The Hong Kong Polytechnic University

Department of Electronic and Information Engineering

Feature-Preserving Processing Techniques for Color Filter Array Images

Chung King Hong

A thesis submitted in partial fulfillment of the requirements

for the degree of Doctor of Philosophy

November 2008

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____ (Signed)

Chung King Hong (Name of student)

To my wife

Abstract

To reduce cost and size, most digital cameras capture scene with a single-sensor image acquisition system in which the sensor is coated with a Bayer color filter array (CFA) and samples only one of the three primary colors (red, green and blue) at each pixel. The mosaic sensor raw data, referred to as Bayer CFA image, is then converted to full-color image by estimating the two missing components of the pixels. This process is called color interpolation or color demosaicing, and it is critical to a digital camera as it determines the output quality of the camera. The demosaiced full-color image may then be enhanced with a post-processing step to attain a visually pleasing output before being compressed for storage.

Recently, such a demosaicing-then-compression arrangement is found to be sub-optimal from the compression point of view because eventually the compression process has to remove the redundancy introduced in the demosaicing process. To solve this problem, the system is modified by adding an additional processing branch or by replacing the original pipeline with a new one to allow direct compression of the Bayer CFA image before demosaicing. As a result, more sophisticated demosaicing and post-processing algorithms can be applied and carried out offline in a powerful computer to attain a higher quality output.

This thesis investigates various processing algorithms to improve the performance of an advanced pipeline in terms of quality and complexity. It deals exclusively with raw Bayer CFA images and focuses on addressing the problem of color demosaicing, digital zoom and image compression respectively.

Color demosaicing is one of the most important processes in a single-sensor image acquisition system as it turns a CFA image into a full-color image. In this thesis, a feature preserving demosaicing algorithm is first presented to restore a full-color

image from a Bayer CFA image. This algorithm uses the variance of color differences as a supplementary criterion to determine the interpolation direction for estimating the missing green components. The missing red and blue components are then estimated based on the interpolated green plane. This algorithm can effectively preserve the details in texture regions and, at the same time, can significantly reduce the color artifacts. Simulation results show that the proposed algorithm produces superior demosaicing results both objectively and subjectively.

Digital zoom is another common process performed in a digital camera. Basically it performs interpolation and hence employs similar signal processing concept as color demosaicing does. However, in a conventional system, the zooming process is generally carried out separately in the post-processing step. Accordingly, the information available on the raw sensor data is not always utilized consistently and efficiently to yield the enlarged output image. To remedy such inefficiency, a joint color demosaicing and digital zooming algorithm is proposed for digital cameras to produce a high quality zoomed output at a reduced computation cost. This algorithm directly extracts the edge information from raw sensor data for interpolation in both demosaicing and zooming to preserve edge features in its output. It allows the extracted information to be exploited consistently in both stages and also efficiently as no separate extraction process is required in different stages. This algorithm can produce a zoomed full-color image as well as a zoomed Bayer CFA image with outstanding performance as compared with conventional approaches which generally combine separate color demosaicing and digital zooming schemes in a straightforward manner.

Simulation results show that our first proposed demosaicing algorithm can provide a high quality output. However, it may not be suitable for real-time realization in a low-profile camera as its computational complexity is comparatively high.

Inspired by the idea of sharing edge information in the realization of the joint demosaicing and zooming algorithm, a more advanced and efficient demosaicing algorithm is proposed for constructing a full-color image. This algorithm exploits a new edge-sensing measure called integrated gradient (IG) to effectively extract gradient information in both color intensity and color difference domains simultaneously. This measure is reliable and supports full resolution, which allows one to interpolate the missing samples along an appropriate direction and hence directly improves the demosaicing performance. By sharing IG in different demosaicing stages to guide the interpolation of various color planes, it guarantees the consistency of the interpolation direction in different color channels and saves the effort required to repeatedly extract gradient information from intermediate interpolation results at different stages. An IG-based green plane enhancement is also proposed to further improve the efficiency of the algorithm. Simulation results confirmed that this proposed demosaicing algorithm outperforms other up-to-date demosaicing algorithms including our first proposed demosaicing algorithm in terms of output quality at a complexity of around 80 arithmetic operations per pixel.

The lossless compression of Bayer CFA images is finally addressed in this thesis. Since a Bayer CFA image acts as a “digital negative” and can be used as an ideal original archive format, lossless compression of Bayer CFA images is highly preferred especially in the field of high-end digital photography. However, as different color channels are interlaced in a Bayer CFA image, the spatial correlation of a Bayer CFA image is seriously damaged. Conventional lossless image compression schemes such as JPEG-LS and JPEG2000 cannot make use of the spatial correlation to attain a good compression performance. A prediction-based lossless compression scheme is proposed in this thesis to effectively de-correlate the pixel redundancy in a Bayer CFA image. This scheme exploits a context matching

technique to rank the neighboring pixels when predicting a pixel, an adaptive color difference estimation scheme to remove the color spectral redundancy when handling red and blue samples, and an adaptive codeword generation technique to adjust the divisor of Rice code when encoding the prediction residues. Simulation results show that the proposed compression scheme can achieve a better compression performance than conventional lossless Bayer CFA image coding schemes.

Author's Publications

(List of publications of the author on which this thesis is based on)

International Journal Papers

1. K. H. Chung and Y. H. Chan, "Color demosaicing using variance of color differences," *IEEE Transactions on Image Processing*, vol. 15, no. 10, pp. 2944–2955, Oct. 2006.

2. K. H. Chung and Y. H. Chan, "A low-complexity joint color demosaicking and zooming algorithm for digital camera," *IEEE Transactions on Image Processing*, vol. 16, no. 7, pp. 1705–1715, Jul. 2007.

3. K. H. Chung and Y. H. Chan, "A lossless compression scheme for Bayer color filter array images," *IEEE Transactions on Image Processing*, vol. 17, no. 2, pp. 134–144, Feb. 2008.

4. K. H. Chung and Y. H. Chan, "On the use of integrated gradient for color demosaicing," *IEEE Transactions on Image Processing* (revised).

International Conference Papers

5. K. H. Chung and Y. H. Chan, "An adaptive color filter array interpolation for digital camera", *IEEE International Conference on Image Processing* (*ICIP'06*), Atlanta, GA, USA, Oct 8–11, 2006.

6. K. H. Chung, Y. H. Chan, C. H. Fu and Y. L. Chan, "An efficient combined demosaicing and zooming algorithm for digital camera", *IEEE International Conference on Image Processing* (*ICIP'07*), San Antonio, Texas, USA, Sep 16–19, 2007.

7. K. H. Chung, Y. H. Chan, C. H. Fu and Y. L. Chan, "A high performance lossless Bayer image compression scheme", *IEEE International Conference on Image Processing* (*ICIP'07*), San Antonio, Texas, USA, Sep 16–19, 2007.

8. K. H. Chung and Y. H. Chan, "Lossless compression scheme for Bayer CFA images", *European Signal Processing Conference* (*EUSPICO'07*), Poland, Europe, Sep 3–7, 2007.

Acknowledgements

I would like to take this opportunity to express my sincere gratitude to various people, particular to whom from the Hong Kong Polytechnic University where I have the opportunity to work with.

First and foremost, I would like to thank my supervisor, Dr. Chris Yuk-Hee Chan. He is my role model of an exceptional scientist and teacher. His patient guidance and professional comments enable me to carry out my research in the best possible way. Without his continuous support and encouragement, I would not have been able to complete this thesis. It is extremely a rare privilege and great honor to work with and study form him.

Second, I would like to thank to my colleagues and friends. Especially to my research team member Dr. Y.H. Fung, he gave me many precious advices on my research works. I gain various benefits from the countless discussions with him on every stage of my study. Thanks are also given to Dr. K.O. Cheng, Dr. S.C. Tan, Mr. W.H. Wong, Mr. W.L. Hui and all other people in DSP lab. They fruit up and inspire me in various aspects towards life and research while discussing and living with them.

All staff members in the Department of Electronic and Information Engineering and in the Research Grants Council of the Hong Kong Special Administrative Region are also greatly appreciated. Their supports make this research possible.

Last, but not least, I would like to dedicate this work to my family. Particularly to my wife, I offer her my deepest appreciation for believing in and encouraging me during the difficult times, for patiently enduring me long hours spent at the lab, and for being my constant companion during the PhD process.

Table of Contents

List of Figures

List of Tables

Statement of Originality

The following contributions reported in this thesis are claimed to be original.

1. The study on how the conventional directional filtering-based demosaicing algorithm destroys the edge features in an image and introduces visual color artifacts into the demosaiced output. (Chapter 3, Section 3.2)

2. The adaptive demosaicing algorithm that determines the interpolation direction for the missing green samples based on the variance of color difference of the pixels along the horizontal and vertical axes in a local window. (Chapter 3, Section 3.3)

3. The two-pass green plane demosaicing scheme which handles the sharp edge regions and the non-sharp edge regions separately in different passes such that they can be handled more effectively with respectively suitable processing techniques. (Chapter 4, Section 4.2)

4. The joint color demosaicing and digital zooming framework which extracts edge information directly from the raw sensor data such that the extracted information can be shared in both demosaicing and zooming to interpolate the missing samples in the enlarged output image. (Chapter 4, Section 4.3)

5. The new edge-sensing measure called integrated gradient (IG), which extracts gradient information from a Bayer CFA image in both color intensity and color difference domains simultaneously for being shared at different stages throughout the demosaicing process to interpolate the missing samples. (Chapter 5, Section 5.2)

6. The green plane enhancement technique which uses the IG to enhance the demosaiced green plane before the interpolation of the red and the blue planes of a Bayer CFA image such that the interpolation of the red and the blue planes can be done with the help of a more reliable green plane. (Chapter 5, Section 5.3.2)

7. The decision-based color demosaicing algorithm which exploits the IG and the green plane enhancement technique to turn a Bayer CFA image into a full-color image at a low computation cost. (Chapter 5, Section 5.3)

8. The predictive coding technique called context matching-based prediction (CMBP), which supports lossless compression of Bayer CFA images by ranking the neighboring available samples of a pixel to determine their weights adaptively when predicting the CFA sample value of the pixel. (Chapter 6, Section 6.2)

9. The adaptive color difference estimation scheme which estimates the color difference at the red and the blue sampling positions of a Bayer CFA image for removing the color spectral redundancy when compressing the red and the blue samples of the CFA image. (Chapter 6, Section 6.3)

10. The adaptive codeword generation technique which adaptively adjusts the divisor of Rice code based on the local statistical mean of the prediction residues when encoding the residues. (Chapter 6, Section 6.4)

11. The lossless Bayer CFA image compression scheme which exploits the aforementioned CMBP technique, the adaptive color difference estimation scheme and the adaptive codeword generation technique to encode a Bayer CFA image. (Chapter 6, Section 6.4)

Chapter 1 Introduction

1.1 Structure of Digital Camera

Over the last decade, digital camera has become one of the most popular portable devices in the field of consumer electronics. Inside a digital camera, a single-sensor image acquisition system is generally employed to lower the device's cost, size and complexity. To acquire scenes in a digital format, a camera first allows light to pass through an optical system, where focusing, shutter control, and optical zoom are performed. The light information is then sampled by an image sensor where the optical signal is converted to electronic signal in digital form. Figure 1.1 shows a typical single-sensor image acquisition system used in a camera [1-3].

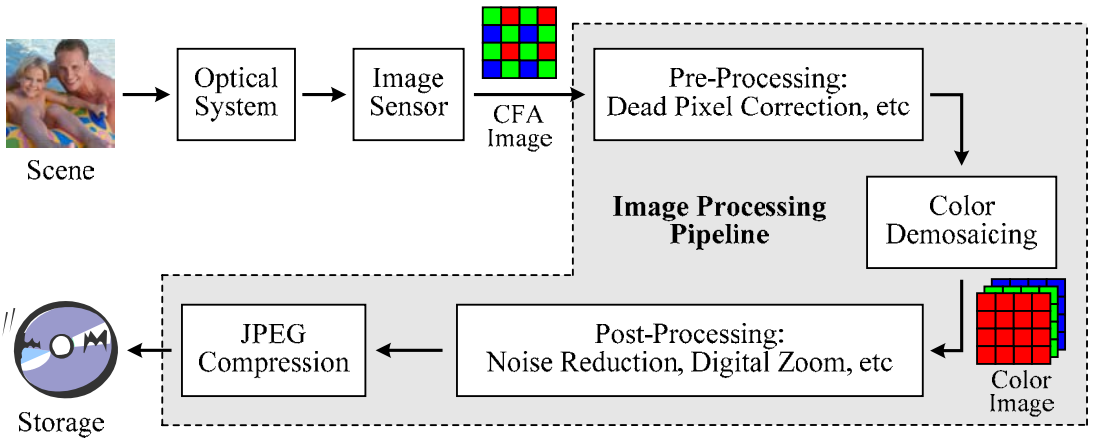


Figure 1.1 A single-sensor image acquisition system inside digital cameras

Charge coupled device (CCD) [4,5] and complementary metal oxide semiconductor (CMOS) [5,6] are the two common types of image sensors used in a acquisition system. These sensors consist of a two-dimensional (2-D) light-sensitive element array. It measures the amount of incident light at each array location, where

the pixel intensity is digitized, to construct an image. Since each light-sensitive element is a monochromatic device which senses the light signal within the same frequency range, it is not allowed to acquire color information directly with the sensor. To make it output color data, a "mosaic" pattern of color filters, called color filter array (CFA), is positioned on top of the sensor to filter out two of the red, green, and blue components of the light falling onto the light-sensitive elements. Consequently, each element and hence each pixel holds only one of the three primary colors and, as a result, a mosaic gray-scale like image is constituted as the sensor output.

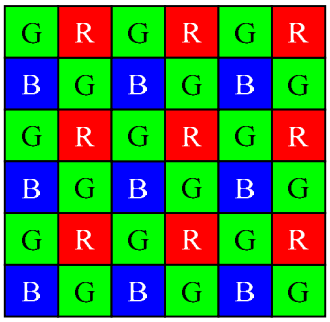


Figure 1.2 Bayer CFA pattern with green-red-green-red phase in the first row

Many CFA patterns are suggested in the literature [7,8]. Among them, the Bayer CFA pattern [9], as shown in Figure 1.2, is the most popular one because it provides the most optimal spatial arrangement of the three primary colors on a square grid [10]. Under the Bayer CFA configuration, green color is measured in a quincunx form with a sampling rate double that of the red and blue channels. Red and blue colors are sampled in a rectangular grid, which interlaces with the green sampling grid, occupying one-fourth area of the whole image individually. The green channel is sampled at a higher sampling rate because the spectral response of green color is close to the luminance response of human visual system [11,12]. This allows the camera to

maintain the maximum perceived sharpness in the luminance channel. The red and the blue colors, on the contrary, serve as chrominance channels for color reproduction.

Though there are some other CFA patterns, the Bayer CFA pattern is the focus in the discussion of this thesis as it is the most popular CFA pattern nowadays and it is easy to modify a processing algorithm for Bayer CFA images to handle images using other CFA patterns. Accordingly, hereafter in this thesis, whenever a CFA image is referred to, a Bayer CFA image is actually referred to unless specified otherwise.

Since the sensors record a color component at each pixel location, a sub-sampled or incomplete image is provided in the sensor output. To make the image become a full-color image, a series of imaging processes have to be carried out in the acquisition pipeline. Figure 1.1 shows a typical processing pipeline used in an image acquisition system. As shown in the figure, the raw CFA data is first pre-processed to remove the defective pixels generated because of malfunctioning of the light-sensitive elements. The pre-processed mosaic CFA image is then converted to a full-color image. This process is called color interpolation or color demosaicing [13]. In practice, it estimates the two missing color components of a pixel from adjacent pixels. The demosaiced full-color image is post-processed next. In post-processing, imaging operations like color correction, noise reduction, digital zoom, edge sharpening, white balance (WB), etc are carried out to improve the visual quality. Finally, the enhanced image is compressed, in JPEG format in general, for storage or transmission.

Recently, this conventional processing pipeline is found to be sub-optimal from the compression point of view. It is because the demosaicing process carried out in the early stage always introduces some redundancy which should be removed in the later compression stage. The task of compression is always made difficult and inefficient accordingly.

To solve this problem, an alternative processing pipeline, as shown in Figure 1.3,

has been proposed in [14-17] recently. This new pipeline allows one to compress the raw sensor image directly without prior demosaicing. Under the new pipeline configuration, more sophisticated demosaicing and post-processing algorithms can be applied and carried out offline in a powerful personal computer to produce a more visually pleasing color output. Besides, as the compression step handles the CFA image whose data size is only one-third of that of a full-color image, the new compression-then-demosaicing configuration can effectively increase the pipeline throughput without degrading the output image quality. It has been proven that this new pipeline outperforms the conventional one when the quality requirement of the color output is high [18].

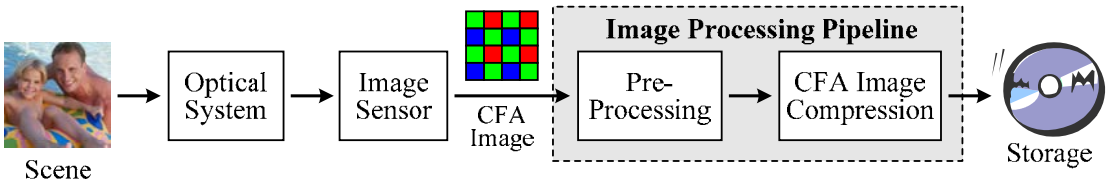


Figure 1.3 The new imaging pipeline added in many prosumer and professional grade digital cameras as an optional imaging path

As a matter of fact, the new pipeline has been adapted in many prosumer and professional grade digital cameras. It serves as an optional imaging path to allow the cameras to deliver a precise and high quality output in a more efficient way. Although the pipeline can provide a satisfactory performance, there is still room for improvement. It is highly desired to investigate some more sophisticated imaging algorithms and camera technologies for this new pipeline.

1.2 The Addressed Problems

The most challenging issue encountered in designing a digital camera is to yield a precise and high quality output image in a short period of time with low power consumption. To achieve this goal, besides upgrading a camera's hardware specification, optimizing its processing pipeline is alternatively a more flexible and cost effective approach. In this thesis, three imaging processes in the processing pipeline including color demosaicing, digital zooming and image compression are addressed. These processes determine the output quality of a camera and hence are important to a camera.

Color demosaicing, as mentioned before, is the process of interpolating two missing color components for each pixel based on the available component and neighboring pixels in a CFA image. This process is critical to a digital camera as it is responsible for the production of a full-color image. In other words, it determines the output quality of the camera. Generally, simple demosaicing algorithms use non-adaptive interpolation techniques to estimate the missing samples. However, these simple demosaicing algorithms always reduce the sharpness of the color output while increasing the visibility of false color artifacts. To achieve higher visual quality output, advanced demosaicing algorithms tend to introduce some more complex operations to estimate the missing samples adaptively. These advanced algorithms can provide a superior result as compared with the non-adaptive algorithms. However, they require high computation effort in general.

Digital zoom is one of the most commonly performed processing operations in a digital camera. To obtain an enlarged full-color image from a small CFA image, the CFA image is either 1) firstly demosaiced and then enlarged or 2) firstly enlarged and then demosaiced. However, no matter which approach is used, the zooming and the

demosaicing processes are executed separately in the processing pipeline. As the actual process behind the zooming and the demosaicing processes is interpolation, similar signal processing techniques are employed in both processes. It would be a good idea to carry out both processes at the same time from efficiency point of view. Besides, if the two processes are performed separately, the information available on the raw sensor data cannot be utilized consistently and efficiently to produce the enlarged output. Accordingly, combining the zooming and the demosaicing processes seems to be a more efficient and cost-effective approach as compared with carrying out them separately.

In the new processing pipeline, the sensor raw CFA data is first compressed before color demosaicing. Whenever it is necessary, the compressed CFA data is decoded and then demosaiced offline to produce a full-color output. This arrangement motivates the need of compression technique for the CFA images. Generally, the compression of a CFA image can be either lossy or lossless. Lossy compression reduces the image data size by discarding the visually redundant information. It offers a higher compression ratio in general as compared with lossless compression, but only an approximation of the original data can be reconstructed afterwards. Lossless compression, on the contrary, allows one to reconstruct the exact original data from the compressed data. It offers an ideal original archive format for high-end photography applications to produce a high quality full-color output from a raw CFA image. Many standard lossless image compression schemes such as JPEG-LS [19] and JPEG2000 [20] can be used to encode a CFA image. However, these schemes are generally proposed for coding natural gray-level images. Because of the interlaced arrangement of the color samples in a CFA image, the spatial correlation among adjacent pixels is weak and hence only a fair performance can be attained by using these schemes.

In this thesis, based on our studies on the structure of the new processing pipeline and the characteristics of a CFA image, four processing algorithms are proposed to tackle the problems. Specifically, two of them are for color demosaicing, one is for joint demosaicing and zooming, and one is for lossless CFA image compression. These algorithms deal executively with Bayer CFA images and aim at improving the performance of the processing pipeline in a camera.

1.3 Thesis Outline

This thesis is structured as follows. In Chapter 2, a brief literature review of relevant works is provided, on which the present works are based. Some important background information on color demosaicing, digital zoom and lossless CFA image compression techniques are covered in this chapter.

In Chapter 3, a study is first carried out to see why visual artifacts are always introduced and why image features cannot be preserved during demosaicing when the conventional adaptive color plane interpolation algorithm [21,22] is used. Based on the study result, a color difference variance-based demosaicing algorithm is presented. This algorithm not only can preserve image features but also can produce color outputs with less color artifacts. It achieves outstanding performance both subjectively and objectively.

In Chapter 4, a low-complexity joint color demosaicing and zooming algorithm is presented. By sharing the edge information extracted directly from a CFA image, missing color samples are interpolated consistently and efficiently in both demosaicing and zooming. This algorithm is proposed for low-profile portable capturing devices, where no optical zooming system is equipped, to produce full-color images with superior visual quality.

In Chapter 5, inspired by the joint algorithm presented in Chapter 4, another demosaicing algorithm is proposed. This algorithm aims at producing high quality output images at a low computation cost. A new gradient measure is defined for being shared in various stages throughout the demosaicing process to interpolate different color channels. Simulation results confirm that, as even compared with the demosaicing algorithm proposed in Chapter 3, this algorithm produces output of superior quality at a relatively low computation cost.

In Chapter 6, a lossless CFA image compression scheme is presented. A context matching-based prediction technique (CMBP) is introduced in the scheme to de-correlate the pixel dependency. The prediction residue is then encoded with context-based entropy coding technique in which Rice code is exploited adaptively. This scheme can effectively remove the CFA data redundancy with a reasonable computational effort.

At last, in Chapter 7, a brief conclusion is given. All the contributions made in this thesis are summarized. Some future possible extensions of the present work are also discussed in the chapter.

Chapter 2 A Comprehensive Literature Review

2.1 Introduction

This chapter reviews some existing works that are relevant to our present works. In Section 2.2, the background of color demosaicing and some common color interpolation algorithms are presented. In Section 2.3, various digital zooming approaches and some state-of-art zooming techniques are described. Finally, in Section 2.4, a brief review of various Bayer CFA image compression schemes is provided.

2.2 Color Demosaicing

As mentioned in Chapter 1, color demosaicing is the process to restore a full-color image from a CFA image. This process determines the output quality of a camera and, hence, is critical to a digital camera.

There are many demosaicing algorithms proposed in the literature [13]. In general, a demosaicing algorithm can be classified as either heuristic or non-heuristic. A heuristic approach does not try to solve a mathematically defined optimization problem while a non-heuristic approach does. The following sub-sections provide reviews on the demosaicing techniques of these two approaches respectively.

2.2.1 Heuristic Approach

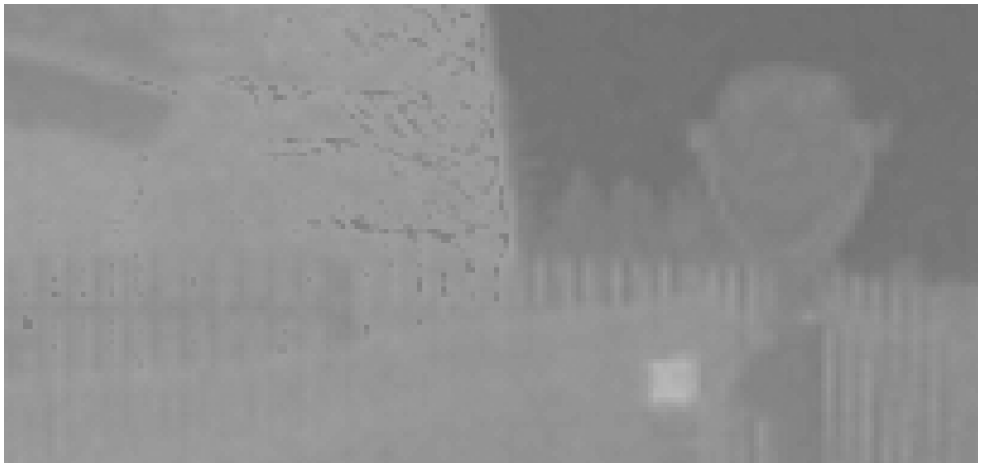
Most existing demosaicing algorithms are heuristic. They generally involve filtering operations that are based on assumptions about color images. Early heuristic

demosaicing algorithms include nearest neighbor replication, bilinear interpolation and bicubic interpolation[23]. These algorithms basically interpolate each color plane separately and then combine the three independently interpolated planes to produce the full-color image. Although these plane-wise demosaicing algorithms are very simple and easy for implementation, they generally introduce severe artifacts such as blurring, false color and zipper effect around edges. As a reference, Figure 2.1 shows a simulation result of the bilinear interpolation. From the result, one can see clearly the visual artifacts.

(a)

(b)

Figure 2.1 (a) original full-color image and (b) bilinear demosaiced image



(a)

(b)

Figure 2.2 (a) G-R and (b) G-B difference planes of the full-color image shown in Figure 2.1a

To diminish the artifacts, more advanced demosaicing algorithms [24-26] exploit the spectral correlation between color channels. They found that hues of a full-color image such as green-to-red and green-to-blue differences vary smoothly within an object. As an example, Figure 2.2 shows two color difference planes of the full-color image shown in Figure 2.1a, which demonstrates the smoothness of the hue planes. Based on this finding, these algorithms estimate the missing red and blue samples by linearly interpolating the hues. Generally, these smooth hue transition-based demosaicing algorithms provide sharper edges as well as lesser color artifacts as

compared with the plane-wise demosaicing algorithms. Nevertheless, for pixels in sharp edges or in fine detail areas where the assumed spectral correlation does not hold, large interpolation errors are still produced. Figure 2.3 shows the interpolation result produced by a smooth hue transition-based demosaicing algorithm, which illustrates the large errors in the fence regions.

Figure 2.3 Simulation result generated by the smooth hue transition-based demosaicing algorithm [24]

To preserve edge structures to where human visual system is sensitive, many adaptive demosaicing algorithms try to perform the interpolation along the edges. In [21,22], Hamilton and Adams propose the well-known second order directional Laplacian interpolation filter for interpolating the missing samples. In their algorithm, the interpolation direction is selected based on the local pixel intensity gradients of a Bayer image. Figure 2.4 shows the demosaicing result generated with the directional Laplacian filter. It reveals that the directional Laplacian filter can produce a better result than the plane-wise and the smooth hue transition-based interpolation algorithms. However, it still cannot completely recover the image details.

Figure 2.4 A demosaiced image produced by applying the directional Laplacian interpolation filter

Figure 2.5 Result of a weighted average-based demosaicing algorithm

In [27-34], the idea of directional interpolation is extended to allow interpolating along edges in any orientation. They calculate weights on the basis of edge directions and interpolate the missing samples by assigning the weights to their neighborhoods. These weighted average demosaicing algorithms generally provide satisfactory results around edges. But for pixel in fine detailed or textured areas where edges are in different directions or not well-defined, these algorithms always introduce undesirable errors. Figure 2.5 shows the simulation result provided by the weighted average demosaicing algorithm [30], which as an example demonstrates the undesirable

errors.

Recently, the directional Laplacian filter has been proven to be a good approximation of the optimal interpolator for Bayer images in one-dimension [35]. Many demosaicing algorithms try to orient the filter horizontally, vertically or sometimes diagonally to give the interpolations with fewer artifacts. These algorithms utilize the smooth property of the color difference signals as prior knowledge to determine the filter orientation. Specifically, they do it by comparing the smoothness of color difference signals in various directions. The demosaicing algorithms proposed in [35-38] are some examples of this decision-based approach. In [35], Hirakawa and Parks first produce two full-color images by horizontally and vertically filtering the CFA image with the Laplacian filter. At each pixel, the final estimate is determined by picking one of the estimates from the two interpolated full-color images at the same position with the least misguidance level of the images as the selection criterion. In [36], Wu *et al*. adopt the pixel selection framework and apply Fisher's discriminant to preserve the highest correlation of color difference signals in the output. In [37], Menon *et al*. compute the local color difference gradients in both horizontal and vertical directions and then determine the interpolation direction with the least color difference gradients. In [38], Tsai and Song propose a hard-decision scheme which selects the direction by comparing the horizontal and vertical heterogeneity values of the pixels. In general, these decision-based demosaicing algorithms perform well in most image areas. They can preserve the edge features even in pattern regions. As an example, Figure 2.6 shows a demosaiced result yielded by the algorithm proposed in [37], which is an algorithm in this category. At the moment, the decision-based demosaicing algorithms are the most effective, in the sense of quality and complexity, for reproducing a full-color image from a CFA image.

Figure 2.6 Simulation result of the decision-based demosaicing algorithm [37]

2.2.2 Non-Heuristic Approach

Some demosaicing algorithms provide solutions in a non-heuristic approach. They interpolate the missing samples by solving a mathematically defined optimization problem. References [10,12,39-42] are some examples of this approach. Specifically, Gunturk *et al.* in [12] uses the Projection-Onto-Convex-Sets (POCS) technique to maintain the output image within the "observation" and the "detail" constraint sets. Li [39] approximates the estimates in color difference (G-R and G-B) domains iteratively with a spatially adaptive stopping criterion. Alleyson *et al.* [10] and Lian *et al.* [40] derive the optimal filters in Fourier domain for recovering the luminance and chrominance terms of the image. Zhang and Wu [41] estimate the missing samples by fusing the primary difference signals using the linear minimum mean square-error estimation (LMMSE) technique, while Muresan and Parks [42] introduces a nonlinear interpolation scheme based on edge information. Generally, these non-heuristic demosaicing algorithms offer a more accurate estimation as compared with the edge-directed algorithms in heuristic approach. As an example, Figure 2.7 shows a result of the non-heuristic demosaicing algorithm proposed in [12] for visual comparison.

Figure 2.7 Demosaiced image produced by the non-heuristic algorithm [12]

2.3 Digital Zoom

Along with color demosaicing, zooming is probably the most commonly performed processing operation in a digital camera. Under practical size and power constraints, a typical portable device is generally not equipped with a complex optical system to achieve high optical zooming power as such an optical system is expensive and sizable. For this reason, digital zooming is usually used to enhance the zooming power of a digital camera.

2.3.1 Zooming-after-Demosaicing Approach

Various approaches can be exploited to produce a full-color zooming result from a CFA image. Traditionally, it is done by performing a digital zooming process after demosaicing the CFA image as shown in Figure 2.8a. In this zooming-after-demosaicing approach, the zooming process is performed on the demosaiced full-color image either in a component-wise manner [43-46] or in a vector manner [47-50].

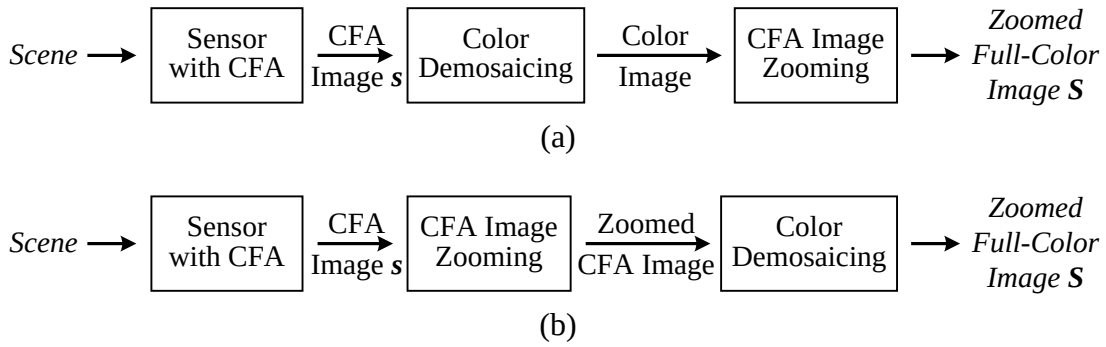


Figure 2.8 Image zooming approaches: (a) zooming-after-demosaicing and (b) demosaicing-after-zooming

In a component-wise interpolation, each color plane is treated as a gray-level image and interpolated with a gray-scale image interpolation technique independently. The zoomed full-color image is then formed by combining the three interpolated color planes. As for a vector color interpolation, each pixel in the image forms a color vector with the three color components as elements. Various vector operations are then performed on the color vectors to produce the zoomed full-color image. As the interpolation carried out in the zooming process is based on the demosaicing result and demosaicing may introduce artifacts such as blurred edges and false colors [51,52], the quality of the resultant zoomed image can be very poor in regions of complicated details.

2.3.2 Demosaicing-after-Zooming Approach

In recent years, an alternative approach as shown in Figure 2.8b is used instead. In this approach, a zooming process is directly applied to the CFA image to generate an enlarged CFA image such that a conventional demosaicing technique can be performed on the enlarged CFA image to produce a zoomed full-color image. In [53-55], one can find three of the successful examples. A linear interpolation-based digital zooming algorithm and an adaptive edge sensing-based digital zooming

algorithm are, respectively, introduced in [53] and [54] to operate on the raw sensor data to produce an enlarged CFA image for further demosaicing. In [55], local color ratio and edge-sensing weight coefficients are utilized wherever necessary in CFA zooming, demosaicing and post-processing to produce the final zoomed full-color image.

2.4 Color Filter Array Image Compression

As mentioned in Chapter 1, a CFA image in general is first interpolated via a demosaicing process to form a full-color image before being compressed for storage. Figure 2.9a shows the workflow of this imaging chain.

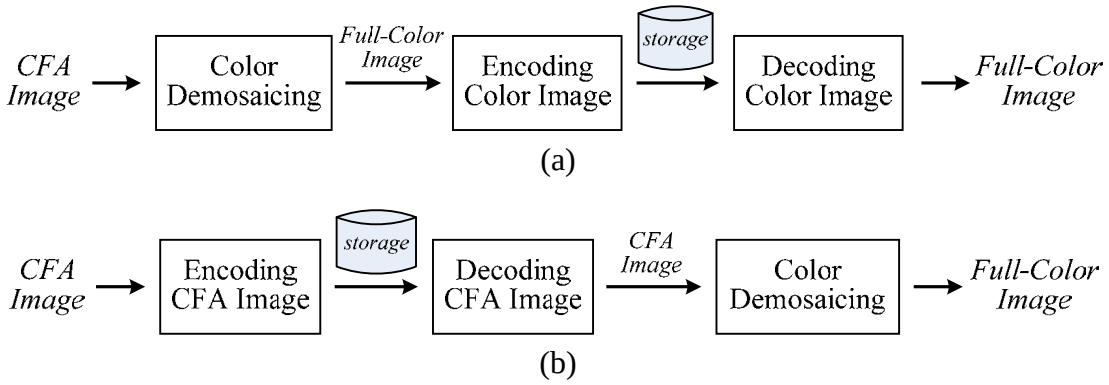


Figure 2.9 Single-sensor camera imaging chain: (a) the demosaicing-first scheme, (b) the compression-first scheme

Recently, some reports [14-18] indicated that such a demosaicing-first processing sequence was inefficient in a way that the demosaicing process always introduced some redundancy which should eventually be removed in the following compression step. As a result, an alternative processing sequence [14-17] which carries out compression before demosaicing as shown in Figure 2.9b has been proposed lately. Under this new strategy, digital cameras can have a simpler design

and lower power consumption as computationally heavy processes like color demosaicing can be carried out in an offline powerful personal computer. This motivates the demand of CFA image compression schemes.

There are two categories of CFA image compression schemes: lossy and lossless. Lossy schemes compress a CFA image by discarding its visually redundant information. These schemes usually yield a higher compression ratio as compared with the lossless schemes. Schemes presented in [14-18,56-61] are some examples of this approach. In these schemes, different lossy compression techniques such as discrete cosine transform (DCT) [56], vector quantization (VQ) [57,58] sub-band coding with symmetric short kernel filters [14], transform followed by JPEG or JPEG 2000 [16,17,59-61] and low-pass filtering followed by JPEG-LS or JPEG 2000 (lossless mode) [15] are used to reduce data redundancy.

In some high-end photography applications such as commercial poster production, original CFA images are required for producing high quality full-color images directly. They serve as a digital version of film negative and make an ideal original archive format. In such cases, lossless compression of CFA images is necessary. Some standard lossless image compression schemes like JPEG-LS [19] and JPEG2000 [20] can be used to encode a CFA image but only a fair performance can be attained. Recently, an advanced lossless CFA image compression scheme (LCMI) [62] was proposed. In the scheme, the mosaic data is first de-correlated by the Mallat wavelet packet transform. Then, the de-correlated wavelet coefficients are compressed by adaptive Rice code. This scheme offers a higher compression performance than the JPEG-LS and JPEG2000 lossless image compression schemes at a lower complexity cost.

Chapter 3 Color Demosaicing Algorithm I: Color Demosaicing Using Variance of Color Differences

3.1 Introduction

As mentioned in Chapter 2, a number of heuristic demosaicing algorithms, to a certain extent, were developed based on the framework of the adaptive color plane interpolation algorithm (ACPI) proposed in [21,22]. For examples, algorithms in [35-37] exploit the same interpolators used in ACPI to generate the green plane. The improved demosaicing performance of these advanced heuristic algorithms is generally achieved by that they can interpolate the missing samples along a correct direction. In this chapter, based on the framework of ACPI, a new heuristic demosaicing algorithm is proposed. This algorithm uses the variance of pixel color differences to determine the interpolation direction for interpolating the missing green samples. Simulation results showed that the proposed algorithm was superior to other state-of-art demosaicing algorithms in terms of both subjective and objective criteria when it was proposed. In fact, it is still one of the best color demosaicing algorithms at the moment. In particular, it can outstandingly preserve the texture details in an image.

This chapter is organized as follows. In Section 3.2, the ACPI algorithm [22] is revisited. An analysis is made and our motivation to develop the proposed algorithm is presented. In Section 3.3, the details of our demosaicing algorithm are presented while, in Section 3.4, some simulation results are presented for comparison study. Finally, in Section 3.5, a brief conclusion is given.

3.2 Observations on the Adaptive Color Plane Interpolation Algorithm

In ACPI [21,22], the green plane is handled first and the other color planes are handled based on the estimation result of the green plane. When the green plane is processed, for each missing green component in the CFA, the algorithm performs a gradient test and then carries out an interpolation along the direction of smaller gradient to determine the missing green component.

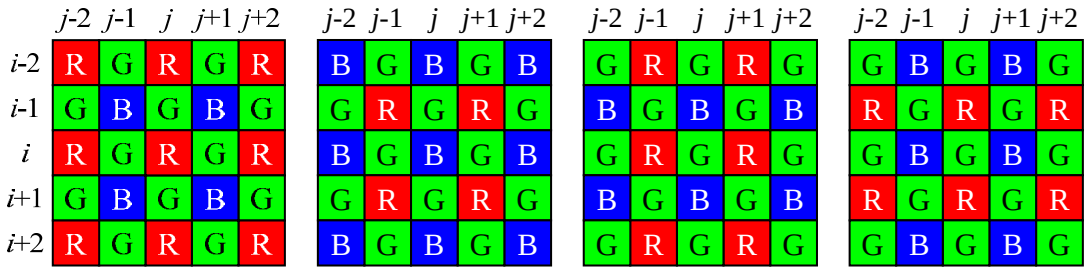


Figure 3.1 Four 5×5 regions of Bayer CFA pattern having their centers at (a) red, (b) blue, (c)-(d) green CFA samples

A pixel without green component in the Bayer CFA may have a neighborhood as shown in either Figure 3.1a or 3.1b. Without losing the generality, here we consider the former case only. In this case, the horizontal gradient $\Delta H_{i,j}$ and the vertical gradient $\Delta V_{i,j}$ at position (i,j) are estimated first to determine the interpolation direction as follows.

$$\Delta H_{i,j} = \left|G_{i,j-1} - G_{i,j+1}\right| + \left|2R_{i,j} - R_{i,j-2} - R_{i,j+2}\right| \tag{3.1}$$

$$\Delta V_{i,j} = \left|G_{i-1,j} - G_{i+1,j}\right| + \left|2R_{i,j} - R_{i-2,j} - R_{i+2,j}\right| \tag{3.2}$$

where $R_{m,n}$ and $G_{m,n}$ denote the known red and green CFA components at position

(m,n). Based on the values of $\Delta H_{i,j}$ and $\Delta V_{i,j}$, the missing green component $g_{i,j}$ in Figure 3.1a is interpolated as follows.

$$g_{i,j} = g_{i,j}^{H} \equiv \frac{G_{i,j-1} + G_{i,j+1}}{2} + \frac{2R_{i,j} - R_{i,j-2} - R_{i,j+2}}{4} \qquad \text{if } \Delta H_{i,j} < \Delta V_{i,j} \qquad (3.3)$$

$$g_{i,j} = g_{i,j}^{V} \equiv \frac{G_{i-1,j} + G_{i+1,j}}{2} + \frac{2R_{i,j} - R_{i-2,j} - R_{i+2,j}}{4} \qquad \text{if } \Delta H_{i,j} > \Delta V_{i,j} \qquad (3.4)$$

$$g_{i,j} = g_{i,j}^{D} \equiv \frac{g_{i,j}^{V} + g_{i,j}^{H}}{2} \qquad \text{if } \Delta H_{i,j} = \Delta V_{i,j} \qquad (3.5)$$

In fact, it is shown in [35] that eqns. (3.3) and (3.4) are the approximated optimal CFA interpolators in horizontal and vertical directions and eqn. (3.5) is good enough for interpolating diagonal image features.

Since the red and the blue color planes are determined based on the estimation result of the green plane and the missing green components are in turns determined by the result of the gradient test, the demosaicing result highly relies on the success of the gradient test. A study was performed here to evaluate how significant the gradient test is to the performance of the algorithm.

In a simulation of our study, twenty-four 24-bit (of bit ratio R:G:B=8:8:8) digital color images of size 512×768 pixels each as shown in Appendix A were sampled according to Bayer CFA to form a set of testing images. These images are part of the Kodak color image database and include various scenes. The testing images were reconstructed with ACPI [22] and the ideal ACPI. The ideal ACPI is basically ACPI except that, in determining a missing green component, it computes all $g_{i,j}$ estimates with eqns. (3.3)-(3.5) and picks the one closest to the real value of the missing component without performing any gradient test. Note that in this simulation all original images are known and hence the real value of a missing green component of

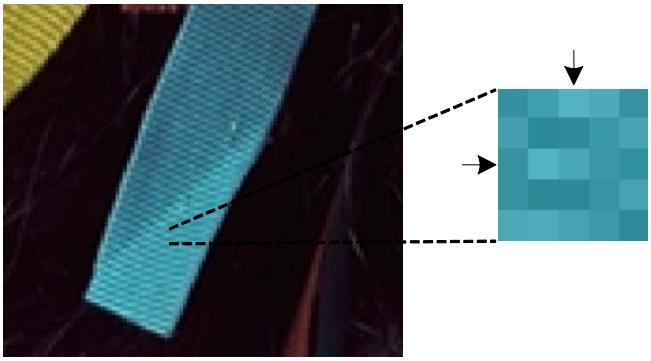
the testing images can be used to implement the ideal ACPI. In practice, the original images are not known and hence the performance of the ideal ACPI is practically unachievable. The ideal ACPI is only used as a reference in our study.

We measured the peak signal-to-noise ratio (PSNR) of the interpolated green planes of both ACPI [22] and the ideal APCI with respect to the original green plane. We found that the average PSNR achieved by the ideal ACPI was 43.83dB while that achieved by ACPI was only 38.18dB. This implies that there is a great room for improving the performance of ACPI. Based on this simulation result, we have two observations. First, the interpolators used in [22] can be very effective if there is an 'effective' gradient test to provide some reliable guidance for the interpolation. Second, the current gradient test used in [22] is not good enough.

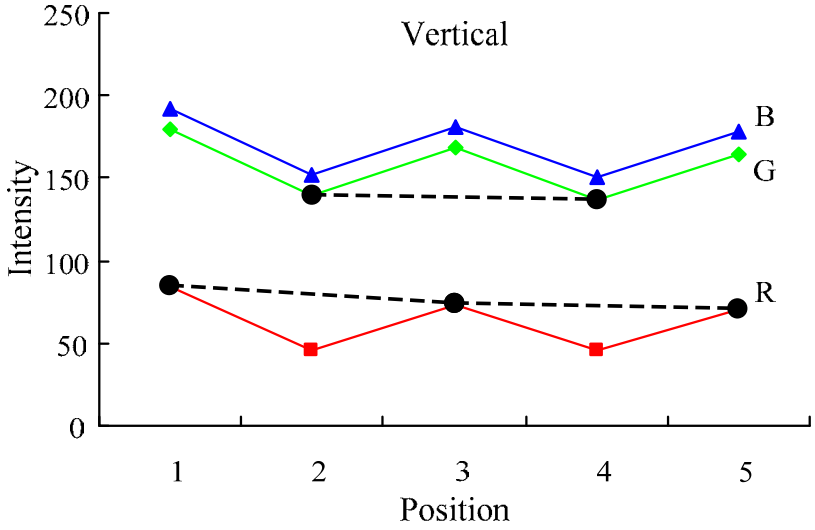
After having the 'ideal' green plane with the ideal ACPI, we proceeded to interpolate the red and the blue planes with it to produce a full-color image with the same procedures as originally proposed in ACPI. The quality of the output was measured in terms of color-peak signal-to-noise ratio (CPSNR) which is defined in eqn. (3.24). As expected, it achieves extremely high score and, subjectively, it is hard to distinguish the recovered image from the original full-color image. Table 3.1 shows the performance of various algorithms for comparison. As shown in Table 3.1, the ideal ACPI provides a very outstanding performance as compared with any other evaluated demosaicing algorithms. This shows that the approach used in [22] to derive the other color planes with a 'good' green plane is actually very effective. As a good green plane relies on a good gradient test, the key of success is again the effectiveness of the gradient test or, to be more precise, the test for determining the interpolation direction. This finding motivates the need to find an effective and efficient gradient test to improve the performance of ACPI.

Image	Non-Heuristic Algorithms			Heuristic Algorithms									
	AP [12]	**DUOR [42,63]**	**DSA [39]**	**BI [23]**	**ACPI [22]**	**ECI [26]**	**PCSD [36]**	**DAFD [28]**	**EECI [30]**	**AHDA [35]**	**VCD w/o Refine.**	**VCD w/ Refine.**	**Ideal ACPI**
1	37.70	34.66	38.32	26.21	33.59	33.81	36.40	35.92	37.99	35.16	35.97	38.53	38.83
2	39.57	39.22	39.95	33.08	39.00	38.88	39.67	39.70	40.49	39.19	39.68	40.43	42.32
3	41.45	41.18	41.18	34.45	40.50	40.87	41.77	41.00	42.64	41.59	41.72	42.54	43.69
4	40.03	38.56	39.55	33.46	38.50	39.43	39.53	39.64	40.51	38.94	39.60	40.50	41.51
5	37.46	35.25	36.48	26.51	34.79	35.29	37.17	36.15	38.04	35.74	36.15	37.89	38.81
6	38.50	37.87	39.08	27.66	34.79	34.95	38.86	36.94	38.10	37.57	38.01	40.03	40.64
7	41.77	41.39	41.50	33.38	40.84	40.54	41.48	41.03	42.73	40.92	41.02	42.15	43.69
8	35.08	31.04	35.87	23.39	32.04	30.52	34.48	33.34	35.20	33.77	34.25	36.41	36.93
9	41.72	41.27	41.94	32.16	40.15	39.55	41.99	40.73	42.58	41.09	41.63	43.04	44.01
10	42.02	40.39	41.80	32.38	39.84	40.30	41.75	41.13	42.52	40.71	41.20	42.51	43.48
11	39.14	37.42	38.92	28.96	35.97	36.24	38.57	38.09	39.46	37.53	37.99	39.86	40.94
12	42.51	42.30	42.37	33.27	40.53	40.02	42.61	41.13	42.63	41.75	42.09	43.45	44.22
13	34.30	31.08	34.91	23.79	29.65	31.33	32.85	33.64	34.38	31.52	32.32	34.90	35.15
14	35.60	35.58	34.52	29.05	35.43	35.43	35.44	35.52	37.13	35.49	36.08	36.88	38.96
15	39.35	37.77	38.97	33.04	37.61	38.94	38.93	39.09	39.49	38.03	38.95	39.78	41.15
16	41.76	41.82	41.60	31.13	38.33	37.91	42.73	40.14	41.16	41.40	41.64	43.64	44.02
17	41.11	39.06	40.97	31.80	38.28	39.26	40.49	40.44	41.36	39.42	39.84	41.21	42.50
18	37.45	35.28	37.27	28.30	34.38	35.79	36.35	37.01	37.73	35.31	35.81	37.49	38.64
19	39.46	38.06	39.96	27.84	37.27	35.29	39.54	37.39	40.13	38.48	39.28	41.00	42.00
20	40.66	39.05	40.51	31.51	38.48	38.70	40.05	40.09	41.33	39.27	39.67	41.07	42.03
21	38.66	36.22	38.93	28.38	35.13	35.72	37.36	37.62	38.96	36.55	37.12	39.12	39.88
22	37.55	36.49	37.67	30.14	36.16	36.45	37.06	37.08	38.28	36.51	37.08	37.97	39.83
23	41.88	41.34	41.79	34.83	41.70	41.68	42.15	41.62	42.91	41.85	42.22	42.89	44.46
24	34.78	32.49	34.82	26.83	32.15	33.76	34.58	34.28	34.82	33.64	34.12	35.04	36.82
Avg.	**39.15**	**37.70**	**39.12**	**30.06**	**36.88**	**37.11**	**38.83**	**38.28**	**39.61**	**37.98**	**38.48**	**39.93**	**41.02**

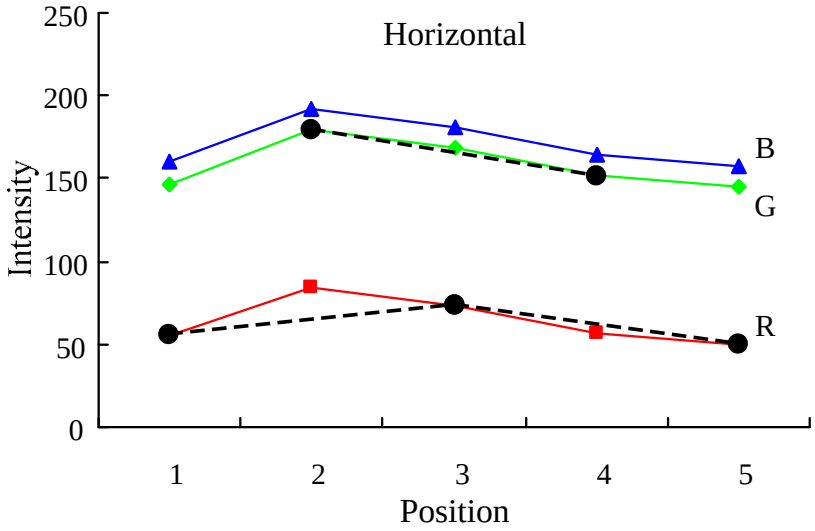
Table 3.1 The CPSNR performance (in dB) of various algorithms



(a)



(b)



(c)

Figure 3.2 An example where a simple gradient test does not work: (a) a 5×5 block in texture region, (b) pixel values along the vertical line across the block center and (c) pixel values along the horizontal line across the block center

When we probed into the gradient test used in [22], we found that the test encountered problems when dealing with pixels in texture regions. Figure 3.2 shows an example where the test does not work properly. A 5×5 block located in a texture region is extracted for inspection as shown in Figure 3.2a. Figures 3.2b and 3.2c show, respectively, the pixel values of the vertical and the horizontal lines across the block center. Suppose the black dots in the plots are the CFA samples while the others are the samples needed to be estimated. Consider that we are going to estimate the green component of the block center. As the black dash lines in vertical direction are flatter than that in horizontal direction, the test provides a misleading result. The algorithm interpolates vertically to determine the missing green component although it should interpolate horizontally. That is the reason why details cannot be preserved in a texture region with ACPI.

3.3 Proposed Color Difference Variance-Based Color Demosaicing Algorithm

Based on the observations presented in Section 3.2, the proposed algorithm put its focus on how to effectively determine the interpolation direction for estimating a missing green component in edge regions and texture regions. In particular, variance of color differences is used in the proposed algorithm as a supplementary criterion to determine the interpolation direction for the green components.

For the sake of reference, hereafter in this chapter, a pixel at location (i,j) in the CFA is represented by either $(R_{i,j}, g_{i,j}, b_{i,j})$, $(r_{i,j}, G_{i,j}, b_{i,j})$ or $(r_{i,j}, g_{i,j}, B_{i,j})$, where $R_{i,j}$, $G_{i,j}$ and $B_{i,j}$ denote the known red, green and blue components and $r_{i,j}$, $g_{i,j}$ and $b_{i,j}$ denote the unknown components in the CFA. The estimates of $r_{i,j}$, $g_{i,j}$ and $b_{i,j}$ are denoted as $\hat{R}_{i,j}$, $\hat{G}_{i,j}$ and $\hat{B}_{i,j}$. To get $\hat{R}_{i,j}$, $\hat{G}_{i,j}$ and $\hat{B}_{i,j}$, preliminary estimates of $r_{i,j}$, $g_{i,j}$ and $b_{i,j}$ may be required in the proposed demosaicing algorithm. These intermediate

estimates are denoted as $\hat{r}_{i,j}$, $\hat{g}_{i,j}$ and $\hat{b}_{i,j}$.

3.3.1 Interpolating Missing Green Components

In the proposed algorithm, the missing green components are first interpolated in a raster-scan manner. As far as a missing green component in the Bayer CFA is concerned, its neighborhood must be in a form as shown in either Figure 3.1a or 3.1b. Without losing the generality, let us consider the case shown in Figure 3.1a only. For the other case, the same treatment used in this case can be done to estimate the missing green component by exchanging the roles of the red components and the blue components.

In Figure 3.1a, the center pixel $p_{i,j}$ is represented by ($R_{i,j}$, $g_{i,j}$, $b_{i,j}$), where $g_{i,j}$ is the missing green component needed to be estimated. The proposed algorithm computes the two following parameters instead of $\Delta H_{i,j}$ and $\Delta V_{i,j}$, as in ACPI algorithm.

$$L^H = \sum_{n=\pm 2}\left[\sum_{m=0,\pm 2}\left|R_{i+m,j+n} - R_{i+m,j}\right| + \sum_{m=\pm 1}\left|G_{i+m,j+n} - G_{i+m,j}\right|\right] + \sum_{n=\pm 1}\left[\sum_{m=0,\pm 2}\left|G_{i+m,j+n} - R_{i+m,j}\right| + \sum_{m=\pm 1}\left|B_{i+m,j+n} - G_{i+m,j}\right|\right] \quad (3.6)$$

$$L^V = \sum_{m=\pm 2}\left[\sum_{n=0,\pm 2}\left|R_{i+m,j+n} - R_{i,j+n}\right| + \sum_{n=\pm 1}\left|G_{i+m,j+n} - G_{i,j+n}\right|\right] + \sum_{m=\pm 1}\left[\sum_{n=0,\pm 2}\left|G_{i+m,j+n} - R_{i,j+n}\right| + \sum_{n=\pm 1}\left|B_{i+m,j+n} - G_{i,j+n}\right|\right] \quad (3.7)$$

These two parameters are used to estimate whether there is sharp horizontal or vertical gradient change in the 5×5 testing window (with $p_{i,j}$ as the window center). A large value implies that there exists a sharp gradient change along a particular

direction. The ratio of the two parameters is then computed to determine the dominant edge direction.

$$e = \max\left(\frac{L^V}{L^H}, \frac{L^H}{L^V}\right) \tag{3.8}$$

A block is defined to be a sharp edge block if $e>T$, where T is a predefined threshold value to be discussed in more details in Section 3.4. If a block is a sharp edge block, the missing green component of the block center is interpolated as follows.

$$g_{i,j} = g^H_{i,j} = \frac{G_{i,j-1} + G_{i,j+1}}{2} + \frac{2R_{i,j} - R_{i,j-2} - R_{i,j+2}}{4} \quad \text{if } L^H < L^V \tag{3.9}$$

$$g_{i,j} = g^V_{i,j} = \frac{G_{i-1,j} + G_{i+1,j}}{2} + \frac{2R_{i,j} - R_{i-2,j} - R_{i+2,j}}{4} \quad \text{if } L^H > L^V \tag{3.10}$$

The block classifier based on eqns. (3.6)-(3.8) and threshold T is used to detect sharp edge blocks and determine the corresponding edge direction for interpolation. In eqns. (3.6) and (3.7), both inter-band and intra-band color information is used to evaluate parameters L^V and L^H. The first summation terms of eqns. (3.6) and (3.7) contribute the intra-band information, which involves the difference between a pixel and its second next pixel. Obviously, the resolution that it supports cannot detect a sharp line of one pixel width. The supplementary intra-band color information contributed by the second summation terms of eqns. (3.6) and (3.7) is used to improve the resolution of the edge detector.

A block which is not classified to be an edge block is considered to be in a flat region or a pattern region. It was found that, in a local region of a natural image, the

color differences of pixels are more or less the same [64]. Accordingly, the variance of color differences can be used as supplementary information to determine the interpolation direction for the green components.

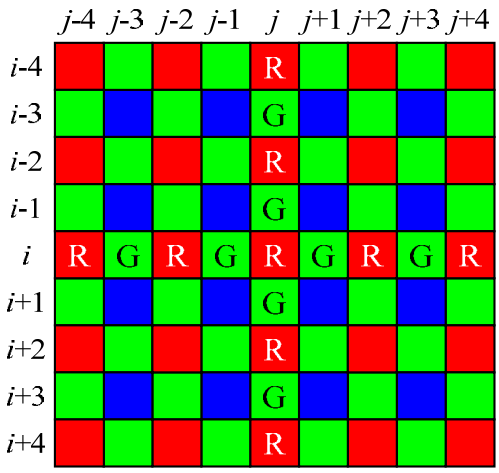


Figure 3.3 9×9 window of Bayer CFA pattern

In the proposed algorithm, we extend the 5×5 block of interest into a 9×9 block by including more neighbors as shown in Figure 3.3 and evaluate the color differences of the pixels along the axis within the 9×9 window. Let ${}_H\sigma_{i,j}^2$ and ${}_V\sigma_{i,j}^2$ be, respectively, the variances of the color differences of the pixels along the horizontal axis and the vertical axis of the 9×9 block. In particular, they are defined as

$$ {}_H\sigma_{i,j}^2 = \frac{1}{9}\sum_{n\in\Psi}\left(d_{i,j+n} - \frac{1}{9}\sum_{k\in\Psi} d_{i,j+k}\right)^2 \tag{3.11}$$

and

$$ {}_V\sigma_{i,j}^2 = \frac{1}{9}\sum_{n\in\Psi}\left(d_{i+n,j} - \frac{1}{9}\sum_{k\in\Psi} d_{i+k,j}\right)^2 \tag{3.12}$$

where $d_{p,q}$ is the color difference of pixel (p,q) and $\Psi=\{0,\pm1,\pm2,\pm3,\pm4\}$ is a set of

indexes which helps to identify a pixel in the 9×9 support region. The values of $d_{i,j+n}$ and $d_{i+n,j}$ for $n \in \Psi$ should be pre-computed and they are determined sequentially as follows.

$$d_{i,j+n} = \begin{cases} R_{i,j+n} - \hat{G}_{i,j+n} & \text{if } n = -2,-4 \\ R_{i,j+n} - \hat{g}_{i,j+n} & \text{if } n = 0,2,4 \end{cases} \tag{3.13}$$

$$d_{i+n,j} = \begin{cases} R_{i+n,j} - \hat{G}_{i+n,j} & \text{if } n = -2,-4 \\ R_{i+n,j} - \hat{g}_{i+n,j} & \text{if } n = 0,2,4 \end{cases} \tag{3.14}$$

and

$$d_{i,j+n} = \frac{d_{i,j+n-1} + d_{i,j+n+1}}{2} \qquad \text{for } n = \pm 1,\pm 3 \tag{3.15}$$

$$d_{i+n,j} = \frac{d_{i+n-1,j} + d_{i+n+1,j}}{2} \qquad \text{for } n = \pm 1,\pm 3 \tag{3.16}$$

Here, bilinear interpolation is used with the concern about the realization complexity to estimate $d_{i,j+n}$ and $d_{i+n,j}$ for n=±1,±3. In fact, there was no obvious improvement in the simulation results when other interpolation schemes such as cubic interpolation were used.

To provide some more information about eqns. (3.13) and (3.14), we note that the missing green components are estimated in a raster-scan fashion and hence the final estimates of the green components in position $\Omega_{i,j}$={(i,j+n),(i+n,j) | n=-2,-4} are already computed. As for the missing green components of the pixels in position {(i,j+n),(i+n,j) | n=0,2,4}, their preliminary estimates $\hat{g}_{i,j+n}$ and $\hat{g}_{i+n,j}$ have to be evaluated. Specifically, $\hat{g}_{i,j+n}$ and $\hat{g}_{i+n,j}$ are, respectively, estimated to be $g^{H}_{i,j+n}$ and $g^{V}_{i+n,j}$ unconditionally. Note the $d_{i,j}$ involved in eqn. (3.11) uses the $\hat{g}_{i,j}$ determined with eqn. (3.3) while the $d_{i,j}$ involved in eqn. (3.12) uses the $\hat{g}_{i,j}$

determined with eqn. (3.4).

The variance of the color differences of the diagonal pixels in the 9×9 window, say ${}_D\sigma_{i,j}^2$, are determined by

$$ {}_D\sigma_{i,j}^2 = \frac{1}{2}\left(\frac{1}{9}\sum_{n\in\Psi}\left(d_{i+n,j} - \frac{1}{9}\sum_{k\in\Psi} d_{i+k,j} \right)^2 + \frac{1}{9}\sum_{n\in\Psi}\left(d_{i,j+n} - \frac{1}{9}\sum_{k\in\Psi} d_{i,j+k} \right)^2 \right). \qquad (3.17) $$

The same set of eqns. (3.13)-(3.16) are used to get the color difference $d_{p,q}$ required in the evaluation of ${}_D\sigma_{i,j}^2$. However, the preliminary estimates $\hat{g}_{i,j+n}$ and $\hat{g}_{i+n,j}$ involved in these equations are determined with eqn. (3.5) instead of eqns. (3.3) and (3.4).

Finally, the interpolation direction for estimating the missing green component at $p_{i,j}=(R_{i,j},\ g_{i,j},\ b_{i,j})$ can be determined based on ${}_H\sigma_{i,j}^2$, ${}_V\sigma_{i,j}^2$ and ${}_D\sigma_{i,j}^2$. It is the direction providing the minimum variance of color difference. The missing green $g_{i,j}$ can then be estimated with either formulation (3.3), (3.4) or (3.5) without concerning $\Delta H_{i,j}$ and $\Delta V_{i,j}$ as follows.

$$ \hat{G}_{i,j} = \begin{cases} \text{unconditional result of eqn. (3.3)} & \text{if } {}_H\sigma_{i,j}^2 = \min({}_H\sigma_{i,j}^2,\ {}_V\sigma_{i,j}^2,\ {}_D\sigma_{i,j}^2) \\ \text{unconditional result of eqn. (3.4)} & \text{if } {}_V\sigma_{i,j}^2 = \min({}_H\sigma_{i,j}^2,\ {}_V\sigma_{i,j}^2,\ {}_D\sigma_{i,j}^2) \\ \text{unconditional result of eqn. (3.5)} & \text{if } {}_D\sigma_{i,j}^2 = \min({}_H\sigma_{i,j}^2,\ {}_V\sigma_{i,j}^2,\ {}_D\sigma_{i,j}^2) \end{cases} \qquad (3.18) $$

Once the missing green component is interpolated, the same process is performed for estimating the next missing green component in a raster-scan manner. For estimating the missing green component in the case shown in Figure 3.1b, one can

replace the red samples by the corresponding blue samples and follow the procedures above to determine its interpolation direction and its interpolated value.

The complexity of the realization of eqn. (3.18) and its preparation work (eqns. (3.11)-(3.17)) is large. When $e>T$, a sharp edge block is clearly identified. In that case, using eqns. (3.9) and (3.10) to interpolate missing components can save a lot of effort and, at the same time, provide a good demosaicing result.

3.3.2 Interpolating Missing Red and Blue Components at Green Sampling Positions

After interpolating all missing green components of the image, the missing red and blue components at green CFA sampling positions are estimated. Figures 3.1c and 3.1d show the two possible cases where a green CFA sample is located at the center of a 5×5 block. For the case shown in Figure 3.1c, the missing components of the center are obtained by

$$\hat{R}_{i,j} = G_{i,j} + \frac{R_{i,j-1} - \hat{G}_{i,j-1} + R_{i,j+1} - \hat{G}_{i,j+1}}{2} \tag{3.19}$$

and

$$\hat{B}_{i,j} = G_{i,j} + \frac{B_{i-1,j} - \hat{G}_{i-1,j} + B_{i+1,j} - \hat{G}_{i+1,j}}{2}. \tag{3.20}$$

As for the case shown in Figure 3.1d, the missing components of the center are obtained by

$$\hat{R}_{i,j} = G_{i,j} + \frac{R_{i-1,j} - \hat{G}_{i-1,j} + R_{i+1,j} - \hat{G}_{i+1,j}}{2} \tag{3.21}$$

and

$$\hat{B}_{i,j} = G_{i,j} + \frac{B_{i,j-1} - \hat{G}_{i,j-1} + B_{i,j+1} - \hat{G}_{i,j+1}}{2}. \tag{3.22}$$

3.3.3 Interpolating Missing Blue (Red) Components at Red (Blue) Sampling Positions

Finally, the missing blue (red) components at the red (blue) sampling positions are interpolated. Figure. 3.1a and 3.1b show the two possible cases where the pixel of interest lies in the center of a 5×5 window. For the case shown in Figure 3.1a, the missing blue sample at the center, $b_{i,j}$, is interpolated by

$$\hat{B}_{i,j} = \hat{G}_{i,j} + \frac{1}{4}\sum_{m=\pm 1}\sum_{n=\pm 1}(B_{i+m,j+n} - \hat{G}_{i+m,j+n}). \tag{3.23}$$

As for the case shown in Figure 3.1b, the missing red sample at the center, $r_{i,j}$, can also be obtained with eqn. (3.23) by replacing the blue estimates with the corresponding red estimates in eqn. (3.23). At last, the final full-color image can be obtained.

3.3.4 Refinement

Refinement schemes are usually exploited to further improve the interpolation performance of various demosaicing algorithms [27-32,35,36]. In fact, there are even some post-processing algorithms proposed as stand-alone solutions to improve the quality of a demosaicing result [51,65]. In the proposed algorithm, we use the refinement scheme suggested in the enhanced ECI algorithm [30] as we found that it matched the proposed algorithm to provide the best demosaicing result after evaluating some other refinement schemes with the proposed algorithm. This refinement scheme processes the interpolated green samples $\hat{G}_{i,j}$ first to reinforce the interpolation performance and, based on the refined green plane, it performs a

refinement on the interpolated red and blue samples. In particular, each of the estimated samples is refined as a weighted-sum of its neighboring samples with the reciprocal of local gradients as the weights. One can see [30] for more details on the refinement scheme.

3.4 Performance Evaluation

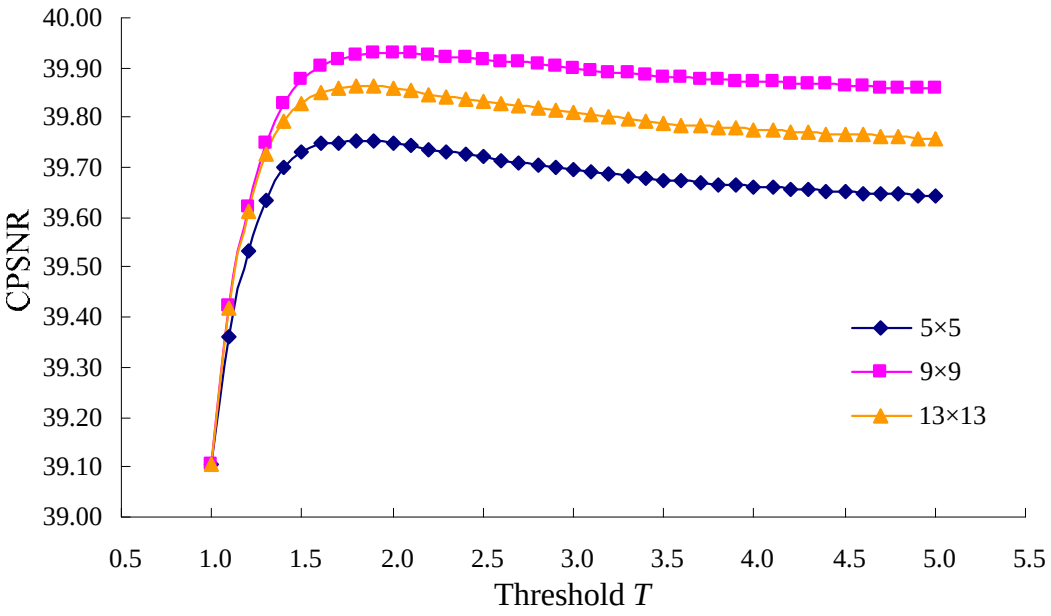
Simulation was carried out to evaluate the performance of the proposed algorithm. The 24 digital color images shown in Appendix A were used to generate a set of testing images as mentioned in Section 3.2. The color-peak signal-to-noise ratio (CPSNR) was used as a measure to quantify the performance of the evaluated demosaicing algorithms, which is defined as

$$CPSNR = 10\log_{10}\left(\frac{255^2}{CMSE}\right) \tag{3.24}$$

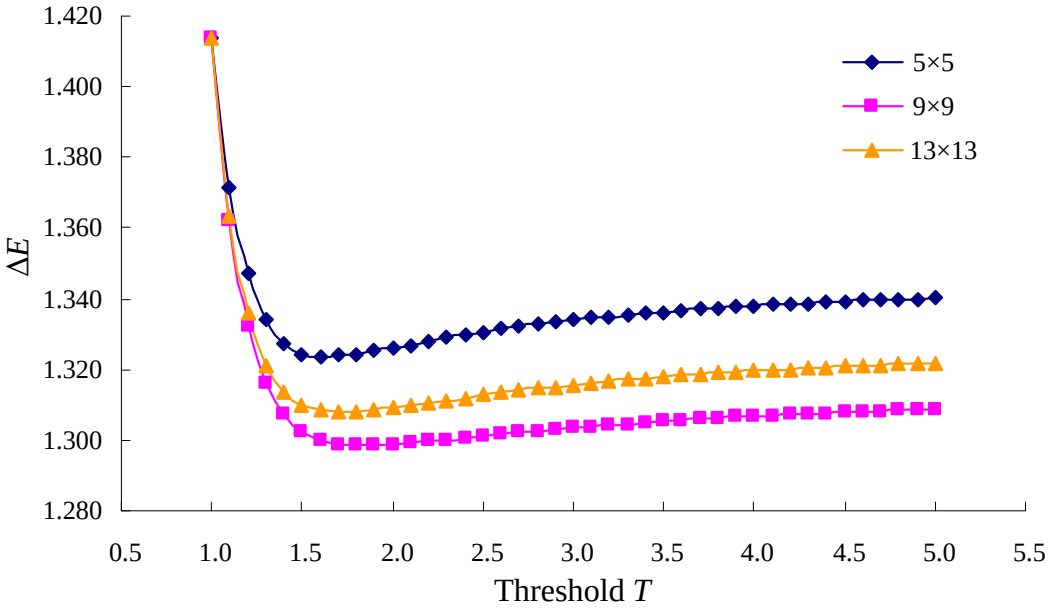
where $CMSE = \frac{1}{3HW}\sum_{k=r,g,b}\sum_{y=1}^{H}\sum_{x=1}^{W}(I_o(x,y,k) - I_r(x,y,k))^2$. I_o and I_r represent, respectively, the original and the reconstructed images of size $H \times W$ each.

In the proposed algorithm, a pixel is interpolated according to the nature of its local region. The 5×5 region centered at the pixel is first classified to be either a sharp edge block or not with threshold T. A non-edge block is then extended from 5×5 to 9×9 to compute ${}_H\sigma_{i,j}^2$, ${}_V\sigma_{i,j}^2$ and ${}_D\sigma_{i,j}^2$ for further classification. An empirical study was carried out to select an appropriate threshold value of T and check if 9×9 is an appropriate size of the extended local region for estimating ${}_H\sigma_{i,j}^2$, ${}_V\sigma_{i,j}^2$ and ${}_D\sigma_{i,j}^2$ in the realization of the proposed algorithm. Figure 3.4 shows the performance at different settings. It shows that T=2 and an extended region of size 9×9 can provide

a performance close to the optimal. Hereafter, all simulation results of the proposed algorithm presented in this chapter were obtained with this setting.



(a) CPSNR



(b) S-CIELab color difference

Figure 3.4 Performance of the proposed algorithm at different settings of threshold T and window size

Ten existing demosaicing algorithms, including BI [23], AP [12], DUOR [42,63], DSA [39], ACPI [22], ECI [26], PCSD [36], EECI [30], AHDA [35] and DAFD [28], were implemented for comparison. The proposed demosaicing algorithm is referred to VCD hereafter in this thesis for clear reference. In the realization of DUOR, the correction step described in [63] was applied to the demosaicing result of [42]. Table 3.1 tabulates the CPSNR performance of different demosaicing algorithms. It reveals that the proposed algorithm produces the best average performance among the tested algorithms. In additional, it was found that the proposed algorithm could handle fine texture patterns well. For images which contain many fine texture patterns such as Image 6, 8, 16 and 19, the proposed algorithm obviously outperforms the other demosaicing solutions. For example, as far as Image 8 is concerned, the CPSNR achieved by the proposed algorithm is 1.21dB higher than that of the CPSNR achieved by EECI, which is the second best among all evaluated algorithms in a way that it achieved the second best average CPSNR performance in the simulation.

Figures 3.5, 3.6 and 3.7 show some demosaicing results of Image 1, 15 and 19 for comparison. They show that the proposed algorithm can preserve fine texture patterns and, accordingly, produce less color artifacts. Recall that the proposed algorithm is actually developed based on ACPI. As compared with ACPI, the proposed algorithm produces a demosaicing result of much less color artifact by interpolating missing components along a better direction. In fact, the average CPSNR of the proposed algorithm (39.93dB) is much closer than that of ACPI (36.88dB) to that achieved by the ideal ACPI (41.02dB). These results show that the gradient test proposed in the proposed algorithm is more reliable than that of ACPI.

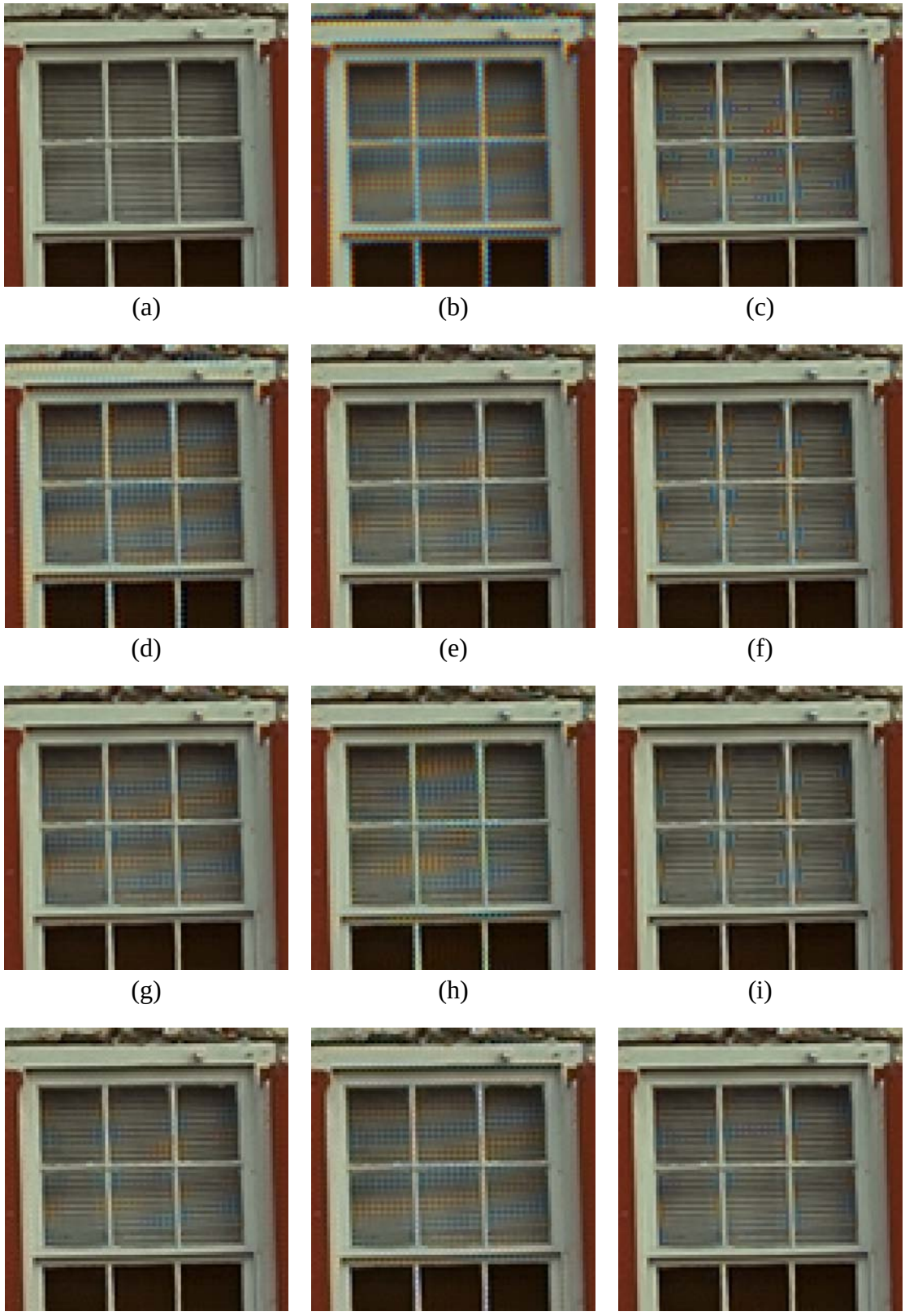


Figure 3.5 Part of the demosaicing results of Image 1: (a) the original, (b) BI, (c) ACPI, (d) ECI, (e) AP, (f) PCSD, (g) EECI, (h) DUOR, (i) AHDA, (j) DSA, (k) DAFD and (l) the proposed algorithm

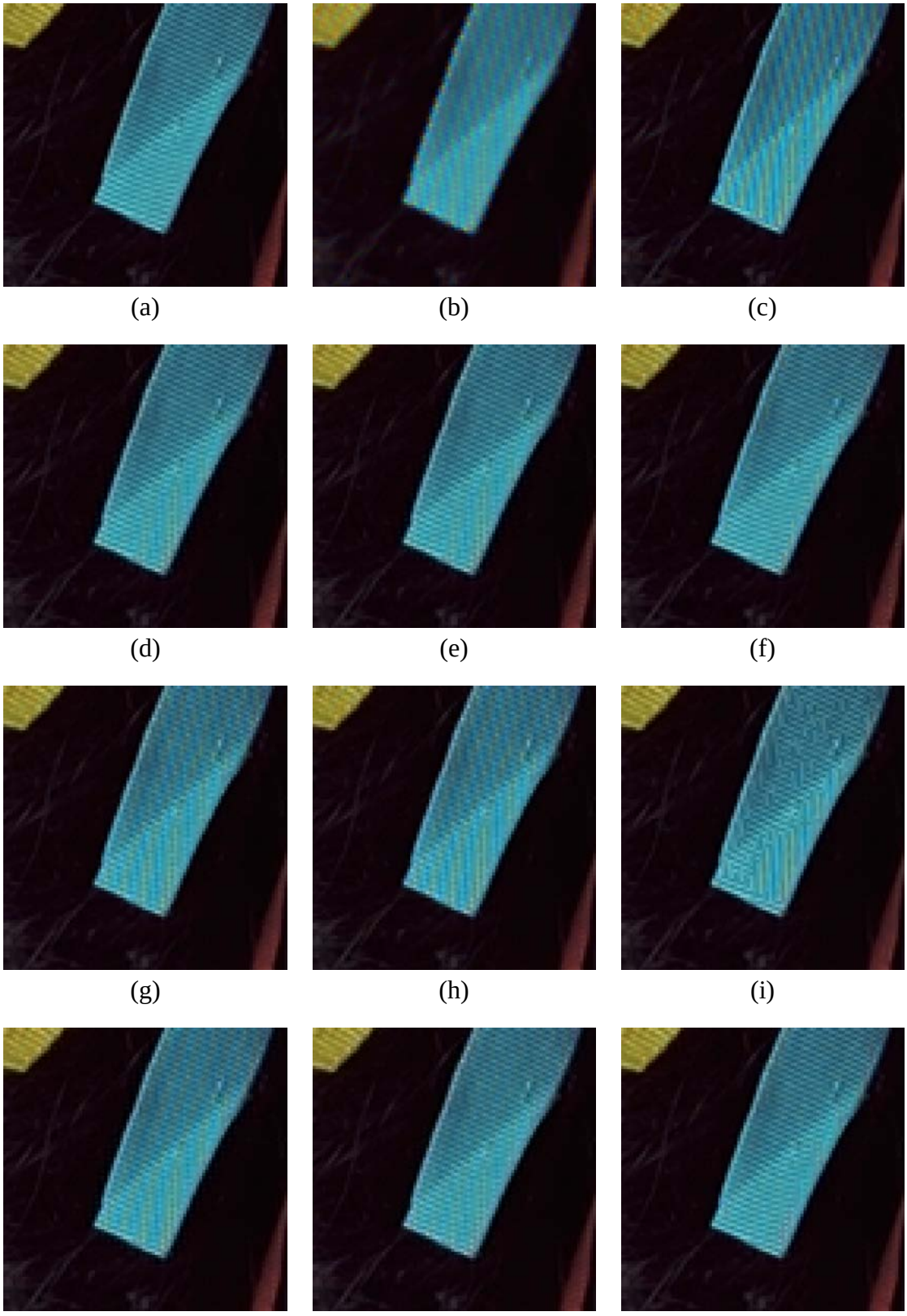


Figure 3.6 Part of the demosaicing results of Image 15: (a) the original, (b) BI, (c) ACPI, (d) ECI, (e) AP, (f) PCSD, (g) EECI, (h) DUOR, (i) AHDA, (j) DSA, (k) DAFD and (l) the proposed algorithm

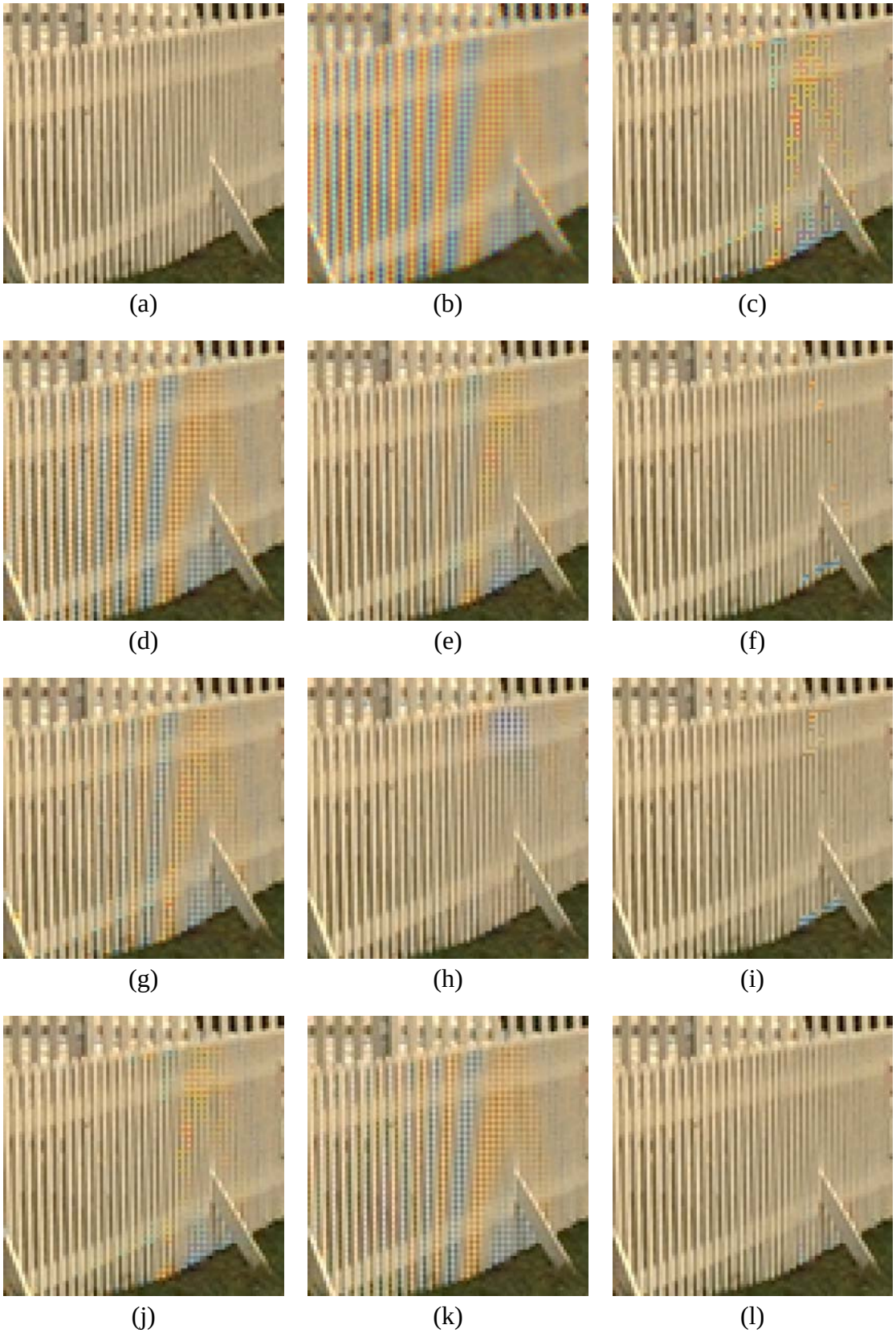


Figure 3.7 Part of the demosaicing results of Image 19: (a) the original, (b) BI, (c) ACPI, (d) ECI, (e) AP, (f) PCSD, (g) EECI, (h) DUOR, (i) AHDA, (j) DSA, (k) DAFD and (l) the proposed algorithm

Image	Non-Heuristic Algorithms			Heuristic Algorithms									
	AP [12]	**DUOR [42,63]**	**DSA [39]**	**BI [23]**	**ACPI [22]**	**ECI [26]**	**PCSD [36]**	**DAFD [28]**	**EECI [30]**	**AHDA [35]**	**VCD w/o Refine.**	**VCD w/ Refine.**	**Ideal ACPI**
1	1.646	2.102	1.570	5.258	2.380	2.359	1.742	1.989	1.506	1.619	1.862	1.463	1.357
2	1.664	1.733	1.609	3.179	1.795	1.829	1.595	1.603	1.488	1.600	1.640	1.499	1.250
3	0.951	0.998	0.995	1.808	1.053	1.018	0.933	1.033	0.861	0.932	0.954	0.880	0.763
4	1.286	1.535	1.354	2.568	1.562	1.381	1.337	1.357	1.210	1.322	1.355	1.216	1.038
5	2.167	2.690	2.426	5.866	2.788	2.725	2.134	2.594	1.943	2.210	2.435	2.031	1.837
6	1.245	1.260	1.203	3.752	1.693	1.720	1.143	1.471	1.214	1.136	1.284	1.063	0.987
7	1.106	1.123	1.149	2.218	1.151	1.246	1.087	1.236	0.974	1.128	1.131	1.033	0.879
8	1.867	2.527	1.737	6.016	2.398	2.845	1.819	2.280	1.690	1.734	1.959	1.587	1.456
9	0.841	0.870	0.833	1.903	0.945	0.984	0.787	0.987	0.773	0.818	0.819	0.742	0.643
10	0.826	0.927	0.835	1.860	0.971	0.924	0.813	0.960	0.766	0.842	0.858	0.773	0.671
11	1.474	1.623	1.473	3.932	1.918	1.976	1.466	1.646	1.341	1.428	1.596	1.330	1.171
12	0.677	0.691	0.688	1.546	0.800	0.811	0.653	0.762	0.640	0.659	0.693	0.614	0.548
13	2.582	3.403	2.503	7.310	4.054	3.338	2.980	2.781	2.438	2.750	3.128	2.445	2.196
14	1.944	2.043	2.135	4.199	2.210	2.169	1.925	1.996	1.717	1.875	1.969	1.748	1.453
15	1.429	1.597	1.493	2.602	1.653	1.484	1.462	1.492	1.323	1.464	1.477	1.337	1.112
16	1.030	0.977	1.018	2.928	1.350	1.439	0.925	1.193	1.035	0.918	1.032	0.871	0.818
17	1.329	1.513	1.341	2.863	1.674	1.529	1.399	1.503	1.279	1.395	1.474	1.308	1.121
18	2.184	2.587	2.352	4.936	2.752	2.372	2.364	2.428	2.070	2.395	2.409	2.132	1.730
19	1.288	1.495	1.247	3.646	1.625	1.734	1.293	1.488	1.186	1.272	1.337	1.144	0.993
20	1.003	1.135	1.029	2.204	1.244	1.207	1.051	1.084	0.928	1.032	1.102	0.963	0.856
21	1.328	1.561	1.311	3.544	1.865	1.726	1.447	1.485	1.241	1.367	1.538	1.262	1.129
22	1.497	1.712	1.538	2.998	1.737	1.613	1.559	1.571	1.395	1.593	1.591	1.450	1.155
23	0.950	1.018	0.990	1.513	0.998	0.968	0.965	0.996	0.893	0.968	0.961	0.918	0.761
24	1.438	1.766	1.483	3.563	1.834	1.693	1.470	1.596	1.335	1.486	1.567	1.363	1.167
Avg.	**1.406**	**1.620**	**1.430**	**3.425**	**1.769**	**1.712**	**1.431**	**1.564**	**1.302**	**1.414**	**1.507**	**1.299**	**1.129**

Table 3.2 Performance of various algorithms in terms of S-CIELab color difference

Table 3.2 shows the performance of various algorithms in terms of S-CIELab color difference (ΔE) [66]. The proposed algorithm provides the best performance among the evaluated algorithms again. With the proposed gradient testing tool, a simple heuristic algorithm can provide a subjectively and objectively better demosaicing performance as compared with many state-of-art algorithms [12,22,26,30,35,36,39,42].

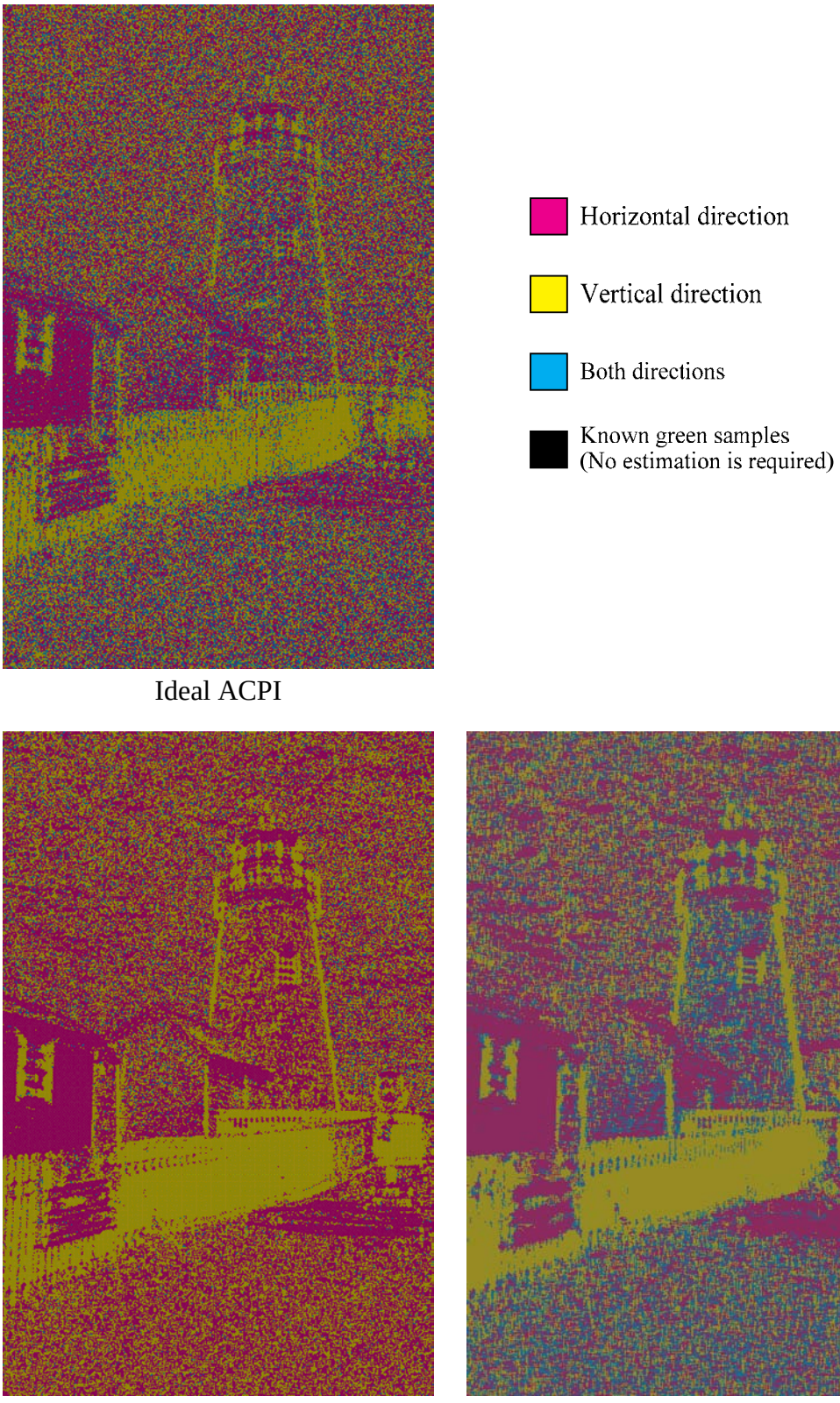


Figure 3.8 Direction maps obtained with different algorithms for interpolating missing green samples

Pixel nature of the group	**Group 1** $D_{ACPI} = D_{iACPI} \neq D_{ours}$		**Group 2** $D_{ACPI} \neq D_{iACPI} = D_{ours}$		**Group 3** $D_{ACPI} \neq D_{iACPI} \neq D_{ours}$	
% of pixels	17.12%		19.07%		34.84%	
	ACPI	Ours w/o Refinement	ACPI	Ours w/o Refinement	ACPI	Ours w/o Refinement
MSE/pixel	8.537	28.779	64.341	7.231	31.766	22.670

Table 3.3 MSE contributed by different groups of pixels

The proposed algorithm is developed based on the fact that the interpolation direction for each missing green sample is critical to the final demosaicing result. A study was carried out to investigate how significant the improvement with respect to ACPI could be when the proposed algorithm was used to find a better interpolation direction for a pixel. Note that demosaicing along edges in a natural image significantly reduces the demosaicing error. Figure 3.8 shows some interpolation direction maps obtained with ACPI, the ideal ACPI and the proposed algorithm for comparison.

Table 3.3 shows the performance of ACPI and the proposed algorithm in finding an appropriate interpolation direction. This Table shows the contribution of 3 different groups of pixels to the minimum square error (MSE) of the green plane in the final demosaicing result. Pixels in the testing images are grouped according to the following three constraints: (1) $D_{ACPI} = D_{iACPI} \neq D_{ours}$, (2) $D_{ACPI} \neq D_{iACPI} = D_{ours}$ and (3) $D_{ACPI} \neq D_{iACPI} \neq D_{ours}$, where D_{ACPI}, D_{iACPI} and D_{ours} are, respectively, a pixel's interpolation directions estimated with ACPI, the ideal ACPI and the proposed algorithm. One can see that the percentage of Group 2 pixels is higher than that of Group 1 pixels. This implies that, when the proposed algorithm is used, there are more hits on D_{iACPI}. Even in the case of $D_{iACPI} \neq D_{ours}$, the estimate of the proposed algorithm is better in a way that the interpolation result provides a lower MSE. One can see the reduction in MSE/pixel achieved by the proposed algorithm in Group 3

pixels. Besides, the average penalty introduced by Group 1 pixels to the proposed algorithm is just 20.3 (=28.8-8.5) while the average penalty introduced by Group 2 pixels to ACPI is 57.1 (=64.3-7.2) in terms of MSE per pixel.

		Original version				Simplified version			
		ADD	**MUL**	**SHT**	**ABS**	**ADD**	**MUL**	**SHT**	**ABS**
Edge/non-edge block classification		26	1	0	8	26	1	0	8
Estimating missing green sample									
	Case: In edge block	9	0	5	0	9	0	5	0
	Case: In non-edge block	97	39	11	0	61	4	7	19
Estimating missing red/blue sample		4	0	1	0	4	0	1	0
Refinement		19	14	0	0	19	14	0	0
Total									
	Case: In edge block	**58**	**15**	**6**	**8**	**58**	**15**	**6**	**8**
	Case: In non-edge block	**146**	**54**	**12**	**8**	**110**	**19**	**8**	**27**

(a)

	Original version				Simplified version			
	ADD	**MUL**	**SHT**	**ABS**	**ADD**	**MUL**	**SHT**	**ABS**
Estimating missing red sample	4	0	1	0	4	0	1	0
Estimating missing blue sample	2	0	1	0	2	0	1	0
Refinement	34	18	0	0	34	18	0	0
Total	**38**	**18**	**2**	**0**	**38**	**18**	**2**	**0**

(b)

Table 3.4 Arithmetic operations required by the proposed algorithm to estimate two missing color components at (a) a red/blue sampling position or (b) a green sampling position

Table 3.4 shows the complexity of the proposed algorithm in terms of number of additions (ADD), multiplications (MUL), bit-shifts (SHT) and absolute-value-taking operations (ABS). A comparison operation is considered as an addition in this thesis for easier comparison. Note that some intermediate computation results can be reused during demosaicing and this was taken into account when the complexity of the proposed algorithm was estimated. Its complexity can be reduced by simplifying the

estimation of ${}_H\sigma_{i,j}^2$, ${}_V\sigma_{i,j}^2$ and ${}_D\sigma_{i,j}^2$. In particular, eqns. (3.11), (3.12) and (3.17) can be simplified by replacing Ψ with $\Psi_s=\{0,\pm2,\pm4\}$ and turning all involved square operations into absolute value operations. Some demosaicing performance is sacrificed due to the simplification. The simplified version provided an average CPSNR of 39.89dB and an average S-CIELab color difference of 1.305 in our simulations. Its complexity is also shown in Table 3.4. In our simulations, the average execution times for the proposed algorithm and its simplified version to process an image on a 2.8GHz Pentium 4 PC with 512MB RAM are, respectively, 0.2091s and 0.1945s.

3.5 Chapter Summary

In this chapter, an adaptive demosaicing algorithm was presented. It makes use of the color difference variance of the pixels located along the horizontal axis and that along the vertical axis in a local region to estimate the interpolation direction for interpolating the missing green samples. With such an arrangement, the interpolation direction can be estimated more accurately and, hence, more fine texture pattern details can be preserved in the output. Simulation results show that the proposed algorithm is able to produce a subjectively and objectively better demosaicing result as compared with a number of advanced algorithms.

Chapter 4 A Low-Complexity Joint Color Demosaicing and Zooming Algorithm for Digital Camera

4.1 Introduction

As mentioned in Chapter 2, there are generally two conventional approaches, namely zooming-after-demosaicing and demosaicing-after-zooming, to produce a zoomed full-color image. However, no matter which approach we take, the zooming and demosaicing processes are carried out independently. Since the actual process behind both zooming and demosaicing is interpolation and similar signal processing concepts are employed in both cases, when the zooming and demosaicing steps are performed separately, the information available on the raw sensor data is not always utilized consistently and efficiently to produce the enlarged output. In this chapter, a low-complexity joint color demosaicing and digital zooming algorithm is proposed for a digital single-sensor camera to solve this problem. Figure 4.1 shows the concept of the proposed joint demosaicing and zooming approach.

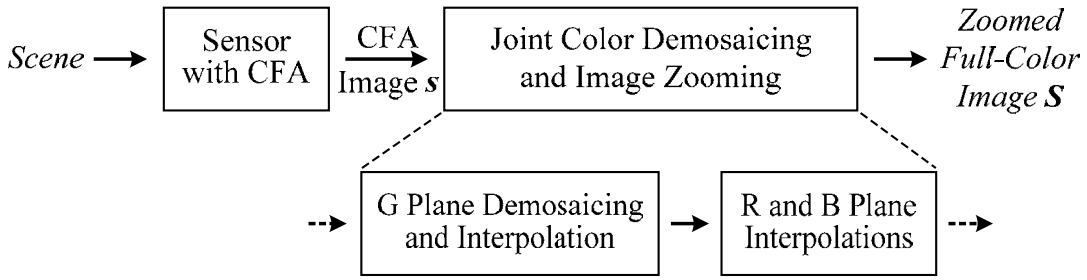


Figure 4.1 Workflow of the proposed joint demosaicing and zooming algorithm

By considering that the green channel provides twice information as compared with the red and the blue channels in a Bayer CFA image [54], the proposed algorithm starts with estimating all missing green samples in the original CFA image. For each missing green sample, it estimates its appropriate interpolation direction based on local intensity gradients and local variances of color difference values, and picks a corresponding linear interpolator to estimate its intensity value. The green plane is then enlarged in a way that all missing green components in the enlarged image are estimated based on (i) the green intensity values of their known or determined neighbors and (ii) the interpolation directions of the closest neighbors whose green components are determined in the previous stage. By so doing, the edge information extracted from the raw sensor data for interpolation is used in demosaicing and zooming consistently. It is also used efficiently as no separate extraction process is required in different stages. Finally, the red and the blue missing samples in the enlarged image are estimated with the interpolated green plane and the color difference model used in [26].

Simulation results show that the proposed algorithm is superior to conventional approaches, which are generally combinations of different demosaicing and zooming algorithms, in zooming CFA images and producing zoomed full-color images in terms of output quality at low complexity.

This chapter is structured as follows. In Section 4.2, a color difference variance-based green plane demosaicing scheme is introduced. In Section 4.3, the details of our joint demosaicing and zooming algorithm is described. In Sections 4.4 and 4.5, simulation results and a complexity analysis of the proposed algorithm are respectively provided. Finally, a conclusion is given in Section 4.6.

4.2 Color Difference Variance-Based Green Plane Demosaicing Scheme

The proposed green plane demosaicing scheme processes the pixels one by one in a raster-scanning order. Based on a symmetric local region of the pixel of interest, the scheme determines the interpolation direction and then interpolates the missing green component of the concerned pixel.

In the proposed scheme, the well-known directional second-order Laplacian interpolators proposed by Hamilton and Adams [22,64] are used to interpolate the missing green component as they were proven to be a simple yet good approximation of the optimal CFA interpolator [35]. In general, in a CFA image, the local region of a pixel without green component is in a pattern as shown in either Figure 4.2a or 4.2b. Without losing generality, only the former case is considered in this chapter of the thesis. For handling the case shown in Figure 4.2b, one can exchange the roles of red samples and blue samples and then follow the procedures for handling the case shown in Figure 4.2a to estimate the missing green components.

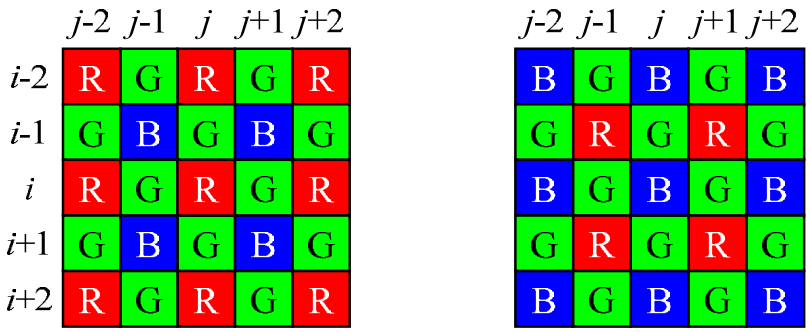


Figure 4.2 Two 5×5 regions of Bayer pattern having centers at (a) red and (b) blue samples

For the case shown in Figure 4.2a, the missing green component of the center pixel (*i*,*j*) is estimated with one of the following directional interpolators.

$$\text{Horizontal}\,(H): \quad g_{i,j}^{H} = \frac{G_{i,j-1} + G_{i,j+1}}{2} + \frac{2R_{i,j} - R_{i,j-2} - R_{i,j+2}}{4} \tag{4.1}$$

$$\text{Vertical}\,(V): \quad g_{i,j}^{V} = \frac{G_{i-1,j} + G_{i+1,j}}{2} + \frac{2R_{i,j} - R_{i-2,j} - R_{i+2,j}}{4} \tag{4.2}$$

$$\text{Diagonal}\,(D): \quad g_{i,j}^{D} = \frac{g_{i,j}^{H} + g_{i,j}^{V}}{2} \tag{4.3}$$

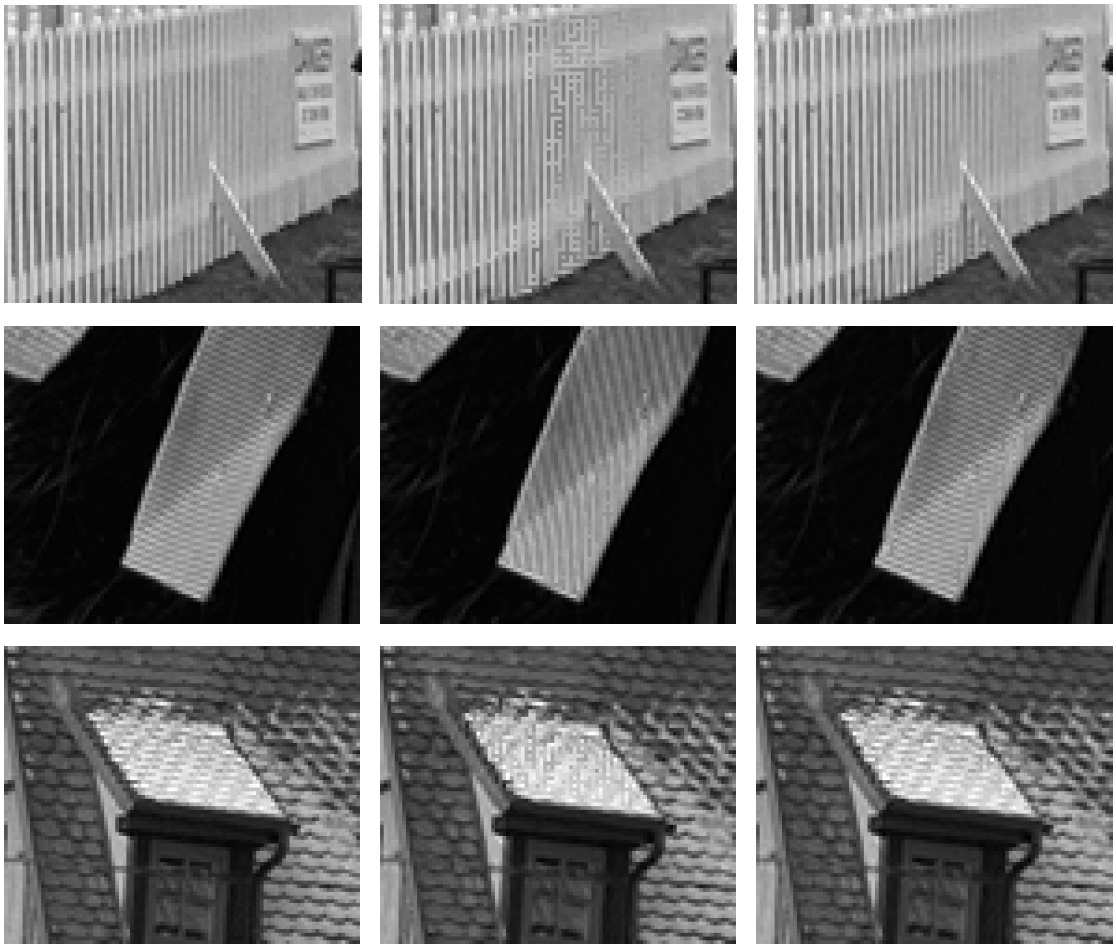
where $R_{m,n}$ and $G_{m,n}$ respectively denote the known red and green CFA components of pixel (*m*,*n*), and $g_{i,j}^{H}$, $g_{i,j}^{V}$ and $g_{i,j}^{D}$ are the three possible estimates of the missing green component obtained with corresponding interpolators.

The selection of the interpolator is a critical factor to the demosaicing performance. Many adaptive demosaicing algorithms use local gradients to determine the interpolation direction and in turn the interpolator [22,64,67]. However, though this determination criterion works very well in simple edge regions, some image pattern features such as those shown in Figure 4.3b may not be preserved well with this criterion. As the color difference of a pixel (either green-to-red or green-to-blue) usually varies smoothly over a local region in a typical image, the local variance of its value is used as supplementary information to determine the interpolation direction in the proposed scheme when a texture region is encountered.

To reach a better decision, the proposed scheme is a two-pass estimation scheme. In the first pass, the scheme raster-scans the CFA image and detects if a particular pixel is in a sharp horizontal or vertical edge region. If it is, interpolation direction will be determined and the missing green component of the pixel will be interpolated accordingly. Otherwise, the pixel will be left behind and processed in the second pass.

In the second pass, the color difference information in a local region is used to determine the interpolation direction. Note that, in Pass 2, the green components estimated in Pass 1 can be used with those known green components to pick appropriate interpolators and perform the interpolation, which helps to improve the interpolation result.

Figure 4.4 summarizes how to select an interpolator for a particular pixel in different passes of the proposed scheme. The details are as follows.



(a) (b) (c)

Figure 4.3 (a) Original green planes of the original full-color images, (b) green planes generated by algorithm [22] and (c) green planes generated by the proposed green plane demosaicing scheme (The input for generating (b) and (c) were obtained by sampling (a) according to the Bayer CFA pattern.)

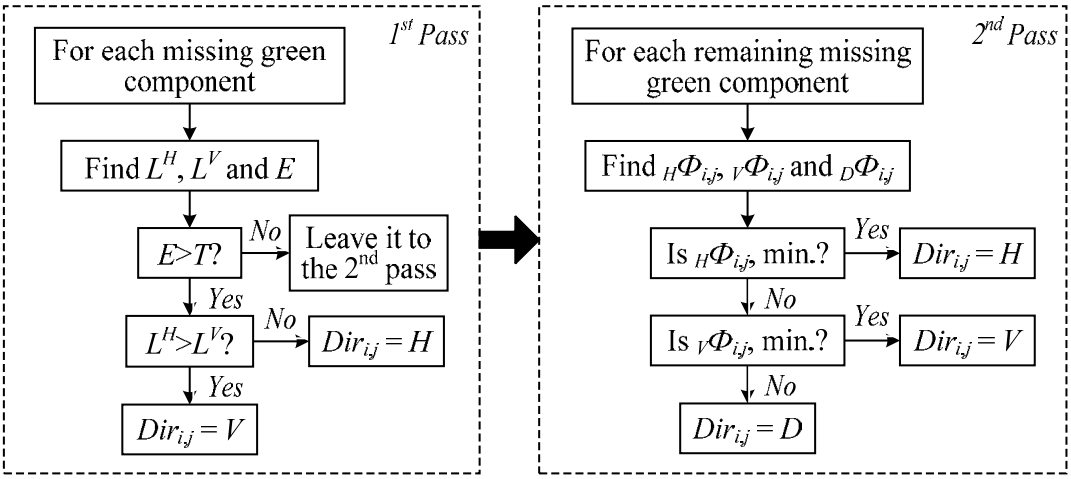


Figure 4.4 Procedures for determining the direction to interpolate a missing green component in the proposed green plane demosaicing scheme

Pass 1

Both local intensity gradient and color difference are exploited in this pass to evaluate two parameters for a 5×5 local region to see whether there is sharp horizontal or vertical gradient change within the region. For the case shown in Figure 4.2a, the two parameters are computed as

$$L^H = \sum_{n=\pm2}\left[\sum_{m=0,\pm2}\left|R_{i+m,j+n} - R_{i+m,j}\right| + \sum_{m=\pm1}\left|G_{i+m,j+n} - G_{i+m,j}\right|\right] + \sum_{n=\pm1}\left[\sum_{m=0,\pm2}\left|G_{i+m,j+n} - R_{i+m,j}\right| + \sum_{m=\pm1}\left|B_{i+m,j+n} - G_{i+m,j}\right|\right] \quad (4.4)$$

and

$$L^V = \sum_{m=\pm2}\left[\sum_{n=0,\pm2}\left|R_{i+m,j+n} - R_{i,j+n}\right| + \sum_{n=\pm1}\left|G_{i+m,j+n} - G_{i,j+n}\right|\right] + \sum_{m=\pm1}\left[\sum_{n=0,\pm2}\left|G_{i+m,j+n} - R_{i,j+n}\right| + \sum_{n=\pm1}\left|B_{i+m,j+n} - G_{i,j+n}\right|\right]. \quad (4.5)$$

The 5×5 window is selected because it matches the support of interpolators (4.1), (4.2) and (4.3). Note that, in both eqns. (4.4) and (4.5), the first summation term

contributes intra-band gradient information while the second summation term contributes supplementary inter-band color information. The second term is used to improve the detection performance in some scenarios when the pixel of interest is on a sharp line of one pixel width.

The ratio of L^H and L^V, which is defined to be

$$E = |\log_2(L^V / L^H)|, \tag{4.6}$$

is then used to classify the region and determine the dominant edge direction. A region is said to be a sharp edge region if its associated E is larger than a predefined threshold value T. Our experimental results showed that T=1 provided a good detection result. We note that eqn. (4.6) is not a completely symmetric function with respect to L^V/L^H. However, this small asymmetry does not affect its detection performance.

If the pixel of interest, say pixel (i,j), locates at the center of a sharp edge region, its interpolation direction $Dir_{i,j} \in \{H,V,D\}$ as well as its green estimate $g_{i,j}$ are determined as follows.

$$\begin{cases} Dir_{i,j} = H \text{ and } g_{i,j} = g^H_{i,j} & \text{if } 2L^{\mathrm{H}} < L^V \\ Dir_{i,j} = V \text{ and } g_{i,j} = g^V_{i,j} & \text{if } L^{\mathrm{H}} > 2L^V \end{cases} \tag{4.7}$$

Otherwise, it is left behind for being processed in Pass 2.

As will be discussed later, to help selecting an interpolator for a pixel, sometimes it is necessary for one to make a preliminary estimation of some other pixels' missing green components. Unlike these estimates which are temporarily obtained for selecting an interpolator, the estimate obtained from eqn. (4.7) is determined with the

selected interpolator. Accordingly, it is referred to as a formal estimate for future reference.

Once a missing green component is handled, the same process is performed for estimating the next missing green component in a raster-scan manner. For estimating the missing green component in the case shown in Figure 4.2b, one can replace the red samples by the corresponding blue samples and follow the procedures above to determine its interpolation direction and its interpolated value. At the end of the first pass, the missing green components of the center pixels in all 5×5 sharp edge regions should be determined.

Pass 2

In the second pass, all missing green components which have not yet been estimated in the first pass are processed in a raster-scan manner. The involved pixels are considered to be in a flat region or a texture region (non-edge region). As a region of size 5×5 does not provide enough information to determine the interpolation direction in pass 1, the local region of interest is extended to 9×9 pixels in Pass 2 to cover more samples such that more useful information can be extracted at a reasonable increase in computation cost. In fact, based on our empirical study, we found that a region of 9×9 pixels provided the best overall zooming performance in terms of signal-to-noise ratio as compared with other possible sizes such as 13×13.

For the case shown in Figure 4.2a, to determine the interpolation direction for the center pixel, the variances of the color difference values of the pixels along the axes of the extended 9×9 local region, say ${}_{H}\Phi_{i,j}$ and ${}_{V}\Phi_{i,j}$, are computed. Figure 4.5 shows the pixels involved in the computation. In formulation, ${}_{H}\Phi_{i,j}$ and ${}_{V}\Phi_{i,j}$ are defined as

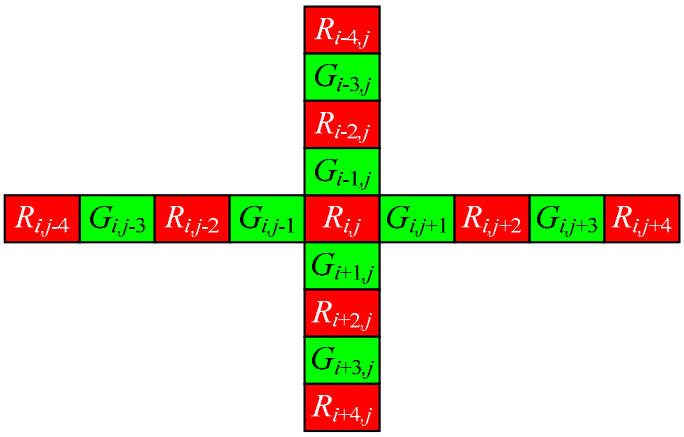


Figure 4.5 The pixels along the horizontal and vertical axes of a 9×9 window in a Bayer CFA image

$$ {}_H\Phi_{i,j} = \sum_{m=-2}^{2}\left(d_{i,j+2m} - \frac{1}{5}\sum_{n=-2}^{2} d_{i,j+2n} \right)^2 $$

and

$$ {}_V\Phi_{i,j} = \sum_{m=-2}^{2}\left(d_{i+2m,j} - \frac{1}{5}\sum_{n=-2}^{2} d_{i+2n,j} \right)^2 \qquad (4.8) $$

where $d_{p,q}$ is a color difference value of pixel (p,q). The values of $d_{i,j+2n}$ and $d_{i+2n,j}$ for $n \in \{0,\pm1,\pm2\}$ are computed as follows.

$$ d_{i,j+2n} = \begin{cases} R_{i,j+2n} - g_{i,j+2n} & \text{if } g_{i,j+2n} \text{ has already been estimated} \\ R_{i,j+2n} - g^{H}_{i,j+2n} & \text{otherwise} \end{cases} \qquad (4.9) $$

and

$$ d_{i+2n,j} = \begin{cases} R_{i+2n,j} - g_{i+2n,j} & \text{if } g_{i+2n,j} \text{ has already been estimated} \\ R_{i+2n,j} - g^{V}_{i+2n,j} & \text{otherwise} \end{cases} \qquad (4.10) $$

Note that the green components for computing $d_{p,q} \in \{d_{i,j+2n}, d_{i+2n,j} \mid n=-1,-2\}$ have been estimated before as all missing green components are estimated in a raster-scan order. As for those for computing $d_{p,q} \in \{d_{i,j+2n}, d_{i+2n,j} \mid n=1,2\}$, they may have been

estimated in the first pass. If they are not, the preliminary estimates $g^{H}_{i,j+2n}$ and $g^{V}_{i+2n,j}$, which are obtained from eqns. (4.1) and (4.2), will be, respectively, used to compute $d_{i,j+2n}$ and $d_{i+2n,j}$.

Sometimes neither a horizontal nor a vertical interpolator can provide a good estimation result and the diagonal interpolator defined in eqn. (4.3) is preferred. Another color difference variance of the pixels within the 9×9 window, say ${}_{D}\Phi_{i,j}$, is used to detect this situation. In formulation, it is defined as

$$ {}_{D}\Phi_{i,j} = \frac{1}{2}\left(\sum_{m=-2}^{2}\left(d_{i,j+2m} - \frac{1}{5}\sum_{n=-2}^{2} d_{i,j+2n} \right)^2 + \sum_{m=-2}^{2}\left(d_{i+2m,j} - \frac{1}{5}\sum_{n=-2}^{2} d_{i+2n,j} \right)^2 \right). \qquad (4.11) $$

Similar to the computation of ${}_{H}\Phi_{i,j}$ and ${}_{V}\Phi_{i,j}$, to get $d_{p,q} \in \{d_{i,j+2n}, d_{i+2n,j} \mid n=0,\pm1,\pm2\}$ for evaluating ${}_{D}\Phi_{i,j}$, the formal estimates of the involved missing green components will be used if they exist. Otherwise, the preliminary estimate $g^{D}_{p,q}$ obtained from eqn. (4.3) is used to get $d_{p,q} = R_{p,q} - g^{D}_{p,q}$.

With parameters ${}_{H}\Phi_{i,j}$, ${}_{V}\Phi_{i,j}$ and ${}_{D}\Phi_{i,j}$, the desired interpolator for estimating the missing green component in the center of the region is selected and its formal estimate is determined by

$$ \begin{cases} Dir_{i,j} = H \text{ and } g_{i,j} = g^{H}_{i,j} & \text{if } {}_{H}\Phi_{i,j} = \min({}_{H}\Phi_{i,j}, {}_{V}\Phi_{i,j}, {}_{D}\Phi_{i,j}) \\ Dir_{i,j} = V \text{ and } g_{i,j} = g^{V}_{i,j} & \text{if } {}_{V}\Phi_{i,j} = \min({}_{H}\Phi_{i,j}, {}_{V}\Phi_{i,j}, {}_{D}\Phi_{i,j}) \\ Dir_{i,j} = D \text{ and } g_{i,j} = g^{D}_{i,j} & \text{if } {}_{D}\Phi_{i,j} = \min({}_{H}\Phi_{i,j}, {}_{V}\Phi_{i,j}, {}_{D}\Phi_{i,j}) \end{cases} . \qquad (4.12) $$

For the case shown in Figure 4.2b, one can treat blue samples as red samples and

follow the above procedures to estimate the missing green component. A complete demosaiced green plane is obtained after Pass 2. Figure 4.3c shows some green planes obtained with the proposed green plane demosaicing scheme.

4.3 Joint Demosaicing and Zooming Algorithm

Since the green channel provides twice information as compared with the red and the blue channels in a CFA image, the proposed joint demosaicing and zooming algorithm starts with green-plane interpolation. The red and the blue plane interpolations then follow with reference to the interpolated green plane. Assume that a CFA image $\boldsymbol{s}$ of size $M \times N$ has to be enlarged to a zoomed full-color image $\boldsymbol{S}$ of size $\lambda M \times \lambda N$. The proposed algorithm supports a zooming factor $\lambda = 2^k$, where k is a positive integer. In this thesis, $\lambda = 2$ is selected for simplicity to facilitate the following discussion.

For the sake of reference, hereafter, a pixel at location (m,n) in image $\boldsymbol{s}$ is denoted by $s_{m,n} = \{s_{r(m,n)}, s_{g(m,n)}, s_{b(m,n)}\}$ with $s_{r(m,n)}$, $s_{g(m,n)}$ and $s_{b(m,n)}$ as its red, green and blue components, while a pixel at location (m,n) in image $\boldsymbol{S}$ is represented by $S_{m,n} = \{S_{r(m,n)}, S_{g(m,n)}, S_{b(m,n)}\}$ with $S_{r(m,n)}$, $S_{g(m,n)}$ and $S_{b(m,n)}$ as the corresponding color components.

Figure 4.6 shows how image $\boldsymbol{S}$ is obtained using CFA image $\boldsymbol{s}$ step by step in different stages for reference. In particular, the circles with segmented edges denote the pixels to be processed in the following processing stage, and the existence of the color in a division denotes that the corresponding color component is known or has been estimated in a previous processing stage.

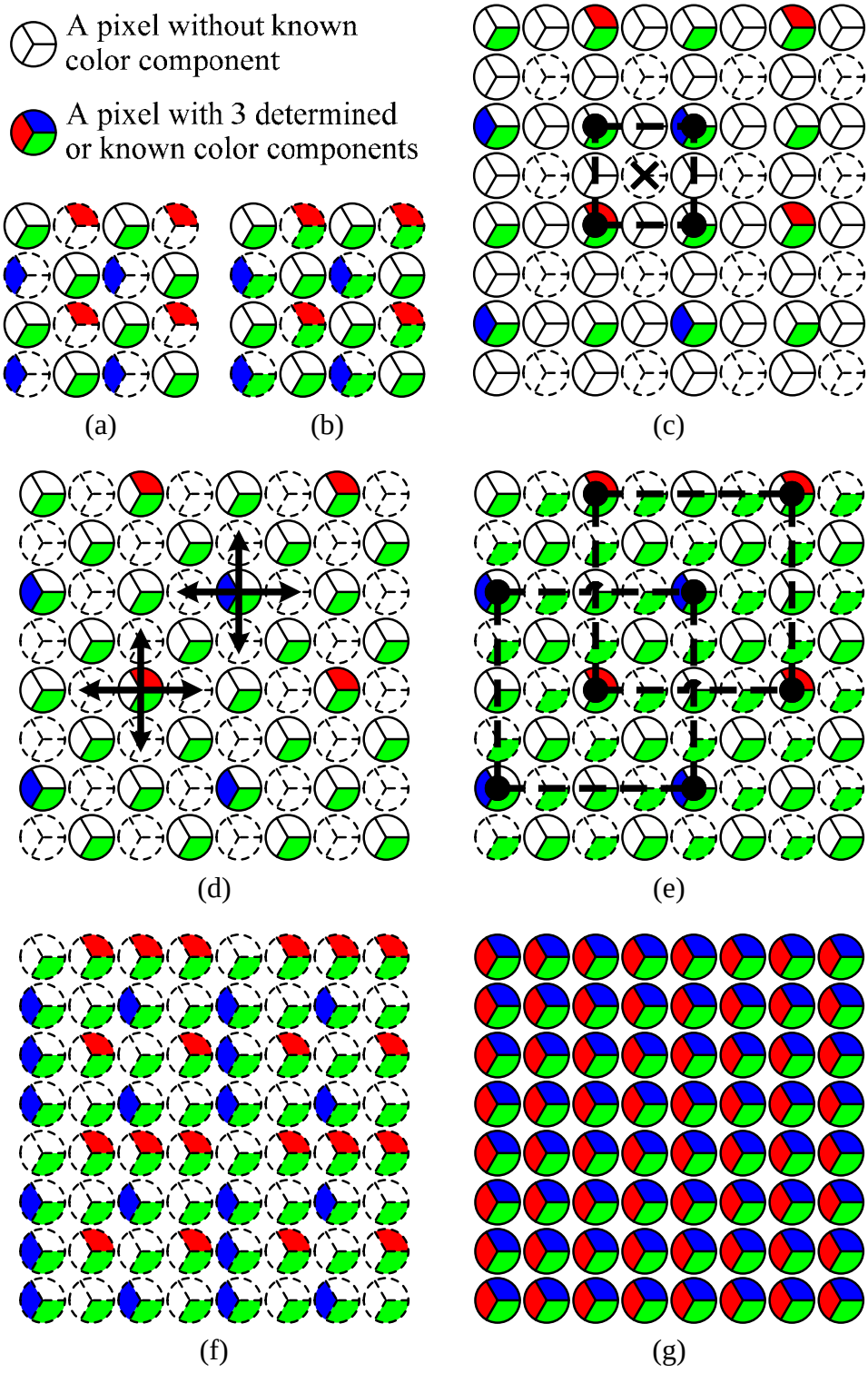


Figure 4.6 Spatial arrangement of the intermediate results of the proposed joint demosaicing and zooming algorithm: (a) raw sensor output CFA image, (b) after demosaicing green-plane, (c) after spatial expansion, (d) after estimating diagonal green components, (e) after estimating all green components (f) intermediate result containing the enlarged CFA image and (g) final enlarged full-color image

4.3.1 Green Plane Demosaicing and Interpolation

In the proposed algorithm, the green plane demosaicing scheme described in Section 4.2 is carried out first to estimate the missing green components in the small CFA image ***s***. It results in a direction map indicating the interpolation directions for the missing green components and a complete demosaiced green plane of image ***s*** as shown in Figure 4.6b.

Median filtering is then applied to the color difference planes of image ***s*** to refine the demosaiced green plane. For a pixel which is in the middle of the pattern shown in Figure 4.2a, its demosaiced green component $s_{g(i,j)}$ is refined by

$$s_{g(i,j)} = \text{median}(d_{(g-r)(i,j)}, d_{(g-r)(i\pm2,j)}, d_{(g-r)(i,j\pm2)}) + s_{r(i,j)} \tag{4.13}$$

where $d_{(g\text{-}r)(m,n)}=s_{g(m,n)}-s_{r(m,n)}$ is the green-to-red color difference of the pixel at location (m,n) and median(•) denotes the median operator which provides the median value of its given inputs. Note that all $s_{r(m,n)}$ involved in eqn. (4.13) are raw sensor components in image ***s***. Median filtering is used here to enhance the demosaiced green plane because of its simplicity and its proven good performance in handling this case. To refine a pixel which is in the middle of the pattern shown in Figure 4.2b, the same refinement step can be used after replacing red samples with corresponding blue samples.

The enhanced partially-demosaiced image ***s*** is then expanded to form an image of the same size as the zoomed image ***S***. In particular, we have

$$S_{2i-1,2j-1} = s_{i,j}, \quad \forall\ (i, j) \ \text{ in image } \boldsymbol{s}. \tag{4.14}$$

Figure 4.6c shows the spatial arrangement of the output of this stage.

In Figure 4.6c, three-fourth of the green components are missing. Those of the pixels marked with segmented edges are interpolated first as each one of them is surrounded by four known green components at its corners as shown in Figure 4.6c. Let $S_{g(p,q)}$ be one of the pixels marked with segmented edges, and $S_{g(p\text{-}1,q\text{-}1)}$, $S_{g(p\text{-}1,q+1)}$, $S_{g(p+1,q\text{-}1)}$ and $S_{g(p+1,q+1)}$ be the four known green components surrounding $S_{g(p,q)}$. The value of $S_{g(p,q)}$ is determined as the weighted sum of $\tilde{S}_{PD}$ and $\tilde{S}_{ND}$, where $\tilde{S}_{PD} \equiv (S_{g(p-1,q+1)} - S_{g(p+1,q-1)})/2$ and $\tilde{S}_{ND} \equiv (S_{g(p-1,q-1)} - S_{g(p+1,q+1)})/2$ are average values of diagonal neighboring green samples of pixel (p,q). In particular, we have

$$S_{g(p,q)} = \frac{\Delta_{PD} \times \tilde{S}_{ND} + \Delta_{ND} \times \tilde{S}_{PD}}{\Delta_{PD} + \Delta_{ND}}, \tag{4.15}$$

$$\Delta_{PD} = \sum_{(a,b) \in \left\{ \begin{matrix} (p,q), \\ (p+2,q),(p,q+2), \\ (p-2,q),(p,q-2) \end{matrix} \right\}} \left| S_{g(a-1,b+1)} - S_{g(a+1,b-1)} \right|$$

and

$$\Delta_{ND} = \sum_{(a,b) \in \left\{ \begin{matrix} (p,q), \\ (p+2,q),(p,q+2), \\ (p-2,q),(p,q-2) \end{matrix} \right\}} \left| S_{g(a-1,b-1)} - S_{g(a+1,b+1)} \right| \tag{4.16}$$

In eqn. (4.15), $\tilde{S}_{ND}$ contributes less to $S_{g(p,q)}$ if Δ_{ND} is larger than Δ_{PD} and vice versa. This weighting mechanism automatically directs the interpolation along an edge if there is. Figure 4.6d shows the spatial arrangement of the output of this stage.

The remaining missing green components in Figure 4.6d are then interpolated by simple directional bilinear interpolation. Let the location of the pixel whose green component is currently being estimated be (p,q). As shown in Figure 4.6d, for each pixel having a missing green component at this stage, there is a neighboring pixel whose green component was determined when demosaicing the green plane of CFA

image $\boldsymbol{s}$. Let the location of this neighboring pixel of $S_{p,q}$ be (i,j). $S_{g(p,q)}$ is then estimated as

$$S_{g(p,q)} = \begin{cases} \dfrac{S_{g(p,q-1)} + S_{g(p,q+1)}}{2} & \text{if } \Gamma(S_{g(i,j)}) = H \\ \dfrac{S_{g(p-1,q)} + S_{g(p+1,q)}}{2} & \text{if } \Gamma(S_{g(i,j)}) = V \\ \dfrac{S_{g(p,q-1)} + S_{g(p,q+1)} + S_{g(p-1,q)} + S_{g(p+1,q)}}{4} & \text{if } \Gamma(S_{g(i,j)}) = D \end{cases} \tag{4.17}$$

where $\Gamma(S_{g(i,j)})$ denotes the interpolation direction for estimating $S_{g(i,j)}$ when demosaicing the green plane of $\boldsymbol{s}$.

In eqn. (4.17), by taking the advantage of the edge consistency in a small region, missing green components in a local region are interpolated with the same direction. This process provides a simple but effective means to preserve edge feature in the zooming result. Figure 4.6e shows the spatial arrangement of the output of this stage where all the missing green components of image $\boldsymbol{S}$ are determined.

4.3.2 Red Plane and Blue Plane Interpolations

To constitute the missing red and blue components in Figure 4.6e, the color difference model used in [26] is employed. For each pixel whose red (blue) component is missing, its green-to-red (green-to-blue) color difference value is bilinearly interpolated from the neighboring pixels having known red (blue) CFA components and its intensity value can then be determined.

For example, when the red components of pixels (p,q), $(p,q+4)$, $(p+4,q)$ and $(p+4,q+4)$ are known and one wants to estimate the red component of pixel $(p+m,q+n)$ for $0 \le m,n \le 4$, the green-to-red color difference value of pixel $(p+m,q+n)$ is first

linearly interpolated as follows.

$$d_{(g-r)(p+m,q+n)} = \frac{4-n}{4}\left(\frac{m}{4}d_{(g-r)(p+4,q)} + \frac{4-m}{4}d_{(g-r)(p,q)}\right) + \frac{n}{4}\left(\frac{m}{4}d_{(g-r)(p+4,q+4)} + \frac{4-m}{4}d_{(g-r)(p,q+4)}\right). \quad (4.18)$$

The missing red color component is then estimated by

$$S_{r(p+m,q+n)} = S_{g(p+m,q+n)} - d_{(g-r)(p+m,q+n)}. \quad (4.19)$$

In practice, the interpolation of all missing red and blue components can be done in parallel so as to reduce the processing time. However, when the components are processed sequentially, one can interpolate some of them first to obtain an intermediate result as shown in Figure 4.6f. This intermediate result contains all color components of image $\boldsymbol{S}$ in CFA format, which allows image $\boldsymbol{S}$ to be stored in CFA format well before its full-color image is finally determined as shown in Figure 4.6g.

Unlike the algorithm proposed in [55] where the color difference information used in the CFA zooming process is obtained from color components located at different positions, the proposed algorithm extract color difference information from color components of the same pixel. This helps to eliminate color artifacts in the resultant image.

4.4 Performance Evaluation

Simulation was carried out to evaluate the performance of the proposed joint demosaicing and zooming algorithm. Some other algorithms for generating zoomed full-color images were also evaluated for comparison. Among them, Lukac's algorithms ((CIZBP) [54] and (DCZPS) [55]) were examples of the demosaicing-after-zooming approach shown in Figure 2.8b. For the zooming-after-demosaicing approach, various demosaicing algorithms in the literature such as ACPI [22], AP [12], PCSD [36], AHDA [35], NCED [27] and BICD [23] were combined with bilinear image zooming algorithms (BI) [44] to produce zoomed full-color images.

Note that some other interpolation algorithms such as cubic interpolation can also work with NCED, PCSD, AHDA, AP, ACPI and BICD to provide a zoomed full-color image. However, BI was used in our simulations as low complexity is one of the key concerns in our study. Other interpolation algorithms are comparatively more complicated. In fact, after demosaicing, sophisticated edge-sensing interpolation algorithms may not provide better zooming performance as compared with BI. When demosaicing and zooming are separately performed, the artifacts introduced during demosaicing may fool an edge-sensing interpolator and an interpolation in a wrong direction may even amplify the artifacts.

The twenty-four original 24-bit (8-bit for each color component) full-color images shown in Appendix A are utilized in the simulation. Each of them is of size 512×768 pixels. They were down-sampled by pixel omission to full-color images of size 256×384 each and then sub-sampled according to the Bayer CFA pattern, with starting sampling sequence of "GRGR…" in the first row, to form a set of small CFA testing images. The CFA testing images were then processed with different evaluated

algorithms to produce zoomed full-color images for comparison.

Table 4.1 tabulates the performance of various algorithms in terms of the color-peak signal-to-noise ratio (CPSNR) of their outputs. The definition of the CPSNR is given by eqn. (3.24) which is described in Chapter 3. From the table, one can observe that the proposed algorithm provides the best performance.

Image	Demosaicing after Zooming Approach						Zooming after Demosaicing Approach				Ours
	ACPI +BI	AP +BI	PCSD +BI	AHDA +BI	NCED +BI	BICD +BI	DCZPS	CIZBP +NCED	CIZBP +AP	CIZBP +AHDA	
1	23.74	24.44	23.97	24.20	24.40	21.60	22.52	22.55	22.23	22.69	24.57
2	30.24	30.46	30.36	30.42	30.69	28.61	29.06	29.12	28.88	29.24	30.55
3	31.20	31.67	31.78	31.87	32.18	29.67	30.17	30.25	29.86	30.37	31.99
4	30.31	30.84	30.72	30.65	31.18	28.49	28.89	28.82	28.55	28.95	30.87
5	23.28	24.33	24.16	24.06	24.50	21.29	22.05	21.91	21.60	22.08	24.43
6	24.79	25.63	25.71	25.73	25.58	23.37	24.06	23.93	23.67	24.09	26.01
7	30.33	30.76	30.68	30.72	31.17	27.23	28.07	28.11	27.71	28.22	30.87
8	21.01	21.57	21.39	21.53	21.56	18.84	19.76	19.81	19.50	19.95	21.86
9	29.69	30.14	30.15	30.22	30.30	26.79	27.87	28.07	27.65	28.25	30.48
10	29.49	30.54	30.26	30.10	30.54	27.38	28.18	28.08	27.78	28.23	30.63
11	26.27	26.97	26.79	26.84	27.04	24.50	25.24	25.19	24.90	25.32	27.12
12	30.67	31.20	31.19	31.24	31.31	28.64	29.37	29.39	29.05	29.57	31.46
13	21.03	22.02	21.71	21.70	22.08	19.82	20.37	20.12	19.91	20.25	22.18
14	25.72	25.95	25.81	25.93	26.40	24.09	24.59	24.52	24.23	24.63	25.98
15	29.91	30.44	30.30	30.29	30.67	28.34	28.66	28.67	28.31	28.79	30.55
16	28.39	29.21	29.33	29.31	29.19	27.04	27.63	27.58	27.31	27.72	29.63
17	29.23	30.14	29.86	29.87	30.24	27.17	27.89	27.81	27.51	27.96	30.28
18	25.36	26.22	25.86	25.86	26.35	23.69	24.36	24.17	23.92	24.30	26.17
19	25.50	26.03	26.19	26.21	25.87	22.95	23.99	24.19	23.85	24.33	26.50
20	28.83	29.43	29.22	29.33	29.65	26.62	27.51	27.66	27.24	27.85	29.77
21	25.51	26.37	26.16	26.18	26.45	23.78	24.53	24.39	24.09	24.53	26.62
22	27.29	27.90	27.64	27.66	28.04	25.63	26.27	26.16	25.91	26.29	27.90
23	31.30	31.79	31.98	32.10	32.16	28.94	29.48	29.44	29.08	29.58	32.22
24	23.79	24.78	24.50	24.43	24.81	22.50	23.04	22.77	22.56	22.92	24.87
Avg.	**27.20**	**27.87**	**27.74**	**27.77**	**28.02**	**25.29**	**25.98**	**25.95**	**25.64**	**26.09**	**28.06**

Table 4.1 CPSNR performance (in dB) of various algorithms in producing a zoomed full-color image

Table 4.2 shows the performance of the evaluated algorithms in terms of S-CIELab color difference. The S-CIELab color difference is defined as the Euclidean distance between the original color of a pixel and its reproduction in S-CIELab color metric space [66]. Again, the proposed algorithm provides the best performance.

Image	Demosaicing after Zooming Approach						Zooming after Demosaicing Approach				Ours
	ACPI +BI	AP +BI	PCSD +BI	AHDA +BI	NCED +BI	BICD +BI	DCZPS	CIZBP +NCED	CIZBP +AP	CIZBP +AHDA	
1	5.755	4.828	5.262	4.460	4.697	8.249	6.547	6.259	6.716	6.134	4.350
2	3.821	3.662	3.724	3.379	3.416	4.848	4.113	3.959	4.151	3.891	3.449
3	2.348	2.213	2.134	1.998	1.974	2.869	2.607	2.435	2.599	2.429	2.060
4	3.398	3.013	3.092	2.859	2.749	4.214	3.620	3.524	3.747	3.509	2.879
5	8.216	6.947	7.104	6.428	6.206	10.490	8.829	8.332	9.183	8.295	6.543
6	4.334	3.536	3.421	3.095	3.454	5.691	4.464	4.389	4.695	4.303	3.081
7	2.821	2.847	2.740	2.698	2.455	4.227	4.069	3.779	4.077	3.876	2.679
8	6.231	5.393	5.433	4.884	5.230	9.220	7.492	6.954	7.577	6.888	4.694
9	2.175	2.009	1.927	1.818	1.858	3.085	2.670	2.447	2.622	2.434	1.808
10	2.247	1.943	1.966	1.840	1.847	3.124	2.619	2.415	2.620	2.402	1.836
11	4.521	3.937	3.978	3.572	3.658	6.049	4.957	4.638	5.055	4.578	3.569
12	1.841	1.652	1.615	1.489	1.539	2.416	2.034	1.914	2.056	1.893	1.489
13	8.754	6.921	7.471	6.580	6.603	10.67	8.426	8.522	9.012	8.330	6.464
14	5.479	5.011	5.088	4.657	4.497	6.899	5.874	5.686	6.100	5.651	4.764
15	3.449	3.181	3.214	2.987	2.930	4.100	3.698	3.518	3.783	3.524	3.041
16	3.332	2.743	2.579	2.365	2.700	4.353	3.492	3.417	3.647	3.359	2.354
17	3.465	2.944	3.078	2.811	2.767	4.571	3.989	3.747	4.030	3.705	2.836
18	6.346	5.500	5.806	5.421	5.131	7.919	6.654	6.555	6.936	6.469	5.462
19	3.925	3.353	3.384	3.061	3.261	5.632	4.497	4.260	4.573	4.216	3.024
20	2.691	2.372	2.448	2.213	2.145	3.476	2.988	2.752	3.041	2.761	2.185
21	4.293	3.537	3.695	3.290	3.329	5.561	4.515	4.374	4.727	4.315	3.256
22	3.880	3.494	3.603	3.414	3.286	4.769	4.043	3.989	4.181	3.954	3.409
23	2.112	2.097	2.069	1.967	1.916	2.489	2.459	2.363	2.470	2.359	1.994
24	4.560	3.847	3.992	3.616	3.574	5.920	4.774	4.621	4.963	4.553	3.574
Avg.	**4.166**	**4.828**	**3.701**	**3.371**	**3.384**	**5.452**	**6.547**	**4.369**	**4.690**	**4.326**	**3.367**

Table 4.2 S-CIELab color difference performance of various algorithms in producing a zoomed full-color image

As mentioned earlier, using sophisticated edge-sensing interpolation algorithms instead of BI after demosaicing in the zooming-after-demosaicing approach may not provide a better zooming performance. Table 4.3 shows the zooming performance of some other combinations [68,69] for comparison. It shows that BI is a good choice in view of both complexity and quality.

Demosaicing algorithm	**Zooming algorithm**		
	Bilinear (BI)	**Cubic [68]**	**Edge-adaptive [69]**
ACPI	**27.20**	27.12	26.72
AP	**27.87**	27.86	27.48
PCSD	**27.74**	27.68	27.29
AHDA	**27.77**	27.70	27.46
NCED	28.02	**28.03**	27.61
BICD	**25.29**	25.26	24.69

(a) CPSNR (in dB)

Demosaicing algorithm	**Zooming algorithm**		
	Bilinear (BI)	**Cubic [68]**	**Edge-adaptive [69]**
ACPI	**4.1664**	4.3199	4.4760
AP	**3.6242**	3.6817	3.7368
PCSD	**3.7009**	3.8086	3.9343
AHDA	**3.3709**	3.4357	3.4744
NCED	**3.3841**	3.4306	3.5173
BICD	**5.4518**	5.7043	6.1156

(b) S-CIELab color difference

Table 4.3 Average performance of various combinations in producing a zoomed full-color image when the zooming-after-demosaicing approach is used. (a) CPSNR (in dB) and (b) S-CIELab color difference

Objective measures may not be accurate and reliable enough to tell the quality difference among the processing results. Figures 4.7 and 4.8 show, respectively, some zooming results of images 19 and 8 for visual comparison. They reveal that the proposed algorithm outstandingly preserves the image features with less color artifacts. For example, as shown in Figure 4.7, the proposed algorithm can reproduce the fence texture with very little color artifacts but the others cannot. One can also see from Figure 4.8 that, while most of the other algorithms totally destroy the letters on the wall in their outputs, the proposed algorithm can preserve most of the details and introduces less color artifacts. These results, to a large extent, reflect the robustness of the proposed joint demosaicing and zooming algorithm in which the interpolation direction is determined directly from the raw sensor data and then used effectively and consistently in different stages to interpolate the green plane.

CIZBP [54] was a dedicated algorithm proposed for zooming CFA images. To make a direct comparison with CIZBP, zoomed CFA images produced by the proposed algorithm were extracted from the intermediate results having the available color components shown in Figure 4.6f for comparison. Corresponding reference CFA images were generated by sub-sampling I_o according to the Bayer pattern. The difference between the zoomed CFA image output of a particular algorithm and its corresponding reference was then measured in terms of $PSNR=10\log_{10}(255^2/MSE)$, where *MSE* is the mean square error of all available color components in the involved CFA image. On average, the PSNRs achieved by CIZBP and the proposed algorithm are 25.51dB and 28.22dB respectively.

(a) input CFA image

(b) the full-color original

(c) PCSD+BI

(d) BICD+BI

(e) ACPI+BI

(f) AHDA+BI

(g) AP+BI

(h) NCED+BI

(i) DCZPS

(j) CIZBP+AP

(k) CIZBP+AHDA

(l) CIZBP+NCED

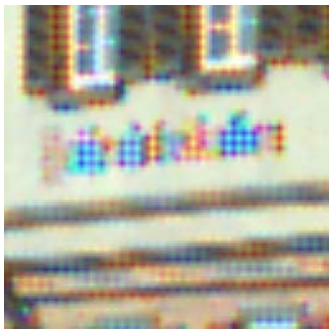
(m) Ours

Figure 4.7 Part of the processing results of Image 19 for visual comparison

(a) input CFA image

(b) the full-color original

(c) PCSD+BI



(d) BICD+BI

(e) ACPI+BI

(f) AHDA+BI

(g) AP+BI

(h) NCED+BI

(i) DCZPS

(j) CIZBP+AP

(k) CIZBP+AHDA

(l) CIZBP+NCED

(m) Ours

Figure 4.8 Part of the processing results of Image 8 for visual comparison

4.5 Computational Complexity

In this section, a computational complexity analysis of the proposed algorithm is provided. The complexity is measured in terms of number of primitive operations including addition (ADD), multiplication (MUL), bit-shift (SHT) and taking absolute value (ABS). A comparison is considered as an addition for easier presentation.

Table 4.4 summarizes the complexity required by the proposed algorithm to estimate a missing component in different stages. The stages into which the algorithm is decomposed in our complexity analysis matches those presented in Figure 4.6 for easy reference. Note that some intermediate computation results can be reused in later stages and this was taken into account when the complexity of the proposed algorithm was estimated.

Action taken in a particular stage			**Operation**			
Action	**Stage**		**ADD**	**MUL**	**SHT**	**ABS**
Estimating a missing G component	(a)→(c) in Figure 4.6	In sharp edge region	35	1	2	8
		In non-sharp edge region	93	25	6	8
	(c)→(d) in Figure 4.6		14	3	2	2
	(d)→(e) in Figure 4.6		3	0	1	0
Estimating a missing R or B component	(e)→(f) in Figure 4.6		2	1	1	0
	(f)→(g) in Figure 4.6		2	1	1	0

Table 4.4 Complexity required for estimating a missing component in different stages of the proposed joint demosaicing and zooming algorithm

In practice, the real number of operations required to produce a zoomed full-color image with the proposed algorithm is image-dependent and it relies on how many missing green components are in sharp edge regions. Table 4.5 lists the average number of operations per pixel required by different algorithms in processing 24 testing images in our simulations. The complexity of BICD+BI and ACPI+BI is lower

than that of the proposed algorithm but their output quality is poor as shown in Figures 4.7 and 4.8. As for PCSD+BI and AHDA+BI, their complexity is close to ours but their output qualities are a bit lower especially in terms of the CIELab color difference criterion. The complexity of the proposed algorithm is around 22% of that of DCZPS [55]. When zooming a CFA image, the complexity of the proposed algorithm is around half of that of CIZBP [54]. In fact, the computation effort for the proposed algorithm to produce a zoomed full-color image is less than that required for CIZBP [54] to produce a zoomed CFA image.

Algorithms	**ADD**	**MUL**	**SHT**	**ABS**	**Total**
BICD+BI	5.25	0.00	4.25	0.00	**9.50**
Producing a zoomed full-color image					
ACPI+BI	6.25	0.00	3.49	0.99	**10.73**
PCSD+BI	28.25	2.00	3.50	3.00	**36.75**
AHDA+BI	26.00	2.00	8.25	1.00	**37.25**
AP+BI	64.90	7.88	32.25	0.50	**105.53**
NCED+BI	71.25	34.50	10.25	2.00	**118.00**
DCZPS	67.50	43.44	0.00	16.88	**127.82**
Ours	19.02	3.68	3.41	1.43	**27.54**
Producing a zoomed CFA image					
CIZBP	25.88	14.25	0.00	4.50	**44.63**
Ours	16.27	3.68	2.03	1.43	**23.41**

Table 4.5 Averaged number of operations per pixel required by various algorithms

In our simulations, the average execution time for the proposed algorithm to produce a zoomed full-color image from a 256×384 CFA image with a zooming factor of 2 on a 3.0GHz Pentium 4 PC with 1024MB RAM is 0.062s.

4.6 Chapter Summary

In this chapter, a low complexity joint demosaicing and zooming algorithm is proposed. With the use of the local color difference variances, the raw sensor data is considered directly to determine the interpolation direction for estimating the missing green components in the zoomed image. With this arrangement, the green plane can be efficiently interpolated with preserved image details. With reference to the interpolated green plane, the red and the blue planes are then interpolated at low complexity. Simulation results show that the proposed algorithm produces images providing the most details and the least color artifacts at the lowest complexity as compared with conventional approaches which generally perform demosaicing-after-zooming or zooming-after-demosaicing.

Chapter 5 Color Demosaicing Algorithm II: On the Use of Integrated Gradient for Color Demosaicing

5.1 Introduction

Though early demosaicing algorithms [22,28,30,67,70] try to extract gradient information from color intensity domains to guide the interpolation in corresponding color planes, recent demosaicing algorithms [36-38] generally put their focus on the color difference domains and exploit the inter-channel spectral correlation to guide the interpolation. However, in either approach, the information extracted from the color difference and the color intensity domains is not properly balanced. This bias may result in misleading information which guides one to interpolate samples along a wrong direction.

In order to improve the output quality, some demosaicing algorithms such as [28] and [30] re-extract gradient information from intermediate interpolation results obtained at different stages. This adapt-to-new-information approach is great but it does not always work properly. For example, in texture areas where the spectral correlation is locally ambiguous along the horizontal and vertical directions, the gradient information extracted at different stages can be mutually contradictory and makes the situation confusing. Besides, gradient estimation is generally computationally expensive and hence repeated estimation increases the complexity a lot.

Color artifacts in high frequency areas are commonly found in the outputs of various demosaicing algorithms. The most common solution for this problem is to

introduce a post-processing step [29,51] to suppress the artifacts after demosaicing. Nevertheless, as all missing samples have to be processed again in the post-processing stage, it seems not an efficient approach from system point of view. Besides, as the interpolation of the red and the blue planes is carried out before the enhancement of the green plane, one cannot make use of the enhanced green plane as a "better" reference to interpolate the red and the blue planes. This also lowers the efficiency.

In this chapter, an efficient decision-based demosaicing algorithm is proposed to reconstruct color images from Bayer images. It aims at producing high quality output images at a low computation cost. Figure 5.1a shows a flow diagram of the proposed demosaicing algorithm. In this algorithm, a new directional edge-sensing parameter called *integrated gradient* (IG), which extracts gradient information in both color intensity (CI) and color difference (CD) domains simultaneously, is defined for being shared in various stages throughout the demosaicing process to interpolate the color channels. This IG is not only used as an edge detector to determine the interpolation direction when interpolating a green sample, but also used to adjust the coefficients of two spatial-variant interpolators when estimating the missing red and blue samples. In addition, a green plane enhancement which works with the IG is introduced to further improve the algorithm's performance. Simulation results confirmed that the proposed demosaicing algorithm provides superior output quality at a comparatively low computation cost.

This chapter is organized as follows. In Section 5.2, the proposed IG is discussed. Section 5.3 presents the details of the proposed demosaicing algorithm. Section 5.4 provides some experimental results of the proposed algorithm for comparison study while Section 5.5 shows the computational complexity of the proposed algorithm. Finally, a conclusion is given in Section 5.6.

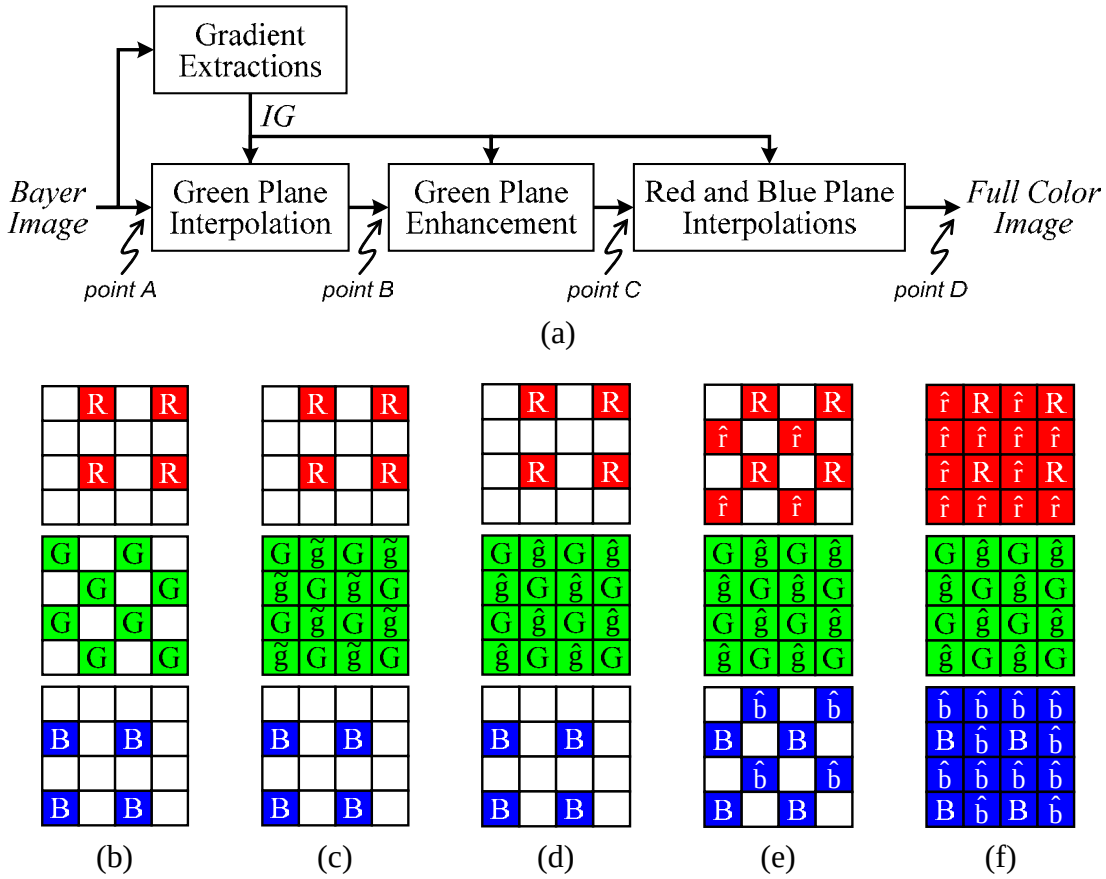


Figure 5.1 The spatial arrangement of the proposed demosaicing algorithm's intermediate results: (a) workflow of the algorithm, (b) raw sensor output Bayer image at point A, (c) green plane interpolation result at point B, (d) green plane enhancement result at point C, (e) intermediate interpolation result during red and blue plane interpolations, and (f) final interpolation result at point D

5.2 Extraction of Integrated Gradient

Since our human visual system is sensitive to edge structures, many demosaicing algorithms try to avoid doing interpolation across edges. To achieve the goal, gradients are estimated in various directions at each pixel to guide the interpolation along an edge.

A similar idea is exploited in the proposed algorithm. The contrast is that, instead of using the intensity gradient as in conventional approaches, the proposed algorithm uses a measure called integrated gradient (IG) to guide the interpolation. This IG is a

combination of the gradients in both the color intensity (CI) domain and the color difference (CD) domain, which provides more information for one to reach a better decision in selecting the interpolation direction. This section presents the definition of this IG and its rationale.

The IG associated with a particular pixel is defined on the neighboring samples of the pixel in its local region. In practice, the available samples in the 5×5 local region of a pixel can be in one of four patterns shown in Figure 5.2.

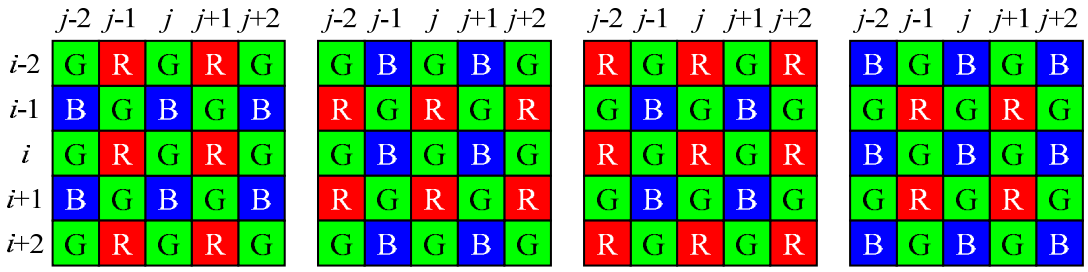


Figure 5.2 Four 5×5 regions of Bayer CFA pattern having their centers at (a)-(b) green, (c) red and (d) blue Bayer samples

Before we define IG, let's first define an intermediate measure called weaker integrated gradient (WIG). It is so called because, as compared with IG, it extracts information from fewer CI and CD planes and hence provides less information for edge detection.

Figure 5.3a shows a cross template for extracting Bayer samples to calculate the WIG of the pixel located at the template center. The template has four extensions. The Bayer samples covered in a particular extension of the template (including the center) are used to compute the WIG along the corresponding direction. No matter which possible 5x5 Bayer pattern is concerned (see Figure 5.2), after rotating the Bayer sample patterns covered by the northbound, the westbound and the southbound

extensions of the cross template by 90º, 180º and 270º respectively, they are all in the same standard pattern form shown in Figure 5.3b as those patterns covered by the eastbound extension do. Hence, as long as the eastbound WIG is defined, the westbound, the northbound and the southbound WIGs can also be defined in the same manner. To save the effort, here we just define the eastbound WIG.

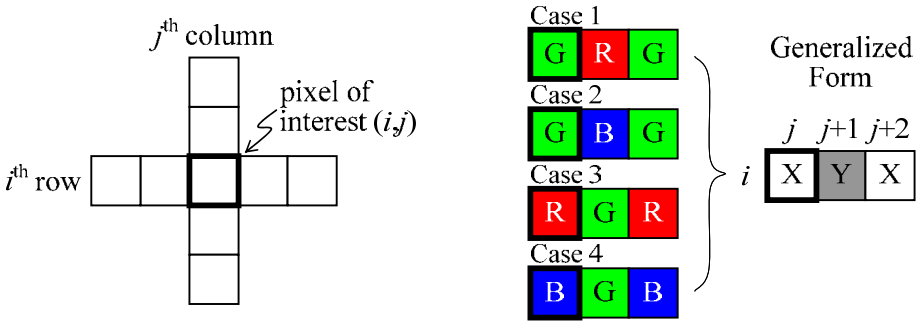


Figure 5.3 (a) a cross template for computing WIGs, (b) all possible Bayer patterns covered for computing the eastbound WIG of a pixel and their generalized form

In the generalized form shown in Figure 5.3b, X and Y denote the colors of the corresponding Bayer samples in a concerned pattern. In formulation, the eastbound WIG is defined as

$$\delta_{XY}^{E}(i,j) = \bar{\delta}_{X}^{E}(i,j) + \alpha\tilde{\delta}_{XY}^{E}(i,j) \tag{5.1}$$

where $\bar{\delta}_{X}^{E}(i,j)$ and $\tilde{\delta}_{XY}^{E}(i,j)$ are, respectively, the eastbound CI and CD gradients of pixel (i,j). α is a weighting factor used to control the contribution of the two gradients. The determination of its value will be discussed later. In the notations, superscript E and subscripts XY and X, respectively, denote the direction of the WIG,

the involved CD plane and the involved CI plane.

The CI gradient $\bar{\delta}_X^E(i, j)$ measures the extent of eastbound CI change and is defined on the color channel that contains the Bayer sample of pixel (*i*,*j*). Specifically, we have

$$\bar{\delta}_X^E(i, j) = \left| X(i, j) - X(i, j+2) \right| \tag{5.2}$$

where $X(i,j)$ is the known Bayer sample at position (*i*,*j*). The CI gradient is used to identify edges and hence only the gradient magnitude is concerned. There is likely an edge if $\bar{\delta}_X^E(i, j)$ is large. In fact, $\bar{\delta}_X^E(i, j)$ is commonly used in a number of conventional demosaicing algorithms for edge detection [13,67]. However, since pixels (*i*,*j*) and (*i*,*j*+2) are two pixels apart, the resolution that $\bar{\delta}_X^E(i, j)$ supports may not be able to detect a thin line of one pixel width.

The CD gradient $\tilde{\delta}_{XY}^E(i, j)$ is introduced to solve this problem. It provides supplementary edge detection information by evaluating the CD change of two successive pixels along the same eastbound direction. In formulation, it is defined as

$$\tilde{\delta}_{XY}^E(i, j) = \frac{|\, d_{XY}(i, j) - d_{XY}(i, j+1) \,| + |\, d_{XY}(i, j+1) - d_{XY}(i, j+2) \,|}{2} \tag{5.3}$$

where $d_{XY}(i, j)$ represents the X-Y CD value at position (*i*,*j*). The determination of $d_{XY}(i, j)$ will be discussed later.

From eqn. (5.3), one can see that $\tilde{\delta}_{XY}^E(i, j)$ is contributed by two items, each of which provides the absolute value of the CD change within one pixel along the eastbound direction. This increase in resolution allows one to detect a line of one pixel

width. In particular, the 1st item is for detecting the edge between pixels (i,j) and (i,j+1), while the 2nd item is for detecting the edge between pixels (i,j+1) and (i,j+2). When there is a line of one pixel width passing pixel (i,j+1), both edges on (i,j+1)'s left and right can be detected. In contrast, $\overline{\delta}_X^E(i, j)$ fails in such a case. $\widetilde{\delta}_{XY}^E(i, j)$ is the average of these two magnitude items and hence the detection result can be faithfully reported. The absolute value nature of the items prevents their detection results from canceling with each other in averaging.

The WIG of pixel (i,j) only provides the edge information extracted from the X plane and the X-Y plane. It might happen that, over the edge to be detected, there is only sharp change in the Y plane or another CD plane. In that case, WIG fails to detect the edge. To solve this problem, more CI or CD planes should be included in the detection. One of the possible solutions is to combine the eastbound WIGs of pixels (i-1,j), (i,j) and (i+1,j) to form the eastbound IG of pixel (i,j).

As an example, for the case shown in Figure 5.2b, the eastbound IG of (i,j) can be defined as

$$\begin{aligned}\Delta^E(i, j) &= 2\delta_{GB}^E(i, j) + \sum_{k=\pm1}\delta_{RG}^E(i+k, j) \\ &= \left(2\overline{\delta}_G^E(i, j) + \sum_{k=\pm1}\overline{\delta}_R^E(i+k, j)\right) + \alpha\left(2\widetilde{\delta}_{GB}^E(i, j) + \sum_{k=\pm1}\widetilde{\delta}_{RG}^E(i+k, j)\right)\end{aligned}$$

for the case shown in Figure 5.2b. (5.4)

Note that $\delta_{GB}^E(i, j)$ is weighted by 2 to balance the contribution of the CD (CI) gradients extracted from different CD (CI) planes to $\Delta^E(i, j)$. By so doing, CD gradients from R-G and G-B planes have equal votes in eqn. (5.4). Similarly, CI gradients from R and G planes also have equal votes.

However, this definition of IG makes IGs along different directions incompatible. Though the Bayer patterns shown in Figures 5.2c and 5.2d are symmetric in all four (*E*,*S*,*W*,*N*) directions, the ones shown in Figure. 5.2a and 5.2b are not. To obtain the northbound IG of the pixel (*i*,*j*) shown in Figure 5.2b, one can rotate Figure 5.2b clockwise by 90º and then compute the eastbound IG of the rotated version with eqn. (5.4). In other words, we have

$$\Delta^N(i,j) = 2\delta_{GR}^N(i,j) + \sum_{k=\pm1}\delta_{BG}^N(i,j+k)$$
$$= \left(2\bar{\delta}_G^N(i,j) + \sum_{k=\pm1}\bar{\delta}_B^N(i,j+k)\right) + \alpha\left(2\tilde{\delta}_{GR}^N(i,j) + \sum_{k=\pm1}\tilde{\delta}_{BG}^N(i,j+k)\right)$$

for the case shown in Figure 5.2b. (5.5)

From eqns. (5.4) and (5.5), one can see that $\Delta^E(i,j)$ carries information from the R plane but not the B plane while $\Delta^N(i,j)$ does the opposite. They are not compatible and hence it does not make sense to compare them.

By considering the aforementioned compatibility constraint, we regulate the definition of IG to eliminate the items that cause the incompatibility and, accordingly, modify eqns. (5.4) and (5.5) as

$$\Delta^E(i,j) = \bar{\delta}_G^E(i,j) + \alpha\left(2\tilde{\delta}_{GB}^E(i,j) + \sum_{k=\pm1}\tilde{\delta}_{RG}^E(i+k,j)\right)$$

for the case shown in Figure 5.2b. (5.6)

and $$\Delta^N(i,j) = \bar{\delta}_G^N(i,j) + \alpha\left(2\tilde{\delta}_{GR}^N(i,j) + \sum_{k=\pm1}\tilde{\delta}_{BG}^N(i,j+k)\right)$$

for the case shown in Figure 5.2b. (5.7)

Note that $\bar{\delta}_G^E(i,j)$ and $\bar{\delta}_G^N(i,j)$'s original scaling factor of 2 is also eliminated in

eqns. (5.6) and (5.7). However, this can be compensated for by adjusting weighting factor α at the end.

Due to the absolute value nature of $\tilde{\delta}^E_{XY}(i,j)$ (see eqn. (5.3)) and the fact that $d_{XY}(i,j) = -d_{YX}(i,j)$, we have $\tilde{\delta}^E_{XY}(i,j) = \tilde{\delta}^E_{YX}(i,j)$. In other words, both $\Delta^E(i,j)$ and $\Delta^N(i,j)$ now carry the information extracted from the same G, G-R and G-B planes, and hence they are compatible. As a matter of fact, with this regulated definition, all four IGs of any particular pixel are compatible.

To maintain this compatibility, though theoretically a full-color image of 3 color components contains 3 CI and 3 CD planes, we do not further include more CI and CD planes in the definition of IG. This final definition of IG is used in the proposed demosaicing algorithm.

Similarly, the eastbound IGs of pixel (*i*,*j*) in cases shown in Figures 5.2a, 5.2c and 5.2d are, respectively, defined as

$$\Delta^E(i,j) = \bar{\delta}^E_G(i,j) + \alpha\left(2\tilde{\delta}^E_{GR}(i,j) + \sum_{k=\pm 1}\tilde{\delta}^E_{BG}(i+k,j)\right)$$

for the case shown in Figure 5.2a. (5.8)

$$\Delta^E(i,j) = \bar{\delta}^E_R(i,j) + \alpha\left(2\tilde{\delta}^E_{RG}(i,j) + \sum_{k=\pm 1}\tilde{\delta}^E_{GB}(i+k,j)\right)$$

for the case shown in Figure 5.2c. (5.9)

and $$\Delta^E(i,j) = \bar{\delta}^E_B(i,j) + \alpha\left(2\tilde{\delta}^E_{BG}(i,j) + \sum_{k=\pm 1}\tilde{\delta}^E_{GR}(i+k,j)\right)$$

for the case shown in Figure 5.2d. (5.10)

The northbound, the westbound and the southbound IGs of a pixel can be defined similarly as before. As a matter of fact, they can be determined by rotating the Bayer

image clockwise by 90º, 180º and 270º respectively and then computing the eastbound IGs of the rotated versions.

As a final remark, we note that, due to the absolute value nature of each component in the definition of IG, we have $\Delta^E(i,j) = \Delta^W(i,j+2)$ and $\Delta^S(i,j) = \Delta^N(i+2,j)$. By making use of this property, one can save an amount of realization effort.

5.2.1 Color Difference Estimation for Computing IGs

Neither $d_{RB}(i,j)$ nor $d_{BR}(i,j)$ is required in the computation of IGs. As for $d_{GB}(i,j)$, $d_{BG}(i,j)$, $d_{GR}(i,j)$ and $d_{RG}(i,j)$, not every one of them is required and their estimation depends on which IG of pixel (*i*,*j*) is evaluated.

When computing an eastbound or westbound (northbound or southbound) IG, preliminary estimates of the required CDs are estimated along a row (column). For example, to estimate $d_{XY}(i,j+1)$ for the pixel (*i*,*j*+1) shown in the generalized pattern in Figure 5.3b when evaluating $\tilde{\delta}^E_{XY}(i,j)$, a preliminary estimate $d'_{XY}(i,j+1)$ is first obtained with

$$d'_{YX}(i,j+1) = \begin{bmatrix} X(i,j) & Y(i,j+1) & X(i,j+2) \end{bmatrix} \cdot \begin{bmatrix} -0.5 \\ 1 \\ -0.5 \end{bmatrix}. \qquad (5.11)$$

Figure 5.4a shows how to generate the CD planes for computing eastbound/westbound IGs. In a Bayer image, the sensor pattern repeats every other row and every other column. For each row, the Bayer sample sequence is convolved with [-0.5, 1, -0.5] to provide preliminary CD estimates with eqn. (5.11). The output

is a sequence of alternate preliminary estimates d'_{GR} and d'_{RG} (d'_{BG} and d'_{GB}). Since we have $d_{XY}(i, j) = -d_{YX}(i, j)$ in theory, by negating every second estimate in a row, we have a row of d'_{GR} (d'_{GB}). A 3-point averaging filter is then applied to remove the potential high frequency noise in each row, which produces the final CD estimates d_{GR} (d_{GB}). The G-R (G-B) plane is constructed with all odd (even) rows.

To get the required G-R and G-B planes for computing the northbound or southbound IG, one can follow the same procedures as presented in Figure 5.4a after rotating the Bayer image by 90º.

Figure 5.5 shows the CD planes of a testing image for computing IGs of different directions. The vertical (horizontal) resolution supported by the CD planes shown in Figure 5.5b (Figure 5.5c) is identical to that of the original image, which enables one to detect a one-pixel thin line. One can see that some edges can only be detected in a particular CD plane but not the other one. That explains why IG instead of WIG is used in our detection process.

5.2.2 Realization in Practice

Figure 5.4a presents the idea how CDs are estimated step by step. These steps can be combined to make their realization much easier by making use of the following facts. First, the functions of the two approaches of implementation shown in Figure 5.4b are identical so one can replace the upper approach with the lower one in the realization. Second, the negation process in the lower approach can be skipped as $|d_{XY}(i, j) + d_{YX}(i, j+1)|$, the absolute value of the sum of two successive output values of the filter, equals to $|d_{XY}(i, j) - d_{XY}(i, j+1)|$. It is already the information required for computing $\tilde{\delta}^{E}_{XY}(i, j)$ and hence the corresponding IG.

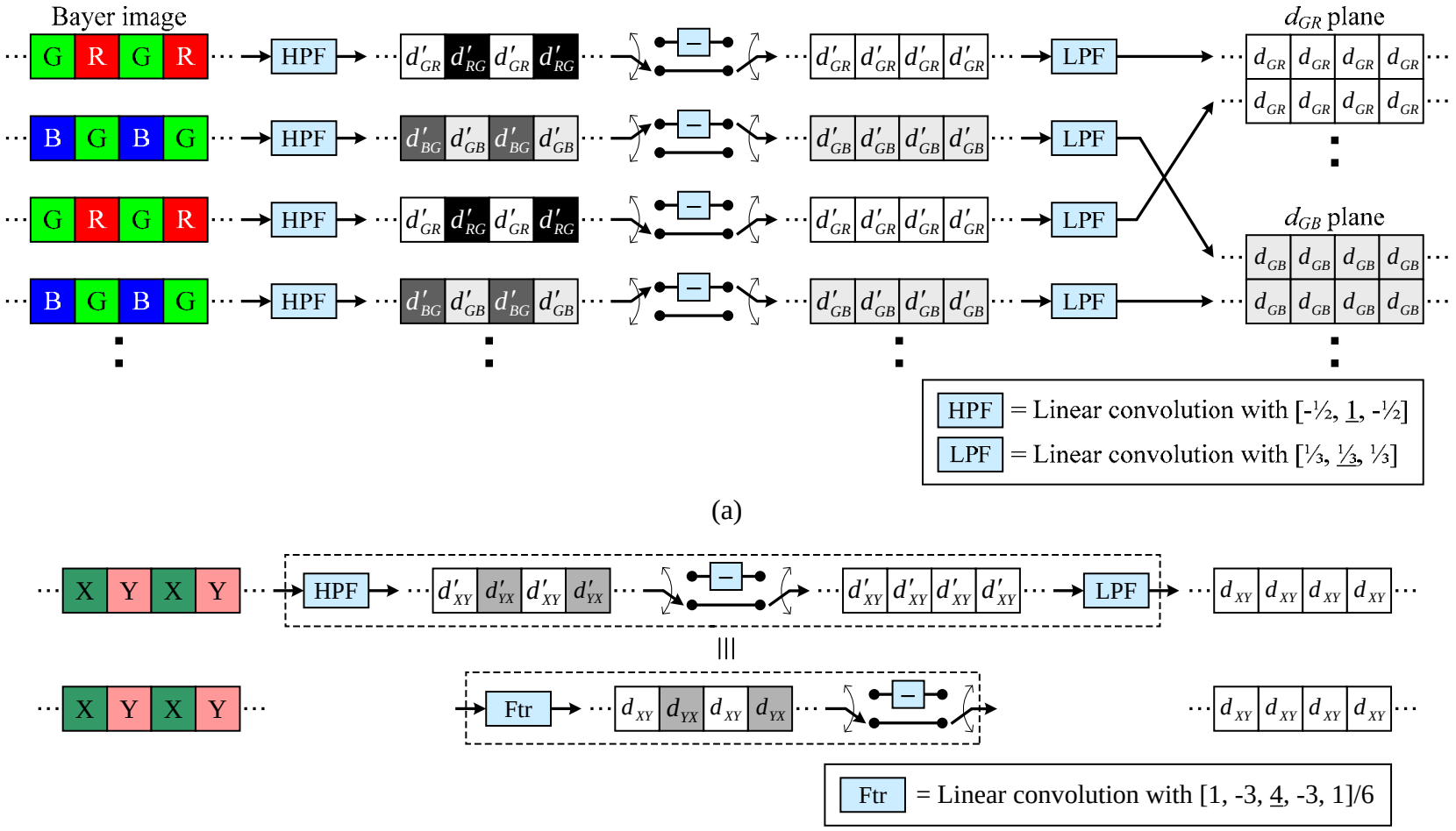
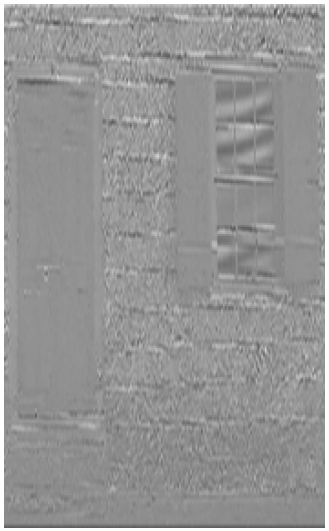


Figure 5.4 Estimation of color difference values for computing eastbound or westbound IGs. (a) realization procedure and (b) alternative implementations of the core module form

Original

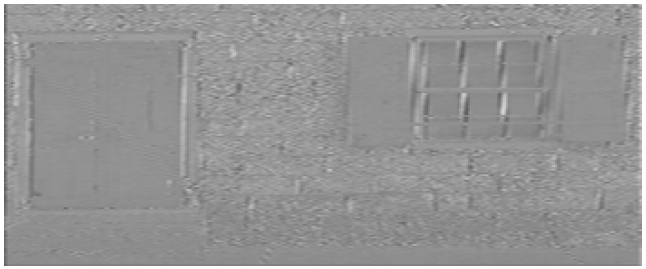
(a)



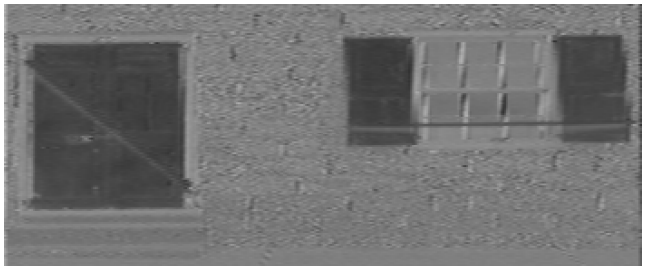
d_{GB} plane

d_{GR} plane

(b)



d_{GB} plane



d_{GR} plane

(c)

Figure 5.5 (a) Part of a testing image, (b) color difference planes for computing northbound or southbound IGs and (c) color difference planes for computing eastbound or westbound IGs

Based on these facts, one can first convolve each row of the Bayer image with filter kernel $\boldsymbol{F}$ = [1,-3,4,-3,1]*[1,1] = [1,-2,1,1,-2,1], where the * sign denotes convolution and the underscore marks the position of the pixel of interest, and combine the filter outputs of all rows to form an intermediate plane $s(i,j)$ ($= 6(d_{XY}(i,j) + d_{YX}(i,j+1))$ if (i,j) carries a X Bayer sample, or $= 6(d_{YX}(i,j) + d_{XY}(i,j+1))$ if (i,j) carries a Y Bayer sample). $\tilde{\delta}^{E}_{XY}(i,j)$ can then be obtained with

$$\tilde{\delta}^{E}_{XY}(i,j) = \frac{1}{12}\left(|s(i,j)| + |s(i,j+1)|\right). \tag{5.12}$$

5.2.3 Determination of Parameter α

Parameter α is a weighting factor used to control the contribution of the CI and CD gradients to IG. An empirical study was carried out to investigate its impact to the demosaicing performance. It was found that the optimum in terms of CPSNR happened at around α=2 and, within the range of $1<\alpha<6$, the CPSNR variation of the results achieved by the proposed demosaicing algorithm was less than 0.05dB as shown in Figure 5.6. By considering this, the value of α is selected to be 3/2 such that the computation of $\alpha\tilde{\delta}^{E}_{XY}(i,j)$ using the approach presented in Section 5.2.2 only involves shift and add operations. This saves realization effort.

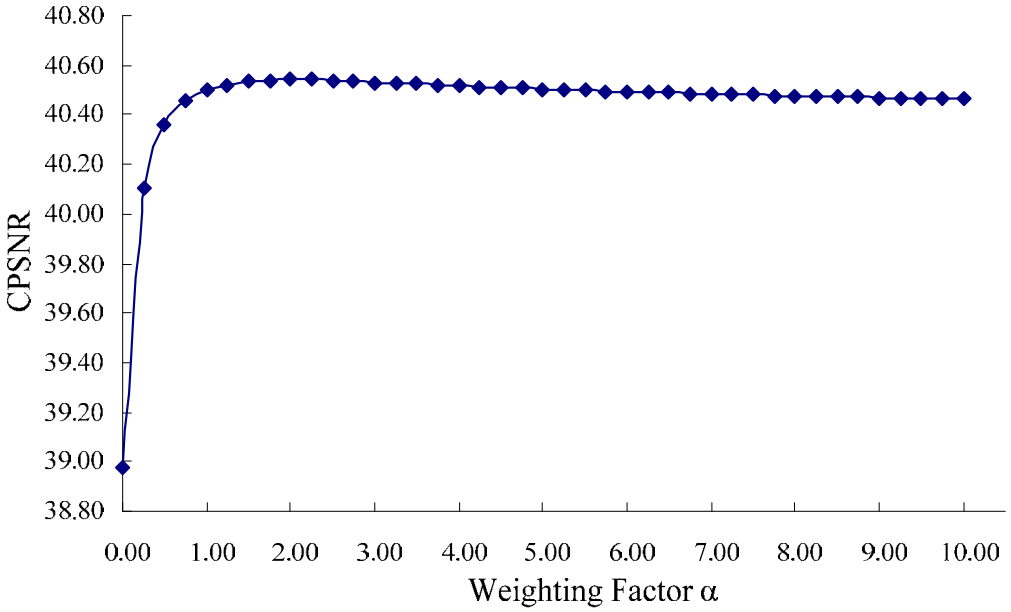


Figure 5.6 CPSNR performances of the proposed demosaicing algorithm achieved with different α values

5.3 Proposed Demosaicing Algorithm

Figure 5.1 briefly illustrates the workflow of the proposed demosaicing algorithm. After IG extraction, the proposed algorithm interpolates the green plane first. The resultant green plane is then enhanced to provide a reference for the subsequent red plane and blue plane interpolations. The extracted IGs are used in various stages of the proposed algorithm to improve the interpolation efficiency. The details of these stages are described in this section.

For the sake of reference, hereafter, a pixel at location (i,j) in the Bayer image is represented by either $(R(i,j), g(i,j), b(i,j))$, $(r(i,j), G(i,j), b(i,j))$ or $(r(i,j), g(i,j), B(i,j))$, where $R(i,j)$, $G(i,j)$, and $B(i,j)$ denote the known red, green and blue Bayer samples and $r(i,j)$, $g(i,j)$ and $b(i,j)$ denote the unknown samples of corresponding color channels in the image. The final estimates of $r(i,j)$, $g(i,j)$ and $b(i,j)$ are denoted as $\hat{r}(i,j)$, $\hat{g}(i,j)$ and $\hat{b}(i,j)$ respectively for clear presentation.

5.3.1 Green Plane Interpolation

As far as a pixel which does not have a green Bayer sample is concerned, the pattern of its local region must be in the form shown in either Figures 5.2c or 5.2d. Without losing generality, the former pattern is discussed here. For the pattern shown in Figure 5.2d, one can exchange the red samples with the corresponding blue samples and then performs the interpolation in the same way.

For the case shown in Figure 5.2c, the missing green sample of pixel (*i*,*j*) can be interpolated with one of the following Laplacian interpolation filters as proposed in [22].

$$g_{i,j}^{H} = \frac{G(i, j-1) + G(i, j+1)}{2} + \frac{2R(i, j) - R(i, j-2) - R(i, j+2)}{4} \tag{5.13}$$

$$g_{i,j}^{V} = \frac{G(i-1, j) + G(i+1, j)}{2} + \frac{2R(i, j) - R(i-2, j) - R(i+2, j)}{4} \tag{5.14}$$

$$g_{i,j}^{D} = \frac{g_{i,j}^{H} + g_{i,j}^{V}}{2} \tag{5.15}$$

where $g_{i,j}^{H}$, $g_{i,j}^{V}$ and $g_{i,j}^{D}$ are, respectively, the estimates obtained with the corresponding horizontal, vertical and diagonal interpolators.

The selection of the interpolators is critical to the demosaicing performance. In the proposed demosaicing algorithm, the high performance two-step estimation scheme proposed in Section 4.2 is modified in three aspects to estimate the missing green samples at a reduced complexity. First, instead of the parameters L^H and L^V used in Section 4.2, IGs are utilized to determine the interpolation direction of the missing green sample to improve the performance. Second, the pixels in flat intensity regions are processed in the first pass rather than in the second pass to save

computation effort. Third, in the second pass, the analysis on the variance of local CD values is simplified without sacrificing its reliability. Figure 5.7 summarizes the procedures of the proposed two-step estimation scheme.

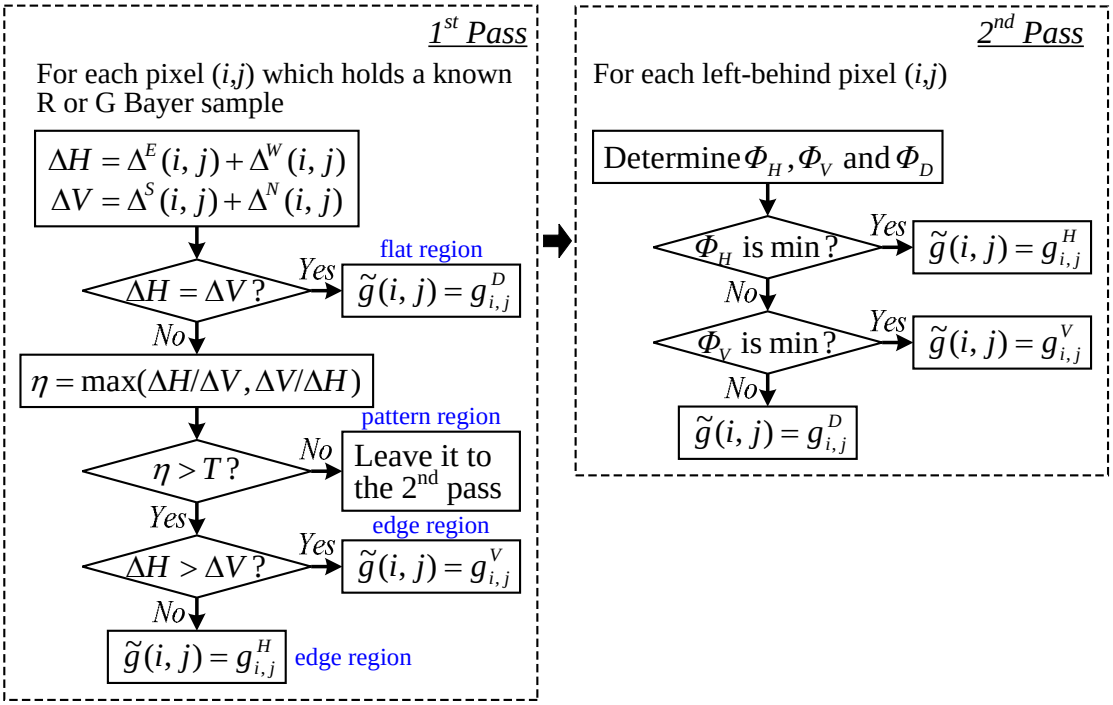


Figure 5.7 Procedures for interpolating a missing green sample in the proposed demosaicing algorithm

In the first pass of the estimation scheme, it raster-scans the Bayer image to classify the local region of each pixel of interest. For this purpose, for each pixel which carries a R or B Bayer sample as the pixel (*i*,*j*) shown in Figures 5.2c or 5.2d, three parameters are defined based on IGs as follows.

$$\Delta H = \Delta^E(i,j) + \Delta^W(i,j) \tag{5.16}$$

$$\Delta V = \Delta^S(i,j) + \Delta^N(i,j) \tag{5.17}$$

$$\eta = \max(\Delta V / \Delta H, \Delta H / \Delta V) \tag{5.18}$$

The classification criteria shown in Figure 5.7 are then used in the proposed scheme to classify a region into a flat, edge or pattern region.

As soon as a pixel's local region is classified, the preliminary estimate of its missing green sample, say $\tilde{g}(i, j)$, is interpolated with an appropriate interpolator as follows unless pixel (i,j) is classified to be in a pattern region.

$$\tilde{g}(i, j) = \begin{cases} g_{i,j}^{D} & \text{if } \Delta H = \Delta V \\ g_{i,j}^{H} & \text{if } \Delta H < \Delta V \text{ and } \eta > T \\ g_{i,j}^{V} & \text{if } \Delta H > \Delta V \text{ and } \eta > T \end{cases} \tag{5.19}$$

where T is a predefined threshold to be discussed later. Those pixels in pattern regions will be handled in pass 2.

The above estimation process is carried out at each pixel which does not have a green Bayer sample. At the end of pass 1, preliminary estimates of all the missing green samples in flat or edge regions are interpolated.

The second pass handles all those pixels left behind after the first pass. By considering that the information available in IGs is not rich enough for one to decide the interpolation direction in the first pass, extra CD information is exploited in the second pass.

Though one may use the CD estimates already obtained in IG extraction to save the effort, a re-estimation using the green estimates obtained in pass 1 is performed in our realization as these green estimates are generally more accurate and hence can guide one to have a better decision.

Assume that the green sample of pixel (i,j) shown in Figure 5.1c cannot be determined in pass 1. In pass 2, the CD estimates of pixel $(m,n) \in \{(i,j \pm 2t),(i \pm 2t,j) \mid t=0,1,2,\ldots L\}$ are re-evaluated as

$$\rho_{GR}^{k}(m,n)=\begin{cases}\tilde{g}(m,n)-R(m,n) & \text{if } \tilde{g}(m,n) \text{ was determined in pass 1}\\ g_{m,n}^{k}-R(m,n) & \text{otherwise}\end{cases}$$

$$\text{for } k\in\{H,V,D\} \qquad (5.20)$$

where $\rho_{GR}^{H}(m,n)$, $\rho_{GR}^{V}(m,n)$ and $\rho_{GR}^{D}(m,n)$ are, respectively, the green-to-red CD estimates of pixel (*m*,*n*) used to estimate the CD variation along the horizontal, the vertical and the diagonal axes passing pixel (*i*,*j*). Note that $g_{m,n}^{H}$, $g_{m,n}^{V}$ and $g_{m,n}^{D}$ are obtained with eqns. (5.13), (5.14) and (5.15) respectively. Parameter *L* determines the number of pixels involved in the estimation of the CD variation along an axis. In particular, the extent of CD variation along the three axes are measured by

$$\Phi_H=\sum_{t=-L}^{L}\left|\rho_{GR}^{H}(i,j)-\rho_{GR}^{H}(i,j+2t)\right|$$

$$\Phi_V=\sum_{t=-L}^{L}\left|\rho_{GR}^{V}(i,j)-\rho_{GR}^{V}(i+2t,j)\right|$$

and
$$\Phi_D=\frac{1}{2}\sum_{t=-L}^{L}\left(\left|\rho_{GR}^{D}(i,j)-\rho_{GR}^{D}(i,j+2t)\right|+\left|\rho_{GR}^{D}(i,j)-\rho_{GR}^{D}(i+2t,j)\right|\right) \qquad (5.21)$$

Note that 1-norm instead of 2-norm is used here to save the computation effort.

Based on the fact that CDs are generally locally constant in pattern areas, the missing green sample of pixel (*i*,*j*) can be estimated by using the interpolator whose associated direction provides the minimum CD variation.

$$\tilde{g}(i,j)=g_{i,j}^{Z} \quad \text{where } z=\arg\min_{k\in\{H,V,D\}}(\Phi_k) \qquad (5.22)$$

For the case that pixel (*i*,*j*) contains a blue Bayer sample as shown in Figure 5.2d,

one can interchange the roles of red samples and blue samples and then follow the same procedures as mentioned before to determine its green estimate $\tilde{g}(i, j)$.

At the end of pass 2, a complete demosaiced green plane is obtained. Figure 5.1c shows the spatial arrangement of the available color samples in the processing image after green plane interpolation.

An empirical study was carried out to investigate the impact of the threshold T used in eqn. (5.19) and the parameter L used in eqn. (5.21) to the demosaicing performance of the proposed algorithm. In the study, 24 full-color images shown in Appendix A were sub-sampled according to the Bayer pattern to form a set of testing CFA images, and the performance is measured in terms of the CPSNR defined in eqn. (3.24). Figure 5.8 reports the connection between their values and the CPSNR performance of the proposed algorithm. Based on this study result, the settings of T=1.7 and L=3 are selected. All the simulation results reported in this thesis are obtained with these settings unless other arrangements are specified.

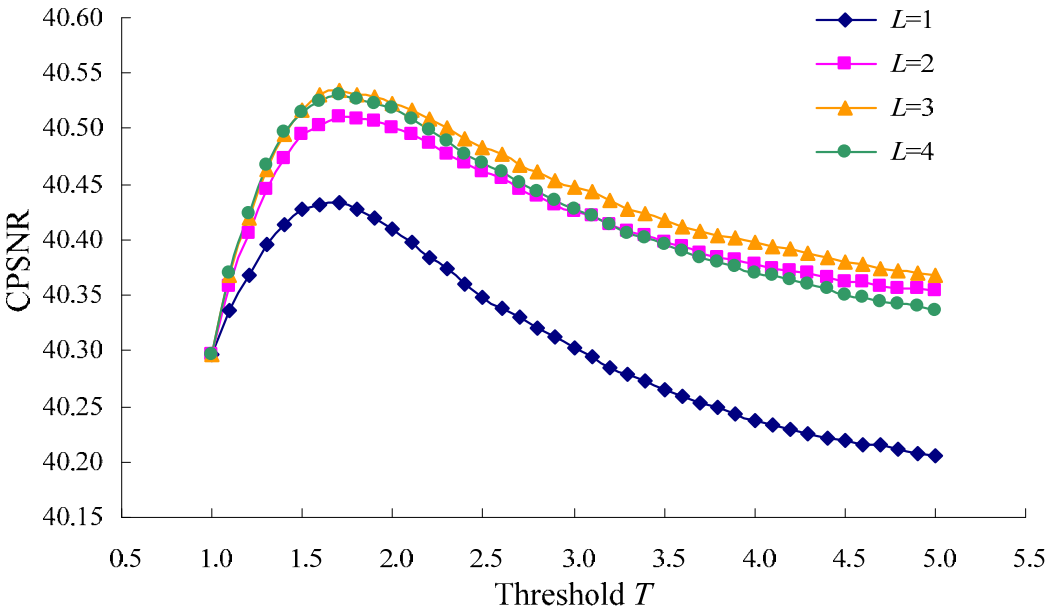


Figure 5.8 CPSNR performances of the proposed demosaicing algorithm under different settings of threshold T and parameter L

5.3.2 Green Plane Enhancement

With the fully populated green plane, the proposed demosaicing algorithm then enhances the demosaiced green samples prior to the interpolation of the red and the blue planes. Since the high frequency proportion of a CD signal is generally weak [26], the proposed enhancement scheme is carried out in the CD domain to produce more accurate green estimates.

Without losing generality, the case of enhancing the demosaiced green sample at a red Bayer sample position, as depicted in Figure 5.2c, is described in this paper. As for the case shown in Figure 5.2d, one can exchange the role of the red and the blue samples and perform the same treatment used in this case to achieve the goal.

After green plane interpolation, $\tilde{g}(i, j)$ is available for all pixels without green Bayer samples and it is the 'best' green estimate so far. Based on the idea that a better temporary green estimate can be used to derive a better CD estimate and hence a better interpolation result, in green plane enhancement the CD estimate of a pixel is re-evaluated with $\tilde{g}(i, j)$. For example, for the pixel (*i*,*j*) shown in Figure 5.2c, its G-R CD is re-evaluated as

$$\bar{d}_{GR}(i, j) = \tilde{g}(i, j) - R(i, j) \qquad \text{for } (i,j) \text{ carrying R Bayer sample.} \qquad (5.23)$$

After re-evaluating the G-R CDs of all pixels carrying R Bayer samples, the CD estimate of pixel (*i*,*j*) is further adjusted to be $\hat{d}_{GR}(i, j)$ by fusing $\bar{d}_{GR}(i, j)$ with $\tilde{d}_{GR}(i, j)$, an interpolation result based on $\bar{d}_{GR}(i, j \pm 2)$ and $\bar{d}_{GR}(i \pm 2, j)$, as follows.

$$\hat{d}_{GR}(i,j) = \beta \bar{d}_{GR}(i,j) + (1-\beta)\tilde{d}_{GR}(i,j) \quad \text{for } (i,j) \text{ carrying R Bayer sample} \quad (5.24)$$

where β is a weighting factor controlling the fusion. In formulation, $\tilde{d}_{GR}(i,j)$ is defined as

$$\tilde{d}_{GR}(i,j) = \frac{w^E \bar{d}_{GR}(i,j+2) + w^W \bar{d}_{GR}(i,j-2) + w^S \bar{d}_{GR}(i+2,j) + w^N \bar{d}_{GR}(i-2,j)}{w^E + w^W + w^S + w^N}$$

for (i,j) carrying R Bayer sample (5.25)

where $w^k = 1/\Delta^k(i,j)$ for $k \in \{E,W,S,N\}$. Since a large value of $\Delta^k(i,j)$ implies that there is a great change of either CD or CI in the corresponding direction, the weighting mechanism in eqn. (5.25) automatically directs the interpolation of $\tilde{d}_{GR}(i,j)$ along an edge when there is.

Parameter β can be determined off-line by linear regression. In our study, a set of training images obtained by sub-sampling half of the images shown in Appendix A according to the Bayer pattern were used in the training process. For the simulation results reported in this chapter, β is selected to be 0.33.

For information, Figure 5.9 shows the average CPSNR performance of the proposed demosaicing algorithm under different values of β when the testing images in the non-training set are demosaiced. It is observed that the CPSNR attains its maximum when the β value is in between 3.0 and 3.5. This result supports our choice of the value of β in this section.

With the adjusted $\hat{d}_{GR}(i,j)$, the demosaiced green sample is updated to be

$$\hat{g}(i,j) = \hat{d}_{GR}(i,j) + R(i,j) \quad \text{for } (i,j) \text{ carrying R Bayer sample.} \quad (5.26)$$

Similar procedures can be performed with the same parameter β to enhance the demosaiced green samples at pixels carrying blue Bayer samples. At the end of the enhancement, all $\tilde{g}(i, j)$ are updated and finalized to be $\hat{g}(i, j)$ as shown in Figure 5.1d.

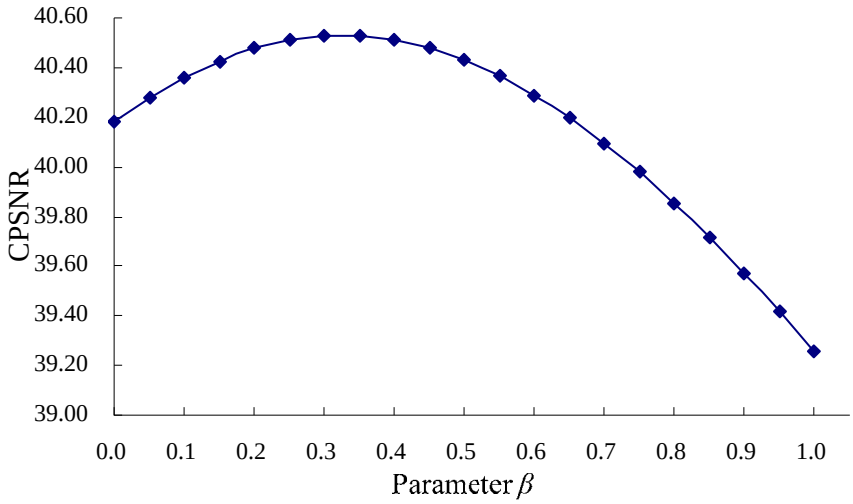


Figure 5.9 Average CPSNR performance of the proposed demosaicing algorithm for the testing images in the non-training set versus the value of parameter β

5.3.3 Red Plane and Blue Plane Interpolations

Based on the enhanced green plane produced in green plane enhancement, a partial G-R (G-B) plane which covers all pixels carrying red (blue) Bayer samples can be constructed. In the proposed demosaicing algorithm, with the help of IGs, an interpolation is carried out in the partial G-R (G-B) plane to derive a full size G-R (G-B) plane. The missing red (blue) samples can then be derived from the G-R (G-B) plane and the enhanced green plane.

Again, without losing generality, only the red plane interpolation is described here. The blue plane interpolation can be achieved in the same manner by interchanging the role of red and blue samples.

During green plane enhancement, the $\hat{d}_{GR}(i, j)$ for pixels carrying R Bayer samples are evaluated with eqn. (5.24) and they are ready to form a partial G-R plane. The complete G-R plane is interpolated with this partial plane through two steps. In step 1, the $\hat{d}_{GR}(i, j)$ for pixels carrying B Bayer samples are first interpolated.

Consider the pixel (*i*,*j*) shown in Figure 5.2d. Its four diagonal neighbors carry R Bayer samples and hence $\hat{d}_{GR}(i \pm 1, j \pm 1)$ are known. Accordingly, $\hat{d}_{GR}(i, j)$ can be interpolated with

$$\hat{d}_{GR}(i, j) = \frac{w^{NW}\hat{d}_{GR}(i-1, j-1) + w^{NE}\hat{d}_{GR}(i-1, j+1) + w^{SE}\hat{d}_{GR}(i+1, j+1) + w^{SW}\hat{d}_{GR}(i+1, j-1)}{w^{NW} + w^{NE} + w^{SE} + w^{SW}}$$

for (*i*,*j*) carrying B Bayer sample (5.27)

where $w^{k_1 k_2} = \frac{1}{\Delta^{k_1}(i, j) + \Delta^{k_2}(i, j)}$ for $k_1 \in \{S,N\}$ and $k_2 \in \{E,W\}$. (5.28)

Theoretically, the weight $w^{k_1 k_2}$ should be derived from the 2-norm of two perpendicular IGs ($\Delta^{k_1}(i,j)$ and $\Delta^{k_2}(i,j)$). However, our empirical study shows that the improvement over 1-norm is insignificant for almost all the testing images.

To complete the G-R plane, the G-R CDs for pixels carrying G Bayer samples are interpolated in step 2. For the pixel (*i*,*j*) shown in Figures 5.2a or 5.2b, $\hat{d}_{GR}(i,j)$ is interpolated with

$$\hat{d}_{GR}(i,j) = \frac{w^E \hat{d}_{GR}(i,j+1) + w^W \hat{d}_{GR}(i,j-1) + w^S \hat{d}_{GR}(i+1,j) + w^N \hat{d}_{GR}(i-1,j)}{w^E + w^W + w^S + w^N}$$

for (*i*,*j*) carrying G Bayer sample (5.29)

where $w^k = 1/\Delta^k(i,j)$ for $k \in \{E,W,S,N\}$.

With the complete G-R plane and the enhanced green plane, all missing red samples can be estimated by

$$\hat{r}(i,j) = \begin{cases} G(i,j) - \hat{d}_{GR}(i,j) & \text{if } G(i,j) \text{ exists} \\ \hat{g}(i,j) - \hat{d}_{GR}(i,j) & \text{if } B(i,j) \text{ exists} \end{cases}. \qquad (5.30)$$

As mentioned, the blue plane interpolation is performed in the same way by exchanging the role of red and blue samples. The interpolations of the two color planes are independent and they can be carried out in parallel. Figure 5.1e shows the spatial arrangement of the intermediate red and blue planes obtained after step 1 and Figure 5.1f shows that of the final reconstructed full-color image.

5.4 Simulation Results

Simulations were carried out to evaluate the performance of the proposed demosaicing algorithm. Some other state-of-art demosaicing algorithms such as AP [12], AF [40], DLMSE [41], AHDA [35], PCSD [36], DFPD [37] and HPHD [38] as well as the color difference variance-based demosaicing algorithm (VCD), presented previously in Chapter 3, were evaluated for comparisons. The 24 digital full-color testing images displayed in Appendix A were used in the simulations. These full-color images were first sub-sampled according to the Bayer pattern to form a set of testing Bayer images. They were then reconstructed to full-color images by the evaluated demosaicing algorithms. Whenever there is a post-processing scheme recommended by the authors of a particular demosaicing algorithm to enhance its demosaicing results, the scheme was performed in the simulation as suggested. For reference and clear presentation, the proposed IG-based demosaicing algorithm is referred to as IGCD hereafter in this thesis.

Similar to the previous chapters, the color peak signal-to-noise ratio (CPSNR) and the S-CIELAB [66] were exploited to measure the quality of the demosaiced images. One can refer to eqn. (3.24) in Chapter 3 for the definition of CPSNR.

Tables 5.1 and 5.2 respectively list the CPSNR and the S-CIELAB measures of various demosaicing algorithms. As mentioned, recommended post-processing was performed to improve the demosaicing performance of HPHD, AHDA, DFPD and VCD. One can see from the tables that the proposed algorithm outperforms the other demosaicing algorithms for the majority of the testing images. For the images which contain many fine structures such as Image 1, 6, 9 and 13, the proposed algorithm provides significant performance improvement over the others. As an example, for Image 1 in which the fine stone structure is almost everywhere, the proposed

algorithm provides a CPSNR of 39.96dB which is around 1dB higher than the second best CPSNR performance provided by other evaluated algorithms in the simulation. Consistent performance improvement can also be found when the S-CIELAB measure is used. These results demonstrate that the proposed demosaicing algorithm is robust to the input when recovering a full-color image from a Bayer image.

Figure 5.10 shows part of the demosaicing results of Image 19 for visual comparison. It can be observed that the proposed demosaicing algorithm produces the best perceptual result as compared with the other evaluated algorithms. Especially in the area near the boundary of the fence and the grass, the proposed algorithm can not only preserve the fine fence details but also produce a result with almost invisible color artifacts. Though some other sophisticated algorithms such as AHDA, DLMSE and HPHD can also reproduce the fence details, noticeable color artifacts can be found in their demosaiced results. Figures 5.11 and 5.12 show the demosaicing results of two other testing images and similar observation can be made.

Image	AP [12]	AF [40]	DLMSE [41]	AHDA [35]	PCSD [36]	DFPD [37]	HPHD [38]	VCD (Ours)	IGCD (Ours)
1	37.70	37.44	38.41	35.16	36.32	36.56	39.00	38.53	39.96
2	39.57	40.63	40.85	39.19	39.98	40.40	40.96	40.43	40.99
3	41.45	42.52	42.56	41.59	41.82	41.95	43.01	42.54	43.26
4	40.03	40.42	40.44	38.94	39.52	39.79	40.89	40.50	40.56
5	37.46	37.91	37.98	35.74	37.15	37.18	38.61	37.89	38.31
6	38.50	37.87	40.11	37.57	38.72	38.93	40.53	40.03	41.00
7	41.77	42.83	42.32	40.92	41.51	41.80	43.01	42.15	42.64
8	35.08	35.10	35.97	33.77	34.39	34.94	36.94	36.41	37.35
9	41.72	42.62	42.98	41.09	42.03	42.27	42.90	43.04	43.42
10	42.02	42.61	42.56	40.71	41.74	42.02	42.56	42.51	42.83
11	39.14	39.17	39.94	37.53	38.52	38.83	40.51	39.86	40.66
12	42.51	42.60	43.38	41.75	42.63	42.84	43.88	43.45	44.13
13	34.30	33.66	34.71	31.52	32.64	32.81	35.32	34.90	36.03
14	35.60	36.93	36.79	35.49	35.69	36.36	37.48	36.88	37.10
15	39.35	39.78	39.80	38.03	38.93	39.20	39.81	39.78	39.84
16	41.76	40.97	43.67	41.40	42.55	42.72	44.08	43.64	44.47
17	41.11	41.14	41.58	39.42	40.40	40.37	41.60	41.21	41.77
18	37.45	37.38	37.75	35.31	36.23	36.40	38.02	37.49	37.96
19	39.46	40.01	40.98	38.48	39.48	39.74	41.35	41.00	41.79
20	40.66	41.08	41.21	39.27	40.02	40.05	41.68	41.07	41.71
21	38.66	38.55	39.03	36.55	37.27	37.47	39.60	39.12	39.99
22	37.55	38.32	38.29	36.51	37.13	37.52	38.43	37.97	38.48
23	41.88	42.99	43.16	41.85	42.21	42.50	43.10	42.89	43.20
24	34.78	34.88	35.56	33.64	34.38	34.55	35.25	35.04	35.39
Avg.	**39.15**	**39.48**	**40.00**	**37.98**	**38.80**	**39.05**	**40.36**	**39.93**	**40.54**

Table 5.1 CPSNR performance (in dB) of various demosaicing algorithms

Image	AP [12]	AF [40]	DLMSE [41]	AHDA [35]	PCSD [36]	DFPD [37]	HPHD [38]	VCD (Ours)	IGCD (Ours)
1	1.6460	1.6328	1.4552	1.6186	1.7422	1.8101	1.4301	1.4630	1.2747
2	1.6639	1.5218	1.4262	1.5995	1.5951	1.5866	1.4445	1.4985	1.4019
3	0.9509	0.8828	0.8699	0.9321	0.9325	0.9334	0.8444	0.8802	0.8270
4	1.2863	1.2195	1.1836	1.3219	1.3367	1.3479	1.1990	1.2156	1.1737
5	2.1668	1.9992	1.9906	2.2099	2.1340	2.2147	1.8585	2.0313	1.8815
6	1.2447	1.2805	1.0472	1.1364	1.1426	1.1871	1.0164	1.0627	0.9682
7	1.1062	0.9704	1.0194	1.1284	1.0871	1.0715	0.9643	1.0329	0.9827
8	1.8668	1.7597	1.6003	1.7335	1.8194	1.8824	1.5243	1.5865	1.4305
9	0.8411	0.7566	0.7408	0.8182	0.7873	0.7930	0.7743	0.7418	0.7274
10	0.8263	0.7597	0.7623	0.8424	0.8134	0.8241	0.7853	0.7727	0.7519
11	1.4742	1.4182	1.3053	1.4281	1.4657	1.4986	1.2759	1.3302	1.2041
12	0.6765	0.6497	0.6071	0.6586	0.6532	0.6605	0.5951	0.6143	0.5782
13	2.5824	2.7044	2.4334	2.7502	2.9799	3.0769	2.3441	2.4453	2.1993
14	1.9443	1.7917	1.7151	1.8750	1.9254	1.9172	1.6292	1.7484	1.6392
15	1.4286	1.3302	1.3121	1.4643	1.4622	1.4475	1.3287	1.3372	1.2963
16	1.0303	1.0776	0.8515	0.9184	0.9249	0.9567	0.8411	0.8710	0.8107
17	1.3292	1.2973	1.2585	1.3952	1.3991	1.4382	1.3023	1.3076	1.2590
18	2.1841	2.0626	2.0821	2.3947	2.3639	2.3462	2.0613	2.1318	2.0837
19	1.2877	1.1971	1.1142	1.2723	1.2928	1.3143	1.1420	1.1439	1.0715
20	1.0030	0.9563	0.9451	1.0316	1.0506	1.0796	0.9158	0.9633	0.8974
21	1.3281	1.3302	1.2510	1.3669	1.4471	1.4995	1.2028	1.2616	1.1567
22	1.4973	1.3690	1.3974	1.5933	1.5594	1.5298	1.3698	1.4495	1.3977
23	0.9500	0.8773	0.8737	0.9676	0.9647	0.9438	0.8911	0.9183	0.8869
24	1.4378	1.3669	1.3300	1.4855	1.4697	1.5032	1.3165	1.3628	1.3074
Avg.	**1.4064**	**1.3421**	**1.2738**	**1.4143**	**1.4312**	**1.4526**	**1.2524**	**1.2988**	**1.2170**

Table 5.2 S-CIELab color difference performance of various demosaicing algorithms

(a) Origin

(b) AP

(c) AHDA

(d) AF

(e) PCSD

(f) DLMSE

(g) VCD

(h) HPHD

(i) DFPD

(j) IGCD

Figure 5.10 Part of the demosaicing results of Image 19 produced by various demosaicing algorithms

(a) Origin

(b) AP

(c) AHDA

(d) AF

(e) PCSD

(f) DLMSE

(g) VCD

(h) HPHD

(i) DFPD

(j) IGCD

Figure 5.11 Part of the demosaicing results of Image 8 produced by various demosaicing algorithms

(a) Origin

(b) AP

(c) AHDA

(d) AF

(e) PCSD

(f) DLMSE

(g) VCD

(h) HPHD

(i) DFPD

(j) IGCD

Figure 5.12 Part of the demosaicing results of Image 1 produced by various demosaicing algorithms

5.5 Computational Complexity

This section reports the computational complexity of the proposed demosaicing algorithm in terms of number of additions (ADD), multiplications (MUL), bit-shifts (SHT) and absolute-value-taking operations (ABS). A comparison operation is considered as an addition in our report.

Table 5.3 shows the number of arithmetic operations required per involved pixel in different stages of the proposed demosaicing algorithm. In this Table, Ω_R, Ω_G and Ω_B denote the set of pixels carrying R, G and B Bayer samples respectively.

Operational Step	ADD	MUL	SHT	ABS
Extraction of IGs for (i,j)$\in\{\Omega_R,\Omega_G,\Omega_B\}$	16	0	6	4
G plane interpolation for (i,j)$\in\{\Omega_R,\Omega_B\}$				
• If (i,j) is in flat regions	12	0	5	0
• If (i,j) is in edge regions	14	1	5	0
• If (i,j) is in pattern regions	52	1	6	12
G plane enhancement for (i,j)$\in\{\Omega_R,\Omega_B\}$	9	9	0	0
Interpolating B samples for (i,j)$\in\{\Omega_R\}$	11	9	0	0
Interpolating R samples for (i,j)$\in\{\Omega_B\}$	11	9	0	0
Interpolating R/B samples for (i,j)$\in\{\Omega_G\}$	14	12	0	0

Table 5.3 Number of arithmetic operations required per involved pixel in the proposed demosaicing algorithm

The computational complexity of the other evaluated demosaicing algorithms is tabulated in Table 5.4 for comparison. The complexity figures for AP, AF, AHDA and PCSD are directly taken from Lian's paper [40]. For VCD, DFPD [37] and HPHD [38], the figures are, respectively, extracted from their corresponding papers. As for DLMSE [41], its complexity is derived based on its realization presented in [41]. A comparison operation is considered as an addition in our derivation.

	ADD	MUL	SHT	ABS	Total
AP	384	384	-	-	**768**
AF	40.5	10.5	11	3	**54**
DLMSE	72.5	36	5.5	-	**114**
AHDA	184	12	-	49	**245**
PCSD	149	60	5	8	**222**
DFPD	28.5	0.5	6	1	**36**
HPHD	176.5-207	81-86	11-18	20.5-31	**289-342**
VCD	48-92	16.5-36	4-7	4	**72.5-139**
IGCD	39-59	15-15.5	8.5-9	4-10	**66.5-93.5**

Table 5.4 Complexity of various demosaicing algorithms in terms of number of arithmetic operations per pixel

From Table 5.4, one can see that the proposed demosaicing algorithm requires the least computation effort in the four demosaicing algorithms providing the best CPSNR and S-CIELAB performance. In the worst case where all red and blue Bayer samples are in pattern regions, the proposed algorithm requires totally at most 93.5 operations to reconstruct a color pixel on average, which is at least 67% lower than that required by HPHD, the second best in terms of CPSNR and S-CIELAB. Although the complexity of AF and DFPD is lower than that of the proposed algorithm, their output quality is low as shown in Figs. 9 and 10. It reveals that the cost performance of the proposed algorithm is high.

In our simulations, the average number of arithmetic operations is around 80 per pixel. The execution time for the proposed demosaicing algorithm to process a Bayer image of size 768x512 on a 3.4GHz Pentium 4 PC with 1024MB RAM is 0.0948s on average.

5.6 Chapter Summary

In this chapter, a new edge-sensing measure called integrated gradient is proposed. This measure effectively extracts gradient information from a Bayer image in both color intensity and color difference domains, and consequently provides reliable and complete information for one to interpolate missing samples in a Bayer image along an appropriate direction.

An efficient decision-based demosaicing algorithm is then developed. Under the guidance of the same integrated gradients, the proposed demosaicing algorithm interpolates different color planes in different stages. Though the algorithm updates the green plane and the corresponding color difference planes in the course to provide better references for interpolation, computationally expensive re-estimation of local gradients based on intermediate interpolation results is avoided. It guarantees the consistency of the interpolation direction in different color channels and saves the effort required to repeatedly extract gradient information from intermediate interpolation results at different stages.

Unlike some other demosaicing algorithms which carry out a post-processing step to enhance all color planes at the end, the proposed demosaicing algorithm enhances the green plane before the interpolation of the red and blue planes based on the integrated gradients. By so doing, it provides a better reference for one to interpolate the red and blue planes. This automatically improves the quality of the resultant red and blue planes, and hence eliminates the necessity of another enhancement step for the red and blue planes after their interpolation.

Simulation results confirmed that the proposed demosaicing algorithm outperforms state-of-art demosaicing algorithms in terms of output quality at a complexity of around 80 arithmetic operations per pixel.

Chapter 6 A Lossless Compression Scheme for Bayer Color Filter Array Images

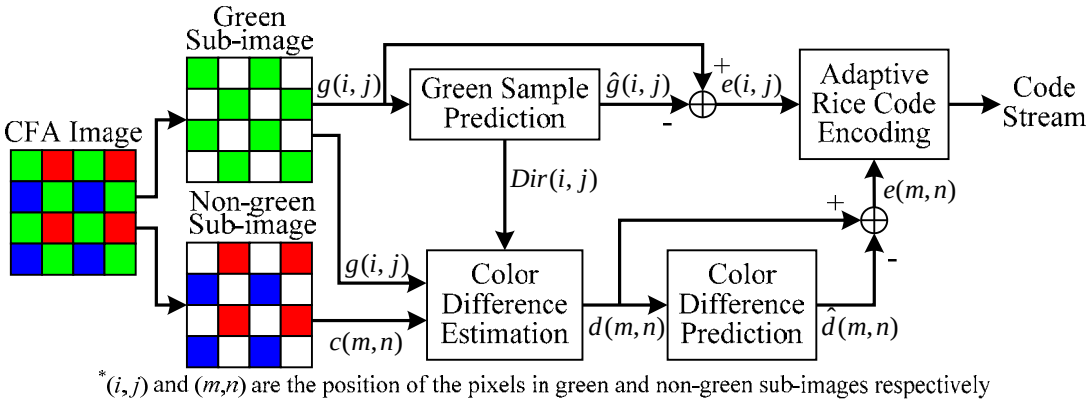
6.1 Introduction

In this chapter, a prediction-based lossless CFA image compression scheme is proposed. Figure 6.1 shows the structure of the proposed scheme. This scheme divides a CFA image into two sub-images: a green sub-image which contains all green samples of the CFA image and a non-green sub-image which holds the red and the blue samples. The green sub-image is coded first and the non-green sub-image follows based on the green sub-image as a reference. To reduce the spectral redundancy, the non-green sub-image is processed in the color difference domain. In contrast, the green sub-image is processed in the intensity domain such that its processing result is used as a reference for coding the color difference content of the non-green sub-image. Both the sub-images are processed in raster-scan sequence with our proposed context matching-based prediction technique to remove the spatial dependency. The prediction residue planes of the two sub-images are then entropy encoded sequentially with our proposed realization scheme of adaptive Rice code.

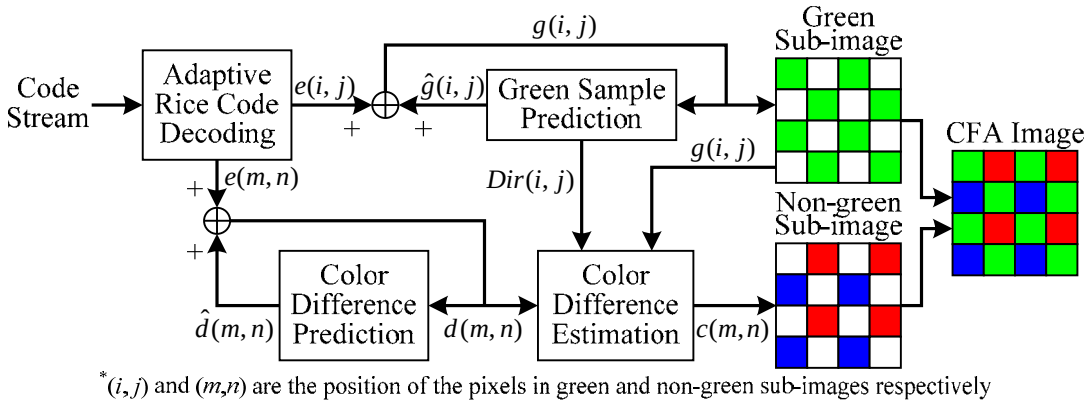
Experimental results show that the proposed compression scheme can effectively and efficiently reduce the redundancy in both spatial and color spectral domains. As compared with the existing lossless CFA image coding schemes such as [14-16], the proposed scheme provides the best compression performance in our simulation study.

This chapter is structured as follows. The proposed context matching-based prediction technique is presented in Section 6.2. Section 6.3 shows how to estimate a missing green sample in the non-green sub-image of a CFA image for extracting the color difference information when compressing the non-green sub-image. In Section

6.4, how the prediction residue is adaptively encoded with Rice Code is provided. Section 6.5 demonstrates some simulation results and, finally, a conclusion is given in Section 6.6.



(a) Encoder



(b) Decoder

Figure 6.1 Structure of the proposed compression scheme: (a) encoder and (b) decoder

6.2 Context Matching-Based Prediction

The proposed prediction technique handles the green plane and the non-green plane separately in a raster-scan manner. It weights the neighboring samples such that the one has higher context similarity to that of the current sample contributes more to the current prediction. Accordingly, this prediction technique is referred to as *context*

matching-based prediction (CMBP) in this chapter.

The green plane (green sub-image) is handled first as a CFA image contains double number of green samples to that of red/blue samples and the correlation among green samples can be exploited easily as compared with that among red or blue samples. Accordingly, the green plane can be used as a good reference to estimate the color difference of a red or blue sample when handling the non-green plane (non-green sub-image).

6.2.1 Prediction on the green plane

As the green plane is raster-scanned during the prediction and all prediction errors are recorded, all processed green samples are known and can be exploited in the prediction of the pixels which have not yet been processed.

Assume that we are now processing a particular green sample $g(i,j)$ as shown in Figure 6.2a. The four nearest processed neighboring green samples of $g(i,j)$ form a candidate set $\Phi_{g(i,j)}=\{g(i,j\text{-}2),g(i\text{-}1,j\text{-}1),g(i\text{-}2,j),g(i\text{-}1,j\text{+}1)\}$. The candidates are ranked by comparing their support regions (i.e. context) with that of $g(i,j)$.

The support region of a green sample at position (p,q), $S_{g(p,q)}$, is defined as shown in Figure 6.3a. In formulation, we have $S_{g(p,q)}=\{(p,q\text{-}2),(p\text{-}1,q\text{-}1),(p\text{-}2,q),(p\text{-}1,q\text{+}1)\}$. The matching extent of the support region of $g(i,j)$ and the support region of $g(m,n)$ for $g(m,n)\in\Phi_{g(i,j)}$ is then measured by

$$D(S_{g(i,j)},S_{g(m,n)}) = \left|g_{(i,j-2)} - g_{(m,n-2)}\right| + \left|g_{(i-1,j-1)} - g_{(m-1,n-1)}\right| + \left|g_{(i-2,j)} - g_{(m-2,n)}\right| + \left|g_{(i-1,j+1)} - g_{(m-1,n+1)}\right|. \quad (6.1)$$

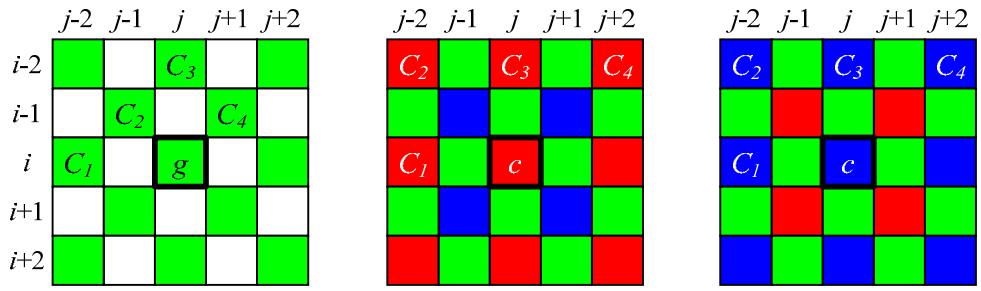
Though a higher order distance such as Euclidian distance can be used instead of

eqn. (6.1) to achieve a better matching performance, we found in our simulations that the improvement was not significant enough to compensate for its high realization complexity.

Let $g(m_k,n_k)\in\Phi_{g(i,j)}$ for k=1,2,3,4 be the 4 ranked candidates of sample $g(i,j)$ such that $D(S_{g(i,j)},S_{g(m_u,n_u)})\le D(S_{g(i,j)},S_{g(m_v,n_v)})$ for $1\le u<v\le 4$. The value of $g(i,j)$ can then be predicted with a prediction filter as

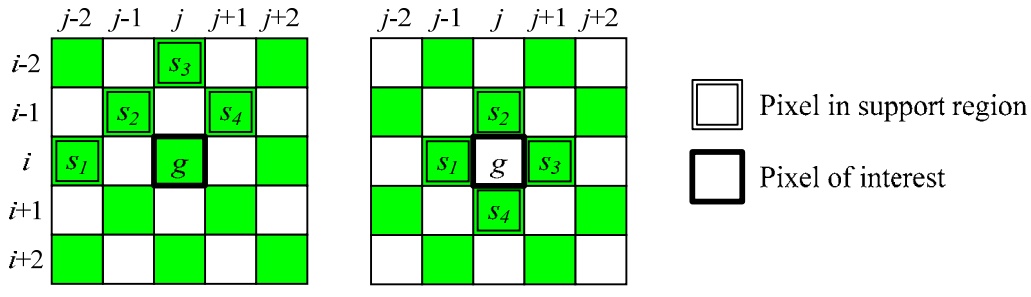
$$\hat{g}(i,j)=round\left(\sum_{k=1}^{4}w_k g(m_k,n_k)\right) \tag{6.2}$$

where w_k for k=1,2,3,4 are normalized weighting coefficients such that $\sum_{k=1}^{4}w_k=1$.



(a) $\Phi_g(i,j)$ (b) $\Phi_c(i,j)$

Figure 6.2 Positions of the pixels included in the candidate set of (a) a green sample and (b) a red/blue sample



(a) $S_{g(i,j)}$ (b) $S_{c(i,j)}$

Figure 6.3 The support region of (a) a green sample and (b) a red/blue sample

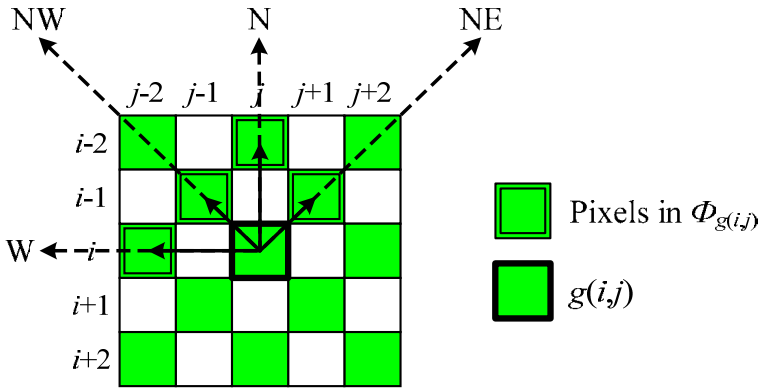


Figure 6.4 The four possible directions associated with a green pixel

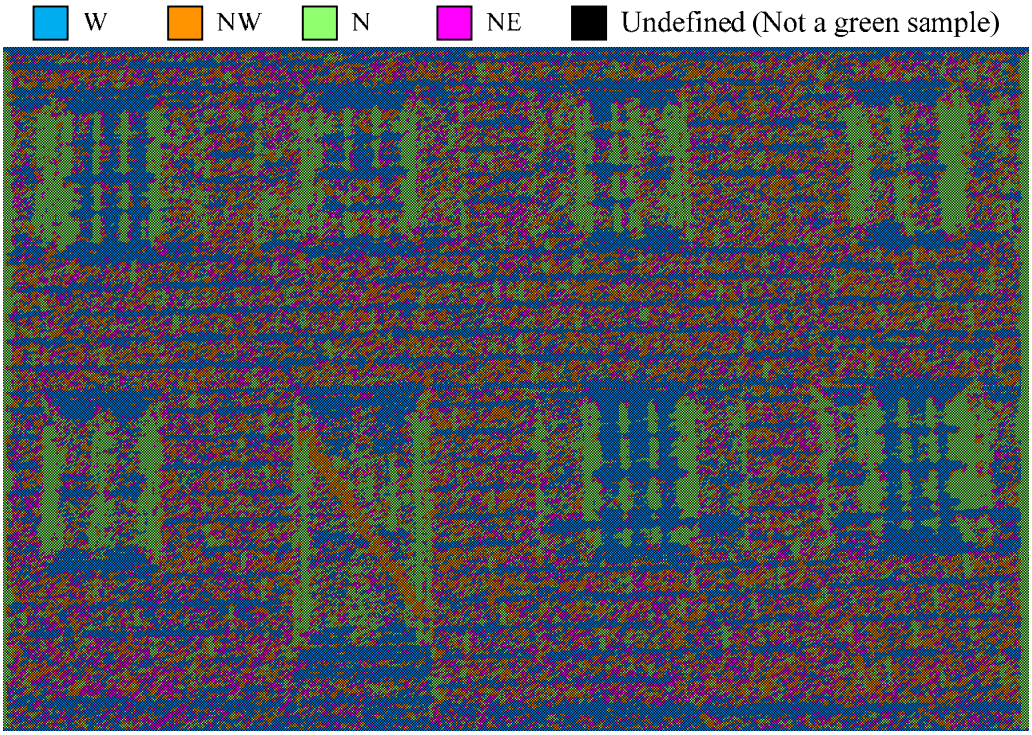


Figure 6.5 The four possible directions associated with a green pixel

Let $Dir(i,j) \in \{W,NW,N,NE\}$ be a direction vector associated with sample $g(i,j)$. It is defined as the direction pointed from sample $g(i,j)$ to $g(i,j)$'s 1st ranked candidate $g(m_1, n_1)$. Figure 6.4 shows its all possible values. This definition applies to all green samples in the green sub-image. As an example, Figure 6.5 shows the direction map of the testing image generated by sub-sampling Image 1 in Appendix A according to Bayer CFA. If the direction of $g(i,j)$ is identical to the directions of all green samples

in $S_{g(i,j)}$, pixel (i,j) will be considered in a homogenous region and $\hat{g}(i, j)$ will then be estimated to be $g(m_1, n_1)$ directly. In formulation, we have

$$\hat{g}(i, j) = g(m_1, n_1) \qquad \text{if } Dir(i,j)=Dir(a,b)\ \forall(a,b)\in S_{g(i,j)} \tag{6.3}$$

which implies $\{w_1,w_2,w_3,w_4\}=\{1,0,0,0\}$. Otherwise, $g(i,j)$ is considered to be in a heterogeneous region and a pre-defined prediction filter is used to estimate $g(i,j)$ with eqn. (6.2) instead.

In our study, w_k are obtained by quantizing the training result derived by linear regression with a set of training images covering half of the testing images. The training images were obtained by sampling the 24 full-color images shown in Appendix A according to Bayer CFA. The quantization is performed in order to reduce the realization effort of eqn. (6.2). After all, when $g(i,j)$ is not in a homogeneous region, the coefficients of the prediction filter used to obtain the result presented in this chapter are given by $\{w_1,w_2,w_3,w_4\}=\{5/8,2/8,1/8,0\}$, which allows the realization of eqn. (6.2) to be achieved with only shift and addition operations as follows.

$$\hat{g}(i, j) = round\left(\frac{4g(m_1, n_1) + g(m_1, n_1) + 2g(m_2, n_2) + g(m_3, n_3)}{8}\right) \tag{6.4}$$

The prediction error is determined with $g(i, j) - \hat{g}(i, j)$. Figure 6.6 summaries how to generate the prediction residue of the green plane of a CFA image.

In CMBP, a green sample is classified according to the homogeneity of its local region to improve the prediction performance. Figure 6.7 shows the effect of this classification step. By comparing Figures 6.7a and 6.7b, one can see that the approach

with classification can handle the edge regions more effectively and more edge details can be eliminated in the corresponding prediction residue planes. Another supporting observation is the stronger de-correlation power of the approach using classification. Figure 6.8 shows the correlation among prediction residues in the green plane of testing image generated with Image 8 under the two different conditions. The correlation of the residues obtained with region classification is lower, which implies that the approach is more effective in data compression. Besides, the entropy of the prediction residues obtained with region classification is also lower. As far as the testing image generated with Image 8 is concerned, their zero-order entropy values are, respectively, 6.195 bpp and 6.039 bpp.

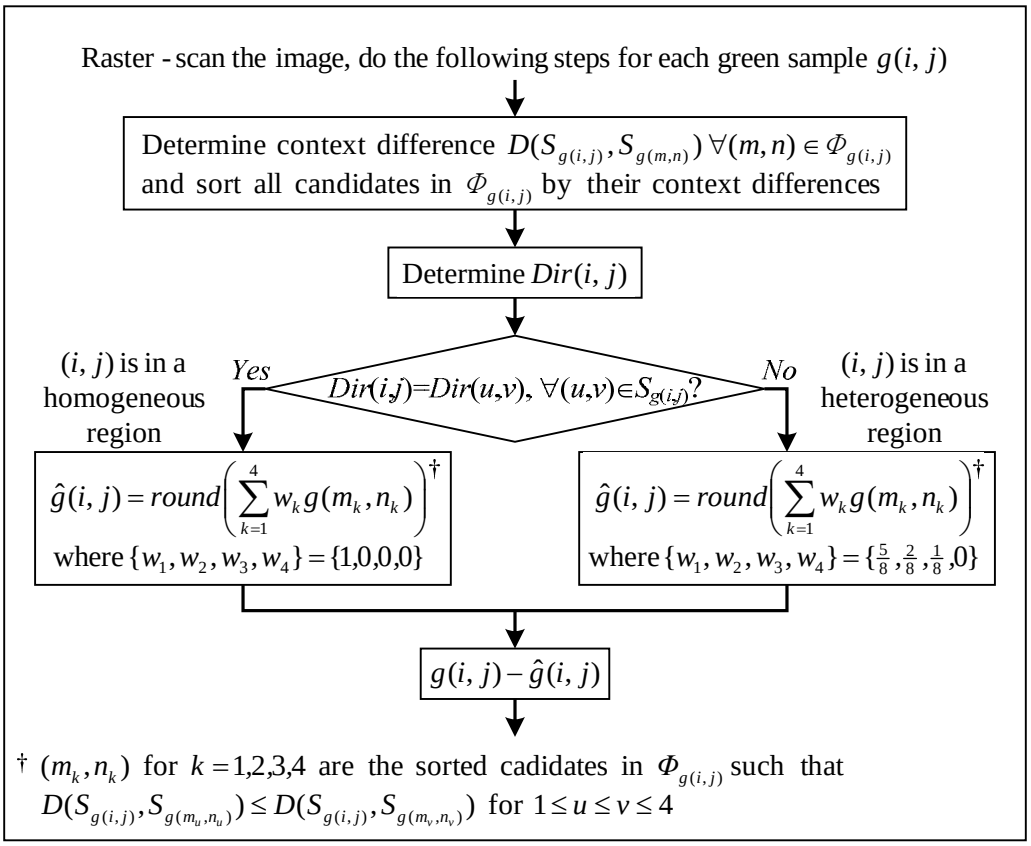
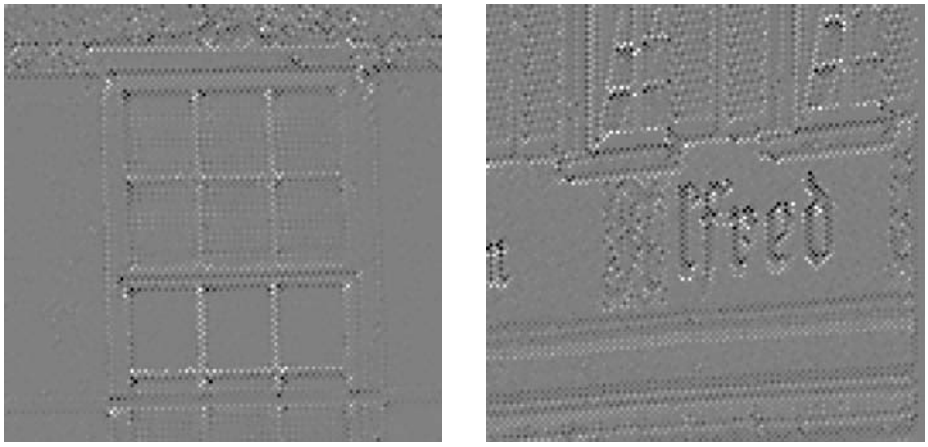


Figure 6.6 Steps to handle the green plane of a CFA image in CMBP

(a)



(b)

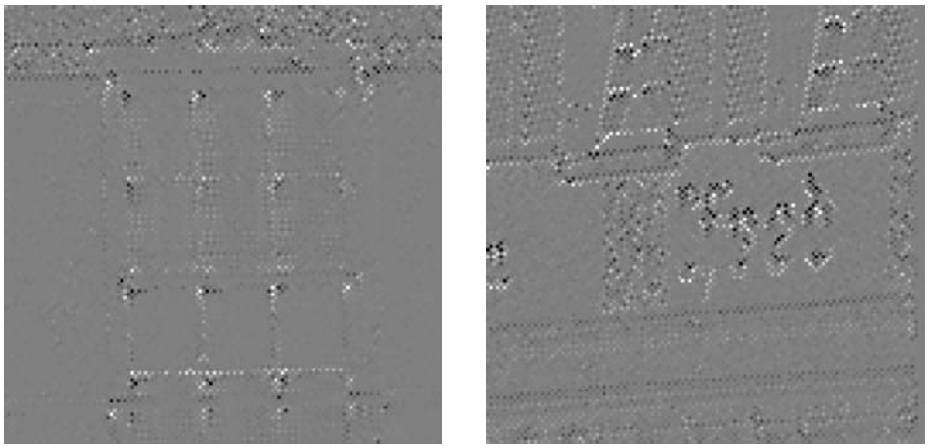
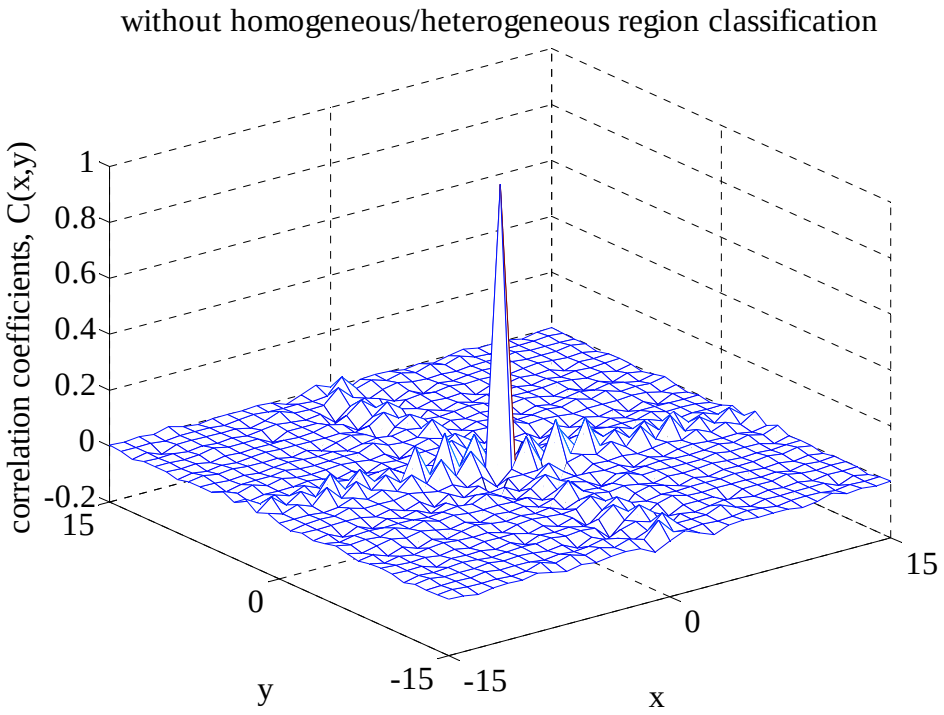
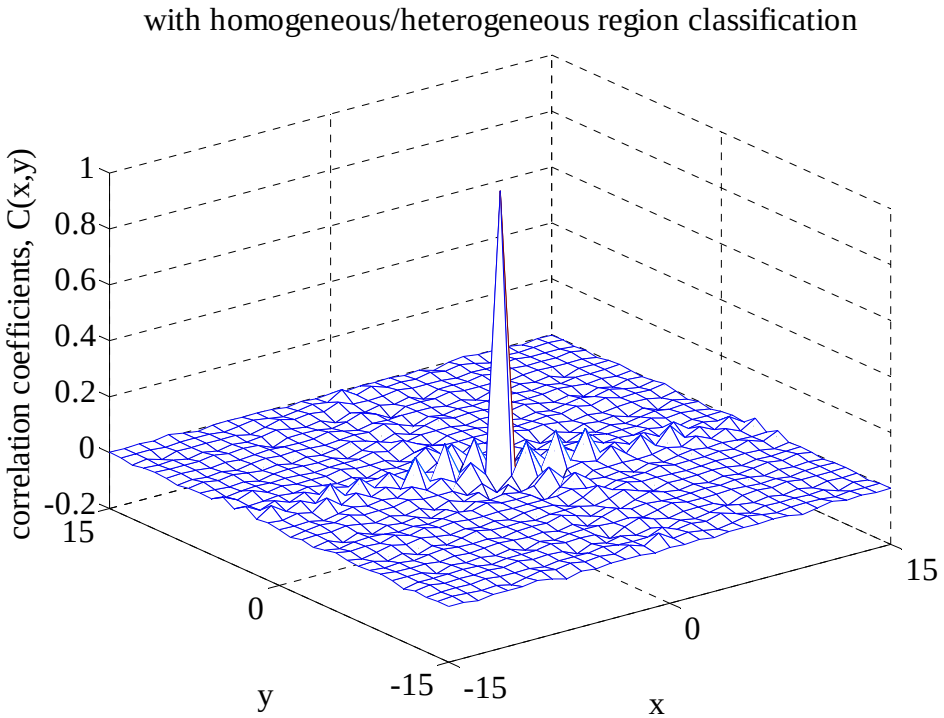


Figure 6.7 Prediction residues of the green planes of testing image Image 1 and 8 (a) without region classification and (b) with region classification



(a)



(b)

Figure 6.8 Correlation among the prediction residues associated with the green sub-image of testing image Image 8: (a) without region classification, and (b) with region classification

6.2.2 Prediction on non-the green plane

As for the case when the sample being processed is a red or blue sample in the non-green plane, the prediction is carried out in the color difference domain instead of the intensity domain as in the green plane. This is done to remove the inter-channel redundancy.

Since the non-green plane is processed after the green plane, all green samples in a CFA image are known and can be exploited when processing the non-green plane. Besides, as the non-green plane is raster-scanned in the prediction, the color difference values of all processed non-green samples in the CFA image should also be known and hence can be exploited when predicting the color difference of a particular non-green sample.

Let $d(p,q)$ be the green-red (or green-blue) color difference value of a non-green sample $c(p,q)$. Its determination will be discussed in detail in Section 6.3. For any non-green sample $c(i,j)$, its candidate set is $\Phi_{c(i,j)}=\{d(i,j-2),d(i-2,j-2),d(i-2,j),d(i-2,j+2)\}$, and its support region (context) is defined as $S_{c(i,j)}=\{(i,j-1),(i-1,j),(i,j+1),(i+1,j)\}$. Figures 6.2b and 6.3b show, respectively, the positions of the pixels involved in the definition of $\Phi_{c(i,j)}$ and $S_{c(i,j)}$.

The prediction for a non-green sample is carried out in the color difference domain. Specifically, the predicted color difference value of sample $c(i,j)$ is given by

$$\hat{d}(i,j) = round\left(\sum_{k=1}^{4} w_k d(m_k, n_k)\right) \tag{6.5}$$

where w_k and $d(m_k,n_k)$ are, respectively, the k^{th} predictor coefficient and the k^{th} ranked candidate in $\Phi_{c(i,j)}$ such that $D(S_{c(i,j)},S_{c(m_u,n_u)}) \le D(S_{c(i,j)},S_{c(m_v,n_v)})$ for $1 \le u < v \le 4$, where

$$D(S_{c(i,j)},S_{c(m,n)}) = |g(i,j-1)-g(m,n-1)| + |g(i,j+1)-g(m,n+1)| + |g(i-1,j)-g(m-1,n)| + |g(i+1,j)-g(m+1,n)|. \quad (6.6)$$

In the prediction carried out in the green plane, region homogeneity is exploited to simplify the prediction filter and improve the prediction result. Theoretically, similar idea can be adopted in handling a non-green sample by considering the direction information of its neighboring samples. For any non-green sample $c(i,j)$, if the directions of all green samples in $S_{c(i,j)}$ are identical, pixel (i,j) can also be considered as in a homogenous region. Its predicted color difference value $\hat{d}(i,j)$ can then be estimated as follows.

$$\hat{d}(i,j) = \begin{cases} d(i,j-2) & \text{if } Dir(m,n) = W \\ d(i-2,j-2) & \text{if } Dir(m,n) = NW \\ d(i-2,j) & \text{if } Dir(m,n) = N \\ d(i-2,j+2) & \text{if } Dir(m,n) = NE \end{cases} \quad \forall (m,n) \in S_{c(i,j)} \quad (6.7)$$

However, such an arrangement is abandoned when a non-green sample is processed in CMBP as edges are generally deemphasized in the color difference domain. As a matter of fact, simulation results showed that this arrangement did not improve the prediction result of $d(i,j)$. For example, as far as the testing image generated with Image 8 is concerned, the zero-order entropy value of $\{ d(i,j) - \hat{d}(i,j) \mid (i,j) \in$ non-green sub-image$\}$ obtained without region classification and that obtained with region classification are, respectively, 5.423 bpp and 5.434 bpp. The entropy of the resultant residue plane is even higher when region classification is exploited. Furthermore, as shown in Figure 6.10, the correlation coefficients of the prediction residues are more or less the same no matter whether region classification is used or

not, which shows that region classification does not effectively contribute to the de-correlation performance. As a result, in the proposed scheme, a single pre-defined prediction filter is used to estimate $d(i,j)$ with eqn. (6.5) no matter whether the pixel is in a homogeneous region.

Again, w_k are trained with the same set of training images used to train the predictor coefficients in eqn. (6.2). For the compression results reported in this chapter, the predictor used for the color difference prediction is

$$\hat{d}(i,j) = round\left(\frac{4d(m_1,n_1) + 2d(m_2,n_2) + d(m_3,n_3) + d(m_4,n_4)}{8}\right). \qquad (6.8)$$

The prediction error is then obtained with $d(i,j) - \hat{d}(i,j)$. Figure 6.9 summaries how to generate the prediction residue of the corresponding color difference plane for the non-green plane of a CFA image.

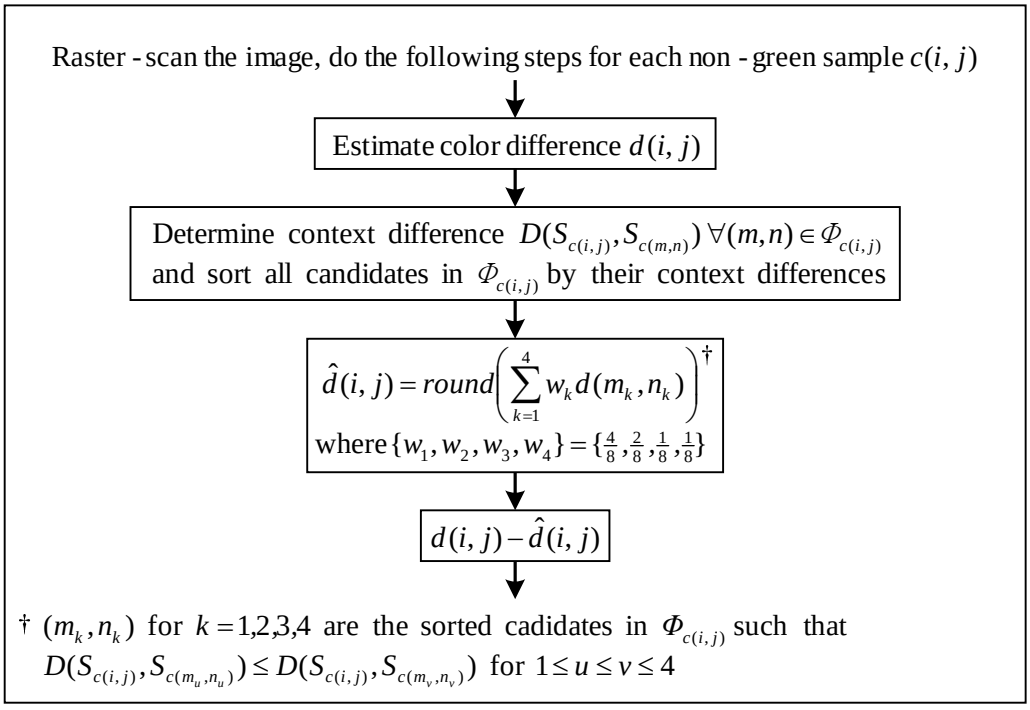
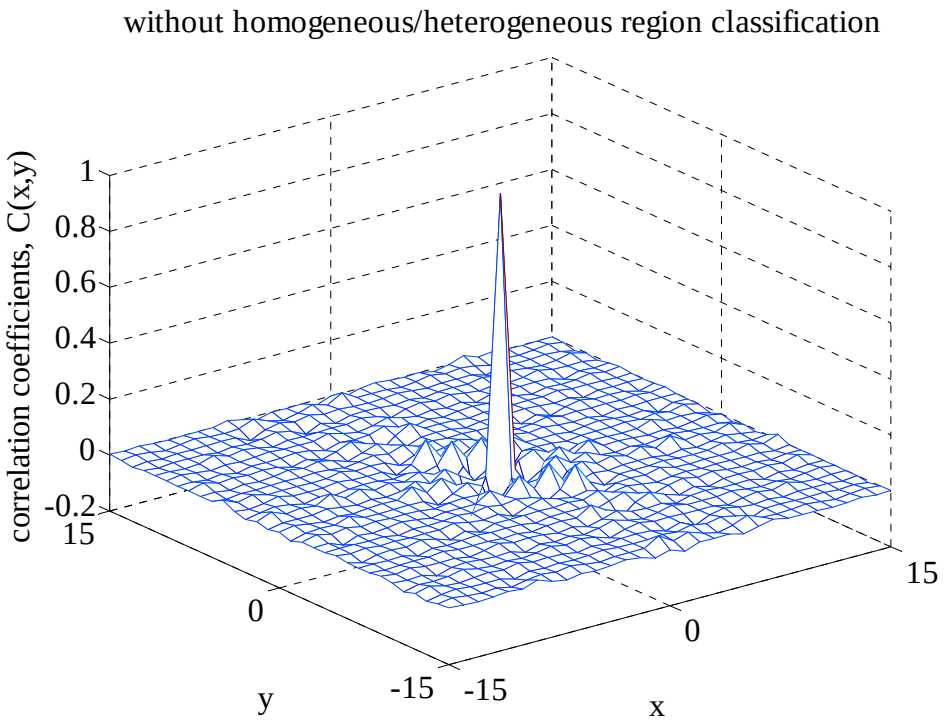
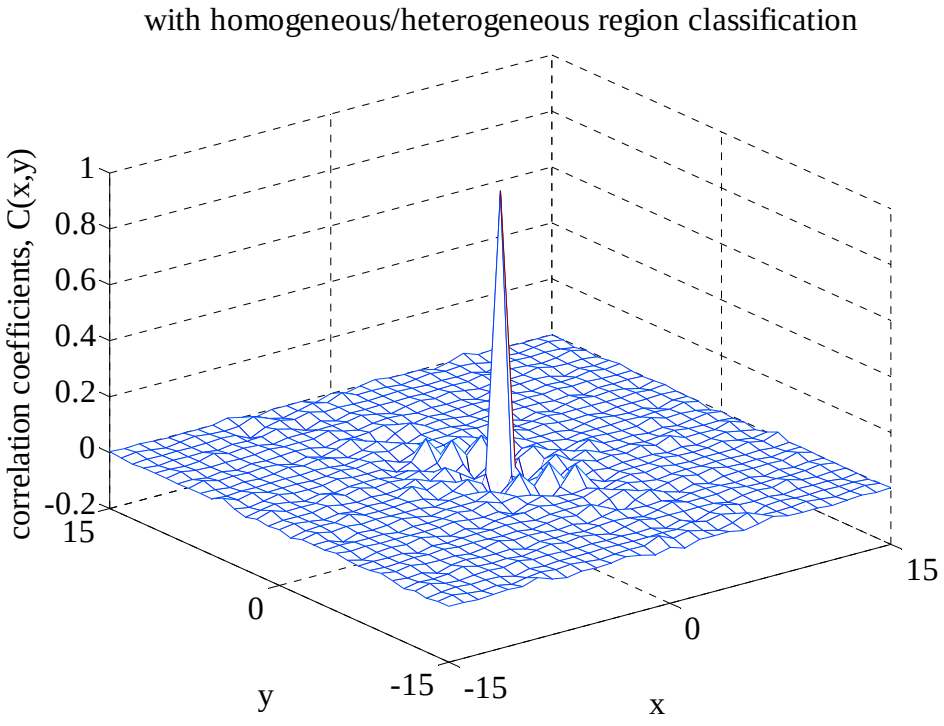


Figure 6.9 Steps to handle the non-green plane of a CFA image in CMBP



(a)



(b)

Figure 6.10 Correlation among the prediction residues associated with the non-green sub-image of testing image Image 8: (a) without region classification, and (b) with region classification

In CMBP, all real green, red and blue samples are encoded in a raster-scan manner. The four samples used for predicting sample $g(i,j)$ in eqn. (6.2) are $g(i,j)$'s closest processed neighboring samples of the same color. They have the highest correlation to $g(i,j)$ in different directions and hence can provide a good prediction result even in an edge region. A similar argument applies to explain why $\Phi_{c(i,j)}$ is used when handling a non-green sample $c(i,j)$.

As for the support region, no matter the concerned sample is green or not, its support is defined based on its four closest known green samples as shown in Figure 6.3. This is because the green channel has a double sampling rate as compared with the other channels in a CFA image and hence provides a more reliable context for matching.

In the proposed compression scheme, as green samples are encoded first in raster sequence, all green samples are known in the decoder and hence the support of a non-green sample can be non-causal while the support of a green sample has to be causal. This non-causal support tightly and completely encloses the sample of interest. It models image features such as intensity gradient, edge orientation and textures better such that more accurate support matching can be achieved.

6.3 Adaptive Color Difference Estimation

When compressing the non-green color plane, color difference information is exploited to remove the color spectral dependency. Here, we show how our proposed method estimates the color difference value of a pixel without having a known green sample of the pixel.

Let $c(m,n)$ be the intensity value of the available color sample (either red or blue) at a non-green sampling position (m,n). The green-red (green-blue) color difference of

pixel (*m*,*n*), *d*(*m*,*n*), is obtained by

$$d(m,n) = \hat{g}(m,n)\text{-}c(m,n) \tag{6.9}$$

where $\hat{g}(m,n)$ represents the estimated intensity value of the missing green component at position (*m*,*n*).

In the proposed estimation, $\hat{g}(m,n)$ is adaptively determined according to the horizontal gradient δH and the vertical gradient δV at (*m*,*n*) as follows.

$$\hat{g}(m,n) = round\left(\frac{\delta H \times G_V + \delta V \times G_H}{\delta H + \delta V}\right) \tag{6.10}$$

where $G_H = (g(m,n-1) + g(m,n+1))/2$ and $G_V = (g(m-1,n) + g(m+1,n))/2$ denote, respectively, the preliminary green estimates obtained by linearly interpolating the adjacent green samples horizontally and vertically. Note that, in eqn. (6.10), the missing green value is determined in such a way that a preliminary estimate contributes less if the gradient in the corresponding direction is larger. The weighing mechanism will automatically direct the estimation process along an edge if there is.

To simplify the estimation of $\hat{g}(m,n)$, one can check if pixel (*m*,*n*) is in a homogenous region by comparing the direction of (*m*,*n*)'s four neighboring green samples in $S_{c(m,n)}$. A straight forward estimation of $\hat{g}(m,n)$ can then be performed if it is. Specifically, we have

$$\hat{g}(m,n) = \begin{cases} round(G_H) & \text{if } Dir(a,b) = W, \forall (a,b) \in S_{c(m,n)} \\ round(G_V) & \text{if } Dir(a,b) = N, \forall (a,b) \in S_{c(m,n)} \end{cases} \tag{6.11}$$

In other words, as far as eqn. (10) is concerned, we have

$$\begin{cases} \delta H = 0 \\ \delta V = 1 \end{cases} \quad \text{if } Dir(a,b) = W, \forall (a,b) \in S_{c(m,n)}$$

and

$$\begin{cases} \delta H = 1 \\ \delta V = 0 \end{cases} \quad \text{if } Dir(a,b) = N, \forall (a,b) \in S_{c(m,n)}$$

when pixel (m,n) is in a homogenous region. Remind that the green plane is encoded first, and hence the directions of all green samples are available for the detection.

When pixel (m,n) is not in a homogenous region or the common direction of all green samples in $S_{c(m,n)}$ is not N or W, a more sophisticated approach is used to estimate gradients δH and δV for realizing eqn. (6.10). Specifically, they are determined by averaging all local green gradients in the same direction within a 5×5 window as

$$\delta H = \frac{1}{5} \sum_{(p,q) \in \begin{Bmatrix} (m-1,n-2),(m+1,n-2), \\ (m,n-1),(m-1,n),(m+1,n) \end{Bmatrix}} \left| g(p,q) - g(p,q+2) \right|$$

and

$$\delta V = \frac{1}{5} \sum_{(p,q) \in \begin{Bmatrix} (m-2,n-1),(m-2,n+1), \\ (m-1,n),(m,n-1),(m,n+1) \end{Bmatrix}} \left| g(p,q) - g(p+2,q) \right| \qquad (6.12)$$

To reduce the effort, a simpler approach can be used to estimate δH and δV with the four adjacent green samples in $S_{c(m,n)}$ as follows.

$$\delta H = \left| g(m,n-1) - g(m,n+1) \right| \quad \text{and} \quad \delta V = \left| g(m-1,n) - g(m+1,n) \right|. \qquad (6.13)$$

In this chapter, all simulation results related to the proposed algorithm were obtained with eqn. (6.12) instead of eqn. (6.13) unless it is specified otherwise.

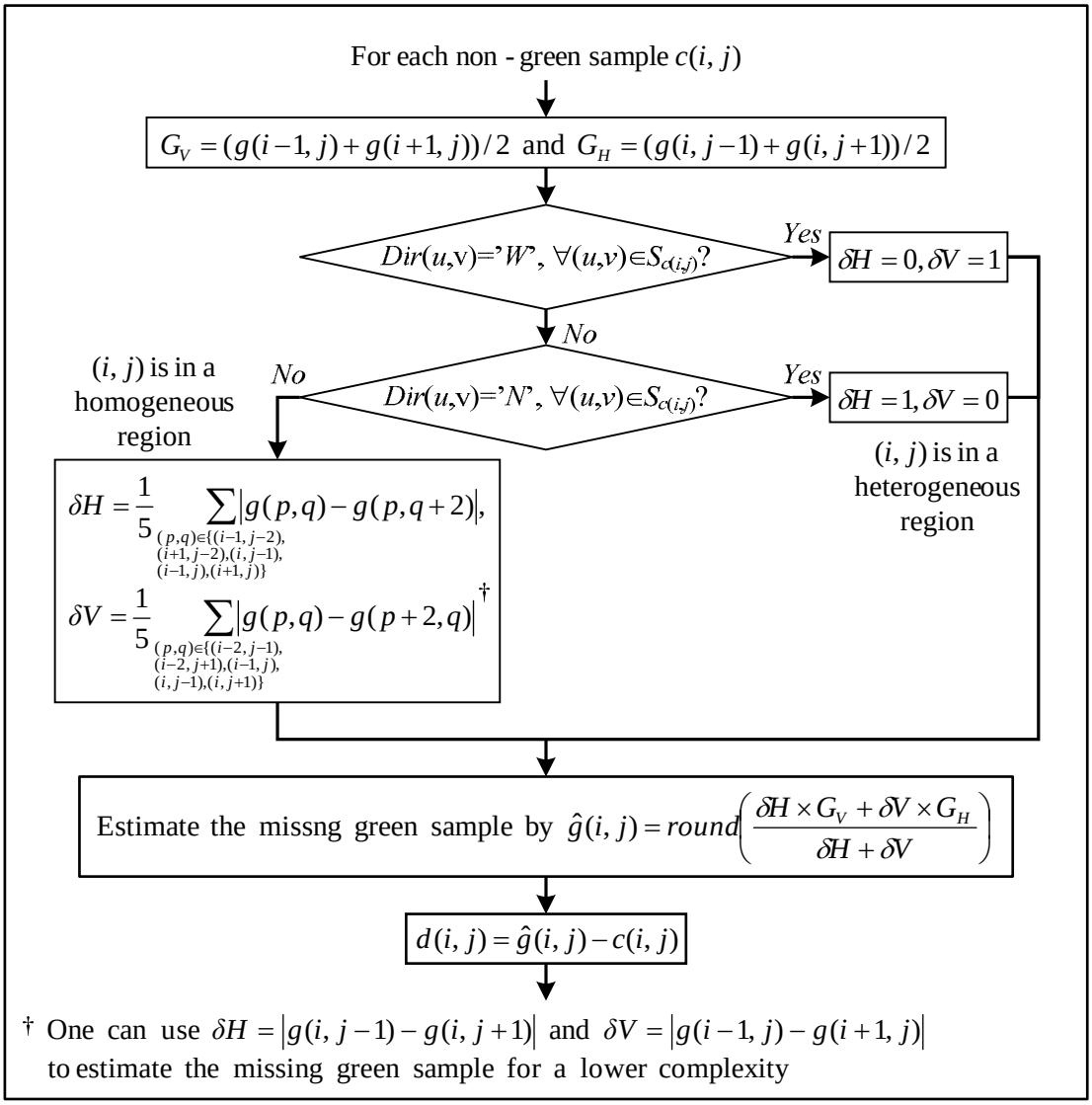
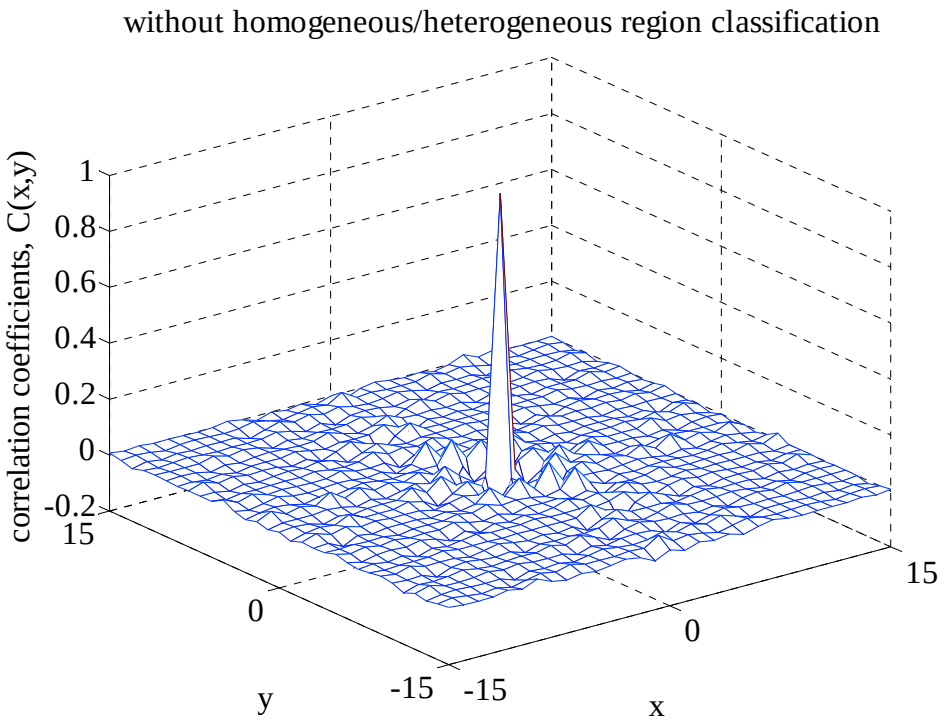
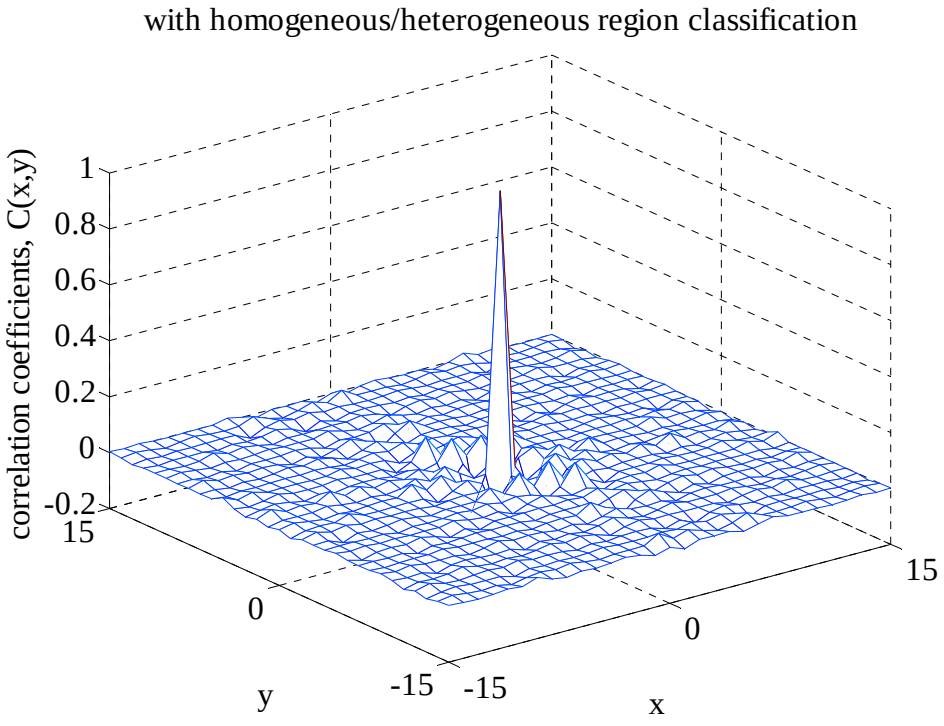


Figure 6.11 Steps to estimate the color difference value of a non-green sample



(a)



(b)

Figure 6.12 Correlation among the prediction residues associated with the non-green sub-image of testing image Image 8: (a) without region classification, and (b) with region classification in determining d(i,j)

Figure 6.11 summaries how to estimate the color difference value of a non-green sample in a CFA image. The proposed method works with CMBP as shown in Figure 6.9 to produce a residue plane associated with the non-green sub-image. One can skip the classification of a region (i.e. the realization of eqn. (6.11)) by bypassing the two decision steps in the flowchart shown in Figure 6.11. Figure 6.12 shows the correlation of the residues obtained under the two different conditions. For the testing image generated with Image 8, the zero-order entropy value of $\{ d(i, j) - \hat{d}(i, j) \mid (i,j) \in$ non-green sub-image$\}$ obtained with region classification and that obtained without region classification are, respectively, 5.434 bpp and 5.437 bpp. The reduction in entropy may not be significant when region classification is exploited, but the reduction in complexity is considerable as the realization of eqn. (6.12) (or (6.13)) can be saved in this case.

6.4 Proposed Compression Scheme

As shown in Figure 6.1, to encode a CFA image, the CFA image is first divided into a green sub-image and a non-green sub-image. The green sub-image is coded first and the non-green sub-image follows based on the green sub-image as a reference.

To code a sub-image, the sub-image is raster-scanned and each pixel is predicted with its 4 neighboring pixels by using the prediction scheme proposed in Section 6.2. The prediction error of pixel (i,j) in the CFA image, say $e(i,j)$, is given by

$$e(i, j) = \begin{cases} g(i, j) - \hat{g}(i, j) & \text{if pixel } (i, j) \text{ is in green sub-image} \\ d(i, j) - \hat{d}(i, j) & \text{if pixel } (i, j) \text{ is in non-green sub-image} \end{cases} \quad (6.14)$$

where $g(i,j)$ and $d(i,j)$ are, respectively, the real green sample value and the color

difference value of pixel (i,j). $d(i,j)$ is estimated by the method described in Section 6.3. $\hat{g}(i,j)$ and $\hat{d}(i,j)$, respectively, represent the predicted green intensity value and the predicted color difference value of pixel (i,j). The error residue $e(i,j)$ is then mapped to a non-negative integer as follows to reshape its value distribution to an exponential one from a Laplacian one.

$$E(i,j)=\begin{cases}-2e(i,j) & \text{if } e(i,j)\le 0\\ 2e(i,j)-1 & \text{otherwise}\end{cases} \tag{6.15}$$

The $E(i,j)$'s from the green sub-image are raster-scanned and coded with Rice code first. The $E(i,j)$'s from the non-green sub-image are further decomposed into two residue sub-planes. One carries the $E(i,j)$'s originated from the red CFA samples while the other one carries those originated from the blue CFA samples. The two residue sub-planes are then raster-scanned and coded with Rice code as well. Their order of processing does not matter as there is no interdependency among these two residue sub-planes. That they are separately handled is just because the Rice code can be made adaptive to their statistical properties in such an arrangement. For reference, the residue sub-planes originated from the red, the green and the blue CFA samples are, respectively, referred to as E_R, E_G and E_B.

Rice code is employed to code $E(i,j)$ because of its simplicity and high efficiency in handling exponentially distributed sources. When Rice code is used, each mapped residue $E(i,j)$ is split into a quotient $Q=\text{floor}(E(i,j)/2^k)$ and a remainder $R=E(i,j)\text{mod}(2^k)$, where parameter k is a non-negative integer. The quotient and the remainder are then saved for storage or transmission. The length of the codeword used for representing $E(i,j)$ is k–dependent and is given by

$$L(E(i,j)\,|\,k) = floor\left(\frac{E(i,j)}{2^k}\right) + 1 + k \tag{6.16}$$

Parameter k is critical to the compression performance as it determines the code length of $E(i,j)$. For a geometric source $\boldsymbol{S}$ with distribution parameter $\rho \in (0,1)$ (i.e. Prob($\boldsymbol{S}$=s) =$(1-\rho)\rho^s$ for s=0,1,2,…), the optimal coding parameter k is given as

$$k = \max\left\{0, ceil\left(\log_2\left(\frac{\log\phi}{\log\rho^{-1}}\right)\right)\right\} \tag{6.17}$$

where $\phi = (\sqrt{5}+1)/2$ is the golden ratio [71]. Since the expectation value of the source is given by $\mu=\rho\,(1-\rho)^{-1}$, as long as μ is known, parameter ρ and hence the optimal coding parameter k for the whole source can be determined easily.

In the proposed compression scheme, μ is estimated adaptively in the course of encoding E_R, E_G and E_B. In particular, it is estimated by

$$\tilde{\mu} = round\left(\frac{\alpha\tilde{\mu}_p + M_{i,j}}{1+\alpha}\right) \quad \text{and} \quad M_{i,j} = \left(\frac{1}{4}\sum_{(a,b)\in\zeta_{i,j}} E(a,b)\right). \tag{6.18}$$

where $\tilde{\mu}$ is the current estimate of μ for selecting the k to determine the codeword format of the current $E(i,j)$, $\tilde{\mu}_p$ is the previous estimate of $\tilde{\mu}$, $M_{i,j}$ is the local mean of $E(i,j)$ in a local region defined by Set $\zeta_{i,j}$, and α is a weighting factor which specifies the significance of $\tilde{\mu}_p$ and $M_{i,j}$ when updating $\tilde{\mu}$. Set $\zeta_{i,j}$ is a set of four processed pixel locations which are closest to pixel (i,j) and, at the same time, possess samples of the same color as pixel (i,j) does. When coding E_G, it is defined to be

$\{(i,j\text{-}2),(i\text{-}1,j\text{-}1),(i\text{-}2,j),(i\text{-}1,j\text{+}1)\}$. For coding E_R and E_B, Set $\zeta_{i,j}$ is defined to be $\{(i,j\text{-}2),(i\text{-}2,j\text{-}2),(i\text{-}2,j),(i\text{-}2,j\text{+}2)\}$. $\tilde{\mu}$ is updated for each $E(i,j)$. The initial value of $\tilde{\mu}_p$ is 0 for all residue sub-planes.

Experimental results showed that α=1 can provide a good compression performance. Figure 6.13 shows how parameter α affects the final compression ratio of the proposed compression scheme. Curve R, G and B respectively show the cases when coding E_R, E_G and E_B. The curve marked with 'All' shows the overall performance when all residue sub-planes are compressed with a common α value.

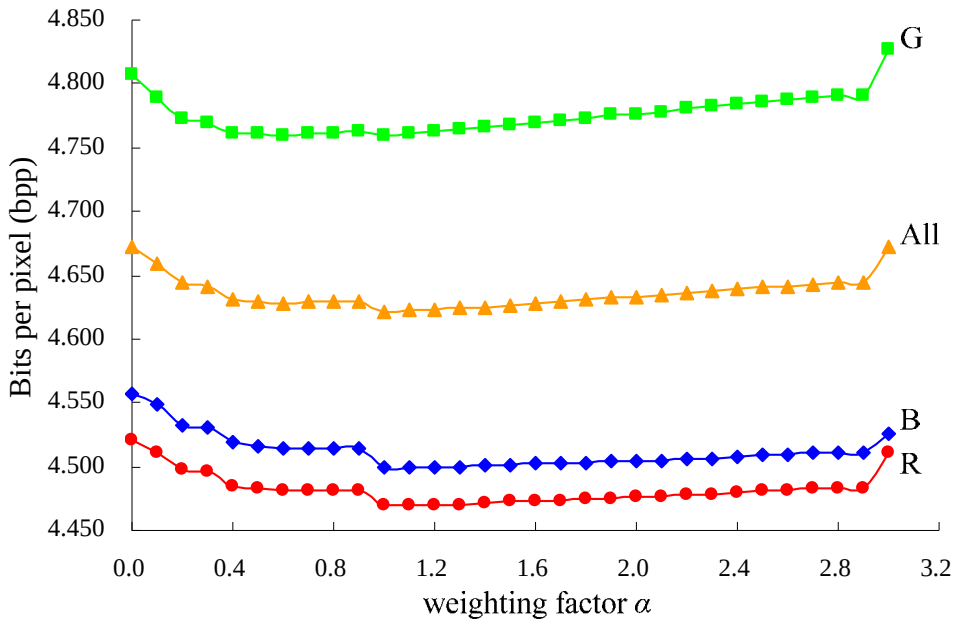


Figure 6.13 Average output bit-rates of the proposed compression scheme achieved with different α values

The decoding process is just the reserve process of encoding. The green sub-image is decoded first and then the non-green sub-image is decoded with the decoded green sub-image as a reference. The original CFA image is then reconstructed by combining the two sub-images.

6.5 Performance Evaluation

Simulations were carried out to evaluate the performance of the proposed compression scheme. The twenty-four 24-bit color shown in Appendix A were sub-sampled according to the Bayer pattern to form a set of 8-bit testing CFA images. These testing CFA images are of size 512×768 each. They were then directly coded by the proposed compression scheme for evaluation. Some representative lossless compression schemes such as JPEG-LS [19], JPEG2000 (lossless mode) [20] and LCMI [72] were also evaluated for comparison.

JPEG-LS and JPEG2000 are dedicated for compressing gray-scale images. To handle a CFA image, one can add a pre-processing step to turn a CFA image into 4 sub-images as shown in Figure 6.14 and then encode the 4 sub-images individually with either JPEG-LS or JPEG2000 as if they were grey-scale images.

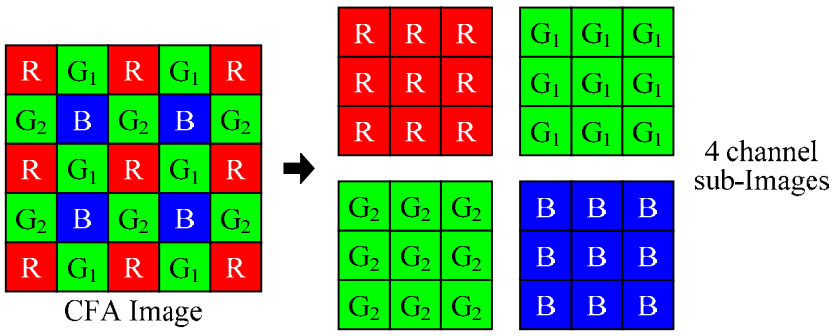


Figure 6.14 The pre-processing step used to divide a CFA image into four channel sub-images for evaluating the performance of JPEG-LS and JPEG2000.

Table 6.1 lists the average output bit-rates of the CFA images achieved by various compression schemes in terms of bits per pixel (bpp). It clearly shows that the proposed scheme outperforms all other evaluated schemes in all testing images. Especially for the images which contain many edges and fine textures such as Image 1, 5, 8, 13, 20 and 24, the bit-rates achieved by the proposed scheme are at least 0.34 bpp lower than the corresponding bit-rates achieved by LCMI, the scheme offers the second best compression performance. These results demonstrate that the proposed compression scheme is robust to remove the CFA data dependency even though the image contains complicated structures. On average, the proposed scheme yields a bit-rate as low as 4.622 bpp. It is, respectively, around 78.3%, 92.1% and 94.5% of those achieved by JPEG-LS, JPEG2000 and LCMI. When the pre-processing step shown in Figure 6.14 is used, the bit-rates achieved by JPEG-LS and JPEG2000 are, respectively, 0.382 bpp and 0.629 bpp higher than that of the proposed method. As a final remark, the proposed method yields an average zero-order entropy of 4.869 bpp, which is 0.156 bpp lower than that attained by LCMI.

In the proposed compression scheme, the non-green sub-image is processed in the color difference domain. Accordingly, the missing green samples in the sub-image have to be estimated for extracting the color difference information of the non-green sub-image. An estimation method for estimating the missing green samples and its simplified version (using eqn. (6.13) instead of eqn. (6.12) to estimate δH and δV), are proposed in Section 6.3. Obviously, one can make use of some other estimation methods such as bilinear interpolation [23] (BI), edge sensing interpolation [28] (ESI) and adaptive directional interpolation [67] (ADI) to achieve the same objective.

Image	JPEG-LS	JPEG2000	JPEG-LS /w pre-proc.	JPEG2000 /w pre-proc.	LCMI	Ours
1	6.403	5.816	5.944	6.211	5.824	5.478
2	6.787	5.134	4.632	4.871	4.629	4.331
3	5.881	4.216	4.117	4.331	3.965	3.746
4	6.682	4.931	4.827	4.988	4.606	4.379
5	6.470	5.947	6.187	6.540	5.859	5.409
6	5.871	5.210	5.220	5.433	5.139	4.881
7	5.974	4.500	4.426	4.737	4.299	3.960
8	6.295	5.899	6.044	6.433	5.966	5.570
9	5.074	4.391	4.440	4.617	4.319	4.188
10	5.395	4.556	4.558	4.833	4.415	4.227
11	5.370	4.986	5.070	5.340	4.952	4.683
12	5.628	4.485	4.404	4.644	4.307	4.089
13	6.747	6.372	6.568	6.817	6.503	6.138
14	6.289	5.555	5.740	6.035	5.487	5.170
15	6.317	4.656	4.335	4.613	4.396	4.102
16	5.289	4.552	4.724	4.934	4.521	4.376
17	4.965	4.547	4.801	5.041	4.499	4.286
18	6.184	5.570	5.766	5.980	5.538	5.284
19	5.470	4.909	5.084	5.234	4.898	4.711
20	4.317	4.026	3.402	3.694	4.054	3.541
21	5.467	5.039	5.073	5.272	4.983	4.803
22	6.188	5.218	5.238	5.432	5.060	4.847
23	6.828	4.525	4.097	4.217	3.960	3.847
24	5.719	5.223	5.401	5.770	5.257	4.873
Avg.	**5.900**	**5.011**	**5.004**	**5.251**	**4.893**	**4.622**

Table 6.1 Achieved bit-rates (in bits/pixel) of various lossless compression schemes

Image	The proposed compression scheme with				
	BI	ESI	ADI	Method in Section 6.3	
				Using eqn. (6.13)	Using eqn. (6.12)
1	5.414	5.411	5.349	5.299	5.245
2	4.366	4.365	4.367	4.343	4.319
3	3.724	3.721	3.709	3.682	3.664
4	4.314	4.312	4.304	4.305	4.275
5	5.312	5.301	5.267	5.221	5.191
6	4.851	4.849	4.802	4.744	4.684
7	3.890	3.893	3.881	3.868	3.858
8	5.574	5.570	5.447	5.326	5.300
9	4.154	4.156	4.113	4.090	4.071
10	4.173	4.171	4.142	4.102	4.081
11	4.625	4.618	4.586	4.526	4.483
12	4.085	4.083	4.063	4.014	3.977
13	5.969	5.967	5.981	5.963	5.918
14	5.030	5.028	5.006	4.980	4.935
15	4.192	4.186	4.161	4.162	4.145
16	4.295	4.291	4.275	4.206	4.152
17	4.202	4.201	4.163	4.147	4.132
18	5.117	5.116	5.126	5.120	5.089
19	4.705	4.703	4.627	4.579	4.552
20	3.701	3.698	3.676	3.665	3.663
21	4.737	4.735	4.723	4.703	4.674
22	4.732	4.732	4.730	4.734	4.697
23	3.844	3.841	3.838	3.837	3.827
24	4.787	4.785	4.771	4.705	4.672
Avg.	**4.575**	**4.572**	**4.546**	**4.513**	**4.484**

Table 6.2 Average bit-rates (in bits/pixel) for coding non-green sub-images with the proposed compression scheme when using a particular estimation method to estimate a missing green sample for reference

For comparison, a simulation was carried out to evaluate the performance of these methods when they were used to compress a non-green sub-image with the proposed compression scheme. In this study, only the non-green sub-images are involved as the compression of green sub-images does not involve the estimation of missing green components. In the realization of BI, a missing green sample is

estimated by rounding the average value of its four surrounding known green samples. For ESI, the four surrounding known green samples are weighted before averaging. The weights are determined according to the gradients among the four known green samples [28]. ADI is a directional linear-based interpolation method in which the interpolation direction is determined by comparing the horizontal and vertical green gradients to a pre-defined threshold [67]. The threshold value was set to be 30 in our simulation as it provided the best compression result for the training set.

Table 6.2 reveals the average bit rates of the outputs achieved by the proposed compression scheme when different methods were used to estimate the missing green samples in the non-green sub-images. It shows that the adaptive estimation methods proposed in Section 6.3 are superior to the other evaluated estimation methods. On average, the best proposed estimation method achieves a bit-rate of 4.484 bpp which is around 0.1 bpp lower than that achieved by BI.

Methods used to estimate missing green samples in non-green planes	**ADD**	**MUL**	**CMP**	**SHT**	**ABS**	**Total**
BI	21.70	0.00	6.00	3.70	3.00	**34.4**
ESI	42.70	12.00	6.00	2.70	9.00	**72.4**
ADI	23.45	0.00	9.86	3.70	5.00	**42.0**
Method in Section 6.3 using eqn. (6.13)	22.35	2.65	7.93	3.70	3.00	**39.6**
Method in Section 6.3 using eqn. (6.12)	29.43	2.65	7.93	3.70	3.00	**46.7**

Table 6.3 Average complexity (in operations/pixel) for different variants of the proposed compression scheme to generate prediction residues of both green and non-green sub-imagesimages

While Table 6.2 reports the compression performance of the proposed compression scheme and its various variants, Table 6.3 lists their complexity cost paid for producing all prediction residues of both green and non-green planes. It is measured in terms of the average number of operations required per pixel in our

simulations. Operations including addition (ADD), multiplication (MUL), bit-shift (SHT), comparison (CMP) and taking absolute value (ABS) are all taken into account.

The proposed compression scheme is composed of four functional components. A study was carried out to evaluate the contribution of each component to the overall performance of the scheme. The same set of 24 testing CFA images were used again in the evaluation.

In particular, when the prediction components are switched off (i.e. $\hat{g}(i, j) = \hat{d}(i, j) = 0$ in Figure 6.1a), the zero-order entropy values of $\{e(i,j) \mid (i,j) \in$ green sub-image$\}$ and $\{e(i,j) \mid (i,j) \in$ non-green sub-image$\}$ are, respectively, 7.114 bpp and 6.295 bpp on average, which are around 40.3% and 34.2% higher than the case when the prediction components are on. As for the component of color difference estimation, the proposed adaptive color difference estimation scheme provided a non-green residue plane of zero-order entropy 4.690 bpp on average, which is 0.114 bpp lower than that provided by using bilinear interpolation instead.

To show the contribution of the proposed adaptive Rice code encoding scheme, we encoded $E(i,j)$ with the conventional Rice code instead of the proposed one for comparison. In its realization, the coding parameter k for coding a sub-image is fixed and determined with eqn. (6.17). The parameter μ is estimated to be the mean of $E(i,j)$ in the sub-image. After all, it achieved an average bit-rate of 5.084 bpp, which is 0.462 bpp higher than that achieved by using the proposed adaptive Rice code encoding scheme.

When the proposed compression scheme (with eqn. (6.12)) was implemented in software with C++ programming language, the average execution time to compress a 512×768 CFA image on a 3.0GHz Pentium 4 PC with 1024MB RAM is around 0.1019s.

6.6 Chapter Summary

In this chapter, a lossless compression scheme for Bayer CFA images is proposed. This scheme separates a Bayer CFA image into a green sub-image and a non-green sub-image and then encodes them separately with predictive coding. The prediction is carried out in the intensity domain for the green sub-image while it is carried out in the color difference domain for the non-green sub-image. In both cases, a context matching technique is used to rank the neighboring pixels of a pixel for predicting the existing sample value of the pixel. The prediction residues originated from the red, the green and the blue samples of the Bayer CFA images are then separately encoded.

The value distribution of the prediction residue can be modeled as an exponential distribution, and hence Rice code is used to encode the residues. We assume the prediction residue is a local variable and estimate the mean of its value distribution adaptively. The divisor used to generate the Rice code is then adjusted accordingly so as to improve the efficiency of Rice code.

Experimental results show that the proposed compression scheme can efficiently and effectively de-correlate the data dependency in both spatial and color spectral domains. Consequently, it provides the best average compression ratio as compared with the latest lossless Bayer CFA image compression schemes.

Chapter 7 Conclusions and Future Works

7.1 General Conclusions

As compared with upgrading hardware specifications, optimizing the processing pipeline is alternatively a more cost effective approach to enhance the overall performance of a digital camera. In this thesis, several elements in the pipeline including color demosaicing, digital zoom and image compression are addressed and four proposals have been made to improve the performance of the processing pipeline.

One of the challenging tasks in color demosaicing is to preserve the image edge features. We conducted a study to investigate why the image features cannot be preserved in the output when the conventional adaptive color plane interpolation algorithm [21,22] were used. In Chapter 3, the study results are reported. Based on the study results, an adaptive feature-preserving demosaicing algorithm is presented. In the algorithm, the color difference variance of the pixels along the horizontal and vertical axes is used as supplementary information to more accurately determine the interpolation direction for the missing green samples. As a result, finer texture pattern details can be preserved in the output. Simulation results show that the proposed algorithm can produce a better demosaicing result, both subjectively and objectively, as compared with a number of advanced demosaicing algorithms.

Chapter 4 addresses the problem of digital zoom. A low complexity joint demosaicing and zooming algorithm is proposed in the chapter to produce an enlarged output in an efficient way. By sharing the edge information extracted from the raw sensor data, the missing samples in the three color planes are estimated one by one. Specifically, the missing green samples in the enlarged image are estimated first. The red and the blue missing samples are then estimated based on the interpolated green

plane as a reference. This arrangement not only can consistently utilize the raw sensor data but also can efficiently interpolate the missing samples in the enlarged image as no separate extraction process is required in different stages. Simulation results reveal that the proposed algorithm outperforms the conventional approaches, which generally perform demosaicing-after-zooming or zooming-after-demosaicing, in terms of both output quality and complexity.

Inspired by the idea of sharing extracted edge information in different stages in the joint algorithm presented in Chapter 4, an efficient decision-based demosaicing algorithm is proposed in Chapter 5 to further improve the processing pipeline performance in terms of both quality and complexity. In the proposed algorithm, a new edge-sensing measure called integrated gradient, which extracts gradient information in both color intensity and color difference domains simultaneously, is developed for being shared in various stages throughout the demosaicing processes to interpolate the color channels. This measure is used not only to determine the interpolation direction for estimating the missing green samples, but to refine the interpolation results of the missing red and blue samples as well. A green plane enhancement which works with the integrated gradient is also introduced to further improve the algorithm's efficiency. Simulation results confirmed that the proposed algorithm outperforms the state-of-art demosaicing algorithms in terms of output quality at a complexity of around 80 arithmetic operations per pixel.

The issue of image compression is finally addressed in Chapter 6. Based on the fact mentioned in recent reports that the compression-first schemes outperform the conventional demosaicing-first schemes in terms of output image quality, an efficient prediction-based lossless compression scheme for Bayer CFA images is proposed in the chapter. In this scheme, a context matching technique, an adaptive color difference estimation scheme and an adaptive codeword generation technique are exploited

respectively to rank the neighboring pixels when predicting a pixel, to remove the color spectral redundancy when handling red and blue samples, and to adjust the divisor of Rice code when encoding the prediction residues. Experimental results show that the proposed compression scheme can efficiently and effectively reduce the data redundancy in both spatial and color spectral domains. As compared with the latest lossless Bayer CFA image coding schemes, the proposed scheme provides the best compression performance in our simulation study.

On the whole, a set of four image processing techniques have been developed to improve the performance of the processing pipeline and in turns the camera. These techniques deal executively with the Bayer CFA image and outperform most state-of-art techniques in terms of output quality, compression ratio and complexity.

7.2 Future Works

Though significant contribution has been made in our work to improve the performance of a digital camera, there is still room for further improvement. To enable a camera to capture a precise and high quality output at low power consumption, besides the issues addressed in this thesis, some other issues should also be addressed in future study. In this section, we will discuss some of the possible research directions or extensions of the current piece of work.

No matter whether it is in a conventional or an updated processing pipeline, noise reduction is generally carried out in the post-processing step on the demosaiced full-color images. However, such an arrangement is unreasonable because the demosaicing process always blends the sensor noise across color channels and correlates the noise with signals in the demosaiced output, which makes the task of denoising complicated and inefficient. To solve this problem, a dedicated noise

reduction algorithm or a joint denoising and demosaicing algorithm for the Bayer CFA images should be developed. As the input Bayer CFA image is always noisy in real situation, the noise-free input assumption made in most existing demosaicing algorithms is not upheld. A Bayer CFA image-based noise reduction algorithm which can handle a more realistic situation should be able to significantly improve the quality of the demosaiced output.

As a matter of fact, a joint demosaicing and denoising algorithm has been recently proposed in [73], in which the authors treat both demosaicing and denoising as an estimation problem and use the total least square (TLS) technique [74] to accomplish the two tasks simultaneously. Although the algorithm is shown to perform better than many demosaicing-then-denoising algorithms, it is not suitable for real-time implementation because of its high computational complexity requirement. Accordingly, investigation of some robust and practical denoising algorithms is still worth being deployed.

Recently, Kodak Image Sensor Solutions announced a new CFA pattern [75] for CMOS and CCD digital image sensors to enhance their performance under low-light conditions. This new pattern, as shown in Figure 7.1, builds upon the standard Bayer pattern by adding a panchromatic pixel next to each red, green and blue pixel. This panchromatic pixel senses all visible wavelengths and allows detecting luminance signals with a very high sensitivity. The sensitivity of the entire sensor is raised accordingly to provide a more detailed image. The remaining red, green and blue pixels presented on the sensor, on the other hand, constitute a set of chrominance channels for collecting color information. Reference [75] shows some image examples for references. They demonstrate that this new pattern can produce the full-color output with much less color artifacts than the conventional Bayer CFA pattern, especially in a low-light situation. In addition, it can get rid of motion blur a

lot as the high sensitive panchromatic pixel enables a faster shutter speed.

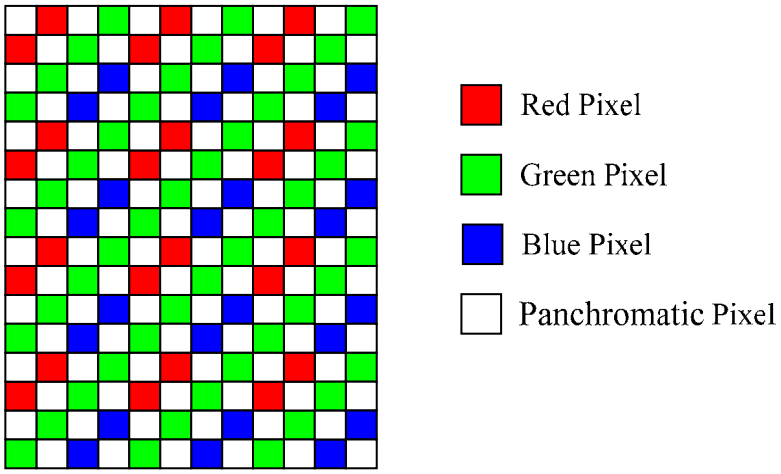


Figure 7.1 Layout of the new CFA pattern

With the new CFA layout, the current image processing chain structure as well as the processing algorithms in a camera must be changed. It requires research on understanding the effects of using this sensor technology in cameras. Besides, the development of new processing pipeline architectures as well as software algorithms specifically designed to work with the raw data generated from the new image sensor are also needed to be performed.

Appendix A Testing Images

Image 1

Image 2

Image 3

Image 4

Image 5

Image 6

Image 7

Image 8

Image 9

Image 10

Image 11

Image 12

Image 13

Image 14

Image 15

Image 16

Image 17

Image 18

Image 19

Image 20

Image 21

Image 22

Image 23

Image 24

References

[1] R. Ramanath, W. E. Snyder, Y. Yoo and M. S. A. D. M. S. Drew, “Color image processing pipeline,” *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 34–43, 2005.

[2] J. Adams, K. Parulski and K. Spaulding, “Color processing in digital cameras,” *IEEE Micro*, vol. 18, no. 6, pp. 20–30, 1998.

[3] K. Parulski and K. Spaulding, "Color image processing for digital cameras," *Digital Color Imaging Handbook*, G. Sharma, ed., pp. 727–757, Boca Raton, FL: CRC Press, 2003.

[4] P. L. P. Dillon, D. M. Lewis and F. G. Kaspar, “Color Imaging System Using a Single CCD Area Array,” *IEEE Journal of Solid-State Circuits*, vol. 13, no. 1, pp. 28–33, 1978.

[5] M. M. S. Battiato, “An Introduction to the Digital Still Camera Technology,” *ST Journal of System Research - Special Issue on Image Processing for Digital Still Camera*, vol. 2, no. 2, pp. 1–9, Dec. 2001.

[6] T. Lule, S. Benthien, H. Keller, F. A. M. F. Mutze, P. A. R. P. Rieve, K. A. S. K. Seibel, M. A. S. M. Sommer and M. A. B. M. Bohm, “Sensitivity of CMOS based imagers and scaling perspectives,” *IEEE Transactions on Electron Devices*, vol. 47, no. 11, pp. 2110–2122, Nov. 2000.

[7] R. Lukac and K. N. Plataniotis, “Color filter arrays: design and performance analysis,” *IEEE Transactions on Consumer Electronics*, vol. 51, no. 4, pp. 1260–1267, Nov. 2005.

[8] J. J. G. Savard, "Color Filter Array Designs," http://www.quadibloc.com/other/cfaint.htm.

[9] B. E. Bayer, *Color imaging array*, U.S. Patent No. 3,971,065, to Eastman Kodak Company (Rochester, NY), 1976.

[10] D. Alleysson, S. Susstrunk and J. Herault, “Linear demosaicing inspired by the human visual system,” *IEEE Transactions on Image Processing*, vol. 14, no. 4, pp. 439–449, Apr. 2005.

[11] R. Ramanath, W. Snyder, G. Bilbro and W. Sander, “Demosaicking methods for Bayer color arrays,” *Journal of Electronic Imaging*, vol. 11, no. 3, pp. 306–315, Jul. 2002.

[12] B. 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, Sep. 2002.

[13] B. K. Gunturk, J. Glotzbach, Y. Altunbasak, R. W. Schafer and R. M. Mersereau, “Demosaicking: color filter array interpolation,” *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 44–54, 2005.

[14] T. Toi and M. Ohta, “A subband coding technique for image compression in single CCD cameras with Bayer color filter arrays,” *IEEE Transactions on Consumer Electronics*, vol. 45, no. 1, pp. 176–180, Feb. 1999.

[15] X. Xie, G. Li, Z. Wang, C. Zhang, D. Li and X. Li, "A novel method of lossy image compression for digital image sensors with Bayer color filter arrays," *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS'05)*, pp. 4995–4998, 2005.

[16] S. Y. Lee and A. Ortega, "A novel approach of image compression in digital cameras with a Bayer color filter array," *Proceedings of IEEE International Conference on Image Processing (ICIP'01)*, 3, pp. 482–485, 2001.

[17] C. C. Koh, J. Mukherjee and S. K. Mitra, “New efficient methods of image compression in digital cameras with color filter array,” *IEEE Transactions on Consumer Electronics*, vol. 49, no. 4, pp. 1448–1456, Nov. 2003.

[18] N. X. Lian, L. Chang, V. Zagorodnov and Y. P. Tan, “Reversing demosaicking and compression in color filter array image processing: Performance analysis and modeling,” *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3261–3278, Nov. 2006.

[19] *Information Technology - Lossless and Near-Lossless Compression of Continuous-Tone Still Images (JPEG-LS)*, ISO/IEC Standard 14495-1, 1999.

[20] *Information technology - JPEG 2000 image coding system - Part 1: Core coding system*, INCITS/ISO/IEC Standard 15444-1, 2000.

[21] J. F. Hamilton and J. E. Adams, *Adaptive color plane interpolation in single sensor color electronic camera,* U.S. Patent No. 5,506,619, to Eastman Kodak Company (Rochester, NY), 1996.

[22] J. F. Hamilton and J. E. Adams, *Adaptive color plane interpolation in single sensor color electronic camera,* U.S. Patent No. 5,652,621, to Eastman Kodak Company (Rochester, NY), 1997.

[23] T. Sakamoto, C. Nakanishi and T. Hase, "Software pixel interpolation for digital still cameras suitable for a 32-bit MCU," *IEEE Transactions on Consumer Electronics,* vol. 44, no. 4, pp. 1342–1352, Nov. 1998.

[24] D. R. Cok, *Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal,* U.S. Patent No. 4,642,678, to Eastman Kodak Company (Rochester, NY), 1987.

[25] W. T. Freeman, *Median filter for reconstructing missing color samples,* U.S. Patent No. 4,724,395, to Polaroid Corporation (Cambridge, MA), 1988.

[26] S. C. Pei and I. K. Tam, "Effective color interpolation in CCD color filter arrays using signal correlation," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 13, no. 6, pp. 503–513, Jun. 2003.

[27] R. Lukac, K. N. Plataniotis, D. Hatzinakos and M. Aleksic, "A novel cost effective demosaicing approach," *IEEE Transactions on Consumer Electronics,* vol. 50, no. 1, pp. 256–261, Feb. 2004.

[28] R. Lukac and K. N. Plataniotis, "Data adaptive filters for demosaicking: A framework," *IEEE Transactions on Consumer Electronics,* vol. 51, no. 2, pp. 560–570, May 2005.

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

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

[31] R. Lukac and K. N. Plataniotis, "Normalized color-ratio modeling for CFA interpolation," *IEEE Transactions on Consumer Electronics,* vol. 50, no. 2, pp. 737–745, May 2004.

[32] N. Kehtarnavaz, H. J. Oh and Y. Yoo, “Color filter array interpolation using color correlation and directional derivatives,” *Journal of Electronic Imaging*, vol. 12, no. 4, pp. 621–632, Oct. 2003.

[33] A. Lukin and D. Kubasov, "An improved demosaicing algorithm," *Proceedings of Graphicon*, 2004.

[34] R. Ramanath and W. E. Snyder, “Adaptive demosaicking,” *Journal of Electronic Imaging*, vol. 12, no. 4, pp. 633–642, Oct. 2003.

[35] K. Hirakawa and T. W. Parks, “Adaptive homogeneity-directed demosaicing algorithm,” *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 360–369, Mar. 2005.

[36] X. Wu and N. Zhang, “Primary-consistent soft-decision color demosaicking for digital cameras - (Patent pending),” *IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1263–1274, Sep. 2004.

[37] D. Menon, S. Andriani and G. Calvagno, “Demosaicing with directional filtering and a posteriori decision,” *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 132–141, Jan. 2007.

[38] C. Y. Tsai and K. T. Song, “Heterogeneity-projection hard-decision color interpolation using spectral-spatial correlation,” *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 78–91, Jan. 2007.

[39] X. Li, “Demosaicing by successive approximation,” *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 370–379, Mar. 2005.

[40] N. X. Lian, L. Chang, Y. P. Tan and V. Zagorodnov, “Adaptive filtering for color filter array demosaicking,” *IEEE Transactions on Image Processing*, vol. 16, no. 10, pp. 2515–2525, Oct. 2007.

[41] 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, Dec. 2005.

[42] D. D. Muresan and T. W. Parks, “Demosaicing using optimal recovery,” *IEEE Transactions on Image Processing*, vol. 14, no. 2, pp. 267–278, Feb. 2005.

[43] S. Battiato, G. Gallo and F. Stanco, “A locally adaptive zooming algorithm for digital images,” *Image and Vision Computing*, vol. 20, no. 11, pp. 805–812, Sep. 2002.

[44] P. Thevenaz, T. Blu and M. Unser, “Interpolation revisited,” *IEEE Transactions on Medical Imaging*, vol. 19, no. 7, pp. 739–758, Jul. 2000.

[45] H. Jung Woo and L. Hwang Soo, “Adaptive image interpolation based on local gradient features,” *IEEE Signal Processing Letters*, vol. 11, no. 3, pp. 359–362, Mar. 2004.

[46] L. Zhang and X. Wu, “An edge-guided image interpolation algorithm via directional filtering and data fusion,” *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2226–2238, Aug. 2006.

[47] N. Herodotou and A. N. Venetsanopoulos, “Colour image interpolation for high resolution acquisition and display devices,” *IEEE Transactions on Consumer Electronics*, vol. 41, no. 4, pp. 1118–1126, Nov. 1995.

[48] R. Lukac, B. Smolka, K. Martin, K. N. Plataniotis and A. N. Venetsanopoulos, “Vector filtering for color imaging,” *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 74–86, Jan. 2005.

[49] R. Lukac, K. N. Plataniotis, B. Smolka and A. N. Venetsanopoulos, "Vector operators for color image zooming," *Proceedings of IEEE International Symposium on Industrial Electronics (ISIE'05)*, 3, pp. 1273–1277, 2005.

[50] Y. Cha and S. Kim, “Edge-forming methods for color image zooming,” *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2315–2323, Aug. 2006.

[51] R. Lukac, K. Martin and K. N. Plataniotis, “Demosaicked image postprocessing using local color ratios,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 6, pp. 914-920, Jun 2004.

[52] P. Longere, Z. Xuemei, P. B. Delahunt and D. H. Brainard, “Perceptual assessment of demosaicing algorithm performance,” *IEEE Proceedings*, vol. 90, no. 1, pp. 123–132, Jan. 2002.

[53] R. Lukac and K. N. Plataniotis, “Digital zooming for color filter array based image sensors,” *Real-Time Imaging*, vol. 11, no. 2, pp. 129–138, Apr. 2005.

[54] R. Lukac, K. N. Plataniotis and D. Hatzinakos, “Color image zooming on the Bayer pattern,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 15, no. 11, pp. 1475–1492, Nov. 2005.

[55] R. Lukac, K. Martin and K. N. Plataniotis, “Digital camera zooming based on unified CFA image processing steps,” *IEEE Transactions on Consumer Electronics,* vol. 50, no. 1, pp. 15–24, Feb. 2004.

[56] Y. T. Tsai, “Color image compression for single-chip cameras,” *IEEE Transactions on Electron Devices,* vol. 38, no. 5, pp. 1226–1232, May 1991.

[57] A. Bruna, F. Vella, A. Buemi and S. Curti, "Predictive differential modulation for CFA compression," *Proceedings of the 6th Nordic Signal Processing Symposium - NORSIG 2004*, pp. 101–104, 2004.

[58] S. Battiato, A. Bruna, A. Buemi and F. Naccari, "Coding techniques for CFA data images," *Proceedings of IEEE International Conference on Image Analysis and Processing (ICIAP'03)*, pp. 418–423, 2003.

[59] A. Bazhyna, A. Gotchev and K. Egiazarian, "Near-lossless compression algorithm for Bayer pattern color filter arrays," *Proceeding of SPIE - the International Society for Optical Engineering*, vol. 5678, pp. 198–209, 2005.

[60] B. Parrein, M. Tarin and P. Horain, "Demosaicking and JPEG2000 compression of microscopy images," *Proceedings of IEEE International Conference on Image Processing (ICIP'04)*, pp. 521–524, 2004.

[61] R. Lukac and K. N. Plataniotis, “Single-sensor camera image compression,” *IEEE Transactions on Consumer Electronics,* vol. 52, no. 2, pp. 299–307, May 2006.

[62] N. Zhang and X. L. Wu, “Lossless compression of color mosaic images,” *IEEE Transactions on Image Processing,* vol. 15, no. 6, pp. 1379–1388, Jun. 2006.

[63] D. D. Muresan and T. W. Parks, "Optimal recovery demosaicing," *Proceedings of IASTED Signal and Image Processing (SIP'02)*, pp. 260–265, 2002.

[64] J. E. Adams, "Design of practical color filter array interpolation algorithms for digital cameras," *Proceeding of SPIE - the International Society for Optical Engineering*, vol. 3028, pp. 117–125, 1997.

[65] R. Lukac and K. N. Plataniotis, “A robust, cost-effective postprocessor for enhancing demosaicked camera images,” *Real-Time Imaging,* vol. 11, no. 2, pp. 139–150, Apr. 2005.

[66] X. Zhang, "S-CIELAB: A Spatial Extension to the CIE L*a*b* DeltaE Color Difference Metric," http://white.stanford.edu/~brian/scielab/scielab.html.

[67] R. H. Hibbard, *Apparatus and method for adaptively interpolating a full color image utilizing luminance gradients*, U.S. Patent No. 5,382,976, to Eastman Kodak Company (Rochester, NY), 1995.

[68] R. Keys, “Cubic convolution interpolation for digital image processing,” *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 29, no. 6, pp. 1153–1160, Dec. 1981.

[69] S. Battiato, G. Gallo and F. Stanc, "A new edge-adaptive zooming algorithm for digital camera," *Proceedings of IASTED Signal Processing and Communications (SPC'00),* pp. 144–149, 2000.

[70] C. A. Laroche and M. A. Prescott, *Apparatus and method for adaptively interpolating a full color image utilizing chrominance gradients*, U.S. Patent No. 5,373,322, to Eastman Kodak Company (Rochester, NY), 1994.

[71] A. Said, *On the determination of optimal parameterized prefix codes for adaptive entropy coding,* Technical Report HPL-2006-74, HP Laboratories Palo Alto, 2006.

[72] X. Wu and Z. Lei, “Improvement of Color Video Demosaicking in Temporal Domain,” *IEEE Transactions on Image Processing,* vol. 15, no. 10, pp. 3138–3151, 2006.

[73] K. Hirakawa and T. W. Parks, “Joint demosaicing and denoising,” *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2146–2157, 2006.

[74] K. Hirakawa and T. W. Parks, "Image Denoising for Signal-Dependent Noise," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05),* 2, pp. 29–32, 2005.

[75] J. Compton and J. Hamilton, "Color Filter Array 2.0," http://johncompton.pluggedin.kodak.com/default.asp?item=624876.