Malaysian Journal of Mathematical Sciences 16(2): 289–316 (2022)
https://doi.org/10.47836/mjms.16.2.08

Malaysian Journal of Mathematical Sciences

Journal homepage: https://einspem.upm.edu.my/journal

Robustness of Modified Non-Separable Haar Wavelet Transform and Singular Value Decomposition for Non-blind Digital Image Watermarking

Razak, M. K. A.[1], Abdullah, K.[1], and Halim, S. A.[*1]

[1]*Center of Mathematics Studies, Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Malaysia*

E-mail: suhaila889@uitm.edu.my
[*]*Corresponding author*

Received: 9 September 2021
Accepted: 15 February 2022

Abstract

Security is a matter of significant concern for image media. An effective way to protect images is by digital watermarking. This paper introduces a non-blind digital watermarking scheme using modified non-separable Haar wavelet transform (NSHWT), singular value decomposition (SVD), Arnold's cat map, and Rabin-*p* cryptosystem to embed a binary watermark image into a color cover image. Aside from robustness, security is also prioritized in the scheme. High robustness is achieved using two transform domain techniques, discrete wavelet transform (DWT) and SVD, while security is heightened with the double encryption by Arnold's cat map scrambling and Rabin-*p* cryptosystem. A disadvantage of the discrete wavelet transform in digital image watermarking is the calculation complexity and the high cost of changing memory and time. The proposed algorithm uses a modified NSHWT instead of the traditional method of DWT. Therefore, the load of the process on the hardware can significantly be reduced while still maintaining the advantages of DWT. The algorithm is objectively evaluated in terms of imperceptibility, robustness, and embedding capacity for both binary and color watermarks, as well as efficiency. It has been concluded that the proposed scheme performs competently compared with other recent watermarking techniques based on DWT and SVD.

Keywords: digital image watermarking; robustness; DWT; SVD; NSHWT.

1 Introduction

The advancement of technology has made it easier to communicate or interact online. However, at the same time, it has also led to illegal distributions and unauthorized use by irresponsible individuals. To protect images, digital image watermarking can be applied to prevent or drastically reduce these cases effectively. Digital image watermarking embeds a watermark in an image, which often requires to be imperceptible by human observation, as well as robust against noise or distortion. Image watermarking is generally applied to protect the cover image or the secret image itself, depending on the intended application of the watermarking algorithm.

Watermark can be embedded in either the spatial domain or the frequency domain. Spatial domain watermarking is a straightforward method by directly adding the watermark to the cover image. Transform domain watermarking means the watermark is added to the transformed version of the cover image. After the watermark is added, the reverse transformation is applied to reconstruct the original image. Common examples of image transforming techniques are discrete wavelet transform (DWT) [22], discrete cosine transform (DCT) [2] and singular value decomposition (SVD) [7]. Watermarking in the spatial domain is fast and simple but lack the robustness to resist attacks. Watermarks in the transform domain are much more robust but significantly increase computing costs, therefore being slow and burdening the hardware where it is performed.

Watermarks need to be imperceptible or invisible. If the watermark can be seen by simple observation of the human eye, an attacker could directly remove the watermark without much loss of important information. If the watermark is invisible, assuming the attacker does not know the exact method used to embed the watermark, they would have to destroy it by image processing attacks and/or geometric attacks. Image processing attacks include adding noise, filtering, and compressing the image, aiming to destroy the watermark so that it cannot be detected anymore, while leaving the cover image intact. Geometric attacks include geometric transformations such as zooming, rotating, removing or adding pixels. Without the proper countermeasures, the watermark may fail to be detected in the extraction process [23].

The DWT transforms an image into four sub-bands, LL, LH, HL, and HH, showing the approximate image, vertical, horizontal, and diagonal, respectively. These sub-bands have properties that can be exploited in various ways to improve watermark imperceptibility, robustness, and/or security. It has excellent spatial localization, multi-resolution, excellent time-frequency analysis, good energy compaction, and is robust against signal processing attacks. The disadvantages of DWT are less potent against geometric attacks and computational complexity [4]. The non-separable Haar wavelet transform (NSHWT) and SVD are applied to cover those weaknesses. The NSHWT [6] is a faster and more efficient alternative to the DWT. The NSHWT has significant advantages over the DWT in hardware utilization, maximum speed, power consumption. Among other 2D-DWT structures, it has advantages in computing time, internal memory usage, and latency.

By using more than one transform domain techniques in a watermarking scheme, computation costs would increase, but a very significant improvement to the watermark robustness can be made. For example, a hybridization of DWT with SVD can further make the watermark more robust because SVD has high robustness against both geometric and signal processing attacks [4].

Roy et al. [21] applied SVD-DWT hybrid for non-blind color image waterkmarking, and Ahmadi et al. [1] applied SVD-DWT for blind color image watermarking. The application of DWT-SVD makes both watermarking schemes very robust against many types of image processing, geometric, and hybrid attacks.

Watermarking algorithms also needs a robust encryption to be secure. Algorithms are often made public, so simply knowing the correct technique allows an irresponsible third party to remove the watermark directly. In short, the watermarks need to be encrypted with secret parameters and should only be directly removed by knowing these parameters beforehand. The most common method of encryption in image watermarking is scrambling. The pixels of an image have their positions scrambled so that the resulting image will look meaningless.

Another way to encrypt images is by applying cryptosystems. Depending on the algorithm, the method of application may differ, such as directly encrypting the image with the cryptosystem, or certain variable parameters in the watermarking process. The Rabin-p cryptosystem [5] is a Rabin-like cryptosystem applied in the proposed scheme. To overcome the critical shortcoming of the Rabin cryptosystem [20], which is the 4-to-1 decryption outputs, the authors avoided the use of Jacobi symbol, message redundancy technique or sending extra information [14]. The Rabin-p cryptosystem has been applied on Internet of Things (IoT) devices, which shows low energy usage and fast execution runtime, and better performance compared to other Rabin cryptosystem variants. Therefore, the Rabin-p cryptosystem could improve the security of the proposed scheme at a low cost. Security is a rapidly growing concern as the development of quantum computers continue, which can compute at an unbelievably higher rate than classical computers, and thus can quickly solve problems considered very difficult [16].

This article proposes a new non-blind watermarking algorithm applying modified NSHWT, SVD, Arnold's cat map and Rabin-p cryptosystem. The use of modified NSHWT and SVD together helps the watermark be imperceptible while also maintaining high levels of robustness. At the same time, the two layers of encryption further keep the secret image secure from the third party with malicious intent. Table 1 shows the main advantage and disadvantage of the techniques used to manipulate the images.

Table 1: Advantage and disadvantage of techniques.

Transform Domain	Advantage	Disadvantage
DWT	Watermark affects image locally instead of whole image [10]	Longer computation time and cost compared to other transforms [17]
SVD	Small changes in image have little effect on the singular values [11]	False positive problem
Arnold's cat map	Effective encryption to hide data and improve robustness by distributing error bit [12]	Image has to be square

The details of each applied technique is briefly explained in the theoretical background section before going through the methodology of the proposed algorithm and its evaluation. The algorithm is evaluated in terms of imperceptibility, robustness, speed, and capacity for both binary and color watermarks. The results are then compared and analyzed to better understand the proposed algorithm.

1.1 Main Contributions

This paper presents a robust and secure watermarking algorithm for sending secret messages. The cover image is decomposed by modified NSHWT to obtain four sub-bands LL, LH, HL, and HH. By using SVD, the singular values of the watermark image is embedded in the singular values of the LL sub-band. The watermark is also encrypted by Arnold's cat map scrambling and Rabin-p cryptosystem.

The DWT is a very common and popular method in the field of watermarking, as it has attractive properties such as multi-resolution representation, spatial localization, and robust embedded watermark. Transform domain techniques generally have much higher time complexity and computation costs compared to spatial domain techniques. Although the DWT has the downsampling process that makes it more efficient, it requires high transposition memory, reaching at least $2N$ for a square matrix of order N [6]. The NSHWT is a method to perform the DWT technique on 2-D signals more efficiently, by removing the need for transposition memory during calculations [6]. The NSHWT is modified in the proposed algorithm to improve the technique in the context of image transforming to reduce the data lost by integer representation of images.

Encryption in watermarking schemes often only include scrambling of the image, with certain parameters sent as the private key. The security can further be improved by incorporating the use of a cryptosystem for encryption. Most cryptosystems are rigorously tested to ensure their security is strong, so the process of decryption is very difficult without the correct keys. However, encrypting an image pixel-by-pixel may be inefficient, so it is encouraged to apply the cryptosystem indirectly. In the proposed scheme, the Rabin-p cryptosystem is used to encrypt the two parameters used in Arnold's cat map scrambling.

2 Theoretical Background

The proposed watermarking algorithm uses four different techniques. The cover image applies two transform domain techniques, modified non-separable Haar wavelet transform (NSHWT) and singular value decomposition (SVD). Two encryption techniques are used for the secret image or watermark image; Arnold's cat map for image scrambling and Rabin-p cryptosystem for encrypting the scrambling parameters The main application of the proposed scheme is for sending secret messages. The application of modified NSHWT and SVD greatly improves the robustness of the watermark against multiple attacks. Other than that, the encryption by scrambling and a cryptosystem makes it hard to extract the secret message, and the transform domain techniques makes the embedded data robust and greatly reduce loss of data when under attack. False positive problem is also not present as the intended information is inside the secret image, and cannot simply be forged.

2.1 Modified Non-Separable Haar Wavelet Transform

The non-separable Haar wavelet transform (NSHWT) [6] is a derivative of the common transform domain technique, discrete wavelet transforms (DWT). The original DWT converts a 1-D signal into two sub-bands, low (L) and high (H). For cases of 2-D signals such as digital images, the operation is done row-by-row, followed by column-by-column, or vice versa, finally resulting in four sub-bands, low-low (LL), low-high (LH), high-low (HL) and high-high (HH). NSHWT is considered non-separable because the row and column operations are combined, so the whole 2-D signal is transformed in one step. Figure 1 shows the results of the horizontal and vertical DWT transform and how the NSHWT transforms it in one step.

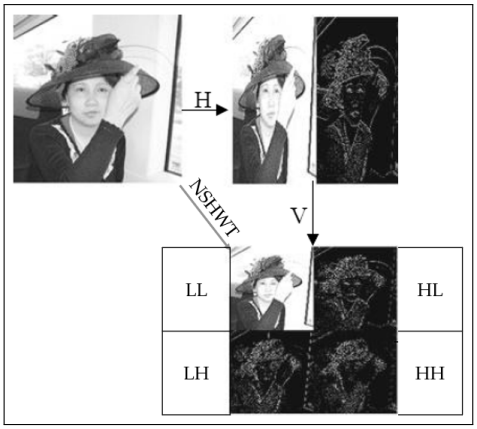


Figure 1: DWT and NSHWT transformation.

The method is proven to be highly efficient compared to other 2-D DWT architectures, including the original DWT [6]. The computing time of the NSHWT is much faster, and no internal memory is needed. The equations below show the original NSHWT operations:

$$
\begin{aligned}
LL &= \frac{x_{11}+x_{21}+x_{12}+x_{22}}{2} \\
LH &= \frac{x_{11}-x_{21}+x_{12}-x_{22}}{2} \\
LH &= \frac{x_{11}+x_{21}-x_{12}-x_{22}}{2} \\
LH &= \frac{x_{11}-x_{21}-x_{12}+x_{22}}{2}
\end{aligned}
\qquad
\begin{aligned}
x_{11} &= \frac{LL+LH+HL+HH}{2} \\
x_{21} &= \frac{LL-LH+HL-HH}{2} \\
x_{12} &= \frac{LL+LH-HL-HH}{2} \\
x_{22} &= \frac{LL-LH-HL+HH}{2}.
\end{aligned}
$$

$$(a) \qquad\qquad (b)$$

where $\begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix}$ is a submatrix of an image matrix, which has integer values, and LL, LH, HL and HH are the corresponding pixels formed by the transform. The order and location at which the submatrices are chosen must be the same as the order and location where each sub-band pixel is put. The equations in (a) is for the forward transform, while the equations in (b) is for the

inverse transform. For use in transforming images, a small modification is made to the NSHWT, resulting in Equations (1) to (4).

$$LL = \frac{x_{11} + x_{21} + x_{12} + x_{22}}{4} \tag{1}$$

$$LH = \frac{x_{11} - x_{21} + x_{12} - x_{22}}{2} \tag{2}$$

$$LH = \frac{x_{11} + x_{21} - x_{12} - x_{22}}{2} \tag{3}$$

$$LH = \frac{x_{11} - x_{21} - x_{12} + x_{22}}{2}. \tag{4}$$

On the other hand, the reverse operation is as shown in Equations (5) to (8):

$$x_{11} = \frac{2LL + LH + HL + HH}{2} \tag{5}$$

$$x_{21} = \frac{2LL - LH + HL - HH}{2} \tag{6}$$

$$x_{12} = \frac{2LL + LH - HL - HH}{2} \tag{7}$$

$$x_{22} = \frac{2LL - LH - HL + HH}{2}. \tag{8}$$

For the forward operation, the denominator in operation for the LL sub-band pixel (1) is increased to 4, and for the reverse operation, the LL coefficients (5-8) are multiplied by 2. This slight modification makes the LL sub-band a closer approximation to the original image. This is achieved by avoiding pixel saturation, which happens when the pixel value goes past the maximum value, 255 for an 8-bit pixel, as commonly used for images, including the cover image used in this article. Since the operation on the numerator is the sum of four-pixel values, dividing by four will ensure the resulting value is never above the maximum. Additionally, only the calculation for the LL sub-band is modified to provide less data is lost in the transformation due to the pixel values being an integer. The difference before and after the modification is shown in Figure 2. During the embedding process, the LL sub-band will be chosen for secret image embedding, as it gives the best watermark robustness [13]. In the case of binary secret images, the watermark is embedded only on the green channel of the red, green, and blue (RGB) cover image for increased robustness [25].

Figure 2: Before and after NSHWT modification.

2.2 Singular Value Decomposition

The singular value decomposition (SVD) is another popular transforming choice for image watermarking. SVD decomposes a rectangular matrix into three singular values of the same size as the original matrix. These singular values are commonly referred to as U, S, and V, as shown in Equation (9).

$$A = USV^T \tag{9}$$

where A is a rectangular matrix of size $m \times n$, U is an orthogonal matrix of size $m \times m$, S is a diagonal matrix of size $m \times n$, and V^T is the transpose of an orthogonal matrix V of size $n \times n$. The diagonal values in the S singular value are highly stable, so slight changes to the values will not have noticeable effects on the visual perception of the cover image, therefore making the watermark imperceptible when embedded in the S singular value [3].

2.3 Arnold's Cat Map

Arnold's cat map [18] is a type of chaotic scrambling and is used to encrypt the image efficiently. Upon scrambling an image, it will turn into an image that seems meaningless, as shown in Figure 3. The equation is as shown in Equation (10) [19].

$$\begin{pmatrix} x' \\ y' \end{pmatrix} \equiv \begin{pmatrix} 1 & a \\ b & ab+1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \pmod N, \tag{10}$$

where (x, y) is the original position of a pixel, (x', y') is the new position of the pixel, a and b are variable and integer scrambling parameters. The reverse of the scrambling is as in Equation (11).

$$\begin{pmatrix} x \\ y \end{pmatrix} \equiv \begin{pmatrix} ab+1 & -a \\ -b & 1 \end{pmatrix} \begin{pmatrix} x' \\ y' \end{pmatrix} \pmod{N}, \tag{11}$$

Each pixel will be moved to a unique position and can be returned to it's original position by applying the inverse operation. However, it is first required to know the correct parameters a and b for the scrambling, which is kept secret. The results of a scrambling can be seen in Figure 3.

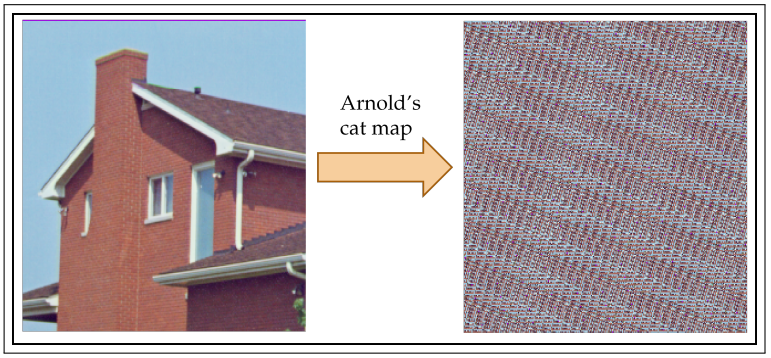


Figure 3: Arnold's cat map scrambling.

2.4 Rabin-p Cryptosystem

The Rabin-p cryptosystem [5] is the method chosen to encrypt the scrambling parameters a and b. The name Rabin-p comes from the fact that the cryptosystem is based on the Rabin cryptosystem [20] and only the secret key p is used in the decryption process. This makes the decryption process much faster, along with other advantages over other variants of the Rabin cryptosystem. Similar to most cryptosystems, the Rabin-p cryptosystem follows three processes, key generation, encryption, and decryption. Algorithm 1, 2 and 3 details each of the processes of the Rabin-p cryptosystem.

Algorithm 1: Rabin-p Key Generation Algorithm

Input: The size k of the security parameter.
Output: The public key $N = p^2q$ and the private key p.
1 Choose two random and distinct primes p and q such that $2^k < p, q < 2^{k+1}$ satisfy $p, q \equiv 3 \pmod 4$.
2 Compute $N = p^2q$.
3 Return the public key N and the private key p.

Using two random and distinct primes p and q which satisfies the two aforementioned conditions, the public key N is generated. Both the public key N and private key p is required for the next steps. The prime q, however, will no longer be used for the encryption or decryption process. As a note, some numbers can easily be factored by certain factoring algorithms, so care must be taken to not choose primes that are more vulnerable to attacks [8].

Algorithm 2: Rabin-p Encryption Algorithm

Input: The plaintext m and the public key N.
Output: A ciphertext c.
1 Choose plaintext $0 < m < 2^{2k-1}$ such that $gcd(m, N) = 1$.
2 Compute $c \equiv m^2 \pmod{N}$.
3 Return the ciphertext c.

With a plaintext m that satisfies the two conditions, one modulo operation is performed to obtain the ciphertext c.

Algorithm 3: Rabin-p Decryption Algorithm

Input: A ciphertext c and the private key p.
Output: The plaintext m.
1 Compute $w \equiv c \pmod{p}$.
2 Compute $m_p \equiv w^{\frac{p+1}{4}} \pmod{p}$.
3 Compute $i = \frac{c - {m_p}^2}{p}$.
4 Compute $j \equiv \frac{i}{2m_p} \pmod{p}$.
5 Compute $m_1 = m_p + jp$.
6 If $m_1 < 2^{2k-1}$, then return $m = m_1$. Else, return $m = p^2 - m_1$.

In the decryption process, only the ciphertext c and the private key p are needed to decrypt the ciphertext. The use of only the private key p is why it is called the Rabin-p cryptosystem. The final result depends on whether the condition $m_1 < 2^{2k-1}$ is fulfilled or not.

In summary, the scrambling parameters a and b are taken in the embedding process to be encrypted and obtain ciphertexts c_1 and c_2. After extracting the scrambled secret image, c_1 and c_2 is first decrypted before being used to descramble the scrambled secret image.

3 Methodology

The experiments use color cover images and binary and color secret images obtained from the USC-SIPI Image Database website. The cover image has 512×512 pixels in while the secret image has 256×256 pixels, both in *.tif format. Lena is used as a cover image while House, Jellybeans, and Fruits are the secret images, as shown in Figure 4 and Figure 5. All the processes in the methodology section are simulated in MATLAB r2019a.

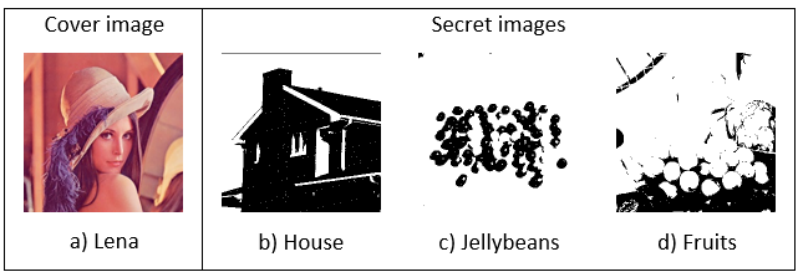


Figure 4: Binary test images.

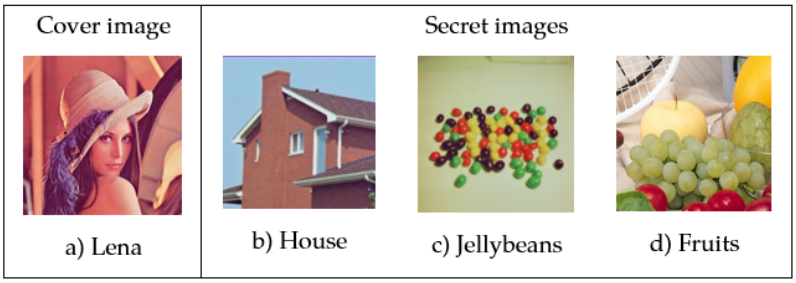


Figure 5: Color test images.

3.1 Embedding Process

Figure 6 shows the flowchart of the embedding process of the proposed algorithm. In the embedding process, the secret image is inserted into the cover image via the LL sub-band of the modified NSHWT coefficient.

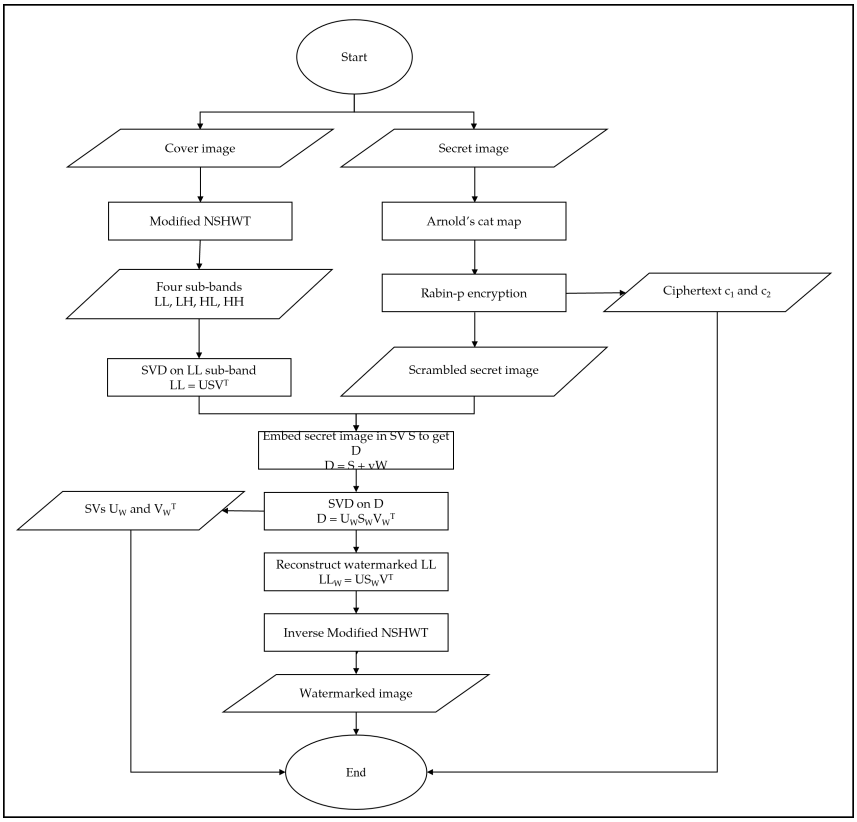


Figure 6: Embedding process.

When embedding data on an image, it has to be scaled by multiplying with a suitable scaling factor, otherwise, it will not be imperceptible or data may be lost due to pixel overflow. The embedding operation is represented as shown in Equation (12).

$$E = C + vS, \tag{12}$$

where E is the embedded image, C is the LL sub-band of the cover image, S is the scrambled secret image, and v is the scaling factor. The steps for the embedding process are as follows:

Step 1) Read the cover image, which is a square $m \times m$ matrix and secret image, square with dimensions half of the cover image.

Step 2) Apply modified NSHWT (Equations (1-4)) to transform the cover image into four sub-bands LL, LH, HL, HH.

Step 3) Apply SVD (Equation (9)) to decompose the LL sub-band into three singular values, U, S, VT.

Step 4) Apply Arnold's cat map transform (Equation (10)) on the secret image to create a scrambled secret image, W.

Step 5) Encrypt the control parameters of the Arnold's cat map transform, a and b with Algorithm 1 and 2, to obtain c_1 and c_2.

Step 6) The scrambled secret image W is added to the S singular value using Equation (12) with scaling factor, $v = 0.1$ to obtain D. The secret binary image is embedded into the green RGB channel. In contrast, the color secret image is embedded in each corresponding channel.

Step 7) Apply SVD (Equation (9)) on the D matrix to obtain three singular values U_w, S_w, ${V_w}^T$.

Step 8) Construct the watermarked LL sub-band, LL_w by multiplying the modified singular values, U, S_w, V^T. The U_w and ${V_w}^T$ singular values are saved for use during extraction process.

Step 9) Apply inverse NSHWT (Equations (5-8)) with the watermarked sub-band LL_w to create the watermarked image, which is saved or sent along with the ciphertext c_1 and c_2, and singular values U_w and ${V_w}^T$.

Figure 7 shows the result of the embedding on a Baboon cover image. It can be observed that there is hardly any difference between the two images. Therefore the watermark is said to be imperceptible.

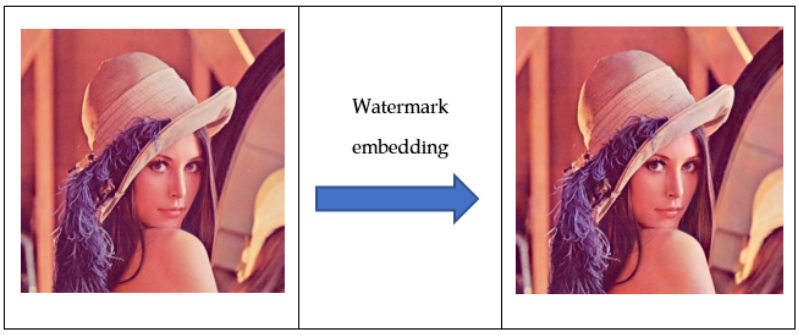


Figure 7: Embedding results.

3.2 Extraction Process

Figure 8 shows the flowchart of the process of the proposed algorithm. In the extraction process, the secret image is extracted from the possibly attacked watermarked image. The original LL sub-band and scrambled secret image are obtained using Equations (13) and (14).

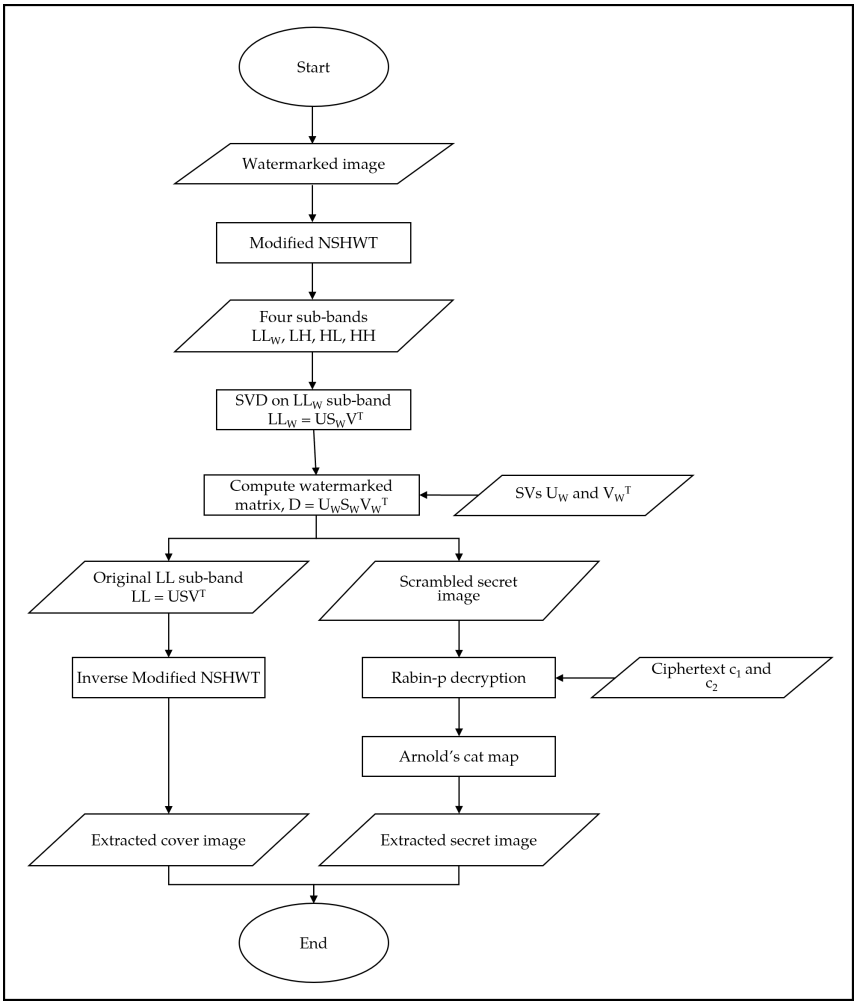


Figure 8: Extraction process.

i. Extracted cover image

$$C = E - vS, \tag{13}$$

ii. Extracted secret image

$$S = \frac{E - C}{v}, \tag{14}$$

The steps for the extraction process are described below:

Step 1) Read the watermarked image.

Step 2) Apply modified NSHWT (Equations (1-4)) to transform the watermarked image into four sub-bands LL_w, LH, HL, HH.

Step 3) Apply SVD (Equation (9)) to the LLw sub-band to obtain U, S_w, V^T, and then compute the D matrix by multiplying the singular values U_w, S_w and ${V_w}^T$.

Step 4) Extract the embedded image from D to get the extracted LL sub-band and scrambled secret image using Equations (13) and (14) respectively.

Step 5) Apply inverse modified NSHWT (Equations (5-8)) on the four DWT coefficients to reconstruct the extracted cover image.

Step 6) Decrypt the ciphertext c_1 and c_2 using Algorithm 3 to recover the control parameters a and b.

Step 7) Descramble the scrambled secret image using Arnold's cat map (Equation (11)) with the control parameters a and b to obtain the extracted secret image.

Figure 9 shows the result of before and after extracting the secret image from the watermarked image:

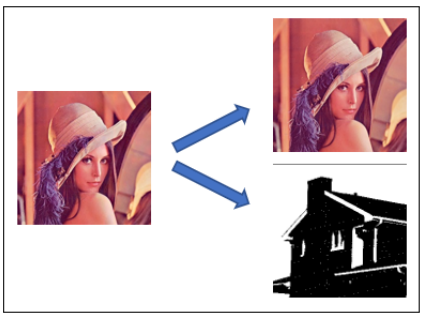


Figure 9: Extracted cover image and extracted secret image.

3.3 Watermark Evaluation

The watermarking algorithm is evaluated based on robustness and speed. Robustness is calculated with three different metrics, which are peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and normalized correlation (NC). The results are then compared with two recent algorithms that apply DWT and SVD for color Lena images. All calculations and methodology are simulated in MATLAB r2019a. PSNR is widely used for quality assessment on images and is the most straightforward metric to compute. Let f be a reference image and g be a test image, both of sizes $M \times N$. The formula for calculating PSNR is as shown in Equation (15) [9],

$$PSNR(f,g) = 10\log_{10}\left(\frac{255^2}{MSE(f,g)}\right), \tag{15}$$

The operation for computing MSE is as in Equation (16) [9]:

$$MSE(f,g) = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}(f_{xy} - g_{xy})^2, \tag{16}$$

where f_{xy} and g_{xy} are the pixel values at the (x, y) coordinate in the respective images; however, PSNR does not clearly show the correlation between two images, so different metrics must also be used. SSIM measures the similarity between two images. For SSIM, the formula is as shown in Equation 17 [15]:

$$SSIM(f,g) = \frac{(2\mu_f\mu_g + c_1)(2\sigma_{fg} + c_2)}{({\mu_f}^2 + {\mu_g}^2 + c_1)({\sigma_f}^2 + {\sigma_g}^2 + c_2)}, \tag{17}$$

where μ_x and μ_y are averages of image x and image y respectively, ${\sigma_x}^2$ and ${\sigma_y}^2$ are variances of image x and image y respectively, σ_{xy} is the covariance of image x and image y, $c_1 = (k_1 L)^2$ and $c_2 = (k_2 L)^2$ are two variables to stabilize the division with the weak denominator, where L is the dynamic range of pixel values and $k_1 = 0.03$ and $k_2 = 0.01$. Lastly, NC measures the correlation between two images. The formula to compute NC is shown in Equation (18) [24]:

$$NC(W,W') = \frac{\sum_{x=1}^{M}\sum_{y=1}^{N}[W(x,y) \times W'(x,y)]}{\sqrt{\sum_{x=1}^{M}\sum_{y=1}^{N}[W(x,y)]^2}\sqrt{\sum_{x=1}^{M}\sum_{y=1}^{N}[W'(x,y)]^2}}, \tag{18}$$

where W and W' are the watermark image and extracted watermark image respectively and sizes $M \times N$, and $W(x, y)$ and $W'(x, y)$ are the pixel values at (x, y) in the binary secret image. For the color secret image, an extra dimension is added to the equation, as shown in Equation (19):

$$NC(W,W') = \frac{\sum_{z=1}^{3}\sum_{x=1}^{M}\sum_{y=1}^{N}[W(x,y,z) \times W'(x,y,z)]}{\sqrt{\sum_{z=1}^{3}\sum_{x=1}^{M}\sum_{y=1}^{N}[W(x,y,z)]^2}\sqrt{\sum_{z=1}^{3}\sum_{x=1}^{M}\sum_{y=1}^{N}[W'(x,y,z)]^2}}, \tag{19}$$

where $W(x, y)$ and $W'(x, y)$ are the pixel values at (x, y) in the z color channel. Before extracting the secret image, the watermarked image is first attacked with six image processing attacks and four geometric attacks. The closer the value of SSIM and NC to 1, the more robust and imperceptible the watermark is.

For robustness evaluation, x represents the original watermark image, and y represents the extracted watermark image, possibly distorted or attacked. Additionally, a speed test is also conducted to show the difference in efficiency between the modified NSHWT and the original DWT. The embedding and extraction process is executed in MATLAB, and the process is timed in the MATLAB profiler. The maximum embedding capacity of the scheme is also computed.

4 Experiment Results

This chapter shows and discusses the results of the calculations. The proposed watermarking algorithm is evaluated in terms of imperceptibility, robustness, speed, and capacity.

4.1 Watermark Imperceptibility

The imperceptibility of both the binary and color watermarks are discussed in this subsection. As there is a lack of references for similar methods embedding color watermarks, only the binary watermarks is compared with other algorithms. Table 2 shows the imperceptibility of the binary and color watermarks based on PSNR, SSIM, and NC, and Table 3 shows the comparison of watermark imperceptibility on Lena image between the proposed scheme and reference algorithms.

Table 2: Binary-attacked watermarked images and extracted watermarks.

Attacks	**PSNR**		**SSIM**		**NC**	
	Color	**Binary**	**Color**	**Binary**	**Color**	**Binary**
House	35.2295	66.9004	0.9957	1.0000	0.9995	1.0000
Jellybeans	34.3463	61.8698	0.9956	1.0000	0.9994	1.0000
Fruits	34.2993	62.6059	0.9953	1.0000	0.9994	1.0000

Based on Table 2, the PSNR value of binary watermark is near double of the color watermarks'. However, the value of SSIM and NC is still high for the color watermark, so the watermark is still imperceptible, despite the degradation of image quality.

Table 3: Imperceptibility comparison with Lena cover image.

Metrics	Ahmadi et al. [1]	Roy et al. [21]	Proposed
PSNR	53.1365	51.1464	66.9004
SSIM	0.9952	-	1.0000

Based on Table 3, the proposed scheme offers better image quality than both reference algorithms in terms of PSNR and SSIM when a binary watermark is embedded.

4.2 Binary Watermark Robustness

The attacked watermarked image and extracted binary watermarks are shown in Table 4. The robustness of the watermark is measured by calculating the PSNR, SSIM, and NC between the original secret image and the extracted secret image. Table 5 shows the robustness against six image processing attacks and four geometric attacks and extraction without attack.

Table 4: Binary-attacked watermarked images and extracted watermarks.

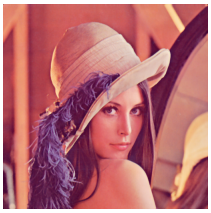
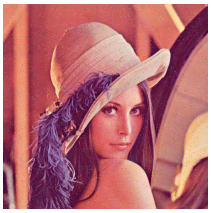
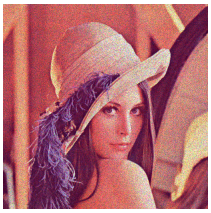
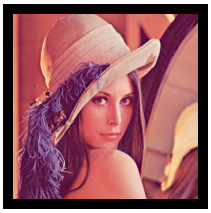
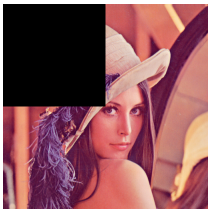
No attacks	Extracted watermark	Median filter 3x3	Extracted watermark	Histogram equalization	Extracted watermark
					
Gamma correction 0.8	Extracted watermark	JPEG compression 50 QF	Extracted watermark	Salt & pepper 0.01	Extracted watermark
					
Speckle noise 0.05	Extracted watermark	Rotation 10 degrees anti-clockwise	Extracted watermark	Cropping 20% frame	Extracted watermark
					
Scaling 50%	Extracted watermark	Cutting 25% top-left	Extracted watermark		
					

Table 5: Robustness evaluation - Binary watermark.

Attacks	PSNR				SSIM				NC			
	House	Jellybeans	Fruits	Average	House	Jellybeans	Fruits	Average	House	Jellybeans	Fruits	Average
No attack	80.8549	81.2441	80.8549	80.9846	1.0000	1.0000	1.0000	1.0000	0.9996	0.9997	0.9996	0.9997
Median filter 3x3	72.9310	72.7931	72.9310	72.8850	0.9999	0.9999	0.9999	0.9999	0.9978	0.9979	0.9978	0.9978
Histogram equalization	76.8508	77.9071	76.8508	77.2029	1.0000	1.0000	1.0000	1.0000	0.9991	0.9994	0.9991	0.9992
Gamma correction 0.8	72.4217	71.9499	72.4217	72.2644	0.9999	0.9999	0.9999	0.9999	0.9976	0.9975	0.9976	0.9975
JPEG compression 50 QF	72.8912	72.4935	72.8912	72.7586	0.9999	0.9999	0.9999	0.9999	0.9978	0.9978	0.9978	0.9978
Salt & pepper 0.01	75.4320	77.3747	75.4320	76.0796	1.0000	1.0000	1.0000	1.0000	0.9988	0.9993	0.9988	0.9989
Speckle noise 0.05	76.0837	76.2524	76.0837	76.1399	1.0000	1.0000	1.0000	1.0000	0.9989	0.9991	0.9989	0.9990
Rotation 10 degrees	75.0899	75.6886	75.0899	75.2895	1.0000	1.0000	1.0000	1.0000	0.9987	0.9989	0.9987	0.9988
Cropping 20% frame	71.5680	71.0842	71.5680	71.4068	0.9999	0.9999	0.9999	0.9999	0.9970	0.9969	0.9970	0.9970
Scaling 0.5	72.6783	72.6407	72.6783	72.6658	0.9999	0.9999	0.9999	0.9999	0.9977	0.9978	0.9977	0.9977
Cutting 25% top-left	71.7472	71.3960	71.7472	71.6301	0.9999	0.9999	0.9999	0.9999	0.9971	0.9971	0.9971	0.9971

Table 6: Robustness comparison - Binary watermark.

Attacks	NC		
	Ahmadi et al. [1]	Roy et al. [21]	Proposed scheme
No attack	**1.0000**	0.9992	0.9997
Median filter 3x3	**1.0000**	0.9796	0.9978
Histogram equalization	**1.0000**	0.9233	0.9992
Gamma correction 0.8	**1.0000**	-	0.9975
JPEG compression 50 QF	0.8987	0.9831	**0.9978**
Salt & pepper 0.01	-	0.9715	**0.9989**
Speckle noise 0.05	0.9962	0.9625	**0.9990**
Rotation 10 degrees	0.9873	-	**0.9988**
Cropping 20% frame	**1.0000**	-	0.9970
Scaling 0.5	**1.0000**	0.9655	0.9977
Cutting 25% top-left	**1.0000**	-	0.9971

Based on the average PSNR and average SSIM values in Table 5, the watermark quality drops the most by the cropping attack. For the NC values, the minimum average result is on the cropping attack, at 0.9970, which is still reasonably high. Table 6 shows the comparison of two SVD and DWT algorithms.

The proposed scheme gives better results for the four attacks; JPEG compression attack, salt and pepper noise attack, speckle noise attack, and rotation attack, with a less value but close difference for the rest of the attacks, as visualized in Figure 10. Therefore, it can be said that the watermark in the proposed algorithm has successfully achieved high levels of robustness.

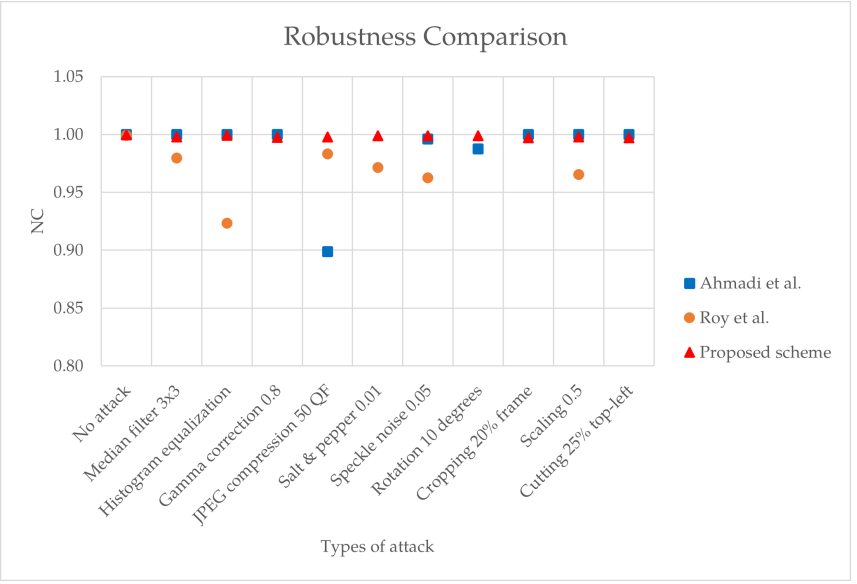


Figure 10: Binary watermark robustness comparison.

4.3 Color watermark robustness

The attacked watermarked image and extracted color watermarks are shown in Table 7. Table 8 shows the robustness of the embedded color secret images. Based on the average PSNR and SSIM values, the watermark quality drops more for histogram equalization, speckle noise, rotation, and cutting attacks. For the NC values, the minimum average result is on the cutting attack, at 0.9700, which is still reasonably high. Table 9 shows the comparison of two SVD and DWT algorithms.

The proposed scheme gives better results for the three attacks JPEG compression attack, salt and pepper noise attack, and rotation attack, yet still, a close difference against the other algorithms on different attacks, as visualized in Figure 11. Therefore, the proposed algorithm has successfully achieved satisfactory levels of robustness for color secret images.

Malaysian J. Math. Sci. 16(2): 289–316 (2022) 289 - 316

Table 7: Color - Attacked watermarked images and extracted watermarks.

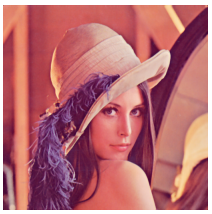
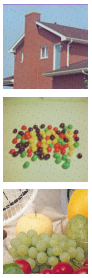
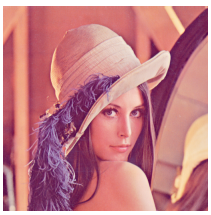
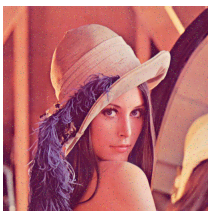
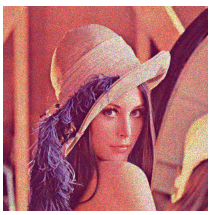
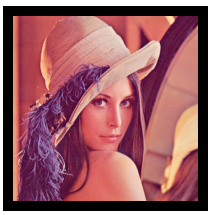
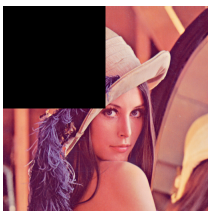
No attacks	Extracted watermark	Median filter 3x3	Extracted watermark	Histogram equalization	Extracted watermark
					
Gamma correction 0.8	Extracted watermark	JPEG compression 50 QF	Extracted watermark	Salt & pepper 0.01	Extracted watermark
					
Speckle noise 0.05	Extracted watermark	Rotation 10 degrees anti-clockwise	Extracted watermark	Cropping 20% frame	Extracted watermark
					
Scaling 50%	Extracted watermark	Cutting 25% top-left	Extracted watermark		
					

Table 8: Robustness evaluation - Color watermark.

Attacks	PSNR				SSIM				NC			
	House	Jellybeans	Fruits	Average	House	Jellybeans	Fruits	Average	House	Jellybeans	Fruits	Average
No attack	51.9258	49.1638	46.3106	49.1334	0.9997	0.9990	0.9991	0.9993	0.9999	1.0000	1.0000	1.0000
Median filter 3x3	26.2784	26.2225	25.6289	26.0433	0.9173	0.9129	0.9089	0.9130	0.9966	0.9972	0.9968	0.9969
Histogram equalization	17.5359	15.2457	16.6605	16.4807	0.8442	0.5240	0.6893	0.6858	0.9960	0.9953	0.9938	0.9950
Gamma correction 0.8	26.0021	25.0829	24.7015	25.2622	0.9387	0.8847	0.8888	0.9041	0.9979	0.9980	0.9979	0.9979
JPEG compression 50 QF	29.1247	29.3948	29.1461	29.2219	0.9394	0.9380	0.9457	0.9410	0.9982	0.9986	0.9986	0.9985
Salt & pepper 0.01	25.1397	25.4916	25.7976	25.4763	0.9017	0.8958	0.9039	0.9005	0.9956	0.9967	0.9969	0.9964
Speckle noise 0.05	18.0454	19.2311	18.8273	18.7013	0.6803	0.6412	0.6630	0.6615	0.9778	0.9863	0.9847	0.9829
Rotation 10 degrees	18.4918	15.3584	16.0058	16.6187	0.7724	0.6367	0.7214	0.7102	0.9916	0.9910	0.9860	0.9895
Cropping 20% frame	24.0560	22.8341	22.1109	23.0003	0.8894	0.8350	0.8602	0.8615	0.9959	0.9955	0.9951	0.9955
Scaling 0.5	25.3615	25.2470	24.6896	25.0994	0.8978	0.8936	0.8919	0.8944	0.9958	0.9965	0.9961	0.9961
Cutting 25% top-left	12.3099	13.6730	17.4847	14.4892	0.3250	-0.1757	0.6960	0.2818	0.9483	0.9740	0.9877	0.9700

Table 9: Robustness comparison - Color watermark.

Attacks	NC		
	Ahmadi et al. [1]	Roy et al. [21]	Proposed scheme
No attack	1.0000	0.9992	1.0000
Median filter 3x3	1.0000	0.9796	0.9969
Histogram equalization	1.0000	0.9233	0.9950
Gamma correction 0.8	1.0000	-	0.9979
JPEG compression 50 QF	0.8987	0.9831	0.9985
Salt & pepper noise 0.01	-	0.9715	0.9964
Speckle noise 0.05	0.9962	0.9625	0.9829
Rotation 10 degrees	0.9873	-	0.9895
Cropping 20% frame	1.0000	-	0.9955
Scaling 0.5	1.0000	0.9655	0.9961
Cutting 25% top-left	1.0000	-	0.9700

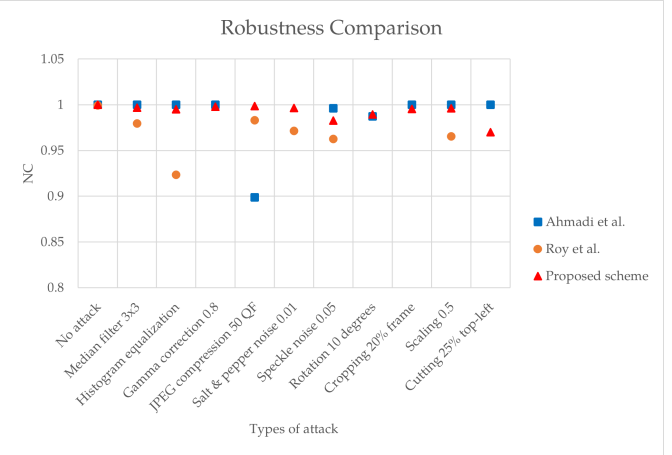


Figure 11: Color watermark robustness comparison.

4.4 Robustness Comparison for Binary Watermark vs. Color Watermark

The average values for PSNR, SSIM, and NC between the extracted secret image and original secret image for both types of secret images is shown in Table 10.

Table 10: Binary watermark vs color watermark robustness.

Attacks	Average PSNR		Average SSIM		Average NC	
	Binary	Color	Binary	Color	Binary	Color
No attack	80.9846	49.1334	1.0000	0.9993	0.9997	1.0000
Median filter 3x3	72.8850	26.0433	0.9999	0.9130	0.9978	0.9969
Histogram equalization	77.2029	16.4807	1.0000	0.6858	0.9992	0.9950
Gamma correction 0.8	72.2644	25.2622	0.9999	0.9041	0.9975	0.9979
JPEG compression 50 QF	72.7586	29.2219	0.9999	0.9410	0.9978	0.9985
Salt & pepper 0.01	76.0796	25.4763	1.0000	0.9005	0.9989	0.9964
Speckle noise 0.05	76.1399	18.7013	1.0000	0.6615	0.9990	0.9829
Rotation 10 degrees	75.2895	16.6187	1.0000	0.7102	0.9988	0.9895
Cropping 20% frame	71.4068	23.0003	0.9999	0.8615	0.9970	0.9955
Scaling 0.5	72.6658	25.0994	0.9999	0.8944	0.9977	0.9961
Cutting 25% top-left	71.6301	14.4892	0.9999	0.2818	0.9971	0.9700

It can be seen that the PSNR value drops drastically for color secret images while under attack, likely due to the high variability of the pixel values in an RGB image. SSIM values for binary watermark is also much less affected than color watermarks. The PSNR and SSIM of color watermark is also especially low when under histogram equalization, speckle noise, rotation attack, and cutting attack. NC values for both binary and secret watermarks are very close to 1, so the watermark itself is highly robust.

4.5 Test for Different Sizes of Binary Watermark

The proposed algorithm is tested with the original size, 256×256, and two other sizes, 128×128 and 64×64. Table 11 shows the imperceptibility results between the watermarked image and

original cover image.

Table 11: Different size imperceptibility of binary watermarks.

Attacks	PSNR			SSIM			NC		
	64x64	128x128	256x256	64x64	128x128	256x256	64x64	128x128	256x256
House	95.0462	76.5336	66.9004	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Jellybeans	83.5081	68.9370	61.8698	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Fruits	83.8241	69.3554	62.6059	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

Naturally, smaller watermarks degrades the cover image less, and give higher PSNR values. The House watermark affects the cover image the least, followed by the Jellybeans and Fruits watermark. The SSIM and NC values for all watermarks of all sizes gives near perfect results, with differences only beyond five decimal places.

Following that, the PSNR, SSIM, and NC between the watermark image and extracted watermark image is computed for robustness analysis. Table 12, 13, and 14 show the robustness against no attacks and ten types of attacks for binary watermark.

Table 12: Different size robustness, House binary watermark.

Attacks	PSNR			SSIM			NC		
	64x64	128x128	256x256	64x64	128x128	256x256	64x64	128x128	256x256
No attack	73.4626	74.7120	78.0349	0.9999	1.0000	1.0000	0.9964	0.9972	0.9987
Median filter 3x3	70.2750	73.1150	75.5401	0.9999	0.9999	1.0000	0.9925	0.9960	0.9977
Histogram equalization	68.3438	71.2441	73.8652	0.9998	0.9999	0.9999	0.9885	0.9938	0.9967
Gamma correction 0.8	70.2750	73.2853	75.3966	0.9999	0.9999	1.0000	0.9925	0.9961	0.9976
JPEG compression 50 QF	70.2750	73.1150	75.3265	0.9999	0.9999	1.0000	0.9925	0.9960	0.9976
Salt & pepper 0.01	68.3438	71.4101	74.0425	0.9998	0.9999	0.9999	0.9885	0.9941	0.9968
Speckle noise 0.05	68.3438	71.4101	73.6711	0.9998	0.9999	0.9999	0.9885	0.9941	0.9965
Rotation 10 degrees	68.5724	71.1902	75.1897	0.9998	0.9999	1.0000	0.9891	0.9938	0.9975
Cropping 20% frame	70.2750	72.9511	73.3071	0.9999	0.9999	0.9999	0.9925	0.9958	0.9962
Scaling 0.5	70.2750	73.0322	75.1562	0.9999	0.9999	1.0000	0.9925	0.9959	0.9975
Cutting 25% top-left	70.2750	72.7931	74.1737	0.9999	0.9999	0.9999	0.9925	0.9956	0.9969

Table 13: Different size robustness, Jellybeans binary watermark.

Attacks	PSNR			SSIM			NC		
	64x64	128x128	256x256	64x64	128x128	256x256	64x64	128x128	256x256
No attack	71.9499	77.7223	81.2441	0.9999	1.0000	1.0000	0.9974	0.9993	0.9997
Median filter 3x3	67.1787	70.1047	72.7931	0.9997	0.9999	0.9999	0.9923	0.9961	0.9979
Histogram equalization	72.4935	75.6510	77.9071	0.9999	1.0000	1.0000	0.9977	0.9989	0.9994
Gamma correction 0.8	67.1787	70.1047	71.9499	0.9997	0.9999	0.9999	0.9923	0.9961	0.9975
JPEG compression 50 QF	67.1787	70.0631	72.4935	0.9997	0.9999	0.9999	0.9923	0.9960	0.9978
Salt & pepper 0.01	73.1150	76.2956	77.3747	0.9999	1.0000	1.0000	0.9980	0.9991	0.9993
Speckle noise 0.05	72.4935	75.2235	76.2524	0.9999	1.0000	1.0000	0.9977	0.9988	0.9991
Rotation 10 degrees	73.4626	76.2956	75.6886	0.9999	1.0000	1.0000	0.9982	0.9991	0.9989
Cropping 20% frame	67.1787	70.0219	71.0842	0.9997	0.9999	0.9999	0.9923	0.9960	0.9969
Scaling 0.5	66.9305	69.9007	72.6407	0.9997	0.9999	0.9999	0.9918	0.9959	0.9978
Cutting 25% top-left	67.1787	70.0219	71.3960	0.9997	0.9999	0.9999	0.9923	0.9960	0.9971

The PSNR, SSIM, and NC values between the extracted watermark and original watermark consistently decreases as the size of watermark decreases, except for the rotation attack on the Jellybeans watermark and salt and pepper attack on the Fruits watermark, which is still only a slight increase from the 256×256 watermark to the 64×64 watermark. Overall, the smaller watermarks are still highly robust against attacks, with the SSIM and NC values very close to 1.

Table 14: Different size robustness, Fruits binary watermark.

Attacks	PSNR			SSIM			NC		
	64x64	128x128	256x256	64x64	128x128	256x256	64x64	128x128	256x256
No attack	69.9408	79.1356	80.8549	0.9999	1.0000	1.0000	0.9956	0.9995	0.9996
Median filter 3x3	67.2647	70.2318	72.9310	0.9997	0.9999	0.9999	0.9918	0.9959	0.9978
Histogram equalization	72.7931	75.3614	76.8508	0.9999	1.0000	1.0000	0.9977	0.9988	0.9991
Gamma correction 0.8	67.2647	70.3187	72.4217	0.9997	0.9999	0.9999	0.9918	0.9960	0.9976
JPEG compression 50 QF	67.2647	70.2318	72.8912	0.9997	0.9999	0.9999	0.9918	0.9959	0.9978
Salt & pepper 0.01	72.7931	75.5038	75.4320	0.9999	1.0000	1.0000	0.9977	0.9988	0.9988
Speckle noise 0.05	72.7931	74.7120	76.0837	0.9999	0.9999	1.0000	0.9977	0.9986	0.9989
Rotation 10 degrees	73.4626	75.0899	75.0899	0.9999	1.0000	1.0000	0.9980	0.9987	0.9987
Cropping 20% frame	67.2647	70.1466	71.5680	0.9997	0.9999	0.9999	0.9918	0.9958	0.9970
Scaling 0.5	67.0116	70.2318	72.6783	0.9997	0.9999	0.9999	0.9913	0.9959	0.9977
Cutting 25% top-left	67.2647	69.8218	71.7472	0.9997	0.9999	0.9999	0.9918	0.9955	0.9971

4.6 Test for Different Sizes of Color Watermark

The same test is also done for color watermarks. Table 15 shows the imperceptibility results between the watermarked image and original cover image.

Table 15: Different size imperceptibility of color watermarks.

Attacks	PSNR			SSIM			NC		
	64x64	128x128	256x256	64x64	128x128	256x256	64x64	128x128	256x256
House	50.3449	42.6707	35.2295	0.9998	0.9991	0.9957	1.0000	0.9999	0.9995
Jellybeans	49.4954	41.7644	34.3463	0.9998	0.9991	0.9956	1.0000	0.9999	0.9994
Fruits	49.4845	41.4123	34.2993	0.9998	0.9989	0.9953	1.0000	0.9999	0.9994

The degradation on the image is significantly less with smaller watermarks, and nearly the same amount between different watermarks, with the House watermark degrading the least and the Fruits watermark degrading the most.

Following that, the PSNR, SSIM, and NC between the watermark image and extracted watermark image is computed for robustness analysis. Tables 16, 17, and 18 show the robustness against no attacks and ten types of attacks for color watermark.

Table 16: Different size robustness, House color watermark.

Attacks	PSNR			SSIM			NC		
	64x64	128x128	256x256	64x64	128x128	256x256	64x64	128x128	256x256
No attack	54.4665	59.6467	54.6375	0.9999	0.9999	0.9998	1.0000	1.0000	1.0000
Median filter 3x3	21.4846	23.9596	26.2630	0.8101	0.8848	0.9170	0.9896	0.9942	0.9966
Histogram equalization	15.9976	16.5801	17.5685	0.8518	0.8895	0.8460	0.9947	0.9968	0.9960
Gamma correction 0.8	15.8624	16.2373	16.3012	0.6684	0.7457	0.6843	0.9827	0.9896	0.9872
JPEG compression 50 QF	25.0978	27.6683	29.1060	0.8972	0.9288	0.9390	0.9955	0.9975	0.9982
Salt & pepper 0.01	19.1804	21.0108	25.1427	0.7639	0.8102	0.8992	0.9826	0.9888	0.9956
Speckle noise 0.05	14.2232	15.3603	18.0369	0.5881	0.6086	0.6780	0.9468	0.9599	0.9777
Rotation 10 degrees	20.3245	19.0145	18.5149	0.8914	0.8752	0.7728	0.9959	0.9961	0.9916
Cropping 20% frame	23.4546	24.8616	24.0564	0.8806	0.9211	0.8894	0.9944	0.9966	0.9959
Scaling 0.5	20.7373	22.9777	25.3428	0.7805	0.8563	0.8974	0.9876	0.9927	0.9958
Cutting 25% top-left	23.0621	25.6240	12.3340	0.8702	0.9247	0.3255	0.9938	0.9966	0.9485

Unlike the robustness of binary watermarks in Tables 12, 13, and 14, due to the much higher range of values of the RGB pixels, the robustness of the watermark does not show a consistent pattern in how the value increases or decreases with the change in watermark size. However, the 256x256 watermarks of House and Jellybean are particularly weakest to the cutting attack.

Table 17: Different size robustness, Jellybeans color watermark.

Attacks	PSNR			SSIM			NC		
	64x64	128x128	256x256	64x64	128x128	256x256	64x64	128x128	256x256
No attack	54.7072	60.0714	49.9662	0.9998	0.9999	0.9991	1.0000	1.0000	1.0000
Median filter 3x3	21.1072	23.7941	26.2013	0.7872	0.8725	0.9125	0.9907	0.9951	0.9972
Histogram equalization	14.0460	14.5092	15.2399	0.6078	0.6134	0.5224	0.9951	0.9962	0.9953
Gamma correction 0.8	15.1083	15.3003	15.4279	0.6477	0.6696	0.6294	0.9838	0.9872	0.9869
JPEG compression 50 QF	25.3464	27.6903	29.3555	0.8860	0.9267	0.9374	0.9965	0.9980	0.9986
Salt & pepper 0.01	20.1209	21.8092	25.3025	0.7349	0.7824	0.8914	0.9884	0.9922	0.9965
Speckle noise 0.05	15.6114	16.6463	19.2206	0.4961	0.5163	0.6390	0.9680	0.9756	0.9862
Rotation 10 degrees	18.2355	20.4373	15.3692	0.7442	0.7837	0.6363	0.9928	0.9948	0.9910
Cropping 20% frame	22.3462	23.2011	22.8290	0.8336	0.8775	0.8350	0.9943	0.9965	0.9955
Scaling 0.5	20.2664	22.6989	25.2226	0.7564	0.8428	0.8931	0.9888	0.9936	0.9965
Cutting 25% top-left	21.3480	24.6117	13.6990	0.8429	0.8663	-0.1734	0.9938	0.9967	0.9741

Table 18: Different size robustness, Fruits color watermark

Attacks	PSNR			SSIM			NC		
	64x64	128x128	256x256	64x64	128x128	256x256	64x64	128x128	256x256
No attack	54.6608	60.0370	48.4759	0.9999	0.9999	0.9992	1.0000	1.0000	1.0000
Median filter 3x3	21.2702	23.4477	25.6161	0.8460	0.8914	0.9086	0.9913	0.9948	0.9968
Histogram equalization	14.8630	15.4030	16.6213	0.7204	0.6750	0.6891	0.9910	0.9916	0.9938
Gamma correction 0.8	14.9944	14.9720	15.1165	0.6750	0.6503	0.6102	0.9859	0.9890	0.9866
JPEG compression 50 QF	25.4175	27.7666	29.1318	0.9239	0.9430	0.9452	0.9966	0.9981	0.9986
Salt & pepper 0.01	19.2796	21.5487	26.0603	0.7790	0.8182	0.9102	0.9861	0.9919	0.9971
Speckle noise 0.05	14.1477	15.4158	18.7585	0.5557	0.5617	0.6597	0.9546	0.9662	0.9844
Rotation 10 degrees	20.1037	23.7473	16.0161	0.8607	0.9106	0.7213	0.9937	0.9974	0.9860
Cropping 20% frame	22.6368	22.3475	22.1130	0.8933	0.8923	0.8603	0.9949	0.9964	0.9951
Scaling 0.5	20.2240	22.3461	24.6762	0.8138	0.8639	0.8914	0.9888	0.9932	0.9961
Cutting 25% top-left	22.1890	23.0580	17.4982	0.8899	0.8284	0.6965	0.9947	0.9951	0.9877

4.7 Speed Test

The speed of embedding and extraction time in seconds is compared between the modified NSHWT and the original DWT. The watermarking algorithm used for comparison is the same as the proposed algorithm, but the original DWT replaces the modified NSHWT. Time is taken from the MATLAB profiler where the code is running and time. Table 19 shows the time taken for embedding and extraction for the Lena cover image, along with three secret images:

Table 19: Speed test evaluation.

Evaluation	Watermark	Modified NSHWT (s)	Original DWT (s)	Percentage difference (%)
Embedding time	House	1.7010	2.7480	38.1004
	Jellybeans	1.7100	2.7660	38.1779
	Fruits	1.6500	2.7490	39.9782
	Average	1.6870	2.7543	38.7522
Extraction time	House	2.2080	3.8940	43.2974
	Jellybeans	2.2290	3.9670	43.8114
	Fruits	2.1920	3.8580	43.1830
	Average	2.2097	3.9063	43.4306

Based on Table 19, the average percentage difference for embedding time between the two methods is 38.7522%, while extraction time is 43.4306%. The modified NSHWT is a massive improvement in terms of efficiency compared to the traditional DWT. Despite having solid advantages in image watermarking, DWT is often said to use too many resources. With the application

of modified NSHWT, this problem could be significantly alleviated and, therefore, chosen over other transforms considered more efficient than DWT.

4.8 Embedding Capacity Analysis

The maximum embedding capacity of the proposed watermarking scheme is analyzed in this subsection. It is important to have a high embedding capacity, especially for sending secure information. The embedding capacity is measured by bits per pixel with Equation (20).

$$BPP = \frac{\text{Number of secret bits embedded}}{\text{Total pixels in the cover image}}. \tag{20}$$

Suppose a cover image is of the size $N \times N$ has , the cover image is decomposed by modified NSHWT to obtain four sub-bands LL, LH, HL, and HH, each of the size $\frac{N}{2} \times \frac{N}{2}$. The LL sub-band is decomposed by SVD to obtain singular values U, S, and V. The whole S singular value which has the same size as the LL sub-band, is chosen for watermark embedding. In this paper, the cover image used is of the size 512×512, so we can simply calculate the number of pixels, P, with the following equation:

$$P = \frac{H}{2} \times \frac{H}{2} = \frac{512}{2} \times \frac{512}{2} = 65536.$$

Binary images has 1 bit of data for each pixel, while RGB images has 24 bits of data for each pixel. Thus, the maximum capacity for binary watermark and color watermark is calculated as in Equations (21) and (22).

$$BPP = \frac{65536}{512 \times 512} = 0.25\text{bpp}, \tag{21}$$

$$BPP = \frac{65536 \times 24}{512 \times 512} = 6\text{bpp}. \tag{22}$$

As shown, the maximum embedding capacity is 0.25bpp for binary watermark and 6bpp for color watermarks. In comparison, the maximum embedding capacity for binary watermark is higher in the proposed scheme than Ahmadi et al. [1] scheme, which embeds a maximum of 0.0104bpp. The relationship between the embedding capacity and imperceptibility or robustness is also discussed. Figure 12 shows a line graph of watermark size against PSNR between watermarked image and original cover image, with data taken from Tables 11 and 15.

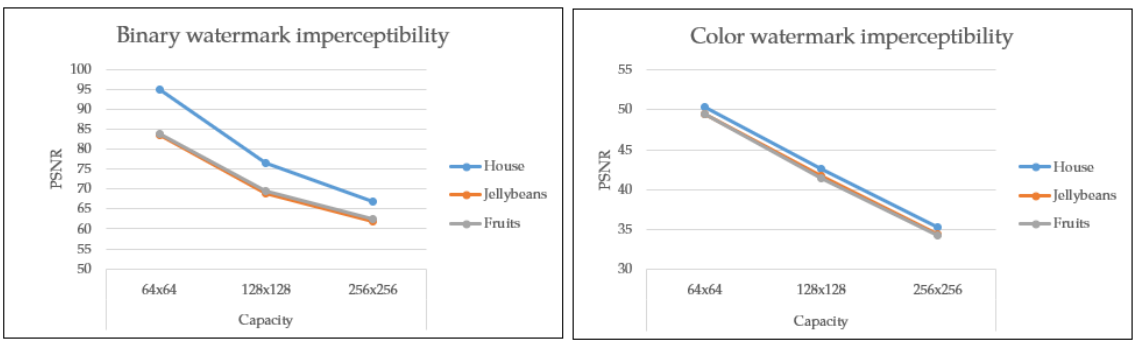


Figure 12: Left: Capacity against imperceptibility for binary watermarks; Right: Capacity against imperceptibility for color watermarks.

From Figure 12 (left) and Figure 12 (right), it can be observed that smaller watermarks degrades the cover image less, with the binary House watermark affecting much less than the Jellybeans and Fruits watemark. Figure 13 shows a line graph of watermark size against PSNR between extracted watermark image and original watermark image. Data is taken from the results of extraction without attacks in Tables 12-14 and 16-18.

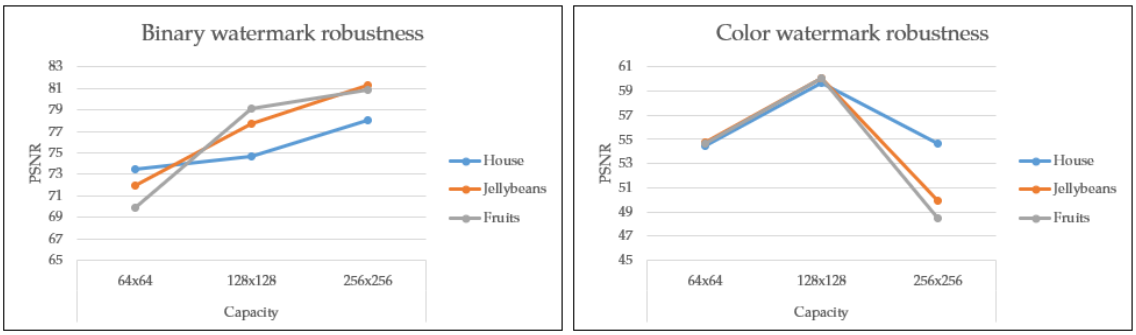


Figure 13: Left: Capacity against robustness for binary watermarks; Right: Capacity against robustness for color watermarks.

From Figure 13 (left), unlike the imperceptibility, the watermarks increase in robustness as their capacity increases, and both the Jellybeans and Fruits watermark robustness increase sharply from size 64×64 to 128×128. Based on Figure 13 (right), the robustness of color watermark is much more erratic than the other results. The robustness increases from size 64×64 to 128×128, but decreases from size 128×128 to 256×256. It is also worth noting that all watermarks show very similar robustness on sizes between each other on sizes 64×64 and 128×128. The nature of color images make the robustness trickier to predict when the size changes. Therefore, it can be concluded from the graphs that as the embedding capacity increase, the binary or color watermark imperceptibility decreases, and the robustness of only the binary watermark increases. This shows that the size of watermark is also an important factor for watermark imperceptibility and robustness.

5 Conclusions

This paper presented a secure and robust non-blind watermarking scheme using DWT, SVD, Arnold's cat map and Rabin-p cryptosystem. The results show that the algorithm performs well when compared with other similar schemes in terms of robustness. The cover image degrades considerably in quality when a color watermark is added, while the results are much better when embedding a binary watermark. The watermarks are also highly robust against the ten attacks which were tested in this paper, showing high values of NC.

Other than that, the high efficiency of the modified NSHWT compared to the original DWT is a promising feature, as well as the flexibility of the proposed scheme in embedding different sizes of binary and color watermarks. More algorithms should apply encryption with a cryptosystem to better protect the watermark. It is hoped that future works will improve the image quality for color watermark embedding, as well as further improving the efficiency and security of the algorithm.

Acknowledgement We would like to thank the Ministry of Education Malaysia, Research Management Centre and Faculty of Computer and Mathematical Sciences, Universiti Teknologi Mara Malaysia for the support in completing this research. This research was financially supported under the Fundamental Research Grant Scheme for Research Acculturation of Early Career Reseachers (FRGS-RACER) with the grant code FRGS-RACER (RACER/1/2019/STG06/UITM//2).

Conflicts of Interest The authors declare no conflict of interest.

References

[1] S. B. B. Ahmadi, G. Zhang, M. Rabbani, L. Boukela & H. Jelodar (2021). An intelligent and blind dual color image watermarking for authentication and copyright protection. *Applied Intelligence*, *51*(3), 1701–1732. https://doi.org/10.1007/s10489-020-01903-0.

[2] N. Ahmed, T. Natarajan & K. R. Rao (1974). Discrete cosine transform. *IEEE Transactions on Computers*, *23*(1), 90–93. https://doi.org/10.1109/T-C.1974.223784.

[3] W. H. Alshoura, Z. Zainol, J. S. Teh & M. Alawida (2020). A new chaotic image watermarking scheme based on SVD and IWT. *IEEE Access*, *8*, 43391–43406. https://doi.org/10.1109/ACCESS.2020.2978186.

[4] T. K. Araghi, A. Manaf, M. Zamani & S. K. Araghi (2016). A survey on digital image watermarking techniques in spatial and transform domains. *International Journal of Advances in Image Processing Techniques*, *3*(1), 6–10.

[5] M. A. Asbullah & M. R. K. Ariffin (2016). Design of Rabin-like cryptosystem without decryption failure. *Malaysian Journal of Mathematical Sciences*, *10*, 1–18.

[6] S. A. Bamerni & A. K. Al-Sulaifanie (2019). An efficient non-separable architecture for Haar wavelet transform with lifting structure. *Microprocessors and Microsystems*, *71*, 102881. https://doi.org/10.1016/j.micpro.2019.102881.

[7] C. Eckart & G. Young (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, *1*(3), 211–218. https://doi.org/10.1007/BF02288367.

[8] A. Ghafar, M. Ariffin & M. Asbullah (2019). A new attack on special-structured RSA primes. *Malaysian Journal of Mathematical Sciences*, *13*, 111–125.

[9] A. Horé & D. Ziou (2010). Image quality metrics: PSNR vs SSIM. In *2010 20th International Conference on Pattern Recognition*, pp. 2366–2369. IEEE, Istanbul, Turkey.

[10] V. S. Jabade & S. R. Gengaje (2011). Literature review of wavelet based digital image watermarking techniques. *International Journal of Computer Applications*, *31*(7), 28–35. https://doi.org/10.5120/3838-5334.

[11] P. Kulkarni, S. Bhise & S. Khot (2015). Review of digital watermarking techniques. *International Journal of Computer Applications*, *109*(16), 40–44. https://doi.org/10.5120/19275-1029.

[12] M. Li, T. Liang & Y.-j. He (2013). Arnold transform based image scrambling method. In (*In Proceedings of 3rd International Conference on Multimedia Technology* (*ICMT-13*), pp. 1309–1316. Atlantis Press, Dordrecht, Netherlands.

[13] J. Liu, J. Huang, Y. Luo, L. Cao, S. Yang, D. Wei & R. Zhou (2019). An optimized image watermarking method based on HD and SVD in DWT domain. *IEEE Access*, *7*, 80849–80860. https://doi.org/10.1109/ACCESS.2019.2915596.

[14] Z. Mahad, M. Asbullah & M. Ariffin (2017). Efficient methods to overcome Rabin cryptosystem decryption failure. *Malaysian Journal of Mathematical Sciences*, *11*, 9–20.

[15] A. Naveed, Y. Saleem, N. Ahmed & A. Rafiq (2015). Performance evaluation and watermark security assessment of digital watermarking techniques. *Science International*, *27*(2), 1271–1276.

[16] A. Nitaj (2017). Post quantum cryptography. *Malaysian Journal of Mathematical Sciences*, *11*, 1–28.

[17] P. Parashar & R. K. Singh (2014). A survey: Digital image watermarking techniques. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, *7*(6), 111–124.

[18] G. Peterson (1997). Arnold's cat map. *Math Linear Algebra*, *45*, 1–7.

[19] D. Qi, J. Zou & X. Han (2000). A new class of scrambling transformation and its application in the image information covering. *Science in China Series E: Technological Sciences*, *43*(3), 304–312.

[20] M. O. Rabin (1979). *Technical report: Digitalized signatures and public-key functions as intractable as factorization*. Massachusetts Institute of Technology, Cambridge, MA.

[21] S. Roy & A. K. Pal (2019). A hybrid domain color image watermarking based on DWT–SVD. *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, *43*(2), 201–217.

[22] M. J. Shensa et al. (1992). The discrete wavelet transform: Wedding the a trous and Mallat algorithms. *IEEE Transactions on Signal Processing*, *40*(10), 2464–2482.

[23] K. F. Tsang & O. C. L. Au (2001). Review on attacks, problems, and weaknesses of digital watermarking and the pixel reallocation attack. In *Security and Watermarking of Multimedia Contents III*, volume 4314 pp. 385–393. SPIE Digital Library, San Jose, CA.

[24] D. Wang, F. Yang & H. Zhang (2016). Blind color image watermarking based on DWT and LU decomposition. *Journal of Information Processing Systems*, *12*(4), 765–778.

[25] C.-q. Yin, L. Li, A.-q. Lv & L. Qu (2007). Color image watermarking algorithm based on DWT-SVD. In *2007 IEEE International Conference on Automation and Logistics*, pp. 2607–2611. IEEE, Jinan.