Highly Effective Iterative Demosaicing Using Weighted-Edge and Color-Difference Interpolations

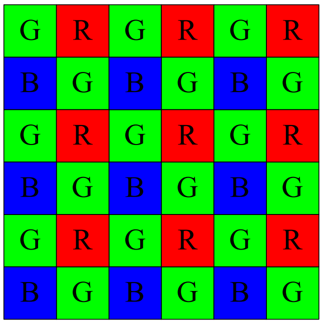
Chung-Yen Su

Abstract — *Demosaicing is a process of obtaining a full-color image by interpolating the missing colors of an image captured from a single sensor color filter array. This paper provides an effective and low-complexity iterative demosaicing algorithm applying a weighted-edge interpolation to handle green pixels followed by a series of color-difference interpolation to update red, blue, and green pixels. Based on our experiments of images, we enable the algorithm a well-designed stopping condition and pre-determine the proper weights of interpolation. Experimental results show that the proposed method performs much better than three state-of-the-art demosaicing techniques in terms of both computational cost and image quality. In comparison to the algorithm of successive approximation, the algorithm proposed here reduces mean squared error up to 14.5% while requiring computational cost only 22% on average. That is, it takes less time but performs better.* [1]

Index Terms — Iterative demosaicing, color interpolation, weighted edge interpolation, color difference.

I. INTRODUCTION

Manipulating a digital still camera (DSC) has been very popular in the field of consumer electronics. In order to reduce the hardware cost, many DSCs use a single sensor equipped with a color filter array (CFA) to capture any of the three primary color components, *R* (Red), *G* (Green), or *B* (Blue), on each pixel location. Among the various suggested CFAs [1], the Bayer CFA pattern [2] shown in Fig. 1 is the most prevalent one, where *G* pixels occupy half of all, and *R* and *B* pixels share the others. A representation of a full-color image needs all the information from the three colors on each pixel location. As a result, the missing two colors on each pixel location have to be interpolated back to get a full-color image. The process of interpolating the missing colors is called as demosaicing or color interpolation whose main objective aims to reconstruct the missing colors as closely to the original ones as possible while keeping the computational complexity as low as possible.



G	R	G	R	G	R
B	G	B	G	B	G
G	R	G	R	G	R
B	G	B	G	B	G
G	R	G	R	G	R
B	G	B	G	B	G

Fig. 1. Bayer CFA pattern

In the lately decade, many algorithms have been proposed. These algorithms can be classified into two categories: noniterative [3]-[9] and iterative [11]-[14]. In general, iterative demosaicing algorithms perform better than noniterative ones in mean squared error (MSE) metric, but pay the price for increasing moderate computational cost because iterative algorithms often have to handle the missing colors on each pixel location many times. The basic assumption of an iterative demosaicing is that an improved estimate of the missing colors in one component will lead to an improved estimate of the missing colors in the others. Therefore, we can begin a rough guess and iteratively refine the values of missing colors. To achieve this, the prerequisite is that the correlation model is sufficiently accurate. The exploitation of interplane correlation can be done on the color ratio rule [11] or on the color difference rule [5][13][14]. The fundamental concept of these rules relates to smooth hue transition. For example, the assumption of color difference rule is that the difference signals of both *G-R* and *G-B* are locally constant.

The successive approximation (SA) algorithm proposed in [14] adopts the color difference rule due to its lower computational complexity and better fits to linear interpolation models. It has shown that SA outperforms the previous iterative demosaicing techniques in terms of both computational cost and image quality. The algorithm of SA includes four main stages: 1) initialization, 2) spatial classification, 3) iteration, and 4) termination. In the initialization stage, bilinear interpolation [4] is used to handle *R* and *B* pixels and edge-sensitive interpolation [3] is used to fill *G* pixels. The color difference signals of both *G-R* and *G-B* are then used for spatial classification. An image is divided into low-aliasing and high-aliasing regions after the spatial classification. In the iteration stage, the color-difference interpolation [5] is applied to update *R*, *B*, and *G* pixels. Finally, the variance of each region of each color plane is evaluated and then the largest variance of each region is compared with a pre-determined threshold. If the largest variance of each region is smaller than the threshold, terminate; otherwise repeat stage 3 until this stopping condition is met.

[1] This work was supported in part by the National Science Council, Taiwan, R.O.C., under Contract NSC-94-2213-E-003-003.

C.-Y. Su is with the Institute of Applied Electronic Technology, National Taiwan Normal University, Taipei 10612, Taiwan, R.O.C. (e-mail: scy@ntnu.edu.tw).

Contributed Paper
Manuscript received March 16, 2006

However, the aforementioned iterative techniques do not probe into the influence of initialization on their final results in the query whether the optimal interpolation to the initial condition yields the smallest MSE value after iteration or not. To stay with this concern, our experiments provide a considered answer by realizing that the following iteration is based on a color model but not accurate enough though (due to the lack of actual values of missing colors). Moreover, even if the variance of each color component is gradually decreased, the MSE performance in each color component is not such the case. From the observation of our tested images, we gain insight into that the MSE values generally reach the smallest ones as the iteration number is equal to 3, and that the values then increase with the iteration number increasing. Besides these, we also find pursuing a low-variance situation in high-aliasing region may not be realistic because it requires lots of iteration and computation, only to contribute a little MSE reduction.

To cope with the difficulties above and to reduce computational cost, we propose a low-complexity iterative demosaicing algorithm based on a weighted-edge interpolation and the color-difference interpolation [5]. The proposed algorithm differs from SA in four aspects: 1) the standard bilinear interpolation [4] to interpolate the missing *R* and *B* pixels is not used any longer; 2) a weighted-edge interpolation is proposed to handle *G* pixels instead of the edge-sensitive interpolation [3], which provides a mechanism to adjust output performance; 3) the spatial classification is omitted so that lots of computation for classification and for the pixel update of high-aliasing region is able to be skipped, which leads to the execution of demosaicing speeded up; 4) the updates of *R*, *B*, and *G* pixels are separately processed, which makes the pixel update more reasonable.

The rest of the paper is organized as follows. Section II provides a brief overview of the proposed algorithm. Section III describes our method for filling *G* pixels in initialization and addresses the color-difference interpolation [5] used in the following pixel update stages. Experimental results are shown in Section IV. Finally, we conclude this paper in Section V.

II. Iterative Demosaicing Algorithm

The proposed iterative demosaicing algorithm includes three stages: 1) initialization, 2) refinement, and 3) iteration. The block diagram of our algorithm is shown in Fig. 2. The main merit of the proposed algorithm is of low complexity because it does not include nonlinear or filtering operations.

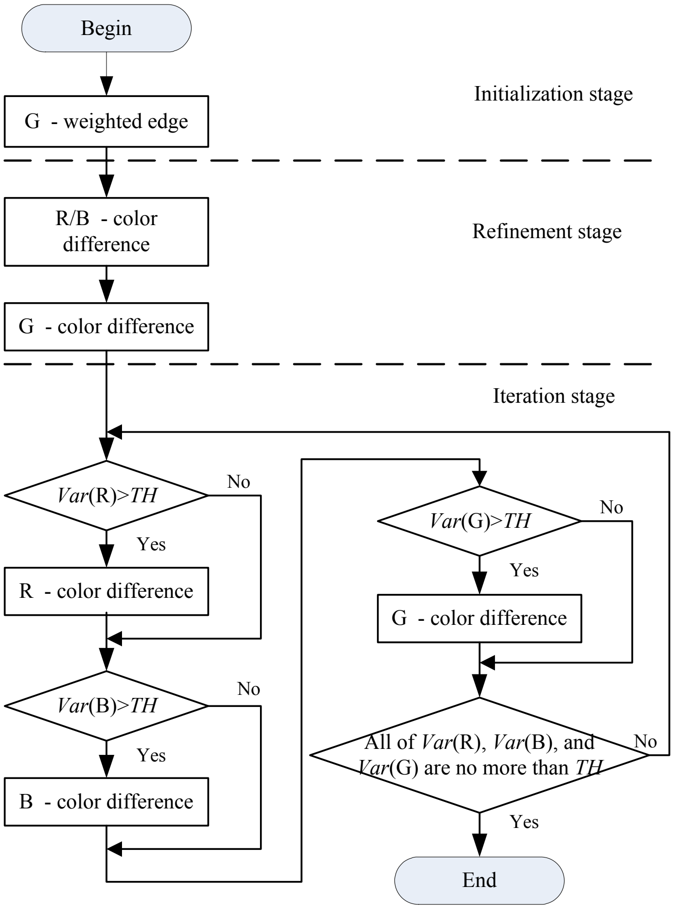


Fig. 2. The proposed iterative demosaicing algorithm

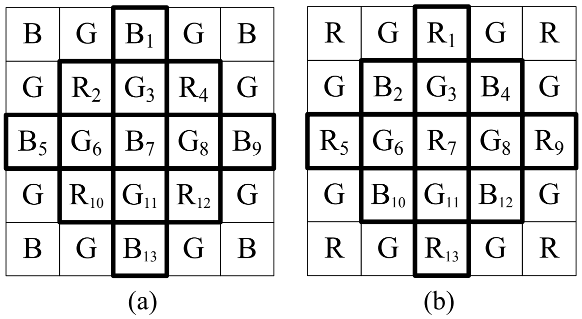
In the initialization stage, the *G* channel is filled by using a weighted-edge interpolation, a technique of intraplane interpolation. The purpose of this approach is to well handle the sharp edges in *G* plane. The weights are pre-determined first in order to reduce computational load. In the refinement stage, the *R* and *B* pixels are first filled and then the *G* pixels are updated by employing the color-difference interpolation [5]. This interpolation is an interplane technique and has shown to be very effective in handling the missing colors by enforcing the color difference rule across different color channels. After this stage, the variance of each channel is evaluated. The variance of each channel is defined as follows.

$$Var(X) = \frac{1}{M \times N} \sum_{j=1}^{M} \sum_{i=1}^{N} \left(X^{n+1}(j,i) - X^{n}(j,i) - m_X \right)^2 \qquad (1)$$

where *X* represents *R*, *G*, or *B*, *M* and *N* separately represent the numbers of rows and columns of the image, and m_X is the mean of the difference signal (X^{n+1}-X^n), i.e.,

$$m_X = \frac{1}{M \times N} \sum_{j=1}^{M} \sum_{i=1}^{N} \left(X^{n+1}(j,i) - X^{n}(j,i) \right). \qquad (2)$$

For the *G* channel, G^0 stands for the outputs of the weighted-edge interpolation, and G^1 for the outputs of the color-difference interpolation, respectively. As to the *R* and *B* channels, R^1 and B^1 are the outputs of the color-difference interpolation, but both of R^0 and B^0 are set to be zero. In the iteration stage, the variance of each channel is compared with a universal threshold *TH*. If the variance of a channel is larger than *TH*, then update this signal by using the color-difference interpolation. After this interpolation, a new variance of the channel is calculated. Finally, if all of the variances are no more than *TH*, then terminate the interpolation, or repeat the steps in iteration stage until the stopping condition is met or the maximum iteration number is reached.



B	G	B_1	G	B
G	R_2	G_3	R_4	G
B_5	G_6	B_7	G_8	B_9
G	R_{10}	G_{11}	R_{12}	G
B	G	B_{13}	G	B

(a)

R	G	R_1	G	R
G	B_2	G_3	B_4	G
R_5	G_6	R_7	G_8	R_9
G	B_{10}	G_{11}	B_{12}	G
R	G	R_{13}	G	R

(b)

Fig. 3. Reference Bayer patterns. (a) the central pixel is B (b) the central pixel is R

III. Color interpolation algorithms

In this section, we present the proposed weighted-edge interpolation and summarize the main ideas of the color difference interpolation in [5] for completeness.

A. Weighted-edge interpolation to G channel

The proposed weighted-edge interpolation is a modified edge-sensitive interpolation [3]. To estimate the green value at blue pixel location (using Fig. 3(a)), the first step is to calculate horizontal and vertical gradients in this location. The definitions of these gradients are as follows.

$$\Delta H_B = |G_6 - G_8| + |2B_7 - B_5 - B_9|, \qquad (3)$$

$$\Delta V_B = |G_3 - G_{11}| + |2B_7 - B_1 - B_{13}|. \qquad (4)$$

In the expressions given above, the first term is the first-order difference of the neighboring green values, and the second term represents the second-order derivative of the neighboring blue values. The procedure of filling the missing green value at blue pixel location is outlined as Fig. 4.

```
If $\Delta V_B > \Delta H_B$ then
    $\hat{G}_7 = w_1[(G_6 + G_8)/2 + (2B_7 - B_5 - B_9)/4]$
        $+ w_2[(G_3 + G_{11})/2 + (2B_7 - B_1 - B_{13})/4];$
Else if $\Delta H_B > \Delta V_B$ then
    $\hat{G}_7 = w_1[(G_3 + G_{11})/2 + (2B_7 - B_1 - B_{13})/4]$
        $+ w_2[(G_6 + G_8)/2 + (2B_7 - B_5 - B_9)/4];$
Else
    $\hat{G}_7 = (G_3 + G_6 + G_8 + G_{11})/4$
        $+ (2B_7 - B_1 - B_{13} + 2B_7 - B_5 - B_9)/8;$
End
```

Fig. 4. Procedure of interpolating the missing green pixel at blue pixel location.

A similar strategy is applied to estimate the missing green value at red pixel location (using Fig. 3(b)). To obtain the corresponding procedure, the only one thing to do is to replace the notation in the foregoing procedure from *B* to *R*.

In Fig. 4, w_1 and w_2 are weighting factors, separately representing the weight of smooth transition term and the weight of sharp transition term. Each term contains two parts. One is the linear average of the neighboring green values, and the other is a correction value based on a second-order derivative of neighboring blue values. Intuitively, we expect a high value of w_1 and a low value of w_2. To reduce the designing complexity, we enforce the constraint $w_1+w_2=1$ to them. Moreover, their values are not optimized to get the best interpolation to *G* channel but to obtain the smallest summation of final MSE values in the proposed iterative demosaicing algorithm. Specifically, the best values of w_1 and w_2, denoted $(w_1, w_2)^*$, are chosen according to

$$(w_1, w_2)^* = \arg \min_{w_1 + w_2 = 1} \{MSE(R) + MSE(G) + MSE(B)\}, \qquad (5)$$

where *MSE*(*X*) represents the MSE value of signal *X* computed from the output pixels of *X* and the known pixels of *X*. To avoid introducing complexity into the demosaicing algorithm, the optimal choices for these weighting factors are estimated prior to demosaicing with average values. We interpolated a large number of images, and calculated the summation of final MSE values for w_1 between 0.99-0.7 (in steps of 0.01). The *TH* value in Fig. 2 is set to be 1 because it is observed that the MSE of each channel generally begins to increase as the variance of the channel is smaller than 1. The best values $(w_1, w_2)^*$ for the test images are recorded and averaged. The averages of the best values are found to be $w_1^*=0.87$ and $w_2^*=0.13$. For fixed-point processors, the suggested values are $w_1^*=7/8$ and $w_2^*=1/8$. This modification does not affect the MSE values much. It is worth mentioning that $w_1=1$ and $w_2=0$ is exactly equivalent to the edge-sensitive interpolation [12][14] used to handle the *G* pixels in initialization.

B. Color-difference interpolation to R and B channels

The color-difference interpolation is mainly dependent on the following two difference signals:

$$D_R = G - R \text{ and } D_B = G - B. \qquad (6)$$

These signals serve as a bridge to pass the refined *G* channel to *R* and *B* channels, and vice versa. The interpolation for *R* and *B* channels is equivalent to perform a bilinear interpolation in D_R and D_B domains. As shown in Fig. 3 (a), the updated values of $\hat{R}_3$ and $\hat{R}_7$ are obtained by

$$\hat{R}_3 = G_3 - \frac{1}{2}[(\hat{G}_2 - R_2) + (\hat{G}_4 - R_4)] \qquad (7)$$

and

$$\hat{R}_7 = \hat{G}_7 - \frac{1}{4}[(\hat{G}_2 - R_2) + (\hat{G}_4 - R_4) + (\hat{G}_{10} - R_{10}) + (\hat{G}_{12} - R_{12})] . \tag{8}$$

A similar operation is applied to update the *B* channel (using Fig. 3(b)).

C. Color-difference interpolation to G channel

Referring to Fig. 3 (a), the updated green value at blue pixel location is given by

$$\hat{G}_7 = B_7 + \frac{1}{4}[(G_3 - \hat{B}_3) + (G_6 - \hat{B}_6) + (G_8 - \hat{B}_8) + (G_{11} - \hat{B}_{11})] . \tag{9}$$

To update the green value at red pixel location (see Fig. 3 (b)), we use

$$\hat{G}_7 = R_7 + \frac{1}{4}[(G_3 - \hat{R}_3) + (G_6 - \hat{R}_6) + (G_8 - \hat{R}_8) + (G_{11} - \hat{R}_{11})] . \tag{10}$$

IV. Simulation Results

The proposed algorithm is compared to three *state-of-the-art* demosaicing algorithms: [6], [12] and [14]. The technique [6] is noniterative but performs very well in S-CIELab metric [16][17] due to its post processing to suppress demosaicing artifacts. The work [12] using convex-set projection theory is another class of iterative demosaicing technique. The scheme [14] has shown that it not only requires less computational load than previous iterative ones but also outperforms them in subjective and objective image qualities. Our algorithm and [14] share the same pixel updating interpolation [5] but apply different initial conditions and stopping criteria.

The tested images (shown in Fig. 5) are Kodak Photo images of 512×768 pixels, which are the same as those used in [14]. We downloaded them from the author's website [15]. The full color images are sampled with Bayer CFA pattern as shown in Fig. 1 to get mosaic images. The mosaic images are then interpolated back to full color images. Experimental results are reported by using MSE metric and S-CIELab metric [17] because S-CIELab metric is more appropriate than MSE metric in the measurement of human perceptual error. In addition, we also include subjective visual quality comparison to support the effectiveness of our algorithm. The MATLAB source code of our algorithm and the reconstructed images can be downloaded and viewed online from http://web.ntnu.edu.tw/~scy/heid_demo.html.

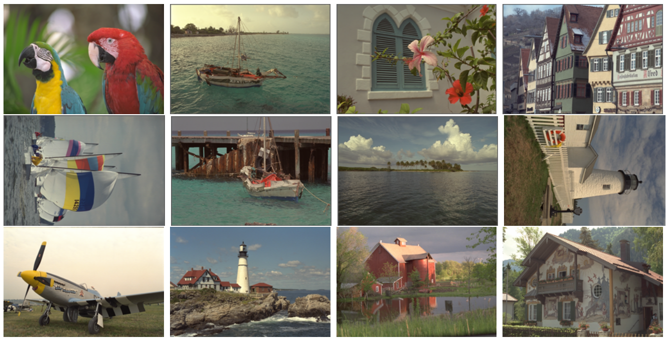


Fig. 5. Images used in the experiments. (images are numbered from 1 to 12 in the order of left-to-right and top-to-bottom).

TABLE I
Mean Squared Error Comparison of Different Demosaicing Schemes

Image.	Plane	Scheme [6]	Scheme [12]	Scheme [14]	Proposed
1	R	3.26	**2.74**	4.41	4.23
	G	1.51	2.11	2.15	**1.47**
	B	**3.16**	4.93	3.71	3.42
2	R	9.78	8.03	7.80	**6.20**
	G	6.46	3.55	3.22	**2.96**
	B	11.15	13.31	9.47	**7.95**
3	R	**3.11**	3.49	4.21	3.81
	G	**1.77**	2.31	2.69	2.45
	B	**4.06**	6.63	5.96	6.77
4	R	20.49	18.34	17.18	**16.78**
	G	12.18	7.55	6.86	**6.08**
	B	19.81	23.10	14.41	**13.87**
5	R	**3.31**	3.35	3.93	3.38
	G	2.07	1.89	2.10	**1.58**
	B	4.03	5.48	4.68	**3.71**
6	R	8.31	**6.89**	8.77	7.92
	G	5.15	3.87	3.49	**3.06**
	B	7.20	7.67	6.97	**5.65**
7	R	4.96	4.59	4.32	**3.27**
	G	3.21	1.77	1.73	**1.56**
	B	5.47	6.29	4.93	**3.96**
8	R	6.80	6.72	6.08	**5.17**
	G	4.53	2.77	2.65	**2.41**
	B	7.99	7.77	7.18	**6.15**
9	R	6.88	10.63	5.56	**4.86**
	G	5.12	**2.26**	2.95	2.62
	B	34.75	7.76	7.94	**7.34**
10	R	8.99	7.48	7.69	**6.03**
	G	6.49	3.37	3.21	**2.92**
	B	9.96	9.40	9.04	**7.36**
11	R	7.75	**7.10**	9.27	8.78
	G	4.72	4.76	5.07	**4.39**
	B	**8.80**	11.00	10.27	9.85
12	R	20.71	16.35	14.47	**12.94**
	G	13.99	8.30	7.41	**6.87**
	B	25.69	24.42	19.33	**18.52**

A. Performance comparison on MSE and S-CIELab metrics

Table I and Table II tabulate the performance comparisons in MSE metric and S-CIELab metric, respectively. The data shown in Table II are the average S-CIELab error magnitudes of output images. We use bold-type font to highlight the smallest value across each row. The S-CIELab results of [12] are not listed because it has shown that their values are larger than those of [6] in all cases. The tabulated results of [14] are obtained by performing the released MATLAB source code

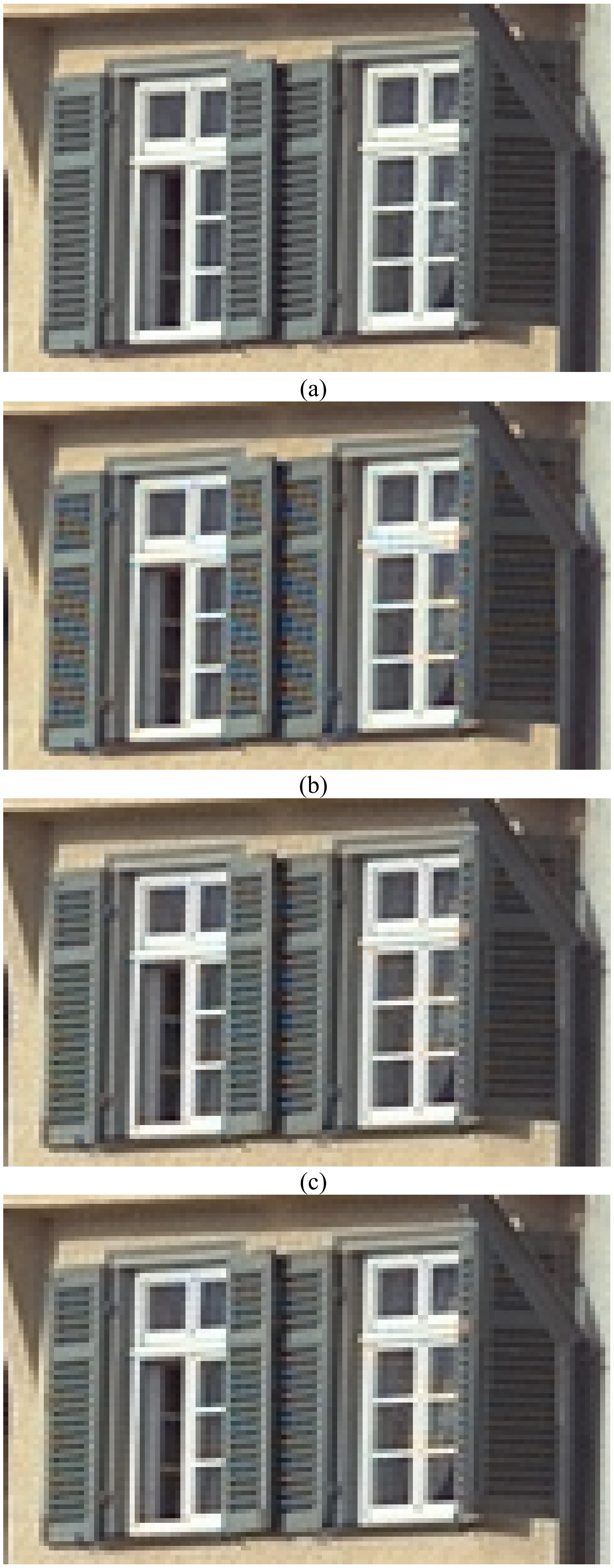
[15] to our computer, different but better, less than those reported in the paper [14]. It is worth mentioning that our scheme has the best MSE performance compared to most cases and achieves a comparable S-CIELab performance to [6]. Let us define the MSE reduction rate of scheme *x* to scheme *y* as (MSE_x-MSE_y) divided by MSE_y. Herein, MSE_x and MSE_y represent the MSE value of scheme *x* and the MSE value of scheme *y*, respectively. Then the MSE reduction rate of scheme [14] to ours can be obtained, near 14.5% on average. One reason for the high performance of our algorithm comes from that we introduce a weighted-edge interpolation, making the edge pixels interpolated very well. Moreover, the stopping condition is well designed so that the proposed algorithm can avoid incurring undesirable zipper artifacts.

TABLE II
AVERAGE S-CIELAB ERROR MAGNITUDE COMPARISON OF DIFFERENT DEMOSAICING SCHEMES

Image.	Scheme [6]	Scheme [14]	Proposed
1	**0.4674**	0.5403	0.5019
2	0.8799	0.8836	**0.8190**
3	**0.5091**	0.6776	0.6788
4	**1.2938**	1.4175	1.3314
5	**0.5333**	0.6046	0.5408
6	0.7253	0.7695	**0.7208**
7	0.6421	0.6380	**0.5944**
8	0.7414	0.7663	**0.7175**
9	0.6228	0.6361	**0.6034**
10	0.8549	0.8701	**0.8016**
11	**0.8525**	0.9586	0.9555
12	**0.9947**	1.1178	1.0494

B. Performance comparison on visual quality

In order to compare the visual quality more precisely, we magnify the reconstructed images of different demosaicing algorithms to show details. Figs. 6 and 7 separately exhibit a portion of image 4 and image 12. These two images abound with large amount of aliasing, which is difficult to demosaicing. From these figures, it is observed that our method results in better visual quality in comparison to all the others. The color misregistration artifacts are successfully suppressed, especially for image 4 (see Fig. 6(d)). It is remarkable that our algorithm does not use any nonlinear, filtering, and post-processing operations to handle these artifacts. In [14], these artifacts are suppressed by a two-class stopping criterion, necessitating additional computation for spatial classification but causing the number of iterations to increase. To discriminate the difference of these figures, readers are suggested to visit the above-mentioned website and check them out on a monitor.



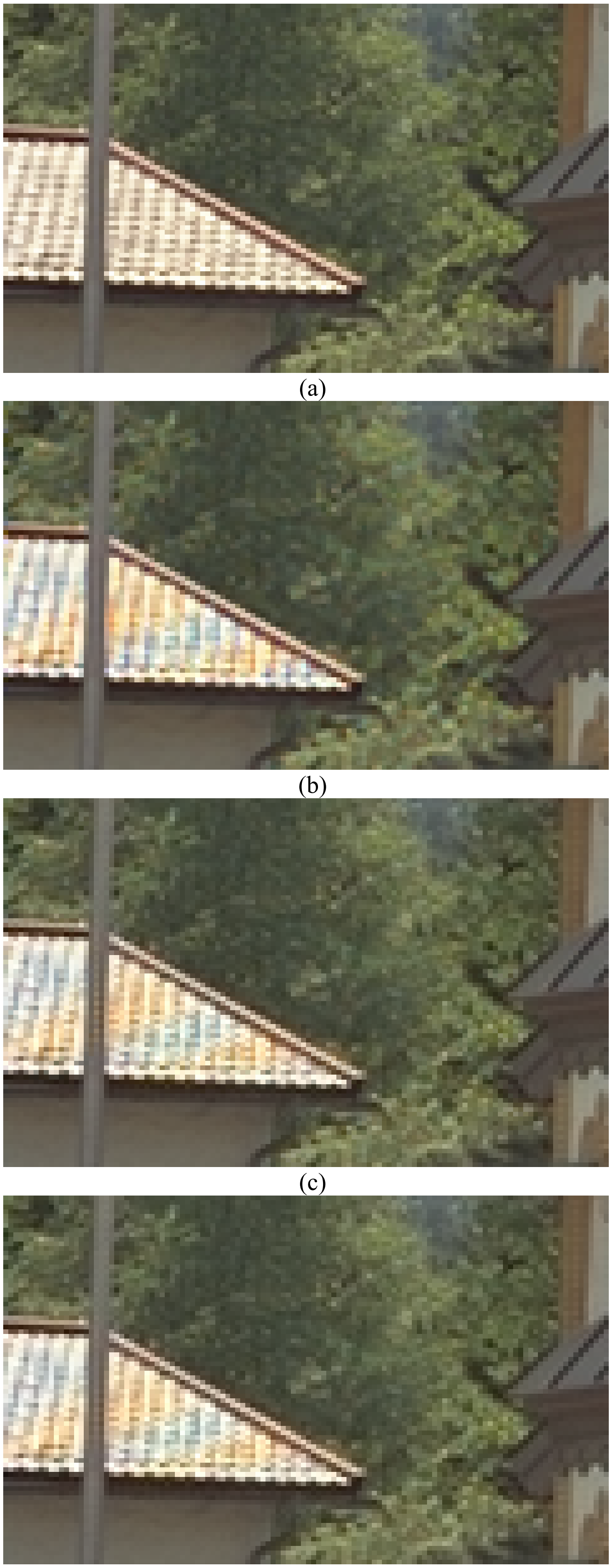
(a)

(b)

(c)

(d)

Fig. 6. Comparison of detailed images of image 4. (a) Original image. (b) Reconstructed by scheme [6]. (c) Reconstructed by scheme [14]. (d) Reconstructed by the proposed scheme.



(a)

(b)

(c)

(d)

Fig. 7. Comparison of detailed images of image 12. (a) Original image. (b) Reconstructed by scheme [6]. (c) Reconstructed by scheme [14]. (d) Reconstructed by the proposed scheme.

C. Complexity comparison

The complexity comparison among the iterative schemes [11], [12] and [14] has been addressed in [14], which shows that the technique in [14] needs less computational cost than the others because it is required neither to perform filtering (wavelet filtering [12]) nor to manipulate nonlinear operations (calculation of color ratio and edge indicator function [11]) at each iteration. The recent iterative method in [13] is also computationally expensive due to the calculation of homogeneity. According to the concerns above, we mainly compare, herein, our computational cost with that in [14].

Table III lists the times of each function executed in [14] and in ours. We sum up the amount of the executions of the same function against the same color channel. From this table, it is obvious that the proposed algorithm requires computational cost only near 22% of that in [14]. This result comes with no surprising, just because the proposed algorithm interpolates each channel only about three times, while [14] uses a two-class stopping criterion to suppress color misregistration artifacts and to reduce MSE values, which requires performing the color-difference interpolation to each channel many times. In most cases, the interpolation of high-aliasing region is performed until the maximum number of iterations (default value is 10 in the released code) is reached. Besides the dominated interpolating times, the proposed algorithm does not need the calculation of spatial classification or the bilinear interpolation to handle *R* and *B* channels for initialization. The improvements explain why the proposed algorithm can achieve a lesser computational cost.

It should be noted that the memory requirement of the proposed algorithm is almost the same as the size of a full-color image because the color updating interpolation can be implemented by in-place operations. Compared with [14], the proposed algorithm is also of a less memory requirement because no map is needed to mark low-aliasing and high-aliasing regions.

V. CONCLUSION

A high-performance and low-complexity iterative algorithm is established in this paper. The high-performance arises from the introduction of a weighted edge interpolation and the well-designed stopping strategy. With them, the proposed algorithm has a good initial condition and can terminate iteration early. The low-complexity comes from that the proposed algorithm is mainly built on the low-complexity color difference interpolation without using any nonlinear, filtering, or post-processing operations. Compared with the other iterative demosaicing techniques reported in the literature, the proposed algorithm not only achieves a large reduction in computational cost but also possesses very good image quality both in subjective and objective measurements, contributing to a high-speed as well as high–quality image.

TABLE III
COMPARISON OF DIFFERENT ALGORITHMS ON THE TIMES OF EACH FUNCTION EXECUTED. (ES, WE, AND SC STAND FOR EDGE SENSITIVE, WEIGHTED EDGE, AND SPATIAL CLASSIFICATION, RESPECTIVELY)

Img.	Scheme	Bilinear		ES	WE	SC	Color difference		
		R	B	G	G		R	B	G
1	[14]	1	1	1	0	1	12	12	12
	Proposed	0	0	0	1	0	2	2	1
2	[14]	1	1	1	0	1	12	12	12
	Proposed	0	0	0	1	0	3	3	2
3	[14]	1	1	1	0	1	12	12	12
	Proposed	0	0	0	1	0	2	3	2
4	[14]	1	1	1	0	1	13	13	13
	Proposed	0	0	0	1	0	4	4	3
5	[14]	1	1	1	0	1	12	12	12
	Proposed	0	0	0	1	0	2	2	1
6	[14]	1	1	1	0	1	12	12	12
	Proposed	0	0	0	1	0	3	3	2
7	[14]	1	1	1	0	1	11	11	11
	Proposed	0	0	0	1	0	3	3	2
8	[14]	1	1	1	0	1	11	11	11
	Proposed	0	0	0	1	0	3	3	2
9	[14]	1	1	1	0	1	12	12	12
	Proposed	0	0	0	1	0	3	3	2
10	[14]	1	1	1	0	1	12	12	12
	Proposed	0	0	0	1	0	3	3	2
11	[14]	1	1	1	0	1	12	12	12
	Proposed	0	0	0	1	0	3	3	2
12	[14]	1	1	1	0	1	13	13	13
	Proposed	0	0	0	1	0	4	4	3

ACKNOWLEDGMENT

The author would like to thank Dr. Li at West Virginia University for providing his demosaicing code as the benchmark.

REFERENCES

[1] K. A. Parulski, “Color filters and processing alternatives for one-chip cameras,” *IEEE Trans. Electron Devices*. vol. ED-32, no. 8. pp. 1381, Aug. 1985.

[2] B. E. Bayer, “Color imaging array,” U.S. Patent 3 971 065, Jul. 1976.

[3] J. F. Hamilton Jr. and J. E. Adams, “Adaptive color plane interpolation in single color electronic camera,” U. S. Patent 5 629 734, May 1997.

[4] A. Jain, *Fundamentals of Digital Image Processing*, Upper Saddle River, NJ: Prentice-Hall, 1989.

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

[6] W. Lu and Y.-P. Tan, “Color filter array demosaicing: New method and performance measures,” *IEEE Trans. Image Process*., vol. 12, no. 10, pp. 1194-1210, Oct. 2003.

[7] X. Li and M. Orchard, “New edge directed interpolation,” *IEEE Trans. Image Process*., vol. 10, no. 10, pp. 1521-1527, Oct. 2001.

[8] P.-S. Tsai, T. Acharya, A. K. Ray, “Adaptive fuzzy color interpolation,” *Journal of Electronic Imaging*, vol. 11, pp. 1-24, July 2002.

[9] D. D. Muresan and T. W. Parks, “Demosaicing using optimal reconvery,” *IEEE Trans. Image Process*., vol. 14, no. 2, pp. 267-278, Feb. 2005.

[10] D. Alleysson, S. Süsstrunk, and J. Hérault, “Linear demosaicing inspired by the human visual system,” *IEEE Trans. Image Process*., vol. 14, no. 4, pp. 439-449, April 2005.

[11] R. Kimmel, “Demosaicing: Image reconstruction from CCD samples,” *IEEE Trans. Image Process*., vol. 8, no.9, pp. 1221-1228, Sep. 1999.

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

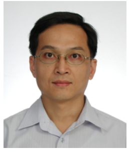
[13] K. Hirakawa and T. W. Parks, “Adaptive homogeneity-directed demosaicing algorithm,” *IEEE Trans. Image Process*., vol. 14, no. 3, pp. 360-369, March 2005.

[14] X. Li, “Demosaicing by successive approximation,” *IEEE Trans. Image Process*., vol. 14, no. 3, pp. 370-379, March 2005.

[15] Kodak test images and the demosaicing code of successive approximation available at http://www.csee.wvu.edu/~xinl/demo/demosaic.html.

[16] X. Zhang, D. A. Silverstein, J. E. Farrell, and B. A. Wandell, “Color image quality metric S-CIELAB and its application on halftone texture visibility,“ in *Proc. IEEE COMPCON*, Feb. 1997, pp. 44-48.

[17] S-CIELab Metric (2003). [Online]. Available at http://white.stanford.edu/~brian/scielab/scielab.html



Chung-Yen Su was born in Tainan, Taiwan, R.O.C., in 1967. He received the B. S. and M. S. degrees in industrial education from the National Taiwan Normal University (NTNU), Taipei, Taiwan, in 1991 and 1994, respectively, and the Ph. D. degree in electrical and control engineering from the National Chiao-Tung University, Hsinchu, Taiwan, in 1999.

He joined the Department of Industrial Education of NTNU as a Teacher Assistant in 1992 and is currently an Associate Professor of the Department and the Institute of Applied Electronic Technology of NTNU. He is currently a visiting scholar in Computer Science Department of University of Nevada at Reno, USA. His research interests include image/video compression, wavelets, signal processing, digital watermarking, and computer vision.