

An Efficient Digital Image-in-Image Watermarking Algorithm Using the Integer Discrete Cosine Transform (IntDCT)

J. Zhang, Anthony T. S. Ho

Division of Information Engineering
School of Electrical and Electronic Engineering
Nanyang Technological University
Nanyang Avenue, Singapore 639798
E-mail: JingZhang@pmail.ntu.edu.sg

Abstract

In this paper, a robust and efficient watermarking algorithm is proposed for copyright protection of digital images. The proposed algorithm can also be extended to video sequences. Based on Ho et al.'s image-in-image watermarking algorithm [1], the proposed algorithm uses an integer discrete cosine transform (IntDCT) [7]. It is based on the IntDCT transform used in the state-of-the-art video compression standard H.264 [7], rather than the conventional DCT used in the prior video standards. The most significant advantage of this transform is that it is free from any floating-point or fixed-point multiplication required by the original DCT and all operations can be carried out with integer arithmetic, without loss of accuracy.

The proposed algorithm can embed a small image such as a company's trademark into the original cover images. The algorithm's performance is evaluated using StirMark 4.0 [10]. Experimental results show that this proposed algorithm can survive attacks such as scaling, additive noise, line removal as well as JPEG compression. The simplicity of the integer DCT transform also offers a significant advantage in shorter processing time and ease of hardware implementation than commonly used DCT techniques.

1. Introduction

Usage of digital multimedia has witnessed a tremendous growth during the last decade as a result of their notable benefits in efficient storage, ease of manipulation and transmission. Unfortunately, the very nature of the digital media makes the work of pirates and hackers easier, since it enables perfect copies with no loss of value. One solution that is gaining popularity in protecting digital contents for copyright owners is digital watermarking. Digital watermarking can hide copyright information (watermark) into the "essence" of the multimedia object. And the hidden information should be imperceptible and robust against malicious attacks.

Digital image watermarking can be performed in both spatial and transform domains. In spatial domain techniques, one of the simplest methods of inserting a digital watermark in a still image is called Least-Significant-Bit (LSB) Watermarking [2]. However, this technique has relatively low information hiding capacity and can be easily erased by lossy image compression. Techniques like superimposing a watermark image over an area of image to be watermarked [3] and signal-adaptive addition [4] are used to embed watermarks in the spatial domain. Watermarks can also be inserted in the frequency domain by applying transforms like Fast Fourier Transform (FFT) [5] and Discrete Cosine Transform (DCT) [6], and then altering the values of selected transform coefficients to store the watermark in still images. However, the main problem of these techniques is that their algorithms are complicated for both software and hardware implementations.

The proposed algorithm is based on Ho et al.'s image-in-image watermarking algorithm [1]. This algorithm applied the block Fast Hadamard Transform with the size of 8x8. The proposed algorithm uses a smaller transform IntDCT which is the essential transform of the newly developed video compression standard H.264 [7]. The H.264 standard is currently the most efficient and network-friendly video coding standard. This standard has already achieved a significant improvement especially in the rate-distortion efficiency, providing a factor of two in bit-rate savings when compared with existing standards such as MPEG-2 [11]. It is now the responsibility of a newly formed Joint Video Team between ITU-T VCEG and ISO/IEC MPEG.

This paper is organized as follows: the derivation and main properties of the IntDCT transform are described in Section 2. The image watermarking algorithm is presented in Section 3. Experimental results are analyzed in Section 4 and conclusion is given in Section 5.

2. Integer Discrete Cosine Transform

The proposed watermarking algorithm uses the integer DCT (IntDCT) which is used in H.264 video standard [7]. H.264 uses this new 4x4 transform to help reduce blocking

artifacts caused by existing video coding standards by using 8x8 DCT. IntDCT basically has the same properties as the original DCT. But there are some fundamental differences. First of all, it is an integer transform. All operations can be carried out with integer arithmetic, without loss of accuracy. The core part of the transform is free from multiplications. It only requires additions and shifts. It does not need floating-point and fixed-point multiplications required by DCT. This reduces the computational complexity and it is much easier for hardware implementation.

To develop this integer DCT, let us examine the 4x4 DCT of a 4x4 matrix X:

$$Y = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} * X * \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix}^T \quad (1)$$

Where

$$a = 1/2, b = \sqrt{1/2} \cos(\pi/8), c = \sqrt{1/2} \cos(3\pi/8)$$

The matrix multiplication can be factorized to the following equivalent form:

$$Y = (B * X * B^T) \otimes Q \quad (2)$$

Where

$$B = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix}, Q = \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}$$

$B * X * B^T$ is a "core" 2-D transform. Q is a matrix of scaling factors and the symbol \otimes indicates that each element of $B * X * B^T$ is multiplied by the scaling factor in the same position in matrix Q . d is c/b (approximately 0.414).

To simplify the implementation of the transform, d is approximated by 0.5. To ensure that the transform remains orthogonal, b also needs to be modified so that:

$$a = 1/2, b = \sqrt{2/5}, d = 1/2$$

The 2nd and 4th rows of matrix B and the 2nd and 4th columns of matrix B^T are scaled by a factor of 2 and the post-scaling matrix Q is scaled down for compensation. This avoids multiplications by 1/2 in the core transform $B * X * B^T$, which would result in loss of accuracy using integer arithmetic. The final forward 4x4 IntDCT becomes:

$$Y = (B_f * X * B_f^T) \otimes Q_f \quad (3)$$

Where

$$B_f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}, Q_f = \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix}$$

This transform is an approximation to the 4x4 DCT. In the context of the H.264 CODEC, it has almost identical compression performance to DCT and has a number of important advantages. The core part of the transform can be carried out with integer arithmetic using only additions, subtractions and shifts (to implement a multiplication by 2).

The inverse transform is given by:

$$X' = B_i^T * (Y \otimes Q_i) * B_i \quad (4)$$

Where

$$B_i = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix}, Q_i = \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}$$

The matrix Y is pre-scaled by multiplying each coefficient by the appropriate weighting factor from matrix Q_i . $\pm 1/2$ in the matrices B_i and in its transposed matrix can be implemented by a right-shift without a significant loss of accuracy because the coefficients Y are pre-scaled. In H.264 scaling matrices Q_f and Q_i are combined into quantization.

For transform performance comparison, we also applied other approximate transforms: Cham [9]'s 8-bit ICT and Liang et al. [8]'s 4/8-bit BinDCT. Cham [9] developed a series of 8x8 Integer Cosine Transforms (ICTs) based on the principle of dyadic symmetry. He claimed that the implementation complexity of an ICT depends on the number of bits required to represent the magnitude of its kernel components. Liang et al. [8] designed several families of fast multiplication-less approximations of DCT with the lifting scheme, named the BinDCT. It can support 3 different block sizes: 4x4, 8x8 and 16x16.

A performance comparison of accuracy and speed is performed among three 4x4 transforms, IntDCT, BinDCT and DCT. First a 4x4 input matrix is generated by randomly selecting its elements from 0 to 255. The input is forward and inverse transformed by the three transforms. This procedure is run 1000 times. The average speed of IntDCT (1.2519 times/ms) is approximately 5.6% faster than DCT (1.1860 times/ms), while the speed of BinDCT (1.1902 times/ms) is approximately the same as that of DCT, with no significant speed improvement. The reconstructed matrix by IntDCT is exactly the same as the input with the

normalized correlation value of 1. However, the outputs of DCT and BinDCT both have very little differences compared to the input, with the correlation value of 0.9999. These results show that IntDCT can perform faster than DCT, without loss of accuracy.

3. Watermarking in IntDCT domain

In this section we present an image-in-image watermarking algorithm using spatial masking. The embedding procedure is implemented in the following five steps: 1) Cover image decomposition and block selection; 2) Watermark image decomposition; 3) Forward IntDCT and watermark rearrangement; 4) Watermark strength control using spatial masking and embedding; 5) Inverse IntDCT.

The original cover image $f(x,y)$ is first decomposed into non-overlapping blocks of 4×4 , denoted by $f_k(x',y')$, $k=0, 1, \dots, K-1$, where the subscript k denotes the index of blocks and K denotes the total number of blocks. That is

$$f(x,y) = \bigcup_k f_k(x',y'), 0 \leq x',y' \leq 3 \quad (5)$$

We applied an M-sequence to pseudo-randomly select the relevant blocks $f_k(x',y')$ for watermark insertion among $f_k(x',y')$ $k=0,1,\dots,K-1$. This process can help to solve the problem of regional processing and attacks. The hidden bits information is spread around the image. It is still possible to retrieve some of the watermark information, even when some other information may have been lost during the attacks. The starting number is generated using a true random seed and is stored in a watermark key.

The watermark image $w(x,y)$ is also decomposed into non-overlapping blocks of 4×4 , denoted by $w_k(x',y')$, $k=0, 1, \dots, L-1$.

$$w(x,y) = \bigcup_k w_k(x',y'), 0 \leq x',y' \leq 3 \quad (6)$$

Every $f_k(x',y')$ and $w_k(x',y')$ are IntDCT transformed:

$$F'_k(u,v) = \text{IntDCT}\{f'_k(x',y')\}, 0 \leq u,v \leq 3$$

$$W_k(u,v) = \text{IntDCT}\{w_k(x',y')\}, 0 \leq u,v \leq 3 \quad (7)$$

All the frequency coefficients within every transformed watermark block $W_k(u,v)$ are scanned in a zigzag fashion and arranged in a one-dimensional sequence, denoted by m_i , $i=0,\dots,I-1$.

Within every transformed cover image block $F'_k(u,v)$, only several middle frequencies x_i of AC components are used, $i=0,\dots,N-1$. The watermark strength factor is denoted by α . The embedding formula is :

$$x_i^* = \alpha * m_i \quad (8)$$

The original coefficient x_i is replaced by the watermarked x_i^* . After that, the cover image block is substituted by the new 4×4 matrix of IntDCT coefficients. The watermarked cover image block is denoted by $F''_k(u,v)$.

Watermark strength α control is based on the edge-detecting mask shown in the following equation:

$$\alpha = \beta * \text{mask}(j,k) \quad (9)$$

Where β is the scaling factor, and j and k indicate the positions of the blocks.

The visual mask $\text{mask}(j,k)$ is adaptively obtained based on edge point density of the cover image by using the Canny Edge Detection algorithm in the spatial domain. Small values of $\text{mask}(j,k)$ indicate that the corresponding block is smoothly textured. Larger values indicate that the block contains outstanding edges.

The watermarked cover image blocks are transformed by inverse IntDCT.

$$f''_k(x',y') = \text{IntDCT}^{-1}\{F''_k(u,v)\}, 0 \leq x',y' \leq 3 \quad (10)$$

A watermark key containing a number of relevant data for private watermarking is then generated. These data include the sizes of the cover image and the watermark image, together with the seed and the starting number of M-sequence for pseudorandom selection of embedding location.

The extraction process of watermark from the attacked and watermarked cover image is relatively similar to the embedding procedure and can be implemented by the following four steps: 1) Cover image decomposition and blocks selection; 2) Forward IntDCT; 3) Extraction and watermark strength control; 4) Watermark rearrangement and inverse IntDCT. All the data stored in the key file during embedding is needed in the blind extraction procedure without any reference to the original cover image.

The watermark information is extracted from the middle frequency components. These components are denoted by x_i^* , and the retrieved watermark IntDCT coefficients are denoted by m_i , $i=0,\dots,I-1$. The watermark extraction formula is given as:

$$m_i = x_i^* / \alpha \quad (11)$$

Inverse zigzag scanning is then performed on these watermark IntDCT coefficients to obtain a set of 4×4 coefficients matrices. The extracted watermark image $w'(x,y)$, can be obtained by applying inverse IntDCT to the watermark blocks.

4. Results and discussions

The results of the proposed image-in-image algorithm using IntDCT are shown in Figure 4 (original 512x512 cover images *lena.bmp* and *baboon.bmp* marked with 32x32 watermark image *dmt.bmp*). The PSNR are as high as 38.7dB and 39.2dB, respectively.

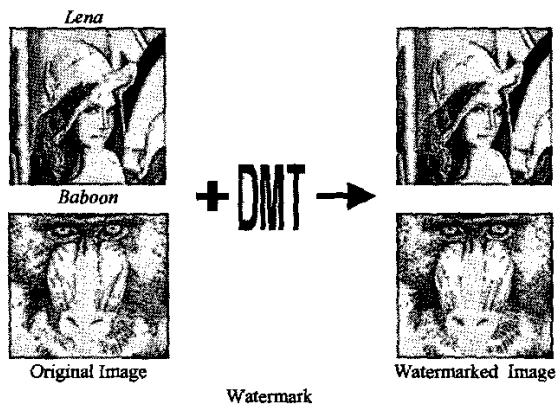


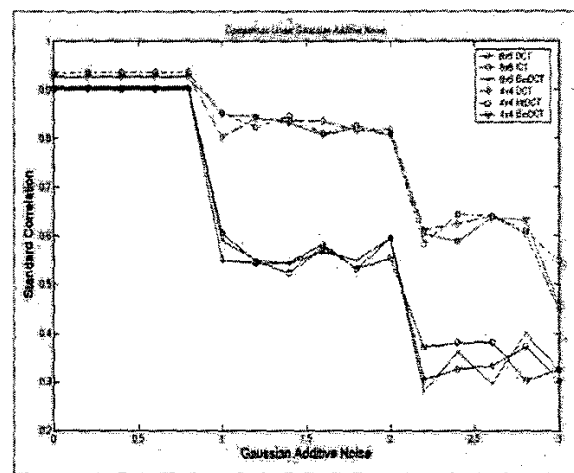
Figure 1 Results of Image-in-Image Algorithm

Table 1 shows the normalized correlation coefficients between the original and extracted watermark images, under different sample StirMark 4.0 attacks [10]. As shown in Table 1, our scheme could resist JPEG compression for even the low quality factor of 30. The additive noise was visible, in the images of *Lena* and *Baboon*, when the noise factor was greater than 1. Image quality would be significantly degraded when this factor was greater than 3. However, our algorithm could survive when image's quality has been degraded to 11dB by additive noise.

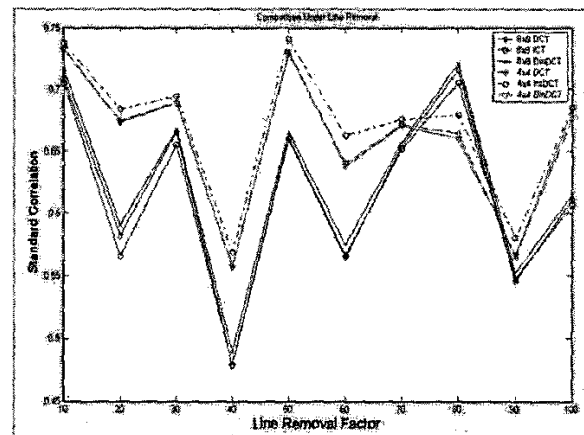
Attacks	Extracted Watermark	Correlation
No Attacks		0.93
Additive noise factor=1		0.93
Additive noise factor=3		0.63
JPEG factor=80		0.90
JPEG factor=40		0.73
3x3 Sharpening		0.71
3x3 Median filtering		0.63
Rescaling factor=200%		0.91
Rescaling factor=50%		0.92

Table 1 Results of Some StirMark Tests for *Lena*

Different transforms were evaluated based on the same image watermarking scheme with fixed PSNR value of watermarked cover images. These transforms are classified into two categories according to the transforming block size. The group with the size 4x4 is consisted of IntDCT, BinDCT and DCT. Another group with the size 8x8 is consisted of DCT, BinDCT and ICT. When 8x8 transforms were used in the scheme, the size of decomposed blocks was changed accordingly. For this evaluation the watermark image *dmt.bmp* was embedded in the cover image *lena.bmp* using the same feature extraction process. Standard correlations of different extracted watermarks are computed for comparison.



(a) Additive Noise



(b) Line Removal

Figure 2 Performance Comparisons among Different DCT Transforms using cover image *Lena*

Figure 2 shows the comparison results under typical additive uniform noise and line removal attacks. For the image *Lena*, the IntDCT transform can survive additive noise as well as 4x4 DCT for additive noise factors lower

than 3. The average standard correlation of IntDCT is 1.8% higher than that of 4x4 DCT, and 1.7% than that of BinDCT.

Since IntDCT is designed by using integer values to approximate the floating-point magnitude of the conventional DCT's kernel components, all operations can be carried out by integer arithmetic with only binary shifts and additions. If these operations are followed correctly, mismatch between the input and output after going through forward and inverse transforms should not occur.

From Figure 2 (b), we can see that IntDCT shows higher robustness than 4x4 DCT with 2.6% correlation increase. And it has better robustness than 8x8 transforms at factors such as 30, 40 and 60 respectively. However, at factor 80 for example, these 4x4 transforms have lower robustness than 8x8 transforms.

The 4x4 IntDCT is designed with a small block size to help in reducing blocking artifacts caused by the prior video coding standards using the 8x8 DCT transform, as well as increasing compression efficiency. Some drawbacks of using the 4x4 IntDCT are its relatively poor performances against some powerful filtering and up-down-sampling attacks. This is because that for a larger 8x8 DCT matrix, one filtered frequency only possesses approximately 1.5% of all the frequency components. Therefore, there is still sufficient information which can be retrieved from the remaining 98.5% of all frequency coefficients. While for a 4x4 IntDCT matrix, if one frequency is filtered, approximately 6.3% frequency information is lost. The small block size becomes the limitation of IntDCT for these powerful filtering and up/down-sampling attacks, such as Median Filtering 4x4.

5. Conclusion

This paper has presented an image-in-image watermarking technique based on the Integer Discrete Cosine Transform (IntDCT) that is used in the state-of-the-art video compression standard H.264. The proposed algorithm could survive such as additive noise, rescaling, line removal attacks and JPEG compression well. For JPEG compression, the watermark was still recognizable even at the low quantization factor of 30. Moreover, the simplicity of the integer DCT transform offered a significant advantage in shorter processing time and ease of hardware implementation than commonly used DCT techniques. Comparison results among IntDCT, DCT and other integer DCTs have also been presented. For future work, the proposed watermarking algorithm using the IntDCT transform will be extended to video watermarking based on the H.264 standard.

References

- [1] A.T.S. Ho, J. Shen, and S. H. Tan, "Digital image-in-image watermarking for copyright protection of satellite images using the fast hadamard transform," IGARSS02', Toronto, Canada, June 16-24, 2002.
- [2] R. G. van Schyndel, A. Z. Tirkel, and C. F. Shamoon, "Secure spread spectrum watermarking for multimedia," IEEE Trans. Image Processing, vol. 6, pp. 1673-1687, Dec. 1997.
- [3] C. Langelaar, J. C. A. Ven der Lubbe, and R. L. Lagendijk, "Robust labeling method for copy protection of images," Proceedings of SPIE/IS&T Electronic Imaging, San Jose, CA, vol. 3022, pp. 298-309, Feb. 1997.
- [4] M. Barni, F. Bartolini, V. Cappellini, and A. Piva, "A DCT-domain system for robust image watermarking," Signal Processing (Special Issue on Watermarking), vol. 66, no. 3, pp. 357-372, May 1998.
- [5] C.-W. Tang and H.-M. Hang, "A feature-based robust digital image watermarking scheme," IEEE Transactions on Signal Processing, vol. 51, no. 4, pp. 950-959, Apr. 2003.
- [6] M. Barni, F. Bartolini, V. Cappellini, and A. Piva, "A DCT-domain system for robust image watermarking," Signal Processing (Special Issue on Watermarking), vol. 66, no. 3, pp. 357-372, May 1998.
- [7] "H.264/MPEG-4 Part 10: Transform & quantization," ITU-T Rec. H.264/ISO/IEC 11496-10, Final Committee Draft, Document JTV-F100, Dec. 2002. (Latest version modified on 19.03.2003) [Online] Available: <http://www.vcodex.fsnet.co.uk/h264.html>
- [8] J. Liang, T. D. Tran and P. Topiwala, "FastVDO's unified 16-Bit transform/quantization approach," JVT-B018.doc, 2nd Meeting of JVT of ISO/IEC MPEG & ITU-T VCTEG, Geneva, CH, 29 January-1 Feb. 2002.
- [9] W. K. Cham and P. C. Yip, "Integer sinusoidal transforms for image processing," International Journal of Electronics, vol. 70, no. 6, pp. 1015-1030, Jun. 1991.
- [10] [Online] Available: <http://www.cl.cam.ac.uk/~fapp2/watermarking/stirmark/>
- [11] R. Schafer, T. Wiegand, and H. Schwarz, "The emerging H.264/AVC standard," EBU Technical Review, Audio/Video Coding, Jan. 2003.