PENGKODEAN CITRA DIGITAL HASIL STEGANOGRAFI DENGAN METODE *LEAST SIGNIFICANT BIT* UNTUK DATA TEKS TERENKRIPSI DENGAN ALGORITMA *HILL CIPHER*

SKRIPSI

Oleh

Ginanjar Tegar Rosdiana
NIM 111810101046

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2015

PENGKODEAN CITRA DIGITAL HASIL STEGANOGRAFI DENGAN METODE *LEAST SIGNIFICANT BIT* UNTUK DATA TEKS TERENKRIPSI DENGAN ALGORITMA *HILL CIPHER*

SKRIPSI

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

Oleh

Ginanjar Tegar Rosdiana
NIM 111810101046

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2015

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. Allah SWT yang telah memberikan kehidupan yang penuh makna;
2. Ayahanda Suparno dan Almarhumah Ibunda Rosidah tercinta, motivator terbesar dalam hidup yang telah memberikan segala cinta dan kasih tanpa pamrih dan segala pengorbanan tanpa mengharapkan imbalan;
3. Kakakku Tegar Parnandi Galih Rosdian dan adik-adikku tersayang, Ana Sarofatin dan Safira Rahmah yang selalu memberikan motivasi;
4. guru-guru sejak taman kanak-kanak sampai perguruan tinggi, yang telah memberikan ilmu dan membimbing dengan penuh kesabaran;
5. Almamater Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam.

MOTTO

Allah tidak akan membebani seseorang melainkan sesuai dengan kesanggupannya (Terjemahan QS. Al- Baqarah :286)*)

Jangan bersedih, sebab bersabar atas sesuatu yang tidak anda sukai dan dalam menghadapi kesulitan adalah jalan menuju kemenangan, kesuksesan, dan kebahagiaan. **)

*) Departemen Agama Republik Indonesia. 2006. *Al Qur'an dan Terjemahannya*. Bandung: CV Penerbit Diponegoro.

**) Al-Qarni, 'Aidh. 2004. *La Tahzan*. Jakarta: Qisthi Press.

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

nama : Ginanjar Tegar Rosdiana

NIM : 111810101046

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "Pengkodean Citra Digital Hasil Steganografi dengan Metode *Least Significant Bit* untuk Data Teks Terenkripsi dengan Algoritma *Hill Cipher*" adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi mana pun, 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 di kemudian hari pernyataan ini tidak benar.

Jember, Juni 2015

Yang menyatakan,

Ginanjar Tegar Rosdiana

NIM 111810101046

SKRIPSI

PENGKODEAN CITRA DIGITAL HASIL STEGANOGRAFI DENGAN METODE *LEAST SIGNIFICANT BIT* UNTUK DATA TEKS TERENKRIPSI DENGAN ALGORITMA *HILL CIPHER*

Oleh

Ginanjar Tegar Rosdiana
NIM 111810101046

Pembimbing:

Dosen Pembimbing Utama	: Ahmad Kamsyakawuni, S.Si., M.Kom.
Dosen Pembimbing Anggota	: Dr. Alfian Futuhul Hadi, S.Si., M.Si.

PENGESAHAN

Skripsi berjudul "Pengkodean Citra Digital Hasil Steganografi dengan Metode *Least Significant Bit* untuk Data Teks Terenkripsi dengan Algoritma *Hill Cipher*" telah diuji dan disahkan pada:

hari, tanggal :

tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Tim Penguji:

Ketua,	Sekretaris,
Ahmad Kamsyakawuni, S.Si., M.Kom. NIP. 197211291998021001	Dr. Alfian Futuhul Hadi, S.Si., M.Si NIP. 197407192000121001
Penguji I,	Penguji II,
Kusbudiono, S.Si., M.Si. NIP. 197704302005011001	M. Ziaul Arif, S.Si., M.Sc. NIP 198501112008121002

Mengesahkan

Dekan,

Prof. Drs. Kusno, DEA., Ph.D.
NIP 196101081986021001

RINGKASAN

Pengkodean Citra Digital Hasil Steganografi dengan Metode *Least Significant Bit* untuk Data Teks Terenkripsi dengan Algoritma *Hill Cipher*; Ginanjar Tegar Rosdiana, 111810101046; 2015; 52 Halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Kemajuan teknologi informasi yang semakin pesat dapat meningkatkan tindak kejahatan terhadap pengaksesan informasi tersebut. Oleh karena itu, diperlukan adanya penyandian atau pengkodean untuk menjaga keamanan informasi. Teknik pengamanan informasi yang dapat digunakan adalah kriptografi dan steganografi. Kriptografi adalah ilmu dan seni yang digunakan untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain. Salah satu algoritma kriptografi klasik adalah *Hill Cipher*, yaitu algoritma kunci simetris dan dikategorikan sebagai *block cipher* karena pesan yang akan diproses akan dibagi menjadi blok-blok dengan ukuran tertentu. Steganografi adalah ilmu dan seni yang digunakan untuk menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui. Salah satu metode steganografi yang dapat digunakan adalah *Least Significant Bit* (LSB), yaitu metode yang mengganti *bit* yang paling rendah pada beberapa *byte* media penyembunyian dengan *bit* data secara berurutan. Tujuan penelitian ini adalah untuk menerapkan steganografi pada citra digital menggunakan metode *Least Significant Bit* dan mengaplikasikan algoritma *Hill Cipher* dalam proses enkripsi dan dekripsi pesan teks maupun citra. Hasil dari penelitian ini diharapkan dapat membangun suatu sistem keamanan yang dapat berguna bagi masyarakat dalam berkomunikasi (bertukar pesan).

Data yang digunakan dalam penelitian ini adalah *printable characters* dan nilai RGB dari setiap piksel citra digital. Citra digital yang digunakan adalah citra

warna yang memiliki komponen RGB. *Printable characters* dan nilai RGB tersebut digunakan sebagai *plaintext* dan *ciphertext*. Pada penelitian ini, *plaintext* (berupa data teks) dienkripsi dengan menggunakan algoritma *Hill Cipher* sehingga dihasilkan *ciphertext*. Matriks kunci yang digunakan dalam proses enkripsi pesan teks merupakan matriks persegi berordo 3×3. Kemudian *ciphertext* tersebut disisipkan di dalam media penyembunyian berupa citra digital dengan menggunakan metode *Least Significant Bit*. Citra digital yang telah disisipkan pesan tersebut dienkripsi kembali menggunakan algoritma *Hill Cipher*. Matriks kunci yang digunakan dalam proses enkripsi citra digital merupakan matriks persegi berordo 2×2 dan berordo 3×3. Kemudian citra digital tersebut didekripsi dan diekstraksi sehingga didapatkan *ciphertext*. Untuk mendapatkan pesan asli, maka *ciphertext* didekripsi kembali.

Proses enkripsi *plaintext* menghasilkan *ciphertext* yang memiliki susunan karakter yang acak, sehingga *ciphertext* sangat sulit untuk dibaca. *Stego object* yang dihasilkan dari proses *encoding* pesan teks terlihat sama dengan citra aslinya. *Stego object* tidak mengalami perubahan atau perbedaan yang signifikan jika dibandingkan dengan citra asli sebelum disisipkan pesan. Pada proses enkripsi *stego object*, menghasilkan citra kripto yang tidak dapat terlihat citra aslinya. Semakin besar elemen matriks kunci yang digunakan dalam mengenkripsi *stego object*, maka citra kripto yang dihasilkan semakin tidak dapat terdeteksi seperti apa citra aslinya sehingga proses enkripsi citra digital semakin baik. Ukuran matriks kunci yang digunakan dalam proses enkripsi citra digital juga mempengaruhi cepat lambatnya proses enkripsi citra digital. Semakin besar ukuran matriks kunci yang digunakan, maka semakin cepat proses enkripsi citra digital. Proses enkripsi *stego object* dengan menggunakan matriks kunci berordo 2×2 membutuhkan waktu komputasi sebesar 123,2983 detik sedangkan proses enkripsi *stego object* dengan menggunakan matriks kunci berordo 3×3 membutuhkan waktu komputasi sebesar 99,9682 detik.

PRAKATA

Alhamdulillah, puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga tugas akhir yang berjudul "Pengkodean Citra Digital Hasil Steganografi dengan Metode *Least Significant Bit* untuk Data Teks Terenkripsi dengan Algoritma *Hill Cipher*" dapat terselesaikan dengan baik. Tugas akhir ini disusum untuk memenuhi syarat menyelesaikan pendidikan strata satu (S1) pada Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Penulisan tugas akhir ini telah mendapatkan bantuan dari berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Ahmad Kamsyakawuni, S.Si., M.Kom selaku Dosen Pembimbing Utama dan Dr. Alfian Futuhul Hadi, S.Si., M.Si. selaku Dosen Pembimbing Anggota yang telah meluangkan waktu, pikiran, dan perhatian dalam penulisan skripsi ini;
2. Kusbudiono, S.Si., M.Si. dan M. Ziaul Arif, S.Si., M.Sc. selaku dosen penguji atas saran-saran yang diberikan;
3. seluruh staf pengajar Jurusan Matematika Fakultas MIPA Universitas Jember yang telah memberikan ilmu serta bimbingannya sehingga penulis dapat menyelesaikan skripsi ini;
4. seluruh keluarga besar di Sidoarjo yang telah memberikan kasih sayang, doa dan motivasi tiada henti;
5. sahabat terbaik, Renny Faridah "Gepeng" dan Rika Mardiana "Gendut" yang selalu setia memberikan dukungan dan motivasi;
6. sahabat Rempongers: Sa'odah, Gembul, Ncrung, Unyil, Nitung, Rimboy dan Ciprudh yang selalu rempong dan selalu memberikan semangat;
7. keluarga dan teman seperjuangan terbaik, KRAMAT'11 yang telah membagi kisah dan tawa serta memberikan motivasi untuk menyelesaikan tugas akhir ini;

8. keluarga kost belakang Warung Bu Judes yang selalu memberikan dorongan dan semangat terbaik;
9. serta semua pihak yang tidak dapat disebutkan satu per satu.

Penulis juga menerima segala kritik dan saran dari semua pihak demi kesempurnaan tugas akhir ini. Akhirnya penulis berharap, semoga tugas akhir ini dapat bermanfaat.

Jember, Juni 2015 Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Kebutuhan manusia terhadap teknologi informasi tidak dapat diabaikan begitu saja. Manusia membutuhkan teknologi informasi terutama dalam melakukan komunikasi suara, teks dan citra. Seiring berkembangnya zaman, kemajuan teknologi informasi semakin pesat. Pertukaran informasi menjadi lebih cepat dan efisien. Namun, perkembangan teknologi turut pula meningkatkan tindak kejahatan terhadap pengaksesan informasi tersebut. Oleh karena itu, diperlukan adanya penyandian atau pengkodean untuk menjaga keamanan terhadap informasi yang akan dipertukarkan. Teknik pengamanan informasi yang dapat digunakan adalah kriptografi dan steganografi.

Kriptografi adalah ilmu dan seni yang digunakan untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain (Ariyus, 2008). Kriptografi digunakan untuk mengamankan data atau pesan dengan cara mengubahnya menjadi data atau pesan lain dengan algoritma sandi tertentu (enkripsi) kemudian data atau pesan yang telah disandikan tersebut diubah ke data semula (dekripsi) sehingga data atau pesan tersebut tidak dapat diakses oleh sembarang pihak. Kriptografi terbagi menjadi dua, yaitu kriptografi klasik dan kriptografi modern. Kriptografi klasik menggunakan sebuah kunci (kunci simetris) sedangkan kriptografi modern menggunakan dua buah kunci (kunci asimetris). Salah satu algoritma kriptografi klasik adalah *Hill Cipher*, yaitu algoritma kunci simetris dan dikategorikan sebagai *block cipher* karena pesan yang akan dienkripsi dibagi menjadi blok-blok dengan ukuran tertentu.

Steganografi adalah ilmu dan seni yang digunakan untuk menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak

dapat diketahui (Munir, 2006). Salah satu metode steganografi adalah *Least Significant Bit* (LSB), yaitu metode yang mengganti *bit* yang paling rendah pada beberapa *byte* media penyembunyian dengan *bit* data secara berurutan.

Hasibuan (2014) membahas perancangan aplikasi steganografi dengan metode *Least Significant Bit* (LSB) untuk data terenkripsi dari algoritma *Hill Cipher*. Data teks (*plaintext*) terlebih dahulu dienkripsi dengan kunci algoritma *Hill Cipher*. Data teks yang telah disandikan (*ciphertext*) tersebut disembunyikan di dalam media citra digital dengan metode steganografi *Least Significant Bit* (LSB), kemudian data teks dapat diekstraksi dan didekripsi kembali. Menurut Tarigan (2014), algoritma *Hill Cipher* dapat digunakan dalam penyandian data gambar. Jika *plaintext*-nya adalah sebuah gambar, maka nilai RGB dari setiap piksel merupakan nilai *plaintext* dalam bentuk matriks yang digunakan pada proses enkripsi dan dekripsi pesan tersebut.

Pada penelitian ini, penulis akan mengkodekan citra digital hasil steganografi dengan metode *Least Significant Bit* untuk data teks terenkripsi dengan algoritma *Hill Cipher*. Penerapan kriptografi dan steganografi tersebut diharapkan dapat meningkatkan keamanan data teks yang disandikan karena memanfaatkan dua teknik pengamanan data.

1.2 Rumusan Masalah

Permasalahan yang akan dibahas pada penulisan tugas akhir ini adalah:

a. bagaimana proses menyisipkan pesan teks yang telah dienkripsi dengan algoritma *Hill Cipher* ke dalam citra digital menggunakan metode *Least Significant Bit*;
b. bagaimana proses enkripsi dan dekripsi citra digital menggunakan algoritma *Hill Cipher*.

1.3 Batasan Masalah

Batasan masalah yang digunakan pada penulisan tugas akhir ini adalah:

a. karakter yang digunakan pada *plaintext* dan *ciphertext* adalah *printable characters*;
b. citra digital yang digunakan adalah citra warna;
c. matriks kunci yang digunakan dalam proses enkripsi citra digital memiliki determinan matriks $= 1$.

1.4 Tujuan

Tujuan dari penulisan tugas akhir ini adalah:

a. menerapkan steganografi pada citra digital menggunakan metode *Least Significant Bit*;
b. mengaplikasikan algoritma *Hill Cipher* dalam proses enkripsi dan dekripsi pesan teks maupun citra.

1.5 Manfaat

Adapun manfaat yang diharapkan dari penelitian ini adalah:

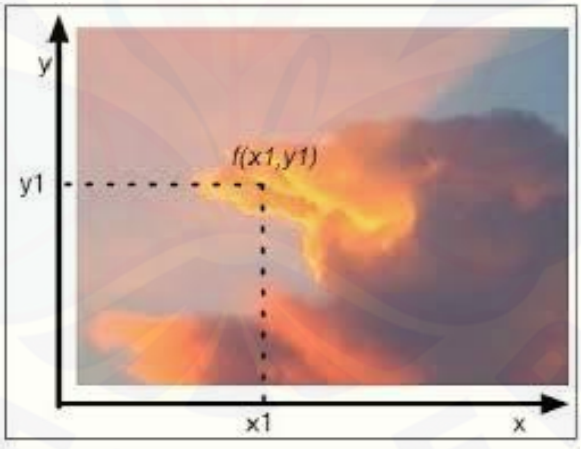
a. mengetahui steganografi citra digital menggunakan metode *Least Significant Bit*;
b. mengetahui cara kerja algoritma *Hill Cipher* dalam proses enkripsi dan dekripsi pesan teks maupun citra;
c. membangun suatu sistem keamanan yang dapat berguna bagi masyarakat dalam berkomunikasi (bertukar pesan).

BAB 2. TINJAUAN PUSTAKA

2.1 Citra Digital

Citra digital adalah citra yang dinyatakan secara diskrit (tidak kontinu), baik untuk posisi koordinatnya maupun warnanya (Prasetyo, 2011). Citra digital dapat digambarkan sebagai suatu matriks dengan indeks baris dan indeks kolom dari matriks yang menyatakan posisi suatu titik di dalam citra dan harga dari elemen matriks menyatakan tingkat keabuan atau warna citra pada titik tersebut. Elemen–elemen matriks suatu citra digital disebut juga dengan istilah piksel yang berasal dari kata *picture element*.

Citra juga dapat didefinisikan sebagai fungsi dua variabel $f(x,y)$. Variabel x dan y adalah koordinat spasial sedangkan nilai $f(x,y)$ adalah intensitas citra pada koordinat tersebut. Ilustrasi citra digital dapat dilihat pada Gambar 2.1.



Gambar 2.1 Ilustrasi citra digital

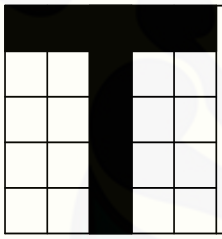
Citra digital dinyatakan dengan matriks berukuran $m \times n$ (m merupakan jumlah baris dan n merupakan jumlah kolom). Setiap titik memiliki koordinat dan dinyatakan dalam bilangan bulat positif (Prasetyo, 2011). Citra digital dinyatakan dengan matriks berukuran $m \times n$ dapat dilihat pada Gambar 2.2.

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,n-1) \\ f(1,0) & f(1,1) & \dots & f(1,n-1) \\ . & . & \dots & . \\ f(m-1,0) & f(m-1,1) & \dots & f(m-1,n-1) \end{bmatrix}$$

Gambar 2.2 Matriks citra digital $m \times n$

2.1.1 Citra Biner

Citra biner diperoleh melalui proses pemisahan piksel-piksel berdasarkan derajat keabuan yang dimilikinya (Ahmad, 2005). Piksel yang memiliki derajat keabuan lebih kecil dari nilai batas yang ditentukan akan diberikan nilai 0, sementara piksel yang memiliki derajat keabuan yang lebih besar dari nilai batas akan diubah menjadi bernilai 1. Gambar 2.3 menunjukan citra biner.



					=	0	0	0	0	0
					=	1	1	0	1	1
					=	1	1	0	1	1
					=	1	1	0	1	1
					=	1	1	0	1	1

$0 =$ warna hitam dan $1 =$ warna putih

Gambar 2.3 Citra biner

2.1.2 Citra Skala Keabuan (*Grayscale*)

Warna yang digunakan pada citra *grayscale* pada umumnya warna hitam sebagai warna minimum dan warna putih sebagai warna maksimum sehingga warna antara kedua warna tersebut adalah abu-abu.

Citra *grayscale* mengandung matriks data yang merepresentasikan nilai dalam suatu range. Elemen-elemen dalam matriks intensitas merepresentasikan berbagai nilai intensitas atau derajat keabuan dengan nilai 0 merepresentaikan warna hitam dan 1 merepresentasikan intensitas penuh atau warna putih (Ahmad, 2005). Gambar 2.4 menunjukkan citra skala keabuan (*grayscale*).

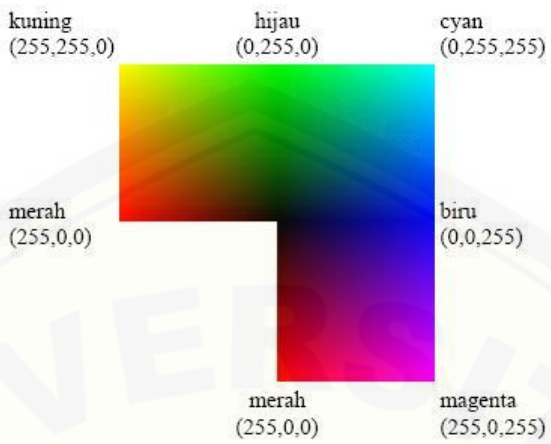
Gambar 2.4 Citra skala keabuan (*grayscale*)

2.1.3 Citra Warna

Menurut Prasetyo (2011) citra warna atau RGB adalah suatu model warna yang terdiri dari warna merah, hijau, dan biru yang digabungkan sehingga membentuk suatu susunan warna yang luas.

Pada monitor komputer, nilai rentang terkecil adalah 0 dan terbesar adalah 255. Pilihan skala 256 ini didasarkan pada cara mengungkap 8 digit bilangan biner yang digunakan oleh mesin komputer. Dengan cara ini, akan diperoleh warna campuran sebanyak $256 \times 256 \times 256 = 16777216$ jenis warna.

Sebuah jenis warna, dapat dibayangkan sebagai sebuah vektor di ruang 3 dimensi yang biasanya dipakai dalam matematika, koordinatnya dinyatakan dalam bentuk tiga bilangan, yaitu komponen-x, komponen-y dan komponen-z. Misalkan sebuah vektor dituliskan sebagai $r = (x, y, z)$. Untuk warna, komponen-komponen tersebut digantikan oleh komponen R(*Red*), G(*Green*) dan B(*Blue*). Jadi, sebuah jenis warna dapat dituliskan sebagai berikut: warna = RGB(30,75,255), Putih = RGB (255,255,255), sedangkan untuk hitam= RGB(0,0,0). Gambar 2.5 menunjukkan contoh kombinasi warna RGB.



Gambar 2.5 Kombinasi warna RGB

2.2 Kriptografi

Kriptografi berasal dari bahasa Yunani, *crypto* dan *graphia*. *Crypto* berarti *secret* (rahasia) dan *graphia* berarti *writing* (tulisan). Menurut terminologinya, kriptografi adalah ilmu dan seni yang digunakan untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain (Ariyus, 2008).

Pada dasarnya komponen kriptografi terdiri dari beberapa komponen, seperti :

a. Enkripsi

   Enkripsi merupakan cara pengamanan data yang dikirimkan sehingga terjaga kerahasiaannya. Enkripsi diartikan sebagai *cipher* atau kode untuk mengubah teks biasa ke dalam bentuk teks kode menggunakan algoritma yang dapat mengkodekan data yang diinginkan.

b. Dekripsi

   Kebalikan dari enkripsi, dekripsi mengembalikan teks yang telah dikodekan ke bentuk asalnya.

c. Kunci

   Kunci digunakan untuk melakukan enkripsi dan dekripsi. Kunci terbagi menjadi dua bagian, yaitu kunci rahasia (*private key*) dan kunci umum (*public key*).

d. *Ciphertext*

Ciphertext merupakan suatu pesan yang telah melalui proses enkripsi. Pesan yang ada pada teks kode tidak dapat dibaca karena berupa karakter-karakter yang tidak dapat diartikan.

e. *Plaintext*

Plaintext merupakan teks asli yang diproses menggunakan algoritma kriptografi untuk diubah menjadi *ciphertext* (teks kode).

f. Pesan

Pesan dapat berupa data atau informasi yang melibatkan dua entitas. Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya , sedangkan penerima (*receiver*) adalah entitas yang menerima pesan.

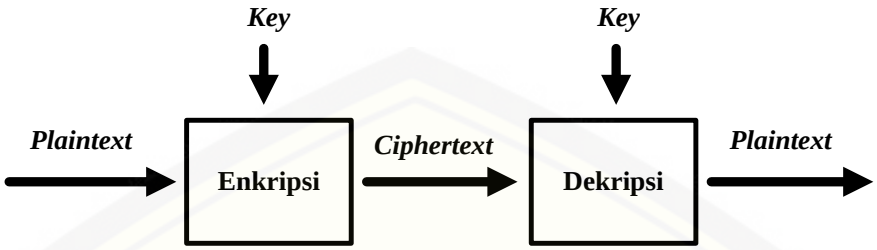
g. Kriptanalisis

Kriptanalisis adalah ilmu dan seni untuk memecahkan *ciphertext* menjadi *plaintext* tanpa mengetahui kunci dan algoritma yang digunakan. Jika suatu teks kode berhasil diubah menjadi teks asli tanpa menggunakan kunci yang digunakan, maka proses tersebut dinamakan *breaking code*. Pelaku kriptanalisis disebut kriptanalis

(Ariyus, 2008).

2.2.1 Prinsip Kerja Kriptografi

Penulisan kriptografi dapat ditulis dalam bahasa matematika. Kriptografi memiliki fungsi-fungsi dasar, yaitu fungsi enkripsi dan fungsi dekripsi. Enkripsi adalah proses mengubah pesan asli (*plaintext*) menjadi pesan yang telah dikodekan (*ciphertext*), sedangkan dekripsi adalah proses mengembalikan pesan yang telah dikodekan menjadi pesan asli semula. Proses enkripsi dan dekripsi ditunjukkan pada Gambar 2.6.



Gambar 2.6 Proses enkripsi dan dekripsi pada kriptografi

Proses enkripsi menyandikan pesan P dengan suatu kunci K sehingga dihasilkan pesan C. Secara sistematis, proses enkripsi dapat ditulis sebagai berikut :

$$C = E(P) \tag{2.1}$$

dimana :

P = pesan asli (*plaintext*);

E = proses enkripsi;

C = pesan yang telah dikodekan (*ciphertext*).

Pada proses dekripsi, pesan C tersebut diuraikan dengan menggunakan kunci K sehingga menghasilkan pesan P. Secara sistematis, proses dekripsi dapat ditulis sebagai berikut :

$$P = D(C) \tag{2.2}$$

dimana :

D = proses dekripsi.

2.2.2 Jenis-Jenis Kunci

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

a. Kunci Simetris

Kunci simetris menggunakan kunci yang sama pada proses enkripsi dan dekripsi. Apabila mengirim pesan menggunakan kunci ini, penerima pesan harus mengetahui kunci dari pesan tersebut agar dapat mendekripsikan pesan yang dikirim. Keamanan pesan yang menggunakan kunci simetris bergantung pada

kunci yang digunakan. Jika kunci tersebut diketahui oleh orang lain maka orang tersebut dapat melakukan enkripsi dan dekripsi terhadap pesan.

b. Kunci Asimetris

Kunci asimetris sering disebut sebagai kunci publik, artinya kata kunci yang digunakan untuk melakukan enkripsi dan dekripsi 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 (dipublikasikan), sedangkan kunci rahasia adalah kunci yang hanya diketahui oleh satu orang (dirahasiakan). Kunci-kunci tersebut saling berhubungan. Kunci umum (*public key*) dapat digunakan untuk mengenkripsi pesan tetapi tidak dapat mendekripsi pesan tersebut. Orang yang dapat mendekripsi pesan tersebut adalah orang yang memiliki kunci rahasia

(Ariyus, 2008).

2.2.3 Jenis-Jenis Serangan

Serangan (*attack*) adalah setiap usaha (*attempt*) atau percobaan yang dilakukan oleh kriptanalis untuk menemukan kunci dan mengungkap *plaintext* (Munir, 2006). Beberapa jenis serangan yang dapat dilakukan oleh kriptanalis, dengan asumsi bahwa kriptanalis telah mengetahui algoritma kriptografi yang digunakan, antara lain :

a. *Ciphertext Only Attack*, yaitu penyerang hanya mendapatkan pesan yang telah dirahasiakan.
b. *Known Plaintext Attack*, yaitu dimana selain mendapatkan sandi, penyerang juga mendapatkan pesan asli.
c. *Chosen Plaintext Attack*, sama dengan *known plaintext attack*, namun penyerang bahkan dapat memilih penggalan dari pesan asli yang akan disandikan.

2.3 Matriks

Menurut Suryadi dan Machmudi (1985), matriks adalah himpunan skalar yang disusun atau dijajarkan menurut baris dan kolom. Misalkan matriks $A = (a_{ij}), i = 1,2, \dots m;\ j = 1,2, \dots n$ artinya bahwa jumlah baris adalah m dan jumlah kolom adalah n.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ . & . & . & . \\ . & . & . & . \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Jika $m = n$, maka matriks tersebut dinamakan matriks persegi atau bujur sangkar. Jika elemen matriks $a_{ij} = 1$ untuk $i = j$ dan $a_{ij} = 0$ untuk $i \neq j$, maka matriks tersebut dinamakan matriks identitas (I). Sebuah matriks B disebut invers dari matriks A jika $AB = I$. B dapat ditulis A^{-1}.

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Macam-macam operasi hitung yang dilakukan terhadap matriks adalah operasi penjumlahan dua buah matriks, perkalian skalar terhadap matriks dan perkalian dua buah matriks.

a. Penjumlahan Matriks (berlaku untuk matriks-matriks berukuran sama)

Jika $A = (a_{ij})$ dan $B = (b_{ij})$ matriks berukuran sama, maka $A + B$ adalah suatu matriks $C = (c_{ij})$ dimana $(c_{ij}) = (a_{ij}) + (b_{ij})$ dengan menjumlahkan setiap elemen matriks yang memiliki posisi sama.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a+e & b+f \\ c+g & d+h \end{bmatrix}$$

b. Perkalian Skalar terhadap Matriks

Jika λ suatu skalar (bilangan) dan $A = (a_{ij})$ maka matriks $\lambda A = (\lambda a_{ij})$, artinya matriks λA diperoleh dengan mengalikan semua elemen matriks A dengan λ.

$$\lambda \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} \lambda a & \lambda b \\ \lambda c & \lambda d \end{bmatrix}$$

c. Perkalian Matriks

Jika $A = (a_{ij})$ berukuran $(p \times q)$ dan $B = (b_{ij})$ berukuran $(q \times r)$, maka perkalian matriks AB adalah suatu matriks $C = (c_{ij})$ berukuran $(p \times r)$ dimana $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{iq}b_{qj}$, untuk setiap $i = 1,2, \dots, p$ dan $j = 1,2, \dots, r$.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \times \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

2.4 Hitungan Modulo

Di dalam hitungan modulo, diberikan sebuah bilangan bulat positif m yang disebut modulo, dan dua bilangan bulat sebarang yang selisihnya sebesar kelipatan bulat dari modulo tersebut dianggap sama atau ekuivalen dengan modulo tersebut (Anton dan Rorres, 1998). Berikut ini merupakan definisi, teorema dan akibat yang berhubungan dengan modulo.

Definisi 2.1:

Jika m sebuah bilangan bulat positif dan a serta b adalah bilangan bulat sebarang, maka dikatakan bahwa a ekuivalen dengan b modulo m, ditulis sebagai $a = b\ (mod\ m)$ jika $a - b$ adalah bilangan bulat kelipatan m.

Untuk sebarang modulo m, setiap bilangan bulat a adalah ekuivalen, modulo m, dengan tepat satu dari bilangan-bilangan bulat

$$0,1,2, \dots, m - 1$$

Bilangan bulat ini disebut residu (sisa) dari a modulo m, dan dituliskan:

$$Z_m = \{0,1,2, \dots, m - 1\}$$

Dalam hitungan biasa setiap bilangan tak nol a mempunyai balikan atau invers perkalian yang dinyatakan oleh a^{-1}, sedemikian rupa sehingga

$$aa^{-1} = a^{-1}a = 1$$

Dalam hitungan modulo, konsep yang berpadanan adalah sebagai berikut:

Definisi 2.2

Jika a adalah sebuah bilangan di dalam Z_m, maka sebuah bilangan a^{-1} di dalam Z_m disebut balikan atau invers perkalian dari a modulo m jika $aa^{-1} = a^{-1}a = 1$ (mod m).

Jika a dan m tidak mempunyai faktor prima bersama, maka a mempunyai balikan modulo m, sebaliknya jika a dan m mempunyai faktor prima bersama maka a tidak mempunyai balikan modulo m.

Teorema 2.1

Matriks bujur sangkar A dengan elemen-elemen di Z_m adalah modulo m yang dapat dibalikkan jika dan hanya jika residu (sisa) dari $det(A)$ modulo m mempunyai balikan modulo m.

Akibat 2.1

Matriks bujur sangkar A dengan elemen-elemen di Z_m adalah modulo m yang dapat dibalikkan jika dan hanya jika m dan residu (sisa) dari $det(A)$ modulo m tidak mempunyai faktor prima yang sama.

Definisi 2.1

Determinan dari matriks $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}_{2\times 2}$ adalah $\det(A) = a \cdot d - b \cdot c$

(Anton dan Rorres, 1998).

2.5 Algoritma *Hill Cipher*

Hill Cipher ditemukan pada tahun 1929 oleh Lester S. Hill. *Hill Cipher* merupakan penerapan hitungan modulo pada kriptografi dengan menggunakan matriks $n \times n$ sebagai kunci yang digunakan untuk melakukan enkripsi dan dekripsi.

Hill Cipher dikategorikan sebagai *block cipher* karena pesan yang akan diproses akan dibagi menjadi blok-blok dengan ukuran tertentu. Setiap karakter dalam satu blok akan saling mempengaruhi karakter lainnya dalam proses enkripsi dan dekripsinya sehingga karakter yang sama tidak dipetakan menjadi karakter yang sama pula.

2.5.1 Dasar Teknik *Hill Cipher*

Dasar teknik *Hill Cipher* adalah hitungan modulo terhadap matriks. Dalam penerapannya, *Hill Cipher* menggunakan teknik perkalian matriks dan teknik invers terhadap matriks. Kunci pada *Hill Cipher* adalah matriks $n \times n$ dengan n merupakan ukuran blok. Matriks K yang menjadi kunci dengan syarat determinan $K \neq 0$ dan harus merupakan matriks *invertible*, yaitu memiliki matriks invers K^{-1} sehingga

$$K \cdot K^{-1} = I$$

Kunci harus memiliki invers karena matriks K^{-1} tersebut merupakan kunci yang digunakan untuk melakukan dekripsi.

2.5.2 Teknik Enkripsi pada *Hill Cipher*

Proses enkripsi *Hill Cipher* dilakukan per blok *plaintext*. Ukuran blok tersebut sama dengan ukuran matriks kunci. Sebelum membagi pesan menjadi deretan blok-blok, *plaintext* terlebih dahulu dikonversikan ke dalam bentuk angka. Secara sistematis, proses enkripsi pada *Hill Cipher* adalah

$$C = K \cdot P \tag{2.3}$$

dimana :

$C =$ *ciphertext;*

$K =$ kunci;

$P =$ *plaintext.*

Jika *plaintext*-nya adalah sebuah gambar, maka nilai RGB dari setiap piksel merupakan nilai *plaintext* dalam bentuk matriks yang digunakan pada proses enkripsi. sehingga di dapatkan nilai *plaintext* dalam bentuk matriks sebagai berikut :

$$P = \begin{bmatrix} 165 & 176 \\ 180 & 195 \end{bmatrix}$$

Plaintext tersebut akan dienkripsi menggunakan algoritma *Hill Cipher*, dengan kunci K yang merupakan matriks berordo 2×2. Blok *plaintext* ini kemudian dienkripsi dengan kunci K melalui persamaan berikut :

$$P_{1,2} = \begin{bmatrix} 5 & 3 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 165 \\ 180 \end{bmatrix} = \begin{bmatrix} 5 \times 165 + 3 \times 180 \\ 3 \times 165 + 2 \times 180 \end{bmatrix} = \begin{bmatrix} 1365 \\ 855 \end{bmatrix} (\text{mod } 256) = \begin{bmatrix} 85 \\ 87 \end{bmatrix}$$

$$P_{3,4} = \begin{bmatrix} 5 & 3 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 176 \\ 195 \end{bmatrix} = \begin{bmatrix} 5 \times 176 + 3 \times 195 \\ 3 \times 176 + 2 \times 195 \end{bmatrix} = \begin{bmatrix} 1465 \\ 918 \end{bmatrix} (\text{mod } 256) = \begin{bmatrix} 185 \\ 150 \end{bmatrix}$$

Dari hasil perkalian matriks di atas, maka didapatkan matriks *ciphertext* sebagai berikut:

$$C = \begin{bmatrix} 85 & 185 \\ 87 & 150 \end{bmatrix}$$

(Tarigan, 2014).

2.5.3 Teknik Dekripsi pada *Hill Cipher*

Proses dekripsi *Hill Cipher* pada dasarnya sama dengan proses enkripsinya. Namun, matriks kunci harus dibalik (invers) terlebih dahulu. Secara sistematis, proses dekripsi pada algoritma *Hill Cipher* dapat diturunkan dari persamaan berikut.

$$C = K \cdot P$$

$$K^{-1} \cdot C = K^{-1} \cdot K \cdot P$$

$$K^{-1} \cdot C = I \cdot P$$

$$P = K^{-1} \cdot C$$

Sehingga persamaan proses dekripsi adalah :

$$P = K^{-1} \cdot C \tag{2.4}$$

Proses dekripsi diawali dengan menghitung invers dari matriks K, maka proses dekripsinya adalah sebagai berikut :

$K = \begin{bmatrix} 5 & 3 \\ 3 & 2 \end{bmatrix}$

Det$(K) = (5 \times 2) - (3 \times 3) = 1$, maka untuk mencari K^{-1} adalah

$$K^{-1} = \frac{1}{|K|} Adj(K)$$

$$K^{-1} = \frac{1}{1} \begin{bmatrix} 2 & -3 \\ -3 & 5 \end{bmatrix} = \begin{bmatrix} 2 & -3 \\ -3 & 5 \end{bmatrix}$$

Setiap bilangan yang bernilai negatif dijumlahkan dengan 256 agar bernilai positif, ini digunakan karena nilai piksel citra adalah $0 - 255$.

$$K^{-1} = \begin{bmatrix} 2 & 253 \\ 253 & 5 \end{bmatrix}$$

Selanjutnya, lakukan proses dekripsi dengan mengalikan matriks K^{-1} dengan *ciphertext* yang telah didapatkan sebelumnya.

$$C_{1,2} = \begin{bmatrix} 2 & 253 \\ 253 & 5 \end{bmatrix} \begin{bmatrix} 85 \\ 87 \end{bmatrix} = \begin{bmatrix} 5 \times 85 + 3 \times 87 \\ 3 \times 85 + 2 \times 87 \end{bmatrix} = \begin{bmatrix} 22181 \\ 21940 \end{bmatrix} (\text{mod } 256) = \begin{bmatrix} 165 \\ 180 \end{bmatrix}$$

$$C_{3,4} = \begin{bmatrix} 2 & 253 \\ 253 & 5 \end{bmatrix} \begin{bmatrix} 185 \\ 150 \end{bmatrix} = \begin{bmatrix} 5 \times 185 + 3 \times 150 \\ 3 \times 185 + 2 \times 150 \end{bmatrix} = \begin{bmatrix} 38320 \\ 47555 \end{bmatrix} (\text{mod } 256) = \begin{bmatrix} 176 \\ 195 \end{bmatrix}$$

Setelah semua blok didekripsi, maka didapatkan hasil *plaintext*-nya sebagai berikut :

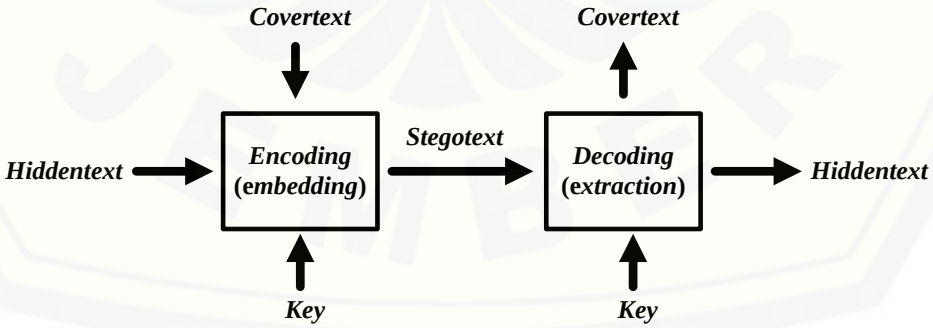
$$P = \begin{bmatrix} 165 & 176 \\ 180 & 195 \end{bmatrix}$$

(Tarigan, 2014).

2.6 Steganografi

Steganografi berasal dari Bahasa Yunani, yaitu "*steganos*" yang artinya tulisan tersembunyi (*covered writing*). Steganografi merupakan ilmu dan seni yang digunakan untuk menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui (Munir, 2006).

Penyisipan pesan ke dalam media *covertext* dinamakan *encoding*, sedangkan ekstraksi pesan dari *stegotext* dinamakan *decoding*. Kedua proses ini memerlukan kunci rahasia agar hanya pihak yang berhak saja yang dapat melakukan penyisipan dan ekstraksi pesan, seperti pada Gambar 2.7 berikut ini.



Gambar 2.7 Proses *encoding* dan *decoding* pesan

Di dalam steganografi citra digital, *hiddentext* atau *embedded message* merupakan teks yang akan disisipkan ke dalam *covertext* atau *cover object*, yaitu file yang digunakan sebagai media penampung pesan yang akan disisipkan. Hasil dari *encoding* atau *embedding* pesan ke dalam file citra akan dihasilkan *stegotext* atau *stego object* yang merupakan file yang berisikan pesan yang telah disisipkan. Gambar 2.8 merupakan contoh dari *hiddentext*, *covertext* dan *stegotext*.

Covertext:
Upakan sal umor tu, aga gar atamu ehat tau urunkan banmu

Hiddentext:
Lari jam Satu

Stegotext:
Lupakan **a**sal **r**umor **i**tu, **j**aga **a**gar **m**atamu **s**ehat **a**tau **t**urunkan **u**banmu

Hiddentext	***Cover Image***	***Stego Image***
Istilah keilmuan serumpun terasa memberikan persepsi pada maksud sebenarnya untuk membentuk rumpun yang mewadahinya		

Gambar 2.8 Contoh *hiddentext*, *covertext* dan *stegotext*

Teknik penyisipan data ke dalam *covertext* dapat dilakukan dalam dua macam domain, yaitu :

d. Domain Spasial (waktu) (*spatial/time domain*)

   Teknik ini memodifikasi langsung nilai *byte* dari *covertext* (nilai *byte* dapat merepresentasikan intensitas atau warna piksel). Metode yang tergolong ke dalam domain spasial adalah metode *Least Significant Bit* (LSB).

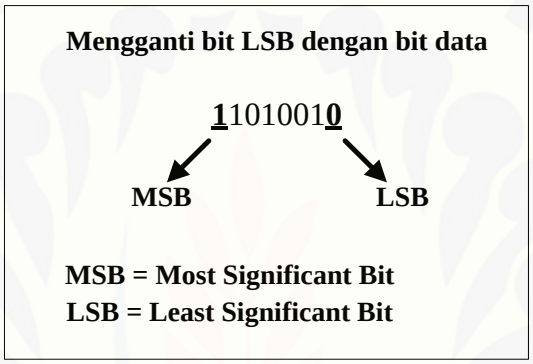
e. Domain Transform (*frequency transform domain*)

   Teknik ini memodifikasi langsung hasil transformasi frekuensi sinyal. Metode yang tergolong ke dalam domain transform adalah *spread spectrum*.

2.7 Metode *Least Significant Bit* (LSB)

Metode *Least Significant Bit* (LSB) merupakan teknik penyembunyian data yang bekerja pada domain spasial atau waktu, yaitu teknik menyembunyikan informasi dengan menyisipkan bit data ke dalam *byte covertext* atau *cover object* (Munir, 2006).

Pada susunan bit dalam *byte* (1 *byte*=8 bit), terdapat bit yang paling berarti (*Most Significant Bit* atau MSB) dan bit yang paling kurang berarti (*Least Significant Bit* atau LSB). Gambar 2.9 merupakan contoh LSB dan MSB.



Gambar 2.9 Contoh LSB dan MSB

Bit yang dapat diganti dengan bit pesan adalah bit LSB karena modifikasi hanya mengubah nilai *byte* tersebut menjadi satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Misalkan *byte* tersebut di dalam sebuah gambar memberikan persepsi warna merah, maka perubahan satu bit LSB hanya mengubah persepsi warna merah tidak terlalu berarti karena mata manusia tidak dapat membedakan perubahan sekecil itu.

Misalkan segmen piksel-piksel citra sebelum penambahan bit-bit pesan adalah.

00110011 10100010 11100010 01101111

Pesan rahasia yang telah dikonversi ke dalam sistem biner adalah 1011. Setiap bit dari pesan tersebut menggantikan posisi LSB dari segmen piksel-piksel citra menjadi:

00110011 10100010 11100011 01101111

Berdasarkan hasil penyisipan atau *embedding* ke dalam sekumpulan piksel citra tersebut akan diperoleh kembali sekumpulan piksel yang telah berubah pada

posisi bit terendah atau LSB dari piksel tersebut, sehingga LSB menggantikan nilai bit-bit terendah dari setiap piksel untuk disisipkan dan digantikan oleh bit yang mengandung pesan (Munir, 2006).

Pesan yang disembunyikan di dalam citra dapat diungkap kembali dengan mengekstrak pesan tersebut. Proses ekstraksi pesan dilakukan dengan mengekstrak LSB dari masing-masing piksel pada *stego object* secara berurutan (Hasibuan, 2014).

2.8 Kode ASCII (*American Standard Code for Information Interchange*)

Kode ASCII merupakan suatu standar internasional dalam kode huruf dan simbol yang bersifat universal. Kode ASCII digunakan oleh komputer dan alat komunikasi lainnya untuk menunjukkan teks (Rachmawanto, 2010).

Karakter kode 32 sampai 127 yang berisi huruf, digit, tanda baca dan beberapa simbol lain yang sebagian besar ditemukan di *keyboard* disebut ASCII *printable characters*. ASCII *printable characters* dapat dilihat pada tabel 2.1 sebagai berikut.

Tabel 2.1 ASCII *printable characters*

ASCII *Code*	*Binnary Code*	*Character*	ASCII *Code*	*Binnary Code*	*Character*
032	00100000	spasi	080	01010000	P
033	00100001	!	081	01010001	Q
034	00100010	“	082	01010010	R
035	00100011	#	083	01010011	S
036	00100100	$	084	01010100	T
037	00100101	%	085	01010101	U
038	00100110	&	086	01010110	V
039	00100111	‘	087	01010111	W
040	00101000	(	088	01011000	X
041	00101001	)	089	01011001	Y
042	00101010	*	090	01011010	Z
043	00101011	+	091	01011011	[
044	00101100	,	092	01011100	\
045	00101101	-	093	01011101	]
046	00101110	.	094	01011110	^
047	00101111	/	095	01011111	_
048	00110000	0	096	01100000	`

ASCII *Code*	***Binnary Code***	***Character***	**ASCII *Code***	***Binnary Code***	***Character***
049	00110001	1	097	01100001	a
050	00110010	2	098	01100010	b
051	00110011	3	099	01100011	c
052	00110100	4	100	01100100	d
053	00110101	5	101	01100101	e
054	00110110	6	102	01100110	f
055	00110111	7	103	01100111	g
056	00111000	8	104	01101000	h
057	00111001	9	105	01101001	i
058	00111010	:	106	01101010	j
059	00111011	;	107	01101011	k
060	00111100	<	108	01101100	l
061	00111101	=	109	01101101	m
062	00111110	>	110	01101110	n
063	00111111	?	111	01101111	o
064	01000000	@	112	01110000	p
065	01000001	A	113	01110001	q
066	01000010	B	114	01110010	r
067	01000011	C	115	01110011	s
068	01000100	D	116	01110100	t
069	01000101	E	117	01110101	u
070	01000110	F	118	01110110	v
071	01000111	G	119	01110111	w
072	01001000	H	120	01111000	x
073	01001001	I	121	01111001	y
074	01001010	J	122	01111010	z
075	01001011	K	123	01111011	{
076	01001100	L	124	01111100	|
077	01001101	M	125	01111101	}
078	01001110	N	126	01111110	~
079	01001111	O	127	01111111	Del

BAB 3. METODE PENELITIAN

3.1 Data Penelitian

Data yang digunakan dalam penelitian ini adalah *printable characters*, yaitu karakter kode 32 sampai 127 yang berisi huruf, digit, tanda baca dan beberapa simbol lain yang sebagian besar ditemukan di *keyboard* dan dapat dicetak serta nilai RGB dari setiap piksel citra digital. *Printable characters* dan nilai RGB tersebut digunakan sebagai *plaintext* dan *ciphertext*. *Printable characters* dalam kode ASCII dan kode biner dapat dilihat pada tabel 2.1.

3.2 Langkah-Langkah Penelitian

Langkah-langkah penelitian yang dilakukan dalam tugas akhir ini adalah :

a. Studi pustaka

Studi pustaka bertujuan untuk memahami secara teoritis mengenai teknik kriptografi algoritma *Hill Cipher* dan steganografi menggunakan metode *Least Significant Bit* pada citra digital.

b. Analisis Hasil

Terdapat enam tahap pada langkah kedua pada penelitian ini, antara lain:

1) Enkripsi *plaintext* menggunakan algoritma *Hill Cipher*

Plaintext (berupa teks) dienkripsi menggunakan algoritma *Hill Cipher* dengan menggunakan matriks *K* sebagai matriks kunci. Proses enkripsi dilakukan seperti pada persamaan 2.3 sehingga dihasilkan *ciphertext*. *Ciphertext* tersebut merupakan *hiddentext* yang kemudian disisipkan ke dalam media citra digital (*cover object*).

2) Penyisipan pesan (*encoding*) menggunakan metode *Least Significant Bit*

   Pada tahapan ini, *hiddentext* disisipkan ke dalam *cover object* menggunakan metode *Least Significant Bit*, sehingga dihasilkan *stego object*. *Hiddentext* yang disisipkan ke dalam *cover object* adalah pesan teks terenkripsi, ukuran matriks kunci dan elemen matriks kunci yang digunakan pada proses enkripsi *plaintext*.

3) Enkripsi *stego object*

   Stego object yang didapatkan dari tahapan kedua akan dienkripsi kembali menggunakan algoritma *Hill Cipher*. Matriks kunci yang digunakan adalah matriks berordo $n \times n$ dengan determinan matriks $= 1$. Proses enkripsi *stego object* dilakukan menggunakan persamaan 2.3 sehingga didapatkan citra kripto.

4) Dekripsi citra kripto

   Proses dekripsi citra kripto dilakukan dengan memasukkan matriks kunci yang sama dengan matriks kunci yang digunakan untuk mengenkripsi *stego object* kemudian mencari invers dari matriks kunci tersebut. Proses dekripsi citra kripto dilakukan menggunakan persamaan 2.4 dengan mengalikan invers matriks kunci yang digunakan untuk mengenkripsi *stego object* dengan matriks RGB citra kripto. Proses dekripsi citra kripto menghasilkan *stego object* semula.

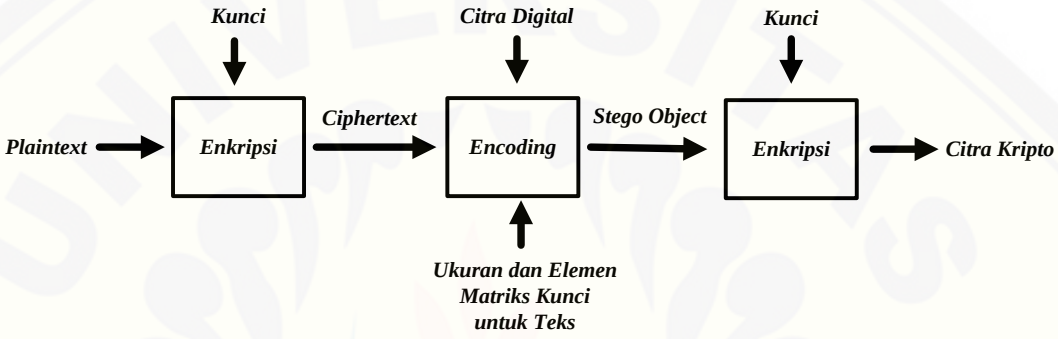
5) Ekstraksi pesan (*decoding*)

   Pesan teks yang telah disisipkan di dalam *cover object* diekstrak dengan menggunakan metode *Least Significant Bit*. *Hiddentext* yang didapatkan dari proses *decoding* berupa *ciphertext* serta ukuran matriks kunci dan elemen matriks kunci yang selanjutnya akan digunakan untuk mendekripsi *ciphertext* ke *plaintext* semula.

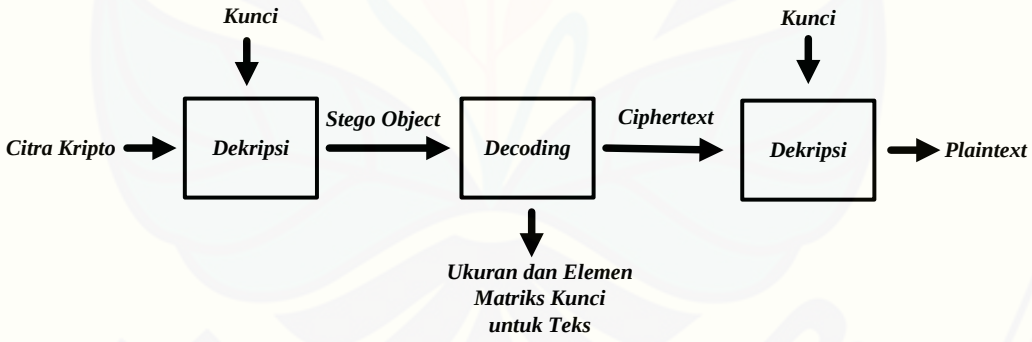
6) Dekripsi *Ciphertext*

Pada proses ini, *ciphertext* akan didekripsi kembali untuk mendapatkan *plaintext* dengan mengalikan invers matriks kunci yang didapatkan dari proses *decoding* dengan matriks *ciphertext* menggunakan persamaan 2.4.

Proses pada langkah b pada penelitian ini dapat dilihat seperti pada diagram berikut:



Gambar 3.1 Proses enkripsi *plaintext*, *encoding pesan* dan enkripsi *stego object*



Gambar 3.2 Proses dekripsi citra kripto, *decoding pesan* dan dekripsi *ciphertext*

c. Pembuatan Program

Pada langkah ini, membuat program dengan bantuan *software* MATLAB 2009a berdasarkan pada algoritma yang telah dibuat pada langkah sebelumnya.

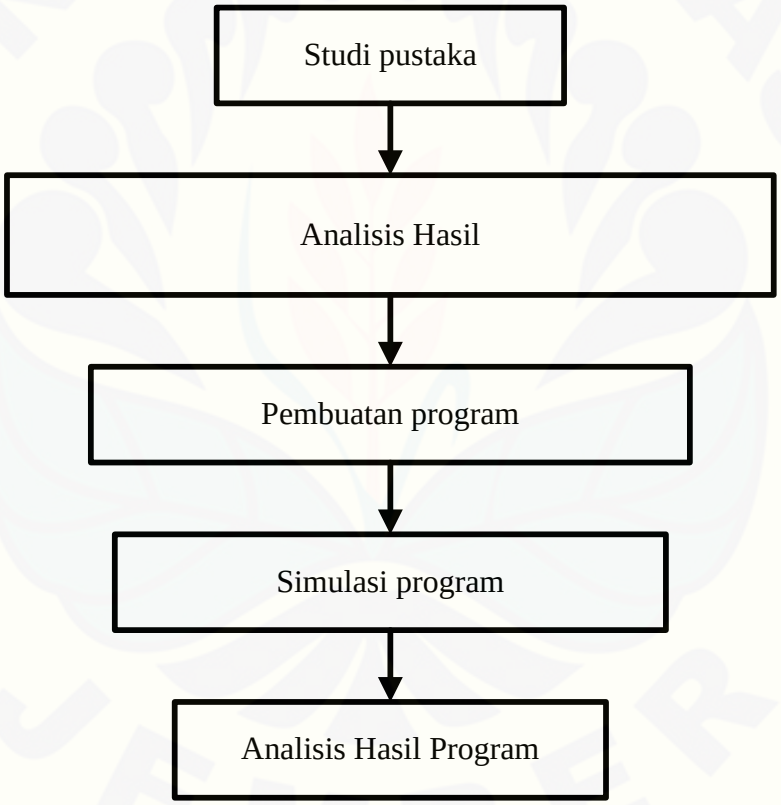
d. Simulasi Program

Langkah selanjutnya adalah melakukan simulasi program dengan *printable character* sebagai data teks terenkripsi dan citra digital sebagai *cover object*.

e. Analisis Hasil Program

Pada langkah ini, dilakukan analisis hasil dengan membandingkan karakter sebelum dan sesudah proses dilakukan.

Secara skematik, alur penelitian tugas akhir ini digambarkan dalam Gambar 3.1



Gambar 3.3 Skema alur penelitian

BAB 4. HASIL DAN PEMBAHASAN

Penelitian ini membahas tentang penerapan dua teknik pengamanan data, yaitu kriptografi pada data teks maupun citra digital menggunakan algoritma *Hill Cipher* dan steganografi menggunakan metode *Least Significant Bit*. Pesan asli atau *plaintext* (berupa teks) terlebih dahulu dienkripsi menggunakan algoritma *Hill Cipher* kemudian pesan tersebut disisipkan ke dalam citra digital menggunakan metode *Least Significant Bit*. Citra digital yang telah disisipkan pesan tersebut dienkripsi menggunakan algoritma *Hill Cipher*.

Implementasi tugas akhir ini adalah merancang suatu program dalam pengamanan data teks menggunakan dua teknik pengamanan data. Steganografi menggunakan metode *Least Significant Bit* dapat dengan mudah dipecahkan karena pesan yang telah dikodekan tersebut disisipkan ke dalam citra digital secara berurutan. Oleh karena itu, citra digital yang telah disisipkan pesan dienkripsi kembali menggunakan algoritma *Hill Cipher* agar kriptanalis tidak mengetahui bahwa pesan rahasia yang dikirimkan sebenarnya berupa data teks, bukan gambar.

4.1 Analisis Hasil

4.1.1 Enkripsi *Plaintext* menggunakan Algoritma *Hill Cipher*

Pesan teks yang akan dienkripsi berupa *printable characters*. Proses enkripsi pesan teks menggunakan modulo 95 karena jumlah seluruh karakter yang dapat digunakan sebanyak 95 karakter (*printable characters*). Dalam proses enkripsi menggunakan algoritma *Hill Cipher, plaintext* terlebih dahulu dikonversikan ke dalam bilangan desimal (kode ASCII), berdasarkan pada Tabel 2.1, kemudian proses enkripsi dilakukan per blok *plaintext*. Berikut merupakan cara kerja algoritma *Hill Cipher* dalam penyandian data teks :

a. Membuat matriks kunci (K)

Misalkan matriks kunci yang digunakan untuk mengenkripsi *plaintext* berordo 3×3 dengan $det(K) \neq 0$ adalah $K = \begin{bmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix}$.

b. Mengkonversikan *plaintext* ke dalam bilangan desimal

Plaintext (P) yang akan dienkripsi adalah "$Besok, jam\ 1$". *Plaintext* tersebut dikonversikan ke dalam bilangan desimal berdasarkan pada Tabel 2.1, sehingga didapatkan *plaintext* dalam bilangan desimal seperti pada Tabel 4.1.

Tabel 4.1 Konversi *plaintext* ke dalam bilangan desimal

B	e	s	o	k	,	spasi	j	a	m	spasi	1
66	101	115	111	107	44	32	106	97	109	32	49

c. Melakukan proses enkripsi *plaintext*

Pada proses enkripsi *plaintext* dengan menggunakan persamaan 2.3, jumlah baris pada matriks *plaintext* harus sama dengan jumlah kolom pada matriks kunci yang digunakan. Jika jumlah baris pada matriks *plaintext* tidak sama dengan jumlah kolom pada matriks kunci, maka *plaintext* ditambah dengan spasi.

Hasil konversi *plaintext* "$Besok, jam\ 1$" membentuk matriks berordo $3 \times n$ didapatkan $P = \begin{bmatrix} 66 & 111 & 32 & 109 \\ 101 & 107 & 106 & 32 \\ 115 & 44 & 97 & 49 \end{bmatrix}$ kemudian *plaintext* tersebut dienkripsi dengan menggunakan matriks kunci K.

$$C = K \cdot P$$

$$C = \left(\begin{bmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 66 & 111 & 32 & 109 \\ 101 & 107 & 106 & 32 \\ 115 & 44 & 97 & 49 \end{bmatrix} \right) mod\ 95$$

$$C = \begin{bmatrix} 66 & 111 & 32 & 109 \\ 629 & 995 & 362 & 904 \\ 779 & 1035 & 533 & 876 \end{bmatrix} mod\ 95$$

$$C = \begin{bmatrix} 66 & 111 & 32 & 109 \\ 59 & 45 & 77 & 49 \\ 114 & 85 & 58 & 116 \end{bmatrix}$$

d. Mengkonversikan *ciphertext* ke karakter yang setara.

Konversikan *ciphertext* tersebut ke karakter yang setara berdasarkan pada Tabel 2.1, sehingga *ciphertext* yang dihasilkan adalah "$B;ro-U\ M:m1t$".

4.1.2 Penyisipan Pesan (*encoding*) menggunakan Metode *Least Significant Bit*

Proses *encoding* pesan teks ke dalam citra digital menggunakan metode *Least Significant Bit*, dengan mengganti bit yang paling tidak berarti pada *cover object* dengan bit data secara berurutan. Citra digital yang digunakan pada penelitian ini adalah citra warna. Setiap komponen piksel pada citra warna terdiri dari komponen RGB yang memiliki panjang 8 bit, sehingga bit citra yang dapat diganti dengan bit data adalah bit kedelapan. Komponen piksel yang digunakan untuk menyisipkan pesan adalah komponen R (*Red*) dengan menyisipkan satu karakter ke dalam satu baris piksel citra warna, sehingga setiap satu karakter yang disisipkan membutuhkan 8 kolom pada setiap baris piksel citra warna. Sedangkan ukuran matriks kunci yang digunakan untuk mengenkripsi pesan teks disisipkan di dalam komponen G (*Green*) dan elemen matriks kunci disisipkan di dalam komponen B (*Blue*). Jumlah karakter yang dapat disisipkan ke dalam citra digital bergantung pada ukuran lebar citra digital (*width*). Semakin besar ukuran lebar citra digital, maka semakin banyak jumlah karakter yang dapat disisipkan.

Berikut merupakan proses *encoding* pesan teks ke dalam citra digital menggunakan metode *Least Significant Bit*:

a. Mengkonversikan *ciphertext* ke dalam bilangan desimal

Ciphertext yang dihasilkan pada langkah sebelumnya adalah "$B;ro-U\ M:m1t$".

Hasil konversi *ciphertext* ke dalam bilangan desimal berdasarkan pada Tabel 2.1 adalah $C = \begin{bmatrix} 66 & 111 & 32 & 109 \\ 59 & 45 & 77 & 49 \\ 114 & 85 & 58 & 116 \end{bmatrix}$

b. Mengkonversikan *ciphertext* ke dalam bilangan biner

Hasil konversi *ciphertext* ke dalam bilangan biner berdasarkan pada Tabel 2.1, seperti pada Tabel 4.2.

Tabel 4.2 Konversi *ciphertext* ke dalam bilangan biner

B	;	r	o	-	U
01000010	00111011	01110010	01101111	00101101	01010101
spasi	M	:	m	1	t
00100000	01001101	00111010	01101101	00110001	01110100

c. Menentukan citra digital yang akan dijadikan *cover object* kemudian mentukan nilai RGB dari setiap piksel citra digital.

Komponen piksel yang digunakan untuk menyisipkan pesan teks adalah komponen R. Misalkan komponen R citra warna (citra mula-mula) yang digunakan seperti pada Tabel 4.3

Tabel 4.3 Komponen R pada citra mula-mula

Baris ke-	**Kolom ke-**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	R=101	R=103	R=110	R=104	R=104	R=104	R=108	R=100
2	R=102	R=108	R=106	R=100	R=97	R=101	R=110	R=109
3	R=106	R=107	R=106	R=103	R=100	R=103	R=103	R=105
4	R=110	R=99	R=98	R=109	R=110	R=107	R=96	R=99
5	R=107	R=98	R=97	R=101	R=104	R=100	R=103	R=99
6	R=111	R=104	R=100	R=100	R=99	R=100	R=110	R=106
7	R=115	R=100	R=99	R=104	R=103	R=100	R=104	R=102
8	R=102	R=99	R=98	R=104	R=101	R=101	R=103	R=100
9	R=101	R=100	R=102	R=106	R=100	R=96	R=101	R=106
10	R=100	R=100	R=99	R=106	R=104	R=101	R=101	R=101
11	R=102	R=101	R=100	R=104	R=100	R=105	R=103	R=101
12	R=105	R=106	R=102	R=101	R=98	R=103	R=104	R=106

sedangkan komponen G dan komponen B citra warna (citra mula-mula) yang digunakan seperti pada Tabel 4.4 dan Tabel 4.5.

Tabel 4.4 Komponen G pada citra mula-mula

Baris ke-	**Kolom ke-**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	G=90	G=92	G=95	G=91	G=91	G=93	G=94	G=86

Tabel 4.5 Komponen B pada citra mula-mula

Baris ke-	Kolom ke-							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	B=58	B=62	B=66	B=58	B=58	B=62	B=64	B=57
2	B=56	B=64	B=64	B=58	B=54	B=58	B=66	B=62
3	B=64	B=66	B=68	B=62	B=62	B=62	B=64	B=62
4	B=66	B=56	B=56	B=68	B=71	B=66	B=56	B=58
5	B=64	B=56	B=56	B=60	B=64	B=60	B=62	B=59
6	B=72	B=66	B=62	B=64	B=60	B=64	B=68	B=62
7	B=74	B=62	B=62	B=68	B=66	B=63	B=63	B=59
8	B=62	B=60	B=60	B=66	B=66	B=64	B=67	B=62
9	B=66	B=62	B=64	B=67	B=64	B=58	B=61	B=65

c. Mengkonversikan nilai RGB ke dalam bilangan biner

Hasil konversi komponen R pada piksel citra mula-mula dalam bilangan biner berdasarkan pada Lampiran A seperti pada Tabel 4.6

Tabel 4.6 Konversi komponen R ke dalam bilangan biner

01100101	01100111	01101110	01101000	01101000	01101000	01101100	01100100
01100110	01101100	01101010	01100100	01100001	01100101	01101110	01101101
01101010	01101011	01101010	01100111	01100100	01100111	01100111	01101001
01101110	01100011	01100010	01101101	01101110	01101011	01100000	01100011
01101011	01100010	01100001	01100101	01101000	01100100	01100111	01100011
01101111	01101000	01100100	01100100	01100011	01100100	01101110	01101010
01110011	01100100	01100011	01101000	01100111	01100100	01101000	01100110
01100110	01100011	01100010	01101000	01100101	01100101	01100111	01100100
01100101	01100100	01100110	01101010	01100100	01100000	01100101	01101010
01100100	01100100	01100011	01101010	01101000	01100101	01100101	01100101
01100110	01100101	01100100	01101000	01100100	01101001	01100111	01100101
01101001	01100101	01100110	01100101	01100010	01100111	01101000	01101010

sedangkan hasil konversi komponen G dan komponen B pada piksel citra mula-mula dalam bilangan biner berdasarkan pada Lampiran A seperti pada Tabel 4.7 dan Tabel 4.8.

Tabel 4.7 Konversi komponen G ke dalam bilangan biner

01011010	01011100	01011111	01011011	01011011	01011101	01011110	01010110

Tabel 4.8 Konversi komponen B ke dalam bilangan biner

00111010	00111110	01000010	00111010	00111010	00111110	01000000	00111001
00111000	01000000	01000000	00111010	00110110	00111010	01000010	00111110
01000000	01000010	01000100	00111110	00111110	00111110	01000000	00111110
01000010	00111000	00111000	01000100	01000111	01000010	00111000	00111010
01000000	00111000	00111000	00111100	01000000	00111100	00111110	00111011
01001000	01000010	00111110	01000000	00111100	01000000	01000100	00111110
01001010	00111110	00111110	01000100	01000010	00111111	00111111	00111011
00111110	00111100	00111100	01000010	01000010	01000000	01000011	00111110
01000010	00111110	01000000	01000011	01000000	00111010	01000001	01000001

d. Melakukan proses penyisipan pesan (*encoding*)

Pada proses ini, *ciphertext* disisipkan ke dalam citra digital dengan mengganti bit kedelapan pada komponen R citra digital dengan bit *ciphertext* secara berurutan, sehingga komponen R citra digital setelah disisipkan *ciphertext* seperti pada Tabel 4.9.

Tabel 4.9 Komponen R dalam bilangan biner setelah disisipkan *ciphertext*

0110010**0**	01100111	01101110	01101000	01101000	01101000	0110110**1**	01100100
01100110	01101100	0110101**1**	0110010**1**	01100001	0110010**0**	0110111**1**	01101101
01101010	01101011	0110101**1**	01100111	01100100	0110011**0**	01100111	0110100**0**
01101110	01100011	0110001**1**	0110110**0**	0110111**1**	01101011	0110000**1**	01100011
0110101**0**	01100010	01100001	0110010**0**	0110100**1**	0110010**1**	0110011**0**	01100011
0110111**0**	0110100**1**	01100100	0110010**1**	0110001**0**	0110010**1**	01101110	0110101**1**
0111001**0**	01100100	01100011	01101000	0110011**0**	01100100	01101000	01100110
01100110	01100011	01100010	01101000	01100101	01100101	0110011**0**	0110010**1**
0110010**0**	01100100	0110011**1**	0110101**1**	0110010**1**	01100000	01100101	01101010
01100100	0110010**1**	01100011	01101010	0110100**1**	01100101	0110010**0**	01100101
01100110	0110010**0**	0110010**1**	0110100**1**	01100100	0110100**0**	0110011**0**	01100101
0110100**0**	01100101	0110011**1**	01100101	01100010	01100111	01101000	01101010

Bit kedelapan yang dicetak tebal dan digaris bawahi merupakan bit yang mengalami perubahan (menjadi satu bit lebih tinggi atau satu bit lebih rendah) setelah disisipkan pesan. Lakukan cara yang sama dengan proses penyisipan *ciphertext* ke dalam komponen R, untuk menyisipkan ukuran matriks kunci dan elemen matriks kunci ke dalam komponen G dan komponen B sehingga diperoleh komponen G dan komponen B citra digital setelah disisipkan ukuran matriks kunci dan elemen matriks kunci seperti pada Tabel 4.10 dan Tabel 4.11.

Tabel 4.10 Komponen G dalam bilangan biner setelah disisipkan ukuran matriks kunci

01011010	01011100	0101111**0**	0101101**0**	0101101**0**	0101110**0**	0101111**1**	0101011**1**

Tabel 4.11 Komponen B dalam bilangan biner setelah disisipkan elemen matriks kunci

00111010	00111110	01000010	00111010	00111010	00111110	01000000	00111001
00111000	01000000	01000000	00111010	00110110	00111010	01000010	00111110
01000000	01000010	01000100	00111110	00111110	00111110	01000000	00111110
01000010	00111000	00111000	01000100	01000111	01000010	00111000	00111010
01000000	00111000	00111000	00111100	01000000	00111100	00111110	00111011
01001000	01000010	00111110	01000000	00111100	01000000	01000100	00111110
01001010	00111110	00111110	01000100	01000010	00111111	00111111	00111011
00111110	00111100	00111100	01000010	01000010	01000000	01000011	00111110
01000010	00111110	01000000	0100001**0**	01000000	00111010	0100000**0**	01000001

e. Mengkonversikan nilai RGB ke dalam bilangan desimal

Komponen RGB citra digital setelah disisipkan pesan merupakan komponen RGB pada *stego object*. Hasil konversi komponen R citra digital pada Tabel 4.9 ke dalam bilangan desimal berdasarkan pada Lampiran A seperti pada Tabel 4.12

Tabel 4.12 Komponen R pada *stego object*

Baris ke-	**Kolom ke-**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	R=100	R=103	R=110	R=104	R=104	R=104	R=109	R=100
2	R=102	R=108	R=107	R=101	R=97	R=100	R=111	R=109
3	R=106	R=107	R=107	R=103	R=100	R=102	R=103	R=104
4	R=110	R=99	R=99	R=108	R=111	R=107	R=97	R=99
5	R=106	R=98	R=97	R=100	R=105	R=101	R=102	R=99
6	R=110	R=105	R=100	R=101	R=98	R=101	R=110	R=107
7	R=114	R=100	R=99	R=104	R=102	R=100	R=104	R=102
8	R=102	R=99	R=98	R=104	R=101	R=101	R=102	R=101
9	R=100	R=100	R=103	R=107	R=101	R=96	R=101	R=106
10	R=100	R=101	R=100	R=106	R=105	R=101	R=100	R=101
11	R=102	R=100	R=101	R=105	R=100	R=104	R=102	R=101
12	R=104	R=106	R=103	R=101	R=98	R=103	R=104	R=106

sedangkan hasil konversi komponen G dan komponen B citra digital pada Tabel 4.10 dan Tabel 4.11 ke dalam bilangan desimal berdasarkan pada Lampiran A seperti pada Tabel 4.13 dan Tabel 4.14.

Tabel 4.13 Komponen G pada *stego object*

Baris ke-	**Kolom ke-**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	G=90	G=92	G=94	G=90	G=90	G=92	G=95	G=87

Tabel 4.14 Komponen B pada *stego object*

Baris ke-	**Kolom ke-**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	B=58	B=62	B=66	B=58	B=58	B=62	B=64	B=57
2	B=56	B=64	B=64	B=58	B=54	B=58	B=66	B=62
3	B=64	B=66	B=68	B=62	B=62	B=62	B=64	B=62
4	B=66	B=56	B=56	B=68	B=71	B=66	B=56	B=58
5	B=64	B=56	B=56	B=60	B=64	B=60	B=62	B=59
6	B=72	B=66	B=62	B=64	B=60	B=64	B=68	B=62
7	B=74	B=62	B=62	B=68	B=66	B=63	B=63	B=59
8	B=62	B=60	B=60	B=66	B=66	B=64	B=67	B=62
9	B=66	B=62	B=64	B=66	B=64	B=58	B=60	B=65

4.1.3 Enkripsi *Stego Object*

Rentang nilai untuk setiap komponen RGB adalah 0 sampai 255 sehingga modulo yang digunakan dalam penyandian citra adah modulo 256. Nilai-nilai pada komponen RGB citra digital disusun sedemikian rupa agar dapat dikalikan dengan matriks kunci yang digunakan dengan syarat jumlah kolom pada matriks kunci sama dengan jumlah baris pada matriks RGB. Pada proses enkripsi *stego object*, misalkan matriks kunci yang digunakan adalah matriks berordo 2×2 dan berordo 3×3 dengan determinan matriks kunci $= 1$.

Jika matriks kunci yang digunakan berordo 2×2 maka penyusunan matriks P yang berisi nilai komponen RGB adalah sebagai berikut.

$$P = \begin{bmatrix} R_1 & G_1 & B_1 \\ R_2 & G_2 & B_2 \end{bmatrix}$$

dimana :

R_1 adalah nilai komponen *red* pada piksel kesatu;

G_1 adalah nilai komponen *green* pada piksel kesatu;

B_1 adalah nilai komponen *blue* pada piksel kesatu;

R_2 adalah nilai komponen *red* pada piksel kedua;

G_2 adalah nilai komponen *green* pada piksel kedua;

B_2 adalah nilai komponen *blue* pada piksel kedua.

Jika matriks kunci yang digunakan berordo 3×3 maka penyusunan matriks P yang berisi nilai komponen RGB adalah sebagai berikut.

$$P = \begin{bmatrix} R_1 & G_1 & B_1 \\ R_2 & G_2 & B_2 \\ R_3 & G_3 & B_3 \end{bmatrix}$$

dimana :

R_3 adalah nilai komponen *red* pada piksel ketiga;

G_3 adalah nilai komponen *green* pada piksel ketiga;

B_3 adalah nilai komponen *blue* pada piksel ketiga.

Berikut ini merupakan langkah-langkah dalam mengenkripsi citra digital menggunakan algoritma *Hill Cipher*.

a. Membuat matriks kunci (K)

Misalkan matriks kunci yang digunakan dalam proses enkripsi *stego object* adalah

$K_1 = \begin{bmatrix} 10 & 3 \\ 33 & 10 \end{bmatrix}$ dan $K_2 = \begin{bmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix}$

b. Menentukan nilai RGB dari setiap piksel *stego object*

Dari proses *encoding*, misalkan diambil potongan *stego object* sebanyak 6 buah piksel pada kolom pertama seperti pada Tabel 4.15.

Tabel 4.15 Komponen RGB *stego object*

Piksel ke-	***Red***	***Green***	***Blue***
1	100	90	58
2	102	90	56
3	106	95	64
4	110	95	66
5	106	91	64
6	110	95	72

c. Melakukan proses enkripsi *stego object*

1) Proses enkripsi menggunakan matriks kunci K_1

Komponen piksel *stego object* dibagi menjadi blok-blok dengan jumlah baris $= 2$, sehingga komponen piksel dalam bentuk matriks yang didapatkan adalah

$P_1 = \begin{bmatrix} 100 & 90 & 58 \\ 102 & 90 & 56 \end{bmatrix}$, $P_2 = \begin{bmatrix} 106 & 95 & 64 \\ 110 & 95 & 66 \end{bmatrix}$ dan $P_3 = \begin{bmatrix} 106 & 91 & 64 \\ 110 & 95 & 72 \end{bmatrix}$ maka dengan menggunakan persamaan 2.3, proses yang digunakan untuk menghasilkan piksel output enkripsi adalah sebagai berikut.

Untuk blok pertama

$$C_1 = K_1 \cdot P_1$$

$$C_1 = \left(\begin{bmatrix} 10 & 3 \\ 33 & 10 \end{bmatrix} \cdot \begin{bmatrix} 100 & 90 & 58 \\ 102 & 90 & 56 \end{bmatrix}\right) mod\ 256$$

$$C_1 = \left(\begin{bmatrix} 1306 & 1170 & 748 \\ 4320 & 3870 & 2474 \end{bmatrix}\right) mod\ 256$$

$$C_1 = \begin{bmatrix} 26 & 146 & 236 \\ 224 & 30 & 170 \end{bmatrix}$$

Untuk blok kedua

$$C_2 = K_1 \cdot P_2$$

$$C_2 = \left(\begin{bmatrix} 10 & 3 \\ 33 & 10 \end{bmatrix} \cdot \begin{bmatrix} 106 & 95 & 64 \\ 110 & 95 & 66 \end{bmatrix}\right) mod\ 256$$

$$C_2 = \left(\begin{bmatrix} 1390 & 1235 & 838 \\ 4598 & 4085 & 2772 \end{bmatrix}\right) mod\ 256$$

$$C_2 = \begin{bmatrix} 110 & 211 & 70 \\ 246 & 245 & 212 \end{bmatrix}$$

Untuk blok ketiga

$$C_3 = K_1 \cdot P_3$$

$$C_3 = \left(\begin{bmatrix} 10 & 3 \\ 33 & 10 \end{bmatrix} \cdot \begin{bmatrix} 106 & 91 & 64 \\ 110 & 95 & 72 \end{bmatrix}\right) mod\ 256$$

$$C_3 = \left(\begin{bmatrix} 1390 & 1195 & 856 \\ 4598 & 3953 & 2832 \end{bmatrix}\right) mod\ 256$$

$$C_3 = \begin{bmatrix} 110 & 171 & 88 \\ 246 & 113 & 16 \end{bmatrix}$$

Komponen RGB citra kripto dengan menggunakan matriks kunci K_1 pada proses enkripsi *stego object* dapat dilihat pada Tabel 4.16.

Tabel 4.16 Komponen RGB citra kripto menggunakan matriks kunci K_1

Piksel ke-	*Red*	*Green*	*Blue*
1	26	146	236
2	224	30	170
3	110	211	70
4	246	245	212
5	110	171	88
6	246	113	16

2) Proses enkripsi menggunakan matriks kunci K_2

Komponen piksel *stego object* dibagi menjadi blok-blok dengan jumlah baris $= 3$, sehingga komponen piksel dalam bentuk matriks yang didapatkan adalah $P_1 = \begin{bmatrix} 100 & 90 & 58 \\ 102 & 90 & 56 \\ 106 & 95 & 64 \end{bmatrix}$ dan $P_2 = \begin{bmatrix} 110 & 95 & 66 \\ 106 & 91 & 64 \\ 110 & 95 & 72 \end{bmatrix}$ maka dengan menggunakan persamaan 2.3, proses yang digunakan untuk menghasilkan piksel output enkripsi adalah sebagai berikut.

Untuk blok pertama

$C_1 = K_2 \cdot P_1$

$$C_1 = \left(\begin{bmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 100 & 90 & 58 \\ 102 & 90 & 56 \\ 106 & 95 & 64 \end{bmatrix} \right) mod\ 256$$

$$C_1 = \left(\begin{bmatrix} 100 & 90 & 58 \\ 902 & 810 & 520 \\ 1010 & 905 & 582 \end{bmatrix} \right) mod\ 256$$

$$C_1 = \begin{bmatrix} 100 & 90 & 58 \\ 134 & 42 & 8 \\ 242 & 137 & 70 \end{bmatrix}$$

Untuk blok kedua

$C_2 = K_2 \cdot P_2$

$$C_2 = \left(\begin{bmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 110 & 95 & 66 \\ 106 & 91 & 64 \\ 110 & 95 & 72 \end{bmatrix} \right) mod\ 256$$

$$C_2 = \left(\begin{bmatrix} 110 & 95 & 66 \\ 986 & 851 & 592 \\ 1092 & 942 & 662 \end{bmatrix} \right) mod\ 256$$

$$C_2 = \begin{bmatrix} 100 & 90 & 58 \\ 218 & 83 & 80 \\ 68 & 174 & 150 \end{bmatrix}$$

Komponen RGB citra kripto dengan menggunakan matriks kunci K_2 pada proses enkripsi *stego object* dapat dilihat pada Tabel 4.17.

Tabel 4.17 Komponen RGB citra kripto menggunakan matriks kunci K_2

Piksel ke-	*Red*	*Green*	*Blue*
1	100	90	58
2	134	42	8
3	242	137	70
4	100	90	58
5	218	83	80
6	68	174	150

4.1.4 Dekripsi Citra Kripto

Pada proses dekripsi citra kripto, terlebih dahulu memasukkan matriks kunci yang sama dengan matriks kunci yang digunakan untuk mengenkripsi *stego object* kemudian mencari invers dari matriks kunci tersebut. Proses dekripsi citra kripto menggunakan persamaan 2.4 dengan mengalikan invers matriks kunci yang digunakan untuk mengenkripsi *stego object* dengan matriks RGB citra kripto. Langkah-langkah dalam mendekripsi citra kripto menggunakan algoritma *Hill Cipher* adalah sebagai berikut :

a. Menentukan nilai RGB dari setiap piksel citra kripto

   Potongan komponen RGB citra kripto seperti pada tabel 4.16 dan 4.17

b. Menentukan invers matriks kunci (K^{-1}) dari proses enkripsi *stego object*

   Invers matriks kunci yang digunakan dalam proses dekripsi citra kripto yaitu

$$K_1{}^{-1} = \begin{bmatrix} 10 & -3 \\ -33 & 10 \end{bmatrix}$$

$$K_1{}^{-1} = \left(\begin{bmatrix} 10 & -3 \\ -33 & 10 \end{bmatrix}\right) mod\ 256$$

$$K_1{}^{-1} = \begin{bmatrix} 10 & 253 \\ 223 & 10 \end{bmatrix}$$

dan $K_2{}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -8 & 1 & 0 \\ 9 & -2 & 1 \end{bmatrix}$ maka $K_2{}^{-1} = \left(\begin{bmatrix} 1 & 0 & 0 \\ -8 & 1 & 0 \\ 9 & -2 & 1 \end{bmatrix}\right) mod\ 95$

$K_2{}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 87 & 1 & 0 \\ 9 & 93 & 1 \end{bmatrix}$

c. Melakukan proses dekripsi citra kripto

1) Proses dekripsi menggunakan matriks kunci $K_1{}^{-1}$

Komponen piksel citra kripto dibagi menjadi blok-blok dengan jumlah baris $= 2$, sehingga komponen piksel dalam bentuk matriks yang didapatkan adalah $C_1 = \begin{bmatrix} 26 & 146 & 236 \\ 224 & 30 & 170 \end{bmatrix}$, $C_2 = \begin{bmatrix} 110 & 211 & 70 \\ 246 & 245 & 212 \end{bmatrix}$ dan $C_3 = \begin{bmatrix} 110 & 171 & 88 \\ 246 & 113 & 16 \end{bmatrix}$ maka dengan menggunakan persamaan 2.4, proses yang digunakan untuk menghasilkan piksel output dekripsi adalah sebagai berikut.

Untuk blok pertama

$P_1 = K_1{}^{-1} \cdot C_1$

$P_1 = \left(\begin{bmatrix} 10 & 253 \\ 223 & 10 \end{bmatrix} \cdot \begin{bmatrix} 26 & 146 & 236 \\ 224 & 30 & 170 \end{bmatrix}\right) mod\ 256$

$P_1 = \left(\begin{bmatrix} 56932 & 9050 & 45370 \\ 8038 & 32858 & 54328 \end{bmatrix}\right) mod\ 256$

$P_1 = \begin{bmatrix} 100 & 90 & 58 \\ 102 & 90 & 56 \end{bmatrix}$

Untuk blok kedua

$P_2 = K_1{}^{-1} \cdot C_2$

$P_2 = \left(\begin{bmatrix} 10 & 253 \\ 223 & 10 \end{bmatrix} \cdot \begin{bmatrix} 110 & 211 & 70 \\ 246 & 245 & 212 \end{bmatrix}\right) mod\ 256$

$P_2 = \left(\begin{bmatrix} 63338 & 64095 & 54336 \\ 26990 & 49503 & 17730 \end{bmatrix}\right) mod\ 256$

$P_2 = \begin{bmatrix} 106 & 95 & 64 \\ 110 & 95 & 66 \end{bmatrix}$

Untuk blok ketiga

$P_3 = K_1{}^{-1} \cdot C_3$

$P_3 = \left(\begin{bmatrix} 10 & 253 \\ 223 & 10 \end{bmatrix} \cdot \begin{bmatrix} 110 & 171 & 88 \\ 246 & 113 & 16 \end{bmatrix}\right) mod\ 256$

$$P_3 = \left(\begin{bmatrix} 63338 & 30299 & 4928 \\ 26990 & 39263 & 19784 \end{bmatrix}\right) mod\ 256$$

$$P_3 = \begin{bmatrix} 106 & 91 & 64 \\ 110 & 95 & 72 \end{bmatrix}$$

2) Proses dekripsi menggunakan matriks kunci ${K_2}^{-1}$

Komponen piksel citra kripto dibagi menjadi blok-blok dengan jumlah baris $= 3$, sehingga komponen piksel dalam bentuk matriks yang didapatkan adalah $C_1 = \begin{bmatrix} 100 & 90 & 58 \\ 134 & 42 & 8 \\ 242 & 137 & 70 \end{bmatrix}$ dan $C_2 = \begin{bmatrix} 100 & 90 & 58 \\ 218 & 83 & 80 \\ 68 & 174 & 150 \end{bmatrix}$ maka dengan menggunakan persamaan 2.4, proses yang digunakan untuk menghasilkan piksel output dekripsi adalah sebagai berikut.

Untuk blok pertama

$$P_1 = {K_2}^{-1} \cdot C_1$$

$$P_1 = \left(\begin{bmatrix} 1 & 0 & 0 \\ 87 & 1 & 0 \\ 9 & 93 & 1 \end{bmatrix} \cdot \begin{bmatrix} 100 & 90 & 58 \\ 134 & 42 & 8 \\ 242 & 137 & 70 \end{bmatrix}\right) mod\ 256$$

$$P_1 = \left(\begin{bmatrix} 100 & 90 & 58 \\ 8834 & 7872 & 5054 \\ 13604 & 4853 & 1336 \end{bmatrix}\right) mod\ 256$$

$$P_1 = \begin{bmatrix} 100 & 90 & 58 \\ 102 & 90 & 56 \\ 106 & 95 & 64 \end{bmatrix}$$

Untuk blok kedua

$$P_2 = {K_2}^{-1} \cdot C_2$$

$$P_2 = \left(\begin{bmatrix} 1 & 0 & 0 \\ 87 & 1 & 0 \\ 9 & 93 & 1 \end{bmatrix} \cdot \begin{bmatrix} 100 & 90 & 58 \\ 218 & 83 & 80 \\ 68 & 174 & 150 \end{bmatrix}\right) mod\ 256$$

$$P_2 = \left(\begin{bmatrix} 100 & 90 & 58 \\ 8918 & 7913 & 5126 \\ 21242 & 8703 & 8112 \end{bmatrix}\right) mod\ 256$$

$$P_2 = \begin{bmatrix} 110 & 95 & 66 \\ 106 & 91 & 64 \\ 110 & 95 & 72 \end{bmatrix}$$

Komponen RGB yang dihasilkan dari proses dekripsi citra kripto dengan menggunakan matriks kunci $K_1{}^{-1}$ dan $K_2{}^{-1}$ seperti pada Tabel 4.15.

4.1.5 Ekstraksi Pesan (*decoding*)

Ekstraksi pesan dilakukan dengan mengambil bit kedelapan pada setiap komponen R yang mengandung pesan. Adapun langkah-langkah yang dilakukan dalam proses *decoding* adalah sebagai berikut :

a. Menentukan *stego object* yang akan diekstrak kemudian mentukan nilai RGB dari setiap piksel *stego object*

   Komponen piksel *stego object* yang telah disisipkan pesan adalah komponen R, seperti pada Tabel 4.12

b. Mengkonversikan nilai RGB ke dalam bilangan biner

   Konversikan nilai R tersebut ke dalam bilangan biner berdasarkan pada Lampiran A, sehingga didapatkan nilai komponen R dalam bilangan biner seperti pada Tabel 4.9.

c. Melakukan proses ekstraksi pesan (*decoding*)

   Untuk mendapatkan *ciphertext* yang telah disisipkan di dalam citra tersebut, dengan mengambil bit kedelapan pada setiap bit komponen R yang telah disisipkan pesan, sehingga didapatkan *ciphertext* dalam bentuk biner sebagai berikut.

01000010	00111011	01110010	01101111	00101101	01010101
00100000	01001101	00111010	01101101	00110001	01110100

   Lakukan cara yang sama untuk mendapatkan ukuran matriks kunci dan elemen matriks kunci dalam bentuk biner yang telah disisipkan di dalam komponen G dan komponen B sehingga ukuran matriks kunci yang didapatkan adalah 00000011 dan elemen matriks kunci yang didapatkan adalah

00000001	00000000	00000000
00001000	00000001	00000000
00000111	00000010	00000001

d. Mengkonversikan *ciphertext* dalam bilangan biner ke karakter yang setara

Konversikan *ciphertext* dalam bilangan biner tersebut ke dalam bilangan desimal berdasarkan pada Tabel 2.1, sehingga *ciphertext* dalam bilangan desimal dalam bentuk matriks yang didapatkan adalah $C = \begin{bmatrix} 66 & 111 & 32 & 109 \\ 59 & 45 & 77 & 49 \\ 114 & 85 & 58 & 116 \end{bmatrix}$ kemudian konversikan *ciphertext* tersebut ke karakter yang setara berdasarkan pada Tabel 2.1, sehingga *ciphertext* yang didapatkan dari proses ekstraksi pesan adalah "$B;ro-U\ M{:}m1t$"

e. Konversikan nilai RGB dalam bilangan biner ke dalam bilangan desimal

Hasil konversi komponen R, komponen G dan komponen B citra digital dalam bilangan biner ke dalam bilangan desimal berdasarkan pada Lampiran A adalah seperti pada Tabel 4.12, Tabel 4.13 dan Tabel 4.14 dan potongan komponen RGB pada piksel citra digital dalam bilangan desimal seperti pada Tabel 4.15.

4.1.6 Dekripsi *Ciphertext*

Proses dekripsi *ciphertext* menjadi *plaintext* sama dengan proses enkripsinya, namun matriks kunci yang digunakan adalah invers dari matriks kunci pada proses enkripsi *plaintext*. Adapun langkah-langkah dalam mendekripsi *ciphertext* adalah sebagai berikut.

a. Menentukan invers matriks kunci (K^{-1})

Invers matriks kunci yang digunakan dalam proses dekripsi *plaintext* yaitu

$$K^{-1} = \left(\begin{bmatrix} 1 & 0 & 0 \\ -8 & 1 & 0 \\ 9 & -2 & 1 \end{bmatrix} \right) mod(95)$$

$$K^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 87 & 1 & 0 \\ 9 & 93 & 1 \end{bmatrix}$$

b. Mengkonversikan *ciphertext* ke dalam bilangan desimal

Ciphertext yang akan didekripsi adalah "$B;ro-U\ M{:}m1t$". Jika dikonversikan ke dalam bilangan desimal dalam matriks berordo $3 \times n$ maka didapatkan

$$C = \begin{bmatrix} 66 & 111 & 32 & 109 \\ 59 & 45 & 77 & 49 \\ 114 & 85 & 58 & 116 \end{bmatrix}$$

c. Melakukan proses dekripsi *ciphertext*

Proses dekripsi *ciphertext* dilakukan dengan menggunakan persamaan 2.4 seperti pada proses di bawah ini

$$P = K^{-1} \cdot C$$

$$P = \left(\begin{bmatrix} 1 & 0 & 0 \\ 87 & 1 & 0 \\ 9 & 93 & 1 \end{bmatrix} \cdot \begin{bmatrix} 66 & 111 & 32 & 109 \\ 59 & 45 & 77 & 49 \\ 114 & 85 & 58 & 116 \end{bmatrix} \right) mod\ 95$$

$$= \left(\begin{bmatrix} 66 & 111 & 32 & 109 \\ 5801 & 9702 & 2861 & 9532 \\ 6195 & 5269 & 7507 & 5654 \end{bmatrix} \right) mod\ 95$$

$$= \begin{bmatrix} 66 & 111 & 32 & 109 \\ 101 & 107 & 106 & 32 \\ 115 & 44 & 97 & 49 \end{bmatrix}$$

sehingga, *plaintext* yang dihasilkan adalah "$Besok, jam\ 1$".

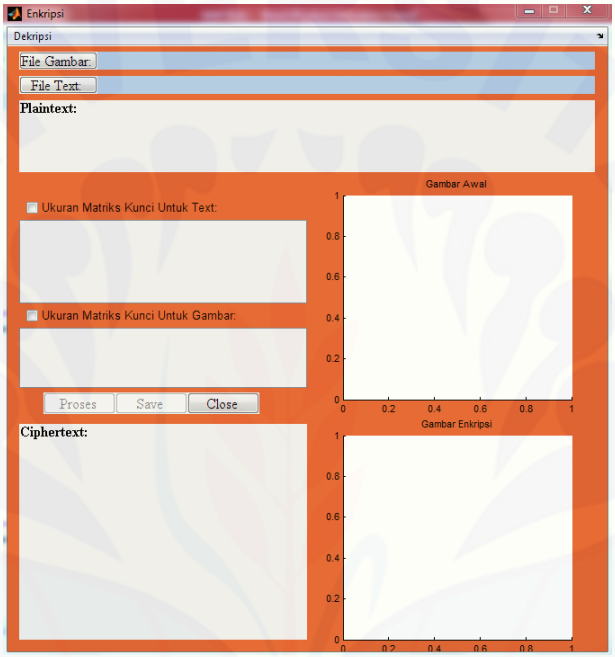
4.2 Program Aplikasi

Program aplikasi pada penelitian ini dibuat dengan bantuan *software* MATLAB 2009a yang bertujuan untuk mempermudah dalam pengamanan data teks menggunakan dua teknik pengamanan data, yaitu kriptografi dan steganografi dengan melakukan pengkodean citra digital hasil steganografi dengan metode *Least Significant Bit* untuk data teks terenkripsi dengan algoritma *Hill Cipher*.

Program aplikasi ini terdiri dari 2 *form*, yaitu *form* Enkripsi dan *form* Dekripsi yang digunakan untuk melakukan proses enkripsi dan dekripsi. Berikut ini merupakan penjelasan dari kedua *form* tersebut.

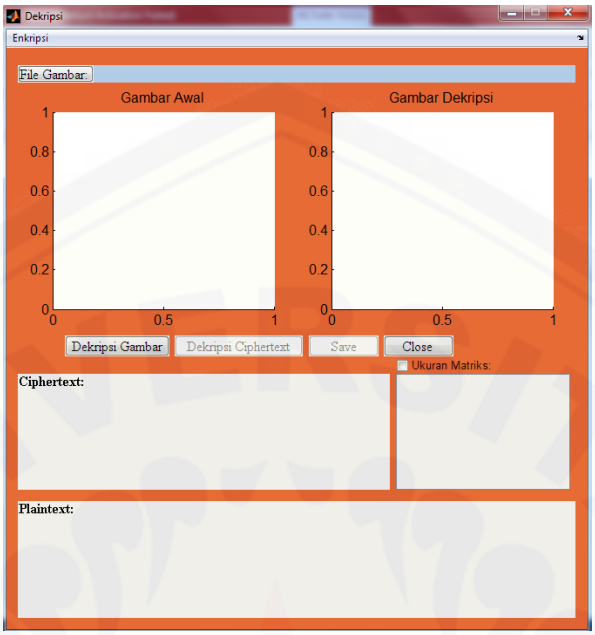
a. *Form* Enkripsi digunakan dalam proses enkripsi *plaintext*, menyembunyikan *ciphertext* ke dalam citra digital dan mengenkripsi citra digital yang telah disisipkan pesan. Di dalam *form* ini, *user* dapat memasukkan citra digital maupun teks yang akan dienkripsi. Pada *form* ini terdapat kolom yang berfungsi untuk menampilkan *plaintext* dan *ciphertext*. Selain itu, juga terdapat kolom yang digunakan untuk

memasukkan ukuran dan elemen matriks kunci yang digunakan dalam proses enkripsi citra digital maupun teks. *Pushbutton* "Proses" digunakan untuk memproses data hingga didapatkan *output* berupa citra kripto dan *pushbutton* "*Save*" digunakan untuk menyimpan citra kripto. Pada *form* ini terdapat menu "Dekripsi" yang digunakan untuk menuju ke *form* Dekripsi. Tampilan *form* Enkripsi dapat dilihat pada Gambar 4.1.



Gambar 4.1 Tampilan *form* Enkripsi

b. Di dalam *form* Dekripsi, *user* dapat memasukkan citra kripto yang akan didekripsi kemudian memasukkan ukuran dan elemen matriks kunci yang sama dengan matriks kunci yang digunakan pada saat mengenkripsi citra digital. *Pushbutton* "Dekripsi Gambar" digunakan untuk mendekripsi citra kripto hingga didapatkan *output* berupa *stego object* dan *ciphertext*. Selain itu, terdapat *pushbutton* "Dekripsi *Ciphertext*" digunakan untuk mendekripsi *ciphertext* sehingga dihasilkan *plaintext* awal serta *pushbutton* "*save*" yang digunakan untuk menyimpan *stego object*. Pada *form* ini terdapat menu "Enkripsi" yang digunakan untuk menuju ke *form* Enkripsi. Tampilan *form* Dekripsi dapat dilihat pada Gambar 4.2.



Gambar 4.2 Tampilan *form* Dekripsi

Secara umum, langkah-langkah dalam menjalankan program adalah sebagai berikut:

a. Masukkan citra digital dengan memilih *pushbutton* "*File* Gambar", maka citra digital yang akan dijadikan sebagai *cover object* akan ditampilkan pada *axes* Gambar Awal.
b. Masukkan *plaintext* dengan memilih *pushbutton* "*File Text*", maka *plaintext* yang akan dienkripsi akan ditampilkan pada kolom *plaintext*.
c. Pilih ukuran matriks kunci yang digunakan untuk mengenkripsi *plaintext*, kemudian masukkan elemen matriks kunci tersebut.
d. Pilih ukuran matriks kunci yang digunakan untuk mengenkripsi citra digital, kemudian masukkan elemen matriks kunci tersebut.
e. Pilih *pushbutton* "Proses" untuk melakukan proses enkripsi *plaintext, encoding* pesan dan enkripsi *stego object*, sehingga hasil enkripsi *plaintext* akan ditampilkan pada kolom *ciphertext* dan citra kripto akan ditampilkan pada *axes* Gambar Enkripsi.

f. Pilih *pushbutton* "*Save*" untuk menyimpan citra kripto hasil dari enkripsi *stego object*.
g. Masukkan citra kripto yang akan didekripsi dengan memilih *pushbutton* "*File* Gambar", maka citra kripto akan ditampilkan pada *axes* Gambar Awal.
h. Pilih ukuran matriks kunci yang digunakan untuk mendekripsi citra kripto, kemudian masukkan elemen matriks kunci tersebut.
i. Pilih *pushbutton* "Dekripsi Gambar" untuk melakukan proses dekripsi citra kripto dan melakukan proses ekstraksi pesan, sehingga *stego object* yang dihasilkan dari proses dekripsi citra kripto akan ditampilkan pada *axes* Gambar Dekripsi dan *ciphertext* yang dihasilkan dari proses ekstraksi pesan akan ditampilkan pada kolom *ciphertext*.
j. Pilih *pushbutton* "Dekripsi *Ciphertext*" untuk melakukan proses dekripsi *ciphertext*, maka hasil dari proses dekripsi *ciphertext* akan ditampilkan pada kolom *plaintext*.
k. Pilih *pushbutton* "*Save*" untuk menyimpan *stego object*, kemudian pilih *pushbutton* "*Close*" untuk keluar dari program.

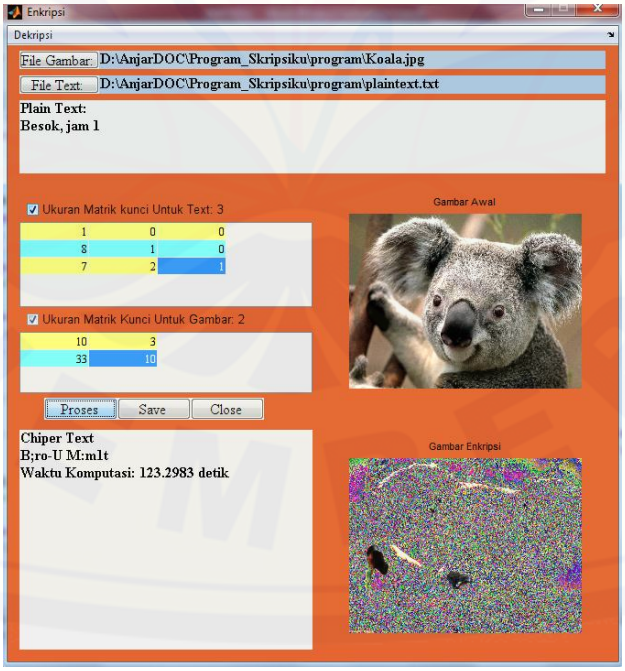
4.3 Simulasi Program

Pada sub-subbab 4.1.1 tentang enkripsi *plaintext*, pesan yang akan dienkripsi adalah "$Besok, jam\ 1$" dan setelah dikodekan, *ciphertext* yang dihasilkan adalah "$B;rP-U\ M{:}m1t$". *Ciphertext* tersebut disisipkan ke dalam citra digital hingga diekstrak kembali sehingga didapatkan *plaintext* semula. Untuk menguji pengkodean dan penyembunyian pesan tersebut dapat berjalan dengan baik dengan menggunakan sebuah matriks kunci pada proses enkripsi pesan teks, yaitu matriks $K = \begin{bmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix}$ dan 2 buah matriks kunci yang digunakan pada proses enkripsi citra digital, yaitu

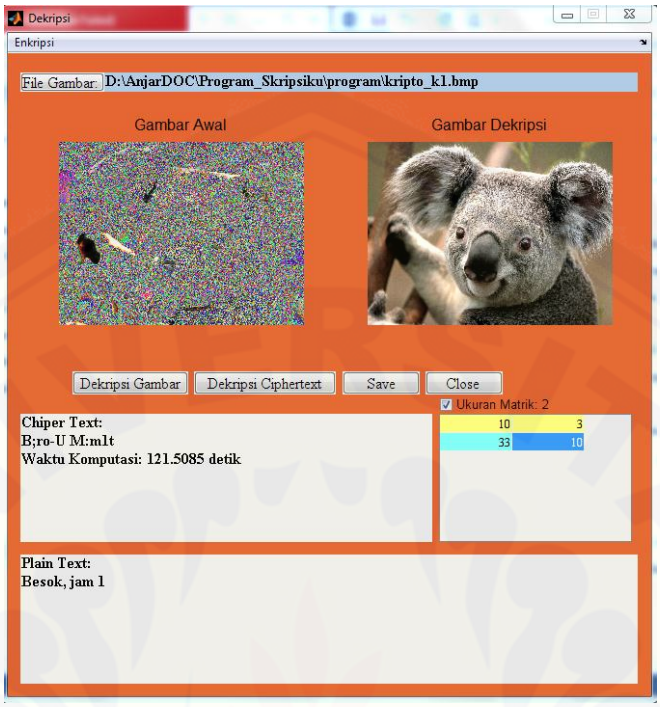
matriks $K_1 = \begin{bmatrix} 10 & 3 \\ 33 & 10 \end{bmatrix}$ dan matriks $K_2 = \begin{bmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix}$. Citra digital yang digunakan pada proses ini adalah seperti pada Gambar 4.3 dengan ukuran file 762 Kb.

Gambar 4.3 Citra digital mula-mula

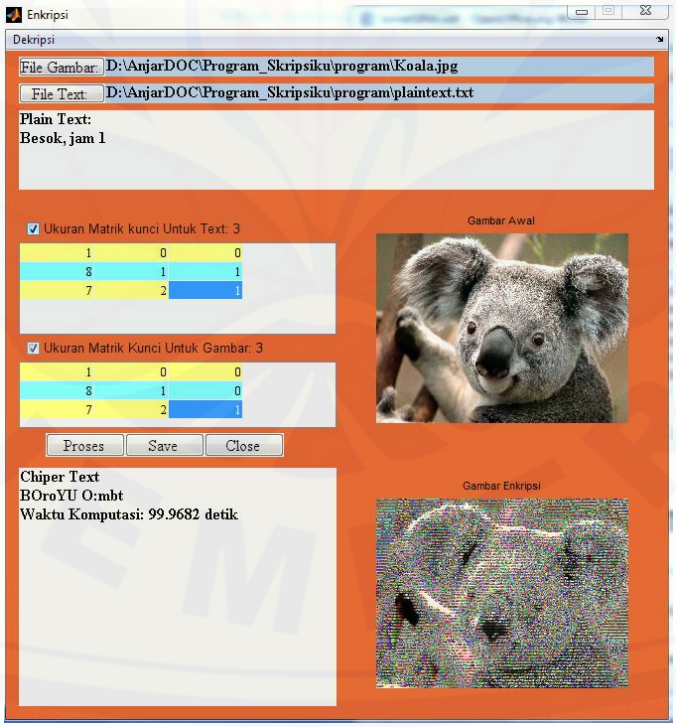
Tampilan *form* Enkripsi dan *form* Dekripsi yang dihasilkan dengan menggunakan matriks kunci K_1 pada proses enkripsi citra digital seperti pada Gambar 4.4 dan Gambar 4.5, sedangkan tampilan *form* Enkripsi dan *form* Dekripsi yang dihasilkan dengan menggunakan matriks kunci K_2 pada proses enkripsi citra digital seperti pada Gambar 4.6 dan Gambar 4.7.



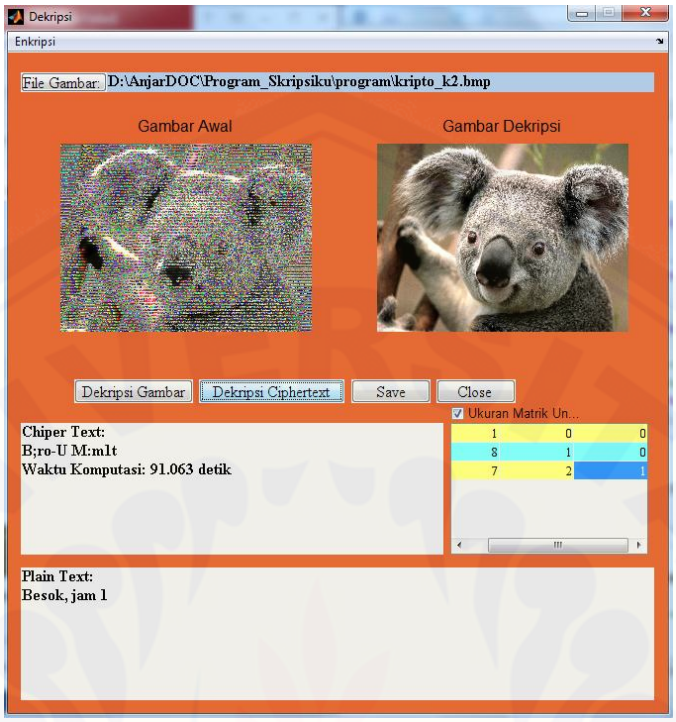
Gambar 4.4 Tampilan *form* Enkripsi menggunakan matriks kunci K_1



Gambar 4.5 Tampilan *form* Dekripsi menggunakan matriks kunci K_1



Gambar 4.6 Tampilan *form* Enkripsi menggunakan matriks kunci K_2



Gambar 4.7 Tampilan *form* Dekripsi menggunakan matriks kunci K_2

Berdasarkan pengujian dengan menggunakan program pada tampilan *form* enkripsi dan *form* dekripsi di atas, menunjukkan bahwa perhitungan program menghasilkan hasil yang sama, yaitu *plaintext* yang telah dikodekan dan disisipkan ke dalam citra digital dapat diekstrak dan didekripsi kembali sehingga dapat dikatakan bahwa program aplikasi dapat berjalan dengan baik.

4.4 Analisis Hasil Program

Berdasarkan pada simulasi program, komponen piksel pada potongan citra mula-mula dan komponen piksel pada potongan *stego object* ditunjukkan pada Gambar 4.8 dan Gambar 4.9.

R:101 G: 90 B: 58	R:103 G: 92 B: 62	R:110 G: 95 B: 66	R:104 G: 91 B: 59	R:104 G: 91 B: 59	R:104 G: 93 B: 63	R:108 G: 94 B: 65	R:100 G: 86 B: 57	R:103 G: 90 B: 56	R:105 G: 91 B: 64

Gambar 4.8 Potongan komponen piksel citra mula-mula

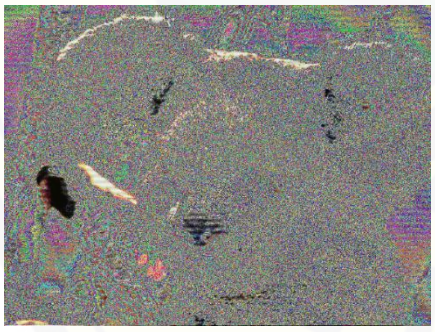
R:100 G: 90 B: 58	R:103 G: 92 B: 62	R:110 G: 94 B: 66	R:104 G: 90 B: 58	R:104 G: 90 B: 58	R:104 G: 92 B: 62	R:109 G: 95 B: 64	R:100 G: 87 B: 57	R:103 G: 90 B: 56	R:105 G: 91 B: 64

Gambar 4.9 Potongan komponen piksel *stego object*

Stego object yang dihasilkan dari proses *encoding* dengan menyisipkan *ciphertext*, ukuran matriks kunci dan elemen matriks kunci *K* seperti pada Gambar 4.10.

Gambar 4.10 *Stego object*

Citra kripto yang dihasilkan dari proses enkripsi *stego object* dengan menggunakan matriks kunci K_1 seperti pada Gambar 4.11 a dan citra kripto yang menggunakan matriks kunci K_2 dalam proses enkripsi *stego object* seperti pada Gambar 4.11 b.



(a)

(b)

Gambar 4.11 Citra kripto

Proses enkripsi *plaintext* menggunakan matriks kunci K menghasilkan *ciphertext* yang memiliki susunan karakter yang acak, sehingga *ciphertext* sangat sulit untuk dibaca. Namun, terdapat beberapa karakter pada *ciphertext* yang sama dengan karakter pada *plaintext*. Hal tersebut disebabkan oleh elemen matriks kunci yang digunakan pada proses enkripsi *plaintext* sangat kecil, yaitu $K = \begin{bmatrix} 1 & 0 & 0 \\ 8 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix}$.

Stego object yang dihasilkan dari proses *encoding* pesan teks terlihat sama dengan citra aslinya. *Stego object* tidak mengalami perubahan atau perbedaan yang signifikan jika dibandingkan dengan citra asli sebelum disisipkan pesan.

Proses enkripsi *stego object* dengan menggunakan matriks kunci K_2 menghasilkan citra kripto yang kurang baik karena masih dapat terlihat citra aslinya. Namun, proses enkripsi *stego object* dengan menggunakan matriks kunci K_1 menghasilkan citra kripto yang lebih baik karena tidak dapat terlihat citra aslinya. Pengkodean citra digital dengan menggunakan elemen matriks kunci yang besar akan menghasilkan citra kripto yang baik. Semakin besar elemen matriks kunci yang digunakan maka citra kripto yang dihasilkan tidak dapat terdeteksi seperti apa citra aslinya. Ukuran matriks kunci yang digunakan dalam proses enkripsi citra digital mempengaruhi cepat lambatnya proses enkripsi citra digital. Semakin besar ukuran matriks kunci yang digunakan, maka semakin cepat proses enkripsi citra digital. Proses enkripsi *stego object* dengan menggunakan matriks kunci berordo 2×2 membutuhkan waktu komputasi sebesar 123,2983 detik sedangkan proses enkripsi *stego object* dengan menggunakan matriks kunci berordo 3×3 membutuhkan waktu komputasi sebesar 99,9682 detik. Ukuran file *stego object* dan citra kripto pada Gambar 4.10 dan 4.11 lebih besar jika dibandingkan dengan citra aslinya pada Gambar 4.3, yakni dari 762 Kb menjadi 2,25 Mb.

BAB 5. PENUTUP

5.1 Kesimpulan

Berdasarkan hasil dan pembahasan yang telah diuraikan sebelumnya, didapatkan kesimpulan sebagai berikut:

a. Proses *encoding* pesan teks ke dalam citra digital menggunakan metode *Least Significant Bit*, yaitu dengan mengganti bit yang paling tidak berarti pada citra digital dengan bit data secara berurutan. Jumlah karakter yang dapat disisipkan ke dalam citra digital bergantung pada ukuran lebar citra digital (*width*). Semakin besar ukuran lebar citra digital, maka semakin banyak jumlah karakter yang dapat disisipkan. *Stego object* yang dihasilkan dari proses *encoding* pesan teks ke dalam citra digital terlihat sama dengan citra aslinya.
b. Kriptografi dengan algoritma *Hill Cipher* menggunakan matriks kunci pada proses enkripsi dan dekripsinya. Semakin besar elemen matriks kunci yang digunakan, maka *ciphertext* maupun citra kripto yang dihasilkan semakin berbeda dengan pesan aslinya atau semakin tidak dapat terdeteksi seperti apa citra aslinya. Ukuran matriks kunci pada proses enkripsi citra digital mempengaruhi cepat lambatnya proses enkripsi citra digital. Semakin besar ukuran matriks kunci yang digunakan, maka semakin cepat proses enkripsi citra digital.

5.2 Saran

Semakin besar ukuran berkas citra digital yang digunakan, maka semakin lama waktu yang dibutuhkan dalam proses enkripsi dan dekripsinya sehingga pada penelitian selanjutnya diharapkan dapat dikembangkan lebih lanjut agar proses enkripsi dan dekripsi citra digital menjadi lebih cepat. Pesan rahasia yang disisipkan ke dalam citra digital diharapkan tidak hanya berformat txt saja, tetapi juga dapat dikembangkan dengan format teks yang lain, seperti format doc dan format pdf. Selain itu, pesan

rahasia yang akan disisipkan tidak hanya berupa teks saja, melainkan dapat dikembangkan dengan media lainnya seperti citra, audio atau video.

DAFTAR PUSTAKA

Ahmad, U. 2005. *Pengolahan Citra Digital & Teknik Pemrogramannya*. Yogyakarta: Graha Ilmu.

Anton, H. & Rorres, C. 2005. *Aljabar Linear Elementer Versi Aplikasi Edisi Kedelapan Jilid 2*. Jakarta: Erlangga.

Aryus, D. 2006. *Kriptografi Keamanan Data dan Komunikasi*. Yogyakarta: Graha Ilmu.

Aryus, D. 2008. *Pengantar Kriptografi*. Yogyakarta: Graha Ilmu.

Hasibuan, Z. 2014. Perancangan Aplikasi Steganografi dengan Metode *Least Significant Bit* (LSB) untuk Data Terenkripsi dari Algoritma *Hill Cipher*: *Jurnal Informatika*, Vol. **6** (3): 150-154.

Munir, R. 2006. *Kriptografi*. Bandung: Informatika.

Prasetyo, E. 2011. *Pengolahan Citra Digital dan Aplikasinya menggunakan Matlab*. Yogyakarta: Andi.

Rachmawanto, E. H. "Teknik Keamanan Data menggunakan Kriptografi dengan Algoritma *Vernam Cipher* dan Steganografi Metode *End of File* (EOF)". Tidak Diterbitkan. Skripsi. Semarang: Universitas Dian Nuswantoro.

Suryadi, D. & Machmudi, S. H. 1985. *Teori dan Soal Pendahuluan Aljabar Linear*. Jakarta: Ghalia Indonesia.

Tarigan, D. B. 2014. Implementasi Algoritma Kriptografi *Hill Cipher* dalam Penyandian Data Gambar. Teknik Informatika STMIK Budi Darma Medan: *Jurnal Informatika*, Vol. **7** (2): 76-81.

LAMPIRAN A. Kode ASCII

ASCII *Code*	*Binnary Code*	*Character*	ASCII *Code*	*Binnary Code*	*Character*
000	00000000	null	128	10000000	Ç
001	00000001	☺	129	10000001	ű
002	00000010	☻	130	10000010	é
003	00000011	♥	131	10000011	â
004	00000100	♦	132	10000100	ä
005	00000101	♣	133	10000101	à
006	00000110	♠	134	10000110	å
007	00000111	•	135	10000111	ç
008	00001000	◘	136	10001000	ê
009	00001001	○	137	10001001	ë
010	00001010	◙	138	10001010	è
011	00001011	♂	139	10001011	ï
012	00001100	♀	140	10001100	î
013	00001101	♪	141	10001101	ì
014	00001110	♫	142	10001101	Ä
015	00001111	☼	143	10001111	Å
016	00010000	►	144	10010000	É
017	00010001	◄	145	10010001	æ
018	00010010	↕	146	10010010	Æ
019	00010011	‼	147	10010011	ô
020	00010100	¶	148	10010100	ö
021	00010101	§	149	10010101	ò
022	00010110	▬	150	10010110	û
023	00010111	↨	151	10010111	Ù
024	00011000	↑	152	10011000	ij
025	00011001	↓	153	10011001	Ö
026	00011010	→	154	10011010	Ü
027	00011011	←	155	10011011	¢
028	00011100	∟	156	10011100	£
029	00011101	↔	157	10011101	¥
030	00011110	▲	158	10011110	Pt
031	00011111	▼	159	10011111	*f*
032	00100000	spasi	160	10100000	á
033	00100001	!	161	10100001	í
034	00100010	“	162	10100010	ó
035	00100011	#	163	10100011	ú
036	00100100	$	164	10100100	ñ

ASCII Code	Binnary Code	Character	ASCII Code	Binnary Code	Character
037	00100101	%	165	10100101	N
038	00100110	&	166	10100110	ª
039	00100111	‘	167	10100111	º
040	00101000	(	168	10101000	¿
041	00101001	)	169	10101001	⌐
042	00101010	*	170	10101010	¬
043	00101011	+	171	10101011	½
044	00101100	,	172	10101100	¼
045	00101101	-	173	10101101	¡
046	00101110	.	174	10101110	«
047	00101111	/	175	10101111	»
048	00110000	0	176	10110000	░
049	00110001	1	177	10110001	▒
050	00110010	2	178	10110010	▓
051	00110011	3	179	10110011	│
052	00110100	4	180	10110100	┤
053	00110101	5	181	10110101	╡
054	00110110	6	182	10110110	╢
055	00110111	7	183	10110111	╖
056	00111000	8	184	10111000	╕
057	00111001	9	185	10111001	╣
058	00111010	:	186	10111010	║
059	00111011	;	187	10111011	╗
060	00111100	<	188	10111100	╝
061	00111101	=	189	10111101	╜
062	00111110	>	190	10111110	╛
063	00111111	?	191	10111111	┐
064	01000000	@	192	11000000	└
065	01000001	A	193	11000001	┴
066	01000010	B	194	11000010	┬
067	01000011	C	195	11000011	├
068	01000100	D	196	11000100	─
069	01000101	E	197	11000101	┼
070	01000110	F	198	11000110	╞
071	01000111	G	199	11000111	╟
072	01001000	H	200	11001000	╚
073	01001001	I	201	11001001	╔
074	01001010	J	202	11001010	╩
075	01001011	K	203	11001011	╦
076	01001100	L	204	11001100	╠

ASCII Code	Binnary Code	Character	ASCII Code	Binnary Code	Character
077	01001101	M	205	11001101	═
078	01001110	N	206	11001110	╬
079	01001111	O	207	11001111	╧
080	01010000	P	208	11010000	╨
081	01010001	Q	209	11010001	╤
082	01010010	R	210	11010010	╥
083	01010010	S	211	11010011	╙
084	01010100	T	212	11010100	╘
085	01010101	U	213	11010101	╒
086	01010110	V	214	11010110	╓
087	01010111	W	215	11010111	╫
088	01011000	X	216	11011000	╪
089	01011001	Y	217	11011001	┘
090	01011010	Z	218	11011010	┌
091	01011011	[	219	11011011	█
092	01011100	\	220	11011100	▄
093	01011101	]	221	11011101	▌
094	01011110	^	222	11011110	▐
095	01011111	_	223	11011111	▀
096	01100000	`	224	11100000	A
097	01100001	a	225	11100001	B
098	01100010	b	226	11100010	Γ
099	01100011	c	227	11100011	Σ
100	01100100	d	228	11100100	Π
101	01100101	e	229	11100101	Ơ
102	01100110	f	230	11100110	M
103	01100111	g	231	11100111	T
104	01101000	h	232	11101000	Φ
105	01101001	i	233	11101001	Θ
106	01101010	j	234	11101010	Ω
107	01101011	k	235	11101011	Ό
108	01101100	l	236	11101100	∞
109	01101101	m	237	11101101	ϕ
110	01101110	n	238	11101110	ϵ
111	01101111	o	239	11101111	ο
112	01110000	p	240	11110000	≡
113	01110001	q	241	11110001	±
114	01110010	r	242	11110010	≥
115	01110011	s	243	11110011	≤
116	01110100	t	244	11110100	⌠

ASCII ***Code***	***Binnary Code***	***Character***	**ASCII** ***Code***	***Binnary Code***	***Character***
117	01110101	u	245	11110101	⌡
118	01110110	v	246	11110110	÷
119	01110111	w	247	11110111	≈
120	01111000	x	248	11111000	**o**
121	01111001	y	249	11111001	∙
122	01111010	z	250	11111010	·
123	01111011	{	251	11111011	√
124	01111100	|	252	11111100	**N**
125	01111101	}	253	11111101	**2**
126	01111110	~	254	11111110	■
127	01111111	Del	255	11111111	**blank**

LAMPIRAN B. Proses Enkripsi *Plaintext* dan Enkripsi Citra Digital

hill_cipher.m

```
function [bil bil2]=hill_chiper(C,desimal)
aa=dec2bin(desimal);
desimal=bin2dec(aa);
n=length(desimal);
m=length(C);
k=mod(n,m);
if k~=0
   k=m-k;
   desimal(n+1:n+k)=ones(1,k)*32;
end
for i=1:m:(n+k)
   b=(desimal(i:(i+m-1)));
   K1=mod(C*(b),95);
   bil(i:(i+m-1))=K1;
   bil2(i:(i+m-1))=fix([C*desimal(i:(i+m-1))]/95);
end
```

hill_cipher_gambar.m

```
function gbr2=hill_chiper_gambar(gbr2,C)
[x y z]=size(gbr2);
n=length(C);
sisa=mod(x,n);
for i=1:n:(x-n-sisa)
   for j=1:y
      %R
      a=dec2bin(gbr2(i:(i+n-1),j,1));
      b=bin2dec(a);
```

```
        gbr2(i:(i+n-1),j,1)=mod(C*b,256);
        %G
        a=dec2bin(gbr2(i:(i+n-1),j,2));
        b=bin2dec(a);
        gbr2(i:(i+n-1),j,2)=mod(C*b,256);
        %B
        a=dec2bin(gbr2(i:(i+n-1),j,3));
        b=bin2dec(a);
        gbr2(i:(i+n-1),j,3)=mod(C*b,256);
    end
end
```

LAMPIRAN C. Proses Penyisipan Pesan (*Encoding*)

LSB.m

```
function [gbr2 a]=LSB(a,gbr,C,bil2)
gbr2=gbr;
n=length(a);
bil_bin=zeros(n,8);
a1=dec2bin(a);
[n m]=size(a1);
for i=1:n
   k=0;
   for j=(8-m+1):8
      k=k+1;
      bil_bin(i,j)=str2num(a1(i,k));
   end
end
%penyisipan pesan pada gambar: posisi Red
[x y z]= size(gbr);
if n<=x
   for i=1:n
         c1=dec2bin([gbr(i,1:8,1)]);
         [m1 m]=size(c1);
      for j=1:8
         c1(j,m)=num2str(bil_bin(i,j));
      gbr2(i,j,1)=bin2dec(c1(j,:));
      end
   end
end
```

```
bil_bin=zeros(n,8);
a1=dec2bin(bil2);
[n m]=size(a1);
for i=1:n
   k=0;
   for j=(8-m+1):8
      k=k+1;
      bil_bin(i,j)=str2num(a1(i,k));
   end
end
[x y z]= size(gbr);
if n<=x
   for i=1:n
         c1=dec2bin([gbr(i,(y-7):y,1)]);
         [m1 m]=size(c1);
      for j=1:8
         c1(j,m)=num2str(bil_bin(i,j));
      gbr2(i,(y-8)+j,1)=bin2dec(c1(j,:));
      end
   end
end
%menyisipkan ukuran kunci pada gambar: Posisi Green
n1=length(C);
q=dec2bin(n1);
n2=length(q);
b=zeros(1,8);
k=0;
for i=(8-n2+1):8
   k=k+1;
```

```
    b(1,i)=str2num(q(k));
end
    c1=dec2bin(gbr(1,1:8,2));
    [m0 m]=size(c1);
for i=1:8
    c1(i,m)=num2str(b(1,i));
    gbr2(1,i,2)=bin2dec(c1(i,:));
end
gbr2(1,1:8,2);
dec2bin(gbr2(1,1:8,2));
%menyisipkan elemen matriks kunci ke gambar:Posisi Blue
matrik=[];
for i=1:n1
    matrik=[matrik C(i,:)];
    k=length(matrik);
end
mat_bin=zeros(k,8);
mat_bin0=dec2bin(matrik);
[nn n]=size(mat_bin0);
for i=1:k
    k1=0;
    for j=(8-n+1):8
        k1=k1+1;
        mat_bin(i,j)=str2num(mat_bin0(i,k1));
    end
end
for i=1:k
    c1=dec2bin(gbr(i,1:8,3));
    [m1 m]=size(c1);
```

```
    for j=1:8
        c1(j,m)=num2str(mat_bin(i,j));
        gbr2(i,j,3)=bin2dec(c1(j,:));
    end
end
```

LAMPIRAN D. Proses Dekripsi Citra Kripto dan Ekstraksi Pesan (*Decoding*)

un_hill_gambar.m

```
function [gbr ]=un_hill_gambar(gbr,C)
[x y z]=size(gbr);
n1=length(C);
%invers matriks C
if n1>0
   if n1==2
  C_inv=[C(2,2) -C(1,2);-C(2,1) C(1,1)];
   for i=1:n1
      for i1=1:n1
         if C_inv(i,i1)<0
            C_inv(i,i1)=C_inv(i,i1)+256;
         end
      end
   end
C=C_inv;
   else
      C=inv(C);
      C=mod(C,256);
   end
%-------------------------------------
n=length(C);
sisa=mod(x,n);
for i=1:n:(x-n-sisa)
   for j=1:y
      %R
      a=dec2bin(gbr(i:(i+n-1),j,1));
```

```
        b=bin2dec(a);
        gbr(i:(i+n-1),j,1)=mod(C*b,256);
        %G
        a=dec2bin(gbr(i:(i+n-1),j,2));
        b=bin2dec(a);
        gbr(i:(i+n-1),j,2)=mod(C*b,256);
        %B
        a=dec2bin(gbr(i:(i+n-1),j,3));
        b=bin2dec(a);
        gbr(i:(i+n-1),j,3)=mod(C*b,256);
    end
end
end
```

dekripsi.m

```
function [matrik sandi sandi1 jmlh_kata]=dekripsi(gbr)
[x y z]=size(gbr);
% ambil ukuran matrik kunci
for i=1:2
    for j=1:8 %10
        kat=dec2bin(gbr(i,j,2));
        aa(j)=kat(end);
    end
    jml(i)=bin2dec(aa);
end
% mengambil elemen matrik kunci
aa='';
for i=1:(n^2)
    for j=1:8
        a=dec2bin(gbr(i,j,3));
```

```
        aa(j)=a(end);
    end
    matrik_b(i)=bin2dec(aa);
end
k=0;
for i=1:n
    matrik(i,:)=matrik_b(k+1:k+n);
    k=k+n;
end
% mengambil sandi
aa='';
for i=1:jmlh_kata
    for j=1:8
        a=dec2bin(gbr(i,j,1));
        aa(j)=a(end);
    end
    sandi(i)=bin2dec(aa);
end
%---------------------------------
% mengambil sandi posisi x,y-8
aa='';
for i=1:jmlh_kata
    for j=1:8
        a=dec2bin(gbr(i,y-8+j,1));
        aa1(j)=a(end);
    end
```

LAMPIRAN E. Proses Dekripsi *Ciphertext*

un_hill_chiper.m

```
function [bil_sandi]=un_hill_chipper(C,sandi,sandi1,jumlah_huruf)
sandi=sandi+95*sandi1;
K=(inv(C));
k=0;
n=length(sandi);
m=length(C);
k1=mod(n,m);
if k1~=0
   k1=m-k1;
   sandi(n+1:n+k1)=ones(1,k1)*(32+95);
   sandi1(n+1:n+k1)=ones(1,k1)*0;
end
for i=1:m:length(sandi)
   bil=(K*(sandi(i:i+(m-1))'))';
   bil_sandi(i:i+(m-1))=round(bil);
   k=length(bil_sandi);
end
```