Image Authentication by Detecting Traces of Demosaicing

Andrew C. Gallagher[1,2]
1. Carnegie Mellon University
agallagh@cmu.edu

Tsuhan Chen[1]
2. Eastman Kodak Company

Abstract

With increasing technical advances, computer graphics are becoming more photorealistic. Therefore, it is important to develop methods for distinguishing between actual photographs from digital cameras and computer generated images. We describe a novel approach to this problem. Rather than focusing on the statistical differences between the image textures, we recognize that images from digital cameras contain traces of resampling as a result of using a color filter array with demosaicing algorithms. We recognize that estimation of the actual demosaicing parameters is not necessary; rather, detection of the presence of demosaicing is the key. The in-camera processing (rather than the image content) distinguishes the digital camera photographs from computer graphics. Our results show high reliability on a standard test set of JPEG compressed images from consumer digital cameras. Further, we show the application of these ideas for accurately localizing forged regions within digital camera images.

1. Introduction

The field of computer graphics is rapidly maturing to the point where human subjects have difficulty distinguishing photorealistic computer generated images (PRCG) from photographic images (PIM). As evidence of the proliferation of computer generated imagery, one need look no further than Hollywood. According to Wikipedia [16], the first feature-length computer animated film was Toy Story, in 1995. In 2007, a total of 14 computer animated films were released, several with stunningly realistic imagery. In addition to computer animated films, computer graphics are routinely used to create imagery in live action motion pictures that would otherwise be nearly impossible to film.

Partly because of the success of computer animation in popular culture, it is well known by the general public that images can be manipulated and are not necessarily a historical record of an actual event. When viewing movies for entertainment, the audience is usually a willing participant when fooled into believing computer generated images represent a fictional version of reality. However, in other situations, it is extremely important to distinguish between PRCG and PIM. In the mass media, there have been embarrassing instances [11] of manipulated images being presented as if they represent photographically captured events. In legal situations, where photographs are used as evidence, it is crucial to understand whether the image is authentic or forged (either computer generated or altered). Furthermore, in the intelligence community, it is of vital importance to establish the origin of an image.

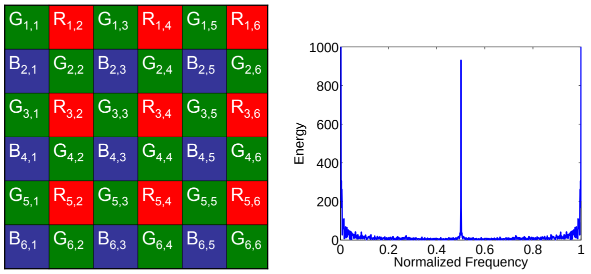

Figure 1. Most digital cameras employ an image sensor with a color filter array such as shown on the left. The process of demosaicing interpolates the raw image to produce at each pixel an estimate for each color channel. With proper analysis, traces of demosaicing are exhibited in the peak of an analysis signal as shown on the right. The presence of demosaicing indicates the image is from a digital camera rather than generated by a computer.

2. Related Work

There are several possible approaches for authenticating the source of a digital image. With active watermarking [13], an image is altered to carry an authentication message by the image capture device. At a later time, the message can be extracted to verify the source of the image. Unfortunately, this method requires coordination between the insertion and extraction of the watermark.

In contrast to the active approach, statistical methods are also used to characterize the difference between PRCG and PIM. For example, in [8], a set of wavelet features are extracted from the images to form a statistical model of PRCG and PIM, and classification is performed with standard machine learning techniques. In [10], it is shown that geometric and physical features are also effective for classifying between PRCG and PIM. In essence, both of these approaches are effective because of the lack of perfection of the state-of-the-art computer graphics. For example, in [10], it is noted that PRCG contain unusually sharp edges and occlusion boundaries. A reasonable explanation for this is that the imperfections such as dirt, smudges, and nicks that are pervasive in real scenes are difficult to simulate. It is far easier to construct a computer graphic of a gleamingly new office than the image of that office after a decade of wear. In any case, as the field of computer graphics matures with more realistic modeling of scene detail and more realistic lighting models, it seems reasonable to assume that the statistical differences between real scenes and computer generated scenes will diminish.

Meanwhile, researchers have recently shown that when an image is resampled through interpolation, statistical traces of resampling are embedded in the image signal itself [5, 6, 11]. In [5], the signature is recovered by applying a Laplacian operator to the image. The Laplacian is shown to have a higher variance at positions corresponding to pixel locations in the original uninterpolated image, and this pattern is recovered with Fourier analysis. Similarly, in [11] the EM algorithm along with Fourier analysis are used to recover the correlations between neighboring pixels that are introduced through interpolation. In addition, because a forgery is generally created by resampling an object and inserting it into a target image, this approach has been shown to be useful for detecting candidate forged image regions and is robust to JPEG compression.

Other researchers have focused on matching images to specific digital camera models [2, 14, 12] using camera-model specific properties of demosaicing. Further, in [7], the authors exploit image sensor imperfections to match images to specific cameras. These approaches demonstrate the utility of exploiting the natural watermarks that are inserted into images as a result of necessary image processing (in the case of demosaicing) or practical hardware issues (in the case of sensor imperfections).

In [12], the authors use the EM algorithm, this time for finding pixels correlated with neighbors and for estimating the coefficients of demosaicing. Based on the estimated coefficients, the image is classified into one of seven bins based on the technique used for demosaicing. Further, the probability maps can be used to suggest local tampering. This work is based mostly on simulated demosaicing without the nonlinearities associated with post-processing.

Our contributions are the following: we describe a novel approach for distinguishing between photorealistic computer graphic images and photographic images captured with a digital camera based on the idea that photographic images will contain traces of demosaicing. We recognize that finding the actual demosaicing parameters is not necessary for distinguishing between photorealistic computer graphics and photographic images. We achieve the highest reported accuracy on a standard test set for distinguishing between photographic images and photorealistic computer graphics by detecting traces of demosaicing. We demonstrate robustness by working only with images captured and processed with consumer-grade digital cameras, including the associated JPEG compression. Further, we extend our algorithm to examine images locally, accurately detecting forged regions in otherwise natural images.

3. Image Sensors and Demosaicing

Nearly all digital cameras [1] contain an image sensor with a color filter array, for example, the Bayer filter array shown in Figure 1. A filter is positioned over each photosite, sensitizing it to either the red, green, or blue component of the incident light. While other color filter array patterns and filters are sometimes used, the Bayer is the most common.

The raw image from the image sensor contains only a single signal value at each pixel position. This pixel value further corresponds to only a single color component (red, green, or blue in the case of the Bayer filter array). Typically, a demosaicing algorithm [1, 3, 15], also called color filter array interpolation, is applied to the raw image to estimate the pixel value for each color component. The interpolation can either be linear or adaptive.

With a naïve interpolation, each color channel is interpolated independently using only samples from the same color, for example, with bilinear or bicubic interpolation. In more complicated linear algorithms, interpolation is performed by considering the local pixel values of multiple color channels. For example, all of the missing green values can first be found. Then missing red pixel values are found by interpolating a red minus green differential. In even more complex nonlinear algorithms, the interpolation kernel is adaptive depending on the characteristics of the pixel values of the local neighborhood.

Generally speaking, demosaicing algorithms have several features in common. Missing color values are determined from a weighted linear combination of neighboring pixels, and the sum of the weights is one. As described in both [5] and [11], it was shown in general that interpolation of this variety leaves a signature that can be reliably detected.

Detailed analysis of the signal traces left by interpolation are found in [5, 11], so we present an example to provide intuitive understanding of our algorithm for detecting the presence of demosaicing. Considering only the green pixel values of the Bayer pattern shown in Figure 1, each missing

[1]The Foveon X3 sensor, used in the Sigma SD14 camera, is a notable exception.

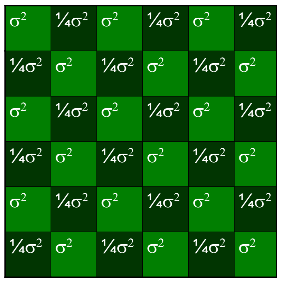

Figure 2. When demosaicing is performed with linear interpolation, the interpolated green pixels have lower variance than the original green pixels. Assuming pixel values from green photosites in the Bayer array are IID with variance σ^2, this image represents the variance from which each pixel value is drawn. The spatial pattern of variances is the basis for detecting the presence of demosaicing.

green pixel value can be interpolated from its four nearest neighbors using bilinear interpolation:

$$\hat{g}(x,y) = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}\begin{bmatrix} g(x-1,y) \\ g(x,y-1) \\ g(x+1,y) \\ g(x,y+1) \end{bmatrix} \quad (1)$$

Assuming that the original green pixel values are IID and drawn from a normal distribution with variance σ^2, the estimated green pixel values can be shown to have a variance of only $\frac{1}{4}\sigma^2$. As shown in Figure 2, the green channel is partitioned into two interleaved quincunx patterns, one corresponding to the original green pixel locations, and the other corresponding to the estimated green pixel locations with lower variance. This analysis oversimplifies the demosaicing and omits the nonlinear image processing for the purpose of illustration. The important point to recognize is that demosaicing introduces periodic patterns into the image signal. In a sense, demosaicing is a form of *passive watermarking*, because the signal processing necessitated by the image sensor leaves a signal in the image. By extracting and recognizing these periodic signals embedded within the image, the source of the image is surmised.

4. Detecting Traces of Demosaicing

An interpolated pixel value is produced with a weighted linear combination of neighboring pixel values. The weights directly affect the variance of the distribution from which the interpolated pixel value is drawn. This pattern of variances can be detected and is the basis for detecting demosaicing. In our implementation, we consider only the green channel of the image to demonstrate our approach. The other color channels (or differences between color channels) can be analyzed in a similar manner.

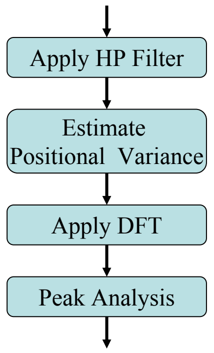

Figure 3. Flow diagram for detecting demosaicing. First a highpass filter is applied, then the variance of each diagonal is estimated. Fourier analysis is used to find periodicities in the variance signal, indicating the presense of demosaicing.

Figure 3 shows the steps in the algorithm for extracting features related to demosaicing. First, the image $i(x,y)$ is convolved with a highpass operator $h(x,y)$ in order to remove low frequency information and enhance the embedded periodicity when demosaicing has occurred. We select the operator:

$$h(x,y) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad (2)$$

Assuming once again that the original green photosites are drawn from a distribution with variance σ^2, the variance of the output of the operator $h(x,y)$ on the green channel can be found, if we again make the simplifying assumption that the green channel is interpolated with linear interpolation:

$$\sigma_o^2 = 4\left(\frac{1}{4}\right)^2\sigma^2 + 4\left(\frac{1}{2}\right)^2\sigma^2 + (1-4)^2\sigma^2 \quad (3)$$

$$= \frac{41}{4}\sigma^2 \quad (4)$$

$$\sigma_i^2 = 0\sigma^2 \quad (5)$$

σ_o^2 is the variance of the output of application of $h(x,y)$ at positions corresponding to original green photosites in the image sensor, and thus nine pixel values from the original sensor contribute to the filter output, four with a coefficient $\frac{1}{4}$, four with a coefficient $\frac{1}{2}$, and position (x,y) itself has coefficient -3. σ_i^2 corresponds to locations where the green value is interpolated (i.e., red or blue photosites), assuming the green channel is interpolated with linear interpolation. In fact, if missing green values were actually estimated with linear interpolation and all other image processing operations in the camera are ignored, then application of the filter $h(x,y)$ yields a value of zero at each pixel location with an interpolated green value. The choice of $h(x,y)$ was made to maintain a large value for $\frac{\sigma_o^2}{\sigma_i^2}$ (in our case, assuming linear interpolation, infinite) and testing using a small number of training images. A large ratio of $\frac{\sigma_o^2}{\sigma_i^2}$ aids in the detection of the periodic pattern of variances characteristic of demosaicing.

Again, we emphasize that the bilinear interpolation example is merely illustrating the mechanics of how traces of demosaicing are recovered from a photographic image. In practice, the situation is far more complex. Our test images are finished images from real consumer cameras where the demosaicing is actually performed with a nonlinear filter, the color filter array pattern is not known, and the image processing path contains nonlinear operations such as noise supression, color enhancement, and JPEG compression. Our algorithm make no assumptions concerning the linearity of the demosaicing, only that the variance of interpolated pixels is distinguishable from the variance of the original pixels. In our experiments (Section 5), we demonstrate that despite these nonlinear complications, the traces of demosaicing are still detectable and useful for distinguishing PIM from PRCG and for accurately detecting evidence of local tampering.

Next, estimates of the variance of each photosite are made using Maximum Likelihood Estimation. After application of $h(x, y)$, each pixel value is assumed to be drawn from a normal distribution with a particular variance, and the variance along diagonals is assumed to be constant for images that have undergone demosaicing (note the variance varies periodically across different diagonals). To compute a MLE estimate of the variance, the statistical variance of the pixel values along each diagonal is found. In keeping with the work of [5], in place of actually computing variance, we use the computationally simpler mean of the absolute values of each diagonal in the image. This projects the image down to a single-dimension signal, $m(d)$, were $m(d)$ represents the estimate of the variance corresponding to the d^{th} diagonal.

$$m(d) = \frac{\sum_{x+y=d} |h(x, y) * i(x, y)|}{N_d} \tag{6}$$

where N_d is the number of pixels along the d^{th} diagonal and is used for normalization.

To find the periodicity in $m(d)$, the DFT is computed to find $|M(e^{j\omega})|$. A relatively high peak at frequency $\omega = \pi$ indicates that the image has undergone interpolation by a factor of two and is characteristic of demosaicing. The peak magnitude at $\omega = \pi$ is quantified as follows:

$$s = \frac{|M(e^{j\omega})|_{\omega=\pi}}{k} \tag{7}$$

where k is the median value of the spectrum, excluding the DC value. Normalizing by k was found to be important to distinguish between true demosaicing and images containing signals or noise with large energy across the frequency spectrum.

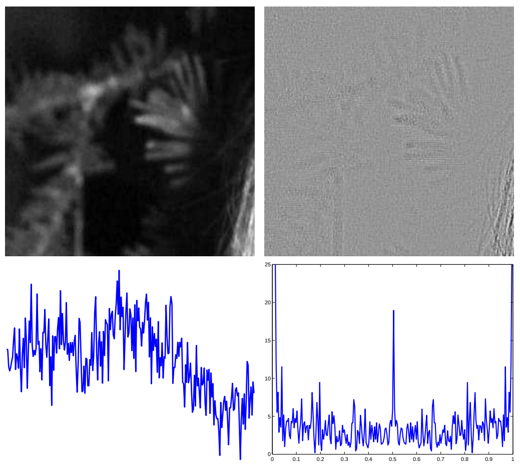

Figure 4. Example signals generated by our algorithm. Top Left: a 256×256 portion of the green channel from an original image (of pine tree branches). Top Right: The absolute value of the image after application of the filter $h(x, y)$. Bottom Left: The signal $m(d)$, which represents an estimate of the variance along each image diagonal. Bottom Right. The spectrum of $m(d)$, exhibiting the characteristic peak at $\omega = \pi$ (or normalized frequency $f = 0.5$). Generally, uninterpolated images do not exhibit a peak in the spectrum of $m(d)$.

Figure 4 shows examples of the signals produced by our feature extraction for a small portion of an image. Figure 5 shows the spectrum $|M(e^{j\omega})|$ for a selection of PIM and PRCG from the ADVENT database [9].

5. Experiments

We validate our approach for two tasks: distinguishing PIM from PRGC and accurately localizing tampered image regions. We emphasize that all of our photographic images are compressed JPEG images directly from the camera. Therefore they have undergone real-world demosaicing, nonlinear rendering, and JPEG compression.

5.1. Distinguishing PIM from PRCG

To validate our approach, we use Columbia's ADVENT dataset from [9]. This set contains 2400 images, including 800 personal PIM from the authors of [9] and Philip Greenspun (*personal*), 800 PIM from Google Image Search (*google*), and 800 PRCG from various 3D artist websites (*CG*). In our work, we omit the *google* images because their origin is not well known (i.e., it appears many have been resized). The images contain a wide variety of subjects such as people, animals, objects, and architecture. Samples of both PIM and PRCG are shown in Figure 5.

For each image, the score s is computed as described in Section 4. An image is classified as PIM if its value s exceeds a threshold t, and as PRCG otherwise. By varying t,

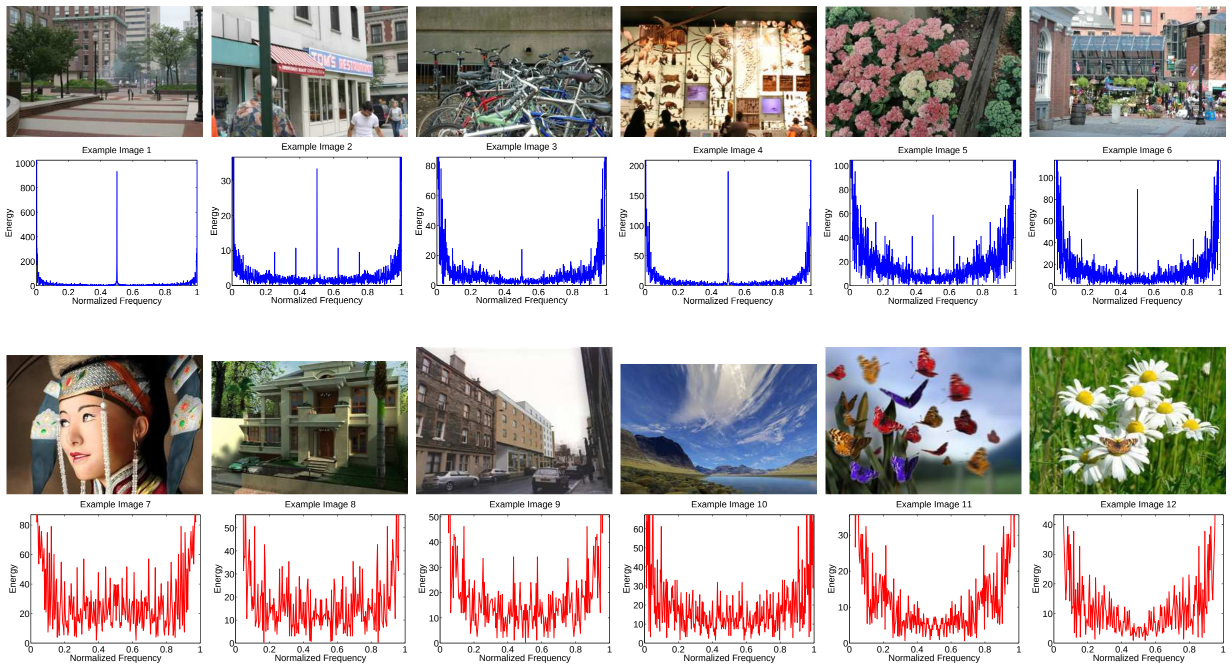

Figure 5. Image examples. The top row shows size PIM from [9]. Beneath each image is displayed the analysis signal. In each case, there is a strong peak present at $f = 0.5$. The third row shows PRCG, also from [9]. These images generally have no discernable peak at $f = 0.5$.

performance curves are generated. Figure 6 shows the result of our experiments. At the point on the performance curve with the fewest errors, only 25 of 1600 images are misclassified (98.4% accuracy). The threshold value corresponding to this operating point is 4.49. Figure 8 shows a selection of PIM that were classified as PRCG. These images seem to contain a large amount of image structure that is not removed by the highpass filter and swamps out the demosaicing signal. Figure 9 shows several PRCG images that were classified as PIM. Our algorithm's accuracy far surpasses the accuracy of about 85% reported by [10], although we omitted the 800 *google* images. Most of the *google* images have undergone subsampling and other image processing operations that destroy the demosaicing watermark. Therefore, it can be said that our method is highly effective at distinguishing between images directly captured at native resolution by a digital camera and other images.

Because our algorithm is dependent on using many pixel samples to estimate variance statistics, the performance suffers when the algorithm is limited to using a central small window of the image as shown in Figure 6. However, even considering a small 64×64 region of the image results in a correct classification rate of about 66%.

Figure 7 demonstrates that our approach is robust even with respect to severe JPEG compression. We recompress the entire ADVENT set using a JPEG Qualify Factor of either 95, 90, 80, 60, 40, 20. The classification accuracy only

Window Size	Native	1024	512	256	128	64
Area under ROC	0.99	0.97	0.94	0.89	0.80	0.71

Table 1. The area under the ROC curve (see Figure 6) as a function of the window size from an image that is inspected.

JPEG Quality Factor	95	90	80	60	40	20
Area under ROC	0.99	0.93	0.91	0.90	0.87	0.82

Table 2. The area under the ROC curve (see Figure 7) as a function of the JPEG quality factor used to recompress the image.

gradually decreases, despite the heavy compression (with a quality factor of 20, the images were compressed by an average of greater than 80:1).

Tables 1 and 2 provide an alternate view of the ROC curves of Figures 6 and 7 by reporting the area under the ROC curves.

Demosaicing is necessarily one of the first operations applied to an image in the digital camera. Subsequent processing includes nonlinear operations such as noise reduction, sharpening, balance, and tonal adjustments as well as compression (JPEG in our case.) The photographic images in our test set include images captured by the Canon 10D and Nikon D70 digital cameras. This experiment provides evidence that the traces of demosaicing survive these image processing operations that can even vary between camera manufacturers.

Figure 8. A selection of photographic images (PIM) that were mistakenly classified as photorealistic computer graphic images (PRCG).

Figure 9. A selection of photorealistic computer graphic images (PRCG) that were mistakenly classified as photographic images (PIM).

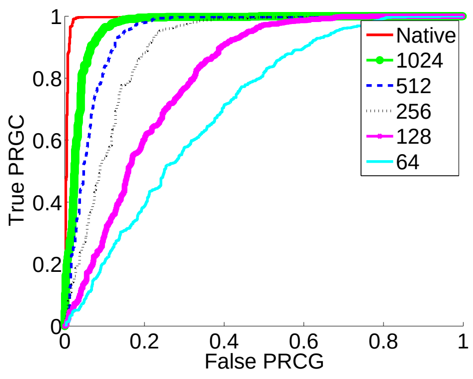

Figure 6. The performance of our algorithm for distinguishing between photographic images and photorealistic computer graphics from the ADVENT dataset [9]. Each curve reports the performance using only a central square window of a specific size of pixels from each image (though "native" includes the entire image). As expected, when the classification is performed on fewer pixels, the performance degrades.

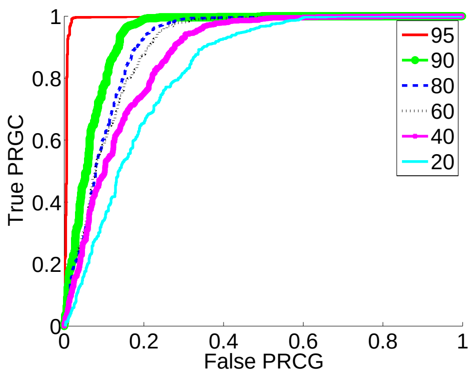

Figure 7. Our algorithm's classification accuracy gracefully fades as a function of compression amount. Each curve reports the classification performance for distinguishing between PRGC and PIM on the ADVENT dataset for size different compression levels.

5.2. Detecting forged image regions

With minor modification, the algorithm described in Section 4 can be applied locally to detect regions of an image that have possibly been tampered with. The intuition is this: demosaicing produces periodic correlations in the image signal. When a forgery is made, an image piece from another source (either another image or a computer graphic) is pasted over a portion of the image. In general, this image piece is resampled to match the geometry of the image. The resampling modifies the correlations of the image piece (and actually introduces new correlations as shown in [11]).

The application of the highpass filter is the same as previously described. Estimating the variance becomes a local operation:

$$m(x,y) = \frac{1}{2n+1} \sum_{i=-n}^{n} o(x+i, y+i) \qquad (8)$$

where $o(x,y) = |h(x,y) * i(x,y)|$, the absolute value of the output of applying the filter $h(x,y)$ to the image $i(x,y)$. The parameter n is the size of the local neighborhood; by default we use $n = 32$. At each position (x,y), a local (256 point) one-dimensional DFT is computed along each row, and the local peak ratio $s(x,y)$ is computed as described before in Eq. (2).

We created three forged images to test this algorithm as shown in Figure 10 [4]. The forgeries were created in Photoshop, by either inserting an object from another image into the target image, or by copying and rescaling an object from the target image itself (in the case of the sheep image). For each image, the local version of our algorithm is applied and the local peak signal $s(x, y)$ is found. The forged region does not contain the expected traces of demosaicing, which results in low values of $s(x, y)$. Using our algorithm, we were able to accurately localize the forged regions in each of the three test images.

6. Discussion

Our algorithm effectively detects the presence of demosaicing in a digital image. Because computer graphics systems do not use an image sensor, we conclude that images containing traces of demosaicing are likely to be photographic images. However, one could argue that a malicious computer animator wishing to add an element of realism to her computer graphic images could simply insert a software module to simulate the effect of color filter array sampling and then apply demosaicing. This attack admittedly will confuse our algorithm, but this type of attack (i.e. modifying or adding image signal components specifically to exploit the assumptions of the algorithm) is a weakness of this entire class of algorithms [5, 8, 10, 11, 12]. For example, modifying the high frequency components of PRCG might confuse the classifier in [8], which relies on the fact that most PRCG do not have as much detail as PIM.

We believe that there will be no single foolproof method for distinguishing between photographic and computer generated or manipulated images. Rather, an arsenal of tests will be necessary, especially as computer graphic capabilities continue to advance. Forgers may fall into one or several traps when attempting to propose a photorealistic computer graphic image is a genuine event.

7. Conclusion

We describe a novel approach to distinguish between photographic images and photorealistic computer generated images. Rather than focusing on characteristics of the scene itself, we exploit the image processing necessitated by the camera hardware. In particular, we note that most cameras' image sensors contain a color filter array and demosaicing must be used to produce three-color images. Demosaicing acts as a type of passive watermarking that leaves a trace embedded within the image signal. When traces of demosaicing are detected, we surmise that the image is a photographic (rather than computer generated) image.

We document the performance of the algorithm on a standard test set of 1600 images compressed with JPEG compression, and achieve classification accuracy in the upper nineties. Further, we show the application of the algorithm for accurately localizing forged image regions. The remarkable attribute of our approach is that is works so well on real images from images from various manufacturers that have undergone all of the nonlinear camera processing such as tonal adjustment and JPEG compression. We demonstrate that it is not necessary to recover the demosaicing parameters to authenticate images using evidence of demosaicing.

References

[1] J. Adams and J. Hamilton. Design of practical filter array interpolation algorithms for digital cameras. *Proc. SPIE*, 1997.

[2] S. Bayram, H. T. Sencar, and N. Memon. Source camera identification based on cfa interpolation. In *Proc. ICIP*, 2005.

[3] D. Cok. Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal. *U.S. Patent 4,642,678*, 1986.

[4] A. Gallagher. A small tampered image database. http://amp.ece.cmu.edu/people/Andy/authentication.html.

[5] A. Gallagher. Detection of linear and cubic interpolation in JPEG compressed images. In *Proc. CRV*, 2005.

[6] A. Gallagher. Method for detecting image interpolation. *U.S. Patent 6,904,180*, 2005.

[7] J. Lukas, J. Fridrick, and M. Goljan. Determining digital image origin using sensor imperfections. *Proc. SPIE*, 2005.

[8] S. Lyu and H. Farid. How realistic is photorealistic? *IEEE Transactions on Signal Processing*, 2005.

[9] T.-T. Ng, S.-F. Chang, J. Hsu, and M. Pepeljugoski. Columbia photographic images and photorealistic computer graphics dataset. Technical report, Columbia University, 2005.

[10] T.-T. Ng, S.-F. Chang, J. Hsu, L. Xie, and M.-P. Tsui. Physics-motivated features for distinguishing photographic images and computer graphics. In *Proc. ACM MM*, 2005.

[11] A. Popescu and H. Farid. Exposing digital forgeries by detecting traces of resampling. *IEEE Trans. on Signal Processing*, 2005.

[12] A. Popescu and H. Farid. Exposing digital forgeries in color filter array interpolated images. *Trans. on Signal Processing*, 2005.

[13] V. Potdar, S. Han, and E. Chang. A survey of digital image watermarking techniques. *Proc. Industrial Informatics*, 2005.

[14] A. Swaminathan, M. Wu, and K. J. R. Liu. Non-intrusive forensic analysis of visual sensors using output images. In *Proc. ICASSP*, 2006.

[15] C.-Y. Tsai and K.-T. Song. A new edge-adaptive demosaicing algorithm for color filter arrays. *Image Vision Comput.*, 2007.

[16] Wikipedia. list of computer-animated films. Downloaded March 2008. http://en.wikipedia.org/wiki/List_of_computer-animated_films.

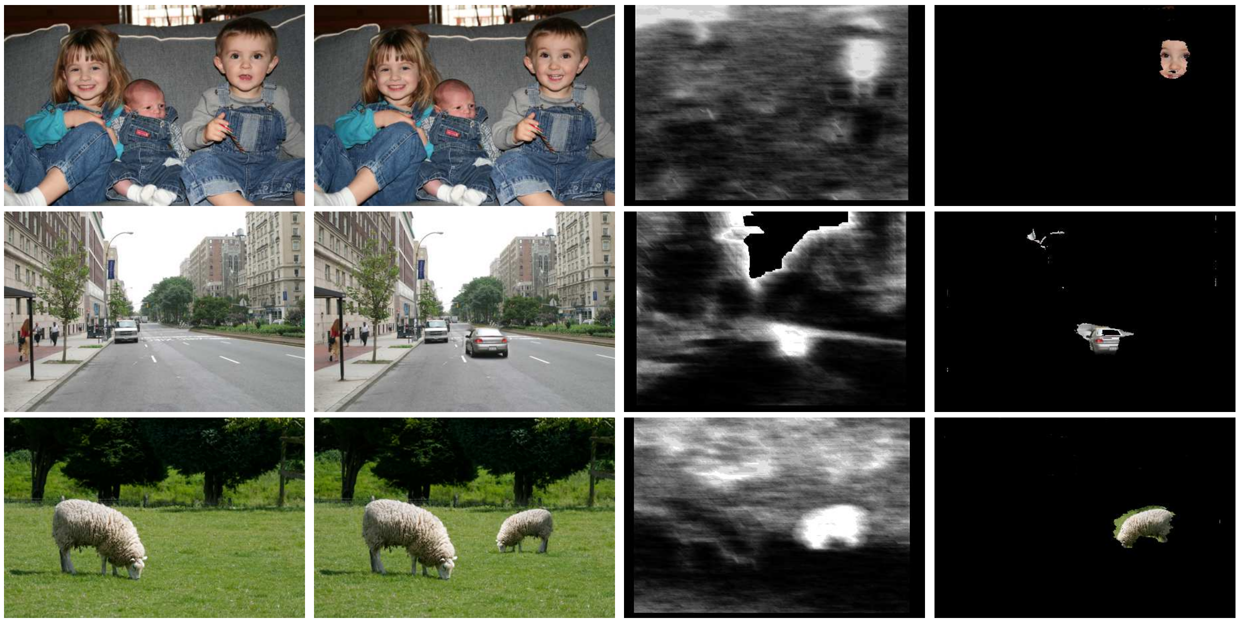

Figure 10. When applied locally, our algorithm is useful for accurately localizing forged image regions. The first column shows the original images (captured by a Canon EOS Digital Rebel XT, a Canon 10D, and a Canon EOS Digital Rebel XTi, respectively. The next column shows the forgeries that were hand-made using Photoshop by splicing other image portions into the target images. The child on the right was modified to smile by superimposing a face from another image. A car from another image was added to the street scene. The sheep was duplicated and pasted back into the same image. Column 3 shows an image of the local peak ratio $s(x, y)$. Bright shades correspond to higher likelihood for being a forged image portion (lower values of $s(x, y)$.) The black area in the map of $s(x, y)$ is a result of a numerical singularity resulting from the clipped sky pixels. The last column shows likely forged regions in each image (i.e., regions with corresponding values of $s(x, y)$ that are smaller than threshold t. In each case, the forged region is found.