Demosaicing Algorithms

Rami Cohen

August 30, 2010

Contents

1 Demosaicing

A demosaicing algorithm is a digital image process used to reconstruct a full color image from the incomplete color samples output from an image sensor overlaid with a color filter array (CFA). Also known as CFA interpolation or color reconstruction [1].

The reconstructed image is typically accurate in uniform-colored areas, but has a loss of resolution (detail and sharpness) and has edge artifacts (for example, the edges of letters have visible color fringes and some roughness).

1.1 Algorithms

1. **Nearest-Neighbour Interpolation** Simply copies an adjacent pixel of the same color channel (2x2 neighbourhood). It is unsuitable for any application where quality matters, but can be useful for generating previews given limited computational resources.

2. **Bilinear Interpolation** The red value of a non-red pixel is computed as the average of the two or four adjacent red pixels, and similarly for blue and green. bilinear interpolation generates significant artifacts, especially across edges and other high-frequency content, since it doesn‘t take into account the correlation among the RGB values [2].

3. **Cubic Interpolation** Taking into account more neighbours than in algorithm no. 2 (e.g., 7x7 neighbourhood). Lower weight is given to pixels which are far from the current pixel.

4. **Gradient-corrected bilinear interpolation** An improvement to algorithm no. 2 is suggested by [3]. The assumption is that in a luminance/chrominance decomposition, the chrominance components don‘t vary much across pixels. It exploits the interchannel correlations between the different color channels and uses the gradients among one color channel, in order to correct the bilinearly interpolated value.

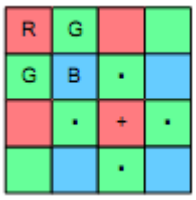

Figure 1: Bayer mosaic of color image

For example, in order to interpolate the green value at an R location ('+' pixel in Figure 1), we correct the bilinearly interpolated value $\hat{g}_b(i,j)$ by a measure of the gradient of R at that location:

$$g_{corrected}(i,j) = \hat{g}_b(i,j) + \alpha\Delta_R$$

where

$$\Delta_R = r(i,j) - r_{avg}$$

and r_{avg} is the average of the 4 nearest red values. α is a gain factor which controls the intensity of such correction. The gradient-corrections for B and R are calculated in a similar manner. The formulas are given in [3], section 3.1, with additional gain factors β and γ. These gain factors are estimated using a data set of images.

This method achieves an improvement of 5.5dB PSNR over bilinear demosaicing, and outperforms many non-linear algorithms [3], when being relatively simple and capable of linear implementation. *This method is used by the function Demosaic in Matlab.*

5. **Smooth Hue Transition Interpolation** First, the green values are interpolated using some desired method. By the assumption that hue is smoothly changing across an objects surface, simple equations for the missing colors can be obtained by using the ratios between the known colors and the interpolated green values at each pixel [2]. Problem can occur when the green value is 0, so some simple normalization methods are proposed [4].

6. **Pattern Recognition Interpolation** In order to prevent flaws when estimating colors on or around edges, Cok [5] describes a way to classify and interpolate three different edge types in the green color plane. 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.

   The green 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. Using median value and an appropriate clipping function, the rest green values are obtained. Later, algorithm 5 is used in order to interpolate R and B values. This algorithm better preserves edge details.

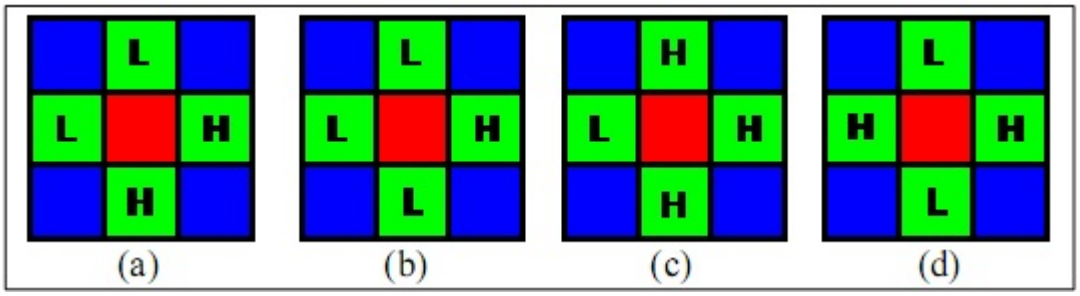

Figure 2: (a) is a high edge pattern, (b) is a low edge pattern, (c) is a corner pattern, and (d) is a stripe pattern.

7. **Adaptive Color Plane Interpolation** A 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

$$\begin{aligned} G &= B + k \\ G &= R + j \end{aligned} \tag{1}$$

are true for constants k, j. With this assumption, Local G information can be used to estimate the first derivative of G in the vertical and horizontal directions, yielding classifiers V and H which are used to sense high frequency data in the corresponding direction. Later, G is calculated according to the ratio between V and H (using Figure 3 as a refernce):

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

Once the green plane is fully interpolated, the red and blue planes are interpolated next with classifiers similar to those used for G interpolation. The appropriate formulas are given in [2], section 2.7.

8. **Directionally Weighted Gradient Based Interpolation** In order to expand the edge detection power of the adaptive color plane method, it is prudent to consider more than two directions (i.e., not only the horizonal and vertical directions).

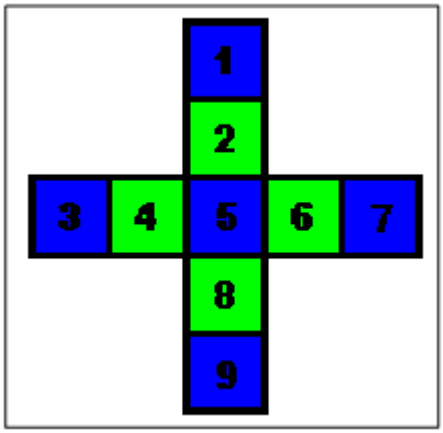

Figure 3: Neighbourhood of B pixel

We can use information from 4 directions (N,S,W,E as shown in Figure 4) in order to calculate 4 gradients for the central missing G value. A weight W_d is assigned for each direction, using the known information about the differences between B and G values. Using Figure 3 as a reference, we can calculate the gradients of G5 as follows:

$$G_N = |G8 - G2| + |B5 - B1|$$
$$G_E = |G4 - G6| + |B5 - B7|$$
$$G_S = |G2 - G8| + |B5 - B9|$$
$$G_W = |G6 - G4| + |B5 - B3|$$

The weight is defined for each direction as $W_d = \frac{1}{1+G_d}$. Using weighted sum the neighbouring green pixels with the gradients, we can calculate G5.

After obtaining G values, R and B are calculated similarly using 4 other directions (NW,NE,SW,SE). All the formulas are given in [2], section 2.8.

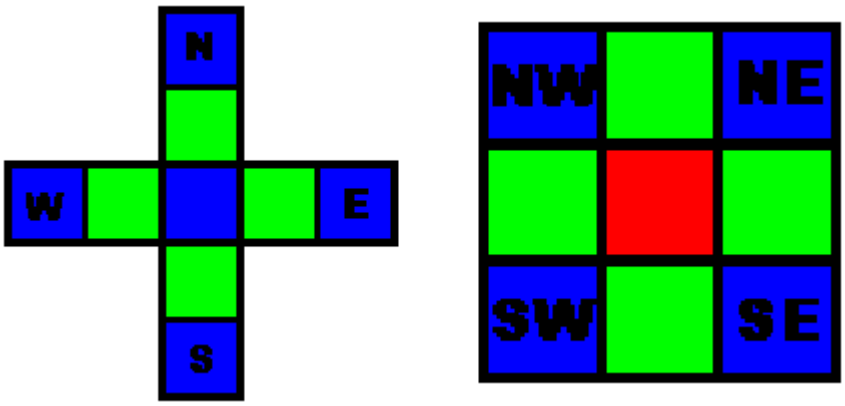

Figure 4: Directions

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 [2].

1.2 Post Processing

Suggested post-processing techniques:

1. **Local Color Ratio Based Post Processing** Using mean values in order to correct unnatural changes in hue values by smoothing the color ratio planes. It stems from the assumptions used in algorithm no. 5. Testing shows that this method performs well when removing false colors, which occur mainly along edges [2].

2. **Median Filtering** We can impose the model introduced by equation 1, by gathering all the color difference values over a square neighborhood around a pixel. Once the differences are gathered, their median is calculated, and then used as an approximation of what the current pixels color difference should be. It is best used when performed only on edge pixels.

1.3 Performance

Reference picture was acquired by sub-sampling it in order to produce a Bayer image. Few tests were conducted [2], such as MMSE, PSNR, blur measure

(using information on the average edges' width), edge slope measure (using information on the edges' height and width) and color artifacts at edges.

Using the above tests, it can be shown that algorithm 8 provides the best average results, when using it with post-processing technique of median filtering 2.

However, if a lower complexity is desired, algorithm 4 is preferred for its good performance [3], and is used by Matlab as default.

References

[1] Wikipedia, the free encyclopedia. Demosaicing, August 2010.

[2] Maschal et al. Review of bayer pattern color filter array (cfa) demosaicing with new quality assessment algorithms. Technical report, U.S. Army Research Laboratory, 2010.

[3] Henrique S. Malvar, Li wei He, and Ross Cutler. High-quality linear interpolation for demosaicing of bayer-patterned color images. In *Proceedings of the IEEE International Conference on Speech, Acoustics, and Signal Processing*, 2004.

[4] R Lukac and K N Plataniotis. Normalized color-ratio modeling for cfa interpolation. *IEEE Transactions on Consumer Electronics*, 2004.

[5] D R Cok. Signal processing method and apparatus for sampled image signals. united states patent 4,630,307, 1987.