Medical & Biological Engineering & Computing (2022) 60:701–717
https://doi.org/10.1007/s11517-021-02499-4

ORIGINAL ARTICLE

A simple embed over encryption scheme for DICOM images using Bülban Map

Veerappan Manikandan[1] · **Rengarajan Amirtharajan**[1]

Received: 18 December 2020 / Accepted: 23 December 2021 / Published online: 17 January 2022

Abstract
With the onset of any pandemic, the medical image database is bound to increase. These medical images are prone to attack by hackers for their medical data and patient health information. To safeguard these medical images, a new algorithm is proposed. The algorithm involves secretly embedding the patient identification number into the medical image and encrypting the medical image, protecting the patient's identity and the patient's medical condition from hackers. The encryption algorithm involved a single stage of confusion and two stages of diffusion. The confusion operation was performed using the key generated by the Bülban map. The first stage of diffusion was done in the transform domain, using 5/3 transformation. The second diffusion stage was performed in the spatial domain by altering the pixel values using the key. The algorithm was tested on over 30 DICOM (Digital Imaging and Communications in Medicine) images taken from Open Science Framework (OSF), a public database for COVID-19 patients. The algorithm could resist the statistical attacks upon analysis, providing a PSNR of 7.084 dB and entropy of 15.9815 bits for the cipher image. The correlation coefficients for the cipher image were 0.0275, -0.0027, 0.018 in horizontal, vertical and diagonal directions. The keyspace was $2^{((M-1)\times N)\times 16}$, with M the number of rows and N the number of columns in the image. The key sensitivity was high. The test results and metric analysis prove that the algorithm is an effective one for embedding and encryption.

Keywords COVID-19 database · DICOM encryption · Bülban map · Shuffling unit · Chaotic maps · IWT

1 Introduction

Safeguarding multimedia content from unintended access is still a work in progress. Over the years, the research community developed many algorithms for protecting multimedia content, with success [1–11]. While some algorithms such as [12–15] focus on the protection of grey and RGB images, some other algorithms such as [7–11, 16–23] focus on providing security to Medical images, especially Digital Imaging and Communications in Medicine (DICOM) [7–11]. It was necessary to design an algorithm that focuses on both robustness and reduced time of operation. While some algorithms focused on algorithm ruggedness and robustness[2, 4, 5, 18, 19], some focused on reducing the execution time [1, 3, 12].

Elliptic curve analogue ElGamal cryptosystem was used to encrypt multiple medical images in [1]. Key was generated using Mersenne Twister pseudo-random generator. A new method was used wherein the pixels were base converted, and thereby the need for discrete coordinate representation was ruled out. Such base conversion in the elliptic curve enabled solving data expansion problems and reducing the number of elliptic curve calculations. Medical image encryption was performed using the 3D-logistic map in [2]. The key was generated using a 2D-logistic-adjusted-sine map [2D-LASM]. The generated random number was diffused into the neighbourhood of each pixel. The image was divided into non-overlapping blocks, and scrambling of non-overlapping blocks was done using the 3D-logistic map. Vigenère cipher algorithm was used for encrypting the image in [3]. Key was generated using Arnold transform, and the key matrix was of the order 16×16. The algorithm

✉ Rengarajan Amirtharajan
amir@ece.sastra.edu

[1] School of Electrical & Electronics Engineering, SASTRA Deemed University, Thanjavur 613 401, India

was implemented as a block cipher. Hermite chaotic neural network was used in medical image encryption in [4]. The 1D chaotic map was used to generate the chaotic sequences. The generated chaotic sequences were used to train the Hermite chaotic neural network, and two such key streams were generated to encrypt the medical images.

Medical image encryption using multiple chaotic maps was done in [5]. The algorithm used sine, cubic and logistic maps for increasing the robustness of encryption. Usage of multiple chaotic maps increased the keyspace and key sensitivity. The algorithm was robust to attacks. Chaotic medical image encryption based on bit-plane decomposition using the permutation-diffusion method was done in [6]. The pixels in the image were subdivided, and higher-order most significant bits were permuted. Permutation was done using the Arnold map, and diffusion was performed using the Henon map. Medical images were encrypted using DNA coding in [12]. Key was generated by merging Henon, Sine and Tent map (HST). Before encryption, the patient information was hidden in the medical image using the least significant bit technique (LSB). HST and DNA coding were used together to encrypt the medical images. As a result, the key sensitivity was higher in the proposed algorithm. An algorithm for medical image security using Arnold map was done in [16]. The key was generated using the Arnold map, and scrambling of pixels was done using the sequence generated using the Arnold map. The medical image encryption using wavelet transform was proposed in [17]. The algorithm involved watermarking patient information in the medical image and encrypting the watermarked image using any reference image and doctor's fingerprint image. The encryption was done through integer wavelet transform. The algorithm provided security through biometric authentication of the doctor's image.

A lossless edge map was used to encrypt medical images in [18]. The medical image was decompressed using Fibonacci bit-plane decomposition. The decomposed bit planes were XORed with the edge maps. The obtained images were diffused using the chaotic sequence generated by a double sine map. Compressive sensing and pixel swapping were used for image encryption in [19]. The algorithm involved two phases. In the first phase, the plain image was compressed and encrypted by the Bernoulli measurement matrix. The chaotic nature of the Bernoulli measurement matrix was done using the Chebyshev map. In the second phase, a logistic map generated the key and permutation diffusion to provide extended security. Dual hyper-chaotic map and DNA was used for providing image encryption in [20]. The permutation and diffusion operations were performed on selected pixels of the image. The DNA coding rules were applied based on the pixel position. Based on the game theory, an image encryption algorithm was proposed in [21]. The region of interest (ROI) selection method was employed to choose the ROI, and encryption was done using the key generated using Quantum cell neural network (QCNN) hyper chaotic system. The block swapping method was used for image encryption in [13]. The image was subdivided into blocks, and the blocks were swapped using the key. The key was generated using a one-dimensional logistic sine map. Diffusion of pixels within and between the blocks was performed to eliminate the correlation between the pixels.

Encryption of pixels based on modular exponentiation and Henon map was proposed in [22]. Key was generated using modular exponentiation and permuted using Henon map. By XORing the image pixels and the random key, the cipher image was produced. Discrete wavelet transform (DWT) was used for image encryption in [23]. The medical image was compressed and applied with DWT. The image was subdivided into blocks, and the block permutation was done. The resultant image was permuted with a key. A new chaotic map named Bülban map was used for image encryption in [14]. The algorithm used spatial domain encryption, using circular shifts of rows and columns in the image. The image pixels were XORed with the generated key to produce the cipher image. The algorithm designed in [14] involved fewer resources and low computation cost. But, the algorithm was developed in the spatial domain and was implemented only for grey-scale and RGB images.

The DICOM image database is increasing day by day, and it is necessary to design a robust, cost-efficient algorithm, with low computation time, for providing security to the images. The proposed algorithm involves low computation cost and reduces algorithm design resources, hence reducing execution time. The proposed algorithm was designed in the transform domain, involves few resources and tested for DICOM images. The proposed algorithm also consists of the embedding of data in the image.

The proposed paper focuses on a new methodology for encryption using the Bülban map and Le Gall 5/3 wavelet and performs steganography by simple embedding. This is explained in Sect. 2, along with relevant figures. The algorithm's efficacy was tested for images of various sizes, and analysis of the algorithm was checked using metrics. The timing analysis was also performed to check the computation time. All these tests are explained in Sect. 3, with relevant tables and figures. The algorithm performance is evaluated with metrics and compared with contemporary algorithms in Sect. 4. Finally, the implications of this proposed algorithm and possible future enhancements and improvements are given in Sect. 5, as concluding remarks.

The significant contributions of the proposed method are:

1. The first use of Bülban map for DICOM image encryption in the transform domain and embedding of patient number
2. Designing an algorithm with increased keyspace of $2^{((M-1)\times N)\times 16}$, mitigating the brute force and chosen plain text attack.
3. The utilisation of fewer resources and low computational cost due to the usage of Bülban map and Le Gall 5/3 wavelet.

2 Proposed Methodology

The proposed method involves generating three keys using the Bülban map, performing Le Gall 5/3 DWT transformations, binarising patient numbers and embedding the patient number into the encrypted image.

The algorithm proposed in this paper is broadly designed for information security, applying both cryptographic techniques and data hiding. Cryptography is employed as a stream cipher, wherein the image size is equal to the key size. The key used here is a symmetric type, wherein a single key is used for encryption and decryption. The advantage of using a single key is that it involves less computation.

2.1 Key Generation

The effectiveness of the algorithm and hence the encryption lies in the effectiveness of the key being unpredictable. Unpredictability could be achieved by providing randomness in the values for the key. Given the initial set of conditions (values), the random generation of values, as a sequence, using chaotic maps, provides the necessary unpredictability. Considering the application and the required simplicity, the Bülban map[24] is chosen for generating the random chaotic sequence from which the keys are generated. This map is one-dimensional and has the simplest equations similar to the 1D logistic map to produce the random chaotic sequence. Besides, the Bülban map provides higher chaotic behaviour on a broader range of parameter values. This also enhances the keyspace and increases the security level. The governing equation for the Bülban Map is given by Eq. 1.

$$x_{n+1} = x_n \times \sqrt{\frac{a}{x_n - b}} \tag{1}$$

Here, a and b are real and positive constants. To avoid the complex values in the sequence, the values for the variables used under the square root, in Eq. 1, must be above zero for all iterations. Hence, a, b and the initial value of x are all chosen to be above zero. Thus, a good random sequence is possible by providing the right value to the variables and the constants. Figure 1 gives the model for key generation. With $b = 4 \times a$, $a = 0.5$ and $x > b$, it is possible to get a good random sequence. As the input parameters are of real values, the generated chaotic sequence is also of type real. With x_1 given as an initial value, the equation is iterated $n + 1$ times(given by i, in Fig. 1), with n ranging from 1 to $(M \times N - 1)$, where M is the number of rows and N is the number of columns of the input image, which has to be encrypted. This gives a chaotic sequence with $M \times N$ values.

With the requirement of two keys, one key is obtained by converting the real values into integer and normalising to fall between 0 and 255, such that the first key set sequence K_1 has values in the range between 0 and 255 and the number of iterations, 'i', is given by $((M-1) \times N)$. The second key set sequence K_2, a position map, is obtained by arranging the original generated real value sequence in ascending order and obtaining the position map between the originally generated sequence and the arranged sequence. In MATLAB R2018b, *[arranged, position map] = sort (key value, 'ascend').* Figure 1, *'arranged'* value is given by *'k'*, and *'position map'* is given by *'p'*. Figure 2 shows how all the keys are generated.

This key set K_2 comprises all values, depicting the position, ranging from 1 to $M \times N$. There is also another key K_3, which has the same initial parameters and undergoes the same process as that of the generation of key K_1, but with a reduced number of values, as the number of iterations is far less and is equivalent to the number of elements in the approximation coefficient of the transformed image. Hence, the number of iterations, 'i', is given by $(rL \times CL)$, where 'rL' is the number of rows and 'cL' is the number of columns in the approximation coefficient of the transformed image. Thus, the key generation can be represented by the pseudo-code as given in Algorithm 1. To check the sensitivity of the system generated using the chaotic equation, the Lyapunov exponent is used. As the Lyapunov exponent for the chaotic system designed using the Bülban map generates positive value, this ascertains that the system is chaotic.

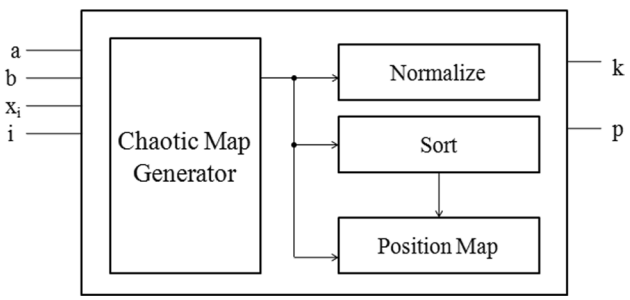

Fig. 1 Generation of key and position map using chaotic map

Algorithm 1: Key Generation

Input: a, b, x_1, M, N, rL, cL
Output: K1, K2, K3
begin
z= 1 to ((M-1) × N)
for all z,
Calculate x_{z+1} using Bülban map;
end for
b1_map=x;
b1_ascend=*sort_ascend*(b1_map);
p_map=*positonfind*(b1_map, b1_ascend);
K2=p_map;
b1_int=*conv2int*(b1_map);
b1_norm=b1_int *mod* 256;
K1 = b_norm;
z= 1 to (rL x cL)
for all z,
Calculate x_{z+1} using Bülban map;
end for
b2_map=x;
b2_int=*conv2int*(b2_map);
b2_norm=b2_int *mod* 256;
K3 = b2_norm;
end

2.2 Le Gall 5/3 Transform

Le Gall 5/3 transform, also known as the 5/3 lifting transform, has two vanishing moments and is the shortest symmetrical bi-orthogonal wavelet to avoid boundary artefacts. Therefore, the image could be decomposed into high-frequency and low-frequency components in the simplest possible way by using Le Gall 5/3 transform. The Le Gall 5/3 wavelet transform involves five taps of low-pass analysis filters and three taps of high-pass analysis filters, producing approximation coefficient LL and detailed coefficient LH, HL and HH.

Decomposing an image into its coefficients by applying transforms, altering the coefficient and applying inverse transform could provide good encryption. As the medical images in general and specifically DICOM have many high-frequency components, using 5/3 transformation would give good results because 5/3 transformations preserve the high-frequency components efficiently. Le Gall 5/3 discrete wavelet transform (DWT) is applied using the lifting scheme. Applying a lifting technique-based integer-to-integer wavelet transform (ITI-WT) would ensure that the pixel value in the image is reconstructed precisely without any data loss. After transformation, the approximate coefficient LL alone is altered without disturbing the detailed coefficients.

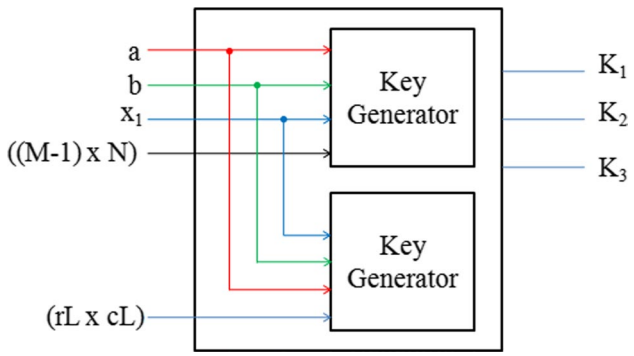

Fig. 2 Three key generation

2.3 Dataset Description

COVID-19 DICOM image datasets were taken from a public database, maintained by Open Science Framework (OSF). The database contains a vast volume of DICOM images of several subjects, compressed in tar.gz format. All these images could be downloaded and decompressed to obtain the.dcm image files. The images state the severity of COVID-19.

2.4 Algorithm Design

The algorithm involves the separation of images, encryption involving confusion and two stages of diffusion and embedding of patient number, known as a tag, into the first row of

the image. The design flow for the proposed algorithm is given in Fig. 3.

2.4.1 Encryption

The input to the algorithm is an (M × N) sized DICOM image, with M rows and N columns. The first row comprising N columns is separated from the input DICOM image of size (M × N). Then, leaving out the remnant (1 × N) pixels, the plain image of size (M-1) × N, extracted from the input DICOM image, is reordered using Key K_2. The algorithm is designed specifically with a focus on DICOM images, and hence the re-ordering of pixels as a first stage gives good results. However, this stage provides confusion and is followed by two stages of diffusion, one in the transform domain. Finally, the image is decomposed into a set of integer coefficients. The diffusion with the key is done in the coefficients, and the other in the spatial domain where the pixel values are altered with the key to produce diffusion. Figure 4 shows the encryption and embedding process of the algorithm.

The confusion, along with two stages of diffusion, provides a good encrypted image. The re-ordered (confused) image is applied with Le Gall 5/3 DWT to perform the first diffusion stage using the lifting scheme. The transformation applied is a 1D and is of level 0. This could be done in MATLAB R2018b using *ls = liftwave('cdf2.2','Int2Int'); [LL,HL,LH,HH] = lwt2(inpimg,ls)*. Keeping the detailed coefficient LH, HL and HH untouched, the approximation coefficient LL is altered by XORing with the key K_1. The modified approximation coefficient LL' is recombined with the detailed coefficients using inverse Le Gall 5/3 DWT, using the lifting scheme.

This could be done in MATLAB R2018b using *outimg = ilwt2(LL_dash,HL,LH,HH,ls)*will suffice. Then, the resultant pre-cipher image is subjected to the second stage of diffusion. Herein, all the pixels of this pre-cipher image are XORed with the key K_3, to produce the encrypted cipher image. Finally, this cipher image is recombined with the remnant image to obtain the complete combined image of the same size (M × N).

2.4.2 Embedding

According to the National Electrical Manufacturers Association (NEMA), the DICOM images (http://dicom.nema.org/Dicom/2011/11_10pu.pdf) do have the patient information in the image. The patient information includes the patient identification and medical records, which had to be kept confidential. Although this algorithm is for storage encryption, the hospital that stores these images has its hospital-specific patient number, which need not be included in the DICOM image information. The patient number serves as a tag for identifying the patient, which is not related to the patient's other data.

The DICOM standardisation is not compromised, as the tag is used only for the algorithm purpose, which is different from the original patient identification number.

The patient number is binarised, and the first row of the combined image is embedded with the binarised patient number. Thus, the altercation in the first row of the encrypted DICOM image would not impact the contents

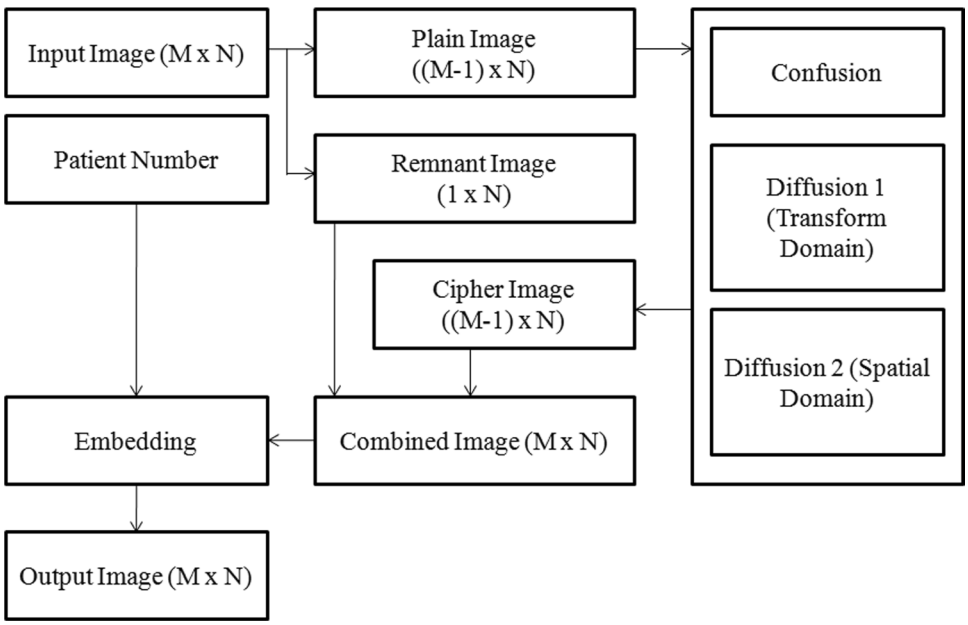

Fig. 3 Design flow

of the DICOM image after decryption. This is because the first row is excluded during the encryption process. The entire process of encryption and embedding represented by pseudo-code is given in Algorithm 2.

Algorithm 2: Encryption and Embedding

Input: DICOM image, Patient number, K1, K2, K3
Output: Encrypted image, embedded with patient information
I ← Input DICOM image; P ← $I_{((M-1)\times N)}$; R ← $I_{(1\times N)}$; PN← Patient number;
begin
[rP, cP] =*size*(P);
Prow=*rowvec*(P);
j= 1 to ((M-1) × N);
for all j,
temp1=K2 (j);
con_im_temp(j) = Prow(temp1);
end for
con_im=*reshape*(con_im_temp)$_{(rP,\ cP)}$
[LL, HL, LH, HH]= *5/3_IWT*(con_im);
[rL, cL] =*size*(LL);
LL_row=*rowvec*(LL);
temp2= K1 ᵗ LL_row;
LL'=*reshape*(temp2)$_{(rL,cL)}$
dif1_im=*5/3_inv_IWT*([LL', HL, LH, HH]);
[rD, cD] =*size*(dif1_im);
temp3=*rowvec*(dif1_im);
temp4= K3 ᵗ temp3;
dif_img=*reshape*(temp4)$_{(rD,\ cD)}$
PN_bin=*binary*(PN);
R_emb=*embed*(PN_bin, R);
Out_im=*combine*(dif_img, R_emb);
end

2.4.3 Classification and Decryption

The embedded encrypted image is stored as a database locally. To access any specific patient's medical images in the database, the patient number is first used to locate the images specific to the patient among the scores of images available in the database. This is done by applying the algorithm that compares the image's first row with the patient number. Once the patient's images are located, they had to be decrypted to access the information in the image. Having access to the patient's number could only be used to locate the patient-related images, but they cannot be accessed, as they are available in encrypted form. Therefore, decryption could be done only by having access to the keys used for encryption. The operations involved in the encryption process are all reversible. Hence, the algorithm is applied to the cipher image to decipher and obtain the plain image using the keys. Decryption involves the removal of two stages of diffusion and removal of one stage of confusion operation. Figure 5 gives the extraction and decryption process. Hence, the decryption of images is done efficiently, and the accurate plain image was recovered from the cipher image for analysis.

2.5 Performance Evaluation

To test the efficacy of the algorithm, the cipher image is subjected to metric evaluation. These metrics are used to check the visual perceptibility, resistance to differential attacks and statistical attacks.

2.5.1 Peak Signal to Noise Ratio

Peak signal to noise ratio (PSNR) could be used as a quality metric to analyse the image's visual quality. PSNR is applied between the input image and the encrypted image and is dependent on the mean square error (MSE) of the

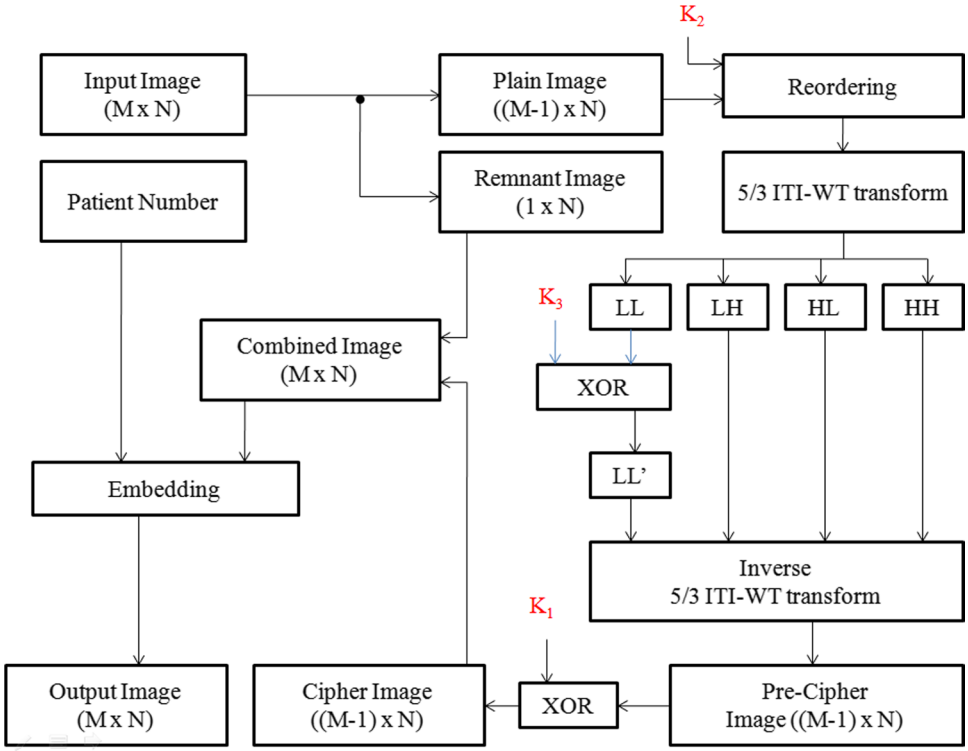

Fig. 4 Encryption and embedding

input image and the encrypted image. MSE is used to estimate the error. If the error is high, PSNR is low. Hence, the higher the error between the input and encrypted images, the poor input quality. This shows that lower the PSNR, the image is of poor quality. PSNR is given by 10 $\log_{10}$ (R^2/(MSE)). Here, 'R' is the maximum possible pixel intensity in an image.

2.5.2 Entropy

The entropy gives the average information of an image. It provides the minimum number of bits required to represent each pixel in an image. The encryption technique's significance can be found by entropy, as it gives the measure of randomness in an image—the higher the entropy, the better

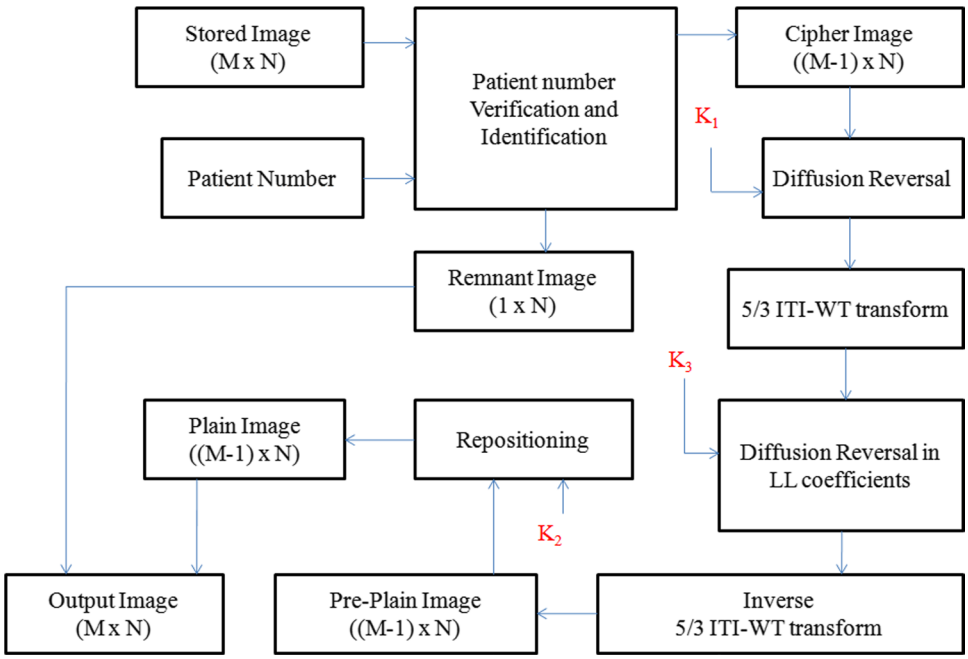

Fig. 5 Extraction and decryption

the encryption technique. For a 16-bit image, the maximum possible entropy is 16.

2.5.3 Correlation co-efficient

The relationship between the neighbouring pixels is found out by finding the degree of correlation between the pixels. The value ranges from -1 to + 1. Therefore, there should be no correlation between the pixels in the image for a cipher image, and the correlation coefficient must be near zero.

The resultant cipher image must be a meaningless object to ensure that the algorithm works perfectly in encrypting the plain image. Therefore, the cipher image is subjected to the PSNR analysis for finding out the visual perception. The lower the PSNR, the higher the error between the plain image and the cipher image, and hence the higher the possibility of the cipher image being a meaningless object.

The number of bits required to represent each pixel in the image could be found by finding the entropy of the image. For example, if each pixel requires 8 bits (for an image of bit depth 8) to get represented, it creates uncertainty about the pixel values, leading to a meaningless object.

Finding the correlation coefficient between the neighbouring pixels makes it easier to determine any relationship between the adjacent pixels in the cipher image. If the encrypted cipher image is a meaningless object, then the correlation coefficient is nearly zero.

3 Results

The algorithm was run in Intel Core i 7–6700 CPU, with the frequency of 3.4 GHz, and 64-bit Windows operating system. The tool used was MATLAB R2018b. The algorithm was fed with DICOM images from the COVID-19 database. Thirty-one images from [25] were tested from the database, and the test results were tabulated. The algorithm worked for images of varied sizes, and all the images in the database were of bit depth 16. The algorithm was analysed and tested at the end of each stage (Confusion, Diffusion 1 and Diffusion 2), and the results are tabulated. This is shown in Table 1. The tabulated results in Table 1 are for a DICOM image of bit depth 16. The entropy gradually increases at each stage and reaches 15.9831 bits, which is close to 16 bits. The PSNR decreases at each stage and reaches 8.2356 dB. The correlation coefficient in horizontal, vertical and diagonal directions becomes 0.0316, -0.0084 and 0.0316 at the end of the three stages, close to zero. The last embedding operation does not have a greater effect on these metric values. The optimum values for each metric are discussed in Sect. 2. It could be observed that an entirely encrypted meaningless output image is obtained at the end of three stages involving confusion and two stages of diffusion. The obtained image is further used for embedding. This is shown in Fig. 6.

3.1 Statistical tests

A good encryption algorithm must produce a visually meaningless object as the output. This could be tested by calculating the PSNR of the output image, and the ideal encrypted image should preferably have a PNSR below 10 dB. [26] [27]. PSNR gives the error between the original image and the encrypted image. The lower the PSNR, the higher the error. Upon testing, the average PSNR obtained was 7.0840 dB, which is considerable, a good value. To mathematically say that the output is meaningless and random, it is sufficient to prove that each image pixel requires all the bits, equivalent to bit depth, to be represented. This is calculated by finding the entropy of the output image. With the plain image of bit depth 16, an ideal encrypted image should have an entropy of 16 bits. On average, this algorithm gives an entropy of 15.9815 bits. This proves that the algorithm results in an excellent encrypted image. The image histogram visualises the count of the number of times each pixel occurs in an image. For an entirely encrypted image, each pixel value should be present an equal number of times, and hence an ideal histogram for a cipher image will be flat. The histogram obtained for the cipher image of the proposed algorithm is near flat, considering the bigger size of the image. Hence, it could be said that the encryption

Table 1 Metrics analysis—stage wise

Stages of encryption *	Metrics				
	Entropy (bits)	PSNR (dB)	Correlation coefficient		
			Horizontal	Vertical	Diagonal
Confusion	0.1621	18.3813	0.1254	-0.0023	0.0204
Diffusion 1	15.6919	10.4380	0.6666	0.6508	0.4184
Diffusion 2	15.9831	8.2356	0.0316	-0.0084	0.0316
Embedding	15.9831	8.2356	0.0559	0.0204	0.0300

***Image File:** 1004172576146205208 6.dcm [25]

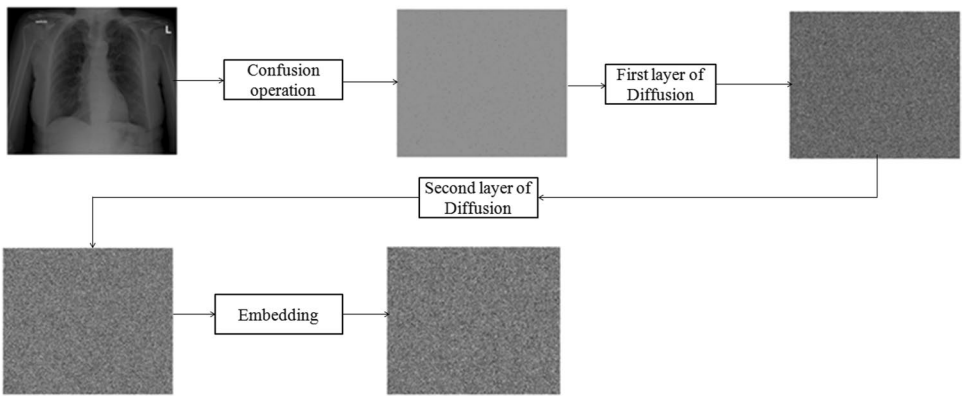

Fig. 6 Encryption stages and corresponding outputs. Input image 10041725761462052086.dcm [25]

algorithm provides a good cipher. Figure 7 depicts the metric analysis pictorially.

The scattered plot obtained while calculating the correlation coefficient determines that the adjacent pixel values are far from one another in all three directions, stating that a cipher image is a meaningless object. The results gain weight if it is found that the nearby adjacent pixels in the output are not correlated. The adjacent pixels should be

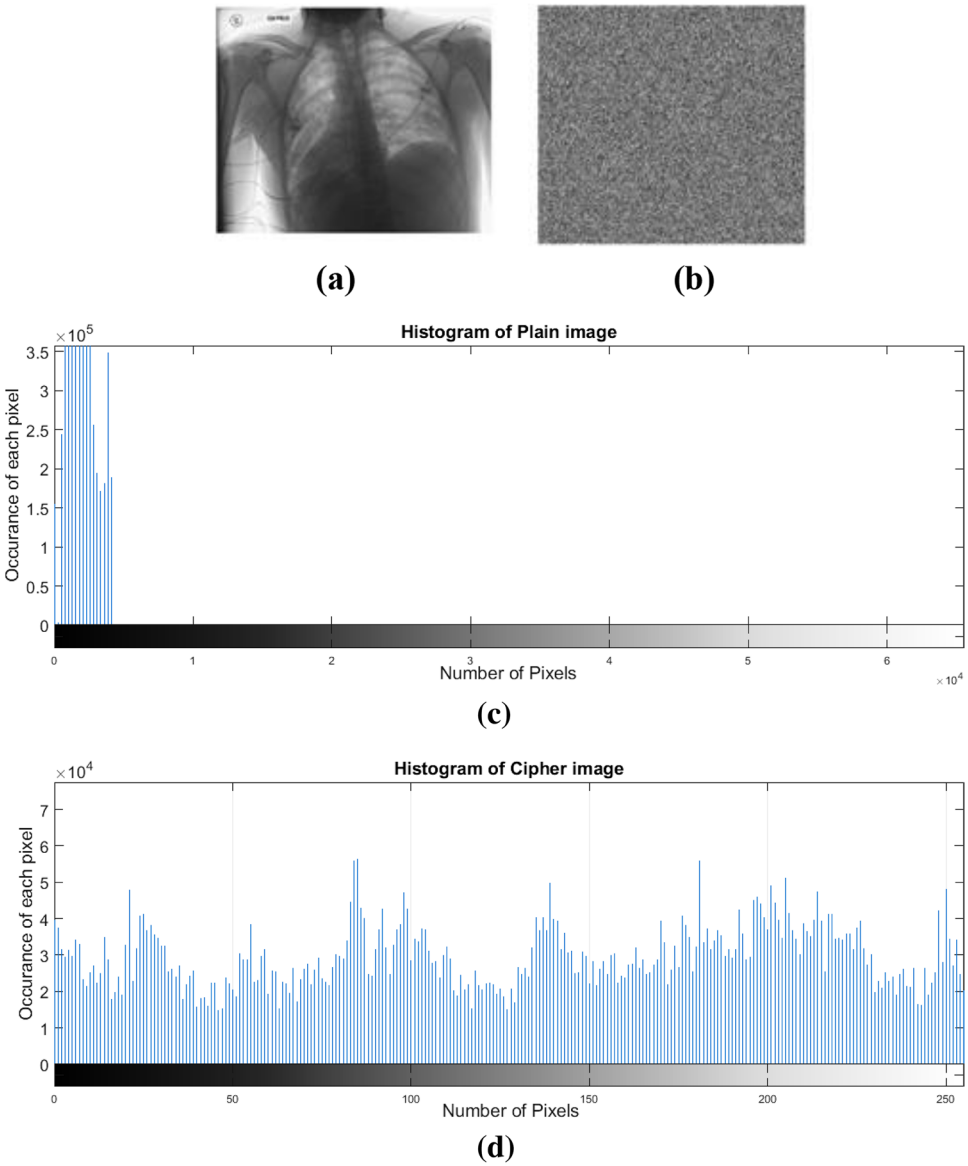

Fig. 7 Metric analysis (**a**) input image 10,104,326,480,143,296,722.dcm [25] (**b**) cipher image (**c**) histogram of input image **d**) histogram of cipher image (**e**) horizontal correlation of plain image (0.9943) (**f**) vertical correlation of plain image (0.9937) (**g**) diagonal correlation of plain image (0.9981) (**h**) horizontal correlation of cipher image (-0.0198) (**i**) vertical correlation of cipher image (-0.0103) (**j**) diagonal correlation of cipher image (0.0196)

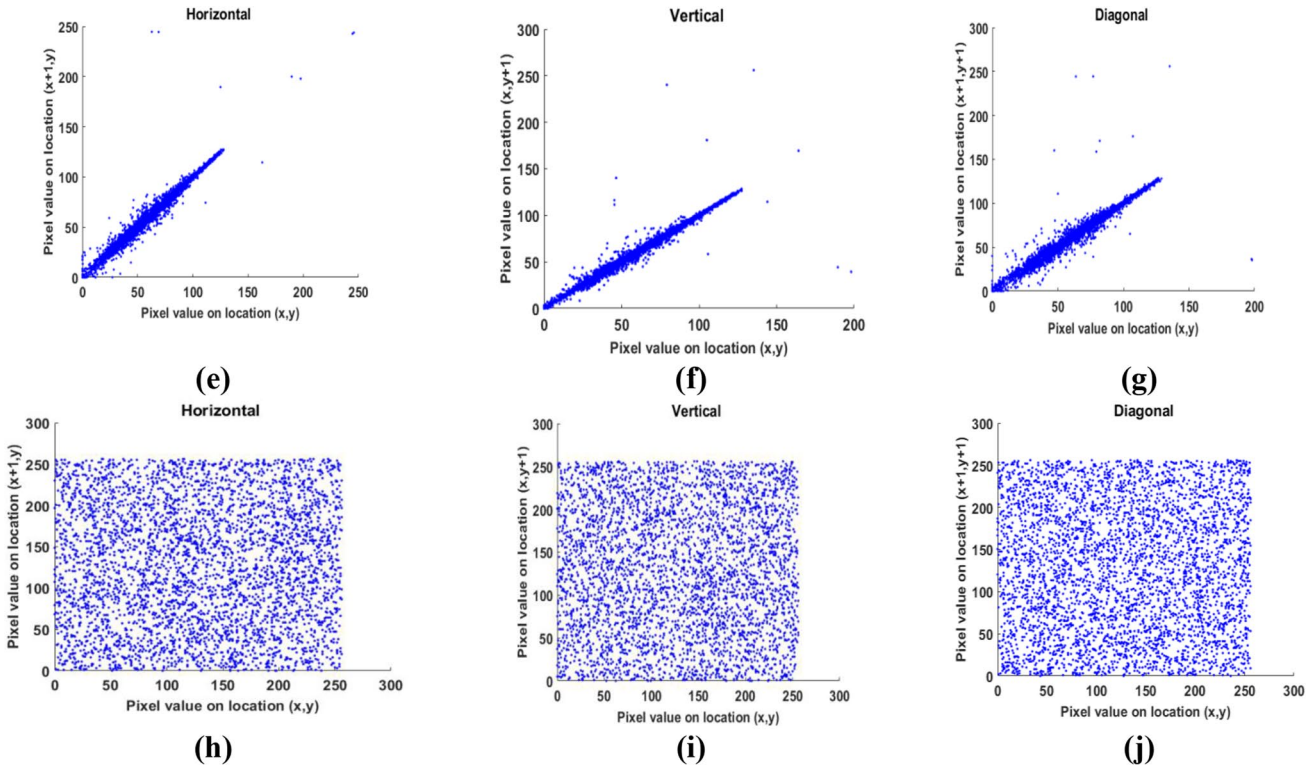

Fig. 7 (continued)

highly uncorrelated with correlation coefficient values near zero in all directions for an entirely encrypted meaningless output image. Upon testing the output image for correlation, on average, the horizontal correlation is 0.0275, vertical correlation is -0.0027, and diagonal correlation is 0.0180. These values depict that the image has highly uncorrelated pixels, and hence the output is an encrypted image. The results are tabulated in Table 2. The results are also graphically represented in Fig. 8, 9, 10, 11 and 12.

Figure 8 gives the entropy of images. Figure 9 provides the PSNR, and Fig. 10 to Fig. 12 show the correlation coefficients of the cipher images.

3.2 Key size and Key Sensitivity

The algorithm is designed to be applied for storage encryption and is implemented as a symmetric cipher algorithm involving stream cipher. Each of the three keys generated has $((M-1)\times N)\times 16$ bits, and hence the key size is $2^{((M-1)\times N)\times 16}$, which is of significant size for the improbable brute force attack of the key. Moreover, as the algorithm is designed for DICOM images, the key size with bigger M and N values would be considerably large.

The initial conditions used for key generation a, b, x_1 can take a range of values and an x_1 value ranging from 2 to far greater values. Even the slightest change in the initial conditions would lead to drastically different chaotic values, leading to nearly impossible key reproduction. The test results showed that the slightest change in all the initial conditions for the key generation completely changed the key and led to incorrect decryption. The NIST test was conducted for a sample key with M = 2520 and N = 3032. The sample size given as input was 488,806,912 bits $((M-1)\times N)\times 16)$, and the results cleared the NIST tests. The randomness in the key generated was verified through these tests.

3.3 Timing Analysis

The encryption algorithm gains traction if the algorithm can get executed in minimal time apart from providing good results. The designed algorithm utilises minimal resources, and the time consumed for key generation and algorithm execution is calculated.

3.3.1 Timing analysis for key generation

As this algorithm uses a stream cipher, the key generation is dependent on the size of the image. Therefore, the time consumed for the generation of key was calculated for different sized images. On average, key generation consumes 0.6449 s. This is shown in Table 3.

Table 2 Statistical analysis

S. No	▪Image	Entropy in bits	PSNR in dB	Correlation		
				Horizontal	Vertical	Diagonal
1	46529543479051320.dcm	15.9814	8.1285	0.0339	-0.0143	-0.0197
2	83154492730325467.dcm	15.9823	8.4295	-0.0035	-0.0256	0.0142
3	107810845726105846.dcm	15.9800	8.2464	0.0536	-0.0021	-0.0008
4	108115246579239728.dcm	15.9822	5.6626	0.0493	-0.0294	0.0411
5	166414673190348825.dcm	15.9816	5.5363	0.0278	0.0108	0.0507
6	315984877759063644.dcm	15.9817	5.5709	0.0238	0.0394	0.0185
7	422370180926276168.dcm	15.9823	5.7465	0.0487	-0.0137	0.0062
8	470216202654547741.dcm	15.9817	5.5636	0.0452	-0.0207	0.0290
9	805240110561130040.dcm	15.9816	8.0652	0.0246	-0.0309	-0.0025
10	832519864833811790.dcm	15.9802	8.4891	0.0224	-0.0242	0.0183
11	834408613386169888.dcm	15.9818	8.5481	0.0562	0.0256	0.0414
12	840453817608439044.dcm	15.9818	5.5437	0.0177	-0.0059	-0.0062
13	841809251237623512.dcm	15.9817	5.5366	0.0351	-0.0142	0.0262
14	842871850070671216.dcm	15.9793	8.3377	0.0355	0.0271	0.0276
15	844376737347041552.dcm	15.9816	5.5462	0.0222	0.0122	0.0262
16	981137163280430165.dcm	15.9817	8.0950	0.0177	-0.0212	0.0372
17	1000186638823204855.dcm	15.9821	7.9005	0.0328	0.0029	0.0255
18	1007988770314491958.dcm	15.9816	7.7964	0.0556	-0.0056	0.0193
19	1008260857234599698.dcm	15.9815	8.1335	0.0066	-0.0369	0.0156
20	1009292177809823491.dcm	15.9814	8.0691	0.0109	0.0067	0.0743
21	10003057218476364498.dcm	15.9827	5.5285	-0.0198	-0.0103	0.0196
22	10005836788378209022.dcm	15.9812	5.5873	0.0145	-0.0061	0.0005
23	10011454155587105152.dcm	15.9820	8.0696	0.0025	0.0079	0.012
24	10015354220486554048.dcm	15.9815	7.9707	0.0209	0.0052	0.0215
25	10026271850367430724.dcm	15.9792	7.9926	0.0592	-0.0151	0.0191
26	10062148331243894356.dcm	15.9818	5.4941	0.0118	0.0269	0.0044
27	10066368160329455712.dcm	15.9811	8.0631	0.0364	-0.0004	-0.0082
28	10073963463917840894.dcm	15.9822	8.4172	0.0421	-0.0051	0.0266
29	10091376699707959251.dcm	15.9823	5.7597	0.0126	-0.0141	0.0138
30	10099932034723900216.dcm	15.9820	8.2046	0.0088	0.0335	0.0139
31	10104326480143296722.dcm	15.9817	5.5720	0.0492	0.0134	-0.0046
Mean		**15.9815**	**7.0840**	**0.0275**	**-0.0027**	**0.0180**
Standard deviation		**0.0008**	**1.3027**	**0.0194**	**0.0197**	**0.0190**

▪Images chosen from public database Ref. [25]

3.3.2 Timing Analysis for Algorithm

The execution time for the algorithm was tested for varied sized images, and the results are tabulated. On average, the algorithm consumes 0.03866 s for confusion, 0.78878 s for the first diffusion, 0.0032 s for second diffusion and 0.01528 s for embedding. Thus, the total average time consumed for the execution of the algorithm is 0.84592 s. This is shown in Table 4.

3.4 Attack Analysis

As the algorithm is designed for storage encryption, they are not prone to channel attacks or attacks during transmission, as there is no data transmission. Anyway, tests were conducted for attacks. Cryptanalysis was performed on the cipher image using CrypTool to check the algorithm's attack performance. To attack the cipher through brute force, key should be attacked. As the key size is $2^{((M-1)\times N)\times 16}$, and M, N are large for DICOM images, the key size is bigger and performing a 256-bit brute force attack takes 5.3×10^{63} years for decryption of a 1024×1024 DICOM image. This is considerably an ample time. Hence, this algorithm effectively resists brute force attacks.

Chosen plain text attack analysis was done on two images, and the results proved that the algorithm could effectively resist the chosen plain text attacks. To perform the chosen plain text attack analysis, two input images are XORed, and

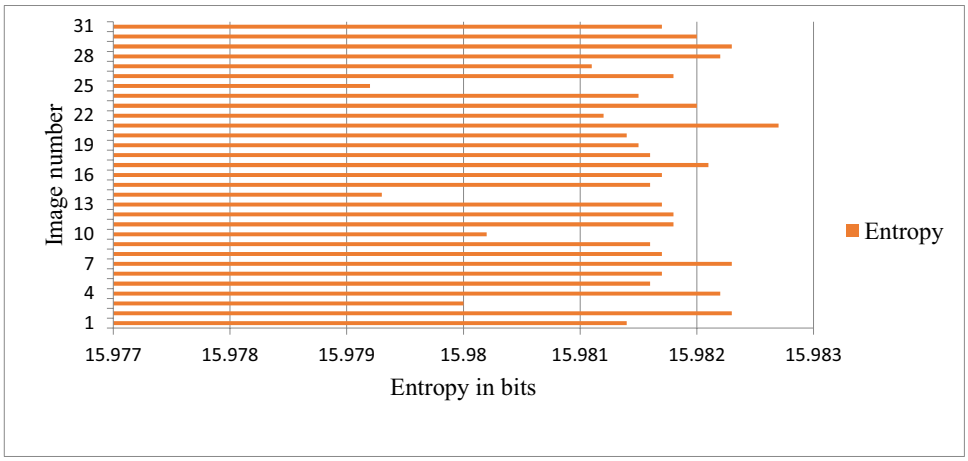

Fig. 8 Entropy analysis

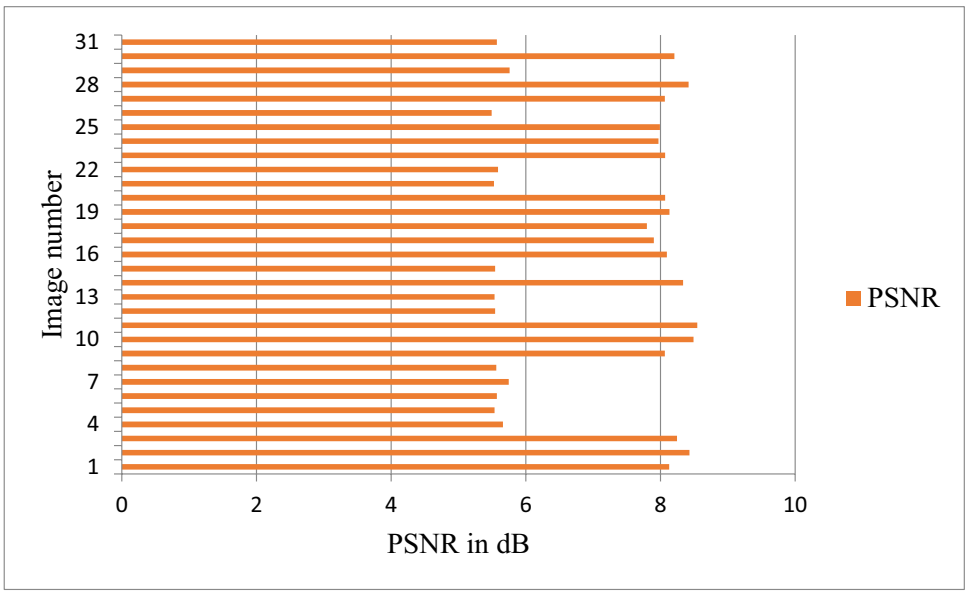

Fig. 9 Visual perception analysis using PSNR

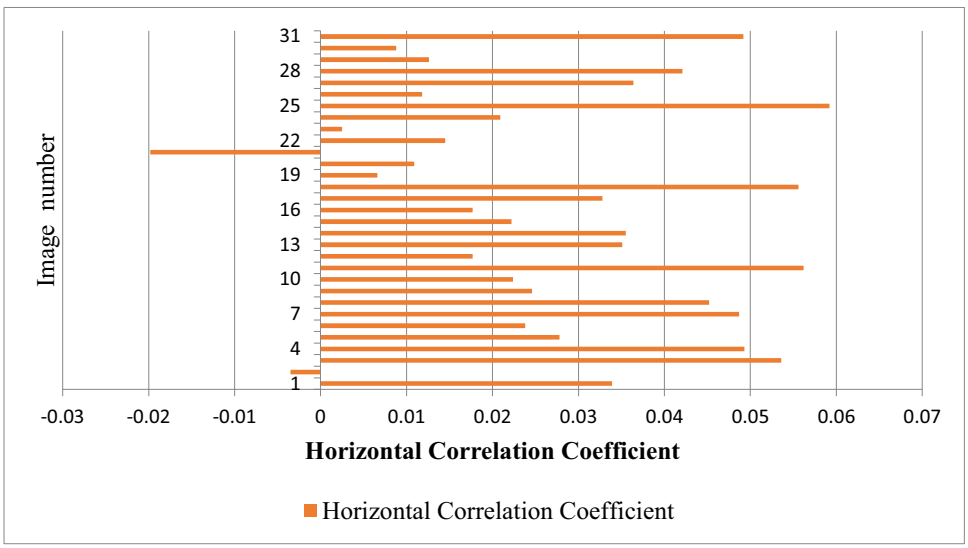

Fig. 10 Horizontal correlation coefficient analysis

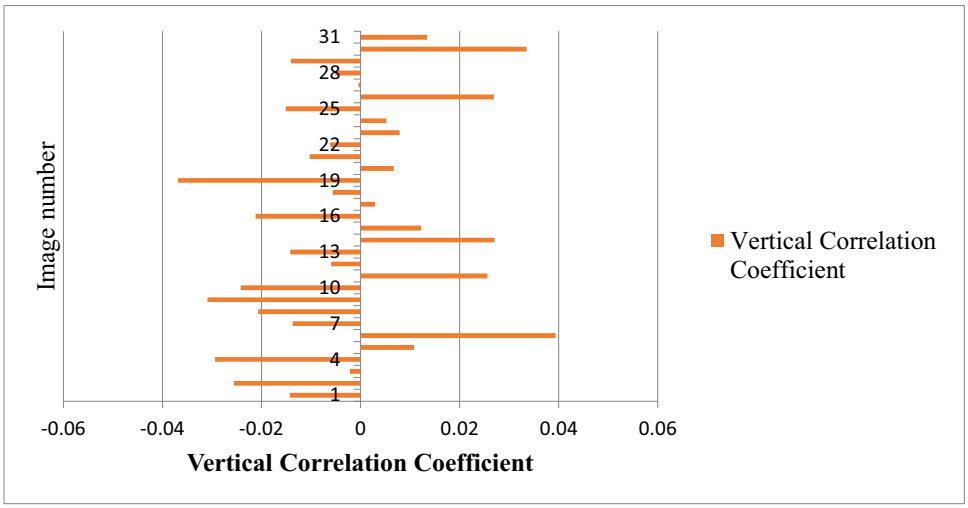

Fig. 11 Vertical correlation coefficient analysis

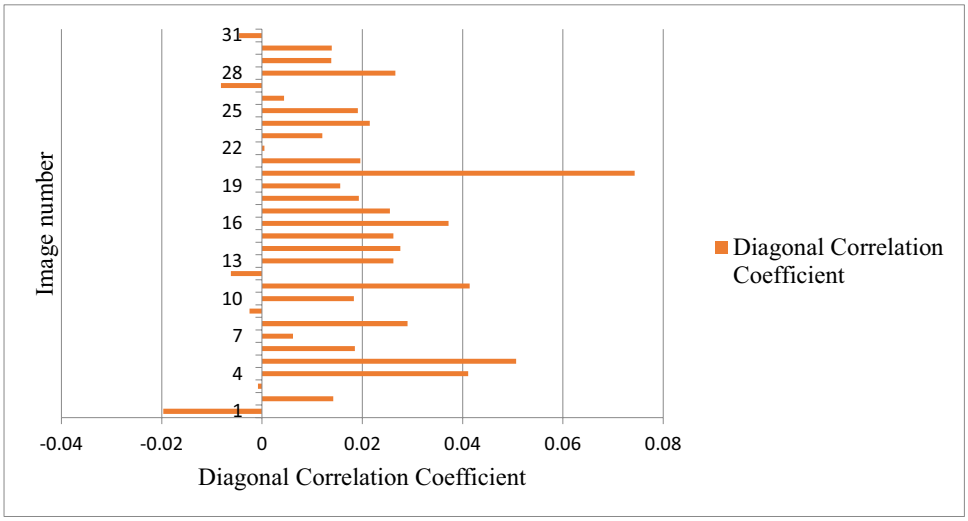

Fig. 12 Diagonal correlation coefficient analysis

Table 3 Timing for key generation

Image size (bit depth is 16)	Time consumed in seconds
2350×2866	0.9942
2320×2868	0.9839
2520×3032	1.1288
2008×2556	0.7896
2008×2556	0.1871
2539×3050	1.1535
2539×3050	0.2959
2350×2866	0.2524
2520×3032	0.2805
2160×2545	0.8264
2160×2545	0.2020
Mean	**0.6449**
Standard deviation	**0.3997**

their corresponding cipher images are XORed. If the XOR of the input images and the XOR of cipher images are not equal, then it is said that the algorithm could effectively mitigate the chosen plain text attack. The two images that were taken for analysis were 981,137,163,280,430,165.dcm and 83,154,492,730,325,467.dcm [25]. Chosen plain text attack analysis is depicted in Fig. 13.

4 Discussion

The test results show that the proposed algorithm is a good candidate for encryption and embedding in DICOM images. This algorithm performs better when compared to some of the contemporary algorithms for the encryption of DICOM images. The proposed algorithm is tested for images of various sizes. Most of the research papers deal with DICOM images of bit depth 8, and even the ones dealing with images

Table 4 Timing for algorithm execution

Image size (bit depth is 16)	Time consumed in seconds				
	Confusion	Diffusion 1	Diffusion 2	Embedding	Total
2160×2545	0.0342	0.6798	0.0030	0.0154	0.7324
2272×2715	0.0362	0.7505	0.0029	0.0158	0.8054
2350×2866	0.0392	0.8051	0.0030	0.0152	0.8625
2520×3032	0.0427	0.9089	0.0038	0.0151	0.9705
2320×2868	0.0410	0.7996	0.0033	0.0149	0.8588
Mean	**0.0386**	**0.7887**	**0.0032**	**0.0152**	**0.8459**
Standard deviation	**0.0034**	**0.0838**	**0.0003**	**0.0003**	**0.0873**

of bit depth 16 use smaller sized images. Compared to the contemporary ones, the compared research reports had used a 512×512 image, and the computation time is far less for the proposed algorithm. The time consumed for a 512×512 DICOM image of bit depth 16 is 0.186 s in [21], whereas, in the proposed algorithm, a 2320×2868 sized DICOM image consumes only 0.845 s. Whereas the PSNR is 10.267 dB in [21], the proposed algorithm has a PSNR of 7.084 dB. In [21], the entropy is 15.6574 bits, whereas the proposed algorithm has better entropy at 15.9815 bits. The compared algorithms show that the proposed algorithm could be validated as an effective candidate for encryption and embedding. The comparison is given in Table 5.

Most of the algorithms [7, 14, 15, 20, 21] taken for comparison had performed either chosen plain text attack or have theoretically mentioned that the algorithm could resist brute force attack. The comparison table is given in Table 6. The proposed algorithm could effectively mitigate the brute force attack, as the keyspace is greater ($2^{((M-1)\times N)\times 16}$) compared to the contemporary algorithms. The brute force attack test was carried out using a crypt tool and briefed in Sect. 3.4.

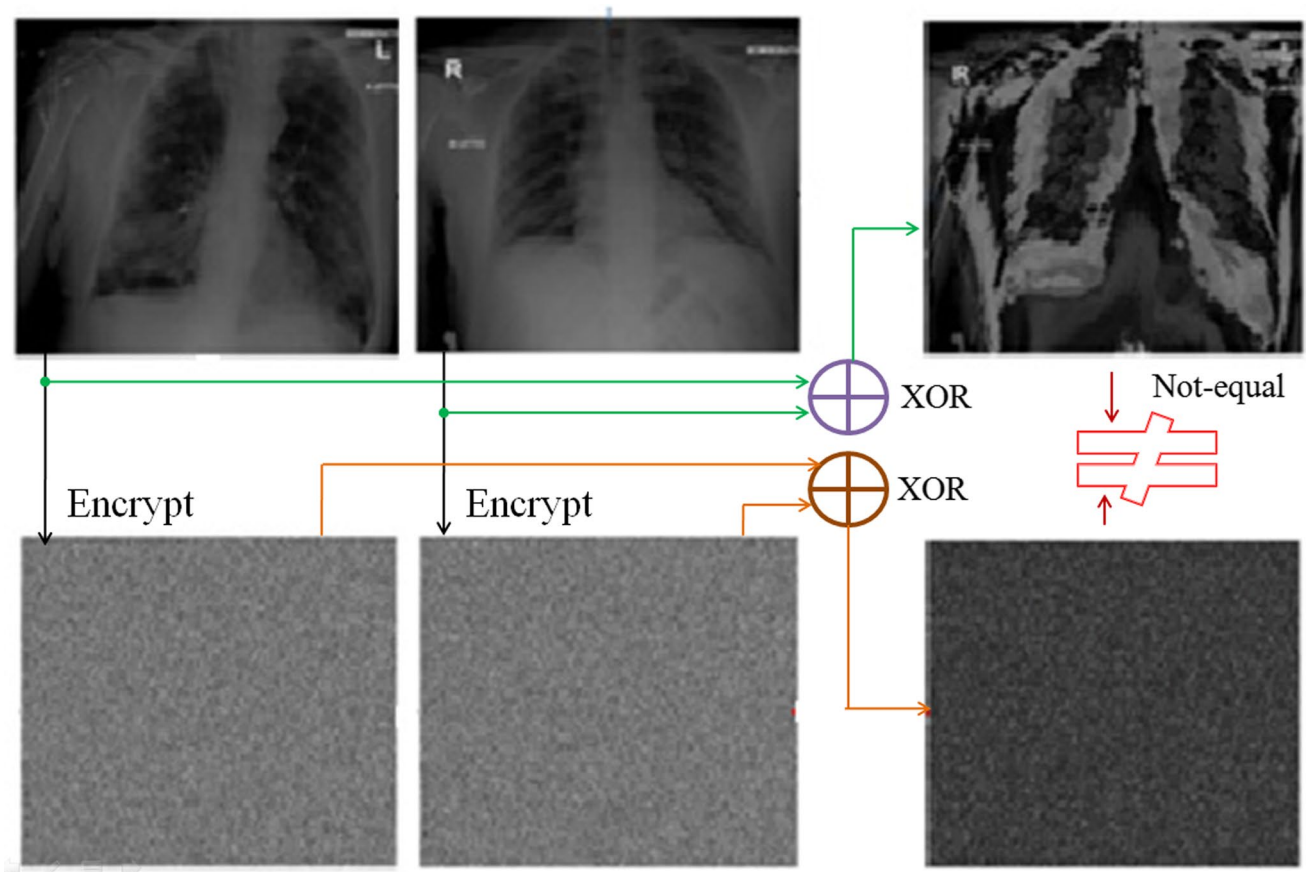

Fig. 13 Chosen plain text attack analysis

Table 5 Algorithm comparison

Reference	Image size	PSNR in dB	Horizontal	Vertical	Diagonal	Time in seconds	Entropy in bits
[20]	512×512 (8 bit)	5.7200	0.0198	0.0213	0.0187	0.2360	-
[22]	512×512 (8 bit)	8.2950	0.0019	−0.0014	−0.0027	4.0818	-
[7]	512×512 (8 bit)	-	−0.0016	0.0043	−0.0061	-	7.9972
[21]	**512×512 (16 bit)**	**10.2670**	**0.0072**	**0.0262**	**0.0024**	**0.1860**	**15.6574**
[9]	256×256(8 bit)	-	0.0061	0.0022	0.0006	-	7.9976
[10]	256×256(8 bit)	-	0.0043	0.0033	0.0027	-	7.9900
[11]	256×256(8 bit)	-	0.0019	0.0018	0.0034	-	7.9890
[28]	256×256(8 bit)	-	**0.0007**	0.0256	- 0.0202	-	7.9973
[15]	256×256(8 bit)	-	0.0011	**-0.0012**	**0.0001**	-	7.9924
[14]	1024×1024 (8 bit)	-	0.0039	0.0059	−0.0050	-	7.9994
Proposed	**2320×2868 (16 bit)**	**7.0840**	**0.0275**	**-0.0027**	**0.0180**	**0.8459**	**15.9815**

The algorithm was implemented in software (MATLAB R2018b) and was tested for DICOM images. However, further tests are needed for greyscale images and RGB images to make this algorithm universal for storage encryption, while implementing the algorithm in hardware platforms such as FPGA, using Bülban map. As key generation using Bülban map involves real numbers, FPGA implementation requires converting real numbers using IEEE 754 floating-point representation (single precision), which consumes many resources. Therefore, some optimisation resources need to be used to reduce the resources while implementing in FPGA.

5 Conclusion

With scores of encryption algorithms available for images and a few for medical images, specifically DICOM, it was essential to design a less complicated, less resource consuming one that could encrypt the DICOM images efficiently and in less time. This proposed algorithm uses fewer resources, which further reduces the time consumption in the algorithm's execution. The analysis of the cipher image using metrics yielded good results, and the algorithm passed all metric tests. The algorithm was tested for many DICOM images and could be effectively used for database maintenance. As more and more DICOM images get stored in the database, maintaining a secure database while identifying the patients' files needs an effective algorithm. The proposed algorithm has direct implications in providing security of the image and identifying patients on the fly. Applying the designed algorithm to the medical images serves dual purposes. Firstly, the medical images and the patient health information is hidden from the prying hackers. Secondly, as patient identification number is secretly embedded within the corresponding medical image, the ambiguity and the mismatch of patient information and the medical image could be avoided. This is achieved without compromising the security of the image. After applying the algorithm, any normal viewer of the image database will only have a huge set of meaningless image objects with no patient identification number. The authorised user could decrypt the medical images and match them with the patient identification number to get the original medical image for inspection and prescription. A further feasibility study is being done on the possible implementation of the design as a hardware unit using FPGA. Implementation of the algorithm in FPGA could help in designing a standalone device, which could function independently. Such independent functionality of the device could further lead to a reduction in the execution time.

Table 6 Attack analysis comparison

Reference	Brute force analysis (keyspace)	Chosen plain text attack analysis
12	2^{399}	✖
15	-	✖
20	2^{256}	✖
13	10^{192}	✖
22	2^{680}	✓
23	10^{203}	✖
24	10^{238}	✓
25	2^{256}	✓
26	10^{58}	✖
17	2^{100}	✖
Proposed	$2^{((M-1)\times N)\times 16}$	✓

Acknowledgements The authors wish to express their sincere thanks to the Department of Science and Technology (DST), India, for their financial support and FIST funding (**SR/FST/ET-I/2018/221 (C)**).

Also, the authors wish to thank the Intrusion Detection Lab at the School of Electrical & Electronics Engineering, SASTRA Deemed University, to provide infrastructural support to carry out this research work.

References

1. Banik A, Shamsi Z, Laiphrakpam DS (Dec. 2019) An encryption scheme for securing multiple medical images. J Inf Secur Appl 49:102398. https://doi.org/10.1016/j.jisa.2019.102398
2. Adedokun EA, Akan JB, Umoh IJ, Nwosu RI, Ibrahim Y (2020) A Secure Chaotic Framework for Medical Image Encryption using a 3D Logistic Map. Appl Model Simul 4(0): 141–148. Accessed: Nov. 27, 2020. [Online]. Available: http://arqiipubl.com/ams
3. Boussif M, Aloui N, Cherif A (May 2020) Securing DICOM images by a new encryption algorithm using Arnold transform and Vigenère cipher. IET Image Process 14(6):1209–1216. https://doi.org/10.1049/iet-ipr.2019.0042
4. Han B, Jia Y, Huang G, Cai L (2020) A Medical Image Encryption Algorithm Based on Hermite Chaotic Neural Network," in Proceedings of 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2020. 2644–2648. https://doi.org/10.1109/ITNEC48623.2020.9085079.
5. Banday SA, Mir AH, Malik S (2020) "Multilevel medical image encryption for secure communication," in Advances in Computational Techniques for Biomedical Image Analysis. Elsevier, pp. 233–252
6. Dai Y, Wang H, Wang Y (2016) Chaotic Medical Image Encryption Algorithm Based on Bit-Plane Decomposition. Int J Pattern Recognit Artif Intell. 30 (4). https://doi.org/10.1142/S0218001416570019.
7. Dagadu JC, Li JP, Aboagye EO (2019) Medical image encryption based on hybrid chaotic DNA diffusion. Wirel Pers Commun 108:591–612. https://doi.org/10.1007/s11277-019-06420-z
8. Lakshmi C, Thenmozhi K, Rayappan JBB et al (2020) Neural-assisted image-dependent encryption scheme for medical image cloud storage. Neural Comput & Applic. https://doi.org/10.1007/s00521-020-05447-9
9. Aashiq Banu S, Amirtharajan R (2020) Tri-level scrambling and enhanced diffusion for DICOM image cipher- DNA and chaotic fused approach. Multimed Tools Appl. https://doi.org/10.1007/s11042-020-09501-5
10. Aashiq BS, Amirtharajan R (2020) Bio-inspired cryptosystem on reciprocal domain-DNA strands mutate to secure health data. Front Inf Technol Electron Eng. https://doi.org/10.1631/FITEE.2000071
11. Aashiq BS, Amirtharajan R (2020) A robust medical image encryption in dual domain: chaos-DNA-IWT combined approach. Med Biol Eng Comput 58:1445–1458. https://doi.org/10.1007/s11517-020-02178-w
12. Ismail Abdelfattah R, Mohamed H, Nasr ME (2020) Secure Image Encryption Scheme Based on DNA and New Multi Chaotic Map. J Phys Conf Ser 1447(1):12053. https://doi.org/10.1088/1742-6596/1447/1/012053
13. Chai X, Gan Z, Zhang M (Jul. 2017) A fast chaos-based image encryption scheme with a novel plain image-related swapping block permutation and block diffusion. Multimed Tools Appl 76(14):15561–15585. https://doi.org/10.1007/s11042-016-3858-4
14. Talhaoui MZ, Wang X, Midoun MA (2020) Fast image encryption algorithm with high security level using the Bülban chaotic map. J Real-Time Image Process. pp. 1–14. https://doi.org/10.1007/s11554-020-00948-1
15. Guan M, Yang X, Hu W (2019) Chaotic image encryption algorithm using frequency-domain DNA encoding. IET Image Proc 13:1535–1539. https://doi.org/10.1049/iet-ipr.2019.0051
16. Madhusudhan KN, Sakthivel P (May 2020) A secure medical image transmission algorithm based on binary bits and Arnold map. J Ambient Intell Humaniz Comput 1:3. https://doi.org/10.1007/s12652-020-02028-5
17. Priya S, Santhi B (2019) A Novel Visual Medical Image Encryption for Secure Transmission of Authenticated Watermarked Medical Images. Mob Networks Appl. 1–8. https://doi.org/10.1007/s11036-019-01213-x.
18. Cao W, Zhou Y, Chen CLP, Xia L (Mar. 2017) Medical image encryption using edge maps. Signal Process 132:96–109. https://doi.org/10.1016/j.sigpro.2016.10.003
19. Zhang LB, Zhu ZL, Yang BQ, Liu WY, Zhu HF, Zou MY (2015) Medical image encryption and compression scheme using compressive sensing and pixel swapping based permutation approach. Math Probl Eng. 2015. https://doi.org/10.1155/2015/940638.
20. Akkasaligar PT, Biradar S (2020) Selective medical image encryption using DNA cryptography. Inf Secur J. 29(2). Taylor and Francis Inc. 91–101. https://doi.org/10.1080/19393555.2020.1718248
21. Zhou J, Li J, Di X (2020) A Novel Lossless Medical Image Encryption Scheme Based on Game Theory with Optimised ROI Parameters and Hidden ROI Position. IEEE Access 8:122210–122228. https://doi.org/10.1109/ACCESS.2020.3007550
22. Chandrasekaran J, Thiruvengadam SJ (2017) A hybrid chaotic and number theoretic approach for securing DICOM images. Secur Commun Networks. 2017. https://doi.org/10.1155/2017/6729896
23. Dagadu JC, Li JP, Shah F, Mustafa N, Kumar K (2017) "DWT based encryption technique for medical images," in 2016 13th International Computer Conference on Wavelet Active Media Technology and Information Processing, ICCWAMTIP 2017. pp. 252–255. https://doi.org/10.1109/ICCWAMTIP.2016.8079849
24. Alpar O (2014) Analysis of a new simple one dimensional chaotic map. Nonlinear Dyn 78(2):771–778. https://doi.org/10.1007/s11071-014-1475-1
25. "OSF | dicom_archive_v2.tar.gz.aa." https://osf.io/rytmh/?view_only=c67e3d4393404981bc6829b6314364fb (accessed Nov. 27, 2020)
26. Li J, Zhang Z, Li S, Benton R, Huang Y, Kasukurthi MV, Li D, Lin J, Borchert GM, Tan S, Li G, Ma B, Yang M, Huang J (2020 Dec 15) A partial encryption algorithm for medical images based on quick response code and reversible data hiding technology. BMC Med Inform Decis Mak 20(Suppl 14):297. https://doi.org/10.1186/s12911-020-01328-2.PMID:33323108;PMCID:PMC7739464
27. Chirakkarottu S, Mathew S (2020) A novel encryption method for medical images using 2D Zaslavski map and DNA cryptography. SN Appl Sci 2:1. https://doi.org/10.1007/s42452-019-1685-8
28. Chen Y, Tang C, Ye R (2020) Cryptanalysis and improvement of medical image encryption using high-speed scrambling and pixel adaptive diffusion. Signal Process 167:107286. https://doi.org/10.1016/j.sigpro.2019.107286

Manikandan Veerappan received B.E. degree in Electrical and Electronics from Anna University, Tamil Nadu, India, in 2007. He received the M.Tech degree in VLSI Design from the SASTRA Deemed-to-be-University, Tamil Nadu, India, in 2009, where he is currently pursuing a PhD degree. He is a research scholar with the Information Security group, SASTRA Deemed Universityfrom 2018. His research interests include RTL design with FPGA, data security and reversible steganography.

Dr. Rengarajan Amirtharajan received his B.E. degree from P.S.G. Tech. Bharathiyar University, Coimbatore, India, in 1997. He received M.Tech. and PhD degrees from SASTRA Deemed University, Thanjavur, India, in 2007 and 2012, respectively. He is currently working as a Professor at SEEE, SASTRA Deemed University. He patented a novel embedding scheme USPTO in March 2015. He has also published more than 250 research articles in National & International journals.