

**IMPLEMENTASI STEGANOGRAFI PADA CITRA DIGITAL  
DENGAN METODE LEAST SIGNIFICANT BIT**

**SKRIPSI**

Disusun untuk melengkapi syarat-syarat guna memperoleh gelar  
Sarjana Komputer



**Oleh:**  
**AMELIA APRILIANI**  
**3145143626**

**PROGRAM STUDI ILMU KOMPUTER**  
**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**  
**UNIVERSITAS NEGERI JAKARTA**

**2018**

## **LEMBAR PERSETUJUAN**

Dengan ini saya mahasiswa Fakultas Matematika dan Ilmu Pengetahuan Alam,  
Universitas Negeri Jakarta

Nama : Amelia Apriliani

No. Registrasi : 3145143626

Jurusan : Ilmu Komputer

Judul : Implementasi Steganografi pada Citra *Digital*  
dengan Metode *Least Significant Bit*.

Menyatakan bahwa skripsi ini telah siap diajukan untuk seminar skripsi.

Menyetujui,

Dosen Pembimbing I

Dosen Pembimbing II

**Drs. Mulyono, M.Kom.**

NIP. 19660517 199403 1 003

**Ratna Widiyati, S.Si, M.Kom.**

NIP. 19750925 200212 2 002

Mengetahui,

Koordinator Program Studi Ilmu Komputer

**Drs. Mulyono, M.Kom.**

NIP. 19660517 199403 1 003

## **HALAMAN PERSEMBAHAN**

*Untuk Ayah, Mama,  
dan Adikku tercinta.*

## **KATA PENGANTAR**

Puji syukur penulis panjatkan ke hadirat Allah SWT Tuhan Yang Maha Kuasa karena hanya dengan ridho-Nya, Skripsi ini dapat terselesaikan tanpa halangan berarti. Keberhasilan dalam menyusun Skripsi ini tidak lepas dari bantuan berbagai pihak yang mana dengan tulus dan ikhlas memberikan masukan yang bermanfaat dalam proses penyusunan Skripsi ini. Oleh karena itu dalam kesempatan ini, dengan kerendahan hati penulis mengucapkan terima kasih kepada:

1. Bapak Drs. Mulyono, M.Kom, selaku Koordinator Program Studi Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Jakarta, dan juga selaku pembimbing pertama yang telah memberikan banyak bantuan, bimbingan, serta arahan dalam Tugas Akhir ini,
2. Ibu Ratna Widyati, S.Si, M.Kom, selaku dosen pembimbing kedua yang juga telah memberikan banyak bantuan, bimbingan, serta arahan dalam Tugas Akhir ini,
3. Seluruh Dosen Prodi Ilmu Komputer FMIPA UNJ yang tidak bisa disebutkan satu per satu, atas ilmu dan bimbingannya selama penulis berkuliahan di Ilmu Komputer FMIPA UNJ,
4. Ayah, Mama dan Icha yang selama ini telah sabar membimbing, mengarahkan, mendoakan penulis tanpa kenal lelah untuk selama-lamanya, dan juga memberikan dukungan secara moral dan financial untuk penulis,
5. Anita, Astia, Dika dan Olga yang selama ini telah menjadi sahabat penulis se-lama menjalankan kuliah di Ilmu Komputer UNJ, yang senantiasa memberikan dukungan dan semangat untuk menyelesaikan kuliah,

6. Kak Reyhan, Ferdiansyah dan Ardiansyah selaku pemberi ide untuk penulis dan juga yang membantu penulis dalam pengembangan program,
7. Teman-teman Ilmu Komputer 2014 atas dorongan, semangat serta hiburan yang senantiasa diberikan kepada penulis dalam keadaan suka maupun duka,
8. Ana, Yuyun, Hasna, Silvana, Fauziah, Dinda, Riska, Feno, Aditha, Nanda, Annisa, Mizalfia, Alfi, Yuni, Febri, Nadia, Rika, Sandra, Rohman, Edi, Banji dan teman-teman yang lain yang selalu memberikan dukungan semangat, dan menghibur penulis,
9. Febyan Nur Aditya, yang selama penyusunan skripsi ini selalu memberikan dukungan dan hiburan kepada penulis
10. Dan seluruh kerabat yang tidak dapat disebutkan satu per satu oleh penulis atas dukungan serta doa yang diberikan kepada penulis.

Penulis menyadari bahwa penyusunan Skripsi ini jauh dari sempurna. Akhir kata, teriring permintaan maaf apabila terdapat kesalahan maupun kekeliruan dalam penulisan Skripsi ini. Besar harapan penulis agar Skripsi ini dapat bermanfaat sebagaimana mestinya. Terima kasih.

Jakarta, ... 2018

**Penulis**

## **DAFTAR ISI**

<b>HALAMAN PERSEMBAHAN</b>	<b>iii</b>
<b>KATA PENGANTAR</b>	<b>iv</b>
<b>DAFTAR ISI</b>	<b>viii</b>
<b>DAFTAR GAMBAR</b>	<b>x</b>
<b>DAFTAR TABEL</b>	<b>xi</b>
<b>ABSTRAK</b>	<b>xii</b>
<b><i>ABSTRACT</i></b>	<b>xiii</b>
<b>I LATAR BELAKANG</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Perumusan Masalah . . . . .	3
1.3 Tujuan Penelitian . . . . .	3
1.4 Manfaat Penelitian . . . . .	3
<b>II KAJIAN TEORI</b>	<b>5</b>
2.1 Steganografi . . . . .	5
2.1.1 Pengertian Steganografi . . . . .	5
2.1.2 Sejarah Steganografi . . . . .	7
2.1.3 Metode Steganografi . . . . .	10
2.2 Perbedaan Steganografi dan Kriptografi . . . . .	13
2.3 LSB ( <i>Least Significant Bit</i> ) . . . . .	14
2.4 ASCII . . . . .	17

2.5 Citra <i>Digital</i> . . . . .	18
2.5.1 Pengertian Citra <i>Digital</i> . . . . .	18
2.5.2 Pengolahan Citra ( <i>Image Processing</i> ) . . . . .	18
2.5.3 Format <i>File</i> pada Citra <i>Digital</i> . . . . .	19
<b>III HASIL DAN PEMBAHASAN</b>	<b>22</b>
3.1 Pengumpulan Data . . . . .	22
3.2 Perancangan Sistem . . . . .	23
3.2.1 Proses Penyisipan ( <i>Encoding</i> ) pesan ke Citra <i>Digital</i> . . . . .	23
3.2.2 Proses Ekstraksi ( <i>Decoding</i> ) pesan dari Citra <i>Digital</i> . . . . .	25
3.2.3 Desain Antar Muka Program . . . . .	26
3.3 Program Steganografi dengan Metode LSB . . . . .	29
3.3.1 <i>Encoding</i> . . . . .	31
3.3.2 <i>Decoding</i> . . . . .	31
3.4 Hasil Steganografi . . . . .	32
3.4.1 Pengujian Berdasarkan Bit pada <i>File</i> Citra . . . . .	32
3.4.2 Pengujian Berdasarkan <i>Imperceptible</i> . . . . .	34
3.4.3 Pengujian Berdasarkan <i>Fidelity</i> . . . . .	35
3.4.4 Pengujian Berdasarkan <i>Recovery</i> . . . . .	37
3.4.5 Kesimpulan Pengujian . . . . .	39
<b>IV KESIMPULAN DAN SARAN</b>	<b>41</b>
4.1 Kesimpulan . . . . .	41
4.2 Saran . . . . .	41
<b>DAFTAR PUSTAKA</b>	<b>45</b>
<b>LAMPIRAN</b>	<b>46</b>



## DAFTAR GAMBAR

Gambar 2.1	Diagram Penyisipan dan Ekstraksi pada Pesan . . . . .	5
Gambar 2.2	Steganografi dengan Media Kepala Budak . . . . .	7
Gambar 2.3	Tablet <i>Wax</i> . . . . .	8
Gambar 2.4	Steganografi Zaman Perang Dunia . . . . .	9
Gambar 2.5	<i>Flowchart Encoding</i> LSB dan <i>Spread Spectrum</i> . . . . .	13
Gambar 2.6	Perbedaan Kriptografi dan Steganografi . . . . .	14
Gambar 3.1	Alur Penelitian . . . . .	22
Gambar 3.2	<i>Flowchart</i> Penyisipan Pesan Rahasia . . . . .	23
Gambar 3.3	<i>Flowchart</i> Ekstraksi Pesan Rahasia . . . . .	25
Gambar 3.4	GUI Steganografi . . . . .	26
Gambar 3.5	GUI - <i>Cover Image</i> . . . . .	27
Gambar 3.6	GUI - Proses <i>Encoding</i> . . . . .	28
Gambar 3.7	GUI - Pesan Hasil <i>Decoding</i> . . . . .	29
Gambar 3.8	<i>Source Code</i> - <i>Browse Image</i> . . . . .	30
Gambar 3.9	<i>Source Code</i> - Menghitung Karakter Maksimal . . . . .	30
Gambar 3.10	<i>Source Code</i> - Menghitung Panjang <i>Hiddentext</i> yang Dimaksimumkan . . . . .	30
Gambar 3.11	<i>Source Code</i> - Cuplikan <i>Encoding</i> . . . . .	31
Gambar 3.12	<i>Source Code</i> - Cuplikan <i>Decoding</i> . . . . .	32
Gambar 3.13	<i>File</i> Citra 8 Bit . . . . .	33
Gambar 3.14	<i>File</i> Citra 24 Bit . . . . .	33
Gambar 3.15	<i>File</i> Citra 32 Bit . . . . .	34
Gambar 3.16	Perbandingan <i>File</i> Citra - lena.bmp . . . . .	34
Gambar 3.17	Perbandingan <i>File</i> Citra - amelia.bmp . . . . .	35

Gambar 3.18 *File Citra lena.bmp Hasil Cropping . . . . .* 38

Gambar 3.19 *File Citra amelia.bmp Hasil Cropping . . . . .* 39

## **DAFTAR TABEL**

Tabel 2.1	Tabel ASCII [21] . . . . .	17
Tabel 2.2	Perbedaan <i>File</i> Citra Digital [7] . . . . .	21
Tabel 3.1	Bit pada <i>File</i> Citra . . . . .	32
Tabel 3.2	Karakter Maksimal Pesan pada <i>File</i> Citra . . . . .	36
Tabel 3.3	Hasil Proses <i>Encoding</i> . . . . .	36
Tabel 3.4	Hasil Proses <i>Decoding</i> . . . . .	37
Tabel 3.5	Hasil Proses <i>Cropping</i> . . . . .	38

## **ABSTRAK**

Steganografi dapat digunakan sebagai salah satu teknik dalam pengamanan informasi. Steganografi dapat menyembunyikan *file* pesan agar orang awam tidak menyadari keberadaan dari *file* pesan yang disembunyikan. Salah satu metode pada steganografi adalah *Least Significant Bit* (LSB). LSB melakukan penyisipan bit pesan ke dalam bit-bit *file* media yang digunakan. Pada tugas akhir ini, media yang digunakan adalah citra *digital RGB (Red, Green dan Blue)*. Pada citra *digital* 24 bit dan 32 bit proses penyisipan pesan berhasil dilakukan, sedangkan pada citra *digital* 8 bit proses penyisipan pesan gagal dilakukan. *File* citra hasil steganografi tidak mengalami perubahan yang cukup berarti dari file citra sebelumnya. Sehingga tidak akan menimbulkan kecurigaan dan keamanan pesan tetap terjaga.

**Kata kunci :** Steganografi, pesan, LSB, citra *digital*.

## ***ABSTRACT***

*Steganography can be used as one of the techniques in information security. Steganography can combine message files so others do not know of hidden message files. One method of steganography is Least Significant Bit (LSB). LSB inserts message bits into original media files of bits. In this final project, the media is digital image RGB (Red, Green and Blue) . In 24 bit and 32 bit digital image, the encoding process is success, but in 8 bit digital image the process is failed. Stego Image has not significant changes from Cover Image. So it will not cause suspicion and the security of the message is safe.*

***Keywords :*** *Steganography, message, LSB, digital image.*

## BAB I

### LATAR BELAKANG

#### 1.1 Latar Belakang

Saat ini *internet* sudah berkembang menjadi salah satu media yang sangat populer di berbagai dunia [4]. Perkembangan *internet* memberikan pengaruh besar terhadap kemudahan dalam berkomunikasi dan menyampaikan informasi. Komunikasi merupakan salah satu hal yang penting bagi manusia. Manusia yang merupakan makhluk sosial cenderung melakukan komunikasi setiap hari, baik secara langsung maupun melalui media elektronik. Manusia melakukan komunikasi untuk bertukar informasi.

Kemudahan dalam berkomunikasi memberikan dampak positif dan negatif. Dampak positifnya yaitu cepatnya informasi dapat tersebar, baik antar daerah maupun antar negara, sedangkan dampak negatifnya adalah semakin berkembangnya kejahatan dalam penggunaan informasi. Dengan berbagai teknik, banyak orang yang mencoba untuk mengakses informasi yang bukan haknya. Oleh karena itu harus berkembang juga pengamanan sistem informasi.

Teknik pengamanan informasi yang ada saat ini seperti kriptografi dan steganografi. Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikan pesan ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Kriptografi telah ada dan digunakan sejak berabad-abad yang lalu dikenal dengan istilah kriptografi klasik, yang bekerja pada mode karakter alfabet [19].

Steganografi adalah seni dan sains komunikasi pesan yang tidak terlihat. Hal ini dilakukan dengan menyembunyikan informasi dalam informasi lain, misalnya menyembunyikan keberadaan informasi yang dikomunikasikan. Kata steganografi bera-

sal dari kata Yunani "stegos" yang berarti "cover" dan "grafia" yang berarti "menulis" yang mendefinisikannya sebagai "tulisan tertutup" [12]. Steganografi merupakan teknik keamanan yang kuat, terutama ketika dikombinasikan dengan citra *digital* [7].

Salah satu metode steganografi adalah *Least Significant Bit* (LSB). Algoritma LSB, menggantikan bit paling signifikan pada *file cover* sesuai dengan bit pesan. Teknik ini adalah teknik yang paling populer digunakan dalam steganografi untuk menyembunyikan pesan. Teknik ini biasanya efektif, karena substitusi LSB tidak menyebabkan degradasi kualitas yang signifikan [10].

Pengimplementasian metode Least Significant Bit pada steganografi sudah pernah dilakukan penelitian oleh Fahri Perdana Prasetyo dengan format file \*.TIFF menggunakan bahasa pemrograman MATLAB [17]. Selain itu juga pernah dilakukan penelitian oleh Adiria dengan format file \*.BMP menggunakan bahasa pemrograman Delphi [1], sedangkan yang akan penulis buat nantinya adalah dengan mengkombinasikan kedua penelitian tersebut.

Penulis juga mencari jurnal-jurnal acuan dalam penelitian ini, seperti: *Comparison of LSB Steganography in BMP and JPEG Images; JPEG versus GIF Images in forms of LSB Steganography; A Survey on Digital Image Steganography Techniques; A New LSB-S Image Steganography Method Blend with Cryptography for Secret Communication, dan Steganography Using Least Significant Bit Algorithm.*

Dengan penjabaran di atas, penulis mengkombinasikan jurnal-jurnal tersebut untuk melakukan penelitian tentang "**Implementasi Steganografi pada Citral Digital dengan Metode Least Significant Bit**". Pada penelitian ini format *file* citra *digital* yang dapat digunakan untuk menyimpan pesan adalah berformat \*.bmp, dan citra hasil keluarannya juga berformat \*.bmp. Pesan yang dapat disimpan adalah pesan teks. Dengan adanya penelitian ini diharapkan dapat memberikan informasi mengenai steganografi.

## 1.2 Perumusan Masalah

Rumusan masalah berdasarkan latar belakang di atas adalah:

1. Bagaimana cara mengimplementasikan steganografi dengan metode *Least Significant Bit* ke dalam citra *digital*?
2. Bagaimana perubahan dalam *file* citra hasil keluaran sebelum dan sesudah disipkan pesan teks?

## 1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Memberikan informasi bagaimana steganografi dapat diimplementasikan ke dalam citra *digital* dengan menggunakan metode *Least Significant Bit*.
2. Mengetahui perubahan yang terjadi dari hasil keluaran *file* citra *digital*.

## 1.4 Manfaat Penelitian

Penelitian ini diharapkan memberikan manfaat sebagai berikut:

1. Bagi Penulis, diharapkan dapat menambah pengetahuan dan pemahaman tentang steganografi.
2. Bagi Program Studi Ilmu Komputer, penulisan penelitian ini dapat memberikan gambaran bagi seluruh mahasiswa khususnya bagi mahasiswa program studi Ilmu Komputer Universitas Negeri Jakarta tentang bagaimana steganografi dalam *file* citra *digital*.

3. Bagi Masyarakat, diharapkan dapat menjadi salah satu solusi dalam mengamankan *file* mereka dari orang-orang yang tidak mempunyai hak untuk melihatnya.

## BAB II

### KAJIAN TEORI

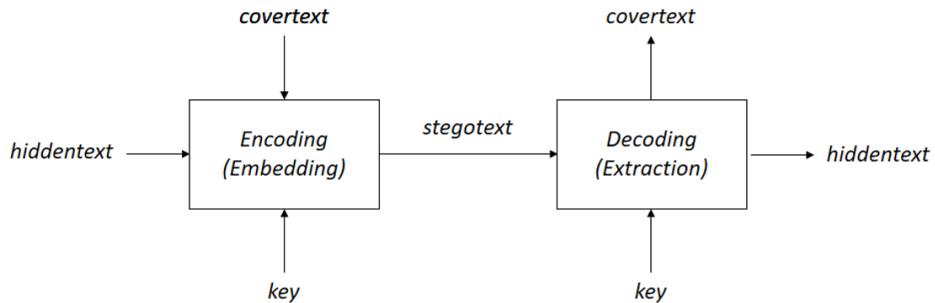
#### 2.1 Steganografi

##### 2.1.1 Pengertian Steganografi

Menurut **Gary C. Kessler** dalam jurnalnya *Steganography Hiding Data Within Data*:

"Steganografi adalah ilmu menyembunyikan informasi. Tujuan steganografi adalah untuk menyembunyikan data dari pihak ketiga." [11].

Secara umum, steganografi adalah seni untuk menyembunyikan pesan ke dalam media lain sedemikian rupa sehingga membuat orang lain tidak menyadari adanya pesan di media tersebut.



**Gambar 2.1:** Diagram Penyisipan dan Ekstraksi pada Pesan

Istilah di dalam steganografi:

1. *Covertext* merupakan media atau tempat pesan yang digunakan untuk menyembunyikan *hiddentext*. *Covertext* bisa berupa teks, gambar, audio, video, dll.

2. *Hiddentext* atau biasa disebut *embedded message* merupakan pesan atau informasi yang ingin disembunyikan. Contohnya bisa berupa teks, gambar, audio, video, dll.
3. *Stegotext* merupakan pesan yang sudah berisi *embedded message*.
4. *Encoding* yaitu penyisipan pesan ke dalam media *covertext*.
5. *Decoding* yaitu ekstraksi pesan dari *stegotext*.

Menurut **Munir**, ada kriteria yang harus diperhatikan dalam penyembunyian pesan, yaitu meliputi *Imperceptible*, *Fidelity*, *Recovery* dan *Capacity*.

#### 1. *Imperceptible*

Keberadaan pesan rahasia tidak dapat dipersepsi secara visual atau secara audio. Jika *covertext* berupa *file* citra, maka *stegotext* yang dihasilkan harus sukar dibedakan oleh kasat mata dengan *covertext*-nya. Dan jika *covertext* berupa *file* audio, maka telinga tidak dapat mendekripsi perubahan yang ada pada audio *stegotext*-nya.

#### 2. *Fidelity*

Kualitas *file* citra penampung tidak jauh berubah. Setelah penambahan pesan rahasia, citra hasil steganografi masih terlihat dengan baik. Pengamat tidak mengetahui kalau di dalam citra tersebut terdapat pesan rahasia.

#### 3. *Recovery*

Pesan yang disembunyikan harus dapat diekstrak kembali. Karena tujuan steganografi adalah menyembunyikan pesan atau informasi, maka jika informasi itu dibutuhkan harus dapat diambil kembali untuk dapat digunakan.

#### 4. Capacity

Ukuran pesan yang akan disembunyikan sedapat mungkin besar. Agar dapat memaksimalkan manfaat dari steganografi itu sendiri [14].

##### 2.1.2 Sejarah Steganografi

Seperti kriptografi, penggunaan steganografi sebetulnya telah digunakan berabad-abad yang lalu bahkan sebelum istilah steganografi itu sendiri muncul. Periode sejarah steganografi dapat dibagi menjadi:

###### 1. Steganografi Kuno (*Ancient Steganography*)

###### (a) Steganografi dengan media kepala budak

Ditulis oleh **Herodatus** (485-525 BC), sejarawan Yunani pada tahun 440 BC di dalam buku: *Histories of Herodatus*). Kisah perang antara kerajaan Persia dan rakyat Yunani. **Herodatus** menceritakan cara **Histaiaeus** mengirim pesan kepada **Aristagoras of Miletus** untuk melawan Persia. Caranya adalah dengan dipilih beberapa budak. Kemudian kepala budak tersebut digunduli dan ditulis pesan dengan cara ditato. Setelah pesan dituliskan, budak harus menunggu hingga rambutnya tumbuh kembali. Setelah rambut pada kepala budak tersebut tumbuh, budak dikirim ke tempat penerima. Di sana kepala budak digunduli agar pesan dapat dibaca.



**Gambar 2.2:** Steganografi dengan Media Kepala Budak

###### (b) Penggunaan tablet *wax*

Orang-orang Yunani kuno menulis pesan rahasia di atas kayu yang kemudian ditutup dengan lilin (*wax*). Di dalam bukunya, **Heradatus** menceritakan **Demaratus** mengirim peringatan tentang serangan yang akan datang ke Yunani dengan menulis langsung pada tablet kayu yang kemudian dilapisi lilin dari lebah.



**Gambar 2.3:** Tablet *Wax*

- (c) Penggunaan tinta tak-tampak (*invisible ink*)

**Pliny the Elder** menjelaskan penggunaan tinta dari getah tanaman *thithymallus*. Jika dituliskan pada kertas maka tulisan dengan tinta tersebut tidak kelihatan, tetapi bila kertas dipanaskan berubah menjadi gelap/coklat.

- (d) Penggunaan kain sutra dan lilin

Orang Cina kuno menulis catatan pada potongan-potongan kecil sutra yang kemudian digumpalkan menjadi bola kecil dan dilapisi lilin. Selanjutnya bola kecil tersebut ditelan oleh si pembawa pesan. Pesan dibaca setelah bola kecil dikeluarkan dari perut si pembawa pesan.

## 2. Steganografi Zaman Renaisans (*Renaissance Steganography*)

Tahun 1499, **Johannes Trithemius** menulis buku *Steganographia*, yang menceritakan tentang metode steganografi berbasis karakter. Selanjutnya tahun 1518 dia menulis buku tentang steganografi dan kriptografi berjudul *Polygraphiae*. **Giovanni Battista Porta** menggambarkan cara menyembunyikan pesan di dalam telur rebus. Caranya, pesan ditulis pada kulit telur yang dibuat dari tinta khusus yang dibuat dengan satu ons tawas dan setengah liter cuka. Prinsipnya penyembunyian adalah tinta tersebut akan menembus kulit telur yang berpori, tanpa meninggalkan jejak yang terlihat. Tulisan dari tinta akan membekas pada permukaan isi telur yang telah mengeras (karena sudah direbus sebelumnya). Pesan dibaca dengan membuang kulit telur.

3. Steganografi Zaman Perang Dunia (*World War Steganography*)



Rinaldi Munir/IF4C

**Gambar 2.4:** Steganografi Zaman Perang Dunia

Selama terjadinya Perang Dunia ke-2, tinta yang tidak tampak (*invisible ink*) telah digunakan untuk menulis informasi pada lembaran kertas sehingga saat kertas tersebut jatuh di tangan pihak lain hanya akan tampak seperti lembaran kertas kosong biasa. Cairan seperti air kencing (*urine*), susu, vinegar, dan jus buah digunakan sebagai media penulisan sebab bila salah satu elemen tersebut dipanaskan, tulisan akan menggelap dan tampak melalui mata manusia [14].

#### 4. Steganografi *Digital*

Sejalan dengan perkembangan maka konsep awal steganografi diimplementasikan pula dalam dunia komputer, yang kemudian dikenal dengan istilah steganografi *digital*. Dalam hal ini, steganografi *digital* memiliki dua properti dasar yaitu media penampung (*cover data* atau *data carrier*) dan data *digital* yang akan disisipkan (*secret data*), dimana media penampung dan data *digital* yang akan disisipkan dapat berupa *file multimedia* (teks/dokumen, citra, audio maupun video). Terdapat dua tahapan umum dalam steganografi *digital*, yaitu proses *embedding* atau *encoding* (penyisipan) dan proses *extracting* atau *decoding* (pemekaran atau pengungkapan kembali (*reveal*)). Hasil yang didapat setelah proses *embedding* atau *encoding* disebut *stego object* (apabila media penampung hanya berupa data citra maka disebut *stego image*) [18].

##### 2.1.3 Metode Steganografi

Berdasarkan ranah operasinya, metode-metode steganografi dapat dibagi menjadi dua kelompok:

###### 1. *Spatial (time) domain methods*

Memodifikasi langsung nilai byte dari *cover-object* (nilai byte dapat merepresentasikan intensitas/warna *pixel* atau amplitudo). Contoh: Metode modifikasi LSB

## 2. Transform domain methods

Memodifikasi hasil transformasi sinyal dalam ranah transform (hasil transformasi dari ranah spasial ke ranah lain (misalnya ranah frekuensi). Contoh: Metode *Spread Spectrum* [14].

Ada empat jenis metode steganografi:

### 1. Least Significant Bit Insertion (LSB)

Metode yang digunakan untuk menyembunyikan pesan pada media *digital* tersebut berbeda-beda. Contohnya, pada berkas *image* pesan dapat disembunyikan dengan menggunakan cara menyisipkannya pada bit rendah atau bit yang paling kanan (LSB) pada data *pixel* yang menyusun *file* tersebut. Pada berkas *bitmap* 24 bit, setiap *pixel* (titik) pada gambar tersebut terdiri dari susunan tiga warna *Red*, *Green* dan *Blue* (RGB) yang masing-masing disusun oleh bilangan 8 bit (*byte*) dari 0 sampai 255 atau dengan format biner 00000000 sampai 11111111. Dengan demikian, pada setiap *pixel* berkas *bitmap* 24 bit kita dapat menggantikan 3 bit data di setiap bit terakhir.

## 2. Algorithms and Transformation

*Algoritma compression* adalah metode steganografi dengan menyembunyikan data dalam fungsi matematika. Dua fungsi tersebut adalah *Discrete Cosine Transformation* (DCT) dan *Wavelet Transformation*. Fungsi DCT dan *Wavelet* yaitu mentransformasi data dari satu tempat (*domain*) ke tempat (*domain*) yang lain. Fungsi DCT yaitu mentransformasi data dari tempat *spatial* (*spatial domain*) ke tempat frekuensi (*frequency domain*).

## 3. Redundant Pattern Encoding

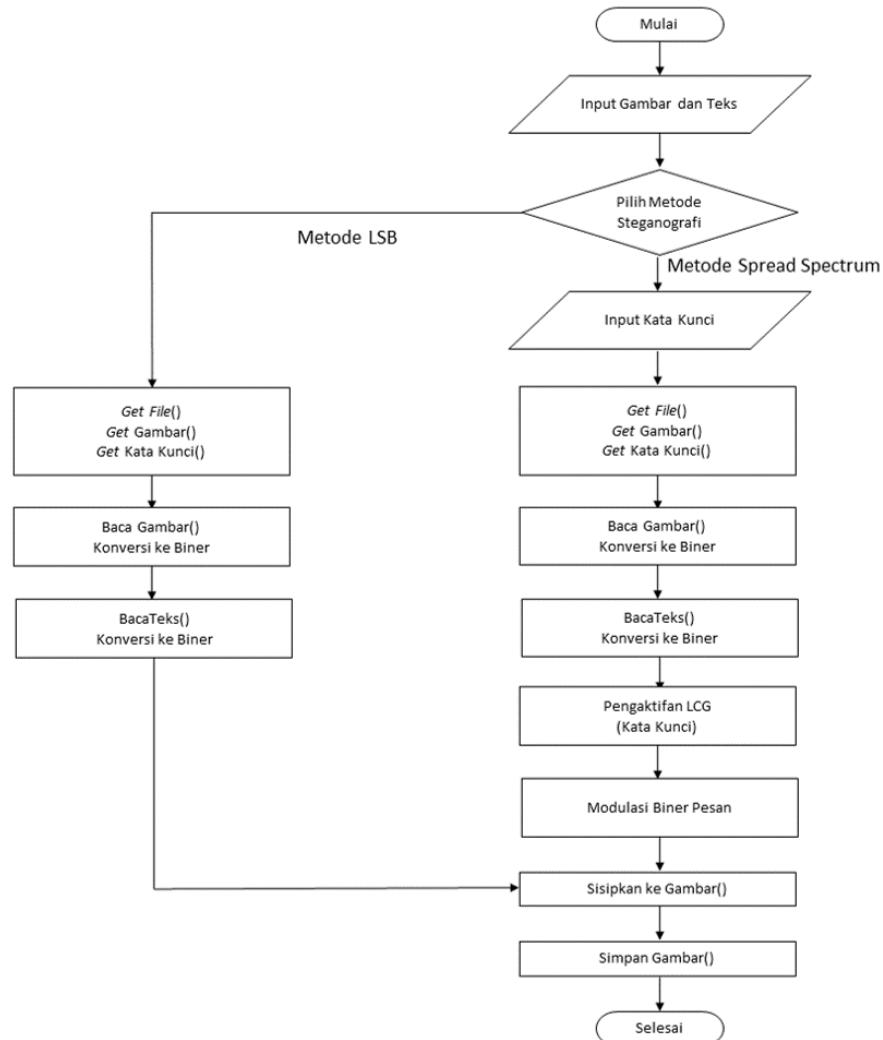
*Redundant Pattern Encoding* adalah menggambar pesan kecil pada kebanyakan

gambar. Keuntungan dari metode ini adalah dapat bertahan dari *cropping* (kegagalan). Kerugiannya yaitu tidak dapat menggambarkan pesan yang lebih besar.

#### 4. *Spread Spectrum Method*

*Spread Spectrum* steganografi terpencar-pencar sebagai pesan yang diacak (*encrypted*) melalui gambar (tidak seperti dalam LSB). Untuk membaca suatu pesan, penerima memerlukan algoritma yaitu *crypto-key* dan *stego-key*. Metode ini juga masih mudah diserang yaitu penghancuran atau pengrusakan dari kompresi dan proses *image* (gambar) [22].

Metode LSB dan *Spread Spectrum* adalah dua metode yang sering digunakan dalam melakukan steganografi. Selain karena metodenya yang sederhana, proses *encoding* dan *decoding* dari kedua metode tersebut juga *relative* cepat[16]. Tetapi LSB memiliki proses *encoding* dan *decoding* yang lebih cepat dari metode *Spread Spectrum* karena proses metode *Spread Spectrum* harus melalui proses XOR antara pesan dan kata kunci, sedangkan LSB langsung menyisipkan pesan ke dalam gambar [15].

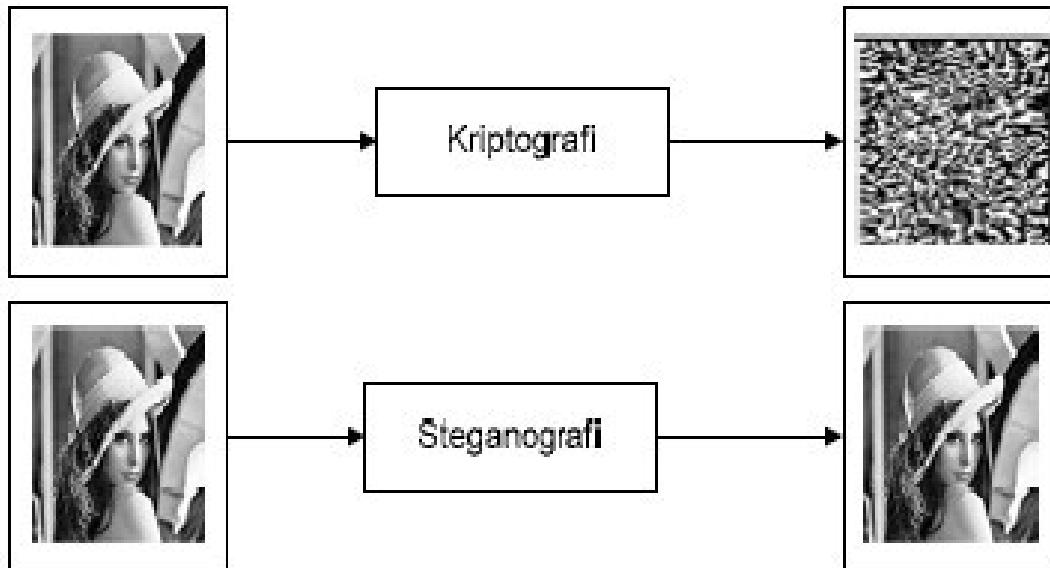


**Gambar 2.5:** Flowchart Encoding LSB dan Spread Spectrum

## 2.2 Perbedaan Steganografi dan Kriptografi

Steganografi dan kriptografi mempunyai prinsip kerja yang berbeda, meskipun keduanya mempunyai hubungan yang dekat dalam dunia keamanan data. Pada kriptografi menghasilkan sebuah *chiphertext* dimana dengan itu seolah-olah dengan sengaja menunjukkan kepada orang lain bahwa ada sesuatu di dalamnya, namun tidak dapat diketahui maknanya. Namun dengan bentuk *chiper*-nya, justru akan membuat

data tersebut terancam oleh usaha-usaha yang dilakukan oleh orang lain untuk dapat membongkarnya dengan tujuan dan atau alasan apapun.



**Gambar 2.6:** Perbedaan Kriptografi dan Steganografi

Steganografi dan kriptografi merupakan seni dan teknik yang dapat digunakan untuk melakukan pengamanan data *digital*. Namun keduanya tidaklah sama. Pada kriptografi, suatu data *digital* diamankan dengan cara mengenkripsi data tersebut dan menghasilkan sebuah data yang berupa sandi, secara visual data tersebut masih dapat terlihat atau diketahui, hanya saja data tersebut menjadi tidak dapat dimengerti. Berbeda dengan steganografi yang tujuannya adalah menyembunyikan data ke dalam sebuah media yang lain, sehingga data tersebut tidak terlihat [20].

### 2.3 LSB (*Least Significant Bit*)

Penyembunyian data dilakukan dengan mengganti bit-bit data di dalam segmen citra dengan bit-bit rahasia. Pada susunan bit di dalam sebuah *byte* (1 *byte*= 8 bit), ada bit yang paling berarti (*most significant bit* atau MSB) dan bit yang paling kurang berarti (*least significant bit* atau LSB). LSB merupakan salah satu metode

yang paling sederhana dalam steganografi. Bit yang cocok untuk diganti adalah bit LSB, sebab perubahan tersebut hanya mengubah nilai *byte* satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya [14].

Pada *file bitmap* RGB (*Red*, *Green*, dan *Blue*), setiap *pixel* memiliki 8 bit *Red*, 8 bit *Green*, dan 8 bit *Blue*. Sehingga dapat menyimpan 3 bit pesan pada setiap *pixel*-nya. Pada gambar 800x600 *pixel* dapat digunakan untuk menyembunyikan 1.440.000 bit (180.000 *Byte*) data rahasia. Sebagai contoh diambil 3 *pixel* dari *file bitmap* 24 bit yang akan disisipkan pesan atau data rahasia karakter "A":

(00001000 00101011 11011100)

(11100000 11000100 00010101)

(00010011 10101010 01100011)

Karakter "A" mempunyai nilai biner 01000001, maka bit-bit yang dihasilkan adalah:

(00001000 00101011 11011100)

(11100000 11000100 0001010**0**)

(0001001**0** 1010101**1** 01100011)

Bit-bit yang nilainya berganti ada 3 dalam 8 *Byte* yang digunakan. Contoh lainnya adalah diambil 8 pixel dari sebuah gambar, maka data rahasia yang dapat dimasukkan adalah 1 kata, contohnya adalah "ADA"

(10011011 01100100 01010000)  
(10010011 01010101 01001000)  
(10011010 01010111 01001110)  
(10011010 01010101 01010000)  
(10001000 01000001 00111111)  
(01101001 00100010 00110100)  
(01101101 00100111 00110010)  
(01111001 00110011 00110101)

Kata "ADA" mempunyai biner A = 01000001, D = 01000100, maka bit-bit yang dihasilkan adalah:

(100110**1** 011001**01** 01010000)  
(100100**10** 010101**00** 01001000)  
(10011010 01010111 01001110)  
(100110**11** 010101**00** 01010000)  
(10001000 01000001 001111**10**)  
(011010**00** 00100010 001101**01**)  
(011011**00** 00100111 00110010)  
(011110**00** 00110011 00110101)

Bit-bit yang nilainya berganti ada 13 dalam 24 *Byte* yang digunakan. Secara rata-rata, LSB hanya menggunakan setengah dari bit dalam gambar yang perlu dimodifikasi untuk menyembunyikan pesan rahasia. Perubahan ini tidak dapat dirasakan oleh mata manusia, dan pesan berhasil disembunyikan [6].

## 2.4 ASCII

ASCII adalah singkatan dari *American Standard Code for Information Interchange*. Komputer hanya dapat memahami angka, jadi kode ASCII adalah representasi numerik dari karakter seperti 'a' atau '@' atau karakter lainnya. Kode ASCII sebenarnya memiliki komposisi bilangan biner sebanyak 7 bit. Namun, ASCII disimpan sebagai sandi 8 bit dengan menambahkan satu angka 0 sebagai bit significant paling tinggi, dan disebut dengan *Extended ASCII*. Dimulai dari 00000000 hingga 11111111. Total kombinasi yang dihasilkan ASCII sebanyak 256, dimulai dari kode 0 hingga 255 dalam sistem bilangan desimal. [3]

**Tabel 2.1:** Tabel ASCII [21]

## 2.5 Citra Digital

### 2.5.1 Pengertian Citra Digital

Citra atau gambar dapat didefinisikan sebagai sebuah fungsi dua dimensi,  $f(x,y)$ ,  $x$  dan  $y$  adalah koordinat bidang datar; dan harga fungsi  $f$  di setiap pasangan koordinat  $(x,y)$  disebut intensitas atau level keabuan (*grey level*) dari gambar di titik itu [8]. Citra ada 2 macam, yaitu:

1. Citra kontinu, yaitu citra yang dihasilkan dari sistem optik yang menerima sinyal analog, misal: mata manusia dan kamera analog.
2. Citra diskrit, yaitu citra yang dihasilkan melalui proses digitalisasi terhadap citra kontinu.

Agar dapat diolah dengan komputer, maka suatu citra harus direpresentasikan secara *numeric* dengan nilai-nilai diskrit. Representasi citra dari fungsi kontinyu menjadi nilai-nilai diskrit disebut digitalisasi, dan citra yang dihasilkan disebut citra *digital*.

Ada 3 bidang studi utama yang menangani pengolahan data atau informasi berbentuk gambar atau citra, yaitu:

1. Grafika Komputer (*Computer Graphics*)
2. Pengolahan Citra (*Image Processing*)
3. Pengenalan Pola (*Pattern Recognition*)

### 2.5.2 Pengolahan Citra (*Image Processing*)

Pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakan komputer, menjadi citra yang kualitasnya lebih baik. Pengolahan Citra bertujuan

memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (dalam hal ini komputer). Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas lebih baik daripada citra masukan [13]

### 2.5.3 Format *File* pada Citra Digital

#### 1. BMP

BMP adalah singkatan dari *Bitmap* yang dahulu dikembangkan oleh MICRO-SOFT. *Bitmap* dapat menyimpan data warna untuk masing-masing *pixel* dalam gambar tanpa kompresi apapun. Format ini dapat digunakan untuk menyembunyikan data tanpa menaikkan kecurigaan pada mata manusia. Gambar yang dihasilkan tanpa kompresi dan format *lossless* yang merupakan salah satu faktor penting. Ekstensi yang digunakan dalam *file* ini adalah .bmp [7].

#### 2. JPEG

Istilah JPEG sebenarnya adalah singkatan dari pengembangnya, yaitu *Joint Photographic Experts Group*. Gambar JPEG tidak terbatas pada sejumlah warna tertentu. Oleh karena itu, format JPEG paling baik untuk mengompresi gambar foto. Gambar dengan format JPEG dapat berisi data gambar beresolusi tinggi berwarna-warni. JPEG memiliki sifat *lossy*, yang berarti beberapa kualitas hilang ketika gambar dikompresi. Jika gambar terlalu banyak dikompres, grafiknya menjadi seperti "tidak berwarna" dan sebagian detailnya hilang. Ekstensi yang digunakan dalam *file* ini adalah .jpeg [5].

#### 3. GIF

GIF adalah singkatan dari *Graphics Interchange Format* yang dikembangkan oleh COMPUSERVICE. GIF digunakan untuk tujuan menyimpan beberapa gambar *bitmap* dalam satu *file* gambar. GIF sering digunakan untuk menyimpan

grafik multi-bit dan data gambar. GIF tidak terkait dengan aplikasi perangkat lunak tertentu tetapi dirancang untuk memudahkan pertukaran dan tampilan data gambar yang tersimpan di lokal atau sistem komputer jarak jauh. GIF digunakan juga karena menerapkan metode kompresi *lossless*. Ekstensi yang digunakan dalam *file* ini adalah .gif [6].

#### 4. TIFF

TIFF adalah singkatan dari *Taged Image Format File*. TIFF dikembangkan oleh ADOBE dan digunakan untuk grafis berkualitas tinggi dengan kompresi *lossless*. Format *file* ini memiliki transparansi dan pilihan warna terindeks untuk menanamkan pesan rahasia di atasnya. TIFF mendukung properti RGB dan *GRAYSCALE* dan digunakan untuk HD *Imaging*. Ini adalah salah satu format *file* paling serbaguna di antara semua format yang tersedia. Ekstensi yang digunakan dalam format *file* ini adalah .tiff [7].

#### 5. PNG

PNG adalah singkatan dari *Portable Network Graphics* yang dikembangkan oleh *PNG Development Group*. PNG mampu menyembunyikan pesan yang besar di dalamnya. Format *file* ini diciptakan untuk meningkatkan format *file* gambar GIF menghilangkan batasan 256 warna tetapi tidak mendukung animasi. Dan PNG menggunakan kompresi data *lossless*. Ekstensi yang digunakan dalam format *file* ini adalah .png [7].

Perbedaan komponen antara masing-masing format *file* citra *digital* dapat dilihat pada Tabel 2.2. [7]

**Tabel 2.2:** Perbedaan *File Citra Digital* [7]

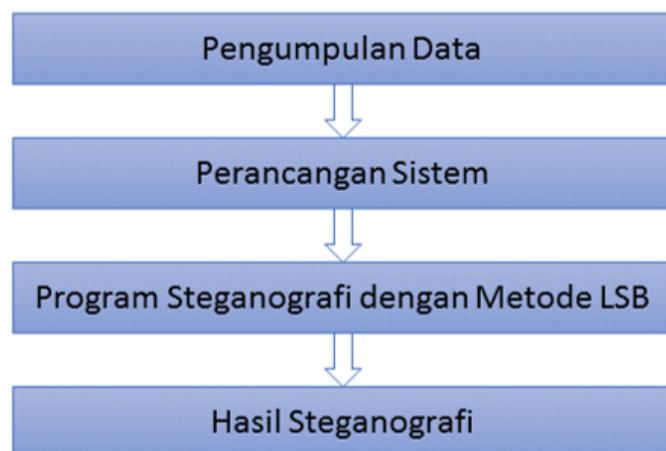
Komponen	BMP	JPEG	GIF	TIFF	PNG
Kompresi <i>Lossless</i>	Ya	Tidak	Ya	Ya	Ya
<i>Grayscale</i>	Ya	Ya	Ya	Ya	Ya
RGB	Terbatas	Ya	Ya	Ya	Ya
Index Pilihan Warna	Ya	Tidak	Ya	Ya	Ya
Transparansi	Tidak	Tidak	Ya	Tidak	Ya
Pilihan Animasi	Tidak	Tidak	Ya	Tidak	Tidak
<i>Color bits</i>	32	24	24	24, 48	24, 48

Pada tabel di atas terlihat bahwa pada komponen kompresi *lossless* JPEG tidak memilikinya, ini dikarenakan JPEG memiliki sifat *lossy*, yang berarti beberapa kualitas hilang ketika gambar dikompresi. Pada komponen RGB menurut jurnal "*A Survey on Digital Image Steganography Techniques*" BMP memiliki sifat *limited*. Untuk format GIF dan PNG keduanya memiliki komponen transparansi. Selain memiliki komponen transparansi, pada format GIF juga memiliki komponen animasi. Semua format mendukung *color bits* 24 bit. BMP mendukung juga sampai 32 bit, sedangkan pada TIFF dan PNG mendukung hingga 48 bit.

## BAB III

### HASIL DAN PEMBAHASAN

Dalam penelitian ini tahapan yang akan dilakukan adalah seperti gambar di bawah ini



**Gambar 3.1:** Alur Penelitian

#### 3.1 Pengumpulan Data

##### 1. Studi Pustaka

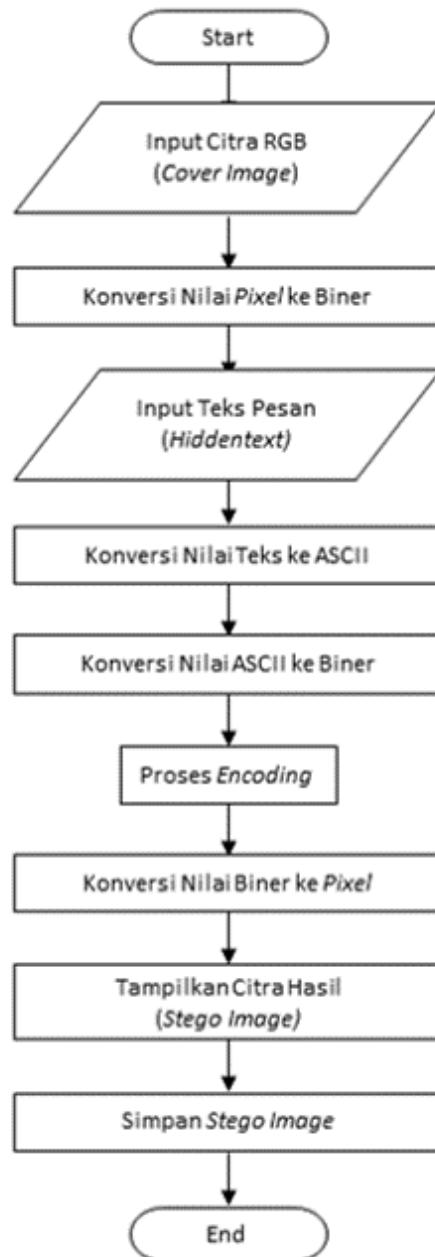
Penulis mendapatkan informasi yang berkaitan dengan steganografi melalui buku referensi dan juga dalam bentuk *e-book*. Penulis juga mencari informasi melalui berbagai situs di internet yang sesuai dengan topik.

##### 2. Studi Literatur

Penulis mencoba mencari perbandingan dengan studi sejenis dari beberapa karya ilmiah lokal maupun internasional, seperti jurnal dan skripsi.

## 3.2 Perancangan Sistem

### 3.2.1 Proses Penyisipan (*Encoding*) pesan ke Citra Digital

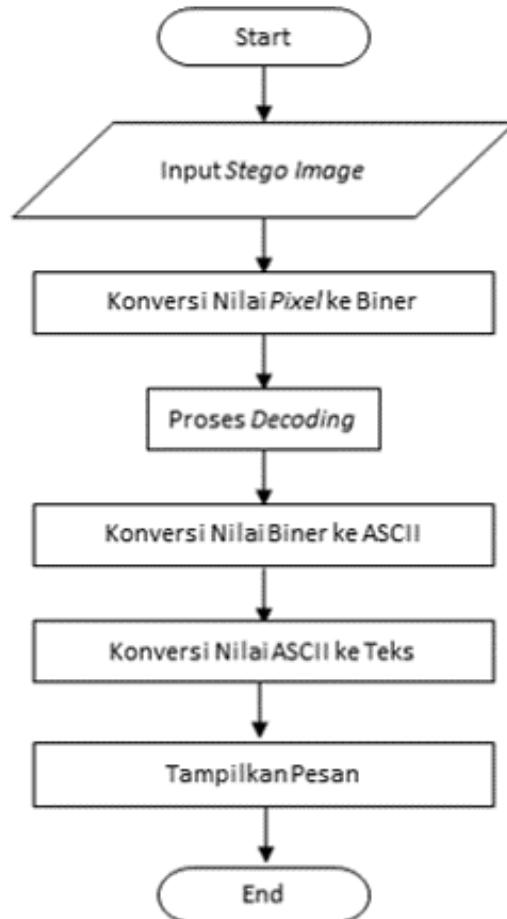


Gambar 3.2: Flowchart Penyisipan Pesan Rahasia

Pada gambar di atas adalah *flowchart* proses penyisipan pesan ke dalam *file* citra (*Cover Image*). Dimulai dengan membaca *file* citra RGB. Untuk *file* bitmap RGB maka setiap *pixel* (titik) pada gambar tersebut terdiri dari susunan tiga warna *Red*, *Green* dan *Blue* yang masing-masing disusun oleh bilangan 8 bit (1 *byte*) dari 0 sampai 255 atau dengan format biner 00000000 sampai 11111111. Setelah membaca *pixel* dari *file* citra langkah selanjutnya menentukan bit terkecil (LSB) pada *Cover Image*.

Selanjutnya adalah menyisipkan pesan (*Hiddentext*) yang akan disembunyikan ke dalam *Cover Image*. Pesan tersebut dikonversi terlebih dahulu menjadi nilai ASCII dan kemudian dikonversi kembali menjadi nilai Biner. Setelah itu terjadilah proses penyisipan (*Encoding*). Selanjutnya biner yang telah disisipkan akan dikonversikan kembali ke dalam *pixel*. Dan menyimpan citra yang telah disisipkan pesan ke dalam *Cover Image* sehingga diperoleh atau dapat ditampilkan sebuah gambar baru (*Stego Image*).

### 3.2.2 Proses Ekstraksi (*Decoding*) pesan dari Citra Digital

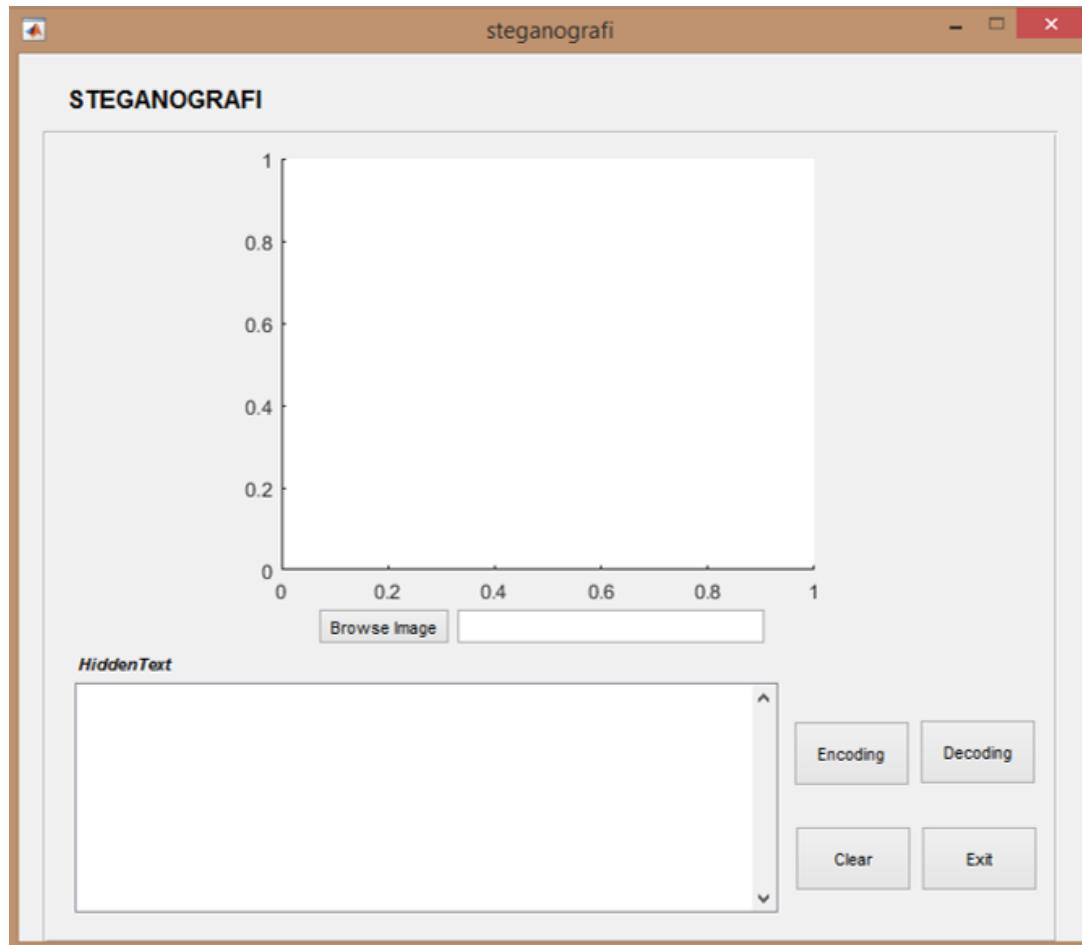


**Gambar 3.3:** Flowchart Ekstraksi Pesan Rahasia

Pada gambar di atas adalah *flowchart* proses ekstraksi pesan dari *Stego Image* yang menghasilkan *Hiddentext* yang terdapat di dalamnya. Prosesnya dimulai dengan membaca *file* citra, dan mengubah *pixel* ke dalam nilai biner. Kemudian proses ekstraksi (*Decoding*). Setelah diperoleh bit-bit yang tersembunyi pada *Cover Image* maka proses berikutnya adalah mengkonversi kembali pesan yang tersembunyi (*Hiddentext*), sehingga pesan dapat ditampilkan kembali.

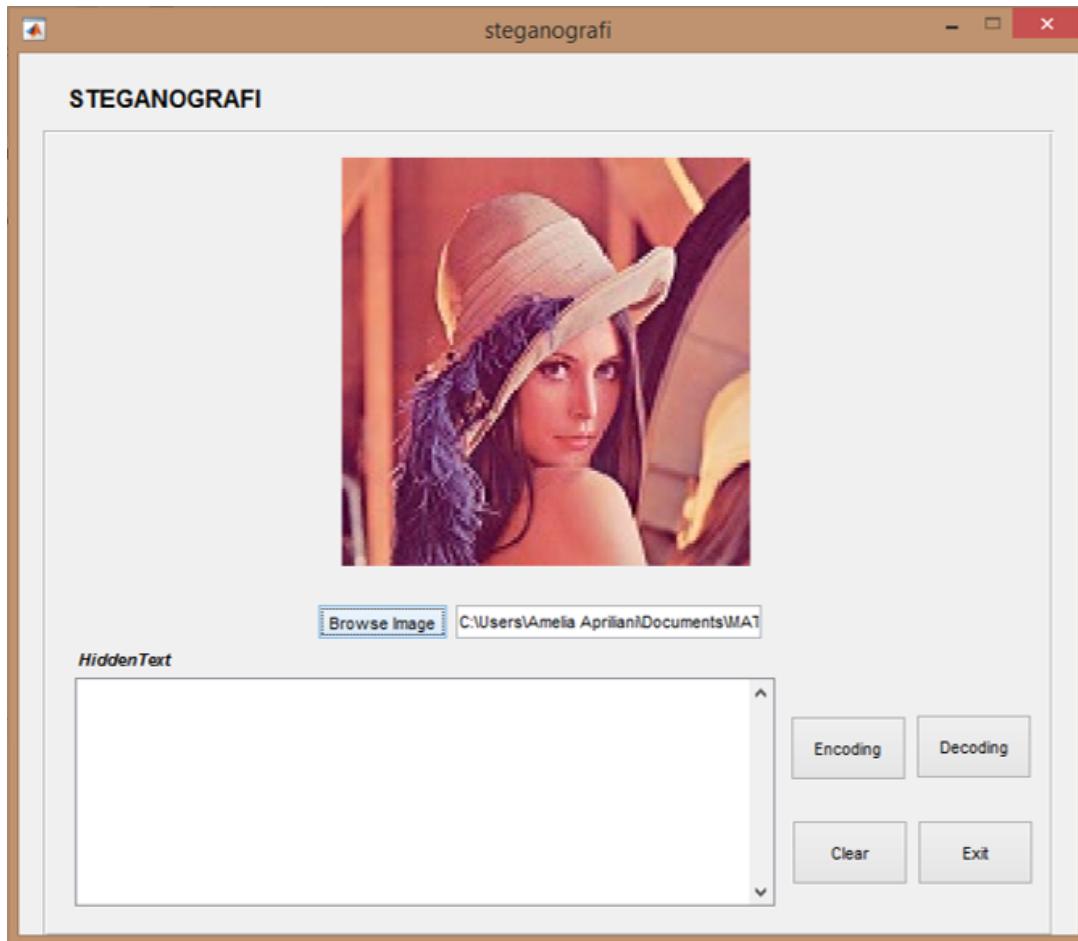
### 3.2.3 Desain Antar Muka Program

Berikut adalah GUI dari program steganografi yang dibangun dengan menggunakan Matlab.



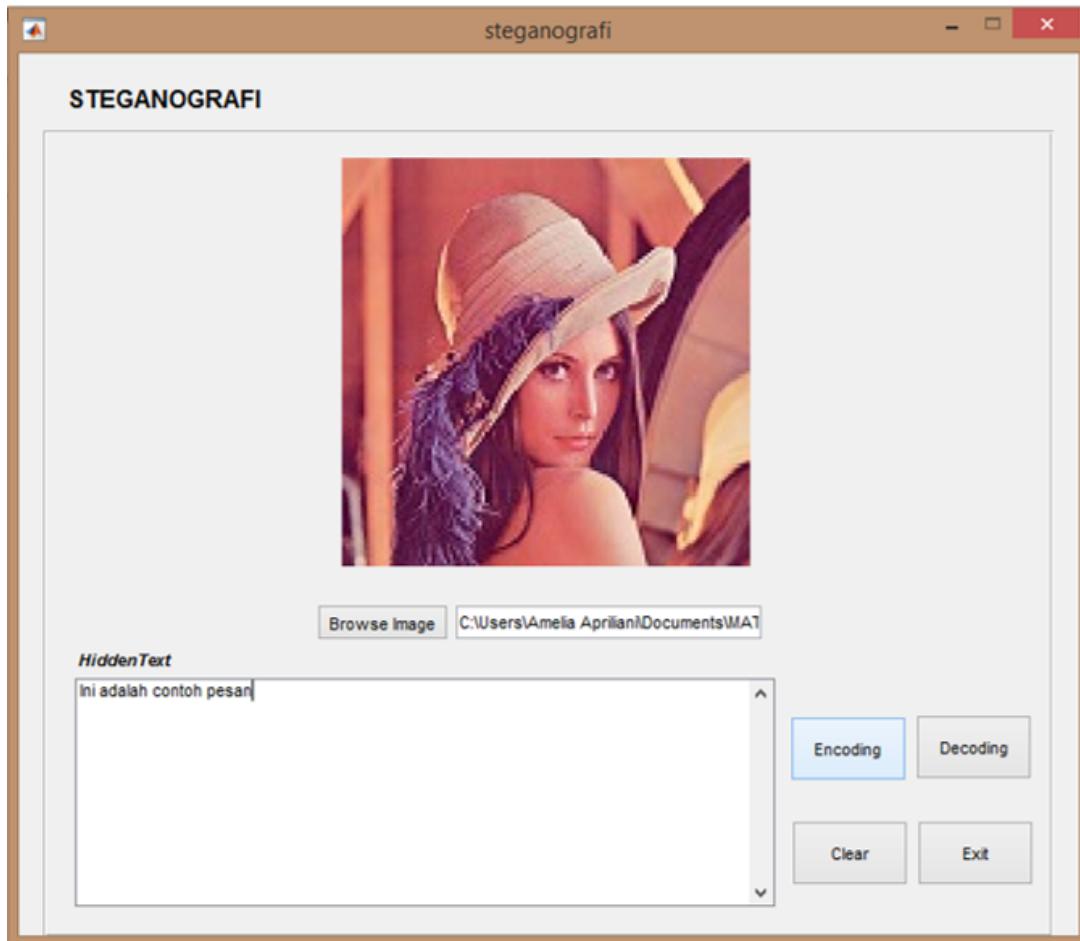
**Gambar 3.4:** GUI Steganografi

Dari tampilan tersebut, pengambilan gambar yang akan dijadikan sebagai *Cover Image* dilakukan dengan menekan tombol "*Browse Image*" dan gambar akan tampil.



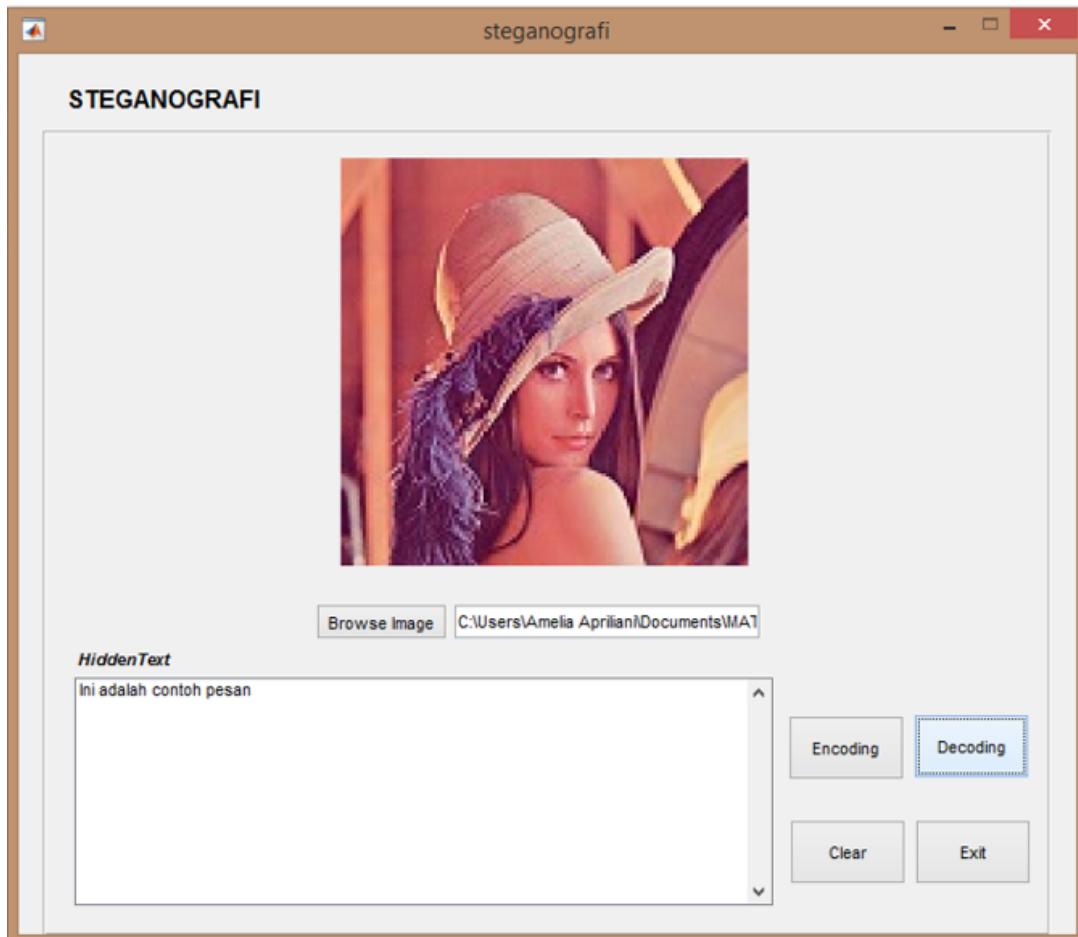
**Gambar 3.5:** GUI - *Cover Image*

Setelah *Cover Image* tampil maka pesan yang akan disisipkan atau *Hiddentext* dapat dituliskan pada kolom pesan. Kemudian klik tombol *Encoding* untuk melakukan proses *Encoding*. Setelah proses *Encoding*, maka akan didapatkan *Stego Image* dan disimpan di dalam folder.



**Gambar 3.6:** GUI - Proses *Encoding*

Jika ingin melakukan proses *Decoding*, maka buka *Stego Image* yang telah disimpan. Kemudian klik tombol *Decoding* dan pesan akan didapatkan.



**Gambar 3.7:** GUI - Pesan Hasil *Decoding*

### 3.3 Program Steganografi dengan Metode LSB

Program diawali dengan pengambilan *file* citra *digital* yang akan digunakan, berikut adalah *source code*-nya:

```

global image;

[nama_file,format_file] = uigetfile({'*.bmp'}, 'Pilih Gambar');
image = imread([format_file,nama_file]);
handles.image = image;
guidata(hObject,handles);
axes(handles.axes2);
imshow(image);

lokasi_image = fullfile(format_file, nama_file);
set(handles.kolom_lokasi,'String',lokasi_image);

```

**Gambar 3.8:** Source Code - Browse Image

Selanjutnya adalah memasukkan pesan atau *Hiddentext*. *Hiddentext* yang akan dimasukkan tidak boleh melebihi batas maksimal karakter. Panjang karakter maksimal disesuaikan dengan besar *pixel* pada *file* citra. Panjang karakter maksimal dapat dihitung, berikut adalah *source code*-nya:

```

%penentuan maksimal karakter pesan
char_max = (row -1)*(column);
char_max = round((char_max*3)/8);

```

**Gambar 3.9:** Source Code - Menghitung Karakter Maksimal

Sedangkan untuk menghitung panjang *hiddentext* yang telah dimasukkan adalah sebagai berikut:

```

%perhitungan panjang pesan
row_max = row;
column_max = column;
hiddentext_length = length(hiddentext);
if hiddentext_length < char_max
    hiddentext_biner = strcat(reshape(dec2bin(double(hiddentext),8)',1,[1]), '00000000')
    hiddentext_save = hiddentext_biner;
    hiddentext_asli = char(bin2dec(reshape(hiddentext_biner,8,[1]) .'))
else
    msgbox('Maaf,pesan terlalu panjang','peringatan','warn');
    return;
end

```

**Gambar 3.10:** Source Code - Menghitung Panjang *Hiddentext* yang Dimasukkan

### 3.3.1 Encoding

Pada tugas akhir ini metode yang digunakan dalam steganografi adalah metode LSB. Proses *Encoding* merupakan proses penyisipan pesan atau *Hiddentext* ke dalam *file* citra. Berikut adalah cuplikan *source code* proses *Encoding* dengan menggunakan metode LSB:

```
%encoding
hiddentext_length = hiddentext_length*8;
for i = 1:row_max-1
    for j = 1:column_max
        if hiddentext_length ~= 0
            image_biner_red = dec2bin(image_red(i,j),8);
            image_biner_red(1,8) = hiddentext_biner(1,1);
            image_red(i,j) = bin2dec(image_biner_red);

            hiddentext_biner(1:1) = [];
            hiddentext_length = length(hiddentext_biner);
        end
    end
end
```

**Gambar 3.11:** *Source Code - Cuplikan Encoding*

### 3.3.2 Decoding

Proses *Decoding* merupakan proses pengambilan kembali pesan atau *Hidden-text* yang telah disisipkan ke dalam *file* citra. Berikut merupakan cuplikan dari *source code* proses *Decoding*:

```
%decoding
hiddentext = '';
var_null = 1;
for i = 1:row_max-1
    for j = 1:column_max
        biner_length = length(hiddentext);
        if biner_length < hiddentext_length && var_null<=8
            image_biner_red = dec2bin(image_red(i,j),8);
            hiddentext_red = image_biner_red(1,8);
            hiddentext = strcat(hiddentext, hiddentext_red);
            if hiddentext_red == 0
                var_null = var_null + 1;
            else
                var_null = 1;
            end
        else
            hiddentext_asli = char(bin2dec(reshape(hiddentext,8,[1]).'));
            set(handles.edit_pesan,'String',hiddentext_asli);
            return;
        end
    end
end
```

**Gambar 3.12:** Source Code - Cuplikan Decoding

### 3.4 Hasil Steganografi

Setelah dilakukan pengujian terhadap program Steganografi, maka didapatkan hasil sebagai berikut:

#### 3.4.1 Pengujian Berdasarkan Bit pada *File* Citra

Pengujian ini menguji berapakah batas Bit *File* Citra yang bisa dimasukkan di dalam program.

**Tabel 3.1:** Bit pada *File* Citra

Citra	Bit	Pixel	Hasil
lena_gray	8	512 x 512	Gagal
lena	24	128 x 128	Berhasil
amelia	32	1576 x 2364	Berhasil

Pada tabel tersebut terlihat bahwa *file* citra dengan ukuran 8 bit tidak berhasil

untuk disisipkan pesan karena tidak mengandung komponen RGB. Sedangkan untuk *file* citra 24 dan 32 bit bisa digunakan sebagai *Cover Image* untuk menyisipkan pesan karena mengandung komponen RGB didalamnya.



**Gambar 3.13:** *File* Citra 8 Bit



**Gambar 3.14:** *File* Citra 24 Bit



**Gambar 3.15:** File Citra 32 Bit

### 3.4.2 Pengujian Berdasarkan *Imperceptible*

Pengujian ini menguji kualitas citra *digital*, apakah citra *digital* akan mengalami perubahan yang mencurigakan atau tidak secara visual. Pengujian ini dikatakan berhasil apabila kualitas dari *Stego Image* yang dihasilkan tidak berbeda jika dibandingkan dengan berkas aslinya atau *Cover Image*.



**Gambar 3.16:** Perbandingan File Citra - lena.bmp



**Gambar 3.17:** Perbandingan *File Citra - amelia.bmp*

Pada gambar 3.16 dan 3.17 menunjukkan jika dibandingkan secara visual maka tidak tampak perubahan yang terjadi dari *Cover Image* ke *Stego Image*. Tidak tampak perubahan warna ataupun bentuk dari *Cover Image*.

### 3.4.3 Pengujian Berdasarkan *Fidelity*

Pengujian ini berdasarkan proses penyembunyiaan pesan atau *Encoding*. Pada proses penyembunyian pesan dapat berhasil apabila ukuran pesan yang akan disembunyikan sesuai dengan kapasitas *file* citra. Apabila ukuran pesan melebihi kapasitas maksimal dari *file* citra, maka program tidak akan melanjutkan proses *Encoding*. Kapasitas pesan dipengaruhi oleh ukuran *pixel* dari citra *digital*. Semakin besar ukuran *pixel* dari citra *digital*, maka semakin besar pula kapasitas pesan yang bisa disembunyikan.

**Tabel 3.2:** Karakter Maksimal Pesan pada *File* Citra

Citra	Pixel	Size (KB)	Kapasitas Karakter Maksimal Pesan
lena_gray	512 x 512	257	32704
lena	128 x 128	48	6096
amelia	1576 x 2364	2700	1396237

**Tabel 3.3:** Hasil Proses *Encoding*

Citra	Pesan (karakter)	Pixel	Size (KB)	Waktu (detik)	Hasil
lena_gray	114	512 x 512	257		Gagal
	786	512 x 512	257		Gagal
	4415	512 x 512	257		Gagal
lena	114	128 x 128	48	1	Berhasil
	786	128 x 128	48	2.63	Berhasil
	4415	128 x 128	48	20.56	Berhasil
	> 6096	128 x 128	48		Gagal
amelia	114	1576 x 2364	2700	1.6	Berhasil
	786	1576 x 2364	2700	2.82	Berhasil
	8832	1576 x 2364	2700	67.46	Berhasil
	> 1396237	1576 x 2364	2700		Gagal

Pada tabel di atas terlihat data *file* citra yang dimasukkan adalah 24 bit dan 32 bit. Setiap *file* citra dimasukkan pesan atau *Hiddentext* dengan beragam panjang karakter. Panjang dari karakter pesan berpengaruh terhadap kecepatan dalam proses *Encoding*. Semakin panjang karakter yang dimasukkan, maka semakin lama juga waktu yg dibutuhkan. Untuk *file* citra *lena<sub>gray</sub>* tidak bisa dilakukan proses *Encoding*, karena *file* tersebut adalah *file* 8 bit. Pada *file* citra lena dan amelia, berhasil dilakukan proses *Encoding*, kecuali ketika panjang karakter pesan melebihi kapasitas maksimal dari *file* citra tersebut.

### 3.4.4 Pengujian Berdasarkan *Recovery*

Pengujian ini berdasarkan proses ekstraksi pesan atau *Decoding*. Untuk membuktikan apakah program steganografi ini berhasil, maka harus dapat dibuktikan bahwa pesan di dalam *Stego Image* dapat diambil kembali. Jika pengujian dilakukan dengan benar, maka *Hiddentext* dapat ditampilkan sesuai dengan yang dimasukkan.

Pada penelitian ini, pesan yang dihasilkan dari proses *Decoding* tidak menghasilkan karakter-karakter aneh yang tidak dapat dibaca. Hasil dari pengujian ekstraksi pesan dapat dilihat pada tabel 3.2

**Tabel 3.4:** Hasil Proses *Decoding*

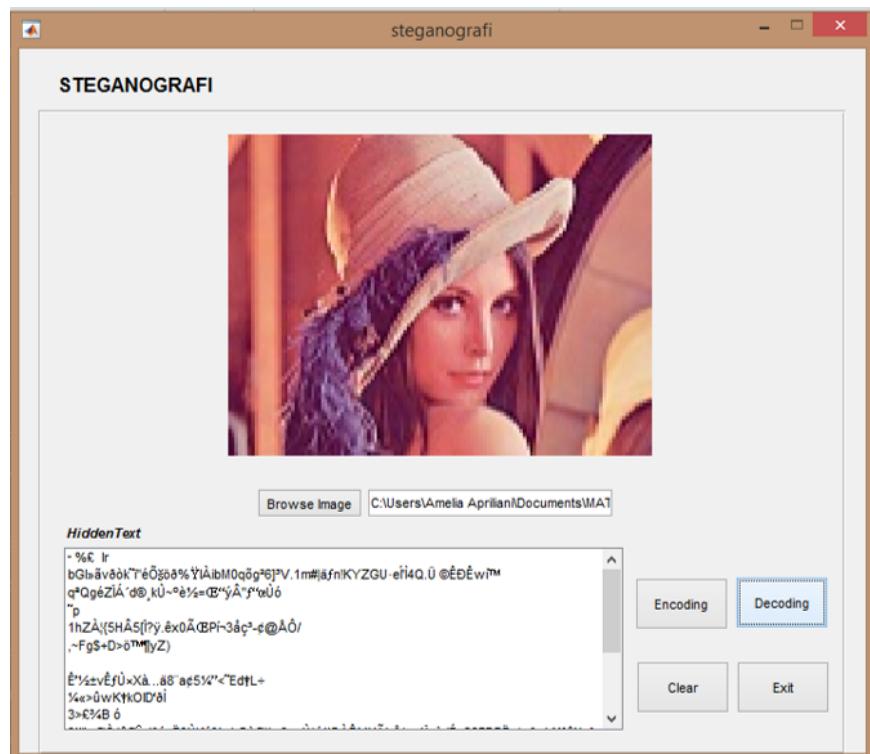
Citra	Pesan (karakter)	Pixel	Size (KB)	Waktu (detik)	Hasil
lena_gray	114				Gagal
	786				Gagal
	4415				Gagal
lena	114	128 x 128	48	1	Berhasil
	786	128 x 128	48	3.4	Berhasil
	4415	128 x 128	48	36	Berhasil
amelia	114	1576 x 2364	2700	1.3	Berhasil
	786	1576 x 2364	2700	3.76	Berhasil
	8832	1576 x 2364	2700	144.26	Berhasil

Pada proses *Decoding*, besar *pixel* dan ukuran pada *Stego Image*, tidak mengalami perubahan. Hal ini tidak akan memberikan kecurigaan terhadap *file* citra. Waktu yang dibutuhkan dalam proses *Decoding* juga berbanding lurus dengan panjang karakter pesan. Semakin banyak karakter pesan atau *Hiddentext* yang dimasukkan, maka semakin lama waktu yang dibutuhkan. Pada proses *Decoding* terhadap kedua *file* citra hasilnya adalah berhasil. Karena pesan yang ada pada *file* citra tersebut bisa ditampilkan kembali.

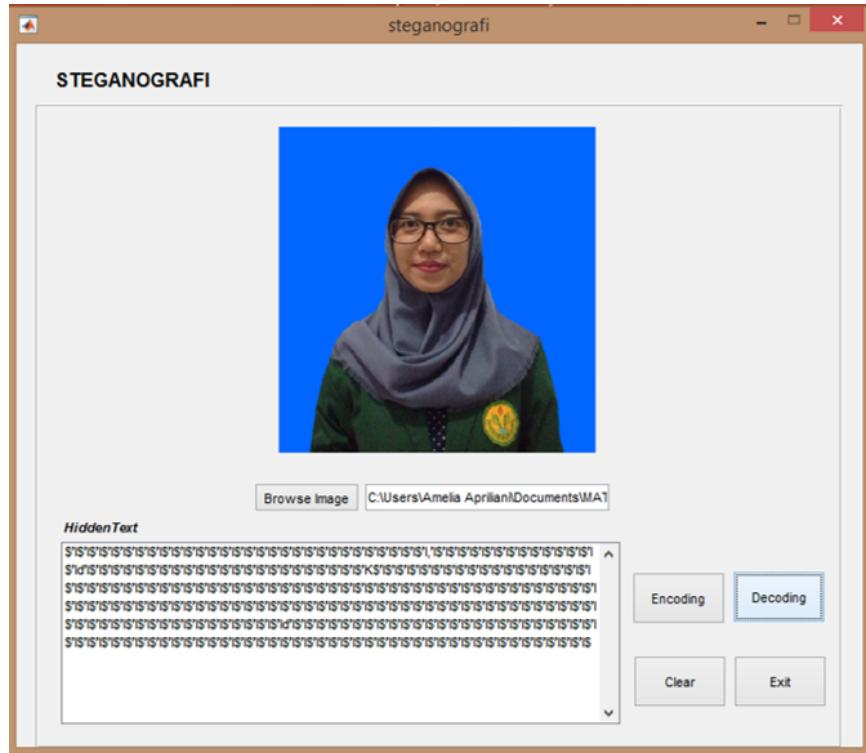
Pada pengujian ini juga dilakukan teknik *cropping* pada beberapa *Stego Image*. Setelah dilakukan *cropping*, *Stego Image* tersebut tidak berhasil untuk menampilkan pesan asli yang ada di dalamnya, hanya karakter-karakter aneh yang ditampilkan.

**Tabel 3.5:** Hasil Proses *Cropping*

Citra	Pixel Sebelum	Pixel Sesudah	Size (KB)	Hasil
lena	128 x 128	128 x 97	36.4	Gagal
amelia	1576 x 2364	480 x 494	694	Gagal



**Gambar 3.18:** File Citra lena.bmp Hasil *Cropping*



**Gambar 3.19:** File Citra amelia.bmp Hasil *Cropping*

Teknik *cropping* dapat menyebabkan *pixel-pixel* pada file citra menjadi hilang. Sehingga pesan yang sudah disisipkan di dalam file citra tersebut tidak dapat ditampilkan kembali.

### 3.4.5 Kesimpulan Pengujian

Dari pengujian di atas dapat disimpulkan bahwa program steganografi yang dibuat ini menghasilkan hasil yang cukup baik untuk setiap penyembunyian pesan ke dalam file citra. Pemilihan *Cover Image* yang akan digunakan dan panjang karakter pesan yang akan dimasukkan sangat berpengaruh, karena semakin besar ukuran file citra yang digunakan dan semakin sedikit karakter yang disisipkan pada file citra maka semakin sedikit perubahan yang terjadi setelah proses penyisipan pada file citra atau kualitas sebelum penyisipan dan setelah penyisipan tidak berpengaruh banyak

pada perubahan kualitas citra sebelumnya.

## **BAB IV**

### **KESIMPULAN DAN SARAN**

#### **4.1 Kesimpulan**

Berdasarkan hasil implementasi dan pengujian program ini, didapat kesimpulan sebagai berikut:

1. Format *file* citra yang hanya bisa digunakan adalah format \*.bmp 24 bit dan 32 bit.
2. Pesan dalam format teks berhasil untuk disisipkan ke dalam *file* citra dengan metode LSB menggunakan *Software* MATLAB R2016b.
3. *File* citra yang digunakan baik sebelum dan sesudah diproses tidak berubah. Perubahan yang terjadi tidak dapat dilihat secara visual.
4. Pesan dalam *Stego Image* dapat ditampilkan kembali dan sesuai dengan pesan awal.

#### **4.2 Saran**

Adapun saran-saran penulis untuk penelitian selanjutnya adalah:

1. Program ini hanya dapat menyisipkan pesan berupa teks, diharapkan untuk ke-depannya dikembangkan sehingga dapat menyisipkan *file*, gambar atau audio.
2. Media yang digunakan berupa *file* citra, diharapkan dapat menggunakan media lain seperti audio.

3. Program steganografi ini masih dikembangkan dengan MATLAB untuk perangkat komputer, diharapkan dapat dikembangkan lebih lanjut untuk perangkat *mobile*.

## **DAFTAR PUSTAKA**

- [1] Adiria. 2010. Analisis dan Perancangan Aplikasi Steganografi pada Citra Digital dengan Menggunakan Metode LSB (Least Significant Bit) [skripsi]. Jakarta: Fakultas Sains dan Teknologi. Universitas Islam Negeri Syarif Hidayatullah Jakarta.
- [2] Arymurthy AM, Setiawan S. 1992. Pengantar Pengolahan Citra. Jakarta. PT Elex Media Komputindo.
- [3] ASCII Table. 2010. ASCII Table and Description. ASCII Table [Online]. Terse-  
dia: <https://www.asciitable.com>. [17 April 2018].
- [4] Bunyamin H, dan Andrian. 2009. Aplikasi Steganography pada File dengan Menggunakan Teknik Low Bit Encoding dan Least Significant Bit. Jurnal In-  
formatika UKM. 5(2): 107-117.
- [5] Elgarab EEA. 2013. Comparison of LSB Steganography in BMP and JPEG Im-  
ages. International Journal of Soft Computing and Engineering (IJSCE). ISSN:  
2231-2307 3(5).
- [6] Elgarab EEA, Mohammed FA. 2013. JPEG versus GIF Images in forms of  
LSB Steganography. International Journal of Computer Science and Network  
(IJCSN). ISSN: 2277-5420 2(6).
- [7] Gautam P, Sharma D. 2015. A Survey on Digital Image Steganography Tech-  
niques. International Journal of Electronics, Electrical and Computational System  
(IJEECS). ISSN: 2348-117X 4(11).
- [8] Hermawati FA. 2013. Pengolahan Citra Digital. Yogyakarta. ANDI.

- [9] Irfan. 2013. Penyembunyian Informasi (steganography) Gambar Menggunakan Metode LSB (Least Significant Bit). *Rekayasa Teknologi*. 5(1).
- [10] Joshi K, Yadav R. 2015. A New LSB-S Image Steganography Method Blend with Cryptography for Secret Communication. *Third International Conference on Image Infomation Processing*.
- [11] Kessler GC. 2001. *Steganography Hiding Data Within Data*.
- [12] Kavitha, Kadam K, Koshti A, Dunghav P. 2012. Steganography Using Least Significant Bit Algorithm. *International Journal of Engineering Research and Applications (IJERA)*. 2(3): 338-341.
- [13] Munir R. 2004. *Pengolahan Citra Digital*. Bandung. Informatika.
- [14] Munir, R. 2006. *Kriptografi*. Bandung. Informatika.
- [15] Pakereng MAI, Beeh YR, Endrawan S. 2010. Perbandingan Steganografi Metode Spread Spectrum dan Least Significant Bit (LSB) Antara Waktu Proses dan Ukuran File Gambar. *Jurnal Infrmatika*. 6(1).
- [16] Pavani M, Naganjaneyulu S, Nagaraju C. 2013. A Survey on LSB Based Steganography Methods. *International Journal Of Engineering And Computer Science*. 2(8): 2464-2467.
- [17] Prasetyo FP. 2010. Steganografi Menggunakan Metode LSB dengan Software Matlab [skripsi]. Jakarta: Fakultas Sains dan Teknologi. Universitas Islam Negeri Syarif Hidayatullah Jakarta.
- [18] Prayudi Y, Kuncoro PS. 2005. Implementasi Steganografi Menggunakan Teknik Adaptive Minimum Error Least Significant Bit Replacement (AMELSBR). Seminar Nasional Aplikasi Teknologi Informasi.

- [19] Rakhmat B, Fairuzabadi M. 2010. Steganografi Menggunakan Metode Least Significant Bit dengan Kombinasi Algoritma Kriptografi Vigenere dan RC4. Jurnal Dinamika Informatika. 5(2).
- [20] Setiana, Mahmudy WF. 2006. Steganografi Pada File Citra Bitmap 24 Bit Untuk Pengamanan Data Menggunakan Metode Least Significant Bit (LSB) Insertion. Kurstor. 2(2): 38-44.
- [21] Table ASCII. (n.d.). Retrieved from <http://theasciicode.com.ar/>
- [22] Wikipedia. (n.d.). Retrieved from <https://id.wikipedia.org/wiki/Steganografi>

## LAMPIRAN A

### *Source Code*

```
char_max = (row -1)*(column);
char_max = round((char_max*3)/8);

%Cek Kondisi Lokasi
lokasi = get(handles.kolom_lokasi, 'String')
if isempty(lokasi)
msgbox('Gambar belum dimasukkan','Peringatan','warn');
return;
end

%Cek Kondisi Kolom Hiddentext
hiddentext = get(handles.edit_pesanan,'String');
if isempty(hiddentext)
msgbox('Pesanan belum dimasukkan','Peringatan','warn');
return;
end

%Menghitung Panjang Pesan
row_max = row;
column_max = column;
hiddentext_length = length(hiddentext)
if hiddentext_length < char_max
hiddentext_biner = strcat(reshape(dec2bin(double(hiddentext),8)',1)
```

```
hiddentext_save = hiddentext_biner;

hiddentext_asli = char(bin2dec(reshape(hiddentext_biner,8,[]).'')).

else

msgbox('Maaf,pesan terlalu panjang','peringatan','warn');

return;

end

%Encoding

hiddentext_length = hiddentext_length*8;

for i = 1:row_max-1

for j = 1:column_max

if hiddentext_length ~= 0

image_biner_red = dec2bin(image_red(i,j),8);

image_biner_red(1,8) = hiddentext_biner(1,1);

image_red(i,j) = bin2dec(image_biner_red);

hiddentext_biner(1:1) = [];

hiddentext_length = length(hiddentext_biner);

end

if hiddentext_length ~= 0

image_biner_green = dec2bin(image_green(i,j),8);

image_biner_green(1,8) = hiddentext_biner(1,1);

image_green(i,j) = bin2dec(image_biner_green);
```

```
hiddentext_biner(1:1) = [];

hiddentext_length = length(hiddentext_biner);
end

if hiddentext_length ~= 0
    image_biner_blue = dec2bin(image_blue(i,j),8);
    image_biner_blue(1,8) = hiddentext_biner(1,1);
    image_blue(i,j) = bin2dec(image_biner_blue);

    hiddentext_biner(1:1) = [];
    hiddentext_length = length(hiddentext_biner);
    end
end
end

stego_image(:,:,1) = uint8(image_red);
stego_image(:,:,2) = uint8(image_green);
stego_image(:,:,3) = uint8(image_blue);

[nama_file, direktori] = uiputfile('*.bmp','Simpan Stego Image');
if direktori == 0
    return;
end
nama = fullfile(direktori, nama_file);
imwrite(stego_image, nama, 'bmp');

msgbox('Stego Image telah berhasil dibuat','pemberitahuan');
```

```
%Decoding

hiddentext = '';
var_null = 1;
for i = 1:row_max-1
for j = 1:column_max
biner_length = length(hiddentext);
if biner_length < hiddentext_length && var_null<=8
image_biner_red = dec2bin(image_red(i,j),8);
hiddentext_red = image_biner_red(1,8);
hiddentext = strcat(hiddentext, hiddentext_red);
if hiddentext_red == 0
var_null = var_null + 1;
else
var_null = 1;
end
else
hiddentext_asli = char(bin2dec(reshape(hiddentext,8,[]).'')).';
set(handles.edit_pesan,'String',hiddentext_asli);
return;
end

biner_length = length(hiddentext);
if biner_length < hiddentext_length && var_null<=8
image_biner_green = dec2bin(image_green(i,j),8);
```

```
hiddentext_green = image_biner_green(1,8);

hiddentext = strcat(hiddentext, hiddentext_green);

if hiddentext_green == 0

var_null = var_null+1;

else

var_null = 1;

end

else

hiddentext_asli = char(bin2dec(reshape(hiddentext,8,[]).'')).';

set(handles.edit_pesan,'String',hiddentext_asli);

return;

end


biner_length = length(hiddentext);

if biner_length < hiddentext_length && var_null<=8

image_biner_blue = dec2bin(image_blue(i,j),8);

hiddentext_blue = image_biner_blue(1,8);

hiddentext = strcat(hiddentext, hiddentext_blue);

if hiddentext_blue == 0

var_null = var_null+1;

else

var_null = 1;

end

else

hiddentext_asli = char(bin2dec(reshape(hiddentext,8,[]).'')).';

set(handles.edit_pesan,'String',hiddentext_asli);
```

```
return;
```

```
end
```

```
end
```

```
end
```