Computers and Electrical Engineering 102 (2022) 108207

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

AuSR2: Image watermarking technique for authentication and self-recovery with image texture preservation

A. Aminuddin [a,b,*], F. Ernawan [a]

[a] Department of Computer Graphic and Multimedia, Faculty of Computing, College of Computing and Applied Sciences, Universiti Malaysia Pahang, Kuantan, Malaysia
[b] Department of Information System, Faculty of Computer Science, Universitas Amikom Yogyakarta, Sleman, Indonesia

ARTICLE INFO

Keywords:
Image authentication
Image watermarking
Image inpainting
Self-recovery
Tamper detection
Tamper coincidence problem

ABSTRACT

This paper presents an image watermarking technique for authentication and self-recovery called AuSR2. The AuSR2 scheme partitions the cover image into 3 × 3 non-overlapping blocks. The watermark data is embedded into two Least Significant Bit (LSB), consisting of two authentication bits and 16 recovery bits for each block. The texture of each block is preserved in the recovery data. Thus, each tampered pixel can be recovered independently instead of using the average block. The recovery process may introduce the tamper coincidence problem, which can be solved using image inpainting. The AuSR2 implements the LSB shifting algorithm to increase the imperceptibility by 2.8%. The experimental results confirm that the AuSR2 can accurately detect the tampering area up to 100%. The AuSR2 can recover the tampered image with a PSNR value of 38.11 dB under a 10% tampering rate. The comparative analysis proves the superiority of the AuSR2 compared to the existing schemes.

1. Introduction

Due to the rapid development of digital imaging and communication technology, the security of images has become a significant challenge. The availability of image editing software such as Adobe Photoshop leads to editing and distributing images becomes challenging to trust the image content. The images may undergo forgery attacks to spread fake propaganda. These attacks compromise the authenticity and the originality of digital images. Therefore, image authentication has an essential role in protecting the authenticity of the images. The image authentication process aims to restore the distrust regarding the image content [1]. Image authentication can be used to detect and localize any modification in the image. There are many types of image forgery, such as color filtering, image composition, object removal, background modification, image splicing, and copy-move forgery [2]. Image splicing transfers an image object to another image, while copy-move forgery is copying an object into another coordinate in the same image [3]. The fragile watermarking technique can be used for image authentication and self-recovery. The authentication and recovery bits can be embedded into the original image to prevent illegal editing and modification of the images [4]. The information contains the authentication data and the recovery data. The authentication data is utilized to authenticate and localize the tampering area on the image. The recovery data is utilized to recover the original image from the tampered image.

Image watermarking techniques are categorized into fragile, semi-fragile, and robust watermarking [5]. Robust watermarking is

* Corresponding author at: Department of Computer Graphic and Multimedia, Faculty of Computing, College of Computing and Applied Sciences, Universiti Malaysia Pahang, Kuantan, Malaysia.
E-mail address: afrig@amikom.ac.id (A. Aminuddin).

https://doi.org/10.1016/j.compeleceng.2022.108207
Received 15 April 2022; Received in revised form 23 June 2022; Accepted 24 June 2022

widely utilized for copyright protection [6], while semi-fragile and fragile watermarking are widely used for image authentication. Semi-fragile watermarking can detect significant image tampering and tolerate minor tampering attacks such as nosing, filtering, and image compression [7]. However, this technique has a limited amount of payload capacity, which is insufficient for recovery purposes [8]. In comparison, fragile watermarking cannot tolerate any tampering attack. Thus, it will detect all the tampering attacks in the image. Furthermore, this technique can store a large amount of watermark data to be embedded into the original image for authentication and recovery [9–15]. Robust and semi-fragile watermarking technique stores the watermark data in frequency domains such as DCT, DWT, and SVD [16]. In comparison, fragile watermarking stores the watermark data into the LSB of the images.

Based on the dependency, image watermarking is categorized into three categories: non-blind, semi-blind, and blind watermarking [17]. In a non-blind scheme, the original image is required for authentication and recovery. In contrast, a semi-blind scheme requires external data such as a block map or watermark data for authentication and recovery. The blind watermarking scheme does not require external data other than the tampered image for authentication and recovery. Therefore, the blind watermarking scheme is preferred when the original image and the external data are unavailable.

This research proposed a fragile blind image watermarking technique for image authentication and self-recovery. Each block generates two authentication bits and 16 recovery bits, including the texture information. The watermark data is embedded into the cover image using the LSB shifting algorithm. Furthermore, the AuSR2 scheme provides a three-layer authentication process to obtain high accuracy in tamper localization. The first layer compares the reconstructed and extracted authentication bits. The second layer checks the undetected area of the surrounding blocks. The third layer merges the result of the previous layer of each RGB channel. Next, the tampered image is then recovered by preserving the texture of each block. The previous research [15] did not consider the texture information as part of the recovery data. Finally, image inpainting is also employed to solve the tamper coincidence problem.

This paper is organized as follows. Section 2 presents related works of the fragile watermarking schemes. Section 3 explains the proposed watermark embedding, authentication, and self-recovery. Section 4 presents the experimental results and analysis of the AuSR2 compared to the existing methods. Finally, this research study is concluded in Section 5.

2. Related works

The fragile image watermarking technique consists of block mapping, watermark generation, watermark embedding, image authentication, and tamper recovery. Many researchers have tailored techniques to solve each step. Block mapping plays an important role in storing the recovery data on another block. The block map is utilized to assign the location to embed the recovery data into another block location. The recovery data is obtained to recover the origin block when modified. Chang et al. [18] presented block mapping using the Arnold transform. The Arnold transform was also employed to encrypt and scramble the cover image for additional security. Hisham et al. [19] and Ernawan et al. [20] used a spiral pattern on the block mapping for embedding the watermark data. Each block was mapped in a spiral manner to provide robustness against a single attack location. However, the scheme may fail to recover the tampered area against multiple tampering attacks.

The watermark generation process produces the authentication and recovery data from each image block. The authentication data is utilized to authenticate a block, while the recovery data is used to recover the tampered block. Dadkhah et al. [9] generated the authentication data by hashing the average value of the image block. The scheme cannot detect a tampered block if the average value of the original block is similar to the tampered block value. In terms of recovery data, researchers [9,12,13,15] utilized an average block as the recovery data. This technique best suits the fragile watermarking scheme that uses 2×2 non-overlapping blocks. The recovered image will have visible pixelated effects when increasing the block size. Singh and Singh [14] generated the recovery data by using the DCT coefficients. Each block is transformed into the DCT coefficients for the recovery data. However, due to the payload limitation for embedding the recovery data, only a limited amount of DCT coefficients is used, which will degrade the recovered image quality.

In the fragile watermarking technique, the watermark data is embedded into the LSB of the cover image to produce minimum error distortions compared to MSB embedding. The researchers [10–12] embedded the watermark data into two LSB. Two LSB embedding will produce an average PSNR value of 44 dB. Researchers [13] and [14] embedded the watermark data into three LSB, which produced an average PSNR value of 37 dB. Dadkhah et al. [9] compared the authentication data to the average of its image block. If both values were equal, the block was not detected as tampered. Otherwise, the block is marked as tampered. The tamper detection rate depends on the number of authentication bits on each block. If each block has a single authentication bit, it has a probability of 50% undetected. Two authentication bits will produce a probability of a 75% detection rate. Hence a multi-layer authentication technique can obtain a high detection rate.

Another challenge in image recovery is the occurrence of the tamper coincidence problem. Tamper coincidence will occur when the current block and its recovery data have also been tampered. Researchers solve the tamper coincidence problem using multiple recovery data [21] and [22]. They utilized 4×4 non-overlapping blocks to accommodate the multiple recovery data. If a pixel value on its image block has been tampered with, then all the pixel values on the block are considered tampered pixels. This problem is called false positive detection. Thus, a large block size may increase the false positive detection. If one recovery data suffers the tamper coincidence problem, another recovery data is employed to recover it. However, there will always be tamper coincidence on a larger tampering area. The existing schemes by Molina-Garcia et al. [12] and [23] solved the tamper coincidence problem by implementing image inpainting techniques. The schemes took eight surrounding pixels to solve the tamper coincidence problem. The average values of those eight pixels were used to replace the tamper coincidence pixel. This technique assumed that those eight surrounding pixels do not suffer the tamper coincidence problem. However, it may occur in any image location. Thus, those eight surrounding pixels may also suffer the same tamper coincidence problem. Therefore, the image inpainting technique employed by Molina-Garcia et al. [12] and

[23] may not accurately solve the tampered coincidence problem.

Image authentication and self-recovery have been intensively investigated in recent years. Some challenges need to be solved, such as improving the quality of the watermarked and recovered image and the tamper detection accuracy. Researchers have tried to overcome these issues with various methods, but it still has not achieved a satisfactory level and can be improved further. The AuSR2 provides contributions to solving the issues in image authentication and self-recovery as follows:

1) Watermarked image quality: The AuSR2 implemented LSB shifting algorithm to improve the watermarked image quality. The algorithm minimizes the variation of pixel intensity level between the cover image and the watermarked image. The results demonstrate that the AuSR2 outperforms the existing scheme by 2.8% in terms of PSNR value.
2) Tamper detection accuracy: The AuSR2 implemented three-layer authentication. Each layer increases the true positive detection and decreases the false-negative detection. The results show that the AuSR2 provides a 2.2% improvement in accuracy compared to the existing methods.
3) Recovered image quality: The AuSR2 preserves the details of each image block by considering its texture. Each pixel in the recovered image will have a different intensity level depending on the texture. In addition, the AuSR2 solves the tamper coincidence problem by implementing the image inpainting technique and multiple recovery data. Therefore, the AuSR2 can produce high-quality recovered images and improves by 2.8% compared to the previous methods.

3. Proposed method

This section proposes a blind fragile image watermarking technique for authentication and self-recovery with image texture preservation. The AuSR2 scheme can be divided into eight primary processes:

1) Block map generation: Three block maps are generated for each RGB channel. Each block map determines the location of the recovery data. A secret key provides the randomness of the block map.
2) Texture sets definition: Various texture sets are defined in this process. Each set may contain up to 16 different textures. The limitation is due to the payload limitation for watermark data.
3) Recovery bits generation: The recovery data are generated in this process. The recovery data consist of minimum value, maximum value, and the result of texture classification of each block.
4) Authentication bits generation: In this process, the authentication data are generated based on the contents of its image block. The binary value of block map location is also involved to ensure the distinction between two identical blocks. Both blocks will have different authentication bits.
5) Watermark embedding: In this process, the watermark data are embedded into the cover image using the LSB shifting algorithms to achieve a high-quality watermarked image.
6) Watermark extraction: In this process, the image may undergo a possible attack in the communication channel. Thus, the AuSR2 extracts and reconstructs the watermark data for authentication and self-recovery.
7) Image authentication: In this process, the extracted and reconstructed authentication bits are compared to localize the tampered area of the image. Three-layer authentication is implemented to obtain a high tamper detection accuracy.
8) Self-recovery: Finally, the tampered areas of the image are recovered using the extracted and reconstructed recovery data. In addition, an image inpainting technique is utilized to solve the tamper coincidence problem.

3.1. Block map generation

The block map is required to determine the recovery data of each block. If a block has been tampered with, then the recovery data is utilized to recover the tampered block. The minimum value, the maximum value, and the texture information of each image block are used as the recovery data. Each of these data is stored in different block locations to provide a second chance for recovery. The block map is generated using the Pseudorandom Number Generator (PRNG) [15] with a seed value as the initialization parameter. The seed value is defined by:

$$seed_{(c,y)} = n \cdot c \cdot y \cdot key \tag{1}$$

where n represents the total number of blocks on an individual channel, c represents the index of RGB channels, y represents the recovery type, and key denotes the secret key. This process creates nine block maps representing three recovery data for each channel. In terms of security, each watermarked image is secured by using a secret key. The image authentication and self-recovery will not be performed if any modification is applied to the key.

3.2. Texture sets definition

Researchers [9,12,13] utilized the average block value as the recovery data. Thus, the schemes recovered a tampered area with the average pixel value of each image block. The utilization of average pixels on the large block size may produce a pixelated effect on the recovered image. This research proposes five sets of textures as a part of the recovery data. The AuSR2 scheme provides sixteen bits of

the recovery data consisting of six bits of the minimum value, six of the maximum value, and four of the texture information. First, the texture is generated from the base texture of a linear gradient at a 0° angle. The texture is then rotated clockwise at 90° three times to produce three different textures. Each texture represents 90°, 180°, and 270°. These four textures are grouped as a texture set *A*. There are five texture sets, called sets *A, B, C, D*, and *E*. Each texture in set *A* is then rotated 45° to produce texture set B, which represents 45°, 135°, 225°, and 315°. These four bits can store up to 16 different textures. The texture sets proposed in this research are shown in Fig. 1.

Next, each texture in sets *A* and *B* is rotated clockwise at 22.5° to produce eight textures grouped as texture set *C*. The texture set *C* represents 22.5°, 67.5°, 112.5°, 157.5°, 202.5°, 247.5°, 292.5°, and 337.5°. Next, each texture in sets *A, B*, and *C* is rotated at 11.25° to produce 16 textures called set *D*. Next, the texture set *D* is transformed to generate the texture set *E*, as visualized in Fig. 2.

Fig. 2 shows the transformation of the first four textures of texture set *D*. The remaining 12 textures have a similar pattern with a 90° rotation. The transformation begins by finding the order of weight values on each texture. It creates a unique pattern for each texture, as shown in Fig. 2. Each weight value of texture set *D* is replaced with the weight value of 0, 31.875, 63.75, 95.625, 127.5, 159.375, 191.25, 223.125, 255 to generate texture set *E*. The weight is obtained from the equal interval values between 0 to 255. This transformation produces an exact standard deviation of 82.30 for each texture set *E*. In contrast, texture set *D* has two different standard deviations of 92.02 and 75.88. Furthermore, each of these texture sets has its unique characteristics listed in Table 1.

According to Table 1, texture sets *A* and *B* consist of four textures with 90° rotation. The texture set *C* consists of eight textures with a 45° rotation. Texture sets *D* and *E* consist of 16 textures with 22.5° rotation. It can be noticed that texture sets *D* and *E* have similar visualization but have different weight values. Each texture set in Table 1 will be investigated in Section 4.3 to find one that produces a high-quality recovered image.

3.3. Recovery bits generation

The recovery bits are generated from each image block. Each block has three recovery data: minimum value, maximum value, and the index of the selected texture from a texture set. This recovery data can be used to recover a block of the image in the self-recovery process. The recovery bits can be obtained by:

1) Partition each channel of the cover image into 3 × 3 non-overlapping blocks. The block size of 3 × 3 is chosen because it provides a large embedding capacity for the authentication bits and recovery data. Eighteen bits can be embedded into two LSB on a 3 × 3 block size. Two bits are utilized as the authentication bits. At the same time, 16 bits are used as the recovery bits. In contrast, a block size of 2 × 2 only provides eight bits embedding capacity on two LSB.
2) Calculate the minimum and maximum value of the selected block as follows:

$$s = min(\bar{x} - \min(p_i), \max(p_i) - \bar{x}) \tag{2}$$

$$r_{min} = \bar{x} - s \tag{3}$$

$$r_{max} = \bar{x} + s \tag{4}$$

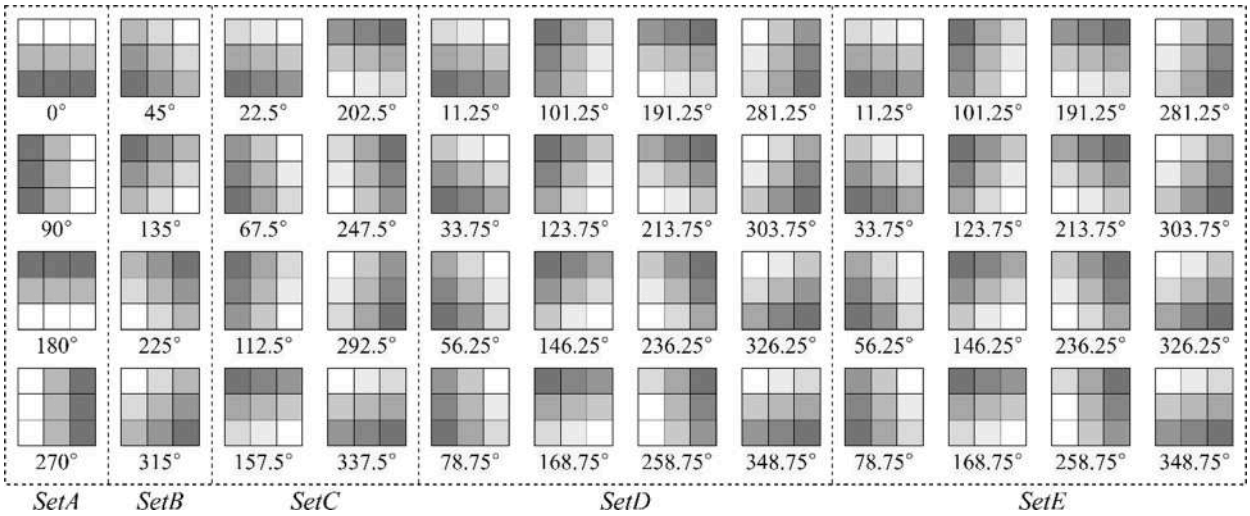

Fig. 1. Five proposed texture sets for recovery bits generation and self-recovery.

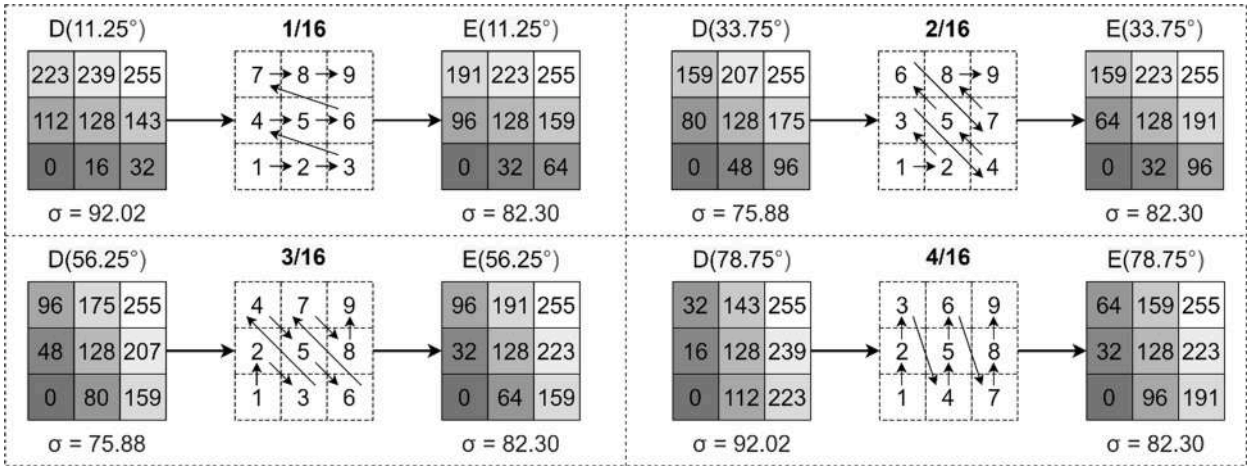

Fig. 2. The transformation from texture set *D* to texture set *E*.

Table 1
Texture set characteristics.

Characteristic	*SetA*	*SetB*	*SetC*	*SetD*	*SetE*
Number of textures	4	4	8	16	16
Required storage	2 bits	2 bits	3 bits	4 bits	4 bits
First texture angle	0.00°	45.00°	22.50°	11.25°	11.25°
Last texture angle	270.00°	315.00°	337.50°	348.75°	348.75°
Angle increment	90.00°	90.00°	45°	22.5°	22.5°
Standard deviation	104.10	73.61	82.30	92.02, 75.88	82.30

where p is the image block, $min(p_i)$ and $max(p_i)$ represent the lowest and the highest pixel value on that block. $\overline{x}$ is the average value of the image block, s represents the minimum distance to the average value, r_{min} denotes a minimum value for recovery, and r_{max} indicates a maximum value for recovery. Six MSB of the r_{min} and r_{max} will be embedded into the cover image.

1) Select the texture by minimizing the error between each block and the selected texture set based on the following equations:

$$z(p,t) = \sum_{i=1}^{3}\sum_{j=1}^{3}\left(p_{(i,j)} - t_{(i,j)}\right)^2 \tag{5}$$

$$r_{txt} = \underset{x \in \{1,\ldots,n\}}{\arg\min}\ z(p,t_x) \tag{6}$$

where p represents the selected block, t denotes the texture weightage, x represents the indexes of the textures, r_{txt} denotes the index of the selected texture, and n represents the number of textures on a selected texture set.

1) Repeat Step 2 to Step 3 for all blocks and channels to obtain the recovery data of the cover image.

3.4. Authentication bits generation

The authentication bits are generated from each image block. This authentication data can be used to authenticate a block in the tamper detection process. The authentication bits can be obtained by:

1) Partition each channel of the cover image into 3 × 3 non-overlapping blocks.
2) Convert each pixel on the image block into binary values. The AuSR2 takes six MSB to generate the authentication data. In total, there are 54 authentication bits a_{val} of each block with the size of 3 × 3 pixels.
3) Randomize the authentication bits of each block using the block map. This process ensures that two identical blocks will produce different authentication bits. In the block map generation process, the block map of each channel consists of the recovery locations of minimum, maximum, and texture recoveries. Those three locations are stored in a 32-bits integer. The OR operation is used to combine those three values to generate a random 32-bit value as defined by:

$$a_{rnd} = m_{min} \oplus m_{max} \oplus m_{txt} \tag{7}$$

where m_{min} represents the recovery location of minimum value, m_{max} represents the recovery location of maximum value, and m_{txt} denotes the recovery location of the texture value.

1) Generate two authentication bits a_{gen} of each block as follows:

$$a_{kv} = \{a_{val},\ a_{rnd}\} \tag{8}$$

$$a_{gen} = \left(\sum_{i=1}^{n} [a_{kv(i)} = 1] \right) mod\ 4 \tag{9}$$

where n represents the length of the a_{kv} which is 86 bits, a_{gen} denotes the authentication bits of the current block. The modulo of four produces decimal values between zero and three which can be stored in two bits value.

1) Repeat Step 2 to Step 4 for all blocks and channels to obtain the authentication data.

3.5. Watermark embedding

The scheme embeds the two authentication bits and sixteen recovery bits into two LSB. Two authentication bits are embedded into the origin block location, and sixteen recovery bits are embedded into three distinct locations based on the block map. The watermark embedding process is visualized in Fig. 3.

In previous research by Dadkhah et al. [9–11], the watermark embedding process replaces the last two bits of each pixel with the watermark data. However, this process will produce four different contrast levels between the original and watermarked pixels. This process contributes to the error distortion on the watermarked image. Thus, the AuSR2 scheme utilizes LSB shifting algorithm to achieve a high-quality watermarked image. The watermark embedding process is explained as follows:

1) Partition each channel of the cover image into 3 × 3 non-overlapping blocks.
2) Retrieve the recovery data that will be embedded into the selected block and permute it as defined by:

$$r_{gen} = \{r_{min}, r_{max}, r_{txt}\} \tag{10}$$

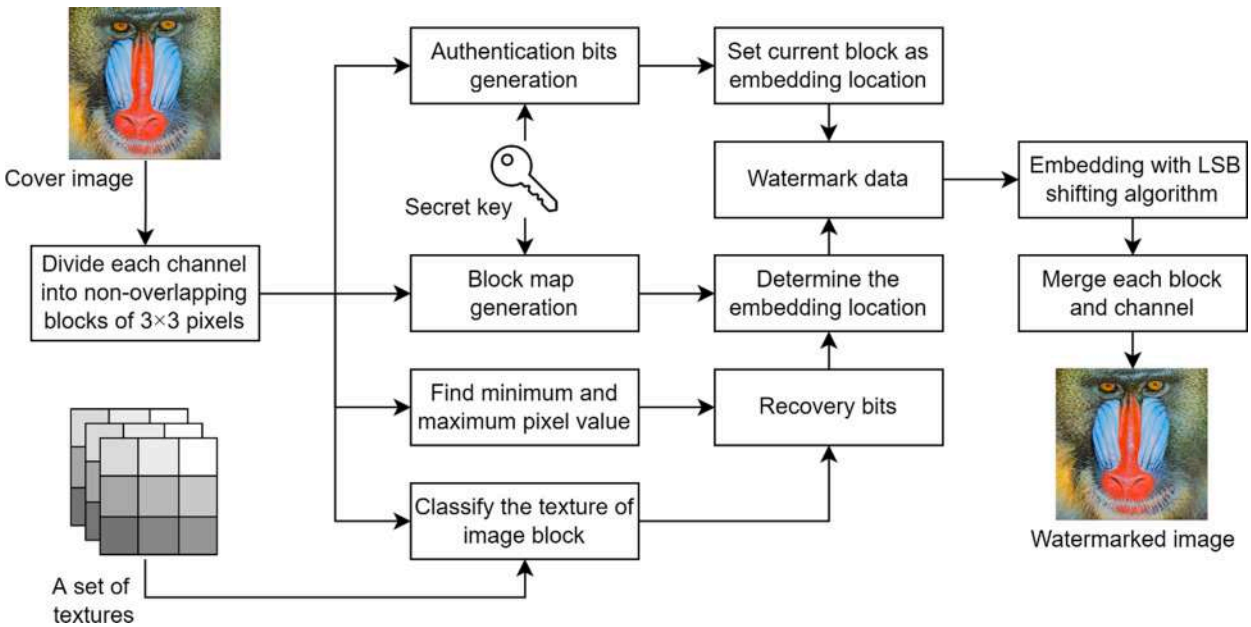

Fig. 3. The proposed watermark embedding of AuSR2.

$$r_{prm} = permute\left(r_{gen}, a_{rnd}\right) \tag{11}$$

where r_{gen} is the generated 16 recovery bits, r_{prm} is the permuted recovery data using a_{rnd} as the permutation key taken from Eq. (7). The permutation function has been applied in the previous research [15]. Note that r_{min}, r_{max}, and r_{txt} are taken from three distinct block locations. The block map determines the locations.

1) Retrieve two authentication bits and define the watermark data as follows:

$$w = \left\{r_{prm}, a_{gen}\right\} \tag{12}$$

where w represents 18-bits watermark data that consist of 16 recovery bits r_{prm} and two authentication bits a_{gen}.

1) The watermark data is embedded into each pixel of the selected blocks. The embedding process is performed using the LSB shifting algorithm previously defined in [15].
2) Repeat Step 2 to Step 4 for all blocks and channels to produce the final watermarked image.

3.6. Watermark extraction

The AuSR2 scheme extracts the watermark data from two LSB of the tampered image. In addition, the AuSR2 scheme also reconstructs the watermark data from the tampered image. The watermark extraction process is explained as follows:

1) Partition each channel of the tampered image into 3 × 3 non-overlapping blocks.
2) Reconstruct authentication bits a_{gen} and a_{kv} from the tampered image as explained in Eqs. (8) and (9). Reconstruct the minimum, maximum, and texture values from the tampered image as defined in Eqs. (2)–(4).
3) Extract the authentication and watermark bits from two LSB in the selected image block. Depermute the watermark bits using the recovery location to obtain sixteen recovery bits. The recovery bits are divided into three parts: the first six bits are a minimum value, the next six are a maximum value, and the last four are a texture value.
4) Repeat Step 2 to Step 3 for all blocks and channels to obtain the reconstructed and extracted watermark data.

The extracted and reconstructed authentication bits will be compared in the authentication process, while the extracted and reconstructed recovery bits will be utilized in the image recovery process. Furthermore, the watermark extraction process can be visualized in Fig. 4.

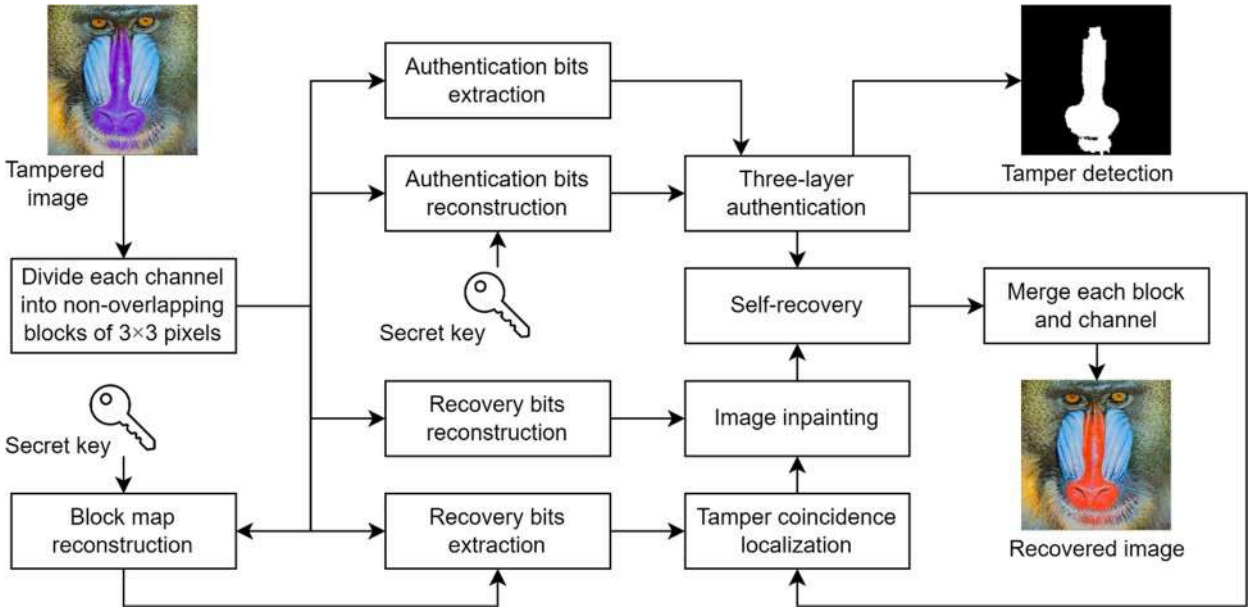

Fig. 4. The proposed watermark extraction of AuSR2.

Based on Fig. 4, the tampered image undergoes watermark extraction, watermark reconstruction, three-layer authentication, and self-recovery. The secret key ensures the watermark embedding and extraction utilize the same block map. Thus, the watermark data cannot be extracted when the watermark extraction utilizes a different key to the embedding process.

3.7. Image authentication

This research investigates the detection rate of the AuSR2 scheme in terms of true positive, false negative, false positive, and true negative. The proposed three-layers authentication process is explained in detail as follows:

1) The first-layer authentication compares the reconstructed and extracted authentication bits from the tampered image. Each block has two authentication bits, which means there is a 25% false-negative and 75% true-positive detection probability.
2) The second-layer authentication checks the false-negative blocks. If the surrounding blocks are detected, then the block is set as a tampered block. The second layer authentication algorithm was previously defined in [15].
3) The third layer authentication compares the output of the second layer authentication in three RGB channels. If an image block of the RGB channels is detected as a tamper, then set all blocks of RGB channels on its block locations to be detected as tampered. This third-layer authentication reduces the false-negative detection further.

3.8. Self-recovery

The proposed self-recovery scheme is explained in detail as follows:

1) Partition each channel of the tampered image into 3 × 3 non-overlapping blocks.
2) Find the tamper coincidence problem using the block map by checking the tampered block and its recovery location. If both locations are detected as tampered, then set both locations as tamper coincidence problems.
3) Find the texture value of the tampered image. This step takes the texture from another channel when the texture is unavailable for the current channel. However, when the texture for all the RGB channels is unavailable, the texture value is interpolated in Step 5.
4) Perform image inpainting to solve the tamper coincidence problem that occurred on the minimum and maximum recovery value. The image inpainting technique finds the non-tamper coincidence in an outward spiral direction. The AuSR2 scheme divides the surrounding blocks into eight regions corresponding to a 45° angle. The illustration of the image inpainting process is visualized in Fig. 5. This process interpolates the tamper coincidence (*tc*) recovery value using the surrounding non-tamper coincidence (*ntc*) recovery value.

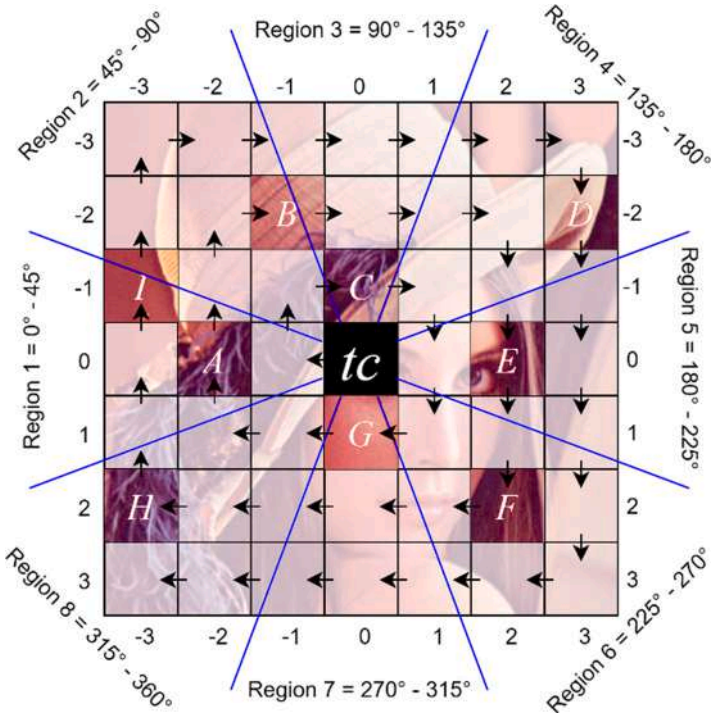

Fig. 5. Illustration of the inpainting process.

Based on Fig. 5, *tc* is the tamper coincidence that must be solved. *A* – *I* are the non-tamper coincidence recovery. The white blocks represent another tamper coincidence problem in the image. The arrow represents the outward spiral search direction. Region 1 is represented by recovery *A* as it is closer to *tc* than recovery *I*. At first, divide the surrounding *ntc* into eight regions. Each region represents a 45° direction relative to the *tc* location. The scheme searches the *ntc* location in the outward spiral direction from the *tc* location. Thus, one *ntc* will represent each region near the *tc* location. The distance between *tc* and *ntc* is computed using the Euclidean distance. The weight of each *ntc* is defined by:

$$ed_i = \sqrt{(ntc_x - tc_x)^2 + (ntc_y - tc_y)^2} \tag{13}$$

$$\alpha = \left(1 - \frac{ed_i}{\max(ed)}\right) \cdot \left(\frac{1}{ed_i^2}\right) \tag{14}$$

where *ed* is the Euclidean distance between *tc* and *ntc*, max(*ed*) is the maximum distance of all available *ntc* to the *tc*, α is the weight of the *ntc*. The final *tc* value is interpolated based on the following Equation:

$$tc = round\left(\frac{\sum_{i=1}^{8}(ntc_i \cdot \alpha_i)}{\sum_{i=1}^{8}\alpha_i}\right) \tag{15}$$

where *tc* represents the solved tamper coincidence value, ntc_i represents eight surrounding non-tamper coincidence values, α_i denotes the weight of each ntc_i value.

1) Solve the remaining texture recovery from the tampered image. First, find the average between the minimum and maximum value of each block. When the minimum or maximum value suffers the tamper coincidence problem, it must be solved in step 4. Next, take eight average values of the surrounding block and find the texture recovery using Eqs. (5) and (6).
2) Recover each tampered block using the recovery process based on the minimum, maximum, and texture using Algorithm 1.

where *n* represents the block size, r_{min} and r_{max} denote the recovery value from the image inpainting process, *textures* indicate the selected texture set, and p_{out} is the output of this algorithm which corresponds to an image block with the size of 3 × 3 pixels.

1) Repeat Step 2 to Step 6 to obtain the recovered image.

4. Experimental results

The experiments were carried out on a computer with a 1.8 GHz octa-core AMD Ryzen 7 5700U processor with Windows 10 operating system and 32 GB memory. The experiments use MATLAB 2021a as the programming environment. The AuSR2 scheme is also tested by using eight color images, as shown in Fig. 6. Each image has a size of 512 × 512 pixels with 24 bits/pixel. The images are taken from the USC-SIPI image database that has been used in the existing research [9–12].

4.1. The performance of watermark embedding

In this set of experiments, PSNR and SSIM metrics are employed to compare the cover and watermarked images. An image with a higher value of PSNR and SSIM means the image has better quality and imperceptibility. The PSNR value shows the degree of

Algorithm 1
Texture recovery algorithm.

Input: n, r_{min}, r_{max}, r_{txt}, *textures*

set = zeros(n, n) + 127;
if (1 <= r_{txt} && r_{txt} <= size(*textures*, 3))
set(:,:) = *textures*(:,:, r_{txt});
End
p_{out} = zeros(n, n);
for i = 1 to n
for j = 1 to n
$weight_{max}$ = set(i, j);
$weight_{min}$ = 255 - $weight_{max}$;
$p_{out}(i, j) = \frac{(weight_{max} \cdot r_{max}) + (weight_{min} \cdot r_{min})}{255}$;
end for
end for

Output: p_{out}

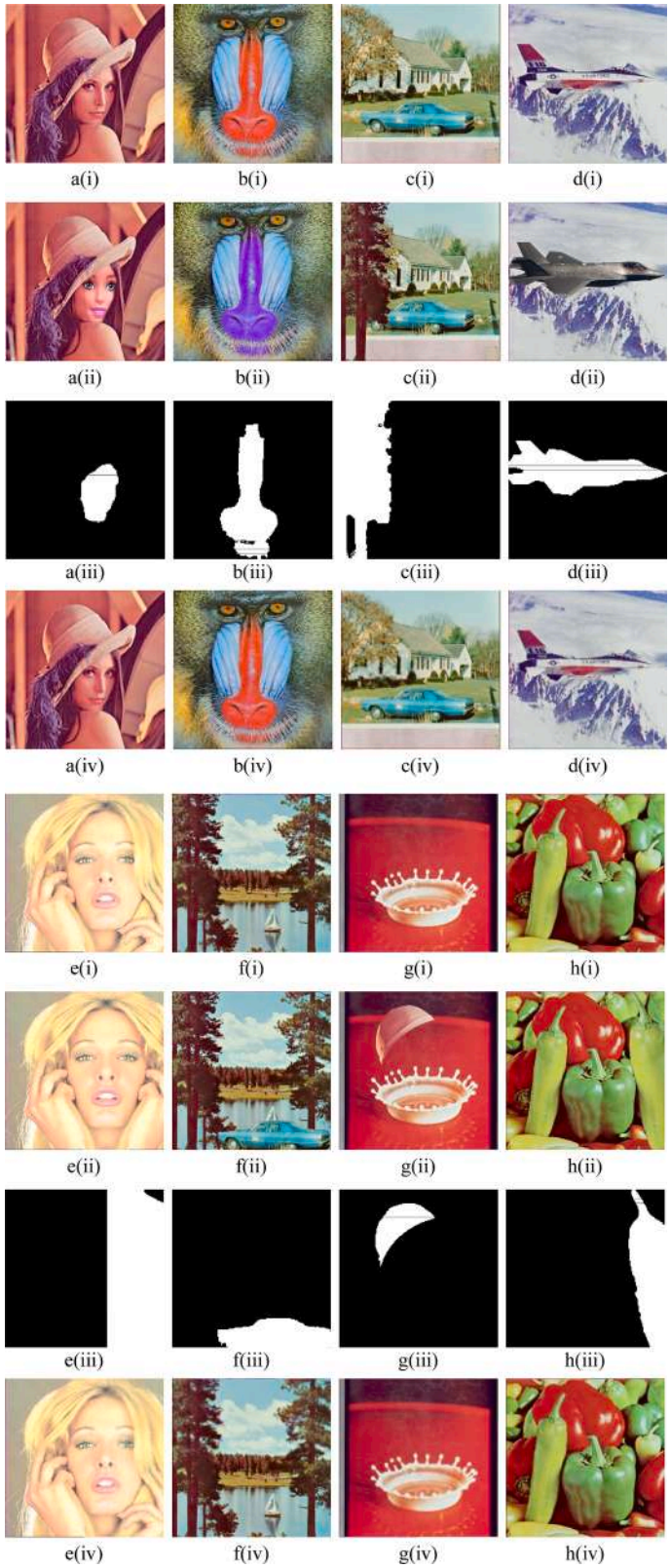

Fig. 6. The irregular attack applied to the test images (a) Lena image (b) Baboon image (c) House image (d) Airplane image (e) Tiffany image (f) Sailboat image (g) Splash image (h) Peppers image (i) Original image (ii) Tampered image (iii) Tamper detection (iv) Recovered image.

invisibility of the image. The SSIM metric measures the image similarity based on the Human Visual System (HVS). It compares the information of structure, luminance, and contrast for quality assessment [21]. Table 2 shows the performance of the watermarked images in terms of PSNR and SSIM values.

According to Table 2, the cover images with less texture, such as Peppers, Splash, and Tiffany, have a lower watermarked image quality than other images. It can be observed from the result presented by [9–11]. This is due to the large amount of watermark data that is embedded into two LSB has created a highly textured watermarked image. In addition, the watermark embedding process in the spatial domain may also contribute to the distortion of the watermarked image. However, the AuSR2 scheme can still maintain the image quality compared to the existing schemes. It can be achieved due to the utilization of the LSB shifting algorithm. It can be noticed that the AuSR2 scheme achieves the average PSNR and SSIM values of 45.91 dB and 0.9975. In comparison, the schemes by [9–11] replaced two LSB of the cover image with the watermark data, which produced an average PSNR and SSIM values of 44.08 dB and 0.9815, respectively. The scheme by [12] employed bit adjustment in the embedding process, producing average PSNR and SSIM values of 44.64 dB and 0.9840, respectively.

4.2. The performance of image authentication

The experiments were performed to test the proposed three-layers authentication scheme in the watermarked image using regular and irregular attacks. For regular attacks, all the test images are added with noises in the central region of the image. The image authentication performance is evaluated using the precision, F-1 score, and accuracy [12] with a confusion matrix. The confusion matrix consists of true positive (TP), false negative (FN), false positive (FP), and true negative (TN). True positive represents the number of modified blocks detected by the authentication algorithm. In contrast, a false negative represents the number of tampered blocks that are not detected by the authentication algorithm. The detection and tampered blocks ratio are TPR (true positive rate) and FNR (false-negative rate). False-positive denotes the number of untampered blocks mislabeled as tampered blocks, while true negative shows the number of unmodified blocks that are not marked as tampered blocks. The ratio between the detection and the untampered blocks is FPR (false positive rate) and TNR (true negative rate). A higher TPR and TNR values demonstrate the excellent performance of tampering detection. While a higher FNR and FPR mean the tamper detection cannot precisely detect the tampered area of the image. The tamper detection results of the AuSR2 scheme are shown in Table 3.

According to Table 3, the AuSR2 scheme achieves a TPR value of 1 and an FNR value of 0. In the meantime, a higher tampering rate produces a high FPR value. The false-positive detection occurred due to the tampering condition. If a pixel in a block has been tampered with, all the pixels on its block are detected as tampered areas. Those non-tampered pixels are considered as false positive detection. However, the false-positive detection will be recovered in the self-recovery process. Furthermore, the AuSR2 scheme outperforms the existing methods in terms of precision and TPR value, as shown in Table 4.

Based on Table 4, the AuSR2 scheme produces a comparable precision value to the existing methods except for an 80% tampering rate. The lower precision value is directly affected by a high FPR value, as shown in Tables 3 and 4.

As suggested in [21], five tampering attacks, such as normal tampering, copy-move, collage, vector quantization, and protocol attack, are used to evaluate the proposed recovery scheme. The tamper detection of the irregular attacks is listed in Table 5.

The irregular attack is visualized in Fig. 6. It shows that the AuSR2 scheme achieves high accuracy of 0.99. It can precisely detect the tampered area of the image. The proposed three-layer authentication provides high precision and F-1 score under various image forgeries. The proposed three-layers authentication scheme can achieve an average TPR value of 0.9996. It can be noticed that the tamper detection under various tampering rates using irregular attacks achieves a higher accuracy value than the regular attacks. The AuSR2 scheme under irregular attacks produces a slightly lower average TPR than under regular attacks due to the three-layers authentication scheme being unable to detect some edges on the tampered area, especially under irregular attacks. However, the AuSR2 scheme can achieve higher precision of 0.9922 and an F1-score of 0.9959 compared to the regular attacks.

4.3. The performance of self-recovery

One of the challenges in image authentication and self-recovery is determining recovery data for each block. Each block contains a limited space for embedding watermark data. Researchers [9,12,13,15] solve this challenge using an average block. As a result, the

Table 2
The comparison of the watermarked image between the existing schemes.

Cover image	Dadkhah [9]		Fan [10]		Tai [11]		Molina-Garcia [12]		Proposed AuSR2	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Airplane	44.12	0.9782	44.11	0.9781	44.12	0.9781	44.69	0.9812	46.05	0.9901
Baboon	44.14	0.9941	44.12	0.9941	44.14	0.9941	44.64	0.9947	46.06	0.9991
House	44.19	0.9815	44.18	0.9815	44.18	0.9815	44.66	0.9834	46.07	0.9970
Lena	44.13	0.9820	44.13	0.9820	44.12	0.9820	44.60	0.9840	46.06	0.9994
Peppers	44.06	0.9791	44.06	0.9791	44.06	0.9791	44.54	0.9816	45.87	0.9992
Sailboat	44.12	0.9868	44.10	0.9867	44.11	0.9868	44.61	0.9884	46.04	0.9982
Splash	44.08	0.9695	44.08	0.9695	44.09	0.9696	44.47	0.9737	45.93	0.9985
Tiffany	43.85	0.9806	43.84	0.9804	43.85	0.9805	44.87	0.9846	45.20	0.9986
Average	44.09	0.9815	44.08	0.9814	44.08	0.9815	44.64	0.9840	45.91	0.9975

Table 3
The tamper detection under various tampering rates on the regular attacks.

Tampering rate	TPR	FNR	FPR	TNR	Precision	F-1 Score	Accuracy
10	1.0000	0.0000	0.0041	0.9959	0.9959	0.9979	0.9979
20	1.0000	0.0000	0.0044	0.9956	0.9957	0.9978	0.9978
30	1.0000	0.0000	0.0031	0.9969	0.9970	0.9985	0.9985
40	1.0000	0.0000	0.0123	0.9877	0.9878	0.9939	0.9938
50	1.0000	0.0000	0.0166	0.9834	0.9836	0.9917	0.9917
60	1.0000	0.0000	0.0151	0.9849	0.9851	0.9925	0.9925
70	1.0000	0.0000	0.0326	0.9674	0.9684	0.9839	0.9837
80	1.0000	0.0000	0.0526	0.9474	0.9500	0.9744	0.9737

Table 4
The tamper detection comparison between the existing schemes.

Tampering rates		Dadkhah [9]	Fan [10]	Tai [11]	Molina-Garcia [12]	Proposed AuSR2
20	Precision	0.9658	0.9025	0.9657	0.9336	**0.9934**
	TPR	1.0000	1.0000	0.9963	1.0000	1.0000
40	Precision	0.9938	0.9697	0.9938	0.9697	**0.9959**
	TPR	1.0000	1.0000	0.9966	1.0000	1.0000
60	Precision	0.9801	0.9801	0.9801	0.9608	**0.9925**
	TPR	1.0000	1.0000	0.9962	1.0000	1.0000
80	Precision	**0.9870**	0.9701	**0.9870**	0.9701	0.9500
	TPR	1.0000	1.0000	0.9962	1.0000	1.0000

Table 5
The tamper detection under various tampering rates on the irregular attacks.

Watermarked images	Irregular attacks	Tampering rates	TPR	FNR	FPR	TNR	Precision	F-1 Score	Accuracy
Airplane	Normal	16.4	0.9998	0.0002	0.0078	0.9922	0.9923	0.9961	0.9960
Baboon	Protocol	14.2	0.9999	0.0001	0.0162	0.9838	0.9841	0.9919	0.9918
House	Collage	23.5	1.0000	0.0000	0.0266	0.9734	0.9741	0.9869	0.9867
Lena	Normal	6.08	0.9978	0.0022	0.0021	0.9979	0.9979	0.9978	0.9978
Peppers	Copy-move	15.4	0.9997	0.0003	0.0041	0.9959	0.9960	0.9978	0.9978
Sailboat	Vector-Q	10.3	0.9998	0.0002	0.0026	0.9974	0.9974	0.9986	0.9986
Splash	Collage	5.87	0.9997	0.0003	0.0031	0.9969	0.9970	0.9983	0.9983
Tiffany	Copy-move	34.9	1.0000	0.0000	0.0012	0.9988	0.9988	0.9994	0.9994

schemes failed to provide details of each pixel on the block. In contrast, the AuSR2 scheme utilizes multiple recovery data consisting of minimum, maximum, and texture values. The texture recovery provides a sharp texture in the recovered image. The experimental results show that the AuSR2 scheme produces a higher PSNR value than the self-recovery scheme using the average block value. It can be noticed in Fig. 7 that the AuSR2 scheme can preserve a sharp texture of the text "U.S. AIR FORCE" compared to the self-recovery scheme with the average block value. The AuSR2 scheme is compared to the self-recovery scheme using the average block with the sizes of 3×3 pixels in Fig. 7.

Fig. 7(b) shows the Airplane image in the center is tampered with using the regular attack with a 10% tampering rate. The AuSR2 scheme successfully localizes the tampered area on the image shown in Fig. 7(c). The AuSR2 scheme can recover the tampered image, as shown in Fig. 7(e). The experiments also evaluate the self-recovery scheme using the average block value of 3×3 pixels, as shown in Fig. 7(f). The visual comparison between the cover image, the texture set *E* recovery, and the average block recovery is shown in Fig. 7 (g), (h), and (i). The AuSR2 scheme produces a higher PSNR value of 37.23 dB of the recovered image. In comparison, the self-recovery scheme with the average block value produces 34.08 dB of PSNR value.

The scheme evaluates five texture sets in this experiment to show their performance under various tampering attacks. First, the image is partitioned into 3×3 non-overlapping blocks. Each block is classified according to the pattern sets. Next, the standard deviation of each texture is calculated. All textures in a texture set should have an equal standard deviation value. The variation in the standard deviation value may lead to the misclassification of an image block. Each image block is classified between five texture sets: *SetA, SetB, SetC, SetD* and *SetE*. Eqs. (5), (6) are utilized to determine the texture of an image block. Texture sets *A* and *B* utilize four textures, texture set *C* uses eight textures, while texture sets *D* and *E* employ 16 textures on their texture sets. Each texture in the texture sets *A, B*, and *C* has an equal standard deviation value. It means that each image block has an equal chance of being assigned by one of the textures. In contrast, texture set *D* has two different standard deviation values of 92.02 and 75.88. The experiment shows that texture set *D* has uneven classifications of image blocks. For example, the 11.25° texture has classified 151 blocks on the Airplane image. In contrast, the 33.75° texture has classified 10.719 blocks for the same image. It concludes that a lower standard deviation value has a higher chance of classifying the image blocks than a higher one. Thus, the texture set *E* is introduced to solve this issue. A single value of the standard deviation in the texture set *E* has spread the blocks' distribution evenly, as shown in texture sets *A, B*, and

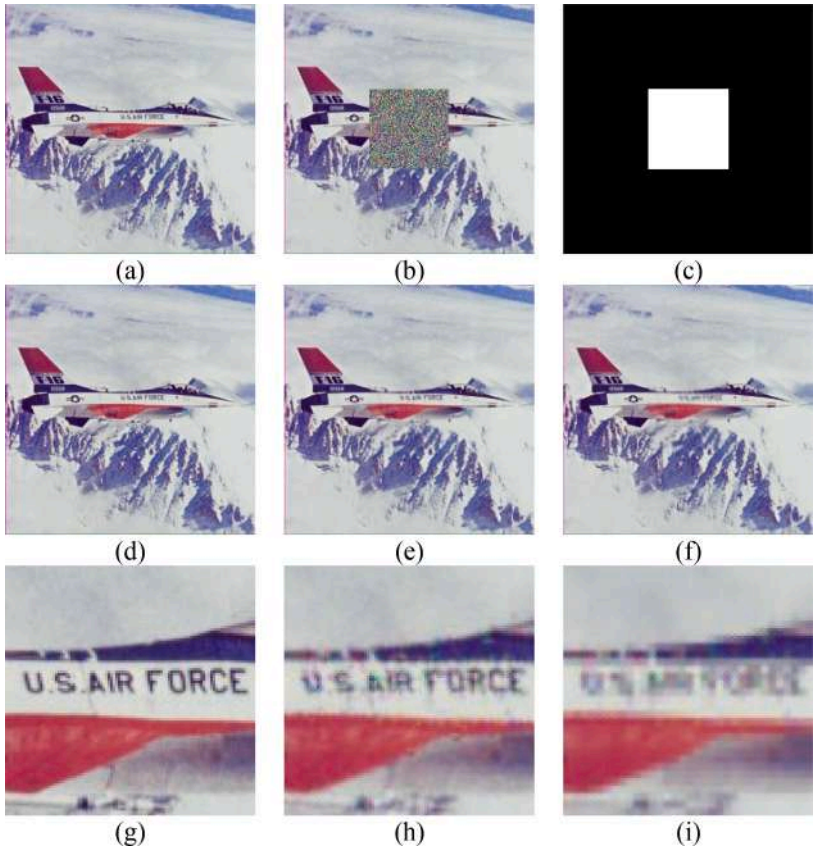

Fig. 7. The Airplane image (a) Watermarked image (b) Regular attack 10% (c) Tamper detection (d) Cover image (e) Texture set *E* recovery (f) Block average recovery (g) Cover image zoomed 500% (h) Texture set *E* recovery zoomed 500% (i) Block average recovery zoomed 500%.

C. Each texture set is then tested under various tampering rates. The quality of the recovered image is shown in Table 6.

The AuSR2 implements multiple recovery data, each with an equal probability of suffering the tamper coincidence problem. The tamper coincidence problem in the minimum and the maximum value can be solved using the image inpainting technique with less compromise. However, the AuSR2 cannot accurately restore the texture value if the texture value undergoes the tamper coincidence problem. Thus, each texture set produces different recovered image quality depending on the level of tampering rate, as described in Table 6. There are two phases to restoring the texture from the tamper coincidence problem. In the first phase, the texture is obtained from the same block location at different RGB channels. In the second phase, the texture is obtained from the surrounding block in the same channel. However, each of these phases has its drawbacks. The texture set *E* performs the highest recovery quality under a 10% tampering rate. At this level of tampering, the image inpainting technique can restore the recovery data with a minimum tamper coincidence problem. Next, texture set *C* provides the highest image quality under 20% to 30% tampering rates. The texture tamper coincidence problem frequently occurs, requiring the first phase of texture recovery. It assumes that the texture of the adjacent channel is identical to the selected channel. However, it has the possibility of receiving a different texture value. Thus, texture set *C* performs

Table 6
The PSNR and SSIM values of the proposed self-recovered image for five texture sets.

Tampering rates	With the average block		The AuSR2 scheme with the texture set									
			SetA		*SetB*		*SetC*		*SetD*		*SetE*	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
10	36.02	0.9901	37.73	0.9930	37.63	0.9929	38.09	0.9934	37.97	0.9933	38.11	0.9935
20	32.88	0.9814	33.88	0.9854	33.95	0.9855	34.25	0.9864	34.19	0.9862	34.21	0.9864
30	30.26	0.9666	30.85	0.9717	30.98	0.9723	31.17	0.9735	31.14	0.9733	31.10	0.9734
40	28.10	0.9444	28.43	0.9512	28.62	0.9525	28.72	0.9538	28.73	0.9537	28.63	0.9534
50	26.27	0.9189	26.26	0.9226	26.49	0.9255	26.53	0.9264	26.55	0.9265	26.43	0.9255
60	24.70	0.8924	24.44	0.8896	24.72	0.8947	24.70	0.8948	24.74	0.8954	24.60	0.8932
70	22.99	0.8580	22.50	0.8444	22.80	0.8524	22.75	0.8513	22.80	0.8526	22.66	0.8490
80	21.09	0.8138	20.50	0.7876	20.78	0.7989	20.71	0.7964	20.77	0.7985	20.64	0.7937

better than texture set *E* under 20% to 30% tampering rates since it has less texture count. Next, texture set *D* produces the highest recovery quality under a 40% to 60% tampering rate. At this level of tampering, the texture of adjacent channels has also suffered the tamper coincidence problem, which requires the second phase of texture recovery. However, it produces a less accurate texture prediction than the first phase. It also ignores the standard deviation of the texture set. Thus, texture set *D* has the highest quality of the recovered image. Next, under 70% to 80% tampering rates, the second phase of texture recovery fails to predict the texture of the tamper coincidence block, which leads to the overfitting problem [24]. As a result, the average block recovery performs slightly better at this level of tampering rate. However, the proposed scheme outperforms the average block recovery below 70% tampering rates. Tables 7 and 8 present the quality of the recovered images using the texture set *E*.

According to Tables 7 and 8, the AuSR2 scheme produces the lowest recovered image quality on the House image with 10% and 20% tampering rates, and the Baboon image in 30% up to 80% tampering rates. In contrast, the Splash image has the highest recovered image quality at 10% tampering rates, and the Tiffany image at 20% up to 80% tampering rates. This discrepancy is caused by the texture complexity of the images and the tampering location. The House image and the Baboon image are highly textured in the center of the image, while the Splash and Tiffany image has the smoothest texture. In the meantime, the regular attack is located in the central region of the images. Thus, a highly textured image has the lowest recovered image quality than other test images.

The proposed self-recovery scheme is evaluated under regular and irregular attacks. The scheme by Molina-Garcia [12] applied the image inpainting technique to produce the appearance of granulated effects on the recovered image. It is caused by only considering the average value of eight surrounding pixels to interpolate the missing pixels. This research proposes an image inpainting technique that considers the outward spiral direction of the tamper coincidence to interpolate the missing pixels. As a result, the AuSR2 scheme can recover the tampered image without any granulated effects, as presented in the scheme by Molina-Garcia [12]. In addition, the proposed self-recovery also considers the texture of each image block in the recovery process. Every pixel on each image block has different intensity levels depending on the image texture. The existing schemes by Dadkhah et al. [9,12,13] utilized an average block for recovering the image data. Consequently, each recovered block has a pixelated effect depending on the block sizes. The comparison of the PSNR and SSIM values between the existing scheme and the AuSR2 is shown in Table 9.

According to Table 9, the AuSR2 scheme outperforms the existing scheme under various tampering rates. In addition, the scheme produces a PSNR value of 38 dB under a 10% tampering rate, while the existing scheme has never achieved a PSNR value of 38 dB under the same tampering rate. The PSNR value comparison of the recovered images is visualized in Fig. 8. According to Fig. 8, the AuSR2 scheme outperforms all the existing schemes for the recovered image. The AuSR2 scheme can improve PSNR value by more than 5 dB. Generally, a higher tampering rate on the watermarked image will produce a low-quality recovered image. The scheme by Molina-Garcia et al. [12] achieved the SSIM value of 0.3958 under an 80% tampering rate, while the AuSR2 scheme can achieve the SSIM value of 0.7937 under the same tampering rate. It is due to the superiority of the image inpainting technique used in the AuSR2 scheme.

The scheme by Molina-Garcia et al. [12] solves the tamper coincidence problem using the average value of the surrounding pixels, which may also suffer the tamper coincidence problem. Thus, the quality of the recovered image is degraded rapidly at a higher tampering rate than the AuSR2 scheme. Furthermore, it can be noticed that the AuSR2 scheme can achieve a high SSIM value on the recovered image under a large tampering rate. Table 10 summarizes the comparison between AuSR2 and the existing methods in various terms.

5. Conclusion

An image authentication and self-recovery scheme called AuSR2 has been presented in this paper. Two authentication bits and sixteen recovery bits have been embedded into each block. Sixteen recovery bits consisted of six bits of minimum value, six bits of maximum value, and four bits of texture information. The authentication bits are embedded into the origin block location, while the recovery bits are embedded into three distinct locations. The embedding of the recovery bits is determined using the block map. The LSB shifting algorithm is then implemented for embedding the watermark data. The experimental results show that the AuSR2 scheme achieves a high PSNR value of 45.91 dB and an SSIM value of 0.9975 on the watermarked images. The LSB shifting algorithm increases the quality of the watermarked image by 2.8% compared to the existing methods. Furthermore, the watermarked images have been tested under various tampering attacks, such as regular and irregular attacks. The AuSR2 scheme provides a high TPR value of 1 and a precision value of 0.9830. In addition, the AuSR2 scheme can accurately detect the tampering area up to 100%. The accuracy is improved by 2.2% compared to the existing methods. The AuSR2 scheme has successfully recovered the tampered image and achieved a PSNR value of 38.11 dB and an SSIM value of 0.9935 under 10% tampering rates. The recovered image quality is improved by 2.8% compared to the existing scheme. The comparison analysis shows that the AuSR2 scheme outperforms the existing methods in terms of watermarked and recovered image quality.

CRediT authorship contribution statement

A. Aminuddin: Conception and design of the study, Analysis and/or interpretation of data, Drafting the manuscript, Approval of the version of the manuscript to be published. **F. Ernawan:** Acquisition of data, Analysis and/or interpretation of data, Revising the manuscript critically for important intellectual content, Approval of the version of the manuscript to be published.

Table 7
PSNR values of the recovered image under regular attacks.

Tampering rates	Airplane	Baboon	House	Lenna	Peppers	Sailboat	Splash	Tiffany	Average
10	37.23	37.43	36.07	37.86	38.00	37.44	40.80	40.07	38.11
20	32.53	32.83	32.00	33.82	34.86	33.53	37.13	36.99	34.21
30	29.78	28.54	29.12	30.75	32.28	30.01	33.61	34.68	31.10
40	27.43	25.32	26.65	28.53	30.00	27.15	31.16	32.79	28.63
50	25.42	22.84	24.48	26.71	27.67	24.77	28.98	30.52	26.43
60	23.84	21.03	22.90	25.04	25.59	22.76	27.11	28.51	24.60
70	21.99	19.42	21.14	23.23	23.09	20.74	25.08	26.55	22.66
80	20.23	17.96	19.19	21.27	20.38	18.77	22.83	24.52	20.64

Table 8
SSIM values of the recovered image under regular attacks.

Tampering rates	Airplane	Baboon	House	Lenna	Peppers	Sailboat	Splash	Tiffany	Average
10	0.9822	0.9959	0.9897	0.9962	0.9961	0.9940	0.9971	0.9966	0.9935
20	0.9677	0.9894	0.9765	0.9908	0.9925	0.9857	0.9943	0.9939	0.9864
30	0.9478	0.9635	0.9581	0.9821	0.9868	0.9691	0.9888	0.9906	0.9734
40	0.9182	0.9162	0.9287	0.9712	0.9789	0.9450	0.9823	0.9864	0.9534
50	0.8788	0.8502	0.8898	0.9586	0.9661	0.9094	0.9725	0.9782	0.9255
60	0.8341	0.7757	0.8476	0.9438	0.9490	0.8683	0.9615	0.9653	0.8932
70	0.7705	0.6847	0.7886	0.9244	0.9191	0.8086	0.9477	0.9482	0.8490
80	0.6936	0.5882	0.7156	0.8986	0.8786	0.7228	0.9230	0.9290	0.7937

Table 9
PSNR and SSIM comparison between the existing schemes and the AuSR2 under regular attacks.

Tampering rates	Dadkhah [9]		Fan [10]		Tai [11]		Molina-Garcia [12]		Proposed AuSR2	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
10	22.51	0.9131	31.47	0.9731	25.89	0.9731	37.34	0.9714	**38.11**	**0.9935**
20	17.32	0.7983	28.36	0.9502	20.57	0.9502	33.98	0.9390	**34.21**	**0.9864**
30	14.52	0.6855	21.62	0.8875	17.43	0.8875	31.28	0.8977	**31.10**	**0.9734**
40	12.64	0.5731	15.79	0.7230	15.21	0.7230	28.47	0.8368	**28.63**	**0.9534**
50	11.40	0.4704	15.69	0.7202	13.54	0.7202	26.00	0.7571	**26.43**	**0.9255**
60	10.39	0.3586	11.57	0.4249	12.01	0.4249	23.51	0.6460	**24.60**	**0.8932**
70	9.61	0.2506	11.57	0.4249	10.80	0.4249	21.23	0.5157	**22.66**	**0.8490**
80	9.03	0.1511	8.10	0.0094	9.81	0.0094	19.20	0.3958	**20.64**	**0.7937**

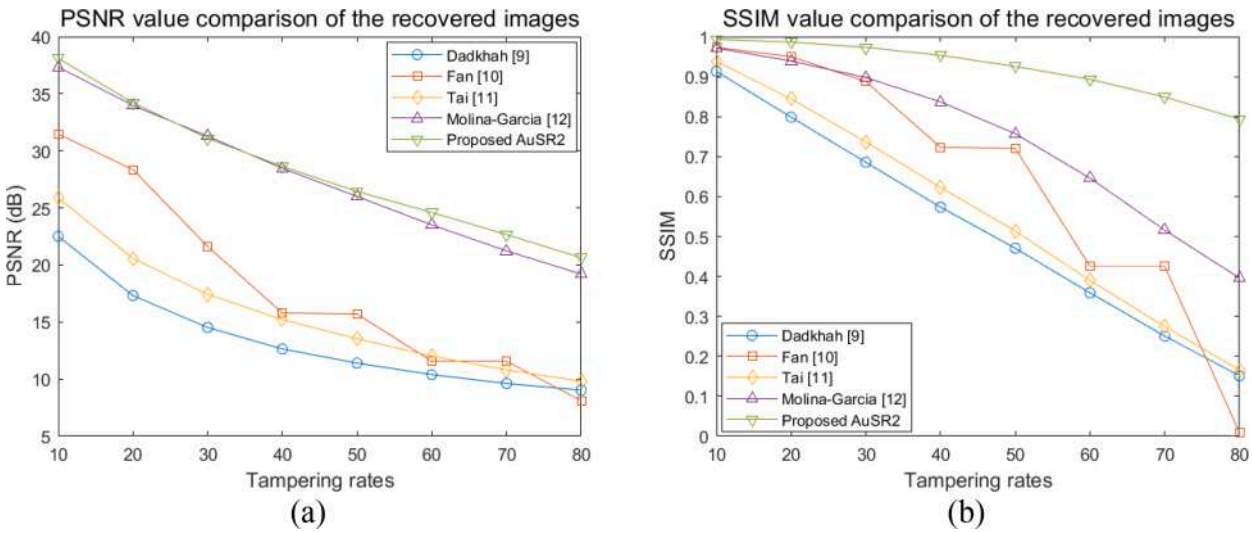

Fig. 8. The recovered images comparison between the existing schemes and the AuSR2 (a) PSNR (b) SSIM.

Table 10
The comparison between AuSR2 and the existing methods in various terms.

Features	Tong [13]	Dadkhah [9]	Singh [14]	Fan [10]	Tai [11]	Molina-Garcia [12]	AuSR1 [15]	AuSR2
Block Size	2 × 2	4 × 4	2 × 2	8 × 8	4 × 4	4 × 4	2 × 2	3 × 3
Authentication bit rate (bpp)	0.5	0.75	0.5	0.5	0.25	0.25	0.5	0.22
Recovery bit rate (bpp)	2.5	1.25	2.5	1.5	1.75	1.75	1.5	1.78
Recovery generation	Average	Average	DCT	SPIHT	IWT	Average	Average	Texture
Embedding location	3 LSB	2 LSB	3 LSB	2 LSB	2 LSB	2 LSB	2 LSB	2 LSB
PSNR of watermarked image (dB)	37.82	44.09	37.82	44.08	44.08	44.64	45.57	**45.91**
SSIM of watermarked image	0.9312	0.9815	0.9312	0.9814	0.9815	0.9840	0.9972	**0.9975**
Precision	0.9905	0.9817	0.9905	0.9556	0.9817	0.9586	**0.9955**	0.9830
True Positive Rate (TPR)	0.7492	1	0.7499	1	0.9963	1	1	1
PSNR of recovered image (dB)*	34.20	22.51	26.55	31.47	25.89	37.34	37.96	**38.11**
SSIM of recovered image (db)*	0.9733	0.9131	0.9290	0.9731	0.9384	0.9714	0.9928	**0.9935**

Note: * The recovered image quality under a 10% tampering rate

Declaration of Competing Interest

The authors declared that there is no conflict of interest regarding the publication of this research article. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was supported by Universiti Malaysia Pahang through the Research Grant Scheme (PDU203210 and PGRS1903186).

References

[1] Jafari Barani M, Yousefi Valandar M, Ayubi P. A new digital image tamper detection algorithm based on integer wavelet transform and secured by encrypted authentication sequence with 3D quantum map. Optik (Stuttg) Jun. 2019;187:205–22. https://doi.org/10.1016/j.ijleo.2019.04.074.
[2] Asghar K, Sun X, Rosin PL, Saddique M, Hussain M, Habib Z. Edge–texture feature-based image forgery detection with cross-dataset evaluation. Mach Vis Appl Oct. 2019;30(7–8):1243–62. Accessed: May 09, 2021. [Online]. Available: https://link.springer.com/article/10.1007/s00138-019-01048-2.
[3] Liu Y, Wang HH, Chen Y, Wu H, Wang HH. A passive forensic scheme for copy-move forgery based on superpixel segmentation and K-means clustering. Multimed Tools Appl Jan. 2020;79(1–2):477–500. Accessed: May 09, 2021. [Online]. Available: https://link.springer.com/article/10.1007/s11042-019-08044-8.
[4] Liu X, et al. A novel zero-watermarking scheme with enhanced distinguishability and robustness for volumetric medical imaging. Signal Process Image Commun Mar. 2021;92:116124.
[5] Zear A, Singh PK. Secure and robust color image dual watermarking based on LWT-DCT-SVD. Multimed Tools Appl Feb. 2021:1–18. https://doi.org/10.1007/S11042-020-10472-W.
[6] Laxmanika, Singh PK. Robust and imperceptible image watermarking technique based on SVD, DCT, BEMD and PSO in wavelet domain. Multimed Tools Appl Aug. 2021:1–26. https://doi.org/10.1007/S11042-021-11246-8.
[7] Fu J, Mao J, Xue D, Chen D. A watermarking scheme based on rotating vector for image content authentication. Soft Comput Sep. 2019;24(8):5755–72. https://doi.org/10.1007/S00500-019-04318-3.
[8] Rhayma H, Makhloufi A, Hamam H, Ben Hamida A. Semi-fragile watermarking scheme based on perceptual hash function (PHF) for image tampering detection. Multimed Tools Appl May 2021;80(17):26813–32. https://doi.org/10.1007/S11042-021-10886-0.
[9] Dadkhah S, Abd Manaf A, Hori Y, Ella Hassanien A, Sadeghi S. An effective SVD-based image tampering detection and self-recovery using active watermarking. Signal Process Image Commun Nov. 2014;29(10):1197–210. https://doi.org/10.1016/J.IMAGE.2014.09.001.
[10] Fan MQ, Wang HX. An enhanced fragile watermarking scheme to digital image protection and self-recovery. Signal Process Image Commun Aug. 2018;66:19–29. https://doi.org/10.1016/J.IMAGE.2018.04.003.
[11] Tai WL, Liao ZJ. Image self-recovery with watermark self-embedding. Signal Process Image Commun Jul. 2018;65:11–25. https://doi.org/10.1016/J.IMAGE.2018.03.011.
[12] Molina-Garcia J, Garcia-Salgado BP, Ponomaryov V, Reyes-Reyes R, Sadovnychiy S, Cruz-Ramos C. An effective fragile watermarking scheme for color image tampering detection and self-recovery. Signal Process Image Commun Feb. 2020;81:115725. https://doi.org/10.1016/j.image.2019.115725.
[13] Tong X, Liu Y, Zhang M, Chen Y. A novel chaos-based fragile watermarking for image tampering detection and self-recovery. Signal Process Image Commun Mar. 2013;28(3):301–8. https://doi.org/10.1016/j.image.2012.12.003.
[14] Singh D, Singh SK. Effective self-embedding watermarking scheme for image tampered detection and localization with recovery capability. J Vis Commun Image Represent Jul. 2016;38:775–89. https://doi.org/10.1016/j.jvcir.2016.04.023.
[15] Aminuddin A, Ernawan F. AuSR1: Authentication and self-recovery using a new image inpainting technique with LSB shifting in fragile image watermarking. J King Saud Univ Comput Inf Sci Feb. 2022. https://doi.org/10.1016/J.JKSUCI.2022.02.009.
[16] Zhang L, Wei D. Dual DCT-DWT-SVD digital watermarking algorithm based on particle swarm optimization. Multimed Tools Appl Jul. 2019;78(19):28003–23. https://doi.org/10.1007/S11042-019-07902-9.
[17] Mahto DK, Singh AK. A survey of color image watermarking: state-of-the-art and research directions. Comput Electr Eng Jul. 2021;93:107255. https://doi.org/10.1016/J.COMPELECENG.2021.107255.
[18] Chang C-CC, Lin C-CC, Su G-DD. An effective image self-recovery based fragile watermarking using self-adaptive weight-based compressed AMBTC. Multimed Tools Appl Sep. 2020;79(33–34):24795–824. https://doi.org/10.1007/s11042-020-09132-w.

[19] Hisham SI, Muhammad AN, Badshah G, Johari NH, Mohamad Zain J. Numbering with spiral pattern to prove authenticity and integrity in medical images. Pattern Anal Appl May 2016;20(4):1129–44. https://doi.org/10.1007/S10044-016-0552-0.
[20] Ernawan F, Aminuddin A, Nincarean D, Razak MFA, Firdaus A. Three layer authentications with a spiral block mapping to prove authenticity in medical images. Int J Adv Comput Sci Appl 2022;13(4). https://doi.org/10.14569/IJACSA.2022.0130425.
[21] Bolourian Haghighi B, Taherinia AH, Mohajerzadeh AH. TRLG: fragile blind quad watermarking for image tamper detection and recovery by providing compact digests with optimized quality using LWT and GA. Inf Sci (Ny) Jun. 2019;486:204–30. https://doi.org/10.1016/j.ins.2019.02.055.
[22] Bolourian Haghighi B, Taherinia AH, Harati A. TRLH: fragile and blind dual watermarking for image tamper detection and self-recovery based on lifting wavelet transform and halftoning technique. J Vis Commun Image Represent Jan. 2018;50:49–64. https://doi.org/10.1016/J.JVCIR.2017.09.017.
[23] Huang R, Liu H, Liao X, Sun S. A divide-and-conquer fragile self-embedding watermarking with adaptive payload. Multimed Tools Appl Sep. 2019;78(18): 26701–27. https://doi.org/10.1007/s11042-019-07802-y.
[24] Li DZ, Wang W, Ismail F. A selective boosting technique for pattern classification. Neurocomputing May 2015;156:186–92. https://doi.org/10.1016/J.NEUCOM.2014.12.063.

Afrig Aminuddin received his Bachelor in Informatics Engineering from Universitas Amikom Yogyakarta in 2014. He received his Master in Information Technology from the Faculty of Engineering, Universitas Gadjah Mada, in 2017. His research interests include digital watermarking, image processing, and computer vision. (Scopus ID: 57221994672)

Ferda Ernawan received his Master in Software Engineering and Intelligence and Ph.D. in image processing from the Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, in 2011 and 2014, respectively. His research interests include image compression, digital watermarking, and steganography (Scopus ID: 53663438800).