IPSJ Transactions on Computer Vision and Applications Vol. 2 16–24 (Mar. 2010)

Research Paper

Watermarking of Raw Digital Images in Camera Firmware

Peter Meerwald[†1] and Andreas Uhl[†1]

In this article we investigate ‘real-time’ watermarking of single-sensor digital camera images (often called ‘raw’ images) and blind watermark detection in demosaicked images. We describe the software-only implementation of simple additive spread-spectrum embedding in the firmware of a digital camera. For blind watermark detection, we develop a scheme which adaptively combines the polyphase components of the demosaicked image, taking advantage of the interpolated image structure. Experimental results show the benefits of the novel detection approach for several demosaicking techniques.

1. Introduction

Digital cameras are in ubiquitous use. Most popular digital cameras use a single, monochrome image sensor with a color filter array (CFA) on top, often arranged in the Bayer pattern, see **Fig. 1**. In order to provide a full-resolution RGB image, the sensor data has to be interpolated — a process called demosaicking — as well as color, gamma and white point corrected. Different demosaicking techniques exist, e.g., Refs. 8) and 3), yet the basic processing steps are shared by most camera implementations [18)].

The digital nature of the recorded images which allows for easy duplication and manipulation, poses challenges when these images are to be used as evidence in court or when resolving ownership claims. Active techniques, such as watermarking [5)], as well as passive or forensic approaches have been suggested to address image integrity verification, camera identification and ownership resolution. Many different forensic techniques have been proposed to detect image forgeries. For example, Chen, et al. [4)] exploit the inherent Photo-Response Non-Uniformity (PRNU) noise of the image sensor for camera identification and image integrity verification. Interpolation artefacts due to demosaicking are used by Popescu, et al. [17)] to verify the integrity of the image. Passive techniques have the disadvantage that camera characteristics such as PRNU must be estimated.

Blythe, et al. [2)] propose a secure digital camera which uses lossless watermarking to embed a biometric identifier of the photographer together with a cryptographic hash of the image data. Their embedding method efficiently changes the JPEG quantization tables and DCT coefficients but precludes watermarking of raw images. Tian, et al. [20)] propose a combined semi-fragile and robust watermarking for joint image authentication and copyright protection during the image capture process. However, the employed wavelet transform is computationally expensive. The image data volume and constrained power resources of digital cameras demand efficient processing. Mohanty, et al. [15)] describe a hardware implementation for combined robust and fragile watermarking. Nelson, et al. [16)] propose an image sensor with watermarking capabilities that adds pseudo-random noise. Lukac, et al. [12)] introduce a visible watermark embossed in sensor data. Few authors have considered watermark protection of the raw images, although the raw sensor data is probably the most valuable asset. The raw data often has a higher dynamic range than the demosaicked copy and does not suffer from post-processing artefacts. Therefore the raw data is the preferential format for high-quality digital camera image archival. Further, it is highly desirable that all potential copies of the same scene shot carry the same watermark which is difficult to guarantee if the watermark is applied later on.

In this article, we extend the simple, additive spread-spectrum watermarking scheme for ‘real-time’ watermarking of single-sensor image data (‘raw’ images) presented in Ref. 14). This application scenario has not received much attention

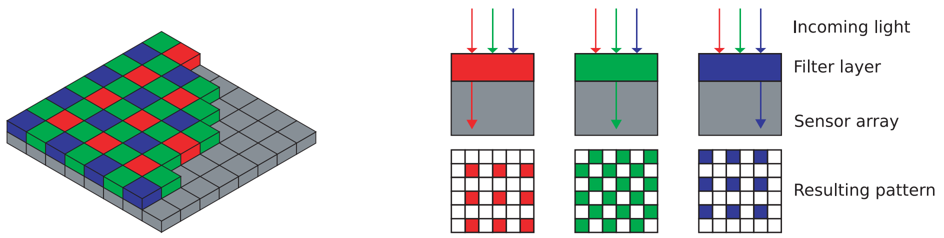

Fig. 1 Color filter array (CFA) arranged in the popular Bayer pattern.

†1 Department of Computer Sciences, University of Salzburg, Austria

before, but is important in order to jointly secure raw image data for archival, as well as the processed JPEG images. We describe the scheme's software-only implementation in the firmware of a digital camera in Section 2. For blind watermark detection in demosaicked images, we propose a method that adaptively combines the polyphase components of the demosaicked image in Section 3, taking advantage of the interpolated image structure [6]. A detector extension incorporating all color bands is discussed in Section 4. Further study on the impact of different demosaicking approaches can be found in Ref. 13). In Section 5, we demonstrate the firmware implementation of the watermark embedding and analyze the performance of the novel detection approach after JPEG compression. Concluding remarks are offered in Section 6.

2. Watermark Embedding in Camera Firmware

Watermarking in digital cameras has not yet gained wide acceptance, although Kodak and Epson both have manufactured cameras with digital watermarking capabilities [2]. For this article, we build on the CHDK project [⋆1], which provides an open-source firmware add-on for Canon consumer cameras, based on the DIGIC II and III image processors — essentially a 32-bit ARM9 architecture processor, augmented with custom hardware functionality for JPEG coding, scaling, color conversion, etc. CHDK provides a Linux-hosted cross-compilation environment to build a firmware loader that partially replaces the original Canon firmware and hooks into the image processing pipeline as illustrated in **Fig. 2**. This way, we gain access to the memory buffer holding the raw single-sensor image data after image acquisition.

For watermark embedding in camera firmware, we opt for a simple, additive spread-spectrum watermark design to meet the runtime requirement. Note that Nelson, et al. [16] essentially perform the same embedding operation, but in the image sensor hardware. Software implementations offer more flexibility than hardware-based approaches and can be implemented at virtually no cost. The flexibility allows perceptual shaping of the watermark to increase either robustness or image fidelity. One of the aims of this article is to show that current embedded processors (in our case the ARM9 architecture CPU in Canon consumer cameras) are powerful enough to perform simple additive watermark embedding in software. The choice to watermark only the perceptually least significant blue color channel helps to reduce the data volume.

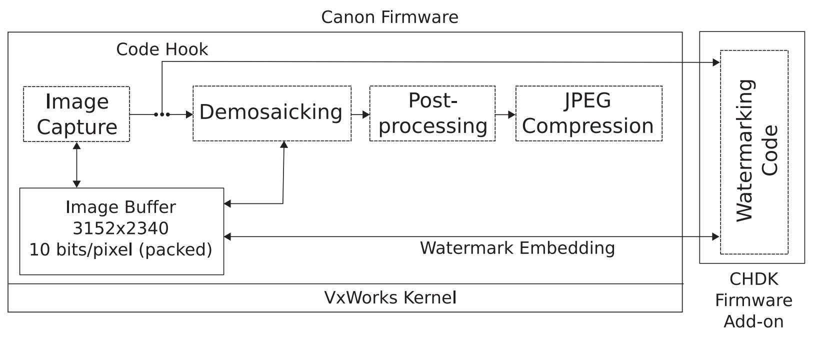

Fig. 2 Architecture of the watermarking firmware add-on.

The raw image data is represented with 10 bits/pixel in packed format in the camera's memory buffer, hence the individual pixels must be shifted into place before further processing. Care must be taken not to watermark dead pixels due to sensor imperfections and to properly clip the pixel values to 10 bits, otherwise visible distortion results. Initially, the pixels were addressed and processed individually consuming approximately 40 seconds to watermark the raw data (3,112 × 2,328 pixels, 9.2 MB, in case of the Canon IXUS 70 camera). Memory throughput is about 45 MB/second, but performance was constrained mainly by the repetitive address computation for unaligned byte memory accesses. Optimized loop unrolling and the implicit arithmetic bit shift option of the load/store instructions in the ARM instruction set help to achieve close to 'real-time' performance with a delay of less that one second [⋆2]. **Figure 3** shows a part of the watermark embedding implementation and the resulting annotated optimized ARM assembler code produced by the GCC 4.3.0 compiler. Note that the implementation is plain C source code, yet it is very well mapped to the three-operand

⋆1 Available at http://chdk.wikia.com.

⋆2 The source code of the watermarking firmware add-on based on CHDK can be downloaded from http://www.wavelab.at/sources.

ARM instruction set. Use of SIMD assembler instructions or hardware assistance may further improve performance. The code processes the first two pixels of a row of packed pixels in the camera's image buffer. 16 bit values are read from the 10 bits/pixel image buffer (`prow_in`) and shifted through a 32 bit buffer (`bit_buf`). Every alternating pixel value is modified by the macro `WATERMARK()` which implements the watermark embedding as given by Eq. (1).

After embedding, the watermarked raw image can be stored at this point for later post-processing with third party software or, alternatively, the data is upsampled in the demosaicking unit of the camera and the image is compressed and stored in JPEG format. Watermarking the raw image data has the advantage that copyright protection is incorporated at an early point in the image life cycle. The most valuable original sensor data as well as all derived images are protected by the same watermark. On the downside, the watermarked raw images has to withstand many processing steps. We provide first results on the impact of demosaicking on an additive watermark in Section 5.

```
...
prow_out = prow_in = (uint16 *)
      &rowbuf[PIXTOBYTES(RAW_LEFT_MARGIN+4)];
bit_buf = *prow_in++;                         // LDRH R7, [SL], #2
bit_buf = *prow_in++;                         // LDRH R7, [SL], #2
out_bit_buf = bit_buf >> 6;                   // MOV R6, R7, ASR #6
bit_buf = (bit_buf << 16) + *prow_in++;       // LDRH R3, [SL], #2
                                              // ADD R7, R3, R7, ASL #16
pixel = bit_buf >> 12 & 0x3ff;                // MOV R3, R7, ASR #12
                                              // MOV R4, R3, ASL #22
                                              // MOV R4, R4, LSR #22
out_bit_buf = WATERMARK(pixel)                // R2 = WATERMARK(R4)
      + (out_bit_buf<< 10);                   // ADD R6, R2, R6 ASL #10
*prow_out++ = out_bit_buf >> 4;               // MOV R3, R6, ASR #4
                                              // STRH R3, [R8], #2
out_bit_buf = (bit_buf >> 2 & 0x3ff)          // MOV R2, R7, ASR #2
      + (out_bit_buf << 10);                  // MOV R4, R2, ASL #22
...                                           // ADD R6, R4, R6, ASL #10
```

Fig. 3 Efficient processing the first two pixels of a packed image buffer row.

The actual camera implementation of the demosaicking, post-processing and compression stage is unknown. However, we can make assumptions on the interpolation and demosaicking step. In the next section, we utilize the interpolated structure of the demosaicked image for efficient watermark detection.

3. Watermark Detection from the Demosaicked Image

Figure 4 depicts the intercalated watermark embedding stage and the following demosaicking, post-processing and JPEG compression stages of the camera's image processing pipeline. Before we proceed, we introduce some notation given in **Table 1** to unambiguously refer to the host, x, and watermarked, x_w, CFA pixel data and identify the components and color channels of the received demosaicked image s.

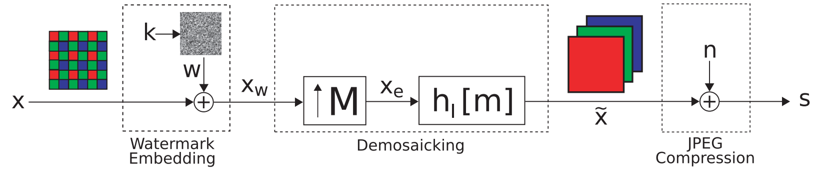

Fig. 4 Watermarking embedding and image processing pipeline.

Table 1 Symbol notation.

Symbol	Definition
x	host pixel matrix of raw CFA data
x^b, x^{g_0}	matrix of blue, green CFA host pixels
x^b_w, $x^{g_0}_w$	matrix of watermarked blue, green CFA pixels
w	matrix with pseudo-random, bipolar $\{-1, 1\}$ watermark symbols
x_e, $\tilde{x}$	expanded and upsampled pixel matrix
s	received RGB image data after demosaicking and post-processing
s^R_i, s^G_i, s^B_i	polyphase components $0 \leq i \leq 3$ of the RGB image bands
h_I	low-pass filter for interpolation
h_i	estimation filter
h^c_i	filter for interference cancellation
γ	watermark embedding strength
β	scaling factor
α_i	image fusion weighting factor

In the embedding stage, a pseudo-random bipolar spread-spectrum watermark w generated from a secret seed value k identifying the copyright owner is added to the blue color component of the sensor data

$$x_w^b[\mathbf{m}] = x^b[\mathbf{m}] + \gamma \cdot w[\mathbf{m}], \tag{1}$$

where $\mathbf{m} \in \mathbb{N}^2$ denotes pixel indices and $\gamma > 0$ controls the embedding strength. Later we will also consider embedding in half of the green color component pixels written as

$$x_w^{g_0}[\mathbf{m}] = x^{g_0}[\mathbf{m}] + \gamma \cdot w[\mathbf{m}] \tag{2}$$

where the Bayer CFA pattern is represented by $[r\ g_0; g_1\ b]$.

The watermark detector does not know which demosaicking algorithm and post-processing operations have been applied on the watermarked raw image. Nevertheless, we know that demosaicking must interpolate missing pixels to obtain the high-resolution data of all three RGB color channels. We propose to approximate the effect of the demosaicking step on each color component's pixels with an expansion of the CFA data with a matrix $\mathbf{M} = [2\ 0; 0\ 2]$ which yields an image $x_e[\mathbf{m}] = x_w[\mathbf{M}^{-1}\mathbf{m}]$ twice the size in each dimension and interpolation with a low-pass filter $h_I = [1/4\ 1/2\ 1/4; 1/2\ 1\ 1/2; 1/4\ 1/2\ 1/4]$ resulting in an upsampled image $\tilde{x}[\mathbf{m}] = h_I[\mathbf{m}] * x_e[\mathbf{m}]$ where $*$ denotes convolution. These assumptions relate to *intra*-only, bilinear demosaicking and directly correspond to the model analyzed by Giannoula, et al.[6] when considering the red and blue color components; see Section 4 for more advanced demosaicking techniques. Finally, we roughly model the impact of the post-processing and JPEG compression stage as an additive noise source n.

Relying on this model, we can adapt the watermark detection strategy proposed by Giannoula, et al.[6] for interpolated, noisy images. While the watermark is embedded in the low-resolution raw sensor data, watermark detection takes place using the high-resolution, interpolated RGB data of the demosaicked and compressed image. In order to exploit the watermark information spread out due to interpolation, we represent the received demosaicked image signal s using polyphase components[21] and perform weighted fusion of the components to improve detection performance. The polyphase component notation allows to express the periodically interleaved subsequences of the image signal due to demosaicking. First we consider embedding in the blue sensor pixels, then green channel watermarking and exploiting *inter* color channel watermark correlation is addressed in Section 4.

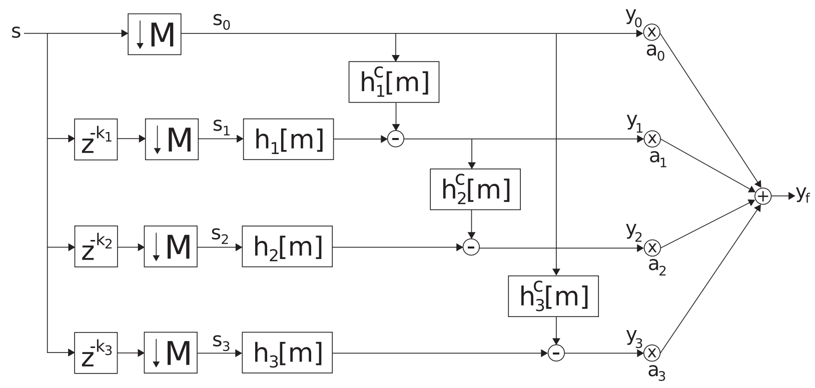

Fig. 5 Polyphase component fusion of one color band of the received image.

Each of the RGB color bands s^R, s^G, s^B of s is split into its noisy polyphase components $s_i^j[\mathbf{m}] = s^j[\mathbf{Mm} + \mathbf{k}_i]$ where $0 \leq i \leq 3$ refers to one of the four components[21], $j \in \{R, G, B\}$ indicates the color band and $\mathbf{k}_i \in \{(0,0), (0,1), (1,0), (1,1)\}$. In **Fig. 5**, we illustrate the polyphase decomposition and fusion process of one color band. Considering the case of watermarking the blue CFA data x_w^b, the polyphase component s_0^B represents the original low-resolution watermarked data, corrupted by a noise component n_0, $s_0^B[\mathbf{m}] = x_w^b[\mathbf{m}] + n_0[\mathbf{m}]$ according to our model. $y_0[\mathbf{m}] = s_0^B[\mathbf{m}]$ is the primary image component used for watermark detection. With the help of two linear filters for estimation and interference cancellation,

$$h_i[\mathbf{m}] = \beta \cdot h_I[\mathbf{m}] \quad \text{and} \quad h_i^c[\mathbf{m}] = \beta \cdot h_I[\mathbf{m}] * h_I[\mathbf{m}] - \delta[\mathbf{m}] \tag{3}$$

respectively, further noisy estimates of x_w^b are computed from the remaining polyphase components, such that

$$y_i[\mathbf{m}] = x_w^b[\mathbf{m}] + n_i[\mathbf{m}] = h_i[\mathbf{m}] * s_i^B[\mathbf{m}] - h_i^c[\mathbf{m}] * s_0^B[\mathbf{m}]. \tag{4}$$

The scaling factor β is adjusted such that $h_i^c[0] = 0$ for $1 \leq i \leq 3$ and $\delta[\mathbf{m}]$ is the Kronecker delta. Finally, the components y_i are *fused* according to optimal weight factors $a_i \in [0,1]$, $\sum a_i = 1$, depending on the estimated noise variance $\sigma_{n_i}^2$ of each component, $y_f[\mathbf{m}] = \sum_i a_i \cdot y_i[\mathbf{m}]$ where

$$(a_0, \ldots, a_3) = \left(\frac{1}{\sigma_{n_0}^2 \sum_i \frac{1}{\sigma_{n_i}^2}}, \ldots, \frac{1}{\sigma_{n_3}^2 \sum_i \frac{1}{\sigma_{n_i}^2}} \right). \qquad (5)$$

Giannoula, et al.[6] suggest to estimate the noise variance $\sigma_{n_i}^2$ by filtering the initial component samples s_0^B and subtracting the result from s_i^B, i.e.,

$$\hat{\sigma}_{n_i}^2 = \text{var}\left(s_i^B[\mathbf{m}] - h_I[\mathbf{m}] * s_0^B[\mathbf{m}]\right). \qquad (6)$$

We apply a linear correlation detector on the *fused* image. See Ref. 6) for a detailed analysis of the detector.

4. Improvements and Experiments

A tremendous amount of research has been published recently on image demosaicking. Although demosaicking can be solved by standard image interpolation techniques such as e.g., bilinear or edge-directed interpolation[11], exploiting the *intra* and *inter* color channel dependencies can significantly improve the visual appearance of the full resolution image. Note that the actual implementation in digital cameras is generally not known and might be covered by patents[1]. Most demosaicking methods belong to the class of *sequential* interpolation algorithms[10] where first the luminance (green) channel is reconstructed which then aids in recovering the chrominance channels. Edge-directed interpolation of the green component forms the basis of many sequential, spatial domain demosaicking methods. Alternatively, the missing luminance data can also be recovered by frequency-domain filtering approaches such as the Adaptive Homogeneity-Directed (AHD) algorithm[8]. Having a full-resolution luminance (green) channel facilitates the recovery of the chrominance channels by enforcing constant hue rules. The assumption is that the ratio of color differences (i.e., R-G, B-G) is constant within image objects. The color-difference signals are then interpolated based on the full-resolution green channel and the down-sampled chrominance channels[7]; the Patterned Pixel Grouping (PPG)[⋆1] algorithm builds on the smooth hue transition assumption. The threshold-based Variable Number of Gradients (VNG) methods[3] computes the most likely pixel value based on the color gradients in a neighborhood.

Due to inter color-channel interpolation, the watermark signal embedded in one CFA color band is carried over to the other color bands as well. Optimal fusion of polyphase components of one color band was discussed in the previous section. In a similar way, we can try to exploit inter band correlation by fusing polyphase components from all color bands. Considering the case where the watermark is embedded in the blue CFA component, we note that the red and green pixels in the full-resolution image have to be reconstructed at those locations where the CFA data has a blue pixel. The original blue pixel values contribute to the reconstruction of the green and red pixels and so the watermark signal is transferred to all bands.

We construct a *color fused* image y_{cf} from polyphase components of all three RGB color bands denoted by s_i^R, s_i^G, s_i^B and noisy image estimates y_i^R, y_i^G, y_i^B where $0 \leq i \leq 3$ denotes the polyphase component as before. The *color fused* image is the weighted sum

$$y_{cf} = a_0^R \cdot y_0^R[\mathbf{m}] + \sum_{i=0}^{3} a_i^G \cdot y_i^G[\mathbf{m}] + a_0^B \cdot y_0^B[\mathbf{m}] \qquad (7)$$

in case of green channel embedding (and accordingly for embedding in blue CFA pixels). The fusion weights a_i^j, $\sum a_i^j = 1$ with $j \in \{R, G, B\}$ indicating the color band, are determined as before depending on the individual components' estimated noise variances

$$\hat{\sigma}_{n_i^j}^2 = \text{var}\left(s_i^j[\mathbf{m}] - h_I[\mathbf{m}] * s_0^G[\mathbf{m}]\right). \qquad (8)$$

In the next section, we will assess the impact of the three demosaicking methods mentioned (AHD, VNG, PPG) and compare the performance of the linear-correlation watermark detector operating on the *fused* and *color fused* images. We distinguish the cases of embedding in the blue CFA data versus embedding in

⋆1 By Chuan-kai Lin, described at http://web.cecs.pdx.edu/~cklin/demosaic/.

half of the green CFA data. We have to omit a more comprehensive assessment of demosaicking methods [1),10)] due to lack of space and refer to Ref. 13). More accurate modelling of the host signal (e.g., assuming a Cauchy distribution [19)]) might improve the detection performance. Especially the interplay between watermarking of raw CFA data and joint denoising and demosaicking as recently proposed [9),22)] is an interesting open area of research.

5. Results

We have implemented watermark embedding in firmware using CHDK for the Canon IXUS 70 and PowerShot A720, 7 and 8 Megapixel cameras, respectively. CHDK adds approximately 150 KB new firmware code to the 3.5 MB Canon firmware image. About 3 KB of code and data is occupied by watermarking functionality, leaving roughly 880 KB free memory available. The watermark embedding stage consumes less than one second, about the same time as storing the raw image data to disk. The experiments by Nelson, et al. [16)] confirm that watermark embedding in CFA sensor data with strength $\gamma = 4$ is imperceptible.

In **Fig. 6**, we present eight test images taken with the Canon IXUS 70 camera and corresponding detection results. A watermark is embedded in the blue channel (embedding strength $\gamma = 4$) of the raw image. Watermark detection is performed on the demosaicked image obtained with the default AHD method [8)] of the `dcraw` [⋆1] program and after JPEG compression with quality factors ranging from 100 to 30. Note that `dcraw` also performs white-balance adjustment and color conversion in addition to demosaicking. The plots show the probability of missing the watermark estimated from 1,000 test runs with four different detectors: the proposed *fused* detector, *direct* correlation of the watermark with the y_0 component, and the reference methods (upsampling the watermark to match the received image dimensions and downsampling the image to match the size of the watermark). The probability of false-alarm (P_{fa}) is set to 10^{-6}. The y_0 component simply corresponds to the originally watermarked pixels and does not contain interpolated pixel data. Clearly, the proposed detector delivers best performance for all images. Similar results were obtained with raw images taken by other digital cameras.

In **Table 2** we compare the impact of different demosaicking methods as implemented by `dcraw` on watermark detection performance. For a false-alarm rate of 10^{-6}, we compare the probability of missing the watermark for our eight test images with the *direct* and *fused* detector after demosaicking the raw images with the AHD [8)], VNG [3)] and PPG algorithm. We found that VNG demosaicking allows for the best watermark detection, followed by the AHD and PPG method. With moderate JPEG compression ($Q = 70$), the *fused* detector shows best per-

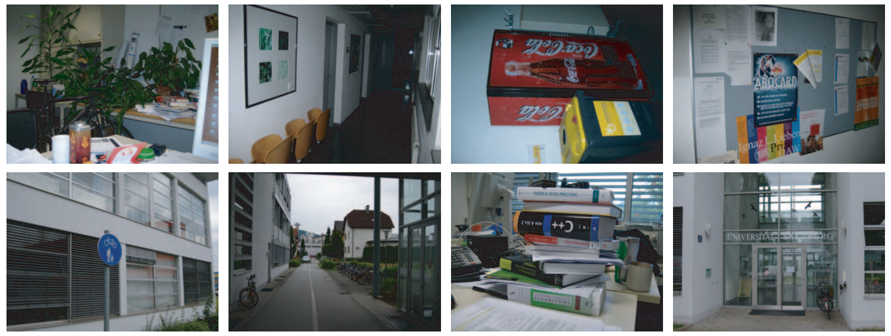
Fig. 6 Test images #1 to #8 (3,112 × 2,328 pixels).

Table 2 Probability of missing the watermark for AHD, VNG and PPG demosaicking and after JPEG compression ($Q = 70$); $P_{fa} = 10^{-6}$.

Image	AHD		VNG		PPG	
	Direct	Fused	Direct	Fused	Direct	Fused
#1	10^{-20}	10^{-84}	10^{-43}	10^{-294}	10^{-12}	10^{-29}
#2	10^{-8}	10^{-35}	10^{-21}	10^{-160}	10^{-6}	10^{-13}
#3	10^{-10}	10^{-38}	10^{-23}	10^{-145}	10^{-7}	10^{-16}
#4	10^{-15}	10^{-80}	10^{-42}	0.0	10^{-9}	10^{-19}
#5	10^{-5}	10^{-15}	10^{-19}	10^{-116}	10^{-4}	10^{-7}
#6	10^{-6}	10^{-18}	10^{-16}	10^{-102}	10^{-4}	10^{-9}
#7	10^{-21}	10^{-69}	10^{-53}	10^{-289}	10^{-14}	10^{-29}
#8	10^{-4}	10^{-11}	10^{-15}	10^{-77}	10^{-3}	10^{-4}

⋆1 `dcraw` is available at http://www.cybercom.net/~dcoffin/dcraw/. Version 8.86 was used for the experiments.

Table 3 Probability of missing the watermark for different resolution and JPEG quality settings (Canon IXUS 70), first test image; $P_{fa} = 10^{-6}$.

Resolution	Quality	Direct	Fused	Downsampled Image	Upsampled Watermark
3,072 × 2,304	SuperFine	10^{-161}	0.0	10^{-15}	10^{-100}
3,072 × 2,304	Fine	10^{-125}	0.0	10^{-15}	10^{-83}
3,072 × 2,304	Normal	10^{-88}	0.0	10^{-14}	10^{-63}
2,592 × 1,944	SuperFine	10^{-68}	0.0	10^{-14}	10^{-50}
2,048 × 1,536	SuperFine	10^{-60}	10^{-223}	10^{-16}	10^{-46}
1,600 × 1,200	SuperFine	10^{-38}	10^{-117}	10^{-8}	10^{-29}
640 × 480	SuperFine	10^{-39}	10^{-98}	10^{-6}	10^{-20}

formance for all images, followed by the *direct* approach. The other two detectors always perform worse and results are omitted.

The impact of the image processing pipeline of the Canon IXUS 70 camera on the watermark is explored in **Table 3**. The raw data of the first test image (depicted in **Fig. 7**) is watermarked ($\gamma = 4$) and then processed by the camera into a JPEG image with varying image quality and resolution settings. Note that the camera stores a slightly cropped version of the raw image (3,072 × 2,304 pixels). The smaller resolution images are upsampled to 3,072 × 2,304 pixels using a bilinear filter before watermark detection. The experiment is repeated 100 times for each setting using the scripting capabilities of the CHDK firmware. We estimate the probability of missing the watermark for each of our four detectors. The *fused* detectors is least likely to miss the watermark in all cases. Repeating the experiment with other test images shows consistent results.

In **Tables 4** and **5** we compare the detection performance for an additive watermark embedded in the blue component of the CFA image versus in the half of green CFA pixels. The received images have been demosaicked and JPEG compressed ($Q = 50$). It is evident that the green channel watermark is far easier to detect. The reason is the sequential demosaicking approach favoring the green color band and the fact that JPEG compression better preserves luminance information than chrominance data. Further, we can observe that the watermark detector based on the *color fused* image can exploit the inter color component

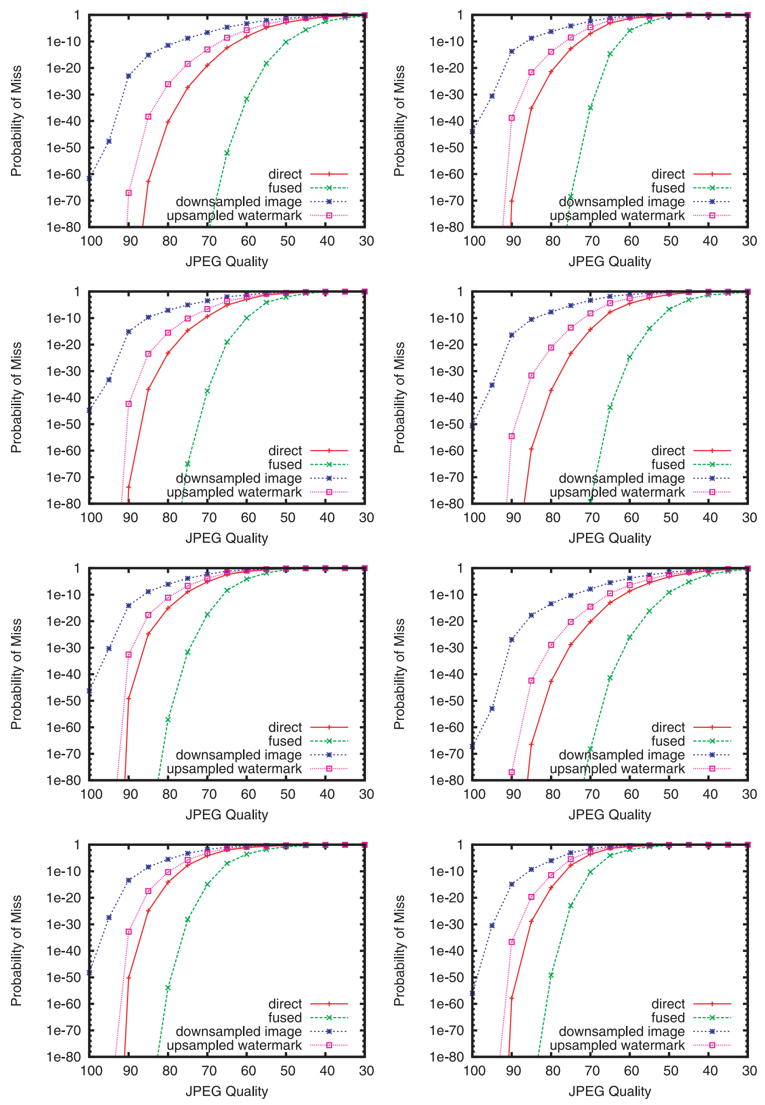

Fig. 7 Simulated watermark detection results on the eight test images after AHD demosaicking and JPEG compression; $P_{fa} = 10^{-6}$.

Table 4 Probability of missing the watermark embedded in the blue CFA pixels for the demosaicking methods AHD, VNG, PPG and after JPEG compression ($Q = 50$); $P_{fa} = 10^{-6}$.

Image	AHD		VNG		PPG	
	Fused	Color Fused	Fused	Color Fused	Fused	Color Fused
#1	10^{-10}	10^{-10}	10^{-62}	10^{-61}	10^{-3}	10^{-3}
#2	10^{-8}	10^{-8}	10^{-57}	10^{-54}	10^{-3}	10^{-3}
#3	10^{-1}	10^{-1}	10^{-12}	10^{-11}	10^{-1}	10^{-1}
#4	10^{-2}	10^{-2}	10^{-16}	10^{-16}	10^{-1}	10^{-1}
#5	10^{-6}	10^{-7}	10^{-61}	10^{-63}	10^{-1}	10^{-1}
#6	10^{-2}	10^{-2}	10^{-19}	10^{-20}	10^{-1}	10^{-1}
#7	10^{-1}	10^{-1}	10^{-10}	10^{-9}	10^{-1}	10^{-1}
#8	10^{-1}	10^{-1}	10^{-15}	10^{-15}	10^{-1}	10^{-1}

Table 5 Probability of missing the watermark embedded in half of the green CFA pixels for the demosaicking methods AHD, VNG, PPG and after JPEG compression ($Q = 50$); $P_{fa} = 10^{-6}$.

Image	AHD		VNG		PPG	
	Fused	Color Fused	Fused	Color Fused	Fused	Color Fused
#1	10^{-309}	0.0	10^{-147}	10^{-172}	10^{-188}	10^{-224}
#2	10^{-173}	10^{-228}	10^{-68}	10^{-87}	10^{-109}	10^{-145}
#3	10^{-227}	10^{-262}	10^{-98}	10^{-110}	10^{-109}	10^{-125}
#4	10^{-234}	10^{-240}	10^{-85}	10^{-87}	10^{-109}	10^{-112}
#5	0.0	0.0	0.0	0.0	0.0	0.0
#6	10^{-121}	0.0	10^{-78}	10^{-207}	10^{-73}	10^{-195}
#7	10^{-131}	10^{-220}	10^{-65}	10^{-105}	10^{-79}	10^{-135}
#8	10^{-183}	10^{-262}	10^{-89}	10^{-123}	10^{-92}	10^{-134}

correlation and significantly outperforms the *fused* detector when the watermark has been embedded in green CFA pixels.

6. Conclusion

Digital watermarking has to be applied close to the image acquisition stage in order to simultaneously protect the copyright of both, the raw and compressed image. Hence, we have implemented additive spread-spectrum watermark embedding of the raw image data in digital camera firmware building on the CHDK firmware add-on for Canon digital cameras. The reported watermarking implementation allows to test the impact of the actual demosaicking and post-processing pipeline of the camera on the watermark.

A framework for blind watermark detection in noisy, interpolated images has been successfully applied to demosaicked images, irrespective of a particular interpolation technique. We evaluated the impact of different demosaicking methods on watermark detection performance, including the particular Canon implementation. We compare results for embedding in green versus blue CFA sensor data pixels and observe that the novel color-fused detection scheme offers improved performance.

Acknowledgments Supported by Austrian Science Fund project FWF-P19159-N13. Thanks to the people working on CHDK and Colin M. L. Burnett for the graphics used in Fig. 1.

References

1) Battiato, S., Guarnera, M., Messina, G. and Tomaselli, V.: Recent Patents on Color Demosaicing, *Recent Patents on Computer Science*, Vol.1, No.3, pp.194–207 (2008).
2) Blythe, P. and Fridrich, J.: Secure Digital Camera, *Digital Forensic Research Workshop*, Baltimore, MD, USA (2004).
3) Chang, E., Cheung, S. and Pan, D.Y.: Color filter array recovery using a threshold-based variable number of gradients, *Proc. SPIE, Sensors, Cameras, and Applications for Digital Photography*, Vol.3650, San Jose, CA, USA, SPIE, pp.36–43 (1999).
4) Chen, M., Fridrich, J., Goljan, M. and Lukas, J.: Determining Image Origin and Integrity Using Sensor Noise, *IEEE Trans. Information Security and Forensics*, Vol.3, No.1, pp.74–90 (2008).
5) Cox, I.J., Miller, M.L., Bloom, J.A., Fridrich, J. and Kalker, T.: *Digital Watermarking and Steganography*, Morgan Kaufmann (2007).
6) Giannoula, A., Boulgouris, N.V., Hatzinakos, D. and Plataniotis, K.N.: Watermark detection for noisy interpolated images, *IEEE Trans. Circuits and Systems*, Vol.53, No.5, pp.359–363 (2006).
7) Hamilton, J. and Adams, J.: Adaptive color plane interpolation in single sensor color electronic camera (1997). US Patent 5,629,734.
8) Hirakawa, K. and Parks, T.W.: Adaptive Homogeneity-Directed Demosaicing Algorithm, *IEEE Trans. Image Processing*, Vol.14, No.3, pp.360–369 (2005).
9) Hirakawa, K. and Parks, T.W.: Joint Demosaicing and Denoising, *IEEE Trans. Image Processing*, Vol.15, No.8, pp.2146–2157 (2006).
10) Li, X., Gunturk, B. and Zhang, L.: Image Demosaicing: A Systematic Survey, *Proc. SPIE, Visual Communications and Image Processing, VCIP '08*, Vol.6822,

San Jose, CA, USA, pp.68221J–68221J–15, SPIE (2008).
11) Li, X. and Orchard, M.: New Edge-Directed Interpolation, *IEEE Trans. Image Processing*, Vol.10, No.10, pp.1521–1527 (2001).
12) Lukac, R. and Plataniotis, K.N.: Camera Image Watermark Transfer by Demosaicking, *Proc. 48th International Symposium ELMAR '06, Multimedia Signal Processing and Communication*, Zadar, Croatia, pp.9–12 (2006).
13) Meerwald, P. and Uhl, A.: Additive spread-spectrum watermark detection in demosaicked images, *Proc. ACM Multimedia and Security Workshop, MMSEC '09*, Princeton, NJ, USA, pp.25–32, ACM (2009).
14) Meerwald, P. and Uhl, A.: Watermarking of raw digital images in camera firmware: Embedding and detection, *Advances in Image and Video Technology: Proc. 3rd Pacific-Rim Symposium on Image and Video Technology, PSIVT '09*, Lecture Notes in Computer Science, Vol.5414, Tokyo, Japan, pp.340–348, Springer (2009).
15) Mohanty, S.P., Kougianos, E. and Ranganathan, N.: VLSI Architecture and Chip for Combined Invisible Robust and Fragile Watermarking, *IET Computers & Digital Techniques*, Vol.1, No.5, pp.600–611 (2007).
16) Nelson, G.R., Julien, G.A. and Yadid-Pecht, O.: CMOS image sensor with watermarking capabilities, *Proc. IEEE International Symposium on Circuits and Systems, ISCAS '05*, Vol.5, Kobe, Japan, pp.5326–5329, IEEE (2005).
17) Popescu, A.C. and Farid, H.: Exposing Digital Forgeries in Color Filter Array Interpolated Images, *IEEE Trans. Signal Processing*, Vol.53, No.10, pp.3948–3959 (2005).
18) Ramanath, R., Snyder, W.E., Yoo, Y. and Drew, M.S.: Color Image Processing Pipeline: A general survey of digital still camera processing, *IEEE Signal Processing Magazine*, Vol.22, No.1, pp.34–43 (2005).
19) Sayrol, E., Vidal, J., Cabanillas, S. and Santamaría, S.: Optimum Watermark Detection in Color Images, *Proc. IEEE International Conference on Image Processing (ICIP'99)*, Vol.2, Kobe, Japan, pp.231–235 (1999).
20) Tian, L. and Tai, H.-M.: Secure images captured by digital camera, *International Conference on Consumer Electronics, Digest of Technical Papers, ICCE '06*, Las Vegas, NV, USA, pp.341–342, IEEE (2006).
21) Vaidyanathan, P.P.: Multirate digital filters, filter banks, polyphase networks, and applications: A tutorial, *Proc. IEEE*, Vol.78, No.1, pp.56–93 (1990).
22) Zhang, L., Wu, X. and Zhang, D.: Color Reproduction from Noisy CFA Data of Single Sensor Digital Cameras, *IEEE Trans. Image Processing*, Vol.16, No.9, pp.2184–2197 (2007).

(Received March 31, 2009)
(Accepted August 22, 2009)
(Released March 11, 2010)

(Communicated by *Stephen Lin*)

Peter Meerwald was born in 1975. He received his Master degree in Computer Sciences from Bowling Green State University (Ohio, USA) in 1999 and University of Salzburg in 2001 (Austria). He is currently working on his Ph.D. thesis in the field of image and video watermarking. His research interests include multimedia security, watermarking, and scalable image and video coding.

Andreas Uhl was born in 1968. He is an associate professor at Department of Computer Sciences (University of Salzburg, Austria) where he leads the Multimedia Processing and Security Lab. His research interests include image and video processing and compression, wavelets, media security, medical imaging, biometrics, and number-theoretical numerics.