

# ADAPTIVE DIRECTIONAL DEMOSAICING

Tae-young Jung, Jechang Jeong

Department of Electronics and Computer Engineering, Hanyang University,  
17 Haengdang-dong, Seongdong-gu, Seoul, Korea  
master@mytears.org, jjeong@ece.hanyang.ac.kr

Keywords: Demosaicing, Bayer pattern, directional, weighted sum, color filter, interpolation

Abstract: Most of commercially available digital cameras use single sensor array with color filter to reduce the size and the cost. The image obtained with single sensor array has only a single color component per pixel. For this reason, we need to interpolate missing two color components per pixel. In generally, this process is called as demosaicing or color filter interpolation. This paper proposes the adaptive directional demosaicing algorithm. This algorithm achieves better visual quality with efficient weighting function. We compare the results with results of various algorithms, the proposed algorithm shows the best results in subjective qualities and PSNR.

## 1 INTRODUCTION

The single sensor array with the color filter array is popularly used in digital camera. However, the image obtained with single sensor array has only a single color component per pixel. In order to reconstruct the color filtered image to the full color image, we need to estimate the missing two color components per pixel. This process is called as demosaicing or color filter interpolation. Bayer pattern as shown in figure 2 is the most frequently used color filter pattern. The human visual system reacts more sensitively on the green components than the red or the blue components. For this reason, half of the pixels are assigned to the green channel, and the red and blue channels share the other half of the pixels. In this paper, we focus on the development of demosaicing algorithm for the Bayer color filter pattern.

In recent years, the demosaicing algorithms are actively developed. Nearest neighborhood interpolation and bi-linear interpolation are the simplest algorithms for the demosaicing. They are easy to implement, however they cause false color errors and the rainbow artifacts as shown in figure

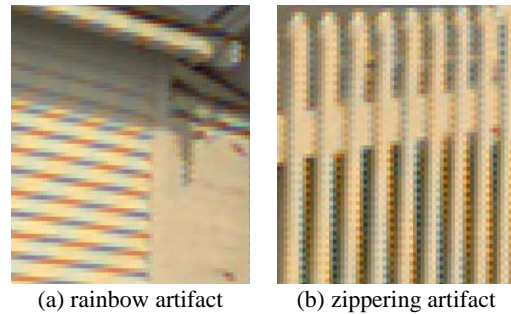


Figure 1: Examples of color artifact.

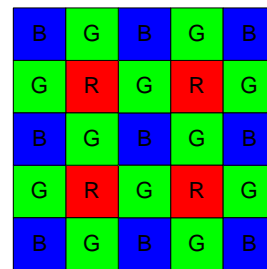


Figure 2: Bayer color filter pattern.

1(a). Cok found that the color ratio varies slowly, and proposed the demosaicing algorithm using color ratio. Likewise, Pei *et al.* found that the color difference shows the similar characteristic in natural image. However, the color difference or the color ratio algorithms still cause the zippering artifacts as

---

This work was sponsored by ETRI SoC Industry Promotion Center, Human Resource Development Project for IT SoC Architect.

shown in figure 1(b) in complex edge region. To improve the quality in edge region, the gradient based algorithms are developed.

## 2 PROPOSED ALGORITHM

### 2.1 Step 1: Interpolating green value on red/blue pixel

The missing pixels are interpolated by nearby pixels. In this paper, we use twelve nearby pixels to increase the edge sensitivity. The directions are illustrated in figure 3. The directions are numbered from 1 to 12 for convenience of expression. In case of interpolating the green value on B33 in figure 3, G23, G32, G34, and G43 are generally used. Additionally, we consider G12, G13, G21, G25, G41, G45, G52, and G54 to detect the edge more sensitively. The vertical offset,  $v_n$ , and horizontal offset,  $h_n$ , of nearby pixels used in this step are listed in Table 1. The difference,  $d_n$ , between the samples in direction  $n$  on position  $(i, j)$  is defined as in (1).

$$d_n(i, j) = |P(i+h_n, j+v_n) - P(i-h_n, j-v_n)| + |P(i+2h_n, j+2v_n) - P(i, j)| \quad (1)$$

where,  $P(i, j)$  denotes the value on position  $(i, j)$ . The vertical gradient,  $dv(i, j)$ , and the horizontal gradient  $dh(i, j)$ , are defined as following.

$$dh(i, j) = |2P(i, j) - P(i+2, j) - P(i-2, j)| + |P(i+1, j) - P(i-1, j)| \quad (2)$$

$$dv(i, j) = |2P(i, j) - P(i, j+2) - P(i, j-2)| + |P(i, j+1) - P(i, j-1)| \quad (3)$$

Since the neighbor pixels within homogeneous region show similar characteristic, the directional difference,  $D_n$ , in direction  $n$  is defined as in (4).

$$D_n(i, j) = \begin{cases} d_n(i-2, j) + d_n(i+2, j) + d_n(i, j) & , dh < dv \\ d_n(i, j-2) + d_n(i, j+2) + d_n(i, j) & , dh > dv \\ d_n(i, j) & , otherwise \end{cases} \quad (4)$$

The weight,  $w_n(i, j)$ , for the nearby pixel in direction  $n$  is defined as in (5).

Table 1: The offsets of nearby pixels in step 1, 3, and 4.

$n$	$h_n$	$v_n$	$n$	$h_n$	$v_n$
1	0	-1	7	+2	+1
2	+1	0	8	+1	+2
3	-1	0	9	-1	+2
4	0	+1	10	-2	+1
5	+1	-2	11	-2	-1
6	+2	-1	12	-1	-2

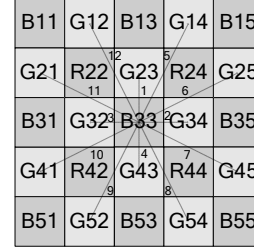


Figure 3: Directions of nearby pixels in step 1, 3, and 4.

$$w_n(i, j) = 1 / (1 + D_n^2(i, j)) \quad (5)$$

The missing green value on the blue sample is interpolated by (6). The color difference,  $K_n^B(i, j)$ , between the green value  $G$  and the blue value  $B$  is obtained by subtracting the blue value from the green value in direction  $n$ . The blue value on green samples is required, however there is no blue sample, average of the nearby blue samples is used instead.

$$G(i, j) = B(i, j) + \frac{\sum_{n=1}^{12} w_n(i, j) K_n^B(x, y)}{\sum_{n=1}^{12} w_n(i, j)} \quad (6)$$

where,  $K_n^B(i, j) = G(i+h_n, j+v_n) - B(i+h_n, j+v_n)$ .

Similarly, the missing green value on the red sample is interpolated by (7). The color difference,  $K_n^R(i, j)$ , between the green value  $G$  and the red value  $R$  is obtained by subtracting the red value from the green value in direction  $n$ . The red value on green samples is required, however there is no red sample, average of the nearby red samples is used instead.

$$G(i, j) = R(i, j) + \frac{\sum_{n=1}^{12} w_n(i, j) K_n^R(x, y)}{\sum_{n=1}^{12} w_n(i, j)} \quad (7)$$

where,  $K_n^R(i, j) = G(i+h_n, j+v_n) - R(i+h_n, j+v_n)$ .

Table 2: The offsets of nearby pixels in step 2 (vertical).

$n$	$h_n$	$v_n$	$n$	$h_n$	$v_n$
1	-1	0	4	-1	+2
2	+1	0	5	+1	-2
3	-1	-2	6	+1	+2

Table 3: The offsets of nearby pixels in step 2 (horizontal).

$n$	$h_n$	$v_n$	$n$	$h_n$	$v_n$
1	0	-1	4	+2	-1
2	0	+1	5	-2	+1
3	-2	-1	6	+2	+1

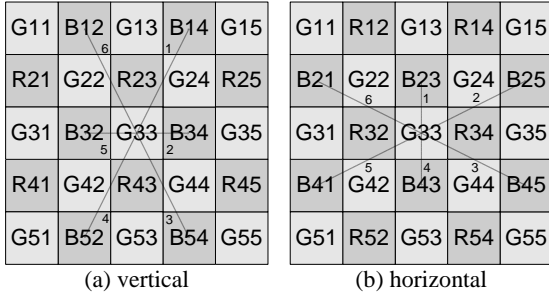


Figure 4: Directions of nearby pixels in step 2.

## 2.1 Step 2: Interpolating red/blue value on green pixel

The missing red/blue values on the green samples are interpolated in this step. As shown in figure 4(a) and figure 4(b), nearby six pixels are available in this step. If the desired colors are laid in vertical direction as shown in figure 4(a), the offsets,  $v_n$  and  $h_n$ , listed in Table 2 are used. In the other case as shown in figure 4(b), the offsets,  $v_n$  and  $h_n$ , listed in Table 3 are used. The missing blue value on green sample is defined as in (8)

$$B(i, j) = G(i, j) - \frac{\sum_{n=1}^6 w_n(i, j) K_n^B(x, y)}{\sum_{n=1}^6 w_n(i, j)} \quad (8)$$

where,  $K_n^B(i, j)$  is obtained in (6),  $w_n(i, j)$  is obtained in (5). Similarly, the missing red value on green sample is defined as in (9).

$$R(i, j) = G(i, j) - \frac{\sum_{n=1}^6 w_n(i, j) K_n^R(x, y)}{\sum_{n=1}^6 w_n(i, j)} \quad (9)$$

where,  $K_n^R(i, j)$  is obtained in (7),  $w_n(i, j)$  is obtained in (5).

## 2.3 Step 3: Interpolating red/blue value on blue/red pixel

Since the missing red/blue values on the green samples are already interpolated, every nearby pixel is available in this step. For the convenience of implementing, we consider twelve directions as in step 1. The missing blue value on the red sample is interpolated as in (10).

$$B(i, j) = G(i, j) - \frac{\sum_{n=1}^{12} w_n(i, j) K_n^B(x, y)}{\sum_{n=1}^{12} w_n(i, j)} \quad (10)$$

where  $K_n^B(i, j)$  is obtained in (6),  $w_n(i, j)$  is obtained in (5). Similarly, the missing red value on green sample is defined as in (11).

$$R(i, j) = G(i, j) - \frac{\sum_{n=1}^{12} w_n(i, j) K_n^R(x, y)}{\sum_{n=1}^{12} w_n(i, j)} \quad (11)$$

where,  $K_n^R(i, j)$  is obtained in (7),  $w_n(i, j)$  is obtained in (5).

## 2.4 Step 4: Re-interpolating green value on red/blue pixel

In step 1, the missing green values on blue/red samples are interpolated with an inaccurate blue/red values, however better blue/red values are obtained in step 2 and 3, recalculate the missing green value with the interpolated blue/red values.

## 3 SIMULATION RESULTS

We compared the proposed demosaicing algorithm with some conventional algorithms for various test images. For the image simulation, we used 24 images from Kodak image database. These images are shown in figure 6.

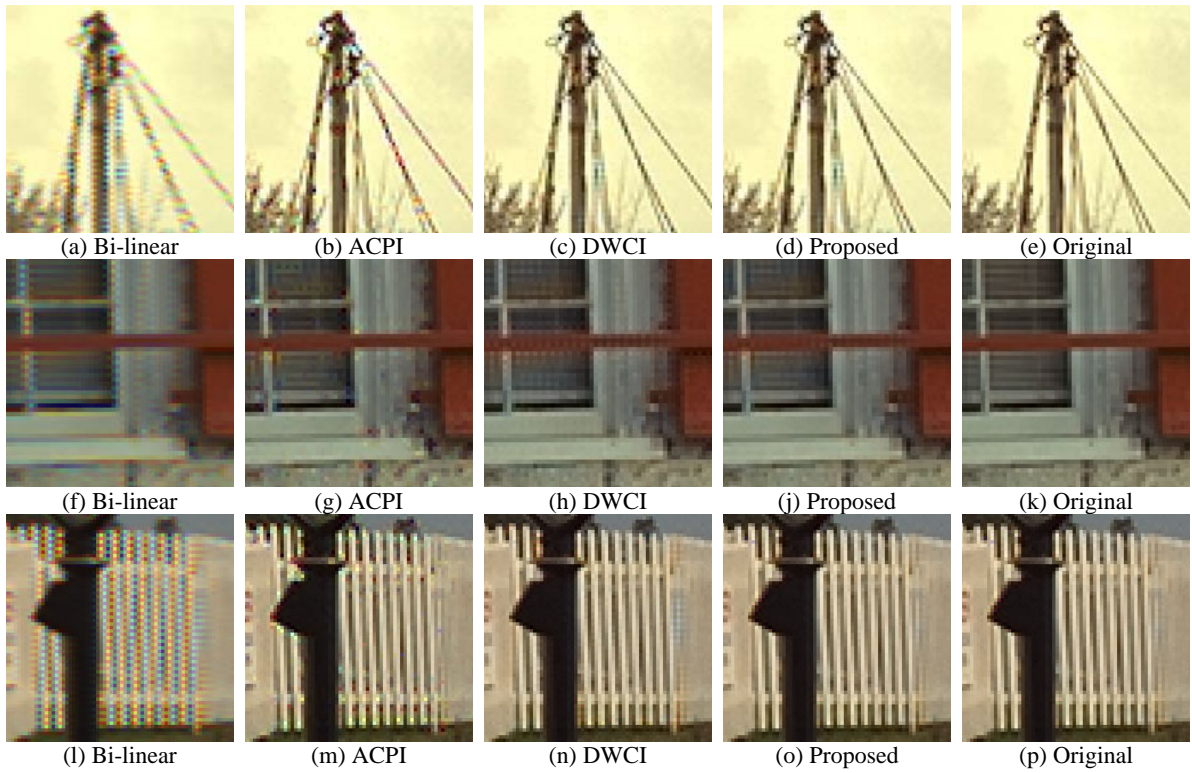


Figure 5: Reconstructed image and the original image.

Images are first down sampled to the Bayer pattern, then reconstructed to the full color image. Figure 5(a, f, l) show the magnified details of the interpolated images of bi-linear interpolation. Figure 5(b, g, m) show the magnified details of adaptive color plain interpolation (ACPI). Figure 5(c, h, n) show the magnified details of directionally weighted color interpolation (DWCI). Figure 5(d, j, o) show the magnified details of proposed algorithm. Figure 5(e, k, p) show the magnified details of original image. Table 4 shows the PSNR comparison for the bilinear interpolation, ACPI, DWCI, and the proposed algorithm. Every case, the proposed algorithm achieves the best PSNR.

## 4 CONCLUSIONS

This paper proposes the adaptive directional demosaicing algorithm for the Bayer color filter array, we use homogeneity of nearby pixels, and color correlation concepts. Proposed algorithm shows the less rainbow artifacts and the zippering artifacts than the conventional algorithms.



Figure 6: Set of images used in simulation. Images are numbered from 1 to 24, in order of left-to-right, top-to-bottom

Table 4. PSNR (dB) comparison.

	Bi-linear			ACPI			DWCI			Proposed		
	R	G	B	R	G	B	R	G	B	R	G	B
1	24.678	28.546	25.292	31.585	31.904	31.578	36.128	40.409	36.079	36.437	41.271	36.195
2	30.539	34.933	31.399	36.391	37.235	37.100	36.730	38.006	38.160	38.001	39.481	39.034
3	31.110	37.797	32.653	37.545	38.318	37.582	40.671	41.411	39.500	40.802	41.984	39.615
4	30.890	33.473	32.199	35.244	36.751	37.215	36.605	39.766	40.208	37.141	41.006	40.180
5	25.174	28.142	25.986	31.575	30.593	31.093	35.665	36.956	34.262	34.405	36.550	33.456
6	25.991	31.950	26.734	32.768	33.088	32.612	36.708	40.410	35.590	36.885	40.885	35.686
7	30.701	35.493	31.998	36.804	36.417	36.368	40.884	40.666	38.274	40.719	41.627	38.428
8	22.294	24.430	22.713	30.473	30.962	30.168	32.749	36.850	32.398	33.400	37.528	32.811
9	30.037	32.259	31.336	37.039	38.066	37.806	40.086	42.395	39.360	40.310	43.338	39.661
10	30.090	32.024	31.187	36.659	37.084	36.868	39.681	41.919	39.069	38.891	41.409	38.353
11	27.503	31.422	28.391	33.877	34.087	34.198	37.278	39.504	37.335	37.516	40.265	37.299
12	30.154	35.483	32.037	37.806	38.853	37.904	40.656	42.102	39.917	41.031	42.799	39.941
13	22.840	26.423	23.117	28.736	28.655	28.290	34.182	37.362	32.381	33.340	37.487	31.864
14	27.440	31.725	28.203	32.973	33.072	32.454	34.773	35.984	34.041	35.639	37.163	34.404
15	28.885	31.971	30.374	35.063	36.282	35.292	36.009	37.823	37.992	36.666	38.648	37.932
16	29.000	36.843	29.684	35.876	36.695	36.218	39.434	42.709	38.641	40.031	43.416	39.075
17	31.437	33.533	30.766	36.761	36.002	36.109	40.327	42.220	38.608	39.811	42.469	38.335
18	27.173	28.889	26.943	32.139	32.064	31.990	35.626	37.830	34.585	35.319	38.451	34.256
19	27.020	29.154	27.150	35.042	35.306	35.065	37.213	41.256	36.834	38.077	42.035	37.623
20	29.071	32.969	28.438	35.828	36.195	34.199	39.478	41.009	36.799	38.940	41.056	36.918
21	27.156	31.785	27.532	33.072	33.072	32.866	38.016	40.712	36.362	37.909	41.280	36.273
22	29.167	31.521	29.338	34.399	34.663	33.752	37.164	37.975	35.547	37.353	39.016	35.755
23	31.316	37.284	33.720	37.156	38.326	37.326	38.417	40.345	39.094	38.753	41.428	38.712
24	26.323	28.686	25.262	31.392	31.095	29.543	34.033	35.761	31.845	33.861	36.127	31.511

## REFERENCES

- J. A. Weldy, "Optimized design for a single-sensor color electronic camera system," in *Proc. SPIE*, pp. 300-307, 1988.
- B. E. Bayer, "Color imaging array," *U.S. Patent 3971065*, Jul. 1976.
- D. R. Cok, "Signal Processing method and apparatus for producing interpolated chrominance values in a sampled color image signal," *U.S. Patent 4642678*, Feb. 1987.
- B. Eamanath, W.E. Synder, and G. L. Bilbro, "Demosaicking methods for Bayer color array," *Journal of Electronic Imaging*, vol. 11, no. 3, pp. 306-315, Jul. 2002.
- Bahadir K. Gunturk, Yucel Altunbasak, and Russell M. Mersereau, "Color plane interpolation using alternating projections," *IEEE Trans. Image Process.*, vol. 11, no. 9, pp 997-1013, Sept. 2002.
- N. Kehtarnavaz, H. Oh, and Y. Yoo, "Color filter array interpolation using correlations and directional derivatives," *Journal of Electronic Imaging*, vol. 12, no. 4, pp 621-632, Oct. 2003.
- Hung-An Chang and Homer Chen, "Directionally weighted color interpolation for digital cameras," in *proc. IEEE Int. Symp. on Cir. and Syst.*, vol. 6, pp. 6284-6287 23-26, May 2005.
- D. Darian Muresan and Thomas W. Parks, "Demosaicing using optimal recovery," *IEEE Transactions Image Processing*, vol. 14, no. 2, pp 267-278, Feb. 2005.
- Wonjae Lee, Seongjoo Lee, and Jaeseok Kim, "Cost-effective color filter array demosaicing using spatial correlation," *IEEE Trans. Consumer Electronics*, vol. 52, no. 2, pp 547-554, May 2006.