PENGAMANAN CITRA DENGAN OPERATOR ALGORITMA GENETIKA

SKRIPSI

Oleh
Ahmad Saiful Rizal
NIM 151810101032

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2019

PENGAMANAN CITRA DENGAN OPERATOR ALGORITMA GENETIKA

SKRIPSI

Diajukan guna melengkapi dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Matematika (S1)
dan mencapai gelar Sarjana Sains

Oleh
Ahmad Saiful Rizal
NIM 151810101032

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2019

PERSEMBAHAN

Dengan menyebut nama Allah SWT yang Maha Pengasih lagi Maha Penyayang, sholawat serta salam atas junjungan kita Nabi Muhammad SAW, Saya persembahkan skripsi ini kepada :

1. Allah SWT, karena atas berkah dan rahmat-Nya skripsi ini dapat terselesaikan;
2. Orang tua saya, Bapak Ady Hadi Wijaya dan Ibu Kholifah, yang selalu memberikan dukungan dan doa yang tiada henti;
3. Adik tersayang, Kamilia Gita Wardani;
4. Teman-teman angkatan 2015 (SIGMA), yang selalu berbagi kisah dalam kasih, selalu memberi *support*, saran, masukan dan pengalaman berharga;
5. UKM Catur UNEJ yang selalu mengajarkan teknik-teknik dalam permainan catur;
6. Almamater tercinta Jurusan Matematika FMIPA Universitas Jember.

MOTO

"*Learn from yesterday, live for today, hope for tomorrow. The important thing is not to stop questioning*"[1]

"*Start where you are. Use what you have. Do what you can*"[2]

[1]Albert Einstein
[2]Arthur Ashe

PERNYATAAN

Saya yang bertanda tangan dibawah ini:

nama : Ahmad Saiful Rizal

NIM : 151810101032

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "Pengamanan Citra dengan Operator Algoritma Genetika" adalah benar-benar hasil karya sendiri, kecuali kutipan yang telah disebutkan sumbernya, belum pernah diajukan di institusi manapun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata dikemudian hari pernyataan ini tidak benar.

Jember, Januari 2019

Yang menyatakan,

Ahmad Saiful Rizal

NIM. 151810101032

SKRIPSI

PENGAMANAN CITRA DENGAN OPERATOR ALGORITMA GENETIKA

Oleh

Ahmad Saiful Rizal
NIM 151810101032

Pembimbing

Dosen Pembimbing Utama : Abduh Riski, S.Si., M.Si

Dosen Pembimbing Anggota : Ahmad Kamsyakawuni, S.Si., M.Kom

PENGESAHAN

Skripsi berjudul "Pengamanan Citra dengan Operator Algoritma Genetika" telah diuji dan disahkan pada:

Hari, tanggal :

Tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Tim Penguji :

Ketua,	Anggota I,
Abduh Riski, S.Si., M.Si NIP. 199004062015041001	Ahmad Kamsyakawuni, S.Si., M.Kom NIP. 197211291998021001
Anggota II,	Anggota III,
Kiswara Agung Santoso, S.Si., M.Kom NIP. 197209071998031003	Ikhsanul Halikin, S.Pd, M.Si NIP. 198610142014041001

Mengesahkan
Dekan,

Drs. Sujito, Ph.D.
NIP. 196102041987111001

RINGKASAN

Pengamanan Citra dengan Operator Algoritma Genetika; Ahmad Saiful Rizal, 151810101032; 2019; 82 Halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Penggunaan data semakin luas dalam berbagai bidang. Semakin luasnya penggunaan data, sering kali terjadi pencurian data dalam dunia internet seperti data keuangan, data privasi dan data citra. Pengamanan sangat diperlukan untuk menjaga kerahasiaan data agar data tersebut tidak disalahgunakan oleh pihak yang tidak berwenang. Teknik enkripsi dan dekripsi adalah teknik yang digunakan untuk menjamin keamanan atau kerahasiaan suatu data. Teknik tersebut dapat dipelajari dalam bidang kriptografi.

Kriptografi adalah salah satu ilmu untuk menjaga kerahasiaan data dengan mengubah data menjadi bentuk sandi sehingga data tersebut sulit dipahami. Data dikodekan dengan algoritma tertentu sehingga data yang dikirimkan sampai kepada penerima dengan aman. Penelitian ini menggunakan Operator Algoritma Genetika yaitu *crossover* dan mutasi. Dalam proses *crossover* akan dipilih dua titik dari dua kromosom kemudian bitstring antara dua titik itu akan mengalami pertukaran. Setelah melalui *crossover*, proses berikutnya adalah mutasi, yaitu perubahan nilai bitstring dari 0 menjadi 1 dan sebaliknya. Kromosom menyatakan piksel dengan kedalaman 8-bit, gen menyatakan nilai kromosomnya yaitu 0 dan 1.

Data yang digunakan dalam penelitian ini adalah data citra yang digunakan sebagai *plain image* dan kunci. Kunci yang digunakan diambil derajat keabuannya, kemudian akan mengalami pergeseran 1-bit kekiri sebanyak 6 kali. Selanjutnya operasi penjumlahan modulo 256 dilakukan antara *plain image* dan kunci. Setelah itu, *plain image* dienkripsi dengan proses *crossover* dan mutasi sehingga dihasilkan *cipher image.* Enkripsi ini menghasilkan *plain image* yang benar-benar berbeda dengan *plain image*. Analisis keamanan dari metode yang digunakan menunjukkan bahwa algoritma aman dari serangan analisis frekuensi dibuktikan dengan analisis histogram, analisis diferensial dan analisis korelasi.

PRAKATA

Puji syukur kehadirat Allah SWT atas segala rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul "Pengaman Citra dengan Operator Algoritma Genetika". Skripsi ini disusun untuk memenuhi salah satu syarat pada program pendidikan strata satu (S1), Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Jember.

Pada kesempatan ini penulis mengucapkan terima kasih atas bantuan dan bimbingan dalam penyusunan tugas akhir ini, terutama yang terhormat:

1. Drs. Sujito, Ph.D., selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
2. Kusbudiono, S.Si., M.Si., selaku Ketua Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
3. Abduh Riski, S.Si., M.Si., selaku Dosen Pembimbing Utama, Ahmad Kamsyakawuni, S.Si., M.Kom., selaku Dosen Pembimbing Anggota, Kiswara Agung Santoso, S.Si., M.Kom., selaku Dosen Penguji I dan Ikhsanul Halikin, S.Pd, M.Si., selaku Dosen Penguji II yang telah meluangkan waktu, pikiran, dan perhatian dalam penulisan skripsi ini;
4. Orang tua tercinta, yang selalu memberikan doa disetiap langkahnya;
5. Dosen dan Karyawan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
6. Teman-teman SIGMA yang memberikan banyak dukungan dan kenangan;
7. Semua pihak yang telah membantu terselesaikannya skripsi ini.

Semoga bantuan, bimbingan, dan motivasi beliau dicatat sebagai amal shaleh, dilipatgandakan pahalanya oleh Allah SWT. Selain itu, penulis juga menerima segala kritik dan saran dari semua pihak demi kesempurnaan skripsi ini. Semoga skripsi ini dapat bermanfaat bagi orang lain.

Jember, Januari 2019

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Semakin pesatnya perkembangan teknologi, penggunaan data semakin luas dalam berbagai bidang. Seiring semakin luasnya penggunanan data, sering kali terjadi pencurian data terutama dalam dunia internet seperti data keuangan, data privasi dan data citra. Pengamanan data menjadi hal yang penting agar data tersebut tidak disalahgunakan oleh pihak yang tidak berwenang. Oleh karena itu, pengamanan sangat diperlukan untuk menjaga kerahasiaan suatu data yaitu dengan teknik enkripsi dan dekripsi. Teknik ini digunakan untuk menjamin komunikasi yang aman antara kedua pelaku sistem informasi, dan tentunya teknik enkripsi dan dekripsi dapat diketahui dan dipelajari dalam bidang kriptografi.

Kriptografi merupakan salah satu ilmu yang digunakan untuk menjaga kerahasiaan dan keaslian suatu data dengan mengubah data menjadi bentuk sandi sehingga data tersebut sulit dipahami oleh orang lain dan hanya dapat dipahami oleh penerima yang berwenang. Proses kriptografi pada dasarnya yaitu mengirim pesan atau data kepada penerima dengan mengubah data asli (*plaintext*) menjadi data rahasia (*ciphertext*), kemudian data rahasia itu diterjemahkan (*dekripsi*) menjadi data asli (*plaintext*) oleh penerima. Dengan kriptografi, data dikodekan dengan algoritma tertentu sehingga data yang dikirimkan sampai kepada penerima dengan aman.

Algoritma genetika merupakan suatu metode pencarian acak yang didasarkan atas prinsip evolusi yang terjadi di alam. Algoritma genetika merupakan algoritma yang digunakan untuk memecahkan suatu permasalahan yang memiliki banyak kemungkinan solusi. Dalam kriptografi, Algoritma Genetika dapat diaplikasikan kedalam bentuk penyandian teks dengan menggunakan teknik *crossover* dan *mutation* dimana teks dikonversi menjadi biner yaitu 0 dan 1. *Crossover* yang digunakan adalah *crossover* dua titik, sedangkan mutasi yang digunakan adalah *flipping of bits* (Sindhuja dan Devi, 2014).

Enayatifar dan Abdullah (2011) telah menggunakan Algoritma Genetika untuk menyandikan suatu citra atau gambar. Kunci yang digunakan diperoleh dari citra yang akan disandikan (*plain image*) dengan melalui proses ekstraksi sehingga didapatkan pesan citra bersandi (*cipher image*). Pada proses enkripsi, kunci yang digunakan hanya 5 piksel untuk masing-masing bagian dan waktu yang dibutuhkan relatif lama bila piksel dari citra yang digunakan berukuran besar. Selain itu, tidak terdapat proses dekripsi sehingga tidak dapat dipastikan hasil enkripsi (*cipher image*) bisa kembali ke bentuk citra sebenarnya (*plain image*).

Pujari et al. (2017) telah melakukan proses enkripsi dan dekripsi citra menggunakan Algoritma Genetika dan rangkaian DNA. Dalam proses enkripsi dan dekripsi terdapat dua tahap yaitu tahap transposisi (pengacakan piksel) dan substitusi. Pada tahap transposisi, piksel-piksel pada citra diacak menggunakan Algoritma genetika dengan salah satu operasi yaitu *crossover* atau *mutation*. Untuk tahap substitusi, terdapat operasi XOR antara bitstring dari nilai piksel dengan kunci yang berupa *random* DNA. Namun dalam proses penyandian citra, citra yang digunakan hanya terbatas di ukuran 256 × 256 piksel dan citra yang telah dienkripsi masih terlihat polanya.

Pada penelitian ini akan dilakukan pengamanan citra dengan Operator Algoritma Genetika yaitu *crossover* dan mutasi, serta kunci yang digunakan akan mengalami pergeseran 1-bit kekiri sebanyak 6 kali dan kunci diperoleh dari citra yang lain. Dalam proses *crossover* nantinya akan dipilih dua titik dari dua kromosom kemudian bitstring antara dua titik itu akan mengalami pertukaran sehingga akan dihasilkan kromosom baru. Sedangkan pada proses mutasi akan terjadi perubahan bitstring hasil *crossover*, jika bit bernilai 0 maka berubah menjadi 1 begitu juga sebaliknya. Penggunaan Operator Algortima Genetika dalam pengamanan citra dilakukan agar pesan citra yang dikirimkan tidak bocor ke orang lain. Pengamanan citra dengan Operator Algoritma Genetika dilakukan dengan harapan agar tingkat keamanan pada penyandian pesan yang berupa citra memiliki tingkat keamanan yang tinggi.

1.2 Rumusan Masalah

Adapun rumusan masalah dalam penelitian ini meliputi:

a. Bagaimanakah proses enkripsi dan dekripsi citra dengan Operator Algoritma Genetika ?
b. Bagaimanakah hasil analisis keamanan dari metode yang digunakan?

1.3 Tujuan

Tujuan dari penelitian ini adalah:

a. Mengenkripsi dan mendekripsi citra dengan Operator Algoritma Genetika.
b. Menganalisis keamanan dari metode yang digunakan melalui analisis histogram, analisis diferensial dan analisis korelasi.

1.4 Manfaat

Manfaat yang diperoleh dari penelitian ini adalah:

a. Mengetahui proses enkripsi dan dekripsi citra dengan Operator Algoritma Genetika.
b. Mengetahui hasil analisis keamanan dari metode yang digunakan.
c. Sebagai bahan studi dan objek literatur bagi penulis dan pembaca untuk bidang teknologi dan informasi.

BAB 2. TINJAUAN PUSTAKA

2.1 Kriptografi

Kriptografi (*cryptography*) berasal dari bahasa Yunani yaitu *crypto* yang berarti rahasia dan *graphia* yang berarti tulisan. Kriptografi adalah ilmu ataupun seni yang mempelajari bagaimana membuat suatu pesan yang dikirim oleh pengirim dapat disampaikan kepada penerima dengan aman. Kriptografi merupakan bagian dari suatu cabang ilmu matematika yang disebut kriptologi (*cryptology*). Kriptografi bertujuan menjaga kerahasiaan informasi yang terkandung dalam data sehingga informasi tersebut tidak dapat diketahui oleh pihak yang tidak sah (Setyaningsih, 2015).

2.1.1 Terminologi dalam Kriptografi

Dalam kriptografi terdapat terminologi atau istilah yang harus dipahami. Menurut Ariyus (2008), terminologi atau istilah yang harus dipahami sebagai berikut:

a. *Plaintext* merupakan pesan asli yang dapat dibaca dan dimengerti yang akan dikirimkan ke penerima (*receiver*).
b. *Ciphertext* merupakan pesan asli yang telah disandikan dan siap untuk dikirimkan oleh pengirim (*sender*) ke penerima (*receiver*). Pesan yang telah disandikan tersebut tidak dapat dibaca karena berupa karakter-karakter yang tidak bermakna.
c. Enkripsi merupakan proses penyandian pesan asli (*plaintext*) menjadi pesan rahasia (*ciphertext*).
d. Dekripsi merupakan proses untuk mengembalikan pesan rahasia (*ciphertext*) menjadi pesan asli (*plaintext*).
e. Kunci, yang dimaksud adalah kunci yang digunakan untuk melakukan proses enkripsi dan dekripsi. Kunci terbagi menjadi dua bagian yaitu kunci rahasia (*private key*) dan kunci umum (*public key*).

f. *Cryptanalysis* yaitu analisis kode atau suatu ilmu untuk mendapatkan *plaintext* tanpa harus mengetahui kunci yang sah. Jika suatu *ciphertext* berhasil diubah menjadi *plaintext* tanpa menggunakan kunci maka proses tersebut dinamakan *breaking code.*

Pada dasarnya, kriptografi mempunyai dua langkah utama yaitu proses enkripsi dan proses dekripsi. Enkripsi adalah proses yang sangat diperlukan dalam kriptografi dan memiliki peran penting dalam pengamanan pesan yang akan dikirimkan sehingga kerahasiaan dari pesan tersebut dapat terjaga. Pesan asli (*plaintext* atau *plain image*) diubah menjadi runtutan karakter kode-kode yang susah dimengerti (*ciphertext* atau *cipher image*) dengan menggunakan kunci. Sedangkan dekripsi adalah kebalikan dari enkripsi. Pesan yang telah dienkripsi (*ciphertext* atau *cipher image*) dikembalikan kebentuk awalnya yaitu pesan asli (*plaintext* atau *plain image*) dengan menggunakan kunci. Proses enkripsi dan dekripsi ditunjukkan pada Gambar 2.1.

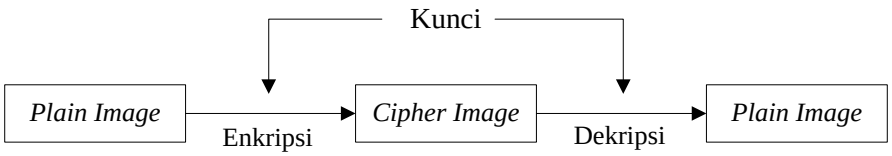
Gambar 2.1 Proses Enkripsi dan Dekripsi

2.1.2 Jenis-jenis Kunci

Berdasarkan kunci yang digunakan, kunci dalam kriptografi dibagi menjadi dua bagian, yaitu:

a. Kunci Simetris

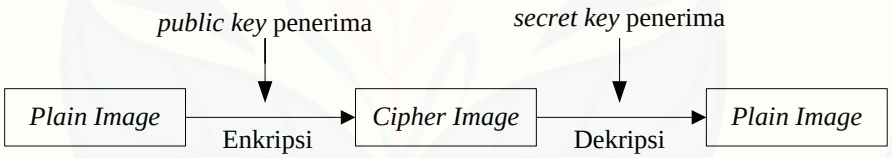
Kunci simetris menggunakan kunci yang sama pada proses enkripsi dan dekripsi. Apabila mengirimkan pesan menggunakan kunci ini, penerima pesan harus mengetahui kunci dari pesan tersebut agar bisa mendekripsikan pesan yang dikirim. Jika kunci yang digunakan diketahui oleh orang lain maka orang tersebut dapat melakukan enkripsi dan dekripsi terhadap pesan. Kunci simetris ditunjukkan pada Gambar 2.2.



Gambar 2.2 Model Sederhana Kunci Simetris

b. Kunci Asimetris

Kunci yang digunakan untuk melakukan enkripsi dan dekripsi adalah berbeda. Kunci asimetris terbagi menjadi dua, yaitu kunci umum (*public key*) dan kunci rahasia (*secret key*). Kunci umum adalah kunci yang dapat diketahui oleh semua orang, sedangkan kunci rahasia adalah kunci yang hanya diketahui satu orang. Kunci-kunci tersebut saling berhubungan. Kunci umum (*public key*) dapat digunakan untuk mengenkripsi pesan tetapi tidak dapat mendekripsikan pesan tersebut. Orang yang dapat mendekripsikan pesan tersebut adalah orang yang memiliki kunci rahasia (*secret key*). Kunci asimetris ditunjukkan pada Gambar 2.3.



Gambar 2.3 Model Sederhana Kunci Asimetris

(Ariyus, 2008).

2.1.3 Tujuan Kriptografi

Menurut Ariyus (2008), terdapat beberapa tujuan dari kriptografi, antara lain sebagai berikut:

a. *Authentication* yaitu berhubungan dengan identifikasi dengan tujuan agar penerima informasi dapat memastikan keaslian pesan, bahwa pesan itu datang dari orang yang dikehendaki.
b. *Integrity* yaitu layanan yang menjamin bahwa pesan masih asli dan tidak dimodifikasi oleh orang yang tidak berhak.

c. *Confidentiality* merupakan usaha untuk menjaga informasi dari orang orang yang tidak berhak mengakses. Ini dilakukan dengan menggunakan sebuah algoritma yang mampu mengubah data asli menjadi data yang sulit dimengerti.
d. *Non-repudiation* merupakan hal yang berhubungan dengan si pengirim, sehingga pengirim tidak dapat mengelak bahwa dialah yang mengirim informasi tersebut.

2.2 Citra

Citra (*image*) adalah salah satu media yang memegang peranan penting sebagai bentuk informasi visual. Citra adalah gambar yang didalamnya terdapat kumpulan-kumpulan piksel yang disusun dalam larik dua dimensi. Sumber cahaya menerangi objek, sebagian dari berkas cahaya dipantulkan kembali oleh objek. Kemudian pantulan cahaya ditangkap oleh alat-alat optik, misalnya mata pada manusia, kamera, dan sebagainya, sehingga bayangan objek yang disebut citra dapat terekam.

Secara matematis, fungsi intensitas cahaya pada suatu citra disimbolkan dengan $f(x,y)$, dimana:

(x,y) : koordinat pada bidang dua dimensi

$f(x,y)$: intensitas cahaya pada titik (x,y)

Bagian terkecil dari citra disebut piksel yang berarti elemen citra. Umumnya citra dibentuk dari kotak-kotak persegi empat yang teratur sehingga jarak horizontal dan vertikal antar piksel adalah sama pada seluruh bagian citra. Dalam komputer setiap piksel diwakili oleh dua buah bilangan bulat untuk menunjukkan lokasinya dalam bidang citra, sedangkan untuk menunjukkan cahaya atau keadaan terang gelap piksel tersebut seringkali menggunakan nilai dengan besar 8 bit yang berarti terdapat 2^8 atau 256 derajat keabuan, dengan selang nilai 0 sampai 255, dimana 0 menyatakan warna hitam, 255 menyatakan warna putih dan tingkat abu-abu berada diantara nilai 0 dan 255 (Ahmad, 2005).

Indeks baris dan kolom (x,y) dari sebuah piksel dinyatakan dalam bilangan bulat. Indeks x bergerak ke kanan dan indeks y bergerak ke bawah. Konvensi ini dipakai merujuk pada cara penulisan yang digunakan dalam pemrograman

komputer. Untuk menunjukkan lokasi suatu piksel, titik asal digunakan untuk posisi kiri atas dalam citra, dan koordinat (*N*,*M*) untuk posisi kanan bawah dalam citra. Intensitas *f* dari gambar hitam putih pada titik (*x*,*y*) disebut derajat keabuan yang mana derajat keabuannya bergerak dari hitam ke putih. Derajat keabuan memiliki rentang nilai $l_{min} < f < l_{max}$. Selang (l_{min}, l_{max}) disebut skala keabuan. Cara untuk menentukan koordinat titik pada citra terlihat pada Gambar 2.4.

Gambar 2.4 Menentukan koordinat titik pada citra
(Sumber: Behnia *et al*., 2007).

Citra digital adalah citra yang dapat diolah langsung oleh komputer secara numerik, disimpan pada komputer sebagai angka untuk menunjukkan besar intensitas pada setiap piksel. Citra digital berukuran $N \times M$ dinyatakan dengan matriks yang berukuran *N* baris dan *M* kolom sebagai berikut :

$$f(x,y) = \begin{bmatrix} f(1,1) & f(1,2) & \cdots & f(1,M) \\ f(2,1) & f(2,2) & \cdots & f(2,M) \\ \vdots & \vdots & \ddots & \vdots \\ f(N,1) & f(N,2) & \cdots & f(N,M) \end{bmatrix}$$

Indeks baris (*x*) dan indeks kolom (*y*) merupakan koordinat suatu titik pada citra, dan *f* (*x*,*y*) merupakan intensitas (derajat keabuan) pada titik (*x*,*y*). Masing-masing elemen pada citra digital (elemen matriks) disebut dengan piksel. Jadi citra yang berukuran $N \times M$ memiliki *NM* buah piksel (Hardjo, 2016).

2.3 Basis Bilangan

Suatu bilangan berdasarkan basisnya terdiri dari beberapa macam diantaranya bilangan desimal dan bilangan biner. Bilangan desimal merupakan susunan bilangan yang mempunyai basis 10 karena bilangan ini menggunakan 10 koefisien yang mungkin yaitu 0, 1, 2, 3, 4, 5, 6, 7, 8 dan 9. Berbeda dengan bilangan biner

yang memiliki susunan basis 2 yaitu 0 dan 1. Suatu basis bilangan dapat dikonversi ke basis bilangan yang lain. Berikut merupakan langkah-langkah untuk melakukan konversi bilangan-bilangan tersebut.

1. Konversi bilangan desimal ke bilangan biner

Salah satu cara untuk mengubah bilangan desimal ke bilangan biner yaitu dengan menggunakan metode pembagian angka desimal dengan angka 2. Kemudian perhatikan hasil bagi dan sisa pembagiannya. Jika hasil bagi tidak nol lakukan pembagian dengan 2 secara terus-menerus dan pembagian akan berhenti ketika diperoleh hasil bagi bernilai 0. Perhatikan sisa-sisa pembagiannya yang nantinya akan membentuk jawaban biner dengan mengurutkan dari bawah ke atas.

Contoh :

Bilangan desimal 23

23 / 2 = 11 sisa 1

11 / 2 = 5 sisa 1

5 / 2 = 2 sisa 1

2 / 2 = 1 sisa 0

1 / 2 = 0 sisa 1

Maka bilangan biner dari 23 adalah 10111

2. Konversi bilangan biner ke bilangan desimal

Untuk mengubah bilangan biner menjadi bilangan desimal digunakan metode penjumlahan bobot seluruh bit dengan memperhatikan angka 1 dan mengabaikan angka 0. Untuk bobotnya yaitu 2^0, 2^1, ..., 2^n dimana 2^0 bobot terkecil yang terletak di bit paling kanan dan 2^n bobot terbesar yang terletak di bit paling kiri.

Contoh :

10101101 $= 1*2^7 + 0*2^6 + 1*2^5 + 0*2^4 + 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0$

Bobot 2^1, 2^4 dan 2^6 tidak dihitung karena memiliki digit nol shingga

10101101 $= 1*2^7 + 1*2^5 + 1*2^3 + 1*2^2 + 1*2^0$

$= 128 + 32 + 8 + 4 + 1 = 173$

Bilangan biner memiliki beberapa operasi bilangan, salah satunya adalah *exclusive-OR* (XOR). Tabel 2.1 menunjukkan hasil dari operasi XOR pada bilangan biner.

Tabel 2.1 Operasi XOR pada bilangan biner

p	q	$p \oplus q$
1	1	0
1	0	1
0	1	1
0	0	0

(Latif *et al*., 2011).

2.4 Pergeseran Bit

Pergeseran bit (bitshift) merupakan pergeseran yang melibatkan angka biner. Pergeseran bit yang digunakan yaitu pergeseran 1-bit kekiri. Dalam pergeserannya terdapat aturan yang ditetapkan yaitu jika angka paling kiri (*most significant bit*) sebelum digeser adalah angka 1 maka akan di-XOR dengan (0001 1011) sedangkan jika angka paling kiri adalah 0 sebelum terjadi pergeseran maka tidak perlu di-XOR dengan (0001 1011).

Contoh : $212_{(10)}$ = 1101 $0100_{(2)}$

= 1101 0100 << 1 (maksudnya terjadi pergeseran 1-bit kekiri)

= 1010 1000 XOR 0001 1011 = 1011 0011

(Stallings, 2006).

2.5 Algoritma Genetika

Algoritma Genetika adalah metode pencarian yang didasarkan pada proses evolusi alam, yaitu terbentuknya populasi awal secara acak yang terdiri dari individu-individu dengan sifat yang tergantung pada gen-gen dalam kromosomnya. Algoritma Genetika merupakan proses pencarian yang *metaheuristic* dan acak sehingga penekanan pemilihan operator yang digunakan sangat menentukan keberhasilan algoritma genetika dalam menemukan solusi

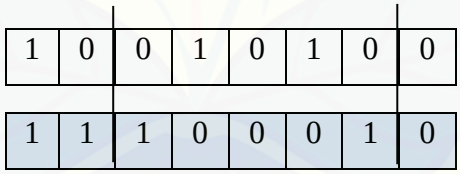
optimum suatu masalah. Pendekatan yang diambil oleh algoritma ini adalah dengan menggabungkan secara acak berbagai pilihan solusi terbaik kedalam satu kumpulan, sehingga dihasilkan generasi terbaik. Terdapat juga operator genetika yang digunakan setelah terbentuk populasi baru. Operator-operator tersebut adalah sebagai berikut:

a. Seleksi (s*election*)

Seleksi adalah proses penentuan individu mana yang akan menjadi *parent.* Berdasarkan seleksi alam, individu yang lemah tidak akan bertahan hidup lama untuk bereproduksi.

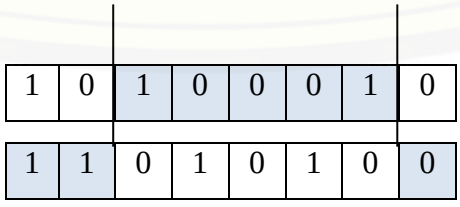
b. *Crossover*

Crossover merupakan pertukaran gen-gen dalam kromosom yang bertujuan untuk menambah keanekaragaman kromosom. Dengan proses pertukaran gen antar kromosom satu dengan kromosom lainnya akan menghasilkan kromosom yang baru. Kromosom merepresentasikan piksel dengan kedalaman 8-bit. Salah satu jenis *crossover* yang akan digunakan yaitu *crossover* yang melibatkan kode biner atau lebih tepatnya *crossover* N-titik, seperti yang ditunjukkan pada Gambar 2.5.



Gambar 2.5 Pemilihan Titik *Crossover*

Titik yang dipilih yaitu 3 dan 7, kemudian diantara titik tersebut akan terjadi pertukaran gen sehingga terlihat seperti Gambar 2.6.



Gambar 2.6 Hasil *Crossover*

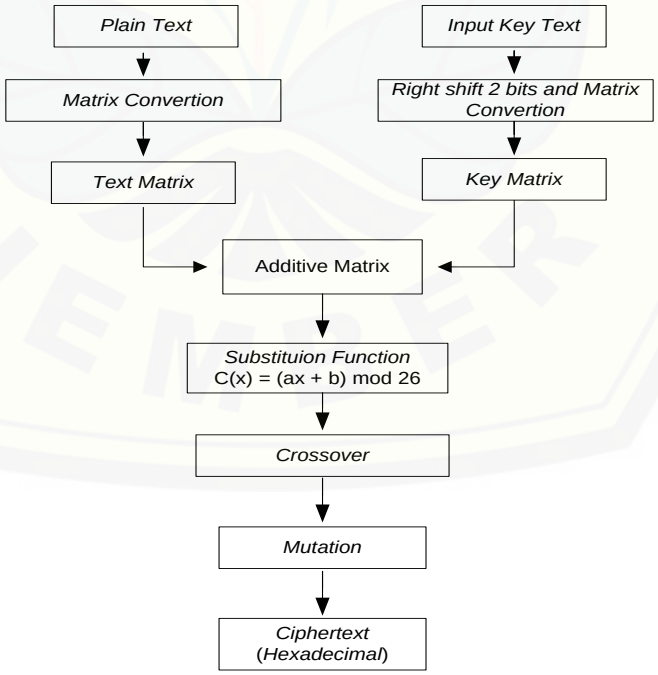
c. Mutasi

Mutasi merupakan proses perubahan nilai gen dalam kromosom. Untuk semua gen yang ada, jika gen tersebut bernilai 0 maka berubah menjadi 1 dan jika bernilai 1 maka akan menjadi 0.

(Dias *et al*., 2016).

2.6 Enkripsi Teks dengan *Crossover* dan Mutasi

Enkripsi teks dilakukan dengan mengubahnya kebentuk matrik. Kunci yang digunakan diubah kedalam kode ASCII kemudian dari kode tersebut dikonversi menjadi biner dan akan mengalami pergeseran kekanan 2-bit. Terdapat juga fungsi substitusi dengan formula $C(x) = (ax + b)$ modulo 26 yang mana a dan b adalah bilangan bulat. *Crossover* yang digunakan yaitu *crossover* dua titik. Teknik *crossover* ini akan dipilih dua titik secara *random* dari dua kromosom induk dan bit-bit diantara dua titik tersebut akan terjadi pertukaran sehingga dihasilkan kromosom baru. Sedangkan proses mutasi yang digunakan yaitu mutasi kebalikan (*flipping of bits*), jika 0 dimutasi menjadi 1 dan sebaliknya. Proses enkripsi teks dapat ditunjukkan pada Gambar 2.7.

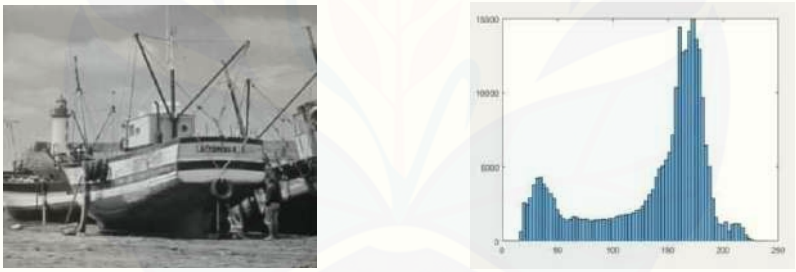


Gambar 2.7 Enkripsi Teks dengan *Crossover* dan Mutasi (Sindhuja dan Devi, 2014).

2.7 Analisis Histogram

Analisis histogram merupakan analisis keamanan didalam pengolahan citra yang memperlihatkan gambaran informasi tentang distribusi nilai piksel pada sebuah citra. Distribusi nilai piksel pada suatu citra yang terenkripsi (*cipher image*) harus merata pada histogramnya. *Cipher image* yang tidak merata penyebaran piksel-nya pada histogram maka akan rentan untuk diserang oleh *attacker* sehingga *cipher image* tersebut tidak cukup aman.

Attacker sering kali menggunakan histogram dengan memanfaatkan frekuensi kemunculan piksel dengan tujuan untuk melakukan kriptanalisis. Agar *attacker* sulit untuk melakukan analisis frekuensi maka histogram *cipher image* seharusnya tidak mempunyai kesamaan dengan histogram *plain image*. Pada analisis histogram terdapat garis mendatar (absis) dan garis tegak (ordinat). Garis mendatar menyatakan nilai-nilai derajat keabuan suatu citra, sedangkan garis tegak menyatakan frekuensi kemunculan dari nilai derajat keabuan, seperti contoh pada Gambar 2.8.



(a) Citra Perahu (b) Histogram Citra Perahu

Gambar 2.8 Analisis dengan Histogram Derajat Keabuan

(Behnia *et al*., 2007).

2.8 Analisis Diferensial

Analisis diferensial digunakan untuk menguji perubahan elemen warna pada *cipher image*. Langkah yang dilakukan dengan menghitung nilai dari *Number of Piksels Change Rate* (NPCR). NPCR merupakan perbandingan posisi piksel antara *plain image* dengan *cipher image*. Adapun perhitungan NPCR dirumuskan seperti pada Persamaan (2.1).

$$NPCR = \left(\frac{1}{m \times n \times p}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} d_{i,j,k}\right) \times 100\% \quad (2.1)$$

dimana *m*, *n* dan *p* merupakan lebar, tinggi dan dimensi dari citra sedangkan $d_{i,j,k}$ adalah fungsi yang ditentukan dengan aturan sebagai berikut :

$$d_{i,j,k} = \begin{cases} 0, & \text{jika } c_{i,j,k}^{(1)} = c_{i,j,k}^{(2)} \\ 1, & \text{jika } c_{i,j,k}^{(1)} \neq c_{i,j,k}^{(2)} \end{cases}$$

yang mana $c_{i,j,k}^{(1)}$ dan $c_{i,j,k}^{(2)}$ merupakan nilai derajat keabuan dari baris *i*, kolom *j*, dan kanal *k* dari citra $c^{(1)}$ dan citra $c^{(2)}$. $c^{(1)}$ ini akan ditetapkan sebagai *plain image* dan $c^{(2)}$ sebagai *cipher image.* Tujuan dari pengujian ini yaitu untuk menjamin bahwa setiap piksel pada citra hasil enkripsi terdapat perubahan elemen warna dengan citra sebelum enkripsi.

Nilai NPCR berkisar antara 0 - 100%. Ketika NPCR bernilai 0% maka *plain image* sama dengan *cipher image*. Namun, saat NPCR bernilai 100% maka terjadi perubahan pada piksel citra secara merata sehingga piksel dari *plain image* berbeda dengan *cipher image*.

Selain dengan NPCR, perhitungan yang digunakan pada analisis ini yaitu *Unified Average Changing Intensity* (UACI). UACI digunakan untuk menghitung persentase rata-rata perubahan intensitas dari piksel yang berubah. Perhitungan UACI dirumuskan seperti Persamaan (2.2).

$$UACI = \left(\frac{1}{m \times n \times p}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} \frac{\left|c_{i,j,k}^{(1)} - c_{i,j,k}^{(2)}\right|}{255}\right) \times 100\% \quad (2.2)$$

Secara teori, nilai minimum yang baik pada indikator NPCR adalah sebesar 99,6094% sedangkan indikator UACI sebesar 33,4635% (Kwok dan Tang, 2007). Menurut Boriga et al. (2014), indikator yang baik harus memenuhi batas minimum dari indikator NPCR yaitu sebesar 98,87% dan UACI sebesar 32,17%.

2.9 Analisis Korelasi

Analisis korelasi digunakan untuk menunjukkan korelasi antara *plain image* dan *cipher image* berdasarkan pada nilai-nilai pikselnya. Algoritma enkripsi yang diajukan akan sangat aman jika *cipher image* yang dihasilkan sungguh berbeda

dengan *plain image*-nya. Adapun perhitungan nilai korelasi dirumuskan seperti Persamaaan (2.3)

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \mu(x))(y_i - \mu(y))}{\sigma(x)\,\sigma(y)} \tag{2.3}$$

dimana r_{xy} menyatakan nilai korelasi, $\mu(x)$ dan $\mu(y)$ adalah rata-rata dari *x* dan *y* dengan perhitungan seperti Persamaan (2.4).

$$\mu(x) = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{dan} \quad \mu(y) = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{2.4}$$

x dan *y* adalah derajat keabuan dari *plain image* dan *cipher image*. Sedangkan perhitungan dari standar deviasi (σ) dirumuskan seperti Persamaan (2.5).

$$\sigma(x) = \sqrt{\sum_{i=1}^{n}\left(x_i - \mu(x)\right)^2} \text{ dan } \sigma(y) = \sqrt{\sum_{i=1}^{n}\left(y_i - \mu(y)\right)^2} \tag{2.5}$$

Jika koefisien korelasi sama dengan 0 maka *cipher image* sepenuhnya berbeda dengan *plain image*. Sedangkan jika koefisien korelasi sama dengan 1 maka *cipher image* dan *plain image* adalah identik (Mousa *et al*., 2013).

BAB 3. METODE PENELITIAN

3.1 Data Penelitian

Data yang penulis gunakan dalam penelitian ini adalah data citra. Berikut ini data-data penelitian yang terdiri dari 5 citra dan 8 citra lainnya terlampir pada lampiran A.

Gambar 3.1 Citra Babon

Gambar 3.2 Citra Gadis

Gambar 3.3 Citra Burung

Gambar 3.4 Citra Lena

Gambar 3.5 Citra Lada

(Sumber: http://informatika.stei.itb.ac.id/~rinaldi.munir/Koleksi/Citra Uji/CitraUji.htm)

3.2 Langkah-langkah Penelitian

Langkah-langkah penelitian ini diuraikan secara sistematis sebagai berikut:

a. Studi Literatur

Pada tahap ini dilakukan dengan mempelajari dan memahami teori-teori yang berkaitan dengan penelitian. Teori yang dipelajari adalah kriptogrfi mengenai teknik enkripsi dan dekripsi, kunci yang digunakan termasuk simetris atau asimetris; Operator Algoritma Genetika yaitu *crossover* dan mutasi; dan teori mengenai citra.

b. Membuat Algoritma Enkripsi dan Dekripsi Menggunakan Operator Algoritma Genetika

Enkripsi dan dekripsi dilakukan dengan mengambil nilai derajat keabuan dari *plain image* dan kunci. Kunci yang digunakan berupa *image*. Setelah itu akan dilakukan manipulasi bit-bit dengan menggunakan Operator Algoritma Genetika yaitu *crossover* dan mutasi.

c. Pembuatan Program

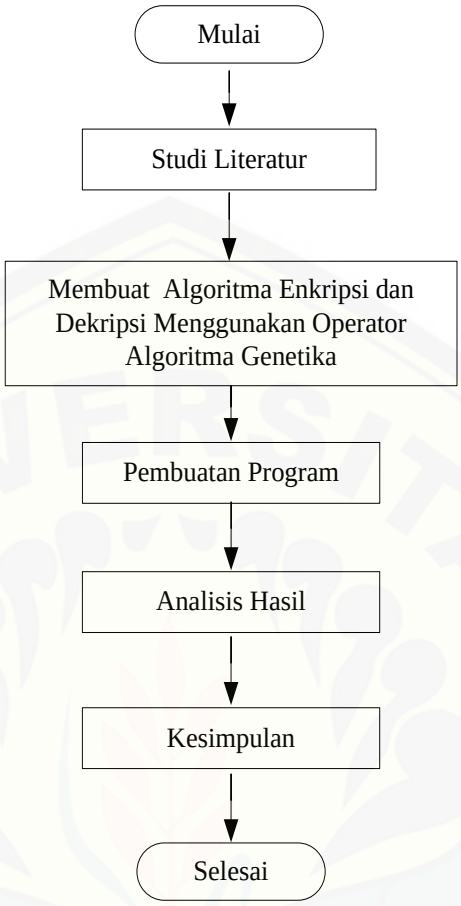
Tahap ini adalah pembuatan program enkripsi dan dekripsi citra berdasarkan Operator dari Algoritma Genetika dengan MATLAB R2015b.

d. Analisis Hasil

Penulis akan menguji program dengan cara menjalankan program tersebut apakah proses telah berjalan dengan baik atau tidak. Pada tahap ini penulis juga menganalisis data yang diperoleh dengan menggunakan analisis histogram derajat keabuan, analisis diferensial dan analisis korelasi dengan tujuan untuk melihat aman tidaknya algoritma yang digunakan.

e. Kesimpulan

Membuat kesimpulan dari penelitian yang dilakukan, yaitu dengan menganalisis proses enkripsi mengubah *plain image* menjadi *cipher image* dan sebaliknya, serta menganalisis keamanan dari algoritma yang digunakan. Langkah-langkah penelitian dengan *flowchart* ditunjukkan pada Gambar 3.6.



Gambar 3.6 Langkah-langkah Penelitian

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, maka dapat diambil beberapa kesimpulan sebagai berikut:

a. Pada proses enkripsi, citra akan melewati tahap proses *crossover* dua titik dan dilanjutkan dengan mutasi yaitu perubahan nilai bit dari 0 menjadi 1 dan sebaliknya, serta kunci yang digunakan akan mengalami pergeseran 1-bit kekiri sehingga didapatkan hasil akhir *cipher image* yang terlihat acak dan tidak berpola. *Cipher image* yang dihasilkan sepenuhnya berbeda dengan *plain image*. Sedangkan pada proses dekripsi, *cipher image* akan melalui tahap proses mutasi dan dilanjutkan dengan proses *crossover* dua titik, serta kunci yang mengalami pergeseran 1-bit kekiri. Akibatnya *cipher image* dapat dikembalikan ke citra yang sebenarnya tanpa menghilangkan informasi yang terkandung didalamnya.
b. Berdasarkan hasil analisis keamanan dilihat dari histogram, NPCR, UACI dan koefisien korelasi, pengamanan citra dengan Operator Algoritma Genetika memiliki tingkat keamanan yang tinggi sehingga sulit untuk diserang oleh kriptanalis.

5.2 Saran

Saran yang dapat diberikan untuk penelitian selanjutnya yaitu menggabungkan modifikasi algoritma genetika dengan algoritma modern seperti AES, DES, dan sebagainya.

DAFTAR PUSTAKA

Ahmad, U. 2005. *Pengolahan Citra Digital dan Teknik Pemrogramannya.* Yogyakarta: GRAHA ILMU.

Ariyus, D. 2008. *Pengantar Ilmu Kriptografi.* Yogyakarta: ANDI.

Behnia, S., A. Akhshani, S. Ahadpour, H. Mahmodi, dan A. Akhavan. 2007. A Fast Chaotic Encryption Scheme Based on Piecewise Nonlinear Chaotic Maps. *Physics Letters A* 366: 391-396.

Boriga, R. E., A. C. Dăscălescu, dan A. V. Diaconu. 2014. A New Fast Image Encryption Scheme Based on 2D Chaotic Maps. *IAENG International Journal of Computer Science* 41 (4).

Dias, M., C. Suhery, dan T. Rismawan. 2016. Penerapan Kriptografi Menggunakan Algoritma Knapsack, Algoritma Genetika, dan Algoritma Arnold's Catmap pada Citra. *Jurnal Coding, Sistem Komputer Untan* 4 (2): 119-129.

Enayatifar, R., dan A. H. Abdullah. 2011. Image Security via Genetic Algorithm. *International Conference on Computer and Software Modeling IPCSIT* 14: 221-226.

Hardjo, A. B. 2016. Enkripsi Citra RGB dengan Algoritma Simplified-Data Encryption Standard (S-DES) dan DNA-Vigenere Cipher. *Skripsi.* Jember: Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Kwok, H.S., dan W.K.S. Tang. 2007. A Fast Image Encryption System based on Chaotic Maps with Finite Precision Representation. *Chaos, Solitons and Fractals* 32 (2007): 1518-1529.

Latif, S., J. Qayyum, M. Lal, dan F. Khan. 2011. Complete Description of Well-Known Number Systems using Single Table. *International Journal of Electrical & Computer Sciences IJECS-IJENS* 11 (3): 23-29.

Mousa, A., O.S. Faragallah, S. El-Rabaie, dan E.M. Nigm. 2013. Security Analysis of Reverse Encryption Algorithm for Databases. *International Journal of Computer Applications (0975-8887)* 66 (14): 19-27.

Pujari, S. K., G. Bhattacharjee, dan S. Bhoi. 2017. A Hybridized Model for Image Encryption through Genetic Algorithm and DNA Sequence. *6th International Conference on Smart Computing and Communications (ICSCC)* 125 (2018): 165-171.

Setyaningsih, E. 2015. *Kriptografi dan Implementasinya Menggunakan MATLAB.* Yogyakarta: ANDI.

Sindhuja, K., dan P. Devi. 2014. A Symmetric Key Encryption Technique Using Genetic Algorithm. *International Journal of Computer Science and Information Technologies (IJCSIT)* 5 (1): 414-416.

Stallings, W. 2006. *Cryptography and Network Security: Principles and Practices*. New Jersey: Pearson Education Inc.

LAMPIRAN

LAMPIRAN A. Citra Penelitian

Citra Mountain

Citra Teks

(Sumber: http://informatika.stei.itb.ac.id/~rinaldi.munir/Koleksi/CitraUji/CitraUji.htm)

Citra Hotel Medan

(Sumber:http://informatika.stei.itb.ac.id/~rinaldi.munir/Koleksi/Foto/Tsunami/Hotel%20Medanedit.jpg)

Citra Bekas Tsunami

(Sumber:http://informatika.stei.itb.ac.id/~rinaldi.munir/Koleksi/Foto/Tsunami/Meulaboh .jpg)

Citra Teluk Ijo

Citra Bunga

Citra Pulau Merah

Citra Malioboro

LAMPIRAN B. Matriks derajat keabuan Citra Babon

Ukuran matriks : 256 × 256, Kanal *Red*

	1	2	3	4	5	6	7	8	9	10	11	12	...	256
1	**141**	**74**	**122**	**78**	**77**	**114**	148	111	177	110	82	92	...	150
2	112	86	81	73	80	68	72	54	111	107	60	67	...	95
3	103	122	81	83	74	92	72	48	61	105	50	54	...	84
4	61	133	117	77	91	108	92	89	68	89	68	68	...	166
5	52	87	146	88	92	160	108	137	90	103	98	102	...	138
6	53	63	87	98	108	189	185	133	107	108	124	140	...	86
7	47	85	75	85	150	129	145	108	135	118	159	157	...	87
8	56	104	100	56	110	155	96	66	94	118	182	136	...	110
9	76	136	125	63	48	79	93	90	88	90	158	149	...	120
10	81	81	112	75	82	88	94	81	131	79	79	121	...	148
11	123	50	66	93	76	132	98	62	107	104	70	123	...	112
12	65	62	55	91	70	87	132	80	96	147	110	141	...	125
13	82	99	74	72	107	95	95	77	71	147	180	144	...	107
14	136	76	100	67	116	141	119	86	74	79	128	125	...	88
15	109	73	66	76	111	136	110	141	164	133	72	128	...	94
16	67	84	81	84	154	121	134	105	110	134	73	101	...	108
17	77	87	138	108	112	134	115	98	106	87	66	68	...	136
18	78	98	84	103	81	90	122	95	135	132	98	106	...	125
19	68	84	79	118	94	60	107	157	162	186	174	137	...	132
20	60	64	94	85	109	92	92	122	123	153	178	185	...	108
21	80	120	149	99	74	78	96	101	110	137	143	159	...	112
22	63	126	164	170	155	84	100	88	86	147	147	109	...	68
23	85	110	149	137	153	135	108	87	60	133	174	118	...	70
24	103	111	105	91	114	94	95	82	44	93	174	185	...	86
25	76	90	88	86	111	81	108	142	95	73	129	166	...	89
26	62	64	97	120	96	82	94	128	114	85	82	119	...	104
27	93	74	134	119	113	98	102	106	103	92	129	144	...	129
28	101	97	128	181	147	154	166	121	117	98	130	150	...	120
29	148	117	76	131	167	123	148	148	128	154	147	186	...	154
30	138	157	92	102	133	126	122	135	164	122	154	164	...	107
31	90	165	163	157	158	160	175	78	89	126	118	105	...	65
32	114	114	72	105	166	173	190	172	74	81	149	136	...	98
33	125	122	133	100	159	169	157	185	147	88	101	168	...	174
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
256	89	85	75	78	84	84	81	86	95	96	93	88	...	61

Ukuran matriks : 256 × 256, Kanal *Green*

	1	2	3	4	5	6	7	8	9	10	11	12	...	256
1	**131**	**54**	**110**	**54**	**57**	**97**	138	96	171	99	76	90	...	170
2	103	71	75	59	89	68	55	29	105	96	45	42	...	93
3	97	109	76	64	71	92	75	32	50	100	37	34	...	96
4	39	118	107	62	85	95	95	87	56	80	71	68	...	188
5	29	75	142	67	85	150	108	143	84	112	111	119	...	137
6	31	39	72	86	91	177	172	128	106	111	134	150	...	72
7	28	78	63	84	143	118	126	110	139	123	168	164	...	104
8	27	106	83	24	99	153	71	74	94	129	185	139	...	117
9	74	131	102	44	31	62	74	96	99	96	168	137	...	144
10	67	71	94	62	90	103	95	79	146	74	71	133	...	163
11	114	33	53	91	63	140	86	52	122	106	64	153	...	124
12	52	44	38	93	58	79	118	69	103	161	113	139	...	132
13	71	100	60	64	113	89	88	69	52	148	178	139	...	107
14	112	76	109	58	113	138	119	88	85	78	122	113	...	81
15	95	62	52	73	114	134	102	145	175	123	82	152	...	106
16	56	85	79	83	155	109	129	97	104	130	68	110	...	116
17	69	96	138	89	108	130	107	103	129	88	57	70	...	162
18	94	102	76	92	75	97	130	97	149	146	109	113	...	145
19	61	79	80	134	87	51	110	158	163	188	190	144	...	148
20	52	56	100	82	101	90	92	123	102	146	172	182	...	124
21	55	127	158	93	71	63	89	111	107	139	138	160	...	124
22	47	128	157	174	164	79	108	84	68	158	147	90	...	67
23	82	112	137	128	148	140	110	73	42	136	170	117	...	62
24	96	113	91	86	102	84	101	82	37	99	184	185	...	72
25	59	89	79	80	106	73	116	154	81	69	128	163	...	88
26	53	57	102	124	93	88	98	132	112	86	87	124	...	113
27	105	78	153	116	111	105	105	115	98	91	145	149	...	139
28	105	108	133	182	140	160	161	120	136	108	145	160	...	143
29	172	112	56	133	168	107	146	149	124	157	150	192	...	167
30	166	163	81	91	134	134	117	134	156	113	151	161	...	90
31	90	156	152	154	150	169	179	69	94	130	122	95	...	68
32	99	105	51	106	171	170	201	173	74	80	160	142	...	135
33	89	118	116	90	152	158	159	187	154	83	95	169	...	192
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
256	116	103	89	99	105	99	94	105	109	106	103	102	...	53

Ukuran matriks : 256 × 256, Kanal *Blue*

	1	2	3	4	5	6	7	8	9	10	11	12	...	256
1	**62**	**22**	**52**	**33**	**39**	**52**	67	39	78	42	44	43	...	105
2	47	34	38	32	46	44	34	19	42	48	44	40	...	71
3	40	49	42	39	42	54	45	30	32	52	37	38	...	60
4	25	49	53	43	52	54	52	49	41	49	51	47	...	103
5	38	30	59	42	45	78	54	58	40	48	50	62	...	72
6	18	20	32	34	39	75	66	53	53	62	60	75	...	38
7	14	42	39	33	59	56	43	64	89	72	90	75	...	65
8	17	36	38	20	29	67	37	45	58	67	119	76	...	67
9	23	41	35	23	20	40	36	42	47	54	102	90	...	70
10	38	29	33	27	30	42	35	40	71	44	41	71	...	103
11	47	29	31	39	31	46	43	35	55	62	34	69	...	66
12	20	30	19	37	36	35	49	43	49	87	57	52	...	71
13	38	37	34	24	38	33	38	33	32	59	86	68	...	53
14	41	31	47	30	34	47	46	42	37	28	59	61	...	44
15	36	37	38	43	56	49	36	54	64	60	50	94	...	75
16	31	43	44	38	82	41	43	34	53	59	62	56	...	79
17	27	52	48	33	43	58	52	42	69	55	45	62	...	112
18	29	55	34	41	31	36	65	48	68	60	72	76	...	103
19	22	36	31	63	47	37	49	48	80	101	90	90	...	85
20	20	26	45	37	38	41	41	34	34	70	74	96	...	90
21	16	60	78	66	44	28	37	50	37	67	50	80	...	86
22	22	58	78	96	87	52	44	52	43	70	72	44	...	42
23	34	49	64	65	74	76	41	37	28	51	74	47	...	39
24	35	42	40	43	57	48	49	47	38	37	97	76	...	46
25	31	34	35	36	50	40	51	62	52	45	66	82	...	50
26	48	29	64	53	50	39	50	66	58	58	55	60	...	54
27	57	43	88	67	59	68	48	66	68	51	77	71	...	64
28	63	58	58	101	74	82	82	61	83	60	81	110	...	89
29	120	68	32	62	86	53	63	84	60	74	84	94	...	114
30	85	90	46	47	56	79	60	51	81	62	66	69	...	63
31	34	63	63	71	82	81	128	36	40	70	54	49	...	31
32	39	46	33	46	96	94	116	106	45	44	57	75	...	67
33	30	47	67	46	44	62	77	100	64	49	46	62	...	101
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
256	118	97	81	93	113	111	107	117	112	100	96	107	...	49

LAMPIRAN C. Matriks derajat keabuan Kunci Citra Lena warna

Ukuran matriks : 128 × 128, Kanal *Red*

	1	2	3	4	5	6	7	8	9	10	11	12	...	128
1	**225**	**225**	**227**	**225**	**221**	**223**	221	222	223	226	231	233	...	237
2	226	225	227	225	222	223	222	222	222	225	231	234	...	214
3	224	225	226	225	226	224	224	223	222	226	233	236	...	103
4	225	226	225	226	226	225	224	224	226	230	233	234	...	84
5	227	225	226	226	226	226	224	224	229	231	231	234	...	91
6	225	225	226	227	227	226	224	225	229	231	231	230	...	93
7	225	224	227	229	227	226	224	227	231	230	229	228	...	100
8	227	226	228	230	228	227	227	231	232	230	228	225	...	102
9	228	228	228	228	229	228	231	233	232	231	228	225	...	99
10	227	227	227	228	229	230	233	234	233	231	228	226	...	98
11	228	228	228	229	230	233	233	233	232	231	229	228	...	93
12	229	228	228	228	233	232	230	230	230	230	231	228	...	93
13	230	229	228	231	235	231	228	226	226	230	231	228	...	95
14	230	230	230	235	235	229	223	220	223	231	231	230	...	93
15	231	231	235	236	233	224	216	214	224	232	233	232	...	92
16	230	233	237	235	230	214	194	210	226	233	235	233	...	98
17	232	236	237	232	224	188	172	210	227	234	235	233	...	101
18	236	238	236	230	209	155	162	210	227	234	235	233	...	76
19	239	239	234	223	178	140	166	209	227	235	235	234	...	76
20	238	236	231	203	155	145	168	209	228	236	236	235	...	156
21	234	232	220	173	148	152	169	211	229	235	236	236	...	216
22	234	228	194	152	155	155	171	210	228	235	236	236	...	217
23	233	212	162	153	162	157	170	208	227	234	236	235	...	200
24	224	182	151	160	161	156	168	209	227	232	235	234	...	198
25	190	158	156	164	163	155	170	208	226	232	235	234	...	212
26	154	154	161	164	163	157	169	205	225	233	234	233	...	216
27	151	158	164	162	161	157	168	206	226	231	232	233	...	213
28	158	161	163	162	162	157	167	206	225	229	231	232	...	213
29	162	161	161	160	159	154	164	205	225	231	233	233	...	212
30	154	157	157	157	159	154	161	202	224	232	232	231	...	211
31	155	156	159	162	164	157	164	203	223	231	234	233	...	211
32	158	159	163	167	169	161	167	201	225	232	235	234	...	211
33	163	164	167	169	171	168	170	203	225	232	234	234	...	211
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
128	87	94	94	98	184	226	223	213	175	205	231	236	...	166

Ukuran matriks : 128 × 128, Kanal *Green*

	1	2	3	4	5	6	7	8	9	10	11	12	...	128
1	**137**	**135**	**136**	**134**	**135**	**134**	131	128	129	137	142	150	...	153
2	137	135	135	132	132	131	130	128	129	134	142	150	...	125
3	132	131	132	130	128	129	127	127	128	133	144	149	...	27
4	131	130	130	131	130	130	128	128	130	140	147	148	...	18
5	130	131	130	131	130	130	127	129	138	145	146	144	...	22
6	130	130	130	131	134	130	127	131	142	144	143	141	...	22
7	129	127	131	132	132	131	129	135	144	145	141	138	...	26
8	130	131	132	133	133	133	134	143	146	144	137	133	...	27
9	132	132	132	135	135	136	142	147	148	143	136	132	...	28
10	134	133	135	138	137	142	145	145	144	141	132	134	...	25
11	137	136	136	137	141	146	142	138	139	135	133	134	...	19
12	136	136	135	138	147	142	131	130	129	133	137	133	...	20
13	138	136	133	143	147	133	121	119	120	135	139	135	...	21
14	135	133	137	147	144	125	110	105	115	137	140	136	...	18
15	133	135	146	147	133	114	92	94	119	137	141	139	...	19
16	134	141	150	140	122	94	72	92	125	140	143	140	...	22
17	140	151	147	130	111	70	56	94	125	141	143	142	...	26
18	148	152	139	121	91	47	52	95	123	142	143	140	...	10
19	151	146	131	107	65	37	60	94	121	142	145	140	...	14
20	146	138	121	82	50	44	61	95	123	142	145	143	...	79
21	139	128	101	61	48	49	61	97	122	140	145	145	...	133
22	130	115	76	48	52	50	63	95	122	140	145	144	...	133
23	117	93	53	50	57	51	64	96	122	139	145	141	...	117
24	98	68	49	59	57	52	61	94	121	136	143	142	...	119
25	70	53	53	61	58	52	59	93	119	135	142	144	...	140
26	55	54	57	58	59	53	60	91	118	135	142	142	...	148
27	54	57	60	61	60	55	65	93	118	134	141	141	...	146
28	56	58	61	59	60	55	64	93	119	134	140	139	...	145
29	62	60	59	60	58	51	59	93	119	134	141	140	...	139
30	61	60	58	56	57	50	57	90	118	134	140	138	...	137
31	58	58	58	59	60	53	57	89	116	134	140	140	...	137
32	59	59	60	61	63	55	58	87	116	134	144	143	...	138
33	61	61	62	64	63	62	62	88	115	136	145	145	...	137
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
128	25	27	27	30	137	198	194	179	98	100	141	158	...	66

Ukuran matriks : 128 × 128, Kanal *Blue*

	1	2	3	4	5	6	7	8	9	10	11	12	...	128
1	**129**	**124**	**124**	**119**	**116**	**111**	107	113	110	116	116	121	...	127
2	126	121	120	114	112	108	106	110	110	113	117	122	...	112
3	110	112	114	109	107	106	106	106	104	108	115	118	...	67
4	109	108	108	106	109	108	104	104	104	110	112	114	...	62
5	111	108	104	106	105	106	105	108	112	114	114	112	...	62
6	109	109	106	106	109	108	109	110	113	112	112	112	...	62
7	107	109	111	107	110	110	109	112	116	114	111	112	...	62
8	108	110	109	107	108	111	113	115	115	114	109	111	...	63
9	108	110	107	106	111	113	115	116	115	111	105	110	...	63
10	116	109	111	110	115	115	114	114	110	107	106	109	...	62
11	117	110	109	112	116	117	112	108	103	104	109	111	...	60
12	112	110	109	113	118	113	105	101	99	104	107	108	...	60
13	117	111	110	116	114	103	97	94	97	106	108	112	...	61
14	113	108	112	114	111	99	89	90	94	106	109	111	...	58
15	107	109	114	112	103	89	80	86	98	105	108	110	...	58
16	112	113	116	109	99	82	73	86	104	108	108	109	...	59
17	112	116	113	103	91	73	69	86	102	108	107	108	...	60
18	116	116	108	99	83	67	69	87	98	108	108	108	...	51
19	124	112	103	90	73	63	74	86	98	108	114	113	...	57
20	119	111	98	80	70	68	75	87	99	107	112	114	...	95
21	113	105	89	73	70	70	76	90	100	106	110	112	...	122
22	107	96	78	70	73	72	77	87	99	105	110	112	...	119
23	93	82	70	72	73	70	77	88	101	110	110	111	...	112
24	81	73	71	76	72	71	74	88	89	109	111	112	...	114
25	76	71	73	76	74	71	74	87	97	105	113	115	...	123
26	76	74	74	74	77	73	74	85	98	107	113	119	...	128
27	76	75	77	75	76	76	79	88	99	111	116	119	...	128
28	74	74	78	75	75	75	79	89	100	110	112	114	...	127
29	78	75	76	76	75	72	75	90	101	108	112	113	...	124
30	79	76	75	73	73	69	72	87	103	110	113	115	...	122
31	77	75	75	75	75	72	73	86	99	111	113	114	...	123
32	80	76	74	76	77	71	73	85	96	107	115	114	...	122
33	77	76	78	78	76	76	76	85	95	107	114	116	...	124
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
128	60	61	63	73	141	183	178	155	98	93	113	123	...	80

LAMPIRAN D. Matriks derajat keabuan *Cipher Image* Citra babon

Ukuran matriks : 256 × 256, Kanal *Red*

	1	2	3	4	5	6	7	8	9	10	11	12	...	256
1	**52**	**247**	**55**	**131**	**43**	**198**	100	137	215	227	88	203	...	70
2	33	219	48	184	88	132	160	146	25	102	62	212	...	255
3	10	87	32	254	103	245	105	120	219	152	181	134	...	35
4	180	76	252	4	22	229	101	104	13	241	51	115	...	93
5	45	202	239	169	165	225	53	88	176	67	24	177	...	36
6	76	66	154	239	133	148	248	12	175	46	206	158	...	143
7	98	92	70	181	203	96	144	53	227	148	59	253	...	176
8	217	201	38	114	108	102	209	168	233	148	244	41	...	41
9	94	130	157	91	138	107	93	189	207	144	236	60	...	52
10	176	176	129	159	8	66	41	86	196	59	203	136	...	120
11	175	88	56	157	62	3	117	89	188	130	84	95	...	117
12	153	92	115	175	161	112	6	218	202	247	76	173	...	216
13	56	7	160	159	188	203	251	116	202	247	70	106	...	58
14	2	222	246	244	5	205	1	34	206	75	218	157	...	125
15	221	161	165	240	56	73	101	209	125	18	127	103	...	199
16	103	19	235	83	112	197	131	28	51	209	94	226	...	176
17	154	165	130	43	161	85	148	211	39	144	117	243	...	143
18	14	154	248	147	196	238	144	118	250	115	69	221	...	12
19	216	232	40	34	104	35	150	136	127	93	233	142	...	85
20	160	60	156	186	75	178	176	11	95	83	122	62	...	23
21	119	47	163	196	155	122	92	176	172	46	77	93	...	227
22	88	60	53	254	221	132	152	61	164	132	121	95	...	223
23	2	96	21	95	33	92	16	126	245	194	206	49	...	153
24	42	94	12	159	152	181	237	243	149	218	185	142	...	204
25	31	89	27	155	219	55	224	119	162	238	214	177	...	37
26	42	40	249	201	106	65	254	180	111	2	69	32	...	155
27	40	169	123	227	89	177	70	250	186	222	38	119	...	77
28	206	9	58	37	23	89	59	11	60	8	104	145	...	214
29	182	85	238	215	236	141	237	208	129	0	244	45	...	4
30	30	214	71	93	238	170	32	184	125	29	125	86	...	90
31	78	206	64	157	19	211	162	193	159	60	65	14	...	100
32	193	1	66	104	118	125	195	243	167	54	226	175	...	67
33	253	39	188	40	189	243	63	54	158	239	194	31	...	103
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
256	74	128	186	78	236	221	215	72	24	36	157	180	...	180

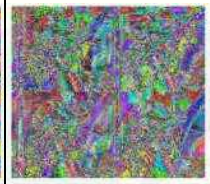
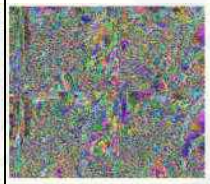
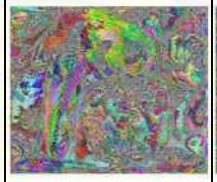
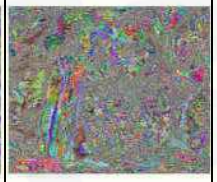
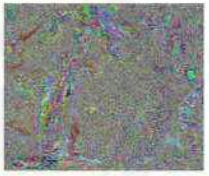
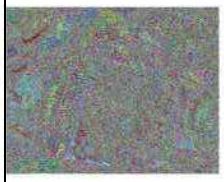
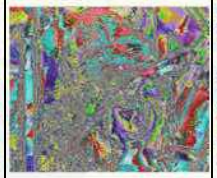
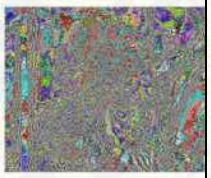
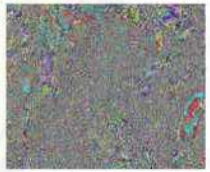
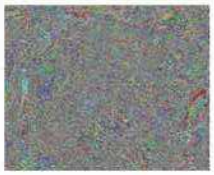
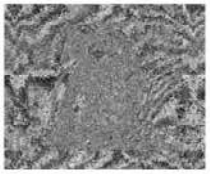
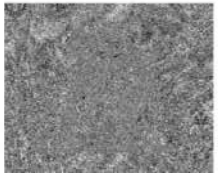
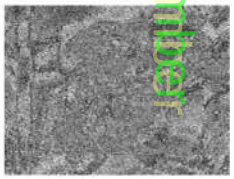
Ukuran matriks : 256 × 256, Kanal *Green*

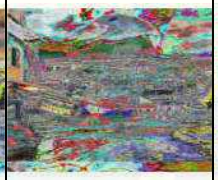
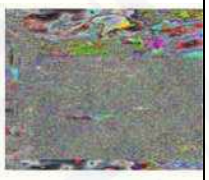
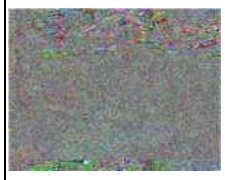
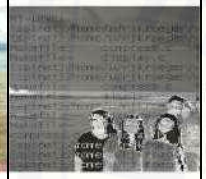
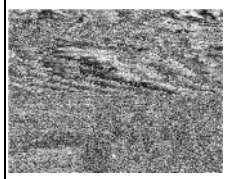
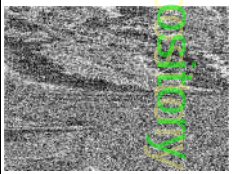
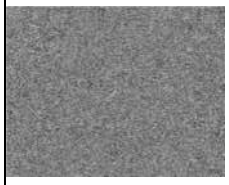
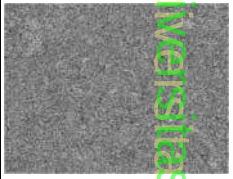
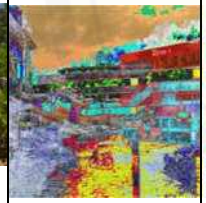
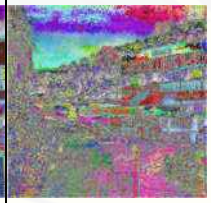
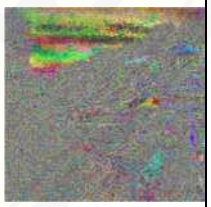
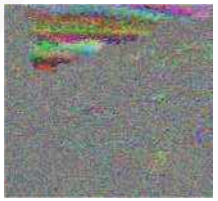
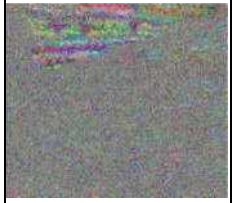
	1	2	3	4	5	6	7	8	9	10	11	12	...	256
1	**49**	**67**	**246**	**19**	**192**	**56**	88	226	103	65	227	219	...	142
2	45	82	110	30	32	94	155	245	201	137	18	251	...	160
3	8	69	253	82	155	102	226	205	128	133	105	179	...	213
4	187	76	55	196	221	211	83	91	74	255	135	209	...	151
5	37	23	36	175	157	220	97	163	32	190	111	231	...	223
6	3	27	26	236	142	193	241	98	101	191	73	217	...	80
7	182	223	83	52	250	28	244	123	35	83	103	48	...	109
8	7	24	214	81	86	128	18	229	80	13	91	14	...	128
9	47	86	51	77	74	70	181	222	242	111	44	208	...	206
10	166	226	203	6	186	104	79	63	156	69	162	100	...	242
11	98	83	111	73	208	162	217	192	250	207	153	160	...	119
12	80	88	163	71	212	224	236	253	75	136	83	94	...	196
13	45	176	29	175	157	160	254	54	210	165	210	158	...	173
14	153	253	231	84	13	99	170	162	123	198	21	35	...	50
15	42	139	218	37	215	122	15	164	76	137	173	140	...	57
16	129	250	70	252	43	184	177	228	21	45	59	17	...	228
17	170	85	212	153	69	253	34	30	140	7	214	41	...	179
18	135	98	56	250	99	137	126	196	129	141	34	254	...	104
19	8	207	242	36	77	185	248	247	99	243	0	239	...	46
20	202	44	98	54	182	96	90	202	128	189	162	249	...	45
21	61	3	201	57	196	12	93	205	27	228	244	30	...	141
22	163	0	256	221	124	172	138	49	2	17	139	4	...	54
23	233	101	71	139	249	175	70	227	188	44	100	250	...	77
24	220	174	96	183	247	220	65	195	33	113	103	166	...	147
25	180	151	209	198	3	151	185	75	42	228	63	155	...	135
26	155	87	135	209	176	104	68	10	27	3	168	195	...	116
27	231	111	189	194	39	247	235	162	201	30	78	74	...	227
28	148	17	1	167	250	208	211	189	179	189	174	4	...	255
29	250	54	229	129	37	80	203	32	143	188	217	79	...	237
30	192	131	44	162	71	181	8	200	191	216	104	3	...	170
31	99	161	37	99	96	215	10	137	173	151	69	32	...	224
32	90	84	179	60	235	246	52	54	209	185	222	49	...	141
33	189	192	242	122	78	8	87	67	54	129	255	53	...	212
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
256	241	126	220	11	27	95	91	61	47	157	8	125	...	159

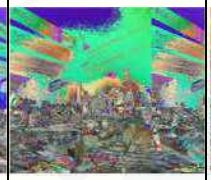
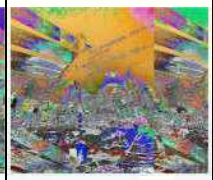
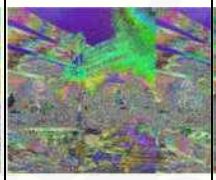
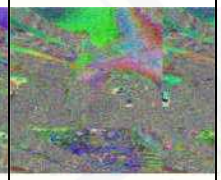
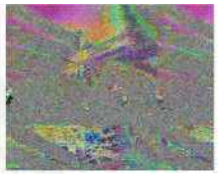
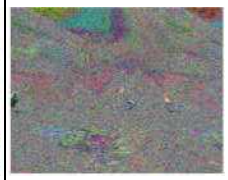
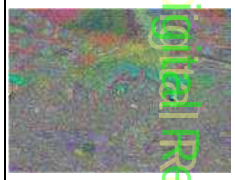
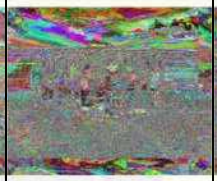
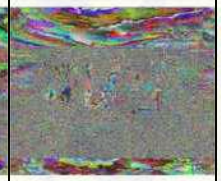
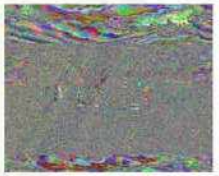
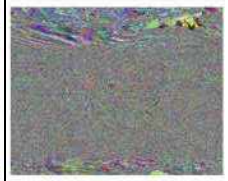
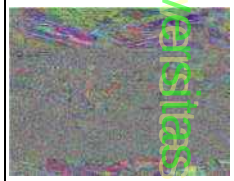
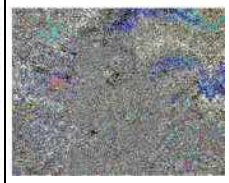
Ukuran matriks : 256 × 256, Kanal *Blue*

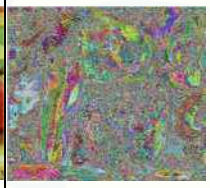
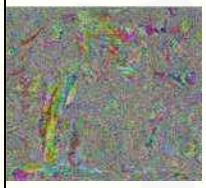
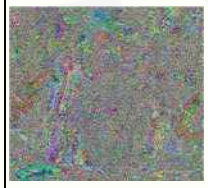
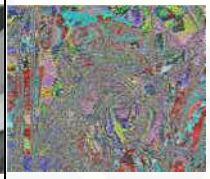
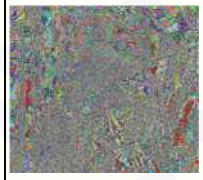
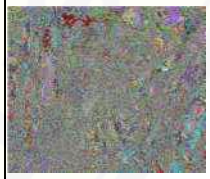
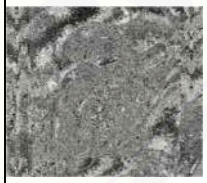
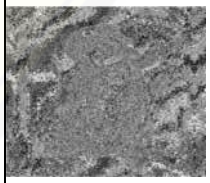
	1	2	3	4	5	6	7	8	9	10	11	12	...	256
1	**180**	**183**	**25**	**154**	**36**	**221**	119	57	211	208	111	203	...	84
2	126	20	64	176	114	181	8	30	119	240	47	142	...	137
3	169	255	182	58	0	84	29	12	106	157	91	245	...	136
4	120	192	12	111	109	171	134	137	241	144	61	161	...	31
5	139	51	31	112	237	204	100	199	56	176	174	34	...	206
6	207	205	26	24	90	182	239	156	171	98	148	165	...	96
7	188	103	90	57	230	233	166	176	18	168	151	5	...	229
8	240	157	107	182	20	142	251	115	22	237	26	101	...	99
9	250	136	39	83	125	71	220	241	33	251	196	151	...	32
10	197	212	0	70	242	6	61	56	74	222	145	26	...	47
11	12	132	178	201	140	221	117	190	112	44	31	188	...	36
12	12	195	126	203	119	93	169	204	107	19	97	77	...	127
13	53	156	239	243	58	54	118	164	252	175	155	76	...	129
14	135	114	65	178	31	61	96	36	16	126	38	164	...	1
15	182	92	186	53	31	20	100	29	188	206	207	99	...	34
16	113	197	191	203	138	255	135	241	101	134	163	105	...	222
17	197	199	48	118	227	168	123	217	98	218	109	67	...	86
18	206	212	47	115	121	208	190	195	104	197	89	85	...	228
19	183	188	72	63	131	241	113	179	60	188	70	6	...	216
20	71	39	31	211	41	166	201	113	58	36	6	144	...	123
21	176	158	224	0	179	163	84	92	242	119	159	224	...	208
22	84	98	107	175	43	206	173	15	177	36	185	212	...	49
23	131	55	47	129	136	211	224	25	235	190	55	210	...	233
24	245	88	71	110	89	47	209	191	182	44	144	4	...	98
25	106	93	95	245	96	183	175	165	200	157	206	94	...	196
26	201	69	162	173	247	11	32	177	66	208	249	111	...	28
27	112	71	81	239	62	53	201	140	40	14	254	164	...	2
28	163	168	63	189	88	80	23	97	228	69	255	130	...	164
29	81	142	57	91	44	77	51	106	91	151	76	130	...	203
30	20	111	244	115	202	224	182	32	230	3	222	91	...	151
31	103	19	83	107	16	81	114	175	20	43	42	111	...	103
32	65	27	65	203	57	209	254	169	79	94	23	229	...	147
33	75	26	198	187	61	75	252	47	69	153	178	61	...	200
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
256	192	245	197	85	110	110	187	19	252	145	160	75	...	167

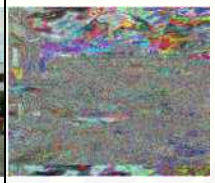
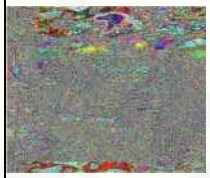
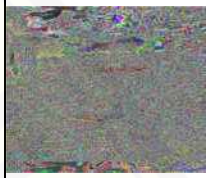
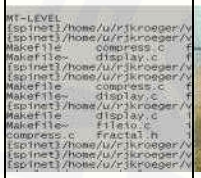
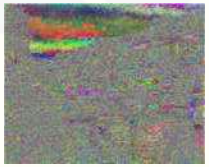
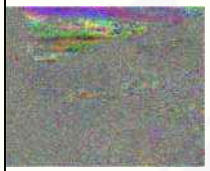
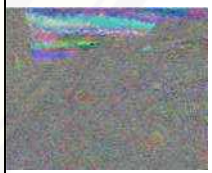
LAMPIRAN E. Hasil Enkripsi *Plain Image*

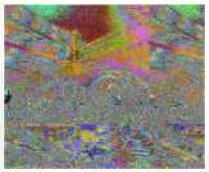
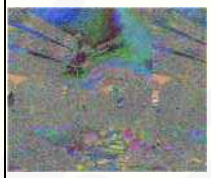
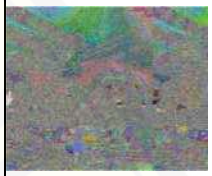
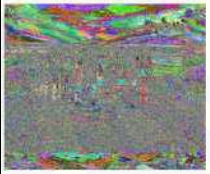
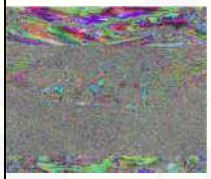
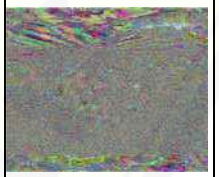
No	*Plain Image*	Kunci	*Cipher Image*							
			Bitshift 0	Bitshift 1	Bitshift 2	Bitshift 3	Bitshift 4	Bitshift 5	Bitshift 6	Bitshift 7
1	 (256 × 256)	 (128 × 128)								
2	 (512×512)	 (512×512)								
3	 (512 × 512)	 (576 × 576)								
4	 (576 × 576)	 (512 × 512)								

No	*Plain Image*	Kunci	*Cipher Image*							
			Bitshift 0	Bitshift 1	Bitshift 2	Bitshift 3	Bitshift 4	Bitshift 5	Bitshift 6	Bitshift 7
5	 (512 × 512)	 (640 × 480)								
6	 (256 × 256)	 (678 × 508)								
7	 (640 × 480)	 (442 × 786)								
8	 (640 × 480)	 (752 × 564)								

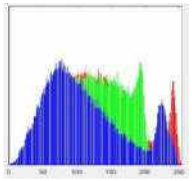
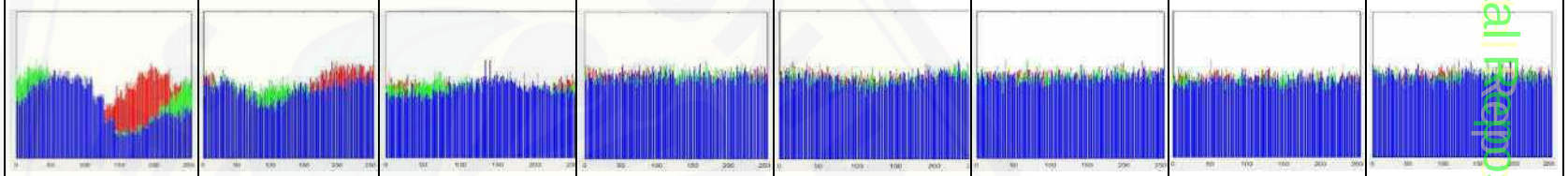
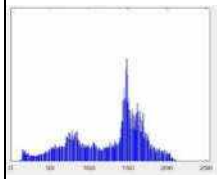
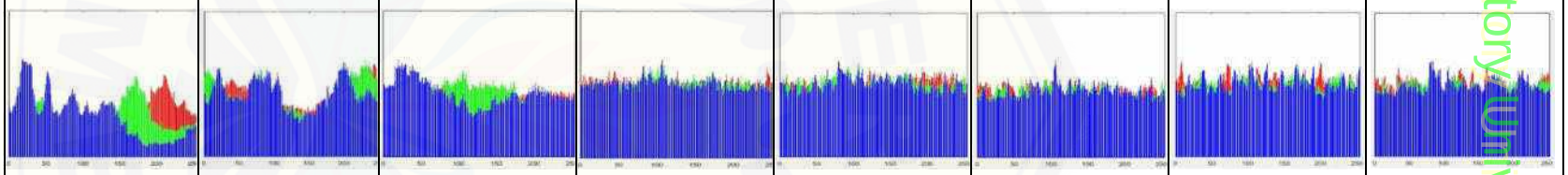
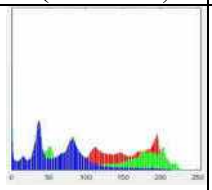
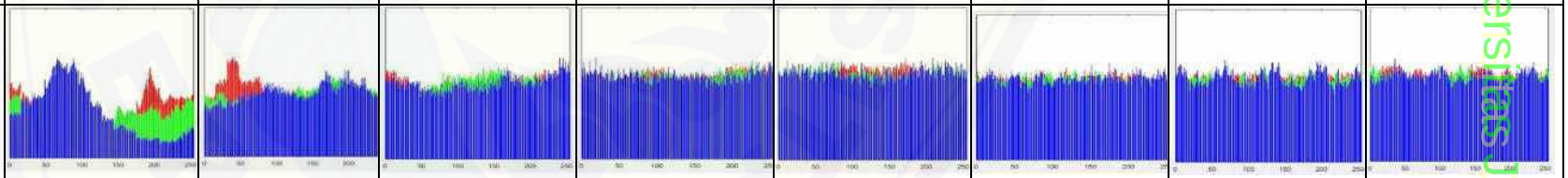
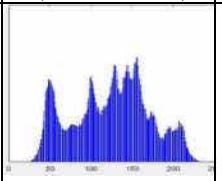
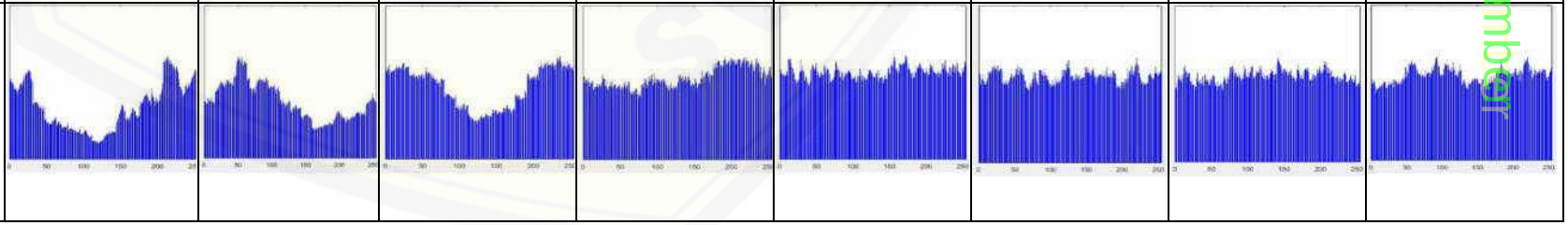
No	*Plain Image*	Kunci	*Cipher Image*							
			Bitshift 0	Bitshift 1	Bitshift 2	Bitshift 3	Bitshift 4	Bitshift 5	Bitshift 6	Bitshift 7
9										
	(800 × 600)	(582 × 437)								
10										
	(678 × 508)	(640 × 480)								
11										
	(752 × 564)	(678 × 508)								
12										
	(582 × 437)	(512 × 512)								

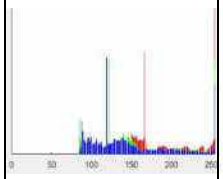
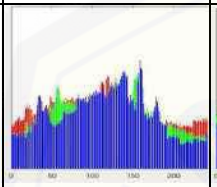
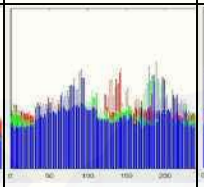
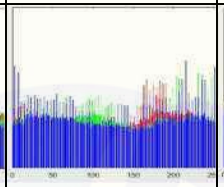
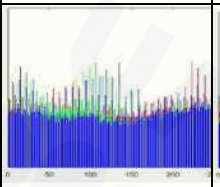
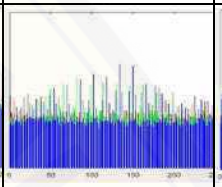
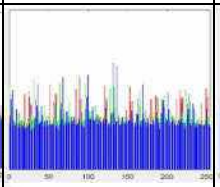
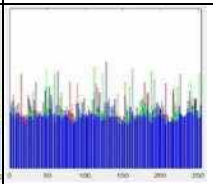
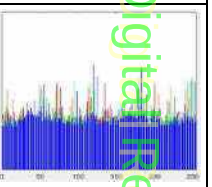
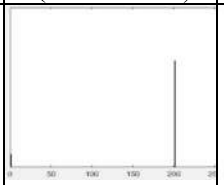
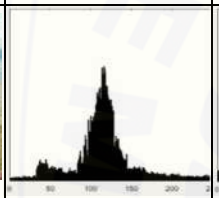
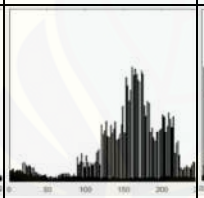
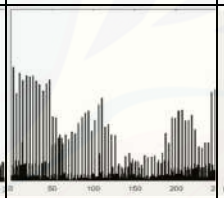
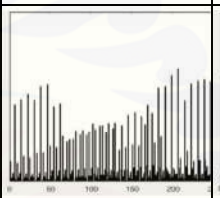
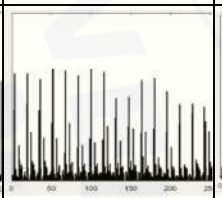
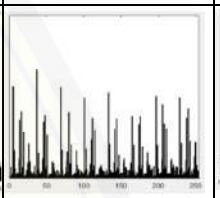
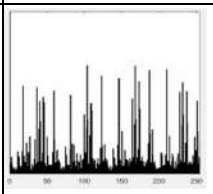
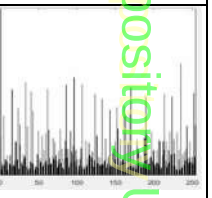
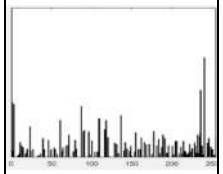
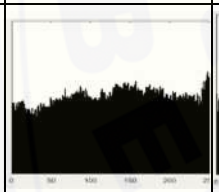
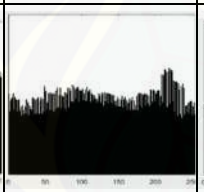
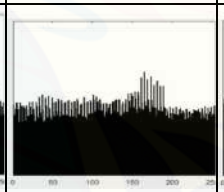
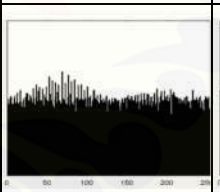
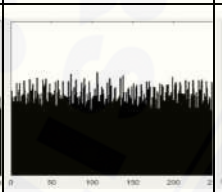
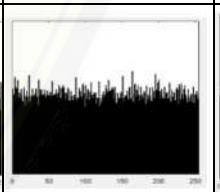
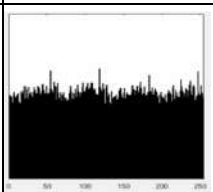
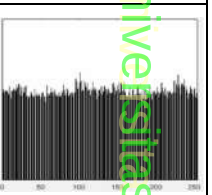
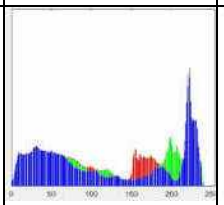
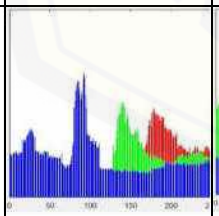
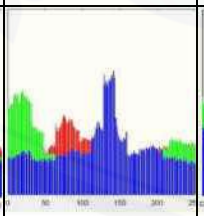
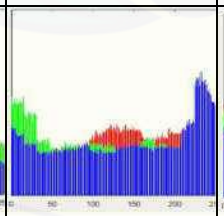
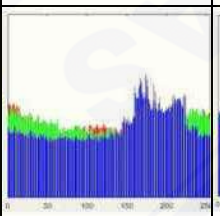
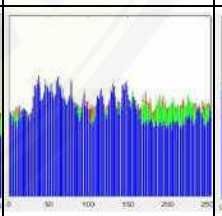
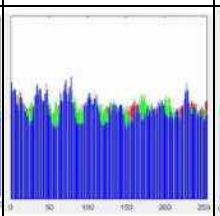
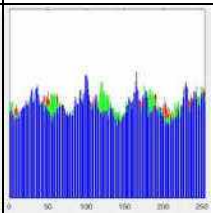
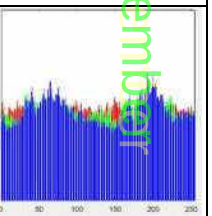
No	*Plain Image*	Kunci	*Cipher Image*		
			Bitshift 8	Bitshift 9	Bitshift 10
1	 (256 × 256)	 (128 × 128)			
2	 (512×512)	 (512×512)			
3	 (512 × 512)	 (576 × 576)			
4	 (576 × 576)	 (512 × 512)			

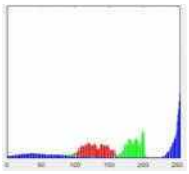
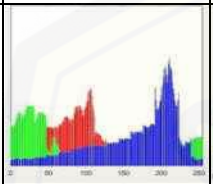
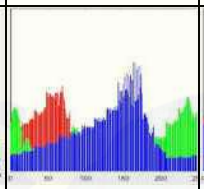
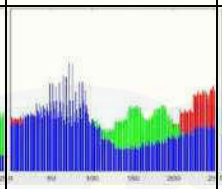
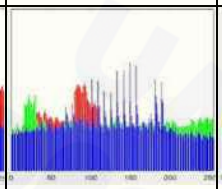
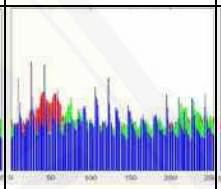
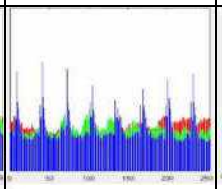
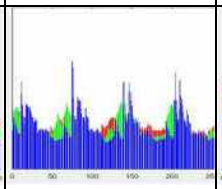
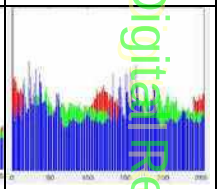
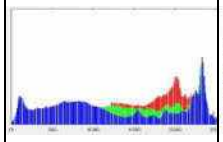
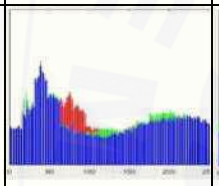
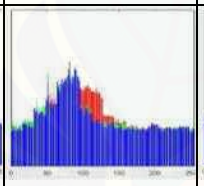
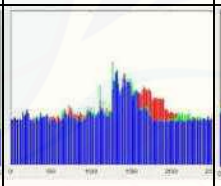
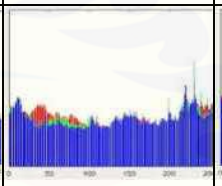
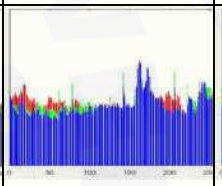
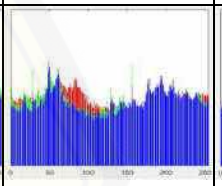
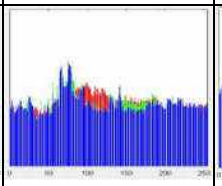
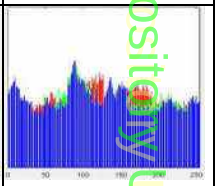
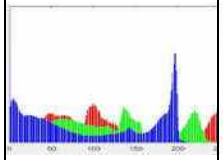
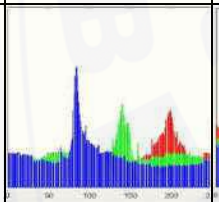
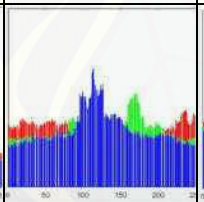
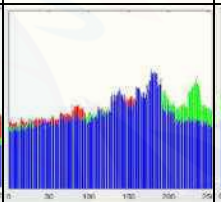
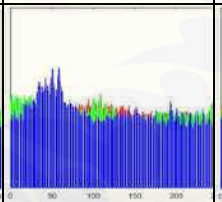
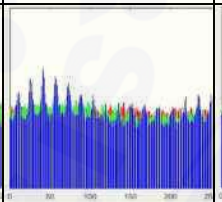
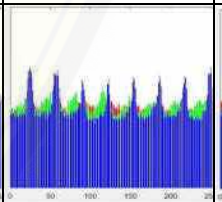
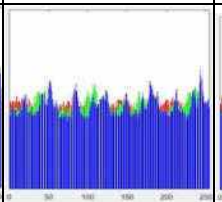
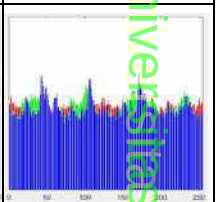
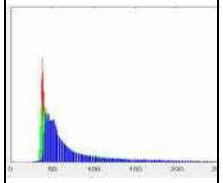
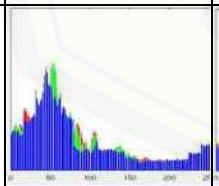
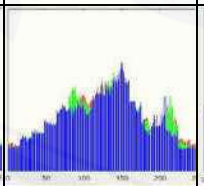
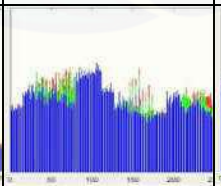
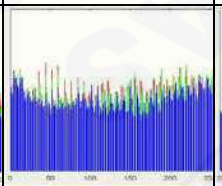
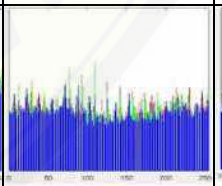
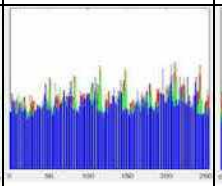
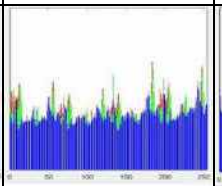
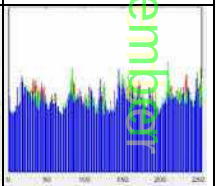
No	*Plain Image*	Kunci	*Cipher Image*		
			Bitshift 8	Bitshift 9	Bitshift 10
5	 (512 × 512)	 (640 × 480)			
6	 (256 × 256)	 (678 × 508)			
7	 (640 × 480)	 (442 × 786)			
8	 (640 × 480)	 (752 × 564)			

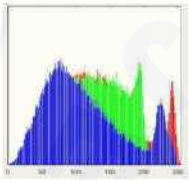
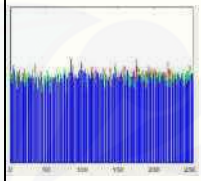
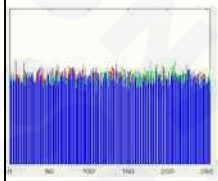
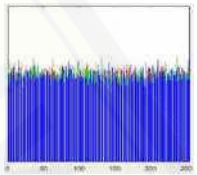
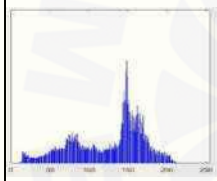
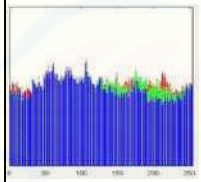
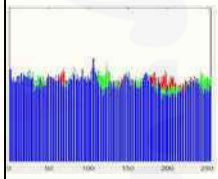
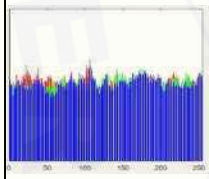
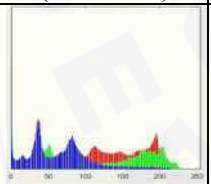
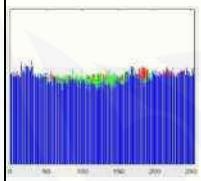
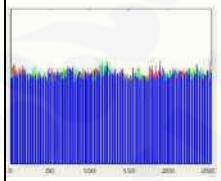
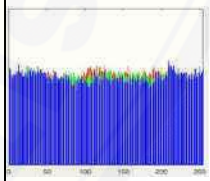
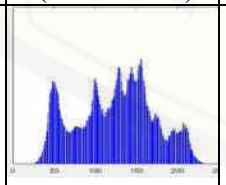
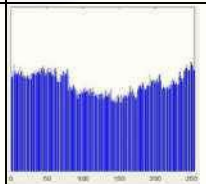
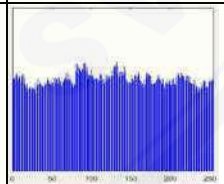
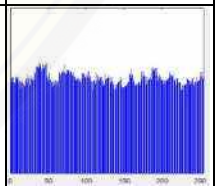
No	*Plain Image*	Kunci	*Cipher Image*		
			Bitshift 8	Bitshift 9	Bitshift 10
9	 (800 × 600)	 (582 × 437)			
10	 (678 × 508)	 (640 × 480)			
11	 (752 × 564)	 (678 × 508)			
12	 (582 × 437)	 (512 × 512)			

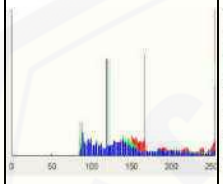
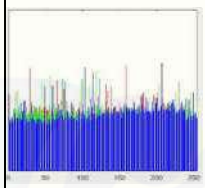
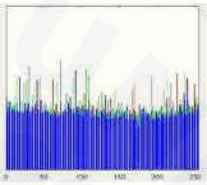
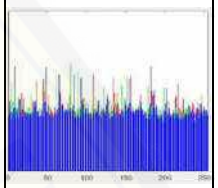
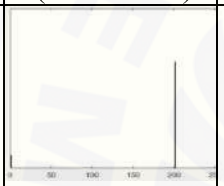
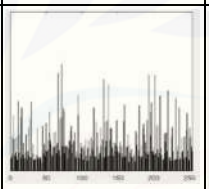
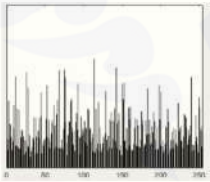
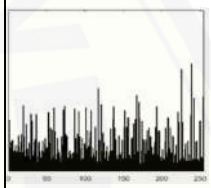
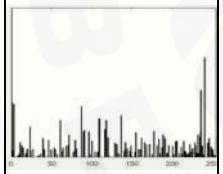
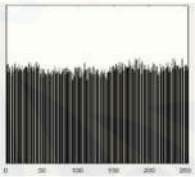
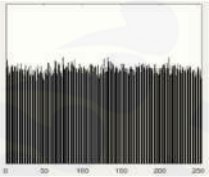
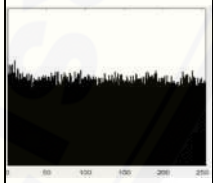
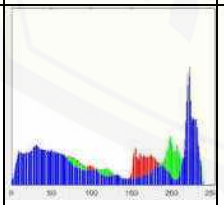
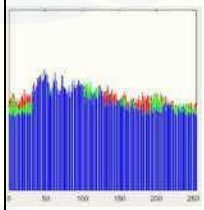
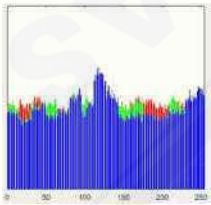
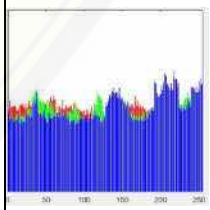
LAMPIRAN F. Hasil Analisis Histogram *Plain Image* dan *Cipher Image*

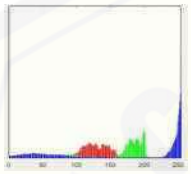
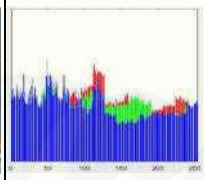
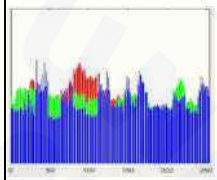
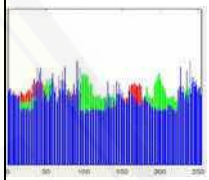
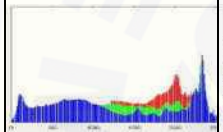
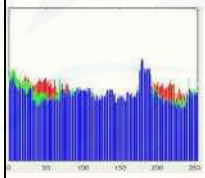
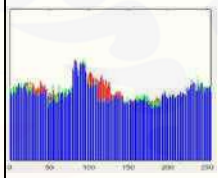
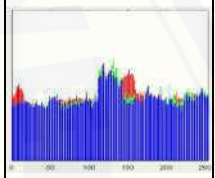
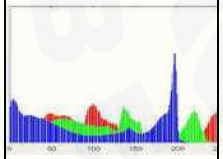
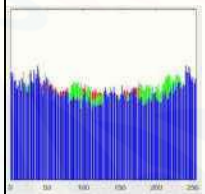
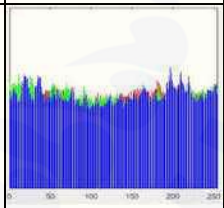
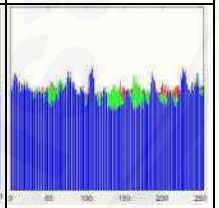
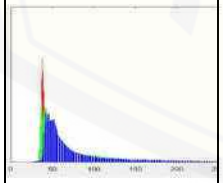
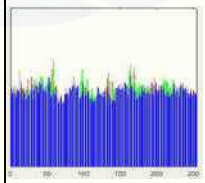
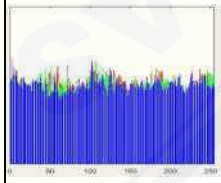
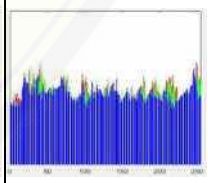
No	*Plain Image*	Kunci	*Cipher Image*							
			Bitshift 0	Bitshift 1	Bitshift 2	Bitshift 3	Bitshift 4	Bitshift 5	Bitshift 6	Bitshift 7
1	 (256 × 256)	 (128 × 128)								
2	 (512×512)	 (512×512)								
3	 (512 × 512)	 (576 × 576)								
4	 (576 × 576)	 (512 × 512)								

No	*Plain Image*	Kunci	*Cipher Image*							
			Bitshift 0	Bitshift 1	Bitshift 2	Bitshift 3	Bitshift 4	Bitshift 5	Bitshift 6	Bitshift 7
5	 (512 × 512)	 (640 × 480)								
6	 (256 × 256)	 (678 × 508)								
7	 (640 × 480)	 (442 × 786)								
8	 (640 × 480)	 (752 × 564)								

No	*Plain Image*	Kunci	*Cipher Image*							
			Bitshift 0	Bitshift 1	Bitshift 2	Bitshift 3	Bitshift 4	Bitshift 5	Bitshift 6	Bitshift 7
9	 (800 × 600)	 (582 × 437)								
10	 (678 × 508)	 (640 × 480)								
11	 (752 × 564)	 (678 × 508)								
12	 (582 × 437)	 (512 × 512)								

No	*Plain Image*	Kunci	*Cipher Image*		
			Bitshift 8	Bitshift 9	Bitshift 10
1	 (256 × 256)	 (128 × 128)			
2	 (512×512)	 (512×512)			
3	 (512 × 512)	 (576 × 576)			
4	 (576 × 576)	 (512 × 512)			

No	*Plain Image*	Kunci	*Cipher Image*		
			Bitshift 8	Bitshift 9	Bitshift 10
5	 (512 × 512)	 (640 × 480)			
6	 (256 × 256)	 (678 × 508)			
7	 (640 × 480)	 (442 × 786)			
8	 (640 × 480)	 (752 × 564)			

No	*Plain Image*	Kunci	*Cipher Image*		
			Bitshift 8	Bitshift 9	Bitshift 10
9	 (800 × 600)	 (582 × 437)			
10	 (678 × 508)	 (640 × 480)			
11	 (752 × 564)	 (678 × 508)			
12	 (582 × 437)	 (512 × 512)			

Lampiran G. Skrip Bitshift (pergeseran bit)

```
function Output=bit1shift(Input)
Output=Input;
Output(Input<128)=Input(Input<128)*2;
Output(Input>=128)=bitxor(mod(Input(Input>=128)*2,256),27);
```

Lampiran H. Skrip Konversi desimal ke biner

```
function bin=dec2binary(dec)
%8 bit binary
[m,n]=size(dec);
if m*n==1
    temp=dec2bin(dec);
    if length(temp)~=8
        bin=[repmat('0',1,8-length(temp)) temp];
    else
        bin=temp;
    end
else
    for i=1:m
        for j=1:n
            temp=dec2bin(dec(i,j));
            if length(temp)~=8
                bin{i,j}=[repmat('0',1,8-length(temp)) temp];
            else
                bin{i,j}=temp;
            end
        end
    end
end
```

Lampiran I. Skrip *Crossover*

```
function [p1, p2]=Crossover(p1,p2)
b1=dec2bin(p1,8);
b2=dec2bin(p2,8);
t1=b1;
t2=b2;
t1(1:4)=b2(1:4);
t2(1:4)=b1(1:4);
p1=bin2dec(t1);
p2=bin2dec(t2);
```

Lampiran J. Skrip Koefisien Korelasi

```
function Corr=CoefCorr(plainimage,cipherimage)
[m,n,o]=size(plainimage);
mup=sum(sum(sum(plainimage)))/(m*n*o);
muc=sum(sum(sum(cipherimage)))/(m*n*o);
sigp=sqrt(sum(sum(sum((plainimage-mup).^2))));
sigc=sqrt(sum(sum(sum((cipherimage-muc).^2))));
Corr=sum(sum(sum((plainimage-mup).*(cipherimage-
muc))))/(sigp*sigc);
```

Lampiran K. Skrip Nilai NPCR

```
function Npcr=NPCRcal(plainimage,cipherimage)
[m,n,o]=size(plainimage);
dij=plainimage-cipherimage;
dij(dij~=0)=1;
Npcr=sum(sum(sum(dij)))/(m*n*o)*100;
```

Lampiran L. Skrip Nilai UACI

```
function Uaci=UACIcal(plainimage,cipherimage)
[m,n,o]=size(plainimage);
Uaci=sum(sum(sum((abs(plainimage-cipherimage))/255)))/(m*n*o)*100;
```

Lampiran M. Skrip Proses Enkripsi

```
Function CipherImage=EncryptProcess(PlainImage,KeyImage,numbitshift)
[m1,n1,o1]=size(PlainImage);
if mod(n1,2)~=0
    PlainImage(:,n1+1,:)=repmat(255,m1,1,o1);
end
[m1,n1,o1]=size(PlainImage);

%%Build Key Image
[m2,n2,o2]=size(KeyImage);
for i=1:numbitshift
    KeyImage=bit1shift(KeyImage);
end
KeyImage=repmat(KeyImage,ceil(m1/m2),ceil(n1/n2));
if o1==1 && o2==3
    KeyImage=KeyImage(:,:,1);
elseif o1==3 && o2==1
    KeyImage=repmat(KeyImage,1,1,3);
end
KeyImage=KeyImage(1:m1,1:n1,:);

%% Process
CipherImage=PlainImage;
CrossoverPattern=importdata('CrossoverPattern.mat');
for k=1:o1

CipherImage(:,:,k)=mod(CipherImage(:,:,k)+KeyImage(:,:,k),256);
    %crossover
    for i=1:m1
        for j=1:2:n1
            temp1=CipherImage(i,j,k);
            temp2=CipherImage(i,j+1,k);
            CipherImage(i,j,k)=CrossoverPattern(temp1+1,temp2+1);

CipherImage(i,j+1,k)=CrossoverPattern(temp2+1,temp1+1);
        end
    end
    %mutation
    CipherImage(:,:,k)=255-CipherImage(:,:,k);
end
```

Lampiran N. Skrip Proses Dekripsi

```
function PlainImage=DecryptProcess(CipherImage,KeyImage,numbitshift)
%%size Cipher Image
[m1,n1,o1]=size(CipherImage);

%%Build Key Image
[m2,n2,o2]=size(KeyImage);
for i=1:numbitshift
    KeyImage=bit1shift(KeyImage);
end
KeyImage=repmat(KeyImage,ceil(m1/m2),ceil(n1/n2));
if o1==1 && o2==3
    KeyImage=KeyImage(:,:,1);
elseif o1==3 && o2==1
    KeyImage=repmat(KeyImage,1,1,3);
end
KeyImage=KeyImage(1:m1,1:n1,:);

%% Process
PlainImage=CipherImage;
CrossoverPattern=importdata('CrossoverPattern.mat');
for k=1:o1
    %mutation
    PlainImage(:,:,k)=255-PlainImage(:,:,k);
    %crossover
    for i=1:m1
        for j=1:2:n1
            temp1=PlainImage(i,j,k);
            temp2=PlainImage(i,j+1,k);
            PlainImage(i,j,k)=CrossoverPattern(temp1+1,temp2+1);
            PlainImage(i,j+1,k)=CrossoverPattern(temp2+1,temp1+1);
        end
    end
    %mod
    PlainImage(:,:,k)=mod(PlainImage(:,:,k)-KeyImage(:,:,k),256);
end
```

Lampiran O. Skrip Histogram

```
function Histogram(cipherimage,vpop)
o=size(cipherimage,3);
color='rgb';
if vpop==1
    for i=1:o
        h=imhist(uint8(cipherimage(:,:,i)));
        if o==3
            bar(0:255,h,0,color(i),'EdgeColor',color(i)); %0
adalah ketebalan batang histogram
        else
            bar(0:255,h,0,'k','EdgeColor','k'); %k warna hitam
            bar(h,'k','EdgeColor','k');
        end
        hold on
        maxv(i)=max(h);
    end
```

```
    hold off
    xlim([0 255]); ylim([0 1.5*max(maxv)]);
elseif vpop==2
    h=imhist(uint8(cipherimage(:,:,vpop-1)));
    bar(0:255,h,0,color(vpop-1),'EdgeColor',color(vpop-1));
    xlim([0 255]); ylim([0 1.5*max(h)]);
elseif vpop==3
    h=imhist(uint8(cipherimage(:,:,vpop-1)));
    bar(0:255,h,0,color(vpop-1),'EdgeColor',color(vpop-1));
    xlim([0 255]); ylim([0 1.5*max(h)]);
elseif vpop==4
    h=imhist(uint8(cipherimage(:,:,vpop-1)));
    bar(0:255,h,0,color(vpop-1),'EdgeColor',color(vpop-1));
    xlim([0 255]); ylim([0 1.5*max(h)]);
end
set(gca,'YTick',[]);
```