OPTIMAL-PICTURE-BASED INFORMATION-HIDING WITH ONE-DIMENSIONAL CHAOTIC SYSTEMS AND DYNAMIC PROGRAMMING

***M.Kannan*[1] *R.Sundar*[2]**

ABSTRACT:

Information-hiding is an innovation focused on the protected hiding of basic realities into ordinal structures, for example, Pictures, recordings and documents. Most definitely, we utilize dim scale Pictures as the media for information-hiding and develop another component for the protected whacking of a category of twofold bits into a dim scale copy. The implanting is done in two phases. Toward the start, the proposed component divides the arrangement into various aftereffects of equivalent length, the parallel bits in the grouping are then rearranged and encoded with a lot of whole number keys and an arrangement of one-dimensional strategic mappings. In the after-arrangement stage, the calculation isolates the rearranged and encoded succession into areas of equivalent length, and successively installs the districts into the dark estimations of chose image dots in the given Picture. The image dots for implanting can be effectively chosen with a powerful programming approach that limits the distinction between a spread Picture and the comparing stego Picture. To implant an area into the comparing image dot, the calculation replaces the least noteworthy bits of the dim estimation of the image dot with the bits in the district. Our examination and tests with both common and clinical Pictures show that this methodology can safely shroud information into dark scale Pictures without creating a lot of progress in Picture content and is accordingly possibly helpful for secure Picture-based information-hiding. A usage of the program in MATLAB is openly accessible upon demand.

Keywords: *information-hiding, gray-scale Pictures, one-dimensional logistic systems, optimization; dynamic programming, key space.*

1. INTRODUCTION

In the past two decades, a huge measure of advanced information was produced alongside the huge turn of events and accomplishments in information innovation. The security of the protected innovation rights related with some significant computerized information has in this manner become a significant issue in information science. Various strategies and methods have been created to determine this issue, including information-hiding innovation [1]. In particular, information-hiding innovation conceals significant information and information inside advanced records. For example, pictures, records and the after-arrangement reports are required to have high closeness with the first ones. A record where the hiding is performed is a spread media and the one that contains the shrouded information is a stego media. All in all, natural eyes can't perceive the contrast between a spread media and the relating stego media.

Since the transmission and capacity of mixed media information are regularly founded on Pictures, inserting information into dark scale Pictures has become a significant methodology for information-hiding. Analysts have built up an enormous number of techniques that can conceal information into dim scale Pictures, for example, the side match innovation [2], the hiding innovation dependent on image dot distinction development and modulus work [3], and the information-hiding innovation dependent on image dot esteem contrast and LSB substitution [4]. The vast majority of these techniques select various neighboring

[1] Assistant Professor, Department of Computer Science
Karpagam Academy of Higher Education, Coimbatore.

[2] Assistant Professor, Department of EEE
Karpagam Academy of Higher Education, Coimbatore.

image dots in the Picture dependent on the distinction in dim qualities. The dark estimations of these neighboring image dots are changed to contain the information that should be inserted. In [5], comparable example squares are utilized to perform Picture-based information-hiding. A standard Picture is scanned for an example square and the example square is then utilized as the reference for the installing of information.

Various techniques have been created to improve the presentation of information-hiding. For instance, a staggered hiding system is created in [6] to accomplish bigger hiding limit while keeping up the high comparability between a stego Picture and its spread Picture. In [7], an information-hiding calculation is created that is dependent on the most extreme histogram hole of Picture squares. In [8], a histogram-moving strategy is utilized to implant information into the quantization levels of the square truncation coding packed codes of an Picture. In [9], relationships among forecast mistakes are used two by two to accomplish improved execution for information-hiding. In [10], another turtle-shell-based information-hiding calculation is created to improve the limit of implanting while at the same time keeping up great Picture quality. In [11], a reversible information-hiding calculation is created that is dependent on square truncation coding. In [12], the corruption of clinical Pictures after a recurrence-space-based hiding calculation conceals information inside them, which is examined. In [13], two novel varieties of the old-style histogram move strategies are created to additionally improve the limit of implanting in clinical Pictures. As of late, another information-hiding calculation is created in [14] to join the Chinese Remainder Theorem and another extraction capacity to broaden the limit of inserting.

A large portion of the current techniques can effectively conceal information into a dark scale Picture while keeping up a high similitude between a spread Picture and the relating stego Picture. The greater part of these techniques can't completely use the image dots in a Picture to upgrade the similitude between a stego Picture and its spread Picture, which may not be attractive for specific applications. What is more, the information installed into an Picture can frequently be legitimately acquired by handling a stego Picture or contrasting a stego Picture and its spread Picture, the inserted information is along these lines not secure when the stego Picture or both the spread Picture and its stego Picture should be transmitted through the web.

As of late, tumultuous frameworks have been utilized in countless calculations to scramble Pictures [15-21]. For instance, in [22], a calculation that can encode shading Pictures by rearranging image dots is created. A rearranging of the image dots in a shaded Picture is produced depending on disorderly frameworks. In [23], a calculation is created to encode a Picture by consolidating stage and dispersion. In [24], a high-dimensional clamorous framework is utilized to create three arrangements of pseudorandom groupings and a Picture can be scrambled depending on these pseudorandom successions. Our past work [25] utilizes two arrangements of one-dimensional calculated frameworks to produce a hearty encryption of a Picture. An inquiry consequently quickly emerges as to whether riotous frameworks can be applied to improve the security of an information-hiding calculation.

In this paper, we utilize various number keys and an arrangement of one-dimensional calculated mappings to improve the security of Picture-based information-hiding and build up another Picture-based information-hiding calculation that can conceal a grouping of double bits into a dark scale Picture with limited contrast between a spread Picture and the relating stego Picture. The calculation plays out the hiding in two phases. In the primary stage, it isolates the grouping into aftereffects of equivalent length. The area of a piece in the arrangement is spoken to by a couple of numbers, where one of them is compared to the aftereffect where the bit is found, and the other is the general situation of the bit in the aftereffect. All bits in the arrangement are in this way mapped to a lattice in two-dimensional space and the bits are rearranged and encoded depending on various number keys and an arrangement of one-dimensional strategic mappings. In particular, each line and section in the framework is related with a positive number key. The movement of bits is performed column by line first. A piece in succession of the two-dimensional matrix is migrated depending on the result of the whole number key of the column and the present situation of the bit in the line. After a line-based movement is finished, bits are gathered in segments and the bits in every section are migrated depending on the whole number key of the segment.

In the second phase of the calculation, it isolates the rearranged and encoded grouping into locales of equivalent length, and the hiding of the bits is finished by successively implanting the areas into the dim scale Picture. To get a stego Picture that has the most noteworthy similitude to the spread Picture, a measure that assesses the contrast between a spread Picture and its stego Picture is created, and the areas for inserting can be controlled by a unique programming calculation that can limit the measure. The installing of a locale is performed by subbing the least critical bits of the dim estimation of the image dot in the decided area with the bits in the district.

The keys for recovering the entrenched arrangement include the integer keys associated with rows and columns of the grid for shuffling, the one-dimensional logistic mappings for encoding and the integers needed to determine the locations of embedding. This algorithm is able to recover the hidden information without the cover Picture or a standard Picture. A simple analysis shows that the algorithm has a key space of large size and the hidden information is thus secure against attacks based on exhaustive search. Our experiments show that this algorithm can generate excellent results when medical Pictures are used for information-hiding. In addition, a comparison with two other existing methods for Picture-based information-hiding shows that our algorithm can generate stego Pictures with higher similarity values to the corresponding cover Pictures.

2. MATERIALS AND METHODS

2.1. The Shuffling of the Arrangement

Let S be an arrangement of binary bits that need to be hidden into a given gray scale Picture. L_S denotes the length of S and $S(i)$ is the i th bit in S. Based on a positive integer c such that $c < L_S$ and $r = \lfloor L_S / c \rfloor + 1$, S can be sequentially divided into r sub-arrangements such that at least $r-1$ of these sub-arrangements are of equal length and each of them contains c bits. The number of bits in the last sub-arrangement could be less than c. If it is the case, a number of bits of 0 can be padded to the end of S such that the length of the last sub-arrangement is also c. We therefore assume that S is sequentially divided into r sub-arrangements of length c in the rest of the paper.

Based on the sub-arrangements in S, the position of $S(i)$ can be described by a pair of integers (u, v) where $u = \lfloor i / c \rfloor$ and $v = i \bmod c$. u is the *row number* of $S(i)$ and v is its *column number*. Each bit in S can thus be uniquely mapped to a point in a two-dimensional space. All bits with the same row number form a *row*. Similarly, all bits with the same column number form a *column*. It is thus clear that all bits with the same row number are in the same row, and those with the same column number are in the same column. The rows formed by bits in S are numbered by integers $1,2 \ldots, r$ and columns can be numbered by $1,2,\ldots,c$.

We then associate each row m, where $1 \leq m \leq r$, with a positive integer k_m such that the greatest common divisor of c and k_m is 1. Similarly, each column n, where $1 \leq n \leq c$, is associated with a positive integer l_n such that the greatest common divisor of l_n and r is 1. As the first step of shuffling, each bit is relocated within its row. Specifically, the bit mapped to an integer pair (s,t) is relocated to $(s, g(t,k_s))$, where $g(t,k_s)$ is computed as follows.

$$g(t,k_s) = (k \times t) \bmod c \ldots (0.1)$$

It is straightforward to see that for two different arbitrary integers t_1 and t_2 such that $0 \leq t_1 < c$ and $0 \leq t_2 < c$,

$g(t_1.k_s)$ and $g(t_2,k_s)$ are also different. Indeed, if it is not the case, there exist two different integers t_1 and t_2 such that $0 \le t_1 < c$ and $0 \le t_2 < c$, and $g(t_1,k_s) = g(t_2,k_s)$. From equation 2.1, the following equation must hold for t_1 and t_2.

$k_s(t_1 - t_2) = cp$ … (0.2)

where p is an integer. However, since the greatest common divisor of c and k_s is 1, equation 2.2 implies that $t_1 - t_2$ must be divisible by c, which is contradictory to the fact that $| t_1 - t_2 | < c$. $g(t_1.k_s)$ and $g(t_2,k_s)$ thus must be different. This fact ensures that no two bits are relocated to the same position and no conflicts would occur in the first step of the designed relocation procedure. In the second step of the shuffling, relocations are performed in columns. Each bit is relocated within its column. Specifically, the bit that is currently in row x and column y is relocated to $(h(x,l_y), y)$, where $h(x,l_y)$ is computed as follows.

$h(x,l_y) = (x \times l_y) \bmod r$ … (0.3)

Similarly, based on the same argument that has been shown above, no two bits are relocated to the same position and no conflicts would occur in the second step of the shuffling process. In the final step of the shuffling process, each bit is mapped from its position in the two-dimensional space back to the corresponding position in the arrangement. Specifically, the bit currently in position (u,v) in the two-dimensional space is mapped back to position $uc + v$ in the arrangement.

2.2. The Encoding of the Shuffled Arrangement

We use R to denote the shuffled arrangement generated from the original arrangement S by the shuffling process described in Subsection 2.1. The length of R is also L_S. Based on a given positive integer parameter b, where $1 \le b \le 8$, R can be sequentially divided into regions of length b. Here, we assume L_S is divisible by b. Since the bits in each region form the binary encoding of an integer between 0 and $2^b - 1$, we use a arrangement of integers $w_1, w_2, \dots, w_a$ to represent R, where $a = L_s / b$.

Based on a one dimensional *logistic mapping* G_μ, a arrangement of real numbers $z_0, z_1, \dots, z_k, z_{k+1}, \dots$, where $k \ge 0$, can be generated based on the following recursion.

$z_{k+1} = \mu z_k (1 - z_k)$ …(0.4)

where μ is a parameter of the recursion and must satisfy $0 < \mu < 4$. It is clear that $0 < z_k < 1$ holds for each $k \ge 0$ if the initial value z_0 of the arrangement is a real number between 0 and 1. In other words, all numbers in the arrangement are positive and less than 1 when the initial value of the arrangement is a real number between 0 and 1.

It is well known that a logistic mapping has the property of a one-dimensional chaotic system. Specifically, when $3.57 \le \mu < 4$, the period of the numbers in the arrangement becomes infinitely large and the numbers in the arrangement are sensitive to the initial value z_0 [15]. When k is large enough, a significantly different z_k would arise from a perturbation of a tiny amount in z_0. The initial value z_0 can thus be used as a key to generate encoded data from a one-dimensional logistic mapping. In fact, many existing Picture encryption algorithms utilize the chaotic property of one-dimensional logistic mappings to encrypt Pictures.

To perform the encoding of R, a system of d one-dimensional logistic mappings $G_{\mu_1}, G_{\mu_2}, \dots, G_{\mu_d}$ is used with different parameters $\mu_1, \mu_2, \dots, \mu_d$. Each logistic mapping is assigned a different initial value and a arrangement of real numbers can be generated for each logistic mapping. We use $z_{1,0}, z_{2,0}, \dots, z_{d,0}$ to denote the initial values for $G_{\mu_1}, G_{\mu_2}, \dots, G_{\mu_d}$ respectively.

We then select a large enough integer k_0 such that the chaotic behavior of a one-dimensional logistic mapping starts to be significant when more than k_0 numbers have been generated in the arrangement. To encode w_i $(1 \le i \le a)$ in R, we need to generate d integers $\lambda_{i,1}, \lambda_{i,2}, \dots, \lambda_{i,d}$ as follows.

$$\lambda_{i,j} = (z_{j,i+k_0} \times M) \bmod 2^b \qquad (0.5)$$

where $1 \le j \le d$, M is a large integer and $z_{j,i+k_0}$ is the

$i+k_0$ th number generated with G_{μ_j} and an initial value of $z_{j,0}$.

From $\lambda_{i,1},\lambda_{i,2},\ldots,\lambda_{i,d}$, w_i can be encoded based on the following recursive relation.

$$\kappa_{i,q+1}=\kappa_{i,q}\oplus\lambda_{i,q+1} \qquad (0.6)$$

where $0\le q\le d-1$, $\kappa_{i,0}=w_i$ and $\kappa_{i,d}$ is the encoded result for w_i.

After all of the integers $w_1,w_2,\ldots,w_a$ have been encoded, the binary forms of $\kappa_{1,d},\kappa_{2,d},\ldots,\kappa_{i,d}$ are sequentially combined into a arrangement of binary bits. The resulting arrangement is the shuffled and encoded arrangement of S.

2.3. The embedding of the Shuffled and Encoded Arrangement

We use B_2 to denote the shuffled and encoded arrangement obtained with the approaches in Subsections 2.1 and 2.2. Let I be the gray scale Picture where B_2 needs to be entrenched. There are m rows and n columns in I and $I(g_p,h_p)$ is the gray value for the image dot in row g_p and column h_p in I. In the second stage, the bits in B_2 are sequentially entrenched into I.

The algorithm embeds the bits in B_2 into I in groups and the numbers of bits contained in all groups are equal. We use w_d to denote the number of bits in each group. Since the gray value of a image dot contains 8 bits, the inequality $1\le w_d\le 8$ must hold for w_d. The bits in B_2 are thus sequentially divided into $\beta=\lfloor cr/w_d\rfloor+1$ sub-arrangements, the bits in each sub-arrangement form a group and each sub-arrangement contains w_d bits. We use integers $1,2,\ldots,\beta$ to sequentially number the sub-arrangements. Let e and f be positive integers that satisfy $e<\lfloor mn/5\beta\rfloor$ and $f\le mn-be$, the bits in sub-arrangement $l\,(1\le l\le\beta)$ is entrenched into the gray value of image dot (g_l,h_l) in I, where g_l and h_l are computed with equations 2.7 and 2.8. We show later that the values of e and f can be determined by a dynamic programming algorithm.

$$g_l=\lfloor (f+(l-1)e)/n\rfloor+1 \qquad (0.7)$$

$$h_l=(f+(l-1)e)\bmod n \qquad (0.8)$$

The gray value of image dot (g_l,h_l) is updated to include the w bits in sub-arrangement l as follows.

$$I_2(g_l,h_l)=\lfloor I(g_l,h_l)/2^{w_d}\rfloor+C_l \qquad (0.9)$$

where $I_2(g_l,h_l)$ is the gray value of image dot (g_l,h_l) after the embedding is performed and C_l is the value represented by the w_d bits in sub-arrangement l. It is clear from equation 2.9 that the algorithm replaces the w_d least significant bits in the gray value of image dot (g_l,h_l) by the bits in sub-arrangement l to complete the embedding.

We then describe the algorithm that can determine the values of e and f. In principle, e and f should be selected to minimize the difference between the stego Picture and the cover Picture. The difference between the cover Picture I and the stego Picture I_2 can be evaluated by $D(I,I_2)$ as defined in equation (10). In practice, we can choose a reasonable value for e and determine the value of f that can minimize $D(I,I_2)$.

$$D(I,I_2)=\sum_{i=1}^{m}\sum_{j=1}^{n}|I(i,j)-I_2(i,j)|$$

(0.10)

Given a fixed e, The value of f can be determined by minimizing $D(I,I_2)$. The minimization can be performed with a dynamic programming approach as follows.

A two dimensional table $T(o,s)$ is maintained to store the difference between I and I_2 in the case where the value of f is equal to o and sub-arrangements from 1 to s have been entrenched into I. It is straightforward to see that $T(o,s+1)$ can be computed based on the recursive relation shown in equations 2.11 and 2.12.

$$D(s+1)=|I(g_{s+1},h_{s+1})-I_2(g_{s+1},h_{s+1})|$$

(0.11)

$$T(o,s+1)=T(o,s)+D(s+1) \qquad (0.12)$$

where g_{s+1} and h_{s+1} can be computed as follows.

$$g_{s+1}=\lfloor (f+so)/n \rfloor+1 \qquad (0.13)$$

$$h_{s+1}=(f+so)\bmod n \qquad (0.14)$$

The value of $T(o,0)$ for any o that satisfies $1<o\le mn-be$ is set to be zero. The value of f that can minimize $D(I,I_2)$ is determined by equation 2.15.

$$f=\underset{1\le o\le mn-be}{\arg\min}\{T(o,b)\} \qquad (0.15)$$

2.4. The salvage of the entrenched arrangement

Let I_2 be the resulting stego Picture after the embedding of B_2 into I is complete. Given the values of c,r,w_d,e,f, the initial values and parameters of the different one - dimensional logistic mappings for encoding, and the integer keys used in the shuffling of the original arrangement S, S can be recovered from I_2 in two stages. In the first stage, a set of image dots whose gray values contain the bits in arrangement B_2 can be efficiently determined. The w_d least significant bits of the gray values of these image dots can then be extracted and combined sequentially to reconstruct B_2. Specifically, for each integer l where $1\le l\le \beta$, the binary value encoded by the w_d bits in sub-arrangement l in B_2 can be computed with equation 2.1

$$C_l=I_2(g_l,h_l)\bmod 2^{w_d} \qquad (0.16)$$

where g_l and h_l are computed with equations 2.7 and 2.8. C_l is the value represented by the w_d bits in sub-arrangement l in B_2. Sub-arrangement l in B_2 can then be determined from C_l.

In the second stage, the original arrangement S can be reconstructed from B_2. Let $\mu_1,\mu_2,...,\mu_d$ be the initial values of the logistic mappings $G_{\mu_1},G_{\mu_2},...,G_{\mu_d}$ used for the encoding of the shuffled arrangement R, and $z_{1,0},z_{2,0},...,z_{d,0}$ be their initial values. Since R can be represented by a arrangement of integers $w_1,w_2,...,w_a$, where each integer is represented by b binary bits and the encoding of R is performed by sequentially encoding the integers in the arrangement, the salvage of R from B_2 can also be performed sequentially such that each individual integer in R can be recovered first and the recovered integers are then combined to reconstruct R. We thus sequentially divide B_2 into a regions of length b and use $\delta_1,\delta_2,...,\delta_a$ to denote the resulting arrangement of integers.

To recover w_i ($1\le i\le a$), we first compute d integers $\lambda_{i,1},\lambda_{i,2},...,\lambda_{i,d}$ from $G_{\mu_1},G_{\mu_2},...,G_{\mu_d}$ by equation (5). Based on $\lambda_{i,1},\lambda_{i,2},...,\lambda_{i,d}$, w_i can be recovered from δ_i by the following recursive relation.

$$\phi_{i,q+1}=\phi_{i,q}\oplus\lambda_{i,d-q} \qquad (0.17)$$

where $0\le q\le d-1$, $\phi_{i,0}=\delta_i$ and $\phi_{i,d}$ is the result of decoding. In other words, $w_i=\phi_{i,d}$.

After all of the integers $w_1,w_2,...,w_a$ have been recovered, R can be reconstructed by sequentially combining the binary forms of $w_1,w_2,...,w_a$ into a single arrangement.

The last step of salvage is to relocate the bits in R to recover the original arrangement S. The salvage of S is based on the integer keys associated with all columns and rows in the two-dimensional grid where the bits are mapped to during the shuffling process. We use $k_1,k_2,...,k_r$ to denote the integer keys associated with the rows and $l_1,l_2,...,l_c$ are the integer keys for the columns.

Bits in R are first mapped to a two-dimensional grid with r rows and c columns. Specifically, the i th bit in R is mapped to the point with a row number of $\lfloor i/c\rfloor$ and a column number of $i \bmod c$. The bits are then relocated by columns after the mapping is completed. Each bit is relocated within its column. Specifically, the bit currently in row $h(x,l_y)$ and column y is relocated to (x,y), where $h(x,l_y)$ is computed with equation (3). To complete the relocation for all bits in column y, for each x that satisfies $1\le x\le r$, the value of $h(x,l_y)$ is calculated and the bit currently in location $(h(x,l_y),y)$ is relocated to (x,y).

The bits are then relocated by rows. Each bit is relocated within its row. Specifically, the bit currently in $(s, g(t, k_s))$ is relocated to (s, t), where $g(t, k_s)$ is computed with equation 2.1. To complete the relocation for all bits in row s, for each t that satisfies $1 \le t \le c$, the value of $g(t, k_s)$ is computed and the bit currently in $(s, g(t, k_s))$ is relocated to (s, t). The resulting arrangement is the original arrangement S. It is clear that the salvage of the entrenched information can be performed without the cover Picture or a standard Picture.

3. EXPERIMENTAL RESULTS AND DISCUSSIONS

3.1. On benchmark and medical Pictures

This information-hiding algorithm has been implemented into a computer program in MATLAB, and its performance is tested by hiding information into four-benchmark ordinary gray-scale Pictures and two medical Pictures. One of the ordinary benchmark Pictures is the well-known benchmark Picture Lena. The other three of the ordinary benchmark Pictures are selected from the Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500), and the dataset can be accessed and downloaded for free from the website.

https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html. The two medical Pictures are both obtained from the database of cancer-imaging archive (TCIA) public access, which can be downloaded and used for free at https://wiki.cancerimagingarchive.net/display/Public/Brain-Tumor-Progression.

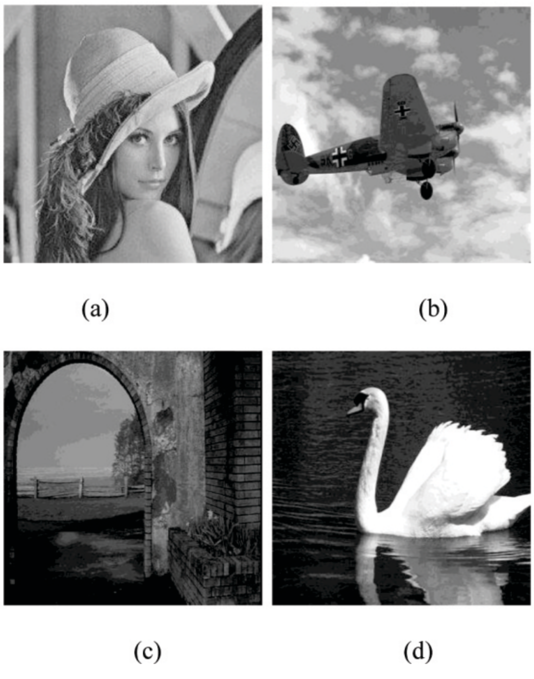

Figure 1. (a) Picture Lena; (b) Picture 3063 from BSD500; (c) Picture 5096 from BSD500; (d) Picture 8068 from BSD500.

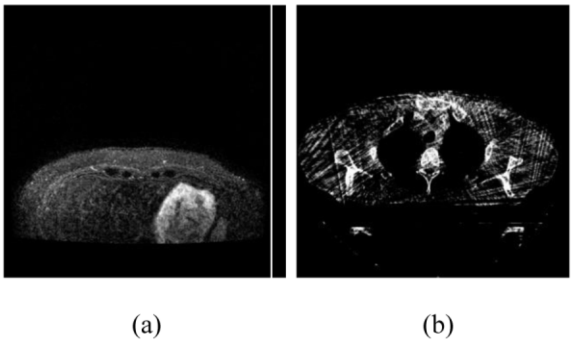

Figure 2. (a) Med1; (b) Med2. Both medical Pictures are downloaded from the cancer imaging archive (TCIA) public access.

The parameters for encoding of the shuffled arrangements are real numbers between 3.7 and 3.9. These parameters can be computed with a linear function. The initial values of the one-dimensional logistic mappings are randomly generated numbers between 0 and 1 and M is set to be 10^6. The four testing ordinary benchmark Pictures are shown in Figure 1 (a) (b) (c) and (d) and the two testing medical Pictures are shown in Figure 2 (a) and (b). All testing Pictures are scaled to the same size, which is 100×100.

We use two measures to evaluate the difference between a cover Picture and the corresponding stego Picture. One of them is the Square Sum of the Difference (SSD) in the gray values of the corresponding image dots in both Pictures. Given a cover Picture I_c and its corresponding stego Picture I_s, both Pictures contain m rows and n columns. The SSD between I_c and I_s can be computed as follows.

$$SSD(I_c, I_s) = \sum_{i=1}^{m} \sum_{j=1}^{n} | I_c(i, j) - I_s(i, j) |^2 \qquad (0.18)$$

It is clear that a higher value of SSD suggests a larger amount of difference between the cover Picture and the corresponding stego one. Another measure is the Peak Signal of Noise Ratio (PSNR) defined between the cover Picture and the corresponding stego Picture. In the case where both I_c and I_s contain m rows and n columns, the PSNR between I_c and I_s can be computed as follows.

$$PSNR(I_c, I_s) = 20\log_{10}(\frac{255\sqrt{mn}}{\sqrt{SSD(I_c, I_s)}}) \qquad (0.19)$$

It can be seen from equation 3.2 that a higher value of $PSNR(I_c, I_s)$ suggests a higher similarity between I_c and I_s. In the case where I_c and I_s are identical, we use $SSD(I_c, I_s)$ as the only measure to represent the difference of I_c and I_s.

We first test the performance of this approach when different values of w_d are used for embedding. The tested values of w_d include 1 and 5. For each given value of w_d, 100 arrangements of binary bits are randomly generated and hidden into each testing Picture. Each arrangement contains 300 binary bits. Tables 1 and 2 show the means and standard deviations of the SSDs and PSNRs between each testing Picture and its stego Picture when different values of w_d are used for embedding.

Table 1. The means and standard deviations (STD) of SSDs and PSNRs obtained with our algorithm on the testing Pictures when w_d is 1.

Testing Picture	SSD		PSNR	
	Mean	STD	Mean	STD
Lena	46.60	4.64	71.47	0.42
3063	51.20	6.38	71.07	0.57
5096	51.20	3.68	71.05	0.31
8068	52.50	4.04	70.94	0.30
Med1	0.00	0.00	—	—
Med2	0.00	0.00	—	—

Table 2. The means and standard deviations (STD) of SSDs and PSNRs obtained with our algorithm on the testing Pictures when w_d is 5.

Testing Picture	SSD		PSNR	
	Mean	STD	Mean	STD
Lena	563.50	109.76	58.12	2.85
3063	478.60	110.05	58.20	2.23
5096	526.30	127.58	58.54	2.15
8068	300.60	101.86	56.17	1.54
Med1	0.00	0.00	—	—
Med2	0.00	0.00	—	—

It can be seen from Tables 1 and 2 that the similarity between a cover Picture and the corresponding stego Picture becomes lower when the word width w_d for embedding increases, which is as expected. In addition, the performance becomes less stable when w_d increases. From Table 2, we can observe that the PSNR values deteriorate significantly when w_d increases from 1 to 5, which suggests that a word width less than 5 should be selected for our algorithm to achieve satisfactory results in practice. However, an unexpected result is that the hiding of all arrangements within the two medical Pictures can be performed without generating any change in cover Pictures, which may suggest that our algorithm should be used with medical Pictures to achieve the most satisfactory results of hiding in practice.

3.2. A comparison with other methods

The overall performance of this algorithm is compared with that of two other existing methods, including the methods developed in [1] and [3]. Specifically, randomly-generated binary arrangements are hidden into all ordinary benchmark Pictures and both medical Pictures with our algorithm, the methods developed in [1] and [3], and the performance of all three methods is evaluated based on the resulting stego Pictures.

We select two different values for arrangement length, including 200 and 600. For each given arrangement length, 100 arrangements of binary bits are randomly generated. The value of w_d is chosen to be 1 for our approach. Tables 3 and 4 compare the means and standard deviations of the SSDs and PSNRs of the stego Pictures obtained with our algorithm and the other two methods on each given arrangement length.

Table 3. The means and standard deviations (STD) of SSDs and PSNRs obtained with three methods on the testing Pictures when the length of the arrangement is 200.

Method		8068	Lena	5096	3063	Med1	Med2
Our method	PSNR	**73.83**	**73.42**	**73.45**	**73.30**	—	—
	STD	0.45	0.62	0.42	0.36	—	—
	SSD	**27.00**	**29.80**	**29.50**	**30.50**	**0.00**	**0.00**
	STD	2.75	4.32	3.38	3.12	0.00	0.00
Method in [1]	PSNR	65.00	67.38	61.70	70.44	—	—
	STD	0.98	2.50	1.53	0.45	—	—
	SSD	210.10	136.10	467.40	59.10	0.00	1.00
	STD	41.08	70.49	130.32	6.25	0.00	0.94

Method in [3]	PSNR	62.99	63.31	60.96	62.60	57.50	57.44
	STD	0.81	0.89	1.16	1.12	0.63	0.47
	SSD	331.40	309.10	539.00	368.00	1167.60	1179.90

Table 4. The means and standard deviations (STD) of SSDs and PSNRs obtained with three methods on the testing Pictures when the length of the arrangement is 600.

Method		8068	Lena	5096	3063	Med1	Med2
Our method	PSNR	**67.69**	**67.45**	**67.54**	**67.40**	—	—
	STD	0.18	0.29	0.16	0.34	—	—
	SSD	**110.80**	**117.10**	**114.70**	**118.60**	**0.00**	**0.00**
	STD	4.71	7.77	4.10	9.59	0.00	0.00
Method in [1]	PSNR	60.98	61.45	56.70	66.20	—	—
	STD	0.38	1.06	1.18	0.22	—	—
	SSD	520.70	477.40	1434.80	156.00	0.00	2.80
	STD	45.85	104.58	143.73	8.29	0.00	1.25
Method in [3]	PSNR	58.19	57.94	56.71	57.83	52.59	52.74
	STD	0.57	0.69	0.55	0.55	0.29	0.30
	SSD	994.50	1057.50	1397.90	1079.50	3592.60	3470.00
	STD	104.76	143.13	142.07	124.11	164.36	147.84

It is clear from Tables 3 and 4 that our approach can achieve perfect hiding on both Med1 and Med2 for all values of arrangement length. The mean values of PSNRs achieved by our approach are significantly higher than those of the other two methods on the four ordinary benchmark Pictures for all arrangement lengths. This suggests that our approach consistently outperforms the other two methods on all testing Pictures. In addition, the standard deviations of SSDs and PSNRs confirm that the performance of our approach is reliable and robust. Tables 12-16 also show that the method developed in [1] can achieve a perfect embedding of all generated arrangements on Med1 and a near perfect embedding of all generated arrangements on Med2.

3.3. The overall performance on Pictures

In order to evaluate the overall performance of this approach and compare it with that of the two other existing methods, we downloaded all Pictures from the Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) and hid randomly-generated arrangements of lengths 200, 300, 400, 500, 600 into each Picture with our approach and the other two methods. The Pictures were stored in three different folders, including a test folder, a train folder and a val folder. The test folder and the train folder both contained 200 Pictures and the val folder 100 Pictures. Tables 5, 6, and 7 show the mean values and standard deviations of the PSNRs and SSDs achieved by all methods on all arrangements and Pictures. The value of w_d is chosen to be 1 for our approach. It can be seen from the tables that the mean values of PSNRs achieved by our approach on Pictures in all three folders are significantly higher than that achieved by the two other methods.

Table 5. The means and standard deviations (STD) of SSDs and PSNRs obtained with three methods on the 200 Pictures in the train folder of the BSDS500 dataset. SL is arrangement length.

Method		SL=200	SL=300	SL=400	SL=500	SL=600
Our method	PSNR	**73.61**	**71.22**	**69.65**	**68.47**	**67.52**
	STD	0.89	0.71	0.63	0.59	0.57
	SSD	**28.76**	**49.63**	**71.14**	**92.98**	**115.84**
	STD	4.33	6.20	7.88	9.48	11.56
Method in [1]	PSNR	68.09	66.38	65.00	63.84	62.92
	STD	4.60	4.46	4.40	4.33	4.29
	SSD	217.57	302.55	404.14	500.91	604.76
	STD	155.04	169.14	172.47	174.36	174.45
Method in [3]	PSNR	62.00	59.99	58.78	57.82	56.95
	STD	2.05	1.79	1.66	1.58	1.49
	SSD	456.80	710.90	931.30	1152.60	1396.80
	STD	129.86	146.56	153.21	157.00	164.92

Table 6. The means and standard deviations (STD) of SSDs and PSNRs obtained with three methods on the 200 Pictures in the test folder of the BSDS500 dataset. SL is arrangement length.

Method		SL=200	SL=300	SL=400	SL=500	SL=600
Our method	PSNR	**73.16**	**71.28**	**69.69**	**68.51**	**67.55**
	STD	0.78	1.12	0.89	0.84	0.76
	SSD	**28.68**	**49.38**	**70.88**	**92.75**	**115.48**
	STD	4.56	6.65	8.51	10.39	12.00
Method in [1]	PSNR	67.72	65.83	64.38	63.26	62.29
	STD	4.20	4.17	4.15	4.19	4.20
	SSD	196.82	300.70	407.38	524.50	643.32
	STD	136.11	165.89	169.43	170.68	172.81
Method in [3]	PSNR	61.75	59.77	58.56	57.55	57.18
	STD	6.16	5.90	5.83	5.78	3.84
	SSD	444.40	685.60	910.30	1154.40	1396.80
	STD	129.55	144.26	154.27	160.67	166.75

Table 7. The means and standard deviations (STD) of SSDs and PSNRs obtained with three methods on the 100 Pictures in the val folder of the BSDS500 dataset. SL is arrangement length.

Method		SL=200	SL=300	SL=400	SL=500	SL=600
Our method	PSNR	**73.55**	**71.13**	**69.59**	**68.41**	**67.47**
	STD	0.52	0.40	0.35	0.31	0.29
	SSD	**28.94**	**50.36**	**71.62**	**94.09**	**116.66**
	STD	3.41	4.53	5.57	6.59	7.70
Method in [1]	PSNR	67.72	65.83	64.38	63.26	62.29
	STD	4.20	4.17	4.15	4.19	4.20
	SSD	196.82	300.70	407.38	524.50	643.32
	STD	136.11	165.89	169.43	170.68	172.81
Method in [3]	PSNR	61.48	60.21	59.05	58.11	57.25
	STD	7.07	2.54	2.72	2.63	2.50
	SSD	461.10	710.50	937.50	1146.80	1389.30
	STD	140.05	152.03	160.74	162.08	167.27

3.3. Additional analysis and discussions

From the steps of the algorithm, it is clear that the algorithm needs $c+r$ integer keys in total to shuffle an arrangement that contains up to cr bits. The key space is thus of size S^{c+r}, where S is the number of positive integers that can be determined to be co-prime with c or r by a computer. In the case where d one-dimensional logistic mappings are used for encoding, the number of possible combinations of initial values for these one-dimensional logistic mappings is at least U^d, where U is the number of real numbers that can be generated by a computer between 0 and 1. It is straightforward to see that both S and U are generally large numbers and at least larger than 1000. The size of the key space is thus at least $10^{3(c+r+d)}$. For an arrangement that contains more than 100 binary bits, if 5 one-dimensional logistic mappings are used for encoding, this number is at least 10^{75}, which suggests that our approach is secure against attacks based on exhaustive search when the hidden arrangement is of a moderate length. In [26], a hybrid approach based on computing error histogram and Picture interpolation with greedy weights is developed for the information-hiding in medical Pictures. Experimental results have shown that this new approach significantly outperforms other existing approaches using interpolation-based reversible watermarking. Our algorithm differs from the algorithm developed in [26] in two major aspects. Firstly, our algorithm searches in the Picture and determines the image dots for embedding with a dynamic programming approach, while the algorithm in [26] performs the embedding with an adaptive Picture-interpolation-based approach. Secondly, our approach encrypts the data that need to be hidden within a Picture before the embedding is performed while the algorithm in [26] directly embeds the data into an Picture.

4. CONCLUSION

In this paper, we build up another calculation for Picture-based information-hiding. The hiding of an arrangement of parallel bits into a given dark scale Picture can be enacted in two phases. In the main stage, the parallel bits in the arrangement are rearranged depending on a lot of positive whole-number keys, and the rearranged grouping is then encoded with an arrangement of one-dimensional calculated mappings. In the after-arrangement stage, the rearranged and encoded succession is separated into locales of equivalent length, and the areas are successively inserted into the dark estimations of the relating image dots in the given dim-scale Picture.

A unique programming calculation is utilized to effectively decide the areas of inserting that can limit the contrast between a spread Picture and the relating stego Picture. This methodology need not be bothered with the spread Picture or a standard Picture to recuperate the information covered up inside a stego Picture. Our tests on both normal benchmark and clinical Pictures show that this calculation can accomplish preferred execution over two other existing strategies. Likewise, our examination shows that the shrouded information is secure against assaults depending on comprehensive pursuit. This methodology is after-arrangemently possibly valuable for Picture-based information-hiding in an assortment of uses.

References

[1] Fabien AP, Anderson RJ and Kuhn MG, "Information hiding-a survey," Proceeding of the IEEE Special Issue on Protection of Multimedia Content, vol. 87, no. 7, pp. 1062-1078, 1999.

[2] Chang CC and Tseng HW., "A steganographic method for digital Pictures using side match," Pattern Recognition Letters, vol. 25, no. 12, pp. 1431-1437, 2004.

[3] Wang CM, Wu NI, Tsai CS, and Hwang MS, "A high quality steganographic method with image dot-value differencing and modulus function," The Journal of Systems and Software, vol. 81, no.1, pp. 150-158, 2008.

[4] Wu HC, Wu NI, Tsai CS and Hwang MS, "Picture steganographic scheme based on image dot-value differencing and LSB replacement methods," IEE Proceedings Vision, Picture &Picture Signal Process, vol. 152, no. 5, pp. 611-615, 2005.

[5] Lu T, Liao S, and Chang C, "The information hiding technology based on the similar sample blocks of grayscale Picture," Proceedings of the Sixth International Conference on Intelligent Information-hiding and Multimedia Signal Processing, Washington D.C., USA. New York: IEEE Computer Society pp.17-20, Oct. 15-17, 2010.

[6] Lin CC, Tai WL, and Chang CC, "Multilevel reversible data hiding based on histogram modification of difference Pictures," Pattern Recognition, vol. 41, pp. 3582-3591, 2008.

[7] Arabzadeh M and Rahimi MR, "Reversible data hiding scheme based on maximum histogram gap of Picture blocks," KSII Transactions on Internet Information Systems, vol. 6, no. 8, pp. 1964-1981, 2012.

[8] Lo CC, Hu YC, Chen WL, and Wu CM, "Reversible data hiding scheme for BTC-compressed Pictures based on histogram shifting," International Journal of Security and Its Applications, vol. 8, no. 2, pp. 301-304, 2014.

[9] Ou B, Li X, Zhao Y, Ni R, and Shi YQ, "Pairwise prediction-error expansion for efficient reversible data hiding," IEEE Transactions on Picture Processing, vol. 22, no. 12, pp. 5010-5021, 2013.

[10] Liu Y, Chang CC, and Nguyen TS, "High capacity turtle shell-based data hiding," IET Picture Process, vol. 10, no. 2, pp. 130-137, 2016.

[11] Zhang S, Gao T, and Yang L, "A reversible data hiding scheme based on histogram modification in integer DWT domain for BTC compressed Pictures," International Journal of Network Security, vol. 18, no. 4, pp. 718-727, 2016.

[12] Khalil MI, "Medical Picture steganography: study of medical Picture degradation when embedding data in the frequency domain," International Journal of Computer Network Information Security, vol. 9, 22, 2017.

[13] Kelkar V, Tuckley K, and Nemade H, "Novel variants of a histogram shift-based reversible watermarking technique for medical Pictures to improve hiding capacity," Journal of Healthcare Engineering, to appear with doi number: https://doi.org/10.1155/2017/3538979, 2017.

[14] Liu Y, Chang CC, Huang PC, and Hsu CY, "Efficient information hiding based on theory of numbers," Symmetry, vol. 10, no. 1, 19, 2018.

[15] Baptista MS, "Cryptography with chaos," Physics Letters, Section A, vol. 240, no. 1-2, pp. 50-54, 1998.

[16] Chen G, Mao Y, and Chui CK, "A symmetric Picture encryption scheme based on 3D chaotic cat maps," Chaos, Solitons and Fractals, vol. 21, no. 3, pp. 749-761, 2004.

[17] Edward O. Chaos in Dynamical Systems. 2nd ed. Cambridge University Press, pp. 223-224, Cambridge, 2003.

[18] Chen E, Min LQ, and Chen GR, "Discrete chaotic systems with one-line equilibria and their application to Picture encryption," International Journal of Bifurcation Chaos, vol. 27, 1750046-1-17, 2017.

[19] Hamza R, "A novel pseudo random arrangement generator for Picture-cryptographic applications," Journal of Information Security Applications, vol. 35, pp. 119-127, 2017.

[20] Jain A and Rajpal N, "A robust Picture encryption algorithm resistant to attacks using DNA and

chaotic logistic maps," Multimedia Tools and Applications, vol. 75, pp. 5455-5472, 2016.

[21] Zhu S and Zhu C, "Picture encryption algorithm with an avalanche effect based on a six-dimensional discrete chaotic system," Multimedia Tools and Applications, to appear with doi number: https://doi.org/10.1007/s11042-018-6078-2, 2018.

[22] Kuang CK and Nien HH, "Multi chaotic systems based image dot shuffle for Picture encryption," Optics Communications, vol. 282, no. 11, pp. 2123-2127, 2009.

[23] Wang Y, Wong KW, Liao X, and Chen G, "A new chaos-based fast Picture encryption algorithm," Applied Soft Computing Journal, vol. 11, no. 1, pp. 514-522, 2011.

[24] Wang XY, Yang L, Liu R, and Kadir A, "A chaotic Picture encryption algorithm based on perceptron model," Nonlinear Dynamics, vol. 62, no. 3, pp. 615-621, 2010.

[25] Song Y, Song J and Qu J, "A secure Picture encryption algorithm based on multiple one-dimensional chaotic systems," Proceedings of the Second IEEE International Conference on Computer and Communications, Chengdu, China. New York: IEEE Computer Society pp.584-587, Oct. 14-17, 2016.

[26] Khosravi MR and Yazdi M, "A lossless data hiding scheme for medical Pictures using a hybrid solution based on IBRW error histogram computation and quartered interpolation with greedy weights," Neural Computing and Applications, to appear with doi number: https//doi.org/10.1007/s00521-018-3489-y, 2018.