New Efficient Methods of Image Compression in Digital Cameras with Color Filter Array

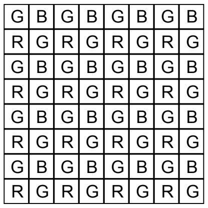
Chin Chye Koh, *Student Member, IEEE*, Jayanta Mukherjee, *Member, IEEE*, and Sanjit K. Mitra, *Life Fellow, IEEE*

Abstract — *Many consumer digital color cameras use a single light sensitive sensor and a color filter array (CFA) with each pixel element recording intensity information of one color component. The captured data is interpolated into a full color image, which is then compressed in many applications. Carrying out color interpolation before compression introduces redundancy in the data. In this paper we discuss methods and issues involved in the compression of CFA data before full color interpolation. The compression methods described operate on the same number of pixels as the sensor data. To obtain improved image quality, median filtering is applied as post-processing. Furthermore, to assure low complexity, the CFA data is compressed by JPEG. Simulations have demonstrated that substantial improvement in image quality is achievable using these new schemes.*

Index Terms — Bayer CFA, false color masking, median filtering, and quincunx pattern.

I. INTRODUCTION

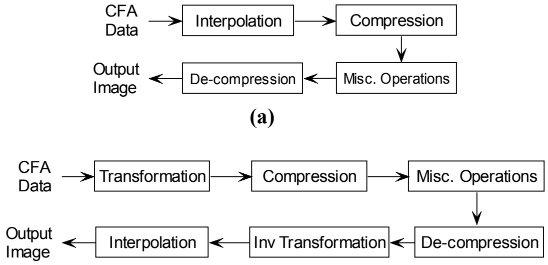
Most inexpensive digital cameras use a color filter array (CFA) with each pixel element of the sensor recording intensity information of one color component, typically red, green, or blue. Although several different CFAs have been proposed, the Bayer CFA [1] shown in Figure 1 is widely used. Here the green filters are in a quincunx (interlaced) grid with the red and blue filters filling up the empty locations. This paper considers only the Bayer CFA, but the algorithm can be adapted to other CFAs as well.



G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G
G	B	G	B	G	B	G	B
R	G	R	G	R	G	R	G

Fig. 1. Bayer patterned color filter array.

The CFA captures only one-third of the necessary color intensities and the full color image is generated from the captured data by interpolation. In most applications, the generated color image is then compressed using JPEG before storage or transmission. Thus JPEG operates on three times the initial CFA data as shown in Figure 2(a). A method of avoiding this data redundancy is obtained by moving the compression stage before color interpolation [2], [3] as illustrated in Figure 2(b). In both figures, the block "miscellaneous operations" refers to the process of storage or transmission of the compressed data.



(a)

(b)

Fig. 2. Flow diagram of compression schemes: (a) conventional method, interpolation before compression, (b) proposed method, interpolation after compression.

By compressing the CFA data before interpolation, more pertinent information is retained, allowing higher compression rates and/or image quality to be achieved. An application would be video conferences where the frame transmission rates and or image quality can be increased. Furthermore, when considering image or video storage, this scheme allows either storage of more images or reconstruction of higher quality color images and videos. Aside from the applications described above, perhaps the most interesting application would be wireless phones that send and receive still or video images. Since present day wireless phones use low quality cameras, applying the compression schemes described in this paper would be advantageous. This is especially important when considering the market for camera phones where it is estimated that by 2007, more than 160 million camera phones will be in use worldwide [4], [5].

A number of issues need to be resolved before the CFA data is compressed, the first being the choice of the interpolation algorithm. Many algorithms exist that can be used in reconstructing the captured scene from the CFA data. Some provide superior image quality at a high computational cost, while others are computationally efficient but yield lower (but still acceptable) image quality. All these interpolation

[1] This work has been supported in part by a University of California MICRO grant with matching support from Philips Research Laboratories and in part by Microsoft Corp.

C. C. Koh and S. K. Mitra is with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106 USA (email:{chinkoh,mitra}@ ece.ucsb.edu).

J. Mukherjee is with the Department of Computer Science & Engineering, Indian Institute of Technology, Kharagpur, INDIA, 721302 (e-mail: jay@cse.iitkgp.ernet.in).

Contributed Paper
Manuscript received September 4, 2003

algorithms behave differently under varying conditions, such as image contents and compression ratios. An issue that must be accounted for is the effect of the image contents. Image contents affect the performance of compression and interpolation, and may or may not adversely affect the quality of the images. Another issue of major concern is the appearance of false colors and it is often necessary to use some form of post-processing to mask these effects.

In this paper we outline two of our algorithms based on the scheme of Figure 2(b) and demonstrate that higher compression ratios and increased image quality are achieved compared to the results obtained using schemes based on Figure 2(a). In addition, by using a simple post-processing scheme, the gains in image quality are improved. We also study the effects of image contents on compression and full color interpolation.

In Section II, we describe the effects of directly compressing the CFA data. In Section III, we review the current state of the art in the compression of CFA data. In Section IV, we describe two new methods of compressing the CFA data. In Section V, we compare the performance of our methods with that of a few full color interpolation algorithms under various conditions. In Section VI, we consider the suppression of false color effects. In Section VII, we present the computational complexity of our algorithms. Finally, we draw conclusions in Section VIII.

II. Direct compression of Bayer CFA data

As indicated earlier, the objective of this work is to compress the Bayer patterned CFA data before full color interpolation. The ultimate goal is to increase the compression ratio of the compressed data while maintaining high image quality. Unfortunately, direct compression of CFA data with JPEG produces poor quality images as shown in Figure 3. Note that both images have been cropped to 128×128 to allow for visual comparison.

The measure of quality used in this paper is the Composite Peak Signal to Noise Ratio (CPSNR) [6] defined as:

$$CPSNR = 10\log_{10}\left(\frac{255^2}{\frac{1}{3MN}\sum_{\forall k}\left[I_{in}(i,j,k)-I_{out}(i,k,k)\right]^2}\right) \qquad (1)$$

Here, I_{in} and I_{out} are the input and output images respectively, M and N are the dimensions of each color component array, i and j are the locations of the pixels, and k represents the color plane.

Figure 3(a) shows an image compressed using the conventional method. Figure 3(b) shows the image generated from the Bayer pattern compressed directly using JPEG. Reconstruction of the image assumes that the CFA pattern in the data is preserved and interpolation is carried out as in the conventional case.

The Bayer patterned compressed image retains the same amount of details compared to the conventionally compressed image. However, there is a greater amount of distortion in Figure 3(b). The CPSNR of the image in Figure 3(a) is 33.37~dB, while that of Figure 3(b) is 30.04~dB. There are several reasons for this increased distortion. The first reason is that the CFA data is a combination of three color planes. These color planes although highly correlated still exhibit different levels of pixel intensities. When the three color planes are interlaced together (as in the CFA pattern), the overall effect is the creation of an image with high frequencies. This high frequency image does not allow high compression ratios when JPEG is used.

Upon closer inspection of Figure 3(b), we find that there is a repeating pattern that distorts the image. This repeating pattern is a result of the positioning of the pixels in the CFA pattern. It is caused by the spread of the color information over the three color planes. To emphasize the effect of the repeating pattern we display the difference between the interpolated and compressed images as shown in Figure 3(c) and (d). The difference image is obtained by first subtracting the pixel values of the compressed image from the interpolated uncompressed image and converting the difference into a gray-scale image. Finally, to increase the visual difference, the difference image is multiplied by a factor of 5. From the difference image we clearly see the distortion caused by the repeating pattern.

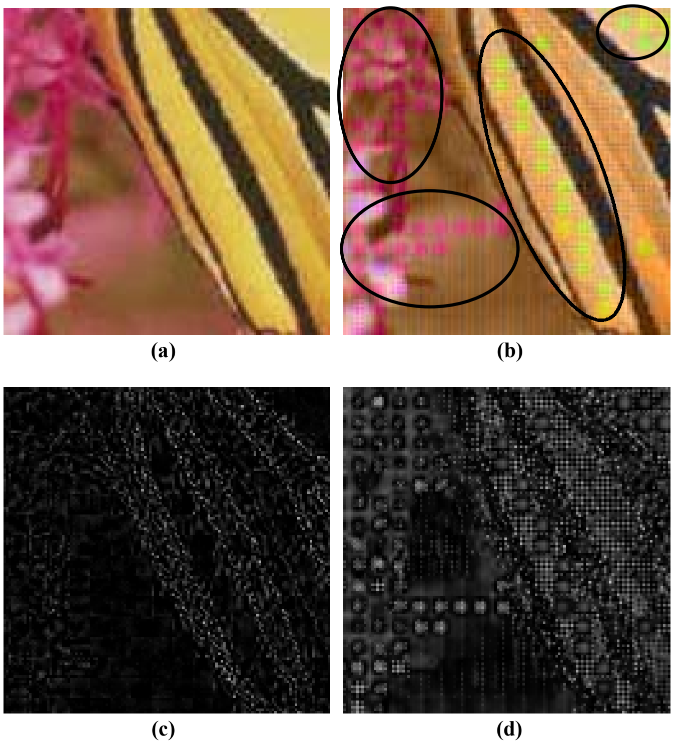


Fig. 3. Result of compressing CCD data directly at a ratio of 24:1. (a) Image obtained from conventional compression of the Bayer pattern CFA data. (b) Image obtained from compressing the Bayer pattern CFA data directly. The circles drawn on the image highlight the distorted areas. (c) Difference image between interpolated image and conventionally compressed image. (d) Difference image between interpolated image and directly compressed Bayer pattern CFA data.

This example illustrates that the Bayer patterned CFA data is not suitable for direct compression using JPEG. If the Bayer patterned CFA data is separated into the three primary color components, the red and blue components can be down-sampled into a compact rectangular array and compressed directly. The problem is to find a transformation of the quincunx green pixels into a suitable form for compression.

In the green component, every line has a missing pixel located at ``odd row, even column" and ``even row, odd column". Applying the DCT on this quincunx array yields transform coefficients that correctly indicate a predominantly high frequency image. On the other hand, applying the DCT on the interpolated green component yields high frequency transform coefficients of lower magnitude. Applying standard quantization matrices and coding schemes on the two transformed images result in a larger coded size for the quincunx array. We see that applying JPEG on the quincunx array results in lower compression ratios compared to applying JPEG on an interpolated quincunx array, with the same quality setting. To obtain good compression ratios, transformation of the green quincunx array (into a rectangular array) is necessary before compression.

III. Existing Solutions

Two methods have been proposed in the literature that applies pre-processing using the size advantage of the data before compression [2], [3]. In both methods, the authors describe transformations that operate on the Bayer patterned CFA data before compression. Lee and Ortega [2] use a reversible transformation that maps the pixel information from the Bayer patterned CFA data into another range. The mapping rotates the original interlaced array into a rhombus, packing the data together. However, the shape of the data to be compressed after transformation is not rectangular and is not suitable for direct application of JPEG. The authors suggest the use of a shape-adaptive DCT. However, using shape-adaptive DCT increases the algorithm complexity, making it less attractive. If the shape-adaptive DCT is not used, then additional information which is not part of the JPEG standard must be provided in the header files of the image. Such information would be the shape of the compressed array.

After processing, the number of pixels in the transformed green component before compression is $0.5(N^2+16N)$ and not $0.5N^2$. This increase is not significant when the size of the CFA data is high. If the size is small, the use of their method increases the bandwidth requirement. For example, if the original CFA data size is 512×512, the increase in data is 3.125%. However, if the CFA data size is 128×128, then the increase in data is 12.5%.

Toi and Ohita [3] apply Subband decomposition [7] to compress the CFA data by using a non-separable two-dimensional diamond filter to process the quincunx green array. The sub-bands are then encoded for optimum rate-distortion. Reconstruction is carried out by decoding, synthesizing, and interpolating into the full color image. The coefficients of the filters used in the analysis and synthesis are all integer-valued allowing simplicity in the hardware implementation.

IV. Proposed Compressions Schemes

We propose two compression algorithms that take into account the Bayer pattern [8] and human visual system [9]. The first stage is to convert the CFA data from the RGB color space to the YCbCr color space. The next stage is to separate the luminance and chrominance components. The chrominance component is then compressed with JPEG. The quincunx luminance component is transformed into a rectangular array and then compressed. In our work we used JPEG compression instead of JPEG2000 as the former is still the most widely used standard.

A. Color Space Conversion

As in JPEG, the CFA data is transformed from the RGB color space into the YCbCr color space. We do so by using the similarity between the Bayer pattern and the sub-sampled 4:2:0 YCbCr format. In the Bayer pattern, there are insufficient pixels to carry out the conversion for each location. Thus, conversion is carried out on blocks of four pixels [2]. Each block contains two green, one red, and one blue pixel as illustrated in Figure 4.

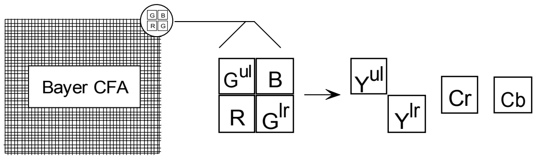


Fig. 4. Blocks of four pixels used in the color space conversion. The luminance and chrominance components are separated into three arrays.

The equations used in the color space conversion are modified to account for the insufficient pixels and is given by:

$$\begin{bmatrix} Y^{ul} \\ Y^{lr} \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 128.6 & 0 & 25 & 65.5 \\ 0 & 128.6 & 25 & 65.5 \\ -37.1 & -37.1 & 112 & -37.8 \\ -46.9 & -46.9 & -18.2 & 112 \end{bmatrix} \begin{bmatrix} G^{ul} \\ G^{lr} \\ B \\ R \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 128 \\ 128 \end{bmatrix} \qquad (2)$$

The color space conversion leaves the luminance component in a quincunx pattern but the chrominance components in a rectangular grid. The chrominance components are immediately compressed via JPEG.

B. Method 1 - Structure Conversion

In our first method, we propose to transform the quincunx array to a rectangular array through a structure conversion process. The simplest transformation of the quincunx grid is by merging the columns, where all even columns are shifted one pixel to the left and all zero columns removed. The disadvantage is that this transformation generates false high frequencies. To reduce the generation of high frequency we used de-interlacing techniques. The quincunx grid can be visualized as two interlaced frames of a scene and by de-interlacing, a smoother image is obtained. Such a process is described in [10] and illustrated in Figure 5.

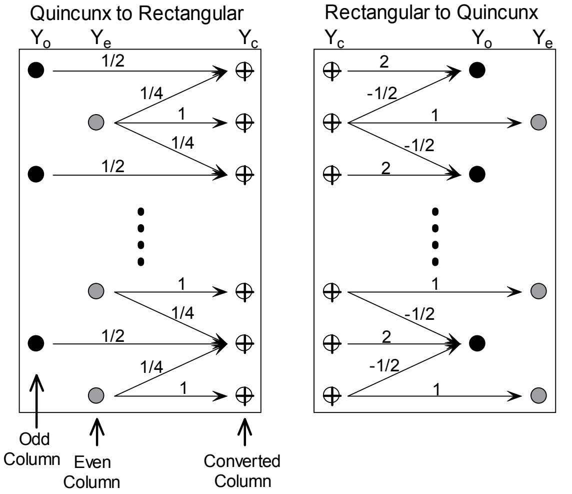


Fig. 5. **Weighted Structure conversion adapted from [10], Here Y_o are odd, Y_e are even and Y_c are converted columns.**

In Figure 5,● denotes pixels obtained from odd columns, while ◍ denotes pixels obtained from even columns of the quincunx grid, and ⊕ denotes the pixels of the converted columns in the rectangular grid. In the left box of Figure 5, we illustrate the conversion of a pair of odd and even column, into one converted column. Likewise, in the right box of Figure 5, the converted columns are transformed back into two interlaced columns.

The transformation into the converted columns can be reduced to low-pass filtering followed by down-sampling of columns. The filter used has an impulse response:

$$h_d[m,n]=\frac{1}{4}\begin{bmatrix}0 & 0 & 1\\ 0 & 2 & 4\\ 0 & 0 & 1\end{bmatrix} \qquad (3)$$

After down-sampling by 2, the array is transformed into a rectangular grid of dimensions of $0.5N^2$ and then compressed using JPEG. Although high frequency contents in the horizontal direction remain, high frequency in the vertical direction is reduced. For the recovery of the original data after de-compression, the converted grid is up-sampled by 2 and processed through a filter with an impulse response:

$$h_u[m,n]=\frac{1}{4}\begin{bmatrix}0 & -1 & 0\\ 2 & 4 & 0\\ 0 & -1 & 0\end{bmatrix} \qquad (4)$$

The filtered output is quincunx sampled to obtain the original data. The final step is the inverse of the color-space conversion in Section IV-A followed by full color interpolation on the reconstructed CFA data.

C. Method 2 - Structure Separation

In our second method, we propose to separate the quincunx grid into two rectangular grids. One grid contains all even (row and column position) pixels and the other contains all odd pixels. Before separation, the quincunx grid is low-pass filtered to limit aliasing. After experimentation with different 2-D filters [7], [11], [12], the impulse response of the non-separable diamond filter [13] used is:

$$h_{lp}[m,n]=\frac{1}{64}\begin{bmatrix}0 & -1 & 0 & -1 & 0\\ -1 & 0 & 10 & 0 & -1\\ 0 & 10 & 32 & 10 & 0\\ -1 & 0 & 10 & 0 & -1\\ 0 & -1 & 0 & -1 & 0\end{bmatrix} \qquad (3)$$

After filtering, the data is separated into odd and even components as in Figure 6.

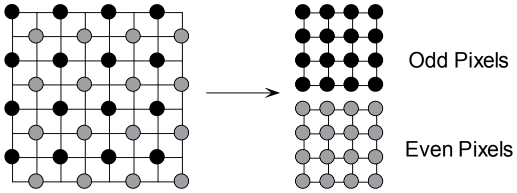


Fig. 6. **Structure separation of the quincunx array into two rectangular arrays. Here, black represents the odd row, odd column pixels, and gray represents the even row, even column pixels.**

After the quincunx array transformed into two rectangular arrays, JPEG compression is immediately applied to each array. As before the number of pixels compressed remains the same as the number of green pixels in the CFA data. Reconstruction of the image starts with de-compression followed by the merging of the odd and even arrays. The inverse of the color-space conversion in Section IV-A is applied followed by full color interpolation on the reconstructed CFA data.

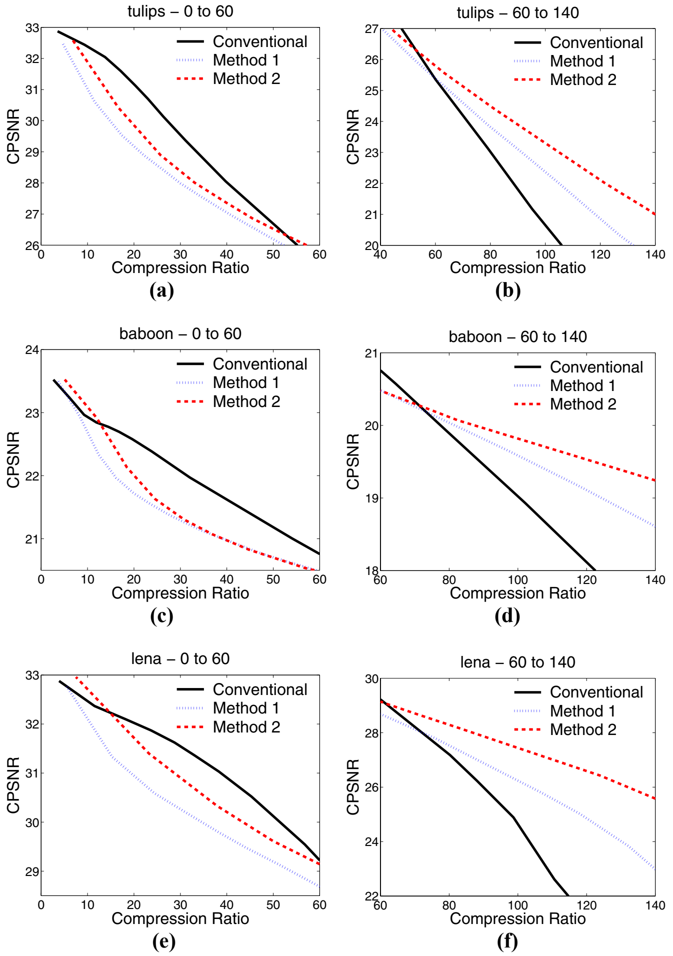
D. Compression Results

Simulations were carried out on twelve 24-bit color images of size 384×256 and 512×512: Lighthouse, Statue, Sail, Baboon, Monarch, Sails, Serrano, Airplane, Tiffany, Lena, Peppers, and Tulips. These images were converted into CFA data by sampling the pixels at locations determined by the Bayer pattern. The interpolation scheme used here is the bilinear method. Other interpolation methods were used and the results are discussed in Section V.

Of the 12 images tested, results for three are provided in this section. The images were chosen based on image content and they were: Tulips, Baboon, and Lena. The Lighthouse image contains a large amount of high frequency content. The Baboon image contains high frequency textures. The Lena image contains mostly low frequencies.

As seen in Figure 7, the proposed compression algorithms allow higher quality at high compression ratios (80:1 - 140:1).

At low compression ratios (up to 10:1), gains in quality is image content dependent and our scheme works well with images that are highly complicated or extremely smooth.



(a)

(b)

(c)

(d)

(e)

(f)

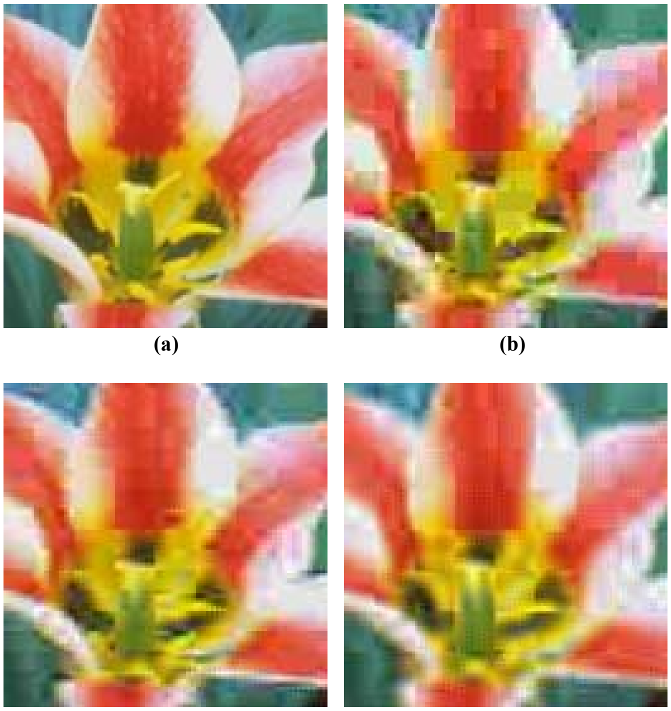
Fig. 7. CPSNR vs. compression ratios: Tulips image, (a) compression ratio up to 60:1, (b) compression ratio from 60:1 up to 140:1. Baboon image, (c) compression ratio up to 60:1, (d) compression ratio from 60:1 up to 140:1. Lena image, (e) compression ratio up to 60:1, (f) compression ratio from 60:1 up to 140:1.

We provide visual comparisons of the compression schemes in Figure 8. The conventional compression scheme (Figure 8(b)) contains severe blocking artifacts. In Figure 8(c), there is also severe blocking artifacts but is less pronounced then in the conventional method. In Figure 8(d), blocking artifacts are also present but is less severe compared to the other two schemes. Overall, the method of structure separation provides the best image quality and is verified by the highest CPSNR value of 25.06dB, a gain of 1.27dB over the conventional compression scheme.

V. Full Color Interpolation

There are many full color interpolation schemes in the literature [6], [14-19] some of which perform better than others. In this paper we have used three interpolation schemes. These schemes perform differently depending on compression ratio and image contents. In this section we study the effects of these factors on the selected interpolation algorithms.

The methods that will be discussed are: Bilinear, Cubic, and a second order interpolation scheme [14] (referred to as Laplacian in this paper). With these methods, we have interpolation schemes that represent the simple to the complicated in terms of computational complexity and provide increasing levels of quality. The bilinear method uses linear averaging to compute a pixel value. It is simple and computationally efficient while providing reasonable image quality. However, it ignores the effect of gradients, edges, and correlation between neighboring pixels and correlation between color planes. The cubic method fits a third order polynomial to compute the missing pixels. The advantage in this scheme is that the interpolated values are continuous (smooth with no abrupt transitions) over the entire range and gives improved image quality over the bilinear scheme. However, the cubic method does not use available information such as correlation between pixels or correlation between color planes. The Laplacian method uses gradients, edge information and rudimentary color plane correlation to compute the missing pixels. In the next subsections we will discuss the effects of interpolation on compression and image content on interpolation in turn.



(a)

(b)

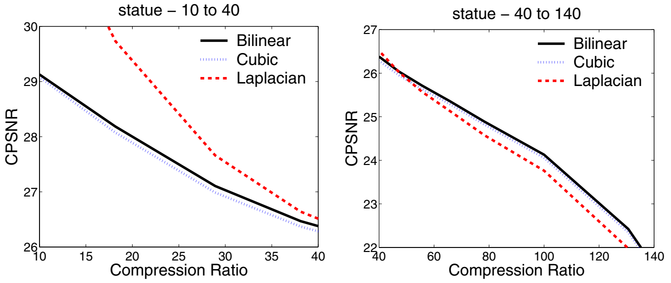
(c)

(d)

Fig. 8. Tulips images at a 70:1 compression ratio: (a) Uncompressed, (b) Conventional - 23.79dB, (c) Method 1 - 24.45dB, (d) Method 2 - 25.06dB.

A. Interpolation and Compression

Different full color interpolation schemes affect the results obtained from our proposed compression schemes. We use the image Statue and the compression scheme structure separation as an illustration. In Figure 9, Laplacian interpolation outperforms the other interpolation schemes at lower compression ratios. When the compression ratio is increased beyond 40:1, the simpler bilinear method produces images of higher quality. The degree of improvement varies and is approximately 0.5dB. The increase at high compression ratios is modest, but this scheme has the advantage of low computationally complexity. In summary, low compression ratios favor higher order interpolations schemes while high compression ratios favor simple interpolation schemes.



(a) (b)

Fig. 9. CPSNR vs. Compression ratios: Statue image, (a) compression ratio up to 40:1, (b) compression ratio from 40:1 up to 140:1.

B. *Interpolation and Image Content*

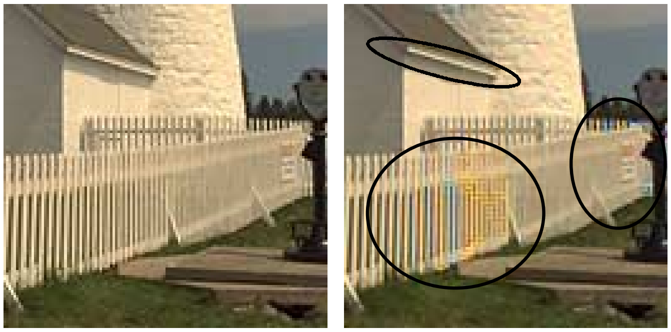
The Laplacian scheme provides the highest quality at low compression ratios, but improvement in image quality depends on image contents. From Table I, images which contain mostly low frequency components or high frequency textures have lower CPSNR gains. Images containing both low and high frequency components with well defined edges exhibit greater improvement in quality. The Tiffany image has the lowest gain, with a 2% improvement from the Laplacian to the bilinear method. Since the image is smooth with very little edges, the use of edge directed interpolation does not offer much in gain. The Baboon image gains only 8% in CPSNR as it contains high frequency texture. The Lighthouse image gains significantly with a 19.5% increase. This image contains well defined edges making it a good candidate for edge directed interpolation schemes. The higher order interpolation scheme does not provide very high gains in images with low frequency components and high frequency textured images. In these cases, it might be advantageous to use the simple interpolation scheme because of the low computational complexity.

TABLE I
UNITS AND CORRESPONDING SYMBOLS

IMAGE	BILINEAR	CUBIC	LAPLACIAN	INCREASE
LIGHTHOUSE	25.7687	25.4323	30.7760	19.43
SAIL	27.9193	27.7665	31.3252	12.20
STATUE	29.6373	29.6470	34.1499	15.23
AIRPLANE	34.2519	34.8726	38.0180	11.00
BABOON	23.5525	23.3598	25.4254	7.95
LENA	33.2332	33.3998	35.0930	5.60
MONARCH	30.4586	31.0401	35.7828	17.48
PEPPERS	35.8590	36.2980	39.0180	8.81
SAILS	28.5197	28.7289	33.2986	16.76
SERRANO	23.9590	23.9308	25.9481	8.30
TIFFANY	33.6803	33.5773	34.3205	1.90
TULIPS	32.8954	33.7508	37.5540	14.16

VI. FALSE COLOR SUPPRESSION

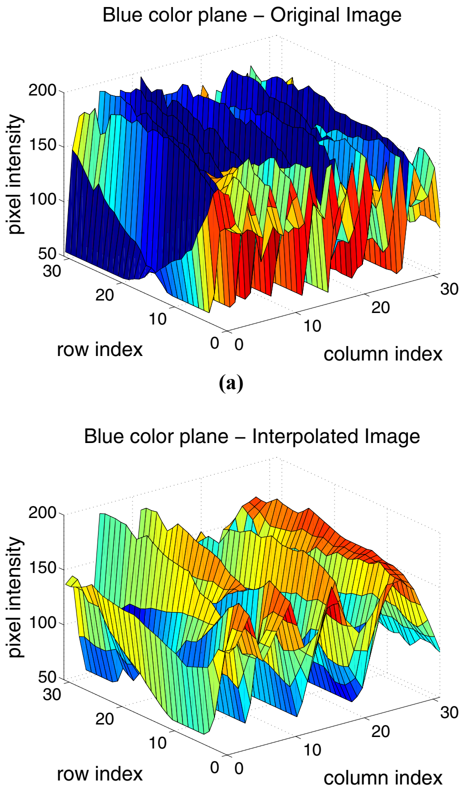
False colors are usually generated after full color interpolation is applied on the CFA data. Even with a higher order interpolation scheme, the effects of false colors are often visible (Figure 10(b)). Note that this example was generated from a full color image sub-sampled to form a Bayer pattern followed by full color interpolation.



(a) (b)

Fig. 10. Result of full color interpolation on Bayer CFA data: (a) Original image, (b) Laplacian interpolated image. The circled areas indicate where false colors are present.

In the context of full color interpolation of the Bayer patterned CFA, false colors are generated due to inadequate red and blue pixel intensities, and is apparent along edges. This can be illustrated by plotting the blue component as a surface as shown in Figure 11, where the original image is shown in Figure 11(a) and the interpolated image is shown in Figure 11(b).



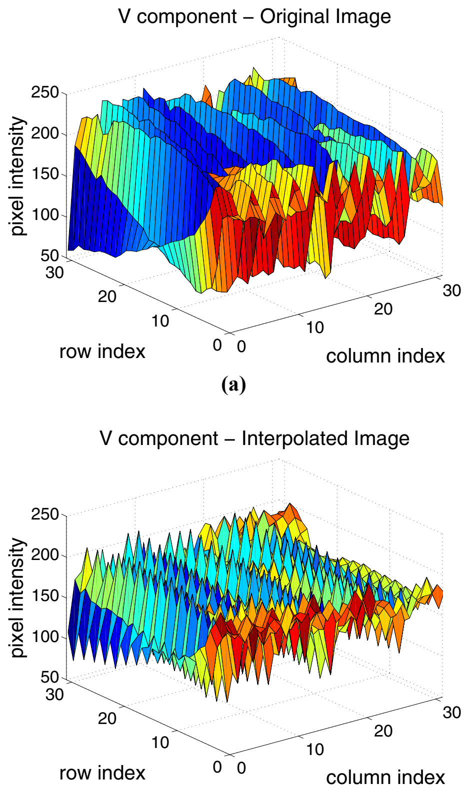
(a)

(b)

Fig. 11. Surface plot of the blue color plane in the false color area: (a) Original image, (b) Interpolated image.

In the surface plots, the X and Y axis represent column and row indices and the Z axis represent pixel intensity. When we compare the original and interpolated blue component, we find that there are several “missing columns” in the interpolated surface. This situation is also present in the red component but the location of these “missing columns” is not overlapping. This is caused by a spatial offset in the positions of the red and blue pixels in the Bayer CFA. This results in erroneous pixel intensities across the edges of an image and the generation of false colors.

This effect is better illustrated in the YUV color space. The V components are displayed in Figure 12. In Figure 12(a), the surface plot of the original image is similar to the plot of the blue component in Figure 12(a). The interpolated V component shown in Figure 12(b) is characterized by impulses and troughs, similar to salt and pepper noise. As before, these impulses are in different spatial locations compared to the U component. The differences in pixel intensities result in false colors. Noting the impulse like behavior, false colors can be suppressed using impulse noise removal algorithms.

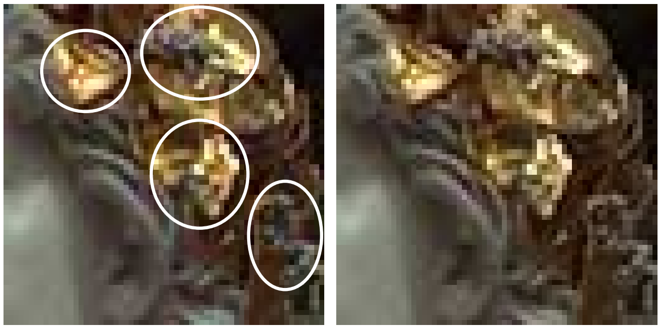


(a)

(b)

Fig. 12. Surface plot of the V color plane in the false color area: (a) Original image, (b) Interpolated image.

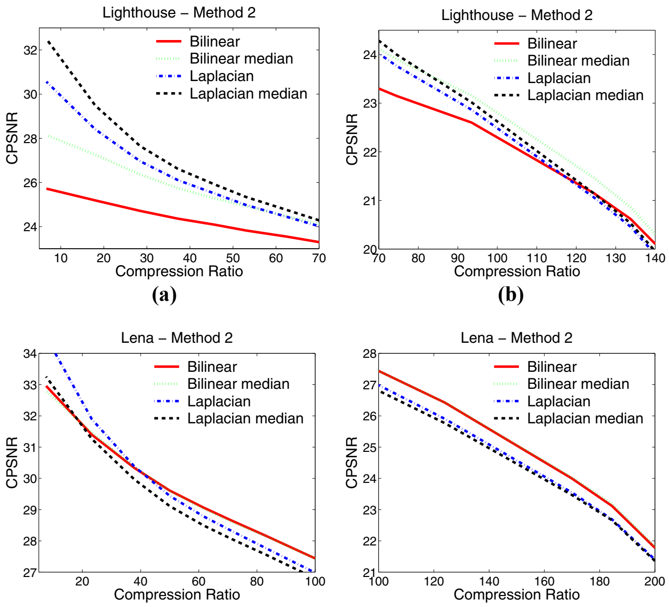
In general, adaptive impulse noise removal schemes will not do as well as non adaptive methods. The surface plot has too many large changes in pixel intensities and adaptive schemes will erroneously treat these impulses as edge details that are to be retained. Any false color suppression scheme implemented has to smooth the surface in order to suppress false color effectively. A simple method of suppressing the false color effects or impulse type behavior is by median filtering to the two chromaticity components, U and V [6]. The effectiveness of median filtering as a false color suppression tool is illustrated in Figure 13, where the false colors have been completely removed and CPSNR has increased by 1.2dB.



(a) (b)

Fig. 13. Result of median filtering applied on the reconstructed Bayer CFA data: (a) Laplacian interpolated image – 34.15dB, (b) Median filtered image – 35.34dB. The circled areas indicate where the false colors have been removed.

The use of median filtering as a post-processing operation after compression is presented in the plots of Figure 14. Image quality is greatly improved at low compression ratios for the Lighthouse image as illustrated by Figure 14(a). However, gains are not always available in other images. At low compression ratios, the Lena image does not have a similar level off improvement in image quality (Figure 14(c)). At high compression ratios, the images are blurred and the chrominance components do not exhibit impulse like peaks (Figures 14(b) and (d)). Median filtering in this case blurs the image and degrades it further.



(a) (b)

(a) (b)

Fig. 14. CPSNR vs. compression ratios: Lighthouse image, (a) compression ratio up to 70:1, (b) compression ratio from 70:1 up to 140:1, Lena image, (c) compression ratio up to 100:1, (d) compression ratio from 100:1 up to 200:1.

VII. Simulation Results

Table II provides a comparison of the compression ranges and best compression scheme based on highest CPSNR achieved. In this table, "CR" refers to the compression range and "Type" refers to the compression scheme employed. Of the different variations, the following six algorithms yielded the best results:

1. CC-L Conventional compression, Laplacian interpolation.
2. CC-B Conventional compression, bilinear interpolation.
3. SS-L Structure separation, Laplacian interpolation.
4. SS-L-M Structure separation, Laplacian interpolation, median filtering.
5. SS-B Structure separation, bilinear interpolation.
6. SS-B-M Structure separation, bilinear interpolation, median filtering.

For example, for the Tulips image (last row of Table II), with a compression ratio up too 8.2:1, SS-L performs the best. With a compression ratio from 8.2:1 to 51.8:1, CC-L performs the best. Finally for compression ratios above 51.8:1, SS-B-M performs the best. From the table, we see that there is no single compression scheme that is universally superior.

The proposed compression schemes perform well at the two ends of the compression scale. This observation is important when we consider present day consumer digital cameras where compression ratios are kept low to retain high image quality. The compression ratios employed usually extend up to 16:1 [20]. Our proposed algorithms have an advantage as it performs better than the conventional scheme through most of this range.

At higher compression ratios, our schemes again have an advantage. The images obtained at such high compression ratios are subjectively of higher quality with fewer artifacts. This is a result of the reduced data size that is to be manipulated compared to the conventional scheme. This improvement in image quality is important in applications that place emphasis on high compression ratios rather than on quality. Such systems would be wireless systems and low bandwidth video conferencing.

VIII. Complexity Comparisons

In this section we study the computational complexity of the compression schemes based on an $M\times N$ data array. We consider only the use of bilinear interpolation as it is computationally efficient and performs well at high compression ratios.

We first examine the number of operations required in color space conversions. In the conventional case, 3 multiplications and 2 additions (with subtractions considered as additions) are required for each luminance pixel. Each chrominance pixel requires 3 multiplications and 3 additions. With 3*MN* pixels in a full color image, the total number of operations required for forward and backward color conversion is 18*MN* multiplications and 16*MN* additions. In our proposed schemes, each luminance pixel requires 3 multiplications and 2 additions, while each chrominance pixel requires 4 multiplications and 4 additions. With 0.5*MN* pixels in the luminance component, and 0.25*MN* pixels in the chrominance component, the total number of operations required for forward and backward color conversion is 7*MN* multiplications and 6*MN* additions.

Next we determine the total number of operations required in the pre and post-processing stages. In structure conversion, an anti-aliasing filter with 13 non-zero coefficients is applied on the quincunx data. Thus, 13 multiplications and 12 additions are required for every processed pixel (except at the edges). With pre and post-processing, we have a total of 26*MN* multiplications and 24*MN* additions. In structure separation, the de-interlacing filter has 4 non-zero coefficients. Each empty pixel location in the quincunx array requires 1 multiplication, while every non empty pixel location requires 3 multiplications and 2 additions. With pre and post-processing, we have a total of 4*MN* multiplications and 2*MN* additions.

We finally determine the total number of operations required during compression. During compression, every pixel requires 2 multiplications and 1 addition ignoring the operations required in the entropy coding and decoding. The multiplications are obtained from the use of the DCT coefficients and normalization factors, while the addition comes from the level shifting operations. With compression and de-compression, the total number of operations in the conventional case is 12*MN* multiplications and 9*MN* additions, while in the proposed case it is 4*MN* multiplications and 2*MN* additions.

In the conventional case, the image is reconstructed into the full color image by interpolating the chrominance component. Assuming that bilinear interpolation is used here, there are 2 multiplications and 1 addition for each pixel. Since there are 2 chrominance components and 0.75*MN* missing pixels in each component, the total number of operations is 3*MN* multiplications and 1.5*MN* additions.

TABLE II
Best Algorithm Over Different Intervals Of Compression

Image	Type	CR	Type	CR	Type	CR	Type	CR	Type
Lighthouse	SS-L-M	16.4:1	CC-L	61.8:1	SS-L-M	82.2:1	SS-B-M		
Sail	SS-L-M	10.6:1	SS-B-M	67.3:1	SS-B-M				
Statue	SS-L-M	64.8:1	CC-L	49.8:1	CC-B	62.7:1	SS-B-M		
Airplane	SS-L	14.8:1	CC-L	16.3:1	CC-B	57.8:1	SS-B-M		
Baboon	SS-L	11.3:1	CC-L	19.2:1	CC-B	62.2:1	SS-B-M		
Lena	SS-L	15.8:1	CC-L	45.5:1	CC-B	61.5:1	SS-B	102.6:1	SS-B-M
Monarch	SS-L	62.6:1	CC-B	68.6:1	SS-B-M				
Peppers	SS-L	10.3:1	CC-L	49.6:1	SS-B	61.1:1	SS-B-M		
Sails	SS-L-M	7.6:1	CC-L	44.4:1	CC-B	63.9:1	SS-B-M		
Serrano	SS-L	19.9:1	CC-L	40.4:1	SS-B-M				
Tiffany	SS-L	25.2:1	CC-B						
Tulips	SS-L	8.2:1	CC-L	51.8:1	SS-B-M				

The total computational requirements are tabulated in Table III. From the table we find that the method of structure conversion requires the most operations among the three. However, it should be noted that the $24MN$ multiplications required by the filtering operations can be implemented as shifts and adds, further reducing the computational complexity. This is a result of the integer valued coefficients used in the filter. The method of structure separation provides the lowest computational complexity of the three methods. Similar to structure conversion, the computational complexity is lowered further with the realization that the coefficients of the de-interlacing filters are integer valued and can be implemented with shifts and adds.

TABLE III
COMPUTATIONAL COMPLEXITY COMPARISON

COMPRESSION SCHEME	MULTIPLICATIONS	ADDITIONS
CONVENTIONAL	$18N^2$	$13N^2$
STRUCTURE CONVERSION	$15N^2$	$10N^2$
STRUCTURE SEPARATION	$37N^2$	$32N^2$

IX. CONCLUDING REMARKS

In this work, we have studied the feasibility of compressing the CFA data directly before full color interpolation. Our proposed algorithms work by processing and compressing the CFA data before interpolation and taking advantage of the reduced size of the CFA data. This allows us to obtain significant gains in image quality. Furthermore, our proposed schemes use the standard JPEG compression algorithm, simplifying setup and hardware requirements.

The proposed methods lead to higher image quality. The improvement in the image quality occurs at low and high compression ratios. The proposed methods allow for high compression ratios that conventional compression is unable to achieve. These methods allow bandwidth reduction where images or videos are transmitted over a communications link while maintaining desired quality. It is also possible to send high quality images or videos with fixed bandwidth requirements.

The performance of the interpolation algorithms is dependent on image contents. At low compression ratios, higher order interpolation techniques (such as Laplacian) yield greater increases in image quality. At high compression ratios, the higher order interpolation schemes lower the image quality. When we take into account the increase in system complexity, the bilinear interpolation scheme provides the best trade off in terms of image quality and computational complexity.

We also investigated the generation of false colors when CFA data is interpolated. These false colors are caused by inadequate red and blue pixels and their spatial offset. Full color interpolation algorithms are unable to compensate for these problems and result in edge misalignment. To mask false colors, we applied median filtering as a post-processing operator. An increase in image quality is obtained at low compression ratios whereas degraded image quality is obtained at high compression ratios. Median filtering is content dependent and works well with images that have well defined edges and not with images characterized by low frequency content or high frequency textures.

The method of structure separation has low complexity (accounting for the use of integer valued coefficients) and combined with the resulting image quality, structure separation is highly attractive.

Finally, our proposed compression schemes have been applied only to the compression of still images, but adaptation to video should not be difficult. By applying these methods to video sequences it is expected that either frame rates or quality can be increased while maintaining the other.

REFERENCES

[1] B.E. Bayer, "Color Imaging Array," U.S. Patent 3,971,065, 1976.

[2] S.Y. Lee and A. Ortega, ``A novel approach of image compression in digital cameras with a Bayer color filter array," *IEEE Int. Conf. Image Processing 2001*, vol. 3, pp. 482-485, Oct 2001.

[3] T. Toi and M. Ohita, ``A subband coding technique for image compression in single CCD cameras with Bayer color filter arrays" *IEEE Trans. Consumer Electronics*, vol. 45, no. 1, pp. 176-180, Feb 1999.

[4] Bitflux : Mobile "28.07.03 The worldwide camera phone market," retrieved August 27, 2003, from http://blog.bitflux.ch/c15.html.

[5] Cellular Online "Wireless Camera Phone Market Set For Boom," retrieved August 27, 2003, from http://www.cellular.co.za/news_002 /112602- wireless_camera_phone_market_set.htm.

[6] J. Mukherjee, M.K. Lang and S.K. Mitra, ``Color demosaicing in yuv color space," *IASTED Con.*, Malaga, Spain, pp. 96-101, Sep 2002.

[7] K. Kotmasu and K. Sezaki, ``Reversible subband coding of images," *SPIE Symp. Visual Communication and Image Processing*, vol.2501, pp. 676-684, May 1995.

[8] C.C. Koh, S.K. Mitra, ``Compression of bayer patterned color filter array data," *IEEE Int. Conf. Image Processing*, submitted for publication, Sep 2003.

[9] B. Wandell. *Foundations of Vision*, Sinauer Associates, 1995.

[10] K. Hisakazu, M. Shogo, T. Ishida and T. Kuge, ``Reversible conversion between interlaced and progressive scan formats and its efficient implementation," *European Signal Processing Conference*, Paper 448, Sep 2002.

[11] S. Bagchi and S.K. Mitra, ``Nonseparable 2D FIR filter design using nonuniform frequency sampling," *SPIE Symp. Electronic Imaging*, San Jose, pp. 104-115, Feb 1995.

[12] M. Vetterli, J. Kovacevic, ``Perfect reconstruction filter banks for HDTV representation and coding," *Image Communication Journal, Special issue on HDTV*, vol. 2, no. 3, pp. 349-364, Oct 1990.

[13] M.K. Tchobanou, Dept. of Electrical Physics, Moscow Power Engineering Institute, Moscow, RUSSIA, Private Communication, 2003.

[14] J.F. Hamilton and J.E. Adams, ``Adaptive color plane interpolation in single sensor color electronic camera," U.S. Patent 5,629,734, 1997.

[15] R. Kimmel, ``Demosaicing: Image reconstruction from color CCD samples," *IEEE Trans. Image Processing*, vol. 8, issue 9. pp. 1221-1228, Sep 1999.

[16] S.C. Pei, I.K. Tam, ``Effective color interpolation in CCD color filter array using signal correlation," *IEEE Int. Conf. Image Processing*, vol. 3, pp. 10-13, Sep 2000.

[17] B.K. Gunturk, Y. Altunbasak and R. Mersereau, ``Color plane interpolation using alternating projections," *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, vol. 4, pp. 3333-3336, May 2002.

[18] T. Kuno and H. Sugiura, `` New interpolation method using discriminated color correlation for digital still cameras," *IEEE Trans. Consumer Electronics*, vol. 45, No. 1, pp. 259-267, Feb 1999.

[19] C. Weerasinghe, I. Kharitonenko and P. Ogunbona, ``Method of color interpolation in a single sensor color camera using green channel separation," *IEEE Int. Conf. Acoustics, Speech and Signal Processing* 2, vol. 4, pp. 3233-3236, May 2002.

[20] Frogprints, ``Compression and File Size," retrieved January 28, 2003, from http://www.frogprints.co.nz/help/compression.cfm.