PES University, Bangalore

(Established under Karnataka Act No. 16 of 2013)

MAY 2020: IN SEMESTER ASSESSMENT (ISA) B.TECH. IV SEMESTER

UE18MA251- LINEAR ALGEBRA

MINI PROJECT REPORT

ON

IMAGE ENCRYPTION CYPHER

Submitted by

1.	**A Anantha Krishna**	**PES1201800506**
2.	**K Venkat Ramnan**	**PES1201801319**
3.	**Vishasagar Udupi**	**PES1201800534**

Branch & Section : ECE - B

PROJECT EVALUATION

(For Official Use Only)

Sl.No.	Parameter	Max Marks	Marks Awarded
1	Background & Framing of the problem	4	
2	Approach and Solution	4	
3	References	4	
4	Clarity of the concepts & Creativity	4	
5	Choice of examples and understanding of the topic	4	
6	Presentation of the work	5	
	Total	25	

Name of the Course Instructor : RENNA SULTANA

Signature of the Course Instructor :

INTRODUCTION:

cryptography
/krɪpˈtɒgrəfi/

noun

the art of writing or solving codes.

In the recent years , with the emergence of the newer technologies in the field of computer science most of the work which was earlier done by man has now been shifted into the hands of the computers. While as much as a boon this may serve to mankind there is a serious problem faced by these newer technologies these days ie: security . Yes , security is the major concern which was overlooked by many companies in the early decades and thus made them pay a heavy price. Every technology or software created , which requires the data provided by the user needs to store those data provided by the user in a safe and secure way which cannot be broken through by people and be exploited in any way. So to overcome this problem many security methods were created which helped to secure the data provided by the user.

The Inspiration of this project was drawn from an article my team and I read which stated various applications of Hill Cypher. We were motivated to pursue something in this fashion and create our own Cypher and thus this is the Product.

In this Project report we aim to focus on how Linear Algebra can be used to encrypt images while maintaining the accuracy of the final product to up to 95%.

We will elaborate on how images can be stored as matrices and how we can manipulate said matrices to encrypt the image while keeping all the data intact.

To conclude, the decryption algorithm and the future scope will be laid out in detail.

REVIEW OF LITERATURE:

1. **The Hill Cypher: [1]**

The stages of the Hill Cipher encryption algorithm are as follows [15]:

1. Organize character alphabetically with numeric A → 1, B → 2, ..., Z → 26 or in ASCII (256 characters)
2. Create a key matrix measuring m x m

$$K_{mxm} = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1m} \\ k_{21} & k_{22} & \cdots & k_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ k_{m1} & k_{m2} & \cdots & k_{mm} \end{bmatrix}$$

3. Matrix K is an invertible matrix that has multiplicative inverse K−1 so that K. K−1 = 1
4. Plaintext P = p1 p2 ... pn, blocked with the same size as the row or column column K

$$P_{qxm} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ p_{q1} & p_{q2} & \cdots & p_{qm} \end{bmatrix}$$

5. Transpose matrix P

$$P^t_{mxq} = \begin{bmatrix} p_{11} & p_{21} & \cdots & p_{1q} \\ p_{12} & p_{22} & \cdots & p_{2q} \\ \cdots & \cdots & \cdots & \cdots \\ p_{1m} & p_{2m} & \cdots & p_{qm} \end{bmatrix}$$

6. Multiply matrix K with transposed P in modulo 26 or 256

$$C^t = K_{mxm} P^t_{mxq}$$

$$C^t = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1m} \\ k_{21} & k_{22} & \cdots & k_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ k_{m1} & k_{m2} & \cdots & k_{mm} \end{bmatrix} \begin{bmatrix} p_{11} & p_{21} & \cdots & p_{1q} \\ p_{12} & p_{22} & \cdots & p_{2q} \\ \cdots & \cdots & \cdots & \cdots \\ p_{1m} & p_{2m} & \cdots & p_{qm} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{21} & \cdots & c_{m1} \\ c_{12} & c_{22} & \cdots & c_{m2} \\ \cdots & \cdots & \cdots & \cdots \\ c_{1q} & c_{2q} & \cdots & c_{mq} \end{bmatrix}$$

7. Then transpose to

$$C = (C^t)^t = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1q} \\ c_{21} & c_{22} & \cdots & c_{2q} \\ \cdots & \cdots & \cdots & \cdots \\ c_{m1} & c_{m2} & \cdots & c_{mq} \end{bmatrix}$$

8. *Change the result of step 7 into the alphabet using alphabetical correspondence with numeric in step 1 to obtain the ciphertext.*

2. **Caeser Cypher: [2]**

The Caesar Cipher technique is one of the earliest and simplest method of encryption technique. It's simply a type of substitution cipher, i.e., each letter of a given text is replaced by a letter some fixed number of positions down the alphabet. For example with a shift of 1, A would be replaced by B, B would become C, and so on. The method is apparently named after Julius Caesar, who apparently used it to communicate with his officials.

Thus to cipher a given text we need an integer value, known as shift which indicates the number of position each letter of the text has been moved down.

The encryption can be represented using modular arithmetic by first transforming the letters into numbers, according to the scheme, A = 0, B = 1,..., Z = 25. Encryption of a letter by a shift n can be described mathematically as:

$$E_n(x) = (x + n) mod\ 26$$

(Encryption Phase with shift n)

$$D_n(x) = (x - n) mod\ 26$$

(Decryption Phase with shift n)

METHODOLOGY:

Data Matrix

[m x n] Rectangular Matrix that holds non-negative values in it.

$$\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn}$$

Key Matrix

[m x n] Rectangular Matrix that holds non-negative values in it.

$$\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn}$$

*NOTE: The Data Matrix and the Key Matrix have to be the same dimensions

Operations to get the Encryptor

The Transpose of the Key Matrix is supposed to be multiple with a unit Column matrix of the appropriate dimension. The resultant matrix is called a buffer matrix, this matrix is transposed and the first row is duplicated m times so as to get an (m x n) matrix. This matrix is called the Encryptor.

1. $$\begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{nxm} X \begin{bmatrix} \cdot \\ \cdot \\ \cdot \\ \cdot \end{bmatrix}_{mx1} = \begin{bmatrix} \cdot \\ \cdot \\ \cdot \\ \cdot \end{bmatrix}_{nx1}$$

Key matrix transpose	Column Matrix	Buffer Matrix

2. $$[\cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot]_{1xn} \longrightarrow \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn}$$

Buffer Matrix Transpose

Duplicating the first row m Times

Encryptor

The Encrypted Matrix

$$\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn} - \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn}$$

The Data Matrix — The Encryptor

This operation results in the encrypted

$$\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn}$$

Encrypted Matrix

Operations to get the Decrypter

1. $\begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{nxm}$ X $\begin{bmatrix} \cdot \\ \cdot \\ \cdot \end{bmatrix}_{mx1}$ =. $\begin{bmatrix} \cdot \\ \cdot \\ \cdot \end{bmatrix}_{nx1}$

Key matrix transpose — Column Matrix — Buffer Matrix

2. $[\cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot]_{1xn} \longrightarrow \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn}$

Buffer Matrix Transpose — Duplicating the first row m Times — Decrypter

The Decrypted Matrix

$$\begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn} + \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}_{mxn}$$

The Data Matrix — The Decrypter

This operation results in the Decrypted Matrix

PROOF OF CONCEPT:

We executed the above algorithm in Matlab to present proof that this type of secret key encryption works.

1. **Initialise a rectangular data Matrix and key Matrix**

```
>>  original_matrix= randi([1,4],[5,7])

original_matrix =

   3   1   1   3   4   3   1
   2   2   2   2   3   3   4
   3   2   4   3   3   2   4
   3   2   1   1   1   2   4
   1   3   4   2   2   4   1
```

```
>> key_matrix=randi([1,4],[5,7])

key_matrix =

   2   1   4   1   3   1   1
   2   3   4   3   3   1   4
   3   2   2   3   4   4   3
   1   4   3   2   4   1   3
   3   3   1   4   3   2   2
```

2. **Transpose the Key Matrix**

```
>> key_transpose= key_matrix.'

key_transpose =

   2   2   3   1   3
   1   3   2   4   3
   4   4   2   3   1
   1   3   3   2   4
   3   3   4   4   3
   1   1   4   1   2
   1   4   3   3   2
```

3. **Multiply the Transpose of the Key Matrix with a unit Column Matrix**

   We take this idea from Eigen vectors concept of Linear Algebra where we multiply the Matrix with a unit matrix to get the sum of all the rows but here we have transposed the matrix so as to get sum of all the columns

```
>> buffer_matrix=key_transpose*unit_matrix

buffer_matrix =

   11
   13
   14
   13
   17
    9
   13
```

4. **Transpose said matrix Increase the dimension by duplicating the first row m times to get the encryptor**

```
encryptor =

   11   13   14   13   17    9   13
   11   13   14   13   17    9   13
   11   13   14   13   17    9   13
   11   13   14   13   17    9   13
   11   13   14   13   17    9   13
```

5. **Form the encrypted matrix by Subtracting the Encryptor from the Data Matrix**

```
>>  Encrypted_matrix=original_matrix - encryptor

Encrypted_matrix =

   -8  -12  -13  -10  -13   -6  -12
   -9  -11  -12  -11  -14   -6   -9
   -8  -11  -10  -10  -14   -7   -9
   -8  -11  -13  -12  -16   -7   -9
  -10  -10  -10  -11  -15   -5  -12
```

6. **Get the Decryptor matrix using the same algorithm**

```
>> buffer_matrix_decryptor_transpose=buffer_matrix_decryptor.'

buffer_matrix_decryptor_transpose =

    11   13   14   13   17    9   13

>> decryptor=[11   13   14   13   17    9   13
;11   13   14   13   17    9   13
;11   13   14   13   17    9   13
;11   13   14   13   17    9   13
;11   13   14   13   17    9   13
]
```

7. **Obtain the reconstructed matrix by adding decryptor to Encrypted and as the result will show the data matrix is same as the reconstructed matrix.**

```
>> solution = Encrypted_matrix+decryptor

solution =

   3   1   1   3   4   3   1
   2   2   2   2   3   3   4
   3   2   4   3   3   2   4
   3   2   1   1   1   2   4
   1   3   4   2   2   4   1
```

ENCRYPTING IMAGES: -[3],[4]

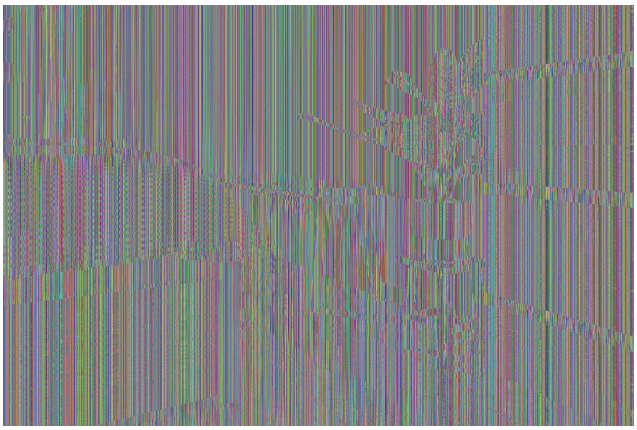
Images are nothing but a huge matrices whose each cell is a pixel in the image. Each cell stores the RGB value of each pixel and thus due to this property images can basically be understood as very large matrices with numbers in them.

Now as we have understood images to be large matrices, we apply the same Algorithm which has been explained in the above sections in detail we can perform a form of cryptography known as secret key cryptography which is best used when privacy and confidentiality is needed.

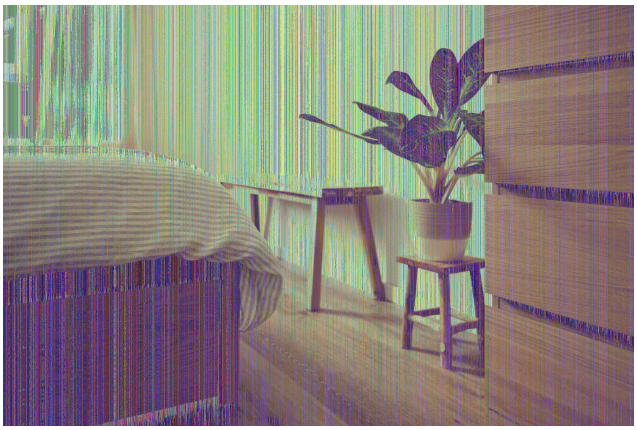
Here the sender and receiver already have an established key image (key matrix), the sender encrypts the image he wants to send with this algorithm using that specific key and once the encoded image is computed, he sends it. The receiver gets the encoded image which at first looks like noise but, when the decryption algorithm is applied the original image gets reconstructed without the account for loss during transmission, encoding and decoding.

Data Image

Key Image



Encrypted Image



Reconstructed Image

**NOTE: All the noise in the reconstructed image is due to data loss while doing the matrix manipulation and during the runtime of the code as it is a very large matrix [3808 x 5712]

RESULTS AND DISCUSSIONS:

- **Matrices can be manipulated** in many ways so as to get desired information.
- Images are **nested matrices in which the RGB** values of each pixel is stored.
- **N - number of cyphers** can be written to produce one's desired effect.
- The data image was encrypted and reconstructed successfully using the Algorithm explained earlier in this report.

SUMMARY AND CONCLUSIONS:

- Matrices can be manipulated in any way but should be done conservatively keeping in mind that all the **manipulations must be reversible when the same is done in reverse.**
- Images can be tricky to work with once it is being shown in a matrix form, hence utmost care should be given while manipulating the RGB values or we might end up corrupting the Data Image itself.
- N-number of Cyphers can be written is true if and only if the matrix is a **non-singular matrix** or else the resulting image will be nothing but a black screen.
- The Algorithm can be used for **any matrix as long as it is a non Singular matrix** and if the image is not very noisy.
- **The Future Scope of this project would be reducing the noise so as to obtain clear reconstructed image and to expand this cypher onto other data types.**

BIBLOGRAPHY:

[1] : ***Application of Hill Cipher Algorithm in Securing Text Messages -***

Muhammad Donni Lesmana Siahaan , Andysah Putera Utama Siahaan

A Secure Encryption Technique based on Advanced Hill Cipher For a Public Key Cr.-
Suman Chandrasekhar, Akash H.P ,Adarsh.K, Mrs.Smitha Sasi

[2]: **Caesar Cypher in Cryptography-**

Geeks for Geeks

[3]: ***Image Encryption using Key Matrix Generation and Lossless Compression -***

Natasha D'Costa , Anusha Pai

[4]: **Debugging queries along the way of writing the code for image encryption-**

www.stackoverflow.com