A Data Hiding Technique in JPEG Compressed Domain

Peter H. W. Wong*, Oscar C. Au**, Justy W. C. Wong***
Department of Electrical and Electronic Engineering
The Hong Kong University of Science and Technology
Email: eepeter@ust.hk*, eeau@ust.hk**, eejusty@ust.hk***

ABSTRACT

JPEG is a common image format in the world wide web. JPEG-compressed images can be used to hide data for secret internet communication and simply any auxiliary data. In this paper, we propose an algorithm called J-Mark to embed invisible watermark information into JPEG compressed images in the compress domain. There are three parts of J-Mark: block selection, DCT coefficient selection, and modification of selected DCT coefficients. Only the texture blocks with significant masking properties are selected in block selection. Only the DCT coefficients with significant energy in the selected blocks are selected. The watermark data are embedded as the "randomized parity" in the selected DCT coefficients. The embedded data can be recovered perfectly in the compressed domain without fully decoding the JPEG image. Experiment results suggest that the proposed J-Mark can hide the watermarking data without detectable visual artifacts. Although the data hiding capacity differs among images, some parameter of J-Mark can be used to achieve tradeoff between data hiding capacity and visual quality.

1. INTRODUCTION

In recent years, digital images can be captured easily with scanners, digital cameras and camcorders, and transmitted over the Internet. As a result, digital images appear widely in the internet and the world wide web (WWW) and in storage media such as CD-ROM and DVD. One of the most popular image formats used is JPEG, which can achieve high compression while retaining high image quality. Associated with the widespread circulation of images are issues of copyright infringement, authentication and privacy. One possible solution is to embed some invisible watermarks into the images.

Digital watermarking [1] is a process to embed some digital data called watermark into an image. While some watermarks are visible[2], most watermarks of interest are invisible. There are many classes of invisible watermarks for different applications such as fragile watermarks and robust watermarks. Fragile watermarks are designed to be broken easily by image processing operations. The broken watermark serves as an indication of alteration of the original image. Major applications include tampering detection of images placed on the WWW and authentication of images received from questionable sources. Robust watermarks are required to remain in the watermarked image even after it has been attacked. The attacks may be hostile attacks such as statistical averaging, signal processing, watermark estimation and removal, watermark counterfeiting, etc. The attacks may also be casual or unintended attacks which are common image processing such as filtering, compression, scaling, cropping, etc. Some methods include spread spectrum in the frequency domain [3] and the incorporation of perceptual models [4]. Major applications include ownership establishment, copyright and distribution control. Data hiding watermarks, also called steganography, are used to embed data in the images with the intention to have the data recovered perfectly at the receiver. Such methods usually assume that there are no hostile or even casual attacks. Error control coding is usually used to combat channel noise and casual signal processing. Major applications include secret communication over the internet and embedding of value-added auxiliary data with such low economic values that there is no motivation for hostile attack. In this paper, we are concern about data hiding watermarks.

There are many existing data hiding watermarks. Some inserts the invisible data into the least insignificant bits (LSB) of the uncompressed image [5]. Some inserts into LSB only around image contours [6]. Some hides small geometric patterns called tags in regions where the tags would be least visible [7], such as the very bright, very dark or texture regions. Some embeds watermark in the histogram [8]. Some chooses random pairs of image points and increase the brightness of one and decrease that of the other [9]. Some adds a small positive number to random locations as specified by the binary watermark pattern and use statistical hypothesis testing to detect the presence of watermark [10]. Some uses dynamic systems (toral automorphism) to generate chaotic orbits which are dense in the spatial domain and hide the watermark at the seemingly chaotic locations [11]. Some uses spread spectrum technique to hide data in the log-2 spatial domain [12]. In this paper, we are concerned about data hiding in the DCT domain in JPEG images.

While there are many robust watermarks in the DCT domain such as [3], there are relatively fewer existing data hiding watermarking techniques in DCT domain. Some embeds watermark bits as pseudorandom sequences in the frequency domain [13]. Some hides watermarks by removing or retaining selected DCT coefficients [14]. Some hides watermark in JPEG images by forcing selected DCT blocks to satisfy certain linear or circular constraint [15]. Some selects from some pre-defined pairs or triplets of DCT coefficients and uses their relative strength to encode a bit of the watermark

Security and Watermarking of Multimedia Contents III, Ping Wah Wong, Edward J. Delp III, Editors, Proceedings of SPIE Vol. 4314 (2001) © 2001 SPIE · 0277-786X/01/$15.00

pattern [16]. Some embeds watermark patterns in the quantization module after DCT [17] or in selected blocks based on human visual models [18]. Some utilizes inter-block correlation by forcing DCT coefficients of a block to be greater or smaller than the average of the neighboring blocks [19]. Some modifies selected DCT coefficients by random shuffling and table lookup embedding [20]. In [21], we proposed an algorithm called DC-Hide which hides watermark in the DC components of texture blocks in JPEG image.

In this paper, we propose a novel algorithm called J-Mark which is an improvement over the DC-Hide algorithm. J-Mark embeds watermark as randomized parity in selected significant DCT coefficients in selected texture blocks in JPEG images.

In this paper, the data to be hidden may be pre-processed, in any arbitrary order, by one or more of these invertible operations: re-ordering, padding, scrambling, encryption, transformation, source coding or compression, error correction/control coding, etc before being embedded into the JPEG images. In all cases, when the embedded data is read out or recovered, the inverse of each of the operations applied during the pre-processing is applied in reverse order to recover the original data.

2. HIDING DATA IN JPEG IMAGES

In this section, we propose an algorithm called J-Mark to hide data in JPEG images with negligible visual degradation. J-Mark can be applied as shown in Figures 1 and 2. In Figure 1, J-Mark embeds hidden watermark into a JPEG compressed image to generate another JPEG image with the hidden data. Three steps are involved: block selection, DCT coefficient selection and the modification of the selected DCT coefficients in the selected blocks. Alternatively, J-Mark can operate on an uncompressed image by compressing it with JPEG with the target quality factor and quantization table and applying the processing in Figure 1 to the resulting JPEG image. We assume that the output JPEG image from J-Mark would not be transcoded or processed further before the hidden data are extracted. In other words, we assume that there are no hostile attacks or casual signal processing on the JPEG compressed image. Under our assumption, the hidden data can be extracted perfectly. The watermark extraction is shown in Figure 2 which involves block selection, DCT coefficient selection and watermark retrieval from the selected DCT coefficients in the selected blocks.

In JPEG compression, an image X is divided into 8x8 disjoint blocks to which discrete cosine transform (DCT), quantization and variable length coding are applied. Consider an 8x8 block in the image. Let $\{x(i,j),\ 0 \le i \le 7,\ 0 \le j \le 7\}$ be the pixels within the block. Let $\{y(i,j),\ 0 \le i \le 7,\ 0 \le j \le 7\}$ be the corresponding DCT coefficients. The quantized DCT coefficients $y_q(i,j)$ are

$$y_q(i,j) = round\left(\frac{y(i,j)}{round\left(Q(i,j)\cdot Q_p\right)}\right)$$

where $Q(i,j)$ is the ij^{th} element of the quantization table and Q_p is an additional scaling factor used to control the bit rate or the image quality. The same quantization table and Q_p are used to encode all the 8x8 blocks within an image. The quantization table can be defined by the user, though JPEG has a default quantization table as shown in Table 1.

J-Mark hides data by modifying the $y_q(i,j)$ at selected low frequency (i,j) in selected blocks. The high frequency DCT coefficients cannot be used because such coefficients tend to be small for natural images while the corresponding entries in the quantization matrix tend to be large due to the insensitivity of the human eye to high frequency details. Consequently, the high frequency $y_q(i,j)$ tend to be zero after quantization, especially when the bit rate is low and the Q_p is large. The selected blocks and the selected low frequency $y_q(i,j)$ need to be identified consistently during encoding and decoding so that the hidden data can be extracted correctly at the receiver.

2.1 Block Selection for Data Hiding

J-Mark may hide 1, 2, 3 or more bits in an 8x8 block. However, not all blocks will be used. From our experiments, we observe that even a slight change in the quantized DC or the low frequency DCT coefficients can cause visual artifacts in the smooth regions, such as the shoulder region of Lena. However, in texture rich regions such as the hair and hat of Lena, a small change in such quantized DCT coefficients can be perceptually undetectable due to the masking effect of the textures. As a result, only the texture-rich blocks are used in the proposed J-Mark to embed the watermark information. This gives rise to two problems. One problem is the way to identify the texture-rich blocks during the data hiding process. The second problem is the way to ensure that the texture-rich blocks can be identified during the data extraction process.

There are many ways to identify texture-rich blocks in other applications such as rate control, etc. Typically, they classify blocks according to some kind of activity measures such as variance[22], texture energy[23], bandpass energy[21], total variation[24], etc. When the activity measure of a block is larger than some threshold, the block is classified as a texture block, and vice versa. Spatial measures such as variance and total variation can be applied on the original image during the compression. However, the original image is not available at the decoder in our JPEG data hiding application and thus there is a possibility of choosing the wrong blocks for watermark retrieval at the decoder using spatial measures.

However, spatial measures can be consistent if they are applied on the reconstructed images during encoding and the data hiding procedure does not change the measures such that the same values are obtained at the decoder. In DC-Hide[21], we used the bandpass energy of the reconstructed image to select the blocks consistently. We applied a bandpass filter and examined the resulting energy within the block to identify the texture-rich blocks. As DC-Hide would only change the DC coefficients, the bandpass energy is preserved in the data hiding and thus the block selection can be performed consistently. Although bandpass energy of the reconstructed image are effective in the block selection, the computational requirement to obtain the reconstructed image at the encoder is high which is undesirable.

In J-Mark, we use another consistent way to select the texture blocks. We use the AC energy of selected quantized DCT coefficients to select the texture blocks. To avoid selecting smooth regions and the edge blocks, we compute the energy E_{AC} of only some of the reconstructed DCT coefficients,

$$E_{AC} = \sum_{(i,j)\in \mathrm{N}} \left(y_q(i,j) round\left[Q(i,j)Q_p\right]\right)^2$$

where N is the shaded region in Figure 3. The region N does not include the DC coefficient such that blocks with different brightness are treated equally. The outer layer of DCT coefficients are not included so as to avoid extracting edge blocks which tend to have lots of energy in these coefficients.

Blocks with E_{AC} larger than a threshold T_1 are declared texture blocks. The threshold T_1 can be a function of the additional quantization factor Q_p. However, false detection of texture blocks are possible. As small texture regions tend not to have good masking properties, J-Mark seeks to avoid such small texture regions. Thus we require a texture block to have at least N_1 texture block neighbors in the 3x3 neighborhood in order to be chosen for data hiding. When $N_1 = 0$, all texture blocks are chosen. Two texture block examples are shown in Figure 4. If $N_1 = 7$, the center texture block in case (a) of Figure 4 will not be selected, but the center block in case (b) will be selected. We typically choose N_1 to be 6 to suppress false detection and avoid small isolated texture blocks and texture boundary blocks.

As will be explained in the next section, some DCT coefficients shown in Figure 5 are candidates for watermarking and may be altered. Although this may change some of the AC coefficients in N of the chosen texture blocks, the AC coefficients are changed in such a way that the E_{AC} of the chosen textures blocks at the decoder remain larger than the threshold T_1 and would continue to be classified as texture blocks at the encoder. And there are no changes to the non-chosen blocks and their E_{AC} such that there is no change to their texture block classification. In other words, the texture block classification remain the same at both the encoder and the decoder. Consequently, the same blocks would be chosen at the decoder and the block selection is consistent.

2.2 DCT Coefficient Selection

Zero, one or more bits are embedded in the chosen blocks. As most high frequency coefficients tend to be zero after quantization especially when the bit rate is low, only the low frequency DCT coefficients shown in Figure 5 are considered as candidates for data hiding. The amount of embedded bits within a chosen block depend on the energy profile of the candidates. Any candidate DCT coefficient whose absolute magnitude is less than a threshold T_2 is eliminated. We choose at most N_2 out of the remaining candidates because too many altered DCT coefficients can result in visually detectable artifacts.

In rare cases, all the candidates will be eliminated. When N_2 or fewer candidates remain, they are selected for watermarking. When there are more than N_2 remaining candidates, they are scanned in a zig-zag manner and the first N_2 candidates are chosen. An example is shown in Figure 6 with $N_2 = 3$. Part (a) of Figure 6 shows four remaining candidates after the energy thresholding and the first $N_2 = 3$ encountered during the zig-zag scanning are chosen as shown in part (b).

2.3 Modification of Selected DCT Coefficients in Selected Blocks

J-Mark uses 'randomized parity' to embed one bit of watermark data into one selected quantized DCT coefficient of a selected 8x8 block. As mentioned before, the watermark data may be preprocessed. Suppose the k^{th} watermark data bit w_k is to be hidden at the DCT coefficient at location (i,j) of a selected block. The 'randomized parity' is defined as

$$\mathrm{mod}\left[round\left(\frac{y_q(i,j)}{n_k}\right),2\right]$$

which is the parity of the rounded quotient of the quantized DCT coefficient $y_q(i,j)$ divided by a pseudo-random number n_k. When the 'randomized parity' and the desired data bit w_k are equal, no change to $y_q(i,j)$ is needed. When they are different, $y_q(i,j)$ is altered in such a way that the resulting 'randomized parity' is the same as w_k.

The pseudo-random number n_k is uniformly distributed between two positive real numbers R_1 and R_2, generated with a user-defined key. The purpose of the 'randomized parity' is to implement user access control. The key is needed for the watermark extraction. If $R_1 = R_2 = 1$, the 'randomized parity' become the parity of the quantized DCT coefficients. According to our experiments, a choice of $R_1 = 1$ and $R_2 = 2$ can give good visual quality. Note that the randomized parity is similar to the table lookup in [20]. However, the general table lookup can have different non-uniform lookup table for each coefficients which might require excessive memory. Our randomized parity is like a uniform lookup table with different cell size for each selected DCT coefficient.

The watermarked DCT coefficient $y_q^w(i,j)$ is computed as follows:

$$y_q^w(i,j) = \begin{cases} \left\lceil n_k\left(round\left(\frac{y_q(i,j)}{n_k}\right)+\beta(i,j)\right)\right\rceil & \mathrm{mod}\left[round\left(\frac{y_q(i,j)}{n_k}\right),2\right] \neq w_k \\ y_q(i,j) & \mathrm{mod}\left[round\left(\frac{y_q(i,j)}{n_k}\right),2\right] = w_k \end{cases}$$

where $\beta(i,j)$ is either 0.5 or –0.5 and $\lceil x \rceil$ is the ceiling function. If x is non-negative, $\lceil x \rceil$ is the smallest integer greater than or equal to x. If x is negative, $\lceil x \rceil$ is the largest integer less than or equal to x. Note that the choice of $\beta(i,j)$ has an impact on the E_{AC} which can affect the consistency of the block selection between the encoder and decoder as mentioned before. The choice of $\beta(i,j)$ also has an impact on the absolute magnitude of the quantized DCT coefficients that can affect the consistency of the DCT coefficient selection.

We propose two ways to choose β. The first way is a simple way which we will call β_1:

$$\beta(i,j) = \begin{cases} 0.5, & y_q(i,j) \geq 0 \\ -0.5, & y_q(i,j) < 0 \end{cases} \qquad (2)$$

such that $\left|y_q^w(i,j)\right| \geq \left|y_q(i,j)\right|$. With this choice, the E_{AC} of the selected block at the decoder is greater or equal to the E_{AC} at the encoder and the block selection would be consistent. The absolute magnitude of the selected DCT coefficients at the decoder is also greater than or equal to that at the encoder such that the DCT coefficient selection remains consistent. With this choice, the worse case distortion to the $(i,j)^{th}$ DCT coefficient caused by the watermarking is about $n_k Q(i,j) Q_p$. This is one of the reasons for our choice of the candidate DCT coefficients shown in Figure 5. The $Q(i,j)$ associated with the candidate locations in Figure 5 are small, upper bounded by 16 in the default JPEG quantization table such that the worst case distortion is not excessive.

To minimize the distortion caused by the watermarking, the $\beta(i,j)$ can be chosen as

$$\beta(i,j) = \begin{cases} 0.5, & y_q(i,j) \geq n_k \cdot round\left(\frac{y_q(i,j)}{n_k}\right) \\ -0.5, & otherwise \end{cases} \qquad (3)$$

such that the worse case distortion to the DCT coefficient is reduced to $n_k Q(i,j)Q_p/2$. However, this may reduce the E_{AC} leading to a possible misclassification at the decoder. The absolute magnitude of the modified DCT coefficients may be below T_2 leading to incorrect DCT coefficient selection at the decoder. As a result, the second way to choose $\beta(i,j)$ which we will call β_2 is to apply Eqn. 3 and then check the resulting E_{AC} and absolute amplitude of the coefficients. If the resulting E_{AC} is larger than T_1 and the resulting absolute magnitude continues to be greater than or equal to T_2, Eqn. 3 is used. Otherwise, Eqn. 2 is used instead.

The two ways to choose $\beta(i,j)$ provide a tradeoff between complexity and distortion. The β_2 has higher computational requirement than β_1, but the distortion of β_2 is minimized which makes the computation worthwhile.

2.4 Watermark Retrieval

To retrieve the watermark at the decoder, the watermarked JPEG image is processed as shown in Figure 2. It is assumed that no transcoding or other processing is performed on the watermarked image. The block selection is the same as that described in section 2.1. In other words, the E_{AC} is computed for every 8x8 block. Those blocks with $E_{AC} > T_1$ are declared texture blocks. The textures blocks with at least N_1 neighboring texture blocks are selected. By design, the selected texture blocks at the decoder are identical to those selected at the encoder. The DCT coefficient selection is also the same as that described in section 2.2. In the selected block, N_2 or fewer candidate DCT coefficients with energy greater than or equal to T_2 are selected.

To decode the watermark, the user-defined key is used to generate the pseudo-random numbers n_k. The watermark w_k is decoded as

$$w_k = \mathrm{mod}\left[round\left(\frac{y_q(i,j)}{n_k} \right), 2 \right]$$

from the selected DCT coefficients $y_q(i,j)$. If preprocessing was applied before the data hiding watermarking, the inverse operation is applied to recover the original data.

3. SIMULATION RESULTS AND DISCUSSIONS

The proposed J-Mark algorithm is simulated on two 512x512 gray scale images, 'Lena' and 'Pepper'. The images were compressed by JPEG using different Q_p to achieve different visual qualities. The visual quality measure used is the peak signal-to-noise ratio, $PSNR = 10\log_{10}\left(255^2/MSE\right)$, where *MSE* is the mean square error between the original image and the reconstructed image.

Two experiments are performed. In the first experiment, the block selection threshold T_1 increases linearly with Q_p and either β_1 or β_2 are used. The results are shown in Table 2. In the second experiment, T_1 is fixed and either β_1 or β_2 are used. The results are shown in Table 3.

For each experiment, six values of the additional quantization factor Q_p are used, namely 0.5, 1.0. 1.5, 2.0, 2.5 and 3.0. The block selection threshold N_1 is chosen to be 6. The DCT candidate selection threshold N_2 is chosen to be 3. The fixed T_1 in the second and fourth experiment is chosen to be 800. The T_1 is chosen to increase from 500 at $Q_p = 0.5$ to 5500 at $Q_p = 3.0$ in the first and third experiment to reflect possible performance in extreme conditions. Without loss of generality, the watermark information used is a random bit sequence. Note that the number of blocks selected to embed the watermark bit sequence depends on the image and the compression ratio. The parameters R_1 and R_2 for the generation of pseudo-random bit sequence (n_k) are chosen to be 1 and 2 respectively.

For Lena at $Q_p = 0.5$, fewer blocks are selected when T_1 is increased from 500 to 800. The chosen blocks are located mainly at the hat, hair and the eyes. When Q_p increased from 0.5 to 3.0 with T_1 fixed at 800, more high frequency

DCT coefficients are quantized to zero resulting in fewer blocks with their E_{AC} greater than $T_1 = 800$ and thus fewer blocks are selected. And significantly fewer blocks are selected if T_1 is increased from 800 to 5500. Similar observations are found in the other test images. Note in particular that very few blocks are selected in Pepper at $Q_p = 3.0$ and $T_1 = 5500$ because Pepper contains mostly smooth regions.

When β_2 is used in Tables 2 and 3, the amount of embedded bits are exactly the same as β_1. The PSNR of β_2 is larger than that of β_1. The bit rate of β_2 is less than or equal to that of β_1. These simply verify that the β_2 has absolutely better performance in terms of PSNR and bit rate than β_1. The only disadvantage of β_2 is the higher computational complexity than β_1. For this reason, only the images associated with β_2 are shown in Figures 7 to 12.

Consider Table 2 in which variable T_1 and the β_2 are used. When $Q_p = 0.5$ and $T_1 = 500$, the original JPEG compressed Lena has a PSNR of 37.83dB at 0.975 bit per pixel (or bpp) and the proposed J-Mark with β_2 can embed 2465 data bits in Lena with only a small PSNR drop of 0.06dB and the same bpp. When Q_p and T_1 increase, the number of embedded data bits of J-Mark decreases and the loss in PSNR increases. When $Q_p = 3.0$ and $T_1 = 5500$, J-Mark can hide 522 data bits with a PSNR loss of 0.25dB. Considering that very few non-zero quantized DCT coefficients are left at $Q_p = 3.0$, this amount of perfectly recoverable hidden data bits is good. Effectively, J-Mark does not change the bit rate of the JPEG images.

Among the two test images, Pepper contains the fewest texture blocks and thus the fewest embedded data bits. In Pepper, J-Mark can embed 1576 bits at $Q_p = 0.5$ and 84 bits At $Q_p = 3.0$, which are significantly lower than the corresponding numbers in Lena. This is probably because the Pepper image contains very few texture blocks. However, the PSNR loss in Pepper is much smaller than in Lena, being less than or equal to 0.06dB in all cases.

In Table 3, the T_1 is fixed at 800 which will be contrasted with the variable T_1 in Table 2. When $Q_p = 0.5$, J-Mark with β_2 can hide 2034 bits in Lena in Table 3, which is fewer than the 2465 bits at $T_1 = 500$ in Table 2. The corresponding PSNR loss of 0.04dB in Table 3 is lower than the 0.06dB in Table 2. For Q_p ranging from 1.0 to 3.0, the T_1 in Table 3 is smaller than that in Table 2 such that more data bits can be hidden in Table 3 than in Table 2. The tradeoff is that when T_1 is smaller in Table 3, the PSNR loss due to J-Mark is also larger in Table 3 than in Table 2 The PSNR loss in Lena is 0.64dB at $Q_p = 3.0$ to embed 1364 bits in Table 3 compared with the 0.25dB to hide 522 bits in Table 2. Similar observations can be made for the other three test images.

The Lena shown in Figures 7, 8 and 9 have effectively the same quality, which shows that the J-Mark does not introduce any visually detectable degradation at $Q_p = 0.5$. The Lena shown in Figures 10 and 11 have effectively the same visual quality, while minor blocking artifacts can be observed in the texture area in Figure 12. This shows that the choice of $T_1 = 5500$ does not introduce any visually detectable degradation at $Q_p = 3.0$. The degradation in Figure 14 would be the price to pay to increase data hiding capacity from 522 bits to 1364 bits using $T_1 = 800$. In other words, the parameter T_1 provides means to achieve different data hiding capacity and visual quality trade-off. Similarly, minor degradation in Pepper is observed at $Q_p = 3.0$ and $T_1 = 5500$ in Figures 18.

In general, J-Mark tends to embed fewer bits at larger Q_p, probably due to increasing number of DCT coefficients being quantized to zero, and vice versa. In addition, J-Mark tends to incur larger loss in PSNR at larger Q_p. For any Q_p, a larger T_1 tends to result in fewer embedded data bits and smaller PSNR loss and vice versa. Consequently, it should be possible to choose the T_1 as a function of Q_p to achieve constant embedded data rate or constant PSNR loss or other desirable profiles. However, such a function would most likely be different for different images. The bit rate change due to J-Mark tends to be negligibly small.

4. CONCLUSION

In this paper, we proposed an algorithm called J-Mark to embed watermarking data into a JPEG image. The J-

Mark algorithm involves three steps: block selection, DCT coefficient selection and DCT coefficient modification. The embedded data can be recovered perfectly. Experiment results show that the proposed J-Mark can embed the data bits without causing detectable degradation. The embedded data capacity can change from one image to another. The parameter T_1 can be used to achieve various embedded data capacity and visual quality tradeoff.

REFERENCE

1. F. Mintzer, et al., "Effective and Ineffective Digital Watermarks", *Proc. of IEEE Int. Conf. on Image Processing*, Vol. 3, pp. 9-13, Oct. 1997.
2. G.W. Braudaway, et al., "Protecting Publicly Available Images with a Visible Image Watermark.", *Proc. of SPIE Conf. On Optical Security and Counterfeit Deterrence Techniques*, Vol.2659, pp.126-33, Feb. 1996.
3. I. J. Cox, et al., "Secure Spread Spectrum Watermarking for Multimedia", *IEEE Trans. of Image Processing*, Vol. 6, No. 12, pp. 1673-1687, Dec 1997.
4. O.H. Kwon, et al., "Watermarking for Still Images Using Human Visual System in the DCT Domain," *Proc. of IEEE Int. Sym. of Circuits and Systems*, Vol. 4, pp. 76-79, Jun. 1999.
5. R.G. van Schyndel, et al., "A Digital Watermark", *Proc. of IEEE Int. Conf. On Image Processing*, Vol. 2, pp. 86-90, Nov. 1994.
6. B.M. Macq, J.J. Quisquater, "Cryptology for Digital TV Broadcasting", *Proc. of the IEEE*, Vol. 83, No. 6, pp. 944-957, Jun 1995.
7. G. Caronni, "Assuring Ownership Rights for Digital Images", *Proc. of Reliable IT Systems*, 1995.
8. D. Coltuc, P. Bolon, "Watermarking by Histogram Specification", *Proc. of SPIE Conf. on Security and Watermarking of Multimedia Contents*, Vol. 3657, pp. 252-263, Jan. 1999.
9. W. Bender, et al., "Techniques for Data Hiding", *Proc. of SPIE Conf. on Storage and Retrieval for Image and Video*, Vol. 2420, pp. 40, Feb. 1995.
10. N. Nikolaidis, I. Pitas, "Copyright Protection of Images using Robust Digital Signatures", *Proc. of, IEEE Int. Conf. on Acoustics, Speech, Signal Processing*, Vol. 4, pp. 2168-2171, May 96.
11. G. Voyatzis, I. Pitas, "Applications of Toral Automorphisms in Image Watermarking", *Proc. of IEEE Int. Conf. On Image Processing*, Vol. 2, pp. 237-240, Sept. 96.
12. P.H.W. Wong, O.C. Au, et al., "Image Watermarking Using Spread Spectrum Technique in Log-2-Spatio Domain", *Proc. of IEEE Int. Sym. on Circuits & Systems*, Jun. 2000
13. W.G. Kim, et al., "An Image Watermarking Scheme with Hidden Signatures," *Proc. of IEEE Int. Conf. On Image Processing, Vol. 2, pp. 205-210, Oct. 99.*
14. G.C. Langelaar, et al., "Watermarking by DCT Coefficient Removal: A Statistical Approach to Optimal Parameter Settings," *Proc. of SPIE Conf. of Security and Watermarking of Multimedia Contents*, pp. 2-13, Jan. 1999.
15. A. G. Borg, I. Pitas, "Image Watermarking using DCT Domain Constraints", *Proc. of IEEE Int. Conf. on Image Processing*, Vol. 3, pp. 231-234, Sept. 96.
16. E. Koch, et al., "Copyright Protection for Multimedia Data", *Proc. of Int Conf. on Digital Media and Electronic Publishing*, 1994.
17. K. Tanaka, et al., "Embedding Secret Information into a Dithered Multilevel Image", *Proc. of IEEE Military Communication Conf.*, pp. 216-220, 1990.
18. M.D. Swanson, et. al., "Multiresolution Scene-based Video Watermarking using Perceptual Models", *IEEE J. on Selected Areas in Communications*, Vol. 16, No. 4, pp. 540-550, May 1998.
19. Y. Choi, K. Aizawa, "Digital Watermarking Using Inter-Block Correlation," *Proc. IEEE Int. Conf. on Image Processing, Vol.2, pp.216-220, Oct. 1999.*
20. M. Wu and B. Liu, "Digital Watermarking using Shuffling," *Proc. of IEEE Int. Conf. on Image Processing,* Vol. 1, pp.291-295, Oct. 1999.
21. P.H.W. Wong, et al., "Data Hiding and Watermarking in JPEG Compressed Domain by DC Coefficient Modification," *Proc. of SPIE Conf.. of Security and Watermarking of Multimedia Contents*, Jan 2000.
22. A. Puri, R. Aravind, "Motion Compensated Video Coding with Adaptive Perceptual Quantization", *IEEE Trans. on Circuits & Systems for Video Technology*, Vol.1, no.4, pp.351-61, Dec. 1991.
23. Soon Hie Tan, et. al., ,"Classified Perceptual Coding with Adaptive Quantization", *IEEE Trans. on Circuits & Systems for Video Technology*, Vol.6, no.4, pp.375-388, Aug. 1996.
24. C.H. Choi, O.C. Au, "Adaptive Image Quantization using Total Variation Classification", *Proc. of Int. Conf. on Signal Processing*, Vol. 2, pp. 796-799, Oct. 1993.

16	11	10	16	24	40	51	64
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Table 1 Default quantization table of JPEG

	Q_p	T_1	JPEG only		J-Mark with β_1			J-Mark with β_2	
			PSNR	bpp	PSNR	bpp	data bit hidden	PSNR	bpp
Lena 512x512	0.5	500	37.83	0.975	37.70	0.977	2465	37.77	0.975
	1.0	1500	35.81	0.625	35.58	0.626	1460	35.70	0.625
	1.5	2500	34.59	0.481	34.31	0.481	1023	34.43	0.481
	2.0	3500	33.70	0.403	33.39	0.404	824	33.49	0.403
	2.5	4500	32.96	0.352	32.60	0.353	654	32.71	0.352
	3.0	5500	32.33	0.316	31.96	0.317	522	32.08	0.317
Pepper 512x512	0.5	500	36.27	1.012	36.22	1.012	1576	36.25	1.012
	1.0	1500	34.75	0.636	34.69	0.634	529	34.72	0.634
	1.5	2500	33.78	0.479	33.68	0.478	268	33.74	0.478
	2.0	3500	33.05	0.402	32.97	0.402	178	33.00	0.402
	2.5	4500	32.43	0.349	32.34	0.350	137	32.37	0.349
	3.0	5500	31.87	0.314	31.79	0.314	84	31.83	0.314

Table 2 Simulation results of J-Mark with variable T_1 using β_1 or β_2

	Qp	T1	JPEG only		J-Mark with β_1			J-Mark with β_2	
			PSNR	bpp	PSNR	bpp	data bit hidden	PSNR	bpp
Lena 512x512	0.5	800	37.83	0.975	37.72	0.976	2034	37.79	0.976
	1	800	35.81	0.625	35.50	0.626	1938	35.64	0.625
	1.5	800	34.59	0.481	34.12	0.482	1667	34.31	0.481
	2	800	33.70	0.403	33.06	0.405	1582	33.27	0.404
	2.5	800	32.96	0.352	32.05	0.353	1653	32.34	0.352
	3	800	32.33	0.316	31.42	0.318	1364	31.69	0.317
Pepper 512x512	0.5	800	36.27	1.012	36.25	1.012	953	36.27	1.012
	1	800	34.75	0.636	34.66	0.636	816	34.70	0.636
	1.5	800	33.78	0.479	33.59	0.480	781	33.67	0.479
	2	800	33.05	0.402	32.85	0.402	557	32.93	0.402
	2.5	800	32.43	0.349	32.11	0.350	597	32.22	0.349
	3	800	31.87	0.314	31.57	0.314	456	31.64	0.314

Table 3 Simulation results of J-Mark with fixed T_1 using β_1 or β_2

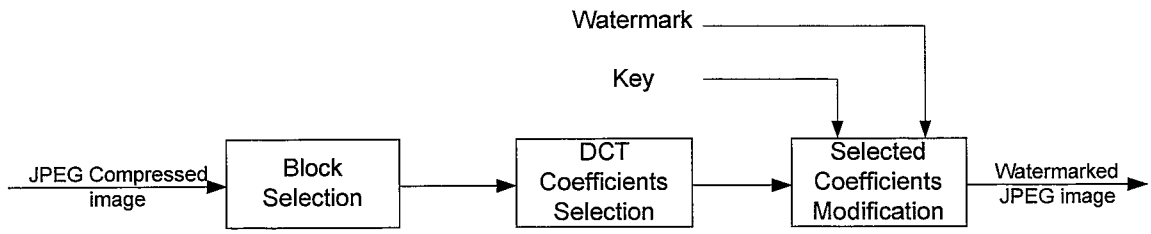


Figure 1 Embedding watermark in a JPEG-compressed image

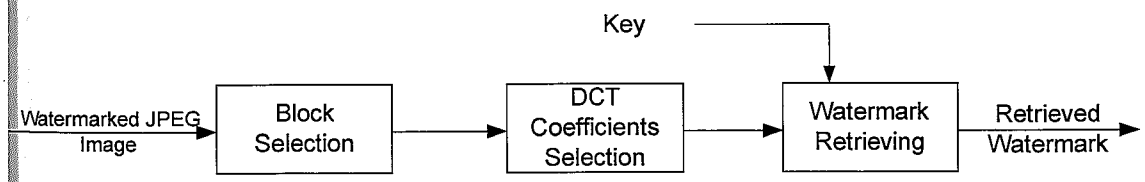


Figure 2 Extracting watermark from the watermarked JPEG image

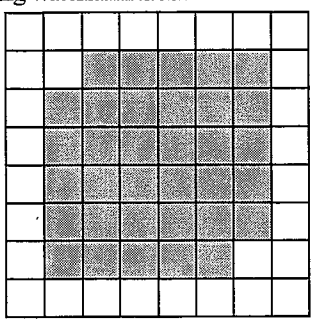


Figure 3 DCT coefficients used to compute E_{AC} (top left = DC coefficient)

(a) (b)

Figure 4 Two texture block examples: 6 texture neighbors in (a), 8 in (b)

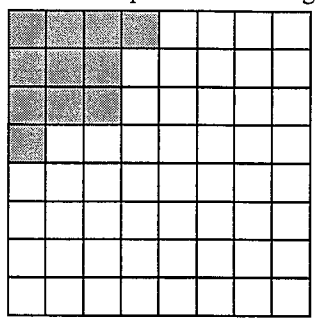
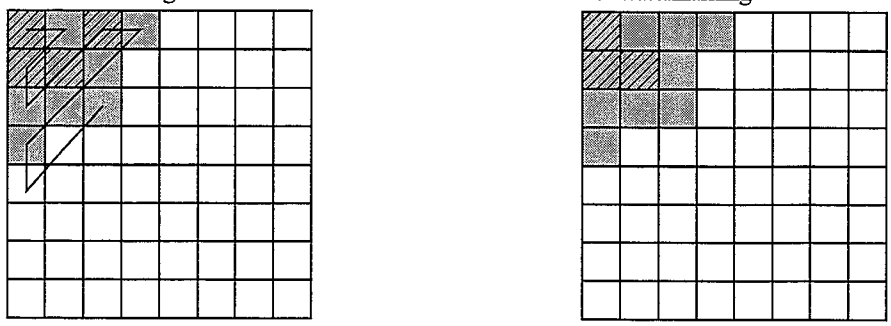


Figure 5 DCT coefficients that are candidates for watermarking



(a) 4 remaining candidates (b) 3 selected coefficient

Figure 6 Example of zig-zag scanning of candidates in (a) to select 3 candidates in (b)

Figure 7 JPEG compressed Lena (Q_p=0.5)

Figure 8 J-Mark (Q_p=0.5, variable T_1=500, β_2)

Figure 9 J-Mark (Q_p =0.5, fixed T_1=800, β_2)

Figure 10 JPEG compressed Lena (Q_p=3.0)

Figure 11 J-Mark (Q_p =3.0, variable T_1, β_2)

Figure 12 J-Mark (Q_p =3.0, fixed T_1=800, β_2)

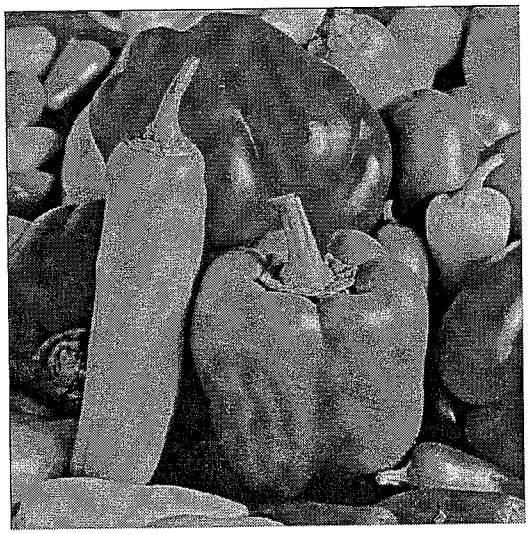


Figure 13 JPEG compressed Pepper (Q_p=0.5)

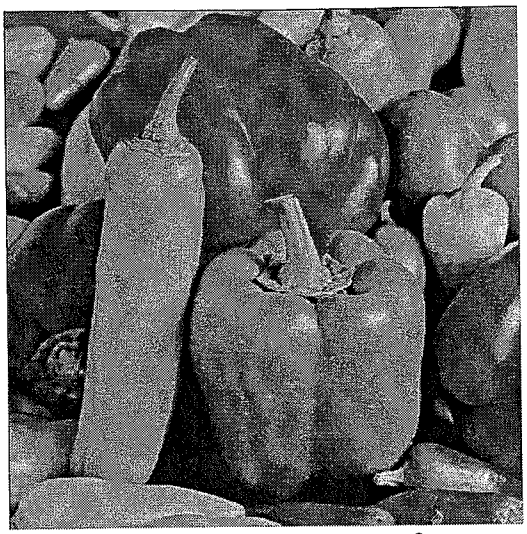


Figure 14 J-Mark (Q_p=0.5, variable T_1=500, β_2)

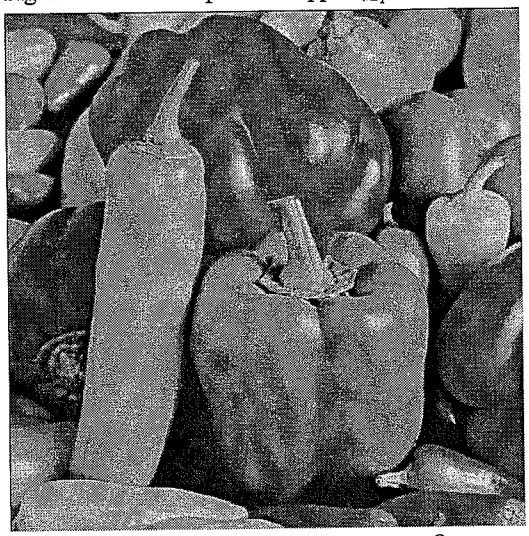


Figure 15 J-Mark (Q_p =0.5, fixed T_1=800, β_2)

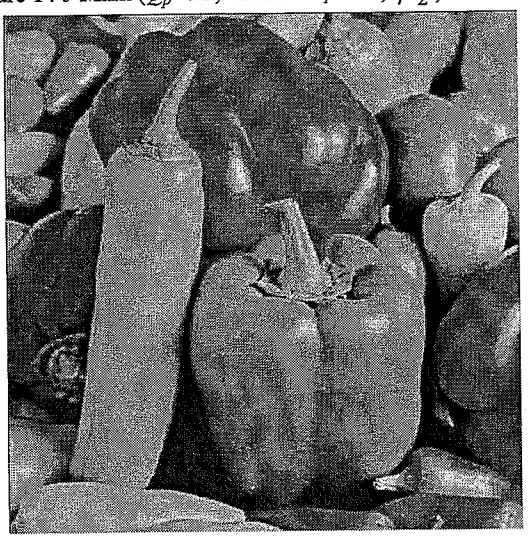


Figure 16 JPEG compressed Pepper (Q_p =3.0)

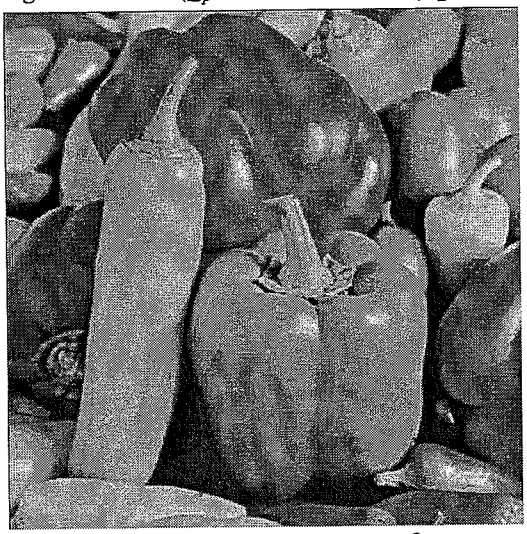


Figure 17 J-Mark (Q_p =3.0, variable T_1, β_2)

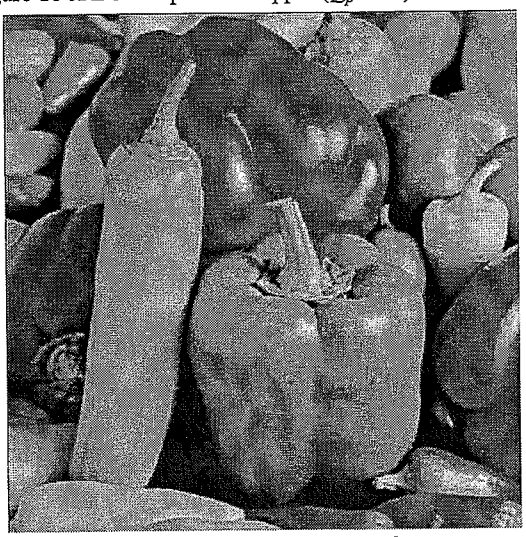


Figure 18 J-Mark (Q_p =3.0, fixed T_1=800, β_2)