Implementation of Robust and Secure Watermarking Algorithm on FPGA using DCT

ANKITA NATUJIRAO PATIL, VAISHALI V. INGALE, FATEMA A. DALAL, VANITA AGARWAL
Electronics and Telecommunication Department,
COEP Technological University,
Pune, Maharashtra,
INDIA

Abstract: - Digital watermarking deals with embedding digital content in a cover signal so that it becomes indiscernible to make it robust against different security threats. This work demonstrates the implementation of a robust and secure watermarking technique on a Field Programmable Gate Array (FPGA) using Discrete Cosine Transform (DCT). Block by block DCT of the cover image was computed and watermark pixels in each of the blocks were hidden using middle-frequency band coefficients. Security is ensured by encrypting the watermarkwith a secret key and Arnold transform. The resultant watermark image is robust to various threats like JPEG compression, scaling, cropping, and noise attacks. This watermarking approach requires a large amount of data to be processed at high speed. FPGA is the best choice for such an application. Also, it is reprogrammable and easily upgradable according to user application. In this work, a watermarking algorithm is implemented using Xilinx design tools on the Artix-7 FPGA board.

Key-Words: - DCT, FPGA, Watermarking, Xilinx, SDK, Arnold transform**.**

Received: March 15, 2024. Revised: August 7, 2024. Accepted: September 2, 2024. Published: October 24, 2024.

1 Introduction

Nowadays use of digital media is increasing rapidly, and hence security, copyright and authentication of that digital information become a major issue. Watermarking is the best solution to avoid copyright misuse and secure digital data. Watermarking is a method of hiding digital content in host media which can be image, video, or audio. Image watermarking is a technique in which we can hide digital information in the host image and the information could be perceptible or imperceptible as per application.

Based on the domain, water marking schemes are grouped into time and transform domains. In the time domain technique, the watermark can be directly insertedby modifying pixel values of the cover signal image such as modification of least significant bit (LSB) or correlation-based techniques. Manipulation of the pixel value accounts for the ease of implementation, very low computational complexity, and good imperceptibility. However this algorithm is not robust against security threats such as compression, geometrical, and addition of external noise. In the transform domain technique, the watermark is embedded in selective coefficients of frequency transformed cover signal. Robustness and imperceptibility depend upon the coefficient selection. Some of the commonly used techniques in the transform domain include Discrete Fourier transform (DFT), discrete cosine transform (DCT), and discrete wavelet transform (DWT). Computing transform results in higher computational complexity. However this method is found to be effective against different threats and provides good imperceptibility. DCT is one of the best techniques in this regard. The content of the image is separated into three regions low, mid, and high frequencies. The visually significant information is concentrated in the low frequency band. As the discrete cosine transform is just the cosine part of the Fourier transform it requires less computation as compared to other transforms. For this reason, DCT is popularly used in the image watermarking, [1], [2], [3], [4], [5].

Watermarking techniques involve processing large amounts of data available from medical imaging taken for clinical analysis, camera images, and legal information for copyright protection and authentication. To process this large amount of data efficiently, we require a hardware platform. Field programmable gate arrays (FPGA) have been extensively used in image processing and signal processing. FPGA is reprogrammable as well as

easily upgradable, [6]. It provides flexibility for general-purpose processors and the hardware-basedperformance of ASICs. It is possible to embed Processors/microcontrollers into the FPGA, so users can define and use processor and user-specific hardware functions both on a single chip.

The Xilinx Embedded Development Kit (EDK), is an integrated development environment consisting of Xilinx platform studio and software toolkit to facilitate the designing of embedded processing systems. In addition, it contains various intellectual property (IP) cores which are required for designing FPGAs with Micro Blaze™ soft processor cores. The included software development kit (SDK) is a set of software development tools on which users can create applications for certain hardware/software platforms, [7], [8].

2 Proposed Watermarking Algorithm

2.1 Discrete Cosine Transform

In the proposed work, the cover image is divided into 4*4 blocks. As our cover image is 256*256, we obtained 4096 blocks. Then DCT coefficients are obtained for each of the blocks

DCT for each block is given as,

$$F(u,v) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} f(i,j)C(u)C(v)\cos\frac{(2i+1)u\pi}{2N}\cos\frac{(2j+1)v\pi}{2N} \quad (1)$$

C (u) =C (v) =$\frac{1}{\sqrt{N}}$; others C (u) =C (v) =$\sqrt{\frac{2}{N}}$ … for u=v=0

As mentioned earlier, the cover image is divided into three regions each corresponding to a different frequency band, low frequency region, [9], mid frequency region, and high frequency region. To embed the watermark, we selected mid frequency region as high frequency coefficients are more susceptible to attacks and low-frequency coefficients contain more visual information.

2.2 Arnold Transform

Arnold transform, also known as Arnold cat map is a 2D chaotic map, [10]. This transform is periodic and it is mainly used for hiding information as well as for encryption. It is suitable for square images, that is images of size N*N. In this work, we used the Arnold cat map method which is a 2D chaotic map to scramble the watermark image. The equation of Arnold cat map is:

$$U = mod(2 * x + y, N) + 1 \quad (2)$$

$$V = mod(x + y, N) + 1 \quad (3)$$

Where,U and Vrepresent new coordinate position of pixel, $x\&y$ are original position, and N is a size of the square image.

2.3 Process to Embed Watermark Image

In the proposed method, we are embedding a binary watermark into the cover image as shown in Figure 1. In the process of embedding information, we decomposed 256*256 cover images into the 4*4 blocks. Then we applied DCT on each block. To obtain more security, we scrambled the watermark using Arnold transform for multiple iterations. After that, the scrambled watermark is XORed with 64-bit symmetric key. Embedding logic is then used to embed the encrypted watermark bits in mid-frequency band coefficients Threshold value is calculated for each block by using the following formula.

$$\text{Th}=\frac{\sum_{n=0}^{k*k} DCT\_f_n(i,j)}{64} \quad (4)$$

Where,k is the size of block and DCT_f_n is DCT coefficients of the cover image. Watermark bits are embedded using following logic.

$$\text{DCT\_}f_n(\text{i,j}) = \begin{cases} -(\text{Th}+\Delta)(\text{W}=0) and\ (DCT\_f\text{n}\ (\text{i,j}) > -T\square) \\ \text{DCT\_}f_n(\text{i,j})(\text{W}=0) \quad and\ (\text{DCT}_{\text{fn}}(\text{i,j}) < -T\square) \\ (\text{Th}+\Delta)(\text{W}=1) \quad and\ (\text{DCT}_{\text{fn}}(\text{i,j}) < T\square) \\ \text{DCT\_}f_n(\text{i,j})(\text{W}=1) \ and\ (DCT\_f\text{n}\ (\text{i,j}) > T\square) \end{cases} \quad (5)$$

W is encrypted watermark image and Δ is a weighting factor. The imperceptibility and robustness of a watermarked image depends upon the value of Δ and amount by which we divide sum of DCT coefficients. After embedding the watermark in DCT coefficients, we took inverse DCT of the block and combined all blocks to form the watermarked image. The detailed embedding process is shown in Figure 1.

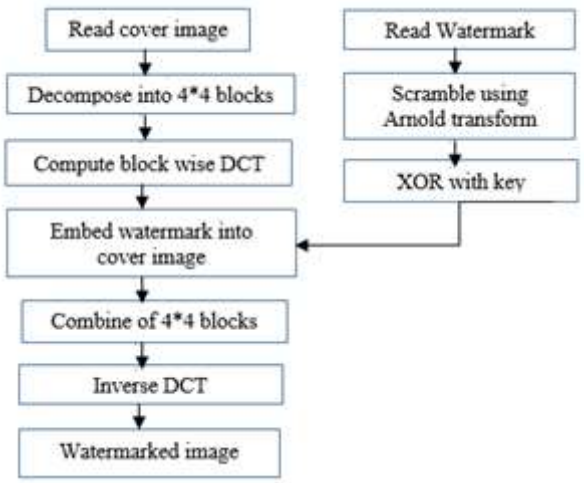


Fig. 1: Proposed watermark embedding scheme

2.4 Process to Extract Watermark Image

The extraction process of the watermark is inverse of the embedding process as shown in Figure 2. Being a blind approach, the initial cover image is not required in the process of extracting the watermark. Decompose the watermarked image in blocks of 4*4 and then compute DCT for each of the blocks. Extract a dimensional encrypted watermark by using the following logic.

$$W = \begin{cases} 1 \; DCT\_F_n^{'}(i,j) \geq 0 \\ 0 \; DCT\_F_n^{'}(i,j) < 0 \end{cases} \quad (6)$$

Where, W is encrypted watermark.

Restore the extracted watermark in two-dimensional image. The obtained watermark is encrypted. To decrypt it, descramble it by using Arnold transform and XOR it with the same 64-bit key which used at the time of encryption. After XOR it with the key, we get a watermark image. Detailed extraction process is given in Figure 2.

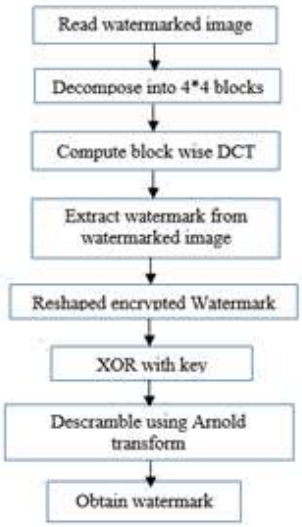


Fig. 2: Blind watermark extraction scheme

3 Hardware Implementation

Hardware design flow consists of preprocessing of the input image using MATLAB, sending the image to FPGA over UART through MATLAB, implementing of complete algorithm using Xilinx ISE tools, and verifying the results on MATLAB obtained from FPGA. Preprocessing of the input image consists of reshaping the image and converting it from RGB to grayscale. The one-dimensional input data is sent over UART by using specific baud rate, data bits, and parity bits to FPGA through MATLAB. Data sent over UART is stored in the block RAM of the FPGA for further processing.

3.1 Project in Xilinx ISE (Integrated Synthesis Environment)

Xilinx ISE project was created with all required ports such as clock, reset, uart_tx and uart_rx port. If we can't use on board system clock directly for our modules, then add clocking wizard intellectual property (IP) from the core generator and architecture wizard. Clock wizard IP converts onboard system clock into low-frequency clock as per the requirement of user with reset, clock_in, and clock_out ports. Add a new embedded processor in design which will have IP's for UART and block RAM, [11], [12], [13].

3.2 Xilinx Platform Studio (EDK)

Whenever a new embedded processor is added to ISE design, the Xilinx Platform Studio (XPS) application will be started. XPS has different IP cores as shown in Figure 3, out of which we have used axi_uartlite, which will run on MicroBlaze soft core processor using AXI4-Lite interface. Set proper parity and baud rate for this UART IP. Here we used 460800 baud rate and no parity. As we are working on a large amount of data, here block RAM of 256 KB was selected. Our data consist of two images of 64kb each and one image of 4kb. The remaining portion of block RAM is used for intermediate variables and code. After adding all required IPs, generate a netlist and export the hardware design to SDK.

Fig. 3: System assembly view of XPS

3.3 Creating Software Development Kit (SDK) Project

In the software development kit we can create applications for selected hardware platforms. After exporting the hardware design to SDK, a new application project was created and the proposed watermarking algorithm is written in SDK using C language. After the SDK project is built, an elf file

is generated which is exported in the Xilinx ISE project.

After including all required modules, UCF file, and elf file from SDK in the Xilinx ISE project, a programming bit file is generated. Selected FPGA device is configured by using that file.

4 Experimental Observations

The algorithm proposed in this work is tested on different types of images as shown in Figure 4 and performance is analyzed. The results for the 256×256, 8-bit gray-scale cover image are presented below. The result is a 64x64 binary watermark image.

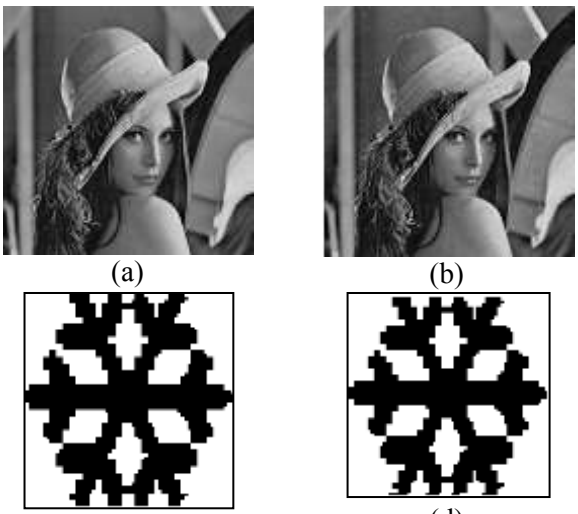


Fig. 4: (a) Original cover image, (b) Watermarked image, (c) Original watermark, (d) Extracted watermark

The peak SNR (PSNR), given by Equation (4.2), and structural similarity index measure are the parameters used for the original cover image with the resulting watermarked image. The watermarked cover image has PSNR= 34.7068 and SSIM= 0.9012. No perceptual degradation was observed in the watermarked images when compared with the original cover image.

$$MSE = \frac{1}{M * N}\sum_{i=1}^{M}\sum_{j=1}^{N}|I(i,j) - I'(i,j)|^2 \quad (7)$$

The PSNR in terms of MSE is defined as,

$$PSN = 20\log_{10}\left(MAX\sqrt{MSE}\right) \quad (8)$$

Normalized correlation (NC) helps to judge the robustness of the watermarking scheme. Here we have extracted a watermark with NC= 0.9951.

We have performed different attacks on watermarked cover images for observing the behavior of the watermark. The robustness of the watermarking algorithm proposed in this workfor various types of distortions introduced in the imageisalso discussed in the preceding sections. The algorithm is found to be robust against different types of threats as evident from the results.

4.1 JPEG Compression Attacks

Most of the time image compression is done to reduce storage requirements and increase transmission speed. One of the most widely used techniques in this respect is the JPEG compression. The JPEG compression is realized with quality factors from 90%, 80%, and 70% to the watermarked cover image respectively. As a result of this attack, the cover image suffers from degradation and loss of a large amount of data as shown in Table 1. On the other hand, the extracted watermark is still identifiable.

Table. 1. Effect of JPEG compression on the watermarked image

JPEG compression ratio	PSNR	SSIM	NC
10%	33.833	0.8867	0.9775
20%	32.5867	0.8603	0.9448
30%	32.0243	0.8593	0.7312

4.2 Geometrical Attacks

Watermarked images after geometrical attacks are shown in Figure 5. Values of PSNR, SSIM, and NC for different types of geometrical attacks are recorded in Table 2.

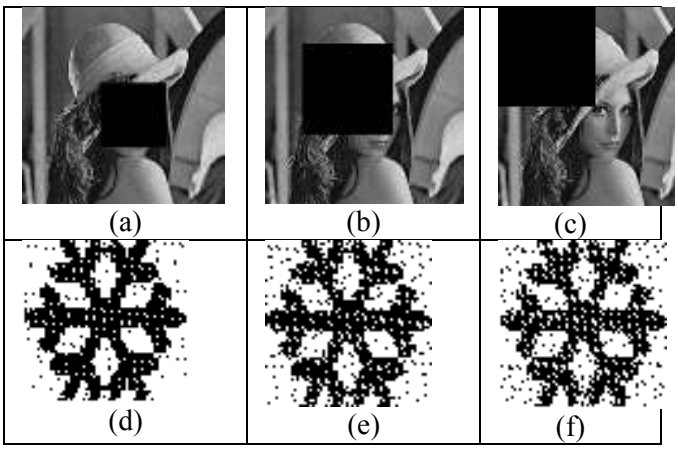


Fig. 5: Attacked watermarked image and extracted watermark after cropping (a) 10% cropping at the center, (b) 20% cropping, (c) 25% cropping

Table 2. Effect of geometrical attack on watermarked image

Geometric attacks	PSNR	SSIM	NC
Scaling (120%)	36.4094	0.9503	0.977
Scaling (110%)	36.2542	0.9479	0.9638
Scaling (90%)	35.5358	0.9532	0.8152
Scaling (80%)	34.5758	0.9562	0.682
Cropping (10%)	16.9294	0.7997	0.8824
Cropping (15%)	16.5129	0.7627	0.8479
Cropping (20%)	13.4736	0.6959	0.7874
Cropping (25%)	13.9105	0.6706	0.7553

4.3 Noise Attacks

Watermarked images after noise attacks are shown in Figure 6.

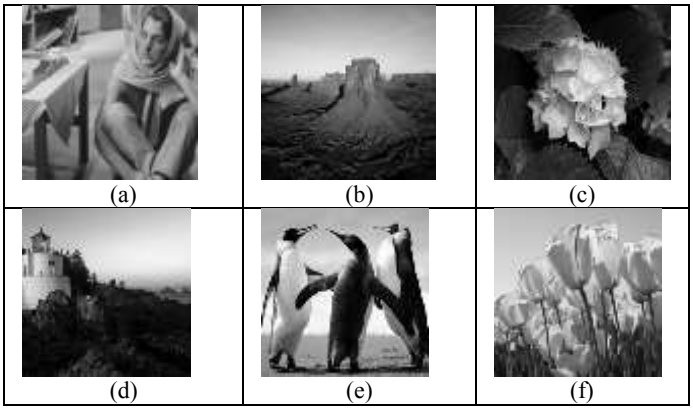
(a) (b)

Fig. 6: extracted watermark after noise attacks (a) salt and pepper noise (0.003), (b) Gaussian noise(0.001).

Values of PSNR, SSIM, and NC for different types of noise attacks on watermarked images are recorded in Table 3.

Table 3. Effect of noise attack on watermarked image

Noise attacks	PSNR	SSIM	NC
Salt and pepper (0.001)	31.9049	0.8834	0.9814
Salt and pepper (0.005)	27.0171	0.8085	0.9428
Salt and pepper (0.01)	24.5088	0.737	0.8905
Gaussian noise (0.001)	28.7543	0.7152	0.8959
Gaussian noise (0.002)	26.3778	0.6177	0.7919



(a) (b) (c) (d) (e) (f)

Fig. 7: Different cover images (a), (b), (c), (d), (e), (f)

We applied the same embedding and extracting process on different cover images. In every case, we got a normalized correlation of the watermark greater than 0.98. Various cover images with different covers is shown in Figure 7. PSNR, and SSIM of cover images for different covers is as recorded in Table 4.

Table. 4. Experimental results for different cover images

Cover images	PSNR	SSIM	NC
Image (a)	33.9542	0.9123	0.9951
Image (b)	35.008	0.8551	0.9946
Image (c)	35.8989	0.9302	0.9853
Image (d)	34.3563	0.8461	0.9927
Image (e)	33.5066	0.8606	0.9868
Image (f)	33.5066	0.8606	0.9951

4.4 Hardware Utilization Summary

As discussed earlier, FPGA consists of different numbers of Combinational Logic Blocks (CLBs), Lookup Tables (LUTs) & logic gates. While evaluating certain designs, FPGA uses a specific amount of components as per its requirement. Table 5 shows the summary of device components used. It presents a comparison between available components and used components; also percentage of usage is given.

Table. 5. Device Utilization Summary

Device Utilization Summary			
Slice Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	2,005	126,800	1%
Number used as Flip Flops	1,971		
Number used as Latches	0		
Number used as AND/OR logics	34		
Number of Slice LUTs	2,757	63,400	4%
Number used as logic	2,481	63,400	3%
Number used as Memory	211	19,000	1%
Number of occupied Slices	1,019	15,850	6%
Number of fully used LUT-FF pairs	1,462	3,125	46%
Number of RAMB36E1/FIFO36E1s	64	135	47%

5 Conclution

In the proposed work, frequency domain blind watermarking is efficiently implemented on a low-cost and reconfigurable Artix-7 FPGA platform using DCT. The computational complexity of DCT is less than other transforms and it sustains JPEG compression efficiently. Selected mid-frequency band coefficients are modified into positive or negative values to embed the watermark in an image. At the receiver end, watermark extraction is done by replacing the watermark bits to 1 or 0 as per the value of modified coefficients. Efficient extraction of the watermark image with NC=0.9951 without using an initial cover image in the extraction process is possible by the proposed method. The overall embedding process requires

approximately 4 to 5 seconds to encrypt and embed the watermark. Watermark is extracted and decrypted in 1 to 1.5 seconds. Results obtained are comparable to existing software-based approaches described in [1] and [2]. It provides a robust and indiscernible watermarking facility. This approach when combined with very good speed and efficient hardware utilization on FPGA makes it suitable for real-time application.

References:

[1] A. R. Yuliani and D. Rosiyadi, "A watermarking scheme based on DCT using HVS characteristic," *2015 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, Bandung, Indonesia, 2015, pp. 165-168, doi: 10.1109/IC3INA.2015.7377766.

[2] Üstübioğlu, Arda & Ulutas, Guzin & Ulutas, Mustafa, "DCT based image watermarking method with dynamic gain", *2015, 38th International Conference on Telecommunications and Signal Processing (TSP)*, Brno, Czech Republic, pp. 550-554, doi: 10.1109/TSP.2015.7296323.

[3] Bhaskar, T., and D. Vasumathi. "DCT Based Watermark embedding into mid frequency of DCT coefficients Using Luminance Component." *International Research Journal of Engineering and Technology (IRJET)*,vol 2, issue 3 (2015): 738-741.

[4] A. Bamatraf, R. Ibrahim and M. N. B. M. Salleh, "Digital watermarking algorithm using LSB," *2010 International Conference on Computer Applications and Industrial Electronics*, Kuala Lumpur, Malaysia, 2010, pp. 155-159, doi: 10.1109/ICCAIE.2010.5735066.

[5] Gomez-Coronel, S.L.; Moya-Albor, E.; Brieva, J.; Romero-Arellano, A. , "A Robust and Secure Watermarking Approach Based on Hermite Transform and SVD-DCT". *Appl.Sci.* 2023, *13*, 8430.

[6] A. Aniyan and J. Deepa, "Hardware implementation of a robust watermarking technique for digital images," *2013 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, Trivandrum, India, 2013, pp. 293-298, doi: 10.1109/RAICS.2013.6745490.

[7] Mittal, Sparsh, Saket Gupta, and Sudeb Dasgupta. "FPGA: An efficient and promising platform for real-time image processing applications." *National Conference on Research and Development In Hardware Systems (CSI-RDHS)*, Kolkata, India, 2008.

[8] Mohamed Ali Hajjaji, Mohamed Gafsi, Abdessalem Ben Abdelali, Abdellatif Mtibaa, "FPGA Implementation of Digital Images Watermarking System Based on Discrete Haar Wavelet Transform", *Security and Communication Networks*, vol. 2019, Article ID 1294267, 17 pages, 2019.

[9] Al-Gindy, Ahmed & Tawfik, Ayman & Ahmad, Hussain & Qahwaji, Rami, "A new blind image watermarking technique for dual watermarks using low-frequency band DCT coefficients." *Proceedings of the IEEE International Conference on Electronics, Circuits, and Systems., 2007. ICECS 2007*, Marrakech, Morocco , pp. 538 - 541. Doi: 10.1109/ICECS.2007.4511047..

[10] Choudhary, Nilesh Y., and Ravindra K. Gupta. "Partial Image Encryption based on Block wise Shuffling using Arnold Map." *International Journal of Computer Applications*, vol 97, issue 10, (2014).

[11] Digilent, "Nexys 4™ FPGA Board Reference Manual", revised September 6, 2013

[12] Documentation, Xilinx Device Drivers. "Product Specification, Xilinx." Inc., Jun (2004).

[13] "Xilinx EDK Concepts, Tools and Techniques", a hands-on guide to effective embedded system design, Oct 2012.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

- Ankita Patil carried out the simulation work.
- Vaishali V. Ingale, Fatema A. Dalal, Vanita Agarwal were responsible for ideating, formulating, guiding organizing and execution of research carried.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare.