ARMY RESEARCH LABORATORY

Review of Bayer Pattern Color Filter Array (CFA) Demosaicing with New Quality Assessment Algorithms

by Robert A. Maschal Jr., S. Susan Young, Joe Reynolds, Keith Krapels, Jonathan Fanning, and Ted Corbin

ARL-TR-5061 **January 2010**

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Adelphi, MD 20783-1197

ARL-TR-5061 **January 2010**

Review of Bayer Pattern Color Filter Array (CFA) Demosaicing with New Quality Assessment Algorithms

Robert A. Maschal, Jr.
Department of Mathematics, University of Maryland
College Park, MD 20742

S. Susan Young
Sensors and Electron Devices Directorate, ARL

Joe Reynolds, Keith Krapels, Jonathan Fanning, and Ted Corbin
Night Vision & Electronic Sensors Directorate
10221 Burbeck Road, Fort Belvoir, VA 22060

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
January 2010	Summary	

4. TITLE AND SUBTITLE
Review of Bayer Pattern CFA Demosaicing with New Quality Assessment Algorithms

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Robert A. Maschal Jr., S. Susan Young, Joe Reynolds, Keith Krapels, Jonathan Fanning, and Ted Corbin

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: RDRL-SES-E
2800 Powder Mill Road
Adelphi, MD 20783-1197

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-5061

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
To address the frequent lack of a reference image or ground truth when performance testing Bayer pattern color filter array (CFA) demosaicing algorithms, we propose two new no-reference quality assessment algorithms. These new algorithms give a relative comparison of two demosaicing algorithms by measuring the presence of two common artifacts in their output images. For this purpose, we reviewed various demosaicing algorithms, especially adaptive color plane, gradient-based methods, and median filtering, paying particular attention to the false color and edge blurring artifacts common to all demosaicing algorithms. We also reviewed classic quality assessment methods that require a reference image (MSE, PSNR, and ΔE), characterized their typical usage, and identified their associated pitfalls. With this information in mind, the motivations for no-reference quality assessment are discussed. From that, we designed new quality assessment algorithms to compare two images demosaiced from the same CFA data by measuring the sharpness of the edges and determining the presence of false colors. Using these new algorithms, we evaluated and ranked the previously described demosaicing algorithms. We reviewed a large quantity of real images, which were used to justify the rankings suggested by the new quality assessment algorithms. This work provides a path forward for future research investigating possible relationships between CFA demosaicing and color image super-resolution.

15. SUBJECT TERMS
Bayer Pattern, CFA Demosaicing, Color Image No Reference Quality Assessment

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			S. Susan Young
Unclassified	Unclassified	Unclassified	UU	42	19b. TELEPHONE NUMBER (*Include area code*) (301) 394-0230

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

1. Introduction

When an image is captured by a monochrome camera, a single charge-coupled device (CCD) or complementary metal-oxide semiconductor (CMOS) sensor is used to sample the light intensity projected onto the sensor. Color images are captured in much the same way, except that the light intensity is measured in separate color bands, usually red, green, and blue. In order to do this, three separate sensors could be used in conjunction with a beam splitter to accurately measure each of the three primary colors at each pixel. However, this approach is expensive and mechanically difficult to implement, making its use in commercial imaging systems infeasible. To overcome this obstacle, the color filter array (CFA) was introduced to capture a color image using only one sensor.

A CFA is an array of alternating color filters that samples only one color band at each pixel location. The most popular CFA pattern is the Bayer pattern (figure 1), which features blue and red filters at alternating pixel locations in the horizontal and vertical directions, and green filters organized in the quincunx pattern at the remaining locations (*1*). This pattern results in half of the image resolution being dedicated to accurate measurement of the green color band. The peak sensitivity of the human visual system lies in the medium wavelengths, justifying the extra green sampling (*2*). Because each pixel now has only one color sampled, a demosaicing algorithm must be employed to recover the missing information.

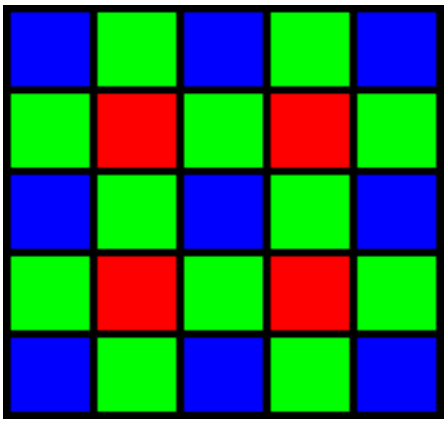


Figure 1. The Bayer pattern features blue and red filters at alternating pixel locations in the horizontal and vertical directions and green filters organized in the quincunx pattern at the remaining locations.

CFA demosaicing is the process through which three fully populated color planes are created from the CFA data. Several algorithms exist for this purpose, ranging from simple linear interpolators to high-end nonlinear interpolators that exploit as much spatial and spectral information as possible.

The rest of the report is organized as follows. Several demosaicing algorithms are introduced in section 2, starting with simple algorithms such as nearest neighbor and ending with more robust gradient-based algorithms. In section 3, the false coloring and zipper effect demosaicing artifacts are introduced and discussed, after which two post-processing techniques for removing false colors are introduced in section 4. The classic performance analysis techniques are introduced and discussed in section 5.1, before one no-reference technique is discussed and two new techniques are proposed in section 5.2 in response to the frequent lack of reference data. In section 6, all algorithms introduced in section 2 are analyzed with these new methods, along with a wealth of images to justify the results.

2. CFA Demosaicing

2.1 Nearest Neighbor

The simplest of all interpolation algorithms is nearest neighbor interpolation. Using a 2x2 neighborhood from the Bayer pattern CFA (figure 2), missing pixel values are interpolated by simply adopting the nearest sampled value (*3*). The sampled blue and red values in this 2x2 neighborhood are used at the three remaining locations. The sampled green values can be moved in either a vertical or horizontal direction to fill in the pixels without green information.

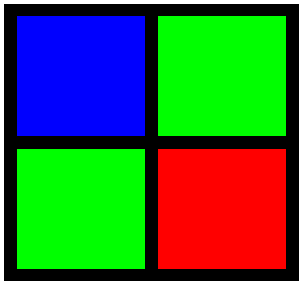


Figure 2. 2x2 Bayer CFA neighborhood.

This method introduces significant color errors, especially along edges. However, since no calculations are performed, this method may be beneficial in applications where speed is critical, such as video imaging systems.

2.2 Linear Interpolation

Another simple interpolation algorithm is linear interpolation. A slightly larger 3x3 neighborhood is taken from the CFA (figure 3), and missing pixel values are estimated by the averaging of nearby values (*3*). Equations used in this interpolation method are as follows (*4*):

$$G5 = \frac{1}{4} * \left(G2 + G4 + G6 + G8\right) \tag{1}$$

$$B5 = \frac{1}{4} * (B1 + B3 + B7 + B9) \tag{2}$$

$$B2 = \frac{1}{2} * (B1 + B3) \tag{3}$$

$$B4 = \frac{1}{2} * (B1 + B7) \tag{4}$$

The missing red values can be estimated using the same equations by replacing the B's with R's.

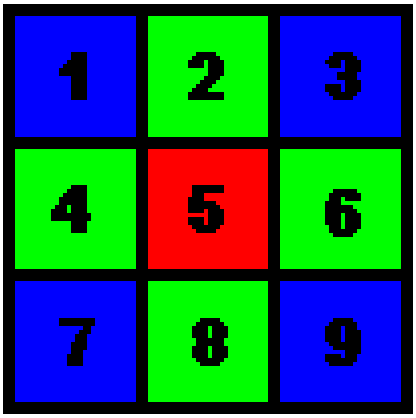


Figure 3. 3x3 Bayer CFA neighborhood.

One benefit of this method is that it can be performed by a convolution with the appropriate kernel. Two kernels are required: one for estimating the missing green values and one for estimating missing red/blue values. The kernels are given below and are identified as *Fg* for the green kernel and *Fc* for the red/blue kernel (*5, 6*).

$$Fg = \frac{1}{4} * \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad Fc = \frac{1}{4} * \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{5}$$

This interpolation method performs well in smooth areas where the color changes slowly from one to the next. However, when performed along edges where color changes occur abruptly, false color and zipper artifacts are introduced, resulting in a poor quality image (see section 3 for information about these types of artifact).

2.3 Cubic Interpolation

Similar in nature to the previously described linear interpolation is cubic interpolation. The algorithm is performed with a simple convolution of a color channel with the appropriate kernel. These kernels are identified as follows, where *Fg* represents the green channel kernel and *Fc* represents the red/blue channel kernel (*6*).

$$Fg = \frac{1}{256} * \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -9 & 0 & -9 & 0 & 0 \\ 0 & -9 & 0 & 81 & 0 & -9 & 0 \\ 1 & 0 & 81 & 256 & 81 & 0 & 1 \\ 0 & -9 & 0 & 81 & 0 & -9 & 1 \\ 0 & 0 & -9 & 0 & -9 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (6)$$

$$Fc = \frac{1}{256} * \begin{bmatrix} 1 & 0 & -9 & -16 & -9 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -9 & 0 & 81 & 144 & 81 & 0 & -9 \\ -16 & 0 & 144 & 256 & 144 & 0 & -16 \\ -9 & 0 & 81 & 144 & 81 & 0 & -9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -9 & -16 & -9 & 0 & 1 \end{bmatrix} \quad (7)$$

Cubic interpolation suffers from the same artifacts as linear interpolation, albeit to a lesser degree. The expanded 7x7 neighborhood reduces the appearance of these artifacts, but they are still present in the final image.

2.4 High Quality Linear Interpolation

Another form of linear interpolation, proposed by Malvar et al. (*7*), expands and improves linear interpolation by exploiting interchannel correlations between the different color channels. A 5x5 neighborhood is used, wherein the nearby pixels of the corresponding color channel are averaged and then added to a correction term calculated from information in a different color channel. Despite only a modest increase in the number of computations performed compared to the linear and cubic interpolations, this method was shown to outperform many more complicated, nonlinear methods, with greatly reduced edge artifacts. However, we decided not test this method in our study.

2.5 Smooth Hue Transition Interpolation

Hue is defined "as the property of colors by which they can be perceived as ranging from red through yellow, green, and blue, as determined by the dominant wavelength of light (*4*)." The key assumption is that hue is smoothly changing across an object's surface. The false color artifact of linear and other methods of interpolation result when this hue changes abruptly, such as near an edge. In this case, hue is defined as the ratio between color channels, in particular the ratio between red/blue and green (*3*). Referring to figure 3, the equations for interpolating the blue channel are as follows:

$$B2 = \frac{G2}{2} * \left(\frac{B1}{G1} + \frac{B3}{G3} \right) \quad (8)$$

$$B4 = \frac{G4}{2} * \left(\frac{B1}{G1} + \frac{B7}{G7} \right) \quad (9)$$

$$B5 = \frac{G5}{4} * \left(\frac{B1}{G1} + \frac{B3}{G3} + \frac{B7}{G7} + \frac{B9}{G9} \right) \tag{10}$$

Equations for red interpolation are defined analogously. Note that this method requires a fully populated green channel, as the green values at every pixel are used. Thus, the green channel must first be fully interpolated by linear, cubic, or some other interpolation method (*3, 4*).

One issue with this method is that it fails when a pixel has a green value of zero (*4*). To resolve this, a normalized approach was proposed by Lukac et al. (*8*). The transformation from the earlier color ratio is defined as

$$\frac{B}{G} \mapsto \frac{B+\beta}{G+\beta} \tag{11}$$

where β is a non-negative number. This generalizes to the earlier definition for the case β=0. In addition to preventing division errors, experimental evidence shows that this normalized model can further improve the demosaicing result in terms of the mean square error (MSE) and peak signal-to-noise ratio (PSNR) by increasing the value of β (*8*). For most cases, β=128 produces sufficient results. In our tests, we use β=128.

2.6 Pattern Recognition Interpolation

Thus far, all of the interpolation algorithms discussed have had flaws estimating colors on or around edges. In attempt to counteract this defect, Cok (*9*) describes a way to classify and interpolate three different edge types in the green color plane. Once the green plane is interpolated, the red and blue color planes are interpolated using the smooth hue transition interpolation described previously.

The first step in his procedure is to find the average of the four neighboring green pixels, and classify the neighbors as either high or low in comparison to this average. For simplicity, equality is labeled as a high value. This pixel is then defined as an edge if three neighbor pixels share the same classification. If not, then the pixel can either be a part of a corner or a stripe. If two adjacent neighbor pixels have the same classification, then the pixel is a corner. If two opposite pixels have the same classification, then the pixel is a stripe. See figure 4 for a picture of each.

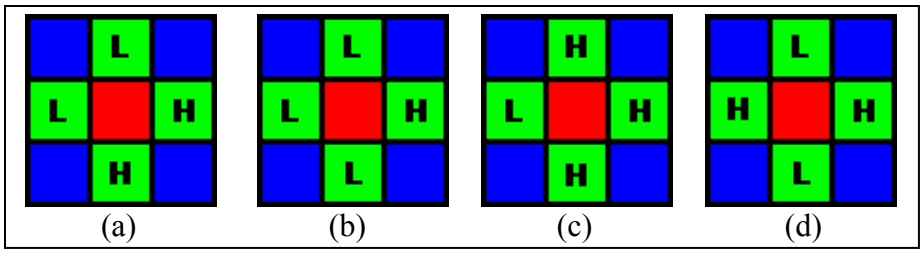


Figure 4. Example neighborhoods for pattern recognition interpolation. (a) is a high edge pattern, (b) is a low edge pattern, (c) is a corner pattern, and (d) is a stripe pattern. Remember that H represents pixels greater than or equal to the average in the 3x3 neighborhood and L represents pixels less than the average.

In the case of both types of edge pattern, the median of the neighbors is calculated and used as the missing green value. For both the stripe and corner patterns, more information must be collected from the surrounding pixels. Furthermore, Cok defined the following clip function for use in the calculation of the missing green value at stripe and corner locations (*9*):

$$clip_C^B(x) = \begin{cases} B & x > B \\ x & C \le x \le B \\ C & x < C \end{cases} \tag{12}$$

where B and C are the values corresponding to A>B>C>D, the rank ordering of the neighboring green values (*3, 9*). In both the stripe and corner cases, the missing green value is defined as

$$G = clip_C^B(2M - S) \tag{13}$$

where M is the median of the H and L pixels, and S is the average of the X pixels in the neighborhoods identified in figure 5 (*3, 9*).

As compared to previous methods, this interpolation algorithm better preserves edge details and minimizes the amount of zipper artifacts along edges.

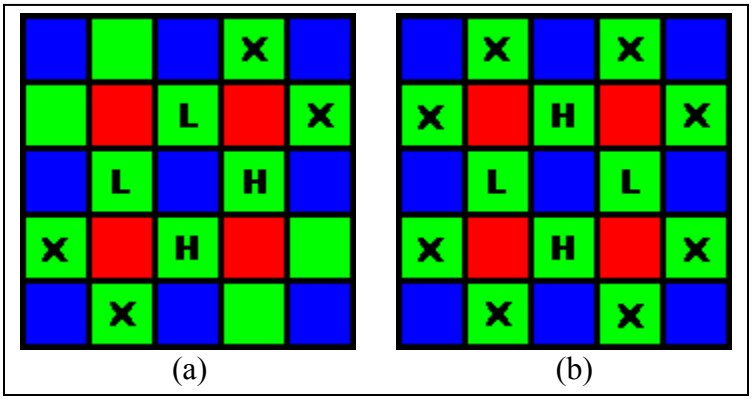


Figure 5. Neighborhoods for the corner and stripe patterns: (a) is the neighborhood for the corner pattern and (b) is the neighborhood for the stripe pattern.

2.7 Adaptive Color Plane Interpolation

Up to this point, the interpolation of the green color plane has occurred using only information from the green samples from the CFA data. However, certain assumptions can be made regarding the correlation between the color planes. One well-known assumption is that the color planes are perfectly correlated in a small enough neighborhood. That is, in a small enough neighborhood, the equations

$$\begin{aligned} G &= B + k \\ G &= R + j \end{aligned} \tag{14}$$

are true for constants k,j. With this assumption, estimations can be made for the second derivate of *G* at a pixel in the vertical and horizontal directions (*4*). Local *G* information can also be used to estimate the first derivate of *G* in the same directions, yielding the following classifiers for the neighborhood shown in figure 6 (*18*):

$$\begin{aligned} V &= |G2 - G8| + |2 * B5 - B1 - B9| \\ H &= |G4 - G6| + |2 * B5 - B3 - B7| \end{aligned} \tag{15}$$

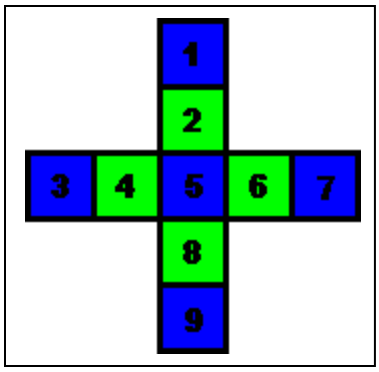


Figure 6. Neighborhood of a B pixel for adaptive color plane interpolation.

The same equations are used for classifiers at an *R* pixel by simply replacing *B* with *R*. Each of these classifiers is used to sense high frequency data in its corresponding direction. Since interpolating across an edge is associated with artifacts such as zippering, the goal is to instead interpolate along the edge, that is, the direction that is changing more slowly (*10*). Thus, the missing *G* value is determined as follows (*4, 10*):

$$G5 = \begin{cases} \dfrac{G2 + G8}{2} + \dfrac{2 * B5 - B1 - B9}{2} & V < H \\ \dfrac{G4 + G6}{2} + \dfrac{2 * B5 - B3 - B7}{2} & V > H \\ \dfrac{G2 + G4 + G6 + G8}{4} + \dfrac{4 * B5 - B1 - B3 - B7 - B9}{4} & V = H \end{cases} \tag{16}$$

Once the green plane is fully interpolated, the red and blue planes are interpolated next. When interpolating a missing *B* value at an *R* pixel, classifiers similar to those used for *G* interpolation are used. With figure 3 representing the relevant neighborhood, the classifiers are (*10*)

$$\begin{aligned} \alpha &= |B1 - B9| + |2 * G5 - G1 - G9| \\ \beta &= |B3 - B7| + |2 * G5 - G3 - G7| \end{aligned} \tag{17}$$

Using these classifiers, *B* interpolation is as follows (*10*):

$$B5 = \begin{cases} \dfrac{B1+B9}{2} + \dfrac{2*G5-G1-G9}{2} & \alpha < \beta \\ \dfrac{B3+B7}{2} + \dfrac{2*G5-G3-G7}{2} & \alpha > \beta \\ \dfrac{B1+B3+B7+B9}{4} + \dfrac{4*G5-G1-G3-G7-G9}{4} & \alpha = \beta \end{cases}$$

$$B2 = \frac{B1+B3}{2} + \frac{2*G2-G1-G3}{2} \qquad (18)$$

$$B4 = \frac{B1+B7}{2} + \frac{2*G4-G1-G7}{2}$$

Interpolation of the red color plane is done in a similar manner as for the blue color plane.

2.8 Directionally Weighted Gradient Based Interpolation

As mentioned in section 2.7, it is best to interpolate missing pixel values along edges, rather than across them. In order to expand the edge detection power of the adaptive color plane method, it is prudent to consider more than two directions. In some methods, as many as 12 directions are considered (*11–13*), in which all the *G* information in a 5x5 neighborhood (shown in figure 7) is used. For the purposes of this report, four directions are considered. They are labeled N, S, E, and W, as indicated in figure 8.

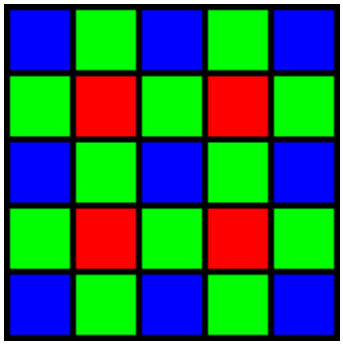


Figure 7. 5x5 Bayer CFA neighborhood.

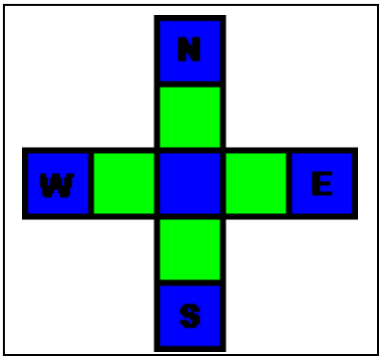


Figure 8. 5x5 Bayer CFA neighborhood with four cardinal directions labeled.

The key distinction of this method between others is that a green value and an associated weight are assigned to each of these four directions. Also, a gradient is calculated for each of these directions based on estimations of the first order derivatives of the green and blue color planes at the pixel under consideration. Using these gradient values, the direction weights are assigned so as to be inversely proportional to the gradient in the associated direction (*11–14*).

Using figure 6 as reference, the equations for calculating the gradients of *G*5 are as follows:

$$\begin{aligned} G_N &= |G8 - G2| + |B5 - B1| \\ G_E &= |G4 - G6| + |B5 - B7| \\ G_S &= |G2 - G8| + |B5 - B9| \\ G_W &= |G6 - G4| + |B5 - B3| \end{aligned} \tag{19}$$

The next step in the procedure is to calculate the directional weights:

$$\begin{aligned} Dirs &= \{N, E, S, W\} \\ \omega_d &= \frac{1}{1 + G_d} \quad d \in Dirs \end{aligned} \tag{2}$$

Since it is certainly possible for a gradient to be zero (representing an estimated no change in the associated direction), a one is added in the denominator of ω_d to prevent division by zero (*11–14*).

Following the calculation of the weights, the directional estimates are calculated (*14*):

$$\begin{aligned} \overline{G}_N &= G2 + \frac{B5 - B1}{2} \\ \overline{G}_E &= G6 + \frac{B5 - B7}{2} \\ \overline{G}_S &= G8 + \frac{B5 - B9}{2} \\ \overline{G}_W &= G4 + \frac{B5 - B3}{2} \end{aligned} \tag{21}$$

The previous equations include a correction term calculated from values in the blue color plane. These formulas come from the assumptions that the green and blue color planes are well correlated with constant offsets and that the rate of change of neighboring pixels along a direction is also constant (*14*). For a derivation of these formulas, please see reference 14.

Once the weights and directional estimates are calculated, the final interpolation of *G*5 can be completed:

$$G5 = \frac{\sum_{d} \omega_d \overline{G}_d}{\sum_{d} \omega_d} \quad d \in Dirs \tag{22}$$

In this example, the missing *G* value is calculated at a *B* pixel location. However, the missing *G* at an *R* location is calculated the same way, replacing all *B*'s in equation 21 with *R*'s.

Following interpolation of all missing *G* values, *B*/*R* values are interpolated at *R*/*B* pixels using the full green color plane. In this instance, gradient values are interpolated in diagonal directions, identified as NE, SE, SW, and NW, as shown in figure 9. A smaller, 3x3 neighborhood is used, since a full green color plane now exists for use in estimating first order derivatives.

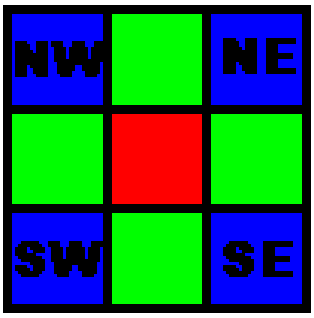


Figure 9. 3x3 Bayer CFA neighborhood with diagonal directions labeled for *B*/*R* interpolation at *R*/*B* pixel.

As an example, equations for *B* interpolation at an *R* pixel are given with reference to figure 3 (*14*):

$$\begin{aligned} B_{NE} &= |B7 - B3| + |G5 - G3| \\ B_{SE} &= |B1 - B9| + |G5 - G9| \\ B_{SW} &= |B3 - B7| + |G5 - G7| \\ B_{NW} &= |B9 - B1| + |G5 - G1| \end{aligned} \tag{23}$$

$$\begin{aligned} Dirs &= \{NE, SE, SW, NW\} \\ \omega_d &= \frac{1}{1 + B_d} \quad d \in Dirs \end{aligned} \tag{24}$$

$$\begin{aligned}
\overline{B}_{NE} &= B3 + \frac{G5 - G3}{2} \\
\overline{B}_{SE} &= B9 + \frac{G5 - G9}{2} \\
\overline{B}_{SW} &= B7 + \frac{G5 - G7}{2} \\
\overline{B}_{NW} &= B1 + \frac{G5 - G1}{2}
\end{aligned} \tag{25}$$

$$B5 = \frac{\sum_d \omega_d \overline{B}_d}{\sum_d \omega_d} \quad d \in Dirs \tag{26}$$

Interpolation of *R* at a *B* pixel is carried out in a similar manner. The remaining *R* and *B* values are interpolated in a similar manner as the missing *G* values, making full use of the previously interpolated *R*/*B* values (*14*). Proceeding in this way allows for a better estimate of edge directions in the red and blue color planes than can normally be achieved by using only measured values.

Perhaps the best attribute of this method of interpolation is its simplicity. Having the adaptive nature of the algorithm encoded within the arithmetic prevents excessive branching, allowing for better pipelining and thus higher speed.

3. Common Demosaicing Artifacts

Because sampling a scene using a CCD with a Bayer pattern CFA measures only 33% of the information of the original scene, several artifacts occur as a result of demosaicing. Two of the most common are false coloring and zippering.

3.1 False Color Artifact

A frequent and unfortunate artifact of CFA demosaicing is what is known as false coloring. This artifact typically manifests itself along edges, where abrupt or unnatural shifts in color occur as a result of misinterpolating across, rather than along, an edge. Figure 10 shows three images demosaiced with bilinear interpolation with examples of false colors. Image (a) has an alternating pattern of red and blue highlights moving along the left edge of the windshield, along with some red and blue highlights on brighter portions of the mirror. Image (b) shows another view of the truck's windshield, where straight lines visible through the windshield appear as alternating red and yellow pixels. Image (c) shows false coloring amidst high frequency information in the Ford logo's lettering.

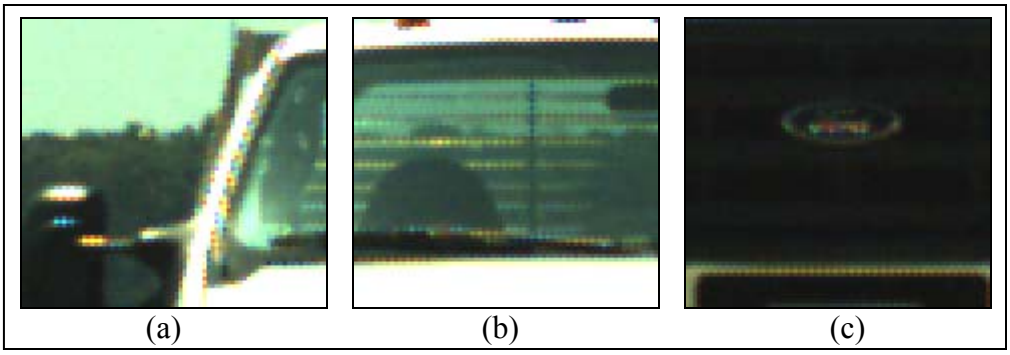


Figure 10. Three images depicting the false color demosaicing artifact. Image (a) shows the corner of a truck with false coloring on the side mirror and along the edge of the windshield. Image (b) shows the window of a truck with false coloring along the edges of the windshield and along edges showing through the windshield. Image (c) depicts the trucks insignia with false coloring amongst the high frequency information contained within.

Several methods exist for preventing and removing this false coloring. Smooth hue transition interpolation, which was reviewed in section 2, is used during the demosaicing to prevent false colors from manifesting themselves in the final image. However, other algorithms exist that can remove false colors after demosaicing. These have the benefit of removing false coloring artifacts from the image while using a more robust demosaicing algorithm for interpolating the red and blue color planes.

3.2. Zippering Artifact

Another side effect of CFA demosaicing, which also occurs primarily along edges, is known as the zipper effect. Simply put, zippering is another name for edge blurring that occurs in an on/off pattern along an edge. Figure 11 shows three images demosaicked with bilinear interpolation featuring the edge blurring zipper effect. Image (a) features a truck with zippering along the upper edge of the grill and also zippering along edges within the headlight. Image (b) features a person with zippering along the stripes in his shirt as well as zippering along the fence poles in the background of the image. Image (c) shows a license plate with zippering along its six characters and more zippering along the upper edge of the bumper.

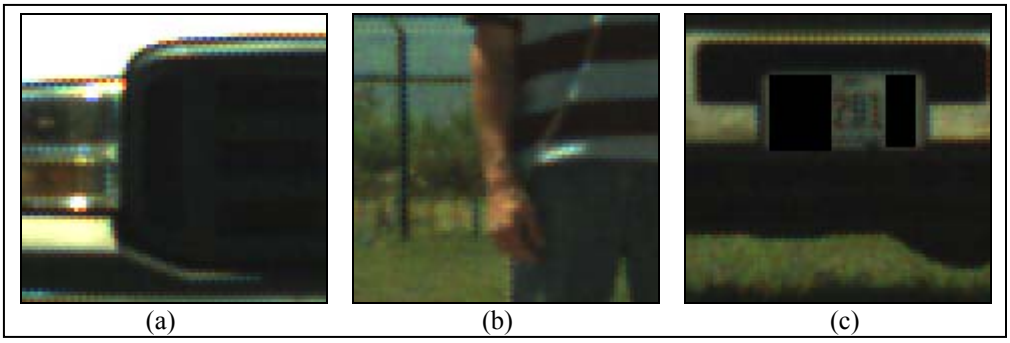


Figure 11. Three images depicting the zippering artifact of CFA demosaicing. (a) features a truck with heavy zippering along edges of the grill and headlights. (b) features a person with zippering along his shirt's stripes and on the fence poles in the background. (c) features a license plate with zippering along its numbers as well as zippering along the edges of the bumper.

This effect occurs when the demosaicing algorithm averages pixel values over an edge, especially in the red and blue planes, resulting in its characteristic blur. The best methods for preventing this effect are the various algorithms which interpolate along, rather than across image edges. Pattern recognition interpolation, adaptive color plane interpolation, and directionally weighted interpolation all attempt to prevent zippering by interpolating along edges detected in the image.

4. Post Demosaicing Artifact Suppression

In section 2, a demosaicing algorithm, known as smooth hue transition interpolation, was introduced, which aims to reduce the occurrence of false coloring in the demosaiced image. However, several post-processing techniques exist that enforce similar smooth hue constraints in the demosaiced image rather than the mosaiced raw data. In this way, more robust interpolation techniques can be used to interpolate the red and blue channels while still reducing the overall appearance of false colors. We summarize two such techniques in this section.

4.1 Local Color Ratio Based Post Processing

Similar to the smooth hue transition algorithm, local color ratio post processing is based on the normalized color ratio model defined in reference 8. The main goal of this algorithm is to correct unnatural changes in hue by smoothing the color ratio planes (*15*).

For the moment, let R(x,y), G(x,y), and B(x,y) be the functions representing the red, green, and blue color planes, respectively, where (x,y) is the pixel at the x-th column and the y-th row. The first part of the algorithm involves adjusting the green color plane based on the newly completed red and blue color planes. It is important to note that this operation is performed only on *R*/*B* pixels. Provided as an example is the equation used to adjust the *G* value at a *B* pixel (*15*):

$$\zeta = \{(p+1,q),(p,q+1),(p-1,q),(p,q-1)\}$$
$$G(p,q) = -\beta + [B(p,q)+\beta] * \underset{(i,j)\in\zeta}{mean}\left\{\frac{G(i,j)+\beta}{B(i,j)+\beta}\right\} \quad (27)$$

where β is a nonnegative constant, as defined by the normalized color ratio model (*8*). This can easily be modified for use on an *R* pixel, by exchanging B(x,y) for R(x,y).

The next step is to perform a similar process for the *B*/*R* values at *R*/*B* pixels. Provided as an example is the equation for the adjustment of the *B* value at an *R* pixel (*15*):

$$\zeta = \{(p+1,q+1),(p+1,q-1),(p-1,q+1),(p-1,q-1)\}$$
$$B(p,q) = -\beta + [G(p,q)+\beta] * \underset{(i,j)\in\zeta}{mean}\left\{\frac{B(i,j)+\beta}{G(i,j)+\beta}\right\} \quad (28)$$

where β has the same value as before. This is, of course, easily modified for adjusting the *R* value at a *B* pixel. Adjustment of *B*/*R* values at *G* pixels is similar, except that it uses the ζ neighborhood from the adjustment of the *G* values.

Testing shows that this method performs well when removing false colors. Moreover, empirical evidence suggests increasing β results in a higher quality image in terms of MSE and PSNR (*8*). However, increasing β too far increases the risk of overflow errors.

4.2 Median Filtering

Unlike methods such as smooth hue transition and local color ratio post processing, median filtering works by acting on color differences, rather than color ratios. Given the image model G=B+k or G=R+j for k,j constant in a local neighborhood, we get that B–G= –k and R–G= –j, or the implication that the color differences are constant within a local neighborhood of the pixel in question. As such, we can adjust a pixel's value by imposing this constant color difference on it.

The first step is to gather all the color difference values over a square neighborhood around the pixel. In many applications, a 5x5 neighborhood is used (*16, 17*) though a smaller 3x3 neighborhood works as well (*4*). Once the differences are gathered, their median is calculated, and then used as an approximation of what the current pixel's color difference should be. Assume we are median filtering the R/B channel at the pixel (p,q) and that ζ is the set of points within the 5x5 neighborhood surrounding the point (p,q). Then the equations are as follows (*17*):

$$\begin{aligned} R(p,q) &= G(p,q) + \underset{(i,j)\in\zeta}{median}\{R(i,j) - G(i,j)\} \\ B(p,q) &= G(p,q) + \underset{(i,j)\in\zeta}{median}\{B(i,j) - G(i,j)\} \end{aligned} \tag{29}$$

This method can occasionally add its own artifacts when used indiscriminately. Thus, it is best used when performed only on edge pixels (*16*).

5. Quality Assessment

The quality of a demosaicing algorithm is assessed by measuring how accurately it interpolates the full image. Typically this is performed by comparing a reference image to an image interpolated from a Bayer pattern subsampling of the reference image. This can be done in the RGB colorspace to which the image already belongs or by transforming the image into an alternate color space, such as the perceptually uniform International Commission on Illumination's L* a* b* color space (CIELAB).

5.1 Quality Assessment with a Reference

The simplest techniques for measuring the quality of a demosaicing algorithm are those that use a reference image. In this case, a reference image is scanned from a film image. This reference is then subsampled in such a way so as to replicate raw data straight from the CCD with the Bayer Pattern CFA. This data is then demosaiced and compared to the reference. Standard techniques for quality assessment with a reference include Color Mean Squared Error (CMSE), Color Peak Signal-to-Noise Ratio (CPSNR), and the International Commission on Illumination's L* a* b* color difference (CIELAB ΔE) (*14*).

5.1.1 Color Mean Squared Error and Color Peak Signal-to-Noise Ratio

CMSE and CPSNR are very simple techniques. Calculating CMSE involves first calculating the squared difference between the reference image and demosaiced image at each pixel and for each color channel. These are then summed and divided by three times the area of the image. CPSNR is then calculated using CMSE. The relevant equations are shown below (*18*):

$$CMSE = \frac{\sum_{i=R,G,B}\sum_{j=1}^{W}\sum_{k=1}^{H}\left(I_{i,j,k} - \bar{I}_{i,j,k}\right)^2}{3WH}$$
$$CPSNR = 10\log\left(\frac{255^2}{CMSE}\right) \tag{30}$$

where I is the reference image, $\bar{I}$ is the demosaiced image, and W and H represent the width and height of the image, respectively.

Both of these methods fail to distinguish the case where the demosaiced image is only slightly different from the reference over many pixels from the case where the demosaiced image is vastly different from the reference image over few pixels. However, the second case represents an image with very severe artifacts, and thus a poorer quality image (*18*).

5.1.2 CIELAB ΔE

Since CMSE and CPSNR given as in section 5.1.1 do not equate to the human perception of color difference, a perceptually uniform color space must be used. In this case, the Euclidean distance between colors represented in the CIELAB perceptually uniform colorspace are calculated as ΔE. Calculating ΔE requires first transforming it from the RGB colorspace into the CIELAB colorspace, and then calculating the average Euclidean distance between the pixel colors in this space. The equation is given by (*14*)

$$\Delta E = \frac{1}{WH}\sum_{i=1}^{W}\sum_{j=1}^{H}\left\|I_{i,j} - \bar{I}_{i,j}\right\| \tag{31}$$

where again W and H represent width and height respectively, and I and $\bar{I}$ represent the reference and demosaiced image, respectively, both now in the CIELAB colorspace.

Though this method better estimates the color differences as perceived by the human visual system, it still does not give a good global estimate of image quality. Like CPSNR, this method cannot distinguish between small differences over the whole image and severe differences over part of the image.

5.1.3 Issues Surrounding CMSE, CPSNR, and ΔE

The main issue with these techniques is that they do not give a good global approximation of image quality. Demosaiced images with similar or equal values in any of these three measures are not necessarily similar or the same, since these values are calculated as averages over the entire image.

Another major detriment to these techniques is that they each require a reference image. Since the quality of an anonymous image is rarely known, finding reference images can be very difficult. Methods that compare demosaiced images without a reference would then be more applicable in real-world situations where references frequently do not exist.

5.2 No-reference Quality Assessment

As previously discussed, there are inherent disadvantages to using quality assessment techniques that require a reference image. In response, two new no-reference techniques are proposed in this report. Before the new techniques are proposed, a current technique known as Blur Measure (*19*) will be introduced, which measures the amount of blur along an image edge. Then, the first new technique, which is an extension of the Blur Measure technique, will be proposed. A second new technique will then be proposed, which operates by measuring the false coloring along an edge within the image. Using these two new techniques, the quality of a demosaicing algorithm's results will be measured based on that algorithm's edge reconstruction abilities.

5.2.1 Blur Measure

A simple measure of blurring in an image proposed in reference 19 involves calculating what is known as an edge's width. Averaging the edge width over the entire image produces what is defined as the blur measure.

In order to calculate blur measure, all edge pixels in an image must be identified, using either the Sobel or Prewitt filters in conjunction with a threshold to reduce noise. Once the edge pixels are found, the edge width is calculated by first getting the profile of the image in the vertical direction passing through the edge pixel. The width of the edge is then found by calculating the distance between the locations of the two local extrema closest to the edge pixel. This process is repeated for all edge pixels, with the blur measure for the image being defined as the average edge width over the entire image.

An example edge profile is shown in figure 12. The edge pixel and corresponding local extrema are marked on the graph with circles. The center circle marks the edge pixel. The right and left circles mark the extrema. The left extremum is located at x = 4 and the right extremum at x = 6. Thus, edge width = 6–4 = 2 in this example. If this is the only edge pixel in the image, then blur measure = 2/1 = 2.

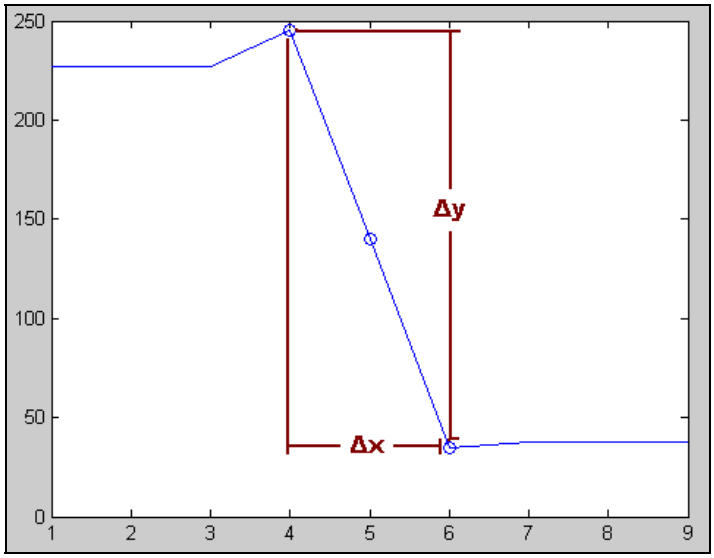


Figure 12. An example edge profile with the edge pixel and local extrema marked with circles. The center circle marks the edge pixel and the outer two circles mark the extrema.

5.2.2 Edge Slope Measure

For the first new no-reference technique, the concept of edge width is expanded to define the notion of edge slope. Moreover, rather than using only the edge profile in either the horizontal or vertical direction, the edge profile is taken along the edge's gradient.

The general procedure for finding edge slope is to first find the edge points in an image, using again either the Sobel or Prewitt filters, as well as a threshold large enough to filter out noise and insignificant edges but small enough to preserve the actual edges in the image. Edges are found in this report using the process outlined in reference 20. For a uniform test, only the edge pixels common to all images are tested so as to ensure only true edges are tested. Using the gradient information gathered from the edge detection stage, an edge profile is taken at an edge point in the direction of that edge point's gradient. The local extrema are then found on this edge profile in the same manner as for blur measure, and edge width is again defined as the distance between the extrema. Edge height is now defined to be the difference between the values of these extrema, and edge slope defined to be the ratio between the edge height and edge width. Edge slope measure is then the average absolute value of edge slope over all edge points in the image. For the example edge shown in figure 12, the edge width would be Δx and the edge height would be Δy. The edge slope would then be $\Delta y / \Delta x$. The necessary equations are

$$\zeta = \{CommonEdgePixels\}$$
$$EdgeSlope(i,j) = \frac{EdgeHeight(i,j)}{EdgeWidth(i,j)} \tag{32}$$
$$EdgeSlopeMeasure = \frac{\sum_{(i,j)\in\zeta} EdgeSlope(i,j)}{card(\zeta)}$$

where edge height and width are defined as previously discussed and $card(\zeta)$ is the cardinality of ζ, or equivalently the number of points in ζ.

The main intention of this assessment technique is to give a relative measure of sharpness of image, for use in comparing the edge reconstruction of various demosaicing algorithms. Thus when edge slope measure is calculated for a set of images demosaiced from the same raw data, the image with the highest edge slope measure will have the highest sharpness and therefore represent the demosaicing algorithm with the best edge reconstruction.

An example edge profile is shown in figure 12. The edge pixel and corresponding local extrema are marked on the graph with circles. The center circle marks the edge pixel. The right and left circles mark the extrema. The left extremum is located at (x,y) = (4,245) and the right extremum at (x,y) = (6,35). Thus, edge slope = (35 – 245)/ (6–4) = –105. If this is the only edge pixel in the image, then the edge slope measure = |–105|/1 = 105.

5.2.3 False Color Measure

The next important demosaicing artifact we seek to identify is the false color artifact defined in section 3. As mentioned previously, false colors occur where there are abrupt changes in color, usually along an edge in the image. Using the constant color difference model G = R+k = B+j for k,j constant in a local neighborhood, the false coloring of an image is measured by divergence from this model by way of MSE. Given that false coloring occurs typically along edges, this method is only performed along edges.

The algorithm involves first finding the edge pixels in the set of images under consideration. As with edge slope measure, only edge pixels common to all images are tested to enforce a uniform test. At each edge pixel, the median color difference is calculated in a 5x5 neighborhood centered at the edge pixel under consideration. The false color measure for the red channel is calculated as follows:

$$\zeta = \{CommonEdgePixels\}$$
$$RFalseColorMeasure = \frac{\sum_{(i,j)\in\zeta} \left[\left(G_{i,j} - R_{i,j}\right) - M_{i,j}\right]^2}{card(\zeta)} \tag{33}$$

where $M_{i,j}$ is the median color difference for the red channel at the point (i,j) and $card(\zeta)$ is the cardinality of ζ, or equivalently the number of common edge pixels. The false color measure for the blue channel is calculated analogously.

Like the edge slope measure, this assessment gives a relative comparison of demosaiced image quality. Given that is technique measures divergence from the expected pattern, the better demosaicing algorithm will produce an image with a lower false color measure.

5.3 Performance Analysis of the Demosaicing Algorithms with New Techniques

Using these two tests in conjunction with one another should give a full comparison of any set of demosaicing algorithms performed on the same set of raw data. The edge slope measure is used to test the overall sharpness of each of the three color channels, thus estimating the relative edge reconstruction accuracy of each demosaicing algorithm. The false measure is performed on the G-R and G-B color difference planes, estimating deviations from the established constant color difference image model, and therefore estimating the R and B channel reconstruction of each demosaicing algorithm. Thus, the overall quality of a demosaicing algorithm is its relative ability to minimize the common blurring and false color artifacts.

For testing purposes, we used real CFA data collected at the Ft. Belvoir Night Vision Lab on June 24, 2009. Twenty images from each of eight image sets were demosaiced using the following algorithms:

- linear interpolation (Linear),
- cubic interpolation (Cubic),
- linear interpolation with smooth hue transition (Linear w/SH),
- pattern recognition interpolation with smooth hue transition (PR w/SH),
- adaptive color plane interpolation (ACP),
- directionally weighted interpolation (DW),
- directionally weighted interpolation with local color ratio based post processing with β=128 (DW w/LCR), and
- directionally weighted interpolation with median filtering (DW w/MF).

Four of the image sets featured a white pickup truck at different distances. The other four image sets featured two men standing at progressively further distances. Since no reference data existed for these images, they were analyzed with the new proposed algorithms. They were analyzed with the following:

- the edge slope measure test in the G channel,
- the edge slope measure test in the R channel,

- the edge slope measure test in the B channel,
- the false color measure test in the G-R color plane, and
- the false color measure test in the G-B color plane.

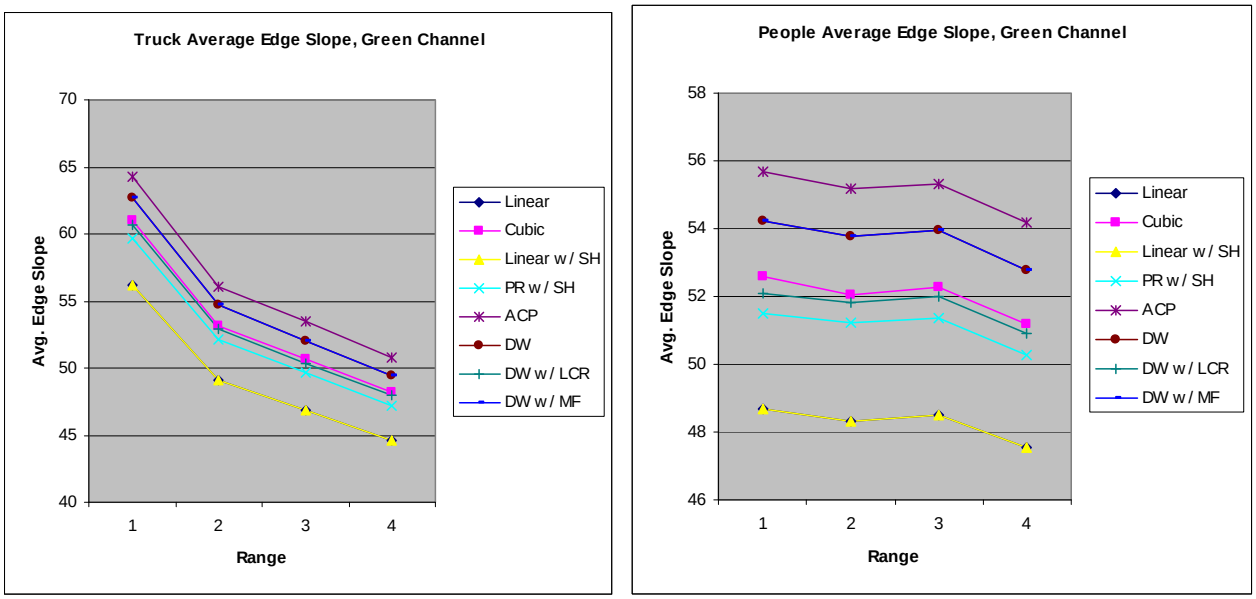
The results are presented as line graphs showing the average measurement for each demosaicing algorithm in each range. For subjective analysis, a large number of image samples are also given.

6. Results and Discussion

6.1 Edge Slope Measure Results

The results for the edge slope measure test are shown in figures 13–15. The results for the green channel are shown in figure 13, the results for the red channel are shown in figure 14, and the results for the blue channel are shown in figure 15. Each figure is split into results for the truck image sets and results for the people image sets.

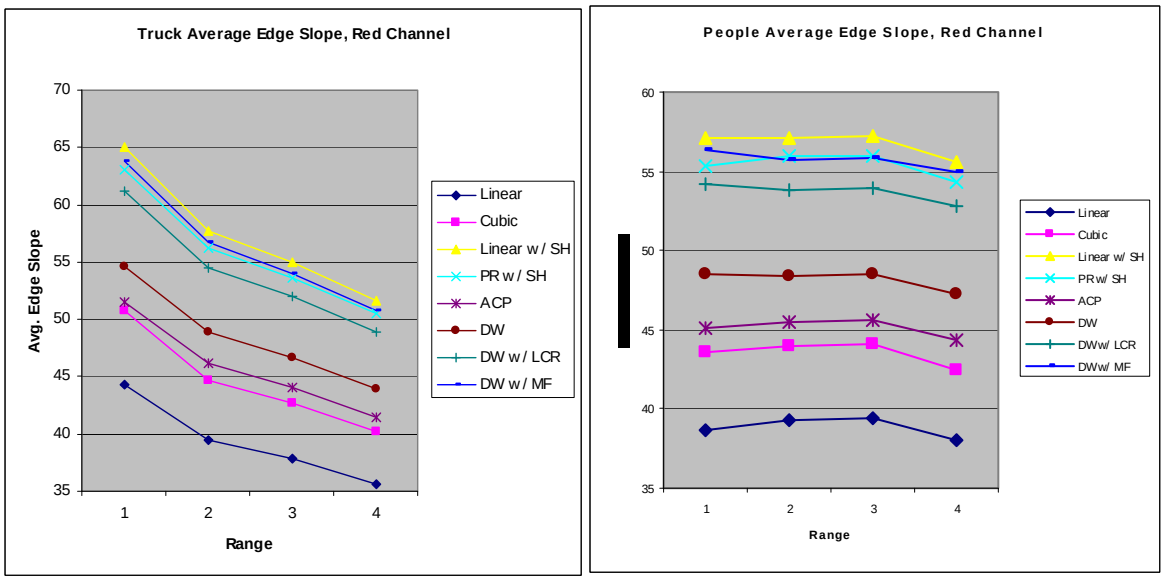
In figure 13, it is clear that ACP interpolation produced the sharpest green color plane from the raw CFA data. It had the highest average edge slope measure in both the truck and people image sets across all ranges of each. The DW and DW w/MF interpolations ranked second highest (they produced the same average slope, since the median filter defined previously does not change the green values) in both image sets. Surprisingly, cubic interpolation achieved the next highest average edge slope measure, despite being a simple kernel-based interpolation scheme. The local color ratio based post processor (in DW w/LCR), which unlike median filtering (in DW w/MF) changes values in the green channel, brought an overall reduction in the edge slope of the DW interpolation's green channel. The linear interpolation performed the worst, as expected, since it operates by averaging the values of nearby pixel, which creates a blurred image as a result. Smooth hue transition does not affect the green channel since by definition it does not operate on these values, so the Linear and Linear w/SH interpolation algorithms produced the same results in the green channel.



(a) (b)

Figure 13. Average edge slope measure for the green channel: (a) The performance of the various algorithms for the images sets showing a truck at four increasing distances and (b) the performance of the algorithms for the image sets showing two people standing at increasing distances.

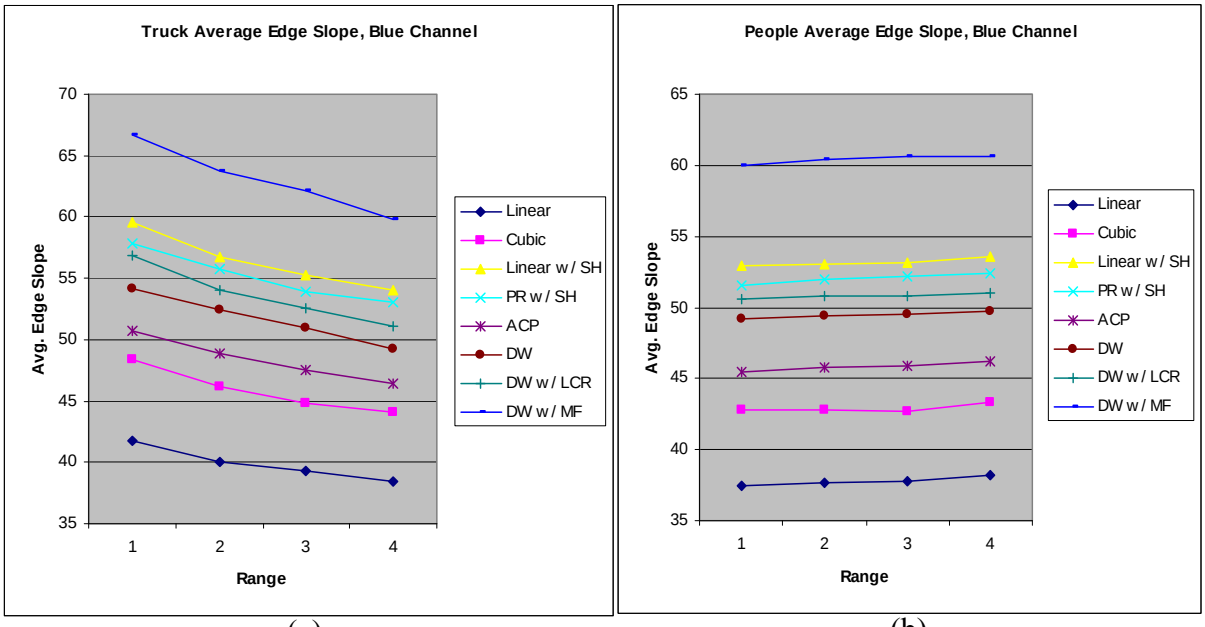
In figure 14, it may be surprising to see that the Linear w/SH interpolation performed best on the red channel. In view of the fact that plain linear interpolation again performed the worst on the red channel, this shows the extent to which exploiting interchannel correlations can improve the quality of the demosaiced image. This is further shown with the improvements attained by post processing the DW interpolation. Both the median filter (in DW w/MF) and local color ratio based post processor (in DW w/LCR) improved the edge slope in the red channel. Both the ACP and cubic interpolations performed worse than expected, given the results in the green channel. PR interpolation performed well in this channel, which is likely the result of the smooth hue transition used to interpolate the red and blue channels.



(a) (b)

Figure 14. Average edge slope measure for the red channel: (a) The performance of the various algorithms for the images sets showing a truck at four increasing distances and (b) the performance of the algorithms for the image sets showing two people standing at increasing distances.

In figure 15, it is immediately obvious the effect median filtering had on the edge slope in the blue channel, with the DW w/MF interpolated images being at least five units higher than any other interpolation technique. Other than this, there were few differences between results for the blue channel and results for the red channel.



(a) (b)

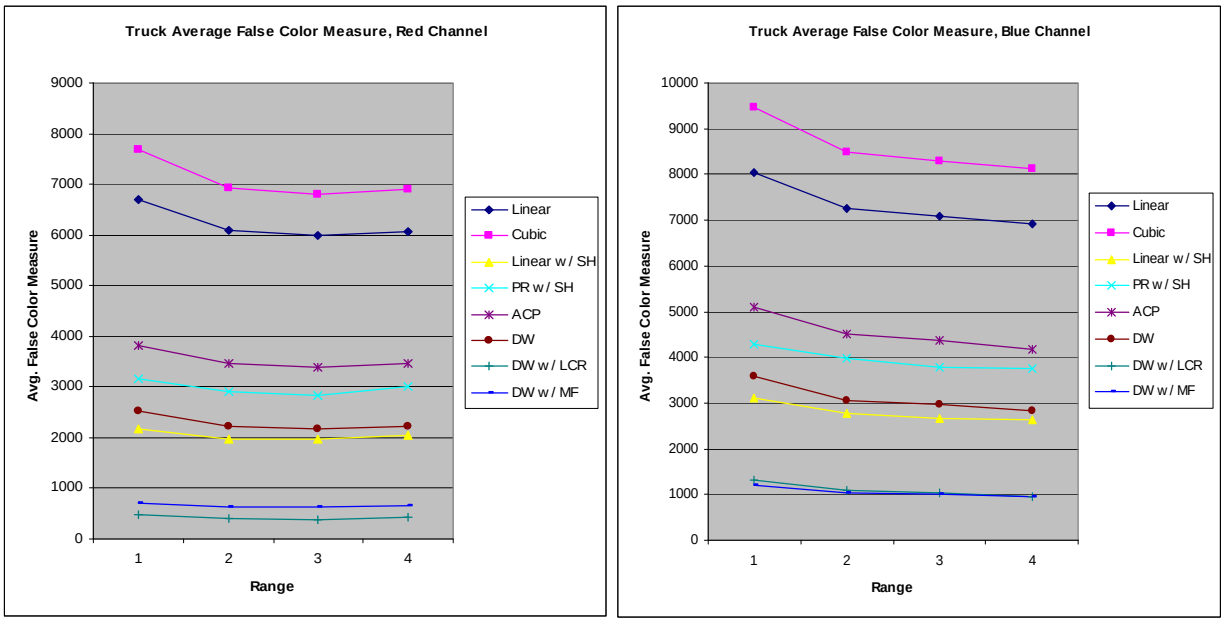
Figure 15. Average edge slope measure for the blue channel: (a) The performance of the various algorithms for the images sets showing a truck at four increasing distances and (b) the performance of the algorithms for the image sets showing two people standing at increasing distances.

One thing to note in these tests is that the performance of these algorithms decreased with the range in the truck image sets. No generalization existed for performance changes with range for the people image sets, as the trends were different in each color channel. This may be due to the fact that manmade objects exhibit sharp edges while people and natural objects exhibit smoother edges. One thing worth noting, however, is that in the blue and red color channels, techniques that used the smoothly changing color ratio and color difference models performed better. The two post-processing techniques both showed improvement in the red and blue color channels in terms of edge slope, despite a decrease in performance in the green channel using the local color ratio based post processing (in DW w/LCR).

6.2 False Color Measure Results

The results for the false color measure test are shown in figures 16–17. The results for the truck image sets are shown in figure 16 and the result for the people image sets in figure 17. Each figure splits the results into results for the red channel and results for the blue channel.

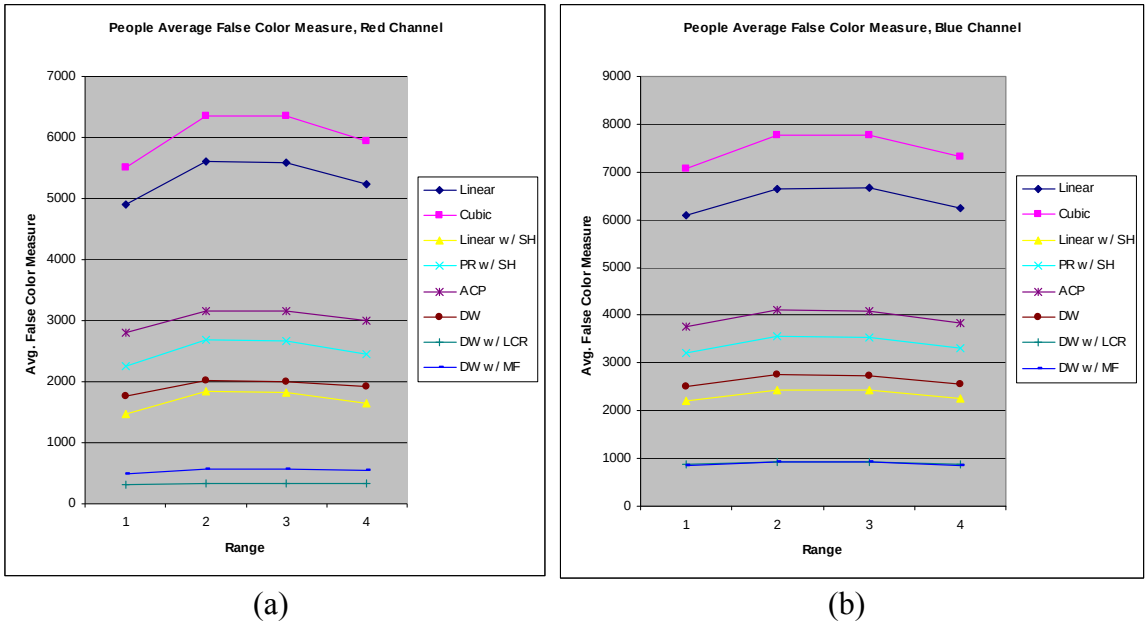
In figure 16, we see that the DW interpolation with the two post-processing techniques performed the best on both the red and blue channels, followed by Linear w/SH and then the regular DW interpolation. Cubic interpolation performed the poorest, followed by linear interpolation and then adaptive color plane interpolation. The false color measure for the truck image sets decreased with range initially, and then increased slightly at the end in the red channel, and continued decreasing in the blue channel, creating a convex graph.



(a) (b)

Figure 16. Average false color measure for images featuring the truck at different ranges: (a) The average false color measures for the red channel and (b) the average false color measures for the blue channel.

In figure 17 the results are nearly the same. The only difference we notice is that initially false color measure increased with the increase in range, and then decreased at the end, creating a concave graph. Otherwise, DW interpolation with post processing performed the best, and the two kernel-based techniques without smooth hue transition performed the worst. All other techniques had the same relative performance as with the truck image set.



(a) (b)

Figure 17. Average false color measure for images featuring the two people at different ranges: (a) The average false color measures for the red channel and (b) the average false color measures for the blue channel.

As with the edge slope measure, we see immediately that techniques exploiting interchannel correlations performed the best, producing the least amount of false colors. Also worth noting is that all interpolation techniques performed worse in the blue channel compared to the red channel across all image sets.

6.3 Image Examples

Figures 18–24 each give eight image samples, one for each demosaicing algorithm tested. Figures 18–21 give image samples from range one of the truck image sets and figures 22–24 give image samples from range one of the people image sets.

In view of the edge slope measure and false color measure results shown in figures 13–17 and the discussion in sections 4.3.1 and 4.3.2, the subjective quality of these images conformed to the objective results. We see in all cases that DW interpolation with some form of post processing produced the least amount of false colors as well as had the sharpest edges (figures 18–24 (g) and (h)). The linear and cubic interpolations produced significant and highly visible false coloring along edges, which tended to be blurry in comparison to other techniques (figures 18–24 (a) and (b)). In comparing (a) to (c) in figures 18–24, it is apparent that the false coloring of

the linear interpolation was removed when the smooth hue transition interpolation was applied, though the edge sharpness did not increase overall. When comparing (f) to (g) and (h) of figures 18–24 we again see the effect post processing has on the presence of false colors. The dark edges of (f) in figures 18–19 have noticeable false color artifacts, which were mostly, if not completely removed by the post processing.

Of particular interest are figures 20–21, which feature the truck's license plate and brand insignia. Images (a)–(c) all feature edge zippering and false colors, which make it difficult to read the license plate and insignia. Moving through to image (f), both figures 18 and 19 slowly become more readable with fewer artifacts. Finally, images (g) and (h) are most easily read with the highest sharpness and the lowest occurrence of false colors. In particular, notice the effect post processing had on the truck's insignia, viewable in figure 21 (f)–(h). For reference, the license plate number is W 42916.

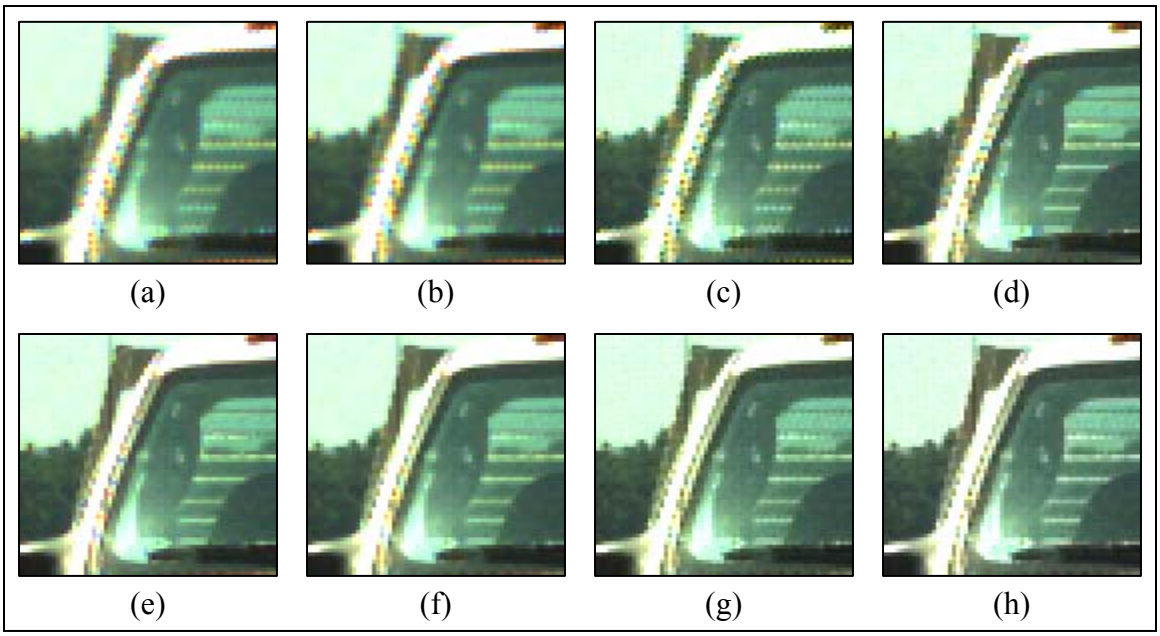


Figure 18. Image samples demosaiced from the first image of the closest range of the truck image sets using the following algorithms: (a) linear, (b) cubic, (c) linear with smooth hue transition, (d) pattern recognition with smooth hue transition, (e) adaptive color plane, (f) directionally weighted, (g) directionally weighted with local color ratio post processing, and (h) directionally weighted with median filter post processing.

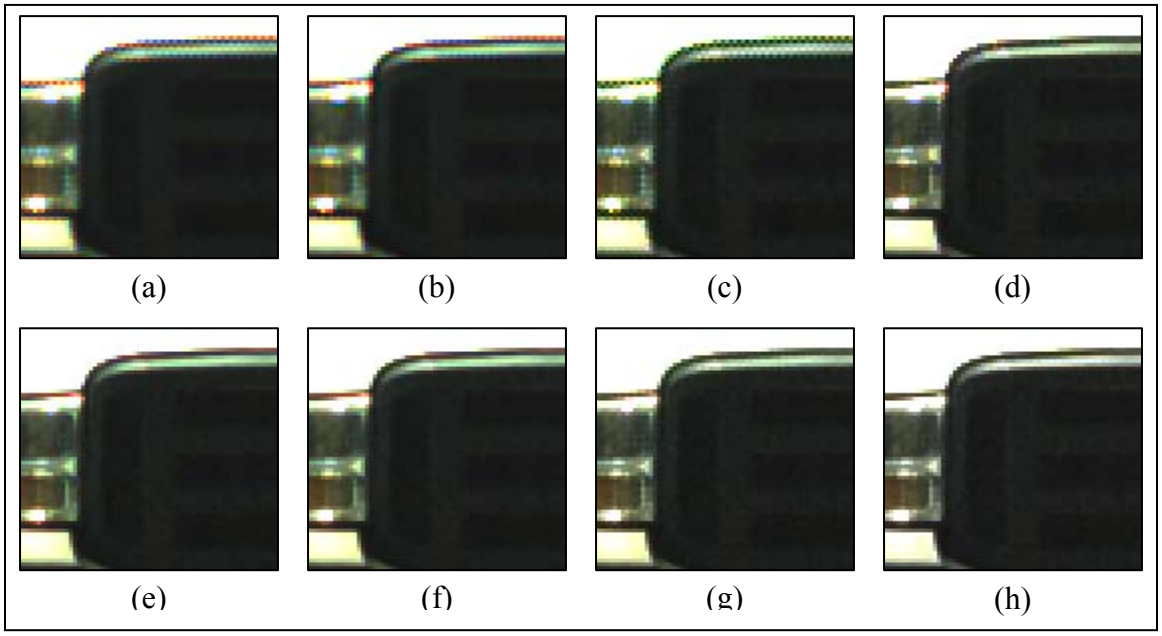


Figure 19. Image samples demosaiced from the first image of the closest range of the truck image sets using the following algorithms: (a) linear, (b) cubic, (c) linear with smooth hue transition, (d) pattern recognition with smooth hue transition, (e) adaptive color plane, (f) directionally weighted, (g) directionally weighted with local color ratio post processing, and (h) directionally weighted with median filter post processing.

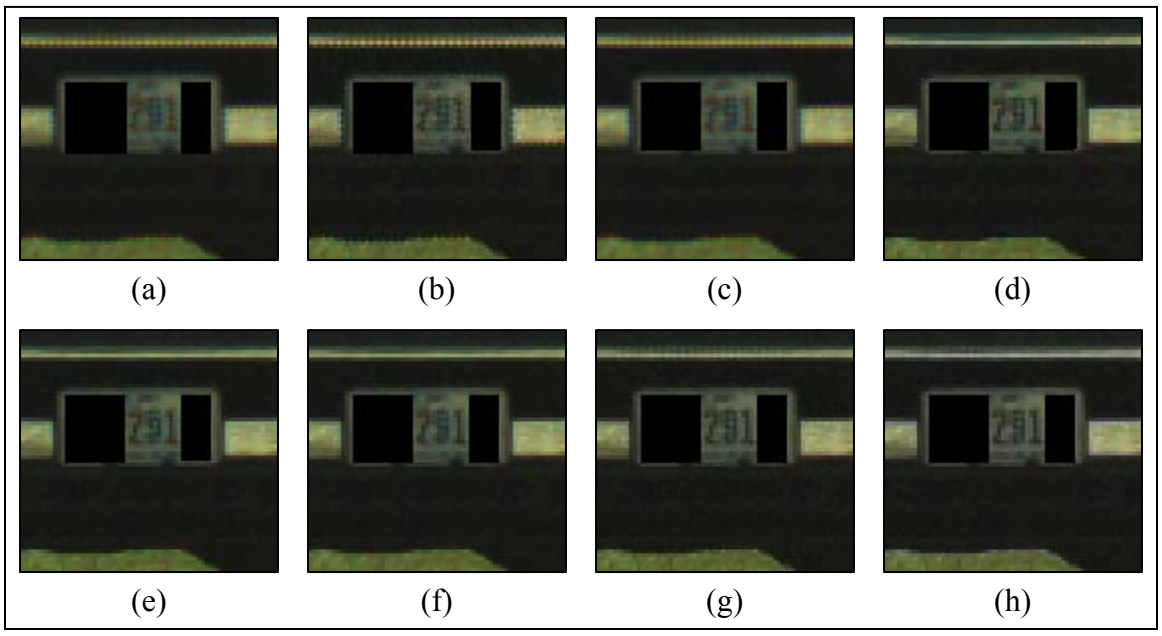


Figure 20. Image samples demosaiced from the first image of the closest range of the truck image sets using the following algorithms: (a) linear, (b) cubic, (c) linear with smooth hue transition, (d) pattern recognition with smooth hue transition, (e) adaptive color plane, (f) directionally weighted, (g) directionally weighted with local color ratio post processing, and (h) directionally weighted with median filter post processing.

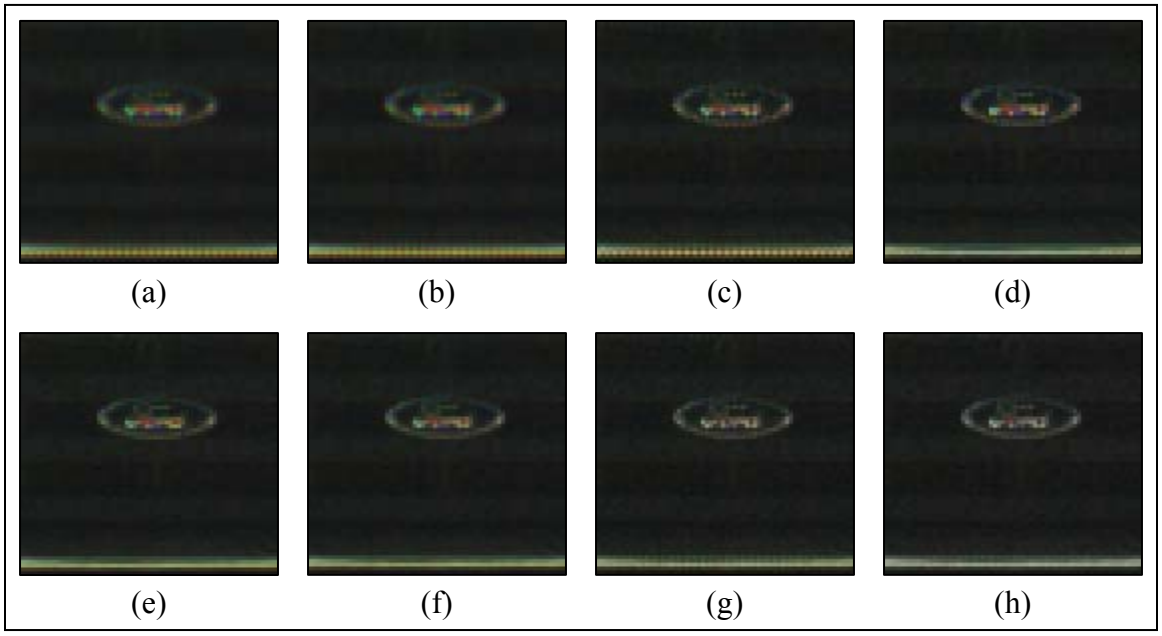


Figure 21. Image samples demosaiced from the first image of the closest range of the truck image sets using the following algorithms: (a) linear, (b) cubic, (c) linear with smooth hue transition, (d) pattern recognition with smooth hue transition, (e) adaptive color plane, (f) directionally weighted, (g) directionally weighted with local color ratio post processing, and (h) directionally weighted with median filter post processing.

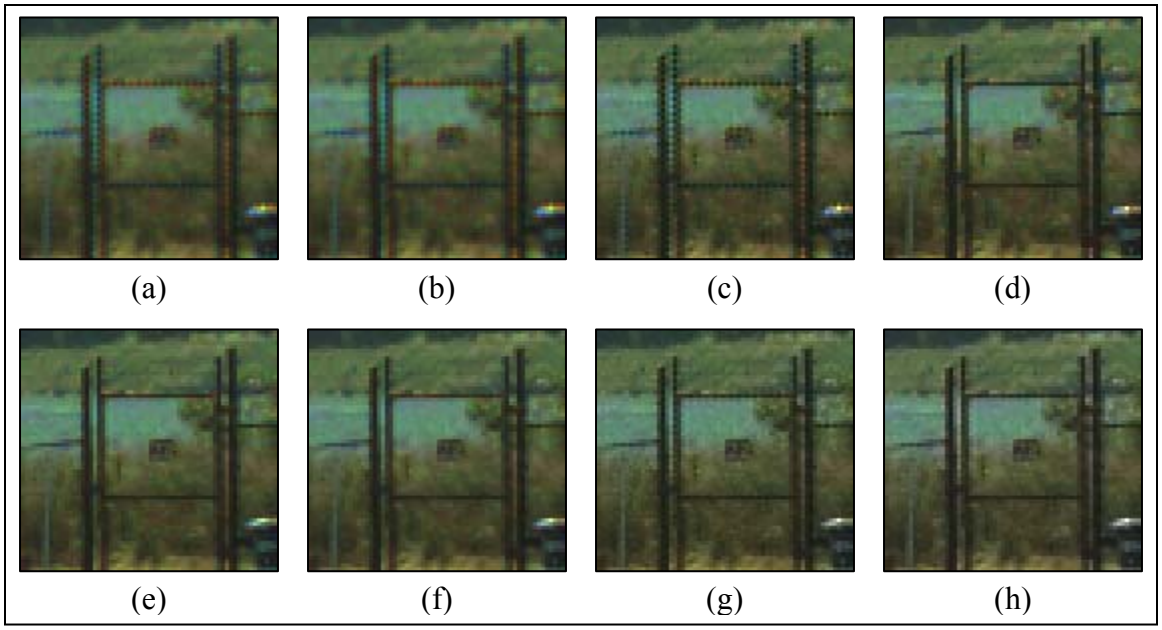


Figure 22. Image samples demosaiced from the first image of the closest range of the truck image sets using the following algorithms: (a) linear, (b) cubic, (c) linear with smooth hue transition, (d) pattern recognition with smooth hue transition, (e) adaptive color plane, (f) directionally weighted, (g) directionally weighted with local color ratio post processing, and (h) directionally weighted with median filter post processing.

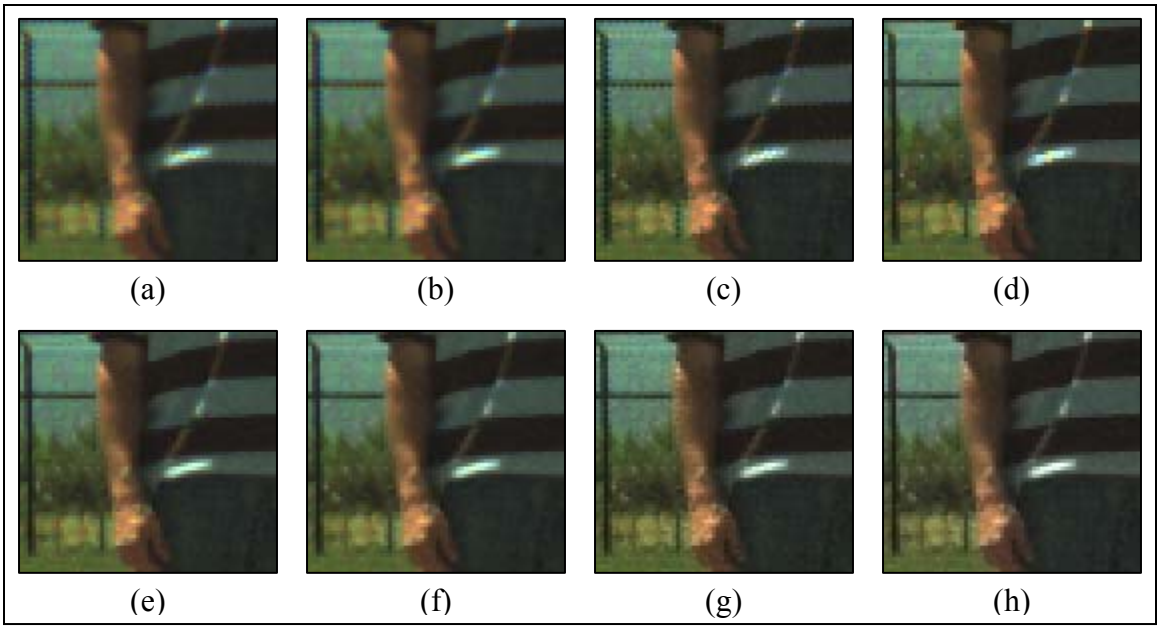


Figure 23. Image samples demosaiced from the first image of the closest range of the people image sets using the following algorithms: (a) linear, (b) cubic, (c) linear with smooth hue transition, (d) pattern recognition with smooth hue transition, (e) adaptive color plane, (f) directionally weighted, (g) directionally weighted with local color ratio post processing, and (h) directionally weighted with median filter post processing.

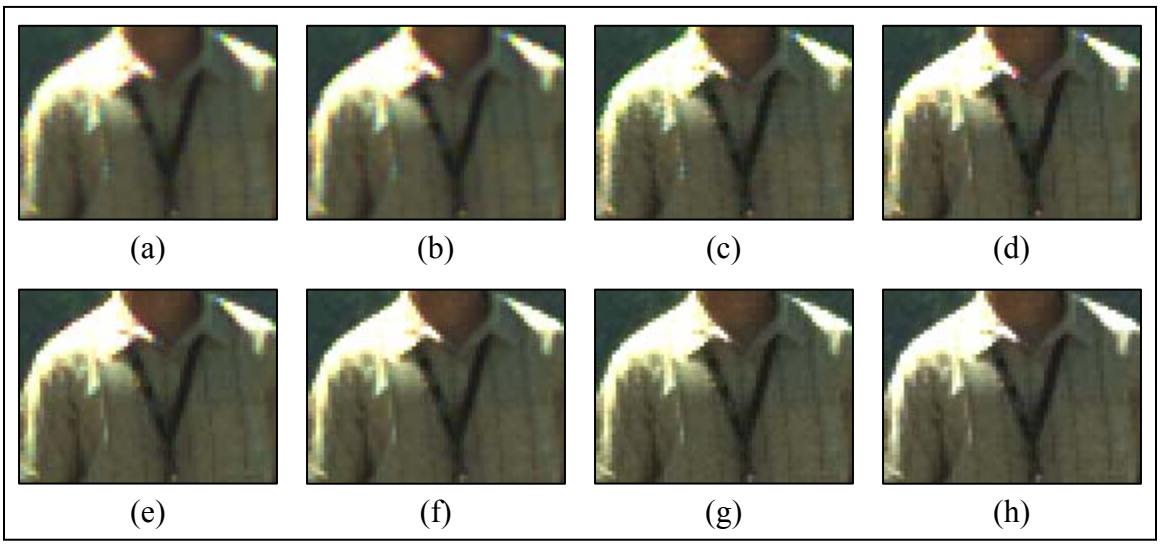


Figure 24. Image samples demosaiced from the first image of the closest range of the people image sets using the following algorithms: (a) linear, (b) cubic, (c) linear with smooth hue transition, (d) pattern recognition with smooth hue transition, (e) adaptive color plane, (f) directionally weighted, (g) directionally weighted with local color ratio post processing, and (h) directionally weighted with median filter post processing.

7. Conclusions

In this report, we proposed two new techniques for assessing the relative quality of images demosaiced from the same CFA data using different demosaicing strategies. The first measures the relative sharpness of the images by calculating the average edge slope along edge points common to all images. The second measures the relative quantity of false coloring by approximating the average squared deviation from the locally constant color difference image model along edge points common to all images. Following the presentation of objective results using each technique, a large number of image samples were given to show that the assessment results were in accordance with the quality of the example images. Moreover, given the image samples and no-reference test results, we concluded that DW w/MF provided the best overall demosaiced image with the fewest color artifacts and sharpest edges.

This report is intended to establish a path forward for future research investigating possible relationships between CFA demosaicing and color image super-resolution. Reference 18 shows that the quality of a demosaiced image decreases with the resolution of the raw data, suggesting possible gains from super-resolving a raw image sequence before demosaicing. With new machinery for comparing image quality without a reference image, a study investigating ties between CFA demosaicing and color image super-resolution is now within reach.

8. References

1. Bayer, B. E. Color Image Array. U.S. Patent 3 971 065, July 1976.
2. Li, X.; Gunturk, B.; Zhang, L. Image Demosaicing: A Systematic Survey. In *Proc. SPIE*, volume 6822, 68221J, 2008.
3. Adams, J. E. Interactions Between Color Plane Interpolation and Other Image Processing Functions in Electronic Photography. In *Proc. SPIE*, volume 2416, pages 144–151, 1995.
4. Ramanath, R.; Snyder, W. E.; Bilbro, G. L.; Sander III, W. A. Demosaicking Methods for Bayer Color Arrays. *Journal of Electronic Imaging* **July 2002**, *11* (3), 633–642.
5. Alleysson, D.; Susstrunk, S.; Herault, J. Color Demosaicing by Estimating Luminance and Opponent Chrominance Signals in the Fourier Domain. *IS&T/SID Tenth Color Imaging Conference,* pages 331–336, 2003.
6. Popescu, C. *Statistical Tools for Digital Image Forensics*. Diss. Dartmouth College, Hanover, New Hampshire, December 2004.
7. Malvar, H. S.; He, L.-W.; Cutler, R. High-quality Linear Interpolation for Demosaicing of Color Images. In *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing,* volume 3, pages 485–488, 2004.
8. Lukac, R.; Plataniotis, K. N. A Normalized Model for Color-Ratio Based Demosaicking Schemes. In *Proc. ICIP,* pages III: 1657–1660, 2004.
9. Cok, D. R. Signal Processing Method and Apparatus for Sampled Image Signals. U.S. Patent 4 630 307, December 1986.
10. Adams, J. E.; Hamilton Jr., J. F. Adaptive Color Plane Interpolation in Single Sensor Color Electronic Camera. U.S. Patent 5 652 621, July 1997.
11. Chang, H.-A.; Chen, H. Directionally Weighted Color Interpolation for Digital Cameras. *IEEE International Symposium on Circuits and Systems* **May 2005**, *6*, 6284–6287.
12. Jung, T.-Y.; Yang, S.; Lee, J.; Jeong, J. Enhanced Directionally Weighted Demosaicing for Digital Cameras. *IEEE International Conference on Signal Image Technology and Internet Based Systems* **2008**, 518–522.
13. Thakur, R. K.; Tripathy, A.; Ray, A. K. A New Edge Adaptive and Directionally Weighted Color Interpolation Algorithm of Digital Camera Images. *International Conference on Advances in Computing, Communication and Control 2009* **January 2009**, 397–402.

14. Lu, W.; Tan, Y.-P. Color Filter Array Demosaicking: New Method and Performance Measures. *IEEE Transactions on Image Processing* **October 2003***, 12* (10), 1194–1210.

15. Lukac, R.; Martin, K.; Plataniotis, K. N. Demosaicked Image Postprocessing Using Local Color Ratios. *IEEE Transactions on Circuits and Systems for Video Technology* **June 2004***, 14* (6), 914–920.

16. Chang, L.; Tan, Y.-P. Adaptive Color Filter Array Demosaicking with Artifact Suppression. *IEEE Proc. of the International Symposium on Circuits and Systems* **May 2004***, 3*, 937–940.

17. Freeman, W. T. Median Filter for Reconstructing Missing Color Samples. U.S. Patent 4 724 395, February 1988.

18. Yang, Y.; Losson, O.; Duvieubourg, L. Quality Evaluation of Color Demosaicing According to Image Resolution. *IEEE International Conference on Signal Image Technology and Internet Based Systems* **December 2007***,* 689–6957.

19. Marziliano, P.; Dufaux, F.; Winkler, S.; Ebrahimi, T. Perceptual Blur and Ringing Metrics: Application to JPEG2000. *Signal Processing: Image Communication* **February 2004**, *19* (2), 163–172, ISSN 0923-5965, DOI: 10.1016/j.image.2003.08.003. (http://www.sciencedirect.com/science/article/B6V08-49DMTV9-1/2/16d4758577133d77d8f56b927ead861d)

20. Young, S. S.; Lee, H.-C. Automatic Location of Cylindrical Bones in Digital Projection Radiographs using Eigenvector Analysis. *Proceedings of SPIE* **2000***, 3979*, 1581.

List of Symbols, Abbreviations, and Acronyms

CCD	charged-coupled device
CFA	color filter array
CIELAB	International Commission on Illumination's L* a* b* color space
CIELAB ΔE	International Commission on Illumination's L* a* b* color difference
CMOS	complementary metal-oxide semiconductor
CMSE	color mean squared error
CPSNR	color peak signal-to-noise ratio
MSE	mean squared error
PSNR	peak signal-to-noise ratio

NO. OF
COPIES ORGANIZATION

1 ELEC ADMNSTR
DEFNS TECHL INFO CTR
ATTN DTIC OCP
8725 JOHN J KINGMAN RD STE 0944
FT BELVOIR VA 22060-6218

1 DARPA
ATTN IXO S WELBY
3701 N FAIRFAX DR
ARLINGTON VA 22203-1714

1 CD OFC OF THE SECY OF DEFNS
ATTN ODDRE (R&AT)
THE PENTAGON
WASHINGTON DC 20301-3080

1 US ARMY INFO SYS ENGRG CMND
ATTN AMSEL IE TD A RIVERA
FT HUACHUCA AZ 85613-5300

1 COMMANDER
US ARMY RDECOM
ATTN AMSRD AMR
W C MCCORKLE
5400 FOWLER RD
REDSTONE ARSENAL AL
35898-5000

1 US ARMY RSRCH LAB
ATTN RDRL CIM G T LANDFRIED
BLDG 4600
ABERDEEN PROVING GROUND MD
21005-5066

16 US ARMY RSRCH LAB
ATTN IMNE ALC HRR
MAIL & RECORDS MGMT
ATTN RDRL CIM L TECHL LIB
ATTN RDRL CIM P TECHL PUB
ATTN RDRL SE J PELLEGRINO
ATTN RDRL SES E R RAO
ATTN RDRL SES J EICKE
ATTN RDRL SES E S S YOUNG
(10 COPIES)
ADELPHI MD 20783

NO. OF
COPIES ORGANIZATION

4 US ARMY NIGHT VISION &
ELECTRONICS SENSORS DIR
SENSOR PERFORMANCE BRANCH
ATTN AMSRD CER NV MS SP
J REYNOLDS
ATTN AMSRD CER NV MS SP
J FANNING
ATTN T CORBIN
ATTN K KRAPELS
10221 BURBECK RD
FT BELVOIR VA 22060

1 NAVAL RESEARCH LABORATORY
ATTN R G DRIGGERS
4555 OVERLOOK AVE
WASHINGTON DC 20375

TOTAL: 27 (1 ELEC, 1 CD, 25 HCs)

INTENTIONALLY LEFT BLANK.