Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/080208

Hill Cipher Modification based on Pseudo-Random Eigenvalues

Ahmed Mahmoud[1,2,*] *and Alexander Chefranov*[1]

[1] Computer Engineering Department, Faculty of Engineering, Eastern Mediterranean University, North Cyprus
[2] Information Technology Department, Faculty of Engineering and Information Technology, AlAzhar University, Gaza, Palestine

Received: 12 Mar. 2013, Revised: 12 Jul. 2013, Accepted: 14 Jul. 2013
Published online: 1 Mar. 2014

Abstract: The Hill cipher is resistant to brute-force and statistical attacks, but it can be broken with a known plaintext-ciphertext attack (KPCA). In this paper, we propose a modification of the Hill cipher, HCM-PRE, which is still resistant to brute-force and statistical attacks, and is resistant also to KPCA due to dynamic encryption key matrix generating. With the modification, the new HCM-PRE can be applied widely in the systems which need high security (e.g., image encryption). Experimental results show that the proposed modification is significantly more effective in the encryption quality of images than original Hill cipher and its known modifications (HCM-PT, HCM-H, HCM-HMAC, and HCM-EE) in the case of images with large single colour areas, and slightly more effective otherwise. HCM-PRE is about two times faster than HCM-EE and HCM-HMAC and four times faster than HCM-H in the frame of our experiments.

Keywords: Keywords: Hill cipher, matrix, eigenvalue, diagonal matrix, exponentiation, pseudo-random number, dynamic key, image encryption

1 Introduction

The Hill cipher (HC) [1] and [2] is a well-known symmetric cryptosystem that multiplies a plaintext vector by a key matrix to get the ciphertext. It is very attractive due to its simplicity and high throughput [3] and [4] ; it is resistant to the frequency letter analysis, but it can be broken by the known plaintext-ciphertext attack (KPCA) [5]. It has a large key space [3]. HC modification [4], HCM-PT, uses a dynamic key matrix obtained by permutations of rows and columns from the master key matrix to get every next ciphertext, and transfers it together with an HC-encrypted permutation to the receiving side. Thus, in HCM-PT, each plaintext vector is encrypted by a new dynamic key matrix that prevents the KPCA; the number of possible dynamic keys is equal to the number of permutations of the key matrix rows, and it may be used as a characteristic of its security. But permutations in HCM-PT are transferred HC-encrypted, which means that master key matrix can be revealed by the KPCA on the transferred encrypted permutations [6]. Modification [7], HCM-NPT, works as HCM-PT does, but without permutations transfer; instead, both communicating parties use a pseudo-random permutation generator, and only the consecutive number of the necessary permutation is transferred to the receiver. It has good computational complexity and the number of its dynamic keys is the same as for HCM-PT.

Another HC modification [8], HILLMRIV, also uses dynamic key matrices: it modifies each row of the matrix key by multiplying the current key by a secret initial vector. But HILLMRIV is still vulnerable to KPCA [9] and [10]. Another HC modification [6], HCM-H, also uses dynamic key matrix produced with the help of a one way hash function applied to an integer picked up randomly by the sender to get the key matrix, and a vector added to the product of the key matrix with a plain text. HCM-H is vulnerable [11] to chosen-ciphertext attack because the selected random number is transmitted in clear over the communication link and is repeated. To avoid this random number transfer, a modification of HCM-H [11], HCM-HMAC, uses only a seed value secure transfer, and then both parties generate necessary numbers synchronously, where HMAC is a hash function, e.g., MD5 [12], SHA-1[13]. The difference between HCM-H and HCM-HMAC is similar to the difference

[*] Corresponding author e-mail: ahmed@alazhar.edu.ps

between HCM-PT and HCM-NPT. In [14], we introduce a modification of the HC, HCM-EE, based on the use of eigenvalues for matrix exponentiation to a pseudo-random power for a new key matrix generating for each plaintext block. In this paper we propose a new modification of HC, HCM-PRE, based on the use of pseudo-random eigenvalues to construct a key matrix [15] and modify it for each new plaintext.

The rest of the paper is organized as follows. Section 2 briefly introduces the Hill cipher, HCM-PT, HCM-NPT, HCM-H, HCM-HMAC, and HCM-EE. Section 3 is devoted to the proposed modification HCM-PRE. Experimental results of image encryption quality and the performance of the proposed modification versus the known ones are presented in Section 4. Security and statistical analysis of the proposed HCM-PRE are discussed in Section 5. Finally, the conclusion is presented in section 6. In the Appendix, description of used encryption quality measures is given.

2 Overview of the Hill Cipher and its Mmodifications

All matrices considered throughout the paper are *m x m* sized with entries over $Z_N = \{0, 1, ...N-1\}$, hence all the operations in encryption/decryption algorithms are assumed *mod N*, where *m* (block size) and *N* (alphabet cardinality) are selected positive integers (e.g., *N=256* for gray scale images). Also, we assume that two parties, *A* and *B*, want to communicate securely, and *A* is a sender, and *B* is a receiver.

First, we introduce HC, HCM-PT, HCM-NPT, HCM-H, HCM-HMAC and then we describe HCM-EE.

When HC is used, *A* and *B* share an invertible key matrix *K*. Sender *A* encrypts a plaintext vector, *P*:

$$C = K \times P. \tag{1}$$

The receiver, B, decrypts the ciphertext vector *C* by

$$P = K^{-1} \times C, \tag{2}$$

where K^{-1} is the key inverse. For existence of K^{-1}, we require

$$gcd(det(K) mod N, N) = 1, \tag{3}$$

where *gcd* is the greatest common divisor and *det(K)* denotes the determinant of *K*.

HCM-PT [4] differs from HC in the following. To encrypt a plaintext *P*, *A* randomly selects a permutation, *t*, of Z_m, and permutes the rows and columns of a key matrix *K* according to *t* producing a new key-matrix $K_t = t(K)$. HCM-PT encryption is then performed by (1), but using K_t instead of *K*. Additionally, sender *A* encrypts *t* by (1) using *K* and getting *u* as a ciphertext, and sends *C* and *u* together to the receiver. In order to decrypt the ciphertext, *B* decrypts *t* from *u* by (2), gets $(K^{-1})_t = (K_t)^{-1}$ [4] from K^{-1}, and then reveals the plaintext by (2), using $(K^{-1})_t$ instead of K^{-1}. The number of dynamic keys used in HCM-PT is

$$NDK(HCM-PT) = m! \tag{4}$$

HCM-NPT [7] uses the same initialization and the same encryption/decryption technique as HCM-PT does. But HCM-NPT assumes that the sender, *A*, and the receiver, *B*, share a secret seed value, *SEED*, which is used to generate a pseudo-random sequence of permutations. In order to encrypt a plaintext, the sender, *A*, selects a number *r*, and calculates

$$t_r = PRPermutationG(SEED, r), \tag{5}$$

getting the *r*-th output permutation from the pseudo-random permutation generator *PRPermutationG* (*r* can be a block number in the sequence of transmitted blocks, or its function). Sender *A* then gets a ciphertext *C* as in HCM-PT, and sends to receiver *B* both *C* and *r*. In order to decrypt, *B* calculates t_r according to (5), and then gets the plaintext as in HCM-PT. The number of dynamic keys used in HCM-NPT, NDK(HCM-NPT), is the same as NDK(HCM-PT) (4).

Proposed in [6], another HC modification, HCM-H, works as follows. The sender, *A*, and the receiver, *B*, share an invertible matrix *K*. To encrypt the plaintext *P*, *A*, selects a random integer *a*, where $0 < a < N$, and applies a one way hash function to compute the parameter $b = f(a \parallel k_{11} \parallel k_{12} ... \parallel k_{mm})$, where $k_{11}, k_{12} ..., k_{mm}$ are the elements of *K*; *b* is used to select the k_{ij} from *K*, where *i* and *j* can be calculated according to (6)

$$i = \left\lfloor \frac{b-1}{m} \right\rfloor .(mod m) + 1, j = b - \left\lfloor \frac{b-1}{m} \right\rfloor .m. \tag{6}$$

Then, *A* generates a vector $V = [v_1, v_2, ..., v_m]$ according to (7)

$$\begin{aligned} &v_1 = f(k_{ij}) mod N, \\ &v_2 = f(v_1) mod N = f^2(k_{ij}) mod N, \\ &..., \\ &v_m = f(v_{m-1}) mod N = f^m(k_{ij}) mod N. \end{aligned} \tag{7}$$

Then, *A* encrypts the plaintext *P* by

$$C = k_{ij} \times P \times K + V, \tag{8}$$

and sends together *C* and *a* to *B*. The decryption process is done by

$$P = k_{ij}^{-1} \times (C - V) \times K^{-1}. \tag{9}$$

The number of dynamic keys used in HCM-H is

$$NDK(HCM-H) = min(m^2, N). \tag{10}$$

Proposed in [11], HCM-HMAC, works as follows. In order to transfer a seed value, the sender, A, transmits the seed value a according to the Hughes key-exchange protocol [16]. Then the seed value a_0 can be used to generate the chain of pseudo-random numbers synchronously by the both parties; a can be calculated by

$$a_t = HMAC_{k'}(a_{t-1}), t = 1,2,..., \tag{11}$$

where k' is the secret key of the hash function, k' can calculated by

$$k' = (k_{11} \parallel k_{12} \parallel k_{13} \parallel ... \parallel k_{mm} \parallel a_{t-1}) mod 2^q, \tag{12}$$

where $\parallel$ denotes the concatenation, q is the number of bits required for the hash function, and a_t is used in recursive calculations of the vector $V = [v_1, v_2, ..., v_n]$, calculated for the encryption of t-th block, $v_0 = 1$, if $a_t \equiv 0(mod\ \ p)$ otherwise $v_0 = a_t mod\ \ p$, p is a prime number.

$$v_i = k_{ij} + \overline{v}_{i-1} a_t mod\ \ p, i = 1,2,...,m, and$$
$$j = (v_{i-1} mod\ \ m) + 1 \tag{13}$$

$\overline{v}_{i-1}$ is calculated by

$$\overline{v}_{i-1} = 2^{\lceil \frac{r}{2} \rceil} + \left(v_{i-1} mod 2^{\lceil \frac{r}{2} \rceil} \right), \gamma = \lfloor log_2 v_{i-1} \rfloor + 1 \tag{14}$$

where $\gamma = \lfloor log_2 v_{i-1} \rfloor + 1$ denotes the bit length of v_{i-1}. Then, A encrypts the plaintext P_t by

$$C_t = v_0 \times P_t \times K + V mod p, \tag{15}$$

and sends together C_t and a to B, t=*1,2,...*. The receiver B calculates the required parameters by using (10)-(15), and then gets the plaintext by

$$P_t = v_0^{-1} \times (C_t - V) \times K^{-1} mod p. \tag{16}$$

HCM-EE [14] works as follows. Sender A selects a set $E = \{e_1, e_2, ..., e_m\} \subset Z_N - \{0\}$, $gcd(e_j, N) = 1$, gcd is the greatest common divisor, $1 \le j \le m$; at least one e_j should have the maximal order which is $\frac{\varphi(N)}{2}$ for N being a power of 2 [17], $\varphi(N)$ is the Eulers totient function [5], giving the number of positive integers less than N and co-prime to it. Then A constructs an invertible matrix Q and calculates the key matrix K [15]:

$$K = Q \times D \times Q^{-1}, \tag{17}$$

where D is a diagonal matrix, diagonal elements of which are its eigenvalues from E. Note that Q and D satisfy (3); A and B share them securely. Additionally, they share the secret values, *SEEDl* and *SEEDt*; *SEEDl* is used to generate the set of pseudo-random numbers $l = \{l_1, l_2, ..., l_n\}$ by (18), $l_i \neq 0$ and $l_i \in \{2, ..., \varphi(N) - 1\}$, $1 \le i \le n$, n is the number of blocks. *SEEDt* is used to generate a pseudo-random sequence of permutations t. In order to encrypt the i-th plaintext block , A selects

$$l_i = PRNG(SEEDl, i) > 0, \tag{18}$$

then calculates

$$E_i = \{e_j^{l_i}\}_{t_r}, 1 \le j \le m, 1 \le i \le n, \tag{19}$$

where $e_j \in E$, n is the number of blocks, and the random permutation t_r can be obtained by (5). Finally, A calculates

$$KM_i = Q \times D_i \times Q^{-1}, \tag{20}$$

where D_i is a diagonal matrix, diagonal elements of which are from E_i after exponentiation to l_i and permutation t_r are performed and

$$i = \frac{\varphi(N)}{2}.r + s, 0 \le s < \frac{\varphi(N)}{2}. \tag{21}$$

The plaintext P_i is encrypted as follows

$$C_i = KM_i \times P_i + diag(D_i), \tag{22}$$

where $dig(D_i)$ is a vector of the main diagonal elements of D_i.

In order to decrypt the ciphertext, B computes l_i according to (18), t_r according to (5) and (21), E_i according to (19), and

$$(KM_i)^{-1} = (Q \times D_i \times Q^{-1})^{-1} = Q \times D_i^{-1} \times Q^{-1}.$$

Then, B retrieves the plaintext:

$$P_i = KM_i^{-1} \times (C_i - diag(D_i)). \tag{23}$$

It is appropriate to mention that for computing KM_i we use a diagonal matrix, and only the diagonal entries of D_i are exponentiated to the power l_i , requiring $O(mlog_2 l_i)$ multiplications. On the other hand, to get D_i^{-1}, we calculate the inverse of m numbers only. Note also that Q^{-1} and D_i^{-1} are calculated only once. The diagonal elements of D_i^{-1} belong to the group G of numbers co-prime to N. Based on Theorem 10.3 [17] we see that for N=*256*, $64 = \frac{\varphi(N)}{2}$ is the maximal order of elements of G (odd numbers in Z_{256}). In HCM-EE, we select at least one element in the diagonal with the maximum order to guarantee the maximum period of the diagonal elements. The number of dynamic keys of HCM-EE is estimated as

$$LB \cdot m! \le NDK(HCM - EE) \le \varphi(N) \cdot m! \tag{24}$$

where LB is the maximum order of the diagonal elements in D_i. If N is a power of 2, $LB = \frac{\varphi(N)}{2}$.

3 The Proposed Scheme

The proposed HCM-PRE uses the same encryption/decryption technique as HCM-EE [14] does. But HCM-PRE differs from the HCM-EE in the key construction. It uses pseudo-random eigenvalues instead of static eigenvalues exponentiated to pseudo-random powers in HCM-EE. If the sender, A, and the receiver, B, want to communicate using HCM-PRE, they share a secret value, *SEED*, that is used to generate pseudo-randomly a sequence of eigenvalue sets, $E = (E_i), 1 \leq i \leq n$:

$$E = PRSetG_{SEED}(n,m), \tag{25}$$

where $E_i = \{e_{ij}\} \sqsubset Z_N - \{0\}$ is is a set of eigenvalues of the matrix to be constructed, e_{ij} is relatively prime to N, $1 \leq j \leq m$, $1 \leq i \leq n$, for positive integers n and m , n is the number of blocks; $PRSetG_{SEED}(n,m)$ is a pseudo-random set sequence generator (using e.g., RC4 initialized by *SEED*) returning n sets, each of which contains m numbers.

Sender A then constructs an invertible matrix Q as in HCM-EE. The key matrix is calculated by (17) but, instead of D, diagonal matrix D_i is used, diagonal elements of which are all the eigenvalues from E_i, $1 \leq i \leq n$. HCM-PRE uses a different set of diagonal elements for every plaintext. It may be easily shown that KM_i is invertible modulo N since Q and D_i have (by construction) determinants relatively prime to N.

Finally, the plaintext P_i is enciphered by (22).

To decrypt a ciphertext, receiver B computes E according to (25), and finds

$$(KM_i)^{-1} = (Q \times D_i \times Q^{-1})^{-1} = Q \times D_i^{-1} \times Q^{-1}. \tag{26}$$

Note that to get D_i^{-1} , we calculate the inverse of m numbers only, and that Q and Q^{-1} are constructed only once. Receiver B then retrieves the plaintext by (23). To generate an invertible key matrix D_i, the eigenvalues must be in the multiplicative group of Z_N, the number of possible eigenvalues in the multiplicative group of Z_N is $\varphi(N)$. Hence the number of dynamic keys of HCM-PRE is

$$NDK(HCM-PRE) = min\left(\varphi(N)^m, \frac{Period(RC4)}{m}\right) \tag{27}$$

where *Period(RC4)* is overwhelmingly likely to be greater than 10^{100} [18].

4 Image Encryption Quality and Performance of the HCM-PRE Versus Known Ones

We developed programs for simulating the encryption schemes in C# on an Intel(R) Core(TM) 2 Duo 1.8 GHz processor with 2-GB RAM and Windows XP.

In our experiments, several RGB images are encrypted. Firstly, the image, P, of size $N \times M$ is converted into its RGB components. Afterwards, each colour matrix (R, G, B) is converted into a vector of integers within {0, 1, ...,255}. Each vector has the length $L = N \times M$. Then, the so obtained three vectors represent the plaintext $P(3 \times L)$ which will be encrypted using the block size m=16.

We examine the encryption quality for three different images containing very large single colour areas: Nike.bmp (Fig. 1), Symbol.bmp (Fig. 2), and Blackbox.bmp (Fig. 3). Also we examined the encryption quality for an image that does not contain many high frequency components: Lena.bmp (Fig. 4). The Girl.bmp (Fig. 5) is used as an example of an image containing many high frequency components. Each image is encrypted using HCM-PT, HCM-H, HCM-HMAC, HCM-EE, and HCM-PRE.

The quality of encryption of these images is studied by visual inspection (Figs. 1-5) and quantitavely (Table 1, used irregular deviation based quality measure ID [8,19, 20] is explained in the Appendix).

Based on visual inspection, it is obvious that the HCM-PRE and HCM-EE are better than the HCM-PT, HCM-H, and HCM-HMAC in hiding all the features of the image containing large single colour areas (Figs. 1-3).

Based on the numerical evaluation of encryption quality measure ID (Table 1), we note that the proposed scheme HCM-PRE versus HCM-EE give better encryption quality. Table 1 shows also that the proposed scheme HCM-PRE is more effective in encryption quality than HCM-PT, HCM-H, and HCM-HMAC. On the other hand, HCM-PT, HCM-H, HCM-HMAC, HCM-EE, and HCM-PRE are all good in encrypting images containing many high frequency components; all the algorithms give nearly the same results but the HCM-PRE and HCM-EE are the most effective ones (Table 1, rows 4-5).

We examined the encryption time for the Nike.bmp image having pixels and 45KB size. The encryption time measured when applying HCM-PT, HCM-H, HCM-HMAC, HCM-EE, and HCM-PRE is shown in Table 2. In our implementation, HCM-EE and HCM-PRE were used with RC4 [5] for the pseudo-random permutation generator (5), pseudo-random number generator (18) for HCM-EE, and pseudo-random set generator (25) for HCM-PRE. We implemented HCM-H with SHA-1 [13] since the latter has been used in [6], and the built-in HMAC from C # with HCM-HMAC-SHA-1. Table 2 shows that HCM-PRE has the best execution time; it is roughly two times faster than HCM-EE and HCM-HMAC, and four times faster than HCM-H. HCM-EE roughly is twice better than HCM-H and it has nearly the same execution time as of HCM-HMAC but HCM-EE has better encryption quality (Figs. 1-5, and Table 1). Table 2 shows that HCM-PT is faster than HCM-EE but equations (4) and (24) show that NDK(HCM-EE) is greater than NDK(HCM-PT), hence HCM-EE is more secure than HCM-PT. Equation (27) shows that NDK(HCM-PRE) is greater than NDK(HCM-EE). Hence HCM-PRE is more secure and is more effective in the encryption time than HCM-PT, HCM-H, HCM-HMAC and HCM-EE.

Table 1: ID for encrypted images using HCM-PT, HCM-H, HCM-HMAC, HCM-EE and HCM-PRE, m=16.

Image/Algorithm	HCM-PT	HCM-H	HCM-HMAC	HCM-EE	HCM-PRE
Nike.bmp	23980.79	13171.75	9983.87	2656.62	1338.04
Symbol.bmp	10482.25	5755.68	4830.91	2378.07	1874.30
Blackbox.bmp	34036.28	18511.62	11491.48	3285.25	1328.63
Lena.bmp	10256	10518.66	10469.33	10172.66	10201.33
Girl.bmp	11459.55	10472.61	10336.77	9942.21	9913.25

Table 2: Encryption time (msec) of Nike.bmp with HCM-PT, HCM-H, HCM-HMAC, HCM-EE and HCM-PRE.

HCM-PT	HCM-H	HCM-HMAC	HCM-EE	HCM-PRE
103	425	214	200	98

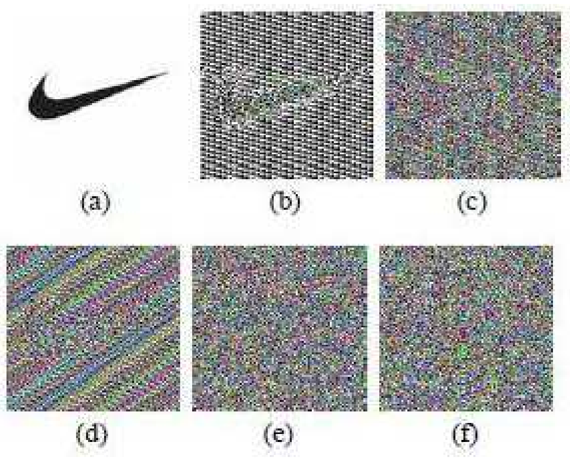

Fig. 1: a) Nike.bmp encrypted by: b) HCM-PT, c) HCM-H, d) HCM-HMAC, e) HCM-EE, f) HCM-PRE.

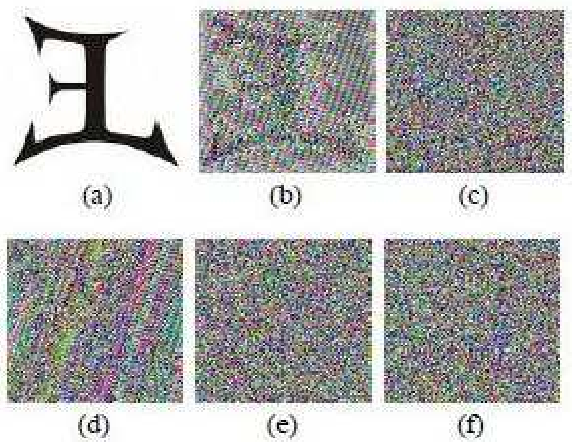

Fig. 2: a) Symbol.bmp encrypted by: b) HCM-PT, c) HCM-H, d) HCM-HMAC, e) HCM-EE, f) HCM-PRE.

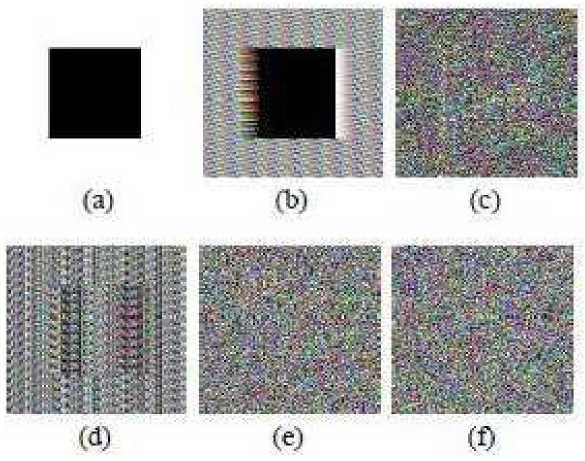

Fig. 3: a) blackbox.bmp encrypted by: b) HCM-PT, c) HCM-H, d) HCM-HMAC, e) HCM-EE, f) HCM-PRE.

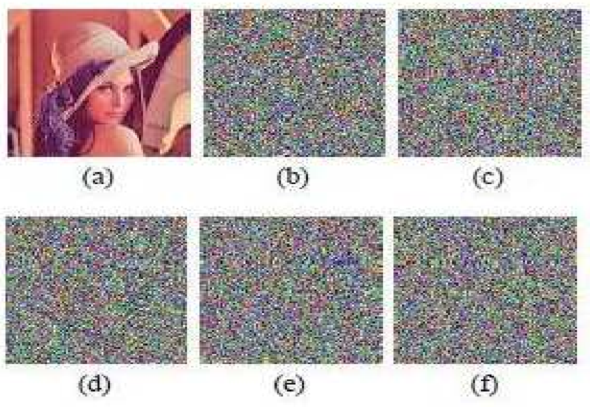

Fig. 4: a) Lena.bmp encrypted by: b) HCM-PT, c) HCM-H, d) HCM-HMAC, e) HCM-EE, f) HCM-PRE.

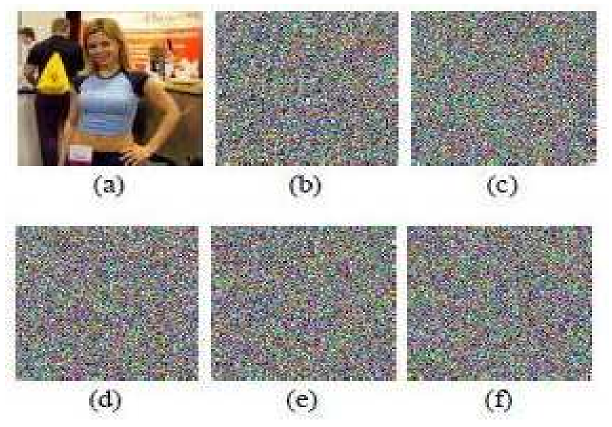

Fig. 5: a) Girl.bmp encrypted by: b) HCM-PT, c) HCM-H, d) HCM-HMAC, e) HCM-EE, f) HCM-PRE.

5 SECURITY AND STATISTICAL ANALYSIS

The ability to withstand all kinds of cryptanalysis and attacks [21,22,23,24] is a good measure of the performance of a cryptosystem. Robustness against attacks is used to evaluate the security of our scheme. It is shown that our proposed scheme is secure from the strongly cryptographic viewpoint. The results show the satisfactory security of the HCM-PRE as explained and discussed in the following subsections.

5.1 Key Space Analysis

Key space is the total number of different keys that can be used in encryption. For a secure encryption scheme, the key space should be large enough to make brute force attacks infeasible. For the HCM-PRE, the key space is the same as that of HC [3] and [4]. Therefore the key space of the scheme is large; hence it is secure against brute force attack.

5.2 Known Plaintext-Ciphertext Attack

The KPCA is effective if a same key is used to encrypt many plaintexts. Similar to HCM-PT [4], our proposed scheme HCM-PRE is secure against the KPCA since each plaintext is encrypted by a different key, and the number of such dynamic keys is significantly large (27). Equations (4), (24), and (27) show that the NDK(HCM-PRE) (27) is larger than the NDK(HCM-PT) (4) and NDK(HCM-EE) (24); hence HCM-PRE is more secure.

5.3 Statistical Analysis Resistance

In [25], it is mentioned that in [26] Shannon said it is possible to solve many kinds of ciphers by statistical analysis. A good cipher should be robust against any statistical attack. To prove the robustness of the proposed scheme, the statistical analysis has been performed. It is usually evaluated by the following measures [21,23,27, 28,29]; calculating the histograms of the encrypted images and the correlation of two adjacent pixels in the plain/encrypted image demonstrating their superior confusion and diffusion property. The obtained results show that our scheme strongly withstands statistical attacks.

5.3.1 Histograms of encrypted images

We have calculated and analyzed the histograms of several encrypted images as well as their original images. Two typical examples are given in Figs. 6-7. The histograms of the encrypted images are very close to uniform distribution; they are significantly different from those of the original image, and bear no statistical resemblance to the original image.

5.3.2 Correlation of Two Adjacent Pixels

There is a very good correlation between adjacent pixels in the plain-image (Nike.bmp: Figs. 1 and 8, Lena.bmp: Figs. 4 and 9. We studied the correlation between two adjacent pixels in plain-image and encrypted image in three different orientations (horizontal, vertical and diagonal). We use the following procedure: first 1000 pairs of two adjacent pixels in three different orientations are selected randomly from image to test correlation, then we calculate the correlation coefficient C.C of each pair. Figs. 8 and 9 show the correlation coefficients (explained in the Appendix) of two adjacent pixels in Nike.bmp and Lena.bmp encrypted by HCM-EE, HCM-PRE, HCM-NPT, HCM-H and HCM-HMAC in three different orientations as a practical example for different image types; Table 3 shows the numerical evaluation of the calculated correlations. It is clear that, the neighboring pixels in the plain-image have a very high correlation while they have a very small correlation (the closer to zero, the better) for encrypted images. This proves that the proposed encryption scheme HCM-PRE satisfies very small correlation and is better than other inspected schemes in the case of images with large single colour areas. We also note that the proposed scheme HCM-PRE versus HCM-EE gives alternately better correlation. On the other hand, the correlation values in Table 3 show that the examined schemes give nearly the same results in images containing many high frequency components.

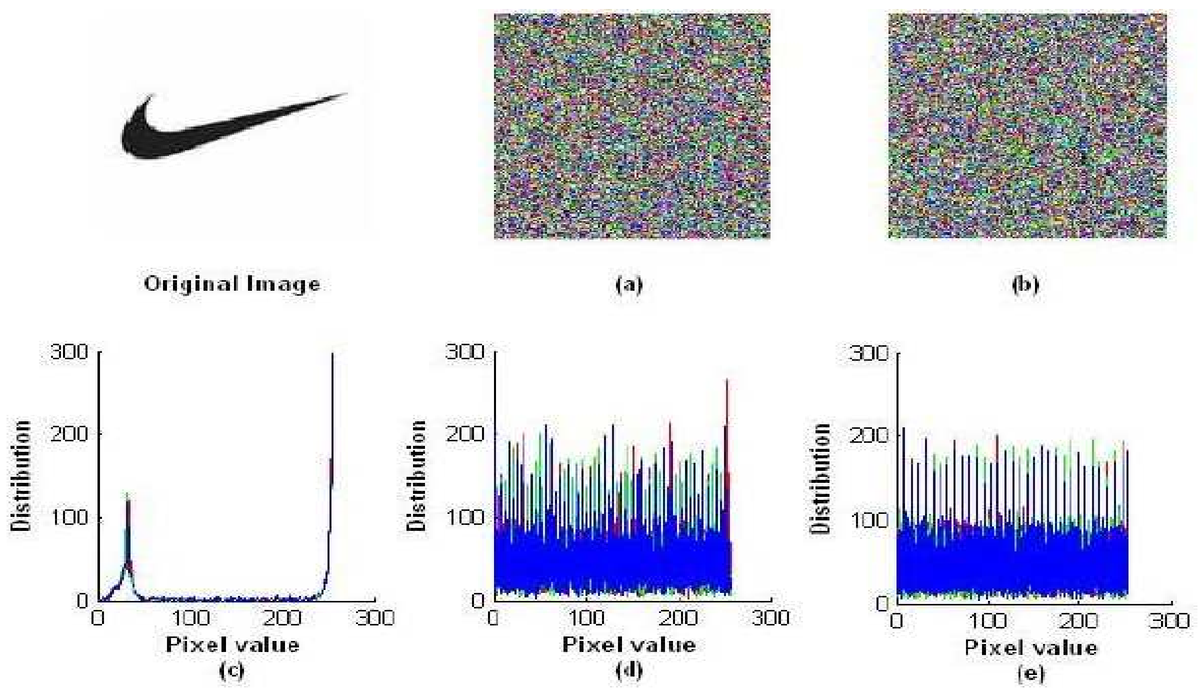

Fig. 6: Histogram of RGB layers for original/encrypted Nike.bmp: a) HCM-EE-encrypted, b) HCM-PRE-encrypted, c) histogram of the original image, d) histogram of HCM-EE-encrypted e) histogram of HCM-PRE-encrypted

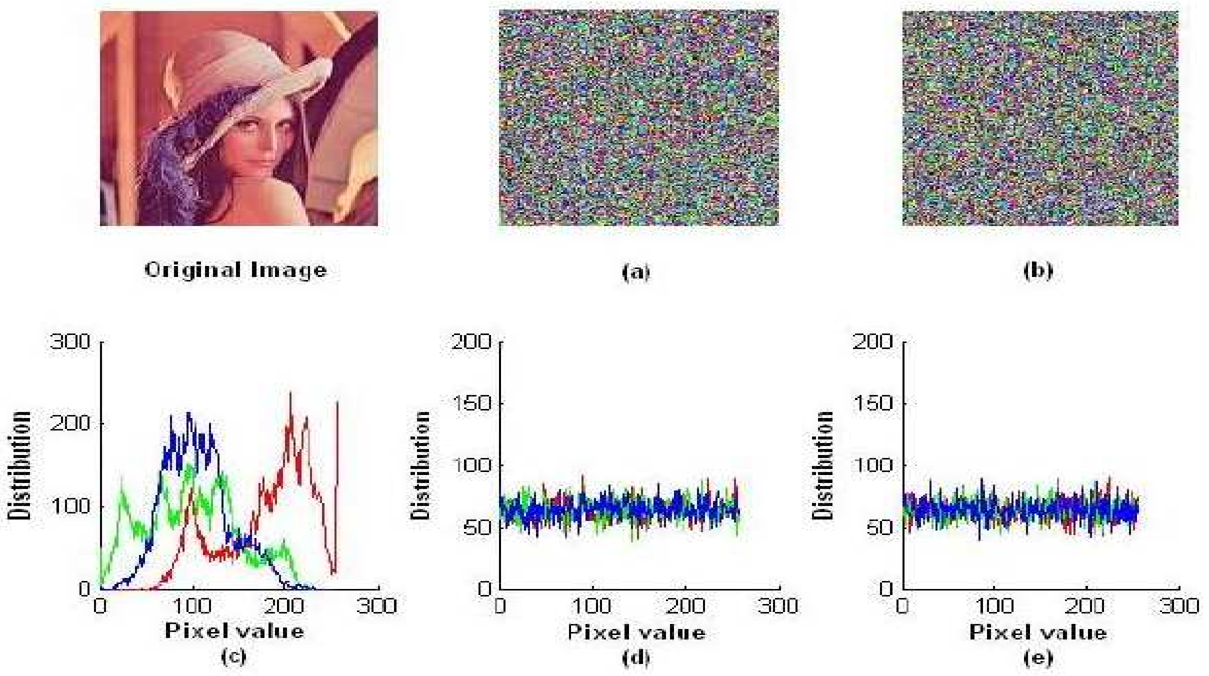

Fig. 7: Histogram of RGB layers for original/encrypted Lena.bmp: a) HCM-EE-encrypted, b) HCM-PRE-encrypted, c) histogram of the original image, d) histogram of HCM-EE-encrypted e) histogram of HCM-PRE-encrypted

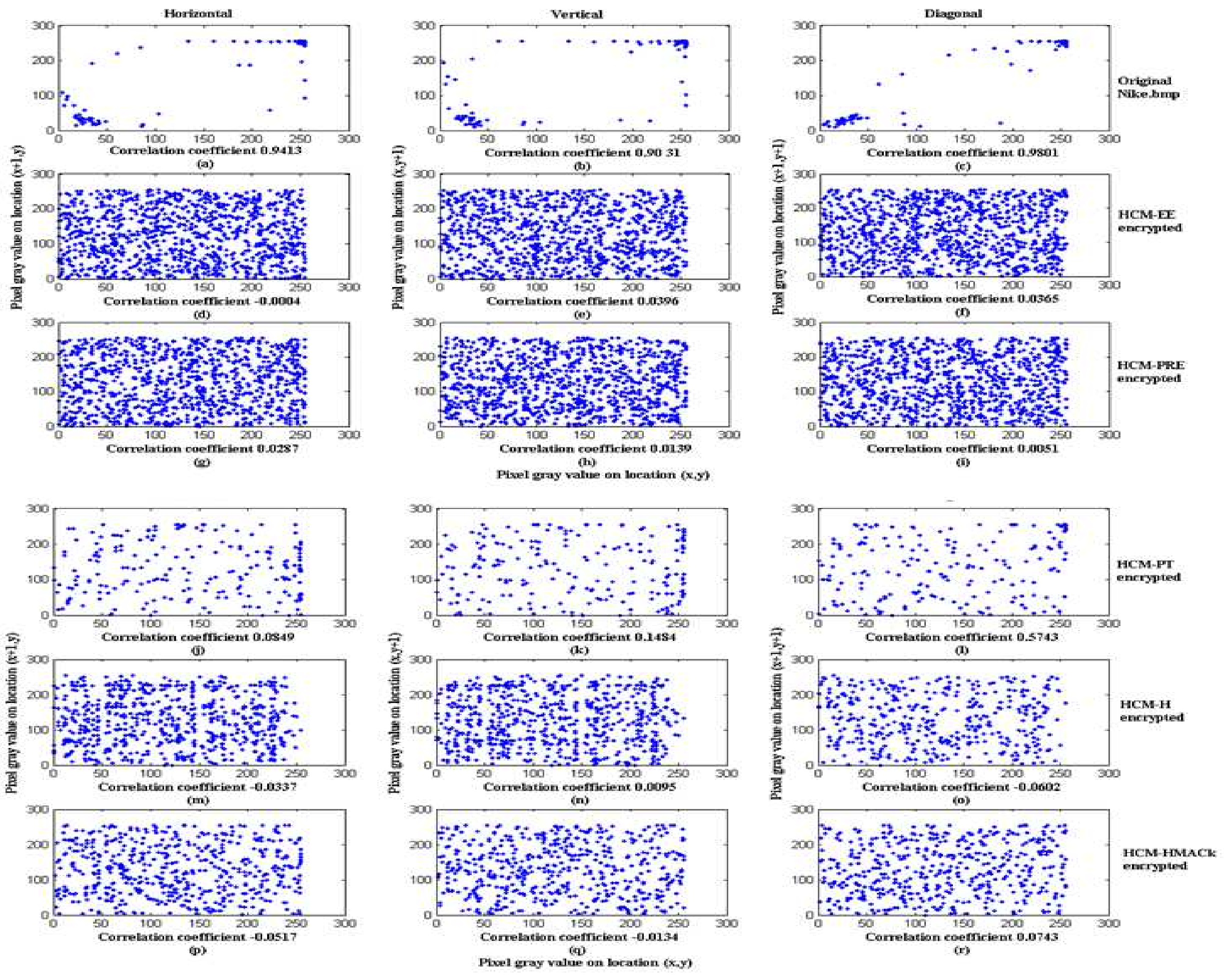

Fig. 8: Correlation coefficients of two adjacent pixels in Nike.bmp encrypted by: HCM-EE, HCM-PRE, HCM-PT, HCM-H, and HCM-HMAC

Table 3: Correlation coefficients of two adjacent pixels in original and HCM-PT-encrypted images, HCM-H-encrypted images, HCM-HMAC-encrypted images, HCM-EE-encrypted images and HCM-PRE-encrypted images.

Image	Direction	Plain Image	Encrypted Image				
			HCM-PT	HCM-H	HCM-HMAC	HCM-EE	HCM-PRE
Nike.bmp	Horizontal	0.9413	0.0849	-0.0337	-0.0517	-0.0004	0.0028
	Vertical	0.9031	0.1484	0.0951	-0.0134	0.0396	0.0013
	Diagonal	0.9801	0.5743	-0.0602	0.0743	0.0365	0.0051
Lena.bmp	Horizontal	0.9349	-0.0074	0.0498	0.0168	0.0282	-0.0435
	Vertical	0.8538	0.0037	-0.0473	0.0026	0.0272	-0.0130
	Diagonal	0.8852	-0.0774	-0.0146	0.0071	-0.0095	0.0084

Table 4: ID for encrypted images using HCM-PRE and AES, m=16.

Image/Algorithm	HCM-PRE	AES
Mecy.bmp	7874.33	47726.75
bicycle.bmp	9214.85	25031.32
Penguin .bmp	4410.31	20745.34

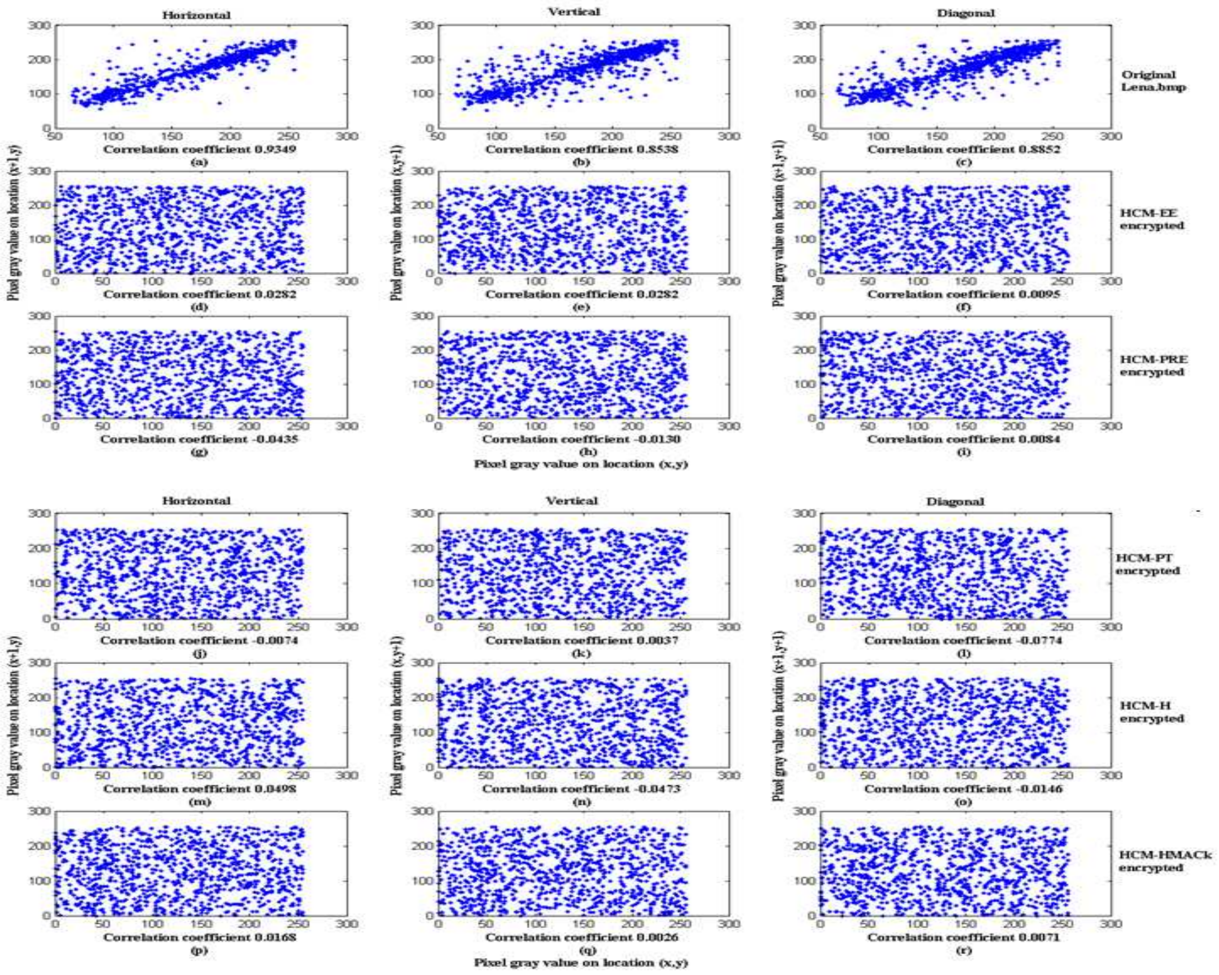

Fig. 9: Correlation coefficients of two adjacent pixels in Lena.bmp encrypted by: HCM-EE, HCM-PRE, HCM-PT, HCM-H, and HCM-HMAC

6 Conclusions

Thus far, we have presented a new HC modification, HCM-PRE, based on the use of eigenvalues for generating a new key matrix for each plaintext block. In this paper, five modifications of Hill cipher algorithms have been implemented for image encryption: HCM-PT, HCM-H, HCM-HMAC, HCM-EE, and proposed here HCM-PRE. Quality of image encryption for all algorithms is studied using visual inspection and numerical quality measures explained in the Appendix. From the obtained results, it follows that the proposed HCM-PRE is more effective in encryption quality than HCM-PT, HCM-H, HCM-HMAC, and HCM-EE. Encryption time for all the algorithms have been considered, the proposed HCM-PRE is about two times faster than HCM-EE and HCM-HMAC, and four times faster than the HCM-H. The proposed modification HCM-PRE resists the KPCA because of the use of dynamically changing key matrices similar to other considered here HC modifications (HCM-PT, HCM-NPT, HCM-H, HCM-HMAC, HCM-EE) but the proposed HCM-PRE is more secure than HCM-H, HCM-PT, HCM-NPT, HCM-EE because of the significantly larger number of dynamic keys generated ((27) versus (24), (10) and (4)). Experimental analysis also shows that the HCM-EE and HCM-PRE resist the statistical attacks.

Appendix

In the following subsections, we describe the quality encryption measures that are used in this paper: the correlation coefficient (C.C), and irregular deviation (ID) [8,19,20].

Correlation Based Quality Measure

A good encryption algorithm must produce an encrypted image of totally random patterns hiding all the features of the original image, and the encrypted image must be independent of the original image. This means that the two images must have a correlation coefficient very close to zero. The correlation coefficient is given by the following expression:

$$C.C = \frac{\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y))}{\sqrt{\sum_{i=1}^{N}(x_i - E(x))^2}\sqrt{\sum_{i=1}^{N}(y_i - E(y))^2}}.$$

The closer C.C to zero, the better.

Irregular Deviation Based Quality Measure

This quality measuring factor is based on how much the deviation affected by encryption is irregular [8,19,20]. This quality measure can be formulated as follows:

1. Calculate the matrix, *D*, which represents the absolute value of the difference between each pixel value of the original and the encrypted image respectively:
$$D = |O - E|,$$
where *O* is the original (input) image and *E* is the encrypted (output) image.
2. Construct a histogram distribution of the *D* which we get from step 1:
$$h = histogram\ (D).$$
3. Get the average value of how many pixels are deviated at every deviation value by:
$$DC = \tfrac{1}{256}\textstyle\sum_{i=0}^{255} h_i.$$
4. Subtract this average from the deviation histogram and take the absolute value by:
$$AC(i) = |h_i - DC|.$$
5. Count:
$$ID = \textstyle\sum_{i=0}^{255} AC(i).$$
The smaller *ID*, the better.

HCM-PRE Versus AES

To give adequate performance comparison, we examine our proposed HCM-PRE versus other well known algorithms (e.g. AES). We examined the encryption quality of several images. Based on visual inspection, the proposed HCM-PRE encrypts the images with large single colour areas (identical plaintext blocks), it successfully hides data patterns. The AES fails to hide the

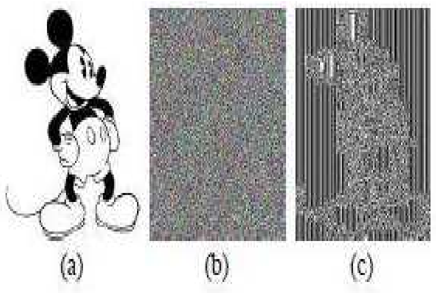

Fig. 10: a) Mecy.bmp encrypted by: b) HCM-PRE, c) AES.

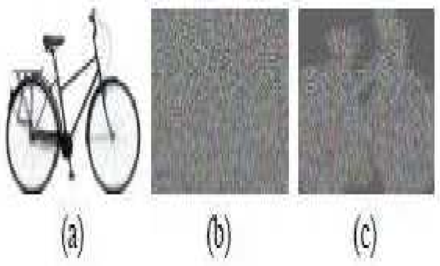

Fig. 11: a) Bicycle.bmp encrypted by: b) HCM-PRE, c) AES.

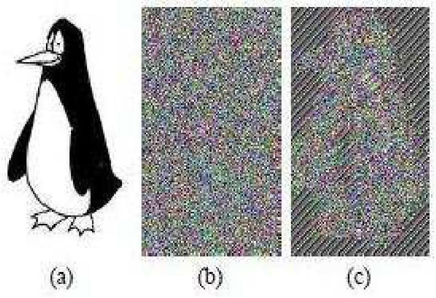

Fig. 12: a) Penguin.bmp encrypted by: b) HCM-PRE, c) AES.

data patterns for the images contain large single colour areas (Mecy.bmp: Fig. 10, Bicycle.bmp: Fig. 11, and Penguin.bmp: Fig. 12). That is, the proposed HCM-PRE has advantage in encryption of identical plaintext blocks over the AES.

The numerical evaluation for encryption quality measure ID of the HCM-PRE and AES is given in Table 4.

References

[1] L. Hill, Cryptography in an Algebraic Alphabet, American Mathematical Monthly, **36**, 306-312 (1929).
[2] L. Hill, Concerning Certain Linear Transformation Apparatus of Cryptography, American Mathematical Monthly, **38**, 135-154 (1931).
[3] J. Overbey, W. Traves and J. Wojdylo, On the Key Space of the Hill Cipher, Journal of Cryptologia, **29**, 59-72 (2005).
[4] S. Saeednia, How to Make the Hill Cipher Secure, Journal of Cryptologia, **24**, 353-360 (2000).
[5] W. Stallings, Cryptography and Network Security Principles and Practices (4th edn.). Prentice Hall: New Jersey (2006).
[6] C. Lin, C. Lee and Y. Lee, Comments on Saeednias Improved Scheme for the Hill Cipher, Journal of the Chinese Institute of Engineers, **27**, 743-746 (2004).
[7] A. Chefranov, Secure Hill Cipher Modification SHC-M, Proc. of the First International Conference on Security of Information and Networks (SIN2007) Gazimagusa (TRNC) North Cyprus, Eli, A., Ors, B., and Preneel, B. (Eds.) Trafford Publishing, Canada. 34-37 (2007).
[8] A. Ismail, M. Amin and H. Diab, How to Repair the Hill Cipher. J. Zhejiang Univ Sci. A, **7**, 2022-2030 (2006).
[9] Y. Romero, et al., Comments on How to Repair the Hill Cipher, Journal of Zhejiang University Science, **A 9**, 211-214 (2007).
[10] C. Li, D. Zhang, and G. Chen, Cryptanalysis of an Image Encryption Scheme Based on the Hill Cipher, Journal of Zhejiang University Science A, **9**, 1118-1123 (2008).
[11] T. Mohsen, F. Abolfazl, A Secure Cryptosystem Based on Affine Transformation. John Wiley & Sons, **4**, 207-215 (2011).
[12] R. Rivest, The MD5 Message-Digest Algorithm. Internet RFC 1321, April (1992).
[13] Federal Information Processing Standard (FIPS) 180-2. Secure Hash Standard, NIST, U. S, Department of Commerce, (2002).
[14] A. Mahmoud, A. Chefranov, Hill Cipher Modification Based on Eigenvalues HCM-EE, Proc. of the Second International Conference on Security of Information and Networks (SIN2009); Gazimagusa (TRNC) North Cyprus, Elci, A., Orgun, M., and Chefranov, A. (Eds.) ACM, New York, USA, **2009**, 164- 167 (2009).
[15] W. Galvin, Matrices with Custom-Built Eigenspaces. this MONTHLY, **91**, 308-309 (1984).
[16] B. Schneier, Applied cryptography: Protocols, Algorithms, and Source Code in C (2nd edn), John Wiley & Sons: New York, (1996)
[17] T. Apostol, Introduction to Analytic Number Theory Springer, (1976).
[18] M. Robshow, Stream Ciphers. RSA Laboratories Technical Report TR-701, July (1995) http://www.rsasecurity.com/rsalabs/index.html.
[19] I. Ziedan, M. Fouad and H. Salem, Application of Data Encryption Standard to Bitmap and JPEG Images. Proc. Twentieth National Radio Science Conference (NRSC), Egypt, 1-16 (2003).
[20] H. Elkamchouchi, A. Makar, Measuring Encryption Quality of Bitmaps Images with Rijndael and KAMKAR Block Ciphers. Proc. Twenty Second National Radio Science Conference (NRSC), Egypt, 1-8 (2005).
[21] E. Hossam, H. Ahmed and S. Osama, An Efficient Chaos-Based Feedback Stream Cipher (ECBFSC) for Image Encryption and Decryption. Journal of Computing and Informatics, **31**, 121-129 (2007).
[22] M. Yaobin, C. Guanrong, Chaos-Based Image Encryption in Eduardo Bayro-Corrochano, editor, Handbook of Computational Geometry for Pattern Recognition, Computer Vision, Neural Computing and Robotics. Springer-Verlag, Heidelberg, (2004).
[23] M. Yaobin, C. Guanrong and L. Shiguo, A Symmetric Image Encryption Scheme Based on 3D Chaotic Cat Maps. Chaos Solitons and Fractals, **21**, 749-761 (2004).
[24] G. Alvarez, S. Li, Some Basic Cryptographic Requirements for Chaos-Based Cryptosystems. International Journal of Bifurcation and Chaos, **16**, 2129-2151 (2006).
[25] E. Hossam, H. Ahmed and S. Osama, Encryption Effeciency Analysis and Security Evaluation of RC6 Block Ciphers for Digital Images. International Journal of Computer and Information Technology, **1**, 33-39 (2007).
[26] C. Shannon, Communication Theory of Secrecy Systems. Bell Systems Technical Journal, **28**, 656-715 (1948).
[27] H. Cheng, L. Xiaobo, Partial Encryption of Compressed Images and Videos. IEEE Trans. Signal Process, **48**, 2439-2451 (2000).
[28] L. Marvel, G. Boncelet and C. Retter, Spread Spectrum Image Stegnography. IEEE Trans. Signal Process, **8**, 1075-1083 (1999).
[29] S. Lian, J. Sun and Z. Wong, Security Analysis of Chaos-Based Image Encryption Algorithm. Phys Lett A, **351**, 645-661 (2005).

Ahmed Mahmoud received a BSc. in Computer Science from Al-Zaytoonah University, Amman, Jordan in 1997, an MSc. in Applied Mathematics and Computer Science and PhD in Computer Engineering from Eastern Mediterranean University, North Cyprus, in 2001 and 2012, respectively. From 2001 to 2006 he was a Lecturer in the Computer Science Department at Al-Azhar University, Gaza Strip, Palestine. From September 2006 to September 2011 he was with the Computer Engineering Department at Eastern Mediterranean University, Famagusta, North Cyprus. Since January 2012 he has been an Assistant professor in the Information Technology Department at Al-Azhar University, Gaza Strip, Palestine. His research interests are in the areas of information security, discrete geometry, parallel programming, and distributed systems

Alexander Chefranov received a diploma of Engineer in Applied Mathematics, PhD, and DSc from Taganrog State Radio-Engineering University, Russia, in 1978, 1984, and 1998, respectively. Since 1999 he has been a Professor of Software Engineering Department of the Institute of Technology, South Federal University, Taganrog, Russia, and, since 2002, he has been an Associate Professor of the Department of Computer Engineering, Eastern Mediterranean University, Famagusta, North Cyprus. His research interests are in the areas of information security, parallel processing, real-time systems, database management systems, and scientific computing.