Watermarking of raw digital images in camera firmware: embedding and detection

Peter Meerwald and Andreas Uhl [⋆]

University of Salzburg, Dept. of Computer Sciences,
Jakob-Haringer-Str. 2, A-5020 Salzburg, Austria
{pmeerw, uhl}@cosy.sbg.ac.at

Abstract. In this paper 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.

Key words: watermarking, demosaicking, signal detection, firmware

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 Figure 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. [1, 2], yet the basic processing steps are shared by most camera implementations.

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 [3], 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. [5] to verify the integrity of the image. Passive techniques have the disadvantage that camera characteristics such as PRNU have to be estimated before use.

[⋆] Supported by Austrian Science Fund project FWF-P19159-N13.

Blythe et al. [6] 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. [7] 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. [8] describe a hardware implementation for combined robust and fragile watermarking. Few authors have considered watermark protection of the raw images, although the raw data is probably the most valuable asset. Nelson et al. [9] propose an image sensor with watermarking capabilities that adds pseudo-random noise. Lukac et al. [10] introduce a visible watermark embossed in sensor data.

In this paper, we propose a simple, additive spread-spectrum watermarking scheme for 'real-time' watermarking of single-sensor image data ('raw' images) and describe its software-only implementation in the firmware of a digital camera in section 2. For blind watermark detection in demosaicked images, we propose a scheme that adaptively combines the polyphase components of the demosaicked image in section 3, taking advantage of the interpolated image structure [11]. In section 4, 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 5.

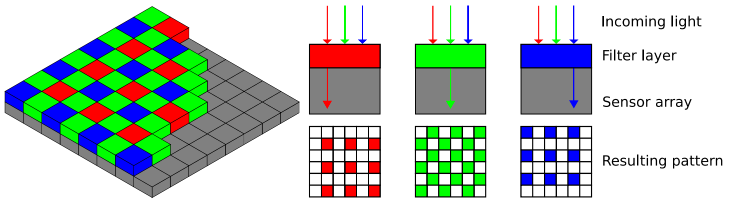

Fig. 1. Color filter array (CFA) arranged in the popular Bayer pattern

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 [6]. For this paper, we build on the CHDK project[1], which provides an open-source firmware add-on for Canon consumer cameras, based on

[1] Available at `http://chdk.wikia.com`. We are using SVN revision 470.

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 Figure 2. This way, we gain access to the memory buffer holding the raw single-sensor image data after image acquisition.

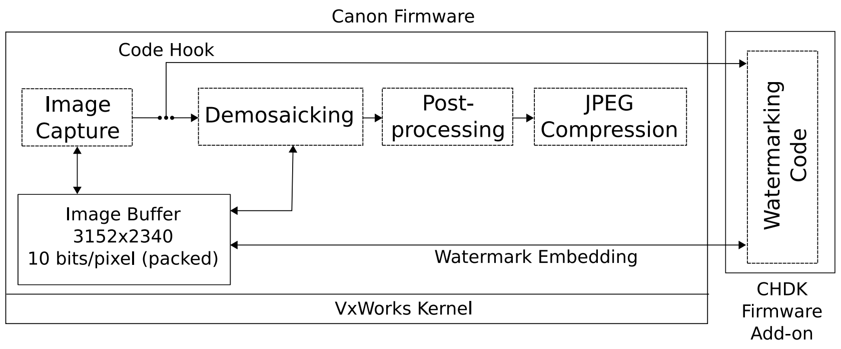

Fig. 2. Architecture of the watermarking firmware add-on

For watermark embedded in camera firmware, we opt for a simple, additive spread-spectrum watermark design to meet the runtime requirement. Note that Nelson et al. [9] essentially perform the same embedding operation, but in the image sensor hardware. Furthermore, the choice to watermark only the perceptually least significant blue color channel helps to reduce the data volume.

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 (3112×2328 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]. Algorithm 1 shows 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

[2] The firmware source based on CHDK is available at `http://wavelab.at/sources`.

Algorithm 1 Processing the first two pixels of a packed image buffer row

```
...
prow_out = prow_in = (uint16 *) &rowbuf[PIXTOBYTES(RAW_LEFT_MARGIN+4)];
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
...
```

plain C source code. Use of SIMD assembler instructions or hardware assistance may further improve performance.

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 stage 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 4.

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 3 depicts the intercalated watermark embedding stage and the following demosaicking, post-processing and JPEG compression stages. 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: $x_w[\mathbf{m}] = x[\mathbf{m}] + \alpha \cdot w[\mathbf{m}]$ where $\mathbf{m}$ denotes pixel indices and $\alpha \geq 0$ controls the embedding strength.

The watermark detector does not know which demosaicking algorithm and post-processing operations have been applied on the watermarked raw image. Nevertheless, we can approximate the effect of the demosaicking step on the watermarked blue color component pixels with an expansion of the data with a matrix $M = [2\ 0; 0\ 2]$ which yields an image x_e 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 $\tilde{x}$. Finally, we roughly model the impact of the post-processing and JPEG compression stage as an additive noise source n.

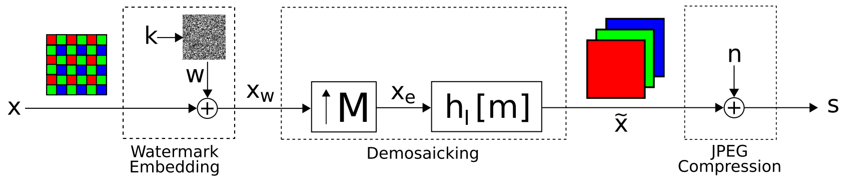

Fig. 3. Watermarking embedding and image processing pipeline

Relying on these assumptions, we can adapt the watermark detection strategy proposed by Giannoula et al. [11] for interpolated, noisy images. While the watermark is embedded in the low-resolution raw data, watermark detection takes place using the high-resolution blue channel of the demosaicked and compressed image, exploiting the watermark information spread out due to interpolation. The received demosaicked image s is split into its noisy polyphase components s_i where $0 \leq i \leq 3$ refers to one of our four components [12]. Figure 4 illustrates this process. s_0 represents the low-resolution watermarked data, corrupted by a noise component n_0, $y_0[\mathbf{m}] = s_0[\mathbf{m}] = x_w[\mathbf{m}] + n_0[\mathbf{m}]$. With the help of two linear filters for estimation and interference cancellation,

$$h_i[\mathbf{m}] = b \cdot h_I[\mathbf{m}] \quad \text{and} \quad h_i^c[\mathbf{m}] = b \cdot h_I[\mathbf{m}] * h_I[\mathbf{m}] - \delta[\mathbf{m}], \tag{1}$$

respectively, further noisy estimates of x_w are computed, such that

$$y_i[\mathbf{m}] = x_w[\mathbf{m}] + n_i[\mathbf{m}] = h_i[\mathbf{m}] * s_i[\mathbf{m}] - h_i^c[\mathbf{m}] * s_0[\mathbf{m}]. \tag{2}$$

The scaling factor b 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 $a_i \in [0, 1]$, $\sum_i a_i = 1$, depending on the estimated noise variance $\sigma_{n_i}^2$ of each component,

$$y_f[\mathbf{m}] = \sum_i a_i \cdot y_i[\mathbf{m}] \quad \text{where} \quad (a_0, ..., a_3) = \left(\frac{1}{\sigma_{n_0}^2 \sum_i \frac{1}{\sigma_{n_i}^2}}, ..., \frac{1}{\sigma_{n_3}^2 \sum_i \frac{1}{\sigma_{n_i}^2}} \right). \tag{3}$$

Giannoula et al. [11] suggest to estimate the noise variance $\sigma_{n_i}^2$ by filtering the initial component samples s_0 and subtracting the result form s_i, i.e.

$$\hat{\sigma}_{n_i}^2 = \text{var}\left(s_i[\mathbf{m}] - h_I[\mathbf{m}] * s_0[\mathbf{m}]\right) \tag{4}$$

We apply a linear correlation detector on the *fused* image. See [11] for a detailed analysis of the detector.

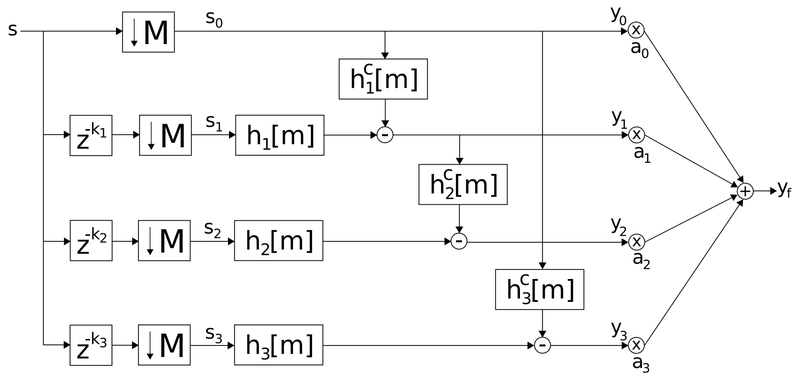

Fig. 4. Polyphase component fusion of the received image

4 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 in [9] confirm that watermark embedding in sensor data with strength $\alpha = 4$ is imperceptible.

In Figure 5, we present nine test images taken with the Canon IXUS 70 camera and corresponding detection results. A watermark is embedded in the blue channel (embedding strength $\alpha = 4$) of the raw image. Watermark detection is performed on the demosaicked image obtained with the default Adaptive Homogeneity-Directed (AHD) method [1] of the `dcraw` [3] 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 1000 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.

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

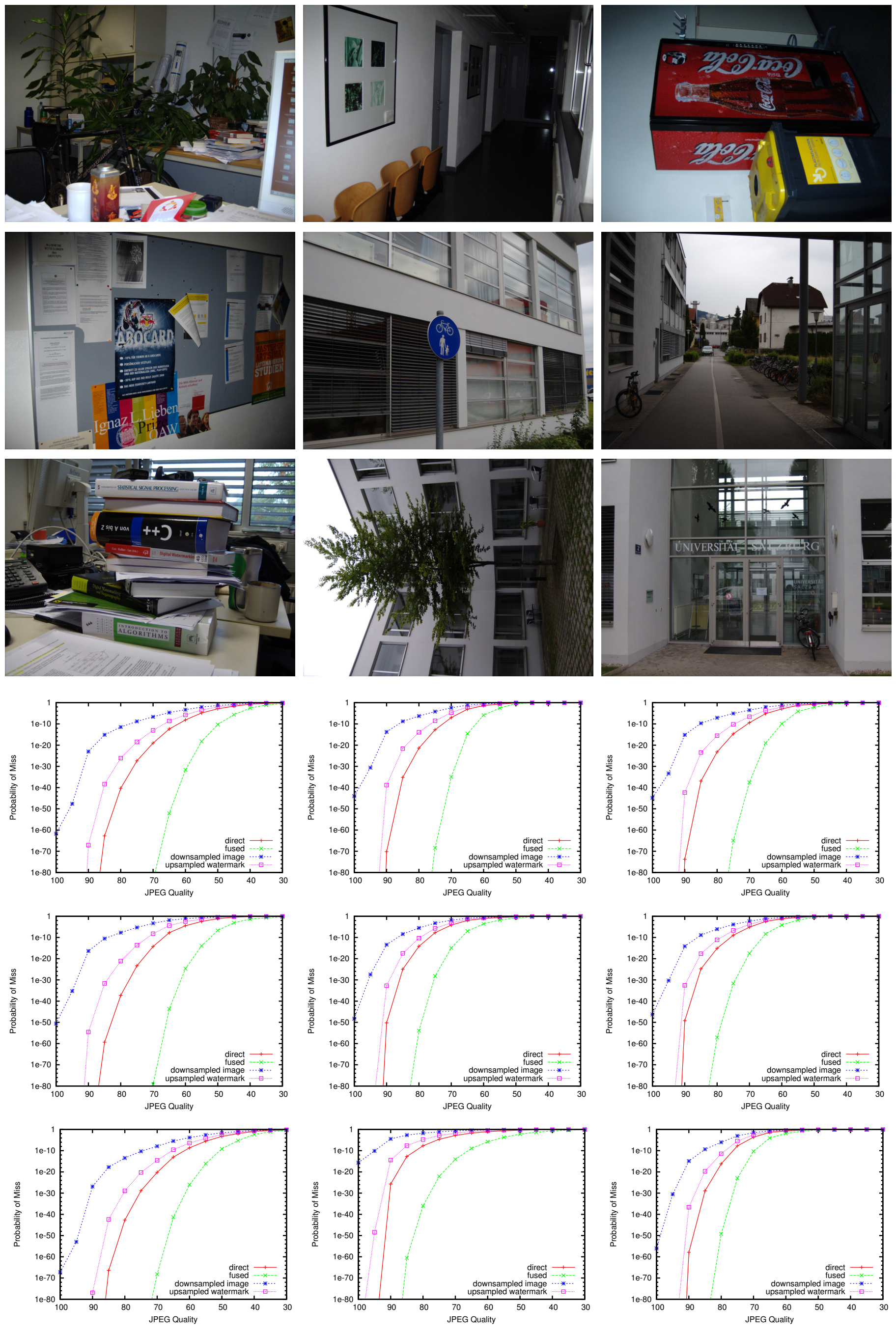

Fig. 5. Test images (3112 × 2328 pixels) and simulated watermark detection results after AHD demosaicking and JPEG compression; $P_{fa} = 10^{-6}$

In Table 1 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 nine test images with the *direct* and *fused* detector after demosaicking the raw images with the AHD [1], threshold-based Variable Number of Gradients (VNG) [2] and Patterned Pixel Grouping (PPG)[4] 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 performance 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 2. The raw data of the first test image (depicted in Figure 5) is watermarked ($\alpha = 4$) 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 (3072×2304 pixels). The smaller resolution images are upsampled to 3072×2304 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.

Table 1. Probability of missing the watermark for the demosaicking methods AHD, VNG, PPG and after JPEG compression ($Q = 70$); $P_{fa} = 10^{-6}$

Image	AHD		VNG		PPG	
	Direct	Fused	Direct	Fused	Direct	Fused
#1	$9.9 \cdot 10^{-20}$	$1.8 \cdot 10^{-84}$	$6.2 \cdot 10^{-43}$	$3.4 \cdot 10^{-294}$	$6.1 \cdot 10^{-12}$	$2.3 \cdot 10^{-29}$
#2	$9.9 \cdot 10^{-08}$	$1.1 \cdot 10^{-35}$	$1.2 \cdot 10^{-21}$	$3.3 \cdot 10^{-160}$	$7.5 \cdot 10^{-6}$	$1.2 \cdot 10^{-13}$
#3	$4.0 \cdot 10^{-10}$	$3.2 \cdot 10^{-38}$	$2.2 \cdot 10^{-23}$	$2.6 \cdot 10^{-145}$	$9.2 \cdot 10^{-7}$	$7.9 \cdot 10^{-16}$
#4	$5.3 \cdot 10^{-15}$	$5.7 \cdot 10^{-80}$	$6.5 \cdot 10^{-42}$	0.0	$6.1 \cdot 10^{-9}$	$4.8 \cdot 10^{-19}$
#5	$6.9 \cdot 10^{-5}$	$1.2 \cdot 10^{-15}$	$5.8 \cdot 10^{-19}$	$1.9 \cdot 10^{-116}$	$4.8 \cdot 10^{-4}$	$5.1 \cdot 10^{-7}$
#6	$7.3 \cdot 10^{-6}$	$3.6 \cdot 10^{-18}$	$2.5 \cdot 10^{-16}$	$2.3 \cdot 10^{-102}$	$2.1 \cdot 10^{-4}$	$3.2 \cdot 10^{-9}$
#7	$6.6 \cdot 10^{-21}$	$6.4 \cdot 10^{-69}$	$3.0 \cdot 10^{-53}$	$3.9 \cdot 10^{-289}$	$1.0 \cdot 10^{-14}$	$3.3 \cdot 10^{-29}$
#8	$1.5 \cdot 10^{-3}$	$8.5 \cdot 10^{-15}$	$8.9 \cdot 10^{-10}$	$5.7 \cdot 10^{-69}$	$1.3 \cdot 10^{-2}$	$1.3 \cdot 10^{-6}$
#9	$2.5 \cdot 10^{-4}$	$5.3 \cdot 10^{-11}$	$8.5 \cdot 10^{-15}$	$9.4 \cdot 10^{-77}$	$4.5 \cdot 10^{-3}$	$1.8 \cdot 10^{-4}$

[4] By Chuan-kai Lin, described at `http://web.cecs.pdx.edu/~cklin/demosaic/`.

Table 2. 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
3072×2304	SuperFine	$2.4 \cdot 10^{-161}$	0.0	$2.4 \cdot 10^{-15}$	$2.5 \cdot 10^{-100}$
3072×2304	Fine	$3.0 \cdot 10^{-125}$	0.0	$2.2 \cdot 10^{-15}$	$2.8 \cdot 10^{-83}$
3072×2304	Normal	$5.1 \cdot 10^{-88}$	0.0	$1.2 \cdot 10^{-14}$	$1.9 \cdot 10^{-63}$
2592×1944	SuperFine	$4.0 \cdot 10^{-68}$	0.0	$3.4 \cdot 10^{-14}$	$1.1 \cdot 10^{-50}$
2048×1536	SuperFine	$3.3 \cdot 10^{-60}$	$4.4 \cdot 10^{-223}$	$1.7 \cdot 10^{-16}$	$4.5 \cdot 10^{-46}$
1600×1200	SuperFine	$2.4 \cdot 10^{-38}$	$2.9 \cdot 10^{-117}$	$1.2 \cdot 10^{-8}$	$6.8 \cdot 10^{-29}$

5 Conclusion

Digital watermarking has to be applied close to image acquisition stage in order to 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.

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.

Acknowledgments

Supported by Austrian Science Fund project FWF-P19159-N13. Thanks to Colin M. L. Burnett for the graphics used in Figure 1.

References

1. Hirakawa, K., Parks, T.W.: Adaptive homogeneity-directed demosaicing algorithm. IEEE Transactions on Image Processing **14**(3) (March 2005) 360–369
2. Chang, E., Cheung, S., Pan, D.Y.: Color filter array recovery using a threshold-based variable number of gradients. In: Proceedings of SPIE, Sensors, Cameras, and Applications for Digital Photography. Volume 3650., San Jose, CA, USA, SPIE (January 1999) 36–43
3. Cox, I.J., Miller, M.L., Bloom, J.A., Fridrich, J., Kalker, T.: Digital Watermarking and Steganography. Morgan Kaufmann (2007)
4. Chen, M., Fridrich, J., Goljan, M., Lukas, J.: Determining image origin and integrity using sensor noise. IEEE Transactions on Information Security and Forensics **3**(1) (March 2008) 74–90

5. Popescu, A.C., Farid, H.: Exposing digital forgeries in color filter array interpolated images. IEEE Transactions on Signal Processing **53**(10) (October 2005) 3948–3959
6. Blythe, P., Fridrich, J.: Secure digital camera. In: Digital Forensic Research Workshop, Baltimore, MD, USA (August 2004)
7. Tian, L., Tai, H.M.: Secure images captured by digital camera. In: International Conference on Consumer Electronics, Digest of Technical Papers, ICCE '06, IEEE (January 2006) 341–342
8. Mohanty, S.P., Kougianos, E., Ranganathan, N.: VLSI architecture and chip for combined invisible robust and fragile watermarking. IET Computers & Digital Techniques **1**(5) (June 2007) 600–611
9. Nelson, G.R., Julien, G.A., Yadid-Pecht, O.: CMOS image sensor with watermarking capabilities. In: Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS '05. Volume 5., IEEE (May 2005) 5326–5329
10. Lukac, R., Plataniotis, K.K.: Camera image watermark transfer by demosaicking. In: Proceedings of the 48th International Symposium ELMAR '06, Multimedia Signal Processing and Communication, Zadar, Croatia (June 2006) 9–12
11. Giannoula, A., Boulgouris, N.V., Hatzinakos, D., Plataniotis, K.N.: Watermark detection for noisy interpolated images. IEEE Transactions on Circuits and Systems **53**(5) (May 2006) 359–363
12. Vaidyanathan, P.P.: Multirate digital filters, filter banks, polyphase networks, and applications: a tutorial. Proceedings of the IEEE **78**(1) (January 1990) 56–93