J. Vis. Commun. Image R. 23 (2012) 729–741

Contents lists available at SciVerse ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Quality-efficient demosaicing for digital time delay and integration images using edge-sensing scheme in color difference domain ☆

Wei-Jen Yang [a], Kuo-Liang Chung [a,*], Hong-Yuan Mark Liao [b]

[a] Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei 10672, Taiwan, ROC
[b] Institute of Information Science, Academia Sinica, No. 128, Section 2, Academia Road, Taipei 11529, Taiwan, ROC

ARTICLE INFO

Article history:
Received 24 October 2010
Accepted 4 April 2012
Available online 16 April 2012

Keywords:
Demosaicing algorithm
Digital time delay and integration (DTDI)
Edge information
Industrial print inspection
Line-scan camera
Mosaic images
Color filter array (CFA)
Sobel edge detector

ABSTRACT

In this paper, we present a novel edge sensing-based demosaicing algorithm for digital time delay and integration (DTDI) mosaic images, which are captured by DTDI line-scan cameras and suitable for industrial print inspection. We propose to use Sobel- and interpolation-based masks to extract more accurate gradient information in the color difference domain. The extracted gradient information is utilized to assist the design of the proposed demosaicing algorithm. By experimenting on more than one thousand and three hundred test DTDI mosaic images, the results demonstrate the efficiency of the proposed demosaicing algorithm in terms of demosaiced image quality.

1. Introduction

Recently, digital cameras are getting more popular in consumer electronics market. Fig. 1 illustrates the structure of a three charge-coupled device or complementary metal-oxide–semiconductor (CCD/CMOS) sensors digital camera. From Fig. 1, it is observed that after passing through the camera lens and the optical filter, a light path would be divided into three color components, which corresponds to the tristimulus values of a scene. However, this structure needs three CCD/CMOS sensors to produce a color image and this is costly. To cut down the cost, most manufacturers use a single CCD/CMOS sensor with the Bayer color filter array (CFA) structure [1,5–7,16,19] to capture the color information. The structure of a single CCD/CMOS sensor digital camera is illustrated in Fig. 2. Based on the Bayer CFA structure shown in Fig. 3, each pixel in the structure has only one color component. Because the green (G) component is the most important factor to determine the luminance of a color image, half of the pixels in the Bayer CFA structure are assigned to G component. The red (R) and blue (B) components share the remaining parts evenly.

In the field of industrial inspection systems, line-scan cameras are mostly used for the examination of fine details [2]. Because of the short exposure time and high-speed transport restrictions, in many cases, it is very difficult to increase the intensity of the illumination. Therefore, some special image formations or techniques, such as high dynamic range imaging (HDR) [8] or time delay and integration (TDI) [21], are demanded to maintain image quality while the quantity of available light decreases. Nowadays, digital time delay and integration (DTDI) line-scan cameras [2,9,12] based on CMOS sensors with the Bayer CFA structure and field programmable gate arrays (FPGAs) have been developed for the industrial applications because they have the advantages of low power consumption, low production cost, and high-speed. For DTDI line-scan cameras, instead of scanning moving objects row by row, a large number of rows, which are sequentially accumulated in their responses, are exposed in parallel. Further, the object will be moved by one row after each exposure, so each object pixel is not captured only once. In fact, the number of times which each object pixel is captured is as many as the DTDI stages. Finally, the output of each pixel is collected by the responses of the partial exposures.

Fig. 4 illustrates an example of the principle of a DTDI line-scan camera. From Fig. 4, it is observed that the design of a DTDI line-scan camera is based on a Bayer CFA and some delay stages. These delay stages are denoted as z^{-1} and are used to integrate two consecutive rows captured at different time instants. For each exposure, either BG-row or GR-row would be captured and delivered. This indicates that each object pixel will be captured by either the G and R components or the G and B components alternately.

☆ Supported by National Science Council of ROC under the contract NSC99-2221-E-011-078-MY3.
* Corresponding author. Fax: +886 2 27301081.
E-mail address: k.l.chung@mail.ntust.edu.tw (K.-L. Chung).

1047-3203/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jvcir.2012.04.002

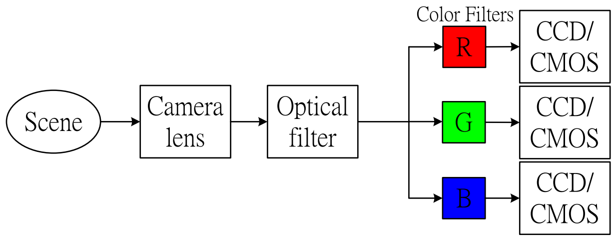

Fig. 1. The structure of a three CCD/CMOS sensor digital camera.

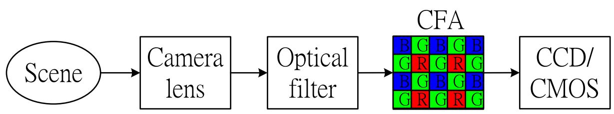

Fig. 2. The structure of a single CCD/CMOS sensor digital camera.

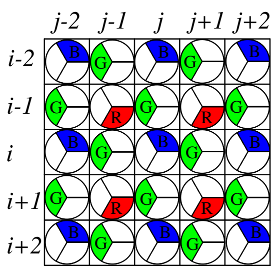

Fig. 3. Bayer CFA structure.

The capturing and delivering process would continue until the number of required DTDI stages is reached, and then the output of each pixel could be computed. For this example with four DTDI stages, the resultant output of each pixel could be computed by the following rule:

$$B'_{2m} = B_{0,2m} + B_{2,2m}z^{-2},$$
$$G'_{2m} = G_{1,2m}z^{-1} + G_{3,2m}z^{-3},$$
$$G'_{2m+1} = G_{0,2m+1} + G_{2,2m+1}z^{-2},$$
$$R'_{2m+1} = R_{1,2m+1}z^{-1} + R_{3,2m+1}z^{-3}.$$

Images captured by a DTDI line-scan camera are called DTDI mosaic images. Fig. 5 illustrates the structure of a DTDI mosaic image where each pixel has two color components, i.e., G and R, or G and B.

Since a full-color image is preferable to the human visual system, the missing color component of each pixel in a DTDI mosaic image should be recovered as much as possible and such a recovery is called a demosaicing process. Among the existing demosaicing algorithms, bilinear interpolation [9,20] is the simplest demosaicing algorithm in which the unknown color component of each pixel is obtained by averaging its two neighboring pixels.

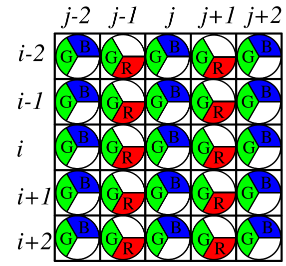

Fig. 5. DTDI structure.

Recently, Heiss-Czedik et al. [12] modified Laroche and Prescott's algorithm [14] and Hamilton and Adams' algorithm [11] to deal with DTDI mosaic images. In [12], Heiss-Czedik et al. also presented DTDI demosaicing algorithms based on the weighted absolute interpolation and least squares approximation, respectively. After examining the previously developed demosaicing algorithms, we find that the quality of a demosaiced image is strongly dependent on the extracted gradient/edge information from the input DTDI mosaic images. In addition, for the previously proposed demosaicing algorithms, such as [4,11,12], the gradient information is extracted in the spacial domain and the interpolation process is based on the color difference domain. The gradient information extracted in the spacial domain might not be able to take on the gradient information in the color difference domain. It would result in the degradation of the estimating accuracy in the demosaicing process. Although for the Bayer CFA with one component in each pixel, Chung and Chan [3] proposed an efficient demosaicing algorithm exploiting the integrated gradient information, which is formed by the combination of the gradient information extracted in the spacial and color difference domains, their algorithm is specifically designed for the Bayer CFA and the design of the related masks widely depends on the Bayer CFA structure, implying that Chung and Chan's algorithm cannot be directly applied to DTDI mosaic images in which each pixel consists of two color components, namely the G and R components or the G and B components. Further, it is very difficult to modify their masks to deal with DTDI structure since these masks are designed according to the arrangement of the color components in the Bayer CFA structure (see Fig. 3), which is quite different from that in the DTDI mosaic image (see Fig. 5). Thus, the main motivations of this work is twofold. In the first place, develop a new approach to extract more accurate gradient information in the color difference domain directly for DTDI mosaic images. Second, develop a new edge sensing-based demosaicing algorithm, which exploits the extracted more accurate gradient information, for DTDI mosaic images.

In our proposed algorithm, instead of using SL-based masks [4] obtained by embedding the luminance estimation mask into Sobel masks to extract gradient information in the spacial domain, we use the Sobel- and interpolation-based (SI-based) masks, which is the combination of Sobel masks and bilinear interpolation, to

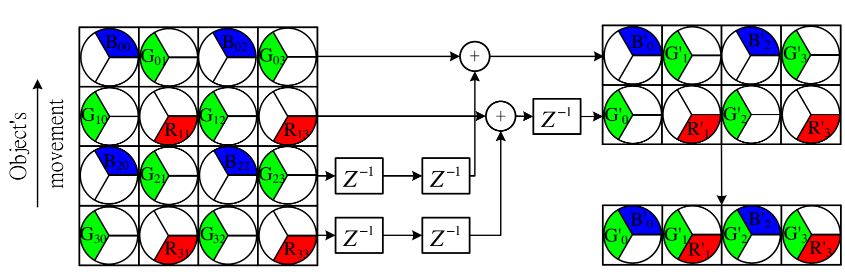

Fig. 4. An example of the principle of a DTDI line-scan camera.

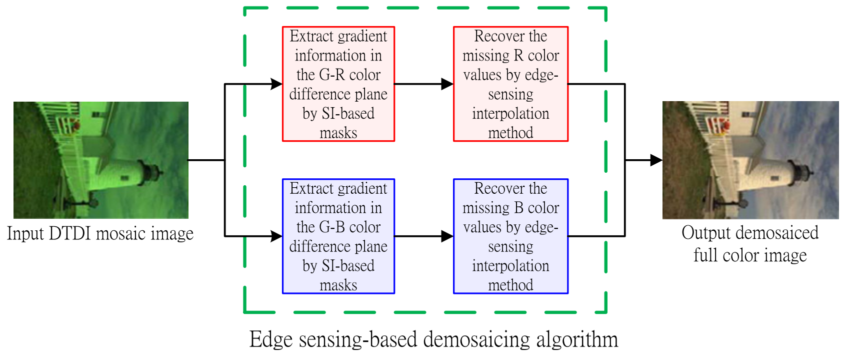

Fig. 6. The flowchart of the proposed demosaicing algorithm.

extract more accurate gradient information in the color difference domain. Based on the extracted gradient information, the proposed new edge sensing-based demosaicing algorithm is developed. Because both of the gradient information extraction and the edge sensing-based interpolation processes are based on the same color difference domain, the proposed demosaicing algorithm would produce better quality of demosaiced images. Note that for DTDI mosaic images, there is no demosaicing algorithm, whose gradient information extraction process and interpolation process are both based on the color difference domain, to be proposed previously. Fig. 6 illustrates the flowchart of the proposed demosaicing algorithm. We have tested our algorithm on more than one thousand and three hundred test DTDI mosaic images. The results demonstrate that the proposed demosaicing algorithm has better demosaiced image quality than five existing demosaicing algorithms in [12], modified Pei and Tam's algorithm [18], and modified Chung et al.'s algorithm [4].

The major novel contributions of this work are stated again as follows. First, we propose new SI-based masks to extract more accurate gradient information in the color difference domain directly. Second, we develop a new edge sensing-based demosaicing algorithm, which exploits the extracted gradient information in the color difference domain, for DTDI mosaic images. To the best of our knowledge, this is the first time such a demosaicing algorithm, whose gradient information extraction process and interpolation process are both based on the color difference domain, has been developed specifically for DTDI mosaic images. Finally, more than one thousand and three hundred test DTDI mosaic images are used to evaluate the demosaiced image quality performance and the results indicate that the proposed algorithm is superior to seven existing state-of-the-art algorithms.

The remainder of this paper is organized as follows. In Section 2, the proposed edge sensing-based demosaicing algorithm for DTDI mosaic images is presented. In Section 3, the experimental results are shown to demonstrate the advantageous features of the proposed demosaicing algorithm. Finally, concluding remarks are drawn in Section 4.

2. The proposed edge sensing-based demosaicing algorithm for DTDI mosaic images

In Section 2.1, we first present how to extract more accurate gradient information in the G–R and G–B color difference planes. Then, the edge sensing-based demosaicing algorithm used to recover the missing R and B color components of a DTDI mosaic image is presented in Section 2.2. One thing to be noted is that in previous demosaicing algorithm, e.g., [4,11,12], the gradient information is extracted in the spacial domain and the interpolation process is based on color difference domain; in the proposed demosaicing algorithm, both of the gradient information extraction process and the interpolation process are based on the color

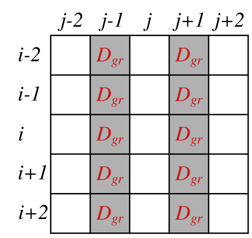

Fig. 7. The pattern of the DTDI G–R color difference plane.

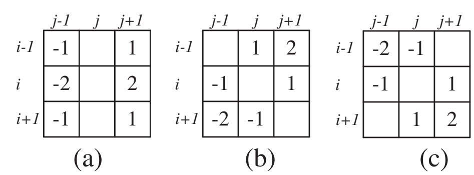

Fig. 8. Sobel edge detector. (a) The horizontal mask. (b) The $\frac{\pi}{4}$-diagonal mask. (c) The $\frac{-\pi}{4}$-diagonal mask.

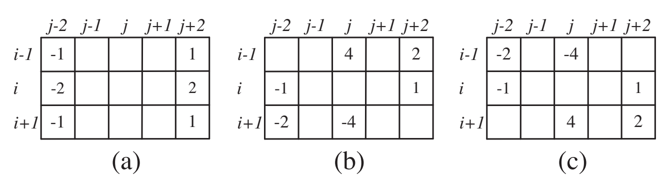

Fig. 9. For the pixels at position $(i,j) \in \Omega_{gr}$, the three SI-based masks. (a) The horizontal mask. (b) The $\frac{\pi}{4}$-diagonal mask. (c) The $\frac{-\pi}{4}$-diagonal mask.

difference domain. Because of the consistency between the gradient information extraction process and the interpolation process in the proposed demosaicing algorithm, it would result in producing better quality of demosaiced images. As shown in Fig. 5, the R, G, and B color values of the pixel located at the position (i,j) of a DTDI mosaic image, I_{Dmo}, are denoted as $I^r_{Dmo}(i,j), I^g_{Dmo}(i,j)$, and $I^b_{Dmo}(i,j)$, respectively.

2.1. Extracting more accurate gradient information in color difference domain

In this sub-section, we present the gradient information extraction process for the G–R and G–B color difference planes. Since the gradient information extraction process for the G–R color difference plane is the same as that for the G–B color difference plane, we only describe it for the G–R color difference plane. According to the structure of a DTDI mosaic image, as shown in Fig. 5, the DTDI G–R color difference plane, D_{gr}, can be obtained by

	j-2	j-1	j	j+1	j+2
i-1		-2		2	
i		-4		4	
i+1		-2		2	

(a)

	j-2	j-1	j	j+1	j+2
i-1		1		5	
i		-2		2	
i+1		-5		-1	

(b)

	j-2	j-1	j	j+1	j+2
i-1		-5		-1	
i		-2		2	
i+1		1		5	

(c)

Fig. 10. For the pixels at position $(i,j) \notin \Omega_{gr}$, the three SI-based masks. (a) The horizontal mask. (b) The $\frac{\pi}{4}$-diagonal mask. (c) The $\frac{-\pi}{4}$-diagonal mask.

$$D_{gr}(i_{gr},j_{gr}) = I^{g}_{Dmo}(i_{gr},j_{gr}) - I^{r}_{Dmo}(i_{gr},j_{gr}), \tag{1}$$

where $\forall(i_{gr},j_{gr}) \in \Omega_{gr} = \{(i \pm a, j \pm (2b+1))\}$. Fig. 7 illustrates the pattern of the DTDI G–R color difference plane. To obtain the fully populated G–R color difference plane, bilinear interpolation is used to estimate the missing pixels in D_{gr}. Thus, the fully populated G-R color difference plane, $\widetilde{D}_{gr}$, can be determined by

$$\widetilde{D}_{gr}(i,j) = \begin{cases} D_{gr}(i,j) & \text{if } (i,j) \in \Omega_{gr}, \\ \frac{1}{2} \sum\limits_{(x,y)\in\{(i,j\pm1)\}} D_{gr}(x,y) & \text{otherwise}. \end{cases} \tag{2}$$

Next, we use Sobel edge detector [10] to extract gradient information. Fig. 8(a)–(c) illustrate the 3×3 horizontal, $\frac{\pi}{4}$-diagonal, and $\frac{-\pi}{4}$-diagonal masks of Sobel edge detector, respectively. After running the horizontal, $\frac{\pi}{4}$-diagonal, and $\frac{-\pi}{4}$-diagonal masks on a 3×3 color difference subplane centered at position (i,j), the horizontal gradient response, $\Delta\widetilde{D}^{h}_{gr}(i,j)$, the $\frac{\pi}{4}$-diagonal gradient response, $\Delta\widetilde{D}^{\frac{\pi}{4}}_{gr}(i,j)$, and the $\frac{-\pi}{4}$-diagonal gradient response, $\Delta\widetilde{D}^{\frac{-\pi}{4}}_{gr}(i,j)$, can be calculated by

$$\begin{aligned}
\Delta\widetilde{D}^{h}_{gr}(i,j) &= \left\{ \begin{array}{l} \left[\begin{array}{l} \widetilde{D}_{gr}(i-1,j+1) + \widetilde{D}_{gr}(i+1,j+1) \\ -\widetilde{D}_{gr}(i-1,j-1) - \widetilde{D}_{gr}(i+1,j-1) \end{array} \right] \\ +2\left[\widetilde{D}_{gr}(i,j+1) - \widetilde{D}_{gr}(i,j-1) \right] \end{array} \right\}, \\
\Delta\widetilde{D}^{\frac{\pi}{4}}_{gr}(i,j) &= \left\{ \begin{array}{l} \left[\begin{array}{l} \widetilde{D}_{gr}(i-1,j) + \widetilde{D}_{gr}(i,j+1) \\ -\widetilde{D}_{gr}(i,j-1) - \widetilde{D}_{gr}(i+1,j) \end{array} \right] \\ +2\left[\widetilde{D}_{gr}(i-1,j+1) - \widetilde{D}_{gr}(i+1,j-1) \right] \end{array} \right\}, \\
\Delta\widetilde{D}^{\frac{-\pi}{4}}_{gr}(i,j) &= \left\{ \begin{array}{l} \left[\begin{array}{l} \widetilde{D}_{gr}(i,j+1) + \widetilde{D}_{gr}(i+1,j) \\ -\widetilde{D}_{gr}(i-1,j) - \widetilde{D}_{gr}(i,j-1) \end{array} \right] \\ +2\left[\widetilde{D}_{gr}(i+1,j+1) - \widetilde{D}_{gr}(i-1,j-1) \right] \end{array} \right\}.
\end{aligned} \tag{3}$$

To make Sobel edge detector workable on the DTDI G–R color difference plane directly, bilinear interpolation should be embedded into Sobel edge detector. Combining Eqs. (2) and (3), the SI-based masks

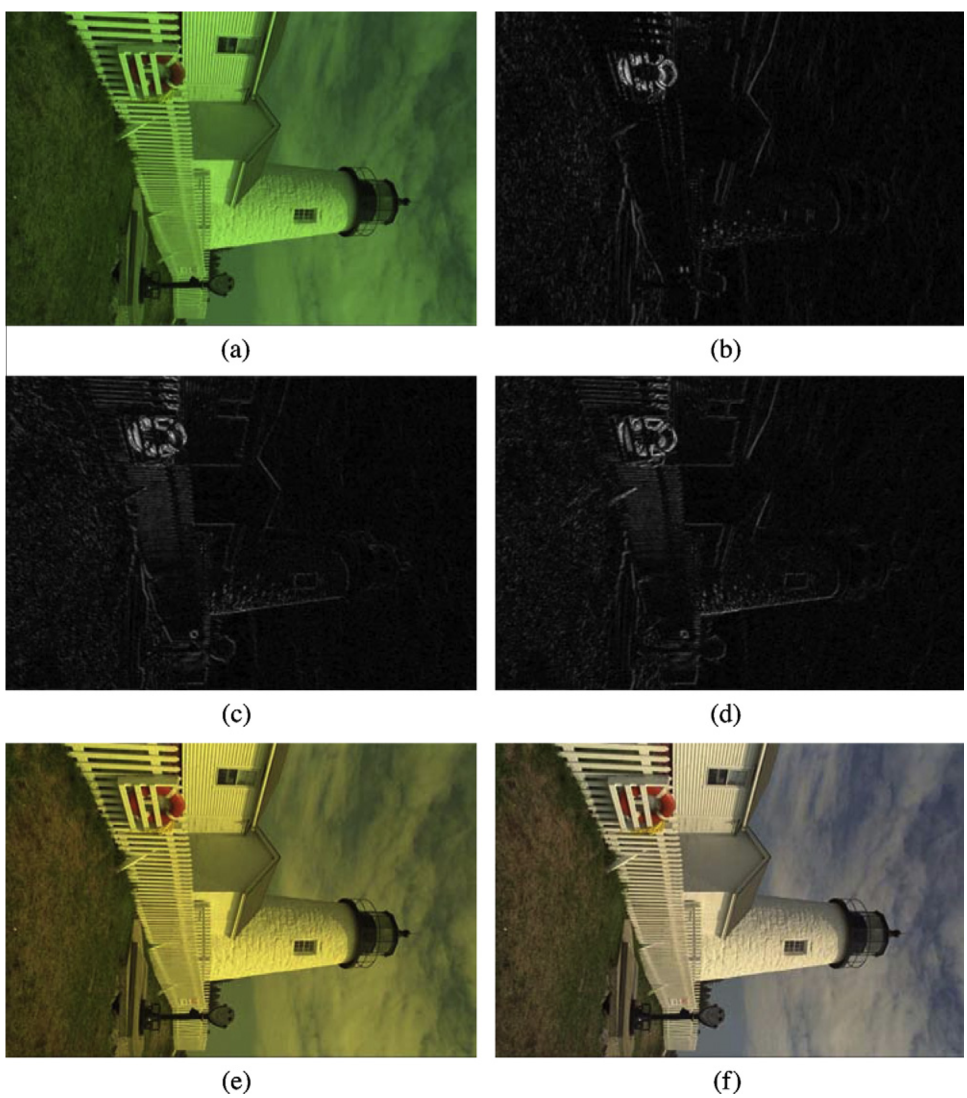

Fig. 11. Based on the DTDI Lighthouse image, the simulation result of each step in the proposed demosaicing algorithm. (a) The original DTDI Lighthouse image. (b) The horizontal gradient response. (c) The $\frac{\pi}{4}$-diagonal gradient response. (d) The $\frac{-\pi}{4}$-diagonal gradient response. (e) The resultant image after recovering the R color values. (f) The demosaiced full color image.

are followed (detailed derivations are shown in Appendix A). Furthermore, the coefficients of the SI-based masks are normalized into integers to avoid floating point computations. Considering two different cases, the SI-based masks for the pixels at position $(i,j) \in \Omega_{gr}$ and $(i,j) \notin \Omega_{gr}$ are shown in Figs. 9 and 10, respectively. After running the appropriate SI-based masks on the 3×5 DTDI color difference subplane centered at the position (i,j), the horizontal gradient response, $\Delta\widetilde{D}^{h}_{gr}(i,j)$, the $\frac{\pi}{4}$-diagonal gradient response, $\Delta\widetilde{D}^{\frac{\pi}{4}}_{gr}(i,j)$, and the $\frac{-\pi}{4}$-diagonal gradient response, $\Delta\widetilde{D}^{\frac{-\pi}{4}}_{gr}(i,j)$, can be obtained directly. For example, based on the input DTDI Lighthouse image shown in Fig. 11(a), Fig. 11(b)–(d) illustrate the horizontal gradient response, the $\frac{\pi}{4}$-diagonal gradient response, and the $\frac{-\pi}{4}$-diagonal gradient response of each pixel in the G–R color difference plane, respectively. From Fig. 11(b)–(d), it is observed that the gradient responses are locally constant in homogeneous regions. In the next sub-section, the extracted gradient information will be used to assist the design of the proposed edge sensing-based demosaicing algorithm.

2.2. Theproposed edge sensing-based demosaicing algorithm for DTDI mosaic images

In this section, the proposed edge sensing-based demosaicing algorithm that is used to recover the missing R and B color values of a DTDI mosaic image is presented. In what follows, let the R, G, and B color values of the pixel located at the position (i,j) of a demosaiced full color image, I_{Ddm}, be denoted as $I^{r}_{Ddm}(i,j), I^{g}_{Ddm}(i,j)$, and $I^{b}_{Ddm}(i,j)$, respectively. Since the proposed recovery method for R color values is the same as that for B color values, we only present the case of R color values.

For easy explanation, we use Fig. 5 to describe how the missing R color values are estimated. The subimage shown in Fig. 5 indicates that the missing R color value of the central pixel at position (i,j) can be estimated from its six neighboring pixels with movement $\Psi = \{(i+a, j+b) | a \in \{0, \pm 1\}, b \in \{\pm 1\}\}$. To estimate the R color value more accurately, we assign six appropriate weights in terms of

Fig. 12. The twenty-four testing images in Kodak PhotoCD [23].

Table 1
PSNR quality comparison in R color plane for twenty-four test images in Kodak PhotoCD.

Image	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
Image01	29.075	32.228	43.721	43.865	33.668	43.282	**45.701**	45.565
Image02	34.087	36.055	40.905	41.302	37.143	40.528	41.064	**42.309**
Image03	37.855	41.247	46.224	46.710	43.126	45.342	46.235	**47.717**
Image04	34.725	36.931	41.391	42.173	38.219	41.169	41.879	**42.894**
Image05	28.397	31.968	42.679	43.129	33.902	41.977	43.139	**44.187**
Image06	32.063	35.111	47.673	47.956	36.614	47.018	47.789	**48.516**
Image07	35.593	39.442	46.021	46.455	41.389	45.407	46.699	**47.665**
Image08	24.442	27.723	41.051	41.185	29.271	40.741	41.181	**42.001**
Image09	34.412	37.675	46.571	46.396	39.383	45.875	47.275	**47.531**
Image10	35.011	38.655	46.159	46.288	40.736	45.511	46.475	**46.845**
Image11	31.322	34.273	44.372	45.010	35.742	43.883	44.989	**45.910**
Image12	36.145	39.565	45.512	45.581	41.293	45.051	46.939	**47.059**
Image13	26.520	29.441	45.396	45.630	30.994	45.030	45.603	**46.064**
Image14	31.399	34.412	40.575	41.671	36.006	40.045	40.929	**42.458**
Image15	32.214	35.453	39.254	39.431	36.328	39.021	**40.120**	39.952
Image16	36.979	39.999	49.927	49.799	41.644	49.297	50.128	**50.446**
Image17	33.606	36.918	47.608	46.732	38.469	46.937	47.505	**47.732**
Image18	29.904	32.731	43.121	43.489	34.132	42.781	43.080	**44.020**
Image19	30.263	33.484	46.654	46.397	34.765	46.191	46.773	**47.291**
Image20	33.190	37.012	48.002	47.928	36.679	47.426	47.985	**48.460**
Image21	31.814	35.096	47.504	47.523	36.757	46.813	47.725	**48.239**
Image22	31.823	34.674	42.044	42.172	35.826	41.433	42.101	**42.516**
Image23	37.405	40.294	45.103	45.147	42.322	44.570	45.614	**46.641**
Image24	29.281	32.249	42.148	41.991	33.675	41.503	41.754	**42.164**
Average	32.397	35.527	44.567	44.748	37.003	44.035	44.945	**45.591**

gradient information along the interpolation directions to the six corresponding neighboring pixels. We first consider the neighboring pixel located at position $(i,j-1)$. When the pixel lies on a vertical edge, it indicates that the horizontal gradient magnitude, $\left|\Delta\widetilde{D}^{h}_{gr}(i,j-1)\right|$, would be large. Based on the color difference assumption [4,15,18], it reveals that this pixel makes less contribution to the estimation of the R color value of the central pixel; otherwise, it reveals that this pixel makes more contribution to the estimation of the R color value of the central pixel. Thus, we use the reciprocal of the gradient magnitude to determine the appropriate weight. Further, besides $\left|\Delta\widetilde{D}^{h}_{gr}(i,j-1)\right|$, another two horizontal gradient magnitudes of the pixels at positions (i,j) and $(i,j-2)$ are also considered to enhance the accuracy of the estimation. According to the above analysis on gradient information and direction effects, the weight of the pixel at position $(i,j-1)$ can be determined by

$$w_{gr}(h,i,j-1)=\frac{1}{1+\left[\left|\Delta\widetilde{D}^{h}_{gr}(i,j-2)\right|+2\left|\Delta\widetilde{D}^{h}_{gr}(i,j-1)\right|+\left|\Delta\widetilde{D}^{h}_{gr}(i,j)\right|\right]},$$

where 1 in the denominator is used to avoid division by zero. For the same reason, the weights of the five other neighbors of the central pixel can be respectively determined by

Table 2
PSNR quality comparison in B color plane for twenty-four test images in Kodak PhotoCD.

Image	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
Image01	28.409	31.245	47.615	47.775	32.299	48.055	47.562	**48.296**
Image02	34.793	37.155	49.939	49.597	38.280	50.855	49.909	**51.034**
Image03	37.691	40.324	47.215	**48.183**	41.619	46.903	45.717	47.667
Image04	34.827	37.915	50.072	49.834	39.416	50.612	50.301	**51.236**
Image05	28.078	31.187	44.341	**45.716**	32.591	44.048	43.821	45.042
Image06	31.614	34.493	45.528	45.644	35.698	45.279	45.795	**46.123**
Image07	35.411	39.113	48.604	**50.089**	41.251	49.122	47.866	49.764
Image08	24.171	27.271	43.369	**43.538**	28.592	43.179	42.975	43.443
Image09	34.130	36.797	48.786	49.015	38.046	49.012	47.611	**49.337**
Image10	34.219	37.327	46.178	46.258	38.859	46.067	45.949	**46.640**
Image11	31.312	34.310	49.510	49.894	35.600	50.157	50.029	**50.908**
Image12	35.011	38.653	47.985	48.138	40.030	48.378	47.727	**48.740**
Image13	26.140	28.787	42.449	42.569	30.111	42.388	42.456	**42.775**
Image14	31.744	34.512	45.798	**47.567**	35.809	45.951	44.546	46.622
Image15	31.521	35.876	46.298	46.598	36.550	**46.731**	45.806	46.676
Image16	36.627	39.456	49.666	49.959	40.840	50.114	50.046	**51.138**
Image17	32.692	35.546	46.398	46.239	36.766	46.991	46.542	**47.079**
Image18	29.339	31.928	43.524	**44.222**	33.189	44.052	43.644	44.175
Image19	30.213	33.128	48.980	49.157	34.206	49.041	48.818	**50.101**
Image20	32.751	35.907	44.934	44.734	31.344	45.086	44.549	**45.222**
Image21	31.641	34.604	45.721	46.351	36.002	45.567	45.451	**46.432**
Image22	31.268	33.836	46.166	**47.270**	35.118	46.364	45.916	47.050
Image23	37.168	39.896	49.258	49.413	42.286	**50.107**	48.269	49.370
Image24	28.143	30.828	38.745	38.959	32.512	38.971	39.075	**39.129**
Average	32.038	35.004	46.545	46.947	36.126	46.793	46.266	**47.250**

Table 3
CPSNR quality comparison for twenty-four test images in Kodak PhotoCD.

Image	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
Image01	30.490	33.469	47.007	47.155	34.690	46.804	48.293	**48.480**
Image02	36.187	38.331	45.165	45.474	39.435	44.914	45.303	**46.534**
Image03	39.533	42.522	48.452	49.145	44.068	47.813	47.729	**49.453**
Image04	36.536	39.156	45.611	46.258	40.537	45.472	46.066	**47.072**
Image05	29.995	33.321	45.192	45.994	34.958	44.651	45.228	**46.354**
Image06	33.594	36.552	48.231	48.409	37.893	47.823	48.439	**48.918**
Image07	37.262	41.035	48.884	49.663	43.081	48.640	49.005	**50.350**
Image08	26.065	29.252	43.818	43.965	30.679	43.552	43.747	**44.423**
Image09	36.030	38.975	49.300	49.272	40.424	48.927	49.201	**50.101**
Image10	36.358	39.701	47.929	48.034	41.458	47.541	47.965	**48.502**
Image11	33.078	36.052	47.982	48.560	37.432	47.734	48.577	**49.488**
Image12	37.302	40.846	48.335	48.435	42.377	48.164	49.076	**49.579**
Image13	28.087	30.863	45.438	45.596	32.291	45.272	45.512	**45.876**
Image14	33.329	36.223	44.205	45.448	37.667	43.824	44.133	**45.820**
Image15	33.615	37.420	43.242	43.440	38.198	43.112	43.853	**43.886**
Image16	38.560	41.480	51.556	51.639	42.984	51.447	51.848	**52.540**
Image17	34.886	37.939	48.722	48.239	39.296	48.725	48.758	**49.154**
Image18	31.373	34.072	45.079	45.601	35.396	45.131	45.114	**45.858**
Image19	31.999	35.063	49.424	49.322	36.237	49.147	49.437	**50.234**
Image20	34.726	38.185	47.963	47.805	35.000	47.861	47.696	**48.307**
Image21	33.487	36.604	48.282	48.659	38.124	47.906	48.202	**49.003**
Image22	33.298	35.996	45.394	45.773	37.219	44.994	45.363	**45.977**
Image23	39.046	41.852	48.462	48.537	44.065	48.271	48.502	**49.556**
Image24	30.436	33.241	41.882	41.976	34.815	41.816	41.972	**42.147**
Average	33.970	37.006	46.898	47.183	38.264	46.648	47.042	**47.817**

$$w_{gr}(h,i,j+1) = \frac{1}{1+\left[\sum_{k=0}^{2}\delta_k\left|\Delta\widetilde{D}_{gr}^{h}(i,j+k)\right|\right]},$$
$$w_{gr}\left(\frac{-\pi}{4},i-1,j-1\right) = \frac{1}{1+\left[\sum_{k=0}^{2}\delta_k\left|\Delta\widetilde{D}_{gr}^{\frac{-\pi}{4}}(i-k,j-k)\right|\right]},$$
$$w_{gr}\left(\frac{-\pi}{4},i+1,j+1\right) = \frac{1}{1+\left[\sum_{k=0}^{2}\delta_k\left|\Delta\widetilde{D}_{gr}^{\frac{-\pi}{4}}(i+k,j+k)\right|\right]},$$
$$w_{gr}\left(\frac{\pi}{4},i-1,j+1\right) = \frac{1}{1+\left[\sum_{k=0}^{2}\delta_k\left|\Delta\widetilde{D}_{gr}^{\frac{\pi}{4}}(i-k,j+k)\right|\right]},$$
$$w_{gr}\left(\frac{\pi}{4},i+1,j-1\right) = \frac{1}{1+\left[\sum_{k=0}^{2}\delta_k\left|\Delta\widetilde{D}_{gr}^{\frac{\pi}{4}}(i+k,j-k)\right|\right]},$$

where $\delta_k = 2$ if $k = 1$; $\delta_k = 1$, otherwise. According to the above description, the R color value of the central pixel, $I_{Ddm}^{r}(i,j)$, can be estimated by the following rules:

$$I_{Ddm}^{r}(i,j) = I_{Ddm}^{g}(i,j) - \frac{\sum_{(d,x,y)\in\xi} w(d,x,y)D_{gr}(x,y)}{\sum_{(d,x,y)\in\xi} w(d,x,y)},$$

where $D_{gr}(x,y) = I_{Dmo}^{g}(x,y) - I_{Dmo}^{r}(x,y)$, $\xi = \{(h,i,j-1),(h,i,j+1),(\frac{-\pi}{4},i-1,j-1),(\frac{-\pi}{4},i+1,j+1),(\frac{\pi}{4},i+1,j-1),(\frac{\pi}{4},i-1,j+1)\}$, and $I_{Ddm}^{g}(i,j) = I_{Dmo}^{g}(i,j)$. For example, based on the input DTDI Lighthouse image shown in Fig. 11(a), the resultant image after recovering R color values is shown in Fig. 11(e).

Finally, we can estimate the missing B color values by the same way, and then the demosaiced full color image can be obtained. Fig. 11(f) illustrates the demosaiced full color Lighthouse image.

Table 4
S-CIELAB ΔE_{ab}^{*} quality comparison for twenty-four test images in Kodak PhotoCD.

Image	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
Image01	3.0827	1.7939	0.4484	0.4331	1.9129	0.4515	0.4158	**0.3735**
Image02	2.0621	1.4768	0.6928	0.6884	1.4665	0.6579	0.6815	**0.5832**
Image03	1.0032	0.6347	0.3872	0.3578	0.6266	0.3880	0.3949	**0.3273**
Image04	1.6594	1.0964	0.5087	0.4889	1.0598	0.4941	0.4836	**0.4233**
Image05	3.9301	2.2525	0.7094	0.6239	2.3495	0.6879	0.6949	**0.6025**
Image06	1.7148	1.0122	0.3668	0.3434	1.0505	0.3722	0.3527	**0.3149**
Image07	1.3133	0.7942	0.4205	0.3796	0.7375	0.4153	0.4229	**0.3575**
Image08	4.0916	2.2944	0.5345	0.5157	2.4097	0.5503	0.5372	**0.4865**
Image09	1.2456	0.8050	0.3182	0.3158	0.8010	0.3238	0.3183	**0.2821**
Image10	1.2286	0.7519	0.3486	0.3434	0.7224	0.3518	0.3395	**0.3052**
Image11	2.3376	1.3949	0.4803	0.4474	1.4605	0.4596	0.4381	**0.3897**
Image12	0.8996	0.5628	0.2787	0.2672	0.5536	0.2754	0.2661	**0.2331**
Image13	4.7080	2.8729	0.6002	0.5272	3.0125	0.5753	0.5858	**0.5252**
Image14	2.4841	1.5260	0.5866	0.5259	1.5310	0.5733	0.5994	**0.5007**
Image15	1.9805	1.3147	0.6292	0.6115	1.3439	0.6233	0.5769	**0.5398**
Image16	1.2053	0.7304	0.3336	0.3146	0.7372	0.3337	0.3108	**0.2737**
Image17	2.0241	1.2672	0.5515	0.5409	1.2683	0.5128	0.5163	**0.4580**
Image18	3.5271	2.2854	0.8016	0.7215	2.2751	0.7647	0.8010	**0.7017**
Image19	2.3556	1.4776	0.4061	0.4082	1.5067	0.4013	0.4052	**0.3594**
Image20	1.5201	0.9746	0.3699	0.3549	1.4386	0.3541	0.3713	**0.3302**
Image21	1.9835	1.1863	0.3837	0.3644	1.2192	0.3863	0.3716	**0.3310**
Image22	2.0835	1.3878	0.5084	0.4710	1.3502	0.4966	0.5148	**0.4462**
Image23	1.0075	0.7419	0.4284	0.4294	0.6736	0.4136	0.4256	**0.3761**
Image24	2.5521	1.5048	0.5175	0.4862	1.5339	0.5186	0.5128	**0.4671**
Average	2.1667	1.3391	0.4838	0.4567	1.3767	0.4742	0.4724	**0.4162**

Table 5
Average PSNR quality comparison in R color plane for the images in each category of CorelDRAW image database.

Category	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
Abstract	32.433	35.474	50.120	50.963	37.134	49.712	48.253	**51.694**
Animals	33.444	36.931	47.947	48.942	38.633	47.285	**50.437**	49.874
Architecture	29.768	32.918	48.405	49.143	34.422	47.872	48.574	**49.910**
Backgrounds	27.008	30.089	47.814	48.664	31.731	47.328	47.894	**49.503**
Business	31.134	34.301	48.128	48.912	35.602	47.544	48.469	**49.807**
Design	31.015	34.405	47.281	47.937	35.856	46.831	47.808	**48.933**
Food_Drink	31.793	34.961	47.101	48.177	36.573	46.795	**49.473**	49.114
Home	27.837	30.839	47.630	48.480	32.238	47.152	48.013	**49.670**
Int_Arch	30.803	34.370	49.400	50.262	36.021	48.867	49.417	**50.947**
Landscape	29.294	32.117	50.987	52.067	33.615	50.518	52.255	**52.725**
Natural	31.087	34.267	47.918	48.844	36.006	47.517	48.181	**49.928**
Objects	31.565	34.637	46.785	47.674	36.355	46.248	47.010	**48.715**
Sunsets	35.921	39.120	47.241	48.444	40.894	46.446	47.187	**48.872**
Technology	32.498	35.871	47.331	48.114	37.486	46.710	47.602	**49.094**
Texture	27.827	31.050	47.454	48.296	32.789	46.941	47.457	**49.287**
Travel	30.110	33.132	49.142	50.100	34.627	48.567	49.277	**50.896**
Undersea	34.471	37.578	49.283	50.485	39.346	48.712	49.290	**50.972**
Water	30.790	33.766	50.844	51.808	35.333	50.258	50.976	**52.467**
Average	31.044	34.213	48.378	49.295	35.815	47.850	48.754	**50.134**

Table 6
Average PSNR quality comparison in B color plane for the images in each category of CorelDRAW image database.

Category	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
Abstract	32.602	35.485	50.743	52.517	37.106	50.478	49.201	**52.694**
Animals	33.696	36.894	49.773	51.610	38.502	49.202	50.650	**51.707**
Architecture	29.874	32.974	49.806	**51.764**	34.449	49.265	49.245	51.567
Backgrounds	27.261	30.339	48.276	50.222	31.960	47.864	47.805	**50.354**
Business	31.250	34.384	50.014	51.556	35.743	49.432	49.521	**51.700**
Design	31.206	34.591	50.638	52.060	35.984	50.104	50.047	**52.160**
Food_Drink	32.021	35.034	49.074	51.245	36.537	48.687	48.544	**51.340**
Home	27.938	30.914	49.327	51.256	32.295	49.045	48.914	**51.363**
Int_Arch	30.809	34.305	49.237	**51.544**	36.072	48.730	48.649	51.520
Landscape	29.466	32.166	50.353	52.512	33.546	50.037	51.032	**52.549**
Natural	31.395	34.297	48.332	50.867	36.037	48.121	47.823	**50.870**
Objects	31.749	34.808	49.041	51.021	36.472	48.526	48.326	**51.022**
Sunsets	36.619	39.337	49.329	**51.081**	40.683	48.482	48.446	50.959
Technology	32.556	35.676	49.302	**51.513**	37.430	48.709	48.420	51.299
Texture	28.088	31.281	48.341	50.585	33.031	48.023	47.785	**50.688**
Travel	30.301	33.188	49.922	52.014	34.624	49.403	49.522	**52.095**
Undersea	34.815	37.812	50.732	52.880	39.714	50.246	50.219	**52.929**
Water	30.977	33.780	50.359	52.578	35.313	49.873	50.158	**52.686**
Average	31.257	34.292	49.589	51.601	35.861	49.124	49.128	**51.639**

Table 7
Average CPSNR quality comparison for the images in each category of CorelDRAW image database.

Category	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
Abstract	34.276	37.238	52.027	53.426	38.876	51.685	50.246	**53.769**
Animals	35.322	38.659	50.304	51.654	40.309	49.643	52.151	**52.212**
Architecture	31.579	34.701	50.481	51.626	36.187	49.951	50.394	**52.117**
Backgrounds	28.889	31.968	49.304	50.878	33.599	48.863	49.166	**51.188**
Business	32.948	36.097	50.446	51.526	37.418	49.865	50.518	**52.155**
Design	32.868	36.256	50.224	51.122	37.677	49.737	50.416	**51.797**
Food_Drink	33.661	36.754	49.500	51.007	38.310	49.151	49.473	**51.486**
Home	29.647	32.634	49.640	51.232	34.021	49.314	49.844	**51.773**
Int_Arch	32.565	36.098	50.927	52.407	37.804	50.411	50.657	**52.628**
Landscape	31.139	33.902	52.225	53.842	35.340	51.845	52.255	**54.217**
Natural	32.992	36.028	49.453	51.082	37.769	49.156	49.385	**51.704**
Objects	33.410	36.465	49.194	50.412	38.151	48.632	49.070	**51.131**
Sunsets	38.008	40.967	49.753	51.261	42.521	48.938	49.403	**51.374**
Technology	34.284	37.482	49.651	50.958	39.178	49.017	49.427	**51.475**
Texture	29.713	32.916	49.060	50.440	34.652	48.645	48.811	**51.133**
Travel	31.963	34.916	51.024	52.383	36.378	50.470	50.913	**52.871**
Undersea	36.398	39.448	51.486	53.023	41.273	50.950	51.236	**53.366**
Water	32.641	35.528	52.114	53.824	37.078	51.573	52.043	**54.021**
Average	32.906	36.003	50.378	51.784	37.586	49.880	50.300	**52.245**

Table 8
Average S-CIELAB ΔE^*_{ab} quality comparison for the images in each category of CorelDRAW image database.

Category	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
Abstract	2.4374	1.4728	0.4094	0.3279	1.5207	0.4267	0.4910	**0.3098**
Animals	2.3462	1.4760	0.5182	0.4073	1.5264	0.5395	0.3976	**0.3874**
Architecture	3.0709	1.8388	0.4462	0.3642	1.9489	0.4764	0.4369	**0.3490**
Backgrounds	5.0452	2.9344	0.6495	0.5255	3.1260	0.6785	0.6781	**0.5131**
Business	2.3937	1.4867	0.4270	0.3473	1.5614	0.4546	0.4143	**0.3292**
Design	2.2533	1.3651	0.4062	0.3409	1.4014	0.4314	0.3981	**0.3210**
Food_Drink	2.6322	1.6857	0.5906	0.4562	1.6702	0.6147	0.6005	**0.4476**
Home	4.2218	2.5020	0.4596	0.3705	2.7225	0.4812	0.4507	**0.3488**
Int_Arch	2.9773	1.7131	0.4364	0.3454	1.7431	0.4728	0.4466	**0.3344**
Landscape	3.4676	2.0625	0.3423	0.2697	2.2466	0.3612	0.3325	**0.2561**
Natural	3.3877	2.0874	0.5493	0.4311	2.1468	0.5697	0.5555	**0.4093**
Objects	2.6064	1.6417	0.5446	0.4347	1.6542	0.5821	0.5491	**0.4133**
Sunsets	1.7976	1.2713	0.6381	0.4993	1.2423	0.6774	0.6133	**0.4990**
Technology	2.3532	1.4960	0.5570	0.4463	1.5426	0.5859	0.5398	**0.4210**
Texture	4.4700	2.6351	0.5584	0.4382	2.7876	0.5887	0.5870	**0.4179**
Travel	3.0628	1.8585	0.3892	0.3115	1.9882	0.4185	0.3831	**0.2982**
Undersea	1.7289	1.0714	0.3793	0.2939	1.0665	0.4083	0.3828	**0.2868**
Water	3.1118	1.8542	0.3862	0.3028	1.9864	0.4119	0.3825	**0.2908**
Average	2.9647	1.8029	0.4826	0.3840	1.8823	0.5100	0.4800	**0.3685**

3. Experimental results

To test the quality performance of the proposed demosaicing algorithm, we used two test image sets to conduct experiments. The first set included twenty-four test images in Kodak PhotoCD [23]. The twenty-four test images are shown in Fig. 12 and they have been widely used for evaluating demosaicing algorithms. The other set was CorelDRAW image database [24] consisting of eighteen categories of images. Since CorelDRAW image database comprised one thousand, three hundred, and fifty-six test images, it could be used to demonstrate the general applicability of the concerned demosaicing algorithms. These test images were first down-sampled to obtain DTDI mosaic images. Then, we ran five existing demosaicing algorithms in [12], modified Pei and Tam's algorithm (MPT algorithm), modified Chung et al.'s algorithm (MCEA algorithm) and the proposed algorithm on the above mentioned DTDI mosaic images. The five existing demosaicing algorithms in [12] were bilinear interpolation algorithm (BI algorithm), modified Laroche and Prescott's algorithm (MLP algorithm), modified Hamilton and Adams's algorithm (MHA algorithm), weighted absolute interpolation algorithm (WAI algorithm), and least squares approximation based on WAI algorithm (L2W algorithm). Further, MPT algorithm was similar to the R/B color interpolation process of original Pei and Tam's algorithm. The only difference was that six neighboring pixels corresponding to interpolation directions were taken into account in the interpolation process. MCEA algorithm, which was similar to the R/B color interpolation process of original Chung et al.'s algorithm, could be developed by modifying the luminance operator of the SL-based marks slightly to deal with DTDI mosaic pattern. The size of each test image in the two sets was 512×768. All algorithms adopted in the experiments were implemented on the IBM compatible computer with Intel Core i5 CPU 2.53 GHz and 3 GB RAM. The operating system used was MS-Windows XP and the program developing environment was Borland C++ Builder 6.0. Furthermore, all the experimental results are available in [25].

To compare the quality performance among the eight demosaicing algorithms, we used three objective measures, i.e., PSNR, CPSNR, and S-CIELAB ΔE^*_{ab} metric [13,15], and one subjective image quality measure, i.e., color artifacts, to test the applicability and the quality of outcome for each algorithm. The PSNR of a demosaiced color plane with size $M \times N$ is defined as

$$\mathrm{PSNR} = 10\log_{10}\frac{255^2}{\frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left[I^c_{ori}(i,j) - I^c_{Ddm}(i,j)\right]^2},$$

where c can be r or b; $I^r_{ori}(i,j)$ and $I^r_{Ddm}(i,j)$, respectively, denote the R color components of the pixels at location (i,j) in an original full color image and a demosaiced image; $I^b_{ori}(i,j)$ and $I^b_{Ddm}(i,j)$, respectively, denote the B color components of the pixels at location (i,j) in an original full color image and a demosaiced image. The lager the PSNR, the better will be the image quality. The CPSNR of a demosaiced image with size $M \times N$ is defined as

$$\mathrm{CPSNR} = 10\log_{10}\frac{255^2}{\frac{1}{3MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\sum_{c\in C}\left[I^c_{ori}(i,j) - I^c_{Ddm}(i,j)\right]^2},$$

where $C = \{r,g,b\}$; $I^r_{ori}(i,j), I^g_{ori}(i,j)$, and $I^b_{ori}(i,j)$ denote the three color components of the pixel at location (i,j) in an original full color image; $I^r_{Ddm}(i,j), I^g_{Ddm}(i,j)$, and $I^b_{Ddm}(i,j)$ denote the three color components of the pixel at location (i,j) in a demosaiced image. The lager the CPSNR, the better will be the image quality. The S-CIELAB ΔE^*_{ab} of a demosaiced color image with size $M \times N$ is defined by

$$\Delta E^*_{ab} = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left\{\sqrt{\sum_{c\in\Psi}\left[LAB^c_{ori}(i,j) - LAB^c_{Ddm}(i,j)\right]^2}\right\}, \quad (4)$$

where $\Psi = \{L,a,b\}$; $LAB^L_{ori}(i,j), LAB^a_{ori}(i,j)$, and $LAB^b_{ori}(i,j)$ denote the three CIELAB color components of the pixel at location (i,j) in an original full color image; $LAB^L_{Ddm}(i,j), LAB^a_{Ddm}(i,j)$, and $LAB^b_{Ddm}(i,j)$ denote the three CIELAB color components of the pixel at location (i,j) in a demosaiced image. The smaller the S-CIELAB ΔE^*_{ab}, the better will be the image quality. The transformation from RGB color space to CIELAB color space can be found in [13].

Based on the twenty-four test images in Kodak PhotoCD, i.e., the first image set, we ran the concerned eight demosaicing

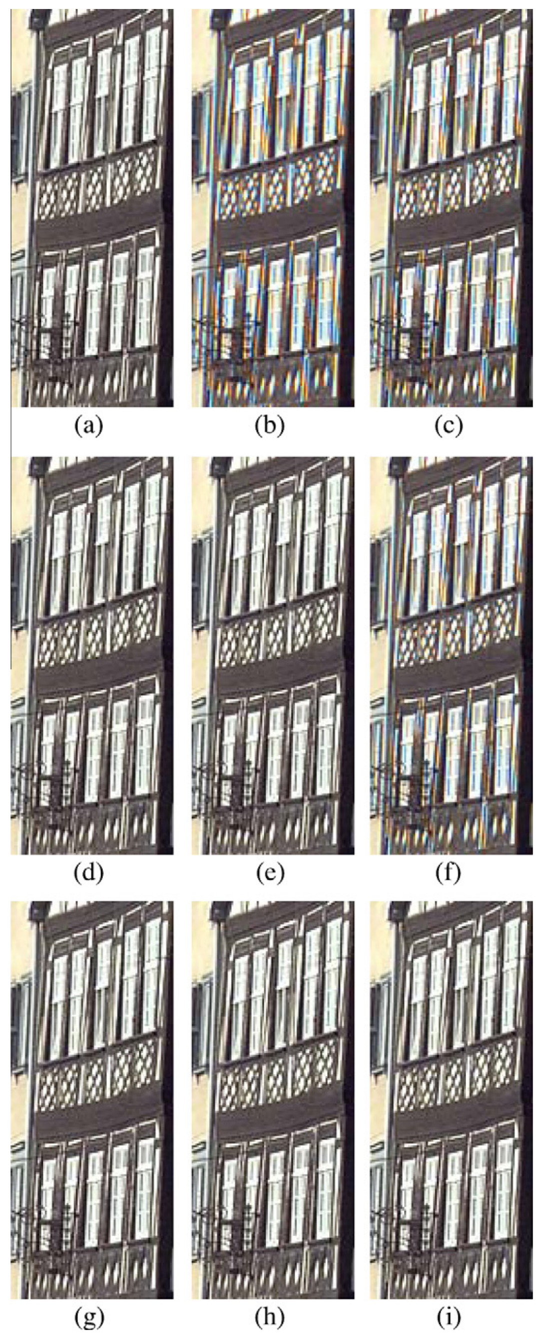

Fig. 13. Magnified subimages cut from the testing image No. 08 in Kodak PhotoCD. (a) Original full color image and the demosaiced images generated by (b) BI algorithm, (c) WAI algorithm, (d) MHA algorithm, (e) MLP algorithm, (f) L2W algorithm, (g) MPT algorithm, (h) MCEA algorithm, and (i) the proposed algorithm.

algorithms. Tables 1 and 2 demonstrate the PSNR quality comparison for the demosaiced R and B color planes, respectively. Tables 3 and 4 show the demosaiced image quality comparison in terms of CPSNR and S-CIELAB ΔE^{*}_{ab}, respectively. In Tables 1–4, the entries with the largest PSNR and CPSNR are highlighted by boldface; the ones with the smallest S-CIELAB ΔE^{*}_{ab} are highlighted by boldface, too. From the tables, it indicates that on average, the proposed demosaicing algorithm has the best demosaiced image quality in terms of PSNR, CPSNR, and S-CIELAB ΔE^{*}_{ab}. Then, we took the test images in CorelDRAW image database, i.e., the second image set, to compare the image quality performance. Based on the eighteen image categories in CorelDRAW image database, Tables 5 and 6 show the average PSNR quality comparison for the demosaiced R and B color planes, respectively, and the demosaiced image quality comparison in terms of CPSNR and S-CIELAB ΔE^{*}_{ab} are demonstrated in Tables 7 and 8, respectively. In Tables 5–8, we also highlight the entries with the best image quality performance by boldface. Tables 5–8 reveal that on average, the proposed demosaicing algorithm has the best image quality performance in terms of PSNR, CPSNR, and S-CIELAB ΔE^{*}_{ab}.

Next, we used the subjective measure to demonstrate the visual quality advantage of the proposed demosaicing algorithm. After demosaicing DTDI mosaic images, some degree of color artifacts may appear on nonsmooth regions of the demosaiced images. We first took the magnified subimages cut from image No. 08 in Kodak PhotoCD to compare the visual effect. Fig. 13(a)–(i) illustrate the nine magnified subimages cut from the original test image and the ones generated by the concerned demosaicing algorithms. Comparing the visual effect between the original magnified subimage shown in Fig. 13(a) and the ones in Fig. 13(b)–(i), it is obvious that MLP algorithm, MHA algorithm, MPT algorithm, MCEA algorithm, and the proposed demosaicing algorithm have the same visual effect and produce less color artifacts than the three other demosaicing algorithms. We subsequently took the magnified subimages cut from image No. 19 in Kodak PhotoCD to depict the visual comparison. Fig. 14(a)–(i) illustrate the magnified subimages cut from the original test image and the demosaiced images generated by the concerned eight demosaicing algorithms. By visual comparison, it is observed that MLP algorithm, MHA algorithm, MPT algorithm, MCEA algorithm, and the proposed demosaicing algorithm have the same visual benefit and better visual effect when compared with the three other demosaicing algorithms. Then, we used magnified subimages cut from the test images in CorelDRAW image database to demonstrate the visual effect comparison. Based on the subimages cut from the testing image No. 127 in the Water category and the testing image No. 12 in the Animal category, Figs. 15 and 16, respectively, show the visual effect comparison among the concerned demosaicing algorithms. From the two figures, it is clear that less color artifacts exist in the demosaiced images produced by MLP algorithm, MHA algorithm, MPT algorithm, MCEA algorithm, and the proposed demosaicing algorithm. Although MLP algorithm, MHA algorithm, MPT algorithm, MCEA algorithm, and the proposed demosaicing algorithm have the same visual effect, Tables 1–8 indicate that the proposed algorithm has the best image quality in terms of PSNR, CPSNR and S-CIELAB ΔE^{*}_{ab}.

Finally, based on all the test images, Table 9 shows the overall performance comparison in terms of average PSNR, average CPSNR,

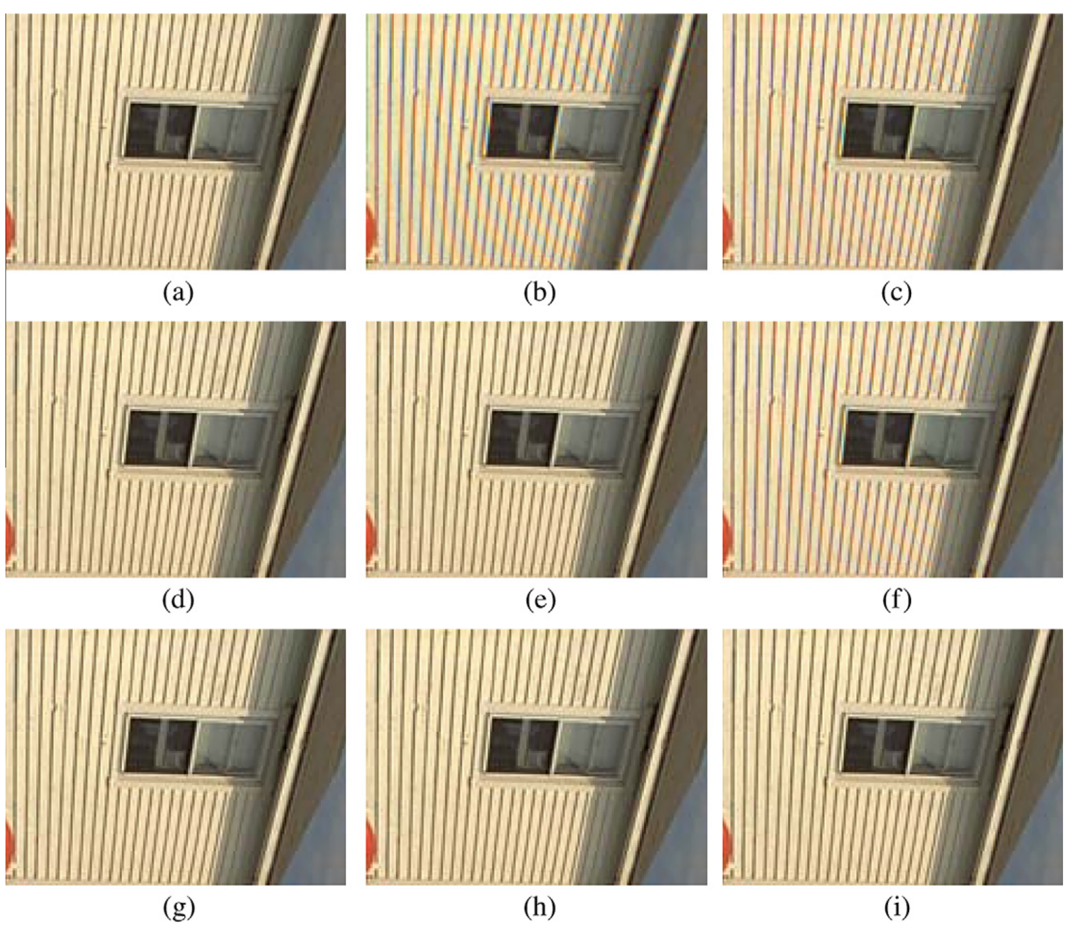

Fig. 14. Magnified subimages cut from the testing image No. 19 Kodak PhotoCD. (a) Original full color image and the demosaiced images generated by (b) BI algorithm, (c) WAI algorithm, (d) MHA algorithm, (e) MLP algorithm, (f) L2W algorithm, (g) MPT algorithm, (h) MCEA algorithm, and (i) the proposed algorithm.

average S-CIELAB ΔE^*_{ab}, average execution-time, and memory requirement between the proposed demosaicing algorithm and the other seven compared algorithms. We measured the memory requirement comparison in terms of the maximum amount of variables required for demosaicing the missing R or B color component in a pixel. The number of required variables were counted for the two different formats, i.e., integer and floating point. For example, twenty-three integer variables and eight floating point variables are required to demosaic the missing R or B color component in a pixel in our proposed algorithm. In Table 9, the variable "W" denotes the number of sampling pixels in the training set of the least squares approximation. From the table, it is observed that the average execution-time and memory requirement of the proposed demosaicing algorithm is moderate when compared with the other seven algorithms. However, the proposed algorithm has the best demosaiced image quality performance in terms of average PSNR, average CPSNR and average S-CIELAB ΔE^*_{ab} among the concerned demosaicing algorithms.

4. Conclusions

In this paper, a novel edge sensing-based demosaicing algorithm for DTDI mosaic images has been presented. In the proposed algorithm, the SI-based masks are first used to extract more accurate gradient information in the color difference domain. Based on the extracted more accurate gradient information, the proposed edge sensing-based demosaicing algorithm can generate good quality of a demosaiced image. By experimenting on

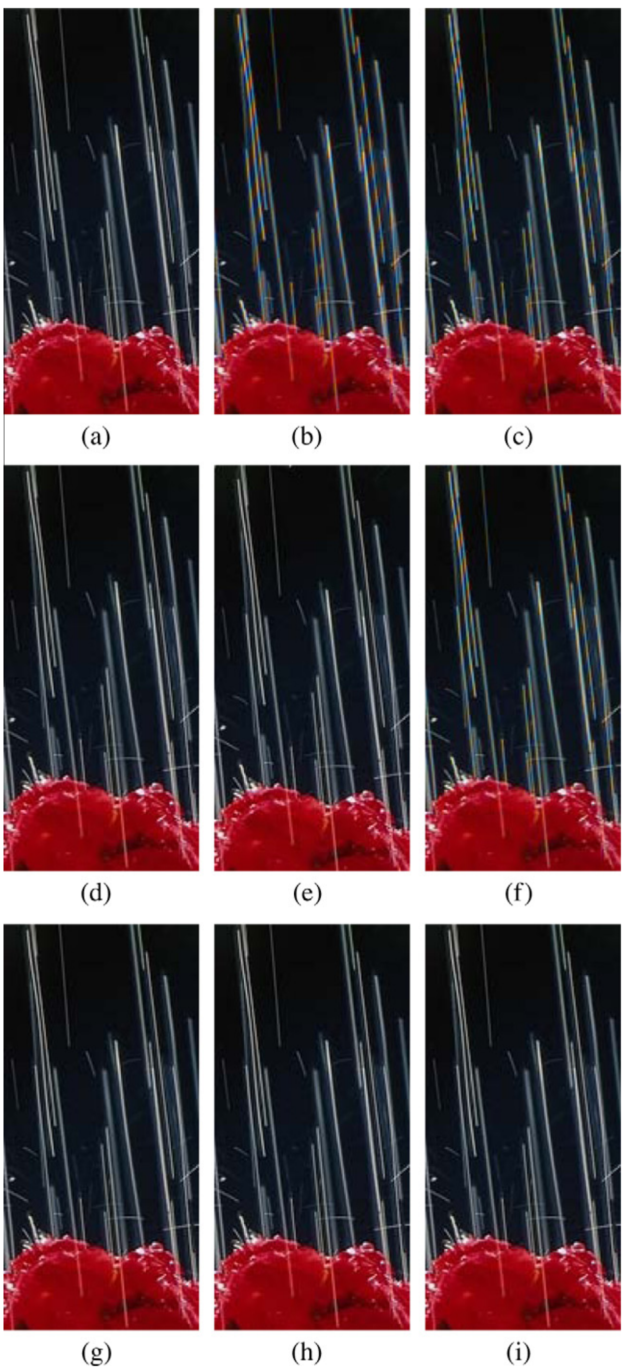

Fig. 15. Magnified subimages cut from the testing image No. 127 in the Water category of CorelDRAW image database. (a) Original full color image and the demosaiced images generated by (b) BI algorithm, (c) WAI algorithm, (d) MHA algorithm, (e) MLP algorithm, (f) L2W algorithm, (g) MPT algorithm, (h) MCEA algorithm, and (i) the proposed algorithm.

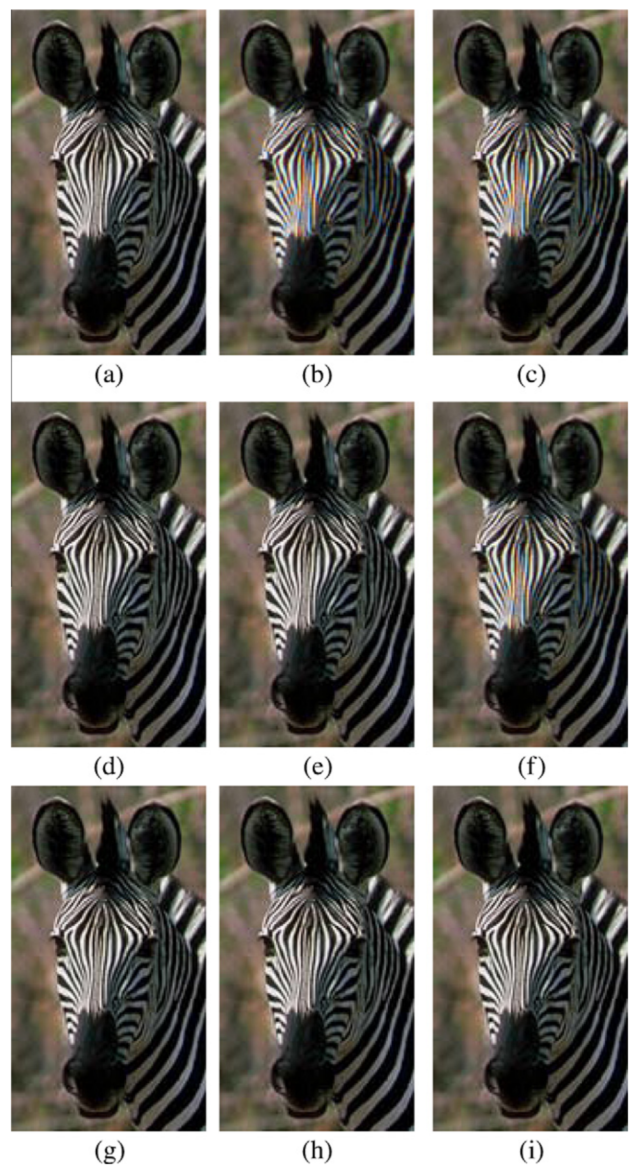

Fig. 16. Magnified subimages cut from the testing image No. 12 in the Animal category of CorelDRAW image database. (a) Original full color image and the demosaiced images generated by (b) BI algorithm, (c) WAI algorithm, (d) MHA algorithm, (e) MLP algorithm, (f) L2W algorithm, (g) MPT algorithm, (h) MCEA algorithm, and (i) the proposed algorithm.

Table 9
The overall performance comparison of the eight concerned demosaicing algorithms for the test DTDI mosaic images in the two sets.

	Algorithm							
	BI [12]	WAI [12]	MHA [12]	MLP [12]	L2W [12]	MPT [18]	MCEA [4]	OURS
PSNR for R plane	31.721	34.870	46.473	47.022	36.409	45.942	46.850	47.862
PSNR for B plane	31.647	34.648	48.067	49.274	35.993	47.958	47.697	49.445
CPSNR	33.438	36.505	48.638	49.483	37.925	48.264	48.671	50.031
S-CIELAB ΔE^{*}_{ab}	2.5657	1.5710	0.4832	0.4204	1.6295	0.4921	0.4762	0.3923
Execution-time (s)	0.014	0.032	0.141	0.016	3.276	0.125	0.506	0.320
Memory requirement								
Integer	2	8	17	5	8	12	36	23
Floating point	0	0	0	0	39 + 2 W	0	13	8

more than one thousand and three hundred test DTDI mosaic images, the results demonstrate that the proposed demosaicing algorithm has better demosaiced image quality when compared with five existing demosaicing algorithms in [12], MPT algorithm, and MCEA algorithm. Because the proposed demosaicing algorithm can generate demosaiced image with better quality, it is an interesting research topic to apply the results of this paper to the field of depth estimation in the panorama [17,22].

Appendix A. The derivation of the SI-based masks

Combining Eqs. (2) and (3), the six SI-based masks used to extract gradient information in the color difference domain directly can be obtained by the following derivation: for the pixels at position $(i,j)\in\Omega_{gr}$, it yields

$$\begin{aligned}\Delta\widetilde{D}^{h}_{gr}(i,j)&=\left\{\begin{array}{l}\begin{bmatrix}\widetilde{D}_{gr}(i-1,j+1)+\widetilde{D}_{gr}(i+1,j+1)\\-\widetilde{D}_{gr}(i-1,j-1)-\widetilde{D}_{gr}(i+1,j-1)\end{bmatrix}\\+2\left[\widetilde{D}_{gr}(i,j+1)-\widetilde{D}_{gr}(i,j-1)\right]\end{array}\right\}\\&=\frac{1}{2}\left\{\begin{array}{l}\begin{bmatrix}D_{gr}(i-1,j+2)+D_{gr}(i+1,j+2)\\-D_{gr}(i-1,j-2)-D_{gr}(i+1,j-2)\end{bmatrix}\\+2\left[D_{gr}(i,j+2)-D_{gr}(i,j-2)\right]\end{array}\right\},\end{aligned}$$

$$\begin{aligned}\Delta\widetilde{D}^{\frac{\pi}{4}}_{gr}(i,j)&=\left\{\begin{array}{l}\begin{bmatrix}\widetilde{D}_{gr}(i-1,j)+\widetilde{D}_{gr}(i,j+1)\\-\widetilde{D}_{gr}(i,j-1)-\widetilde{D}_{gr}(i+1,j)\end{bmatrix}\\+2\left[\widetilde{D}_{gr}(i-1,j+1)-\widetilde{D}_{gr}(i+1,j-1)\right]\end{array}\right\}\\&=\frac{1}{2}\left\{\begin{array}{l}\left[D_{gr}(i,j+2)-D_{gr}(i,j-2)\right]\\+2\left[D_{gr}(i-1,j+2)-D_{gr}(i+1,j-2)\right]\\+4\left[D_{gr}(i-1,j)-D_{gr}(i+1,j)\right]\end{array}\right\},\quad\text{and}\end{aligned}$$

$$\begin{aligned}\Delta\widetilde{D}^{\frac{-\pi}{4}}_{gr}(i,j)&=\left\{\begin{array}{l}\begin{bmatrix}\widetilde{D}_{gr}(i,j+1)+\widetilde{D}_{gr}(i+1,j)\\-\widetilde{D}_{gr}(i-1,j)-\widetilde{D}_{gr}(i,j-1)\end{bmatrix}\\+2\left[\widetilde{D}_{gr}(i+1,j+1)-\widetilde{D}_{gr}(i-1,j-1)\right]\end{array}\right\}\\&=\frac{1}{2}\left\{\begin{array}{l}\left[D_{gr}(i,j+2)-D_{gr}(i,j-2)\right]\\+2\left[D_{gr}(i+1,j+2)-D_{gr}(i-1,j-2)\right]\\+4\left[D_{gr}(i+1,j)-D_{gr}(i-1,j)\right]\end{array}\right\},\end{aligned}$$

For the pixels at position $(i,j)\notin\Omega_{gr}$, it yields

$$\begin{aligned}\Delta\widetilde{D}^{h}_{gr}(i,j)&=\left\{\begin{array}{l}\begin{bmatrix}\widetilde{D}_{gr}(i-1,j+1)+\widetilde{D}_{gr}(i+1,j+1)\\-\widetilde{D}_{gr}(i-1,j-1)-\widetilde{D}_{gr}(i+1,j-1)\end{bmatrix}\\+2\left[\widetilde{D}_{gr}(i,j+1)-\widetilde{D}_{gr}(i,j-1)\right]\end{array}\right\}\\&=\frac{1}{2}\left\{\begin{array}{l}2\begin{bmatrix}D_{gr}(i-1,j+1)+D_{gr}(i+1,j+1)\\-D_{gr}(i-1,j-1)-D_{gr}(i+1,j-1)\end{bmatrix}\\+4\left[D_{gr}(i,j+1)-D_{gr}(i,j-1)\right]\end{array}\right\},\end{aligned}$$

$$\begin{aligned}\Delta\widetilde{D}^{\frac{\pi}{4}}_{gr}(i,j)&=\left\{\begin{array}{l}\begin{bmatrix}\widetilde{D}_{gr}(i-1,j)+\widetilde{D}_{gr}(i,j+1)\\-\widetilde{D}_{gr}(i,j-1)-\widetilde{D}_{gr}(i+1,j)\end{bmatrix}\\+2\left[\widetilde{D}_{gr}(i-1,j+1)-\widetilde{D}_{gr}(i+1,j-1)\right]\end{array}\right\}\\&=\frac{1}{2}\left\{\begin{array}{l}\left[D_{gr}(i-1,j-1)-D_{gr}(i+1,j+1)\right]\\+2\left[D_{gr}(i,j+1)-D_{gr}(i,j-1)\right]\\+5\left[D_{gr}(i-1,j+1)-D_{gr}(i+1,j-1)\right]\end{array}\right\},\quad\text{and}\end{aligned}$$

$$\begin{aligned}\Delta\widetilde{D}^{\frac{-\pi}{4}}_{gr}(i,j)&=\left\{\begin{array}{l}\begin{bmatrix}\widetilde{D}_{gr}(i,j+1)+\widetilde{D}_{gr}(i+1,j)\\-\widetilde{D}_{gr}(i-1,j)-\widetilde{D}_{gr}(i,j-1)\end{bmatrix}\\+2\left[\widetilde{D}_{gr}(i+1,j+1)-\widetilde{D}_{gr}(i-1,j-1)\right]\end{array}\right\}\\&=\frac{1}{2}\left\{\begin{array}{l}\left[D_{gr}(i+1,j-1)-D_{gr}(i-1,j+1)\right]\\+2\left[D_{gr}(i,j+1)-D_{gr}(i,j-1)\right]\\+5\left[D_{gr}(i+1,j+1)-D_{gr}(i-1,j-1)\right]\end{array}\right\}\end{aligned}$$

References

[1] B.E. Bayer, Color imaging array, US Patent# 3 971 065, 1976.
[2] E. Bodenstorfer, J. Fürtler, J. Brodersen, K.J. Mayer, C. Eckel, K. Gravogel, H. Nachtnebel, High speed line-scan camera with digital time delay integration, in: Proceedings of Electronic Imaging Conference on Real Time Imaging, 2007, pp. 64960I-1–64960I-10.
[3] K.H. Chung, Y.H. Chan, An edge-directed demosaicing algorithm based on integrated gradient, in: Proceedings of IEEE International Conference on Multimedia and Expo 2010 (ICME 2010), 2010, pp. 388–393.
[4] K.L. Chung, W.J. Yang, W.M. Yan, C.C. Wang, Demosaicing of color filter array captured images using gradient edge detection masks and adaptive heterogeneity-projection, IEEE Trans. Image Process. 17 (2008) 2356–2367.
[5] K.L. Chung, W.J. Yang, P.Y. Chen, W.M. Yan, C.S. Fuh, New joint demosaicing and zooming algorithm for color filter array, IEEE Trans. Consumer Electron. 55 (2009) 1477–1486.
[6] K.L. Chung, W.J. Yang, J.H. Yu, W.M. Yan, C.S. Fuh, Novel quality-effective zooming algorithm for color filter array, J. Electron. Imag. 19 (2010) 013005-1–013005-15.
[7] K.L. Chung, W.J. Yang, W.M. Yan, C.S. Fuh, New joint demosaicing and arbitrary-ratio resizing algorithm for color filter array using DCT approach, IEEE Trans. Consumer Electron. 56 (2010) 783–791.
[8] P.E. Debevec, J. Malik, Recovering high dynamic range radiance maps from photographs, in: Proceedings of SIGGRAPH, 1997, pp. 369–378.
[9] J. Fürtler, E. Bodenstorfer, K.J. Mayer, J. Brodersen, D. Heiss, H. Penz, C. Eckel, K. Gravogel, H. Nachtnebel, High performance camera module for fast quality inspection in industrial printing applications, in: Proceedings of the Electronic Imaging Conference on Machine Vision Applications in Industrial Inspection, 2007, pp. 65030J-1–65030J-12.
[10] R. Gonzalez, R. Woods, Digital Image Processing, Addison Wesley, New York, 1992.
[11] J. Hamilton, J. Adams, Adaptive color plane interpolation in single sensor color electronic camera, US Patent# 5 629 734, 1997.
[12] D. Heiss-Czedik, R. Huber-Mörk, D. Soukup, H. Penz, B.L. García, Demosaicing algorithms for area- and line-scan cameras in print inspection, J. Visual Commun. Image Represent. 20 (2009) 389–398.
[13] R.W.G. Hunt, Measuring Colour, second ed., Ellis Horwood, Chichester, U.K, 1995.
[14] C. Larocheand, M. Prescott, Apparatus and method for adaptively interpolating a full color image utilizing chrominance gradients, US Patent# 5 373 322, 1994.
[15] W. Lu, Y.P. Tan, Color filter array demosaicking: new method and performance measures, IEEE Trans. Image Process. 12 (2003) 1194–1210.
[16] R. Lukac, K.N. Plataniotis, Color filter arrays: design and performance analysis, IEEE Trans. Consumer Electron. 51 (2005) 1260–1267.

[17] H. Nagahara, A. Ichikawa, M. Yachida, Depth estimation from the color drift of a route panorama, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, pp. 4072–4077.
[18] S.C. Pei, I.K. Tam, Effective color interpolation in CCD color filter arrays using signal correlation, IEEE Trans. Circ. Syst. Video Technol. 13 (2003) 503–513.
[19] I. Pekkucuksen, Y. Altunbasak, Edge strength filter based color filter array interpolation, IEEE Trans. Image Process. 21 (2012) 393–397.
[20] T. Sakamoto, C. Nakanishi, T. Hase, Software pixel interpolation for digital still camera suitable for a 32-bit MCU, IEEE Trans. Consumer Electron. 44 (1998) 1342–1352.
[21] H.S. Wong, Y.L. Yao, E.S. Schlig, TDI charge-coupled devices: design and applications, IBM J. Res. Develop. Arch. 36 (1992) 83–105.
[22] J.Y. Zheng, M. Shi, Scanning depth of route panorama based on stationary blur, Int. J. Comput. Vision 78 (2008) 169–186.
[23] <http://www.site.uottawa.ca/~edubois/demosaicking/>.
[24] <http://www.corel.com/>.
[25] <http://140.118.175.164/WJYang/paper/DTDIDM/>.