Architectural Framework of Image Cryptography by Hybrid Approach of Logistic Map and Cheat Image

Deepak Pant[*], Manish Madhav Tripathi[**], Sonali Yadav[**]

[*] Student, Computer Science and Engineering, Integral University, Lucknow
[**] Associate Professor, Computer Science and Engineering, Integral University, Lucknow

Abstract- This paper proposes a novel confusion and diffusion algorithm for image encryption based on logistic map and cheat image. We choose the initial condition and control parameter of logistic map as the secret key. The cheat image selected from the most common images in public network, together with the chaotic matrices generated by logistic maps, is employed both in encryption and decryption processes to encrypt and recover the plain image. One cheat image can be used to encrypt a great number of plain images if the cheat image does not attract the attention of the attackers. The computer experiments such as statistical analysis, sensitivity analysis, differential attack analysis and cheat characteristic analysis, prove that the proposed image encryption algorithm is robust and secure enough to be used in practical communication.

Index Terms- image encryption; logistic map; cheat image;confusion;diffusion

I. Introduction

Recently images are widely used and the security of the information which images bring is becoming a serious problem. With the rapid development of computer science and network technology, people are obtaining, using and processing digital images more frequently. This situation brings us convenience, as well as potential threats. How to protect the information within the digital images from the attacks of intruders is becoming a more and more serious problem. Encryption is a good solution of this problem.

Image encryption is the most secure solution for storing images on a web server. Encrypted images cannot be displayed without first being decrypted otherwise they cannot be displayed, making any images stored on a server not only secure from unauthorized linking by the public, but they are also secure from web hosting staff and your web master.

In this project we introduce a novel confusion and diffusion algorithm for image encryption based on logistic map and cheat image. An image encryption using encryption scheme is a novel method particularly in that it needs an assistant image called cheat image to fulfill the encryption and decryption processes. The cheat image may theoretically be any image, but those images frequently appeared in public network are expected to be cheat images which are easily neglected by attackers. For image encryption a secure scheme, the key space should be large enough to make the brute-force attack infeasible.

II. Software Requirement Specification

2.1 Matlab

MATLAB (matrix laboratory) is a numerical computing environment and fourth-generation programming language. Developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, and Fortran.

Syntax

The MATLAB application is built around the MATLAB language. The simplest way to execute MATLAB code is to type it in the Command Window, which is one of the elements of the MATLAB Desktop. When code is entered in the Command Window, MATLAB can be used as an interactive mathematical shell.

Variables

Variables are defined with the assignment operator, =. MATLAB is a weakly dynamically typed programming language. It is a weakly typed language because types are implicitly converted.For example:

```
>> x = 17
x = 17
>> x = 'hat'
x =hat
>> y = x + 0
y =      104       97      116
>> x = [3*4, pi/2]
x =   12.0000    1.5708
>> y = 3*sin(x)
y =   -1.6097    3.0000
```

MATLAB has several functions for rounding fractional values to integers:

- round(X): round to nearest integer, trailing 5 rounds to the nearest integer away from zero. For example, round (2.5) returns 3; round (-2.5) returns -3.
- fix(X): round to nearest integer toward zero (truncate). For example, fix(2.7) returns 2; fix(-2.7) returns -2
- floor(X): round to the nearest integer toward minus infinity (round to the nearest integer less than or equal to X). For example, floor (2.7) returns 2; floor (-2.3) returns -3.

- ceil(X): round to the nearest integer toward positive infinity (round to the nearest integer greater than or equal to X); for example, ceil(2.3) returns 3; ceil(-2.7) returns -2

Vectors/matrices

MATLAB is a "Matrix Laboratory", and as such it provides many convenient ways for creating vectors, matrices, and multi-dimensional arrays. In the MATLAB vernacular, a *vector* refers to a one dimensional (1×*N* or *N*×1) matrix, commonly referred to as an array in other programming languages. MATLAB provides a simple way to define simple arrays using the syntax: *init*: *increment: terminator*. For instance:

```
>> array = 1:2:9
array = 1 3 5 7 9
```

defines a variable named array (or assigns a new value to an existing variable with the name array) which is an array consisting of the values 1, 3, 5, 7, and 9. That is, the array starts at 1 (the init value), increments with each step from the previous value by 2 (the increment value), and stops once it reaches (or to avoid exceeding) 9 (the terminator value).

```
>> array = 1:3:9
array = 1 4 7
```

the increment value can actually be left out of this syntax (along with one of the colons), to use a default value of 1.

```
>> ari = 1:5
ari = 1 2 3 4 5
```

assigns to the variable named ari an array with the values 1, 2, 3, 4, and 5, since the default value of 1 is used as the incrementer.
The list of elements should be surrounded by square brackets: []. Parentheses: () are used to access elements and subarrays (they are also used to denote a function argument list).

```
>> A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
   A = 16 3 2 13
       5 10 11 8
     9 6 7 12
     4 15 14 1

>> A (2, 3)
ans = 11
```

Sets of indices can be specified by expressions such as "2:4", which evaluates to [2, 3, 4]. For example, a submatrix taken from rows 2 through 4 and columns 3 through 4 can be written as:

```
>> A (2:4, 3:4)
ans = 11 8
 7 12
 14 1
```

A square identity matrix of size n can be generated using the function eye, and matrices of any size with zeros or ones can be generated with the functions zeros and ones, respectively.

```
>> eye (3)
ans =
 1 0 0
 0 1 0
 0 0 1
>> zeros (2, 3)
ans =
 0 0 0
 0 0 0
>> ones (2, 3)
ans =
 1 1 1
 1 1 1
```

Most MATLAB functions can accept matrices and will apply themselves to each element. For example, mod (2*J, n) will multiply every element in "J" by 2, and then reduce each element modulo "n". MATLAB does include standard "for" and "while" loops, but using MATLAB's vectorized notation often produces code that is easier to read and faster to execute. This code, excerpted from the function *magic.m*, creates a magic square *M* for odd values of *n* (MATLAB function *meshgrid* is used here to generate square matrices I and J containing 1: n).

```
[J, I] = meshgrid (1: n);
A = mod (I+J-(n+3)/2, n);
B = mod (I+2*J-2, n);
M = n*A + B + 1;
```

III. Implementation

3.1 Introduction:

The implementation phase of any project development is the most important phase as it yields the final solution, which solves the problem at hand. The implementation phase involves the actual materialization of the ideas, which are expressed in the analysis document and developed in the design phase. Implementation should be perfect mapping of the design document in a suitable programming language in order to achieve the necessary final product. The factors concerning the programming language and platform chosen are described in the next couple of sections.

The implementation stage in a system project in its own right. It involves

- Careful planning
- Investigation of the current system and the constraints on implementation.
- Training of staff in the newly developed system.

3.2 Implementation:

Implementation of any software is always preceded by important decisions regarding selection of the platform, the language used, etc. There are three major implementation

decisions that have been made before the implementation of this project. They are as follows:

1. Selection of the platform (Operating System).
2. Selection of the programming language for development of the application.
3. Coding guideline to be followed.

3.3 Selection of the platform:

Windows® XP provides the most dependable version of Windows ever—with the best security and privacy features Windows has ever provided. Overall, security has been improved in Windows XP to help you Windows XP Professional includes all of the security capabilities of Windows XP Home Edition, plus other security management features. These important new security features will reduce your IT costs and enhance the security of your business systems. For example, if you use the Internet to chat online or to send and receive e-mail, you may be vulnerable to hacker attacks. To protect you from these threats.Let's take a look at the important security and privacy features in Windows XP Home Edition that make you and your information more secure while you're having the most productive Windows user experience ever.These features include access control lists (ACLs), security groups, and Group Policy—in addition to the tools that allow businesses to configure and manage these features.

Windows XP offers thousands of security-related settings that can be implemented individually.Businesses can apply these security templates when they:

- Create a resource, such as a folder or file share, and either accept the default access control list settings or implement custom access control list settings.
- Place users in the standard security groups, such as Users, Power Users, and Administrators, and accept the default ACL settings that apply to those security groups.
- Use the Basic, Compatible, Secure, and Highly Secure Group Policy templates that have been provided with the operating system.

Many of these tools, such as the Microsoft Management Console snap-ins, are components of Windows XP Professional. Other tools are included with the Windows XP Professional Resource Kit.

3.4 Selection of Language

For the implementation of our project we need flexible systems implementation language. Compilation should be relatively straightforward compiler, provide low-level access to memory, provide language constructs that map efficiently to machine instructions, and require minimal run-time support. Program should be compiled for a very wide variety of computer platforms and operating systems with minimal change to its source code. For Graphical User Interface programming, language chosen must be simple to uses, secure, architecture neutral and portable. Additional requirements of GUI are: 1) User interface management: Windows, menus, toolbars and other presentation components be supported by the language.2) Data and presentation management: language must contains a rich toolset for presenting data to the user and manipulating that data. 3) The Editor: The language should have a editor, a powerful and extensible toolset for building custom editors. 4) The Wizard framework: A toolset for easily building extensible, user-friendly Wizards to guide users through more complex tasks. 5) Configuration management: Rather than tediously write code to access remote data and manage and save user-configurable settings, etc., all of this is can be well handled by Matlab.

IV. SNAPSHOT

4.1 Transmitter

4.1.1 Start-up Page

Fig 4.1.1

4.1.2 Input Image

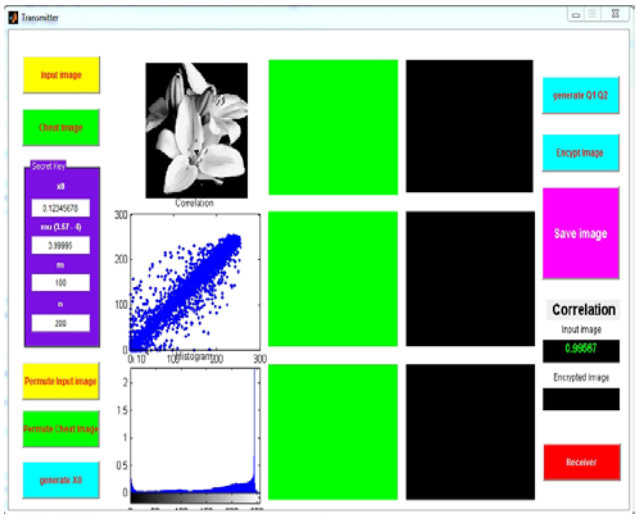

Fig 4.1.2

4.1.3 Cheat Image

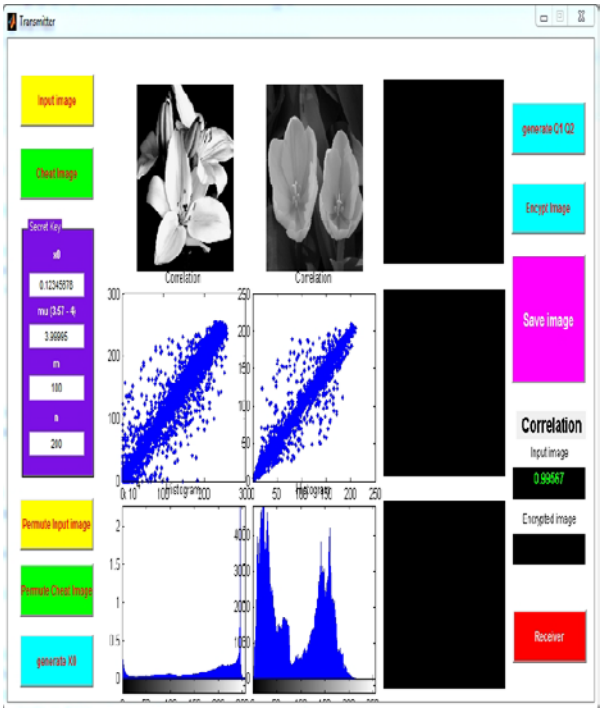

Fig 4.1.3

4.1.4 Permute Input Image

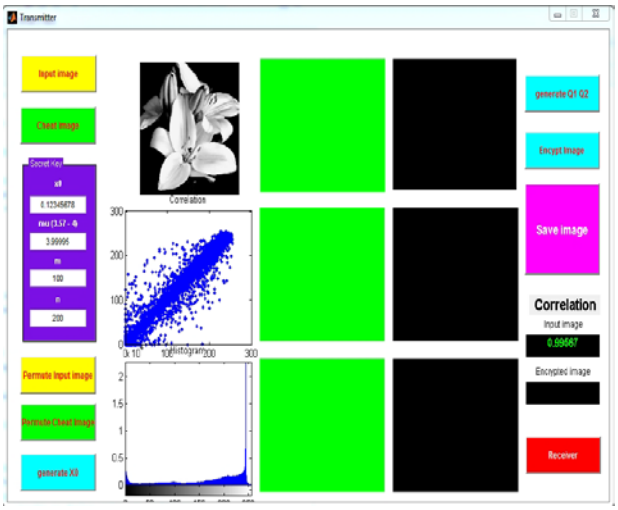

Fig 4.1.4

4.1.5 Generate X0

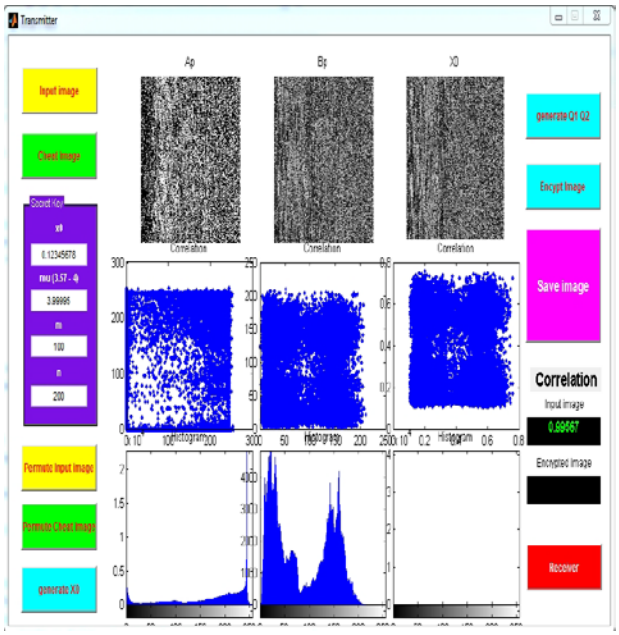

Fig 4.1.5

4.1.6 Generate Q1 and Q2

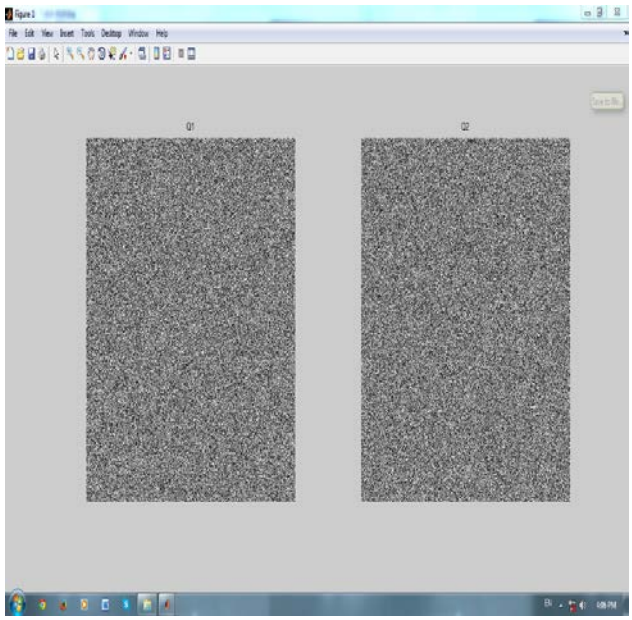

Fig 4.1.6

4.1.7 Encrypt the Image

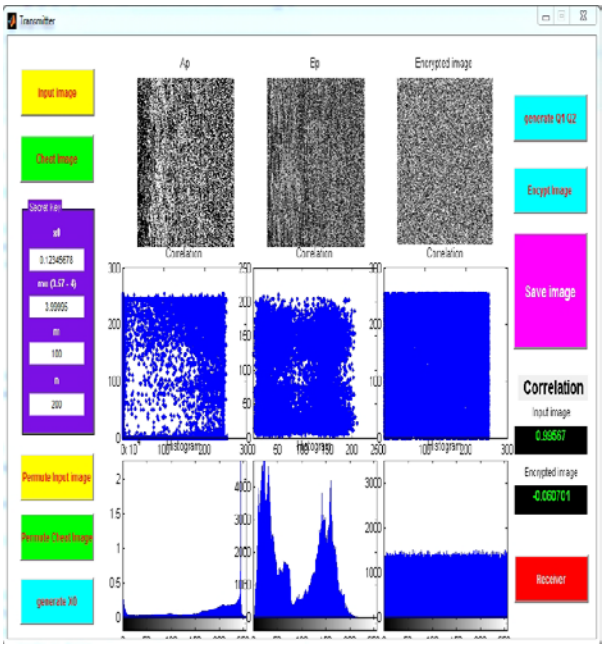

Fig 4.1.7

4.1.8 Save Encrypt Image

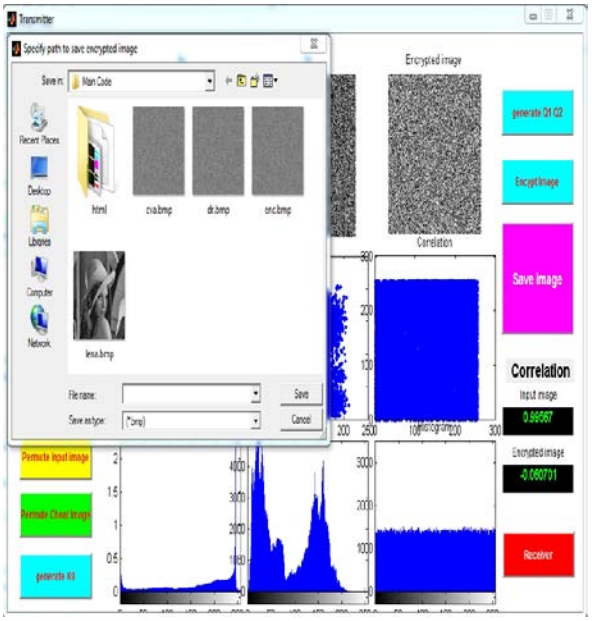

Fig 4.1.8

4.2 Receiver

4.2.1 Input Save Encrypted Image

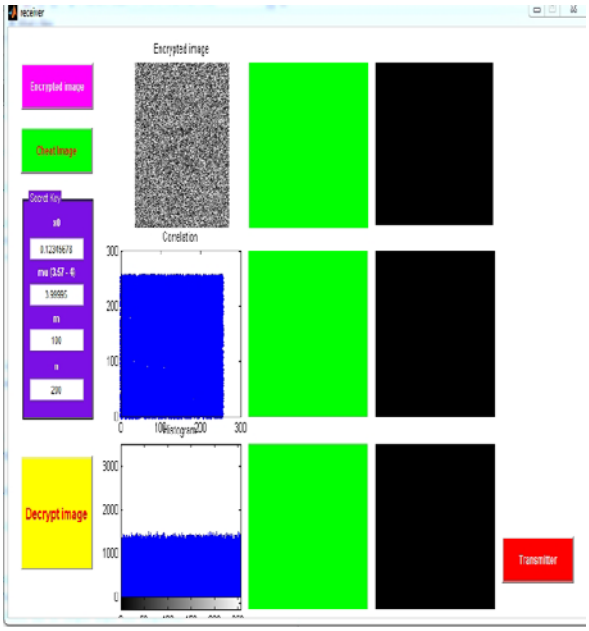

Fig 4.2.1

4.2.2 Input Cheat Image

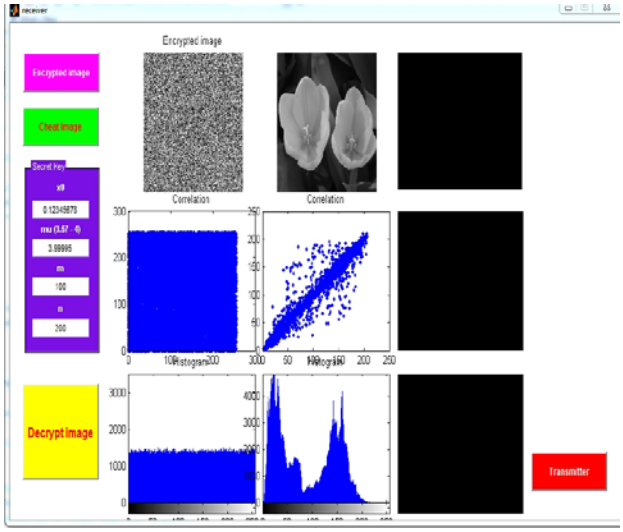

Fig 4.2.2

4.2.3 Decrypt Image

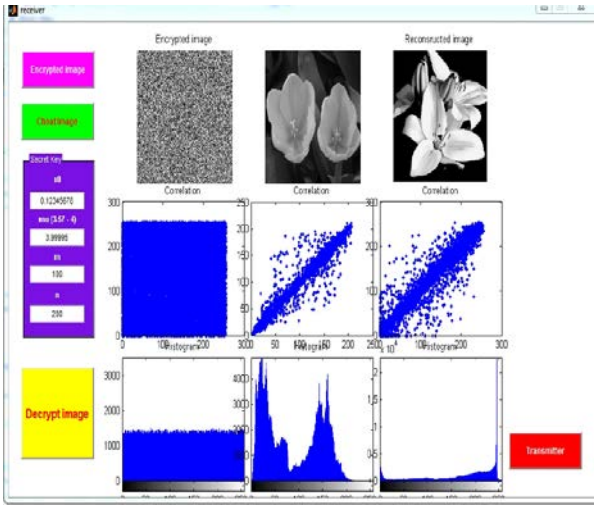

Fig 4.2.3

V. Conclusion

In this conclusion part we present an algorithm scheme have been proposed a novel confusion and diffusion algorithm for image encryption based on logistic map and cheat image. We choose the initial condition and control parameter of logistic map as the secret key. The cheat image selected from the most common images in public network, together with the chaotic matrices generated by logistic maps, is employed both in encryption and decryption processes to encrypt and recover the plain image. One cheat image can be used to encrypt a great number of plain images if the cheat image does not attract the attention of the attackers. The computer experiments such as statistical analysis, sensitivity analysis, and differential attack characteristics are discussed in detail to demonstrate that the proposed cryptosystem is robust and secure enough to be used in practical communication.

References

[1] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," Int. J. Bifur. Chaos, vol. 8(6), 1998, pp. 1259–1284.

[2] X. Tong and M. Cui, "Image encryption with compound chaotic sequence cipher shifting dynamically," Image. Vision. Comput, vol. 26(6), 2008, pp. 843–850.

[3] K. W. Wong, BS. H. Kwok and W. S. Law, "A fast image encryption scheme based on chaotic standard map," Phys. Lett. A, vol. 372(15), 2008, pp. 2645–2652.

[4] S. Lian, "Efficient image or video encryption based on spatiotemporal chaos system," Chaos. Soliton. Fract, vol. 40(5), 2009, pp. 2509–2519.

[5] S. Behnia, A. Akhshani, H. Mahmodi and A. Akhavan, "A novel algorithm for image encryption based on mixture of chaotic maps," Chaos. Soliton. Fract, vol. 35(2), 2008, pp. 408–419.

[6] N. K. Pareek, V. Patidar and K. K. Sud, "Discrete chaotic cryptography using external key," Phys. Lett. A, vol. 309(1–2), 2003, pp. 75–82.

[7] A. N. Pisarchik, N. J. Flores-Carmona and M. Carpio-Valadez, "Encryption and decryption of images with chaotic map lattices," Chaos, vol. 16(3), 2006, pp. 033118–033123.

[8] N. Pareek, V. Patidar and K. K. Sud, "Image encryption using chaotic logistic map," Image. Vision. Comput, vol. 24(9), 2006, pp. 926–934.

[9] A. N. Pisarchik and M. Zanin, "Image encryption with chaotically coupled chaotic maps," Physica D, vol. 237(20), 2008, pp. 2638–2648.

[10] G. Alvarez, F. Montoya, M. Romera and G. Pastor, "Cryptanalysis of a discrete chaotic cryptosystem using external key," Phys. Lett. A, vol. 319(3–4), 2003, pp. 334–339.

[11] D. Arroyo, R. Rhouma, G. Alvarez, S. Li and V. Fernandez, "On the security of a new image encryption scheme based on chaotic map lattices," Chaos, vol. 18(3), 2008, pp. 033112–033118.

[12] E. Solak and C. Cokal, "Comment on 'Encryption and decryption of images with chaotic map lattices'," Chaos, vol. 18(3), 2008, pp. 038101–038103.

[13] C. Li, S. Li, M. Asim, J. Nunez, G. Alvarez and G. Chen, "On the security defects of an image encryption scheme," Image. Vision. Comput, vol. 27(9), 2009, pp. 1371–1381.

[14] D. Arroyo, S. Li, J. M. Amigo, G. Alvarez and R. Rhouma, "Comments on 'image encryption with chaotically coupled chaotic maps'," Physica D, vol. 239(12), 2010, pp. 1002–1006.

Authors

First Author – Deepak Pant, Student, Computer Science and Engineering, Integral University, Lucknow, Email: deepakpant02@yahoo.com
Second Author – Manish Madhav Tripathi, Associate Professor, Computer Science and Engineering, Integral University, Lucknow, Email:- mmt@iul.ac.in
Third Author – Sonali Yadav, Tripathi, Associate Professor, Computer Science and Engineering, Integral University, Lucknow, Email:- sonali@iul.ac.in