

## ENKRIPSI CITRA WARNA MENGGUNAKAN *RUBIK'S CUBE* DAN *THREE CHAOTIC LOGISTIC MAP*

<sup>1</sup>Ronsen Purba, <sup>2</sup>Frans Agus, <sup>3</sup>Sari Fatmawati  
Program Studi Teknik Informatika STMIK Mikroskil

<sup>1</sup>ronsen@mikroskil.ac.id, <sup>2</sup>purba07frans@gmail.com, <sup>3</sup>sarimieckeyfatmawati@gmail.com

### Abstract

An image encryption is a technique to protect the image secrecy from illegal accessing. One of the encryption algorithm that are used is Rubik's Cube where this algorithm used to permute the pixel in the color image. However, the key used in the Rubik's Cube is not random so the security of image is still weak. To further improve the security of the image, then used the chaotic systems for generating random keys. Excess of chaos is the sensitivity to initial conditions, behave randomly and do not have a recurring period. One of the chaos function used is Three Chaotic Logistic Maps where necessary keys generated are used in encryption and decryption processes. The test results show that the algorithm Rubik's Cube and Three Chaotic Logistic Maps on the encryption and decryption processes do not cause an error in the original image and the result image. Neighboring pixels have a low coefficient indicating that the quality of the results encryption is good and resistant toward to attack by adding noise. The nature of chaotic sensitivity cause cipher image will not be recovered if the keys used in decryption are slightly different from the encryption keys. that the decrypted image does not revert to plain image so the image obtained more secure.

**Keywords:** *Image Encryption, Rubik's Cube, Three Chaotic Logistic Maps.*

### 1. Pendahuluan

Enkripsi citra merupakan teknik untuk melindungi kerahasiaan citra dari pengaksesan ilegal. Salah satu algoritma enkripsi yang digunakan adalah *Rubik's Cube* dimana algoritma ini menggunakan prinsip-prinsip kerja kubus rubik untuk mengacak susunan piksel pada citra warna [1]. Terdapat dua buah kunci yang diperlukan untuk proses permutasi piksel citra dan proses operasi XOR. Namun kunci yang digunakan pada proses XOR tidak acak sehingga keamanan citra masih lemah [2].

Untuk lebih meningkatkan keamanan citra, maka digunakan algoritma *Three Chaotic Logistic Maps* untuk membangkitkan kunci yang acak dimana kunci yang dibangkitkan diperlukan untuk proses XOR pada saat enkripsi dan dekripsi [3]. Algoritma ini dapat diterapkan karena sensitivitas terhadap nilai awal sehingga menghasilkan bilangan yang acak tanpa pola yang berulang walaupun proses dilakukan berkali-kali [4]. Dengan mengubah nilai awal *Three Chaotic Logistic Maps* sedikit saja, maka kunci yang dihasilkan akan mengalami perubahan yang signifikan.

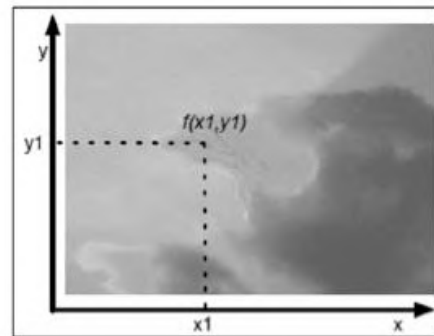
Tujuan dari penelitian ini adalah untuk mengetahui keamanan citra digital dengan menggunakan algoritma *Rubik's Cube* dan *Three Chaotic Logistic Maps*. Untuk mengetahui keamanan tersebut maka dilakukan pengujian kualitas citra untuk melihat kualitas citra setelah dilakukan proses dekripsi, analisis korelasi yang berfungsi untuk mengetahui kualitas enkripsi, analisis sensitivitas kunci untuk mengetahui seberapa sensitif parameter *Three Chaotic Logistic Maps* dan penambahan *noise* untuk mengetahui ketahanan *cipher image* terhadap serangan *noise*.

### 2. Kajian Pustaka

#### Konsep Dasar Citra Digital

Kata citra yang dikenal secara luas dengan kata gambar dapat diartikan sebagai suatu fungsi intensitas cahaya dua dimensi, yang dinyatakan oleh  $f(x,y)$ , dimana nilai amplitudo  $f$  pada koordinat spasial  $(x,y)$  menyatakan intensitas citra pada titik tersebut. Citra digital merupakan citra  $f(x,y)$  yang telah dilakukan digitalisasi pada area koordinat maupun level *brightness*. Nilai  $f$  di koordinat  $(x,y)$  menunjukkan level *brightness* atau *grayness* dari

citra pada titik tersebut. Hal tersebut diilustrasikan pada gambar 1.



Gambar 1 Citra Digital  
(Saputra et al, 2013)

Dari gambar di atas, citra digital dapat didefinisikan sebagai fungsi dua variabel  $f(x_1,y_1)$ , dimana  $x_1$  dan  $y_1$  adalah koordinat spasial dan nilai  $f(x_1,y_1)$  adalah intensitas citra pada koordinat tersebut.

Teknologi dasar untuk menciptakan dan menampilkan warna pada citra digital berdasarkan pada penelitian bahwa sebuah warna merupakan kombinasi dari tiga warna dasar yaitu merah, hijau dan biru (RGB) [5].

#### Teori Chaos

Karakteristik utama sistem *chaos* adalah sensitivitasnya terhadap nilai awal (*initial value*). Sensitivitas ini berarti jika *chaos map* diiterasikan sejumlah kali, maka perubahan kecil pada nilai awal menghasilkan perbedaan yang signifikan pada nilai fungsinya. Karakteristik ini penting di dalam kriptografi sebab bersesuaian dengan prinsip *diffusion* dari Shannon dalam merancang sebuah algoritma kriptografi [6]. Dengan prinsip *diffusion* ini maka perubahan satu bit nilai awal *chaos* dapat menyebabkan *cipherteks* tidak bisa diprediksi lagi dan sebagai konsekuensinya *cipherteks* tetap tidak dapat didekripsi. Fungsi *chaos* yang digunakan di dalam algoritma ini adalah *Logistic Map*.

#### Three Chaotic Logistic Maps

Ide utama dari *Three Chaotic Logistic Maps* adalah fungsi *chaos Logistic Map* yang dilakukan sebanyak tiga kali dan diletakkan dalam satu algoritma

yang sama untuk menghasilkan tingkat keamanan yang lebih tinggi dibanding dengan *chaotic Logistic Map* biasa. Dari tiga persamaan berikut, akan dihasilkan bilangan acak dan akan dipilih secara random untuk digunakan sebagai kunci. Berikut adalah persamaan dari *Three Chaotic Logistic Maps* :

$$\begin{aligned} x_{n+1} &= 3.5699456 x_n (1 - x_n) \\ y_{n+1} &= 3.5699456 y_n (1 - y_n) \\ z_{n+1} &= 3.5699456 z_n (1 - z_n) \end{aligned}$$

Barisan nilai acak yang dihasilkan dari iterasi *Three Chaotic Logistic Maps* sensitif terhadap perubahan kecil nilai awal. Nilai awal  $x_0$  yang di-input merupakan bilangan riil dalam selang  $0 < x_0 < 1$ , nilai awal  $y_0$  diambil dari hasil iterasi persamaan  $x_{n+1} = 3.5699456 x_n (1 - x_n)$  dimana  $y_0 = x_{n-1}$  dan  $n$  merupakan jumlah iterasi pengacakan, begitu juga dengan nilai awal  $z_0$  diambil dari hasil iterasi persamaan  $y_{n+1} = 3.5699456 y_n (1 - y_n)$  dimana  $z_0 = y_{n-1}$ . Nilai akhir  $z_n$  merupakan kunci yang digunakan dalam proses XOR pada saat enkripsi dan dekripsi.

### Algoritma Rubik's Cube

*Rubik's Cube* merupakan algoritma yang bekerja dengan menggunakan prinsip-prinsip kerja dari kubus rubik. Algoritma ini digunakan untuk mengacak piksel-piksel citra. Dimisalkan  $I$  adalah citra digital dengan  $\alpha$ -bit dan ukuran panjang x lebar adalah  $M \times M$ . Nilai matriks piksel dari  $I$  diwakili dengan notasi  $I_0$ . *Input* pada algoritma ini adalah kunci  $Kr$  untuk mengetahui jumlah langkah pergeseran baris ke kiri atau ke kanan, kunci  $Kc$  untuk mengetahui pergeseran kolom ke atas atau ke bawah dan iterasi *maximum* untuk jumlah iterasi pengacakan yang akan dilakukan. Pada prinsip ini digunakan dua buah kunci yaitu kunci  $Kr$  dan  $Kc$  untuk melakukan proses rotasi atau pergeseran piksel citra dan kunci yang dibangkitkan secara acak untuk proses XOR dengan piksel citra.

### 3. Metode Penelitian

#### Proses Enkripsi

Proses enkripsi meliputi proses pembangkitan kunci, proses pengacakan piksel citra dan proses XOR. *Input* pada proses enkripsi yaitu citra warna, parameter *Three Chaotic Logistic Maps* untuk pembangkitan kunci, kunci  $Kc$  dan  $Kr$  serta iterasi *maximum*. Proses pembangkitan kunci dimulai dengan menginput nilai awal  $x_0$ ,  $r$  dan  $n$ . Kemudian lakukan proses perulangan untuk menghasilkan nilai  $x_i$  yang akan dijadikan nilai awal untuk menghitung nilai  $y_i$ . Setelah nilai  $y_i$  selesai dihitung maka nilai  $y_i$  dijadikan nilai awal untuk menghitung nilai  $z_i$  dan nilai akhir  $z_i$  diambil nilai desimalnya kemudian ekstrak menjadi delapan digit. Setelah itu modulokan dengan 256 dan hasilnya digunakan sebagai kunci ( $I_{ciph}$ ) untuk proses XOR. Langkah-langkah proses pengacakan piksel dan proses XOR adalah sebagai berikut :

1. Input Iterasi<sub>max</sub>,  $Kc$  dan  $Kr$ , kemudian inisialisasi iterasi = 0.
2. Iterasi = iterasi + 1
3. Pada tiap-tiap baris  $i$  dan kolom  $j$  pada matriks  $I_0$  lakukan proses berikut:

- a. Hitung jumlah elemen dalam baris ke  $i = 1$  sampai  $M$

$$S_{baris(i)} = \sum_{j=1}^M I_0(i, j)$$

- b. Cari jumlah langkah pergeseran yang akan dilakukan

$$M_{baris(i)} = S_{baris(i)} \bmod Kr$$

- c. Cari arah pergeseran yang akan dilakukan
 
$$W_{baris(i)} = M_{baris(i)} \bmod 2$$
 if  $W_{baris(i)} = 0$ 
  - baris  $i$  geser ke kanan
  - else → baris  $i$  geser ke kiri

- d. Hitung jumlah elemen dalam kolom ke  $j = 1$  sampai  $M$

$$S_{kolom(j)} = \sum_{i=1}^M I_0(j, i)$$

- e. Cari jumlah langkah pergeseran yang akan dilakukan

$$M_{kolom(j)} = S_{kolom(j)} \bmod Kc$$

- f. Cari arah pergeseran yang akan dilakukan

$$W_{kolom(j)} = M_{kolom(j)} \bmod 2$$

$$\text{if } W_{kolom(j)} = 0$$

→ kolom  $j$  geser ke atas

else → kolom  $j$  geser ke bawah

4. If iterasi = iterasi<sub>max</sub> → lanjut ke langkah 5, else ke langkah 2.

Pada langkah 2 - 4 akan menghasilkan citra acak dengan simbol  $I_{scr}$  (pergeseran horizontal dan vertical).

5. Lakukan proses XOR pada masing-masing baris dan kolom citra acak dengan kunci yang dibangkitkan  $I_{ciph}$ .

- a.  $i = 1 ; j = 0$

- b.  $j = j + 1$

- c.  $I_{enc(i, j)} = I_{scr(i, j)} \text{ xor } I_{ciph}$

- d. if  $i = M$  → ke langkah f.

else lanjut ke langkah e

- e. if  $j = M$  →  $i = i + 1 ; j =$

0 lalu kembali ke langkah b

else ke langkah b

- f. Citra yang terenkripsi disimpan dalam  $I_{enc}$ .

#### Proses Dekripsi

Proses dekripsi merupakan kebalikan dari proses enkripsi dimana proses awal yang dilakukan adalah pembangkitan kunci kemudian proses pengacakan piksel citra dan proses XOR. Berikut adalah langkah-langkah proses dekripsi.

1. Input Iterasi<sub>max</sub>,  $Kc$  dan  $Kr$ , kemudian inisialisasi iterasi = 0.

2. Lakukan proses XOR dari gambar enkripsi  $I_{enc}$  dengan kunci yang dibangkitkan  $I_{ciph}$ .

- a.  $i = 1 ; j = 0$

- b.  $j = j + 1$

- c.  $I_{scr(i, j)} = I_{enc(i, j)} \text{ xor } I_{ciph}$

- d. if  $i = M$  → ke langkah f.

else lanjut ke langkah e

- e. if  $j = M$  →  $i = i + 1 ; j =$

0 lalu kembali ke langkah b.

else ke langkah b

- f. Citra yang telah di XOR disimpan dalam  $I_{scr}$ .

3. Iterasi = Iterasi + 1

4. Pada tiap-tiap kolom  $j$  dan baris  $i$  pada matriks  $I_0$  lakukan proses berikut:

- a. Hitung jumlah elemen dalam kolom ke  $j$  sampai  $M$ .

$$S_{kolom(j)} = \sum_{i=1}^M I_{scr}(j, (M + 1 - i))$$

- b. Cari jumlah langkah pergeseran yang akan dilakukan

$$M_{kolom(j)} = S_{kolom(j)} \bmod Kc$$

- c. Cari arah pergeseran yang akan dilakukan

$$W_{kolom(j)} = M_{kolom(j)} \bmod 2$$

- if  $W_{kolom(j)} = 0$   
 → kolom  $j$  geser ke bawah  
 else → kolom  $j$  geser ke atas
- d. Hitung jumlah elemen dalam baris ke  $i$  sampai  $M$ .
- $$S_{baris(i)} = \sum_{j=1}^M I_{scr}((M+1-i, j))$$
- e. Cari jumlah langkah pergeseran yang akan dilakukan
- $$M_{baris(i)} = S_{baris(i)} \text{ mod } Kr$$
- f. Cari arah pergeseran yang akan dilakukan
- $$W_{baris(i)} = M_{baris(i)} \text{ mod } 2$$
- if  $W_{baris(i)} = 0$  → baris  $i$  geser ke kiri  
 else → baris  $i$  geser ke kanan
5. If iterasi = iterasi<sub>max</sub> → simpan dengan nama  $I_0$ , else ke langkah 3.

**Pengujian**

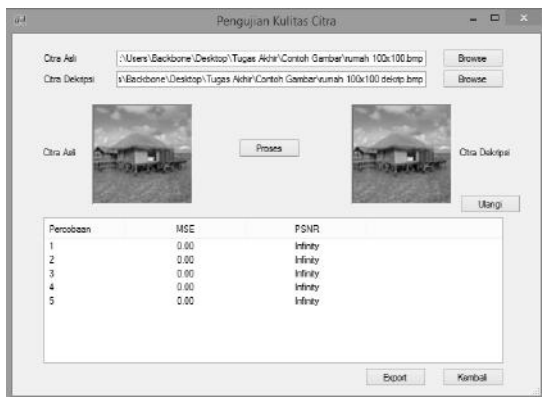
Pengujian yang dilakukan pada penelitian ini meliputi (1) pengujian kualitas citra untuk melihat kembali/tidaknya citra setelah dilakukan proses dekripsi, (2) pengujian kualitas enkripsi untuk melihat keacakan *cipher image* yang dihasilkan dengan menghitung nilai koefisien korelasi antara dua piksel yang bertetangga secara horizontal, vertikal dan diagonal, (3) pengujian sensitivitas kunci untuk melihat pengaruh perubahan parameter *Three Chaotic Logistic Maps* terhadap *cipher image* pada saat proses dekripsi dan (4) pengujian penambahan *noise* untuk mengetahui ketahanan *cipher image* setelah ditambah *noise*.

**4. Hasil**

Untuk melakukan proses pengujian maka dirancang tampilan untuk masing-masing pengujian sehingga diperoleh hasil sebagai berikut :

**Pengujian Kualitas Citra**

Pengujian ini dilakukan untuk mengetahui kualitas citra setelah dilakukan proses enkripsi. Kualitas citra dapat dilihat dari nilai MSE dan PSNR yang dihasilkan. Citra uji yang digunakan untuk proses pengujian adalah citra warna dengan ukuran yang berbeda. Pengujian dilakukan dengan membandingkan citra asli dengan citra hasil dekripsi untuk mendapatkan nilai MSE dan PSNR. Gambar 2 adalah tampilan *form* pengujian kualitas citra.



Gambar 2 Tampilan Pengujian Kualitas Citra

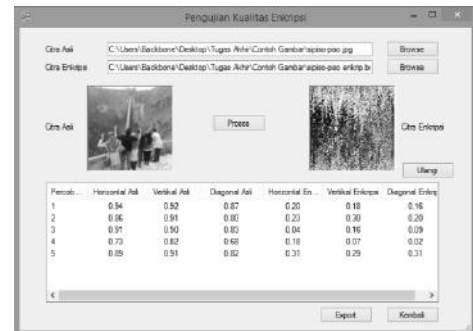
Hasil pengujian dapat dilihat pada Tabel 1.

Tabel 1 Hasil Pengujian Kualitas Citra

Perco baan	Nama Citra	Citra	Ukuran	MSE	PSNR
1	buku.jpg		50 x 50	0.00	Infinity
			100 x 100	0.00	Infinity
			512 x 512	0.00	Infinity
2	minion.jpg		50 x 50	0.00	Infinity
			100 x 100	0.00	Infinity
			512 x 512	0.00	Infinity
3	rumah.bmp		50 x 50	0.00	Infinity
			100 x 100	0.00	Infinity
			512 x 512	0.00	Infinity

**Pengujian Kualitas Enkripsi**

Pengujian kualitas enkripsi digunakan untuk mengetahui kualitas hasil enkripsi dari proses enkripsi yang dilakukan. Untuk mengetahui nilai korelasi pada *plain image* dan *cipher image*, maka dihitung koefisien korelasi antara dua piksel yang bertetangga secara horizontal, vertikal dan diagonal. Pada skenario pengujian kualitas enkripsi ini digunakan citra yang berbeda dengan ukuran yang sama yaitu 100 x 100 piksel dan iterasi *maximum* sebesar 100, 200 dan 300. Gambar 3 adalah tampilan *form* pengujian kualitas citra.



Gambar 3 Tampilan Pengujian Kualitas Enkripsi

Hasil pengujian dapat dilihat pada Tabel 2, Tabel 3 dan Tabel 4







Tabel 2 Pengujian Dengan Iterasi *Maximum* 100

Perco baan	Citra	Hori zontal	Ver tikal	Diagonal
1	<i>Plain image</i>	0,94	0,92	0,87
	<i>Cipher image</i>	0,20	0,18	0,16
2	<i>Plain image</i>	0,86	0,91	0,80
	<i>Cipher image</i>	0,23	0,30	0,20
3	<i>Plain image</i>	0,89	0,91	0,82
	<i>Cipher image</i>	0,31	0,29	0,31

Tabel 3 Pengujian Dengan Iterasi *Maximum* 200

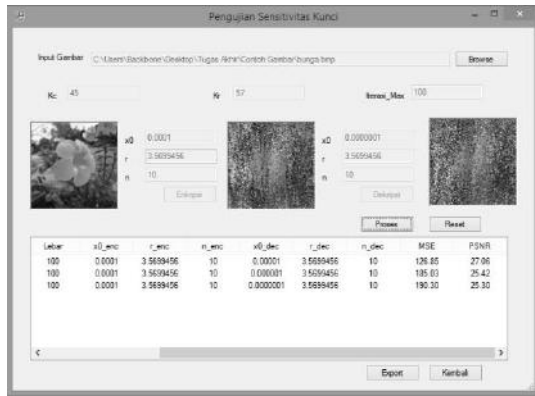
Perco baan	Citra	Hori zontal	Ver tikal	Dia gonal
1	<i>Plain image</i>	0,94	0,92	0,87
	<i>Cipher image</i>	0,04	0,13	0,13
2	<i>Plain image</i>	0,86	0,91	0,80
	<i>Cipher image</i>	0,16	0,17	0,16
3	<i>Plain image</i>	0,89	0,91	0,82
	<i>Cipher image</i>	0,21	0,16	0,17

Tabel 4 Pengujian Dengan Iterasi *Maximum* 300

Percobaan	Citra	Hori zontal	Ver tikal	Diag onal
1	 Plain image	0,94	0,92	0,87
	 Cipher image	0,11	0,11	0,11
2	 Plain image	0,86	0,91	0,80
	 Cipher image	0,17	0,16	0,16
3	 Plain image	0,89	0,91	0,82
	 Cipher image	0,10	0,13	0,09

**Pengujian Sensitivitas Kunci**

Untuk mengetahui sensitivitas kunci yang dibangkitkan maka dilakukan pengujian dengan mengubah nilai parameter *Three Chaotic Logistic Maps* ( $x_0$ ,  $r$ ). Kunci dikatakan sensitif apabila *cipher image* tidak dapat kembali menjadi *plain image* semula karena adanya perubahan kunci pada proses dekripsi. Citra uji yang digunakan pada pengujian ini adalah citra dengan ukuran 100 x 100 piksel. Citra awal (bunga.bmp) akan dienkripsi dengan nilai  $x_0 = 0,0001$ ,  $r = 3,5699456$ ,  $n = 10$ ,  $Kc = 45$ ,  $Kr = 57$ , Iterasimax = 100. Gambar 4 adalah tampilan *form* pengujian sensitivitas kunci dengan melakukan perubahan nilai  $x_0$ .



Gambar 4 Tampilan Pengujian Sensitivitas Kunci

Hasil pengujian dapat dilihat pada Tabel 5 dengan perubahan nilai parameter *Three Chaotic Logistic Maps* ( $x_0$ ,  $r$ ) sebagai berikut:

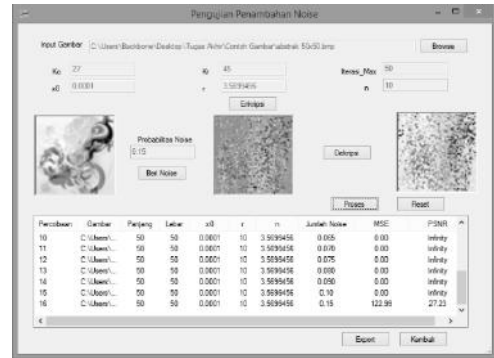
Tabel 5 Hasil Pengujian Sensitivitas Kunci dengan Merubah Nilai  $x_0$  dan  $r$

Citra	ukuran	Perubahan nilai parameter <i>Three Chaotic Logistic Maps</i>	MSE	PSNR
	100 x 100	$x_0 = 0,00001$	126,85	27,06
		$x_0 = 0,000001$	185,03	25,42
		$x_0 = 0,0000001$	190,3	25,3
		$r = 3,5699457$	200,95	25,07
		$r = 3,5699458$	164,12	25,95
		$r = 3,5699459$	122,78	27,21

**Pengujian Penambahan Noise**

Pengujian ini dilakukan untuk mengetahui seberapa tahan algoritma terhadap serangan *noise gaussian*. Pada pengujian ini, *user* akan secara sengaja memberikan *noise* pada *cipher image* dengan nilai

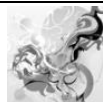

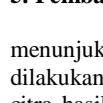
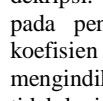
probabilitas yang berbeda-beda. Citra uji yang digunakan adalah citra warna berbeda dengan ukuran 50 x 50 dan 100 x 100 piksel. Pada semua skenario pengujian digunakan kunci yang sama dengan iterasi *maximum* 100. Gambar 5 adalah tampilan *form* pengujian penambahan *noise*.



Gambar 4 Tampilan Pengujian Penambahan Noise

Hasil pengujian dapat dilihat pada Tabel 6 berikut.

Tabel 6 Hasil Pengujian Penambahan Noise

Citra	Ukuran	Proba bilitas Noise	MSE	PSNR
	50 x 50	0,10	0,00	Infinity
		0,15	122,99	27,23
		0,20	122,99	27,23
	100 x 100	0,070	0,00	Infinity
		0,075	17,40	3,74
		0,080	17,40	3,74
	50 x 50	0,090	0,00	Infinity
		0,095	76,72	27,76
		0,100	76,72	27,76
	100 x 100	0,070	0,00	Infinity
		0,075	17,40	3,74
		0,080	17,40	3,74

**5. Pembahasan**

Percobaan dengan menguji kualitas citra menunjukkan bahwa proses enkripsi dan dekripsi yang dilakukan tidak menyebabkan *error* pada citra asli dan citra hasil karena tidak ada kerusakan yang terjadi pada citra asli ketika proses enkripsi sehingga citra dapat kembali seperti citra semula setelah dilakukan proses dekripsi. Percobaan dengan merubah iterasi *maximum* pada pengujian kualitas enkripsi menghasilkan nilai koefisien korelasi pada *cipher image* mendekati 0 yang mengindikasikan bahwa piksel-piksel yang bertetangga tidak lagi berkorelasi sehingga diperoleh kualitas enkripsi yang bagus.

Sedangkan percobaan pada perubahan parameter *Three Chaotic Logistic Maps* berpengaruh terhadap proses dekripsi. Perubahan sedikit saja pada nilai awal  $x_0$  dan  $r$  menyebabkan kegagalan dalam mengembalikan *cipher image* menjadi *plain image* semula. Kemudian untuk pengujian penambahan *noise*, *cipher image* tahan terhadap serangan *noise gaussian* dengan nilai probabilitas di bawah 0,10 (10%). Nilai tersebut tergantung pada citra uji yang digunakan. Setiap citra memiliki ketahanan yang berbeda-beda terhadap serangan.

**6. Kesimpulan**

Berdasarkan hasil pengujian yang dilakukan dapat ditarik kesimpulan sebagai berikut :

1. Hasil citra dekripsi tidak mengalami perubahan jika dibandingkan dengan citra asli sebelum proses enkripsi sehingga menunjukkan bahwa proses enkripsi dan dekripsi yang dilakukan tidak menyebabkan *error* pada citra asli dan citra hasil.
2. Perubahan iterasi *maximum* memiliki pengaruh yang besar pada proses enkripsi, semakin besar jumlah iterasi yang dilakukan maka semakin teracak citra yang dihasilkan.
3. Perubahan parameter *Three Chaotic Logistic Maps* pada proses dekripsi menyebabkan kegagalan dalam mengembalikan *cipher image* menjadi *plain image* semula sehingga membuktikan bahwa sifat *chaos* yang sensitif terhadap nilai awal.
4. *Cipher image* yang dihasilkan dari proses enkripsi tahan terhadap serangan *noise gaussian* tergantung pada citra uji yang digunakan.

## 7. Referensi

- [1] Loukhaoukha, K., Chouinard, J.-Y., dan Berdai, A., 2012, *A Secure Image Encryption Algorithm Based on Rubik's Cube Principle*, volume 2012, Article ID 173931, 13 pages.
- [2] Saputra, W., Affif, S. A., dan Eka, R. D., 2013, Enkripsi Citra Digital Dengan Menggunakan Algoritma *Rubik's Cube*.
- [3] Francois, M. dan Defour, D., 2013, *A Pseudo Random Bit Generator Using Three Chaotic Logistic Maps*, Article ID 00785380.
- [4] Patidar, V. dan Sud, K. K., 2008, *A Pseudo Random Bit Generator Based on Chaotic Logistic Map and its Statistical Testing*, *Informatica* 33.
- [5] Gonzalez, R. C., dan Woods, R. E., 2002, *Digital Image Processing*, edisi 2, Prentice Hall Inc., New Jersey.
- [6] Schneier, B., 1996, *Applied Cryptography*, edisi 2, Wiley & Sons.