Bayer Demosaicking with Polynomial Interpolation

Jiaji Wu, Marco Anisetti, Wei Wu, Ernesto Damiani, and Gwanggil Jeon

***Abstract*—Demosaicking is a digital image process to reconstruct full color digital images from incomplete color samples from an image sensor. It is an unavoidable process for many devices incorporating camera sensor (e.g. mobile phones, tablet, etc.). In this paper, we introduce a new demosaicking algorithm based on polynomial interpolation-based demosaicking (PID). Our method makes three contributions: calculation of error predictors, edge classification based on color differences, and a refinement stage using a weighted sum strategy. Our new predictors are generated on the basis of on the polynomial interpolation, and can be used as a sound alternative to other predictors obtained by bilinear or Laplacian interpolation. In this paper we show how our predictors can be combined according to the proposed edge classifier. After populating three color channels, a refinement stage is applied to enhance the image quality and reduce demosaicking artifacts. Our experimental results show that the proposed method substantially improves over existing demosaicking methods in terms of objective performance (CPSNR, S-CIELAB ΔE^*, and FSIM), and visual performance.**

***Index Terms*—Demosaicking, color interpolation, polynomial interpolation, edge classifier.**

I. Introduction

Digital cameras are increasingly widespread, and camera modules are now embedded in a variety of handheld devices including mobile phones and tablet PCs. In most digital cameras, images are captured by a single image sensor whose surface is covered with a color filter array (CFA). This widely adopted solution keeps cost and size of digital cameras under control, because the image sensor is the most expensive component of the camera [1]. A single image sensor obtains only one color component at each pixel position; therefore, the captured image is in the form of a mosaic pattern. To reconstruct a full color image from the CFA-sampled image, an image processing technique that estimates the missing color values is needed. Such a technique is known as CFA interpolation or *demosaicking*.

The most common CFA pattern is the Bayer pattern, which is shown in Fig. 1 [2]. Because the spectral response of a green (G) channel corresponds to that of the human visual system's luminance channel [3], [4], the green components are sampled at twice the rate of the red (R) or blue (B) components. In other words, the green components are sampled using a quincunx grid, and the red and blue components are obtained by a rectangular grid. This technique is used in most digital cameras [5].

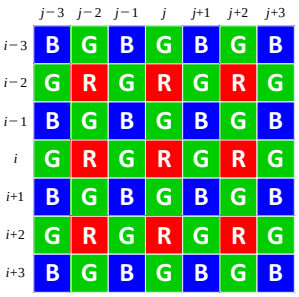


Fig. 1. A 7×7 Bayer pattern. Parameters i and j are row and column numbers, respectively.

In demosaicking, it is commonly assumed that color ratios and color differences are constant over small regions. These are referred to as the color ratio model and color difference models, respectively [6]. In [7], Cok reconstructed the missing green components and then estimated the red and blue ones using bilinear interpolation of the red-to-green and blue-to-green difference ratios. In [8], Lukac *et al*. proposed a normalized model for color ratio-based demosaicking. Because of its simplicity and effectiveness, many methods utilize the color difference model, i.e., red-green and blue-green [9]. In both cases, correct interpolation of the missing green components is crucial because interpolated green components are used to reconstruct the other colors.

Improper interpolation of neighboring pixel values leads to demosaicking artifacts, such as false colors and the so-called *zipper effect* [10]. One strategy for interpolation of neighboring pixels is the edge-directed method. In this method, two or more predictors are estimated along the candidate directions, and one of them is selected as the value of the missing pixel [11-15]. The objective of this strategy is to perform interpolation along edges rather than across them. Another strategy is computing a weighted sum of predictors [16-22]. After estimating predictors along the candidate directions, some weights are calculated on the basis of the edge directions. Each missing pixel is then interpolated by the weighted sum of predictors with the calculated weights. Other strategies use various schemes such as pattern matching, median filtering, bilateral filtering, and optimization-based filtering. Wu *et al*. exploited pattern matching [23] while Freeman utilized the median filter to improve reconstruction near edges [24]. Ramanth *et al*. proposed the use of a bilateral

Manuscript received January 15, 2015, revised August 06, 2015 and Nov. 05, 2015. This work is supported by National Natural Science Foundation of China under 61377011, 61271330, and is supported under the framework of international cooperation program managed by National Research Foundation of Korea(NRF-2016K1A3A1A25003543).

J. Wu and G. Jeon are with the School of Electronic Engineering, Xidian University, Xi'an, China, 710071, China (email: {wujj,gjeon}@mail.xidian.edu.cn).

M. Anisetti is with the Department of Computer Science, Università degli Studi di Milano, Via Bramante 65, 26013 Crema (CR), Italy (email: marco.anisetti@unimi.it).

W. Wu is with the College of Electronics and Information Engineering, Sichuan University, Chengdu, 610207, China, (email: wuwei@scu.edu.cn).

E. Damiani is with the Department of Electrical and Computer Engineering, Khalifa University, Al Zafranah, Abu Dhabi, UAE (email: ernesto.damiani@kustar.ac.ae).

G. Jeon is with the Department of Embedded Systems Engineering, Incheon National University, Incheon 406-772, Korea (email: gjeon@inu.ac.kr).

filter [25]. Zhang and Wu introduced the directional linear minimum mean square error estimation (LMMSE) method for demosaicking [26]. Menon and Calvagno incorporated regularization approaches to demosaicking (RAD) [27].

Demosaicking has been studied in the frequency domain as well. One promising class of algorithms is based on a frequency-domain explanation of the spatial multiplexing of red, green and blue components; it has been termed *luma-chroma demultiplexing* in [28-32]. In this paper, we introduce a new demosaicking algorithm based on polynomial interpolation-based demosaicking (PID). Our method makes three contributions with respect to the state of the art: i) new calculation of error predictors, ii) edge classification based on color differences and predicted error, and iii) a refinement stage using a weighted sum strategy for artefacts reduction.

The remainder of the paper is arranged as follows. Conventional edge-directed color channel interpolation methods including adaptive color plane interpolation (ACPI) and effective color interpolation (ECI) are explained in Section II. The polynomial interpolation, a motivation of our proposed method, is summarized and its application to demosaicking is presented in Section III. Our proposed method based on the polynomial interpolation is explained in Section IV. The experimental results comparing the proposed polynomial interpolation-based demosaicking (PID) method with other reference methods are provided in Section V. We conclude the paper in Section VI.

II. Edge-directed Color Channel Interpolation Methods

Several methods generate predictors along all candidate directions, as mentioned above; then, either one of them is selected, or they are fused with appropriate weights based on the direction. In this paper, we assume $X_{i,j}$ is the intensity value at (i,j) in the X color plane. $\widehat{X}_{i,j}$ is the estimated value after demosaicking and $\widetilde{X}_{i,j}$ is temporary variable value to calculate $\widehat{X}_{i,j}$.

In [11], the authors proposed ACPI and generate two predictors, defined as

$$\begin{aligned} G^H_{i,j} &= \frac{G_{i,j-1}+G_{i,j+1}}{2}+\frac{2R_{i,j}-R_{i,j-2}-R_{i,j+2}}{4}, \\ G^V_{i,j} &= \frac{G_{i-1,j}+G_{i+1,j}}{2}+\frac{2R_{i,j}-R_{i-2,j}-R_{i+2,j}}{4}, \\ G^A_{i,j} &= \frac{G^H_{i,j}+G^V_{i,j}}{2}, \end{aligned} \tag{1}$$

where $G^H_{i,j}$ and $G^V_{i,j}$ are estimates for the horizontal and vertical directions, and $G^A_{i,j}$ is an estimate for the omni-direction. To select one of them, an edge classifier composed of the Laplacian second-order terms for R/B and the gradient terms for G is utilized. This technique is known to provide good results, and it has been further studied by many researchers. For example, an edge classifier using the variances of color differences was presented by Chung *et al.* [12]. The directional filtering and *a posteriori* decision method was proposed by Menon *et al*., who used a different edge classifier composed of the sum of the gradients in the horizontal and vertical directions [13]. Dengwen *et al*. improved the DFPD [14], while Su and Kao exploited discrete wavelet transformed coefficients as the classifier [15].

Another method for estimating predictors is to exploit the weighted sum of color differences in the vicinity. In early study in effective color interpolation (ECI) [16], the missing green components were reconstructed as follows:

$$\begin{aligned} \widehat{G}_{i,j} &= R_{i,j}+\widetilde{K}_{i,j} \\ &= R_{i,j}+\frac{\widetilde{K}_{i,j-1}+\widetilde{K}_{i,j+1}+\widetilde{K}_{i-1,j}+\widetilde{K}_{i+1,j}}{4}, \end{aligned} \tag{2}$$

where $\widetilde{K}_{i,j}$ is considered as difference between original $R_{i,j}$ and estimated $\widehat{G}_{i,j}$, and the color difference values of the four neighbors are computed as

$$\begin{aligned} \widetilde{K}_{i,j-1} &= G_{i,j-1}-\widetilde{R}_{i,j-1}=G_{i,j-1}-\frac{R_{i,j-2}+R_{i,j}}{2}, \\ \widetilde{K}_{i,j+1} &= G_{i,j+1}-\widetilde{R}_{i,j+1}=G_{i,j+1}-\frac{R_{i,j+2}+R_{i,j}}{2}, \\ \widetilde{K}_{i-1,j} &= G_{i-1,j}-\widetilde{R}_{i-1,j}=G_{i-1,j}-\frac{R_{i-2,j}+R_{i,j}}{2}, \\ \widetilde{K}_{i+1,j} &= G_{i+1,j}-\widetilde{R}_{i+1,j}=G_{i+1,j}-\frac{R_{i+2,j}+R_{i,j}}{2}. \end{aligned} \tag{3}$$

Chang and Tan improved ECI by using adaptive weight obtained by a gradient operator instead of the fixed weight, $1/4$ [17]. An effective demosaicking algorithm based on edge property (EDAEP) is an another approach based on ECI, in which the weights are determined according to the edge direction [18]. Pekkucuksen and Altunbasak utilized the edge strength filter (ESF) [19] and proposed multiscale gradients-based method (MSG) [20]. Kim *et al*. used geometric duality and the dilated directional differentiation of color differences for demosaicking (GD) [21]. Chen *et al*. proposed a demosaicking method by introducing a voting strategy and provided accurate directional interpolation method [22].

In fact, the predictor for the horizontal direction in Eq. (1) can be represented by

$$\begin{aligned} G^H_{i,j} &= R_{i,j}+\frac{1}{2}\left(G_{i,j-1}-\frac{R_{i,j}+R_{i,j-2}}{2}\right) \\ &\quad +\frac{1}{2}\left(G_{i,j+1}-\frac{R_{i,j}+R_{i,j+2}}{2}\right) \\ &= R_{i,j}+\frac{1}{2}\{(G_{i,j-1}-\widetilde{R}_{i,j-1})+(G_{i,j+1}-\widetilde{R}_{i,j+1})\}. \end{aligned} \tag{4}$$

In the same manner, $G^V_{i,j}$ can be obtained. As shown in Eqs. (2) and (4), both of ECI-based and ACPI-based methods use temporally interpolated red components at neighboring green sampling positions. This property is used in our study because temporal interpolation is as important for demosaicked image quality as the fusing strategy. Based on this observation, we propose a new demosaicking algorithm, including a new scheme to generate the predictors and a new edge classifier. A new refinement scheme is also presented, which follows the concept of the initial interpolation.

III. Interpolation Error and Its Application to Demosaicking

A. Interpolation Error

Assume that $f \in C^{n+1}[a,b]$ and $x_0, x_1, \ldots, x_n$ are distinct nodes in $[a,b]$. Let f be $n+1$ times continuously differentiable on a closed interval $[a,b]$, and $p_n(x)$ be the unique polynomial of degree with maximum n that interpolates f at $n+1$ distinct points $\{x_i\}$, where $i = 0, 1, \ldots, n$, in that interval. Then, for each x in the interval, there exists c in that interval such that

$$f(x) - p_n(x) = e_n(x), \tag{5}$$

where the error function is given by

$$e_n(x) = \frac{1}{(n+1)!} f^{(n+1)}(c) \prod_{i=0}^{n} (x - x_i). \tag{6}$$

That is, the error function is given in the form of the Taylor series [33]. The expression for the error e_n can be simpler if we select to distribute our nodes so that they are uniformly spaced. That is, fix n and define the step length $h = (b-a)/n$, then we can clarify the partition of the interval $[a,b]$ as follows: $x_0 = a, x_i = a + ih, x_n = b$, for $i = 1, 2, \ldots, n$. When the points are equally spaced[1], the interpolation error is bound. If we consider $n = 2$ for two points x_0 and $x_0 + h$, it follows from Eq. (6) that there is a point ξ such that

$$\begin{aligned} f(x) \cong f(x_0) + \frac{f(x_0+h) - f(x_0)}{h}(x - x_0) \\ + \frac{f''(\xi)}{2}(x - x_0)(x - (x_0 + h)). \end{aligned} \tag{7}$$

For $x = x_0 + h/2$, we have

$$\begin{aligned} f\left(x_0 + \frac{h}{2}\right) &\cong f(x_0) + \frac{f(x_0+h) - f(x_0)}{h} \cdot \frac{h}{2} \\ &\quad + \frac{f''(\xi)}{2} \cdot \frac{h}{2} \cdot \left(-\frac{h}{2}\right) \\ &= \frac{f(x_0) + f(x_0+h)}{2} - \frac{f''(\xi)}{8} \cdot h^2. \end{aligned} \tag{8}$$

In general, we cannot calculate $f''(\xi)$ (i.e. the error-amender term) because we do not have any information for the value between $f(x-1)$ and $f(x+1)$. In the case of demosaicking, however, we can estimate a good candidate value for ξ by considering the color difference model.

B. Application to Demosaicking

In this section, we explain the application of the polynomial interpolation introduced in the previous section to the demosaicking process. The application of this theorem is performed using the same color components for the first term and different color components for the second term in Eq. (8). The pixels in CFA are spaced one pixel apart, $(x_0 + \frac{1}{2}h) - (x_0) = (x_0 + h) - (x_0 + \frac{1}{2}h) = 1$; that is, $h = 2$. Then, the estimate of $R_{i,j-1}$, as shown in Fig. 1, can be calculated using Eq. (8):

$$\begin{aligned} \widetilde{R}_{i,j-1} &\cong f'(\xi_1) - \frac{f''(\xi_1)}{8} \cdot h^2 \\ &= \frac{R_{i,j-2} + R_{i,j}}{2} - \frac{f''(\xi_1)}{8} \cdot h^2, \end{aligned} \tag{9}$$

where the second term on the right side is defined as

$$\begin{aligned} \frac{f''(\xi_1)}{8} \cdot h^2 &\cong \frac{1}{2h}\left(\frac{R_{i,j+1} - R_{i,j-1}}{h} - \frac{R_{i,j-1} - R_{i,j-3}}{h}\right) \\ &= \frac{R_{i,j+1} - 2R_{i,j-1} + R_{i,j-3}}{2h^2}. \end{aligned} \tag{10}$$

Let the color difference plane, Λ, between G and $\widehat{R}$ be defined as

$$\Lambda = G - \widehat{R}, \tag{11}$$

Then, by applying the color difference model of Eq. (11), we obtain

$$\begin{aligned} \frac{f''(\xi_1)}{8} \cdot h^2 &\cong \frac{(G_{i,j+1} - \Lambda_{i,j+1}) - 2(G_{i,j-1} - \Lambda_{i,j-1})}{2h^2} \\ &\quad + \frac{(G_{i,j-3} - \Lambda_{i,j-3})}{2h^2}. \end{aligned} \tag{12}$$

The color differences are quite constant over small regions; i.e., $\Lambda_{i,j+1} \doteq \Lambda_{i,j-1} \doteq \Lambda_{i,j-3}$. Therefore, Eq. (12) can be approximated to

$$\frac{f''(\xi_1)}{8} \cdot h^2 \cong \frac{G_{i,j+1} - 2G_{i,j-1} - G_{i,j-3}}{2h^2}, \tag{13}$$

Substituting Eq. (13) into Eq. (9) gives

$$\widetilde{R}_{i,j-1} \cong \frac{R_{i,j-2} + R_{i,j+1}}{2} - \frac{G_{i,j+1} - 2G_{i,j-1} - G_{i,j-3}}{2h^2}. \tag{14}$$

We can analogously compute the estimate of $R_{i,j+1}$.

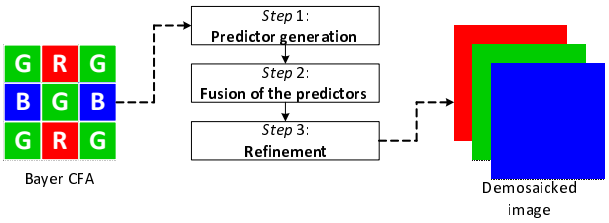


Fig. 2. Flowchart of the proposed algorithm.

IV. Demosaicking Implementation

A. Green Channel Interpolation

Our proposed method is presented as a flowchart in Figure 2. The first step in reconstructing the missing green components is to generate the directional predictors. Because the color difference plane is smoother than the R/B plane, interpolating a pixel using the color difference plane is much more effective than using the original values such as R or B [9]. In our method, we interpolate the green plane with the use of the color difference model. For a missing green component at a

[1]In case of non-equal but linear spacing, this bound can be easily generalized to be a symmetric function of the spacing step [34].

red sampled position, $R_{i,j}$ in Fig. 1, we define the predictor for the horizontal direction as

$$\widehat{G}^H_{i,j} = R_{i,j} + \widetilde{\Lambda}_{i,j} \cong R_{i,j} + \frac{\widetilde{\Lambda}_{i,j-1} + \widetilde{\Lambda}_{i,j+1}}{2}, \quad (15)$$

where $\widetilde{\Lambda} = G - \widetilde{R}$. Using the color difference model, we have

$$\begin{aligned}\widehat{G}^H_{i,j} &\cong R_{i,j} + \frac{G_{i,j-1} - \widetilde{R}_{i,j-1} + G_{i,j+1} - \widetilde{R}_{i,j+1}}{2} \\ &= R_{i,j} + \frac{G_{i,j-1} + G_{i,j+1}}{2} - \frac{\widetilde{R}_{i,j-1} + \widetilde{R}_{i,j+1}}{2}, \quad (16)\end{aligned}$$

where the third term of the right side can be calculated using Eq. (14). The predictor for the vertical direction is analogously obtained as

$$\begin{aligned}\widehat{G}^V_{i,j} &\cong R_{i,j} + \frac{G_{i-1,j} - \widetilde{R}_{i-1,j} + G_{i+1,j} - \widetilde{R}_{i+1,j}}{2} \\ &= R_{i,j} + \frac{G_{i-1,j} + G_{i+1,j}}{2} - \frac{\widetilde{R}_{i-1,j} + \widetilde{R}_{i+1,j}}{2}. \quad (17)\end{aligned}$$

Note that in order to calculate a predictor, 12 addition operations and seven shift operations are required. This calculation can be reduced to seven addition operations and four shift operations by reusing some intermediate computation results.

The second step is to fill in the missing pixels based on the edge classifier. In our method, we consider two directions: horizontal and vertical. Let ψ^H and ψ^V be the costs for the horizontal and vertical directions, respectively, defined as

$$\begin{aligned}\psi^H &= \sum_{m=\{-2,0,2\}} \sum_{n=\{-2,0,2\}} |\widehat{G}^H_{i+m,j+n} - R_{i+m,j+n}|, \\ \psi^V &= \sum_{m=\{-2,0,2\}} \sum_{n=\{-2,0,2\}} |\widehat{G}^V_{i+m,j+n} - R_{i+m,j+n}|. \quad (18)\end{aligned}$$

The ratio, Ψ, of the two cost values is used as the edge classifier, which is computed by

$$\Psi = \max\left(\frac{\psi^H}{\psi^V}, \frac{\psi^V}{\psi^H}\right). \quad (19)$$

If Ψ is larger than a predefined threshold, τ, the pixel is defined to be in a strong edge region. In this case, the missing green component is interpolated as follows:

$$\begin{aligned}\widehat{G}_{i,j} &= \widehat{G}^H_{i,j} && \text{if } (\Psi > \tau) \& \left(\Psi = \tfrac{\psi^V}{\psi^H}\right), \\ \widehat{G}_{i,j} &= \widehat{G}^V_{i,j} && \text{if } (\Psi > \tau) \& \left(\Psi = \tfrac{\psi^H}{\psi^V}\right), \\ \widehat{G}_{i,j} &= \tfrac{\omega^H \widehat{G}^H_{i,j} + \omega^V \widehat{G}^V_{i,j}}{\omega^H + \omega^V} && \text{if } (\Psi \le \tau).\end{aligned} \quad (20)$$

Pixels which are not classified into a strong edge region ($\upsilon \le \tau$) are interpolated using the weighted sum method strategy. Here, ω^H and ω^V are weights for the horizontal and vertical directions, respectively, and are defined as

$$\begin{aligned}\omega^H &= \frac{1}{d^H + \varepsilon}, \\ \omega^V &= \frac{1}{d^V + \varepsilon}, \quad (21)\end{aligned}$$

where ε is a small number added to the denominators to avoid division by zero. For computational simplicity, we use the directional differences used in ACPI as d^H and d^V in Eq. (21), which are respectively defined as

$$\begin{aligned}d^H &= |G_{i,j-1} - G_{i,j+1}| + |2R_{i,j} - R_{i,j-2} - R_{i,j+2}|, \\ d^V &= |G_{i-1,j} - G_{i+1,j}| + |2R_{i,j} - R_{i-2,j} - R_{i+2,j}|. \quad (22)\end{aligned}$$

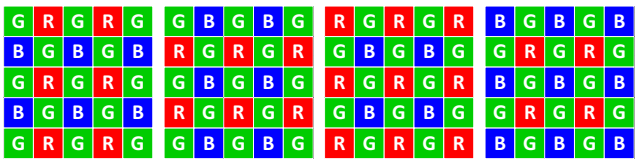


Fig. 3. Four cases of red and blue channel interpolation (a) and (b): green pixel-centered cases, (c) red pixel-centered case, (d) blue pixel-centered case.

B. Red and Blue Channel Interpolation

After populating the G channel, we can estimate R and B channels using G channel information. Figure 3 shows four possible cases. Figures 3(a) and 3(b) represent the cases in which a green pixel is centered, and Figs. 3(c) and 3(d) are the cases in which a B component interpolation at an R pixel is required and vice versa. When interpolating an R component in a GR line, as shown in Fig. 3(a), there is no vertical information for red pixels. A similar problem also occurs in a GB line, where horizontal information for red pixels is not available, as shown in Fig. 3(b). For that reason, we horizontally apply the polynomial interpolation-based demosaicking algorithm in a GR line and vertically in a GB line.

Recall that Λ is a color difference plane between the populated green channel and the red channel, $\widehat{\Lambda} = G - \widehat{R}$. Then, an R component at the G sampling position in the GR line in Fig. 3(a) is interpolated as follows:

$$\begin{aligned}\widehat{R}_{i,j} &= G_{i,j} - \widehat{\Lambda}_{i,j} \\ &= G_{i,j} - \left\{\frac{\Lambda_{i,j-1} + \Lambda_{i,j+1}}{2} - \frac{f''(\xi_2)}{8}\right\}. \quad (23)\end{aligned}$$

Because we apply a polynomial interpolation-based demosaicking scheme to the color difference plane, information for ξ_2 between two pixels located at $(i, j-1)$ and $(i, j+1)$ is not directly available. One solution for estimating ξ_2 is to average the second derivatives at $(i, j-1)$ and $(i, j+1)$. Therefore, Eq. (23) can be approximated as

$$\begin{aligned}\widehat{R}_{i,j} &= G_{i,j} - \frac{\Lambda_{i,j-1} + \Lambda_{i,j+1}}{2} \\ &\quad + \frac{1}{8}\left(\frac{\Lambda_{i,j+1} - 2\Lambda_{i,j-1} + \Lambda_{i,j-3}}{2}\right) \\ &\quad + \frac{1}{8}\left(\frac{\Lambda_{i,j+3} - 2\Lambda_{i,j+1} + \Lambda_{i,j-1}}{2}\right) \\ &= G_{i,j} - \frac{\Lambda_{i,j-1} + \Lambda_{i,j+1}}{2} \\ &\quad + \frac{\Lambda_{i,j-3} - \Lambda_{i,j-1} - \Lambda_{i,j+1} + \Lambda_{i,j+3}}{16}. \quad (24)\end{aligned}$$

For the reconstruction of a missing red component in a GB line in Fig. 3(b), we use a similar equation:

$$\widehat{R}_{i,j} = G_{i,j} - \frac{\Lambda_{i-1,j} + \Lambda_{i+1,j}}{2} + \frac{\Lambda_{i-3,j} - \Lambda_{i-1,j} - \Lambda_{i+1,j} + \Lambda_{i+3,j}}{16}. \quad (25)$$

When reconstructing a missing red component at a blue sampling position, as shown in Fig. 3(d), we use the four adjacent color values along the diagonal directions. Namely, we employ the averaging strategy given as follows:

$$\widehat{R}_{i,j} = G_{i,j} - \widehat{\Lambda}_{i,j} = G_{i,j} + \frac{\Lambda_{i-1,j-1} + \Lambda_{i-1,j+1} + \Lambda_{i+1,j-1} + \Lambda_{i+1,j+1}}{4}. \quad (26)$$

C. Refinement Process

After reconstructing the full color image, we refine it to reduce demosaicking artifacts, such as false colors and the zipper effect. At this point, we have three color values at every sampling position; therefore, we can make effective use of additional color information compared to when performing the initial interpolation described in the previous section. In the refinement step for the green components, we utilize four directions, north (N), west (W), south (S), and east (E). The gradient values, Δ^{Φ}, for each direction, Φ, are defined as

$$\begin{aligned} \Delta^N &= |R_{i,j} - R_{i-2,j}| + |G_{i-1,j} - G_{i-3,j}|, \\ \Delta^W &= |R_{i,j} - R_{i,j-2}| + |G_{i,j-1} - G_{i,j-3}|, \\ \Delta^S &= |R_{i,j} - R_{i+2,j}| + |G_{i+1,j} - G_{i+3,j}|, \\ \Delta^E &= |R_{i,j} - R_{i,j+2}| + |G_{i,j+1} - G_{i,j+3}|, \end{aligned} \quad (27)$$

where we use original sampling values to compute the gradients because they are more reliable than the interpolated values. Based on these gradient values, we calculate the weights for the four directions as follows:

$$\omega^{\Phi} = \frac{\psi^{\Phi}}{\sum_{X=\{N,W,S,E\}} \psi^X} \quad \text{for } \Phi = \{N, W, S, E\}, \quad (28)$$

where ψ^{Φ} is the cost value in Φ direction, for each direction and are respectively defined as

$$\psi^{\Phi} = \frac{\prod_{X=\{N,W,S,E\}} \Delta^X}{\Delta^{\Phi}} \quad \text{for } \Phi = \{N, W, S, E\}, \quad (29)$$

The final green component, $\overline{G}_{i,j}$, is obtained using the weighted sum of the four neighboring color difference values with the use of the weights computed by Eq. (28) as:

$$\overline{G}_{i,j} = \widehat{R}_{i,j} + (\omega^N \cdot \Lambda_{i-1,j} + \omega^W \cdot \Lambda_{i,j-1} + \omega^S \cdot \Lambda_{i+1,j} + \omega^E \cdot \Lambda_{i,j+1}). \quad (30)$$

Following green channel refinement, the red and blue channels are updated. The red components are refined using Eqs. (28-30) with the refined green components, and the blue components are similarly updated by exchanging R for B in Eqs. (28-30).

(a) (b) (c)

Fig. 4. Test sets for comparison: (a) 18 McM dataset, (b) 16 CMLA dataset, and (c) 24 Kodak dataset.

V. Experimental Results

In this section, the performance of the proposed method is evaluated and compared with some reference benchmarks. In the experiments, three image sets were used for comparison test and three image sets were used for training. Test sets include 18 McM dataset [41], 16 CMLA dataset [36], and 24 Kodak dataset, while training sets include 150 LC dataset [37], 25 Zahra dataset [38], and 9,907 Stanford dataset [39]. Figure 4 shows test image sets: McM, CMLA, and Kodak datasets. Compare to McM dataset, the Kodak images are statistically less colorful and sharper.

A. Simulation Setup for Comparison

In this section, the performance of the proposed method is evaluated and compared with some reference benchmarks. In the experiments, two image sets were used for comparison test and three image sets were used for training. Test sets include 18 McM dataset [35] and 16 CMLA dataset [36], while training sets include 150 LC dataset [37], 25 Zahra dataset [38], and 9,907 Stanford dataset [39]. Figure 4 shows test image sets: McM and CMLA datasets.

The McM dataset has been used in many recent survey papers. The McM dataset consists of 18 full color images with 500×500 pixel resolution. We also used a CMLA dataset of 16 images obtained with a Nikon D80 camera using ISO 1250 and exposure time $1/640$ s. In CMLA images the darkest pixels are saturated, and therefore the noise curve does not follow the linear variance model. CMLA dataset contains some views of buildings, dice, people, and outdoor images. Especially, two building images are highly textured. The CMLA dataset consists of 16 full color images with 704×469 pixel resolution, respectively.

The full color images were first downsampled in the Bayer CFA pattern and then interpolated using the proposed PID method. The demosaicked images were compared to the original full color images, and the results are reported in terms of color peak signal-to-noise ratio (CPSNR), S-CIELAB ΔE^* [40], and feature-similarity (FSIM) [41]. For comparison, we have selected the following reference methods:

(A) A luma-chroma demultiplexing (LSLCD) method uses a least-squares design methodology for the required bandpass filters [31]. LSLCD examines the tradeoff relationship between demosaicking quality and speed using trained filters in order to recommend filter specifications for the best system that balances quality and speed. Before applying LSLCD, training set is required for filter design, and the selection of training set may affect the sensitivity of the performance.

(B) There are two steps in color demosaicking with directional filtering and weighting (CDDFW) method [14], initial step and refinement step. In the initial step, the initial estimates of the R, G and B planes are computed, whereas in the refinement step the G plane can be further refined using the initial R and B planes and vice versa. CDDFW classifies region into two, strong edge region and smooth region, and applies DFPD for strong edge region and applies weighted average for smooth region.

(C) In effective demosaicking algorithm based on edge property for color filter arrays (EDAEP) method [15], authors apply initial interpolation and refinement process. The initial interpolation is modification of EDI and refinement process is identical to [42]. The obtained accurate weights are used for image demosaicking, before refinement is made in post-processing. We note that, refinement is performed only to the interpolated green pixel values.

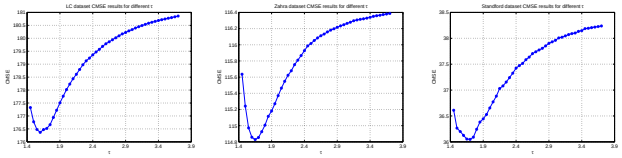
(D) In edge strength filter based CFA interpolation (ESF) [19], authors proposed an orientation-free edge strength filter and applied it to the demosaicking issue. The ESF output is utilized both to improve the initial green channel interpolation and to apply the constant color difference rule adaptively.

(E) The same authors of [19] proposed the multi-scale gradients-based CFA interpolation (MSG) [20], where the authors proposed a method that uses multiscale color gradients to adaptively combine color difference estimates from different directions. The MSG method does not require any thresholds as it does not make hard decision, and all process is non-iterative.

(F) In geometric duality and dilated directional differentiation based method (GD) [21], the authors noticed that a given high resolution image and its low resolution image obtained by sampling have similar edge properties. By using this property, GD computes the interpolation errors for the candidate directions in the low resolution image, and exploits them as a cost term for the direction. By combining the edge classifier and the weighted sum of the estimates obtained by approximation, demosaicking is performed.

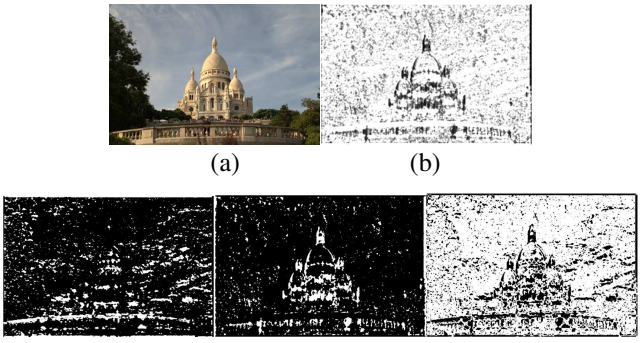
(G) In voting-based directional interpolation method (VDI) [22], authors introduced the voting strategy and its application to the interpolation direction of the center missing color component. Along the determined interpolation direction, the center missing color component is determined accurately and interpolated using the gradient weighted interpolation method by exploring the intra-channel gradient correlation of the neighboring pixels.

(H) Our polynomial interpolation-based demosaicking (PID) method has three steps, (1) calculation of error predictors, (2) edge classification based on color differences, (3) a refinement stage using a weighted sum strategy. The proposed predictors are generated on the basis on the polynomial interpolation. After interpolating three color channels, a refinement stage is applied to enhance the image quality and reduce demosaicking artifacts.



(a) (b) (c)

Fig. 5. CMSE results for: (a) 150 LC dataset ($\tau = 1.8$), (b) 25 Zahra dataset ($\tau = 1.9$), and (c) 9,907 Stanford dataset ($\tau = 2.0$). Threshold value τ is determined as 1.9.



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

Fig. 6. An example of Zahra #4 image: (a) original image, (b) weight maps of ψ^V/ψ^H, (c) pixels with $\widehat{G}_{i,j} = \widehat{G}^H_{i,j}$, (d) pixels with $\widehat{G}_{i,j} = \widehat{G}^V_{i,j}$, and (e) pixels with $\Psi \leq \tau$. We note that we assumed $\tau = 1.9$.

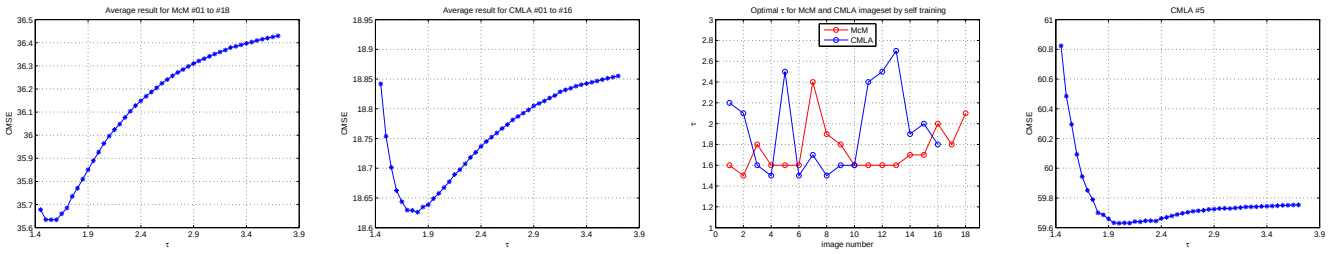
Seven recently presented demosaicking methods, which are published after 2011, were implemented for comparison: LSLCD [31], CDDFW [14], EDAEP [15], ESF [19], MSG [20], GD [21], and VDI [22]. All of the simulations were carried out on the Intel Core i7-3537U CPU at 2.00 GHz.

B. Threshold Parameter Tuning

A proper setting for threshold parameter τ is important for the success of our method. Parameter τ plays an important role to balance performance of Eq. (20). In our scheme, $\widehat{G}^H_{i,j}$ or $\widehat{G}^V_{i,j}$ is assigned to $\widehat{G}_{i,j}$ when Ψ is larger than predetermined threshold τ because the pixel is assumed to be in a strong edge region. Otherwise, weighted sum strategy is used to compute $\widehat{G}_{i,j}$.

To obtain the most suitable parameter τ, we tested our method with several values of τ, ranging from 1.0 to 10.0, with 0.1 increments. Figure 5 shows the average CMSE performance for 150 LC dataset, 25 Zahra dataset, and $9,907$ Stanford dataset using Eq. (20). From Fig. 5, we found that $\tau = 1.8, 1.9$, and 2.0 yield the least CMSE for LC, Zahra, and Stanford datasets, respectively. Therefore, we concluded that $\tau = 1.9$ is the most reliable parameter value.

Figure 6 shows an example of #4 Zahra image. Figure 6(a) is the original image, Fig. 6(b) is the weight maps of $\frac{\psi^V}{\psi^H}$, Figs. 6(c-e) show pixels with $\widehat{G}_{i,j} = \widehat{G}^H_{i,j}$, $\widehat{G}_{i,j} = \widehat{G}^V_{i,j}$, and $\Psi \leq \tau$, respectively. It was found that 13.80% of pixels were determined as $\widehat{G}_{i,j} = \widehat{G}^H_{i,j}$, that 8.94% of pixels were



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

Fig. 7. (a) Average CMSE for various τ for McM dataset, (b) average CMSE for various τ for CMLA dataset, (c) optimal τ for McM and CMLA imageset by self-training, and (d) CMSE for varying τ values on #5 of the CMLA dataset.

TABLE I
FOUR METRICS TO ASSESS PERFORMANCE FOR TWO τ CONDITIONS

	CPSNR	S-CIELAB ΔE^*	FSIM	CMSE
$\tau = 1.9$	30.35	2.0112	0.99378	59.9902
$\tau = 2.6$	30.3727	2.0097	0.99379	59.6772
Difference	0.0227	-0.0015	0.00001	-0.313

determined as $\widehat{G}_{i,j} = \widehat{G}^V_{i,j}$, and 77.26% of pixels were determined as $\Psi \leq \tau$.

Figure 7 shows the optimal τ for McM and CMLA imageset. For McM imageset, best τ was 1.8, which is very close to our assumption $(\tau = 1.9)$. It can be found from Fig. 7(c) that throughout 18 McM images, τ values range from 1.5 (#2) to 2.4 (#7), and most τ values locate around 1.9. For CMLA imageset, best τ was 2.2, which is 0.3 bigger value than our assumption. The τ values range from 1.5 (#4, 6, 8) to 2.7 (#13), and most τ values locate around 2.2. However, MSE of #13 for $2.1 \leq \tau \leq 2.7$ showed similar performance. Therefore, we conclude that our assumption $\tau = 1.9$ is reasonable.

Although $\tau = 1.9$ was selected for our system, this value caused worse performance for #5 of the CMLA dataset. Figure 7(d) shows CMSE result for varying τ values. It is obvious that $\tau = 2.6$ is the optimal selection for this image with CMSE (=59.6772), while globally selected $\tau = 1.9$ gives worse CMSE (59.9902). Table I shows results of four metrics (CPSNR, S-CIELAB ΔE^*, FSIM, and CMSE) for two τ conditions: $\tau = 1.9$ and $\tau = 2.6$. It is noted that all numerical results were obtained from PID_{WR} to clearly show the effect of the τ selection. It can be found from Table I that the optimal selection $(\tau = 2.6)$ outperformed the global selection $(\tau = 1.9)$ for all four metrics.

C. Performance Comparison in CFA images

Tables II and III show the CPSNR results of the 18 McM and 16 CMLA images and the average error measures and three rankings. R_{WR}, R_P, and R_M stand for PID performance without refinement and with refinement, and ranking of each reference method. We rejected an eleven pixels wide border from each image during the performance calculations to leave edge effects not representative of the error performance of the various methods. These numerical results may not match directly to those informed in original published articles because of differences in the test platforms, such as different versions of the same images being adopted or slight differences in the metric calculations.

TABLE II
CPSNR RESULTS ON MCM DATASET FOR VARIOUS DEMOSAICKING METHODS (IN DB)

Im#	LSLCD	CDDFW	EDAEP	ESF	MSG	GD	VDI	PID_{WR}	PID	R_{WR}	R_P
1	27.644	27.133	27.591	26.061	27.193	27.145	28.054	26.487	29.082	8	1
2	33.579	33.539	33.981	33.07	33.807	33.868	34.281	33.345	34.619	8	1
3	31.272	32.616	32.055	32.291	33.029	33.023	32.643	32.535	32.19	5	7
4	34.501	34.709	34.347	34.647	35.713	35.8	35.989	34.646	36.395	7	1
5	32.365	31.08	32.076	30.279	31.251	31.471	32.696	30.619	34.059	8	1
6	35.622	33.749	35.037	32.109	33.773	33.797	35.783	32.68	37.596	8	1
7	34.64	38.971	36.215	38.796	39.138	38.627	36.025	39.061	34.811	2	8
8	36.581	37.312	37.114	37.286	37.635	37.927	37.428	37.538	36.61	3	8
9	35.656	34.794	35.244	33.968	34.868	35.029	36.159	34.015	36.778	8	1
10	36.86	36.917	36.994	35.553	36.669	36.428	37.452	35.709	38.281	8	1
11	37.828	37.501	37.829	36.438	37.44	37.373	38.144	36.597	39.468	8	1
12	37.14	36.827	37.148	35.888	36.901	36.916	37.54	36.415	38.209	8	1
13	39.232	38.629	39.332	38.271	38.952	39.1	39.843	38.454	40.879	8	1
14	37.738	37.127	37.641	36.652	37.235	37.313	38.04	36.815	38.598	8	1
15	37.958	37.321	37.757	36.708	37.302	37.305	38.153	36.759	39.094	8	1
16	31.68	30.058	31.402	28.989	30.26	30.51	31.89	29.685	34.194	8	1
17	30.859	29.774	30.578	28.334	29.486	29.559	31.196	28.471	33.167	8	1
18	34.062	33.82	34.066	33.502	34.172	34.296	34.545	33.918	35.549	7	1
Avg.	34.734	34.549	34.8	33.825	34.712	34.749	35.326	34.097	36.088	8	1
R_M	5	7	3	9	6	4	2	8	1		

TABLE III
CPSNR RESULTS ON CMLA DATASET FOR VARIOUS DEMOSAICKING METHODS (IN DB)

Im#	LSLCD	CDDFW	EDAEP	ESF	MSG	GD	VDI	PID_{WR}	PID	R_{WR}	R_P
1	31.211	33.06	33.525	32.802	33.495	33.512	33.67	33.291	33.744	6	1
2	30.512	32.429	32.63	32.446	32.921	32.975	32.799	32.854	32.791	3	5
3	30.922	35.106	34.675	34.145	35.123	35.051	34.947	34.583	35.141	7	1
4	38.127	43.888	43.638	42.421	43.737	43.449	44.324	42.594	46.482	7	1
5	28.157	30.136	30.868	29.812	30.496	30.693	30.993	30.356	31.593	6	1
6	32.453	38.399	38.827	36.988	38.383	38.585	40.111	37.189	41.993	7	1
7	40.665	43.806	43.93	43.727	44.046	44.153	44.148	43.814	44.373	6	1
8	37.409	41.393	41.555	41.325	41.829	41.804	41.69	41.585	42.162	5	1
9	41.318	44.885	45.169	44.305	44.969	44.887	45.336	44.547	46.025	7	1
10	39.799	43.463	43.491	43.143	43.786	43.736	43.781	43.389	43.498	7	4
11	33.923	35.615	36.205	35.641	36.11	36.185	36.406	36.041	36.303	6	2
12	34.701	37.666	37.22	37.712	38.098	37.969	37.21	37.93	37.09	3	8
13	31.232	34.119	34.114	34.188	34.721	34.745	34.446	34.537	34.206	3	5
14	33.205	35.668	35.746	35.639	36.169	36.229	36.073	35.982	36.138	5	3
15	29.081	32.119	31.956	31.773	32.397	32.325	32.13	32.165	32.536	4	1
16	32.72	35.138	35.248	35.198	35.701	35.773	35.551	35.498	35.391	4	5
Avg.	34.09	37.306	37.425	36.954	37.624	37.629	37.726	37.272	38.092	7	1
R_M	9	6	5	8	4	3	2	7	1		

It can be seen from Table II that the estimates of the demosaicked images by our PID method are very competitive with the reference methods. The PID method outperformed most other methods on average: 1.354 dB (LSLCD), 1.539 dB (CDDFW), 1.288 dB (EDAEP), 2.263 dB (ESF), 1.376 dB (MSG), 1.339 dB (GD), 0.762 dB (VDI), and 1.991 dB (PID_{WR}), respectively. We note that TO filters were used for LSLCD results. Although the PID method did not provide better performance than MSG or GD for #3, #7, #8 McM images, the average results outperformed both methods.

The CPSNR results on CMLA dataset are shown in Ta-

TABLE IV
S-CIELAB ΔE^* RESULTS ON MCM DATASET FOR VARIOUS DEMOSAICKING METHODS

Im#	LSLCD	CDDFW	EDAEP	ESF	MSG	GD	VDI	PID$_{WR}$	PID	R$_{WR}$	R$_P$
1	2.8057	3.1895	2.9027	3.7844	3.0946	3.0861	2.62	3.4693	2.6171	8	1
2	1.1486	1.2964	1.2458	1.4533	1.2202	1.2612	1.1444	1.3689	1.1682	8	3
3	1.736	2.0829	2.2897	2.0759	1.7848	1.7993	1.8931	2.0834	1.7059	8	1
4	0.8595	1.2531	1.4838	1.4612	1.0304	1.177	1.113	1.4308	0.8617	7	2
5	1.381	1.5506	1.5258	1.771	1.5053	1.5503	1.3735	1.7586	1.2147	8	1
6	1.0866	1.4644	1.2781	1.8569	1.4524	1.4904	1.1271	1.7557	0.9875	8	1
7	1.2851	0.9321	1.2284	0.944	0.9011	0.9489	1.1994	0.9074	1.2386	2	8
8	0.5988	0.6315	0.7229	0.6586	0.5843	0.5723	0.6125	0.6126	0.6205	5	6
9	1.0033	1.2169	1.3045	1.3957	1.1659	1.2124	1.0579	1.4224	0.9494	9	1
10	0.9064	0.9759	0.9988	1.1917	0.99	1.0606	0.9318	1.1718	0.8929	8	1
11	0.6503	0.7647	0.7246	0.9074	0.7633	0.7857	0.6752	0.8865	0.6348	8	1
12	0.9422	1.0672	0.94	1.2573	1.0581	1.0943	0.9558	1.1194	0.9107	8	1
13	0.7041	0.7067	0.6831	0.7394	0.6841	0.6876	0.6495	0.7349	0.5812	8	1
14	0.7257	0.7958	0.7893	0.8597	0.7687	0.7854	0.7415	0.84	0.7113	8	1
15	0.752	0.8222	0.8191	0.9129	0.821	0.8487	0.7734	0.911	0.7098	8	1
16	1.7041	2.2949	1.6727	2.8385	2.2738	2.1906	1.4987	2.4828	1.5217	8	2
17	1.8179	2.3518	2.2904	3.0903	2.5054	2.6275	1.9595	3.1425	1.7021	9	1
18	1.335	1.5224	1.451	1.6007	1.442	1.447	1.2873	1.5266	1.3743	8	3
Avg.	1.1912	1.3844	1.3528	1.5999	1.3359	1.3681	1.2008	1.5347	1.1335	8	1
R$_M$	2	7	5	9	4	6	3	8	1		

TABLE V
S-CIELAB ΔE^* RESULTS ON CMLA DATASET FOR VARIOUS DEMOSAICKING METHODS

Im#	LSLCD	CDDFW	EDAEP	ESF	MSG	GD	VDI	PID$_{WR}$	PID	R$_{WR}$	R$_P$
1	1.6366	1.3848	1.2673	1.4721	1.3014	1.329	1.1924	1.3729	1.2393	6	2
2	1.9195	1.5179	1.4491	1.569	1.4318	1.4356	1.346	1.4921	1.3354	6	1
3	1.6503	1.0097	0.9311	1.255	1.0557	1.0979	0.9172	1.1179	0.8667	7	1
4	0.7588	0.348	0.3723	0.4022	0.3475	0.369	0.3409	0.4068	0.2985	8	1
5	2.4966	2.0371	1.7959	2.1799	1.9475	1.9241	1.7065	2.0105	1.7355	6	2
6	1.3464	0.6307	0.5954	0.7916	0.6358	0.6283	0.5106	0.7765	0.4355	7	1
7	0.4178	0.3004	0.3018	0.3112	0.2964	0.2968	0.2968	0.3057	0.2865	7	1
8	0.7708	0.4296	0.4268	0.4737	0.4237	0.4346	0.3947	0.4454	0.3809	7	1
9	0.4392	0.2849	0.2824	0.3105	0.2845	0.2922	0.2777	0.303	0.2578	7	1
10	0.4454	0.2949	0.2986	0.3163	0.2892	0.295	0.2971	0.3067	0.2933	7	2
11	0.9293	0.7544	0.6954	0.7599	0.7013	0.7044	0.6556	0.7245	0.703	6	4
12	0.953	0.6747	0.6681	0.6989	0.651	0.6562	0.6357	0.6736	0.6188	6	1
13	1.8882	1.2832	1.2897	1.3431	1.2059	1.2156	1.1843	1.2604	1.2066	5	3
14	1.1072	0.7715	0.7616	0.8382	0.7319	0.7531	0.6993	0.7825	0.6844	7	1
15	2.5457	1.8231	1.8012	1.9379	1.7375	1.7842	1.662	1.8307	1.5524	7	1
16	1.1079	0.8801	0.884	0.9163	0.8168	0.8302	0.7927	0.893	0.7736	7	1
Avg.	1.2758	0.9016	0.8638	0.9735	0.8661	0.8779	0.8068	0.9189	0.7918	7	1
R$_M$	9	6	3	8	4	5	2	7	1		

TABLE VI
FSIM RESULTS ON MCM DATASET FOR SEVERAL DEMOSAICKING METHODS

Im#	LSLCD	CDDFW	EDAEP	ESF	MSG	GD	VDI	PID$_{WR}$	PID	R$_{WR}$	R$_P$
1	0.9921	0.99208	0.99287	0.99066	0.99269	0.99258	0.99363	0.99124	0.99396	8	1
2	0.99635	0.9964	0.99668	0.99603	0.99666	0.99667	0.99678	0.99619	0.99694	8	1
3	0.99614	0.99756	0.99698	0.99734	0.99784	0.99777	0.99745	0.9975	0.99724	4	7
4	0.99729	0.9986	0.99822	0.99852	0.99884	0.9988	0.99871	0.9985	0.99888	7	1
5	0.99678	0.99653	0.99708	0.99602	0.99684	0.99689	0.9974	0.9961	0.99832	8	1
6	0.99802	0.99793	0.99838	0.9971	0.99811	0.99809	0.99851	0.99748	0.99891	8	1
7	0.99713	0.99879	0.99775	0.99869	0.99882	0.99867	0.99778	0.9988	0.9975	2	8
8	0.99781	0.99767	0.99783	0.99777	0.99788	0.99801	0.99804	0.99782	0.99761	5	9
9	0.99737	0.99742	0.99752	0.99667	0.99728	0.99733	0.99789	0.99662	0.99798	9	1
10	0.99815	0.99866	0.99844	0.99815	0.99857	0.99839	0.99858	0.99809	0.99884	9	1
11	0.9983	0.99845	0.99846	0.99799	0.99845	0.99839	0.99844	0.99801	0.99876	8	1
12	0.99827	0.99843	0.99833	0.99807	0.99846	0.99842	0.99838	0.99818	0.99865	8	1
13	0.99829	0.99783	0.9983	0.99773	0.99804	0.99816	0.99845	0.99776	0.99879	8	1
14	0.99831	0.99788	0.99823	0.99766	0.99799	0.99803	0.99855	0.99767	0.99871	8	1
15	0.99789	0.99804	0.99799	0.99777	0.99804	0.99799	0.99807	0.99767	0.99834	9	1
16	0.99709	0.99654	0.99768	0.99542	0.99695	0.99721	0.9979	0.99631	0.99814	8	1
17	0.99432	0.99543	0.99568	0.99358	0.99532	0.99506	0.99596	0.99364	0.99735	8	1
18	0.99759	0.99786	0.9977	0.99766	0.99803	0.99804	0.99802	0.99781	0.99806	6	1
Avg.	0.99707	0.99734	0.99745	0.99682	0.99749	0.99747	0.9977	0.99697	0.99794	8	1
R$_M$	7	6	5	9	3	4	2	8	1		

TABLE VII
FSIM RESULTS ON CMLA DATASET FOR SEVERAL DEMOSAICKING METHODS

Im#	LSLCD	CDDFW	EDAEP	ESF	MSG	GD	VDI	PID$_{WR}$	PID	R$_{WR}$	R$_P$
1	0.99294	0.99611	0.99652	0.99568	0.99641	0.99643	0.99687	0.99616	0.99622	6	5
2	0.99192	0.99522	0.99534	0.99514	0.99548	0.9956	0.99563	0.99548	0.99493	3	8
3	0.99253	0.99812	0.99752	0.99756	0.99804	0.99789	0.99754	0.99771	0.99767	4	5
4	0.99758	0.99919	0.99899	0.99903	0.99912	0.99907	0.99909	0.99898	0.99938	8	1
5	0.9877	0.99353	0.99403	0.99304	0.99391	0.9941	0.99433	0.99378	0.99395	6	4
6	0.9955	0.99906	0.9992	0.99865	0.99903	0.99909	0.99935	0.99877	0.99943	7	1
7	0.99789	0.9988	0.9988	0.9988	0.99883	0.99886	0.99878	0.9988	0.99883	4	2
8	0.99727	0.99886	0.99887	0.99882	0.99891	0.99889	0.9988	0.99888	0.99887	3	5
9	0.99832	0.99893	0.99897	0.99888	0.99896	0.99896	0.99896	0.99891	0.99903	7	1
10	0.99788	0.99899	0.99901	0.99893	0.99902	0.99902	0.99902	0.99896	0.99903	7	1
11	0.99404	0.99639	0.99678	0.99635	0.99669	0.99672	0.99689	0.99664	0.99632	5	8
12	0.99543	0.99698	0.99693	0.99703	0.99714	0.99715	0.9969	0.99711	0.9971	3	4
13	0.99135	0.99608	0.99602	0.99604	0.99636	0.99645	0.99636	0.99629	0.99562	4	8
14	0.99375	0.99735	0.99729	0.99728	0.99756	0.9976	0.99762	0.99745	0.99724	4	8
15	0.98966	0.99577	0.99522	0.9952	0.99583	0.99574	0.99574	0.99555	0.99583	6	2
16	0.99377	0.99705	0.99696	0.99698	0.99723	0.9973	0.99719	0.99715	0.9966	4	8
Avg.	0.99422	0.99728	0.99728	0.99709	0.99741	0.99743	0.99744	0.99729	0.99725	4	7
R$_M$	9	6	5	8	3	2	1	4	7		

ble III. The PID method outperformed the tested methods with most CMLA images except #2, #10-14, and #16. The average CPSNR achieved by the proposed method was 0.463 dB and 0.366 dB higher than those of GD and VDI, respectively, which were the second and third best performing methods.

For color images, human vision system-based S-CIELAB ΔE^* metric may be more appropriate than CPSNR metric. S-CIELAB ΔE^* measure is a spatial extension of the CIELAB ΔE^* and is one of the most effective measures considering the human visual system in order to determine the difference between the S-CIELAB ΔE^* representation of an original image and that of the reconstructed image. In terms of S-CIELAB ΔE^* metric on McM dataset, PID yields better performance on average than all the other reference images (-0.0577, -0.2509, -0.2193, -0.4664, -0.2024, -0.2346, -0.0673, -0.4012) as shown in Table IV. We note that LSLCD and VDI are the second the third best methods in terms of S-CIELAB ΔE^* measure.

For assessing S-CIELAB ΔE^* metric on CMLA dataset, as shown in Table V, PID outperformed all the existing methods with -0.484, -0.1098, -0.072, -0.1817, -0.0743, -0.0861, -0.015, and -0.1271, on average. The second and the third best methods for CMLA were VDI and EDAEP, respectively.

To strongly verify the performance of our method, we also used a third metric, the FSIM index. We note that FSIM results close to 1.0 indicate better image quality. Tables VI and VII show FSIM results on McM and CMLA dataset for several demosaicking methods. It can be found from Table VI that, PID outperformed all the other methods with 0.00087 (LSLCD), 0.0006 (CDDFW), 0.00049 (EDAEP), 0.00112 (ESF), 0.00045 (MSG), 0.00047 (GD), 0.00024 (VDI), 0.00097 (PID$_{WR}$), on average. PID provides the best FSIM results for all McM images, except #3, #7, and #8. However, PID did not outperform VDI (-0.00015), GD (-0.00014), and MSG (-0.00012) on average. Although FSIM results of PID ranked 4^{th} for CMLA images, the difference with VDI was very close.

We now compare the performance of the PID method against the other methods in terms of computation time. The speed of demosaicking was assessed as the measured average

TABLE VIII
AVERAGE CPU TIME(S) TO DEMOSAIC ONE 500×500 IMAGE OF THE MCM DATASET AND 704×469 CMLA DATASET

Im#	LSLCD	CDDFW	EDAEP	ESF	MSG	GD	VDI	PID$_{WR}$	PID	R$_{WR}$	R$_P$
McM	0.3498	9.8442	5.5962	10.435	8.8071	3.9116	5.1609	3.64	5.0708	2	4
CMLA	0.4211	12.979	6.4433	13.1322	11.1902	4.4394	6.5502	4.0831	5.8595	2	4
Avg.	0.3855	11.4116	6.0198	11.7836	9.9987	4.1755	5.8556	3.8615	5.4652	2	4
R$_M$	1	8	6	9	7	3	5	2	4		

TABLE IX
PERCENTAGE OF THE PROCESSING TIME FOR PID (%)

	Time (sec)		Percentage of the processing time for PID (%)	
Im#	PID_{WR}	PID	PID_{WR}	Refinement
1	4.4162	6.3903	69.107(%)	30.893(%)
2	6.3198	8.3571	75.622(%)	24.378(%)
3	6.6352	9.0124	73.623(%)	26.377(%)
4	4.9574	7.2701	68.188(%)	31.812(%)
5	4.7056	6.5798	71.516(%)	28.484(%)
6	2.8568	4.3035	66.383(%)	33.617(%)
7	3.1346	4.8201	65.032(%)	34.968(%)
8	3.0910	4.5335	68.182(%)	31.818(%)
9	2.7991	4.2936	65.193(%)	34.807(%)
10	4.5827	6.2355	73.494(%)	26.506(%)
11	3.0252	4.9078	61.640(%)	38.360(%)
12	4.3075	6.3714	67.606(%)	32.394(%)
13	3.7160	5.3206	69.841(%)	30.159(%)
14	3.7795	5.3082	71.201(%)	28.799(%)
15	3.7929	5.3405	71.021(%)	28.979(%)
16	3.2100	4.7077	68.187(%)	31.813(%)
Avg.	4.0831	5.8595	69.683(%)	30.317(%)

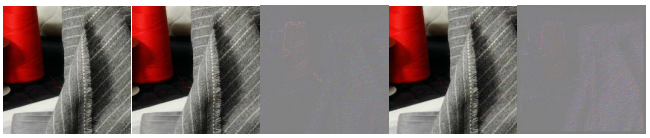
time (in second) required reconstructing a full color image, which are tabulated in Table VIII.

Here, R_{WR}, R_P, R_M stand for rankings of PID without and with refinement, and ranking of CPU time consumption for each method. As we can see in Table VIII, the computation and implementation complexities of the PID method is considerably lower than VDI, MSG, ESF, EDAEP and CDDFW, which are known to provide good quality-speed tradeoffs among conventional methods. The elapsed computation time of PID is 93.3% of VDI, 54.7% of MSG, 46.4% of ESF, 90.8% of EDAEP, and 47.9% of CDDEW. The LSLCD and GD methods only required 7.1% and 76.4% computational time of PID, however the objective performance of the proposed method outperformed these methods.

We also compare complexity in terms of number of operation for LSLCD, EDAEP, and PID methods. The LSLCD is filter-based demosaicking which is processed in frequency domain, thus the speed is fast. To demosaic a pixel using LSLCD, total number of multiplication is 101. The EDAEP method is one of 'inter-channel correlation' and 'edge-directed interpolation' based demosaicking methods. To demosaic a pixel using EDAEP, total number of operations with or without refinement are $(18M + 62A + 23S + 2C)$ or $(2M + 30A + 7S+2C)$, where M, A, S, C are multiplication, addition, shift, and comparison operations. Meanwhile, number of operations to demosaic a pixel with PID with or without refinement is $(16M + 94A + 4S + 4C)$ or $(4M + 46A + 4S + 4C)$.

In addition, we evaluate the computational effort of our approach considering the refinement process separately. Table IX shows the percentage of the processing time for PID (in %). It can be found that refinement step requires 30.3% of the total CPU time, while the PID_{WR} the 69.7%.

From Table IX, it can be found that the processing time of the PID_{WR} and PID were about 69.7% and 30.3%, respectively. As can be seen in Tables II, IV, and VI, refinement process raised performance for most test images. However, for some images, such as #7 and #8 of McM dataset, refinement process was not recommendable and the result images became even worse. Figure 8(a) shows original #7 McM image, and its corresponding PID_{WR} and PID images are shown in Figs. 8(b) and 8(d). The difference between original and the PID_{WR} and



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

Fig. 8. (a) Original image, (b) PID_{WR}, (c) PID, (d) difference between original and PID_{WR} images, and (e) difference between original and PID images.

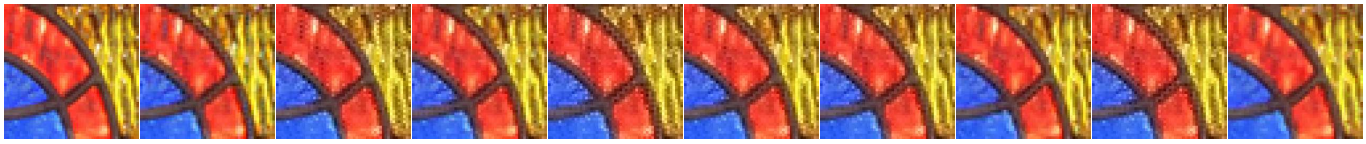
PID result images are shown in Figs. 8(c) and 8(e). It is obvious that the refinement process was helpful for the red yarn area located in upper left. However, refinement process was not beneficial for gray cloth located in the right area. One reason we can consider is that, our setting was deterministic and it caused over-smoothing the image and unwanted artifacts were brought in textile area. However, the refinement process successfully reduced color artifact in yarn area produced by PID_{WR}.

Figures 9-14 illustrate the subjective demosaicking performance using the partial zoomed images of #1, #5 and #17 McM images and #3, #6, and #15 CMLA images. After the mosaicking process using the Bayer pattern, demosaicking was then implemented in order to reconstruct the full color image using the proposed method and the conventional methods mentioned in this section. The visual quality can be appraised with regard to restoration of edges, textures, and various kinds of geometric details such as corners, diagonals, and fine patterns. Images in Figs. 9-14 reveal how each benchmark method fares in restoring the demosaicked images in these difficult regions.

The results of reference methods, such as those using LSLCD, CDDFW, EDAEP, ESF, MSG, and GD, suffer from severe demosaicking artifacts. The VDI method gives images of similar characteristics in terms of visual quality assessment compared with PID appears much smoother than the one obtained demosaicking with VDI, as shown in Figs. 9(h), 10(h), and 11(h).

This visual improvement is more distinct in CMLA dataset. Figures 12 and 14 prove that PID generated the least color artifacts among all compared methods. In addition, edges demosaicked by PID are very similar to the original image, while the other methods still generated unwanted color artifacts. The edge part appears much smoother after demosaicking with PID, as shown in Figs. 12(i), 13(i), and 14(i), especially when compared with most of the other methods. In addition, Figs. 12(b-g), 13(b-g), and 14(b-g) produced salt-and-pepper noise-like artifacts along the edges. When comparing the texture detail preservation ability using McM #1 of stained glass and #17 of flowers and their leaves, the reconstructed texture using PID can be preserved. However, the other methods produced many visible color artifacts like zipper artifacts and false colors along the edges of stained glass and the leaves and flowers where abrupt color changes exist, which can be seen in Figs. 9(b-g), 10(b-g), and 11(b-g).

The LSLCD method is one of the fastest methods and it gives the best performance when it uses ST filters. Its high speed comes from using trained filters, therefore if one uses other filters such as TO and RE filters, performance become

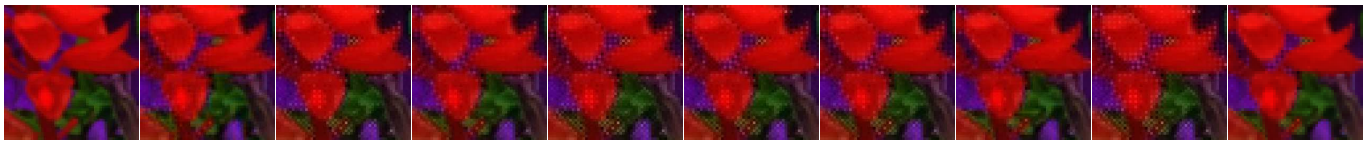


(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 9. Demosaicking results of #1 McM image: (a) Original, (b) LSLCD, (c) CDDFW, (d) EDAEP, (e) ESF, (f) MSG, (g) GD, (h) VDI, (i) PID_{WR}, and (j) PID.

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 10. Demosaicking results of #5 McM image: (a) Original, (b) LSLCD, (c) CDDFW, (d) EDAEP, (e) ESF, (f) MSG, (g) GD, (h) VDI, (i) PID_{WR}, and (j) PID.



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 11. Demosaicking results of #17 McM image: (a) Original, (b) LSLCD, (c) CDDFW, (d) EDAEP, (e) ESF, (f) MSG, (g) GD, (h) VDI, (i) PID_{WR}, and (j) PID.

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 12. Demosaicking results of #3 CMLA image: (a) Original, (b) LSLCD, (c) CDDFW, (d) EDAEP, (e) ESF, (f) MSG, (g) GD, (h) VDI, (i) PID_{WR}, and (j) PID.

worse. As noted in the preceding section, TO filters were used in this paper for LSLCD results and the visual performance was not pleasant. The performance would be even worse if we select RE filters. In addition, LSLCD does not include refinement process, which would be helpful to LSLCD results.

The CDDFW is based on two conventional methods [13] and [17]. Although CDDFW is similar to [17] but CDDFW performed in the color-difference planes and used more effective weights. CDDFW classifies region into two, strong edge and smooth regions, and applies [13] for strong edge region and applies weighted average for smooth region. However, decision on strong and smooth edge region is deterministic which appears unwanted artifacts for wrongly determined region.

As what CDDFW did, EDAEP also has two steps: initial interpolation and refinement process. The initial interpolation is modification of conventional directional interpolation. However, the main drawback of this method is that the refinement process is identical to [42], which is outdated and it is only applied on green pixel values.

The ESF method has three steps: green channel interpolation and its update, and red/blue channel interpolation. In the second step, ESF identifies the regions where constant color difference assumption is likely to fail, and applies edge strength filter to avoid averaging across edge structures. The neighbor of strong edge region neighbor will contribute less to the update result. However, refinement is performed only to the interpolated green pixel values in ESF, which causes red/blue channel artifact.

In MSG, multiscale color gradients were used to adaptively combine color difference estimates from various directions. The main contribution of this method is that it does not make hard decision, and all process is non-iterative. This method is improved version of ESF and its structure is similar. One of main differences is that the edge strength filters were replaced by multiscale color gradients, which were used in initial green channel interpolation. However, MSG method focused on green channel interpolation and update, and red/blue channel reconstruction was restored by 7-by-7 simple filter where directional information was not considered.

The high resolution image and its low resolution one has similar property. GD is based on this idea and calculates reconstruction errors for the possible directions in the low resolution image and exploits them as a cost term for the direction. The edge classifier and the weighted sum were used to restore demosaicking. As the presented filter is discontinuous function, GD only uses large filter over smooth region and uses small filter for edge region. However, GD does not refine red/blue channel which causes artifacts at edge region. In addition, as small filter was used for edge region,

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 13. Demosaicking results of #6 CMLA image: (a) Original, (b) LSLCD, (c) CDDFW, (d) EDAEP, (e) ESF, (f) MSG, (g) GD, (h) VDI, (i) PID$_{WR}$, and (j) PID.

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 14. Demosaicking results of #15 CMLA image: (a) Original, (b) LSLCD, (c) CDDFW, (d) EDAEP, (e) ESF, (f) MSG, (g) GD, (h) VDI, (i) PID$_{WR}$, and (j) PID.

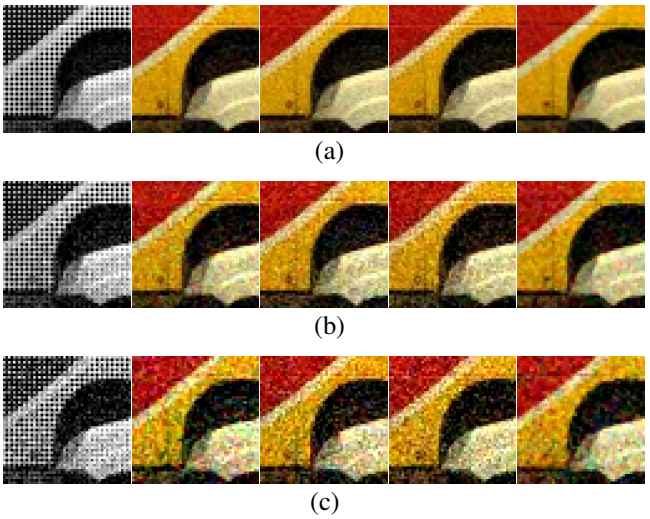
restored detail images are blurred. Moreover, predetermined four direction (0^o, 45^o, 90^o, 135^o) Sobel filters were used for directional differentiation of image. These filters were useful for speed, but two interpolation directions, 45^o and 135^o, were considered the same directions during the process which cause blurred image.

VDI uses color channel correlation to estimate the missing green component, and then applies local adaptive gradient inverse weighted interpolation method to refine and enhance reconstructed image. To investigate the directions of the missing color components, VDI uses flags to classify edge direction. The sum of flags indicates vertical, horizontal, and undefined directions. VDI works well for both edge and complex regions because the estimated edge direction is accurate for edge region and the edge detection is also accurate if the domain edge exists. However, if domain edge is not determined by VDI, the edge direction will be omnidirectional.

The PID method uses error predictor calculation, color difference-based edge classification, and a refinement stage using a weighted sum strategy. Following green channel refinement, the red and blue channels were updated. From the above objective and subjective evaluations of the proposed method, it can be concluded that PID has advantages because it better preserved edge details and created fewer artifacts when compared with conventional methods. Moreover, the proposed PID algorithm appears to yield visually more friendly color images with color artifacts well removed, consistent with the objective quality measures.

TABLE X
AVERAGE PERFORMANCE RESULTS FOR CMLA IMAGESET WITH METRICS OF PSNR (M_1), S-CIELAB ΔE^* (M_2), AND FSIM (M_3)

	σ	LSLCD	CDDFW	EDAEP	ESF	MSG	GD	VDI	PID$_{WR}$	PID	R$_{WR}$	R$_P$
M_1	0	34.09	37.306	37.425	36.954	37.624	37.629	37.726	37.272	38.092	7	1
	10	27.145	27.962	28.169	27.869	28.134	28.224	28.138	28.177	29.442	3	1
	20	22.276	22.778	22.99	22.667	22.919	23.018	22.939	23.007	24.354	3	1
	30	19.166	19.581	19.797	19.454	19.714	19.811	19.743	19.819	21.18	2	1
M_2	0	1.2758	0.9016	0.8638	0.9735	0.8661	0.8779	0.8068	0.9189	0.7918	7	1
	10	3.0454	2.8261	2.9827	2.8641	2.7654	2.8102	3.0006	2.7765	2.8144	2	4
	20	5.6094	5.3743	5.7572	5.3767	5.2932	5.3516	5.8251	5.251	5.4302	1	6
	30	8.7121	8.3146	8.902	8.2976	8.1693	8.2584	9.0077	8.0861	8.3974	1	6
M_3	0	0.99422	0.99728	0.99728	0.99709	0.99741	0.99743	0.99744	0.99729	0.99725	4	7
	10	0.92912	0.9291	0.92609	0.92894	0.92971	0.92948	0.92491	0.92975	0.93336	2	1
	20	0.83509	0.83435	0.82701	0.83353	0.83473	0.83327	0.82531	0.83428	0.83926	5	1
	30	0.75545	0.75482	0.74505	0.75552	0.7556	0.75305	0.74279	0.75507	0.75895	5	1



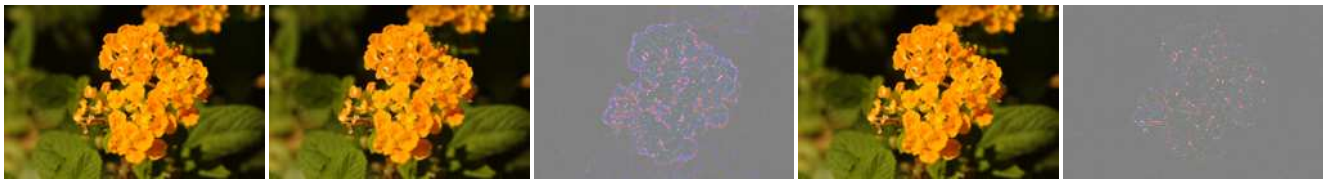
(a) (b) (c)

Fig. 17. Visual performance comparison of EADEP, GD, PID$_{WR}$, and PID for AWGN added #15 CMLA image: (a) $\sigma = 10$, (b) $\sigma = 20$, and (c) $\sigma = 30$.

D. Effect of Refinement Process in Various AWGN Condition

In the Section V.C we claimed that the proposed refinement step is able to improve the final results. To support its reliability, we show some examples when the refinement step improves the result and when refinement step is not available. Figures 15 and 16 show good and bad cases of refinement process, where original images, PID$_{WR}$ and PID, and their corresponding color difference are shown. Figure 15 shows an example of good case where color difference significantly decreased after the refinement process. However, Fig. 16 shows an opposite case where color difference slightly increased after refinement process. Therefore, it can be concluded that refinement process mostly raises image quality, but not always.

We also investigated the effect of refinement process along with different level of AWGN ($\sigma = 10, 20, 30$). Table X shows average PSNR (M_1), S-CIELAB ΔE^* (M_2), and FSIM (M_3) results for CMLA imageset. It can be observed that PID always yields the best CPSNR regardless AWGN noise level. However, in terms of S-CIELAB ΔE^*, refinement process degrades the performance as AWGN noise level increases. On the other hand, FSIM metric showed opposite results of S-



(a) (b) (c)

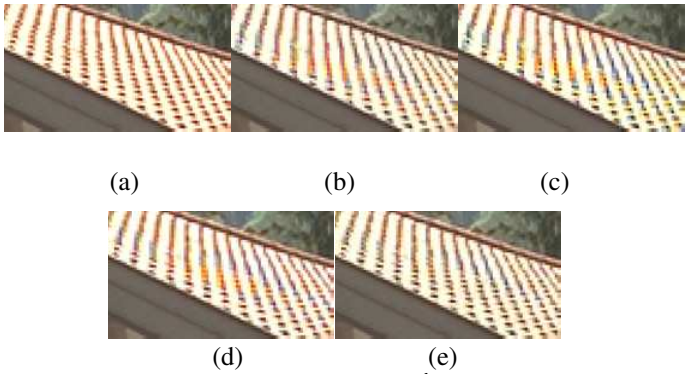
Fig. 15. Visual performance comparison on #6 CMLA image for good case with and without refinement process: (a) original, (b) PID_{WR} and difference with original, and (c) PID and difference with original.

(a) (b) (c)

Fig. 16. Visual performance comparison on #13 CMLA image for good case with and without refinement process: (a) original, (b) PID_{WR} and difference with original, and (c) PID and difference with original.

CIELAB ΔE^*. When AGWN is 0, PID was ranked in 7. However, as AWGN increases, PID gave the best performance. Figure 17 shows visual performance comparison for #15 CMLA image.

In addition, we also found that M_2 results after refinement process on noisy images were worse than that of M_2 results without refinement process. However, refinement process was helpful for M_2 under noise-free condition. On the other hand, for M_3 metric, refinement process was helpful when CFA images have noise. However, if the images are noise-free, it is recommended not to use refinement process. For M_1 metric, refinement process was always helpful to raise the performance.



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

Fig. 18. Visual performance comparison on 24^{th} Kodak image: (a) Original, (b) CSD, (c) AICD, (d) MLRI, and (e) PID.

E. Performance Comparison of Other Types of Methods on Kodak Dataset

The classic Kodak database is widely used to easily compare demosaicking methods. We compare three other types of methods with the proposed one: contour stencil-based demosaicking (CSD) [43], adaptive inter-channel correlation-based demosaicking (AICD) [44], and minimized-Laplacian residual interpolation (MLRI) [45]. There are many demosaicking approaches which adapt the interpolation in accord with obtained edge orientations. However, while the preciseness of edge orientations is significant, correct orientation is hard to obtain from the CFA image.

The CSD extends contour stencils to compute image contours based on total variation along curves and to obtain contour orientations directly on CFA images. The color restoration is achieved as an energy minimization, using a graph regularization adapted according to the orientation estimates. The energy function regularizes the luminance to suppress zipper artifacts and at the same time it regularizes the chrominance to suppress color artifacts. The AICD considers inter-channel correlation and locally selects the best interpolation orientation. The AICD uses nonlocal image self-similarity in order to decrease interpolation artifacts when local geometry is obscure. The AICD has two steps: a local directional interpolation and a nonlocal filtering based in NL-means process. In the preceding step, demosaicked image is populated by deciding a posteriori among a set of local directionally restored images. In the later step, a patch based algorithm takes advantage of image similarities to refine the locally restored image. The MLRI utilizes residual differences, which are differences between observed and tentatively estimated pixel values. The temporary estimate is produced by upsampling the observed pixel values by using the guided filter. The MLRI estimates the temporary pixel values by minimizing the Laplacian energies of the residuals.

Figure 18 tests the methods on a 24^{th} Kodak image (cropped area of roof), where strong color discontinuities and curving boundaries exist. Most conventional methods produce significant yellow and purple color artifacts. Figures 18(b-d) compare how conventional methods handle texture details with a crop from the roof image. The texture details and complicated geometry of the roof pattern make this simulation hard. The roof image includes a pattern approaching the Nyquist limit so that restoration critically aliases the red and blue channels. On the other hand, the PID method shows less aliasing than the other methods, and performs well for such images with limited color artifacts.

TABLE XI
AVERAGE PERFORMANCE RESULTS FOR KODAK IMAGESET WITH METRICS OF PSNR (M_1), S-CIELAB ΔE^* (M_2), AND FSIM (M_3)

Metric	M_1				M_2				M_3			
Method	CSD	AICD	MLRI	PID	CSD	AICD	MLRI	PID	CSD	AICD	MLRI	PID
1	***40.574***	35.724	36.779	39.685	***0.8984***	1.3565	1.2032	0.9825	***0.99924***	0.99716	0.99826	0.99896
2	40.775	39.284	40.747	***41.006***	***0.5931***	0.771	0.6615	0.6461	***0.99898***	0.99742	0.99832	0.99843
3	42.867	42.013	***42.9***	42.537	0.4926	0.5305	***0.4706***	0.4932	0.99926	0.99859	***0.99927***	0.99924
4	40.649	39.715	***41.423***	40.702	0.6862	0.7457	***0.6406***	0.6999	***0.99931***	0.99846	0.99917	0.9992
5	37.597	36.503	37.624	***38.151***	1.3063	1.1874	1.1107	***1.0582***	***0.99896***	0.99829	0.99884	0.99895
6	39.84	37.01	39.084	***41.103***	0.8785	1.0299	0.8347	***0.7106***	0.99912	0.99768	0.99891	***0.9993***
7	42.024	41.432	***42.76***	42.308	0.6042	0.5942	***0.5364***	0.5675	0.99951	0.99902	0.99951	***0.99951***
8	36.338	34.566	34.869	***37.129***	1.2599	1.5029	1.4597	***1.1791***	***0.99854***	0.99727	0.99769	0.99851
9	42.415	41.111	42.298	***43.236***	0.6163	0.6653	0.5772	***0.5459***	0.99928	0.9985	0.9991	***0.99931***
10	42.119	41.027	42.352	***42.982***	0.603	0.629	0.5554	***0.5279***	0.99927	0.99856	0.99919	***0.99931***
11	39.816	37.729	39.256	***40.734***	0.8338	0.9215	0.7695	***0.6929***	0.99909	0.99775	0.99876	***0.9991***
12	***43.946***	41.576	43.175	43.852	0.4615	0.5446	0.4794	***0.4457***	***0.99936***	0.99825	0.99913	0.99923
13	***36.062***	32.091	33.12	35.814	1.5149	2.0829	1.8216	***1.4479***	***0.99869***	0.99529	0.99691	0.99839
14	37.207	35.99	***37.574***	36.585	1.0133	1.1591	***0.9364***	1.0038	***0.99892***	0.99787	0.9987	0.99857
15	39.281	38.898	39.306	***39.65***	***0.6464***	0.6558	0.6829	0.6737	0.99847	0.99751	0.9983	***0.99869***
16	43.597	40.35	42.79	***44.594***	0.5795	0.7151	0.5628	***0.4999***	0.99946	0.99829	0.9993	***0.99952***
17	41.168	39.482	40.919	***42.276***	0.5859	0.6705	0.5713	***0.4971***	0.99942	0.99861	0.99927	***0.99943***
18	36.626	35.496	36.587	***37.953***	1.1669	1.2426	1.0518	***0.9991***	0.99818	0.99695	0.99785	***0.99865***
19	40.402	38.63	39.912	***41.794***	0.8186	0.919	0.7853	***0.6862***	0.99928	0.99818	0.99899	***0.9993***
20	41.235	39.816	40.65	***41.726***	0.5758	0.6141	0.675	***0.5199***	0.99914	0.99849	0.99909	***0.99918***
21	39.996	37.017	38.141	***40.017***	0.9251	1.0651	0.962	***0.8402***	***0.99892***	0.9974	0.99841	0.99883
22	37.953	36.922	38.568	***38.668***	1.08	1.1515	***0.914***	0.9726	0.99841	0.99717	***0.99846***	0.99843
23	42.607	41.817	***43.706***	42.737	0.5382	0.5752	***0.4709***	0.5334	0.99934	0.99891	***0.99936***	0.99924
24	34.409	33.432	34.709	***35.802***	1.2765	1.321	1.143	***0.9895***	0.99725	0.99562	0.99731	***0.99788***
Avg.	39.979	38.235	39.552	***40.46***	0.8315	0.9438	0.8282	***0.7589***	***0.99897***	0.9978	0.99867	0.99896

Table XI shows the PSNR, S-CIELAB ΔE^*, and FSIM results over the Kodak dataset. The ***bold-italic*** entries indicate either the highest CPSNR or FSIM, or the smallest S-CIELAB in each row. The table shows that while the proposed method is relatively weak on FSIM (CSD is the best for this metric), it has the best performance on the PSNR and S-CIELAB ΔE^* metrics.

VI. CONCLUSION

In this paper, a polynomial interpolation-based demosaicking (PID) method was proposed. The proposed method has three contributions, (1) predictors that consider error, (2) an edge classifier based on color differences, and (3) a refinement scheme using a weighted sum strategy. This method generates directional predictors based on the polynomial interpolation and combines them according to the proposed edge classifier that uses the color difference model. After interpolating three color channels, the proposed refinement was applied to enhance the demosaicked image quality and reduce demosaicking artifacts, such as false color.

Although we used three training sets with different characteristics, determined threshold parameter τ was not very different ($\tau = 1.8, 1.9$, and 2.0 for LC, Zahra, and Stanford dataset, respectively). McM and CMLA datasets were used as test datasets. Simulation results confirmed that our method outperforms the tested existing demosaicking methods in terms of objective (CPSNR and S-CIELAB ΔE^*) and subjective performance. For FSIM metric comparison, the proposed method was not able to provide the best performance for CMLA dataset, while it gives the best performance for McM dataset. We also compare other type of demosaicking methods on widely used Kodak database.

Further work is underway to better deal with FSIM performance on CMLA dataset. In addition, we also will investigate the reason why PID gives 4.327 dB worse CPSNR performance than reference method MSG.

REFERENCES

[1] J. Adams, K. Parulski, and K. Spaulding, "Color processing in digital cameras," *IEEE micro*, vol. 18, no. 6, pp. 20–30, Nov. 1998.

[2] B. E. Bayer, *Color Imaging Array*, U.S. patent 3 971 065, 1976.

[3] S. J. Robinson and J. T. Schmidt, "Fluorescent penetrant sensitivity and removability - what the eye can see, a fluorometer can measure," *Materials Evaluation*, vol. 42, no. 8, pp. 1029–1034, July 1984.

[4] E. F. Schubert, *Light Emitting Diodes*, 2nd Ed., Cambridge University Press, 2006.

[5] D. Menon and G. Calvagno, "Color image demosaicking: an overview," *Signal Process. Image Commun.*, vol. 26, no. 8-9, pp. 518–533, 2011.

[6] X. Li, B. K. Gunturk, and L. Zhang, "Image demosaicing: a systematic survey," in *Proc. SPIE VCIP*, vol. 6822, pp. 68221J, San Jose, CA, January 2008.

[7] D. R. Cok, *Signal Processing Method andApparatus for Producing Interpolated Chrominance Values in a Sampled Color Image Signal*, U.S. Patent 4 642 678, 1987. Eastman Kodak Company.

[8] R. Lukac and K.N. Plataniotis, "A normalized model for color-ratio based demosaicking schemes," in *Proc. ICIP*, vol. 3, Oct. 2004, pp. 1657–1660.

[9] J. E. Adams, "Intersections between color plane interpolation and other image processing functions in electronic photography," in *Proc. SPIE*, vol. 2416, pp. 144–151, 1995.

[10] W. Lu and Y.-P. Tan, "Color filter array demosaicking: new method and performance measures," *IEEE Trans. Image Process.*, vol. 12, no. 10, pp. 1194–1210, Oct. 2003.

[11] J. Hamilton and J. Adams, *Adaptive Color Plane Interpolation in Single Sensor Color Electronic Camera*, U.S. Patent 5 629 734, 1997.

[12] K.-H. Chung and Y.-H. Chan, "Color demosaicing using variance of color differences," *IEEE Trans. Image Process.*, vol. 15, no. 10, pp. 2944–2955, Oct. 2006.

[13] D. Menon, S. Andriani, and G. Calvagno, "Demosaicing with directional filtering and *a posteriori* decision," *IEEE Trans. Image Process.*, vol. 16, no. 1, pp. 132–141, Jan. 2007.

[14] Z. Dengwen, S. Wxialiu, and Weiming, "Colour demosaicking with directional filtering and weighting," *IET Image Process.*, vol. 6, no. 8, pp. 1084–1092, Nov. 2012.

[15] C.-Y. Su and W.-C. Kao, "Effective demosaicking using subband correlation," *IEEE Trans. Consumer Electron.*, vol. 55, no. 1, pp. 199–204, Feb. 2009.

[16] S.-C. Pei and I.-K. Tam, "Effective color interpolation in CCD color filter arrays using signal correlation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 6, pp. 503–512, June 2003.

[17] L. Chang and Y.-P. Tan, "Effective use of spatial and spectral correlations for color filter array demosaicking," *IEEE Trans. Consumer Electron.*, vol. 50, no. 1, pp. 355–365, Feb. 2004.

[18] W.-J. Chen and P.-Y. Chang, "Effective demosaicking algorithm based on edge property for color filter arrays," *Digit. Signal Process.*, vol. 22, vol. 1, pp. 163–169, Jan. 2012.

[19] I. Pekkucuksen and Y. Altunbasak, "Edge strength filter based color filter array interpolation," *IEEE Trans. Image Process.*, vol. 21, no. 1, pp. 393–397, Jan. 2012.

[20] I. Pekkucuksen and Y. Altunbasak, "Multiscale gradients-based color filter array interpolation," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 157–165, Jan. 2013.

[21] J. Kim, G. Jeon, and J. Jeong, "Demosaicking using geometric duality and dilated directional differentiation," *Optics Communications*, vol. 324, pp. 194–201, 2014.

[22] X. Chen, G. Jeon, and J. Jeong, "Voting-based directional interpolation method and its application to still color image demosaicking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 2, pp. 255–262, Feb. 2014.

[23] X. Wu, W. K. Choi, and P. Bao, "Color restoration from digital camera data by pattern matching," in *Proc. SPIE*, vol. 3018, pp. 12–17, 1997.

[24] W. T. Freeman, *Method and Apparatus for Reconstructing Missing Color Samples*, U.S.Patent, No.4774565, Sep. 1988.

[25] R. Ramanath and W. E. Snyder, "Adaptive demosaicking," *J. Electron. Imaging*, vol. 12, no. 4, pp. 633–642, Oct. 2003.

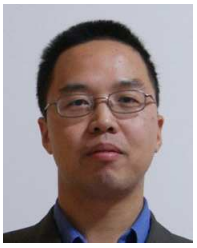
[26] L. Zhang and X. Wu, "Color demosaicking via directional linear minimum mean square-error estimation," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2167–2178, Dep. 2005.

[27] D. Menon and G. Calvagno, "Regularization approaches to demosaicking," *IEEE Trans. Image Process.*, vol. 18, no. 10, pp. 2209–2219, Oct. 2009.

[28] D. Alleysson, S. Susstrunk, and Jeanny Herault, "Linear demosaicking inspired by the human visual system," *IEEE Trans. Image Process.*, vol. 14, no. 4, pp. 439–449 Apr. 2005.
[29] E. Dubois, "Frequency-domain methods for demosaicking of Bayer-sampled color images," *IEEE Signal Process. Lett.*, vol. 12, no. 12, pp. 847–850, Dec. 2005.
[30] E. Dubois, "Filter design for adaptive frequency-domain Bayer demosaicking," in *Proc. ICIP*, pp. 2705–2708, Atlanta, GA, Oct. 2006.
[31] B. Leung, G. Jeon, and E. Dubois, "Least-squares luma-chroma demultiplexing algorithm for Bayer demosaicking," *IEEE Trans. Image Process.*, vol. 20, no. 7, pp. 1885–1894, July 2011.
[32] G. Jeon and E. Dubois, "Demosaicking of noisy Bayer-sampled color images with least-squares luma-chroma demultiplexing and noise level estimation," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 146–156, Jan. 2013.
[33] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, 3rd ed., Springer-Verlag: New York, 2010.
[34] E. Damiani, O. D'Antona, and D. E. Loeb, "The complementary symmetric functions: connection constants using negative sets," *Advances in Mathematics*, vol. 135, no. 2, pp. 207–219, 1998.
[35] L. Zhang, X. Wu, A. Buades, and X. Li, "Color Demosaicking by Local Directional Interpolation and Non-local Adaptive Thresholding," *Journal of Electronic Imaging*, vol. 20, no. 2, pp. 023016, Apr-Jun 2011.
[36] M. Colom, A. Buades, and J.-M. Morel, "Nonparametric noise estimation method for raw images," *J. Opt. Soc. Am. A*, vol. 31, no. 4, pp. 863–871, April 2014.
[37] Available http://www.gipsa-lab.grenoble-inp.fr/ laurent.condat/imagebase.html Accessed 21 May 2014
[38] Z. Sadeghipoor, Y. M. Lu, and S. Susstrunk, "Correlation-based joint acquisition and demosaicing of visible and near-infrared images," in *Proc. IEEE ICIP*, pp. 3165–3168, Sept. 2011.
[39] Available http://www-db.stanford.edu/ wangz/image.vary.jpg.tar Accessed 3 Sept. 2013
[40] X. Zhang and B. A.Wandell, "A spatial extension of CIELAB for digital color image reproduction," *J. Soc. Inf. Display*, vol. 5, no. 1, pp. 61–67, Mar. 1997.
[41] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "FSIM: a feature similarity index for image quality assessment," *IEEE Trans. Image Process.*, vol. 8, no. 8, pp. 2378–2386, Aug. 2011.
[42] C.-Y. Tsai and K.-T. Song, "A new edge-adaptive demosaicing algoritm for color filter arrays," *Image and Vision Computing*, vol. 25, pp. 1495–1508, 2007.
[43] P. Getreuer, "Color demosaicing with contour stencils," in *Proc. IEEE DSP*, July 2011, pp. 1–6.
[44] J. Duran and A. Buades, "Self-similarity and spectral correlation adaptive algorithm for color demosaicking," *IEEE Trans. Image Processing*, vol. 23, no. 9, pp. 4031–4040, Sept. 2014.
[45] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, "Minimized-Laplacian residual interpolation for color image demosaicking," in *Proc. IS&T/SPIE Electronic Imaging*, March 2014, pp. 90230L–90230L.

Jiaji Wu received the BS degree in electrical engineering from Xidian University, Xi'an, China, in 1996, the MS degree from National Time Service Center, the Chinese Academy of Sciences in 2002, and the PhD degree in electrical engineering from Xidian University in 2005. Currently, he is a professor at Xidian University, Xi'an China. His current research interests include image processing, still image coding, hyperspectral/multispectral image compression, communication, and high-performance computing.

Marco Anisetti is an Assistant Professor at the Università degli Studi di Milano. He received the Ph.D. degree in computer science from the Università degli Studi di Milano in 2009. His research interests are in the area of Computational Intelligence and its application to i) the design of complex systems and services , ii) Humanized Computing, Human Machine Interaction and Ambient Intelligence, iii) signal processing. Recently, he has been investigating the adoption of Computational Intelligence techniques in the area of security mechanisms for distributed systems and Big Data assurance.



Wei Wu was born in 1975. He received the B.S. degree from Tianjin University, Tianjin, China, in 1998 and the M.S. and Ph.D. degrees in communication and information systems from Sichuan University, Chengdu, China, in 2003 and 2008, respectively. He is currently an Associate Professor with the College of Electronics and Information Engineering, Sichuan University. His current research interests include image processing and video communications, and super resolution.

Ernesto Damiani is a Full Professor at the Università degli Studi di Milano, where he leads the SESAR research lab, (http://sesar.di.unimi.it), and the leader of the Big Data Initiative at the Etisalat British Telecom Innovation Center EBTIC/Khalifa University in Abu Dhabi, UAE. Ernesto is the Principal Investigator of the H2020 TOREADOR project on Big-Data-as-a-Service funded by the EU Commission on the ICT call on Big Data Research. He was a recipient of the Chester-Sall Award from the IEEE IES Society (2007). He was named ACM Distinguished Scientist (2008) and received the IFIP TC2 Outstanding Contributions Award (2012).

Gwanggil Jeon received the BS, MS, and PhD (summa cum laude) degrees in Department of Electronics and Computer Engineering from Hanyang University, Seoul, Korea,in 2003, 2005, and 2008, respectively. From 2008 to 2009, he was with the Department of Electronics and Computer Engineering, Hanyang University, from 2009 to 2011, he was with the School of Information Technology and Engineering (SITE), University of Ottawa, as a postdoctoral fellow, and from 2011 to 2012, he was with the Graduate School of Science & Technology, Niigata University, as an assistant professor. He is currently an associate professor with the Department of Embedded Systems Engineering, Incheon National University, Incheon, Korea and a full professor with the School of Electronic Engineering, Xidian University, Xi'an, China. His research interests fall under the umbrella of image processing, particularly image compression, motion estimation, demosaicking, and image enhancement as well as computational intelligence such as fuzzy and rough sets theories. He was the recipient of the IEEE Chester Sall Award in 2007, the 2008 ETRI Journal Paper Award, and the Thousand Talents Program for Distinguished Young Scholars by Chinese Government.