Hindawi
Security and Communication Networks
Volume 2020, Article ID 8848356, 22 pages
https://doi.org/10.1155/2020/8848356

Research Article

Chaotic Cryptosystem for Selective Encryption of Faces in Photographs

Jorge Aguilar Santiago,[1] **Octavio Flores Siordia**,[1] **José T. Guillen Bonilla**,[2] **Juan C. Estrada Gutiérrez**,[1] **María G. González Novoa**,[1] **and Maricela Jiménez Rodríguez**[1]

[1]*División de Desarrollo Biotecnológico, Centro Universitario de la Ciénega, Universidad de Guadalajara, Av. Universidad No. 1115, Col. Lindavista C.P., 47810 Ocotlán, Jalisco, Mexico*
[2]*Departamento de Electrónica, Centro Universitario de Ciencias Exactas e Ingeniería, Universidad de Guadalajara, Blvd. Marcelino García Barragán No. 1421, esq Calzada Olímpica, C.P., 44430 Guadalajara, Jalisco, Mexico*

Correspondence should be addressed to Maricela Jiménez Rodríguez; maricela.jimenez@cuci.udg.mx

Received 24 March 2020; Revised 23 October 2020; Accepted 7 November 2020; Published 1 December 2020

Academic Editor: Salvatore D'Antonio

Copyright © 2020 Jorge Aguilar Santiago et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In this article, a safe communication system is proposed that implements one or more portable devices denominated SBC (single-board computers), with which photographs are taken and that later utilizes the OpenCV Library for the detection and identification of the faces that appear in them. Subsequently, it consults the information in a stored database, whether locally in SBC or in a remote server, to verify that the faces should be coded, and it encrypts these, implementing a new cryptosystem that executes mathematical models to generate chaotic orbits, one of which is used for application on two occasions the technique of diffusion with the purpose of carrying out a small change in one of the pixels of the image, generating very different cryptograms. In addition, in order to make a safer system, it implements other chaotic orbits during the technique of confusion. With the purpose of verifying the robustness of the encryption algorithm, a statistical analysis is performed employing histograms, horizontal, vertical, and diagonal correlation diagrams, entropy, number of pixel change rate (NPCR), unified average change intensity (UACI), sensitivity of the key, encryption quality analysis, and the avalanche effect. The cryptosystem is very robust in that it generates highly disordered cryptograms, supports differential attacks, and in addition is highly sensitive to changes in the pixels as well as in the encrypted keys.

1. Introduction

Facial recognition is very important today due to its application in security, surveillance systems, forensic analyses, etc. Therefore, it is important to conduct investigation in this area, as well as in research in which a system is developed to detect the drowsiness of a conductor, considering its positions, the form around the eye, and whether the head is inclined [1]. These authors also carried out a balance between image similarities and differences in order to detect human faces even when they had noise or facial expressions and different poses [2]. However, facial recognition can also be utilized to generate damage; thus, it is necessary permit the protection of identity by means of encryption systems, which can be employed to code photographs completely, although it is most recommendable to perform selective encryption, whether on the borders or using facial detection because this considerably reduces the system's time and resources [3, 4]. Shakir et al. detected faces in the images, considering a database that contains information of different parts of the skin [5]. Rachmawati et al. elaborated a cryptosystem for Android that employs cryptography [6]. Vela Medina et al. carried out an embedded system that utilizes an intelligent camera, in addition to algorithms to detect and recognize faces in a video surveillance system that performs

the capture and storage of the images and subsequently, it sends a message to inform the user [7]. Rodrigues et al. coded the human face in a video sequence using AES and VLC (variable length coding); these authors detected the skin, but the method does not render the faces totally unrecognizable [3]. Khashan et al. implemented border and facial detection methods utilizing OpenCV and employing the Blowfish algorithm for encryption [4]. Also, in the work developed by Hong and Jung, the authors implemented a partial encryption method that only codes the facial region in images and videos; they used a Gaussian skin-color model for detection and, in addition, DES (data encryption standard) and AES (advanced encryption standard) for encryption [8]. Gerhardt et al. selectively encrypted faces in videos using Sophisticated High-speed Object Recognition Engine (SHORE) for the detection and use of AES-256 [9]. In another research, these authors encrypted facial regions by applying multilayer encoding on images in the frequency domain, but they do not divide the human region and it may also encrypt clothing [10]. Also, they encrypted important objects in the image that are detected with the machine learning technique through the OpenCV Library and employed an algorithm called RNS to encrypt these, increasing the encryption speed but losing the context of the original image [11]. They extract all of the faces that appear in the videos using high-speed support functions with the OpenCV Library, implementing NewHope, public, private, and SHA-3 keys on order to encrypt; faces are encoded in parallel using GPU [12]. Sardar et al. use the tree-structured part model (TSPM) to extract the region of the face; for training, they employ sparse representation coding (SRC), coordinate descent (CD) and block coordinate descent (BCD), and the SVM classifier. They also implement a variant of BioHashing to generate face code from the characteristic vector. Then, to provide greater security, they adapt the RSA algorithm [13]. Asgari-Chenaghlu et al. use YoloV3 to automatically detect objects or the human body of the ROI delimiter and utilize a cryptosystem based on the chaotic sine map and logistic map systems. They only detect objects of interest and encode them [14]. Additionally, investigations have also been developed in which the properties of the chaotic systems are taken advantage of, such as the high sensitivity of the initial conditions as well as of the parameters. This is due to that the latter are excellent candidates for the development of cryptographic systems, such as those in which they detect the face in order to later extract regions of the body and encrypt them [15]. Wen et al. selectively encrypt objectives in infrared images employing logistic chaotic maps and sine [16]. These authors also developed a selective facial encryption system in gray tones utilizing the tent chaotic map [17]. Prabhavathi et al. selectively encrypt regions of interest in medical images utilizing multiple chaotic systems [18]. In addition to the latter, discrete chaotic systems have also been taken advantage of to encrypt images [19–22]. Ayoup et al. divide the image into blocks and, according to the entropy involved, determine which of these to code using pseudorandomized numbers (PRNs), Arnold's cat map, and AES [23]. The proposed communication system performs selective facial detection, identification, and encryption, encoding only the object of interest and leaving the environment visible, unlike general encryption, which hides everything. Thus, encryption time and resources are reduced compared to the encryption of a full image and increases the security of people who do not want their faces to be visible when others post their photographs. It is also useful for protecting identity in the news when someone is presented who is under criminal proceedings. It allows protecting the identity of minors when adults wish to publish photographs, and it can also be used in closed-circuit video surveillance networks or in photographs published through social media and communication. Our system, in contrast to those described in the current research, can selectively encode only faces indicated in a database or manually when the face cannot be identified. In addition, there is an encryption algorithm that implements novel confusion and diffusion techniques; these allow the system to be robust on confronting different attacks, and especially that of the avalanche effect, which not all cryptosystems approve.

The remainder of this article is organized as follows: in Section 2, the mathematical models are explained that are implemented during encryption. Section 3 details the functioning and the stages carried out by the communications system, while the results and the tests conducted on the cryptogram are presented in Section 4. Finally, Section 5 presents the conclusions of this investigation.

2. Mathematical Models

2.1. Rössler Oscillator. This comprises a simultaneous system of 3 differential equations:

$$\begin{cases} \dfrac{dx}{dt} = -(y + z), \\ \dfrac{dy}{dt} = x + ay, \\ \dfrac{dz}{dt} = b + z(x - c), \end{cases} \tag{1}$$

where x, y, and z are the dependent variables of time, while a, b, and c are the parameters. In the system, the third equation makes this possess nonlinear chaotic behavior [24].

3. Communication System

This comprises a server and can have one or more portable devices through which recognition can be performed with the Open Computer Vision (OpenCV) Library or, in a particular case, code selected faces in photographs as well as in videos; this is due to that the latter are composed of a series of photograms, in which it is possible to detect the faces and to code them later.

Figure 1 shows the general procedure that the communication system carries out, which includes the following stages:

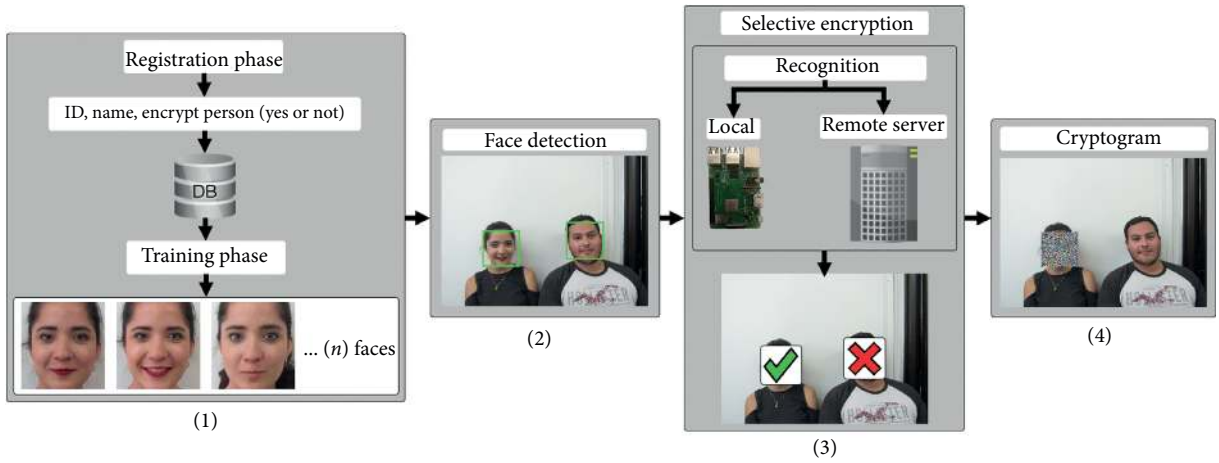


Figure 1: Stages of selective facial encryption.

(1) Registration and training
(2) Face detection
(3) Selective encryption
(4) Cryptogram generation

3.1. Registration and Training. First, it is necessary to carry out registration and training with the information of all persons who are to be added to the database, for which the process is performed that is depicted in Figure 2.

(1) Register the data: the information is stored in a database performed in SQLite3 on the persons involved, such as the following: ID, name, and, in addition, it is indicated whether it is necessary to code the face once it is detected in the photographs. This information can be stored in three ways:
   (i) In a base of local data (limited), stored in the portable device.
   (ii) Base of complete data, located in a remote server.
   (iii) Conduct the capture of the data of both databases, that is, local and remote.

(2) Training: this is carried out with the OpenCV Library. In this process, different photographs are taken of the person, whether from the SBC or with another camera, and the process continues with the generation of the vector characteristic of each face, which is stored in the registry of its ID in the database.

3.2. Face Detection. First, the photograph is captured or stored in the single-board computer (SBC). Subsequently, the OpenCV Library is utilized to locate the faces, which are identified and marked with a square, as can be observed in Figure 3.

3.3. Selective Encryption. The system permits manual selection through an interface, in which the face is selected and encrypted, as can be observed in Figure 4.

Also, encryption can be automatic, by means of a communications schema, which performs recognition, taking information locally, remotely, or considering both databases, as demonstrated in Figure 5.

(i) Local: the faces extracted from the photograph are compared with the information that was stored in the database of the local device (SBC) during the training phase, in order to identify them and to determine whether it is necessary to code them.

(ii) Remote: it is verified whether there exists information on the face in the local database of the device. In the case of there not being information, optimized encryption is conducted with the purpose of this being more agile, but safe for transmission via Internet; that is, the diffusion technique of the algorithm is employed only for encryption. Later, this is sent to the remote server to carry out the search in its database; in the case of finding it, a signal is returned to the source, indicating that faces should be encrypted.

3.4. Generation of the Cryptogram. Once a certain face is detected that should be coded according to the information stored in the database, the cryptogram should be generated utilizing the following procedure.

3.4.1. Algorithm to Encrypt. The encryption algorithm can work with any chaotic mathematical model one wishes to employ, as long as the required number of orbits is generated. The Rössler oscillator is implemented because it generates three complex chaotic sequences, is highly sensitive to initial conditions and system parameters, and uses six keys each time it is resolved, generating a very wide key space.

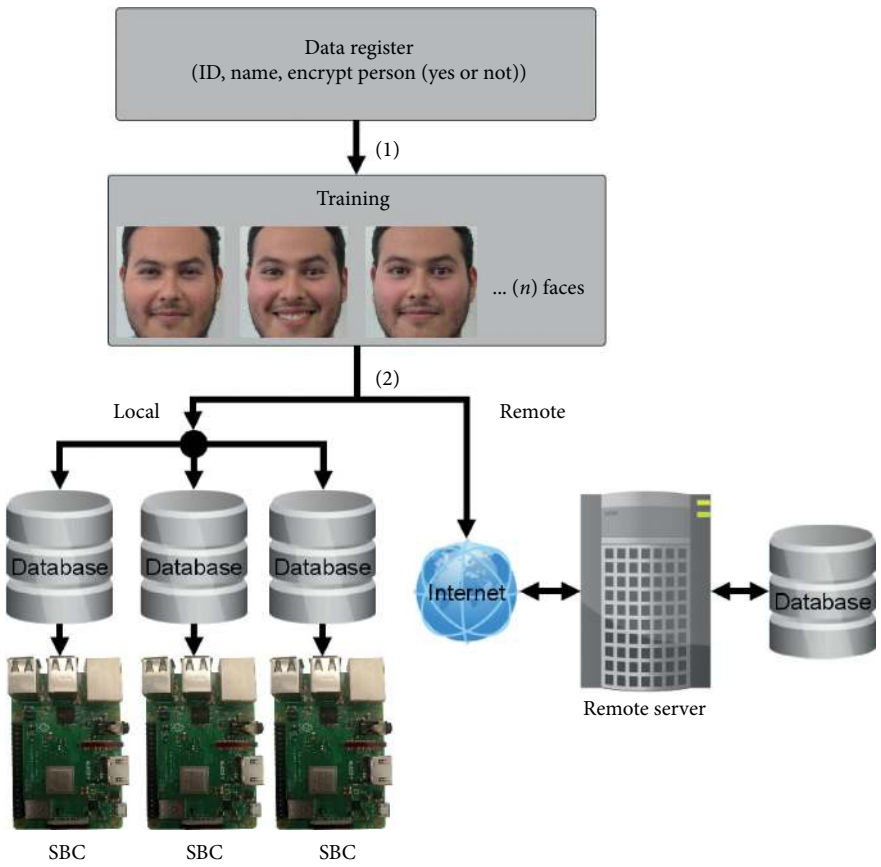


Figure 2: Data registration and training phase.

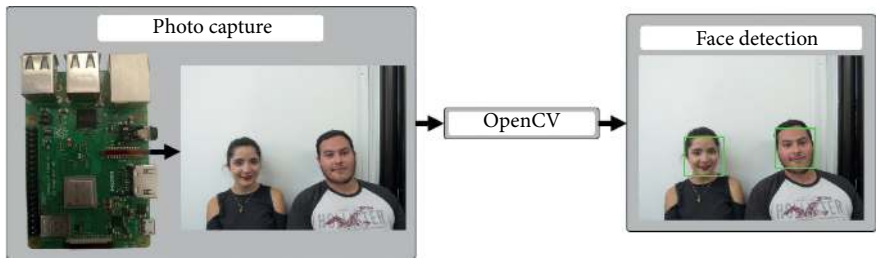


Figure 3: Face detection phase.

Keys

(i) *Diffusion*

$x1_0, y1_0, z1_0, a1, b1, c1$: values *corresponding* to the initial conditions and parameters used to apply Diffusion I, when the chaotic system of the Rössler oscillator is resolved (equation (1)). $x3_0, y3_0, z3_0, a3, b3, c3$: keys utilized to apply Diffusion II.

startdif 1, stardif 2: the keys necessary in the two diffusion processes, which are utilized as initial position and that can be taken between 1 and L; where L = width_image $*$ height_*image* $*$ 3.

(ii) *Confusion*

$x2_0, y2_0, z2_0, a2, b2, c2$: these initial *conditions* and parameters are also utilized to resolve the Rössler oscillator (equation (1)).

keyconf: initial position between 0 and 5.

keyconf 2: value between 0 and 16777215, which is employed to select a position.

Apply Diffusion I

Each face is broken down into the pixels comprising it, and later into their 3 subpixels (red, green, blue (RGB)), which have a value between 0 and 255.

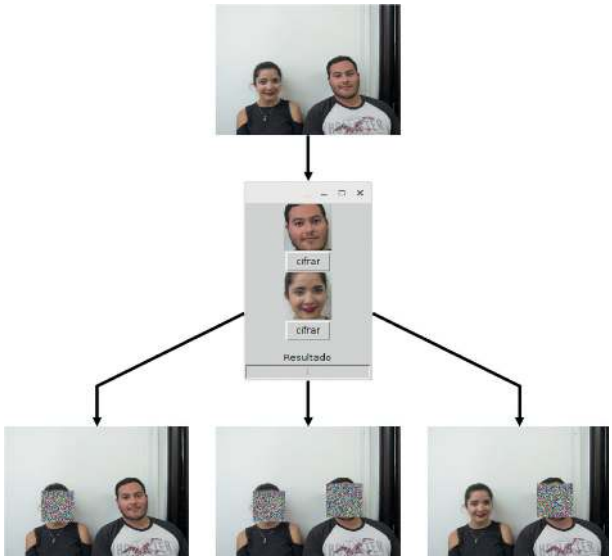


Figure 4: Manual facial encryption.

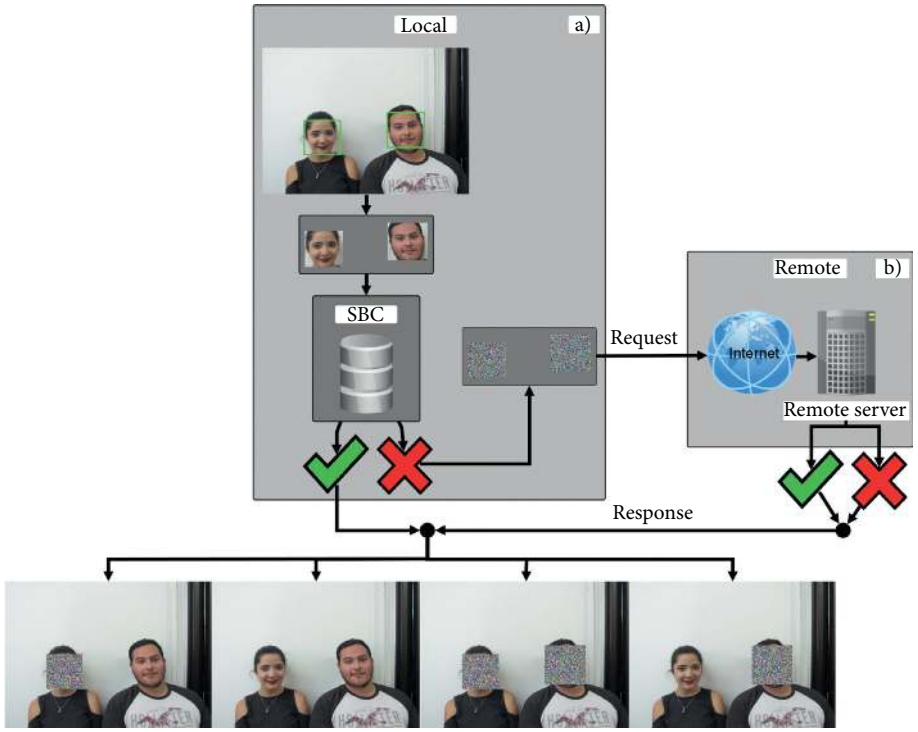


Figure 5: Techniques for facial recognition.

Step 1: store in a vector of length L, called image, the subpixels of the face:

$$\text{image} = [R_1, G_1, B_1, \ldots, R_n, G_n, B_n]. \quad (2)$$

Step 2: create a vector called dif 1 of *size L*.

Step 3: resolve R times the system of equation (1), implementing the encryption keys of $x1_0, y1_0, z1_0, a1, b1, c1$; where $R = \text{width\_image} * \text{height\_image}$. The values generated of $x1_R, y1_R, z1_R$ are stored in a vector called *chaos* of length $L = 3 * R$:

$$\text{chaos} = [x1_1, y1_1, z1_1, \ldots, x1_R, y1_R, z1_R]. \quad (3)$$

Step 4: assign, to the variable Loc, the value of the key startdif 1.

Step 5: take a value of image and assign it to location Loc of vector dif 1, as can be observed in Figure 6:

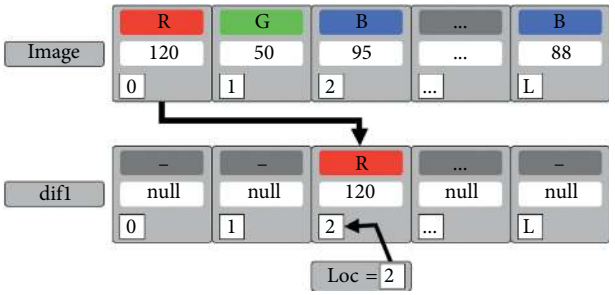


Figure 6: Assign a value of image to another position of vector dif 1.

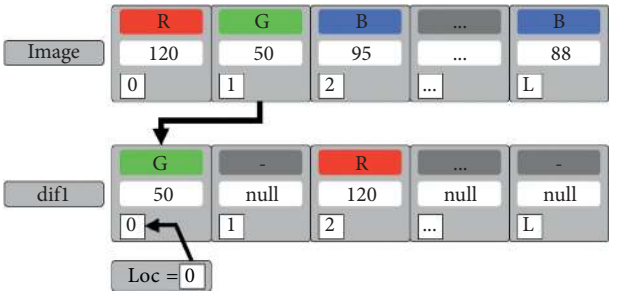


Figure 7: Assign a value of image to an empty position of vector dif 1.

$$\text{dif1}[\text{Loc}] = \text{image}[\text{i}]. \tag{4}$$

Step 6: take the absolute value of an element of the vector chaos $[j]$, later adding to it dif 1 [Loc]; finally, the result is rounded off and is assigned to the variable inc:

$$\text{inc} = \text{round}(\text{abs}(\text{chaos}[j]) + \text{dif1}[\text{Loc}]). \tag{5}$$

Step 7: verify if chaos $[j] \geq 0$, then

$$\begin{aligned} \text{Loc} &= \text{Loc} + \text{inc} \\ \text{seq} &= 1 \end{aligned}. \tag{6}$$

Step 8: else, if chaos $[j] < 0$, then

$$\begin{aligned} \text{Loc} &= \text{Loc} - \text{inc} \\ \text{seq} &= 0 \end{aligned}. \tag{7}$$

Step 9: verify if dif 1 [Loc] is empty, assigning to it an element of image; else, go to Step 10 (see Figure 7).

Step 10: verify if dif 1 [Loc] is not empty and seq = 0, Loc is decreasing, until finding a formerly empty location; contrariwise, if seq = 1, Loc increases. On localizing an empty position, assign to it the element of image. This procedure is presented in Figure 8.

$$\text{dif 1}[\text{Loc}] = \text{image}[i]. \tag{8}$$

Step 11: repeat Steps 6 through 10, until arranging all of the subpixels of the vector image in dif 1.

Diffusion I algorithm is shown in Algorithm 1.

Confusion

Step 1: convert, to their value in pixel [0, 16777215], the elements of dif 1 and store them in a pix_image vector of R length:

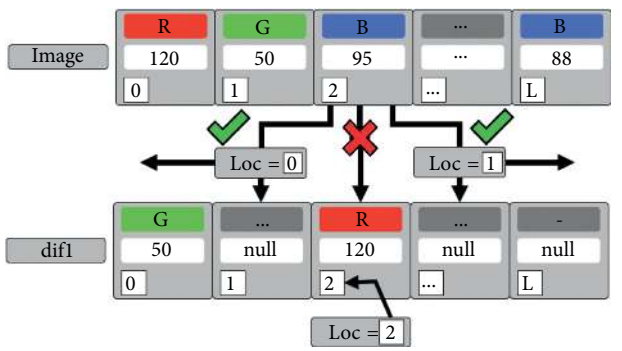


Figure 8: Locate an empty location in dif 1 to store the data.

$$\text{pix\_image} = [P_1, P_2, P_3, \ldots, P_R]. \tag{9}$$

Step 2: resolve R times the system of equation (1) using encryption keys $x2_0, y2_0, z2_0, a2, b2, c2$, yielding as a result three orbits:

$$\begin{aligned} \text{orb}\, x &= [x2_1, x2_2, \ldots, x2_R], \\ \text{orb}\, y &= [y2_1, y2_2, \ldots, y2_R], \\ \text{orb}\, z &= [z2_1, z2_2, \ldots, z2_R]. \end{aligned} \tag{10}$$

Step 3: generate integer values between 0 and 16777215, dividing an element of orb x between the absolute value of the highest number of the vector, subsequently multiplying this by 16777215. The result is rounded off and the value of orb $x[i]$ is substituted.

$$\text{orb}\, x[i] = \text{round}((\text{orb}\, x[i])/(\text{abs}(\text{Highest}(\text{orb}\, x)) * 16777215). \tag{11}$$

Step 4: convert, to a 6-digit hexadecimal, each element of pix_image, as shown in Figure 9.

Confusion algorithm is shown in Algorithm 2.

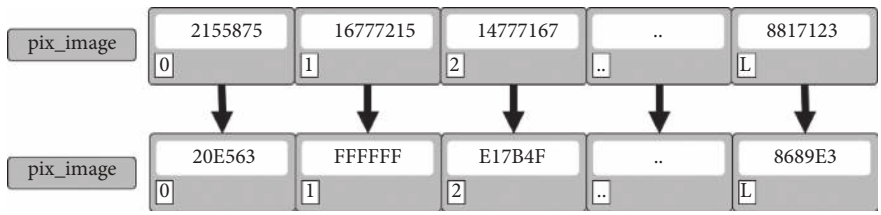


Figure 9: Convert from decimal to hexadecimal.

Step 5: store, in rotation, the first 6 decimals of an element of the vector orb y, which will be utilized to indicate the number of times that each hexadecimal digit moves among the elements stored in the vector pix_image.

$$Orby[i] = 23.64456834467$$
$$\downarrow$$
$$rotation = [6, 4, 4, 5, 6, 8]$$

Step 6: apply the algorithm of diffusion I to an element of pix_image, taking into account the key keyconf and the positions indicated in rotation; later this is converted into a decimal, as can be observed in Figure 10.

Step 7: repeat Steps 5 and 6 until applying the diffusion I algorithm to all elements of pix_image.

Step 8: store in the variable before an element of pix_image, later applying XOR between pix_image$[i]$ and keyconf 2. Repeat this procedure to encrypt all elements of pix_image. However, from the second iteration, the value stored in before is used to apply XOR with the pix_image elements instead of keyconf 2. The results will be stored in vector cipher. In this step, each pixel value is taken into account in order to apply confusion; thus, a modification in one of these will generate cascade changes in the cryptogram. This process is exhibited graphically in Figure 11.

Step 9: apply XOR between vectors cipher and orb x, with the result stored in vector conf. The procedure is shown in Figure 12.

Diffusion II. Take vector conf and apply the procedure indicated in Diffusion I, but with keys $x3_0, y3_0, z3_0, a3, b3, c3$ and stardif 2; in the end, the cryptogram is generated.

3.4.2. Algorithm for Decryption

Eliminate Diffusion II

Step 1: store the cryptogram's subpixel vector image, which has a length of $L = R_n + G_n + B_n$.

$$\text{image} = [R_1, G_1, B_1, \ldots, R_n, G_n, B_n]. \quad (12)$$

Step 2: create a vector called conf of size L.

Step 3: resolve R times the system of equation (1), with encryption keys $x3_0, y3_0, z3_0, a3, b3, c3$. The values generated of $x3_R, y3_R, z3_R$ are stored in a vector denominated chaos:

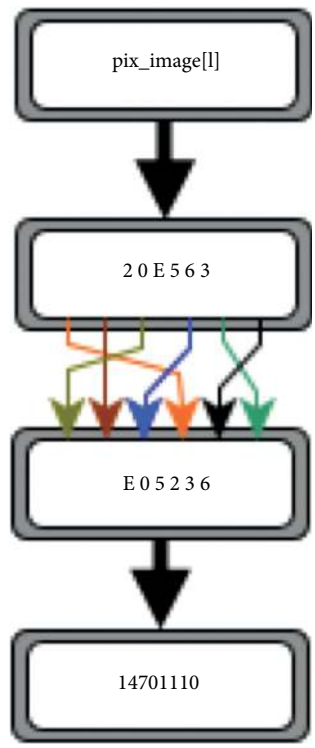


Figure 10: Change the value of the pixel.

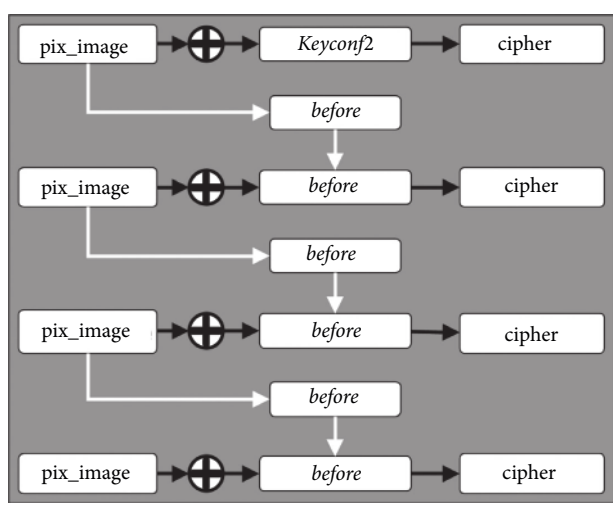


Figure 11: Apply XOR to substitute the elements of pix_image.

$$\text{chaos} = [x3_1, y3_1, z3_1, \ldots, x3_R, y3_R, z3_R]. \quad (13)$$

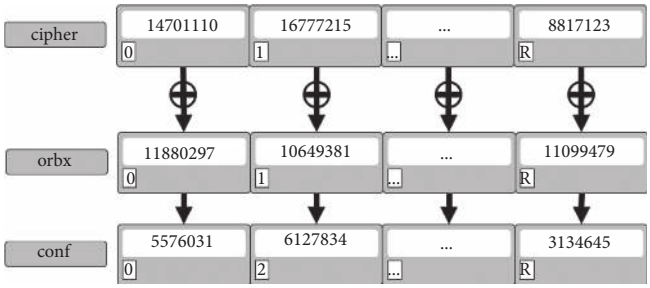


Figure 12: Apply XOR between cipher and orb x.

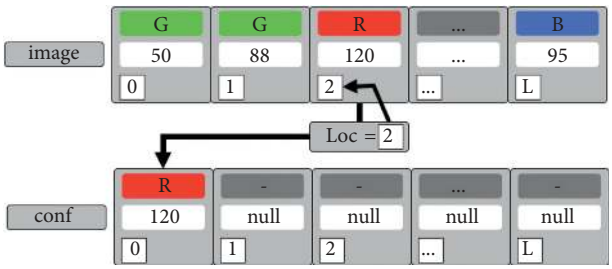


Figure 13: Place the value of image in its original position in vector.

Step 4: assign, to the variable Loc, the value of key startdif 2.

Step 5: take, from location Loc in vector image, the value and place it in conf. This procedure is shown in Figure 13.

$$\text{conf}[i] = \text{image}[\text{Loc}]. \tag{14}$$

Step 6: take the absolute value of an element of the vector chaos$[j]$, later adding it to a value of conf$[i]$, rounded off the result and assigning it to inc:

$$\text{inc} = \text{round}(\text{abs}(\text{chaos}[j]) + \text{conf}[i]). \tag{15}$$

Step 7: verify if chaos$[j] \geq 0$, then

$$\begin{aligned} \text{Loc} &= \text{Loc} + \text{inc} \\ \text{seq} &= 1. \end{aligned} \tag{16}$$

Step 8: contrariwise, if chaos$[j] < 0$, then

$$\begin{aligned} \text{Loc} &= \text{Loc} - \text{inc} \\ \text{seq} &= 0. \end{aligned} \tag{17}$$

Step 9: verify if Image[Loc] has a value, assigning it to the next position in conf; contrariwise, proceed to Step 10 (see Figure 14).

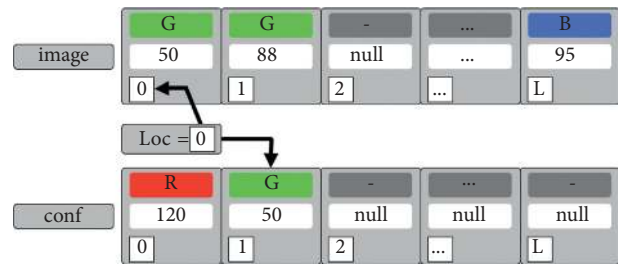


Figure 14: Assign the value of the vector image in position Loc in vector conf.

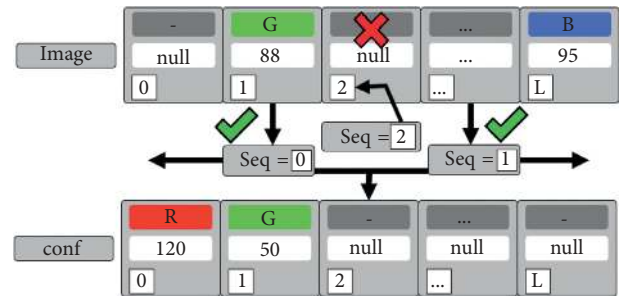


Figure 15: Localize the following site with a datum in the vector image.

Step 10: in the case of Image[Loc] being empty and seq = 0, Loc will decrease; contrariwise, if seq = 1, Loc increases. This procedure is carried out until finding a value and assigning it to conf (see Figure 15).

$$\text{conf}[i] = \text{image}[\text{Loc}]. \tag{18}$$

Step 11: repeat Steps 6–10 until arranging all Image elements in conf.

Eliminate diffusion II algorithm is shown in Algorithm 3.

Eliminate Confusion

Step 1: convert the elements of vector conf from subpixels into pixels:

$$\text{conf} = [P_1, P_2, P_3, \ldots, P_R]. \tag{19}$$

Step 2: create orb x, orb y, orb z (Step 2, confusion).

Step 3: again, generate whole values between 0 and 16777215 (Step 3, confusion):

$$\text{orb}\,x[i] = \text{round}\left(\frac{\text{orb}\,x[i]}{\text{abs}(\text{Highest}(\text{orb}\,x))} * 16777215\right). \tag{20}$$

Step 4: apply XOR between conf and orb x, with the result stored in cipher, as can be observed in Figure 16.

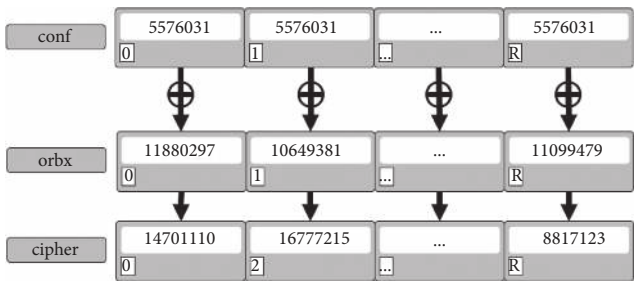


Figure 16: Apply XOR between conf and orb x.

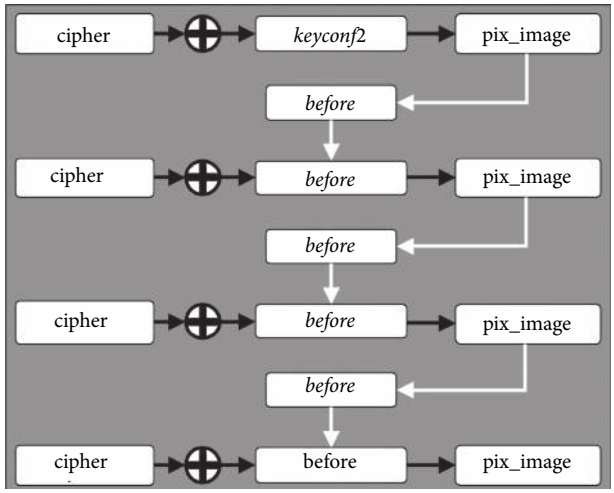


Figure 17: Restore the vector pix_image, reverting XOR to cascade.

Step 5: assign, to variable before and to vector pix_image, the result of applying XOR between the first element of cipher and keyconf 2. Repeat this procedure to decrypt the entire vector cipher. But, from the second iteration, the value stored in before is utilized to apply XOR with the elements after cipher, instead of keyconf 2, as can be observed in Figure 17.

Step 6: regenerate rotation, taking the 6 first decimals of an element of vector orb y.

$$Orby[i] = 23.64456834467$$
$$\downarrow$$
$$rotation = [6, 4, 4, 5, 6, 8]$$

Step 7: implement the algorithm to eliminate diffusion II, taking an element of pix_image, the key keyconf and the positions indicated in rotation; later, convert this into a decimal, as can be observed in Figure 18.

Step 8: repeat Steps 6 and 7 until rearranging each element of vector pix_image.

Step 9: convert pix_image from a hexadecimal into a decimal.

Eliminate confusion algorithm is shown in Algorithm 4.

Eliminate Diffusion I. Take vector pix_image and apply the procedure indicated in Eliminate Diffusion II, but with keys $x1_0, y1_0, z1_0, a1, b1, c1$, and stardif 1; finally, the original image is recovered.

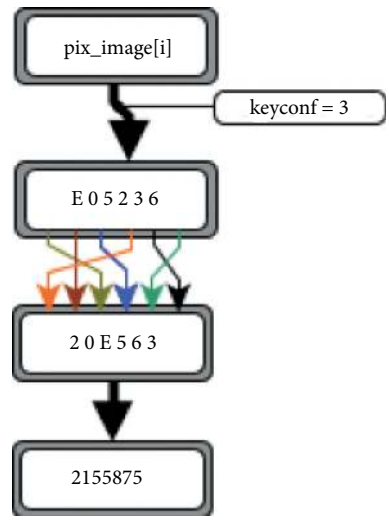


Figure 18: Restore the value of the pixel.

4. Results

The proposed communication system works with images in the .png format and implements the OpenCV Library for face detection and identification. We use this library to determine the characteristics that photographs must possess to encrypt faces more effectively; 171 photographs of 1,280 x 960, 4,032 x 3,024, and 2,704 x 2,652, with controlled, semicontrolled, and uncontrolled environments, were used. According to the tests, it was discovered that detection affects the lighting, the angle of the face, and the facial expressions; also, in the OpenCV Library in semicontrolled environments where one wants to identify more than one person, some faces were correctly identified and encrypted, but others were not. Based on these results, it was determined that it is best to use the photographs in controlled environments for the system to function properly. In semicontrolled and uncontrolled environments, it is best to use the system manually.

Table 1 presents the results generated with the tests. Positive indicates that detection, recognition, and encryption worked correctly. In the intermediate, some faces were not recognized or were confused with that of another person, and in the negatives, it did not recognize persons.

To verify the robustness of the encryption system, the statistical analysis is presented in this section, as well as different attacks of the algorithm proposed, and this is compared with the results of other investigations. Figure 19(a) shows the Lena image of 256 x 256 that is used for all tests, Figure 19(b) shows the cryptogram generated when applying diffusion technique I, Figure 19(c) represents the cryptogram when implementing only the confusion technique, Figure 19(d) shows the cryptogram that was generated using diffusion I and confusion, and Figure 19(e) presents the cryptogram with the diffusion I-confusion-diffusion II techniques. The decryption process is presented in Figure 19(f), which presents the cryptogram obtained when eliminating diffusion II, Figure 19(g) shows the cryptogram when confusion and diffusion II are eliminated,

```
Input: vector image, stardif1, x1_0, y1_0, z1_0, a1, b1, c1
Output: vector dif 1
 (1) L = subpixels of image
 (2) R = (L/3)
 (3) i = 1
 (4) for j = 1 to R do
 (5)       Chaos [j] = Solve Equation (1) with RungeKutta − 4 and keys x1_0, y1_0, z1_0, a1, b1, c1.
 (6) end
 (7) Loc = stardif 1
 (8) dif 1 [Loc] = image [i]
 (9) i = i + 1
(10) for j = 1 to L do
(11)      inc = round (|chaos [j]| + dif 1 [Loc])
(12)       if chaos [j] ≥ 0 then
(13)          Loc = Loc + inc
(14)         seq = 1
(15)      end
(16)       else if chaos [j] < 0 then
(17)          Loc = Loc − inc
(18)          seq = 0
(19)       end
(20)      if (dif 1 [Loc] is NULL) then
(21)         dif 1 [Loc] = image [i]
(22)       end
(23)      else if ((dif 1 [Loc] is not NULL) and (seq = 0)) then
(24)          while (dif 1 [Loc] is not NULL)
(25)             Loc = Loc − 1
(26)         end
(27)          dif 1 [Loc] = image [i]
(28)       end
(29)       else if ((dif 1 [Loc] is not NULL) and (seq = 1)) then
(30)          while (dif 1 [Loc] is not NULL)
(31)             Loc = Loc + 1
(32)         end
(33)          dif 1 [Loc] = image [i]
(34)      end
(35)      i = i + 1
(36) end
```

Algorithm 1: Diffusion I.

that is, only with the diffusion I technique, and Figure 19(h) shows the decrypted image. It can be clearly observed that, individually, both the diffusion technique (Figure 19(b)) and the confusion technique (Figure 19(c)) render the image unrecognizable.

4.1. Statistical Analysis

4.1.1. Histograms. In Figures 20(a), 20(c), and 20(e), the histograms are presented of the RGB channels corresponding to Figure 19(a) and to the images of Figures 20(b), 20(d), and 20(f)), those of the cryptogram of Figure 19(e). As can be observed, there are no similarities in the distributions of the color frequencies between the original image and the cryptogram. That is, on observing the histograms corresponding to the cryptogram, it can be clearly seen that there is greater homogeneity in contrast with the originals; therefore, it is difficult for an attacker to find a relation between the original image and the cryptogram.

4.1.2. Correlation. These permit the detection of the linear relation between two variables or images; this analysis yields an R correlation coefficient, which is found within the range of [−1 to 1]: the more it nears 1 and −1, the more both images are similar or alike if they yield these exact values. On the other hand, the nearer they are to 0, the greater the number of differences, or they are completely different if $R = 0$.

Figure 21 presents the correlation diagram of the original image with respect to the cryptogram, and it can be observed that coefficient $R = 0.00000267625$ is very close to 0; thus, it is more difficult for an attacker to attempt to find a relation between both of these.

In Figure 22, the correlation is able to be observed of the original image vs. the decrypted one, which yields the result

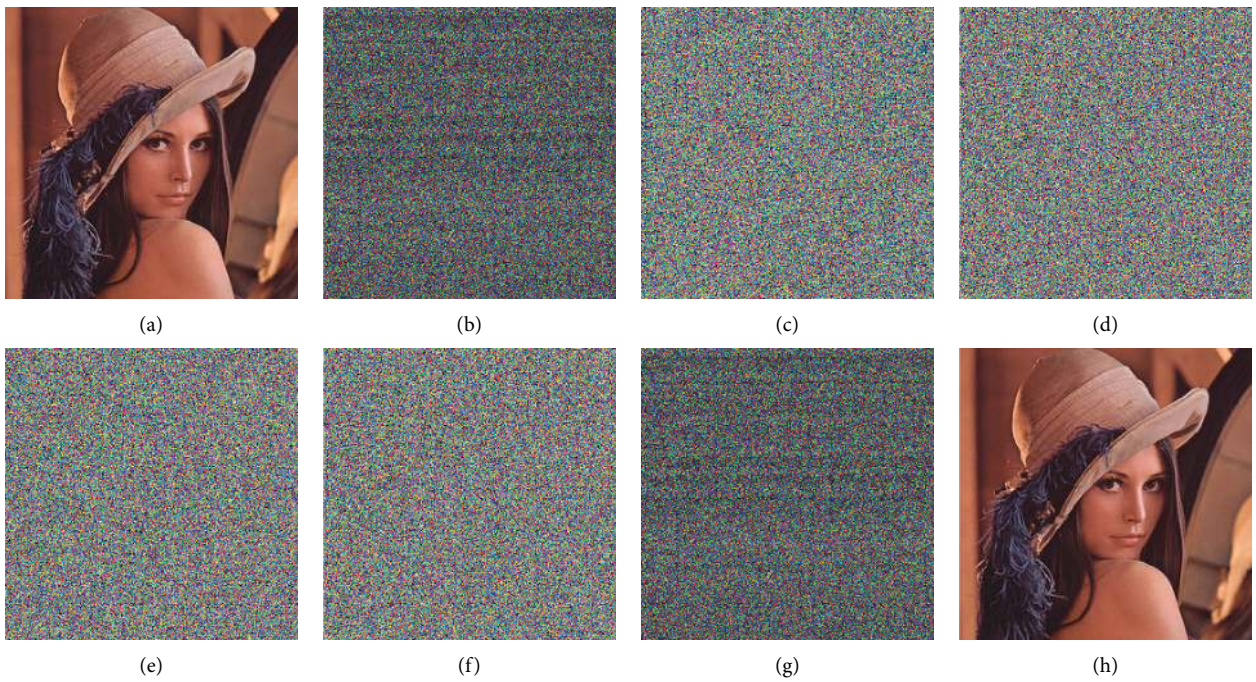


Figure 19: Encryption and decryption process: (a) original image, (b) cryptogram with diffusion I, (c) cryptogram with confusion, (d) cryptogram with diffusion I and confusion, (e) full cryptogram with diffusion I, confusion, and diffusion II, (f) cryptogram recovered by removing diffusion II, (g) cryptogram recovered by removing diffusion II and confusion, and (h) decrypted image.

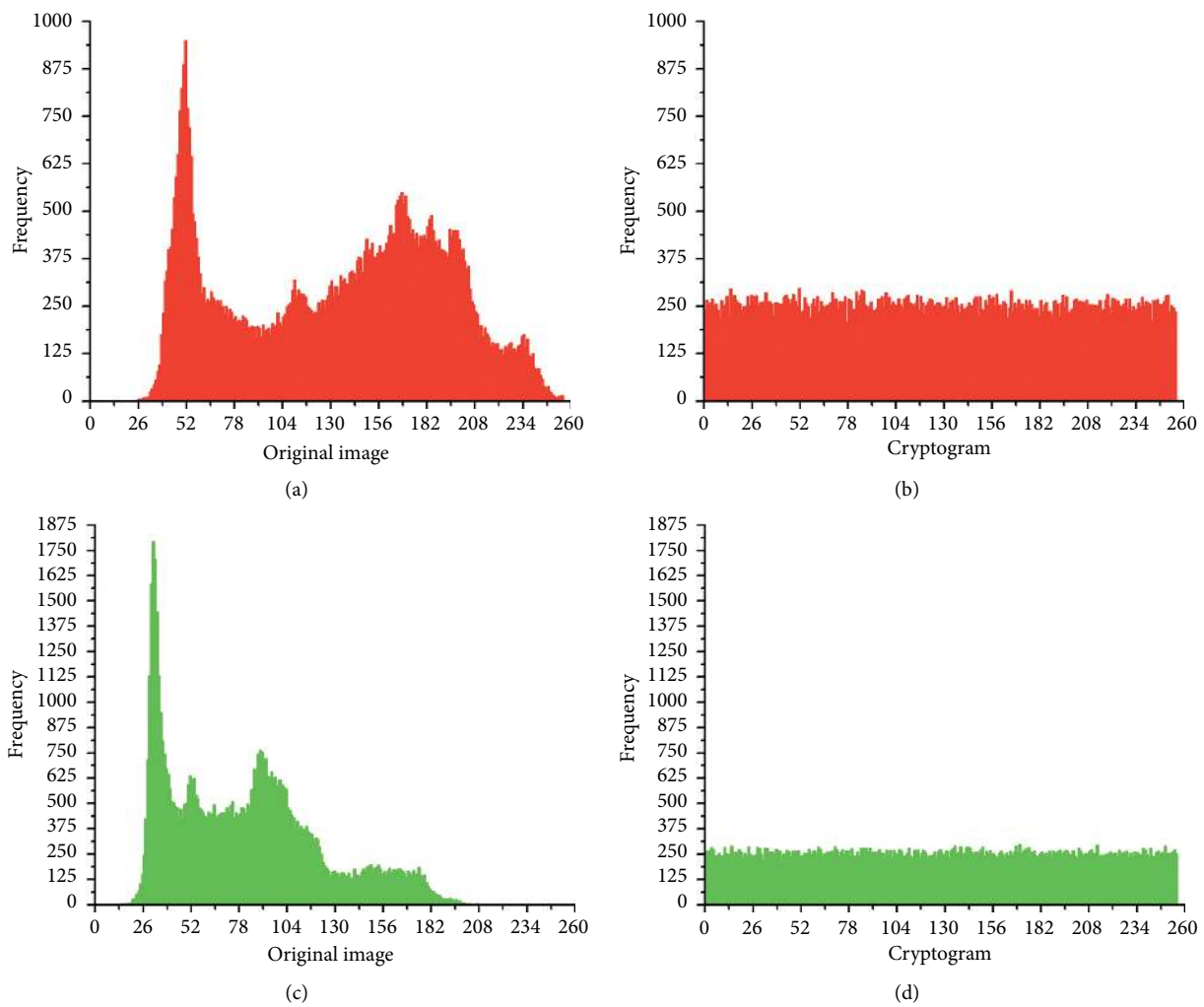


Figure 20: Continued.

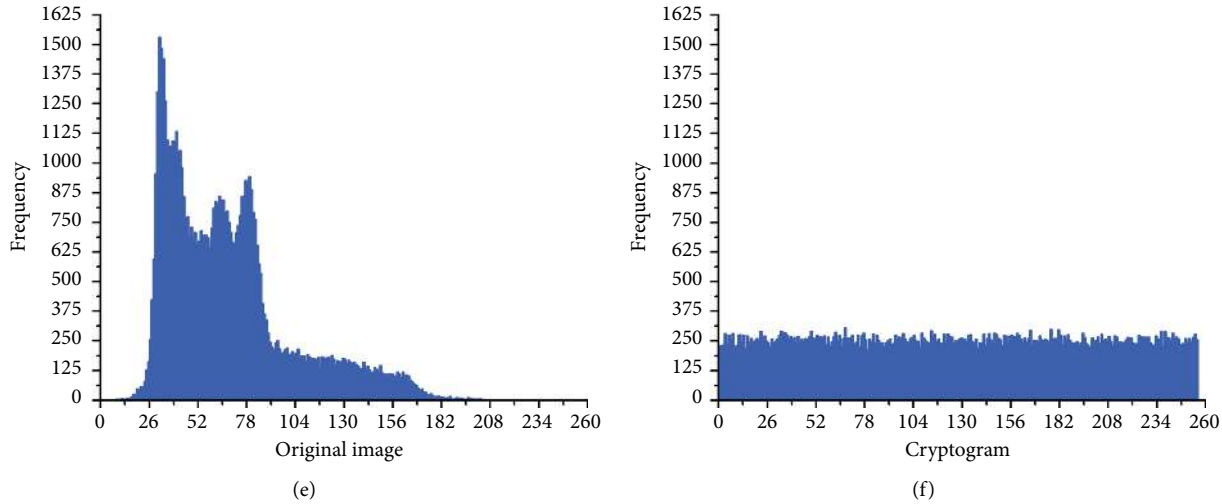


Figure 20: Analysis of histograms: (a, c, e) original image of Lena red, green, and blue, respectively; (b, d, f) the histograms of the red, green, and blue channel cryptogram.

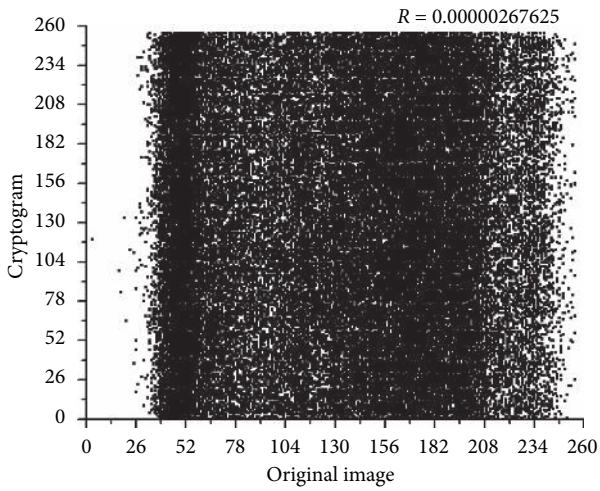


Figure 21: Diagram of the correlation between the original image vs. the cryptogram.

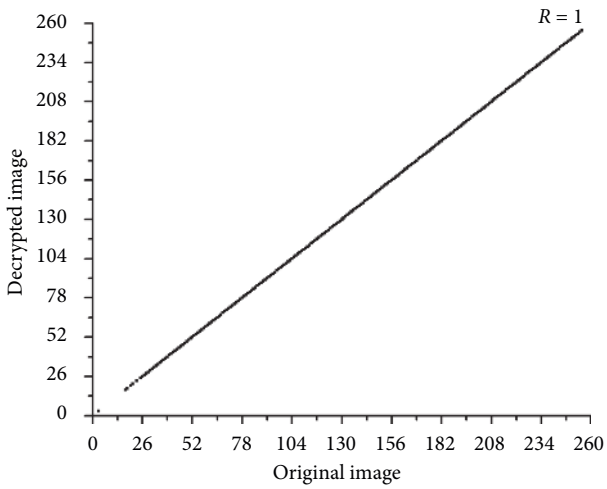


Figure 22: Diagram of the correlation of the original image vs. the decrypted.

of $R = 1$. Thus, the encrypted system guarantees integrity on recovering 100% of the information.

In Table 2, the results are shown of the correlation analysis. In the first column, the coefficient correlation is presented as generated on comparing the original image vs. the cryptogram; later, the original image was compared vs. the decrypted, with the results presented in column 2. As can be observed, our cryptosystem presents excellent results in comparison with those of other state-of-the-art cryptosystems.

The correlation coefficient was calculated of 1,000 pairs of pixels taken at random, that is, the original image as well as the cryptogram horizontally, vertically, and diagonally, according to components R, G, and B. Figures 23(a)–23(i) present the correlation diagrams generated from the original image, while those of Figures 24(a)–24(i) depict the distribution of the cryptograms.

Table 3 displays the correlation coefficients of Figures 23(a)–23(i) and of Figures 24(a)–24(i), which are compared with those of other investigations.

Table 4 presents a comparison of the horizontal, vertical, and diagonal distribution of channels R, G, and B, but now taking into account 5,000 pairs of adjacent pixels.

In the results of Tables 3 and 4, it can be clearly observed that our system generates a correlation in the cryptograms that is much closer to 0, nearly null. Therefore, the proposed encryption technique renders the loss of the relation among adjacent pixels in the three directions.

4.1.3. Entropy. Entropy allows verification of the level of randomness or of how unpredictable the pixel values are in an image, which is the determination of the level of disorder of a cryptogram: the more the latter nears 8 or is equal to it, the better the processes of diffusion and confusion of the cryptogram and is more robust to an entropy attack, due to the degree of randomness of the

Input: vector dif 1, $x2_0$, $y2_0$, $z2_0$, $a2$, $b2$, $c2$, keyconf[0, 5], keyconf 2 [0, 16777215].
Output: vector conf
(1) pix_image = convert dif 1 to its values in pixels.
(2) R = length of pix_image.
(3) **for** i = 1 to R **do**
(4) orb $x[i]$, orb $y[i]$, orb $z[i]$ = Solve Equation (1) with RungeKutta − 4 and keys $x2_0$, $y2_0$, $z2_0$, $a2$, $b2$, $c2$.
(5) **end**
(6) **for** i = 1 to R **do**
(7) orb x $[i]$ = round (orb x $[i]$/|highest (orb x)|) ∗ 16777215
(8) **end**
(9) pix_image = convert to hexadecimal (pix_image)
(10) **for** i = 1 to R **do**
(11) rotation = store 6 decimals of orb $y[i]$
(12) pix_image$[i]$ = apply diffusion I pseudocode with keyconf, rotation and pix_image$[i]$
(13) pix_image[i] = convert to decimal (pix_image[i])
(14) **end**
(15) before = pix_image[1]
(16) cipher[1] = pix_image[1] XOR keyconf 2
(17) **for** i = 2 to R **do**
(18) cipher [i] = before XOR pix_image[i]
(19) before = pix_image$[i]$
(20) **end**
(21) **for** i = 1 to R **do**
(22) conf $[i]$ = orb x $[i]$ XOR cipher$[i]$
(23) **end**

Algorithm 2: Confusion.

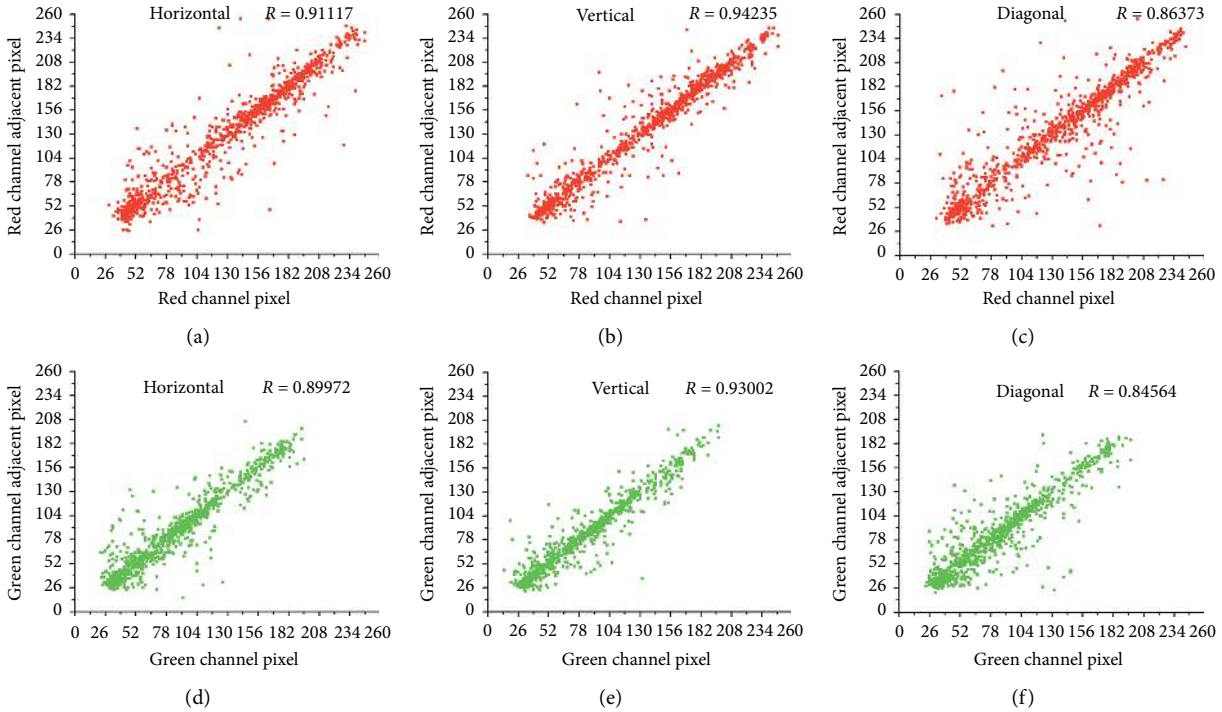


Figure 23: Continued.

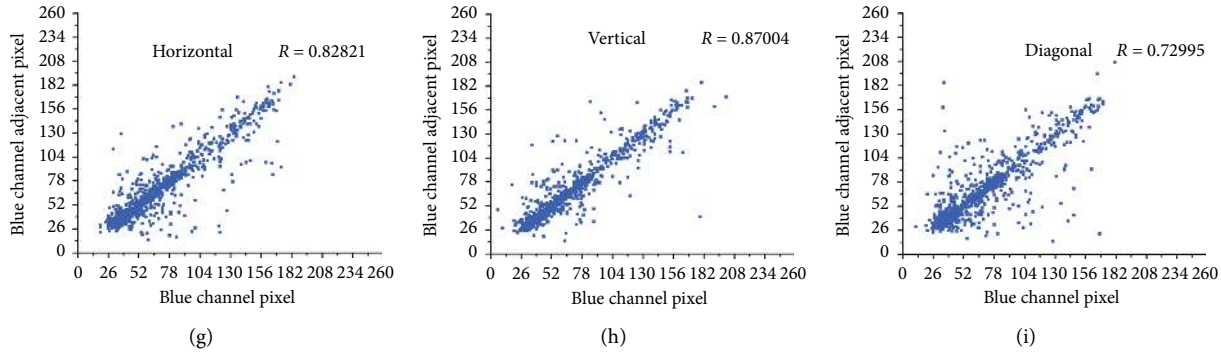


Figure 23: Correlation of pairs of the horizontal, vertical, and diagonal pixels of the original image according to the RGB channels.

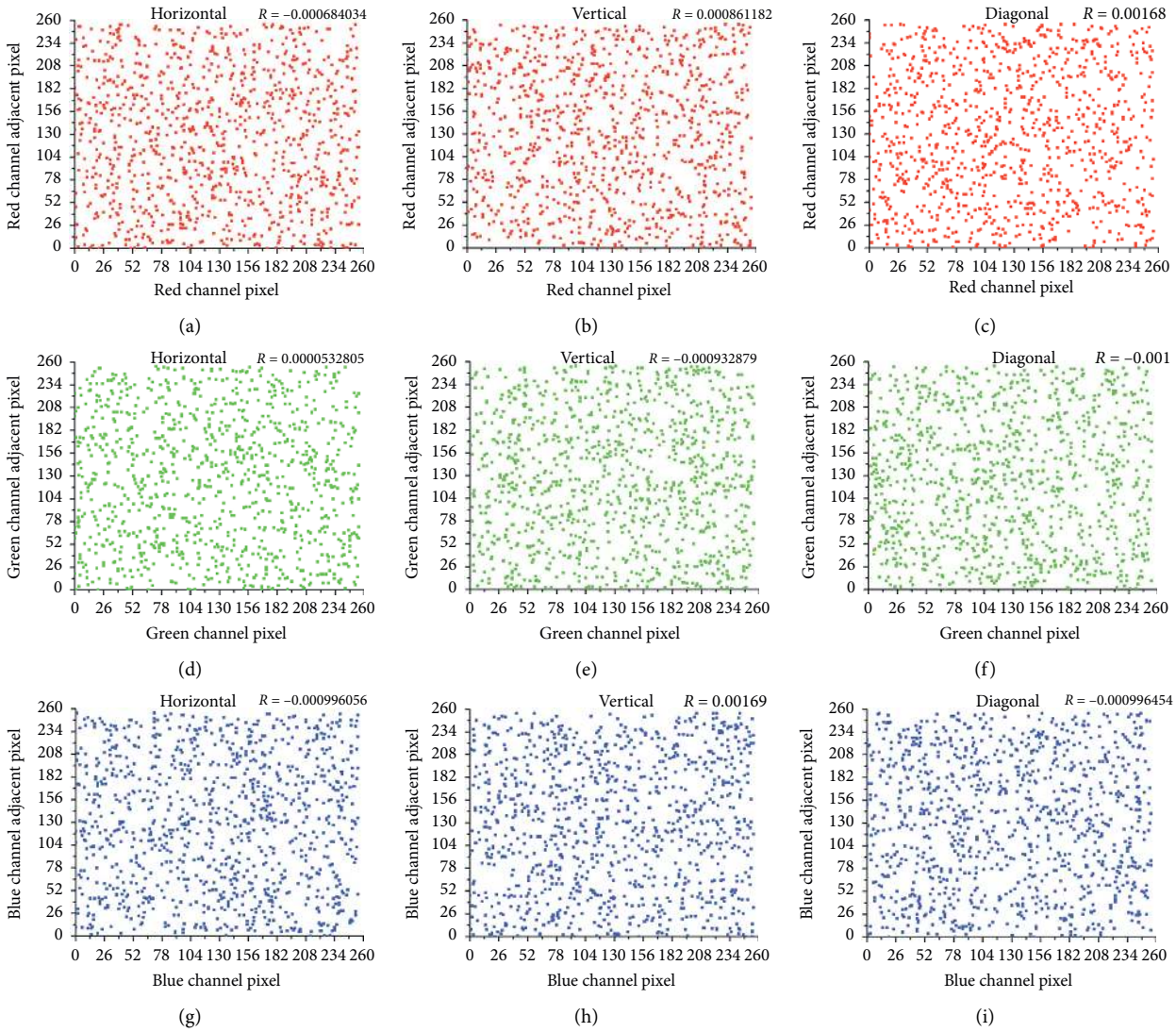


Figure 24: Correlation of horizontal, vertical, and diagonal pixels of the cryptogram according to the RGB channels.

cryptograms. Thus, this is unpredictable. Table 5 depicts the entropy of the proposed system and compares it with other systems.

4.1.4. NPCR (Number of Pixel Change Rate). The NPCR permits the determination of the strength of the encryption with respect to a differential attack; that is, it determines the

```
Input: vector cryptogram, stardif 2, x3_0, y3_0, z3_0, a3, b3, c3.
Output: vector conf
(1) image = convert cryptogram to its values in subpixels.
(2) L = subpixels of image
(3) R = (L/3)
(4) i = 1
(5) for j = 1 to R do
(6)       Chaos[j] = Solve Equation (1) with RungeKutta − 4 and keys x3_0, y3_0, z3_0, a3, b3, c3.
(7) end
(8) Loc = stardif 2
(9) conf[i] = image[Loc]
(10) i = i + 1
(11) for j = 1 to L do
(11)       inc = round(|chaos[j]| + conf[i])
(12)      if chaos[j] ≥ 0 then
(13)        Loc = Loc + inc
(14)        seq = 1
(15)       end
(16)       else if chaos[j] < 0 then
(17)        Loc = Loc − inc
(18)        seq = 0
(19)       end
(20)       if (image [Loc] is not NULL) then
(21)        conf [i] = image[Loc]
(22)      end
(23)       else if ((image[Loc] is NULL) and (seq = 0))then
(24)      while (image[Loc] is NULL)
(25)           Loc = Loc − 1
(26)         end
(27)         conf [i] = image[Loc]
(28)       end
(29)       else if ((image[Loc] is NULL) and (seq = 1))then
(30)         while (image[Loc] is NULL)
(31)          Loc = Loc + 1
(32)        end
(33)         conf [i] = image[Loc]
(34)      end
(35)       i = i + 1
(36) end
```

Algorithm 3: Eliminate diffusion II.

percentage of changes in the location of the pixels of two images. NPCR should yield a value near 100%, indicating that both images are very different.

4.1.5. UACI (Unified Average Changed Intensity). The average is calculated of the differences in the intensity of the pixels of the two images. This should yield a result greater than 33%.

NPCR and UACI were calculated with two cryptograms generated with nearly all of the same keys, with the exception of one carried out with a small change in the least significant decimal digit. NPCR results were obtained of above 99% and those of UACI were above 33%. Therefore, the proposed system takes as input a plane image and with a small change, generating a very different cryptogram. Table 6 presents the results of this analysis.

In Table 7, the result is presented of the analysis of the NPCR and UACI of the two cryptograms that were generated from the two images that were only differentiated because in one of these, the least significant bit of a red subpixel was changed; it also yielded NPCR> 99% and UACI> 33 as a result. Therefore, the proposed system is very sensitive to the plane image and would provide robustness when encountering a differential attack.

4.1.6. Avalanche Effect. This analysis reflects the sensitivity of the encryption algorithms to a small change in the parameters (in the key or in the plane text) of the entry

Input: vector conf, $x2_0$, $y2_0$, $z2_0$, $a2$, $b2$, $c2$, keyconf[0, 5], keyconf 2 [0, 16777215].
Output: vector pix_image
(1) conf = convert conf to its values in pixels.
(2) R = pixels of conf.
(3) **for** i = 1 to R **do**
(4) orb $x[i]$, orb $y[i]$, orb $z[i]$ = Solve Equation (1) with RungeKutta − 4 and keys $x2_0$, $y2_0$, $z2_0$, $a2$, $b2$, $c2$.
(5) **end**
(6) **for** i = 1 to R **do**
(7) orb $x[i]$ = round(orb $x[i]$/|highest (orb x)|) ∗ 16777215)
(8) **end**
(9) **for** i = 1 to R **do**
(10) cipher$[i]$ = orb $x[i]$XOR conf$[i]$
(11) **end**
(12) before = cipher[1]XOR keyconf 2
(13) pix_image[1] = before
(14) **for** i = 2 to R **do**
(15) pix_image [i] = before XOR cipher$[i]$
(16) before = pix_image$[i]$
(17) **end**
(18) pix_image = convert to hexadecimal (pix_image)
(19) **for** i = 1 to R **do**
(20) rotation = store 6 decimals of orb $y[i]$
(21) pix_image$[i]$ = apply eliminate diffusion II with keyconf, rotation and pix_image$[i]$
(22) pix_image[i] = convert to decimal (pix_image[i])
(23) **end**

Algorithm 4: Eliminate confusion.

Table 1: Results of the tests with 171 images.

Type	Total photos	Positive	Intermediate	Negative
Controlled	44	44	—	—
Semicontrolled	108	67	33	8
Uncontrolled	19	—	—	19
Total	171	111	33	27

Table 2: Comparison of correlation coefficients.

Cryptogram	Decrypt	Reference
0.00000267625	1	Proposed
0.002851	1	[25]
0.000686	—	[26]

information, which should cause a great change in the cryptogram [35].

This test was utilized to verify the differences between the two Lena images, in which, in one of these the least significant bit was modified in channel R in one of the initial pixels, and later compared with another without any change. Figure 25(a) depicts the remainder between the two pixels and difference was able to be appreciated clearly. Figure 25(b) presents the remainder of the two pixels of the 2 cryptograms generated on encrypting the 2 Lena images, and it can be clearly seen that these are different. The proposed algorithm passes this test due to that the diffusion technique is applied on two occasions and, in addition, the pixels were taken into account on coding during confusion.

Figures 26(a) and 26(b) present the tests carried out of the avalanche effect, but with the least significant bit of the R channel altered at the end of one of the Lena images.

4.1.7. Sensitivity of the Key. To verify whether the image can be recovered with different keys, Figure 19(a) was encrypted with the keys of Table 8, generating the cryptogram of Figure 27(a), which was recovered correctly utilizing the same keys with which it was encrypted (Figure 27(b)). Recovery was also attempted by the alteration of $x3_0$ of 0.75749573951 to 0.75749573952, yielding as a result Figure 27(c). With the latter, it was proven that, on performing a small change in one of the keys, the image was unable to be recovered correctly.

4.1.8. Speed Tests. Algorithm speed was tested in an Omen 17 An101la laptop, with an Intel Core i7-8750 H 2.20 GHz processor, a Wester Digital Black 512 GB SSD hard drive, 16 GB RAM DDR4-2666 MHz, and the encryption algorithm was performed with Java language programming. Table 9 presents a comparison of the encryption speed of the proposed system with other investigations.

4.1.9. Key Space. The key space defines the number of different keys that can be used by a cryptosystem. The proposed algorithm uses six in diffusion technique I $x1_0, y1_0, z1_0, a1, b1, c1$, another six when diffusion II is implemented $x3_0, y3_0, z3_0, a3, b3, c3$, and also, when

Table 3: Coefficient of the horizontal, vertical, and diagonal correlation of pairs of adjacent pixels.

Distribution	Plane image			Encrypted image			Reference
	R	G	B	R	G	B	
Horizontal	0.9567	—	—	-0.0026	—	—	
Vertical	0.9239	—	—	-0.0038	—	—	[27]
Diagonal	0.8888	—	—	0.0017	—	—	
Horizontal	0.9843	0.9793	0.8784	0.0181	−0.0067	0.0154	
Vertical	0.9928	0.9878	0.9204	−0.0099	0.0126	0.0063	[21]
Diagonal	0.9810	0.9742	0.8717	0.0085	0.0127	−0.0155	
Horizontal	0.933561	0.905638	0.859151	−0.00020268	−0.00153669	−0.00343206	
Vertical	0.9638	0.946469	0.905638	−0.00014400	0.00405293	−0.00555131	[28]
Diagonal	0.935289	0.907703	0.860089	−0.00309766	−0.000400902	−0.00026181	
Horizontal	0.93338	0.9049	0.8597	−0.0137	−0.0049	0.0038	
Vertical	0.9641	0.9467	0.9061	0.0038	−0.0008	0.0017	[29]
Diagonal	0.9074	0.8802	0.8383	0.0036	−0.0012	0.0008	
Horizontal	0.91117	0.89972	0.82821	−0.000684034	0.0000532805	−0.000996056	
Vertical	0.94235	0.93002	0.87004	0.000861182	−0.000932879	0.00169	Proposed
Diagonal	0.86373	0.84564	0.72995	0.00168	−0.001	−0.000996454	

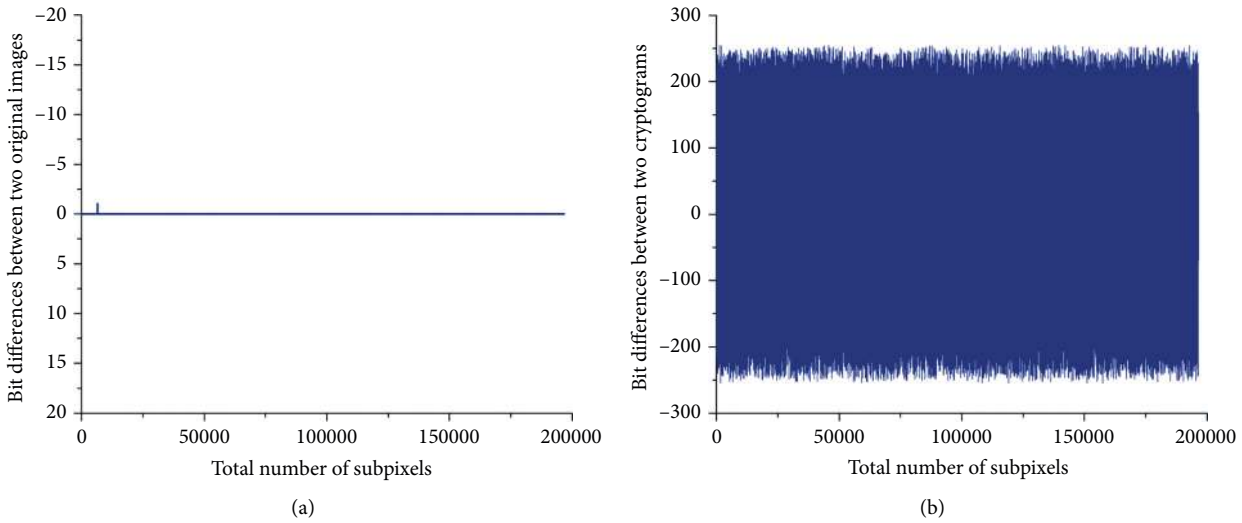


Figure 25: Avalanche effect: (a) difference at the initiation of a Lena image; (b) differences in the cryptograms.

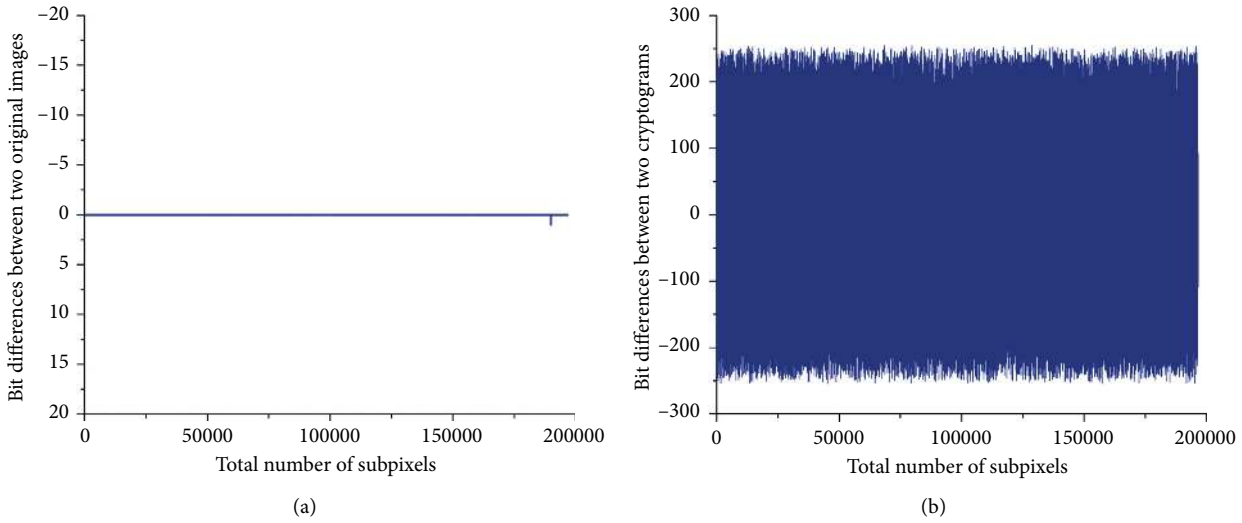
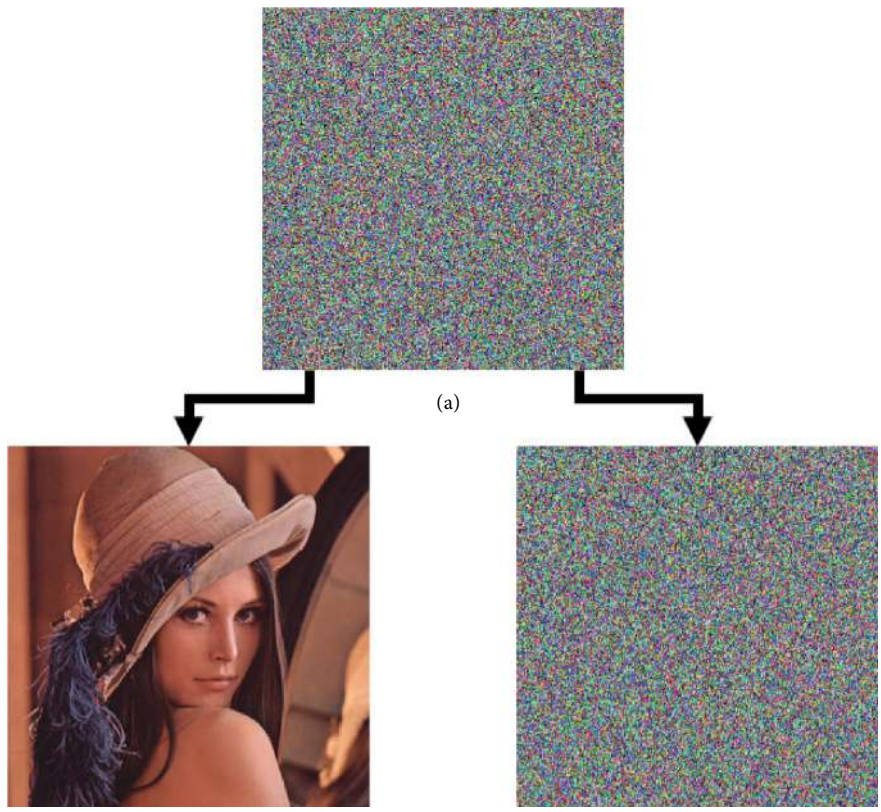


Figure 26: Avalanche effect: (a) difference of channel R at the end of one of the Lena images; (b) difference of the cryptograms.

Table 4: Coefficient of horizontal, vertical, and diagonal correlation of 5,000 pairs of adjacent pixels.

Distribution	Plane image			Encrypted image			Reference
	R	G	B	R	G	B	
Horizontal	0.932667029586456	0.922281493233349	0.893825219391821	0.003535161409385	−0.009706871635486	0.018571551282381	
Vertical	0.962408763131362	0.954693507829754	0.934300708191352	−0.004076324725952	0.005312032028056	0.010691355134514	[30]
Diagonal	0.907966009398585	0.880408776021311	0.863453990135567	−0.041028497842243	−0.008522993738987	−0.017458909511577	
Horizontal	0.9777	0.9604	0.9101	0.0135	−0.0835	−0.0170	[31]
Horizontal	0.91669	0.86071	0.82829	0.00011123	−0.000197324	−0.000194282	
Vertical	0.935	0.91865	0.87415	−0.000126717	0.000388285	−0.000196329	Proposed
Diagonal	0.87405	0.84351	0.78519	−0.0000893574	−0.000165072	−0.000191976	



(a) (b) (c)

Figure 27: Test of the sensitivity of the key.

Table 5: Entropy of different systems of encryption.

Entropy	Reference
7.999	[32]
7.9988	[33]
7.997	[31]
7.997	[34]
7.997133	[28]
7.99654	[29]
7.9998	[19]
7.9988	Proposed

Table 6: NPCR and UACI of cryptograms encrypted with a small change in one of their keys.

Analysis	R	G	B	Reference
NPCR	99.6552	99.6277	99.588	[27]
UACI	33.4846	33.4132	33.3441	
NPCR	99.5971	99.6002	99.6368	Proposed
UACI	33.3895	33.2925	33.5537	

Table 7: NPCR and UACI of cryptograms encrypted with a change in one of the subpixels.

Analysis	R	G	B	Reference
NPCR	99.63	99.60	99.61	[31]
UACI	33.31	33.34	33.43	
NPCR	99.5659	99.5658	99.5959	[21]
UACI	33.2829	33.3459	33.3270	
NPCR	99.6429	99.6140	99.6277	[20]
UACI	33.3935	33.5637	33.4814	
NPCR	99.6124	99.6140	99.6201	[28]
UACI	33.4235	33.4838	33.5983	
NPCR	99.5986	99.6170	99.6231	Proposed
UACI	33.4074	33.5645	33.5042	

implementing confusion, yet other six keys are used $x2_0, y2_0, z2_0, a2, b2, c2$. The precision of the floating-point number is 10^{-15}; therefore, the key space is $10^{270} \approx 2^{896}$. Additionally, the keys keyconf, keyconf 2,

Table 8: Keys employed for encryption.

Diffusion I	Confusion	Diffusion II
$x1_0 = 3.75846375846$	$x2_0 = 2.58395739578$	$x3_0 = 0.75749573951$
$y1_0 = 1.37483648365$	$y2_0 = 1.27490458294$	$y3_0 = -4.4839573944$
$z1_0 = 7.29347658364$	$z2_0 = 7.57395729493$	$z3_0 = -12.391038593$
$a1 = 0.25783958392$	$a2 = 0.15639204821$	$a3 = 0.22491840583$
$b1 = 5.74950385024$	$b2 = 18.1749374935$	$b3 = 15.1947394573$
$c1 = 0.57583759382$	$c2 = 0.16739275921$	$c3 = 0.10385738593$
startdif 1 = 15	keyconf = 3	startdif 2 = 89483
—	keyconf 2 = 138495	—

Table 9: Encryption time (seconds).

Image (pixels)	Proposed	[31]	[19]	[26]	[36]
256 x 256	0.134438	0.0657			0.21
512 x 512	0.41285	0.2432	0.007	0.290	1.19

Table 10: Secret key space.

Proposed	[31]	[37]	[36]	[26]	[19]	[34]	[28]
2^{896} 4 variable keys 10^{270}	2^{128}	2^{192}	2^{200}	2^{298}	2^{256}	10^{112}	10^{58}

Table 11: Encryption quality analysis.

Maximum deviation			Irregular deviation			Deviation from ideality			Reference
R	G	B	R	G	B	R	G	B	
83,912	81,294.5	87,629.5	32,102	33,453	38,159	0.04852	0.04739	0.05227	[28]
5.0069×10^4	36,473	65,826	45,906	25,140	2750	0.7734	0.5492	0.9882	[29]
36,437	59,526	70,232	37,468	60,530	71,238	0.003463745	0.00382995	0.00369262	Proposed

startdif 1, and stardif 2 are implemented with a variable key space. Table 10 depicts a comparison of the space key of the proposed system with other investigations.

For the cryptosystem to be safe, there must be a key space greater than 2^{100} [38]; therefore, the proposed system will support a brute force attack.

4.1.10. Encryption Quality Analysis. To verify the quality of the encryption, we calculated the maximum deviation: to evaluate the deviation between the intensity of pixels of the original image and the cryptogram. If the value is high, the quality of the encryption is good. Irregular deviation allows to statistically evaluate the quality of the intensity of the pixels in the cryptogram. If it is high, the pixels are uniformly distributed. Deviation from ideality: this detects the probability of distribution of the pixels in the cryptogram. The closer to zero it is, the better the quality of the cryptogram [39]. (Table 11)

5. Conclusions

In this article, a communication system is proposed that utilizes portable SBC devices and a server that performs detection, identification, and selective facial encryption, and the system implements diffusion and confusion techniques that, when applied individually, render cryptograms unrecognizable. However, to strengthen security, the techniques can be applied in layers, that is, the order in which diffusion or confusion is implemented and the number of times they are applied depends on the user. In fact, each time one of these techniques is used, the number of keys increases; thus, it is more difficult for an attacker to retrieve the original image. Based on the tests, it is recommended to apply the algorithm in the following order: diffusion I-confusion-diffusion II, for optimal encryption in matters of time and security. Furthermore, the algorithm considers the order and value of the pixels to encrypt; therefore, a small change in these or in the keys generates different cryptograms resistant to an avalanche attack, which not many cryptographic systems resist. Encryption time is longer than in other systems because it is preferred to provide more security when implementing a much larger key space, but if speed is preferred over security, only diffusion or confusion can be implemented.

To verify the system's robustness, a statistical analysis was carried out, in which it can be observed that the histograms generated from the cryptogram present excellent homogeneity in pixel distribution. In addition, by means of the diagrams of correlation and entropy, it was determined

that the algorithm provides relevant disorder. Additionally, through NPCR and UACI, the differences were verified between the location and the intensity of the pixels, generating results of NPCR>99 and UACI>33; therefore, the system is robust on confronting differential attacks. Additionally, the cryptosystem does not obligate the implementation of a unique mathematical model; it is possible to utilize any model that generates chaotic orbits, which renders it highly sensitive to the encryption keys. For correct facial detection and identification, it is necessary to conduct the training using photographs of the persons to be identified and encrypted. The greater the number of photographs employed during this stage, the more accurate the recognition will be. In addition, the faces should not be superimposed or incomplete, and it is recommended that the photographs be controlled or semicontrolled in order to avoid that, at the time of detection and recognition, the position of the camera, the angle of the faces, the facial expressions, luminosity, and distance can affect. Also, when performing the recognition, this can affect whether persons have different makeup, hair tone, and natural aging. Therefore, it is recommended to train the system with different changes. During the tests carried out with the OpenCV Library, there have been cases in which persons are very similar and are confused with others. The library can also sometimes recognize some objects as faces. To correct this problem, the proposed system allows for manually selecting those one wants to encode. Another problem that can be generated is that the cryptograms overlap when the faces are close together.

On utilizing the SBC, the use is reduced of hardware resources and it provides greater portability in that only when necessary is a superior database consulted in a remote server; also, the algorithm can be utilized in mobile devices. The communications system can be employed in police departments and military units to carry out searches for persons, for video surveillance, to safeguard the identity of persons in companies, for journalistic uses, and for uses in the social and communications media.

Data Availability

The images, algorithms, and study data are included in this article.

Conflicts of Interest

The authors report that they have no conflicts of interest about the publication of this article.

Acknowledgments

This project was supported by CONACYT in project 701231.

References

[1] I. H. Choi, C. H. Jeong, and Y. G. Kim, "Tracking a driver's face against extreme head poses and inference of drowsiness using a gidden markov model," *Applied Sciences MDPI*, vol. 6, no. 137, pp. 1–14, 2016.

[2] N. A. Shnain, Z. M. Hussain, and S. F. Lu, "A feature-based structural measure: an image similarity measure for face recognition," *Applied Sciences MDPI*, vol. 7, no. 786, pp. 1–17, 2017.

[3] J. M. Rodrigues, W. Puech, P. Meuel, J. C. Bajard, and M. Chaumont, "Face protection by fast selective encryption in a video," in *Proceedings of the IET Conference on Crime and Security*, pp. 420–425, London, UK, June 2006.

[4] O. A. Khashan, A. M. Zin, and E. A. Sundararajan, "Performance study of selective encryption in comparison to full encryption for still visual images," *Journal of Zhejiang University Science C*, vol. 15, no. 6, pp. 435–444, 2014.

[5] H. R. Shakir, L. E. George, and G. K. Tuma, "Partial encryption for colored images based on face detection," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, no. 8, pp. 25–35, 2015.

[6] D. Rachmawati, M. A. Budiman, and M. I. Wardhono, "Hybrid cryptosystem for image security by using Hill cipher 4x4 and ElGamal elliptic curve algoritm," in *Proceedings of the IEEE International Conference on Communication, Networks and Satellite (comnetsat)*, pp. 49–54, Medan, Indonesia, November 2018.

[7] J. C. Vela Medina, A. E. Guerrero Sánchez, J. E. Rivas Araiza, and E. A. Rivas Araiza, "Face detection for efficient video surveillance IoT based embedded system," in *Proceedings of the 2018 IEEE International Conference on Automation/XXIII Congress of the Chilean Association of Automatic Control (ICA-ACCA)*, pp. 1–6, Chile, South America, October 2018.

[8] K. Hong and K. Jung, "Partial encryption of digital contents using face detection algorithm," *Guilin*, vol. 4099, pp. 632–640, 2006.

[9] C. Gerhardt, P. Aichroth, and S. Mann, "Selective face encryption in H.264 encoded videos," in *Proceedings of the 2017 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, St. Petersburg, FL, USA, December 2017.

[10] Q. Jiang, W. Zeng, W. Ou, and R. Xu, "A scrambling and encryption algorithm for selective block of identification photo," in *Proceedings of the 2016 8th International Conference on Wireless Communications & Signal Processing (WCSP)*, pp. 1–5, Yangzhou, China, October 2016.

[11] K. Soumya, R. Kumar, and K. Raju, "Selective area encryption using machine learning Technique2," *Innovations in Power and Advanced Computing Technologies*, vol. 1, pp. 1–7, 2019.

[12] P. Duong-Ngoc, T. N. Tan, and H. Lee, "Efficient NewHope cryptography based facial security system on a GPU," *IEEE Access*, vol. 8, pp. 108158–108168, 2020.

[13] A. Sardar, S. Umer, C. Pero, and M. Nappi, "A novel cancelable FaceHashing technique based on non-invertible transformation with encryption and decryption template," *IEEE Access*, vol. 8, pp. 105263–105277, 2020.

[14] M. Asgari-Chenaghlu, M. R. Feizi-Derakhshi, N. Nikzad-Khasmakhi et al., "Chaotic yolo for user intended image encryption and sharing in social media," *Information Sciences*, vol. 542, pp. 212–227, 2020.

[15] T. Paraskevi, N. Klimis, and K. Stefanos, "Security of human video objects by incorporating a chaos-based feedback cryptographic scheme," in *Proceedings of the 12th ACM International Conference on Multimedia*, pp. 1–5, New York, NY, USA, October 2004.

[16] W. Wen, Y. Zhang, Z. Fang, and J.-x. Chen, "Infrared target-based selective encryption by chaotic maps," *Optics Communications*, vol. 341, pp. 131–139, 2015.

[17] T. Xiang, K. Wong, and X. Liao, "Selective image encryption using a spatiotemporal chaotic system," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 17, no. 2, 2007.

[18] K. Prabhavathi, C. P Sathisha, and K. M Ravikumar, "Region of interest based selective medical image encryption using multi chaotic system," in *Proceedings of the 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, pp. 1–5, Mysuru, India, December 2017.

[19] M. Amin, O. S. Faragallah, and A. A. Abd El-Latif, "A chaotic block cipher algorithm for image cryptosystems," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 11, pp. 3484–3497, 2010.

[20] L. Teng, X. Wang, and J. Meng, "A chaotic color image encryption using integrated bit-level permutation," *Multimedia Tools and Applications*, vol. 77, no. 6, pp. 6883–6896, 2018.

[21] M. Kumar, P. Powduri, and A. Reddy, "An RGB image encryption using diffusion process associated with chaotic map," *Journal of Information Security and Applications*, vol. 15, pp. 20–30, 2014.

[22] O. Flores, J. C. Estrada, C. E. Padilla, J. Aguilar, and M. Jiménez, "System to safeguard the identity of persons in photographs through cryptography and steganography techniques using chaos," *Security and Communication Networks*, vol. 2018, pp. 1–16, Article ID 4853134, 2018.

[23] A. M. Ayoup, A. H. Hussein, and M. A. Attia, "Efficient selective image encryption using pseudo random number sequences and AES technique," *Multimedia Tools and Applications*, vol. 12, no. 1, pp. 187–198, 2015.

[24] K. M. Ibrahim, R. K. Jamal, and F. H. Ali, "Chaotic behaviour of the Rossler model and its analysis by using bifurcations of limit cycles and chaotic attractors," *Journal of Physics: Conference Series*, vol. 1008, pp. 2–7, 2018.

[25] Z. Congxu, "A novel image encryption scheme based on improved hyperchaotic sequences," *Optics Communications*, vol. 285, no. 1, pp. 29–37, 2011.

[26] B. Stoyanov and K. Kordov, "Image encryption using Chebyshev map and rotation equation," *Entropy*, vol. 17, no. 4, pp. 2117–2139, 2015.

[27] C. Pak and L. Huang, "A new color image encryption using combination of the 1D chaotic map," *Signal Processing*, vol. 138, pp. 129–137, 2017.

[28] N. Chidambaram, P. Raj, K. Thenmozhi, and R. Amirtharajan, "Advanced framework for highly secure and cloud-based storage of colour images," *ITM image processing*, vol. 14, no. 13, pp. 1–12, 2020.

[29] R. Sivaraman, S. Rajagopalan, J. B. Balaguru, and R. Amirtharajan, "Ring oscillator as confusion – diffusion agent: a complete TRNG drove image security," *ITM Image Processing*, vol. 14, no. 13, pp. 1–10, 2020.

[30] M. Kumar, A. Iqbal, and P. Kumar, "A new rgb image encryption algorithm based on dna encoding and elliptic curve Diffie-Hellman cryptography," *Signal Processing*, vol. 125, pp. 187–202, 2016.

[31] M. A. Murillo, C. Cruz, F. Abundiz, R. M. López, and O. R. Acosta, "A rgb image encryption algorithm based on total plain image characteristics and chaos," *Signal Processing*, vol. 109, pp. 119–131, 2014.

[32] P. Praveenkumar, R. Amirtharajan, K. Thenmozhi, J. B. B. Rayappan, and J. B. Balaguru, "Triple chaotic image scrambling on RGB - a random image encryption approach," *Security and Communication Networks*, vol. 8, no. 18, pp. 3335–3345, 2015.

[33] M. Khan and H. M. Waseem, "A novel image encryption scheme based on quantum dynamical spinning and rotations," *PLoS One*, vol. 13, pp. 1–23, 2018.

[34] C. Lakshmi, K. Rayappan, and R. Amirtharajan, "Hopfield attractor-trusted neural network: an attack-resistant image encryption," *Neural Computing and Applications*, vol. 32, no. 15, pp. 11477–11489, 2019.

[35] M. A. F. Al-Husainy and H. A. A. Al-Sewadi, "Implementing binary search tree concept for image cryptography," *International Journal of Advanced Science and Technology*, vol. 130, pp. 21–32, 2019.

[36] F.-P. An and J.-e. Liu, "Image encryption algorithm based on adaptive wavelet chaos," *Journal of Sensors*, vol. 2019, pp. 1–12, Article ID 2768121, 2019.

[37] A.-V. Diaconu and K. Loukhaoukha, "An improved secure image encryption algorithm based on rubik's cube principle and digital chaotic cipher," *Mathematical Problems in Engineering*, vol. 2013, pp. 1–10, Article ID 848392, 2013.

[38] G. Bhatnagar and Q. M. J. Wu, "Enhancing the transmission security of biometric images using chaotic encryption," *Multimedia Systems*, vol. 20, pp. 203–214, 2014.

[39] N. Chidambaram, P. Raj, K. Thenmozhi, S. Rajagopalan, and R. Amirtharajan, "A cloud compatible DNA coded security solution for multimedia file sharing & storage," *Multimedia Tools and Applications*, vol. 78, pp. 22837–22863, 2019.