Available online at www.sciencedirect.com

SCIENCE DIRECT®

Real-Time Imaging 11 (2005) 370–377

www.elsevier.com/locate/rti

Real-time lossless compression of mosaic video sequences

Lei Zhang[a,*], Xiaolin Wu[a,1], Paul Bao[b]

[a]*Department of Electrical and Computer Engineering, McMaster University, Hamilton, Ontario, Canada L8S 4K1*
[b]*School of Computer Engineering, Nanyang Technological University, Singapore 639798, Singapore*

Available online 10 August 2005

Abstract

This paper presents a simple, fast coding technique for lossless compression of mosaic video data. The design of a video codec needs to strike a balance between the compression performance and the codec throughput. Aiming to make the encoding throughput high enough for real-time lossless video compression, we propose a hybrid scheme of inter and intraframe coding. Interframe predictive coding is invoked only when the motion between adjacent frames is modest and a simple motion compensation operation can significantly improve the compression performance. Otherwise, still frame compression is performed to keep the complexity low. Experimental results show that the proposed scheme achieves higher lossless video compression ratio than existing methods such as JPEG-LS and JPEG-2000.
© 2005 Elsevier Ltd. All rights reserved.

1. Introduction

Digital video has become a prevalent and rich information source in all walks of life. But unlike film cameras, which can capture the red, green and blue color channels simultaneously, most digital video cameras record color signals by sub-sampling color bands in particular mosaic patterns, such as the popular Bayer color filter array (see Fig. 1) [1]. At each pixel, only one of the three primary colors is captured, and the other two missing components are interpolated via color demosaicking to reconstruct the full color image [2–15]. In the current design of digital video cameras, color demosaicking is carried out first. The demosaicked video is then subjected to lossy compression before being stored or transmitted [4,9].

In this research we question the merits of the above workflow: color demosaicking followed by lossy compression, in overall system performance and ultimate video quality. In the current design, constrained by low cost, throughput bottleneck and power conservation, the color demosaicking process is done inexpensively but also sub-optimally on camera. Unfortunately, the "demosaicking-first and lossy-compression-later" process is irreversible, and once done it will deny the opportunity to improve video quality by running state-of-the-art, albeit computationally more expensive, demosaicking algorithms on the original sensor data when ample computation resources are available and new demosaicking technology is developed in the future.

Compression of demosaicked video rather than of raw mosaic video directly can also be counterproductive. Color demosaicking triples the amount of raw data by interpolating red, green and blue channels. Ironically, a key task of color video compression is to decorrelate the interpolated color bands. This essentially attempts to reverse the demosaicking process. In other words, the current scheme of "demosaicking first and compression later" increases the algorithm complexity, reduces the compression ratio, and burdens the on-camera I/O bandwidth.

With these considerations we argue that raw mosaic color video data should be first compressed on camera without any loss or under a very tight bound on

*Corresponding author. Tel. +1 905 5259140.
E-mail addresses: johnray@mail.ece.mcmaster.ca (L. Zhang), xwu@mail.ece.mcmaster.ca (X. Wu).
[1]This work is supported by Natural Sciences and Engineering Research Council of Canada through NSERC-DALSA Industrial Research Chair in Digital Cinema.

1077-2014/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rti.2005.07.001

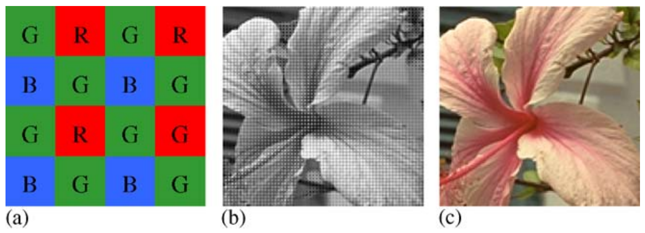


Fig. 1. (a) Bayer pattern; (b) an original mosaic image; and (c) the demosaicked color image.

compression errors (near-lossless coding). Color video demosaicking should be performed later, most likely off camera. This new workflow permits the raw mosaic video to be decoded without any loss, and then post-processed by the most suitable sophisticated joint spatiotemporal color demosaicking techniques to extract full color video of the best possible quality [11,13,14]. For high-end applications where the visual quality has paramount importance, this is particularly important. It is necessary to compress the raw mosaic video losslessly and store it on camera.

Some lossy and near-lossless compression methods for digital cameras with a color filter array have been proposed [16–20]. These schemes use techniques such as discrete cosine transform [20], subband coding by symmetric short kernel filter [19], format transformation followed by JPEG [17,18] and structural transformation followed by JPEG-LS [16] to achieve the data reduction. However, few literatures have been reported on the lossless compression of mosaic video sequences.

In this paper we propose a simple, fast hybrid scheme of inter and intraframe lossless compression for mosaic video data. The main goal is to make the encoder throughput high enough for real-time video coding, while maintaining the compression ratio at a competitive level. For each input frame to be encoded, we compute the intraframe difference of it and the interframe distance between it and the previous frame, which serve as the measurements in the adaptive switching between the two coding modes. If the interframe distance value is smaller than the intraframe difference value, the previous frame can give a better prediction of the current frame and thus a fast interframe coding technique is used. Otherwise, the interframe prediction accuracy is considered to be lower than the intraframe prediction and then the current frame is encoded by using an intraframe coding method. This hybrid strategy judiciously spends computation resources when simple temporal prediction can remove a bulk of data redundancy in time.

The paper is organized as follows. Section 2 presents an intraframe lossless compression scheme of mosaic images, while Section 3 presents the switch mechanism between the intra and interframe coding together with a fast interframe lossless coding method. Experimental results are reported in Section 4, and finally conclusion is drawn in Section 5.

2. Intraframe lossless coding method

The pixels in the current frame $\mathbf{F}_j$ are sampled with the Bayer pattern, as shown in Fig. 1(a). Most existing lossless image compression schemes developed for continuous-tone images are not suitable for mosaic images, since they assume a degree of smoothness of the image signal, which is not true for mosaic images. To avoid this problem we separate the raw red, green and blue interlaced data into three parts. For instance, the mosaic image

$$\mathbf{I}=\begin{array}{|cccccccc|}\hline G_{0,0} & R_{0,1} & G_{0,2} & R_{0,3} & G_{0,4} & R_{0,5} & G_{0,6} & R_{0,7}\\ B_{1,0} & G_{1,1} & B_{1,2} & G_{1,3} & B_{1,4} & G_{1,5} & B_{1,6} & G_{1,7}\\ G_{2,0} & R_{2,1} & G_{2,2} & R_{2,3} & G_{2,4} & R_{2,5} & G_{2,6} & R_{2,7}\\ B_{3,0} & G_{3,1} & B_{3,2} & G_{3,3} & B_{3,4} & G_{3,5} & B_{3,6} & G_{3,7}\\ G_{4,0} & R_{4,1} & G_{4,2} & R_{4,3} & G_{4,4} & R_{4,5} & G_{4,6} & R_{4,7}\\ B_{5,0} & G_{5,1} & B_{5,2} & G_{5,3} & B_{5,4} & G_{5,5} & B_{5,6} & G_{5,7}\\ G_{6,0} & R_{6,1} & G_{6,2} & R_{6,3} & G_{6,4} & R_{6,5} & G_{6,6} & R_{6,7}\\ B_{7,0} & G_{7,1} & B_{7,2} & G_{7,3} & B_{7,4} & G_{7,5} & B_{7,6} & G_{7,7}\\ \hline\end{array} \tag{2.1}$$

is divided into red and blue sub-images

$$\mathbf{R}=\begin{array}{|cccc|}\hline R_{0,1} & R_{0,3} & R_{0,5} & R_{0,7}\\ R_{2,1} & R_{2,3} & R_{2,5} & R_{2,7}\\ R_{4,1} & R_{4,3} & R_{4,5} & R_{4,7}\\ R_{6,1} & R_{6,3} & R_{6,5} & R_{6,7}\\ \hline\end{array} \qquad \mathbf{B}=\begin{array}{|cccc|}\hline B_{1,0} & B_{1,2} & B_{1,4} & B_{1,6}\\ B_{3,0} & B_{3,2} & B_{3,4} & B_{3,6}\\ B_{5,0} & B_{5,2} & B_{5,4} & B_{5,6}\\ B_{7,0} & B_{7,2} & B_{7,4} & B_{7,6}\\ \hline\end{array} \tag{2.2}$$

and green sub-image $\mathbf{G}$ that contains the remained green samples.

The green sub-image $\mathbf{G}$ should be first coded because the green channel has twice as many samples as the red/blue channel and hence it has higher spatial correlation. Once $\mathbf{G}$ is coded, it will be used as a reference in predictive coding of the red and blue channels through spectral correlation. However, the samples in $\mathbf{G}$ form a diamond grid, while all existing lossless image compression standards operate on square sample grid. To circumvent this problem, one can transform the green quincunx lattice into the rectangular lattice before compression. Many transforms are possible, such as separation, rotation, merge and reversible de-interlacer [17]. For simplicity, we split $\mathbf{G}$ into two rectangular parts:

$$\mathbf{G}_e=\begin{array}{|cccc|}\hline G_{0,0} & G_{0,2} & G_{0,4} & G_{0,6}\\ G_{2,0} & G_{2,2} & G_{2,4} & G_{2,6}\\ G_{4,0} & G_{4,2} & G_{4,4} & G_{4,6}\\ G_{6,0} & G_{6,2} & G_{6,4} & G_{6,6}\\ \hline\end{array} \qquad \mathbf{G}_o=\begin{array}{|cccc|}\hline G_{1,1} & G_{1,3} & G_{1,5} & G_{1,7}\\ G_{3,1} & G_{3,3} & G_{3,5} & G_{3,7}\\ G_{5,1} & G_{5,3} & G_{5,5} & G_{5,7}\\ G_{7,1} & G_{7,3} & G_{7,5} & G_{7,7}\\ \hline\end{array} \tag{2.3}$$

where $\mathbf{G}_e$ consists of the green pixels $\mathbf{G}_{even,even}$ and $\mathbf{G}_o$ consists of the pixels $\mathbf{G}_{odd,odd}$.

The resulting sub-image $\mathbf{G}_e$ can be coded using any of the existing lossless image codecs, such as JPEG-LS [21] and JPEG 2000 lossless mode [22]. Of course, one can also compress sub-image $\mathbf{G}_o$ independently of $\mathbf{G}_e$. This is, though, clearly suboptimal for that the correlation between $\mathbf{G}_o$ and $\mathbf{G}_e$ is totally ignored. Instead, we propose a predictive coding scheme of $\mathbf{G}_o$ based on $\mathbf{G}_e$. Referring to Fig. 2, we compute a prediction of each sample $\mathbf{G}_{odd,odd}$ in $\mathbf{G}_o$, denoted by $\hat{G}_{odd,odd}$, using its neighbors in $\mathbf{G}_e$. The residual signal $e_{odd,odd} = G_{odd,odd} - \hat{G}_{odd,odd}$ is losslessly coded so that $\mathbf{G}_\text{o}$ can be perfectly reconstructed from $\mathbf{G}_e$. Many interpolation methods can be used to predict $\mathbf{G}_o$ from $\mathbf{G}_e$, such as bi-linear interpolation, bi-cubic interpolation and the more sophisticated linear minimum mean square-error estimation (LMMSE) based interpolator [23]. In order for simplicity and fast implementation, we use the bi-linear interpolator so that $\hat{G}_{odd,odd}$ is computed as the average of its four nearest neighbors.

After the green sub-image $\mathbf{G}$ is compressed, the red sub-image $\mathbf{R}$ and blue sub-image $\mathbf{B}$ are to be compressed by exploiting the spectral correlation between the color channels. In particular, we compress the color difference signals $\mathbf{G}-\mathbf{R}$ and $\mathbf{G}-\mathbf{B}$ rather than the red signal $\mathbf{R}$ and blue signal $\mathbf{B}$ themselves. This is because the difference signal is much smoother than the individual color component signal. Interestingly, many of color demosaicking algorithms also assume the smoothness of the color difference signals in the color interpolation process [3–5,8,10,15]. In Figs. 3(a) and (c), the red and blue channels of a test image are shown, and they are compared with the green/red and green/blue color difference images placed in Figs. 3(b) and (d) respectively. It is evident that color difference signals have much lower energy than the individual color channels themselves.

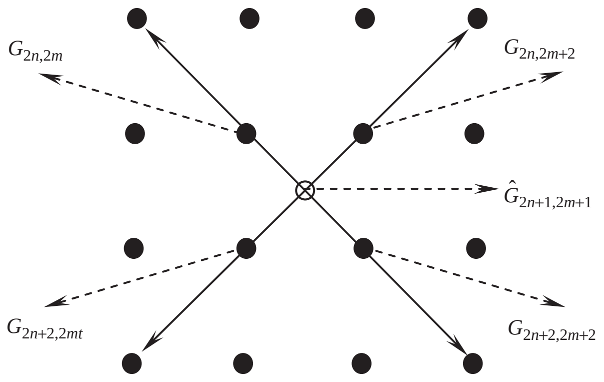


Fig. 2. $G_{odd,odd}$ is estimated from its available neighbors.

In order to code the color difference signals $\mathbf{G}-\mathbf{R}$ and $\mathbf{G}-\mathbf{B}$, we need to estimate the missing green values from the existing green samples at the pixels $R_{even,odd}$ and $B_{odd,even}$. Denote these estimates as $\hat{G}_{even,odd}$ and $\hat{G}_{odd,even}$ respectively. Fig. 4 shows the case that $\hat{G}_{even,odd}$ is to be interpolated by its neighbors. The case to interpolate $\hat{G}_{odd,even}$ is symmetrical. Being similar to computing $\hat{G}_{odd,odd}$, computing the estimates $\hat{G}_{even,odd}$ and $\hat{G}_{odd,even}$ can be implemented by using bi-linear or bi-cubic convolution but in horizontal and vertical directions, if encoder speed is of main concern, or by using more advanced methods such as [23] for higher estimation accuracy and hence better compression, if there is computation power to spare.

Denote by $\hat{\mathbf{G}}_R$ and $\hat{\mathbf{G}}_B$ the estimated green sub-images that consist of $\hat{G}_{even,odd}$ and $\hat{G}_{odd,even}$ respectively. The color difference images are now formed as

$$\mathbf{D}_R = \hat{\mathbf{G}}_R - \mathbf{R}, \quad \mathbf{D}_B = \hat{\mathbf{G}}_B - \mathbf{B} \tag{2.4}$$

$\mathbf{D}_R$ and $\mathbf{D}_B$ are then compressed by JPEG-LS as gray scale images. They are far more compressible than $\mathbf{R}$ and $\mathbf{B}$ due to spectral de-correlation between the color channels. Finally, the decoder can get the original mosaic samples $\mathbf{R}$ and $\mathbf{B}$ back from $\mathbf{D}_R$ and $\mathbf{D}_B$ and $\hat{\mathbf{G}}_R$ and $\hat{\mathbf{G}}_B$.

3. Interframe lossless coding scheme

In the presence of relative motions between adjacent frames, video compression technique of choice is interframe predictive coding, facilitated by motion estimation and compensation. Accurate motion estimation is, however, computationally intensive, and its real

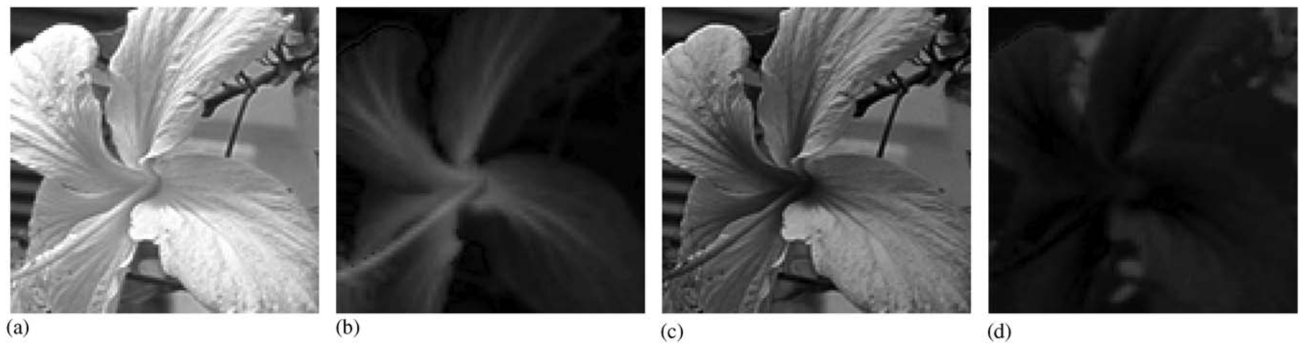


Fig. 3. (a) The red and (b) blue channels of a test image. (c) The green/red and (d) green/blue difference signals of the test image.

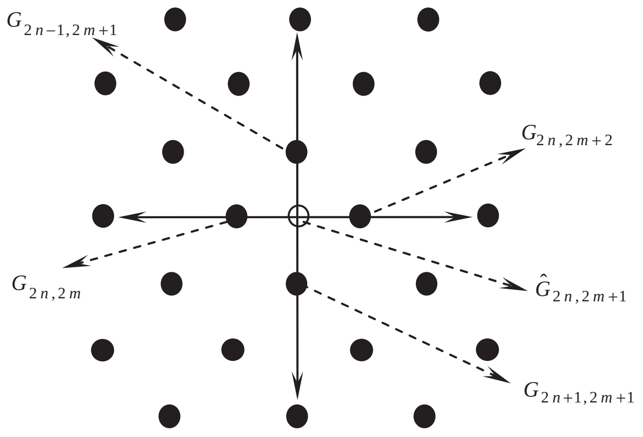


Fig. 4. $G_{even,odd}$ is estimated from its available neighbors.

time implementation is difficult and costly. This is why motion-based video coding standard MPEG was primarily developed for stored video applications, where the encoding process is typically carried out off-line on powerful computers [24]. The MPEG design is asymmetrical in terms of complexity, with the encoder being much slower than the decoder, because only the decoder needs to play back the compressed video in real time. For lossless coding of mosaic data on video cameras, the system requirement is exactly opposite: high encoder throughput is needed. For this reason we develop in this section an interframe coding scheme that requires only a small fraction of the cost of full-search motion estimation. In addition, we switch from interframe coding mode to intraframe coding mode if the former does not offer a significant coding gain.

3.1. Switch of coding mode

The selection between intraframe and interframe coding modes depends on the degree and complexity of the motion. To assess the motion, we first need to interpolate the missing green samples in frame $\mathbf{F}_{j-1}$. This can be efficiently done with a linear directional filtering method, for example, the second order Laplacian correction filter proposed by Hamilton and Adams [5].

Let us examine the case depicted by Fig. 5: a column and a row of alternating green and red samples intersect at a red sampling position, where the missing green value needs to be estimated. The symmetric case of estimating the missing green values at the blue sampling positions of the Bayer pattern can be handled in the same way. Denote the red sample at the center of the window by R_0. Its interlaced red and green neighbors in horizontal direction are labeled as R_i^h, $i \in \{-2, 2\}$, and G_i^h, $i \in \{-1, 1\}$ respectively; similarly, the red and green neighbors of R_0 in vertical direction are R_j^v, $j \in \{-2, 2\}$, and G_j^v, $j \in \{-1, 1\}$.

Let $\Delta_0 = G_0 - R_0$ be the unknown difference between green and red channels at the sample position of R_0. We

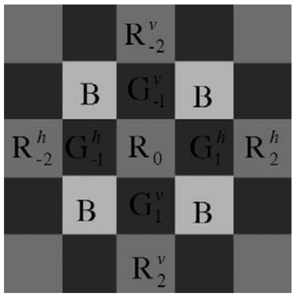


Fig. 5. A row and a column of mosaic data that intersect at a red sampling position.

first obtain an estimate of Δ_0, denoted by $\hat{\Delta}_0$, and then recover the missing green sample by $\hat{G}_0 \approx R_0 + \hat{\Delta}_0$. The reason for estimating the color difference signal $\Delta = G - R$ rather than the green signal G directly is that Δ is much smoother than G. Referring to Fig. 5, the horizontal and vertical differences between the green and red channels at R_0 can be estimated as

$$\Delta_0^h = \frac{1}{2}\left(G_{-1}^h + G_1^h\right) - \frac{1}{4}\left(2R_0 + R_{-2}^h + R_2^h\right) \tag{3.1}$$

$$\Delta_0^v = \frac{1}{2}\left(G_{-1}^v + G_1^v\right) - \frac{1}{4}\left(2R_0 + R_{-2}^v + R_2^v\right) \tag{3.2}$$

Note that all the filter coefficients are of form 2^{-n}, $n \in Z$, which makes the hardware implementation very fast. In [5], the second order gradient of red samples and the first order gradient of green samples are calculated respectively in horizontal and vertical directions. If the horizontal gradient is smaller than that of vertical, then $\hat{\Delta}_0 = \Delta_0^h$; otherwise $\hat{\Delta}_0 = \Delta_0^v$. This decision is to avoid the interpolation across image edges, which will affect the visual quality. For speed considerations we may omit these operations and simply let $\hat{\Delta}_0 = (\Delta_0^h + \Delta_0^v)/2$. Our experiments show that this has almost no effect on compress performance.

Denote the interpolated green channel of frame $\mathbf{F}_{j-1}$ as

$$\mathbf{G}_{j-1} = \begin{array}{|ccccc|} G^{j-1}_{0,0} & \hat{G}^{j-1}_{0,1} & G^{j-1}_{0,2} & \hat{G}^{j-1}_{0,3} & \cdots \\ \hat{G}^{j-1}_{1,0} & G^{j-1}_{1,1} & \hat{G}^{j-1}_{1,2} & G^{j-1}_{1,3} & \cdots \\ G^{j-1}_{2,0} & \hat{G}^{j-1}_{2,1} & G^{j-1}_{2,2} & \hat{G}^{j-1}_{2,3} & \cdots \\ \hat{G}^{j-1}_{3,0} & G^{j-1}_{3,1} & \hat{G}^{j-1}_{3,2} & G^{j-1}_{3,3} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array} \tag{3.3}$$

$\mathbf{G}_{j-1}$ consists of half the original green samples and half the interpolated samples. Denote the green channel of frame $\mathbf{F}_j$ as

$$\mathbf{G}_{j} = \begin{array}{|ccccc|} G^{j}_{0,0} & \times & G^{j}_{0,2} & \times & \cdots \\ \times & G^{j}_{1,1} & \times & G^{j}_{1,3} & \cdots \\ G^{j}_{2,0} & \times & G^{j}_{2,2} & \times & \cdots \\ \times & G^{j}_{3,1} & \times & G^{j-1}_{3,3} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array} \tag{3.4}$$

Let (d_x, d_y) be the motion vector between the two frames $\mathbf{G}_{j-1}$ and $\mathbf{G}_j$. In video compression, the estimation accuracy of (d_x, d_y) is one of the most important factors that affect the final compression performance. However, the heavy computation load of accurate motion estimation makes it impractical for our purposes. In this paper, we restrict $d_x, d_y \in \{-1, 0, 1\}$. To further streamline computations we determine d_x and d_y separately in x and y search windows:

$$\begin{cases} d_x = \arg \min\limits_{d \in \{-1,0,1\}} \sum\limits_{n+m=even}\sum \left| G^{\mathrm{j}}_{n,m} - G^{\mathrm{j-1}}_{n,m+d} \right| \\ d_y = \arg \min\limits_{d \in \{-1,0,1\}} \sum\limits_{n+m=even}\sum \left| G^{\mathrm{j}}_{n,m} - G^{\mathrm{j-1}}_{n+d,m} \right| \end{cases} \tag{3.5}$$

After (d_x, d_y) is computed, a distance between $\mathbf{G}_j$ and $\mathbf{G}_{j-1}$

$$D_1 = \sum_{n+m=even}\sum \left| G^{\mathrm{j}}_{n,m} - G^{\mathrm{j-1}}_{n+d_y,m+d_x} \right| \tag{3.6}$$

is computed to quickly evaluate the merit of interframe coding. A large D_1 value indicates that big relative motions or abrupt scene changes may occur in these frames. Consequently, interframe coding may be inefficient compared with intraframe coding. Therefore, an intraframe coding performance measure D_0 is needed, which will be compared against D_1 in the switch between intraframe and interframe coding modes. The measure D_0 is to predict, with minimum computation, the compressibility of the current green frame $\mathbf{G}_j$ by exploiting the intraframe redundancies only.

Referring to (3.6), for each pixel $G^j_{n,m}$ satisfying $n + m = even$, we predict it by only its two nearest neighbors within frame $\mathbf{F}_j$:

$$\hat{G}^j_{n,m} = (G^j_{n-1,m-1} + G^j_{n-1,m+1})/2 \tag{3.7}$$

The intraframe coding performance is then measured by

$$D_0 = \sum_{n+m=even}\sum \left| G^j_{n,m} - \hat{G}^j_{n,m} \right| \tag{3.8}$$

which reflects the prediction error. A small D_0 value implies that the intraframe correlation is high, and thus simple intraframe coding should suffice to compress $\mathbf{G}_j$. Finally, we have the following simple coding mode selection criterion:

```
If $D_1 \geqslant D_0$
    go to intraframe coding described in Section 2;
Else
    go to interframe coding described in Section 3.2;
End.
```

3.2. Interframe coding

Fig. 6 illustrates the whole encoding procedure of the proposed scheme. When $D_1 \geqslant D_0$, the intraframe algorithm described in Section 2 is used. If $D_1 < D_0$, $\mathbf{G}_j$ is to be compressed by using interframe coding with the help of $\mathbf{G}_{j-1}$. We define the difference image between $\mathbf{G}_j$ and $\mathbf{G}_{j-1}$ as

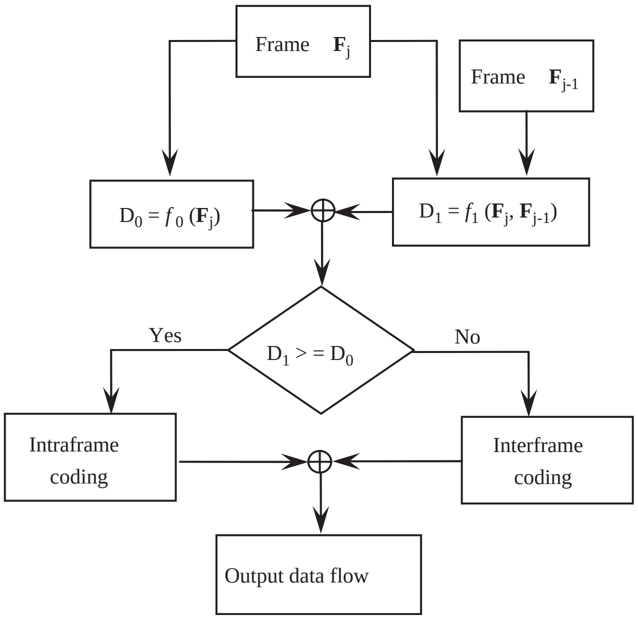


Fig. 6. Illustration of the encoding procedure.

$$\tilde{\mathbf{G}} = \begin{array}{|ccccc|} \tilde{G}_{0,0} & \times & \tilde{G}^{\mathrm{j}}_{0,2} & \times & \cdots \\ \times & \tilde{G}^{\mathrm{j}}_{1,1} & \times & \tilde{G}^{\mathrm{j}}_{1,3} & \cdots \\ \tilde{G}_{2,0} & \times & \tilde{G}^{\mathrm{j}}_{2,2} & \times & \cdots \\ \times & \tilde{G}^{\mathrm{j}}_{3,1} & \times & \tilde{G}^{\mathrm{j-1}}_{3,3} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array} \tag{3.9}$$

where

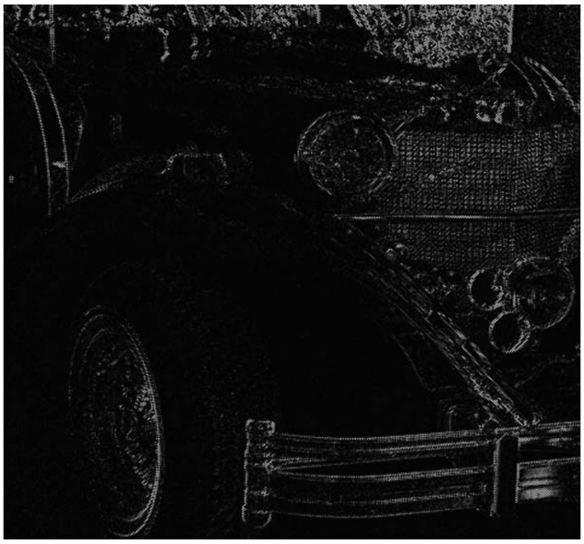
$$\tilde{G}_{n,m} = G^j_{n,m} - G^{j-1}_{n+d_y,m+d_x} \tag{3.10}$$

and d_x and d_y are computed in (3.5).

Now the lossless compression of $\mathbf{G}_j$ becomes the problem of entropy coding of $\tilde{\mathbf{G}}$. The residual terms in $\tilde{\mathbf{G}}$ are coded in raster scan order from left to right and top to bottom. After the interframe prediction, the samples of difference image $\tilde{\mathbf{G}}$ remain somewhat correlated. Large values of $\tilde{G}_{n,m}$ tend to register with edges, and the neighbors of $\tilde{G}_{n,m}$ with significant magnitude are also likely to have high magnitude, as shown by Fig. 7(a). This form of correlation can be exploited by applying context-based coding to $\tilde{\mathbf{G}}$ [25]. Specifically, the coding of $\tilde{G}_{n,m}$ is conditioned on the energy level in its neighborhood. Define the neighborhood energy of $\tilde{G}_{n,m}$ as

$$P_{n,m} = \left|\tilde{G}_{n-1,m-1}\right| + \left|\tilde{G}_{n-1,m+1}\right| + \left(\left|\tilde{G}_{n-2,m}\right| + \left|\tilde{G}_{n-2,m}\right|\right)/2 \tag{3.11}$$

We quantize $P_{n,m}$ into some conditioning contexts in which $\tilde{G}_{n,m}$ will be coded. Fig. 7(a) shows a difference image $\tilde{\mathbf{G}}$ computed from a video sequence and Fig. 7(b) plots the distribution of $P_{n,m}$ for this difference image.


(a)

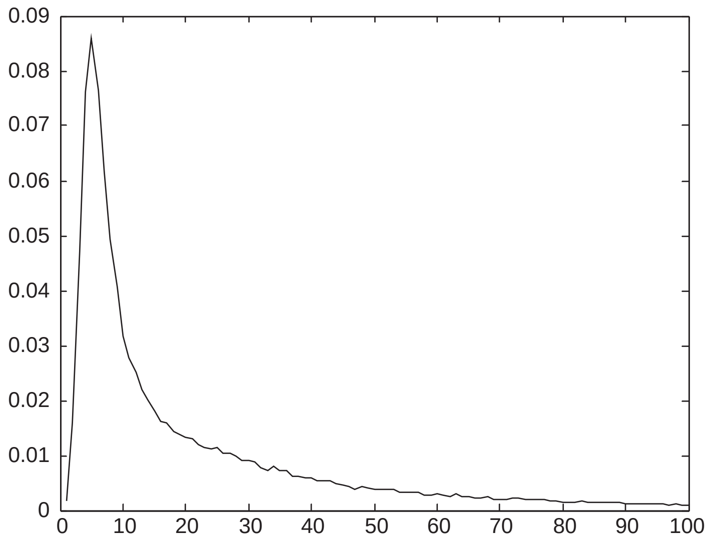

(b)

Fig. 7. (a) A difference image $\tilde{\mathbf{G}}$ for the test sequence shown in Fig. 8(a); (b) The distribution of $P_{n,m}$ for the difference image. The horizontal axis represents the value of $P_{n,m}$ and the vertical axis is the probability density.

We quantize $\tilde{G}_{n,m}$ into K contexts Θ_κ, $\kappa = 1, 2, ..., K$, by partitioning the range of $P_{n,m}$. The optimal partition can be obtained via dynamic programming. Empirically we find that the simple dyadic partition of $P_{n,m}$ works well in practice. Let

$$\begin{cases} \tilde{G}_{n,m} \in \Theta_K & P_{n,m} \geqslant 2^{K-1} \\ \tilde{G}_{n,m} \in \Theta_\kappa & 2^{\kappa-1} \leqslant P_{n,m} < 2^\kappa, \kappa = 2, 3, ..., K-1 \\ \tilde{G}_{n,m} \in \Theta_1 & 0 \leqslant P_{n,m} < 2 \end{cases} \tag{3.12}$$

After the green channel $\mathbf{G}_j$ of frame $\mathbf{F}_j$ is coded, we proceed to code the red and blue channels of $\mathbf{F}_j$, $\mathbf{R}_j$ and $\mathbf{B}_j$:

$$\mathbf{R}_j = \begin{array}{|ccccc|} \times & R^j_{0,1} & \times & R^j_{0,3} & \cdots \\ \times & \times & \times & \times & \cdots \\ \times & R^j_{2,1} & \times & R^j_{2,3} & \cdots \\ \times & \times & \times & \times & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}, \quad \mathbf{B}_j = \begin{array}{|ccccc|} \times & \times & \times & \times & \cdots \\ B^j_{1,0} & \times & B^j_{1,2} & \times & \cdots \\ \times & \times & \times & \times & \cdots \\ B^j_{3,0} & \times & B^j_{3,2} & \times & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array} \tag{3.13}$$

The definition of $\mathbf{R}_{j-1}$ and $\mathbf{B}_{j-1}$ for frame $\mathbf{F}_{j-1}$ is the same as (3.13). Since the compression of the blue channel is symmetrical to that of the red channel, it suffices to only discuss the coding of $\mathbf{R}_\mathrm{j}$.

Notice that if (d_x, d_y), the displacement between frame $\mathbf{F}_j$ and $\mathbf{F}_{j-1,}$ is equal to (0,0), then the pixel $R^j_{n,m}$ in $\mathbf{R}_j$ can be matched to an original red pixel $R^{j-1}_{n,m}$ in frame $\mathbf{F}_{j-1.}$ Otherwise, a reference sample of $R^j_{n,m}$, denoted by $\hat{R}^{j-1}_{n+d_y,m+d_x}$, has to be interpolated in frame $\mathbf{F}_{j-1.}$ However, because the sampling frequency of red channel is only half of that of green channel, the interpolation accuracy of missing red samples is much lower than that of missing green samples. Therefore, though $\hat{G}^{j-1}_{n+d_y,m+d_x}$ may be a good prediction of $G^j_{n,m}$, $\hat{R}^{j-1}_{n+d_y,m+d_x}$ may not predict $R^j_{n,m}$ well. With this consideration, we apply interframe coding to $\mathbf{R}_j$ only when $(d_x, d_y) = (0{,}0)$. Denote by

$$\tilde{\mathbf{R}} = \mathbf{R}_\mathrm{j} - \mathbf{R}_\mathrm{j\text{-}1} = \begin{array}{|ccccc|} \times & \tilde{R}_{0,1} & \times & \tilde{R}_{0,3} & \cdots \\ \times & \times & \times & \times & \cdots \\ \times & \tilde{R}_{2,1} & \times & \tilde{R}_{2,3} & \cdots \\ \times & \times & \times & \times & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array} \tag{3.14}$$

the difference red image, where $\tilde{R}_{n,m} = R^j_{n,m} - R^{j-1}_{n,m}$. The encoding procedure for $\mathbf{R}_j$ and $\mathbf{B}_j$ is simply as follows:

```
If $(d_x, d_y) = (0, 0)$
    Entropy coding of $\tilde{\mathbf{R}}$ and $\tilde{\mathbf{B}}$ as that of $\tilde{\mathbf{G}}$;
Else
    Intraframe coding of $\mathbf{R}_j$ and $\mathbf{B}_j$ as described in
    Section 2;
End
```

4. Experimental results

We first analyze the complexity of the proposed scheme. Suppose the size of an input mosaic frame is $N \times M$. For the intraframe algorithm proposed in Section 2, the input image is divided into four $N/2 \times M/2$ sub-images: $\mathbf{G}_e$, $\mathbf{G}_o$, $\mathbf{R}$ and $\mathbf{B}$. The employed interpolation technique is bi-linear convolution. Among the four sub-images, $\mathbf{G}_e$ is directly encoded by using the

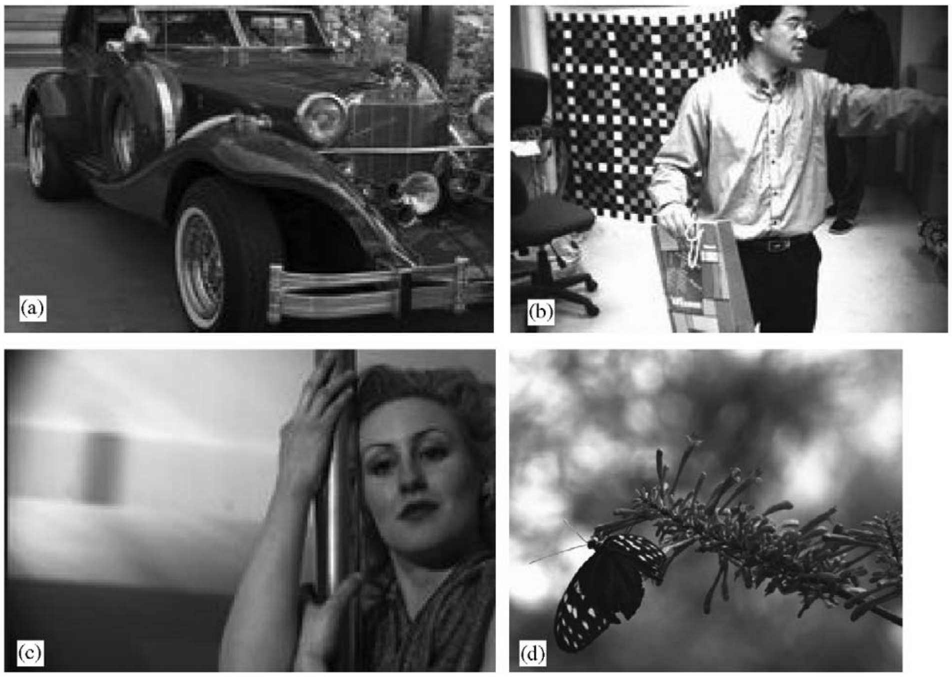


Fig. 8. The scenes in the (a) the first test clip (512×768); (b) the second test clip (480×640); (b) the third test clip (512×768); (b) the fourth test clip (1280×1536).

JPEG-LS standard [21], which is very easy to implement in real time. For $\mathbf{R}$ and $\mathbf{B}$, we need only 4 additions and 1 division (by 4, which is actually a bit shift operation in hardware implementation) per pixel to get difference images $\mathbf{D}_R$ and $\mathbf{D}_R$, which are then encoded with JPEG-LS. As for $\mathbf{G}_o$, 4 additions and 1 division (by 4) per pixel are needed to compute the residual signal, which is then clustered by the context modeling as described in Section 3.2, and this clustering process needs 3 additions, 1 division (by 2) and 1 logical operation per pixel. The remaining computation in encoding $\mathbf{G}_o$ is traditional entropy coding, such as Huffman coding, arithmetic coding or Golomb coding as used in JPEG-LS. Overall, the needed computation for the intraframe algorithm is at the same level as JPEG-LS.

For the interframe algorithm in Section 3, measured by the whole image, we need 5 additions and 2.5 divisions (by 4 or 2) per pixel to interpolate the green channel; 5 additions to determine the motion vector (d_x, d_y); 1 addition to compute the distance D_1; 1.5 additions and 0.5 division (by 2) to compute the distance D_0; 1 addition to compute the residual signal; 3 additions, 1 division (by 2) and 1 logical operation to cluster the residual signal. Totally, we spend 16.5 additions, 4 divisions and 1 logical operation in the predicting and clustering process. The remaining computation depends on the used entropy coder of the clustered residual signal. In our PC (3 GHz CPU, 2 GB RAM), a 768×512 sequence can be compressed in real time, i.e. the used time for each frame is less than $1/24$ s.

The proposed mosaic video lossless coding techniques are evaluated in comparison with JPEG-LS and JPEG 2000 lossless mode. Four mosaic video sequences were used in our experiments. The first video sequence is originally captured on film at a rate of 24-frames/s and then digitized by a high-resolution scanner. The mosaic data are simulated by subsampling the true color image using the Bayer pattern. Fig. 8(a) shows the scene of it, whose size is 512×768. The second to fourth video sequences are captured by a digital video camera at a rate of 24-frames/second. The spatial resolution for the second clip is 480×640 (Fig. 8(b)), the third clip is 512×768 (Fig. 8(c)) and the last clip is 1280×1536 (Fig. 8(d)). All the videos are stored in 8-bits depth.

Four lossless compression methods were used to code the clips: JPEG-LS, JPEG-2000 (in its lossless mode using 5-3 integer wavelet), the pure intraframe lossless coding method presented in Section 2 (denoted as IFC-LS), and the hybrid interframe and intraframe coding scheme described in Section 3 (denoted as HC-LS). In the implementation of the hybrid method, the input mosaic image is divided into blocks (such as 128×128, 64×64, etc.) and the HC-LS algorithm is applied to each block. In our experiments, the block size is set as 64×64. Table 1 lists the coding rates of the four methods on the four clips. The reported compression results are the average of 50 consecutive frames for the first clip, 100 frames for the second clip, 24 frames for the third clip and 48 frames for the fourth clip.

Table 1
Lossless compression results (bpp) for the four video sequences by JPEG-LS, JPEG-2000, the proposed intraframe coding and the hybrid intra-interframe coding

Video	JPEG-LS	JPEG-2000	IFC-LS	HC-LS
Clip 1	6.33	5.10	4.78	4.48
Clip 2	5.77	4.99	4.65	4.41
Clip 3	5.86	4.49	4.38	4.30
Clip 4	5.38	4.30	4.09	3.98

From Table 1 we see that the proposed hybrid coding scheme for lossless compression of mosaic video achieves the lowest bit rates. The presented intraframe mosaic coding method also outperforms the JPEG-LS and JPEG-2000 standards.

5. Conclusion

This paper presents two fast lossless compression techniques for raw mosaic video sequences: an intra-frame coding technique that exploits both spatial and spectral correlations of the mosaic data within a frame, and a hybrid inter- and intraframe coding technique that also exploits temporal correlation. The hybrid method can significantly improve the lossless compression performance at a slightly higher computational cost than intraframe coding techniques. This is made possible by a judicious use of motion estimation and by greatly simplified motion vectors.

References

[1] Bayer BE and Eastman Kodak Company. Color imaging array. US patent 3 971 065, 1975.
[2] Cok DR and Eastman Kodak Company. Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal. US patent 4 642 678, 1987.
[3] Gunturk BK, Altunbasak Y, Mersereau RM. Color plane interpolation using alternating projections. IEEE Transactions on Image Processing 2002;11:997–1013.
[4] Gunturk BK, Glotzbach J, Altunbasak Y, Schaffer RW, Murserau RM. Demosaicking: color filter array interpolation. IEEE Signal Processing Magazine 2005;22(1):44–54.
[5] Hamilton Jr JF, Adams JE. Adaptive color plane interpolation in single sensor color electronic camera. US Patent 5 629 734, 1997.
[6] Hirakawa K, Parks TW. Adaptive homogeneity-directed demosaicing algorithm. IEEE Transactions on Image Processing 2005;14:360–9.
[7] Kakarala R, Baharav Z. Adaptive demosaicing with the principal vector method. IEEE Transactions onConsumer Electronics 2002;48:932–7.
[8] Lukac R, Martin K, Plataniotis KN. Demosaicked image postprocessing using local color ratios. IEEE Transactions on Circuits and Systems for Video Technology 2004;14(6):914–20.
[9] Lukac R, Plataniotis KN. On a generalized demosaicking procedure: a taxonomy of single-sensor imaging solutions. Lecture Notes in Computer Science 2005;3514:687–94.
[10] Lukac R, Plataniotis KN. Normalized color-ratio modeling for CFA interpolation. IEEE Transactions on Consumer Electronics 2004;50(2):737–45.
[11] Lukac R, Plataniotis KN. Fast video demosaicking solution for mobile phone imaging applications. IEEE Transactions on Consumer Electronics 2005;51(2):675–81.
[12] Lukac R, Plataniotis KN, Hatzinakos D, Aleksic M. A novel cost effective demosaicing approach. IEEE Transactions on Consumer Electronics 2004;50(1):256–61.
[13] Wu X, Zhang L. Color Video Demosaicking via Motion Estimation and Data Fusion, revised in IEEE Transactions on Circuits and Systems for Video Technology.
[14] Wu X, Zhang N. Joint temporal and spatial color demosaicking. Proceedings of SPIE 2003;5017:307–13.
[15] Zhang L, Wu X. Color demosaicking via directional linear minimum mean square-error estimation. IEEE Transactions on Image Processing (to appear).
[16] Bazhyna A, Gotchev A, Egiazarian K. Near-lossless compression algorithm for Bayer pattern color filter arrays digital photography. Proceedings of SPIE-IS &T Electronic Imaging, SPIE 2005;5678:198–209.
[17] Koh CC, Mukherjee J, Mitra SK. New efficient methods of image compression in digital cameras with color filter array. IEEE Transactions on Consumer Electronics 2003;49(4):1448–56.
[18] Lee SY, Ortega A. A novel approach of image compression in digital cameras with a Bayer color filter array. Proceedings of IEEE International Conference on Image Processing 2001;3: 482–5.
[19] Toi T, Ohta M. A subband coding technique for image compression in single CCD cameras with bayer color filter arrays. IEEE Transactions on Consumer Electronics 1999;45(1):176–80.
[20] Tsai YT. Color image compression for single-chip cameras. IEEE Transactions on Electron Devices 1991;38(5):1226–32.
[21] ISO/IEC JTC1/SC29/WG1. Information technology – Lossless and near-lossless compression of continuous-tone still images. ISO FDIS 14495-1(JPEG-LS), also ITU Recommendation T.87, 1998.
[22] ISO/IEC JTC1/SC29/WG1 Document N1890. JPEG 2000 Part I Final Draft International Standard. 2000.
[23] Zhang L, Wu X. An edge guided image interpolation algorithm via directional filtering and data fusion. IEEE Transactions on Image Processing (under review).
[24] Kuhn P. Algorithms, complexity analysis and VLSI architectures for MPEG-4 motion estimation. Boston: Kluwer Academic Publishers; 1999.
[25] Wu X, Memon N. Lossless interframe image compression via context modeling. IEEE Transactions on Image Processing 2000;9(5):994–1001.