A New Digital Image Watermarking Algorithm Using DCT and DWT

Yan Zheng
University of Florida
Email: zhengyan666@ufl.edu

***Abstract*—digital watermark can protect our authenticity from being copied or downloaded by others. A watermarking algorithm usually combines transformation method, embedding algorithm and encryption algorithm. In our project, we make a combination of discrete wavelet transforms (DWT) and discrete cosine transforms (DCT), Cox's embedding algorithm, chaotic encryption algorithm, and Arnold encryption algorithm. In addition, we perform this algorithm under noise like Gaussian noise and Pepper and Salt noise to evaluate it on different DWT's sub-band and different level DWT using PSNR, correlation coefficient. It performs well when comparing to other algorithms like Ref 5] and Ref.[6].**

***Index Terms*—Digital watermarking, DWT, DCT, Arnold, PSNR.**

I. Introduction

Image Watermarking procedure is shown in figure 1.

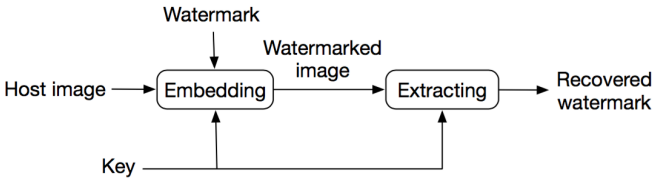

Figure 1. Digital watermarking procedure

The original watermark image first gets encrypted by a key, then embedded into host image to get watermarked host image. The watermark image can be extracted from the watermarked host image with the same key. There are a variety of algorithm to encrypt watermark and embed watermark into host image. So it is easy to make a new algorithm for watermarking, but it is not easy to get a good result. In order to success, we should think about watermark structure, embedding method, and how to make a trade-off between the difference between original host image and watermarked host image and the difference between original watermark and recovered watermark.

This paper indicates a new image watermarking algorithm using DCT and DWT. Watermark embedding is done by using a watermark image and perform Arnold algorithm and chaotic encryption algorithm to scrambling it with a key. Then we perform 2-level DWT to decompose the host image and followed by a DCT on the selected sub-band. When performing DCT on host image, the scrambled watermark also performs DCT and get embedded into sub-band, after IDCT and 2-level IDWT to the sub-band, we get the watermarked host image. Watermark extracting is done by performing 2-level DWT to get the selected sub-band. Then using DCT to get the scrambled watermark, this watermark should be similar to watermark in embedding process after chaotic encryption algorithm. Then we use chaotic encryption algorithm followed by Arnold algorithm to decode scrambled watermark and get recovered watermark. The watermark size and host image size have to be adjusted when we perform different levels DWT, and we select different sub-bands to get different watermarked image. After a noise attack, like adding Gaussian noise to watermarked image, we can get different recovered watermark. We use PSNR to evaluate if watermark has been hided safely and see the similarity between host image and watermarked image, and we use correlation coefficient to measure the similarity between watermark and recovered watermark.

II. Watermark Embedding Procedure

DWT can separate host image into four sub-bands which are LL (high scale low frequency components), HL (Horizontal low-scale, high-frequency components), LH (Vertical low-scale, high-frequency components), and HH (Diagonal low-scale, high-frequency components)[2]. For example, if host image size is 512*512, then after 1-level DWT, each sub-band size is 256*256, and if we do another DWT on a selected sub-band, since host image size is 256*256 now, we will get four sub-bands whose size is 128*128, figure 2 shows the DWT procedure. Because the relationship between size of watermark and sub-band will change the performance, so it is a important factor to take into account. In our project, when we perform 2- level DWT on a 512*512 host image, we get 128*128 sub-band, so we use 32*32 watermark image to perform our new algorithm. Figure 3 shows the steps, the following steps are an example which can present our embedding procedure.

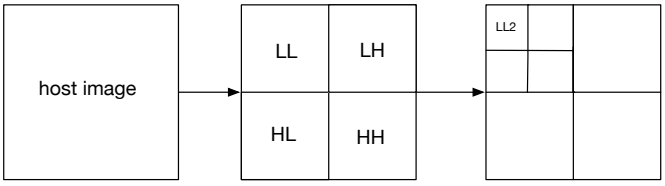

Figure 2. DWT procedure

- Step 1 Perform Arnold encryption algorithm on 32*32 watermark to get a 32*32 scrambled watermark.

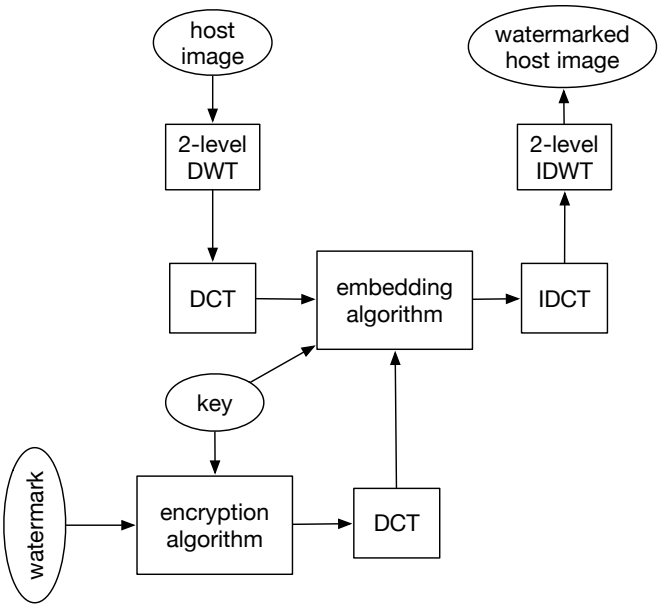

Figure 3. watermarking embedding procedure

- Step 2 Perform chaotic encryption algorithm with a key on 32*32 scrambled watermark to get a new 32*32 scrambled watermark.
- Step 3 Perform DWT on 512*512 host image to get four 256*256 sub-bands: LL, LH, HL, HH.
- Step 4 Perform DWT on 256*256 LL to get four 128*128 sub-bands:LL2, LH2, HL2, HH2.
- Step 5 Since we have 128*128 LL2 image and 32*32 watermark image, we decompose LL2 into 32*32 blocks, and each block size is 4*4. We embed one watermark image pixel in one block, performing DCT on each block and use key in Step 2 to decide where to embed in block.
- Step 6 Perform IDCT on each embedded block to get embedded LL2.
- Step 7 Perform 2-level IDWT on LL2 to get watermarked host image.

If we want to perform 3-level DWT, we should use a 1024*1024 host image and add a DWT procedure between Step 4 and Step 5 to get 128*128 LL3. And perform 3-level IDWT in Step 6. If we still use 512*512 host image, the watermark size should be 16*16 which are not able to contain much information, another solution is we can change block size to 2*2 and still use 32*32 watermark, but the block size is too small that will make a change to the similarity between host image and watermarked image. So, we choose to use a larger host image. And we can change LL to other sub-bands to evaluate performance in different DWT sub-bands.

A. Arnold encryption algorithm

Arnold encryption algorithm is based on Arnold's Cat Map, it can scramble a image to be unnoticeable and after many times to ACM transformation, it return to be the original image. It is defined mathematically as following:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{pmatrix} x' \\ y' \end{pmatrix} mod\ N \tag{1}$$

ACM transformation has a period. For any integer $N > 2$, we have period $T \leq N^2/2$[10]. To calculate T, T is the min number n that can sacrifice the formula below:

$$\begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}^n (mod\ N) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{2}$$

So after T times ACM transformation, the image returns to be the original image.

Here is ACM period for special size image:

Table I
ARNOLD PERIOD IN DIFFERENT SIZE IMAGES

Size N	4*4	8*8	32*32	64*64	128*128	256*256
Period M	3	6	24	48	96	192

B. chaotic encryption algorithm

Shannon says, if a random sequence can be generated by a key, and any small changes in input will get a totally difference at out put, then this sequence can be used for encryption. Chaos sequence satisfied this requirement. It is very sensitive with the initial value. Chaos model in one dimensional logistic map can be defined as follows:

$$Xn + 1 = \mu Xn(1 - Xn), 0 < Xn < 1, 0 < \mu < 4 \tag{3}$$

When $\mu > 3.57$ sequence Xn has chaotic features[11]. Since using any key to decode any data can get a result without error, so it is impossible to use brute force attack to decode chaos sequence.

In our algorithm, we use $\mu = 4$ and Xn to generate a chaos sequence and using a key to get part of the sequence, the sequence size we get is the same as watermark pixel size 32*32. Then we compare each number in sequence to a number N which is between 0 and 1. If it is larger than N, it changes to 0, otherwise, it changes to 1. Then we get a sequence with 0 and 1. We transform the sequence to a 32*32 matrix and make a "XOR" with watermark to get a encrypted watermark matrix.

In the decode procedure, we use the same initial value to generate the same matrix, and make a "XOR" with encrypted watermark to get recovered watermark.

C. Cox's embedding algorithm

Embedding is done with a scaling factor str[8].

$$block(X, Y) = str * watermark(j, i) \tag{4}$$

Block is the result we decompose the host image, X and Y defines where to embed watermark pixel and watermark is one pixel of watermark.

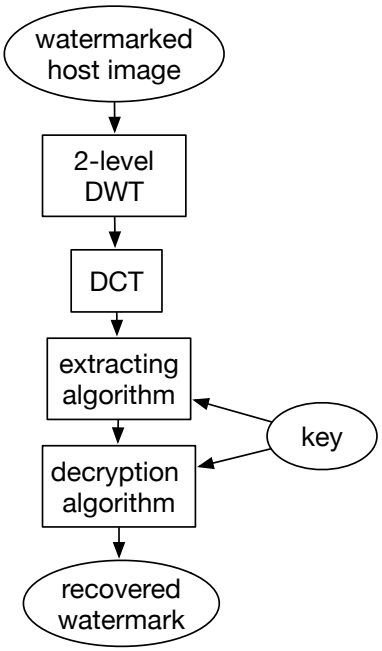

Figure 4. watermarking extracting procedure

III. Watermark Extracting Procedure

Figure 4 shows the extracting procedure and steps are described as follows:

- Step 1 Perform DWT on 512*512 watermarked host image to get sub-band 256*256 LL.
- Step 2 Perform DWT on 256*256 LL to get sub-band 128*128 LL2.
- Step 3 decompose LL2 to marked blocks in 4*4 size, and use the same key to determine where to extract watermark.
- Step 4 Perform DCT on each marked block and get each watermark pixel from each marked block and combine these pixels to watermark.
- Step 5 Using chaotic encryption algorithm to decode watermark.
- Step 6 Using Arnold encryption algorithm to decode watermark and get recovered watermark.

IV. Performance Evaluation

We test imperceptibility of host image and robustness of watermarked image to see performance.

A. Imperceptibility

Imperceptibility means host image should not be distorted by embedding watermark image. PSRN can measure the similarity between host image and watermarked image. When $PSNR > 35$, we can hardly tell the difference between them. PSNR can be defined via the mean squared error (MSE), MSE is used to test the similarity between two images[2]. The equation is written as below:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \tag{5}$$

$$\begin{aligned} PSNR &= 10 * log\left(\frac{MAX^2}{MSE}\right) \\ &= 20 * log\left(\frac{MAX}{MSE}\right) \\ &= 20 * log(MAX) - 10 * log(MSE) \end{aligned} \tag{6}$$

I is the host image and K is the watermarked image. m is the row number of host image, n is the column number of host image. MAX is the maximum possible pixel value of the image.

B. Robustness

The robustness can be tested by compare original watermark and recovered watermark after a noise. We use correlation coefficient to measure the similarity between these two images. The equation is written as below:

$$r = \frac{\sum\sum(A - \overline{A})(B - \overline{B})}{\sqrt{\sum\sum(A - \overline{A})\sum\sum(B - \overline{B})}} \tag{7}$$

C. Simulation data

Table II
DIFFERENT ALGORITHM COMPARISON

Image attacks	Ref.[5]	Ref.[6]	Ref.[7]	The proposed
JPEG(Q=50)	1.0000	0.6519	1.0000	0.9950
Gaussian noise(0.001)	0.6820	0.3545	0.9354	0.8923
Gaussian noise(0.002)	0.3553	0.1257	0.7772	0.7129
Gaussian noise(0.003)	0.1512	*	0.5628	0.6093
pepper salt(0.001)	0.9397	0.7468	0.9354	0.9476
pepper salt(0.002)	0.8861	0.6058	0.9154	0.9083
pepper salt(0.003)	0.8403	0.5396	0.8320	0.8679
3*3 media filter	0.8349	0.5432	0.8543	0.7335
Cutting 50:200	0.8654	0.8215	0.8965	0.8154
scale (0.5)	*	*	0.9733	0.7798

Table III
DIFFERENT LEVEL DWT

Image attacks	1-level	2-level	3-level	PSNR
JPEG(Q=50)	0.9731	0.9950	0.9123	33.80
Gaussian noise(0.001)	0.8666	0.8923	0.7261	29.35
Gaussian noise(0.002)	0.6796	0.7129	0.5892	26.63
Gaussian noise(0.003)	0.5570	0.6093	0.4968	24.96
pepper salt(0.001)	0.9374	0.9476	0.8977	33.30
pepper salt(0.002)	0.8667	0.9083	0.8236	31.36
pepper salt(0.003)	0.8310	0.8679	0.7144	30.22
3*3 media filter	0.8988	0.7335	0.7377	33.83
Cutting 50:200	0.7761	0.8154	0.8123	15.91
scale (0.5)	0.9067	0.7798	0.7324	32.87

Table IV
DIFFERENT SUB-BAND DWT

Image attacks	LL	LH	HL	HH
JPEG(Q=50)	0.9950	0.5759	0.3010	0.0315
Gaussian noise(0.001)	0.8923	0.8846	0.8843	0.8225
Gaussian noise(0.002)	0.7129	0.7329	0.6484	0.6429
Gaussian noise(0.003)	0.6093	0.5826	0.5453	0.5286
pepper salt(0.001)	0.9476	0.9292	0.9424	0.9566
pepper salt(0.002)	0.9083	0.8846	0.9002	0.9313
pepper salt(0.003)	0.8679	0.8393	0.8553	0.8608
3*3 media filter	0.7335	0.5745	0.6156	0.5118
Cutting 50:200	0.8154	0.8282	0.8338	0.8371
scale (0.5)	0.7798	0.2922	0.2028	0.0805

Table two is a comparison of different algorithm. Figure 5 is a plotting of this comparison. The simulation data of other algorithm are taken from reference 7. The proposed algorithm is 2-level DWT in LL sub-band, this is a best solution we found for our algorithm. And we made a comparison between our algorithm and others. From table two, we can see our algorithm perform much better in noise attack and compression than Ref.[5] and Ref.[6]. Its performance are about 4 percents lower than Ref.[7] in 0.002 Gaussian noise attack and 8 percents lower than Ref.[7] in 0.003 Gaussian noise attack, but it performs 4 percents higher in 0.003 pepper and salt noise attack. Although Ref.[7] has a better performance in filter attack, cutting attack, and scale attack, watermark through our algorithm can still be recognized.

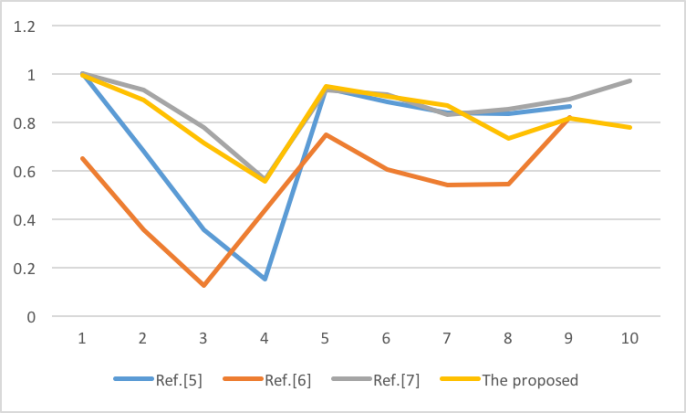

Figure 5. different algorithm comparison

Table three is a comparison of our algorithm in different level DWT. In 1-level DWT, we use 1024*1024 host image and 128*128 watermark, in 2-level DWT, we use 512*512 host image and 32*32 watermark, and in 3-level DWT, we use 1024*1024 host image and 32*32 watermark. 3-level DWT perform much lower in noise attack, because host image is much larger than watermark, watermark are effected more by noise in this situation. 1-level DWT perform lower than 2-level DWT in noise attack, but higher in filter attack and scale attack. We made a trade-off between the level of DWT and the size of watermark and choose 2-level DWT to be proposed algorithm. PSNR is between host image and watermarked host image after being attacked under 2-level DWT. The images after being attacked are still very similar to original images, which means the proposed algorithm has a good imperceptibility.

Table four is a comparison of our algorithm performing DWT in different sub-band. We conclude that our algorithm perform best in LL sub-band. This is because HL, LH, and HH sub-bands are high frequency sub-bands. HL contains horizontal detail, LH contains vertical details, and HH contains diagonal details, they are stable and difficult to be detected by eyes, but easy to be effected by noise and lose watermark information[1]. On the other hand, LL contains most energy of original host image, so it is a better choice for our algorithm to embed watermark in LL sub-band.

D. *Simulation image*

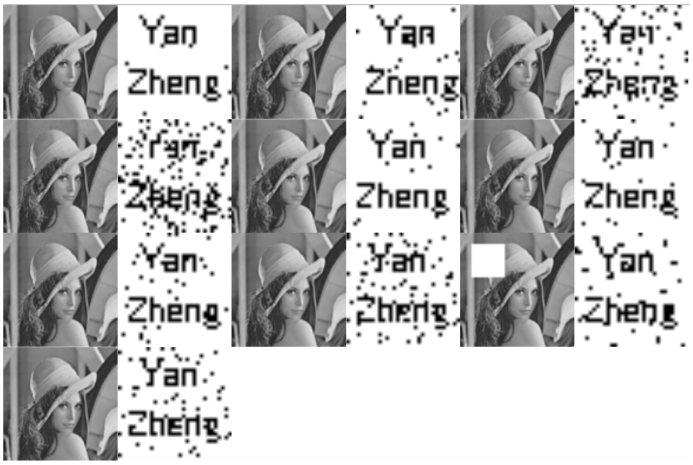

Figure 6. 2-level DWT in LL

The simulation result images are watermarked host image and recovered watermark according to table two in the proposed algorithm. The order is from left to right, from top to bottom.

V. RELATED WORKS

I find a lot of watermark algorithms during this project. Akter *et al* [2] use DWT followed by DCT, and he invents a new embedding algorithm which embed watermark into two sub-bands. Lagzian *et al* [3] use DWT and SVD, embed watermark into all sub-bands. Bajaj [9] use DWT, DCT, SVD to develop a watermark algorithm. There are still many other algorithms I have to read. After reading many papers, I sum up an experience: a watermark algorithm can be separated into three part — transformation method, encryption algorithm, and embedding algorithm. This is the reason why there are a large amount of watermark algorithms.

VI. Summary and Conclusion

I read Ref.[2], and decide to make a algorithm using DCT and DWT. I use chaotic encryption algorithm from Ref.[4], and combined it with Arnold encryption algorithm. I learn Cox's embedding algorithm from Ref.[2] and Ref.[8]. There three place using keys: in Arnold encryption algorithm, we use a key to decide Arnold transformation times in embedding procedure, in chaotic encryption algorithm, we use a key to create a sequence, and in Cox's embedding algorithm, we use a key to decide position to insert watermark pixel into a block. I learn performance evaluation method from Ref.[2] and Ref.[7]. I search source code about these three parts and learn MATLAB on Internet, and then I write my own code. After this project, I totally understand DCT and DWT, I also learn many encryption algorithm for matrix, evaluating method like normalized correlation, and my MATLAB skill improves much. I got stuck on calculating correlation coefficient, some papers didn't take mean value into account, so their correlation coefficient is always higher than 0.9 even under much noise condition like Ref.[9]. I followed my text book using zero-mean normalized cross correlation.

As for my algorithm, its performance is lower than Ref.[7] under Gaussian noise but higher than Ref.[7] under pepper and salt noise. It is a little bit worse than Ref.[7], but much better than Ref.[5] and Ref.[6] according to figure 5. I think it is a successfully algorithm, I learned much about encryption and MATLAB from making it. Our extended work is to embed watermark into four sub-bands, and explore the best type of data in watermark that should be embedded into each sub-band.

References

[1] Darshana Mistry, Asim Banerjee, "Discrete wavelet transform using MATLAB", IJCET ,Volume 4, Issue 2, March – April (2013), pp. 252-259.

[2] Afroja Akter, Nur-E- Tajnina, Muhammad Ahsan Ullah," Digital Image Watermarking based on DWT-DCT: Evaluate for a New Embedding Algorithm",IEEE 3rd International Conference on informatics, electronics & vision, 2014.

[3] S. Lagzian, M. Soryani, and M. Fathy, "Robust watermarking scheme based on RDWT-SVD: Embedding Data in All subbands," in Artificial Intelligence and Signal Processing (AISP), 2011 International Symposium on, 2011, pp. 48–52.

[4] R. Matthews, On the derivation of a "Chaotic" encryption algorithm, Cryptologia, v.8 n.1, pp.29-41, Jan. 1984

[5] Vladimir Gorodetski, Leonard Popyack. "A SVD-based Approach to Transparent Embedding Data into Digital Images," Information Assurance in Computer Networks. Methods, Models and Architectures for Network Security, 2001, pp. 263-274.

[6] Paul Bao, Xiaohu Ma. "Image Adaptive Watermarking Using Wavelet Domain Singular Value Decomposition," IEEE Transactions on Circuits and Systems for Video Technology, 2005, 1(15): 96-102.

[7] Ye Xueyi, Deng Meng, Wang Yunlu, Zhang Jing, "A robust DWT-SVD blind watermarking algorithm based on Zernike moments," Communications Security Conference (CSC 2014), 2014.

[8] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. In IEEE Trans. on Image Processing, 1997, pp. 1673–1687.

[9] Anu Bajaj. "Robust and reversible digital image watermarking technique based on RDWT-DCT-SVD," ICAETR, 2014.

[10] Freeman J. Dyson, Harold Falk, "Period of a Discrete Cat Mapping," The American Mathematical Monthly, Vol. 99, No. 7 , Aug. - Sep., 1992, pp. 603-614

[11] X. Wang, L. Ma, and X. Du, "An Encryption Method Based on Dual- Chaos Systems," Second International Conference on Intelligent Networks and Intelligent Systems, ICINIS '09, pp. 217-220, Nov. 2009.