Least-Squares Luma-Chroma Demultiplexing Algorithm for Bayer Demosaicking

Brian Leung, *Student Member, IEEE*, Gwanggil Jeon, *Member, IEEE*, and Eric Dubois, *Fellow, IEEE*

***Abstract*— This paper addresses the problem of interpolating missing color components at the output of a Bayer color filter array (CFA), a process known as demosaicking. A luma-chroma demultiplexing algorithm is presented in detail, using a least-squares design methodology for the required bandpass filters. A systematic study of objective demosaicking performance and system complexity is carried out, and several system configurations are recommended. The method is compared with other benchmark algorithms in terms of CPSNR and S-CIELAB ΔE^* objective quality measures, and demosaicking speed. It was found to provide excellent performance, and the best quality-speed tradeoff among the methods studied.**

***Index Terms*— color demosaicking, color filter array, Bayer sampling**

I. Introduction

THE single charge-coupled device (CCD) sensor with color filter array (CFA) combination remains the predominant implementation inside digital cameras for capturing color images [1]. Because of the CFA, only one color component is measured at each CCD sensor element. CFAs vary by the color classes they admit and the arrangement of these color samples. The Bayer array [2], the most commonly used CFA in digital cameras, is a spatially periodic mosaic of red, green, and blue samples where the green samples have twice the sampling density over the red or blue samples. The problem of estimating the missing color components at each spatial location by using only the measured samples is called demosaicking.

The direct approach to demosaicking involves spatial interpolation of the undersampled red, green, and blue components to the full CFA grid. Because each of these components is severely undersampled, this approach yields very visible artifacts, especially false colors and "zippering". Many approaches have been proposed to reduce the level of such artifacts, generally based on exploiting correlation between the three components and using edge-directed interpolation, with varying levels of success. Comprehensive reviews of demosaicking algorithms can be found in [3] and [4] and so will not be repeated here.

Alleysson *et al.* [5] showed that the spatial multiplexing of red, green, and blue color components in a Bayer CFA signal is equivalent to the frequency multiplexing of an achromatic "luma" component and two modulated "chroma" components. Moreover, the luma and chroma components are sufficiently well isolated in the frequency domain to consider constructing demosaicking algorithms from a frequency analysis approach. Such an algorithm involves extracting estimates of the luma and modulated chroma components separately by filtering the Bayer CFA signal using two-dimensional filters with the appropriate passbands, and then converting the estimated luma and two demodulated chroma values at each spatial location into RGB values. However, the algorithm along these lines proposed in [5] did not give state-of-the-art performance.

In [6], Dubois introduced a locally-adaptive luma-chroma demultiplexing algorithm which did give state-of-the-art performance at the time. This algorithm exploits the redundancy of one chroma component in the Bayer CFA signal by locally selecting the better estimate using the copy less corrupted by crosstalk with the luma signal. The work in [7] introduced a least-squares approach for optimal filter design that replaced the window filter design method used in [6]. This new filter design method was successful in producing lower order filters (11×11) that achieved virtually identical demosaicking quality as the higher order filters (21×21) used in [6]. We refer to this method as the least-squares luma-chroma demultiplexing (LSLCD) algorithm. The present paper further examines the tradeoff relationship between demosaicking quality and speed of demosaicking for this algorithm in order to recommend filter specifications for the best system that balances quality and speed. We argue in light of our experimental results that this algorithm gives state-of-the-art performance, especially when computational complexity is considered. Note that the LSLCD formulation has been extended to arbitrary CFA structures in [8].

The rest of this paper is organized as follows. In Section II, the adaptive luma-chroma demultiplexing algorithm is reviewed, and the least-squares filter design method is described in detail. Section III presents the computational complexity analysis of the system, and gives a structured approach to reducing the computational complexity to give the best tradeoff between demosaicking performance and speed. Detailed simulation results and comparisons with five competing methods in terms of objective quality measures and computational time are reported in Section IV. Note that a complete set of demosaicked images can be found on the

Copyright(c) 2010 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Manuscript received April, 2010; revised November, 2010. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

E. Dubois and G. Jeon are with the School of Information Technology and Engineering, University of Ottawa, Ottawa, ON, K1N 6N5, Canada. (email: edubois@uottawa.ca, gjeon@uottawa.ca).

B. Leung is with the Institute of Biomaterials & Biomedical Engineering, University of Toronto, Toronto, ON, M5S 1A1, Canada. (email: brpw.leung@utoronto.ca).

associated web page [9]. Conclusions are drawn in Section V.

II. Demosaicking System Design

A. Demosaicking by Adaptive Luma-Chroma Demultiplexing

The algorithm for Bayer demosaicking by adaptive luma-chroma demultiplexing used in this paper is precisely the one described in [6]; it is summarized here for completeness. We assume that an underlying color image with RGB components f_R, f_G, and f_B is sampled on the rectangular integer lattice $\Lambda = \mathbb{Z}^2$, with the upper left point of the image at coordinate (0,0). The unit of length used in this paper is the vertical spacing between sample elements in the CFA signal, denoted 1 px. The standard *spatial multiplexing model* of the Bayer CFA signal is

$$\begin{aligned} & f_{\text{CFA}}[n_1, n_2] \\ &= f_R[n_1, n_2] m_R[n_1, n_2] + f_G[n_1, n_2] m_G[n_1, n_2] \\ &+ f_B[n_1, n_2] m_B[n_1, n_2] \\ &= \frac{1}{4} f_R[n_1, n_2](1 - (-1)^{n_1})(1 + (-1)^{n_2}) \\ &+ \frac{1}{2} f_G[n_1, n_2](1 + (-1)^{n_1 + n_2}) \\ &+ \frac{1}{4} f_B[n_1, n_2](1 + (-1)^{n_1})(1 - (-1)^{n_2}). \end{aligned} \quad (1)$$

Expanding the expression and collecting like terms, one obtains

$$\begin{aligned} & f_{\text{CFA}}[n_1, n_2] \\ &= (\frac{1}{4} f_R[n_1, n_2] + \frac{1}{2} f_G[n_1, n_2] + \frac{1}{4} f_B[n_1, n_2]) \\ &+ (-\frac{1}{4} f_R[n_1, n_2] + \frac{1}{2} f_G[n_1, n_2] \\ &- \frac{1}{4} f_B[n_1, n_2])(-1)^{n_1 + n_2} \\ &+ (-\frac{1}{4} f_R[n_1, n_2] + \frac{1}{4} f_B[n_1, n_2])((-1)^{n_1} - (-1)^{n_2}) \\ &\triangleq f_L[n_1, n_2] + f_{C1}[n_1, n_2](-1)^{n_1 + n_2} \\ &+ f_{C2}[n_1, n_2]((-1)^{n_1} - (-1)^{n_2}). \end{aligned} \quad (2)$$

This expression offers a different interpretation to the spatial representation of the Bayer CFA signal. Specifically, the CFA is treated as the multiplexing of one baseband signal and two modulated difference signals. The baseband signal f_L identifies an achromatic *luma* component and the two modulated signals f_{C1} and f_{C2} identify two separate chromatic color difference components, referred to here as *chroma* components.

Substituting for $-1 = e^{j\pi}$ in equation (2), one obtains

$$\begin{aligned} f_{\text{CFA}}[n_1, n_2] &= f_L[n_1, n_2] + f_{C1}[n_1, n_2] e^{j\pi(n_1 + n_2)} \\ &+ f_{C2}[n_1, n_2](e^{j\pi n_1} - e^{j\pi n_2}) \\ &\triangleq f_L[n_1, n_2] + f_{C1m}[n_1, n_2] \\ &+ f_{C2ma}[n_1, n_2] + f_{C2mb}[n_1, n_2], \end{aligned} \quad (3)$$

with Fourier transform

$$\begin{aligned} F_{\text{CFA}}(u, v) &= F_L(u, v) + F_{C1}(u - 0.5, v - 0.5) \\ &+ F_{C2}(u - 0.5, v) - F_{C2}(u, v - 0.5), \end{aligned} \quad (4)$$

where frequencies are expressed in c/px.

The key observation of [6] is that the CFA signal contains two separate modulated versions of f_{C2} which suffer differently from crosstalk. Generally, in a local area, one version suffers less from crosstalk than the other, and the adaptive algorithm attempts to identify and favor the less corrupted version to estimate f_{C2}. The adaptive luma-chroma demultiplexing algorithm of [6] is summarized in Fig. 1. Three FIR linear shift-invariant bandpass filters h_1, h_{2a}, and h_{2b} are used to estimate the modulated components as $\hat{f}_{C1m}$, $\hat{f}_{C2ma}$, and $\hat{f}_{C2mb}$ respectively. The design of these filters is one key aspect of this paper. In [6], 21×21 filters were designed using the window method; in this paper, they are designed using a least-squares criterion (as in [7]), and the effect of filter size will be studied. Returning to Fig. 1, the estimated modulated components are demodulated to baseband; $\hat{f}_{C1}$ is retained as is, whereas $\hat{f}_{C2a}$ and $\hat{f}_{C2b}$ are adaptively combined to yield the estimate $\hat{f}_{C2} = w\hat{f}_{C2a} + (1 - w)\hat{f}_{C2b}$. The weighting coefficient is $w = e_Y/(e_X + e_Y)$, where e_X and e_Y represent the average local energy in the vicinity of frequencies $(f_m, 0)$ and $(0, f_m)$. Specifically, h_{G1} is a Gaussian filter with horizontal and vertical standard deviation r_{G1} and r_{G2}, modulated to the frequency $(\pm f_m, 0)$, and h_{G2} is the transpose of the same Gaussian filter, modulated to $(0, \pm f_m)$. Then $e_X = (f_{\text{CFA}} * h_{G1})^2 * h_{MA}$, where h_{MA} is a 5×5 moving average filter, and $e_Y = (f_{\text{CFA}} * h_{G2})^2 * h_{MA}$. In [6], f_m was set to 0.375 c/px, and we could not improve on this value. The standard deviations $(r_{G1}, r_{G2}) = (3.0, 1.0)$ were found to give the best performance. The effect of the Gaussian filter order is studied in this paper. Although this adaptive weighting algorithm is empirical, it gives excellent results, and we have not been able to improve on it.

B. Least-Squares Filter Design

We have seen that the estimate for component $X \in \{\text{C1m}, \text{C2ma}, \text{C2mb}\}$ is obtained by the spatial filtering operation $\hat{f}_X = f_{\text{CFA}} * h_{X'}$ where $X' \in \{1, 2a, 2b\}$ respectively. Suppose we have a model for the original signal f_X so that the difference between $\hat{f}_X$ and f_X can be expressed as a stationary random field. Then, a suitable design criterion is to minimize the expected squared error, resulting in the filter $h_{X'} = \arg\min_h E[(f_X[n_1, n_2] - (f_{\text{CFA}} * h)[n_1, n_2])^2]$, which is independent of (n_1, n_2) due to stationarity. Because good models for f_X do not exist yet, we can instead compile a set of training images from typical color images and compute filters that minimize squared errors over the training set; these filters are the solution to standard least-squares problems [10].

Assume that we have chosen a training set of original RGB color images. Thus, we also have access to the signals f_{C1m}, f_{C2ma} and f_{C2mb}, which are respectively the original baseband signals f_{C1} and f_{C2} modulated to the appropriate centering frequencies. Let us first consider C1 and filter h_1. Recall that the estimate $\hat{f}^{(i)}_{C1m}$ for the i^{th} training image is obtained by $\hat{f}^{(i)}_{C1m} = f^{(i)}_{\text{CFA}} * h_1$. If h_1 has region of support $\mathcal{B}$ that is a subset of the i^{th} image sampling raster Λ_i, then

$$\hat{f}^{(i)}_{C1m}[n_1, n_2] = \sum_{[k_1, k_2] \in \mathcal{B}} h_1[k_1, k_2] f^{(i)}_{\text{CFA}}[n_1 - k_1, n_2 - k_2]. \quad (5)$$

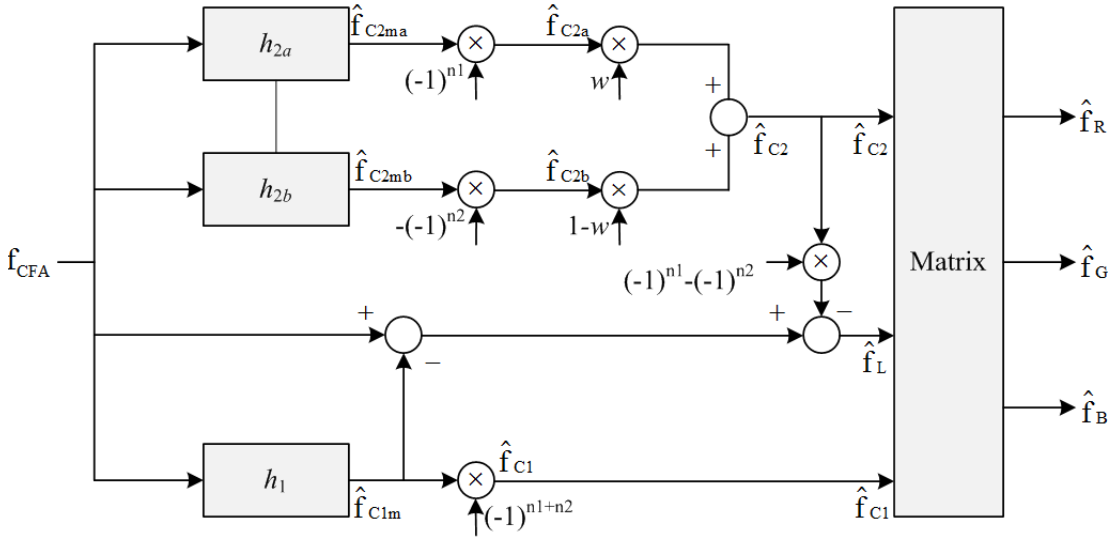


Fig. 1. Block diagram of adaptive luma-chroma demultiplexing algorithm for the Bayer CFA structure.

We define the *total squared error* (TSE) on C1 over every pixel in a training set of K images by

$$\text{TSE}_{C1} = \sum_{i=1}^{K} \sum_{[n_1,n_2]\in\Lambda_i} (\hat{f}_{C1m}^{(i)}[n_1,n_2] - f_{C1m}^{(i)}[n_1,n_2])^2. \quad (6)$$

The least-squares filter h_1^* that minimizes the estimation error on C1 is the solution to the least-squares problem

$$h_1^* = \arg\min_{h_1} \text{TSE}_{C1}. \quad (7)$$

We can reformulate the least-squares problem using matrices. Let $N_B = |\mathcal{B}|$ be the number of h_1 filter coefficients and let $N_W = |\Lambda_i|$ be the number of pixels in the i^{th} training image. Assume for now that N_W is the same for every training image. We may reshape $f_{C1m}^{(i)}$ into a $N_W \times 1$ column vector $\mathbf{f}_{C1m}^{(i)}$ by scanning $f_{C1m}^{(i)}$ column-by-column over Λ_i. Now, reshape h_1 into a $N_B \times 1$ column vector $\mathbf{h}_1$ by scanning h_1 column-by-column over $\mathcal{B}$. Finally, construct a $N_W \times N_B$ matrix $\mathbf{A}^{(i)}$ by scanning $f_{\text{CFA}}^{(i)}$ in alignment with $\mathbf{h}_1$ such that each entry of the matrix product $\mathbf{A}^{(i)}\mathbf{h}_1$ realizes equation (5). The result of $\mathbf{A}^{(i)}\mathbf{h}_1$ is the $N_W \times 1$ column vector $\hat{\mathbf{f}}_{C1m}^{(i)}$ aligned pixel-wise with $\mathbf{f}_{C1m}^{(i)}$. These matrices reformulate equation (7) into

$$\begin{aligned}\mathbf{h}_1^* &= \arg\min_{\mathbf{h}_1} \sum_{i=1}^{K} \|\hat{\mathbf{f}}_{C1m}^{(i)} - \mathbf{f}_{C1m}^{(i)}\|^2 \\ &= \arg\min_{\mathbf{h}_1} \sum_{i=1}^{K} \|\mathbf{A}^{(i)}\mathbf{h}_1 - \mathbf{f}_{C1m}^{(i)}\|^2, \end{aligned} \quad (8)$$

which is a standard least-squares problem with solution

$$\mathbf{h}_1^* = \left[\sum_{i=1}^{K} \mathbf{A}^{(i)^T}\mathbf{A}^{(i)}\right]^{-1} \left[\sum_{i=1}^{K} \mathbf{A}^{(i)^T}\mathbf{f}_{C1m}^{(i)}\right]. \quad (9)$$

Finally, we reshape $\mathbf{h}_1^*$ back onto support $\mathcal{B}$ to get the least-squares filter h_1^*.

The same framework is used on C2 to obtain the least-squares filters h_{2a} and h_{2b} defined over supports $\mathcal{D}$ and $\mathcal{D}'$ (where $\mathcal{D}'$ is the transpose of $\mathcal{D}$). Here, we have

$$\text{TSE}_{C2} = \sum_{i=1}^{K} \sum_{[n_1,n_2]\in\Lambda_i} (\hat{f}_{C2m}^{(i)}[n_1,n_2] - f_{C2m}^{(i)}[n_1,n_2])^2 \quad (10)$$

with the adaptive estimate (see Fig. 1)

$$\begin{aligned}\hat{f}_{C2m}^{(i)}[n_1,n_2] = {} & w_i[n_1,n_2](-1)^{n_1}\times \\ & \sum_{[k_1,k_2]\in\mathcal{D}} (h_{2a}[k_1,k_2] f_{\text{CFA}}^{(i)}[n_1-k_1, n_2-k_2]) \\ & - (1-w_i[n_1,n_2])(-1)^{n_2}\times \\ & \sum_{[k_1,k_2]\in\mathcal{D}'} (h_{2b}[k_1,k_2] f_{\text{CFA}}^{(i)}[n_1-k_1, n_2-k_2]). \end{aligned} \quad (11)$$

The set of weighting coefficients w_i is obtained in the same manner described previously. The sets w_i and $(1-w_i)$ are modulated accordingly to match the centering frequencies of f_{C2ma} and f_{C2mb} respectively.

As before, we cast the least-squares problem into matrix form. Furthermore, we can simultaneously find the least-squares filters h_{2a}^* and h_{2b}^* by temporarily merging the two filter kernels. Once again, let $N_W = |\Lambda_i|$ be the number of pixels in the i^{th} training image and assume N_W to be the same for all images. Let $N_D = |\mathcal{D}|$ be the number of h_{2a} (or h_{2b}) filter coefficients. First, reshape $f_{C2m}^{(i)}$ into a $N_W \times 1$ column vector $\mathbf{f}_{C2m}^{(i)}$ by scanning $f_{C2m}^{(i)}$ column-by-column over Λ_i. Next, reshape h_{2a} and h_{2b} into two $N_D \times 1$ column vectors $\mathbf{h}_{2a}$ and $\mathbf{h}_{2b}$ respectively by scanning column-by-column and then stack $\mathbf{h}_{2a}$ over $\mathbf{h}_{2b}$ to form the $2N_D \times 1$ column vector

$$\mathbf{h}_2 = \begin{bmatrix} \mathbf{h}_{2a} \\ \mathbf{h}_{2b} \end{bmatrix}.$$

Finally, construct a $N_W \times 2N_D$ matrix $\mathbf{B}^{(i)}$ by scanning the product values of $f_{\text{CFA}}^{(i)}$ and modulated weighting coefficients in alignment with $\mathbf{h}_2$ such that each entry of the matrix

product $\mathbf{B}^{(i)}\mathbf{h}_2$ realizes equation (11). The matrix product is the $N_W \times 1$ column vector $\hat{\mathbf{f}}_{C2m}^{(i)}$ aligned pixel-wise with $\mathbf{f}_{C2m}^{(i)}$. With these matrices we can express the standard least-squares problem on C2 as

$$\begin{aligned}\mathbf{h}_2^* &= \arg\min_{\mathbf{h}_2} \sum_{i=1}^{K} \|\hat{\mathbf{f}}_{C2m}^{(i)} - \mathbf{f}_{C2m}^{(i)}\|^2 \\ &= \arg\min_{\mathbf{h}_2} \sum_{i=1}^{K} \|\mathbf{B}^{(i)}\mathbf{h}_2 - \mathbf{f}_{C2m}^{(i)}\|^2, \end{aligned} \tag{12}$$

with solution

$$\mathbf{h}_2^* = \left[\sum_{i=1}^{K} \mathbf{B}^{(i)^T}\mathbf{B}^{(i)}\right]^{-1} \left[\sum_{i=1}^{K} \mathbf{B}^{(i)^T}\mathbf{f}_{C2m}^{(i)}\right]. \tag{13}$$

Now extract $\mathbf{h}_{2a}^*$ from the first N_D entries of $\mathbf{h}_2^*$ and $\mathbf{h}_{2b}^*$ from remaining entries and then reshape $\mathbf{h}_{2a}^*$ and $\mathbf{h}_{2b}^*$ separately back onto supports $\mathcal{D}$ and $\mathcal{D}'$ to get the least-squares filters h_{2a}^* and h_{2b}^*.

We have relied on the assumption that each training image has the same number of pixels. Although this may not be true in general, we can enforce this assumption by dividing each training image into sub-images of the same dimensions. Then, the sub-image size N_W is constant for each piece and we train over all sub-images instead. In [7] the sub-image window has dimensions 96 pixels $\times$ 96 pixels giving $N_W = 9216$ pixels. The choice of N_W has negligible effect on the demosaicking results.

III. System Optimization

A. Filter Optimization Based on Quality versus Complexity

The least-squares filter design is completely automated as long as we specify the rectangular kernel dimensions for filters h_1, h_{2a}, and h_{2b}. Therefore, we can program a scheme to systematically explore the effects on demosaicking quality of lowering the orders of the three filters. This is a worthwhile endeavour because preliminary results in [7] have shown that demosaicking quality and computational complexity form a tradeoff relationship. More importantly, there exists a "sweet spot" in terms of computational complexity where more complex realizations of the algorithm using larger filter orders give negligible improvements to objective and subjective demosaicking quality.

We shall elaborate on the notion of computational complexity as it pertains to the adaptive luma-chroma demultiplexing algorithm. The algorithm comprises primarily spatial signal filtering realized by the convolution operator, where the multiplication and addition are the only necessary atomic operations. The amount of required computational work depends only on the sizes of the signal and the filter involved. Of these two factors, only the filter kernel sizes can be adjusted to produce different realizations of the algorithm; the signal size depends on the unknown input CFA signal and so, without loss of generality, we consider the work to demosaic one image pixel. Furthermore, multiplications take significantly more computation time than additions on most processors. Therefore, counting the number of multiplications required to demosaic one pixel is a good indicator of computational complexity. Finally, because the algorithm uses the exact same steps to demosaic any arbitrary image pixel, the single-pixel count strictly reflects demosaicking speed — the fewer multiplications required (i.e., less complexity), the faster the demosaicking is.

By tracing the demosaicking steps described in Fig. 1 and omitting any multiplications involving a constant operand, the number of multiplications S required to demosaic one image pixel is

$$S = 2(P_1 + P_2 + 1) + M_1M_2 + 2N_1N_2 + 3, \tag{14}$$

where filter h_1 has order $M_1 \times M_2$, filter h_{2a} has order $N_1 \times N_2$, filter h_{2b} has order $N_2 \times N_1$, and the *separable* Gaussian filters h_{G1} and h_{G2} involved in the selection of weighting coefficients have orders $P_1 \times P_2$ and $P_2 \times P_1$ respectively. Equation (14) shows that these filter orders are the only adjustable parameters that have any effect on computational complexity. The remaining adjustable parameters include the two-dimensional standard deviations (r_{G1}, r_{G2}) and the centering frequency parameter f_m for the Gaussian filters h_{G1} and h_{G2}. However, they do not change Gaussian filter orders and so they do not exert any influence on computational complexity. A systematic full search showed that $(r_{G1}, r_{G2}) = (3.0, 1.0)$ is near optimal. Also, following the recommendation in [6], we set $f_m = 0.375$ c/px.

We used a greedy algorithm to explore the relationship between objective demosaicking quality and computational complexity. Let us represent the set of filter orders using vectors of the form
$[\ M_1 \quad M_2 \quad N_1 \quad N_2 \quad P_1 \quad P_2\ ]$. Each valuation of this vector defines a *configuration* that completely describes a realization of the adaptive frequency-domain algorithm. The greedy algorithm iterates from the initial configuration $[\ 11 \quad 11 \quad 11 \quad 11 \quad 11 \quad 11\ ]$, which is the recommendation from [7], to the end configuration $[\ 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1\ ]$. At each iteration the algorithm generates a maximum of six temporary configurations (fewer if some entries have already reached their end value) by decrementing each entry of the current configuration by 2, one at a time. We chose the step size of 2 to maintain odd-valued kernel dimensions. For each temporary configuration the greedy algorithm prepares an implementation of the demosaicking algorithm with the appropriate least-squares and Gaussian filters that match the configuration, and then performs demosaicking using this system over a test set of color images. Finally, the greedy algorithm exits the current iteration by retaining the temporary configuration that yields the best objective quality as the new "current" configuration. Regardless of whether it is discarded or kept, a temporary configuration will never be tested again at a later iteration.

Objective quality is determined by the average color mean-square error (CMSE) metric on the RGB color space. Specifically, the calculations involve first computing the average MSE for each of the red, green, and blue channel over the test images and then averaging the three MSEs to obtain the CMSE. A smaller CMSE indicates better objective quality,

which of course does not necessarily precisely reflect *subjective* quality. As an alternative measure of subjective quality, the S-CIELAB ΔE^* error measure has been proposed [11] and used to evaluate demosaicking performance [4], and we report those values as well in the paper. We used the MATLAB implementation of the S-CIELAB ΔE^* calculations available in [12] and kept the default assumption of a 72 dpi display seen from 18 inches away. There are two complexity reduction techniques worth considering. Both techniques involve generating filters at post-design time that approximate the least-squares filters h_1, h_{2a}, and h_{2b} while possessing some computationally desirable properties. The first technique is to create a quadrantally symmetric approximation from a least-squares filter by averaging every set of four quadrantally related filter coefficients and then redistributing this average value to those four locations. Quadrantal symmetry generally reduces the number of required multiplication in the convolution by a factor of close to four. The number of multiplications S_{QS} required to demosiac one pixel is then

$$S_{QS} = 2(P_1 + P_2 + 1) + \frac{(M_1+1)(M_2+1)}{4} + \frac{(N_1+1)(N_2+1)}{2} + 3. \quad (15)$$

The second technique is to create a separable filter by computing the singular value decomposition of the least-squares filter and then keeping only a first-order approximation that corresponds to the largest singular value. The number of multiplications S_{SF} required to demosaic one pixel becomes

$$S_{SF} = 2(P_1+P_2+1)+(M_1+M_2)+2(N_1+N_2)+3. \quad (16)$$

The greedy algorithm is performed three times in total: once without any complexity reduction technique, once with quadrantally symmetric filters, and once with separable filters.

Notice that the greedy algorithm does not exhaust every single configuration. In this case, it takes 30 iterations to reach the end configuration starting from the initial configuration. This means the algorithm tests fewer than 180 unique configurations in one run, which is much less than the $5^6 (= 15625)$ configurations contained within the boundaries of the initial and end configurations. Moreover, the computational complexity is always decreasing after every iteration. This is easily verified using equations (14) – (16) and the fact that one filter order is decremented at each iteration. These two properties make the greedy algorithm an efficient optimization technique. It is able to generate a sequence of successively less complex configurations that best preserve objective quality while considering only a small subset of configurations during the analysis.

B. Optimization Results

The filter design training set and the demosaicking test set of color images consisted of subsets from the 24 Kodak photosampler images of size 512×768 that are widely used in the demosaicking literature. Fig. 2 shows the changes to objective demosaicking quality, measured by CMSE and S-CIELAB ΔE^*, as a function of computational complexity, measured by the number of multiplications, for one complete experiment

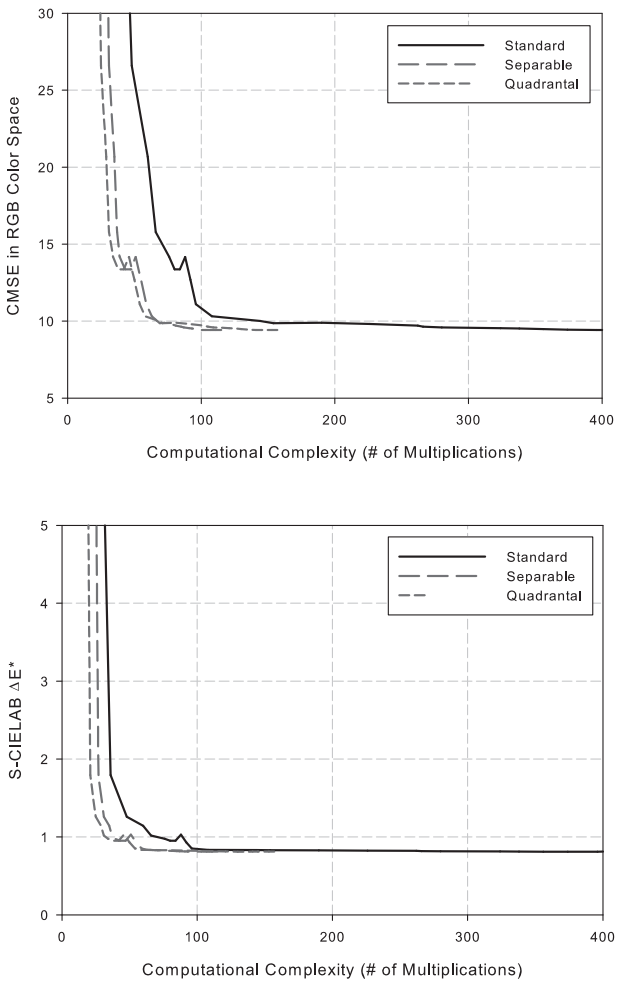


Fig. 2. Example plots of objective demosaicking quality as a function of computational complexity, generated from one experiment with the greedy algorithm.

with the greedy algorithm. Table I lists the configurations that correspond to the plot points of the "standard" scenario (i.e., no quadrantal symmetry and no separable filters) in Fig. 2. In this particular experiment the first twelve Kodak images formed the training set and the last twelve images formed the test set.

The experiment results confirm the tradeoff relationship between demosaicking quality and speed. The plots also show that the recommended filter specifications from [7], which formed the initial configuration, can be simplified further without much degradation in objective quality. The configurations that best balance demosaicking quality and speed are the ones clustered about the transition from negligible changes to significant changes in CMSE and S-CIELAB ΔE^*. Using Table I and favoring the least complex system possible, we identify the configuration $[\ 5 \quad 5 \quad 9 \quad 3 \quad 11 \quad 1\ ]$ as a good choice, corresponding to a 5×5 least-squares filter h_1, 9×3 least-squares filter h_{2a}, 3×9 least-squares filter h_{2b}, 11×1 Gaussian filter h_{G1} centered at $(0.375, 0.0)$ c/px, and 1×11 Gaussian filter h_{G2} centered at $(0.0, 0.375)$ c/px. Both 1D

TABLE I
THE UNDERLYING CONFIGURATIONS AND PLOT DATA SUPPORTING THE "STANDARD" CURVE OF FIG. 2

Configuration	# of Mult.	CMSE	S-CIELAB
[11 11 11 11 11 11]	412	9.42	0.812
[11 11 11 11 11 09]	408	9.42	0.812
[11 11 11 11 11 07]	404	9.42	0.812
[11 11 11 11 11 05]	400	9.42	0.812
[11 11 11 11 11 03]	396	9.42	0.811
[11 09 11 11 11 03]	374	9.44	0.811
[09 09 11 11 11 03]	356	9.48	0.811
[07 09 11 11 11 03]	338	9.52	0.812
[07 07 11 11 11 03]	324	9.54	0.814
[07 07 11 09 11 03]	280	9.59	0.815
[07 05 11 09 11 03]	266	9.64	0.818
[07 05 11 09 11 01]	262	9.71	0.820
[07 05 09 09 11 01]	226	9.81	0.824
[07 05 09 07 11 01]	190	9.89	0.827
[07 05 09 05 11 01]	154	9.86	0.829
[05 05 09 05 11 01]	144	10.01	0.832
[05 05 09 03 11 01]	108	10.31	0.833
[05 05 07 03 11 01]	96	11.09	0.852
[05 05 07 03 09 01]	92	12.68	0.930
[05 05 07 03 07 01]	88	14.17	1.029
[05 05 07 03 05 01]	84	13.36	0.952
[05 05 07 03 03 01]	80	13.36	0.952
[05 05 07 03 01 01]	76	14.18	0.977
[05 03 07 03 01 01]	66	15.78	1.017
[03 03 07 03 01 01]	60	20.67	1.142
[03 03 05 03 01 01]	48	26.59	1.259
[03 03 03 03 01 01]	36	56.64	1.795
[03 01 03 03 01 01]	30	154.42	6.360
[01 01 03 03 01 01]	28	207.44	9.324
[01 01 03 01 01 01]	16	300.46	10.992
[01 01 01 01 01 01]	12	735.79	17.401

Gaussian filters have a standard deviation of $r_G = r_{G1} = 3.0$. During our own subjective evaluation, the demosaicked images produced by this configuration were perceived as being virtually identical to those from the more complex configurations. Of course, designers can choose whichever configuration gives their preferred tradeoff between quality and complexity.

The "quadrantal" and "separable" plots from Fig. 2 show that using quadrantally symmetric or separable filter approximations do not affect the tradeoff relationship. Quandrantal symmetry should definitely be used, whereas there is no advantage to use separability.

Some additional observations can be made about the filters themselves. In the first steps, the Gaussian filters h_{G1} and h_{G2} are reduced to one-dimensional filters. This is reasonable because at the centering frequencies $(0.375, 0.0)$ c/px and $(0.0, 0.375)$ c/px, there is usually little energy along the dimension not of interest (vertical for h_{G1} and horizontal for h_{G2}) in the local frequency spectrum of the pixel undergoing demosaicking. Then, the order of h_1 is reduced more quickly than the order of h_2. Also, it appears the least-squares filter h_1 prefers a square kernel, while the kernels for least-squares filters h_{2a} and h_{2b} can be made significantly rectangular.

The entire experiment was repeated by assigning different subsets of the Kodak dataset as the training and test sets. This included trials where the training set and test set were identical. In every case we observe the same tradeoff relationship reported by Fig. 2. The sequences of chosen configurations do differ amongst the different setups. However, some configurations are consistently chosen at the same iteration. In fact, the identified configuration recommended earlier is one such configuration.

IV. SIMULATION RESULTS

We have tested the proposed LSLCD algorithm under numerous conditions and configurations. This section presents some representative results and comparisons with other reference methods from the recent literature. For the main comparison, we have used the base configuration [11 11 11 11 11 11], tested on the 24 Kodak photosampler images. A different set of filters was used for each image, trained on the other 23 images in the dataset; this set of filters is denoted f_{TO}. Other configurations have also been tested, including: using the same filters, trained on the entire set, for all images; dividing the dataset into two subsets of 12 images, and using the same filters for all the images in one subset, where they were obtained by training on the other subset of 12 images. There was little difference in the results obtained with these alternate configurations. Thus, we consider this result to be representative of the performance of LSLCD. For reference, we also present the result with filters trained on the test image itself (self-trained), denote f_{ST}. This gives a lower bound on the mean-square error achievable with the LSLCD algorithm, but it is not implementable since the original RGB image is not available for training. The frequency response of the sets of three 11×11 filters obtained for image #1 in these two scenarios f_{ST} and f_{TO} are illustrated in Fig. 3 (a)-(f). We note that the two versions of h_1 are quite similar, whereas differences in h_{2a} and h_{2b} are quite evident.

For comparison, we have selected the following reference methods:

A. The method of directional linear minimum mean-square error estimation (LMMSE) of Zhang and Wu [13]. Similar to the present paper, LMMSE attempts to minimize mean-square error, but of different quantities.

B. The adaptive homogeneity directed (AHD) method of Hirakawa and Parks [14], with $\delta = 2$. Although this method gives higher objective error than [18], it was judged to give better subjective performance in the survey of [3].

C. The original adaptive luma-chroma demultiplexing algorithm of [6], with 21×21 filters designed using the window method, denoted here as FDM (frequency domain method).

D. The edge estimation for analyzing the variance of color differences approach (VCD) of Chung and Chan [15].

E. The K-means singular value decomposition approach to demosaicking (K-SVD) of Mairal *et al.* [16]. The sparse nature of color images is manipulated to obtain an appropriate dictionary that is used for demosaicking with the iterative K-SVD algorithm.

F. The regularization approach to demosaicking (RAD) of Menon and Calvagno [17].

Two objective quality measures (CPSNR and S-CIELAB ΔE^*) and computational time were used to evaluate the performance of demosaicking algorithms. The S-CIELAB ΔE^*

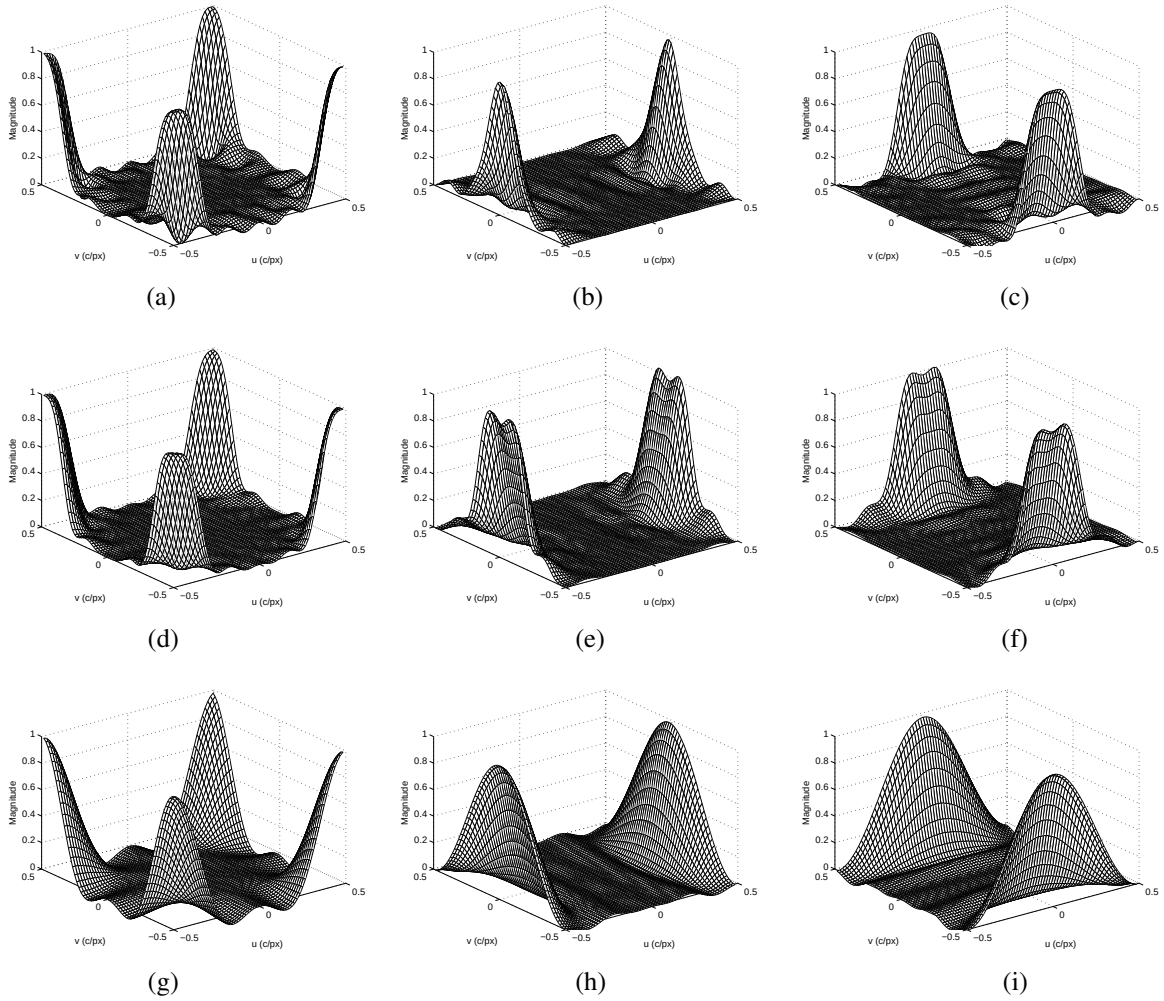


Fig. 3. Perspective view of frequency responses of filters for demosaicking image #1. (a-c) f_{ST}: h_1, h_{2a}, and h_{2b} of size 11×11 obtained by training with image $\#1$. (d-f) f_{TO}: h_1, h_{2a}, and h_{2b} of size 11×11 obtained with training set consisting of images $\#2 - \#24$. (g-i) h_1, h_{2a}, and h_{2b} of sizes 5×5, 9×3 and 3×9 obtained by training on images $\#2 - \#24$.

metric uses a perceptual model in an attempt to measure the spatial color reproduction accuracy of an estimated image against the original as seen by a human viewer [11]. A larger ΔE^* value denotes more error in the estimated image and hence poorer perceived visual quality. Additionally, demosaicking speeds were tested by measuring the times that each algorithm took to demosaic the 24 Kodak images. The experiments were conducted on an Intel(R) Core(TM) i7-920 CPU (2.67 GHz) with 6 GB of RAM and using MATLAB implementations of the algorithms by the original authors[1]. Our software and all of the 24 original Kodak images and those demosaicked using each of the tested methods are available at [9]. The readers are invited to visit the website to assess the visual quality of demosaicked images using different demosaicking schemes.

A. Objective Performance Comparison with the Benchmarks

Table II shows the CPSNR and S-CIELAB ΔE^* of the 24 test images and the average error measures. We discarded an eleven pixel wide border from each image during the error calculations to exclude edge effects not representative of the performance of the various methods. The bold-italic entries indicate either the highest CPSNR or the smallest S-CIELAB ΔE^* in each row (not including the non-implementable LSLCD with self-trained filters, which is separated from other results with a double line). The italic entries indicate the second best result for these measures in each row. These numerical results may not correspond directly to those reported in other published papers, including [6], because of differences in the test platforms, such as different versions of the same images being used or slight differences in the metric calculations (including the size of the excluded border).

It can be seen from Table II that the estimates of the demosaicked images by the proposed LSLCD algorithm are very competitive with the reference methods. The LSLCD algorithm using f_{TO} outperforms most of the other methods on average, except [16]. The margins of improvement in CPSNR are 0.08, 2.60, 0.55, 0.22, -0.34, and 0.42 dB, respectively. Although the LSLCD algorithm does not give better performance than K-SVD on average, the LSLCD algorithm

[1]Since VCD [15] and K-SVD [16] implementations were provided as executable files, they are not included in this comparison.

TABLE II

CPSNR (M_1) AND S-CIELAB ΔE^* (M_2) METRICS FOR 24 DEMOSAICKED IMAGES USING EIGHT ALGORITHMS. (A) LMMSE [13]. (B) AHD [14]. (C) FDM [6]. (D) VCD [15]. (E) K-SVD [16]. (F) RAD [17]. (G) LSLCD USING f_{TO}. (H) LSLCD USING f_{ST}.

	A		B		C		D		E		F		G		H	
Metric	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2	M1	M2
1	38.51	1.108	34.98	1.248	38.13	1.156	38.59	*1.105*	***39.71***	***1.009***	38.28	1.198	*38.67*	1.133	39.55	1.052
2	*40.92*	0.648	39.11	0.801	40.01	0.671	40.12	0.744	***41.41***	***0.612***	39.84	0.718	40.81	*0.642*	41.54	0.581
3	*42.73*	***0.479***	40.91	0.574	41.65	0.534	42.67	0.518	***43.21***	*0.491*	42.10	0.548	42.37	0.509	43.22	0.481
4	41.08	0.671	38.73	0.851	40.62	0.688	40.60	0.757	***42.02***	***0.623***	40.85	0.706	*41.34*	*0.644*	42.01	0.610
5	38.09	1.070	35.26	1.309	37.85	1.052	37.91	1.063	***38.81***	***0.935***	38.01	1.117	*38.30*	*1.024*	38.78	0.941
6	*40.25*	*0.774*	37.36	0.864	40.03	0.777	40.06	0.780	40.10	0.829	39.84	0.851	***40.42***	***0.771***	41.01	0.721
7	42.38	0.574	40.36	0.686	42.12	0.537	42.25	0.614	***43.30***	***0.513***	42.43	0.577	*42.86*	*0.526*	43.59	0.468
8	36.06	*1.332*	33.60	1.434	35.22	1.482	***36.45***	***1.282***	*36.42*	1.405	36.02	1.400	35.66	1.449	36.16	1.433
9	42.84	***0.569***	40.71	0.649	42.06	0.591	***43.12***	0.574	*43.09*	*0.573*	42.07	0.648	42.85	0.581	43.05	0.569
10	42.61	0.544	40.45	0.620	42.21	0.549	42.48	0.575	***42.89***	***0.536***	42.47	0.601	*42.85*	*0.541*	43.00	0.532
11	40.08	*0.737*	37.25	0.856	39.74	0.759	39.69	0.784	*40.19*	0.779	39.70	0.800	***40.33***	***0.731***	40.43	0.725
12	*43.51*	***0.461***	41.53	0.520	43.05	*0.464*	43.43	0.491	***43.67***	0.484	43.19	0.519	43.46	0.479	43.72	0.472
13	34.80	1.560	31.02	1.943	*35.07*	1.583	34.89	1.588	***35.31***	***1.531***	34.84	1.625	34.89	*1.573*	36.57	1.421
14	*37.01*	*0.981*	35.12	1.202	35.85	1.079	36.61	1.081	***37.77***	***0.948***	36.10	1.079	36.75	0.991	37.82	0.923
15	39.87	0.657	37.57	0.798	39.67	0.659	39.83	0.675	***40.92***	***0.600***	39.89	0.685	*40.03*	*0.636*	40.67	0.584
16	*43.81*	*0.531*	41.22	0.587	43.66	0.531	43.77	0.551	43.47	0.582	43.24	0.593	***44.09***	***0.526***	44.45	0.497
17	*41.84*	*0.517*	38.93	0.612	41.57	0.524	41.47	0.574	41.84	0.529	41.49	0.559	***42.00***	***0.507***	42.17	0.501
18	37.47	1.008	34.39	1.290	37.44	1.022	37.21	1.064	***37.88***	***0.957***	37.47	1.033	*37.80*	*0.965*	37.89	0.983
19	40.90	***0.752***	38.18	0.874	40.35	0.790	*41.02*	0.757	***41.04***	*0.755*	40.01	0.844	40.87	0.755	41.04	0.735
20	41.27	***0.545***	38.88	0.642	40.39	0.601	*41.29*	*0.557*	***41.51***	0.564	41.00	0.594	41.25	0.567	41.44	0.524
21	39.16	0.901	36.32	1.030	38.74	0.935	*39.30*	0.901	***39.62***	***0.884***	39.22	0.940	39.28	*0.892*	39.65	0.868
22	38.44	0.964	36.17	1.215	38.09	0.940	38.20	1.028	***38.73***	*0.911*	38.29	0.972	*38.67*	***0.908***	38.86	0.884
23	*43.28*	0.495	41.46	0.614	42.12	0.506	42.77	0.566	***43.83***	***0.467***	42.35	0.536	43.15	*0.479*	44.25	0.440
24	35.53	1.018	32.47	1.312	35.39	1.012	35.23	1.057	***35.72***	*1.011*	35.57	1.055	*35.60*	***0.969***	35.75	0.990
Avg.	40.10	0.787	37.58	0.939	39.63	0.810	39.96	0.820	***40.52***	***0.772***	39.76	0.842	*40.18*	*0.783*	40.69	0.747

provides comparable CPSNR performance with significantly lower computational cost. To obtain the result images of [16], K-SVD first calculates rough results with a globally trained dictionary followed by two times 20 learning iterations with a patch sparsity equal to 20. This process involves heavy computational costs, and is not currently applicable for real-time systems.

Compared with the LMMSE method of [13], LSLCD algorithm with f_{TO} yields just slightly better CPSNR result (0.08 dB). The LSLCD algorithm fails to provide better average CPSNR result (-0.34 dB) than K-SVD algorithm. However, the CPSNR results are comparable to or better than the images obtained using K-SVD for images # 6, 11, 16, 17. One should keep in mind that increases to the CPSNR do not always imply better subjective quality. For color images, human vision system-based S-CIELAB metric may be more appropriate than CPSNR metric. In terms of S-CIELAB ΔE^* metric, LSLCD with f_{TO} yields better performance on average than most of other reference methods except K-SVD (-0.004, -0.156, -0.027, -0.037, 0.011, and -0.059, respectively), although again LMMSE and K-SVD are very close.

B. CPSNR & S-CIELAB ΔE^* vs. Computational Time

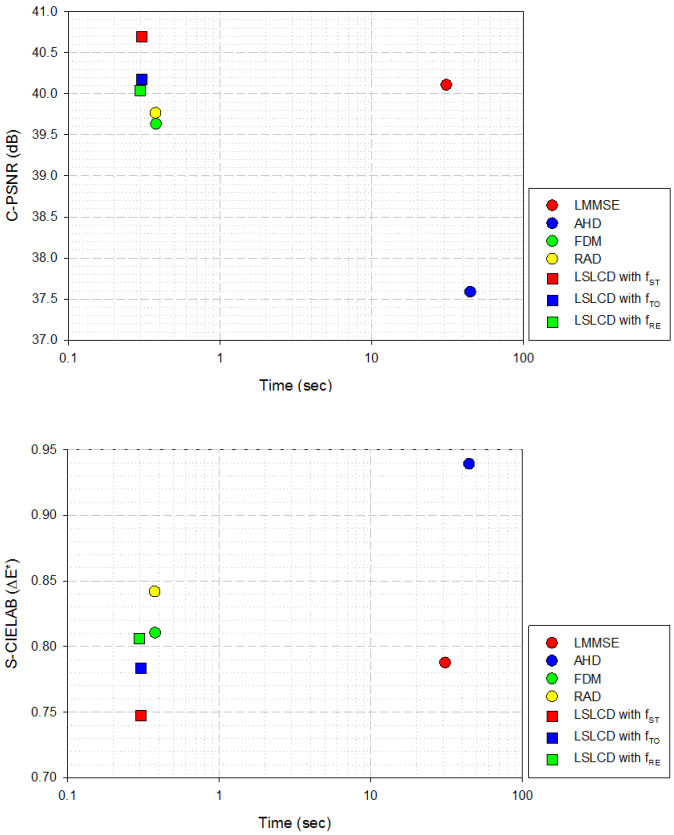
We now compare the performance of the LSLCD algorithm against the other methods in terms of computation time. Table III shows the average elapsed time to demosaic one 512×768 image of the Kodak dataset, where (I) represents LSLCD algorithm using recommended filter f_{RE} with configuration $[\ 5\ \ 5\ \ 9\ \ 3\ \ 11\ \ 1\ ]$. As can be seen in Table III, the computation and implementation complexities of the LSLCD algorithm are considerably lower than the method (F) of [17], which is known as the most efficient of the conventional methods. The elapsed computation time of LSLCD algorithm with f_{ST}, f_{TO}, and f_{RE} is 80.9 %, 80.7 %, and 78.5 %, respectively, of that of [17].

TABLE III

AVERAGE TIME (S) TO DEMOSAIC ONE 512×768 IMAGE OF THE KODAK DATASET. ALGORITHM LABELS (A) TO (C) AND (F) TO (H) CORRESPOND TO THOSE USED IN TABLE II. (I) LSLCD USING RECOMMENDED FILTER f_{RE}.

Method	A	B	C	F	G	H	I
$\frac{Time}{Image}$	31.15	44.80	0.380	0.378	0.306	0.305	0.297

As a further evaluation, we follow the method of comparison of Portilla *et al.* [19]. In their paper Portilla *et al.* compared their algorithm against other solutions in terms of demosaicking visual quality (S-CIELAB ΔE^*) and speed of demosaicking. In addition to their comparison, we have added CPSNR results in the comparison. Speed of demosaicking is evaluated by the measured average time (in seconds) to demosaic one image. For each algorithm we calculate the



Method	CPSNR	S-CIELAB	Time	$\frac{Time_{Method}}{Time_I}$
A	40.102	0.7873	31.1470	104.99
B	37.582	0.9388	44.8023	151.02
C	39.626	0.8101	0.3801	1.28
F	39.760	0.8415	0.3779	1.27
G	40.692	0.7473	0.3056	1.03
H	40.178	0.7834	0.3051	1.03
I	40.035	0.8062	0.2967	1.00

Fig. 4. Plot of demosaicking CPSNR (dB) and visual quality (S-CIELAB ΔE^*) versus speed (s) of demosaicking for the algorithms under comparison. The red, blue, and green squares correspond to the LSLCD algorithms with f_{ST}, f_{TO}, and f_{RE}, respectively. The red, blue, green, and yellow circles correspond to the algorithms LMMSE, AHD, FDM, and RAD, respectively. Algorithm labels (A) to (I) correspond to those used in Table III.

CPSNR and S-CIELAB ΔE^* value of every test image and then compute the average CPSNR and S-CIELAB ΔE^* for the set. At the same time, we measure the total elapsed time to demosaic the 24 images and then compute the average time consumed by the demosaicking of one image. Finally, we plot the average CPSNR, S-CIELAB ΔE^* values and the average demosaicking times on a 2D scatter plot.

Fig. 4 shows the 2D scatter plot of the data collected from an instance of the experiment. Portilla *et al.* remarked in [19] that demosaicking algorithms can be grouped into three broad categories: "high speed, low quality", "medium-low speed, high quality", and those "in between". Fig. 4 certainly confirms these three classifications of conventional demosaicking methods. However, the LSLCD demosaicking algorithm and its variants do not belong to any of those categories because they give "high speed, high quality". We conclude from Fig. 4 that the LSLCD demosaicking algorithm offers the best demosaicking visual quality while maintaining the fastest demosaicking speeds. We recognize that these complexity comparisons are very rough and dependent on the specific MATLAB implementations. However, they are the best mechanism we have to evaluate relative computational complexity, and we have used the implementations of the original authors in all cases.

C. Subjective Quality Comparison

In [9], all results of the original and the demosaicked images by benchmark schemes ([13], [14], [6], [15], [16], and [17]) and our proposed algorithm (LSLCD with f_{ST}, f_{TO}, and f_{RE}) are shown to allow subjective quality assessment by the reader. The subjective quality can be appraised with regard to reconstruction of textures, edges, and different kinds of geometric details such as diagonals, corners, and fine patterns. Images in [9] indicate how each benchmark method fares in reconstructing the demosaicked images in these difficult regions. The proposed LSLCD algorithm appears to yield visually more friendly color images with color artifacts well removed, consistent with the objective quality measures.

V. Conclusion

This paper has presented a detailed analysis of the adaptive least-squares luma-chroma demultiplexing algorithm for Bayer demosaicking. It has been shown to provide state-of-the-art demosaicking performance in terms of CPSNR and S-CIELAB ΔE^* with low complexity, and gives the best performance-complexity tradeoff of the methods studied. Essentially optimal performance is achieved with 11×11 bandpass filters to extract the modulated chroma components, compared to the 21×21 filters used in [6]. However, a systematic study of the objective performance as a function of the filter size shows that a significant further reduction in complexity can be achieved with little loss in performance. For example, a system using a 5×5 filter for h_1 and 9×3 and 3×9 filters for h_{2a} and h_{2b} gives near optimal performance. Quadrantal symmetry should definitely be exploited, but no advantage was obtained with separable filters. It was also found that one-dimensional Gaussian filters are sufficient to form the adaptation signal.

Although a training set is required for filter design, the performance is not very sensitive to the choice of training set, so any set of images representative of the given application can be used to design a fixed set of filters. The methodology presented in this paper is applicable to any CFA structure using the methods described in [8]. Further work is underway to better deal with crosstalk near the Nyquist region boundary, as in the picket fence in the well-known lighthouse image (image 19 in the Kodak dataset), and to obtain a more solid theoretical basis for the adaptation rule (although the present one works very well). An interesting problem for further investigation is to design filters for the LCD demosaicking approach in the absence of a training set.

REFERENCES

[1] P. L. P. Dillon, D. M. Lewis, and F. G. Kaspar, "Color imaging system using a single CCD area array," *IEEE Jounal of Solid-State Circuits*, vol. 13, no. 1, pp. 28–33, Feb. 1978.

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

[3] B. K. Gunturk, J. Glotzbach, Y. Altunbasak, R. W. Schafer, and R. M. Mersereau, "Demosaicking: Color filter array interpolation," *IEEE Signal Process. Mag.*, vol. 22, no. 1, pp. 44–54, Jan. 2005.

[4] X. Li, B. Gunturk, and L. Zhang, "Image demosaicing: a systematic survey," in *Proceedings of the SPIE*, vol. 6822, Jan. 2008, p. 68221J.

[5] D. Alleysson, S. Süsstrunk, and J. Hérault, "Linear demosaicing inspired by the human visual system," *IEEE Trans. Image Process.*, vol. 14, no. 4, pp. 439–449, Apr. 2005.

[6] E. Dubois, "Frequency-domain methods for demosaicking of Bayer-sampled color image," *IEEE Signal Process. Lett.*, vol. 12, pp. 847–850, Dec. 2005.

[7] ——, "Filter design for adaptive frequency-domain Bayer demosaicking," in *IEEE International Conference on Image Processing (ICIP) 2006*, Oct. 2006, pp. 2705–2708.

[8] ——, "Color filter array sampling of color images: Frequency-domain analysis and associated demosaicking algorithms," in *Single Sensor Imaging: Methods and Applications for Digital Cameras*, R. Lukac, Ed. CRC Press, Taylor & Francis Group, 2009.

[9] E. Dubois, G. Jeon, and B. Leung. (2010) Least-squares luma-chroma demultiplexing algorithm for bayer demosaicking: Additional results. [Online]. Available: http://www.site.uottawa.ca/ edubois/lslcd/

[10] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing*. Upper Saddle River, NJ: Prentice Hall, 2000.

[11] X. Zhang and B. A. Wandell, "A spatial extension of CIELAB for digital color image reproduction," *Journal of the Society for Information Display*, vol. 5, no. 1, pp. 61–67, Mar. 1997.

[12] X. Zhang. (1998) S-CIELAB: A spatial extension to the CIE L*a*b* DeltaE color difference metric. [Online]. Available: http://white.stanford.edu/˜brian/scielab/scielab.html

[13] L. Zhang and X. Wu, "Color demosaicking via directional linear minimum mean square-error estimation," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2167–2177, Dec. 2005.

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

[15] 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.

[16] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Trans. Image Process.*, vol. 17, no. 1, pp. 53–69, Jan. 2008.

[17] D. Menon and G. Calvagno, "Regularization approaches to demosaicking," *IEEE Trans. Image Process.*, vol. 18, no. 10, pp. 2209–2220, Oct. 2009.

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

[19] J. Portilla, D. Otaduy, and C. Dorronsoro, "Low-complexity linear demosaicing using joint spatial-chromatic image statistics," in *IEEE International Conference on Image Processing (ICIP) 2005*, vol. 1, no. 11–14, Sep. 2005, pp. I–61–4.

Brian Leung (S'07) received the B.A.Sc. (honours) degree in computer engineering and B.Sc. (honours) degree in mathematics from University of Ottawa in 2006 and 2007, and the M.A.Sc. degree in biomedical engineering from University of Toronto in 2009.

He completed undergraduate internships in web application development with the National Research Council of Canada (2004), ASIC verification with Tundra Semiconductors (2005), and image processing with the School of Information Technology and Engineering (SITE) at University of Ottawa. He is currently a Ph.D. candidate from the Institute of Biomaterials and Biomedical Engineering at the University of Toronto. His research interests include image processing, biomedical signal processing, machine learning, and rehabilitation engineering. His current research is focused on access and assistive technologies for children with severe physical disabilities.

Gwanggil Jeon received B.S., M.S., and Ph.D. (Summa Cum Laude) degrees in electronic engineering from Hanyang University, Korea, in 2003, 2005, and 2008, respectively.

From 2008 to 2009, he was with the Department of Electronics and Computer Engineering, Hanyang University, as a postdoctoral fellow. He is currently a postdoctoral fellow in the School of Information Technology and Engineering (SITE), University of Ottawa. 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.

Dr. Jeon was with the recipient of the IEEE Chester Sall Award in 2007 and 2008 ETRI Journal Paper Award.

Eric Dubois (S'72-M'77-SM'85-F'96) received the B.Eng. (honours) degree with great distinction and the M.Eng. degree from McGill University in 1972 and 1974, and the Ph.D. from the University of Toronto in 1978, all in electrical engineering.

He joined the Institut national de la recherche scientifique (University of Quebec) in 1977, where he held the position of professor in the INRS-Télécommunications centre in Montreal, Canada. Since July 1998, he is Professor with the School of Information Technology and Engineering (SITE) at the University of Ottawa, Ottawa, Canada. He was Vice-Dean (Research) and Secretary of the Faculty of Engineering from 2001 to 2004. From January 2006 to December 2008 he was Director of SITE. His research has centered on the compression and processing of still and moving images, and in multidimensional digital signal processing theory. His current research is focused on stereoscopic and multiview imaging, image sampling theory, image-based virtual environments and color signal processing. The research has been carried out in collaboration with such organizations as the Communications Research Centre, the National Research Council, the RCMP, and the Learning Objects Repositories Network (LORNET).

Dr. Dubois is a Fellow of the Canadian Academy of Engineering and of the Engineering Institute of Canada. He is a registered professional engineer in Quebec as member of the Order of Engineers of Quebec. He is a member of the Society for Information Display (SID) and the Society for Imaging Science and Technology (IS&T). He is corecipient of the 1988 Journal Award from Society of Motion Picture and Television Engineers. He is a member of the editorial board of the EURASIP journal Signal Processing: Image Communication and was an associate editor of the IEEE Transactions on Image Processing (1994-1998). He was technical program co-chair for the IEEE 2000 International Conference on Image Processing (ICIP) and a member of the organizing committee for the IEEE 2004 International Conference on Acoustics, Speech and Signal Processing (ICASSP).