Malvar-He-Cutler Linear Image Demosaicking

Pascal Getreuer

CMLA, ENS Cachan, France
(pascal.getreuer@cmla.ens-cachan.fr)

Abstract

Image demosaicking (or demosaicing) is the interpolation problem of estimating complete color information for an image that has been captured through a color filter array (CFA), particularly on the Bayer pattern. In this paper we review a simple linear method using 5 x 5 filters, proposed by Malvar, He, and Cutler in 2004, that shows surprisingly good results.

Source Code

The source code (ANSI C), its documentation, and the online demo are accessible at the IPOL web page of this article[1]. This software includes the implementations of algorithms potentially linkable to patents. Various distribution terms apply. Some files are distributed under the terms of the simplified BSD license, some are for scientific and education use only. Please see the readme file or the online documentation for details.

Keywords: image demosaicking; Bayer pattern; linear method

1 Introduction

Image demosaicking (or demosaicing) is the interpolation problem of estimating complete color information for an image that has been captured through a color filter array (CFA), particularly on the Bayer pattern. While many complicated methods for demosaicking have been proposed, Malvar, He, and Cutler [5, 8] showed that surprisingly good results are possible with a simple linear method using 5×5 filters. Figure 1 shows an example of demosaicking.

In the Bayer pattern [1], green pixels cover half the array in a quincunx lattice. The red and blue pixel locations are spaced uniformly every two pixels. The pattern alternates between "red rows" and "blue rows". In a red row the pattern is R, G, R, G, . . . , and in a blue row it is G, B, G, B, ... The pattern is displayed in Figure 2.

---

[1]http://dx.doi.org/10.5201/ipol.2011.g_mhcd

Figure 1: Example of demosaicking. Left, original image. Center, simulated image displaying the Bayer pattern. Right, demosaicking result of Malvar, He, and Cutler's method, after processing the simulated image. The images are enlarged to show individual pixels more clearly.
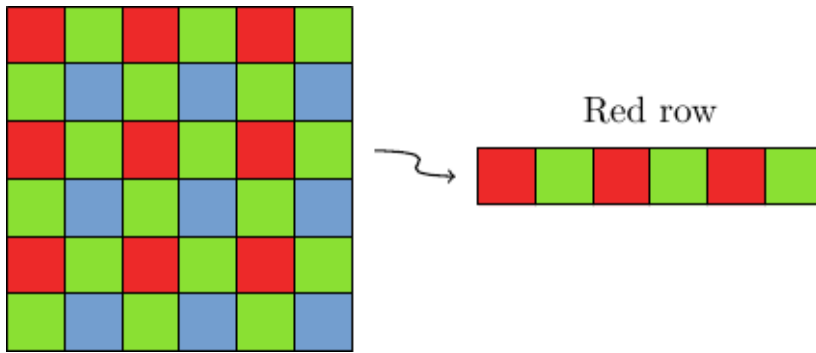

Figure 2: The Bayer pattern.

2 Algorithm

The method is derived as a modification of bilinear interpolation. Let R, G, B denote the red, green, and blue channels. At a red or blue pixel location, the bilinear interpolation of the green component is the average of its four axial neighbors,

$$\hat{G}^{bl}(i,j) = \frac{1}{4}\big(G(i-1,j) + G(i+1,j) + G(i,j-1) + G(i,j+1)\big).$$

Bilinear interpolation of the red and blue components is similar, but using instead the four diagonal neighbors.

The visual quality of bilinear demosaicking is generally quite poor. Since the channels are interpolated independently, the misalignments near edges produce strong color distortions and zipper artifacts.

To improve upon the quality of the bilinear method, Malvar, He, and Cutler follow the work of Pei and Tam [3] by adding Laplacian cross-channel corrections. The green component at a red pixel location is estimated as

$$\hat{G}(i,j) = \hat{G}^{bl}(i,j) + \alpha\Delta_R(i,j),$$

where Δ_R is the discrete 5-point Laplacian of the red channel,

$$\Delta_R(i,j) := R(i,j) - \frac{1}{4}\big(R(i-2,j) + R(i+2,j) + R(i,j-2) + R(i,j+2)\big).$$

To estimate a red component at a green pixel location,

$$\hat{R}(i,j) = \hat{R}^{bl}(i,j) + \beta\Delta_G(i,j),$$

where Δ_G is a discrete 9-point Laplacian of the green channel.

To estimate a red component at a blue pixel location,

$$\hat{R}(i,j) = \hat{R}^{bl}(i,j) + \gamma\Delta_B(i,j),$$

where Δ_B is the discrete 5-point Laplacian of the blue channel. By symmetry, blue components are estimated in a manner similar to the estimation of the red components. These formulas are shown in more detail in the next section.

The parameters α, β and γ control the weight of the Laplacian correction terms. To set these parameters optimally, the values producing the minimum mean squared error over the Kodak image suite were computed. These values were then rounded to dyadic rationals to obtain

$$\alpha = \frac{1}{2}, \quad \beta = \frac{5}{8}, \quad \gamma = \frac{3}{4}.$$

The advantage of this rounding is that the filters may be efficiently implemented with integer arithmetic and bitshifting. The filters approximate the optimal Wiener filters within 5% in terms of mean squared error for a 5×5 support.

The demosaicking is implemented by convolution with a set of linear filters. There are eight different filters for interpolating the different color components at different locations. The filters are shown in Figure 3.

For example, at a green pixel location in a red row, the red component is interpolated by

$$\hat{R}(i,j) = \frac{1}{8}\Big(\begin{array}{ccccc} & & \tfrac{1}{2}G(i,j-2) & & \\ & -G(i-1,j-1) & & -G(i+1,j-1) & \\ -G(i-2,j) + & 4R(i-1,j) & + \ 5G(i,j) \ + & 4R(i+1,j) & + \ -G(i+2,j) \\ & -G(i-1,j+1) & & -G(i+1,j+1) & \\ & & + \tfrac{1}{2}G(i,j+2) & & \end{array} \Big).$$

The filters can be implemented using integer arithmetic. Suppose that the input CFA data is given as a 2D integer array $F(i,j)$, then the interpolation above can be implemented as

$$\begin{aligned} R(i,j) = (\ & F(i,j-2) + F(i,j+2)+ \\ & -2(F(i-1,j-1) + F(i+1,j-1) + F(i-2,j) + F(i+2,j) + F(i-1,j+1) + F(i+1,j+1))+ \\ & +8(F(i-1,j) + F(i+1,j))+ \\ & +10F(i,j) \ \ )/16, \end{aligned}$$

and the division by 16 can be efficiently implemented by bitshifting. The other filters are implemented similarly.
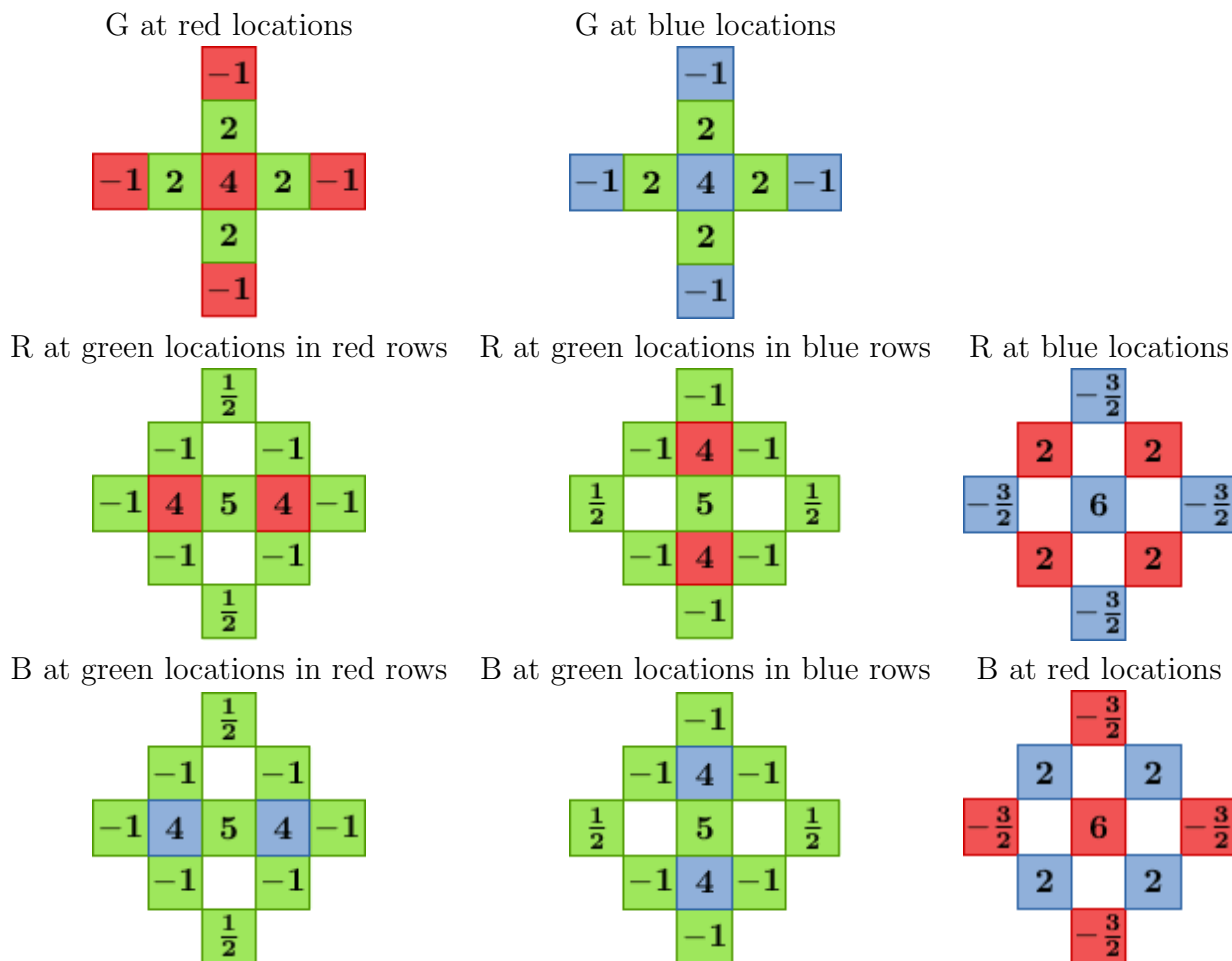
85

Figure 3: The 5×5 linear filters. The coefficient values are scaled by 8.

3   Examples

Considering its simplicity, the method works surprisingly well compared to more complicated methods. See Figure 4 for a comparison. Aside from the bilinear interpolation, all other methods shown here are either nonlinear or use larger filter support.

Bilinear interpolation and Hamilton-Adams methods are also low-cost demosaicking methods, so it is interesting to compare particularly with them. The example in Figure 5 shows that Malvar-He-Cutler is visually sharper than bilinear, but not as sharp as Hamilton-Adams. This is because Hamilton-Adams uses a nonlinear interpolation strategy.
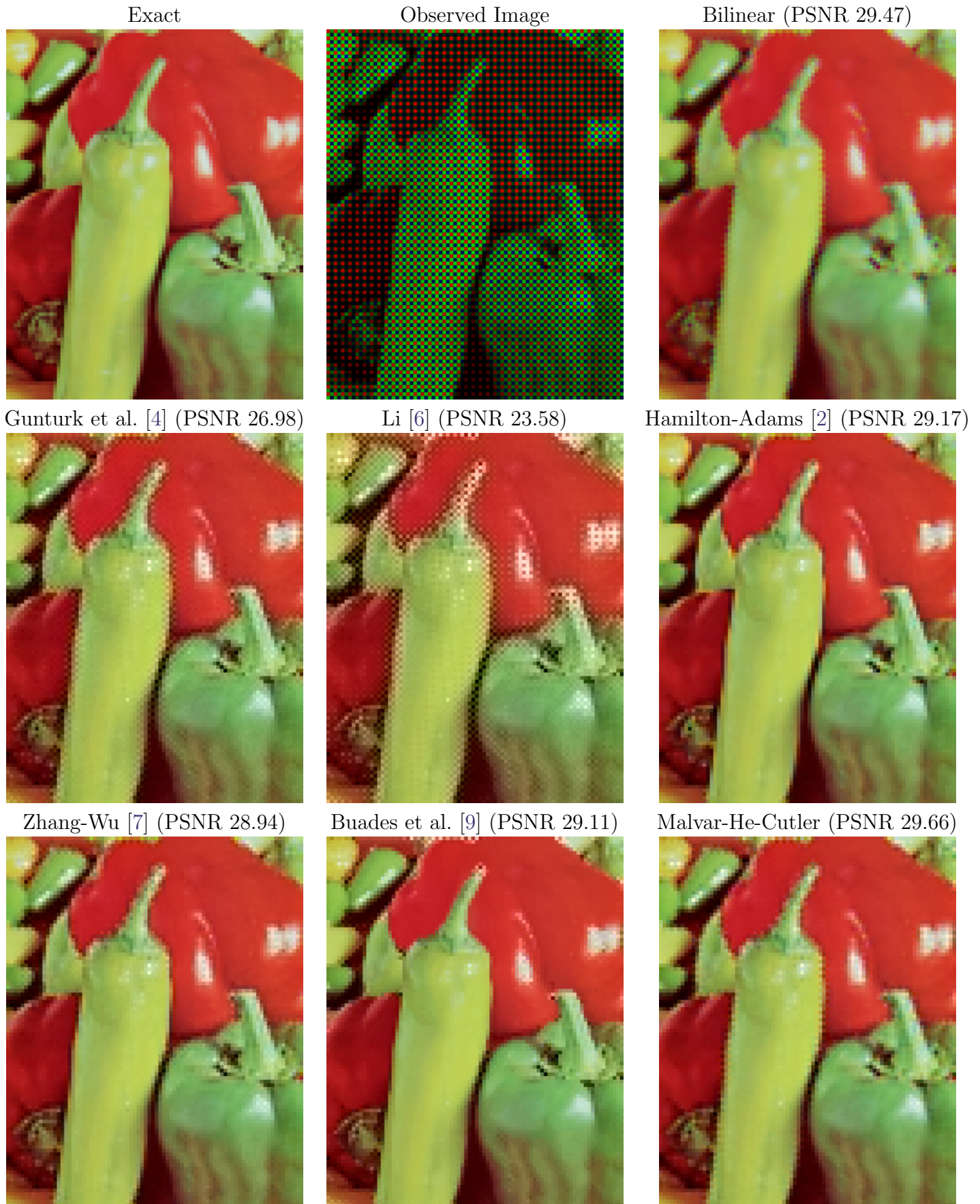
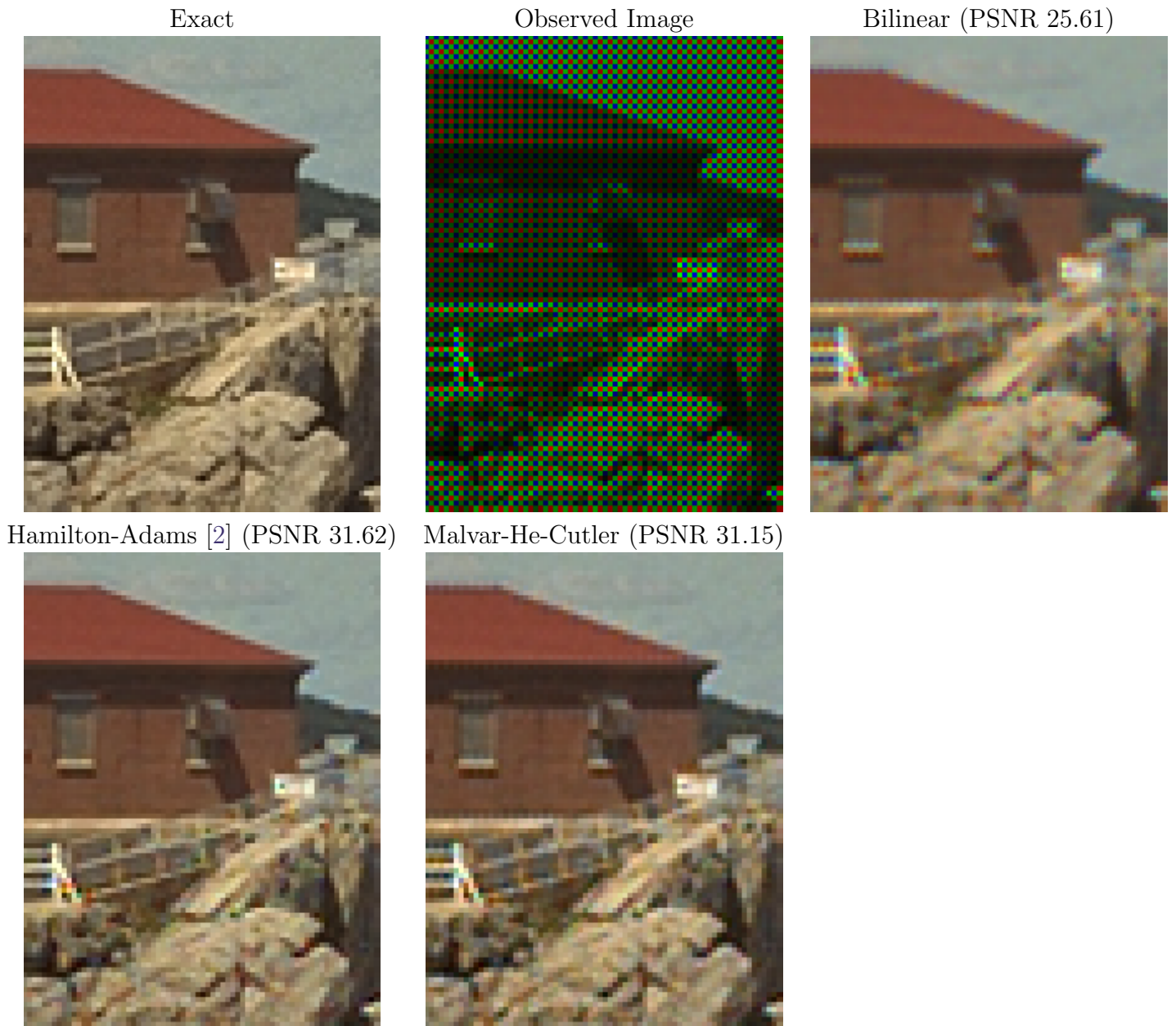Figure 4: Comparison of demosaicking results of different methods.

Exact

Observed Image

Bilinear (PSNR 25.61)

Hamilton-Adams [2] (PSNR 31.62)

Malvar-He-Cutler (PSNR 31.15)

Figure 5: Comparison of Malvar-He-Cutler demosaicking to bilinear and Hamilton-Adams methods.

Image Credits

 Kodak Image Suite[2], images 7 and 21.

 Standard test image.

References

[1] E. Bayer, "Color imaging array", U.S. Patent 3971065, 1976.

[2] F. Hamilton, Jr. and J. E. Adams, Jr., "Adaptive color plan interpolation in single sensor color electronic camera", U.S. Patent 5629734, 1997.

[3] S.-C. Pei and I.-K. Tam, "Effective color interpolation in CCD color filter array using signal correlation", in Proceedings of International Conference on Image Processing (ICIP), pp. 488–491, September 2000.

[4] K. Gunturk, Y. Altunbasak, and R. M. Mersereau, "Color plane interpolation using alternating projections", IEEE Transactions on Image Processing, vol. 11, no. 9, pp. 997–1013, 2002. http://dx.doi.org/10.1109/TIP.2002.801121

[5] H.S. Malvar, L. He, R. Cutler, "High-quality linear interpolation for demosaicing of Bayer-patterned color images", in Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2004. http://dx.doi.org/10.1109/ICASSP.2004.1326587

[6] X. Li, "Demosaicing by successive approximation", IEEE Transactions on Image Processing, vol. 14, no. 3, pp. 370–379, 2005. http://dx.doi.org/10.1109/TIP.2004.840683

[7] L. Zhang and X. Wu, "Color demosaicking via directional linear minimum mean square-error estimation", IEEE Transactions on Image Processing, vol. 14, no. 12, pp. 2167–2178, 2005. http://dx.doi.org/10.1109/TIP.2005.857260

[8] H.S. Malvar, L. He, R. Cutler, "High-quality gradient-corrected linear interpolation for demosaicing of color images", U.S. Patent 7502505, 2009.

[9] A. Buades, B. Coll, J-M. Morel, C. Sbert, "Self-Similarity Driven Demosaicking", Image Processing On Line, 1 (2011). http://dx.doi.org/10.5201/ipol.2011.bcms-ssdd

---

[2]http://r0k.us/graphics/kodak/