A Robust Image Authentication Method Distinguishing JPEG Compression from Malicious Manipulation

Ching-Yung Lin and Shih-Fu Chang

Department of Electrical Engineering &
New Media Technology Center
Columbia University
New York, New York 10027

ABSTRACT

Image authentication verifies the originality of an image by detecting malicious manipulations. This goal is different from that of image watermarking which embeds into the image a signature surviving most manipulations. Existing methods for image authentication treat all types of manipulation equally (*i.e.*, as unacceptable). However, some applications demand techniques that can distinguish acceptable manipulations (*e.g.*, compression) from malicious ones. In this paper, we describe an effective technique for image authentication which can prevent malicious manipulations but allow JPEG lossy compression. The authentication signature is based on the invariance of the relationship between DCT coefficients at the same position in separate blocks of an image. This relationship will be preserved when these coefficients are quantized in a JPEG compression process. Our proposed method can distinguish malicious manipulations from JPEG lossy compression regardless of how high the compression ratio is. We also show that, in different practical cases, the design of the authenticator depends on the number of recompression times, and whether the image is decoded into integral values in the pixel domain during the recompression process. Theoretical and experimental results indicate that this technique is effective for image authentication.

Keywords: image authentication, JPEG lossy compression, manipulation, digital signature

1 Introduction

The well-known adage that "the photograph doesn't lie" is no longer true due to the availability of powerful image manipulation software. Digital images have been adopted because of their ease of manipulation, processing, and storage. It is almost impossible to distinguish subjectively which images are original, and which have been manipulated. This technical development has decreased the credibility that photography used to achieve.

Image authentication techniques protect images from malicious manipulation at every stage of transmission and storage. Reliable image authentication technology must be able to protect an image from the time it was first produced until the final stage of use.

If we consider a digital image to be merely an ordinary bitstream on which no modification is allowed, then there is not much difference between this problem and other data cryptography problems. Two methods have been suggested for achieving the authenticity of digital images: having a digital camera sign the image using a digital signature[5], or embedding a secret code in the image[17]. The first method uses an encrypted digital "signature" which is generated in the capturing devices. A digital signature is based on the method of Public Key Encryption[4][13]. A private key is used to encrypt a hashed version of the image. This encrypted message is called the "signature" of the image, and it provides a way to ensure that this signature cannot be forged. This signature then travels with the image. The authentication process of this image needs an associated public key to decrypt the signature. The image received for authentication is hashed and compared to the codes of the signature. If they match, then the received image is authenticated. The second method embeds a "watermark" in an image[9][17][18]. The fragile watermark usually will be destroyed after manipulation. Authenticity is determined by examining the watermark extracted from the received image.

Both the above methods have clear drawbacks. In their propositions, authenticity will not be preserved unless every pixel of the images is unchanged. However, since lossy compression such as

JPEG is often acceptable - or even desired - in practical applications, an authentication method needs to be able to distinguish lossy compression from malicious manipulations.

Manipulations on images can be considered in two ways: *method* and *purpose*. Manipulation methods include:

- Compression: including lossless and lossy compression, such as JPEG, EZW, *etc.*
- Format Transformation: change of file format, such as changing from GIF to TIFF, or from RGB color space to YUV.
- Shifting: pixels in an image are translated vertically and/or horizontally.
- Scaling: image size is changed. The image is either magnified or reduced.
- Cropping: some part of an image is extracted.
- Quantization: pixel values are quantized. For instance, 256 gray levels are quantized to 64 levels.
- Filtering: some filters such as low pass filter, edge enhancement, median filter, etc. may be used for improving the image quality.
- Replacement: intentional change of some pixel values. The replaced values may be generated manually, by the computer, or from nearby pixels.

The purpose of manipulations may be *representation transformations* and *attacks*. The former are usually acceptable, and the latter, unacceptable. There are two kinds of representation transformations:

1. Format transformation and Lossless Compression. Disregarding the noise caused by the precision limitation during computation, pixel values in an image are not changed after these manipulations. Therefore, when we talk about the word "manipulation" in the following sections, these manipulations are usually excluded.
2. Application-specific transformations. Some applications may require the lossy compression in order to satisfy the resource constraints on bandwidth or storage. Some applications may also need to enhance the image quality, crop the image, change the size, or perform some other

operations. A common aspect of these manipulations is that they change the pixel values, which results in different levels of visual distortion on the image. Among all the application-specific transformations, the lossy compression is one of the operations that tries to minimize the visual distortion.

Attacks, or malicious manipulations, change the image to a new one which carries a different visual meaning to the image observer. One typical example is replacing some parts of the image with different content.

It is difficult for an authenticator to know the purpose of manipulation. A practical approach is to design an authenticator based on the manipulation method. In this paper, we design an authenticator which accepts format transformation, loseless compression, and the popular JPEG lossy compression. The authenticator rejects replacement manipulations because they are frequently used for attacks. Our authenticator does not aim to reject or accept, in absolute terms, other manipulation methods because the problem of whether they are acceptable depends on their applications. But, if necessary, some manipulations can be clearly specified by users, such as shifting, cropping, or constant intensity enhancement. We will discuss this more rigorously later.

For an image, there are some invariance properties which can be preserved during JPEG lossy compression. Let us consider the relationship between two DCT coefficients of the same position in two separate 8x8 blocks of an image. This relationship will hold even if these coefficients are quantized by an arbitrary quantization table in a JPEG compression process. In this paper, we will propose a robust authentication method which can distinguish malicious manipulations from JPEG lossy compression. Our technique can prevent malicious attacks that aim to change the image's meaning. We briefly review the JPEG system in Section 2. In Section 3, a general system for authentication will be proposed. Also, we will describe how to control parameters for different practical uses. A simple example is shown in this section. We will present rigorous performance analysis in Section 4. Experimental results will be shown in Section 5. In Section 6, we will present conclusions and discuss future work.

2 Review of JPEG Lossy Compression

In this section, we briefly review the JPEG lossy compression standard. Figure 1 shows the key steps of the JPEG Baseline (Lossy) Compression. At the input to the JPEG[16] encoder, the source image, X, is grouped into $\wp$ nonoverlapping 8×8 blocks, $X = \bigcup_{p=1}^{\wp} \mathbf{X_p}$. Each block is sent sequentially to the Discrete Cosine Transform (DCT). Instead of representing each block, $\mathbf{X_p}$, as a 8×8 matrix, we can rewrite it as a 64×1 vector following the "zig-zag" order[16]. Therefore, the DCT coefficients, $\mathbf{F_p}$, of the vector, $\mathbf{X_p}$, can be considered as a linear transformation of $\mathbf{X_p}$ with a 64×64 transformation matrix $\mathbf{D}$, *s.t.*,

$$\mathbf{F_p} = \mathbf{D}\mathbf{X_p}. \tag{1}$$

Each of the 64 DCT coefficients is uniformly quantized with a 64-element quantization table $\mathbf{Q}$. In JPEG, this table is used on all blocks of an image. (For color images, there could be three quantization tables for YUV domains, respectively.) Quantization is defined as the division of each DCT coefficient by its corresponding quantizer step size, and rounding to the nearest integer:

$$\tilde{\mathbf{f}}_\mathbf{p}(\nu) \equiv \ \mathit{Integer\ Round}(\frac{\mathbf{F_p}(\nu)}{\mathbf{Q}(\nu)}), \tag{2}$$

where $\nu = 1...64$. In eq.(2), $\tilde{\mathbf{f}}_\mathbf{p}$ is the output of the quantizer. For the convenience of later discussion, we can define $\tilde{\mathbf{F}}_\mathbf{p}$, a quantized approximation of $\mathbf{F_p}$, as

$$\tilde{\mathbf{F}}_\mathbf{p}(\nu) \equiv \tilde{\mathbf{f}}_\mathbf{p}(\nu) \cdot \mathbf{Q}(\nu). \tag{3}$$

After quantization, the inter-block differences of DC coefficients are encoded. The AC terms are ordered following the "zig-zag" order. Both DC and AC coefficients are then entropy encoded. The final JPEG file, $\tilde{B}$, includes the Huffman Table, the Quantization Table, the encoded data and some other information.

At the decoder, first the above two tables have to be reconstructed. Then the bitstream is sent to the Entropy Decoder and, then, the Dequantizer. The output of dequantizer, $\tilde{\mathbf{F}}_\mathbf{p}$, is the same as that defined in eq.(3). We then use Inverse DCT (IDCT) to convert $\tilde{\mathbf{F}}_\mathbf{p}$ to the spatial-domain image block $\tilde{\mathbf{X}}_\mathbf{p}$.

$$\tilde{\mathbf{X}}_\mathbf{p} = \mathbf{D}^{-1}\tilde{\mathbf{F}}_\mathbf{p}. \tag{4}$$

All blocks are then tiled to form a decoded image frame.

Theoretically, the results of IDCT will be real numbers. However, the brightness of an image is usually represented by an 8-bit integer from 0 to 255 and thus a rounding process mapping those real numbers to integers may be necessary. We have found that popular JPEG softwares such as PhotoShop, xv, *etc.* use the integer rounding functions in several steps of their DCT and IDCT operators in order to save computation or memory. In other words, the input and output of their DCT and IDCT operators are all integers. This approximation may not introduce too much visual distortion but may affect the authentication system performance that we will discuss in more detail in Section 4.

3 Authentication System

The proposed authentication method is shown in Figure 2. Our method uses a concept similar to that of the digital signature method proposed by Friedman[5], but their technique doesn't survive lossy compression. A signature and an image are generated at the same time. The signature is an encrypted form of the feature codes or hashes of this image, and it is stored as a file. When a user needs to authenticate the image he receives, he should decrypt this signature and compare the feature codes (or hashed values) of this image to their corresponding values in the original signature. If they match, this image can be claimed to be "authenticated." The most important difference between our method and Friedman's "trustworthy camera" is that we use invariance properties in JPEG lossy compression as robust feature codes instead of using hashes of the raw images.

3.1 Invariants of an image before and after JPEG compression

The generation of a signature can be divided into two parts: feature extraction and feature encryption. Feature extraction is the focus of this paper. From the compression process of JPEG, we have found that some quantitative invariants or predictable properties can be extracted.

Two steps in the JPEG compression process reduce the required bits representing an image: 1.) *quantization and rounding of the DCT coefficients*, and 2.) *entropy coding*. The second step is a lossless operation. The first step is a lossy operation. This operation alternates pixel values but keeps the important visual characteristics of the image. It is the source of image quality

degradation. Therefore, if robust feature codes are expected for authentication, they must survive this step. The following theorems provide a technical basis for generating such robust feature codes.

Theorem 1 *Assume* $\mathbf{F_p}$ *and* $\mathbf{F_q}$ *are DCT coefficient vectors of two arbitrarily* 8×8 *nonoverlapping blocks of image X, and* $\mathbf{Q}$ *is the quantization table of JPEG lossy compression.* $\forall\nu \in [1,..,64]$ *and* $p,q \in [1,..,\wp]$, *where* $\wp$ *is the total number of blocks. Define* $\mathbf{\Delta F_{p,q}} \equiv \mathbf{F_p} - \mathbf{F_q}$ *and* $\mathbf{\Delta \tilde{F}_{p,q}} \equiv \mathbf{\tilde{F}_p} - \mathbf{\tilde{F}_q}$ *where* $\mathbf{\tilde{F}_p}$ *is defined as* $\mathbf{\tilde{F}_p}(\nu) \equiv$ *Integer Round*$(\frac{\mathbf{F_p}(\nu)}{\mathbf{Q}(\nu)}) \cdot \mathbf{Q}(\nu)$. *Then, the following properties must be true:*

- *if* $\mathbf{\Delta F_{p,q}}(\nu) > 0$, *then* $\mathbf{\Delta \tilde{F}_{p,q}}(\nu) \geq 0$,
- *else if* $\mathbf{\Delta F_{p,q}}(\nu) < 0$, *then* $\mathbf{\Delta \tilde{F}_{p,q}}(\nu) \leq 0$,
- *else* $\mathbf{\Delta F_{p,q}}(\nu) = 0$, *then* $\mathbf{\Delta \tilde{F}_{p,q}}(\nu) = 0$.

□

Proof of Theorem 1: See the Appendix.

In summary, because all DCT coefficients matrices are divided by the same quantization table in the JPEG compression process, the relationship between two DCT coefficients of the same coordinate position will not be changed after the quantization process. The only exception is that "*greater than*" or "*less than*" may become "*equal*" due to the rounding effect of quantization. The above theorem assumes that the same quantization table is used for the whole image. Extension of the invariance property to the case of variable quantization table is included in Appendix B.

In some practical cases, the quantization table $\mathbf{Q}$ is known or can be estimated in the authenticator. This is true because a JPEG compression file includes the quantization table. If the recompression process is not allowed in application, *i.e.*, the image can not be compressed, decompressed and compressed again, this quantization table will be the only one that has been used during the compression process. Therefore, the DCT coefficients after compression will be limited in a specific range according to the following theorem.

Theorem 2 *Use the parameters defined in Theorem 1. Assume a fixed threshold* $k \in \Re$. $\forall \nu$, *define* $\tilde{k}_\nu \equiv$ *Integer Round* $(\frac{k}{\mathbf{Q}(\nu)})$. *Then,*

if $\mathbf{\Delta F_{p,q}}(\nu) > k$,

$$\mathbf{\Delta \tilde{F}_{p,q}}(\nu) \geq \begin{cases} \tilde{k}_\nu \cdot \mathbf{Q}(\nu), & \frac{k}{\mathbf{Q}(\nu)} \in Z, \\ (\tilde{k}_\nu - 1) \cdot \mathbf{Q}(\nu), & elsewhere, \end{cases} \tag{5}$$

else if $\mathbf{\Delta F_{p,q}}(\nu) < k$,

$$\mathbf{\Delta \tilde{F}_{p,q}}(\nu) \leq \begin{cases} \tilde{k}_\nu \cdot \mathbf{Q}(\nu), & \frac{k}{\mathbf{Q}(\nu)} \in Z, \\ (\tilde{k}_\nu + 1) \cdot \mathbf{Q}(\nu), & elsewhere, \end{cases} \tag{6}$$

else $\mathbf{\Delta F_{p,q}}(\nu) = k$,

$$\mathbf{\Delta \tilde{F}_{p,q}}(\nu) = \begin{cases} \tilde{k}_\nu \cdot \mathbf{Q}(\nu), & \frac{k}{\mathbf{Q}(\nu)} \in Z, \\ (\tilde{k}_\nu \ or \ \tilde{k}_\nu \pm 1) \cdot \mathbf{Q}(\nu), & elsewhere. \end{cases} \tag{7}$$

□

In Theorem 2, k is a designated threshold value used to bound the difference of two DCT coefficients of the same position in two separate 8×8 blocks of an image. In contrast, Theorem 1 only describes the invariance property of the sign of $\mathbf{\Delta F_{p,q}}$. We can consider Theorem 1 as a special case of Theorem 2 (with k set to be 0). Several different k's (*e.g.*, a series of binary division of a fixed dynamic range) can be used for a single authentication system of different levels of strength. Based on Theorem 2, we can read the quantization table $\mathbf{Q}$ from a bitstream and use Equations (5), (6) and (7) to predict the difference relationships those coefficients have after compression.

As shown in Figure 2, by applying Theorem 1 and Theorem 2, we can extract feature codes Z of an image from the relationship between two DCT coefficients of the same position in two separate 8×8 blocks. These feature codes are then encrypted as a signature. For the authentication process, a user has to calculate the DCT coefficients of the image, and compare them to the features decrypted from the digital signature S. This image is said to be matched if all the DCT coefficient relationships satisfy the criteria predicted by the features of the original image.

3.2 Image Analyzer: Feature Extraction

Figure 3(a) is the flow chart of the feature extraction process. First, an image X captured directly by a digital camera, a digital camcorder or computer software is sent into the image ana-

lyzer. Each 8×8 block of this image is then transformed to the DCT coefficients.

After all these processes, there are three loops for generating feature codes:

- Loop 1: Generate N sets of feature codes, $Z_{n,p}$, $n = 1$ to N. Each set uses different k and b_n, where k is defined in Theorem 2, b_n is the number of DCT coefficient compared in each block pair. A different threshold, k, and the number of coefficients, b_n, are used for each set.
- Loop 2: Iterate over all possible block pairs, $p = p_1$ to $p_{\frac{\wp}{2}}$, where $\wp$ is the total number of blocks in the image.
- Loop 3: Iterate over each of the b_n selected coefficient pairs.

In Loop 1, N sets of feature codes are generated. For each set, different threshold k and number of coefficients b_n are used. Parameter b_n represents how many bits are generated in each block. Parameter k represents the precision threshold used in Theorem 2. The first set, $k = 0$, protects the sign of $\Delta F_{p,q}$. From the second set to the last set, k's are set to protect the magnitude of $\Delta F_{p,q}$ with increasing accuracy. For example, if the dynamic range of ΔF is 256; k's can be set to 0, 128, 64, 32, 16, *etc.* The larger N is, the more precisely the coefficient differences are protected. We will discuss how to define the thresholds in more detail later in this section.

In Loop 2, we need to form DCT blocks into pairs. As defined in Theorem 2, the DCT coefficient difference between block p and block q is computed. Let us denote one set of blocks $P_p = \{p_1, p_2, ..., p_{\frac{\wp}{2}}\}$ and another set of blocks $P_q = \{q_1, q_2, ..., q_{\frac{\wp}{2}}\}$. For example, P_p can be all the even blocks, $\{0, 2, 4, ..., \wp - 1\}$, and P_q can be all the odd blocks, $\{1, 3, 5, ..., \wp - 2\}$. The formation of all blocks in an image into pairs can be based on any arbitrary mapping function, W, as long as the following conditions are held.

$$P_q = W(P_p), \tag{8}$$

and

$$P_p \cap P_q = \emptyset, \quad P_p \cup P_q = P. \tag{9}$$

If redundancy is allowed, P_p and P_q each may contain more blocks than $\frac{\wp}{2}$.
The choice of the mapping function W can be served as a secret parameter used to enhance the security of authenticator. For example, the image analyzer uses a seed to generate the mapping

function W and provides the seed with the feature codes to the authenticator. Each authenticator can transform this seed into the mapping function. This transformation method will not be publicized. Therefore, each manufacturer of the image authenticator can implement his/her own transformation method.

In Loop 3, for each block, we compare the b_n selected values (indexed in the zig-zag order) in the DCT domain. Both DC and AC values in the DCT domain are used. At first, the difference of DC values in block p and q, $\mathbf{\Delta F}_{p,q}(1)$, is used for comparison. If this value is smaller than k, then a feature code bit $z = 0$ is added to the end of the previous feature code. Otherwise, if this value is greater or equal to k, we will assign $z = 1$. (We classify two cases, "greater" and "equal", to the same type because the probability of $\mathbf{\Delta F_{p,q}}(\nu) = 0$ is quite small. If they are classified into three different types, i.e., "greater", "equal" and "less than", two bits should be used in this case. This will result in the increase of feature codes.) Thereafter, the differences in selected AC values are compared with k. Only b_n parameters, including one DC difference and $b_n - 1$ AC differences, are used in this process. After Loops 1, 2 and 3 are completed, the feature codes, Z, of this image are generated.

Usually, the b_n selected positions are located in the low and middle frequency bands for the following two reasons: 1.) they are usually larger than the high-band coefficients because of energy concentration, and 2.) their values are usually conserved after JPEG compression because the values in the quantization table, $\mathbf{Q}$, in these bands are small. For security, the manufacturer can choose from the first b_n positions or from the first position and arbitrarily $b_n - 1$ AC positions in the DCT domain.

3.2.1 Choosing Precision Thresholds and Other Considerations

Theoretically, threshold values, k, can be determined arbitrarily, and they may vary for different n and ν. In our system, for the first set, all $k_{1,p}(\nu)$ are set to zeros. We use a binary division method to set thresholds for other sets in the proposed authentication system. Assume the dynamic range of $\mathbf{\Delta F_{p,q}}(\nu)$ is from $-\zeta$ to ζ. If we know that $\mathbf{\Delta F_{p,q}}(\nu) < 0$ in the first set, then we can set the threshold in the second set as $-\zeta/2$. Furthermore, if we know that this value $\mathbf{\Delta F_{p,q}}(\nu) > -\zeta/2$

in the second set, the threshold in the third set can be set as $-\zeta/4$. These thresholds result in dynamic binary decision ranges. This method protects the magnitude of $\mathbf{\Delta F_{p,q}}(\nu)$ with an increasing accuracy as more sets are being used. The larger N is, the more precisely will the coefficient differences be limited.

Define a constant ζ which is a power of 2. If $Z_{1,p}(\nu) = 0$, that is, $\mathbf{\Delta F}_{p,q}(\nu) < 0$, then we can divide the negative values into two ranges with threshold, $k_{2,p}(\nu) = -\frac{1}{2}\zeta$. Otherwise, if $Z_{1,p}(\nu) = 1$, that is, $\mathbf{\Delta F}_{p,q}(\nu) \geq 0$, then we can divide the positive values into two ranges with threshold, $k_{2,p}(\nu) = \frac{1}{2}\zeta$. This binary division will be recursively executed until $n = N$. A closed form of $k_{n,p}(\nu)$ is

$$k_{n,p}(\nu) = \zeta \sum_{i=1}^{n-1} (\frac{1}{2})^i (-1)^{Z_{i,p}(\nu)+1}, \quad n > 1. \tag{10}$$

To simplify the notation in later discussions, we use $k = k_{n,p}(\nu)$ instead.

In addition to the parameters used in the three loops, some extra information about the image is necessary for defeating attacks. In our authentication system, the signature uses the difference of the DCT values at the selected coefficient positions of block pairs. A possible attack is to make a constant change to DCT coefficients at the same location in all blocks. This will not change the difference values between pairs of DCT coefficients from two different blocks. For instance, raising the image image intensity uniformly changes the DC parameter in all blocks and defeats the previous approach. Therefore, we have to record the mean value of DCT coefficients in each (selected) position for all blocks in the signature. These additional feature codes need no more than 64 bytes. Once the DCT coefficients are changed by constant values, they will be easily detected by the deviation of their mean values.

3.3 Authentication Process

Encryption and decryption can be used to protect the feature codes, but they will not affect the codes themselves. Fig. 2 includes the authentication process. It is composed of three parts. First, the received image, $\hat{X}$ or $\hat{B}$, has to be transformed to the DCT domain, $\hat{F}$. This involves the DCT transform block by block if a raw image, $\hat{X}$, is used. If the JPEG compressed bitstream, $\hat{B}$, is used, a parser has to be used for reconstructing the Huffman Table and Quantization Table, $\hat{\mathbf{Q}}$.

The signature, S, has to be decrypted to reconstruct feature codes, Z. After $\hat{F}$ and Z are available, they will be sent to the authentication comparator in order to determine whether this image has been manipulated.

Authentication Comparator is shown in Fig. 3(b). Similar to the three loops in the image analyzer, there are also three corresponding loops here. In Loop 1, the number of loops, n, can be different from the one used in the Image Analyzer. Fewer loops may be used for some special cases. Loop 2 and Loop 3 are the same as those used in the Image Analyzer. Inside these loops, we have to compare each of the DCT coefficient relationships obtained from the original image and that of the image of interest.

From Theorem 2, we can define

$$\hat{k} = \begin{cases} \tilde{k}_\nu \cdot \mathbf{Q}(\nu), & \frac{k}{\mathbf{Q}(\nu)} \text{ is an integer,} \\ (\tilde{k}_\nu + 1) \cdot \mathbf{Q}(\nu), & \frac{k}{\mathbf{Q}(\nu)} \text{ is not an integer } \textit{and } Z_n(\nu) = 0, \\ (\tilde{k}_\nu - 1) \cdot \mathbf{Q}(\nu), & \frac{k}{\mathbf{Q}(\nu)} \text{ is not an integer } \textit{and } Z_n(\nu) = 1. \end{cases} \tag{11}$$

(*It should be noted that $\hat{k}$ is a function of ν, p, and n.*) Observe from Fig. 3(b), if $Z_n(\nu) = 0$, that is, $\mathbf{\Delta F_{p,q}}(\nu) < k$, then $\mathbf{\Delta \hat{F}_{p,q}}(\nu) - \hat{k} \leq 0$ must be satisfied. Therefore, if $\mathbf{\Delta \hat{F}_{p,q}}(\nu) - \hat{k} > 0$, we know that some parameters of block p or q must have been modified. Similar results can be obtained in the case of $\mathbf{\Delta F_{p,q}}(\nu) \geq k$.

However, because some integer rounding noise may be introduced if the image is converted back to integral pixel values during the decode-reencode process, if the compressor and the signature generator use different chromatic resolution decimation algorithms for color images, or if the JPEG compressor calculates not-so-precious DCT, we must introduce a tolerance bound τ to the detection function. We augment the comparing process with Proposition 1:

Proposition 1: *Block p or q can be claimed as being manipulated*

if

$$\mathbf{\Delta \hat{F}_{p,q}}(\nu) - \hat{k} > \tau, \tag{12}$$

for the case of $\mathbf{\Delta F_{p,q}}(\nu) - k < 0$, (or equivalently $Z_n(\nu) = 0$),

or if

$$\Delta\hat{\mathbf{F}}_{\mathbf{p},\mathbf{q}}(\nu) - \hat{k} < -\tau, \tag{13}$$

for the case of $\Delta\mathbf{F}_{\mathbf{p},\mathbf{q}}(\nu) - k \geq 0$, *(or equivalently* $Z_n(\nu) = 1$.*)*

The tolerance, τ, is determined by the level of integer rounding errors. Optimal values for the rounding tolerance will be discussed in Section 4.1. It is worthwhile to note that the invariance properties observed in Theorem 2 and Theorem 3.3 hold despite either multiple iterations of decoding-reencoding or any compression rate.

As discussed earlier, the authenticator has to check whether this image has been manipulated by some constant change of DCT coefficients in all blocks. This process can by done by checking the mean values of DCT coefficients of each position in the authenticated image and in the original image. If their difference is larger than a predefined threshold, $\pm\tau_{\mathrm{s}}(\nu)$, then we can say this image may have been manipulated by constant changes on that position, and report it as a fake.

The result of authentication can be a binary indicator, *true or false*, for the whole image, or it may indicate the authenticity or forgery of specific parts in an image.

3.3.1 Other Considerations

Manipulation in specific block pairs can be located by the proposed technique. However, the authenticator using non-overlapping sets in Eq.(9) will not be able to identify which block in the pair has been modified. If identification of specific blocks is needed, we can use overlapping sets in Eq.(9). Identifying local changes is very useful to some applications in which both global and local contents are important. For instance, in a picture of ten people, even if a man's face has been substituted by that of other person or has been removed, other parts of the image can still be verified to authenticate the appearance of the another nine people. Another advantage is that the system can verify authenticity in a selected area (*e.g.*, some news agency may cut out boundary areas of file photos).

Boundary cropping and/or position shifting are often performed on images to suit application needs. The proposed authentication signature is sensitive to cropping and shifting. However, for

cropping, image block pairs that are not affected by cropping may still be authenticated. For shifting, if no DCT quantization is done on the shifted image (*e.g.*, shifting in the pixel domain only), the shifted image can be adjusted to the right position that results in the matched DCT block structure. Then, the DCT domain signature can be verified.

Constant intensity change in the image is sometimes expected, especially when the image is too dark or too bright. Our proposed authenticator can adapt it by loosening the threshold $\tau_s(1)$ of the mean value of DC coefficients or even not checking it.

The authenticator is sometimes expected to pass only those images that are compressed by JPEG up to a certain compression ratio or quality factor. For example, if the image is JPEG compressed below 20:1 ratio, the image is acceptable. Otherwise, if it is compressed more than 20:1 or manipulated by non-JPEG manipulation, it will fail the test. The argument for failing highly compressed images is that such images are already with poor quality and should not be considered as authentic. For this purpose, we can apply one of the following methods. The first one is to calculate the compression ratio from the raw image size and the actual file size. If it is too high, the authenticator can reject it before any authenticating process. The second method is to calculate the difference between the number of the "equal" coefficients in the block pairs from the original image and the received image. For natural images with reasonable compression ratios, the percentage of this difference should not be too high. We can set a threshold on this percentage to reject those images that have so many "equal" coefficients in the block pairs.

3.4 Encryption, Decryption and Signature Length

The feature codes are encrypted by a secret private key of the Public Key method. Refer to Section 3.2, the length, l_f, of feature codes is the summation of the comparison bits $\frac{\varphi}{2} \cdot (\sum_{n=1}^{N} b_n)$, the seeds of mapping function and selected DCT positions, and the recorded mean values. For instance, assume the image size is $320 \times 240 = 76800(bytes)$. If $N = 2$ and $b = 10, 6$ are used for generating feature codes, the two seeds are all 4 bytes, and only 16 DCT coefficient averages are recorded, then its feature code length, l_f, will be $\frac{1200}{2} \cdot 16 \cdot \frac{1}{8} + 4 + 4 + 16 = 1224(bytes)$. If necessary, an entropy coding method can be used to reduce this length.

The Public Key algorithm is used because everyone who wants to authenticate the image can easily access a public key to decrypt the signature. The most famous public key algorithm is RSA (named after its inventors, Rivest, Shamir, and Adleman)[6][14]. The key length of RSA is variable and the most commonly used length is 512 bits[14] and the data size must be smaller than the key length. If we choose to divide the feature codes into B-bit blocks, it needs $l_f \cdot \frac{1}{B}$ RSA calculations. Assume the output length of each RSA is l_r, then the signature length will be $l_f \cdot \frac{1}{B} \cdot l_r$. For instance, in previous example, if $B = 500$ and $l_r = 511$ are used, then the RSA algorithm has to be run 20 times and the signature length will be 1278 bytes. It is about $\frac{1}{60}$ of the original image size.

A problem with Public Key algorithms is their speed. In hardware, Public Key algorithm RSA is 1000 times slower than Secret Key algorithm DES, and it is about 100 times slower in software [14]. Therefore, if efficiency is more important, we can choose the Secret Key algorithm instead of the Public Key algorithm. The drawback is that users have to keep their secret keys safe, and the image can be authenticated by only the few people who own the secret key.

3.5 Example: A small 16×8 image

We will use a small 16×8 image as an example in illustrating our proposed image authenticator. The pixel values of this small image are shown in Table 1. This image is divided into two 8×8 blocks in the JPEG compression process. Therefore, $\wp = 2$. The DCT coefficients of these two blocks are shown in Table 2. For simplicity, only integral values of them are shown in the table.

First, let us consider the case of $N = 1$, *i.e.*, only one set which contains the threshold value $k = 0$ is used for the feature codes generation. Assume the first 10 coefficients ($b_1 = 10$) of the two DCT coefficients blocks are compared to generate the feature codes. Then the length of the feature codes, Z, will be 10 bits. From our observation of Table 2, $\mathbf{F_1}(0,0) = 486$ and $\mathbf{F_2}(0,0) = 727$. This position $(0,0)$ is the first value in the zig-zag order. We then change the two-dimensional representation of $\mathbf{F_1}(0,0)$ and $\mathbf{F_2}(0,0)$ to one-dimensional $\mathbf{F_1}(1)$ and $\mathbf{F_2}(1)$. We can obtain $\mathbf{\Delta F_{1,2}}(1) = -241 < 0$. Therefore, the first bit of the feature codes, Z, is 0. The second coefficients in the zig-zag order are: $\mathbf{F_1}(2) = 91$ and $\mathbf{F_2}(2) = -188$, respectively. Since $\mathbf{\Delta F_{1,2}}(2) = 279 > 0$, the second bit of the feature codes is 1. This procedure goes on until $\nu = 10$.

We then get the feature codes, Z, as: 0111100110.

If we consider the case of longer feature codes, we can set $N = 4$, $b_1 = 10$, $b_2 = 6$, $b_3 = 3$ and $b_4 = 1$. The reason for a decreasing number of b_n is that the lower frequency coefficients need more protection than the higher frequency ones. Following the procedure in Fig. 3(b), the first 10 bits of Z are the same as the previous case. The threshold values, k's , in the second set of Loop 1 depend on the Z_1 codes in the first set. In the first comparison of the second set, k is set to be a negative value, such as -128, because we have known that $\mathbf{\Delta F_{1,2}}(1) < 0$ in the calculation of the first set. Since $\mathbf{\Delta F_{1,2}}(1) = -240.875 < -128$, the 11th bit of Z is 0. Similarly, in the second comparison comparator, since we know that $\mathbf{\Delta F_{1,2}}(2) > 0$ in the first set, the threshold k should be a positive value. Therefore, $\mathbf{\Delta F_{1,2}}(2) = 279 > 128$ and the 12th bit of Z is 1. A similar procedure continues until $n = 4$. Readers can check the result of the feature codes Z to be: 01111001100100010110. The length of Z is equal to $\sum_{n=1}^{4} b_n = 20$. The feature codes Z obtained are then encrypted by a private key. For the sake of simplicity, we do not show the encrypted signature in this example.

The image in Table 1 is compressed by JPEG. Assume the Quantization Table $\mathbf{Q}$ is set to be a constant matrix with all its values equal to 16. Then the values of $\tilde{\mathbf{F}}_\mathbf{1}$ and $\tilde{\mathbf{F}}_\mathbf{2}$ are shown in Table 3. They are entropy encoded and compose a JPEG bitstream $\tilde{B}$. This bitstream is sent into the authenticator. In the authenticator, $\hat{\mathbf{F}}_\mathbf{1} = \tilde{\mathbf{F}}_\mathbf{1}$ and $\hat{\mathbf{F}}_\mathbf{2} = \tilde{\mathbf{F}}_\mathbf{2}$. According to the process in Fig. 3(b), we have to compare $\mathbf{\Delta \tilde{F}_{1,2}}$ to the feature codes Z. For instance, since $\mathbf{\Delta \tilde{F}_{1,2}}(1) = -240 < 0$ and $Z_1(1) = 0$, this value is authenticated to be true. If we choose $\hat{N} = 4$, a similar process continues until all values are compared. Otherwise, if the Quantization Table is not available in some cases, we can choose $\hat{N} = 1$ in the authenticator. The first situation , $\hat{N} = 4$, compares the coefficient difference to a finer scale, while the second situation, $\hat{N} = 1$, compares the sign of them only. However, in both situations, all values are authenticated as true.

Consider an example of manipulation. Assume the gray level of $X(0,2)$ and $X(0,3)$ are modified from 72 and 26 to 172 and 126. After manipulation, this image is also compressed by JPEG with the same Quantization Table is used as in the previous case. The difference of the DCT coefficients from the compressed bitstream are shown in Table 4. This compressed bitstream is then sent into the authenticator. The authenticator will indicate the manipulation because the mismatch of

the 4th bit of the feature codes Z, 1, and the 4th zig-zag coefficient $\mathbf{\Delta F_{1,2}}(4) = -16 < 0$ in Table 4.

3.6 Color Images and DCT-based Variable Quantization Table Compressions

In the JPEG compression standard, the color image is considered to be in the YC_bC_r format. Ordinary RGB format images have to be transformed to this format. Chromatic data are usually down-sampled at the rate of 1:2 (horizontal direction only) or 1:4 (one-half in both horizontal and vertical directions). To authenticate a color image, we first down-sample the chromatic component of the image with the sampling rate $1:4$. Then, we can generate the feature codes of Y, C_b, C_r components separately by the process in Fig. 3(a). In the authenticator, if the chromatic data of an image are sampled by $1:2$, they must be sampled again in the other direction in order to obtain a $1:4$ image. Similar authentication procedure in Fig. 3(b) is then applied to the color images.

The authentication process proposed in this section was designed according to the rules of JPEG, in which only one quantization table is used for an image. However, some other image/video compression techniques use different quantization tables in different image blocks for adaptive compression rate control, such as MPEG or future JPEG standards. In these applications, the previous proposed image authentication system is still effective if some modifications are applied. A theorem as well as some necessary modifications for the authenticator, which accepts images that are compressed by DCT-based, variable quantization table methods, is shown in Appendix A.2.

4 Performance Analysis

The image authenticator is a manipulation detector with two types of error involved: *miss* and *false alarm*[12]. 'Miss' refers to the situation in which an image is manipulated by unacceptable manipulations but the system reports the image as authentic. 'Miss' is also called *Type II error* in Hypotheses Testing. 'False alarm' means that the system reports the existence of manipulation while the image is, in fact, not modified by unacceptable manipulations. It is also called a *Type I error*. In our authentication system, the test is based on block pairs. For each block pair, we perform the following test: H_0: *the pixels in the image block pair are not modified, or modified to new values that are reachable by the JPEG compression processes.*, versus H_1: *the pixels in the image*

block pair are modified to new values that are not reachable by any JPEG process. The test function is defined in Proposition 1. Conceptual illustration of 'Miss', 'False alarm' and other scenarios are shown in Fig. 4.

The Probability of Miss, P_m, and the Probability of False Alarm, P_f, are estimated by the signature generator and are useful information for users of the authenticator. An additional evaluation, the Probability of Success, P_s, can also be used from the attacker's viewpoint. The attacker may try to manipulate the image based on his best knowledge of the authentication technique. Detailed discussion will be shown in this section.

Several variables are needed to estimate these probabilities. We can classify variables to three types: *pre-determined values*, *selectable variables*, and *stochastic variables*. The signature generator estimates a list of P_f and P_m based on different quantization tables and tolerances. Based on the quantization table used in the compressed image, the user may choose tolerances, τ, to satisfy constraints on P_f and P_m. A list of the property of system variables is shown in Table 5.

4.1 Noises from the JPEG Process and the Probability of False Alarm

Rounding noises may be added during the JPEG compression process and they may cause false alarm. In practice, computer software and hardware calculate the DCT with finite precision. For some cases, not only the input and the output of DCT operations are integers, but also some of the intermediate values. This will add rounding noises to the DCT values. The other noises come from the integer rounding process of the quantized DCT values. In general, the integral value of rounding a real number is its nearest integer. However, some application software, may drop small values in the high frequency positions. In other words, the rule of integer rounding may change at different DCT positions in different softwares. Combining these considerations, we can modify Eq.(2) to

$$\tilde{\mathbf{f}}_{\mathbf{p}}(\nu) = \ Integer\ Round(\frac{\mathbf{F}_{\mathbf{p}}(\nu) + N_d}{\mathbf{Q}(\nu)}) + N_r, \tag{14}$$

where N_d is the noise of DCT operation and N_r is the noise of integer rounding. Both of them are random variables. N_d usually depends on specific implementations and the number of recompression processes. Also, in most systems, the rounding rules are consistent across positions and thus N_r can be assumed to be zero.

The Probability of False Alarm of a block pair, P_f, represents the probability that at least one DCT difference value in the block pair triggers the detector in Proposition 1, merely because of the effect of rounding noise. We can write P_f as

$$P_f = \sum_{n=1}^{N} \sum_{\nu=1}^{b_n} \alpha_{n,\nu}, \tag{15}$$

where $\alpha_{n,\nu}$ is the probability that a DCT difference value $\mathbf{\Delta \tilde{F}_{p,q}}(\nu)$ does not satisfy the constraint of $Z_n(\nu)$. That is,

$$\alpha_{n,\nu} = \begin{cases} P[\mathbf{\Delta \tilde{F}_{p,q}}(\nu) - \hat{k} < -\tau], & given\ \mathbf{\Delta F_{p,q}}(\nu) \geq k, \\ P[\mathbf{\Delta \tilde{F}_{p,q}}(\nu) - \hat{k} > \tau], & given\ \mathbf{\Delta F_{p,q}}(\nu) < k. \end{cases} \tag{16}$$

Because of symmetry, these two probabilities will be the same. If the probability density function (pdf) of N_d is known, then we can find the Probability of False Alarm as follows. First, define a random variable $N'_{d,p}$, *s.t.*, $\lfloor f_p + \frac{1}{2} \rfloor + N'_{d,p} \equiv\ Integer\ Round(\frac{\mathbf{F_p}(\nu) + N_{d,p}}{\mathbf{Q}(\nu)})$ where $f_p = \frac{\mathbf{F_p}(\nu)}{\mathbf{Q}(\nu)}$. Its probability density function is

$$P[N'_{d,p} = n_d] = P[\ (n_d + \lfloor f_p + \frac{1}{2} \rfloor - f_p + \frac{1}{2}) \cdot \mathbf{Q}(\nu) > N_{d,p} \geq (n_d + \lfloor f_p + \frac{1}{2} \rfloor - f_p - \frac{1}{2}) \cdot \mathbf{Q}(\nu)\ ]. \tag{17}$$

The probability density function of $N'_{d,q}$ can be obtained in a similar way. With some transformations, we can obtain $\alpha_{n,\nu}$ as

$$\alpha_{n,\nu} = P[N'_{d,p} - N'_{d,q} < \hat{k}' - \tau' - \lfloor f_p + \frac{1}{2} \rfloor + \lfloor f_q + \frac{1}{2} \rfloor], \tag{18}$$

where $\hat{k}' = \frac{\hat{k}}{\mathbf{Q}(\nu)}$ and $\tau' = \frac{\tau}{\mathbf{Q}(\nu)}$. The symbol of $\lfloor \cdot \rfloor$ represents the 'floor' function.

From Eq.(15), Eq.(17), and Eq.(18), the user of the image authenticator can set suitable tolerance value τ depending on the Quantization Table reconstructed from the bitstream, the estimated variances of noises, and the thresholds. However, in practical applications, the user has to assume the pdf of N_r *a priori*. If the model assumption of N_r is not available, rules of thumbs can be used to set τ to *zero* for a JPEG bitstream which is directly compressed from an origianl gray-level image, or $\mathbf{Q}(\nu)$ for color images and most other cases.

4.2 Manipulation and the Probability of Miss

The Probability of Miss represents the *reliability* of the authenticator. To obtain the Probability of Miss of a manipulated block pair, we may assume the block p of the image is manipulated

and its corresponding block q is untouched. From the viewpoints of the signature generator, any manipulation on the block p of image can be modeled as an additive random variable matrix $\mathbf{M_p}$, *s.t.*,

$$\hat{\mathbf{f}}_{\mathbf{p}}(\nu) = \ Integer\ Round(\frac{\mathbf{F_p}(\nu) + \mathbf{M_p} + N_d}{\mathbf{Q}(\nu)}) + N_r. \tag{19}$$

In general, M_p is usually much larger than N_d, and N_r can be neglected. Therefore, the difference value of the DCT block pair is

$$\mathbf{\Delta}\hat{\mathbf{F}}_{\mathbf{p,q}}(\nu) = [\ Integer\ Round(\frac{\mathbf{F_p}(\nu) + \mathbf{M_p}(\nu)}{\mathbf{Q}(\nu)}) - Integer\ Round(\frac{\mathbf{F_q}(\nu)}{\mathbf{Q}(\nu)})\ ] \cdot \mathbf{Q}(\nu). \tag{20}$$

From Section 3.2, we know that the range of $\mathbf{\Delta F_{p,q}}(\nu)$ is confined by the multiple sets in the authentication signature, *i.e.*,

$$\mathbf{\Delta F_{p,q}}(\nu) \in [\ k_{l,\nu}, k_{u,\nu}\ ), \tag{21}$$

where $k_{l,\nu}$ is the lower bound of $\mathbf{\Delta F_{p,q}}(\nu)$ and $k_{u,\nu}$ is the upper bound. $k_{l,\nu}$ can be $-\infty$ if no lower bound exist, or $k_{u,\nu}$ can be ∞ if there is no upper bound. After JPEG compression, the range of $\mathbf{\Delta}\tilde{\mathbf{F}}_{\mathbf{p,q}}(\nu)$ should be bounded in the range of $[\hat{k}_{l,\nu}, \hat{k}_{u,\nu}]$ as proposed by Proposition 1. Therefore, the probability that the authenticator fails to detect a manipulation on the position ν of the block pair (p, q) is

$$\beta_\nu = P[\ \mathbf{\Delta}\hat{\mathbf{F}}_{\mathbf{p,q}}(\nu) \in [\hat{k}_{l,\nu}, \hat{k}_{u,\nu}]\ ], \quad given\ \mathbf{\Delta F_{p,q}}(\nu) \in [\ k_{l,\nu}, k_{u,\nu}\ ). \tag{22}$$

To obtain β_ν, we have to refer to Proposition 1 to find the probability that a manipulation passes an inequality property defined in the signature. That is

$$p_b = \begin{cases} P[\mathbf{\Delta}\hat{\mathbf{F}}_{\mathbf{p,q}}(\nu) - \hat{k} \geq -\tau], & given\ \mathbf{\Delta F_{p,q}}(\nu) \geq k, \\ P[\mathbf{\Delta}\hat{\mathbf{F}}_{\mathbf{p,q}}(\nu) - \hat{k} \leq \tau], & given\ \mathbf{\Delta F_{p,q}}(\nu) < k. \end{cases} \tag{23}$$

Because the manipulation value is the only random variable in the Eq.(23), after some transformations, we can compute the probability with

$$p_b = \begin{cases} P[\ \mathbf{M_p}(\nu) \geq \hat{k} - \tau + (\lfloor \frac{\mathbf{F_q}(\nu)}{\mathbf{Q}(\nu)} + \frac{1}{2} \rfloor - \frac{1}{2}) \cdot \mathbf{Q}(\nu) - \mathbf{F_p}(\nu)\ ], & given \mathbf{\Delta F_{p,q}}(\nu) \geq k, \\ P[\ \mathbf{M_p}(\nu) \leq \hat{k} + \tau + (\lfloor \frac{\mathbf{F_q}(\nu)}{\mathbf{Q}(\nu)} + \frac{1}{2} \rfloor + \frac{1}{2}) \cdot \mathbf{Q}(\nu) - \mathbf{F_p}(\nu)\ ], & given \mathbf{\Delta F_{p,q}}(\nu) < k. \end{cases} \tag{24}$$

We can derive β_ν from Eq.(22) and Eq.(24), i.e.,

$$\beta_\nu = P[\ m_{l,\nu} \leq \mathbf{M_p}(\nu) \leq m_{u,\nu}\ ], \tag{25}$$

where

$$\begin{cases} m_{l,\nu} &= \hat{k}_{l,\nu} - \tau + (\lfloor \frac{\mathbf{Fq}(\nu)}{\mathbf{Q}(\nu)} + \frac{1}{2} \rfloor - \frac{1}{2}) \cdot \mathbf{Q}(\nu) - \mathbf{F_p}(\nu), \\ m_{u,\nu} &= \hat{k}_{u,\nu} + \tau + (\lfloor \frac{\mathbf{Fq}(\nu)}{\mathbf{Q}(\nu)} + \frac{1}{2} \rfloor + \frac{1}{2}) \cdot \mathbf{Q}(\nu) - \mathbf{F_p}(\nu). \end{cases} \tag{26}$$

To estimate the Probability of Miss, P_m, of a specific image block pair, we need to know the probability distribution of $\mathbf{M_p}$. Use a $b_n \times 1$ vector $\hat{\mathbf{M}}_\mathbf{p}$ to represent the selected b_n elements of $\mathbf{M_p}$ with its probability density function $f(\hat{\mathbf{M}}_\mathbf{p})$ and the range set $\mathbf{RB}$ (i.e., $\mathbf{RB} = \{\hat{\mathbf{M}}_\mathbf{p}(\nu) : m_{l,\nu} \leq \hat{\mathbf{M}}_\mathbf{p}(\nu) \leq m_{u,\nu}\}$) specified by the signature. Then,

$$P_m = \int_{\mathbf{RB}} f(\hat{\mathbf{M}}_\mathbf{p})\, d\hat{\mathbf{M}}_\mathbf{p}. \tag{27}$$

Since the possible manipulation to an image block is arbitrary, from the signature generator's viewpoint, there is no exact distribution function to depict it. However, we can assume that the manipulated image block must be homogeneous to its adjacent blocks, otherwise this manipulated image block will cause a severely artificial effect, which is easily detected by people. We may use a multivariate zero-mean Gaussian distribution, $\mathbf{\Delta X_p} : N[\mathbf{0}, \sigma^2\mathbf{R}]$, to model the probability of additive intensity change on each pixel in the block. The variance parameter σ^2 depends on what kind of manipulation is expected. Some experimental values are shown in Table 6. We can observe that manipulations performed by replacing or cloning have higher variance values than manipulations performed by quantization, filtering or shifting, because they have to change more in intensities to result in different visual meaning. The probability distribution of $\mathbf{M_p}$ in the DCT domain can be obtained from the probability distribution of intensity changes in the pixel domain, given the covariance matrix $\mathbf{R}$ of the random changes of pixels. That is,

$$\mathbf{M_p} : \; N[\, \mathbf{0}, \sigma^2\mathbf{DRD}^t \,], \tag{28}$$

where $\mathbf{D}$ is the DCT transform matrix defined in Eq. (1).

To evaluate an authentication system, we can calculate the probability of miss based on the two extreme cases of $\mathbf{\Delta X_p}$, uncorrelated and fully correlated. In the uncorrelated case, i.e. $\mathbf{R} = \mathbf{I}$, manipulations on each pixels are totally uncorrelated. They are similar to Additive White Gaussian Noise. Therefore, $\mathbf{M_p} : \; N[\, \mathbf{0}, \sigma^2\mathbf{I} \,]$ because $\mathbf{DD}^t = \mathbf{I}$ in DCT. In this case, the probability of miss

P_m will be

$$P_m = \prod_{\nu=1}^{b_n} \beta_\nu = \prod_{\nu=1}^{b_n} [\ \Phi(\frac{m_{u,\nu}}{\sigma}) - \Phi(\frac{m_{l,\nu}}{\sigma})\ ], \tag{29}$$

where $\Phi(.)$ is the *standard normal distribution function*, $\Phi(z) = \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} e^{\frac{-u^2}{2}} du$. In the fully correlated case, assume there are no weighting on specific positions in the pixel domain. The intensity change of each pixel is the same, i.e., $\mathbf{R} = [r_{ij} :\ i,j = 1..64]$ where $r_{ij} = 1$. Then, $\mathbf{M_p} : N[\ \mathbf{0},\ \sigma^2\hat{\mathbf{R}}]$ can be obtained where $\hat{\mathbf{R}} = [r_{ij} :\ i,j = 1..64]$ with $r_{1,1} = 64$ and $r_{i,j} = 0, elsewhere$. This can be easily verified because only the DC value in the DCT domain changes in this case. In this case, P_m will be

$$P_m = \Phi(\frac{m_{u,1}}{8\sigma}) - \Phi(\frac{m_{l,1}}{8\sigma}). \tag{30}$$

Finally, given a specific image block pair with the Quantization Table, the tolerance values, and the thresholds, we can use Eq. (29), (30), and Table 6 to estimate the range of Probability of Miss in an image block pair.

The above computation estimates the Probability of Miss for a single block pair. For some applications, the final authentication result is reported for the entire image and detection of manipulated blocks may not be necessary. In these cases, the miss probability is the product of the miss probability of all the block pairs manipulated, and thus is usually very low.

4.3 The Probability of Success of the Attacker

From the attacker's point of view, it is desirable to know the chances of success. There are two kinds of attack. First, attackers may manipulate the image to a new image with different visual meaning. In this case, the attacker may use replacing, deletion, or cloning to change the pixel values. The strategy of this kind of manipulation is to make the change too difficult to figure out by people. Second, attackers may manipulate the image (or synthesize an image) based on their knowledge about the authentication algorithm and the information in the signature. The strategy of these attacks is to generate a different image to fool the authenticator. In our system, this cannot happen if all the information used in designing the signature is kept secret. However, it is possible that attackers would obtain the information and try to manipulate the image accordingly. Despite this possibility, the proposed system is still effective, because these manipulations must be conducted in the DCT domain, and they will cause artifacts noticeable by people. In the following

two subsections, we consider these two types of attacks and their probabilities of success.

4.3.1 Attacks on Visual Meaning

Manipulation of the visual meaning of an image is a common attack. To evaluate the probability of his success, an attacker has to use the result of his change to the blocks and an estimation of the other parameters. For instance, he can compute the DCT values of the changed blocks. If the manipulated compressed image will not be further recompressed, he knows the quantization table. Otherwise, he can generate a list of Probabilities of Success based on different quantization tables. The information regarding the thresholds used may be obtained by extracting the signature. If this information is not available, for a one-set signature, setting the thresholds to zero is a reasonable estimation. The tolerances that would be used in the authenticator are not predictable by the attacker. But he can guess some reasonable values such as zero or $\mathbf{Q}(\nu)$, and observe their effect. The only stochastic parameter for estimating the Probability of Success would be the values of the DCT coefficients in the other block, by way of comparison, because its position is unknown to the attacker.

The Probability of Success, P_s, of a manipulated block should be

$$P_s = \prod_{\nu=1}^{b_n} \gamma_\nu, \tag{31}$$

where γ_ν is the probability of success for a DCT coefficient. For each coefficient, because the attacker does not know the exact value of the other coefficient in the block pair, he will not know the exact comparison bits in the signature. In other words, the attacker does not know where the difference values would be, in the ranges divided by the thresholds of authentication signature. Therefore, γ_ν should be

$$\begin{aligned}
\gamma_\nu &= \textstyle\sum_K \{P[\mathbf{\Delta\hat{F}_{p,q}}(\nu) - \hat{k}_l \geq -\tau,\ \mathbf{\Delta F_{p,q}}(\nu) \geq k_l] + P[\mathbf{\Delta\hat{F}_{p,q}}(\nu) - \hat{k}_u \leq \tau,\ \mathbf{\Delta F_{p,q}}(\nu) < k_u]\} \\
&= \textstyle\sum_K \{P[\ \mathbf{F_q}(\nu) \leq (\hat{\mathbf{f}}_\mathbf{m}(\nu) + \frac{1}{2}\ )\mathbf{Q}(\nu) - \hat{k}_l + \tau\ ,\ \mathbf{F_q}(\nu) \leq \mathbf{F_p}(\nu) - k_l] + \\
&\qquad P[\ \mathbf{F_q}(\nu) \geq (\hat{\mathbf{f}}_\mathbf{m}(\nu) - \tfrac{1}{2}\ )\mathbf{Q}(\nu) - \hat{k}_u - \tau\ ,\ \mathbf{F_q}(\nu) > \mathbf{F_p}(\nu) - k_u]\} \\
&\equiv \textstyle\sum_K \gamma_{\kappa,\nu}
\end{aligned} \tag{32}$$

with

$$\hat{\mathbf{f}}_\mathbf{m}(\nu) = \lfloor \frac{\mathbf{F_p}(\nu) + \mathbf{M_p}(\nu)}{\mathbf{Q}(\nu)} + \frac{1}{2} \rfloor, \tag{33}$$

where the whole real numbers are divided into K ranges by the thresholds of the authentication signature, while the upper bound and the lower bound for each range are k_l and k_u. It should be noted that the k_l and k_u are different in each range. For instance, if there is only one threshold, k, used in a specific DCT position, then $[k_l, k_u) = [-\infty, k)$ in the first range and $[k_l, k_u) = [k, \infty)$ in the second range. The lower bound of a range is equal to the upper bound of its previous range. The relation of the range numbers, K, and the set numbers, N, is $K = 2^{N+1}$.

In general cases, we can assume $\mathbf{F_q}(\nu)$ to be a zero-mean Gaussian distribution random variable, with a variance of σ_ν^2. Therefore, $P[\mathbf{F_q}(\nu) \leq \mathbf{F_p}(\nu) - k_l]$ can be written as $\Phi(\frac{\mathbf{F_p}(\nu)-k_l}{\sigma_\nu})$. The probability of $P[\mathbf{F_q}(\nu) > \mathbf{F_p}(\nu) - k_u]$ and other probabilites can be representated with similar format. We can obtain the success probability of each coefficient, $\gamma_{\kappa,\nu}$, as

$$\gamma_{\kappa,\nu} = \begin{cases} \min\ [\ 0,\ \Phi(\frac{(\hat{\mathbf{f}}_\mathbf{m}(\nu)+\frac{1}{2})\mathbf{Q}(\nu)-\hat{k}_l+\tau}{\sigma_\nu}) - \Phi(\frac{\mathbf{F_p}(\nu)-k_u}{\sigma_\nu})\ ]\ , & \hat{\mathbf{f}}_\mathbf{m}(\nu) < \frac{\mathbf{F_p}(\nu)-\tau+\hat{k}_l-k_l}{\mathbf{Q}(\nu)} - \frac{1}{2} \\ \Phi(\frac{\mathbf{F_p}(\nu)-k_l}{\sigma_\nu}) - \Phi(\frac{\mathbf{F_p}(\nu)-k_u}{\sigma_\nu})\ , & elsewhere, \\ \min\ [\ 0, \Phi(\frac{\mathbf{F_p}(\nu)-k_l}{\sigma_\nu}) - \Phi(\frac{(\hat{\mathbf{f}}_\mathbf{m}(\nu)-\frac{1}{2})\mathbf{Q}(\nu)-\hat{k}_u-\tau}{\sigma_\nu})\ ]\ , & \hat{\mathbf{f}}_\mathbf{m}(\nu) > \frac{\mathbf{F_p}(\nu)+\tau+\hat{k}_u-k_u}{\mathbf{Q}(\nu)} + \frac{1}{2}. \end{cases} \tag{34}$$

Therefore, the attacker can estimate P_s from Eq.(31)-(34). It should be noticed that the attacker has to transform his manipulations to the DCT domain and find the statistical variance of the DCT coefficients of all blocks in the image.

From eq.(34), we can observe that $\forall\nu$, if $\hat{\mathbf{f}}_\mathbf{m}(\nu) \in [\ \frac{\mathbf{F_p}(\nu)-\tau+\hat{k}_l-k_l}{\mathbf{Q}(\nu)} - \frac{1}{2},\ \frac{\mathbf{F_p}(\nu)+\tau+\hat{k}_u-k_u}{\mathbf{Q}(\nu)} + \frac{1}{2}\ ]$ in all ranges, then the probability of success P_s will be equal to 1. After some transformation, we can represent this range with the form of the manipulation,$\mathbf{M_p}(\nu)$, on the DCT domain, *i.e.*,

$$\{\lfloor\frac{\mathbf{F_p}(\nu)-k_l}{\mathbf{Q}(\nu)}+\frac{1}{2}\rfloor-\frac{1}{2}-\frac{\mathbf{F_p}(\nu)-\hat{k}_l}{\mathbf{Q}(\nu)}\}\cdot\mathbf{Q}(\nu)-\tau\ \leq\ \mathbf{M_p}(\nu) < \{\lfloor\frac{\mathbf{F_p}(\nu)-k_u}{\mathbf{Q}(\nu)}+\frac{1}{2}\rfloor+\frac{1}{2}-\frac{\mathbf{F_p}(\nu)-\hat{k}_u}{\mathbf{Q}(\nu)}\}\cdot\mathbf{Q}(\nu)+\tau. \tag{35}$$

Eq.(35) specifies the range of manipulation in the DCT domain in which an attacker can definitely change the coefficients without triggering the authentication alarm. The size of this range is equal to $\mathbf{Q}(\nu)$.

An attacker may have no idea how to calculate the range specified in eq.(35). In general cases of this situation, is there an undetectable manipulation range? Derived from Eq.(35), the range of

undetectable manipulation would be

$$\mathbf{M_p}(\nu) \in [a - \mathbf{Q}(\nu), a], \tag{36}$$

where a is a practical case dependent variable located in the range of $[\tau - 1.5\mathbf{Q}(\nu), \tau + 2.5\mathbf{Q}(\nu)]$. In other words, no universal undetectable bound exists for general authenticators. Without the knowledge of the authenticator and the signature, an attacker has no way to manipulate maliciously an image without taking the risk of triggering the authentication alarm.

4.3.2 Attacks by the Rule of Authentication

Some attackers may try to manipulate an image based on their knowledge no matter what the manipulated image looks like. Attackers may want to manipulate or even synthesize an image that can fool the system without triggering the alarm. In our authentication system, the security mechanism is based on: 1.) the private key used for the signature encryption, which ensures the signature cannot be forged; 2.) the secret transformation mechanism to generate the mapping function of block pairs from a seed; and 3.) the secret method used to represent the selected DCT coefficient positions in block pairs from another seed. In the following paragraphs, we will discuss the four possible situations, which depend on the extent of knowledge that the attacker has about the system and the signature.

Security Level I: All information in the signature is secret

If all information in the signature is kept secret from the attacker, the performance of the proposed authenticator has been analyzed in the above subsections. The only possible attack is to make a constant change to DCT coefficients at the same location in all blocks. We have solved this problem by protecting the mean values of DCT coefficients as discussed in Section 3.2.1 and Section 3.3.

Security Level II: The selected DCT coefficient positions are known

The two secret transformation mechanisms to represent the selected DCT coefficient positions and the mapping function of block pairs from the two seeds are pre-designed by the manufacturer of authentication system. For instance, they can be secret bytecodes embeded in the system.

Therefore, even though these two seeds are easily accessed by the attacker because anyone use the corresponding public key can extract the signature, the real selected positions and mapping function are still unknown to the attacker.

These two mechanisms are supposed to be kept secret from the attacker. However, an attacker may try his best to find out these rules. Once he knows the real selected positions represented by the seed in the signature, he can arbitrarily change the coefficients that are not compared in the authentication process without triggering the authentication alarm. To avoid this problem, the authenticator can change the rule of selected positions, block by block, in a more complicated method, to make this rule more difficult to discover. Furthermore, if this threat is still considerable, the signature generator can eventually use all the 64 DCT coefficients in the signature.

Security Level III: The mapping function of block pairs is known

Once the mapping function is known, the attacker also knows the DCT differences for each pair of the image. For example, if only the sign of the DCT differences are used for authentication, and the attacker knows $\mathbf{\Delta}\hat{\mathbf{F}}_{\mathbf{p},\mathbf{q}}(\nu) = 10$ in the origianl compressed image, he can manipulate this value to $\mathbf{\Delta}\hat{\mathbf{F}}_{\mathbf{p},\mathbf{q}}(\nu) = 60$, which will not be detected by the authenticator. In this case, multiple threshold sets, $\mathbf{k}$, should be used because they can protect each coefficient with a higher accuracy. Although the DCT differences are known to the attacker, he still cannot manipulate those DCT coefficients too much, because the range of manipulation in the multiple sets is limited.

Security Level IV: The private key used for signature encryption is known

To ensure that the authentication signature is generated by the original source, not the attacker, we use the private key which is known to the source only. This is similar to the approach used in [5]. Anybody may use the corresponding public key to extract the signature. But only the authorized signature generator has the right private key to encrypt the seeds and the feature codes of an image to generate the signature. In the extreme hypothetical case, the private key used by the original source may be known as well. This is a general problem for any secure communication and is out of the scope of this paper. However, once the private key is also known by the attacker, he can produce another signature for the new manipulated image, and pretends it is the authentic one. One possible way to solve this problem is to ask the image owner to register and store its signature

in a trustable institution. The institution stamps a digital postmark on the signature to prove its receiving time and its originality. Therefore, the forged signature will be considered invalid because its originality cannot be proven.

It is also worth noting that subjective inspection may provide another way of protecting the image authenticity. The attacker may try to develop special manipulations in the DCT domain in order to break the proposed scheme. But at the same time, it is difficult for the attacker to control the resulting artifacts in the pixel domain. These artifacts may be very obvious to humans, even as they are able to circumvent the authentication protection.

5 Experimental Results

5.1 Experiments

In evaluating the proposed image authenticator, we took the well-known 'Lenna' image as the raw image X in Fig. 2. The original image is shown in Fig. 5. In our experiment, the authentication results together with the DCT coefficients $\hat{F}$ are sent into an IDCT to convert those coefficients to the pixel domain. Those blocks detected as manipulated will be highlighted in the pixel domain. The highlighted intensity added to a block is proportional to the number of manipulated coefficients. Therefore, the more pixels modified, the brighter this block will be.

Experiment 1: Lossy Compression

The 'Lenna' image is compressed with compression ratio $9:1$. The authentication signature is generated based on the original 'Lenna' image. The compressed bitstream is sent to the system for authentication. The tolerance bound of the authenticator is set as $\tau = 0$, since no integral rounding is involved. As previously predicted, the authenticator will verify the compressed image as authentic and decompress this image perfectly. The authentication result is shown in Fig. 6.

Experiment 2: Recompression and False Alarm

The original image is compressed with a compression ratio $6:1$. Then, this image is decompressed by Photoshop, rounded to integral values, and recompressed into an image with compression

ratio 9 : 1. In this case, if we use $\tau = \mathbf{Q}(\nu)$, the recompression process (9:1) will not trigger the manipulation detector and the final compressed image is still verified as authentic. The final decoded image is similar to Fig. 6.

Experiment 3: Detection of Manipulation

The third experiment is made by manipulating the image by deleting the feather fringe hanging over the hat brim, just above Lenna's right eye. This feather area (16×16 pixels) is removed and cloned by its neighboring pixels. This image is shown in Fig. 7. The authentication result is shown in Fig. 8. It is clearly shown that the manipulated part has been detected as fake; it is highlighted by the authenticator. The other example is shown in Fig. 9. In this image, Lenna's mouth was flipped in the vertical direction. Its authenticaiton result is shown in Fig. 10.

5.2 Practical System Performance

From Fig. 11 to Fig. 14, we show the practical system performance by revealing the Probability of Miss and the Probability of Success in different cases. Fig. 11 shows the median values of the Probability of Miss in several images. The tolerance value, $\tau = 0$, the threshold constant, $\zeta = 32$, and the standard deviation of manipulations, 35, are used in this figure. (If not specified, these settings are kept the same for other figures.) In these figures, a (b_1, b_2) symbol means b_1 bits are used in the first set of the feature codes, and b_2 are used in the second set. For instance, 10 bits used per block pair are denoted by a $(10, 0)$ symbol.

Fig. 12 is an indication of the Probability of Miss with different standard deviations of manipulations. Users can refer to Table 6 for estimating the authentication system performance during different kinds of manipulations.

Although the authentication system is always valid for an image with different quality factors of JPEG compression, the Probability of Miss is still variant because the allowable range for any modification on the images is larger as the quality factor decreases. This is shown in Fig. 13.

Because the Probability of Success is case dependent, it can only be estimated by the attacker's

real manipulations. It is impractical to compute a single P_s for an image. Therefore, as an example, we manipulate the DCT coefficients on each position of a block, with different values to see the Probability of Success for that block pair. They are shown in Fig. 14.

Observing these figures, we know that the more bits used, the less the Probability of Miss will be. Also, we know that if the same number of bits was used, the performance of authentication signature with two threshold sets will be better than the one with only one set.

6 Conclusion

In this paper, we have proposed an image authentication technique that distinguishes the JPEG lossy baseline compression from other malicious manipulations. In practical applications, images may be compressed and decompressed several times and still considered as authentic. Some manipulations, *e.g.*, integral value rounding, color space transformation and cropping, are also considered acceptable in some applications. We propose a technique that allows JPEG lossy compression but prevents malicious manipulations. Our proposed technique can be customized to different requirements and protect such "desirable" manipulations. Our analytic and empirical performance analysis has shown the effectiveness of this system.

Appendix

A.1 Proof of Theorem 1

Proof 1 : $\forall a, b, c \in \Re$, *assume* $a = A + r(a)$, $b = B + r(b)$, *and* $c = C + r(c)$, *where* $A, B, C \in Z$ *are the rounding integers of a,b,c, respectively, and* $-0.5 \leq r(a), r(b), r(c) < 0.5$.
Assume $a - b > c$, *then*

$$A + r(a) - B - r(b) > C + r(c). \tag{37}$$

Therefore,

$$A - B - C > r(c) + r(b) - r(a). \tag{38}$$

If $r(c) \neq 0$, *then* $-1.5 < r(c) + r(b) - r(a) < 1.5$. *Therefore, since* $A, B, C \in Z$,

$$A - B - C > -1.5 \geq -1, \tag{39}$$

then,

$$A - B \geq C - 1. \tag{40}$$

If c is an integer, then $-1.0 < r(c) + r(b) - r(a) < 1.0$ *because* $r(c) = 0$. *Therefore,*

$$A - B - C > -1.0 \geq 0. \tag{41}$$

Then,

$$A - B \geq C. \tag{42}$$

Then, eq.(5) can be proved by substituting a *by* $\mathbf{F_p}(u,v)$, A *by* $\tilde{\mathbf{F}}_\mathbf{p}(u,v)$, b *by* $\mathbf{F_q}(u,v)$, B *by* $\tilde{\mathbf{F}}_\mathbf{q}(u,v)$, c *by* $\frac{k}{\mathbf{Q}(u,v)}$, C *by* $\tilde{k}_{u,v}$, *and with every parameter multiplied by* $\mathbf{Q}(u,v)$. *Also, eq.(6) and eq.(7) can be proved by using similar methods.*

□

A.2 DCT-based Variable Quantization Table Compressions

In some image/video compression techniques, different quantization tables are used in different image blocks for adaptive compression rate control, such as in MPEG or future JPEG standards. In these cases, the previous proposed image authentication system is still effective if we add Theorem 3.

Theorem 3 *Assume* $\mathbf{F_p}$ *and* $\mathbf{F_q}$ *are DCT coefficient vectors of two arbitrarily* 8×8 *nonoverlapping blocks of image X, and* $\mathbf{Q_p}$ *and* $\mathbf{Q_q}$ *are their corresponding quantization tables.* $\forall\nu \in [1,..,64]$ *and* $p,q \in [1,..,\wp]$, *where* $\wp$ *is the total number of blocks. Define* $\mathbf{\Delta F_{p,q}} \equiv \mathbf{F_p} - \mathbf{F_q}$ *and* $\mathbf{\Delta\tilde{F}_{p,q}} \equiv \tilde{\mathbf{F}}_\mathbf{p} - \tilde{\mathbf{F}}_\mathbf{q}$ *where* $\tilde{\mathbf{F}}_\mathbf{p}$ *is defined as* $\tilde{\mathbf{F}}_\mathbf{p}(\nu) \equiv$ *Integer Round*$(\frac{\mathbf{F_p}(\nu)}{\mathbf{Q_p}(\nu)}) \cdot \mathbf{Q_p}(\nu)$ *and* $\tilde{\mathbf{F}}_\mathbf{q}$ *is defined as* $\tilde{\mathbf{F}}_\mathbf{p}(\nu) \equiv$ *Integer Round*$(\frac{\mathbf{F_q}(\nu)}{\mathbf{Q_q}(\nu)}) \cdot \mathbf{Q_q}(\nu)$. *Assume a fixed threshold* $k \in \Re$. *Therefore, the following properties hold:*

- *if* $\mathbf{\Delta F_{p,q}}(\nu) \geq k$, *then* $\mathbf{\Delta\tilde{F}_{p,q}}(\nu) \geq k - \frac{1}{2}(\mathbf{Q_p}(\nu) + \mathbf{Q_q}(\nu))$,
- *else if* $\mathbf{\Delta F_{p,q}}(\nu) < k$, *then* $\mathbf{\Delta\tilde{F}_{p,q}}(\nu) \leq k + \frac{1}{2}(\mathbf{Q_p}(\nu) + \mathbf{Q_q}(\nu))$.

□

Theorem 3 can be used to handle the case with variable quantization tables. We can refine Eq.(11) as

$$\hat{k} = \begin{cases} k + \frac{1}{2}(\mathbf{Q_p}(\nu) + \mathbf{Q_q}(\nu)), & Z_n(\nu) = 0,\ i.e.,\ \mathbf{\Delta F_{p,q}}(\nu) < k, \\ k - \frac{1}{2}(\mathbf{Q_p}(\nu) + \mathbf{Q_q}(\nu)), & Z_n(\nu) = 1,\ i.e.,\ \mathbf{\Delta F_{p,q}}(\nu) \geq k. \end{cases}$$

Therefore, observe from Fig. 3(b), if $Z_n(\nu) = 0$, that is, $\mathbf{\Delta F_{p,q}}(\nu) < k$, then $\mathbf{\Delta \hat{F}_{p,q}}(\nu) - \hat{k} \leq 0$ must be satisfied. In other words, if $\mathbf{\Delta \hat{F}_{p,q}}(\nu) - \hat{k} > 0$, we know that some pixels in block p or q must have been modified. Similar results can be obtained for the case of $\mathbf{\Delta F_{p,q}}(\nu) \geq k$.

Except the above modifications, the authentication system designed for the variable quantization table cases would be the same as the proposed system for the case with equal quantization tables. A detailed discussion of this case is in [8].

References

[1] G. W. Braudaway, K. A. Magerlein and F. Mintzer, "Protecting Publicly-Available Images with a Visible Image Watermark," *IBM T.J. Watson Research Center Research Report*, RC 20336, January 1996.

[2] I. J. Cox, J. Kilian, T. Leighton and T. Shamoon, "Secure Spread Spectrum Watermarking for Multimedia," *NEC Research Institute Technical Report*, 95-10, 1995.

[3] S. Craver, N. Memon, B. L. Yeo and M. Yeung, "Can Invisible Watermarks Resolve Rightful Ownerships?," *IBM T.J. Watson Research Center Research Report*, RC 20509, July 1996.

[4] Whitfield Diffie and Martin E. Hellman, "New Directions in Cryptography," *IEEE Trans. on Information Theory*, Vo. IT-22, No. 6, pp.644-654, November 1976.

[5] Gary L. Friedman, "The Trustworthy Digital Camera: Restoring Credibility to the Photographic image," *IEEE Trans. on Consumer Electronics*, Vol.39, No.4, pp.905-910, November 1993.

[6] Charlie Kaufman, Radia Perlman and Mike Speciner, "Network Security: Private Communication in a Public World," *Prentice-Hall Inc.*, pp.129-162, 1995.

[7] C.-Y. Lin and S.-F. Chang, "A Robust Image Authentication Method Surviving JPEG Lossy Compression," *SPIE Storage and Retrieval of Image/Video Databases*, San Jose, January 1998.

[8] C.-Y. Lin and S.-F. Chang, "An Image Authenticator Surviving DCT-based Variable Quantization Table Compressions," *CU/CTR Technical Report 490-98-24*, November 1997.

[9] C.-Y. Lin and S.-F. Chang, "A Watermark-Based Robust Image Authentication Method Using Wavelets," ADVENT Project Report, Columbia University, April 1998.

[10] B. M. Macq and J. J. Quisquater, "Cryptology for Digital TV Broadcasting," *Proceedings of the IEEE*, vol. 83, No. 6, pp. 944-957, June 1995.

[11] Kineo Matsui and Kiyoshi Tanaka, "Video-Steganography: How to Secretly Embed a Signature in a Picture," *IMA Intellectual Property Project Proceedings*, Vol. 1, pp. 187-206, 1994.

[12] Louis L. Scharf, "Statistical Signal Processing – Detection, Estimation, and Time Series Analysis," *Addison Wesley Inc.*, pp. 103-178, 1991.

[13] Marc Schneider and Shih-Fu Chang, "A Robust Content Based Digital Signature for Image Authentication," *IEEE International Conf. on Image Processing*, Laussane, Switzerland, October 1996.

[14] Bruce Schneier, "Applied Cryptography,"*John Willey & Sons. Inc.*, pp.461-502, 1996

[15] S. D. Silvey, "Statistical Inference," *Chapman & Hall Inc.*, pp. 94-107, 1975.

[16] Grogory K. Wallace, "The JPEG Still Picture Compression Standard," *Communications of the ACM*, Vol.34(4). pp. 30-44, April 1991.

[17] Steve Walton, "Image Authentication for a Slippery New Age," *Dr. Dobb's Journal*, pp. 18-26, April 1995.

[18] M. Yeung and F. Mintzer, "An Invisible Watermarking Technique for Image Verification," *Proce. of ICIP*, Santa Barbara, October 1997.

[19] Jian Zhao and Eckhard Koch, "Embedding Robust Label into Images for Copyright Protection," *Proc. of the International Congress on Intellectual Property Rights for Specialized Information, Knowledge and New Technologies*, Vienna, Austria, August 1995.

List of Tables

List of Figures

114	105	93	81	61	34	36	36	53	70	106	130	150	152	153	157
113	130	127	109	88	74	54	45	36	32	41	67	112	138	145	145
72	118	131	117	90	97	63	63	56	50	47	51	71	104	138	138
26	67	116	80	61	66	68	88	68	76	65	69	68	91	116	126
16	30	47	24	20	25	52	87	65	92	85	81	82	89	102	116
21	25	26	25	16	11	14	28	85	106	92	91	89	93	104	111
16	38	67	57	28	18	14	13	27	75	102	97	90	92	100	107
28	70	151	150	88	36	13	12	14	42	88	99	89	90	99	101

Table 1: Pixel values of a small image of two 8 blocks. This sub-image is a part of Lenna's right eye and eyebrow

486	91	-66	-91	-17	-1	14	-0
140	41	44	35	-8	-12	-6	-4
43	108	-54	5	16	13	-9	-0
-143	-21	84	34	22	-0	-12	6
9	-18	-2	-32	8	5	5	12
-23	-9	1	-1	-8	1	2	-0
3	10	-14	4	6	-1	-1	-6
-8	-10	14	3	-1	-2	-2	-3

(a)

727	-188	-3	-28	-16	-4	-6	-1
51	-77	22	45	11	1	2	3
31	-52	-73	-8	5	5	10	7
73	40	-21	-7	1	-13	-2	-2
19	12	-21	-17	4	2	2	-1
20	15	-2	-17	-5	2	-0	-1
16	16	13	1	2	6	-2	0
-1	-3	-6	-12	-6	-1	1	3

(b)

Table 2: (a) The DCT coefficients $\mathbf{F_1}$ of the left 8×8 block in Table 1; (b) The DCT coefficients $\mathbf{F_2}$ of the right block.

480	96	-64	-96	-16	0	16	0
144	48	48	32	-16	-16	0	0
48	112	-48	0	16	16	-16	0
-144	-16	80	32	16	0	-16	0
16	-16	0	-32	16	0	0	16
-16	-16	0	0	-16	0	0	0
0	16	-16	0	0	0	0	0
-16	-16	16	0	0	0	0	0

(a)

720	-192	0	-32	-16	0	0	0
48	-80	16	48	16	0	0	0
32	-48	-80	-16	0	0	16	0
80	48	-16	0	0	-16	0	0
16	16	-16	-16	0	0	0	0
16	16	0	-16	0	0	0	0
16	16	16	0	0	0	0	0
0	0	0	-16	0	0	0	0

(b)

Table 3: (a) The quantized DCT coefficients $\tilde{\mathbf{F}}_\mathbf{1}$ of the first block in Table 2; (b) The quantized DCT coefficients $\tilde{\mathbf{F}}_\mathbf{2}$.

-208	320	-32	-32	16	16	32	0
112	144	48	0	-16	0	0	0
-16	128	0	0	0	0	-32	0
-256	-112	64	0	0	0	-32	0
0	-32	16	-16	16	0	0	16
-16	0	32	32	16	16	16	0
0	0	-16	16	16	0	0	0
-16	-16	16	16	0	0	0	0

Table 4: The differential DCT coefficients of the manipulated image

	Image	Mapping Function & Number of Bits in Sets	Manipulation	Rounding Noise
Signature Generator	fixed	selected	random	random
Authenticator	fixed	fixed	random	random
Attacker	fixed	random	fixed	random
System Evaluation	random	random/fixed	random/fixed	random

Table 5: Properties of different system variables with respect to different parties

Image	Replacement	Blur	Sharpen	Histogram Equalization
Lenna	25.8 – 55.0	8.7 – 10.7	9.43 – 12.7	23.1

Table 6: Some experimental values of the variance with different operations (by using Photoshop 3.0 software)

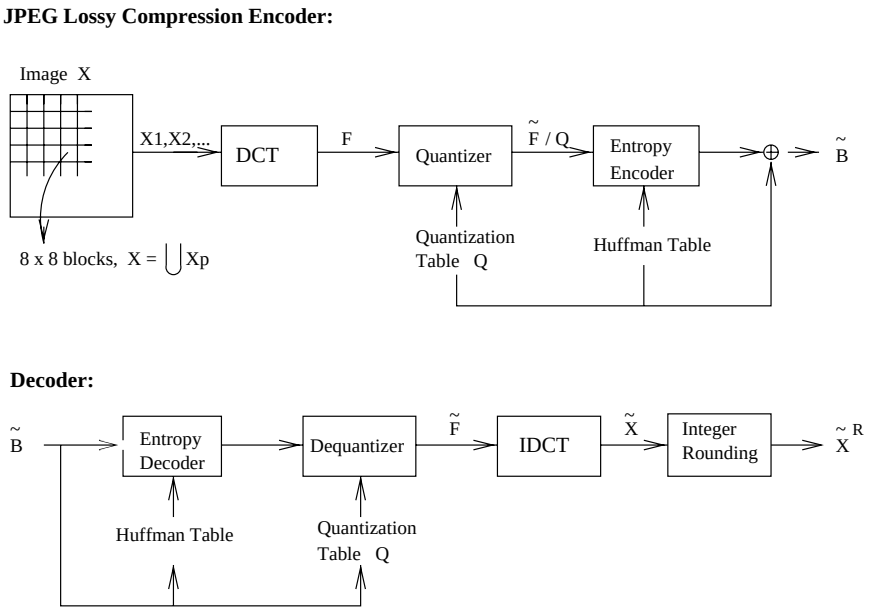

Figure 1: JPEG lossy compression

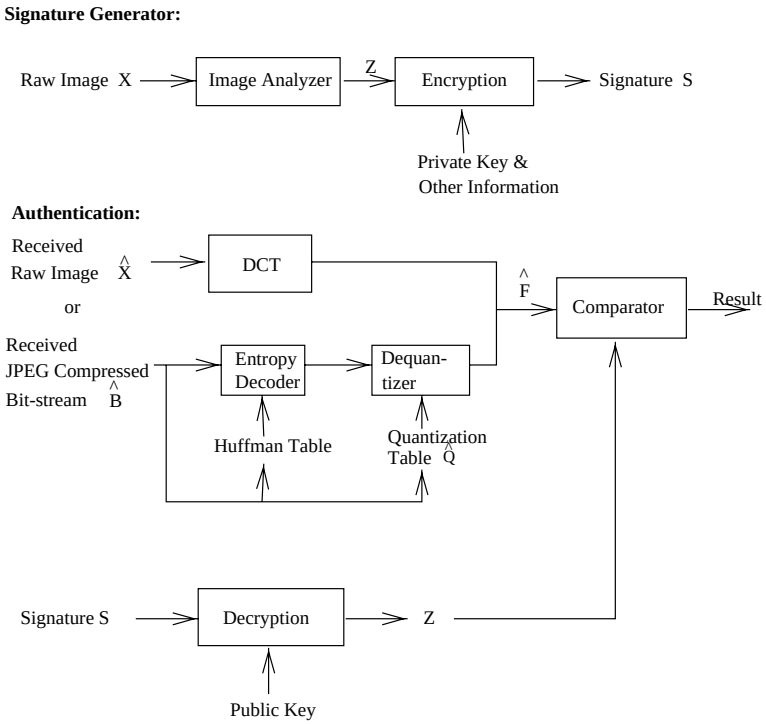

Figure 2: Signature Generator & Authentication Process

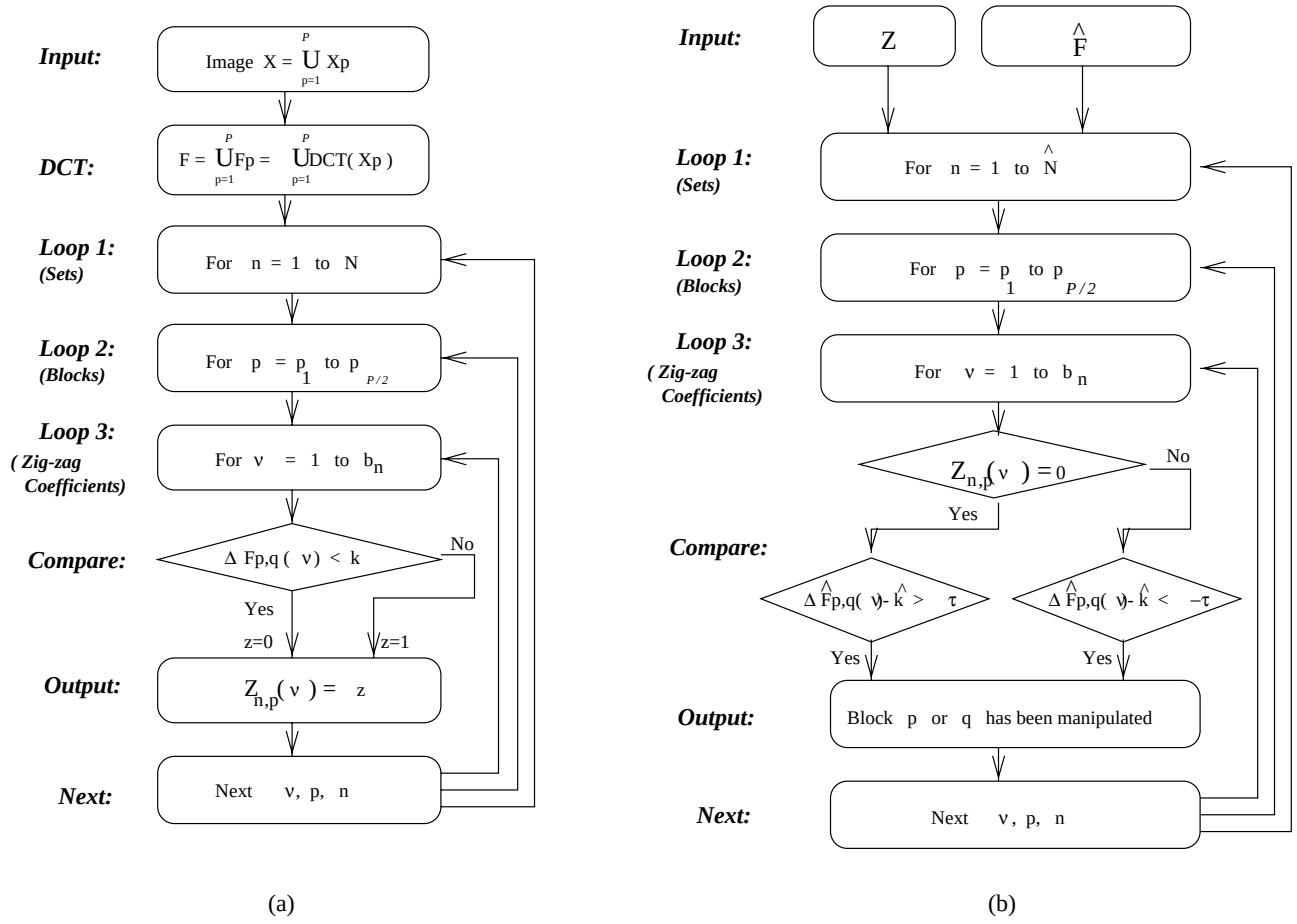

Figure 3: (a) Feature Extraction, (b) Authentication: Comparator

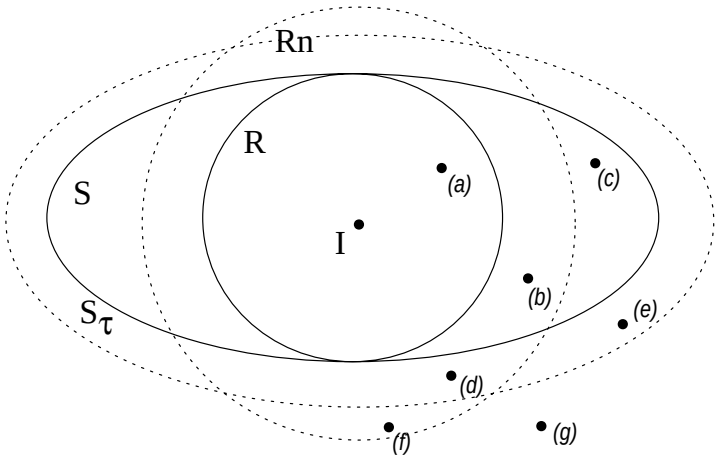

Figure 4: Conceptual illustration of 'miss', 'false alarm' and other scenarios. The definition of parameters: I, the original image; R, the range set of all JPEG operations on I, including multiple re-encoding processes; R_n, the range set of JPEG operations on I with rounding noise introduced; S, the accepted image set of the authenticator without tolerance values; S_τ, the accepted image set of the authenticator with tolerance τ. The points (a)-(g) represent: (a) Successful authentication of a JPEG compressed I; (b) Successful authentication of a rounded JPEG compressed I; (c) Miss; (d) False alarm by S but successful authentication by S_τ; (e) Successful detection of a manipulation by S but miss by S_τ; (f) False alarm; (g) Successful detection of a manipulation.

Figure 5: Original image : Lenna

Figure 6: JPEG compressed Lenna: compression ratio 9:1

Figure 7: Middle of hat brim cloned

Figure 8: Authentication result of Figure 7

Figure 9: Mouth manipulated

Figure 10: Authentication result of Figure 9

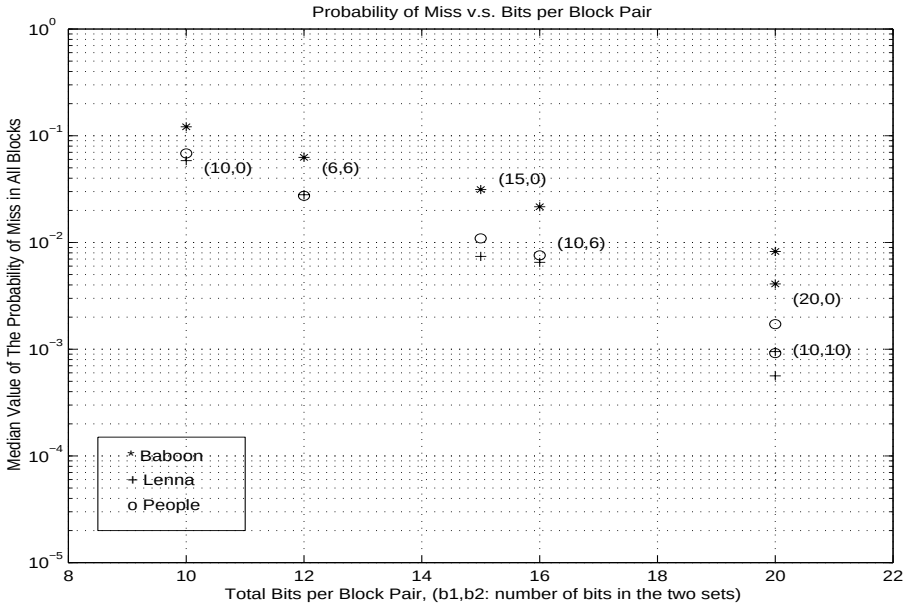

Figure 11: The Probability of Miss with different images

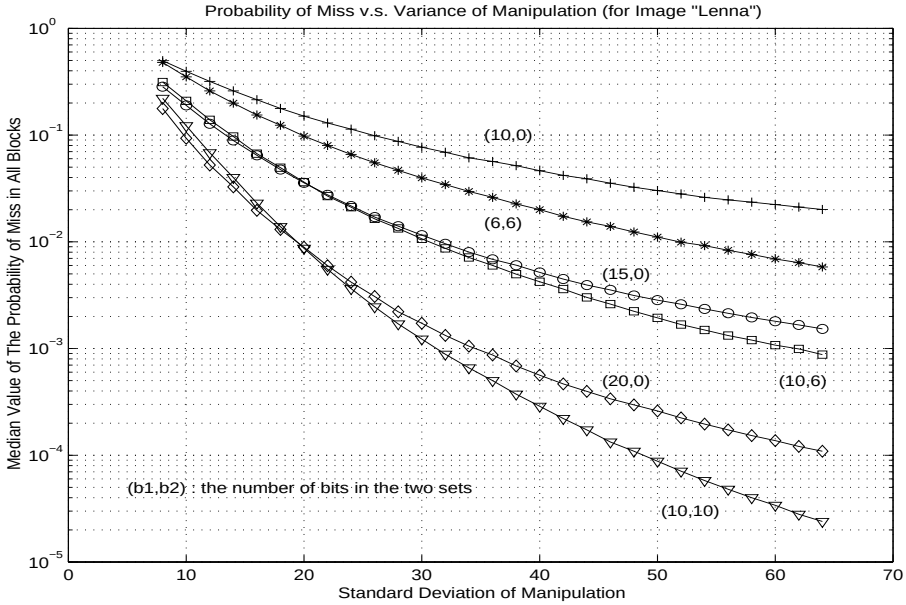

Figure 12: The Probability of Miss with different signature lengths

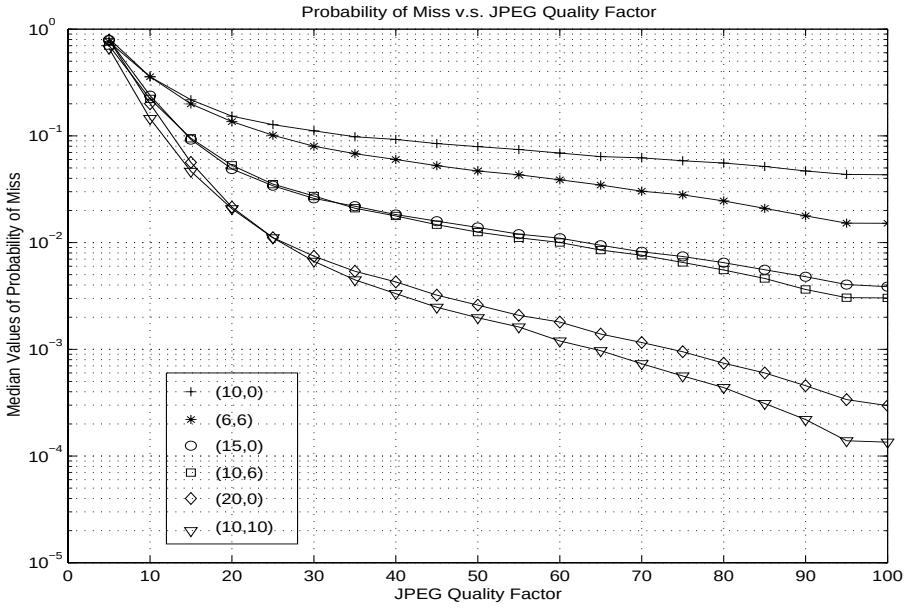

Figure 13: The Probability of Miss of images with different JPEG quality factors

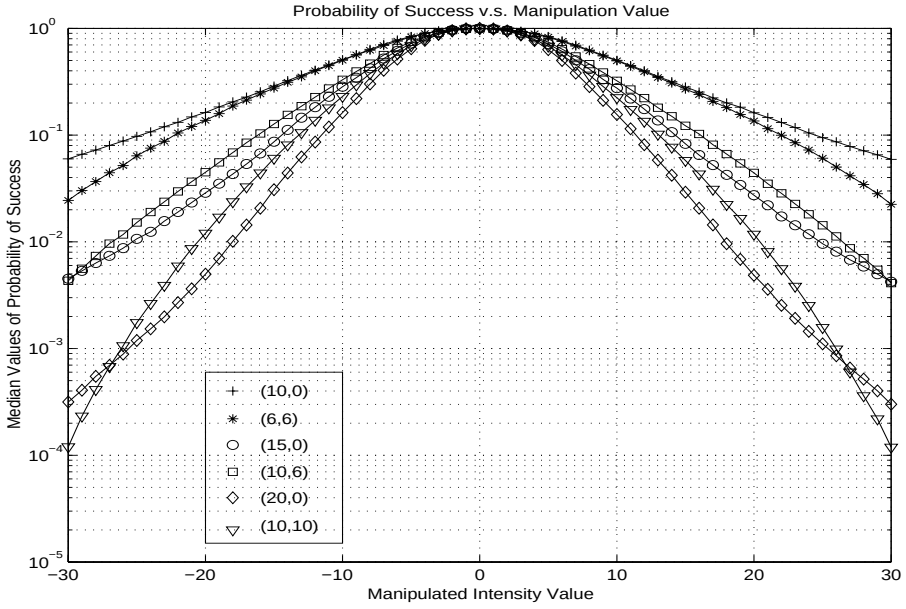

Figure 14: The Probability of Success with different thresholds