Directional Variances Based Demosaicing Method

Joohyeok Kim
Electronics and Computer Engineering
Hanyang University
Republic of Korea
kjh76363@gmail.com

Gwanggil Jeon
Embedded Systems Engineering
Incheon National University
Republic of Korea
gjeon@incheon.ac.kr

Jechang Jeong*
Electronics and Computer Engineering
Hanyang University
Republic of Korea
jjeong@hanyang.ac.kr

***Abstract*—In this paper, we propose a new demosaicing method, which has the improved edge detection method and the refinement scheme. The proposed method finds the interpolation direction based on the directional variances, and then interpolates the missing green components. The missing red and blue components are populated with the use of the fully interpolated green components and color differences. According to the edge direction, two or six neighboring pixels are used to interpolate the red and blue channels. A full colored image, after that, is refined by using median filter with 5×5 cross-shaped kernel. The experimental results show that the proposed algorithm provides a better demosaiced image with relatively low computational complexity.**

Keywords- demosaicing; color filter array; adaptive color plane interpolation

I. INTRODUCTION

A pixel of a full color image is composed of three colors; hence, three separate spectrally selective sensors are required to capture a particular color channel. However, the sensor is one of the most expensive components of a camera system; specifically, it takes about 10-25% of the total cost [1]. For this reason, most cameras use a single sensor covered with a color filter array (CFA) in order to reduce the cost. Fig. 1 shows a popular CFA pattern, known as the Bayer CFA, which is composed of red (*R*), green (*G*), and blue (*B*) filter elements. As one can observe from this CFA, each pixel has only one color component and accordingly the two missing components at each pixel must be estimated. Such an estimation process is called as CFA interpolation or demosaicing.

The simplest method for demosaicing is to use conventional interpolation methods such as bilinear or cubic interpolation [2]-[4]. However, such methods produce some color artifacts because each color channel is independently interpolated without the use of the inter-channel correlation. One solution to consider the inter-channel correlation is to use the color difference rule, which is based on the assumption that the color differences such as *G-R* and *G-B* are quite constant over small regions [2],[5].

Adaptive color plane interpolation (ACPI) proposed in [2] has provided a framework of demosaicing. In order to interpolate a missing green component, ACPI uses the mean term of two neighboring green components and the second order directional Laplacian term of red or blue components two pixels apart on the same column or same row. Effective color interpolation (ECI) calculates the estimates of color differences and utilizes them to interpolate missing components by averaging [6]. Enhanced ECI (EECI) is another method that utilizes color differences for interpolation. In EECI, weight factors on neighbor color differences are utilized for interpolation [7]. EECI shows better results and its complexity is comparable to that of ECI. Recently, voting-based directional interpolation (VDI) is proposed, which adds the voting strategy to determine interpolation direction [8]. A missing pixel is interpolated by using the gradient weights.

In this paper, we propose a new demosaicing method based on ACPI. The proposed method uses variance of directional neighboring pixels to determine interpolation direction of a missing component. Interpolation is performed along the determined direction using the same predictors as those of ACPI.

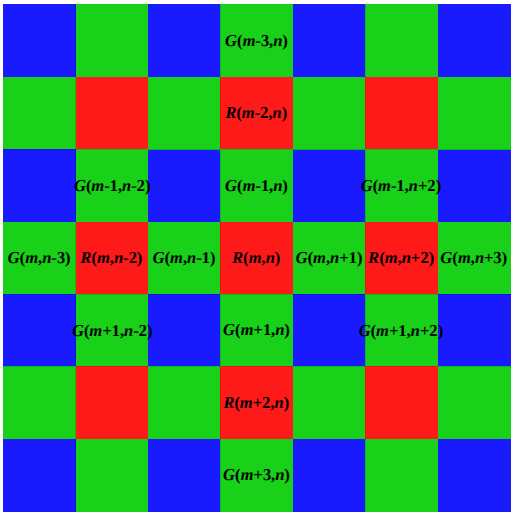


Figure 1. Bayer color filter array.

The remainder of the paper is organized as follows. In Section II, we explain the details of the proposed algorithm, including each color component interpolation and the refinement. Some simulation results are presented and analyzed for comparison in Section III. Finally, we conclude the paper in Section IV.

* Corresponding author

 ISBN: 978-1-61208-320-9

II. PROPOSED METHOD

In the proposed method, we first interpolate the missing green components because the green channel contains important spatial information. After green channel interpolation, red and blue channels are interpolated with the use of the populated green channel. Finally, a fully populated image is refined by median filter.

A. Green Component Interpolation

In order to interpolate the missing green components at the red sampling positions in Fig. 1, three predictors are used as follows:

$$\begin{aligned}\hat{G}_H(m,n) &= \frac{G(m,n-1)+G(m,n+1)}{2} \\ &\quad + \frac{2R(m,n)-R(m,n-2)-R(m,n+2)}{4}, \\ \hat{G}_V(m,n) &= \frac{G(m-1,n)+G(m+1,n)}{2} \\ &\quad + \frac{2R(m,n)-R(m-2,n)-R(m+2,n)}{4}, \\ G_A(m,n) &= \frac{\hat{G}_H+\hat{G}_V}{2}.\end{aligned} \tag{1}$$

These predictors are same as in ACPI. One of them is chosen by edge detection, and used as the estimate of the missing green component at (*m*,*n*). For edge detection, we use the directional variance of neighboring pixels. When calculating the horizontal variance, we exploit green components on the upper and lower lines, the *m*-1th and *m*+1th row, and green and red components on the line that the target pixel is belonging, the *m*th row. Let sets of the positions $\Omega_{H,R0}$, $\Omega_{H,G0}$, $\Omega_{H,G+}$, and $\Omega_{H,G-}$ be defined as

$$\begin{cases}\Omega_{H,G-} = \{(-1,-2),(-1,0),(-1,2)\}, \\ \Omega_{H,G0} = \{(0,-3),(0,-1),(0,1),(0,3)\}, \\ \Omega_{H,G+} = \{(1,-2),(1,0),(1,2)\}, \\ \Omega_{H,R0} = \{(0,-2),(0,0),(0,2)\},\end{cases} \tag{2}$$

and then the variances of pixels on the sets are calculated as follows:

$$\begin{aligned}\sigma^2_{H,R0} &= \frac{1}{3}\sum_{(i,j)\in\Omega_{H,R0}} \left[\mu_{H,R0} - R(m+i,n+j)\right]^2, \\ \sigma^2_{H,G0} &= \frac{1}{4}\sum_{(i,j)\in\Omega_{H,G0}} \left[\mu_{H,G0} - R(m+i,n+j)\right]^2, \\ \sigma^2_{H,G+} &= \frac{1}{3}\sum_{(i,j)\in\Omega_{H,G+}} \left[\mu_{H,G+} - R(m+i,n+j)\right]^2, \\ \sigma^2_{H,G-} &= \frac{1}{3}\sum_{(i,j)\in\Omega_{H,G-}} \left[\mu_{H,G-} - R(m+i,n+j)\right]^2\end{aligned} \tag{3}$$

where $\mu_{H,R0}$, $\mu_{H,G0}$, $\mu_{H,G+}$, and $\mu_{H,G-}$ are the mean values of those pixels, and they are calculated as

$$\begin{aligned}\mu_{H,R0} &= \frac{1}{3}\sum_{(i,j)\in\Omega_{H,R0}} R(m+i,n+j), \\ \mu_{H,G0} &= \frac{1}{4}\sum_{(i,j)\in\Omega_{H,G0}} R(m+i,n+j), \\ \mu_{H,G+} &= \frac{1}{3}\sum_{(i,j)\in\Omega_{H,G+}} R(m+i,n+j), \\ \mu_{H,G-} &= \frac{1}{3}\sum_{(i,j)\in\Omega_{H,G-}} R(m+i,n+j).\end{aligned} \tag{4}$$

The cost for the horizontal direction is defined as sum of the variances as

$$C_H = \sigma^2_{H,R0} + \sigma^2_{H,G0} + \sigma^2_{H,G+} + \sigma^2_{H,G-}. \tag{5}$$

The cost for vertical direction is obtained analogously by using (3)-(5) with the sets defined by

$$\begin{cases}\Omega_{H,G-} = \{(-2,-1),(0,-1),(2,-1)\}, \\ \Omega_{H,G0} = \{(-3,0),(-1,0),(1,0),(3,0)\}, \\ \Omega_{H,G+} = \{(-2,1),(0,1),(2,1)\}, \\ \Omega_{H,R0} = \{(-2,0),(0,0),(2,0)\}.\end{cases} \tag{6}$$

After calculating variances for the horizontal and vertical directions, the missing green component is estimated by

$$\hat{G}(m,n) = \begin{cases}\hat{G}_H(m,n), & \text{if } C_H + \delta < C_V \\ \hat{G}_V(m,n), & \text{else if } C_V + \delta < C_H \\ \hat{G}_A(m,n), & \text{else.}\end{cases} \tag{7}$$

where δ is an offset. With this offset, we can distinguish clear edge from flat or omni-directional edge region. The missing green components on the blue sampling positions are interpolated with the same process except for swapping *R* for *B*.

B. Red/blue Component Interpolation

There are two configurations for interpolating red/blue components at green components as shown in Fig. 2. Because the interpolation process is same for each configuration, we explain only the case of Fig. 2 (a).

The directional variances are also utilized in red/blue component interpolation. That is, we first determine the edge direction at *G*(*m*,*n*) using variances described in the previous subsection. If the direction is horizontal, we interpolate the missing red component at (*m*,*n*) by horizontal average of two color differences at red sampling positions. Because there is no blue component on the horizontal line, we average color differences at six closest blue sampling positions. If the direction is determined as vertical one, we interpolate the

missing red component with six color differences, and the missing blue component with two color differences. When the pixel at (*m*,*n*) is not in edge region, two color differences are used as in ACPI. Let *KR* be the color difference between green and red, and *KB* be the color difference between green and blue. Then, this process can be represented by

$$\text{If } C_H + \delta < C_V$$

$$\hat{R}(m,n) = G(m,n) + \frac{KR(m,n-1) + KR(m,n+1)}{2}$$

$$\hat{B}(m,n) = G(m,n) + \frac{1}{6}\sum_{j=-2,0,2} KB(m-1,n+j) + KB(m+1,n+j)$$

$$\text{elseif } C_V + \delta < C_H$$

$$\hat{R}(m,n) = G(m,n) + \frac{1}{6}\sum_{i=-2,0,2} KR(m+i,n-1) + KR(m+i,n+1)$$

$$\hat{B}(m,n) = G(m,n) + \frac{KB(m-1,n) + KB(m+1,n)}{2}$$

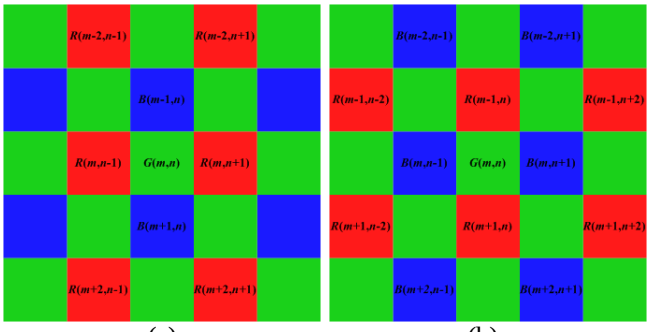
$$\text{else}$$

$$\hat{R}(m,n) = G(m,n) + \frac{KR(m,n-1) + KR(m,n+1)}{2}$$

$$\hat{B}(m,n) = G(m,n) + \frac{KB(m-1,n) + KB(m+1,n)}{2}$$

For interpolating the missing red components at blue sampling positions and the missing blue components at red sampling positions, the average on color differences of four diagonal neighbors is used as the estimates as:

$$\begin{aligned} R(m,n) &= G(m,n) + \big[KR(m-1,n-1) + KR(m-1,n+1) \\ &\quad + KR(m+1,n-1) + KR(m+1,n+1)\big]/4 \\ B(m,n) &= G(m,n) + \big[KB(m-1,n-1) + KB(m-1,n+1) \\ &\quad + KB(m+1,n-1) + KB(m+1,n+1)\big]/4. \end{aligned} \tag{8}$$



(a) (b)

Figure 2. Two configurations for red/blue component interpolation: (a) horizontal GR line, (b) horizontal GB line.

C. *Refinement*

The fully populated image is refined so as to improve image quality. In the proposed method, we use a simple refinement scheme instead of iterative schemes proposed in [9]-[11]. The following median filter is applied to the estimated green component at the red sampling position as shown in Fig. 1 to suppress color artifacts:

$$\tilde{G}(m,n) = \text{median}\begin{bmatrix} KR(m-2,n), KR(m-1,n), KR(m,n), \\ KR(m+1,n), KR(m+2,n), KR(m,n-2), \\ KR(m,n-1), KR(m,n+1), KR(m,n+2) \end{bmatrix}. \tag{9}$$

We choose not a 3×3 window, but a 5×5 cross-shaped window because the 3×3 window includes the *KR* values generated by the estimated green and red components. The green components at the blue sampling positions are similarly obtained by exchanging *KR* for *KB*. After refining all the estimated green components, we refine the estimated red and blue components using the line average of the color differences proposed in ACPI.

III. Experimental Results

To evaluate the performance of the proposed demosaicing method, we simulated it and the four existing methods: ACPI [2], ECI [6], EECI [7], and VDI [8]. The 24 digital color images shown in Fig. 3 were utilized as a set of testing images, each having 768×512 pixels. As a measure, the color peak signal-to-noise ratio (CPSNR) were used as defined as

$$\text{CPSNR} = 10\log_{10}\left(\frac{255^2}{\text{CMSE}}\right) \tag{10}$$

where

$$\text{CMSE} = \frac{1}{3HW}\sum_{k=r,g,b}\sum_{m=1}^{H}\sum_{n=1}^{W}\big(I_o(m,n,k) - I_d(m,n,k)\big)^2. \tag{11}$$

In this equation, I_o and I_d represent the original and the demosaiced images of size $H \times W$. All the testing images are sampled according to the Bayer CFA pattern, and then interpolated. To measure the reconstructed image quality, the interpolated images are compared to the original images.

Table I tabulates the CPSNR results of different methods. It is shown that the proposed algorithm achieved higher PSNR measures than the other methods. The proposed method achieved the best CPSNR results among the compared methods while ACPI, which is the base of the proposed method, showed the worst CPSNR scores. The difference was of 2.95 dB on average and the proposed method showed higher PSNR scores for all the testing images. This signifies that the edge detection scheme of the proposed method outperforms the original one.

 ISBN: 978-1-61208-320-9

Figure 3. Test images (referred to as image 1 to image 24, enumerated from left-to-right and top-to-bottom.)

TABLE 1. Comparison of CPSNR Results

Image	ACPI	ECI	EECI	VDI	Prop.
1	33.48	33.43	37.81	34.13	**38.35**
2	38.50	36.32	**40.36**	39.30	40.26
3	40.44	38.35	**42.74**	40.92	42.39
4	38.56	38.41	**40.53**	38.88	40.15
5	34.59	34.60	**38.09**	35.23	37.96
6	34.78	34.19	38.11	35.99	**38.97**
7	40.55	38.70	**42.60**	41.28	41.99
8	31.85	30.53	35.22	33.03	**36.22**
9	40.12	38.77	42.76	41.00	**42.92**
10	39.47	39.07	**42.43**	40.60	42.05
11	36.06	35.95	39.54	36.91	**39.61**
12	40.45	38.85	**42.67**	41.30	42.25
13	29.89	31.30	34.27	30.09	34.81
14	35.51	34.59	**37.60**	36.11	36.68
15	37.36	35.94	39.15	37.44	**39.27**
16	38.29	36.35	41.28	39.88	**41.62**
17	38.60	39.05	**41.79**	38.87	41.75
18	33.68	35.18	36.82	33.61	**37.02**
19	36.87	35.45	40.12	37.82	**40.31**
20	36.83	36.02	**40.53**	38.44	40.22
21	34.89	35.47	38.90	35.84	**39.19**
22	35.92	36.28	**38.40**	36.69	38.19
23	38.84	38.78	**41.16**	40.77	41.12
24	32.03	33.56	34.58	31.93	**34.64**
average	36.56	36.05	39.48	37.34	**39.51**

Fig. 4 shows the demosaiced images of image 19. Fig. 4 (a) is the cropped version of the original image, and Fig. 4 (b)-(f) are images reconstructed by different methods. The demosaiced image from ECI suffered from color artifacts, and that from EECI had the relatively reduced artifacts but they are still serious. VDI showed the much reduced results, but it cannot avoid zig-zag artifacts. The proposed method reduced those artifacts: color artifacts and zig-zag artifacts; hence, the demosaiced image was smoother than that of VDI. Compared to ACPI, the proposed method showed much better result although both used the same predictors. This demonstrates again that the edge detection and the refinement scheme of the proposed method are excellent.

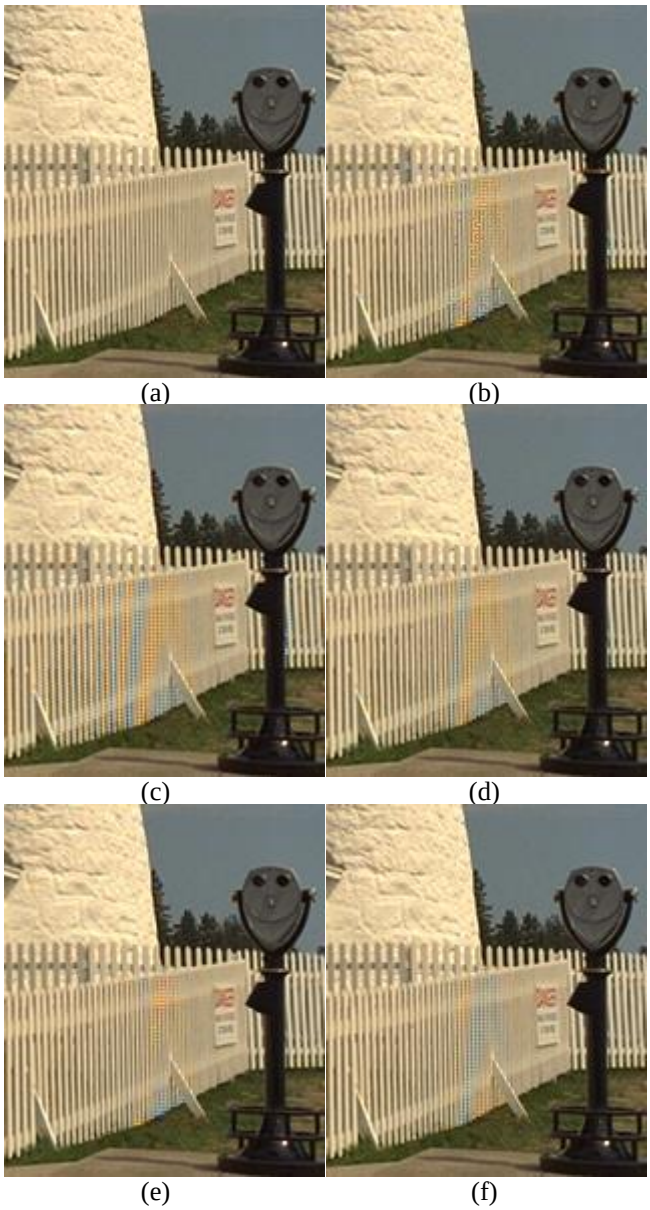


Figure 4. Comparison of Reconstructed Images: (a) original, (b) ACPI, (c) ECI, (d) EECI, (e) VDI, (f) the proposed method.

In order to evaluate the computational complexity, we compared the number of arithmetic operations required for generating a fully populated image from a CFA image. To reduce the complexity, the variance calculation was performed as

$$\begin{aligned}\sigma^2 &= \frac{1}{N}\sum_{i=1}^{N}\left[x_i - \bar{x}\right]^2 \\ &= \frac{1}{N}\sum_{i=1}^{N} x_i^{\,2} - \bar{x}^2,\end{aligned} \tag{12}$$

and the other computations were also optimized. As one can observe from Table II, the proposed method required 6*HW* multiplication operations, which was much less compared to those of EECI and VDI, each requiring 28*HW* and 13*HW*. In the simulations using MATLAB, the proposed method consumed 0.671s while EECI and VDI took 0.925s and 0.702s, respectively, to populate an image. The proposed method required 6.295*HW* comparison operations. In our analysis, we found that it was from the refinement step. In a simulation without the refinement, the number of comparison operations required was 2*HW*, CPU time was reduced to 0.191s, and the average CPSNR result was 37.62dB.

TABLE II. Comparison of Computational Complexity

Image	ACPI	ECI	EECI	VDI	Prop.
ADD	6.75 *HW*	10 *HW*	58 *HW*	42 *HW*	24.5 *HW*
SHT	2.5 *HW*	4 *HW*	2 *HW*	4.5 *HW*	7 *HW*
CMP	0.5 *HW*	0 *HW*	0 *HW*	1.5 *HW*	6.295 *HW*
MUL	0 *HW*	0 *HW*	28 *HW*	13 *HW*	6 *HW*

IV. CONCLUSION

In this paper, we presented a new demosaicing method. In order to determine the interpolation direction, the proposed method used the horizontal and vertical directional variances. Because the predictors proposed in ACPI is fast and has a good performance, we interpolated a missing pixel using the predictors of ACPI along the direction determined by our proposed edge detection method. For interpolating the red and blue channels, we used again the directional variances. Based on the direction determined by the directional variances, two or six neighboring pixels were used for interpolation. A fully interpolated image, then, went through the refinement step. In the simulation for comparison with other conventional methods, it was demonstrated that the proposed algorithm has low computational complexity, while showing better quality than the tested conventional methods.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea(NRF) Grant funded by the Korean Government(MOE) (NRF-2011-0011312).

REFERENCES

[1] J. Adams, K. Parulski, and K. Spaulding, “Color processing in digital cameras,” IEEE Micro, vol. 18, no. 6, Nov.-Dec. 1998, pp. 20–30.

[2] J. H. Hamilton and J. E. Adams, “Adaptive Color Plane Interpolation in Single Sensor Color Electronic Camera,” U.S. Patent 5 629 734, 1997.

[3] T. Sakamoto, C. Nakanishi, and T. Hase, “Software pixel interpolation for digital still camera suitable for a 32-bit MCU,” IEEE Trans. Consum. Electron., vol. 44, no. 4, Nov. 1998, pp. 1342–1352.

[4] J. Adams, “Interactions between colorplane interpolation and other image processing functions in electronic photography,” Proc. SPIE, vol. 2416, Mar. 1995, pp. 144–151.

[5] E. Chang, S. Cheung, and D. Y. Pan, “Color filter array recovery using a threshold-based variable number of gradients,” Proc. SPIE, vol. 3650, Mar. 1999, pp. 36–43.

[6] S. C. Pei and I. K. Tam, “Effective color interpolation in CCD color filter arrays using signal correlation,” IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 6, June 2003, pp. 503–513.

[7] L. Chang and Y. P. Tam, “Effective use of spatial and spectral correlations for color filter array demosaicing,” IEEE Trans. Consum. Electron., vol. 50, no. 1, Feb. 2004, pp. 355–365.

[8] X. Chen, G. Jeon, and J. Jeong, "Voting-based directional interpolation method and its application to still color image demosaicking," IEEE Trans. Circuits Syst. Video Technol., accepted, 2013.

[9] R. Kimmel, “Demosaicing: Image reconstruction from color CCD samples,” IEEE Trans. Image Process., vol. 8, Sep. 1999, pp. 1221–1228.

[10] B. K. Gunturk, Y. Altunbasak, and R. Mersereau, “Color plane interpolation using alternating projections,” IEEE Trans. on Image Process., vol. 11, no. 9, Sep. 2002, 997–1013.

[11] K. Hirakawa and T. W. Parks, “Adaptive homogeneity-directed demosaicking algorithm,” IEEE Trans. Image Process., vol. 14, no. 3, Mar. 2005, pp. 360–369.

 ISBN: 978-1-61208-320-9