National Taiwan University of Science and Technology
Department of Computer Science and Information Engineering

Efficient High-Quality Demosaicing Algorithm for Color Filter Array

鍾國亮 Kuo-Liang Chung
楊偉楨 Wei-Jen Yang
顏文明 Wen-Ming Yan
王中周 Chung-Chou Wang


July 2007
Abstract Most modern digital cameras use a signal CCD/CMOS sensor with a color filter array (CFA) to capture a color image in which each pixel has only one primary color, and the captured image is called the mosaic image. Without demosaicing processing, this paper first proposes a new approach to capture more accurate color gradient/edge information on mosaic images directly. Next, based on spectral–spatial correlation, a novel adaptive heterogeneity–projection for mosaic images is presented. Combining the captured color gradient/edge information and the adaptive heterogeneity–projection values, a new edge–sensing demosaicing algorithm is presented. Based on twenty–four popular testing mosaic images, experimental results demonstrated that our proposed high–quality demosaicing algorithm has the best image quality performance when compared to currently published four algorithms by Pei and Tam, Lu and Tan, Dubois, and Tsai and Song.

Index Terms: Adaptive heterogeneity projection, Color filter array (CFA), Demosaicing algorithm, Digital cameras, Mosaic images, CPSNR, Sobel operator.

I. INTRODUCTION

Recently, digital cameras have become more and more popular in consumer electronics market. In order to economize the hardware cost, instead of using three sensors, most digital cameras capture a color image with a single sensor array based on the well–known Bayer color filter array (CFA) [1], where each pixel in the captured image has only one measured color and this kind of images is called mosaic images. Fig. 1 depicts the Bayer CFA structure. Because $G$ (green) color channel is the most important factor to determine the luminance of the color image, half of the pixels in Bayer CFA structure are assigned to $G$ channel. $R$ (red) and $B$ (blue) color channels, which share the other half pixels in the Bayer CFA structure, are considered as the chrominance signals. Fig. 2 shows the full color Lighthouse image. After filtering Fig. 2 via the Bayer CFA structure, the resultant mosaic image is shown in Fig. 3.
In order to recover the full color image from the input mosaic image, the demosaicing process is used to estimate the other two color channels for each pixel [9]. Bilinear interpolation (BI) [21] is the simplest demosaicing algorithm in which the unknown two color channels of each pixel are obtained by averaging its proper adjacent pixels. In [8], Gunturk et al. presented an efficient demosaicing algorithm by using alternating projections. Based on color correlation concept, Pei and Tam [20] presented an efficient demosaicing algorithm. In [14], Lu and Tan presented an efficient demosaicing algorithm based on exploiting spatial and spectral correlations among adjacent pixels and they also presented a quality measure. Based on primary-consistent soft-decision framework, Wu and Zhang
[28] presented a multiple estimation to estimate unknown color values according to different edge directions. Based on projection–onto–convex–set approach, Li [13] presented the first iterative demosaicing algorithm. In [4], Dubois presented a novel demosaicing algorithm based on frequency–domain representation. Su [23] presented an improved iterative demosaicing algorithm using weighted–edge and color-difference interpolations. In [11], Lai and Liaw presented an modified mean–removed vector quantization algorithm to improve the quality image performance of the previous algorithm by Pei and Tam [20]. Based on the spatial correlation and the edge–directional information of the neighboring pixels, Lee et al. [12] presented a weighted edge–sensing demosaicing algorithm. Currently, based on the Nth–order directional finite derivative of spectral–special correlation, empirically $N = 11$, Tsai and Song [27] presented a lined–based demosaicing algorithm and their algorithm has the best image quality performance in average when compared to all other existing demosaicing algorithms. The digital zooming issue on CFA model has been discussed in [3, 16, 17, 18, 29].

After examining most of these previously published edge–sensing demosaicing algorithms, we find that the quality of demosaiced images is heavily dependent on the captured gradient/edge information from input mosaic images, but usually the captured gra-
dient/edge information on mosaic images is not so accurate. Since each pixel in the mosaic image only has one color channel, the previous color edge detectors [2, 5, 6, 24, 25, 26] can not work well on mosaic images directly. The motivations of this research are two–fold: (1) proposing a new approach to capture more accurate color gradient/edge information on mosaic images directly and (2) developing a new high–quality edge–sensing demosaicing algorithm.

In this paper, without demosaicing processing, a new approach to capture more accurate color gradient/edge information on mosaic images directly is first proposed. Next, based on spectral–spatial correlation [27], a novel adaptive heterogeneity–projection for mosaic images is presented. Combining the captured color gradient/edge information and the adaptive heterogeneity–projection values, we present a new high–quality edge–sensing demosaicing algorithm. Based on twenty–four popular testing mosaic images, experimental results demonstrated that our proposed new demosaicing algorithm has the best quality performance when compared to current published four demosaicing algorithms by Pei and Tam [20], Lu and Tan [14], Dubois [4], and Tsai and Song [27].

The remainder of this paper is organized as follows. In Section II, a new approach to capture more accurate color gradient/edge information on mosaic images is presented. In Section III, a novel adaptive heterogeneity–projection for mosaic images is first presented and then combining the captured color gradient/edge information and the adaptive heterogeneity–projection values, our proposed new edge–sensing demosaicing algorithm is presented. In Section IV, some experimental results are carried out to illustrate the quality advantage of our proposed demosaicing algorithm when compared to four previous demosaicing algorithms. Finally, some conclusions are addressed in Section V.

II. NEW APPROACH TO CAPTURE MORE ACCURATE COLOR GRADIENT INFORMATION ON MOSAIC IMAGES

This section presents a new approach to capture more accurate color gradient in-
formation on mosaic images directly. Although previously, several efficient demosaicing algorithms have been developed [13, 14, 20, 21, 28], we plug the bilinear interpolation technique [21] into our proposed approach on mosaic images due to its simplicity and computational efficiency. In what follows, we first survey the bilinear interpolation technique for demosaicing mosaic images. Next, our proposed new approach to capture more accurate color gradient information on mosaic images directly is presented.

2.1. Bilinear interpolation–based demosaicing process

Based on bilinear interpolation estimation, the other two unknown color values of each pixel on the mosaic image can be estimated by considering the color information of its adjacent pixels. For a demosaiced full color image, suppose the R, G, and B color values of the pixel at position \((i, j)\) are denoted by \(I_{dm}^r(i, j)\), \(I_{dm}^g(i, j)\), and \(I_{dm}^b(i, j)\), respectively; for a mosaic image, the R, G, and B color pixels located at position \((i, j)\) are denoted by \(I_{mo}^r(i, j)\), \(I_{mo}^g(i, j)\), and \(I_{mo}^b(i, j)\), respectively, where \(i\) denotes the location on the vertical axis and \(j\) denotes the location on the horizontal axis (see Fig. 1). The full color image can be recovered from the mosaic image by the following rules:

Case 1: Compensate red and blue colors for the pixel at position \((i, j)\), \(i, j \in \text{odd}\), with green color by computing

\[
\begin{align*}
I_{dm}^r(i, j) &= \frac{1}{2}[I_{mo}^r(i, j - 1) + I_{mo}^r(i, j + 1)] \\
I_{dm}^g(i, j) &= I_{mo}^g(i, j) \\
I_{dm}^b(i, j) &= \frac{1}{2}[I_{mo}^b(i - 1, j) + I_{mo}^b(i + 1, j)].
\end{align*}
\]  

Case 2: Compensate red and blue colors for the pixel at position \((i, j)\), \(i, j \in \text{even}\), with green color by computing

\[
\begin{align*}
I_{dm}^r(i, j) &= \frac{1}{2}[I_{mo}^r(i - 1, j) + I_{mo}^r(i + 1, j)] \\
I_{dm}^g(i, j) &= I_{mo}^g(i, j) \\
I_{dm}^b(i, j) &= \frac{1}{2}[I_{mo}^b(i, j - 1) + I_{mo}^b(i, j + 1)].
\end{align*}
\]
**Case 3:** Compensate green and blue colors for the pixel at position \((i, j)\), \(i \in \text{odd} \) and \( j \in \text{even} \), with red color by computing

\[
I_{dm}^{r}(i, j) = I_{mo}^{r}(i, j) \\
I_{dm}^{g}(i, j) = \frac{1}{4} \left[ I_{mo}^{g}(i-1, j-1) + I_{mo}^{g}(i+1, j+1) + I_{mo}^{g}(i, j-1) + I_{mo}^{g}(i, j+1) \right] \\
I_{dm}^{b}(i, j) = \frac{1}{4} \left[ I_{mo}^{b}(i-1, j-1) + I_{mo}^{b}(i+1, j+1) + I_{mo}^{b}(i, j-1) + I_{mo}^{b}(i, j+1) \right].
\]

**Case 4:** Compensate red and green colors for the pixel at position \((i, j)\), \(i \in \text{even} \) and \( j \in \text{odd} \), with blue color by computing

\[
I_{dm}^{r}(i, j) = \frac{1}{4} \left[ I_{mo}^{r}(i-1, j-1) + I_{mo}^{r}(i-1, j+1) + I_{mo}^{r}(i+1, j-1) + I_{mo}^{r}(i+1, j+1) \right] \\
I_{dm}^{g}(i, j) = \frac{1}{4} \left[ I_{mo}^{g}(i-1, j-1) + I_{mo}^{g}(i+1, j+1) + I_{mo}^{g}(i, j-1) + I_{mo}^{g}(i, j+1) \right] \\
I_{dm}^{b}(i, j) = I_{mo}^{b}(i, j).
\]

The data dependence of compensation for the above four cases is depicted in Fig. 4.

![Fig. 4: The data dependence in the bilinear interpolation estimation. (a) Compensating red and blue colors for case 1. (b) Compensating red and blue colors for case 2. (c) Compensating blue and green colors for case 3. (d) Compensating red and green colors for case 4.](image)

### 2.2. Capturing the full color gradient information on mosaic images directly

In this subsection, our proposed new approach to capture more accurate color gradient information on mosaic images directly is presented. The captured gradient information will be used in our proposed new edge–sensing demosaicing algorithm in Section III and it would lead to a high–quality advantage.
Before presenting the proposed new approach, for completeness, we first introduce how to run Sobel operator [7] on full color images. In [30], Sobel operator for gray images is extended for capturing the gradient information from the gray image domain to the full color image domain. The four masks used in $3 \times 3$ Sobel operator for full color image are illustrated in Fig. 5. Given a demosaiced full color image, it has been known that the

$$
\begin{array}{ccc}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1 \\
\end{array} & \begin{array}{ccc}
-1 & -2 & -1 \\
0 & 0 & 0 \\
-2 & -1 & 0 \\
\end{array} & \begin{array}{ccc}
0 & 1 & 2 \\
-1 & 0 & 1 \\
0 & -1 & -2 \\
\end{array} & \begin{array}{ccc}
2 & 1 & 0 \\
1 & 0 & -1 \\
0 & -1 & -2 \\
\end{array}
$$

(a) (b) (c) (d)

Fig. 5: The $3 \times 3$ Sobel operator. (a) The horizontal mask. (b) The vertical mask. (c) The $\frac{\pi}{4}$–diagonal mask. (d) The $-\frac{\pi}{4}$–diagonal mask.

$R$, $G$, and $B$ color values of the pixel at position $(i, j)$ are denoted by $I_{rm}^{dm}(i, j)$, $I_{gm}^{dm}(i, j)$, and $I_{bm}^{dm}(i, j)$, respectively. After running Sobel horizontal, vertical, $\frac{\pi}{4}$–diagonal, and $-\frac{\pi}{4}$–diagonal masks as shown in Fig. 5 on the $3 \times 3$ full color subimage centered at position $(i, j)$, the horizontal response $\Delta I_{H}^{dm}(i, j)$, the vertical response $\Delta I_{V}^{dm}(i, j)$, the $\frac{\pi}{4}$–diagonal response $\Delta I_{\frac{\pi}{4}}^{dm}(i, j)$, and the $-\frac{\pi}{4}$–diagonal response $\Delta I_{-\frac{\pi}{4}}^{dm}(i, j)$ are calculated by

$$
\begin{align*}
\Delta I_{H}^{dm}(i, j) &= 0.299 \Delta I_{H,r}^{dm}(i, j) + 0.587 \Delta I_{H,g}^{dm}(i, j) + 0.114 \Delta I_{H,b}^{dm}(i, j) \\
\Delta I_{V}^{dm}(i, j) &= 0.299 \Delta I_{V,r}^{dm}(i, j) + 0.587 \Delta I_{V,g}^{dm}(i, j) + 0.114 \Delta I_{V,b}^{dm}(i, j) \\
\Delta I_{\frac{\pi}{4}}^{dm}(i, j) &= 0.299 \Delta I_{\frac{\pi}{4},r}^{dm}(i, j) + 0.587 \Delta I_{\frac{\pi}{4},g}^{dm}(i, j) + 0.114 \Delta I_{\frac{\pi}{4},b}^{dm}(i, j) \\
\Delta I_{-\frac{\pi}{4}}^{dm}(i, j) &= 0.299 \Delta I_{-\frac{\pi}{4},r}^{dm}(i, j) + 0.587 \Delta I_{-\frac{\pi}{4},g}^{dm}(i, j) + 0.114 \Delta I_{-\frac{\pi}{4},b}^{dm}(i, j)
\end{align*}
$$

(5)
where for $C \in \{r, g, b\}$,

\[
\Delta I_{\text{dm}}^{H.C}(i, j) = \begin{cases} 
[I_{\text{dm}}^{C}(i - 1, j + 1) + 2I_{\text{dm}}^{C}(i, j + 1) + I_{\text{dm}}^{C}(i + 1, j + 1)] \\
- [I_{\text{dm}}^{C}(i - 1, j - 1) + 2I_{\text{dm}}^{C}(i, j - 1) + I_{\text{dm}}^{C}(i + 1, j - 1)]
\end{cases}
\]

\[
\Delta I_{\text{dm}}^{V.C}(i, j) = \begin{cases} 
[I_{\text{dm}}^{C}(i + 1, j - 1) + 2I_{\text{dm}}^{C}(i + 1, j) + I_{\text{dm}}^{C}(i + 1, j + 1)] \\
- [I_{\text{dm}}^{C}(i - 1, j - 1) + 2I_{\text{dm}}^{C}(i - 1, j) + I_{\text{dm}}^{C}(i - 1, j + 1)]
\end{cases}
\]

\[
\Delta I_{\text{dm}}^{\pi r.C}(i, j) = \begin{cases} 
[I_{\text{dm}}^{C}(i - 1, j) + 2I_{\text{dm}}^{C}(i - 1, j) + I_{\text{dm}}^{C}(i - 1, j + 1)] \\
- [I_{\text{dm}}^{C}(i + 1, j) + 2I_{\text{dm}}^{C}(i + 1, j) + I_{\text{dm}}^{C}(i + 1, j + 1)]
\end{cases}
\]

\[
\Delta I_{\text{dm}}^{\pi b.C}(i, j) = \begin{cases} 
[I_{\text{dm}}^{C}(i - 1, j) + 2I_{\text{dm}}^{C}(i - 1, j) + I_{\text{dm}}^{C}(i - 1, j + 1)] \\
- [I_{\text{dm}}^{C}(i + 1, j) + 2I_{\text{dm}}^{C}(i + 1, j) + I_{\text{dm}}^{C}(i + 1, j + 1)]
\end{cases}
\]
Fig. 6: For $i + j \in \text{even}$, the four SI–based masks for $G$ channel. (a) The horizontal SI–based mask. (b) The vertical SI–based mask. (c) The $\frac{\pi}{4}$–diagonal SI–based mask. (d) The $-\frac{\pi}{4}$–diagonal SI–based mask.

Fig. 7: For $i + j \in \text{odd}$, the four SI–based masks for $G$ channel. (a) The horizontal SI–based mask. (b) The vertical SI–based mask. (c) The $\frac{\pi}{4}$–diagonal SI–based mask. (d) The $-\frac{\pi}{4}$–diagonal SI–based mask.

are denoted by $\Delta I_{dm}^{\frac{\pi}{4},r}(i, j)$, $\Delta I_{dm}^{\frac{\pi}{4},g}(i, j)$, and $\Delta I_{dm}^{\frac{\pi}{4},b}(i, j)$, respectively. By Eq. (5), combining the three horizontal responses, we have the total horizontal response $\Delta I_{dm}^{H}(i, j)$. Similarly, we have the total vertical response $\Delta I_{dm}^{V}(i, j)$, the total $\frac{\pi}{4}$–diagonal response $\Delta I_{dm}^{\frac{\pi}{4}}(i, j)$, and the total $-\frac{\pi}{4}$–diagonal response $\Delta I_{dm}^{-\frac{\pi}{4}}(i, j)$.

III. THE PROPOSED NEW EDGE–SENSING DEMOSAICING ALGORITHM

Based on the accurate gradient information obtained in last section, this section presents our proposed new high–quality edge–sensing demosaicing algorithm. Our proposed novel adaptive heterogeneity–projection for mosaic images is first presented in subsection 3.1 to capture more accurate horizontal and vertical heterogeneity–projection value. Based on the two obtained heterogeneity–projection values and those obtained gra-
For $i \in \text{odd}, j \in \text{even}$ ($i \in \text{even}, j \in \text{odd}$), the four SI–based masks for $R$ ($B$) channel. (a) The horizontal SI–based mask. (b) The vertical SI–based mask. (c) The $\frac{\pi}{4}$–diagonal SI–based mask. (d) The $-\frac{\pi}{4}$–diagonal SI–based mask.

For $i \in \text{even}, j \in \text{odd}$ ($i \in \text{odd}, j \in \text{even}$), the four SI–based masks for $R$ ($B$) channel. (a) The horizontal SI–based mask. (b) The vertical SI–based mask. (c) The $\frac{\pi}{4}$–diagonal SI–based mask. (d) The $-\frac{\pi}{4}$–diagonal SI–based mask.

3.1. Novel adaptive heterogeneity–projection for mosaic images

In this subsection, a novel adaptive heterogeneity–projection for mosaic images is presented. Given an original mosaic image $I_{mo}$, its horizontal heterogeneity–projection map $HP_{H-map}$ and vertical heterogeneity–projection map $HP_{V-map}$ can be obtained by running the following two 1–D Laplacian operations [27]:

$$HP_{H-map} = |I_{mo} \ast (F^{N\times(N-3)T(N-3)\times1})^t|$$

$$HP_{V-map} = |I_{mo} \ast (F^{N\times(N-3)T(N-3)\times1})|$$
where \( N (=11 \text{ empirically [27]}) \) denotes the vector length (or the 1-D mask size); \( \mathbf{F}^{N \times (N-3)} = [1 \quad -1 \quad -1 \quad 1]^t \) denotes a \( N \times (N-3) \) coefficient matrix; \( \mathbf{T}^{(N-3) \times 1} = \prod_{x=1}^{N-4} [1 \quad -1]^t \) denotes a \((N-3) \times 1 \) coefficient vector; \( \mathbf{I}^{M \times M} \) denotes an \( M \times M \) identity matrix; the symbol “\(*\)” denotes the 2-D convolution operator; \(| \cdot |\) denotes the absolute value operator and the operator “\(^t\)” denotes the transpose operator.

In what follows, our approach can determine the suitable value of \( N \) adaptively.

Let us examine some cases of the current pixel on the mosaic image. If the surrounding region of the current pixel is homogenous, the two responses by Eq. (7) are almost the same whether a large mask size or small mask size is adopted. If there is one tiny horizontal (vertical) edge passing through the current pixel, a small mask size for \( HP_{v-map} \)


\( (HP_{H-map}) \) is enough rather than a large mask size. In [27], the adopted mask size is fixed and is set to \( N = 11 \). According to the above discussion, we now present our proposed adaptive heterogeneity–projection for mosaic images such that the used mask size is as small as possible and the computed responses are more accurate than those computed by the mask with size 11. Experimental results reveal that our proposed adaptive heterogeneity–projection approach under proper horizontal and vertical mask sizes has the computation–saving and more accurate advantages.

We utilize the horizontal (vertical) spectral–spatial correlation (SSC) [27] between the current pixel at location \((i, j)\) and its neighboring pixel at location \((i, j + 1) ((i+1, j))\) to determine the proper horizontal (vertical) mask size \(N_H(i, j) (N_V(i, j))\). For a horizontal SSC map, the horizontal SSC value at location \((i, j)\) can be obtained by using the following rules:

\[
S_H(i, j) = \begin{cases} 
|I^g_{mo}(i, j) - I^g_{mo}(i, j + 1)| & \text{if the current pixel is green and } i \in \text{odd}. \\
|I^r_{mo}(i, j) - I^r_{mo}(i, j + 1)| & \text{if the current pixel is red.} \\
|I^b_{mo}(i, j) - I^b_{mo}(i, j + 1)| & \text{if the current pixel is blue.}
\end{cases}
\]

Similarly, the vertical SSC value at location \((i, j)\) can be obtained by using the following rules:

\[
S_V(i, j) = \begin{cases} 
|I^g_{mo}(i, j) - I^g_{mo}(i + 1, j)| & \text{if the current pixel is green and } j \in \text{even}. \\
|I^r_{mo}(i, j) - I^r_{mo}(i + 1, j)| & \text{if the current pixel is red.} \\
|I^b_{mo}(i, j) - I^b_{mo}(i + 1, j)| & \text{if the current pixel is blue.}
\end{cases}
\]

The horizontal SSC map and the vertical SSC map of the mosaic Lighthouse image in Fig. 3 are illustrated in Fig. 12(a) and Fig. 12(b), respectively, and it is observed that the SSC values are locally constant in homogeneous regions.

Because the determination of the proper horizontal mask size \(N_H(i, j)\) is the same as that for the vertical mask size \(N_V(i, j)\), we thus only focus on the determination of horizontal mask size. Fig. 13 depicts our proposed approach to determine the proper horizontal mask size \(N_H(i, j)\). Centered at location \((i, j)\), Fig. 13 illustrates a \(1 \times 11\) row data extracted from the horizontal SSC map. The procedure to determine the proper horizontal mask size \(N_H(i, j)\) consists of the following three steps.
Fig. 12: The two SSC maps of the mosaic Lighthouse image. (a) The horizontal SSC map. (b) The vertical SSC map. (In order to show the images more clear, the gray value 80 is used to represent the value 0)

Fig. 13: The depiction of our proposed approach to determine the proper horizontal mask size $N_H(i, j)$.

<table>
<thead>
<tr>
<th>$S_H(i,j-5)$</th>
<th>$S_H(i,j-4)$</th>
<th>$S_H(i,j-3)$</th>
<th>$S_H(i,j-2)$</th>
<th>$S_H(i,j-1)$</th>
<th>$S_H(i,j)$</th>
<th>$S_H(i,j+1)$</th>
<th>$S_H(i,j+2)$</th>
<th>$S_H(i,j+3)$</th>
<th>$S_H(i,j+4)$</th>
<th>$S_H(i,j+5)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>←</td>
<td>←</td>
<td>←</td>
<td></td>
<td>←</td>
<td>←</td>
<td></td>
<td>←</td>
<td>←</td>
<td>←</td>
<td>←</td>
</tr>
</tbody>
</table>

**Step 1:** Initially, we compute the two SSC values $S_H(i,j-2)$ and $S_H(i,j+2)$, and temporarily set $N_H(i,j) = 5$, $x_{left} = j - 2$, and $x_{right} = j + 2$.

**Step 2:** If the condition $N_H(i,j) = 11$ holds, output $N_H(i,j)$ and stop. Here, the maximal horizontal mask size is bounded by eleven. Otherwise, go to Step 3.

**Step 3:** Examine whether the neighboring horizontal SSC values are locally chosen by using the following testing condition:

$$Max(DS_H^{left} , DS_H^{right}) < T_h$$

where $DS_H^{left} = |S_H(i, x_{left}) - S_H(i, x_{left} - 1)| + |S_H(i, x_{left}) - S_H(i, x_{left} + 1)|$, $DS_H^{right} = |S_H(i, x_{right}) - S_H(i, x_{right} - 1)| + |S_H(i, x_{right}) - S_H(i, x_{right} + 1)|$, and the threshold is set to $T_h = 8$ empirically. If the above testing condition
holds, output the value of $N_H(i, j)$ as the proper horizontal mask size and stop. Otherwise, perform the operation $N_H(i, j) = N_H(i, j) + 2$, $x_{left} = x_{left} - 1$, and $x_{right} = x_{right} + 1$, and then go to Step 2.

The above three-step procedure can be easily applied to determine the proper horizontal mask size. In order to normalize the masks for different sizes, the normal factor $\frac{1}{Q(N)}$ is used to normalize the coefficients of the mask. In other words, instead of using $F^{N \times (N-3)}(N-3 \times 1)$, we use $\frac{1}{Q(N)}(F^{N \times (N-3)}(N-3 \times 1))$ to obtain the normalized heterogeneity–projection map. The value of $Q(N)$ is defined as the sum of the positive coefficients in the mask. For example, if $N = 5$, the mask $F^{5 \times (5-3)}(N-3 \times 1) = [1 -2 0 2 -1]^t$, can be normalized to $\frac{F^{5 \times 2 \times 2 \times 1}}{Q(5)} = [1 -2 0 2 -1]^t$.

Finally, in order to reduce the estimation error, we use the local mean to tune the heterogeneity–projection maps. For the $HP_{H\text{–map}}$ ($HP_{V\text{–map}}$), the horizontal (vertical) heterogeneity–projection value at location $(i, j)$ is denoted by $HP_H(i, j)$ ($HP_V(i, j)$). The tuned horizontal heterogeneity–projection value $HP'_H(i, j)$ and the tuned vertical heterogeneity–projection value $HP'_V(i, j)$ can be computed by the following operations:

$$HP'_H(i, j) = \frac{1}{6} \sum_{k=-2}^{2} \delta_k HP_H(i, j + k)$$

$$HP'_V(i, j) = \frac{1}{6} \sum_{k=-2}^{2} \delta_k HP_V(i + k, j)$$

where $\delta_k = 2$ if $k = 0$; $\delta_k = 1$, otherwise.

After performing the above adaptive heterogeneity–projection for mosaic images, the values $HP'_H(i, j)$ and $HP'_H(i, j)$ can be obtained for each mosaic pixel. In next two subsections, the six responses $HP'_H(i, j)$, $HP'_H(i, j)$, $\Delta I_{dmH}(i, j)$, $\Delta I_{dmV}(i, j)$, $\Delta I_{dm}(i, j)$, and $\Delta I_{dm}^{-\pi}(i, j)$ will be used in our proposed new edge–sensing interpolation estimation for demosaicing mosaic images.

3.2. The edge–sensing interpolation estimation for $G$ channel
In this subsection, we present the proposed high–quality edge–sensing interpolation estimation for $G$ channel. For exposition, let us take a $7 \times 7$ mosaic subimage as shown in Fig. 14 to explain how to estimate the $G$ channel value located at the center position of Fig. 14. Before performing the interpolation estimation for $G$ channel, assume the gradient information of the current mosaic pixel at position $(i, j)$ and the eight neighboring pixels at positions $\Omega_n = \{(x, y)|(x, y) = (i \pm 1, j), (i \pm 2, j), (i, j \pm 1), (i, j \pm 2)\}$ have been captured by using our proposed method described in last section, and the captured nine horizontal gradient magnitudes and nine vertical gradient magnitudes are denoted by $\Delta I_{dm}^H(\ast, \ast)$ and $\Delta I_{dm}^V(\ast, \ast)$, respectively.

According to the tuned horizontal heterogeneity–projection value $HP_H'(i, j)$ and the tuned vertical heterogeneity–projection value $HP_V'(i, j)$ of the current mosaic pixel at position $(i, j)$, the interpolation estimation scheme in our proposed edge–sensing demosaicing algorithm considers three cases, namely (1) horizontal variation as shown in Fig. 15(a), (2) vertical variation as shown in Fig. 15(b), and (3) the other variations as shown in Fig. 15(c). The arrows in Fig. 15 denote the data dependence.

In addition, in order to estimate $I_{dm}^g(i, j)$ more accurately from its four neighbors,
four proper weights in terms of gradient information are assigned to corresponding four
spectral-correlation terms in the interpolation estimation. Considering the neighboring
pixel located at location \((i-1, j)\), if there is (is no) a horizontal edge passing through it, i.e.
the vertical gradient magnitudes of the pixel at location \((i-1, j)\) is large (small), the color
difference assumption [14, 20] reveals that the green component of this pixel makes less
(more) contribution to the estimation of green component for the current pixel at location
\((i, j)\). On the other hand, if the gradient magnitudes of the pixels at location \((i-2, j)\)
and \((i, j)\) are large (small), the pixel at location \((i-1, j)\) is located in a nonhomogeneous
(homogeneous) region and we claim that the pixel at location \((i-1, j)\) makes less (more)
green contribution to the estimation of green component for the current mosaic pixel at
location \((i, j)\). Combining the above analysis of gradient and direction effects, the weight of
pixel at location \((i-1, j)\) can be given by
\[
{w_g}(i - 1, j) = \frac{1}{1 + \beta [\delta_{1}^V(i-1, j) + \delta_{1}^H(i, j)]}.
\]

Following the similar discussion, the weights of the four neighbors of the current pixel
are expressed by
\[
{w_g}(i - 1, j) = \frac{1}{1 + \beta [\delta_{1}^V(i-1, j)]}, \quad {w_g}(i + 1, j) = \frac{1}{1 + \beta [\delta_{1}^V(i+1, j)]},
\]
\[
{w_g}(i, j - 1) = \frac{1}{1 + \beta [\delta_{1}^H(i, j-1)]}, \quad {w_g}(i, j + 1) = \frac{1}{1 + \beta [\delta_{1}^H(i, j+1)]}
\]
where \(\delta_k^t\) = 2 if \(k = 1\); \(\delta_k^t = 1\), otherwise; the parameter \(\beta\) is set to \(\beta = 0.7\) empirically.

According to the above description, the value of \(I^g_{dm}(i, j)\) of the current pixel at location
\((i, j)\) can be estimated by the following rules:
\[
I^g_{dm}(i, j) = I^b_{mo}(i, j) + \sum_{(x,y) \in \Omega_g} w_g(x, y) D_g(x, y)
\]
\[
\Omega_g = \begin{cases}
\{(i \pm 1, j)\} & \text{if } HP^v(i, j) < \alpha HP^h(i, j) \text{ (horizontal variation).} \\
\{(i, j \pm 1)\} & \text{if } HP^h(i, j) < \alpha HP^v(i, j) \text{ (vertical variation).} \\
\{(i \pm 1, j), (i, j \pm 1)\} & \text{otherwise (other variations).}
\end{cases}
\]
where for \((x_1, y_1) \in \{(i \pm 1, j)\}\), \(D_g(x_1, y_1) = I^g_{mo}(x_1, y_1) - \frac{I^b_{mo}(x_1+1, y_1)+I^b_{mo}(x_1-1, y_1)}{2}\); for
\((x_2, y_2) \in \{(i, j \pm 1)\}\), \(D_g(x_2, y_2) = I^g_{mo}(x_2, y_2) - \frac{I^b_{mo}(x_2+1, y_2)+I^b_{mo}(x_2-1, y_2)}{2}\); the parameter \(\alpha\)
is set to \(\alpha = 0.5\).

After performing the above edge-sensing interpolation estimation for \(G\) channel, the
\(G\) channel of the demosaiced image is fully populated. In next subsection, the fully
populated \(G\) channel of the image will be used to assist the interpolation of \(R\) and \(B\)
channels.

Fig. 15: The data dependence of our proposed interpolation estimation for $G$ channel. (a) Horizontal variation (vertical edge). (b) Vertical variation (horizontal edge). (c) The other variations.

3.3. The edge–sensing interpolation estimation for $R$ and $B$ channels

Because the number of $R$ pixels or $B$ pixels is less than $G$ in the mosaic image, the interpolation estimation for $R$ and $B$ channels should be partitioned into two steps: (1) estimating the red (blue) values at blue (red) pixels and then (2) recovering the remains of the red and blue values at green pixels. Because the interpolation estimation for $R$ channel is the same as it for $B$ channel, we thus only present it for $R$ channel. In our proposed interpolation estimation for $R$ and $B$ channels, the fully populated $G$ channel is used to assist the interpolation of $R$ and $B$ channels. For convenience, we still use Fig. 14 to explain how to estimate the $R$ channel value for the current pixel at position $(i, j)$.

Similar to the interpolation estimation for $G$ channel, assume the gradient information of the current mosaic pixel at position $(i, j)$ and the eight pixels at positions $\Omega'_n = \{(x, y) | (x, y) = (i \pm 1, j \pm 1), (i \pm 2, j \pm 2)\}$, respectively, have been captured by using our proposed method. By the same arguments, in order to estimate $I_{dm}(i, j)$ more accurately from its four red neighbors, four proper weights in terms of gradient/direction information are assigned to the corresponding four spectral–correlation terms in the interpolation estimation. For estimating $I_{dm}(i, j)$, we consider four diagonal variations of
the mosaic pixel at position \((i, j)\) to determine the four weights which will be used in the proposed edge–sensing estimation of \(I_{dm}^r(i, j)\). Considering the neighboring pixel at location \((i - 1, j - 1)\), if there is (is no) a \(\frac{\pi}{4}\)–diagonal edge passing through it, i.e. the \(\frac{\pi}{4}\)–diagonal gradient magnitude of the pixel at location \((i - 1, j - 1)\) is large (small), and then it claims that the red component of this pixel makes less (more) contribution to the estimation of red component for the current pixel at location \((i, j)\). On the contrary, if the gradient magnitudes of the pixel at location \((i - 1, j - 1)\) and \((i - 2, j - 2)\) are large (small), i.e. the pixel at location \((i - 1, j - 1)\) is inside nonhomogeneous (homogeneous) region, the color difference assumption indicates that the green component of this pixel makes less (more) contribution to the estimation of red component for the current pixel at location \((i, j)\). Following the above analysis, the weights of the four diagonal red neighbors of the current pixel can be expressed by

\[
w_r(i - 1, j - 1) = \frac{1}{1 + \beta \left[ \sum_{k=0}^{2} \delta'_k \Delta I_{dm}^r(i-k,j-k) \right]}
\]

and

\[
w_r(i + 1, j + 1) = \frac{1}{1 + \beta \left[ \sum_{k=0}^{2} \delta'_k \Delta I_{dm}^r(i+k,j+k) \right]}
\]

where \(\delta'_k = 2\) if \(k = 1\); \(\delta'_k = 1\), otherwise; the parameter set to \(\beta = 0.5\) empirically. Based on the four weights obtained above and the color difference concept, the demosaiced full red color for the blue pixel in the mosaic image, \(r_{f}^{d}(i, j)\), can be estimated by

\[
I_{dm}^r(i, j) = I_{dm}^g(i, j) + \frac{\sum_{(x,y) \in \Omega_r} w_r(x, y) D_r(x, y)}{\sum_{(x,y) \in \Omega_r} w_r(x, y)}
\]

where for \((x, y) \in \{(i \pm 1, j \pm 1)\}\), \(D_r(x, y) = I_{dn}^r(x, y) - I_{dm}^r(x, y)\).

After describing how to estimate demosaiced full red colors for those blue pixels in the mosaic image, we now introduce how to estimate full red colors for those green pixels. Fig. 16 illustrates the pattern of \(R\) channel. Referring to Fig. 16, the full red color for the green pixel, \(I_{dn}^r(i, j)\), can be estimated by the following rules:

\[
\Omega_r' = \begin{cases} 
\{(i \pm 1, j)\} & \text{if } HP_{V}^{r}(i, j) < \alpha HP_{H}^{r}(i, j) \text{ (horizontal variation).} \\
\{(i, j \pm 1)\} & \text{if } HP_{H}^{r}(i, j) < \alpha HP_{V}^{r}(i, j) \text{ (vertical variation).} \\
\{(i \pm 1, j), (i, j \pm 1)\} & \text{otherwise (other variations).}
\end{cases}
\]
where for \((x, y) \in \{(i \pm 1, j), (i, j \pm 1)\}\), we perform \(D_r(x, y) = I^r_{mo}(x, y) - I^g_{dm}(x, y)\). If \(k = 1\), we set \(\delta'_k = 2\); otherwise, we set \(\delta'_k = 1\). We further perform \(w_r(i - 1, j) = \frac{1}{1 + \beta \sum_{k=0}^{2} \delta'_k \Delta I^r_{dm}(i-k,j)}\), \(w_r(i + 1, j) = \frac{1}{1 + \beta \sum_{k=0}^{2} \delta'_k \Delta I^r_{dm}(i+k,j)}\), \(w_r(i, j - 1) = \frac{1}{1 + \beta \sum_{k=0}^{2} \delta'_k \Delta I^g_{dm}(i,j-k)}\), and \(w_r(i, j + 1) = \frac{1}{1 + \beta \sum_{k=0}^{2} \delta'_k \Delta I^g_{dm}(i,j+k)}\). Empirically, the parameter \(\beta\) is set to \(\beta = 0.7\).

After performing the interpolation estimation, the postprocessing proposed in [15] could be used to improve the de-mosaiced image quality. After presenting our proposed new edge-sensing demosaicing algorithm, experimental results in next section will illustrate the quality advantage of our proposed new edge-sensing demosaicing algorithm.

**IV. EXPERIMENTAL RESULTS**

In this section, based on twenty-four popular testing mosaic images, some experimental results are demonstrated to show that our proposed new demosaicing algorithm has better image quality performance when compared to the previous four algorithms by Pei and Tam [20], Lu and Tan [14], Dubois [4], and Tsai and Song [27], respectively. The concerned algorithms are implemented on the IBM compatible computer with Intel Core 2 Duo CPU 1.6GHz and 1GB RAM. The operating system used is MS–Windows XP and the program developing environment is Borland C++ Builder 6.0. Our program has been uploaded in [32].

Fig. 17 illustrates the twenty-four testing images from Kodak PhotoCD. In our ex-
periments, the twenty–four testing images shown in Fig. 17 are first down–sampled to obtain the mosaic images. Furthermore, the boundaries of the image are dealt with using the mirroring method. Here, we adopt two objective image quality measures, color peak signal–to–noise ratio (CPSNR) and S-CIELAB $\Delta E_{ab}^*$ metric [10, 14], and one subjective image quality measure, color artifacts, to justify the better quality performance of our proposed novel demosaicing algorithm. The CPSNR of a color image with size $X \times Y$ is defined by

$$\text{CPSNR} = 10 \log_{10} \frac{255^2}{3XY \sum_{i=0}^{X-1} \sum_{j=0}^{Y-1} \sum_{c \in C} [I_{\text{ori}}^c(i,j) - I_{\text{dm}}^c(i,j)]^2}, \quad C = \{r, g, b\}$$

where $I_{\text{ori}}^r(i,j)$, $I_{\text{ori}}^g(i,j)$, and $I_{\text{ori}}^b(i,j)$ denote the three color components of the pixel at location $(i,j)$ in the original full color image; $I_{\text{dm}}^r(i,j)$, $I_{\text{dm}}^g(i,j)$, and $I_{\text{dm}}^b(i,j)$ denote the three color components of the pixel at location $(i,j)$ in the demosaiced image. The S-CIELAB $\Delta E_{ab}^*$ of a color image with size $X \times Y$ is defined by

$$\Delta E_{ab}^* = \frac{1}{XY} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left\{ \sum_{c \in \mathcal{L}} \left[ \sqrt{\sum_{r \in \mathcal{R}} \left[ \text{LAB}_{\text{ori}}^c(i,j) - \text{LAB}_{\text{dm}}^c(i,j) \right]^2} \right] \right\}, \quad \mathcal{L} = \{L, a, b\}$$

where $\text{LAB}_{\text{ori}}^L(i,j)$, $\text{LAB}_{\text{ori}}^a(i,j)$, and $\text{LAB}_{\text{ori}}^b(i,j)$ denote the three CIELAB color components of the pixel at location $(i,j)$ in the original full color image; $\text{LAB}_{\text{dm}}^L(i,j)$, $\text{LAB}_{\text{dm}}^a(i,j)$, and $\text{LAB}_{\text{dm}}^b(i,j)$, and
and $L A B_{dn}^b(i,j)$ denote the three CIELAB color components of the pixel at location $(i,j)$ in the demosaiced image.

Based on twenty-four testing mosaic images, Table 1 and Table 2 demonstrate the demosaiced image quality comparison in terms of CPSNR and S-CIELAB $\Delta E_{ab}^*$ among our proposed algorithm and the other four existing well known demosaicing algorithms, respectively. In Table 1 and Table 2, the entries with the best CPSNR or the least $\Delta E_{ab}^*$ are highlighted by bold black. It is observed that in average, our proposed demosaicing algorithm has the best demosaiced image quality in terms of CPSNR and $\Delta E_{ab}^*$. Table 1 and Table 2 also reveal that the quality performance of our proposed demosaicing algorithm is rather stable for different kinds of testing images.

Next, we adopt the subjective image visual measure, color artifacts, to demonstrate the quality advantage of our proposed novel demosaicing algorithm. After performing the demosaicing processing, some degree of color artifacts may happen on edges or textures of the demosaiced image. We first take the magnified subimages cut from the testing image No. 19 as shown in Fig. 18 to compare the visual effect among the concerned five algorithms. Figs. 18(a)–(f) illustrate the six magnified subimages cut from the original testing image No. 19, the demosaiced image obtained by Pei and Tam’s demosaicing algorithm, the one obtained by Lu and Tan’s demosaicing algorithm, the one obtained by Dubois’ demosaicing algorithm, the one obtained by Tsai and Song’s demosaicing algorithm, and the one obtained by our proposed demosaicing algorithm, respectively. Comparing the visual effect between the original full color image and the one in Figs. 18(b)–(f), it is observed that our proposed demosaicing algorithm creates less color artifacts when compared to the other four demosaicing algorithms. Then, we take the magnified subimages cut from the testing image No. 8 to depict the visual comparison. Figs. 19(a)–(f) illustrate the magnified subimages cut from the original full color testing image No. 8 and the five demosaiced images. From visual comparison, it is observed that our proposed demosaicing algorithm produces less color artifacts when compared to the other
Table 1: CPSNR quality comparison for twenty four testing images.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Image01</td>
<td>36.1191</td>
<td>39.1344</td>
<td>38.0169</td>
<td>39.1284</td>
<td>39.8907</td>
</tr>
<tr>
<td>Image03</td>
<td>40.7687</td>
<td>41.6042</td>
<td>41.6925</td>
<td>42.0839</td>
<td>42.3487</td>
</tr>
<tr>
<td>Image04</td>
<td>39.4017</td>
<td>39.9928</td>
<td>40.5148</td>
<td>40.0421</td>
<td>40.1687</td>
</tr>
<tr>
<td>Image05</td>
<td>36.7950</td>
<td>37.0751</td>
<td>37.8749</td>
<td>37.7793</td>
<td>37.9770</td>
</tr>
<tr>
<td>Image06</td>
<td>37.1539</td>
<td>39.0822</td>
<td>39.9591</td>
<td>40.2916</td>
<td>40.7492</td>
</tr>
<tr>
<td>Image07</td>
<td>40.5159</td>
<td>41.2677</td>
<td>41.9731</td>
<td>41.5070</td>
<td>41.6697</td>
</tr>
<tr>
<td>Image08</td>
<td>32.5782</td>
<td>36.3112</td>
<td>35.2721</td>
<td>37.0667</td>
<td>37.6053</td>
</tr>
<tr>
<td>Image09</td>
<td>40.7434</td>
<td>42.0427</td>
<td>42.1739</td>
<td>42.6826</td>
<td>42.9313</td>
</tr>
<tr>
<td>Image10</td>
<td>41.3372</td>
<td>41.6658</td>
<td>42.1659</td>
<td>41.9793</td>
<td>42.3200</td>
</tr>
<tr>
<td>Image11</td>
<td>38.2779</td>
<td>39.6385</td>
<td>39.8045</td>
<td>40.1601</td>
<td>40.5631</td>
</tr>
<tr>
<td>Image12</td>
<td>40.755</td>
<td>42.3641</td>
<td>42.9763</td>
<td>43.3003</td>
<td>43.5368</td>
</tr>
<tr>
<td>Image13</td>
<td>34.3581</td>
<td>35.7019</td>
<td>35.1269</td>
<td>35.5947</td>
<td>36.2313</td>
</tr>
<tr>
<td>Image14</td>
<td>35.4253</td>
<td>35.4617</td>
<td>36.0349</td>
<td>36.1989</td>
<td>36.2937</td>
</tr>
<tr>
<td>Image15</td>
<td>38.2933</td>
<td>38.9026</td>
<td>39.2652</td>
<td>39.0081</td>
<td>39.3298</td>
</tr>
<tr>
<td>Image16</td>
<td>39.9072</td>
<td>42.4001</td>
<td>43.7054</td>
<td>43.9256</td>
<td>44.3225</td>
</tr>
<tr>
<td>Image17</td>
<td>41.0680</td>
<td>41.6828</td>
<td>41.7058</td>
<td>41.7525</td>
<td>42.0447</td>
</tr>
<tr>
<td>Image18</td>
<td>37.0195</td>
<td>37.2864</td>
<td>37.2658</td>
<td>37.2617</td>
<td>37.6919</td>
</tr>
<tr>
<td>Image19</td>
<td>37.2838</td>
<td>40.7753</td>
<td>40.4161</td>
<td>41.3102</td>
<td>41.7566</td>
</tr>
<tr>
<td>Image20</td>
<td>40.0749</td>
<td>41.0464</td>
<td>40.3860</td>
<td>41.2631</td>
<td>41.6033</td>
</tr>
<tr>
<td>Image21</td>
<td>37.8859</td>
<td>39.7048</td>
<td>38.8571</td>
<td>39.7107</td>
<td>40.199</td>
</tr>
<tr>
<td>Image22</td>
<td>37.5141</td>
<td>38.0973</td>
<td>38.2231</td>
<td>38.1381</td>
<td>38.4420</td>
</tr>
<tr>
<td>Image23</td>
<td>41.0553</td>
<td>41.6861</td>
<td>42.1933</td>
<td>41.7760</td>
<td>42.0554</td>
</tr>
<tr>
<td>Image24</td>
<td>34.9857</td>
<td>34.7855</td>
<td>35.3381</td>
<td>34.7793</td>
<td>35.2610</td>
</tr>
</tbody>
</table>
Table 2: S–CIELAB $\Delta E_{ab}^*$ quality comparison for twenty four testing images.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Image01</td>
<td>1.7773</td>
<td>1.3748</td>
<td>1.5592</td>
<td>1.3680</td>
<td>1.2756</td>
</tr>
<tr>
<td>Image02</td>
<td>1.7874</td>
<td>1.5982</td>
<td>1.7603</td>
<td>1.5584</td>
<td>1.5304</td>
</tr>
<tr>
<td>Image03</td>
<td>1.0050</td>
<td>0.9087</td>
<td>0.9465</td>
<td>0.8897</td>
<td>0.8921</td>
</tr>
<tr>
<td>Image04</td>
<td>1.2817</td>
<td>1.1893</td>
<td>1.2177</td>
<td>1.2119</td>
<td>1.1919</td>
</tr>
<tr>
<td>Image05</td>
<td>2.2565</td>
<td>2.0580</td>
<td>2.0353</td>
<td>1.9745</td>
<td>1.9369</td>
</tr>
<tr>
<td>Image06</td>
<td>1.3383</td>
<td>1.1263</td>
<td>1.0926</td>
<td>1.0131</td>
<td>0.9756</td>
</tr>
<tr>
<td>Image07</td>
<td>1.2586</td>
<td>1.1369</td>
<td>1.0755</td>
<td>1.1252</td>
<td>1.1200</td>
</tr>
<tr>
<td>Image08</td>
<td>2.1716</td>
<td>1.5836</td>
<td>1.7774</td>
<td>1.4984</td>
<td>1.4290</td>
</tr>
<tr>
<td>Image09</td>
<td>0.8860</td>
<td>0.8126</td>
<td>0.8078</td>
<td>0.7796</td>
<td>0.7730</td>
</tr>
<tr>
<td>Image10</td>
<td>0.8452</td>
<td>0.8080</td>
<td>0.8170</td>
<td>0.8005</td>
<td>0.7899</td>
</tr>
<tr>
<td>Image11</td>
<td>1.5881</td>
<td>1.3177</td>
<td>1.3677</td>
<td>1.2814</td>
<td>1.2486</td>
</tr>
<tr>
<td>Image12</td>
<td>0.7489</td>
<td>0.6584</td>
<td>0.6569</td>
<td>0.6261</td>
<td>0.6204</td>
</tr>
<tr>
<td>Image13</td>
<td>2.3943</td>
<td>2.1574</td>
<td>2.4274</td>
<td>2.2369</td>
<td>2.0844</td>
</tr>
<tr>
<td>Image14</td>
<td>2.0050</td>
<td>1.8052</td>
<td>1.8755</td>
<td>1.7252</td>
<td>1.7054</td>
</tr>
<tr>
<td>Image15</td>
<td>1.4960</td>
<td>1.3966</td>
<td>1.4358</td>
<td>1.4038</td>
<td>1.3745</td>
</tr>
<tr>
<td>Image16</td>
<td>1.1752</td>
<td>0.9370</td>
<td>0.8828</td>
<td>0.8472</td>
<td>0.8293</td>
</tr>
<tr>
<td>Image17</td>
<td>1.4391</td>
<td>1.3113</td>
<td>1.3296</td>
<td>1.3269</td>
<td>1.3158</td>
</tr>
<tr>
<td>Image18</td>
<td>2.3071</td>
<td>2.2650</td>
<td>2.2634</td>
<td>2.2796</td>
<td>2.2158</td>
</tr>
<tr>
<td>Image19</td>
<td>1.4089</td>
<td>1.1580</td>
<td>1.2176</td>
<td>1.1383</td>
<td>1.0995</td>
</tr>
<tr>
<td>Image20</td>
<td>1.0446</td>
<td>0.9499</td>
<td>1.0351</td>
<td>0.9456</td>
<td>0.9205</td>
</tr>
<tr>
<td>Image21</td>
<td>1.3317</td>
<td>1.1644</td>
<td>1.2943</td>
<td>1.1681</td>
<td>1.1177</td>
</tr>
<tr>
<td>Image22</td>
<td>1.4972</td>
<td>1.4501</td>
<td>1.3643</td>
<td>1.4541</td>
<td>1.4213</td>
</tr>
<tr>
<td>Image23</td>
<td>1.0353</td>
<td>0.9759</td>
<td>0.9263</td>
<td>0.9854</td>
<td>0.9762</td>
</tr>
<tr>
<td>Image24</td>
<td>1.4904</td>
<td>1.4366</td>
<td>1.4323</td>
<td>1.4201</td>
<td>1.3715</td>
</tr>
<tr>
<td>Average</td>
<td>1.4821</td>
<td>1.3158</td>
<td>1.3583</td>
<td>1.2941</td>
<td>1.2589</td>
</tr>
</tbody>
</table>
four demosaicing algorithms.

Fig. 18: The magnified subimages cut from the testing image No. 19. (a) Original full color image. (b) The demosaiced image obtained by Pei and Tam’s algorithm. (c) The demosaiced image obtained by Lu and Tan’s algorithm. (d) The demosaiced image obtained by Dubois’ algorithm. (e) The demosaiced image obtained by Tsai and Song’s algorithm. (f) The demosaiced image obtained by our proposed algorithm.

Finally, in order to demonstrate the advantage of our proposed adaptive heterogeneity–projection, the mask–use ratio $\mathcal{R}_N$ for demosaicing a mosaic image with size $X \times Y$ is defined as

$$\mathcal{R}_N = \frac{T_N}{XY} \times 100\%$$

where $T_N$ denotes the number of times which the mask with size $N$ is adopted during the demosaicing processing. Based on twenty–four testing mosaic images, in average, the four mask–use ratios for $N$ in $\{5, 7, 9, 11\}$ are $\mathcal{R}_5 = 29.96\%$, $\mathcal{R}_7 = 5.28\%$, $\mathcal{R}_9 = 4.24\%$, and $\mathcal{R}_{11} = 60.52\%$, respectively. According to the above statistical information, $N = 11$ could be set for most cases, but for some other cases, smaller mask size could be more suitable for capturing the heterogeneity–projection values.

VI. CONCLUSION
Without demosaicing processing, this paper first proposes a new approach to capture more accurate color gradient/edge information on mosaic images directly. Next, based on spectral–spatial correlation [27], a novel adaptive heterogeneity–projection for mosaic images is presented. Combining the captured color gradient/edge information and the adaptive heterogeneity–projection values, a new edge–sensing demosaicing algorithm is presented. Some experimental results have been carried out to demonstrate the quality advantage in terms of CPSNR and S–CIELAB $\Delta E_{ab}^*$ of our proposed new demosaicing algorithm when compared to previous four algorithms by Pei and Tam, Lu and Tan, Dubois, and Tsai and Song.

References


APPENDIX I: Derivation of two quad–masks for $G$ channel.

This appendix presents the detailed derivation of the two quad–masks for $G$ channel. Refer to condition $i + j \in \text{even}$. When $i, j \in \text{even}$ or $i, j \in \text{odd}$, this condition $i + j \in \text{even}$ will be hold. Since it is not hard to verify that the compensations of $G$ channel $I_{dm}^g(*,*)$ are the same for $i, j \in \text{even}$ or $i, j \in \text{odd}$, we only consider the case $i, j \in \text{even}$. Combining Eq. (1)–Eq. (4) and Eq. (6), the compensation of $G$ channel $I_{dm}^g(*,*)$ can be achieved by using following rules:

\[
\begin{align*}
I_{dm}^g(i - 1, j - 1) &= I_{mo}^g(i - 1, j - 1) \\
I_{dm}^g(i - 1, j) &= \frac{1}{4} \left[ I_{mo}^g(i - 2, j) + I_{mo}^g(i, j) \\
&+ I_{mo}^g(i - 1, j - 1) + I_{mo}^g(i - 1, j + 1) \right] \\
I_{dm}^g(i - 1, j + 1) &= I_{mo}^g(i - 1, j + 1) \\
I_{dm}^g(i, j - 1) &= \frac{1}{4} \left[ I_{mo}^g(i - 1, j - 1) + I_{mo}^g(i + 1, j - 1) \\
&+ I_{mo}^g(i, j) \right] \\
I_{dm}^g(i, j + 1) &= \frac{1}{4} \left[ I_{mo}^g(i - 1, j + 1) + I_{mo}^g(i + 1, j + 1) \\
&+ I_{mo}^g(i, j) \right] \\
I_{dm}^g(i + 1, j - 1) &= I_{mo}^g(i + 1, j - 1) \\
I_{dm}^g(i + 1, j) &= \frac{1}{4} \left[ I_{mo}^g(i, j) + I_{mo}^g(i + 1, j) \\
&+ I_{mo}^g(i + 1, j - 1) + I_{mo}^g(i + 1, j + 1) \right] \\
I_{dm}^g(i + 1, j + 1) &= I_{mo}^g(i + 1, j + 1)
\end{align*}
\]

According to the above eight green values, the horizontal response $\Delta I_{dm}^{H,g}(i, j)$, the vertical response $\Delta I_{dm}^{V,g}(i, j)$, the $\frac{s}{4}$–diagonal response $\Delta I_{dm}^{\frac{s}{4},g}(i, j)$, and the $-\frac{s}{4}$–diagonal response $\Delta I_{dm}^{-\frac{s}{4},g}(i, j)$ are computed by

\[
\Delta I_{dm}^{H,g}(i, j) = \sum \left\{ \begin{array}{l}
I_{mo}^g(i - 1, j + 1) \\
+ \frac{1}{4} \left[ I_{mo}^g(i - 1, j + 1) + I_{mo}^g(i + 1, j + 1) \right] \\
+ I_{mo}^g(i + 1, j + 1) \\
I_{mo}^g(i - 1, j - 1) \\
- \frac{1}{4} \left[ I_{mo}^g(i - 1, j - 1) + I_{mo}^g(i + 1, j - 1) \right] \\
+ I_{mo}^g(i + 1, j - 1) \\
- \frac{s}{2} \left[ I_{mo}^g(i, j + 2) - I_{mo}^g(i, j - 2) \right] \\
+ \frac{1}{2} \left[ I_{mo}^g(i, j + 2) + I_{mo}^g(i, j - 2) \right]
\end{array} \right\}
\]

\[
= \sum \left\{ \begin{array}{l}
\frac{1}{2} [I_{mo}^g(i, j + 2) - I_{mo}^g(i, j - 2)] \\
+ \frac{1}{2} [I_{mo}^g(i, j + 2) + I_{mo}^g(i, j - 2)]
\end{array} \right\}
\]

30
\[
\Delta I_{dm}^{V,g}(i, j) = \sum \left\{ \begin{array}{l}
I_{mo}^g(i + 1, j - 1) \\
+ 2 \left[ \frac{1}{3} I_{mo}^g(i, j) + I_{mo}^g(i + 2, j) + I_{mo}^g(i + 1, j - 1) + I_{mo}^g(i + 1, j + 1) \right] \\
+ I_{mo}^g(i + 1, j + 1) \\
- \frac{1}{2} I_{mo}^g(i - 1, j - 1) \\
+ 2 \left[ \frac{1}{2} I_{mo}^g(i - 2, j) + I_{mo}^g(i, j) + I_{mo}^g(i - 1, j - 1) + I_{mo}^g(i - 1, j + 1) \right] \\
I_{mo}^g(i - 1, j + 1) \\
= \sum \left\{ \begin{array}{l}
\frac{1}{2} [I_{mo}^g(i + 2, j) - I_{mo}^g(i - 2, j)] \\
+ \frac{3}{2} \left[ I_{mo}^g(i + 1, j - 1) + I_{mo}^g(i + 1, j + 1) \right] \\
= \sum \left\{ \begin{array}{l}
\frac{1}{4} [I_{mo}^g(i + 2, j) + I_{mo}^g(i, j)] \\
+ \frac{1}{4} I_{mo}^g(i - 1, j - 1) + gm(i - 1, j + 1) \\
+ I_{mo}^g(i - 1, j + 1) + I_{mo}^g(i + 1, j + 1) \\
+ I_{mo}^g(i - 1, j - 1) + gm(i + 1, j + 1) \\
- \frac{1}{4} I_{mo}^g(i + 1, j - 1) + gm(i + 1, j + 1) \\
- \frac{1}{4} I_{mo}^g(i + 1, j - 1) + gm(i + 1, j + 1) \\
- \frac{1}{4} I_{mo}^g(i + 1, j - 1) + gm(i + 1, j + 1) \\
= \sum \left\{ \begin{array}{l}
\frac{1}{4} [I_{mo}^g(i + 2, j) + I_{mo}^g(i, j)] \\
+ \frac{1}{4} I_{mo}^g(i - 1, j - 1) + gm(i - 1, j + 1) \\
+ \frac{1}{4} I_{mo}^g(i - 1, j - 1) + gm(i - 1, j + 1) \\
+ \frac{1}{4} I_{mo}^g(i - 1, j - 1) + gm(i - 1, j + 1) \\
+ \frac{1}{4} I_{mo}^g(i - 1, j - 1) + gm(i - 1, j + 1) \\
respectively.

Besides the discussion of the case \(i + j \in \text{even} \) for calculating the four responses \( \Delta I_{dm}^{H,g} \), \( \Delta I_{dm}^{V,g} \), \( \Delta I_{dm}^{\pi,g} \), and \( \Delta I_{dm}^{\pi g} \), we now consider the case \(i + j \in \text{odd} \). By the same argument,
for the case \( i + j \in \text{odd} \), the four responses \( \Delta I_{dm}^{H,g} \), \( \Delta I_{dm}^{V,g} \), \( \Delta I_{dm}^{\pi g} \), and \( \Delta I_{dm}^{\pi^* g} \) are calculated by

\[
\Delta I_{dm}^{H,g} = \sum \left\{ \begin{array}{l}
\frac{1}{4} \left[ I_{mo}^g(i - 2,j + 1) + I_{mo}^g(i,j + 1) \\
+ 2I_{mo}^g(i,j + 1) \\
+ I_{mo}^g(i,j + 1) + I_{mo}^g(i + 2,j + 1) \\
+ I_{mo}^g(i + 1,j) + I_{mo}^g(i + 1,j + 2) \\
+ I_{mo}^g(i - 2,j - 1) + I_{mo}^g(i,j - 1) \\
+ I_{mo}^g(i - 1,j - 2) + I_{mo}^g(i - 1,j) \\
- \frac{1}{4} \left[ I_{mo}^g(i,j + 1) + I_{mo}^g(i,j - 1) \\
+ I_{mo}^g(i + 1,j - 2) + I_{mo}^g(i + 1,j) \\
\right]
\end{array} \right. 
\]

\[
\Delta I_{dm}^{V,g} = \sum \left\{ \begin{array}{l}
\frac{1}{4} \left[ I_{mo}^g(i,j - 1) + I_{mo}^g(i + 1,j) \\
+ 2I_{mo}^g(i + 1,j) \\
+ I_{mo}^g(i + 1,j - 2) + I_{mo}^g(i + 1,j) \\
+ I_{mo}^g(i,j + 1) + I_{mo}^g(i + 2,j + 1) \\
+ I_{mo}^g(i + 1,j) + I_{mo}^g(i + 1,j + 2) \\
+ I_{mo}^g(i - 2,j - 1) + I_{mo}^g(i,j - 1) \\
+ I_{mo}^g(i - 1,j - 2) + I_{mo}^g(i - 1,j) \\
- \frac{1}{4} \left[ I_{mo}^g(i + 2,j + 1) + I_{mo}^g(i,j + 1) \\
+ I_{mo}^g(i + 1,j + 1) + I_{mo}^g(i + 1,j) \\
\right]
\end{array} \right. 
\]

\[
\Delta I_{dm}^{\pi g} = \sum \left\{ \begin{array}{l}
\frac{1}{4} \left[ I_{mo}^g(i + 1,j - 1) + I_{mo}^g(i + 1,j + 1) \\
+ 2I_{mo}^g(i + 1,j + 1) \\
+ I_{mo}^g(i + 1,j) + I_{mo}^g(i + 1,j + 2) \\
+ I_{mo}^g(i,j + 1) + I_{mo}^g(i + 2,j + 1) \\
+ I_{mo}^g(i + 1,j) + I_{mo}^g(i + 1,j + 2) \\
+ I_{mo}^g(i - 2,j - 1) + I_{mo}^g(i,j - 1) \\
+ I_{mo}^g(i - 1,j - 2) + I_{mo}^g(i - 1,j) \\
- \frac{1}{4} \left[ I_{mo}^g(i + 2,j + 1) + I_{mo}^g(i,j + 1) \\
+ I_{mo}^g(i + 1,j + 1) + I_{mo}^g(i + 1,j) \\
\right]
\end{array} \right. 
\]

\[
\Delta I_{dm}^{\pi^* g} = \sum \left\{ \begin{array}{l}
\frac{1}{4} \left[ I_{mo}^g(i + 1,j - 1) + I_{mo}^g(i + 1,j + 1) \\
+ 2I_{mo}^g(i + 1,j + 1) \\
+ I_{mo}^g(i + 1,j) + I_{mo}^g(i + 1,j + 2) \\
+ I_{mo}^g(i,j + 1) + I_{mo}^g(i + 2,j + 1) \\
+ I_{mo}^g(i + 1,j) + I_{mo}^g(i + 1,j + 2) \\
+ I_{mo}^g(i - 2,j - 1) + I_{mo}^g(i,j - 1) \\
+ I_{mo}^g(i - 1,j - 2) + I_{mo}^g(i - 1,j) \\
- \frac{1}{4} \left[ I_{mo}^g(i + 2,j + 1) + I_{mo}^g(i,j + 1) \\
+ I_{mo}^g(i + 1,j + 1) + I_{mo}^g(i + 1,j) \\
\right]
\end{array} \right. 
\]
we thus only derive the masks for channel. Because the derivation of masks for \( \Delta \) channel is the same as it for \( R \) channel, we thus only derive the masks for \( R \) channel.

By the same argument as the derivation of \( G \) channel, for each case, the four responses \( \Delta R_{dm}^{r} \), \( \Delta V_{dm}^{r} \), \( \Delta R_{dm}^{g} \), and \( \Delta V_{dm}^{g} \) are calculated by

\[
\Delta I_{dm}^{r,g}(i,j) = \sum \left\{ \begin{array}{l}
I_{mo}^{g}(i-1,j) \\
+2 \left[ \frac{1}{3} \left[ I_{mo}^{g}(i-2,j+1) + I_{mo}^{g}(i,j+1) \\
+ I_{mo}^{g}(i-1,j) + I_{mo}^{g}(i-1,j+2) \right] \right] \\
+ I_{mo}^{g}(i+1,j) \\
- \left[ \frac{1}{4} \left[ I_{mo}^{g}(i,j-1) + I_{mo}^{g}(i+2,j-1) \right] \\
+ I_{mo}^{g}(i+1,j) + I_{mo}^{g}(i+1,j+2) \right] \\
\end{array} \right. \\
\]

\[
\Delta I_{dm}^{g}(i,j) = \sum \left\{ \begin{array}{l}
I_{mo}^{g}(i,j) \\
+2 \left[ \frac{1}{3} \left[ I_{mo}^{g}(i-2,j-1) + I_{mo}^{g}(i,j-1) \\
+ I_{mo}^{g}(i-1,j-2) + I_{mo}^{g}(i+1,j) \right] \right] \\
+ I_{mo}^{g}(i+1,j) \\
- \left[ \frac{1}{4} \left[ I_{mo}^{g}(i,j) + I_{mo}^{g}(i+2,j+1) \right] \\
+ I_{mo}^{g}(i+1,j) + I_{mo}^{g}(i+1,j+2) \right] \\
\end{array} \right. \\
\]

**APPENDIX II: Derivation of four quad–masks for \( R \) (or \( B \)) channel.**

This appendix presents the detailed derivation of the four quad–masks for \( R \) (or \( B \)) channel. Because the derivation of masks for \( R \) channel is the same as it for \( B \) channel, we thus only derive the masks for \( R \) channel.
Case 1: \((i \in \text{odd}, j \in \text{even})\)
\[ \Delta I_{dm}^{vr}(i, j) = \sum \left\{ \begin{array}{c} \frac{1}{2} I_r(i - 1, j - 1) + I_m(i + 1, j - 1) \\ + 2 \left[ \frac{1}{2} I_r(i - 1, j - 1) + I_m(i + 1, j - 1) \right] \\ + 2 \left[ \frac{1}{2} I_r(i - 1, j - 1) + I_m(i + 1, j - 1) \right] \\ - \left[ \frac{1}{2} I_r(i - 1, j - 1) + I_m(i + 1, j - 1) \right] \\ - 2 \left[ I_r(i - 1, j - 1) + I_m(i + 1, j - 1) \right] \\ - I_m(i - 1, j - 1) - I_m(i + 1, j - 1) \end{array} \right\} \]

Case 2: \((i \in \text{even}, j \in \text{odd})\)

\[ \Delta I_{dm}^{hr}(i, j) = \sum \left\{ \begin{array}{c} I_r(i - 1, j + 1) \\ + 2 \left[ \frac{1}{2} I_r(i - 1, j + 1) + I_m(i + 1, j + 1) \right] \\ + I_m(i + 1, j + 1) \\ - \left[ \frac{1}{2} I_r(i - 1, j + 1) + I_m(i + 1, j + 1) \right] \\ - 2 \left[ I_r(i - 1, j + 1) + I_m(i + 1, j + 1) \right] \\ - I_m(i - 1, j + 1) - I_m(i + 1, j + 1) \end{array} \right\} \]

\[ \Delta I_{dm}^{lr}(i, j) = \sum \left\{ \begin{array}{c} I_r(i + 1, j - 1) \\ + 2 \left[ \frac{1}{2} I_r(i + 1, j - 1) + I_m(i + 1, j + 1) \right] \\ + I_m(i + 1, j + 1) \\ - \left[ \frac{1}{2} I_r(i + 1, j - 1) + I_m(i + 1, j + 1) \right] \\ - 2 \left[ I_r(i + 1, j - 1) + I_m(i + 1, j + 1) \right] \\ - I_m(i + 1, j - 1) - I_m(i + 1, j + 1) \end{array} \right\} \]

\[ \Delta I_{dm}^{pr}(i, j) = \sum \left\{ \begin{array}{c} I_r(i + 1, j + 1) \\ + 2 I_r(i + 1, j + 1) \\ + I_m(i + 1, j - 1) \\ - \left[ 2 I_m(i + 1, j + 1) \right] \\ - I_m(i + 1, j - 1) - I_m(i + 1, j + 1) \end{array} \right\} \]

\[ \Delta I_{dm}(i, j) = \sum \left\{ 3 [I_r(i - 1, j - 1) - I_m(i + 1, j - 1)] \right\} \]
\[ \Delta I_{dm}^{\pi,r}(i, j) = \sum \left\{ \right. \\
\begin{aligned}
&\frac{1}{2} [I_{mo}(i - 2, j) + I_{mo}(i + 2, j - 1)] \\
&+ 2[I_{mo}(i, j) + I_{mo}(i + 2, j)] \\
&+ \frac{1}{2} [I_{mo}(i - 2, j - 1) + I_{mo}(i + 2, j + 1)] \\
&- 2[I_{mo}(i, j) + I_{mo}(i + 2, j + 1)] \\
&+ \frac{1}{2} [I_{mo}(i, j)]
\end{aligned}
\]

\[ = \sum \{ 3[I_{mo}(i, j - 1) - I_{mo}(i, j + 1)] \}
\]

**Case 3:** \( i \in \text{odd}, j \in \text{odd} \)

\[ \Delta I_{dm}^{H,r}(i, j) = \sum \left\{ \right. \\
\begin{aligned}
&\frac{1}{2} [I_{mo}(i, j) + I_{mo}(i + 2, j + 1)] \\
&+ 2[I_{mo}(i, j) + I_{mo}(i + 2, j + 1)] \\
&+ \frac{1}{2} [I_{mo}(i, j) + I_{mo}(i + 2, j + 1)] \\
&- 2[I_{mo}(i, j) + I_{mo}(i + 2, j + 1)] \\
&+ \frac{1}{2} [I_{mo}(i, j + 1)]
\end{aligned}
\]

\[ = \sum \{ 3[I_{mo}(i, j - 1) - I_{mo}(i, j + 1)] \}
\]

\[ \Delta I_{dm}^{V,r}(i, j) = \sum \left\{ \right. \\
\begin{aligned}
&\frac{1}{4} [I_{mo}(i, j - 1) + I_{mo}(i + 2, j + 1)] \\
&+ I_{mo}(i, j - 1) + I_{mo}(i, j + 1) \\
&+ 2[I_{mo}(i, j) + I_{mo}(i, j + 1)] \\
&+ \frac{1}{4} [I_{mo}(i, j) + I_{mo}(i + 2, j + 1)] \\
&- 2[I_{mo}(i, j) + I_{mo}(i + 2, j + 1)] \\
&+ \frac{1}{4} [I_{mo}(i, j + 1)]
\end{aligned}
\]

\[ = \sum \{ I_{mo}(i + 2, j) + I_{mo}(i + 2, j + 1) \}
\]
\[ \Delta I_{dm}^{r,s}(i, j) = \sum \left\{ \begin{array}{l}
\frac{1}{4} \left[ I_{mo}(i - 2, j - 1) + I_{mo}(i - 2, j + 1) \right] \\
+ \frac{1}{4} \left[ I_{mo}(i, j - 1) + I_{mo}(i, j + 1) \right] \\
+ 2 \frac{1}{2} [I_{mo}(i - 2, j - 1) + I_{mo}(i, j - 1)] \\
+ I_{mo}(i, j - 1) \\
\end{array} \right. \\
- \left\{ \begin{array}{l}
\frac{1}{4} \left[ I_{mo}(i, j - 1) + I_{mo}(i, j + 1) \right] \\
+ \frac{1}{4} \left[ I_{mo}(i + 2, j - 1) + I_{mo}(i + 2, j + 1) \right] \\
+ 2 \frac{1}{2} [I_{mo}(i, j + 1) + I_{mo}(i + 2, j + 1)] \\
+ I_{mo}(i, j + 1) \\
\end{array} \right. \\
= \sum \left\{ \begin{array}{l}
\frac{1}{4} [I_{mo}(i - 2, j + 1) - I_{mo}(i + 2, j - 1)] \\
+ \frac{1}{4} [I_{mo}(i - 2, j - 1) - I_{mo}(i + 2, j + 1)] \\
+ 2 [I_{mo}(i, j - 1) - I_{mo}(i, j + 1)] \\
\end{array} \right. \]

**Case 4:** \((i \in \text{even}, j \in \text{even})\

\[ \Delta I_{dm}^{H,r}(i, j) = \sum \left\{ \begin{array}{l}
\frac{1}{4} [I_{mo}(i - 1, j + 2) + I_{mo}(i - 1, j - 2)] \\
+ \frac{1}{4} [I_{mo}(i - 1, j) + I_{mo}(i - 1, j + 2)] \\
+ \frac{1}{4} [I_{mo}(i + 1, j + 2) + I_{mo}(i + 1, j - 2)] \\
+ I_{mo}(i - 1, j - 2) + I_{mo}(i + 1, j) \\
\end{array} \right. \\
- \left\{ \begin{array}{l}
\frac{1}{4} [I_{mo}(i - 1, j) + I_{mo}(i - 1, j - 2)] \\
+ \frac{1}{4} [I_{mo}(i + 1, j) + I_{mo}(i + 1, j - 2)] \\
+ \frac{1}{4} [I_{mo}(i - 1, j + 2) + I_{mo}(i + 1, j + 2)] \\
- I_{mo}(i - 1, j - 2) - I_{mo}(i + 1, j - 2) \\
\end{array} \right. \\
= \sum \left\{ \begin{array}{l}
I_{mo}(i - 1, j + 2) + I_{mo}(i + 1, j + 2) \\
- I_{mo}(i - 1, j - 2) - I_{mo}(i + 1, j - 2) \\
\end{array} \right. \]

\[ \Delta I_{dm}^{V,r}(i, j) = \sum \left\{ \begin{array}{l}
\frac{1}{4} [I_{mo}(i + 1, j - 2) + I_{mo}(i + 1, j)] \\
+ 2 I_{mo}(i + 1, j) \\
+ \frac{1}{2} [I_{mo}(i + 1, j + 2) + I_{mo}(i + 1, j - 2)] \\
+ \frac{1}{2} [I_{mo}(i + 1, j) + I_{mo}(i + 1, j + 2)] \\
- \frac{1}{2} [I_{mo}(i + 1, j) + I_{mo}(i + 1, j + 2)] \\
\end{array} \right. \\
- \left\{ \begin{array}{l}
\frac{1}{4} [I_{mo}(i + 1, j - 2) + I_{mo}(i + 1, j)] \\
+ 2 I_{mo}(i + 1, j) \\
+ \frac{1}{2} [I_{mo}(i + 1, j + 2) + I_{mo}(i + 1, j - 2)] \\
+ \frac{1}{2} [I_{mo}(i + 1, j) + I_{mo}(i + 1, j + 2)] \\
- \frac{1}{2} [I_{mo}(i + 1, j) + I_{mo}(i + 1, j + 2)] \\
\end{array} \right. \\
= \sum \left\{ \begin{array}{l}
3 I_{mo}(i + 1, j) - I_{mo}(i - 1, j) \\
+ \frac{1}{2} [I_{mo}(i + 1, j - 2) + I_{mo}(i + 1, j + 2)] \\
- I_{mo}(i - 1, j - 2) - I_{mo}(i - 1, j + 2) \\
\end{array} \right. \]
\[
\Delta I_{dm}^{\pi r}(i, j) = \sum \left\{ \begin{array}{c}
I_{m o}(i - 1, j) \\
+2\left[ \frac{1}{2} \left( I_{m o}(i - 1, j) + I_{m o}(i - 1, j + 2) \right) \right] \\
+ \frac{1}{4} \left( I_{m o}(i - 1, j) + I_{m o}(i + 1, j) \right) \\
- 2\left[ \frac{1}{2} \left( I_{m o}(i + 1, j - 2) + I_{m o}(i + 1, j) \right) \right] \\
+ \frac{1}{4} \left( I_{m o}(i + 1, j - 2) + I_{m o}(i + 1, j) \right)
\end{array} \right\}
\]

\[
\Delta I_{dm}^{\pi^+ r}(i, j) = \sum \left\{ \begin{array}{c}
I_{m o}(i - 1, j) \\
+2\left[ \frac{1}{2} \left( I_{m o}(i - 1, j - 2) + I_{m o}(i - 1, j) \right) \right] \\
+ \frac{1}{4} \left( I_{m o}(i - 1, j - 2) + I_{m o}(i + 1, j) \right) \\
- 2\left[ \frac{1}{2} \left( I_{m o}(i + 1, j - 2) + I_{m o}(i + 1, j) \right) \right] \\
+ \frac{1}{4} \left( I_{m o}(i + 1, j - 2) + I_{m o}(i + 1, j) \right)
\end{array} \right\}
\]

38