Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis[☆]

Zhouchen Lin[a,*], Junfeng He[b], Xiaoou Tang[a], Chi-Keung Tang[b]

[a]*Microsoft Research, Asia, Zhichun Road #49, Haidian District, Beijing 100190, PR China*
[b]*The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, PR China*

ARTICLE INFO

Article history:
Received 12 October 2007
Received in revised form 24 January 2009
Accepted 7 March 2009

Keywords:
Pattern analysis
Tampered image detection
JPEG
DCT coefficient
Double quantization

ABSTRACT

The quick advance in image/video editing techniques has enabled people to synthesize realistic images/videos conveniently. Some legal issues may arise when a tampered image cannot be distinguished from a real one by visual examination. In this paper, we focus on JPEG images and propose detecting tampered images by examining the double quantization effect hidden among the discrete cosine transform (DCT) coefficients. To our knowledge, our approach is the only one to date that can automatically locate the tampered region, while it has several additional advantages: fine-grained detection at the scale of 8 × 8 DCT blocks, insensitivity to different kinds of forgery methods (such as alpha matting and inpainting, in addition to simple image cut/paste), the ability to work without fully decompressing the JPEG images, and the fast speed. Experimental results on JPEG images are promising.

1. Introduction

Nowadays the saying "to see is to believe" might not be true any more. The trustworthiness of photos is being undermined by the advance of image/video editing techniques (e.g., [1–7], to name just a few). There has been a long history of image forgery. In the early days, dark-room skills were used to print multiple fragments of photos onto a single photograph paper. In the current digital era, image/video forgery becomes much easier. The techniques involve naive cutting and pasting [20], matting for perfect blending [3,5], graph cut for finding optimal composition boundaries [2,7], texture synthesis [2,6], and variational approaches [1,4] for synthesizing new contents. With these more and more sophisticated techniques, realistic synthetic images/videos can be produced conveniently without leaving noticeable visual artifacts (e.g., Fig. 1(a)). This has caused many problems, including legal and ethical issues [11]. For example, Walski presented news reports with degraded fidelity[1] and the internet image showing Fonda and Kerry sharing the same speaker platform[2] has caused some damage to Kerry's career. Therefore, developing technologies to judge whether the content of an image/video has been altered is very important. This paper contributes by addressing such a difficult problem in computational photography and vision, and provides the first fast, fully-automatic, and fine-grained algorithm to detect tampered JPEG images. We shall demonstrate the encouraging results that have been obtained.

1.1. Related work

Image forensic technologies can be categorized as active ones and passive ones. Active image forensic methods mainly insert digital watermark [8] to images/videos at the instant of their acquisition. The integrity of images/videos can be checked by detecting the change in the watermark. This would require that the image/video capturing devices being equipped with watermarking functionality, and that both the watermark embedder and detector be standardized. Unfortunately, up to date this is still not true for most of the commodity cameras, leaving plenty of images/videos unprotected by watermark. On the other hand, watermarking relies on the assumption that the digital watermark cannot be easily removed and reinserted, which is not guaranteed either [9]. These limitations prevent broad applications of active approaches.

In contrast, passive image forensic aims at developing technologies for tampered image/video detection without using *any*[3]

[☆] A preliminary version of this paper appeared on ECCV'06 [26].

[*] Corresponding author. Tel.: +86 10 58963143; fax: +86 10 88099511.
E-mail addresses: zhoulin@microsoft.com (Z. Lin), hejf@ust.hk (J. He), xitang@microsoft.com (X. Tang), cktang@cse.ust.hk (C.-K. Tang).

[1] Available at: http://angelingo.usc.edu/issue01/politics/ward.html

[2] Available at: http://urbanlegends.about.com/library/bl_kerry_fonda.htm

[3] As a mid-way approach, Lukas et al. [16] require either the camera that took the images or other images taken by the same camera.

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.03.019

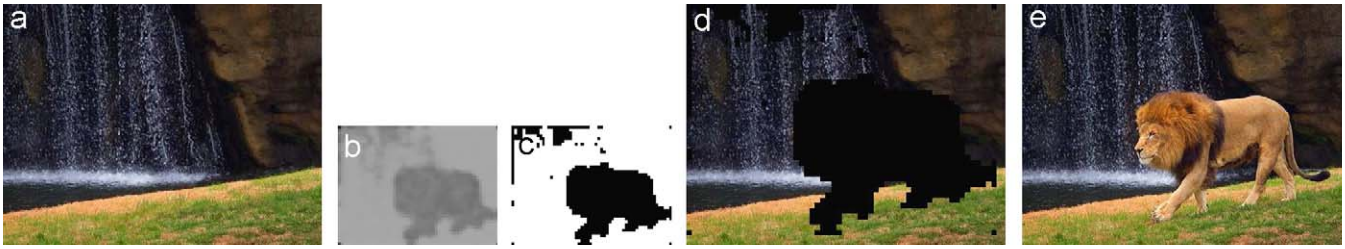

Fig. 1. Examples of our detection results. (a) Tampered JPEG image by masking the lion and inpainting with structure propagation [6]. (b) Block posterior probability map (BPPM) of (a). (c) The segmentation of (b). (d) The output of the tampered region (shown in black) according to (c). For comparison, the original image is given in (e).

knowledge beyond the image/video itself [10]. Rather, it focuses on checking various kinds of consistency in the image/video. Farid et al. have performed some pioneering work in this direction. They have proposed some statistics of images that may be changed after tampering [13], including the interpolation relationship among the nearby pixels if resampling was done during synthesis [15], the double quantization (DQ) effect of two JPEG compression steps with different qualities, before and after the images are synthesized [13], the gamma consistency via blind gamma estimation using the bicoherence [17], the signal to noise ratio (SNR) consistency, and the color filter array (CFA) interpolation relationship among the nearby pixels [14]. They have also proposed checking the lighting consistency [12] and detecting the duplicated regions [18] and the chromatic aberration [19]. Bicoherence is also used by Ng et al. [20] to detect spliced images. Lin et al. [25] and Hsu and Chang [21,22] also considered exploiting the camera response function (CRF) to detect image composition, where the former computed the whole CRF curves from different sets of image patches to check their normality and consistency, whereas the latter modeled the CRF by a linear exponent function and estimated the corresponding parameters. A more recent survey on the above-mentioned techniques can be found in [24].

However, all the existing literature only report the *possibility* of using the proposed statistics or low-level cues to detect image forgery. More specifically, most of the work only shows its effectiveness by comparing the difference in the characteristics of the authentic region and the forgery region which are usually known *a priori*. To our best knowledge, no complete solution has been reported to determine the forgery region *blindly* and *automatically*. For example, although the computation of the bicoherence [20] and the detection of the periodic resampling pattern [15] and the interpolation pattern [14] can judge at some accuracy whether each *given* image block is forgery, to determine the forgery region blindly requires testing a large amount of image blocks across the whole image, possibly resulting in a prohibitive computation load. As a result, more or less, the suspicious regions have to be prescribed manually. However, in some cases, manual prescription may not be easy. For example, it is hard to guess the suspicious region if the image is synthesized by inpainting or texture synthesis after removing an object (Figs. 1(a) and 9(a) and (e)). Moreover, none of the previous work reports how to fuse the local judgement on image blocks into a global decision to determine the forgery region. Consequently, all the previous work can only produce coarse-grained output of the forgery region, i.e., at the level of large-size regions (e.g., at a size of 128 × 128 [20]) or only decides that the given image is fake [10,21,22].

In this paper, we propose a fast and fully automatic detection method for JPEG images. The reason we target JPEG images is because JPEG is the most widely used image format. Particularly in digital cameras, JPEG may be the most preferred image format due to its efficiency of storage. Our method is based on the DQ effect (to be detailed in Section 2.2. Intuitively speaking, the DQ effect is the exhibition of periodic peaks and valleys in the histograms of the discrete cosine transform, DCT, coefficients.) in forged JPEG images and can produce fine-grained output of the forgery region at the scale of 8 × 8 image blocks.

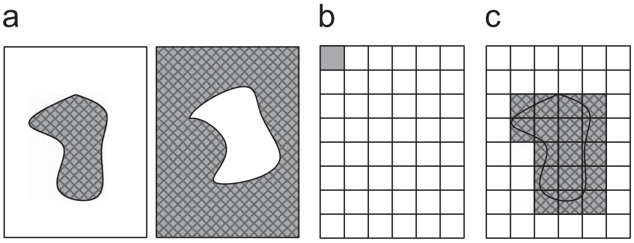

Fig. 2. Illustrations of some terminologies. (a) A tampered image must contain the unchanged region (blank area) and the tampered region (shaded area). Note that the unchanged region can either be the background (left figure) or the foreground (right figure). (b) A DCT block is a group of pixels in an 8 × 8 window on which DCT is operated when compression. The gray block is one of the DCT blocks. The DCT grid is a grid that partitions the image into DCT blocks. (c) A tampered block (shaded blocks) is a DCT block that is inside the tampered region or across the synthesis edge. An unchanged block (blank blocks) is a DCT block that is completely inside the unchanged region.

Although DQ effect was already presented in [13,27] and the underlying theory was also exposed therein, those papers actually only *suggested* that DQ effect could be utilized for image authentication: they only verified, under different compression qualities, that double compression can result in detectable DQ effect, but did not explore how to utilize DQ effect for image forgery detection. Moreover, they believed that those images having DQ effects are possibly forgery. We find that this is *not* true (see Section 3.1). On the other hand, our algorithm is much more sophisticated than simple DQ effect detection. To date, our paper is the only one that applies DQ effect for image forgery detection. The basis of our proposed methodology is the exact *opposite* to that of [13,27]: the regions that do *not* have the DQ effect are possibly forged.

Ye et al. [23] also detected image forgery by analyzing the histograms of DCT coefficients. However, their method needs user assistance to segment an image into correct regions, in order to estimate the quantization tables and compute the blocking artifact measures appropriately.

1.2. Terminologies

To proceed, we first give some definitions (Fig. 2). That an image is "tampered" (Fig. 2(a)) means part of the content of a real image is altered. Note that this concept does not include those wholly synthesized images, e.g., an image completely rendered by computer graphics or by texture synthesis. But if part of the content of a real image is replaced by those synthesized or copied data, then it is

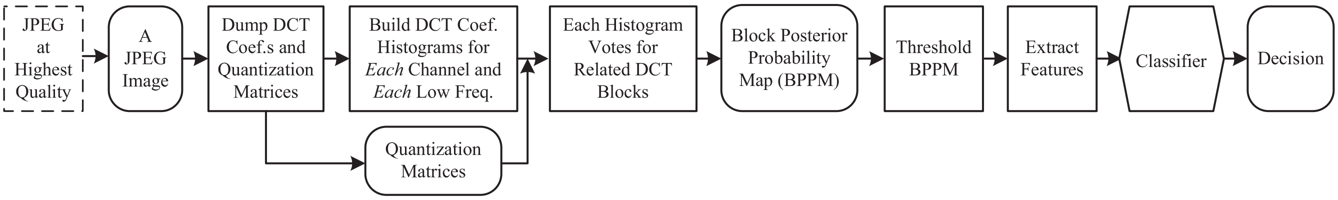
JPEG at Highest Quality
A JPEG Image
Dump DCT Coef.s and Quantization Matrices
Build DCT Coef. Histograms for Each Channel and Each Low Freq.
Each Histogram Votes for Related DCT Blocks
Block Posterior Probability Map (BPPM)
Threshold BPPM
Extract Features
Classifier
Decision
Quantization Matrices

Fig. 3. The work flow of our algorithm. The dashed block at the left is an optional step if the given test image is stored in a lossless format. The classifier has to be trained in advance, where the training images also undergo all the steps before "Classifier".

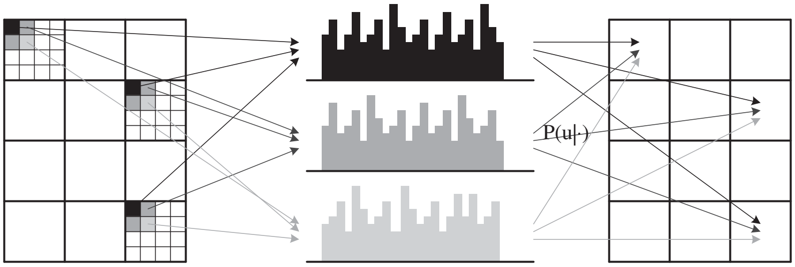

Fig. 4. The illustration of building histograms of DCT coefficients for each of YUV channel and each frequency and voting for the probabilities of the DCT blocks to be unchanged. Only three blocks, three histograms, and three arrows are shown for clarity of the figure.

viewed as "tampered". In other words, that an image is tampered implies that it must contain two parts: the unchanged region and the tampered region. A DCT block (Fig. 2(b)), or simply called a "block", is a group of pixels in an 8 × 8 window. It is the unit of DCT that is used in JPEG. A DCT grid is the horizontal lines and the vertical lines that partition an image into blocks during JPEG compression. A tampered block (Fig. 2(c)) refers to a block in the tampered region or along the synthesis edge and an unchanged block is a block in the unchanged region.

1.3. Outline of our approach

Fig. 3 shows the work flow of our algorithm. Given a JPEG image, we first dump its DCT coefficients and quantization matrices for YUV channels. If the image is originally stored in other lossless format, we first convert it to the JPEG format at the highest compression quality. Then we build histograms for each channel and each frequency (Fig. 4). Note that the DCT coefficients are of 64 frequencies in total, varying from (0,0) to (7,7). For each frequency, the DCT coefficients of all the blocks can be gathered to build a histogram. Moreover, a color image is always converted into the YUV space for JPEG compression. Therefore, we can build at most $64 \times 3 = 192$ histograms of DCT coefficients of different frequencies and different channels. However, as high frequency DCT coefficients are often quantized to zeros, only the histograms of low frequencies of each channel are useful. For each block in the image, using one histogram we can compute one probability of it being a tampered block, by checking the DQ effect of this histogram (more details will be presented in Section 3.2). With all the available histograms, we can accumulate the probabilities to give the posterior probability of this block being unchanged (Fig. 4). Then the block posterior probability map (BPPM) is thresholded to differentiate the *possibly* tampered region and *possibly* unchanged region. With such a segmentation, a four-dimensional feature vector is computed for the image. Finally, a trained SVM is applied to decide whether the image is tampered. If it is tampered, then the segmented tampered region is also output.

Our method has several advantages. First, it is capable of locating the tampered region automatically without the user to prescribe the suspicious region. This is a feature that is rarely possessed by the existing methods. The duplicated region detection [18] may be the only exception. But copying a part of an image to another position of the same image is not a common practice in image forgery. Second, the detection is at a fine-grained scale, i.e., at the level of DCT blocks, while the previous methods [20,10,13,15,14,25] can only produce coarse-grained output. Third, most of the existing methods aim at detecting tampered images synthesized by the cut/paste skill ([20,21], interpolation or resampling may precede [13,15]). In contrast, our method could deal with images whose tampered region is produced by various kinds of methods such as inpainting [1,6], alpha matting [3,5], texture synthesis [2], and other editing skills [4] besides image cut/paste. Third, our algorithm directly analyzes the DCT coefficients without fully decompressing the JPEG image. This greatly saves the memory cost and the computation load. Finally, our method is much faster than the bi-coherence based approaches [13,20], iterative methods [13], and the CRF based algorithms [25,21,22].

The rest of this paper is organized as follows. We first give the background of our approach in Section 2, then introduce the core part of our algorithm in Section 3. Next we present the experimental results in Section 4. Finally, we conclude our paper with discussions and future work in Section 5.

2. Some background

2.1. The model of image tampering and the double JPEG compression

We model the image tampering process in three steps (Fig. 5):

(1) Choose a JPEG-compressed image I_1 and decompress it.
(2) Replace a region of I_1 by pasting or matting (either with interpolation or resampling or not) a region from another image I_2 (either in JPEG or not), or inpainting or synthesizing new content inside the region.
(3) Save the forgery image in JPEG or any lossless format.[4]

[4] In this case, we will re-save the image as JPEG with a compression quality 100 when detection starts (Fig. 3). Note that most of the existing image formats other than JPEG and JPEG2000 are lossless.

Please cite this article as: Z. Lin, et al., Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis, Pattern Recognition (2009), doi: 10.1016/j.patcog.2009.03.019

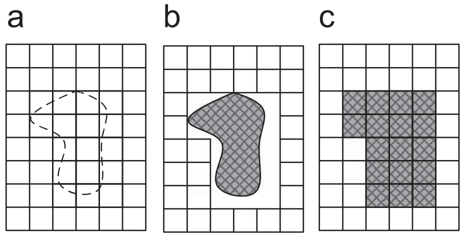

Fig. 5. The three steps of our JPEG image tampering model. (a) Decompress a JPEG image (the grid represents DCT blocks) and choose (implicitly or explicitly) a region (inside the dashed curve) to alter. (b) Replace the region (shaded) with new content. (c) Recompress the tampered image. After these three steps, the unchanged region (blank blocks) is doubly compressed while the tampered region (shaded blocks) is singly compressed.

To explain the DQ effect that results from double JPEG compression, we shall give a brief introduction of JPEG compression. The compression of JPEG images involves three basic steps [28]:

(1) DCT: An image is first divided into DCT blocks. Each block is subtracted by 128 and transformed to the YUV color space. Finally DCT is applied to each channel of the block.
(2) Quantization: the DCT coefficients are divided by a quantization step and rounded to the nearest integer.
(3) Entropy coding: lossless entropy coding of quantized DCT coefficients (e.g., Huffman coding).

The quantization steps for different frequencies are stored in quantization matrices (luminance matrix for Y channel or chroma matrix for U and V channels). The quantization matrices can be retrieved from the JPEG image. Here, two points need to be mentioned:

(1) The higher the compression quality is, the smaller the quantization step will be, and vice versa.
(2) The quantization step may be different for different frequencies and different channels.

The decoding of a JPEG image involves the inverse of the pervious three steps taken in reverse order: entropy decoding, de-quantization, and inverse DCT (IDCT). Unlike the other two operations, the quantization step is not invertible as will be discussed in Section 2.2. The entropy encoding and decoding step will be ignored in the following discussion, since it has nothing to do with our method.

Consequently, when an image is doubly JPEG-compressed, it will undergo the following steps and the DCT coefficients will change accordingly:

(1) The first compression:
    (a) DCT (suppose after this step a coefficient value u is obtained).
    (b) The first quantization with a quantization step q_1 (now the coefficient value becomes $Q_{q_1}(u) = [u/q_1]$, where $[x]$ means rounding x to the nearest integer).
(2) The first decompression:
    (a) Dequantization with q_1 (now the coefficient value becomes $Q_{q_1}^{-1}(Q_{q_1}(u)) = t[u/q_1]q_1$.
    (b) IDCT.
(3) The second compression:
    (a) DCT.
    (b) The second quantization with a quantization step q_2 (now the coefficient value u becomes $Q_{q_1q_2}(u) = [[u/q_1]q_1/q_2]$).

We will show in the following section that the histograms of double quantized DCT coefficients have some unique properties that can be utilized for forgery detection.

2.2. DQ effect

The DQ effect has been discussed in [13], but their discussion is based on quantization with the floor function. However, in JPEG compression it is the rounding function, rather than the floor function, that is utilized in the quantization step. So we provide the analysis of DQ effect based on quantization with the rounding function here, which can more accurately explain the DQ effect caused by double JPEG compression.

Denote h_1 and h_2 the histograms of DCT coefficients of a frequency *before* the first quantization and *after* the second quantization, respectively. We will investigate how h_1 changes after DQ. Suppose a DCT coefficient in the u_1-th bin of h_1 is relocated in a bin u_2 in h_2, then

$$Q_{q_1q_2}(u_1) = \left[\left[\frac{u_1}{q_1}\right]\frac{q_1}{q_2}\right] = u_2.$$

Hence,

$$u_2 - \frac{1}{2} \leqslant \left[\frac{u_1}{q_1}\right]\frac{q_1}{q_2} < u_2 + \frac{1}{2}.$$

Therefore,

$$\left\lceil \frac{q_2}{q_1}\left(u_2 - \frac{1}{2}\right)\right\rceil - \frac{1}{2} \leqslant \frac{u_1}{q_1} < \left\lfloor \frac{q_2}{q_1}\left(u_2 + \frac{1}{2}\right)\right\rfloor + \frac{1}{2},$$

where $\lceil x \rceil$ and $\lfloor x \rfloor$ denote the ceiling and floor function, respectively.

If q_1 is even, then

$$q_1\left(\left\lceil \frac{q_2}{q_1}\left(u_2 - \frac{1}{2}\right)\right\rceil - \frac{1}{2}\right) \leqslant u_1 < q_1\left(\left\lfloor \frac{q_2}{q_1}\left(u_2 + \frac{1}{2}\right)\right\rfloor + \frac{1}{2}\right).$$

If q_1 is odd, then

$$q_1\left(\left\lceil \frac{q_2}{q_1}\left(u_2 - \frac{1}{2}\right)\right\rceil - \frac{1}{2}\right) + \frac{1}{2} \leqslant u_1 \leqslant q_1\left(\left\lfloor \frac{q_2}{q_1}\left(u_2 + \frac{1}{2}\right)\right\rfloor + \frac{1}{2}\right) - \frac{1}{2}.$$

In either cases, the number $n(u_2)$ of the original histogram bins contributing to bin u_2 in the double quantized histogram h_2 depends on u_2 and can be expressed as

$$n(u_2) = q_1\left(\left\lfloor \frac{q_2}{q_1}\left(u_2 + \frac{1}{2}\right)\right\rfloor - \left\lceil \frac{q_2}{q_1}\left(u_2 - \frac{1}{2}\right)\right\rceil + 1\right). \tag{1}$$

Note that $n(u_2)$ is a periodic function, with a period:

$$p = q_1/\gcd(q_1, q_2),$$

where $\gcd(q_1, q_2)$ is the greatest common divider of q_1 and q_2. This periodicity is the reason of the periodic pattern in histograms of double quantized signals (Figs. 6(c), (d) and 7).

One notable observation is that if $q_2 < q_1$, then $n(u_2) = 0$ for some u_2. For example, when $q_1 = 5$, $q_2 = 2$, then $n(5k+1) = 0$. This means that the histogram after DQ can have periodically missing values (also refer to Fig. 6(c)). While when $q_2 > q_1$ the histogram can exhibit some periodic pattern of peaks and valleys (Figs. 6(d) and 7). In both cases, it could be viewed as showing peaks and valleys periodically. This is called the DQ effect.

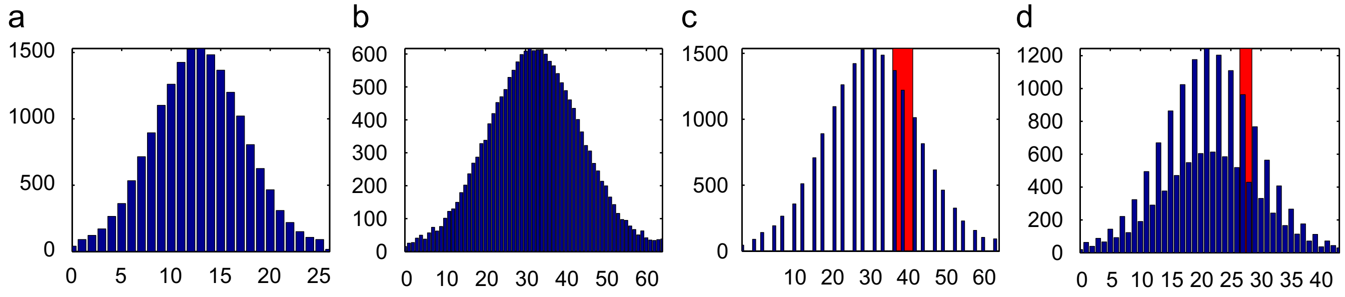

Fig. 6. The left two figures are histograms of single quantized signals with steps 5 (a) and 2 (b), respectively. The right two figures are histograms of double quantized signals with steps 5 followed by 2 (c), and 2 followed by 3 (d). Note the periodic artifacts in the histograms of double quantized signals. The shaded rectangles show one period of the histograms.

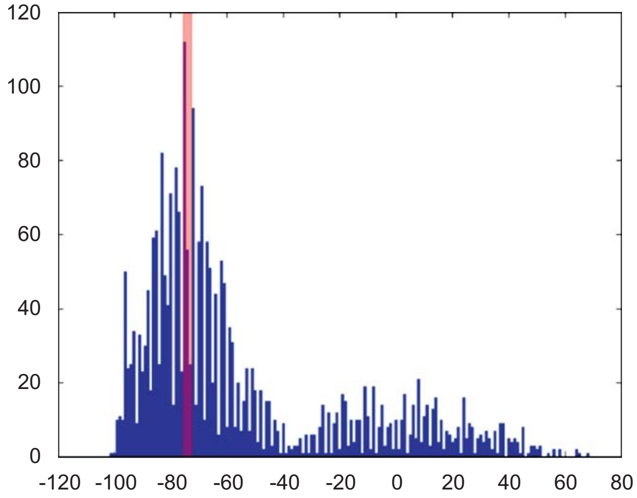

Fig. 7. A typical DCT coefficient histogram of a tampered JPEG image. This histogram can be viewed as the sum of two histograms. One has high peaks and deep valleys and the other has a random distribution. The first "virtual" histogram collects the contribution of unchanged blocks, while the second one collects the contribution of tampered blocks. The shaded rectangle shows one estimated period of the histogram.

2.3. Estimating the period

To utilize the DQ effect, we need to know the period p of a histogram h. It can be estimated as follows. Suppose s_0 is the index of the bin that has the largest value. For each p between 1 and $s_{max}/20$, we compute the following quantity:

$$H(p)=\frac{1}{i_{max}-i_{min}+1}\sum_{i=i_{min}}^{i_{max}}[h(i\cdot p+s_0)]^{\alpha},$$

where $i_{max}=\lfloor(s_{max}-s_0)/p\rfloor$, $i_{min}=\lceil(s_{min}-s_0)/p\rceil$, s_{max} and s_{min} are the maximum and minimum index of the bins in the histogram, respectively, and α is a parameter (can be simply chosen as 1). $H(p)$ evaluates how well the supposed period p gathers the high-valued bins. Then one estimate of the period p is: $p_{hist}=\arg\max_p H(p)$. On the other hand, we can use the fast Fourier transform to find the peak of the spectrum of the histogram with the direct current component removed. This gives another estimate p_{FFT} of the period p. Then the final estimate of the period is $p=\min(p_{hist},p_{FFT})$. If $p=1$, this histogram suggests that the JPEG image is singly compressed. Therefore, it cannot tell whether a block is tampered or not and we should not use this histogram.

3. Core of our algorithm

3.1. DQ effect analysis in tampered JPEG images

Although DQ effect has been presented in [13,27], their authors actually did not develop a working algorithm for real-world tampered image detection. They only suggested that images having DQ effect might be forged. However, since people may simply compress a real image twice with different qualities, the presence of DQ effect does not necessary imply the existence of image forgery. Moreover, what is even more critical is that our observation is exactly the other way round: the unchanged region will have DQ effect, while the tampered region will not. Unchanged regions produce DQ effect because this part of the tampered image is the same as that of the doubly compressed original image. On the other hand, the tampered region will not exhibit DQ effects because of following reasons:

(1) *Absence of the first JPEG compression in the tampered region.* Suppose the tampered region is cut from a BMP image or other kind of images rather than JPEG images, then, the tampered region will not undergo the first JPEG compression, and of course does not produce any DQ effect. Similarly, when the tampered region is synthesized by alpha matting or inpainting, or other similar skills, then the tampered region will not produce any DQ effect either.

(2) *Mismatch of the DCT grid of the tampered region with that of the unchanged region.* Suppose the tampered region is cut from a JPEG image, or even from the original JPEG image itself, the tampered region is still of little possibility to show the DQ effect. Recall the description in Section 2.1, one assumption to assure the existence of DQ effect is that the DCT in the second compression should be just the inverse operation of IDCT in the first decompression. But if there is mismatch of the DCT grids, then the assumption is violated. For example, if the first block of a JPEG image, i.e., the block from pixel (0,0) to (7,7), is pasted to another position of the same image, say, to the position from pixel (18,18) to (25,25), then in the second compression step, the tampered region will be divided into four sub-blocks: (18,18)–(23,23), (24,18)–(25,23), (18,24)–(23,25), and (24,24)–(25,25). None of these sub-blocks can recover the DCT coefficients of the original block.

(3) *Composition of DCT blocks along the boundary of the tampered region.* There is little possibility that the tampered region exactly consists of 8 × 8 blocks, so blocks along the boundary of the tampered region will consist of pixels in the tampered region and also pixels in the unchanged region. These blocks do not follow the rules of DQ effect either. Moreover, some post-processing, such as smoothing or alpha matting, along the boundary of the tampered region can also render those blocks to break the rules of DQ effect.

In summary, when the tampered region is synthesized or edited, there may be some reasons, which may not be limited to those listed above, that cause the absence of DQ effect in the tampered region. Therefore, the histogram of the whole tampered JPEG image could be regarded as the superposition of two histograms: one has periodical peaks and valleys, and the other has random bin values. They are contributed by the unchanged region and the tampered region, respectively. Fig. 7 shows a typical histogram of a tampered JPEG image.

However, given a test image, we do not know *a priori* the tampered region and the unchanged region. We only have total histograms like Fig. 7 that are contributed by both tampered and unchanged blocks, rather than separated histograms. So we have to infer the possibility of a block being tampered or not.

3.2. Bayesian approach to detecting tampered blocks

From the above analysis, we see that tampered blocks and unchanged blocks have different bias in terms of contributing to the bins of a histogram h: an unchanged one favors the high peaks of h, while a tampered one tends to contribute randomly to the bins of h. Our inference is based on this key observation.

Suppose a period starts from the s_0-bin and ends at the (s_0+p-1)-th bin, then the possibility of an unchanged block which contributes to that period occurring in the (s_0+i)-bin can be estimated as

$$P_u(s_0+i) = h(s_0+i) \Bigg/ \sum_{k=0}^{p-1} h(s_0+k) \tag{2}$$

because it indeed gives high values at high peaks. Here, $h(k)$ denotes the value of the k-th bin of the DCT coefficient histogram h. On the other hand, the possibility of a tampered block which contributes to that period appearing in the bin (s_0+i) can be estimated as

$$P_t(s_0+i) = 1/p \tag{3}$$

due to its randomness of contribution. From the naive Bayesian approach, if a block contributes to the (s_0+i)-th bin, then the posterior probability of it being a tampered block or an unchanged block is, respectively,

$$P(\text{tampered}|s_0+i) = P_t/(P_t+P_u) \quad \text{and} \tag{4}$$

$$P(\text{unchanged}|s_0+i) = P_u/(P_t+P_u). \tag{5}$$

Note that we need to know the period p in advance in order to compute P_u or P_t. The method has been described in Section 2.3, and we should use those histograms whose periods are greater than 1. Then each period of *each* of such histograms assigns a probability to every block that contributes to the bins in that period (Fig. 4), using Eq. (5). Consequently, we obtain a BPPM of blocks of the image under examination (Fig. 1(b)). Each pixel of the BPPM stands for a DCT block of the image and its value is the accumulated posterior probabilities of the DCT block.

3.3. Feature extraction

If the image is tampered, we expect that tampered blocks cluster, i.e., the BPPM should be segmented into a small number of regions, where each region has a high probability of being either unchanged or tampered (Fig. 1(c)). While any image segmentation algorithm can be applied to the BPPM, to save computation time, we simply threshold the BPPM by choosing a threshold:

$$T_{opt} = \arg\max_T (\sigma/(\sigma_0+\sigma_1)), \tag{6}$$

where given a T, the pixels of the BPPM are classified into to classes C_0 and C_1, respectively. σ_0 and σ_1 are the variances of the probabilities in each class, respectively, and σ is the squared difference between the mean probabilities of the classes.

With the optimal threshold, we expect that those pixels in class C_0 (i.e., those having probabilities below T_{opt}) correspond to the tampered blocks in the image. However, this is still insufficient for confident decision because any BPPM can be segmented in the above manner as long as its variance is nonzero. So we have to elaborate more. Based on the segmentation, we can extract four features: T_{opt}, σ, $\sigma_0+\sigma_1$, and the connectivity K_0 of C_0. We need the connectivity of C_0 as a feature because we expect that the tampered blocks cluster if they exist. Again, there are many methods to define the connectivity K_0. Considering the computation load, our choice is as follows. First, the BPPM is denoised by using a medium filter. Then, for each pixel i in C_0, find the number e_i of pixels in class C_1 in its four-neighborhood. Finally, we compute $K_0 = \sum_i \max(e_i-2,0)/|C_0|$, where $|C_0|$ is the number of pixels in C_0. This definition is inspired by the perimeter–area ratio for shape description. As we can see, the more connected C_0 is, the smaller K_0 is. Note that we use $\max(e_i-2,0)$ instead of e_i directly, to allow narrowly shaped C_0. If e_i is used directly, round-shaped C_0 will be preferred.

With the four-dimensional feature vector $(T_{opt}, \sigma, \sigma_0+\sigma_1, K_0)$, we can proceed to decide whether the image is tampered, by feeding the feature vector into a trained SVM. If the output is positive, then the DCT blocks that correspond to C_0 of the BPPM are decided as the tampered region of the image (Fig. 1(d)).

4. Experiments

We first ask some students, who are unaware of our detection method, to build a database for us. As we know well under what situations our method works (see next section), the bias in the database can be reduced at the best without our involvement. The students randomly choose 50 images (in JPEG or not), shrink and rotate them slightly, crop the first two rows and columns and save them in BMP. In this way, we believe that any trace of JPEG in the original images is removed and clean "raw" images are obtained. Then, for each image the students perform JPEG compression with quality $Q_1 = 50, 55, 60, \ldots, 95$, respectively, resulting in a set of derived images. Next, for each set of derived images, the students make a copy of them and alter their content in the same region using either lazy snapping [7], Poisson matting [5], image completion [6], or image inpainting (which is a part of the image completion tool), depending on their content.[5] If the content is from other images, it is applied identically to all the derived images. Finally, the students perform JPEG compression again on each obtained image (derived authentic images, and those further tampered) with quality $Q_2 = 50, 55, 60, \ldots, 95$, respectively. So we have a database of 10 000 images, half of which are the authentic images with compression qualities Q_1 and Q_2,[6] each varying from 50 to 95, while the other half are their tampered counterparts. In this way, we simulate all the compression qualities at a step size of 5.[7]

Next, we randomly choose 20 sets of derived images and their forgery counterparts to train an SVM. Then we apply the SVM to the rest images in the database, or other test images donated by our colleagues which we are sure about their sources and their synthesis process, after extracting the features of those images. Note that the donated images are created before the invention of our detection

[5] As our goal is to test our algorithm, the visual quality of the synthesized images is not as good as those donated by our colleagues, most of which aim at being published in Siggraph.

[6] Note that these images will be judged as forgery if checking the DQ effect naively as in [13] because obviously they have the DQ effect.

[7] Experimentation with even lower compression qualities is possible but is deemed unnecessary. Moreover, it requires more computation.

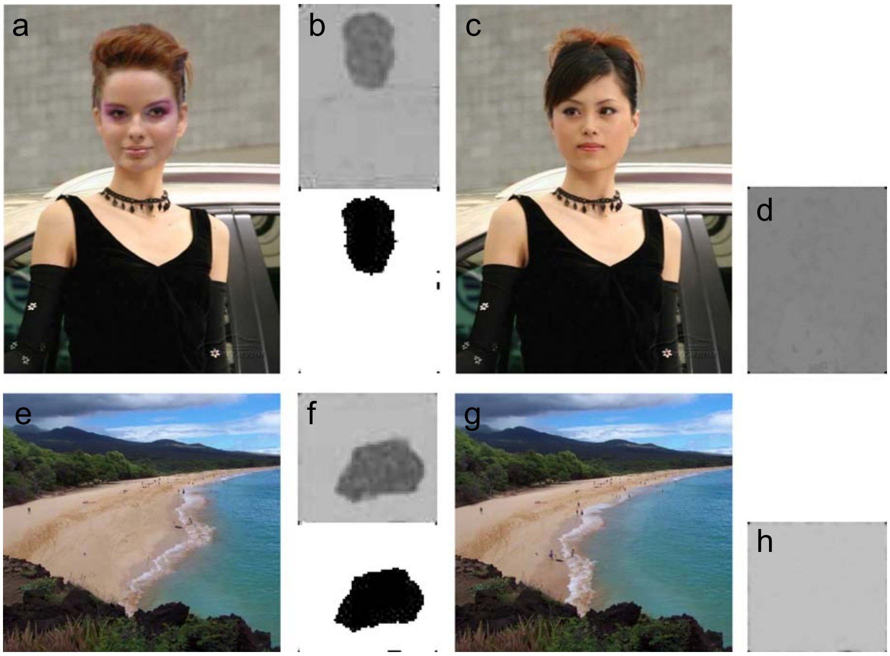

Fig. 8. Some detection results of our algorithm. The images are donated by our colleagues. Image (a) is tampered by cut and paste followed by some touch-ups. Image (e) is tampered by texture synthesis. The left columns are the tampered images. The third column are the original images. The BPPMs and the masks of tampered regions are shown in the middle column. For comparison, the BPPMs of the original images are also shown on the right-most column. Visual examination may fail for these images.

method. So they are not deemed to introduce bias to the database either.

Figs. 8 and 9 show some examples of successful detection on the donated images. Given the tampered images shown in the first column, human inspection may fail. However, our algorithm can detect the tampered regions almost correctly. In comparison, the BPPMs of the original images are almost uniform. It is worth noting that it is hard to guess the suspicious regions in images tampered by inpainting (Fig. 9(a) and (e)).

Our algorithm is fast. Analyzing an image of a size 500 × 500 only requires about 4 s on our Pentium 1.9 GHz PC, with unoptimized Matlab codes.

To evaluate the accuracy of region level detection, we define the region level detection rate as the proportion of DCT blocks that are correctly classified.[8] Fig. 10(a)–(j) show the image level and the region level detection rates of our algorithm on the rest images in our database, under different Q_1 and Q_2. Although the curves look somewhat random, most of the curves have a peak between $Q_2 = 80$ and 90. As the DQ effect breaks down when $Q_1 = Q_2$, the image level detection becomes random guess at $Q_2 = Q1$. The average detection rates (averaged on Q_2) are about 60% (Fig. 10(k)).

For comparison, we expand the algorithm presented in [25] slightly so that it can output image level detection rates. First, for each image 10 sets of CRFs are computed from 10 random selection of the set of image patches. Next, the SVM therein decides the normality of each set of CRFs and the variance among the sets of *normal* CRFs is computed as

$$var = var_R + var_G + var_B, \quad \text{where}$$

$$var_i = \sum_k \int_0^1 (r_i^{(k)}(x) - \bar{r}_i(x))^2 \, dx, \quad i = R, G, B,$$

in which $r_i^{(k)}(x)$ is the inverse response function of channel i computed from the k-th choice of image patch set and $\bar{r}_i(x)$ is the average of $r_i^{(k)}(x)$. Finally, the image is decided to be forgery if at least two sets of the CRFs are abnormal, or the variance among the normal CRFs is larger than 0.115.[9] The corresponding image level detection rates are shown in Fig. 10(j). As this method is too time consuming (computing the CRFs requires nonlinear optimization), we were unable to test its performance on all Q_1's and Q_2's.

Another comparison is with blind gamma estimation [13], which is possibly effective only when the forgery region is across the whole image horizontally or vertically. Fig. 11 (a) and (b) show the estimated gammas for each column of Fig. 9(i) and (k), respectively. Our algorithm only took 4.1 s to analyze Fig. 9(i) or (k) and gave the correct results, while the blind gamma estimation algorithm [13] took 610 s on each image and the gammas (Fig. 11(b)) computed from the authentic image Fig. 9(k) seem abnormal.

5. Discussions and future work

In this paper, we have proposed an algorithm for tampered JPEG image detection by analyzing the DQ effects hidden among the histograms of the DCT coefficients. The four advantages of our algorithm, namely automatic tampered region determination, resistent to different kinds of forgery techniques in the tampered region, ability to work without full decompression and fast detection speed, make our algorithm very attractive.

However, although our proposed method produces encouraging results, more effort is still needed to improve the accuracy of our approach. For example, some tampered images may not be detected and the detected tampered regions may not be 100% correct either (see Figs. 1, 8, and 9). This may be improved by defining (2) more

[8] The ground truth is quantized to the DCT blocks beforehand by the percentage of pixels it occupies a DCT block. When the sizes of an image cannot be divided by 8, the incomplete DCT blocks are not taken into account.

[9] This value is found by checking the CRFs computed from authentic images.

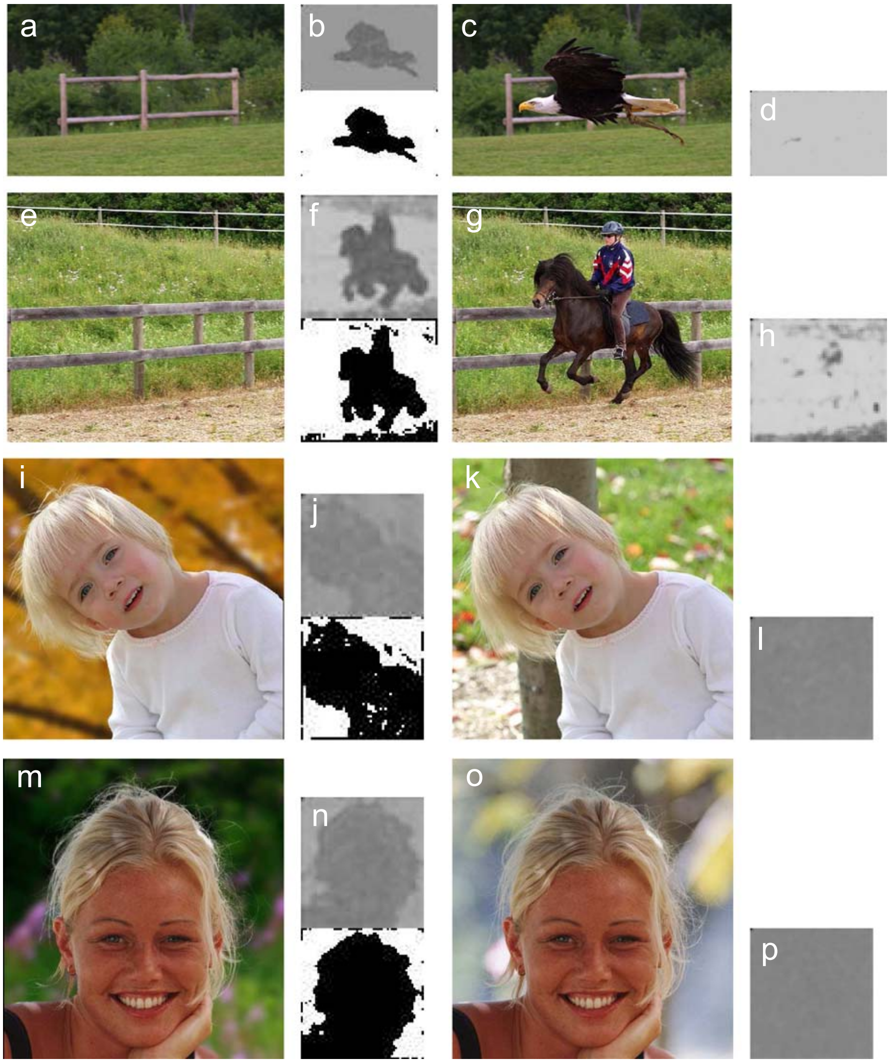

Fig. 9. More detection results of our algorithm on donated images. Images (a) and (e) are tampered by inpainting. Images (i) and (m) are tampered by matting. It is hard to guess that some objects were removed from images (a) and (e).

accurately by

$$P_u(s_0+i)=n(s_0+i)\Bigg/\sum_{k=0}^{p-1}n(s_0+k)\,.$$

But we need to know q_1 and q_2 in order to compute $n(k)$ according to (1). Actually q_2 can be dumped from the JPEG image. Unfortunately, q_1 is lost after the first decompression and hence has to be estimated. Although Lukas and Fridrich [27] have proposed an algorithm to estimate the first quantization matrix, the algorithm is too restrictive and may not be reliable. Hence we are exploring a simple yet practical method to estimate q_1. A possible way is by machine learning techniques: we first collect a large number of training images with known q_1 and q_2; given a DCT coefficient histogram of a JPEG image, whose second quantization step is q_2, if we could define appropriate features of histograms, many regression or multi-class classification methods can be employed to infer the most probable q_1.

It is possible to adapt our algorithm to handle JPEG2000 images. In JPEG2000, an image is first partitioned into tiles, whose sizes are lager than those of DCT blocks. For each tile, wavelet transform is applied and then the wavelet coefficients of each subband are quantized with the same quantization step. So the basic principles supporting tampered JPEG image detection also apply to JPEG2000 images, but the differences in the tile/block sizes, the subband/frequency numbers and the characteristics of wavelet transform/DCT may cause the difference in the effectiveness of DQ.

Finally, it is already recognized that, like watermarking, passive image forensic techniques can easily have counter measures [10,13] if the detection algorithm is known to the public. For example, resampling test [13] fails when the image is further resampled after synthesis. The SNR test [13] fails if the same noise is added across the whole synthesized image. The blind gamma estimation [13] and CRF computation [25] do not work if the forger synthesizes in the irradiance domain by converting the graylevel into irradiance using the CRF [25] estimated in the component images, and then applying a consistent CRF to convert the irradiance back into graylevel. And the CFA checking [14] fails if the synthesized image is downsampled into a Bayer pattern and then demosaicked again. As for our method, it is not surprising that there are cases under which our method does not work well:

(1) The original image which contributes to the unchanged region is not a JPEG image. In this case, there will not be the DQ effect in the unchanged region.

Please cite this article as: Z. Lin, et al., Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis, Pattern Recognition (2009), doi: 10.1016/j.patcog.2009.03.019

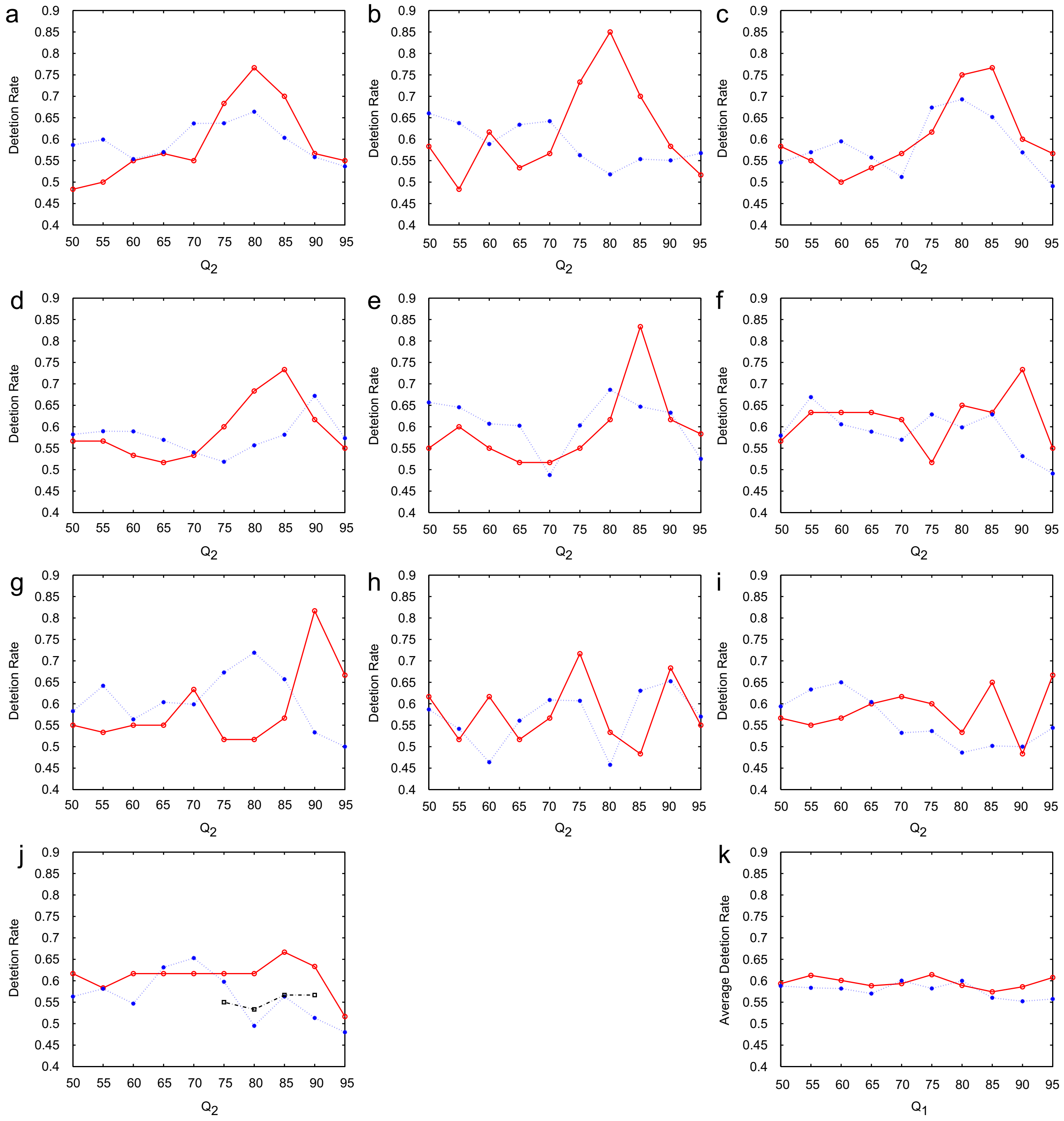

Fig. 10. The image level (solid lines) and region level (dotted lines) detection rates at different compression qualities Q_1 and Q_2. The dash-dot line in (j) shows the image level detection rate of the method by Lin et al. [25]. (k) is the average detection rate for different Q_1's. (a) $Q_1 = 50$, (b) $Q_1 = 55$, (c) $Q_1 = 60$, (d) $Q_1 = 65$, (e) $Q_1 = 70$, (f) $Q_1 = 75$, (g) $Q_1 = 80$, (h) $Q_1 = 85$, (i) $Q_1 = 90$, (j) $Q_1 = 95$.

(2) The whole image is resized, rotated, or cropped such that the DCT grid changes.

Due to the above reasons, we could not test our method on images downloaded from the web because we are not sure of: (1) whether the original images were in JPEG; and (2) whether the images had undergone any postprocessing. That is why we have to test on an image database built by our own (Section 4). We are working on more robust methods that can still detect the trace of DQ effect when unfavorable postprocessing happens after image forgery. Nonetheless, the current paper contributes to tampered image detection by providing a fast, fully automatic, and fine-grained algorithm and presenting encouraging results.

Due to the easy loss of low-level cues for passive image forensic techniques, Popescu and Farid had to conclude at the end of [13] that developing image forensic techniques will increase the difficulties in creating convincing image forgeries, rather than solving the problem completely. So we expect that in the battle between image forgery

Please cite this article as: Z. Lin, et al., Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis, Pattern Recognition (2009), doi: 10.1016/j.patcog.2009.03.019

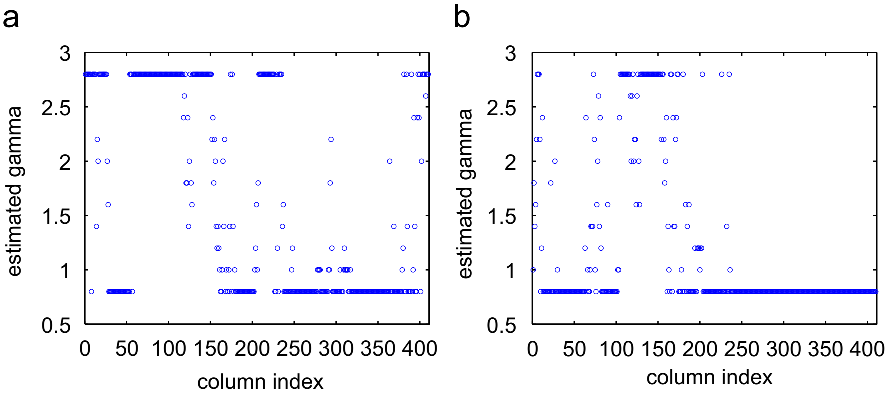

Fig. 11. The estimated column-wise gammas using the blind gamma estimation algorithm in [13]. (a) and (b) correspond to Fig. 9(i) and (k), respectively. The horizontal axis is the column index and the vertical axis is the gamma value. The gamma is searched from 0.8 to 2.8 with a step size 0.2. By the methodology in [13], Fig. 9(k) is more likely to be classified as tampered than Fig. 9(i) is because the gamma distribution in (b) is more abnormal than that in (a).

and forgery detection, the techniques of both sides will improve alternately.

Acknowledgments

The authors would like to thank Dr. Yin Li, Dr. Jian Sun, and Dr. Lu Yuan for sharing us test images, Mr. Lincan Zou for collecting the training samples, and Dr. Yuwen He and Dr. Debing Liu for providing us the code to dump the DCT coefficients and the quantization matrices in the JPEG images.

References

[1] M. Bertalmio, G. Sapiro, V. Caselles, C. Ballester, Image inpainting, in: ACM Siggraph, 2000, pp. 417–424.
[2] V. Kwatra, et al., Graphcut textures: image and video synthesis using graph cuts, in: ACM Siggraph, 2003, pp. 277–286.
[3] Y.-Y. Chuang, et al., Video matting of complex scenes, in: ACM Siggraph, 2002, pp. 243–248.
[4] P. Pérez, M. Gangnet, A. Blake, Poisson image editing, in: ACM Siggraph, 2003, pp. 313–318.
[5] J. Sun, et al., Poisson matting, in: ACM Siggraph, 2004, pp. 315–321.
[6] J. Sun, L. Yuan, J. Jia, H.-Y. Shum, Image completion with structure propagation, in: ACM Siggraph, 2005, pp. 861–868.
[7] Y. Li, et al., Lazy snapping, in: ACM Siggraph, 2004, pp. 303–308.
[8] S.-J. Lee, S.-H. Jung, A survey of watermarking techniques applied to multimedia, in: Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE2001), vol. 1, pp. 272–277.
[9] S. Craver, et al., Reading between the lines: lessons from the SDMI challenge, in: Proceedings of the 10th USENIX Security Symposium, 2001.
[10] T.-T. Ng, S.F. Chang, C.-Y. Lin, Q. Sun, Passive-blind image forensics, in: W. Zeng, H. Yu, C.-Y. Lin (Eds.), Multimedia Security Technologies for Digital Rights, Elsevier, Amsterdam, 2006.
[11] C. Amsberry, Alterations of photos raise host of legal, ethical issues, The Wall Street Journal, January 26, 1989, Section B, p. 1.
[12] M. Johnson, H. Farid, Exposing digital forgeries by detecting inconsistencies in lighting, in: ACM Multimedia and Security Workshop, New York, NY, 2005.
[13] A.C. Popescu, H. Farid, Statistical tools for digital forensics, in: Proceedings of the 6th International Workshop on Information Hiding, Toronto, Canada, 2004.
[14] A.C. Popescu, H. Farid, Exposing digital forgeries in color filter array interpolated images, IEEE Transactions on Signal Processing 53 (10) (2005) 3948–3959.
[15] A. Popescu, H. Farid, Exposing digital forgeries by detecting traces of re-sampling, IEEE Transactions on Signal Processing 52 (2) (2005) 758–767.
[16] J. Lukas, J. Fridrich, M. Goljan, Detecting digital image forgeries using sensor pattern noise, SPIE Electronic Imaging, Photonics West, January 2006.
[17] H. Farid, Blind inverse gamma correction, IEEE Transactions on Image Processing 10 (10) (2001) 1428–1433.
[18] A.C. Popescu, H. Farid, Exposing digital forgeries by detecting duplicated image regions, Technical Report, TR2004-515, Dartmouth College, Computer Science, 2004.
[19] M.K. Johnson, H. Farid, Exposing digital forgeries through chromatic aberration, in: ACM Multimedia and Security Workshop, Geneva, Switzerland, 2006.
[20] T.-T. Ng, S.-F. Chang, Q. Sun, Blind detection of photomontage using higher order statistics, in: IEEE International Symposium on Circuits and Systems (ISCAS 2004), Vancouver, Canada, pp. 688–691.
[21] Y.-F. Hsu, S.-F. Chang, Detecting image splicing using geometry invariants and camera characteristics consistency, in: International Conference on Multimedia and Expo (ICME 2006), Toronto, Canada.
[22] Y.-F. Hsu, S.-F. Chang, Image splicing detection using camera response function consistency and automatic segmentation, in: International Conference on Multimedia and Expo (ICME 2007), Beijing, China.
[23] S. Ye, Q. Sun, E.-C. Chang, Detecting digital image forgeries by measuring inconsistencies of blocking artifact, Int'l Conf. Multimedia and Expo (ICME 2007), Beijing, China.
[24] T.V. Lanh, K.-S. Chong, S. Emmanuel, M. Kankanhalli, A survey on image forensic methods, in: International Conference on Multimedia and Expo (ICME 2007), Beijing, China.
[25] Z. Lin, R. Wang, X. Tang, H.-Y. Shum, Detecting doctored images using camera response normality and consistency, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 1087–1092.
[26] J. He, Z. Lin, L. Wang, X. Tang, Detecting doctored JPEG images via DCT coefficient analysis, in: Proceedings of the 9th European Conference on Computer Vision, Lecture Notes in Computer Sciences, vol. 3953, Springer, Berlin, 2006, pp. 423–435.
[27] J. Lukas, J. Fridrich, Estimation of primary quantization matrix in double compressed JPEG images, in: Proceedings of the Digital Forensic Research Workshop, 2003.
[28] A.M. Tekalp, Digital Video Processing, Prentice-Hall, Englewood Cliffs, NJ, 1995.

About the Author—ZHOUCHEN LIN received the Ph.D. degree in Applied Mathematics from Peking University in 2000. He is currently a Researcher in Visual Computing Group, Microsoft Research, Asia. His research interests include computer vision, computer graphics, pattern recognition, statistical learning, document processing, and human computer interaction. He is a Senior Member of the IEEE.

About the Author—JUNFENG HE received the Master degree from Tsinghua University in 2005. His research interests include pattern recognition and statistical learning.

About the Author—XIAOOU TANG received the Ph.D. degree from the Massachusetts Institute of Technology, Cambridge in 1996. He is a Professor and the Director of Multimedia Lab in the Department of Information Engineering, the Chinese University of Hong Kong. He is also the Group manager of the Visual Computing Group at the Microsoft Research Asia. He is an Associate Editor of IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). His research interests include computer vision, pattern recognition, and video processing.

About the Author—CHI-KEUNG TANG received his Ph.D. from the University of Southern California in 2000. He is an Associate Professor in the Department of Computer Sciences, the Hong Kong University of Science and Technology. His research interests include computer vision and computer graphics.