Abstract

Technology today evolves at a faster rate than it could have been predicted. The benefits of the these rapid changes has not diminished the concerns for security of data. The paper deals with implementation of security measures for images based on the principles of Rubik’s cube. Here the image is encrypted by scrambling of pixels and performing XOR operations and it is decrypted similarly. This technique enables a better procedure for ensuring security for the image files that are being transmitted literally every second. The algorithm has been tested on different kinds of images and the obtained results have been presented.

Keywords: Encryption, Decryption, Rubik’s Cube.

1. INTRODUCTION

Images have been the fundamental source of conveying ideologies right from the start of mankind which stands true even today. In this day and age of accelerated and convenient media for transmission and use of data resources the need for security has never been more profound. This has led to the development and analysis of several encrypting algorithms that come with their own distinct characteristics.

In recent years, several encryption schemes have been proposed [1]–[12]. Image encryptions are broadly classified into three types. They are
1. Pixel Transformation
2. Value Transformation
3. Chaotic Transformation

Value Transformation:
Based on Intensity Hue Saturation (IHS), the transformation of pixel can be done. The total image is divided into different color spaces and each color space is encrypted by using the different approach.

Pixel Transformation:
The transformation is possible by using different mechanisms such as Peano-Hilbert curves and permutation diffusion architecture. These types of processes may use the pixel transformation or pixel shuffling, depends on the criteria.

Chaotic Transformation:
The origin for this one is pixel transformation only. It is novel pixel shifting method which generates chaotic sequences and is used as an encryption sequences. It used 3D cat map to shuffle pixels within image. The paper presents encryption techniques for secure transfer of images from one port to another. It has been designed based on the Principles established by Rubik’s cube. The technique involves encrypting the image by dividing into a sequence of byte arrays and applying XOR operations to obtain a scrambled image. The image is decrypted by taking the same sequence of byte arrays and applying the before mentioned techniques to obtain the original image.

2. RUBIK’S CUBE PRINCIPLE AND PROCEDURE

The very concept of Rubik’s cube depends on rows and columns. The principle is applied to rows and these rows rotated according to the requirement. The image is divided into rows and each row pixel transformation is applied.

2.1 Rubik’s Cube Image Encryption

In this section, the proposed encryption algorithm based on Rubik’s cube principle is described along with the decryption algorithm.

2.1.1 Rubik’s Cube Based Encryption Algorithm

Let Io represent a α-bit image of the size M × N. Here, Io represents the pixels values matrix of image Io. The steps of encryption algorithm are as follows:

1) Generate randomly two byte arrays Br and Bk of variable length.
(2) Store all the pixel values of a selected image in a byte array \( B_R \)
(3) Divide the byte array \( B_R \) into three parts and store it in three different byte arrays.
   (a) The first byte array \( B_1 \) will store pixel values from 0 to one-third part of \( B_R \)
   (b) The second byte array \( B_2 \) will store pixel from the end of \( B_1 \) till one-sixth of \( B_R \)
   (c) The third byte array \( B_3 \) will store remaining pixel from end of \( B_2 \) till end of \( B_R \)
(4) For each byte array \( i \) of image \( I_0 \),
   (a) Compute the sum of all elements in the byte array \( i \), the sum is denoted by \( \alpha(i) \)
   \[
   \alpha(i) = \sum_{j=1}^{N} I_0(i, j), \quad i = 1, 2, \ldots, M, \quad (1)
   \]
   (b) Compute modulo2 of \( \alpha(i) \), denoted by \( M_{\alpha(i)} \)
   (C) Byte array \( i \) is left or right shifted depending on the \( M_{\alpha(i)} \)
      If \( M_{\alpha(i)} = 0 \) \( \rightarrow \) right circular shift
      else \( \rightarrow \) left circular shift
(5) Using byte array \( B_K \), the bitwise XOR operator is applied to first and third byte array of the scrambled image.
(6) Combine all the byte arrays \( B_1, B_2, B_3 \) into another byte array \( I_{enc} \) in order to obtain an encrypted image.

2.1.2 Rubik’s cube decryption Algorithm.
The decrypted image is \( I_{dec} \) is recovered from the encrypted image \( I_{enc} \). The steps for decryption algorithm is
(1) Divide the byte array \( I_{enc} \) into three parts and store it in three different byte arrays.
   (a) The first byte array \( B_1 \) will store pixel values from 0 to one-third part of \( B_R \)
   (b) The second byte array \( B_2 \) will store pixel one sixth part of \( B_R \)
   (c) The third byte array \( B_3 \) will store remaining pixel values of \( B_R \)
(2) Using byte array \( B_K \), the bitwise XOR operator is applied to first and third byte array of the scrambled image.
(3) For each byte array \( i \) of image \( I_0 \),
   (a) Compute the sum of all elements in the byte array \( i \), the sum is denoted by \( \alpha(i) \)
   \[
   \alpha(i) = \sum_{j=1}^{N} I_0(i, j), \quad i = 1, 2, \ldots, M, \quad (1)
   \]
   (b) Compute modulo2 of \( \alpha(i) \), denoted by \( M_{\alpha(i)} \)
   (c) Byte array \( i \) is left or right shifted depending on the \( M_{\alpha(i)} \)
      If \( M_{\alpha(i)} = 0 \) \( \rightarrow \) right circular shift
      else \( \rightarrow \) left circular shift
(4) Combine all the byte arrays \( B_1, B_2, B_3 \) into another byte array \( I_{dec} \) in order to obtain an original image.

3. RESULT AND ANALYSIS
In this section, we present the tests that were conducted to assess the efficiency and security of the proposed image encryption algorithm. These tests involve visual testing and security analysis [13].

3.1 Visual Testing
For visual testing we test by using different types of images with different sizes. These selected images will be encrypted using our proposed Rubik’s cube encryption. From this we can observe the difference in pixel position of the encrypted images and original images shown in figure 1(a) to figure 1(f). There will be a drastic change between the encrypted and original images.

3.2 Analyzing Security
Security is a major concern in any image encryption. There are certain major methods that will help in breaking the image encryption. Methods such as Brute force, Plain text and cipher text attack etc are some of the major methods for breaking the encryption. The assessment of these security issues can be done mainly by KEY SPACE and KEY SENSIBILITY.

3.2.1 Key Space
A brute force attack is an attack that basically works on guessing. So practically speaking, larger the key space, the harder it will be the for the brute force to break the encryption. In our proposed Rubik’s cube image encryption, a brute force
attack is practically impossible because the key space huge. The key which we use to encrypt the image is multiplied by fifteen times. Since the key space is too big to break using brute force our image encryption is very secure.

3.2.2 Key Sensibility
A good encryption should always have a high sensibility feature. Which means any small change should lead to a huge difference in the encrypted image. In our encryption the each pixel’s position depends on each byte present in the secret key which we provide for encryption shown in figure 2(a) to figure 2(c). Since there will be a huge change in the image for every small change in the key, key sensibility is very high in our encryption.

![Figure 1(a): Leena (original)](image1.png)
![Figure 1(b): Leena (encrypted) using key k1](image2.png)

![Figure 1(c): Penguins (original)](image3.png)
![Figure 1(d): Penguins (encrypted) using key k1](image4.png)

![Figure 1(e): Murali (original)](image5.png)
![Figure 1(f): Murali (encrypted) using key k1](image6.png)
3.3 Speed Test
One more important aspect that should be kept in mind is the speed at which the encryption of the image is done. Our proposed Rubik’s cube image encryption is very fast compared to other image encryptions. Our encryption’s average speed is less than a second when it runs on a normal Intel Core I3 (3.06 GHZ) Processor on a .Net framework 4.0. Speed test is performed on images with various resolutions, but the encryption was finished in less than a second.

4. CONCLUSION
This paper explains the encrypting and decrypting techniques using the principles of Rubik. Performance assessment tests demonstrate that the proposed image encryption algorithm is highly secure. It is also capable of fast encryption/decryption which is suitable for real-time Internet encryption and transmission applications. The before mentioned algorithms are efficient and have been repeatedly tested for concrete results, but, the ethical aspect of the solution must not be forgotten. The techniques mentioned above need to be implemented ethically to produce desired results without breaching any established security protocols.

References


AUTHOR

M.SIRISHA completed her graduation from Andhra University, Visakhapatnam. She is currently pursuing her final semester post graduation in MCA in ANITS Engineering college a premier private engineering college in Visakhapatnam.

SVVS LAKSHMI has received her MCA from Andhra University, Visakhapatnam. Presently she is perusing her MTECH in Andhra University. She is working as Assistant Professor in ANITS Engineering College, Visakhapatnam.