

IMPLEMENTASI TEKNIK STEGANOGRAFI DENGAN METODE LSB PADA CITRA DIGITAL

Putri Alatas / 11104284

*Tugas Akhir. Jurusan Sistem Informasi, Fakultas Ilmu Komputer & Teknologi
Informasi, Universitas Gunadarma, 2009*

Abstrak

Berbagai macam teknik digunakan untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak, salah satunya adalah teknik steganografi. Steganografi sebagai suatu seni menyembunyikan pesan ke dalam pesan lainnya yang telah ada sejak sebelum masehi dan kini seiring dengan kemajuan teknologi jaringan serta perkembangan dari teknologi digital, steganografi banyak dimanfaatkan untuk mengirim pesan melalui jaringan Internet tanpa diketahui orang lain dengan menggunakan media digital berupa file citra.

Kata Kunci : Steganografi, LSB, Citra Digital

(xiii + 52 + lampiran)

Daftar Pustaka (2000 – 2007)

PENDAHULUAN

Saat ini internet sudah berkembang menjadi salah satu media yang paling populer di dunia. Karena fasilitas dan kemudahan yang dimiliki oleh internet maka internet untuk saat ini sudah menjadi barang yang tidak asing lagi. Sayangnya dengan berkembangnya internet dan aplikasi menggunakan internet semakin berkembang pula kejahatan sistem informasi. Dengan berbagai teknik banyak yang mencoba untuk mengakses informasi yang bukan haknya. Maka dari itu sejalan dengan berkembangnya media internet ini harus juga dibarengi dengan perkembangan pengamanan sistem informasi.

Berbagai macam teknik digunakan untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak, salah satunya adalah teknik steganografi. Steganografi sebagai suatu seni menyembunyikan pesan ke dalam pesan lainnya yang telah ada sejak sebelum masehi dan kini seiring dengan kemajuan teknologi jaringan serta perkembangan dari teknologi digital, steganografi banyak dimanfaatkan untuk mengirim pesan melalui jaringan Internet tanpa diketahui orang lain dengan menggunakan media digital berupa file citra.

Atas dasar uraian diatas, maka pada penulisan skripsi ini akan membahas mengenai bagaimana mengamankan suatu pesan dengan menyisipkan kedalam pesan lainnya yaitu file citra dengan menggunakan algoritma LSB (*Least Significant Bit*) pada suatu aplikasi steganografi.

1. STEGANOGRAFI

Steganografi berasal dari bahasa Yunani yaitu *Steganós* yang berarti menyembunyikan dan *Graptos* yang artinya tulisan sehingga secara keseluruhan artinya adalah tulisan yang disembunyikan. Secara umum steganografi merupakan seni atau ilmu yang digunakan untuk menyembunyikan pesan rahasia dengan segala cara sehingga selain orang yang dituju, orang lain tidak akan menyadari keberadaan dari pesan rahasia tersebut.

Steganografi sudah digunakan sejak dahulu kala sekitar 2500 tahun yang lalu untuk kepentingan politik, militer, diplomatik, serta untuk kepentingan pribadi sebagai alat komunikasi.

Akhir-akhir ini kata steganografi menjadi sering disebut di masyarakat bersama-sama dengan kata kriptografi setelah pemboman gedung WTC di AS, telah disebutkan oleh Pejabat Pemerintah dan Para Ahli dari Pemerintahan Amerika Serikat "yang tidak disebut namanya bahwa" bahwa Para Teroris menyembunyikan peta-peta dan foto-foto target dan juga perintah untuk aktivitas teroris di ruang chat sport, bulletin boards porno dan web site lainnya. Walaupun demikian sebenarnya belum ada bukti nyata dari pernyataan-pernyataan tersebut diatas. Novel Da Vinci Code pun turut mempopulerkan steganografi dan kriptografi.

Sebuah contoh klasik untuk menjelaskan steganography adalah seperti persoalan tahanan penjara. Diilustrasikan Alisa dan Bobi berada sama-sama di penjara. Pada suatu saat keduanya akan menyusun rencana untuk kabur dari penjara. Alisa harus mengirimkan pesan tersebut melalui kurir (Fred). Jika Alisa mengenkripsi pesan rahasianya, kurir tentu akan curiga. Oleh karena itu Alice dan Bobi perlu menggunakan suatu teknik sehingga kurir tidak dapat mendeteksi adanya pesan rahasia. Teknik tersebut dikenal dengan sebutan steganografi.



Gambar 1.1 Ilustrasi tahanan penjara

Pada gambar 2.1 diatas satu-satunya cara untuk berkomunikasi antara Alisa dan Bobi adalah melalui seorang sipir penjara yaitu Fred. Alisa menulis surat pada selembur kertas, lalu surat tersebut diserahkan kepada Fred. Fred tentu saja dapat membaca isi

surat tersebut sebelum disampaikan kepada Bobi. Hal yang sama juga dapat dilakukan Bobi bila ia hendak membalas pesan dari Alice.

Jika Alisa ingin menulis pesan rahasia kepada Bobi mengenai rencana waktu pelarian mereka dari penjara. Pesan tersebut bunyinya adalah “Lari jam satu”, bagaimana cara Alisa mengirim pesan tersebut tanpa diketahui oleh Fred, ada dua cara yang dapat digunakan, solusi pertama yaitu kriptografi dimana Alisa harus mengenkripsi pesan tersebut menjadi ciphertext. Fred yang menyampaikan pesan tersebut pasti curiga dan menduga ciphertext tersebut merupakan pesan rahasia karena tulisannya yang tidak lazim sehingga kemungkinan Fred akan menahan surat tersebut.

- Alternatif 1: mengenkripsinya menjadi ciphertext

xjT#9uvmY!rc\$

Fred pasti curiga!

Solusi yang kedua yaitu Alisa menyembunyikan pesan rahasia tersebut di dalam tulisan lain dengan cara menyisipkan setiap huruf pesan rahasia pada awal setiap kata seperti pada contoh alternatif 2 di bawah, Fred yang menyampaikan pesan tentunya tidak akan curiga dan tidak menyadari keberadaan pesan rahasia di dalamnya.

- Alternatif 2: menyembunyikannya di dalam pesan lain

Lupakan asal rumor itu jangan ambil manfaatnya setelah aku tutup usia

Fred tidak akan curiga!

Information hiding dengan steganografi!

1.1 Pengertian Steganografi

Steganografi merupakan suatu ilmu atau seni dalam menyembunyikan informasi dengan memasukkan informasi tersebut ke dalam pesan lain. Dengan demikian keberadaan informasi tersebut tidak diketahui oleh orang lain.

Ilustrasi Steganography Alisa dan Bob tersebut sudah sangat usang jika digunakan dalam dunia modern seperti sekarang ini. Teknik Steganography yang digunakan dalam

dunia modern sekarang ini sudah amat beragam. Beragam mulai dari algoritma yang digunakannya sampai pada media yang digunakannya.

Beberapa contoh media penyisipan pesan rahasia yang digunakan dalam teknik Steganography antara lain adalah :

1. Teks

Dalam algoritma Steganography yang menggunakan teks sebagai media penyisipannya biasanya digunakan teknik NLP sehingga teks yang telah disisipi pesan rahasia tidak akan mencurigakan untuk orang yang melihatnya.

2. Audio

Format ini pun sering dipilih karena biasanya berkas dengan format ini berukuran relatif besar. Sehingga dapat menampung pesan rahasia dalam jumlah yang besar pula.

3. Citra

Format pun paling sering digunakan, karena format ini merupakan salah satu format file yang sering dipertukarkan dalam dunia internet. Alasan lainnya adalah banyaknya tersedia algoritma Steganography untuk media penampung yang berupa citra.

4. Video

Format ini memang merupakan format dengan ukuran file yang relatif sangat besar namun jarang digunakan karena ukurannya yang terlalu besar sehingga mengurangi kepraktisannya dan juga kurangnya algoritma yang mendukung format ini.

Tujuan dari steganografi adalah menyembunyikan keberadaan pesan dan dapat dianggap sebagai pelengkap dari kriptografi yang bertujuan untuk menyembunyikan isi pesan. Oleh karena itu, berbeda dengan kriptografi, dalam steganografi pesan disembunyikan sedemikian rupa sehingga pihak lain tidak dapat mengetahui adanya pesan rahasia. Pesan rahasia tidak diubah menjadi karakter 'aneh' seperti halnya

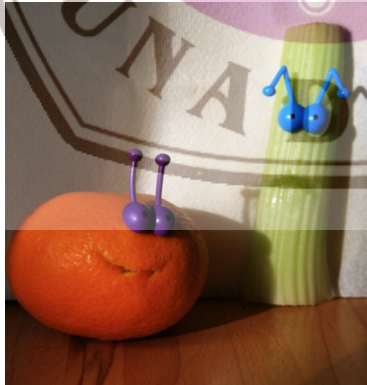

kriptografi. Pesan tersebut hanya disembunyikan ke dalam suatu media berupa gambar, teks, musik, atau media digital lainnya dan terlihat seperti pesan biasa.

1.1.1 Konsep dan Terminologi

Terdapat beberapa istilah yang berkaitan dengan steganografi:

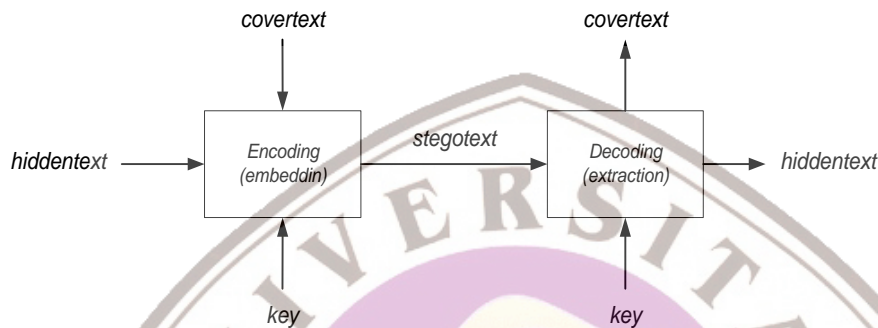
1. *Hiddentext* atau *embedded message*: pesan yang disembunyikan.
2. *Coverttext* atau *cover-object*: pesan yang digunakan untuk menyembunyikan *embedded message*.
3. *Stegotext* atau *stego-object*: pesan yang sudah berisi *embedded message*

Di dalam Steganografi citra digital ini, *hidden text* atau *embedded message* yang dimaksudkan adalah adalah teks yang akan disisipkan ke dalam *coverttext* atau *cover-object* yaitu file citra digital yang digunakan sebagai media penampung pesan yang akan disisipkan. Dari hasil *encoding* atau *embedding* pesan kedalam file citra akan dihasilkan *stegotext* atau *stego-object* yang merupakan file citra yang berisikan pesan *embedding*. Pada gambar 2.2 di bawah ini merupakan contoh dari *hiddentext*, *coverttext* dan *stegotext*.

Semua file - file berharga perusahaan kita tersimpan di ruang bawah tanah rumah kita, file tersebut tersimpan pada sebuah lemari besi yang tersembunyi dibalik lemari tua dengan serial kunci 16A05m11p.		
Hiddentext	Coverttext / Cover-object	Stegotext / Stego-object

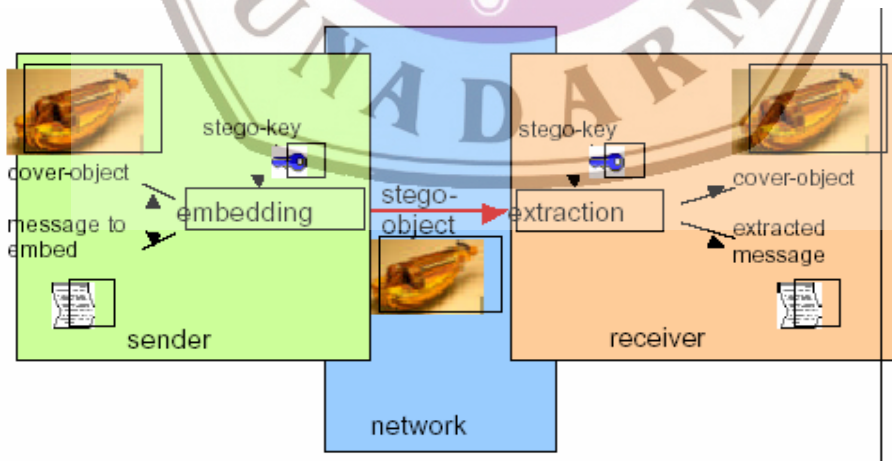
Gambar 1.2 Contoh Hiddentext, Coverttext dan Stegotext

Penyisipan pesan ke dalam media *coverttext* dinamakan encoding, sedangkan ekstraksi pesan dari *stegotext* dinamakan decoding. Kedua proses ini memerlukan kunci rahasia (*stegokey*) agar hanya pihak yang berhak saja yang dapat melakukan penyisipan dan ekstraksi pesan, seperti yang terlihat pada gambar 2.3 di bawah.



Gambar 1.3 Diagram penyisipan dan ekstraksi pesan

Applikasi steganografi pada file citra ini dapat memudahkan kita dalam menjaga keamanan data saat bertukar informasi rahasia atau mengirimkan pesan melalui internet karena hanya pihak yang memiliki stego-key yang dapat memperoleh pesan yang terdapat pada file citra digital tersebut, seperti yang terlihat pada gambar 2.4 di bawah ini:



Gambar 1.4 Proses pengiriman stego-object melalui internet

1.1.2 Kriteria Steganography

Penilaian sebuah algoritma steganography yang baik dapat di nilai dari beberapa faktor yaitu :

1. *Imperectibility*. Keberadaan pesan rahasia dalam media penampung tidak dapat dideteksi oleh inderawi. Misalnya, jika *coverttext* berupa citra, maka penyisipan pesan membuat citra *stegotext* sukar dibedakan oleh mata dengan *coverttext*-nya. Jika *coverttext* berupa audio (misalnya berkas file *mp3*, *wav*, *midi* dan sebagainya), maka indera telinga tidak dapat mendeteksi perubahan pada file *stegotext*-nya.
2. *Fidelity*. Mutu media penampung tidak berubah banyak akibat penyisipan. Perubahan itu tidak dapat dipersepsi oleh inderawi. Misalnya, jika *coverttext* berupa citra, maka penyisipan pesan dapat membuat citra *stegotext* sukar dibedakan oleh mata dengan citra *coverttext*-nya. Jika *coverttext* berupa audio (misalnya berkas file *mp3*, *wav*, *midi* dan sebagainya), maka audio *stegotext* tidak rusak dan indera telinga tidak dapat mendeteksi perubahan pada file *stegotext*-nya.
3. *Recovery*. Pesan yang disembunyikan harus dapat diungkapkan kembali (*reveal*). Karena tujuan steganography adalah *data hiding*, maka sewaktu-waktu pesan rahasia di dalam *stegotext* harus dapt diambil kembali untuk digunakan lebih lanjut.

1.1.3 Teknik Penyembunyian Data

Teknik penyembunyian data ke dalam *coverttext* dapat dilakukan dalam dua macam domain:

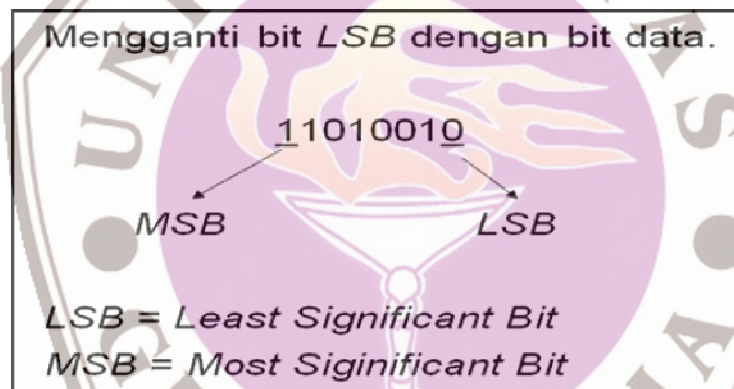
1. Domain spasial/waktu (*spatial/time domain*).

Teknik ini memodifikasi langsung nilai byte dari *coverttext* (nilai byte dapat merepresentasikan intensitas/warna pixel atau amplitude). Metode yang tergolong ke dalam teknik ranah spasial adalah metode LSB.

2. Domain transform (*frequency transform domain*).

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

Pada penelitian ini, akan digunakan metode LSB (*Least Significant Bit*) yang merupakan teknik penyembunyian data yang bekerja pada domain spatial atau waktu. Untuk menjelaskan teknik penyembunyian LSB yang dipakai ini kita menggunakan citra digital sebagai *covertext*. Setiap *pixel* yang ada di dalam file citra berukuran 1 sampai 3 *byte*. Pada susunan bit dalam setiap *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*)



Gambar 1.5 Contoh LSB dan MSB

Dari contoh *byte* 11010010 pada gambar 2.5 diatas bit 1 pertama yang (di garis bawah) adalah bit *MSB* dan bit 0 terakhir yang digaris bawah adalah bit *LSB*. Bit yang cocok untuk diganti dengan bit pesan adalah bit *LSB*, karena modifikasi hanya mengubah nilai *byte* tersebut satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Misalkan *byte* tersebut di dalam gambar memberikan persepsi warna merah, maka perubahan satu bit *LSB* hanya mengubah persepsi merah tidak terlalu berarti karena mata manusia tidak dapat membedakan perubahan sekecil ini.

Sebagai ilustrasi misalkan *cover-object* adalah citra sekumpulan citra berwarna merah seperti yang terlihat pada contoh di bawah ini:

00110011 10100010 11100010 01101111

Dan misalkan pesan rahasia (yang telah dikonversi ke system biner) *embedded message* adalah 0110. Setiap bit dari *watermark* menggantikan posisi LSB dari segmen *pixel-pixel* citra menjadi :

00110010 10100011 11100011 01101110

Dari hasil penanaman atau *embedding* kedalam sekumpulan *pixel* citra berwarna merah tadi diperoleh kembali sekumpulan *pixel* berwarna merah yang telah berubah sedikit pada posisi bit terendah atau LSB dari *pixel* tersebut. Demikianlah contoh sederhana bagaimana algoritma LSB bekerja untuk menggantikan nilai bit-bit terendah dari setiap *pixel* untuk disisipkan atau digantikan oleh bit baru yang mengandung pesan.

Untuk dapat membuat *hiddentext* tidak dapat dilacak, bit-bit pesan tidak mengganti *byte-byte* yang berurutan, namun dipilih susunan *byte* secara acak. Misalnya jika terdapat 50 *byte* dan 6 bit data yang akan disembunyikan, maka *byte* yang diganti bit *LSB*-nya dipilih secara acak, misalkan *byte* nomor 36, 5, 21, 10, 18, 49. Pembangkitan bilangan acak dilakukan dengan *pseudo-random-number-generator* (*PRNG*) yang berlaku sebagai kunci stegano.

Pada citra 8-bit yang berukuran 256 x 256 *pixel* terdapat 65536 *pixel*, setiap *pixel* berukuran 1 *byte* sehingga kita hanya dapat menyisipkan 1 bit pada setiap *pixel*. Pada citra 24-bit yang berukuran 256 x 256 *pixel*, satu *pixel* berukuran 3 *byte* (atau 1 *byte* untuk setiap komponen R, G, B), sehingga kita bisa menyisipkan pesan sebanyak $65536 \times 3 \text{ bit} = 196608 \text{ bit}$ atau $196608/8 = 24576 \text{ byte}$.

Pesan yang disembunyikan di dalam citra dapat diungkap kembali dengan mengekstraksinya. Posisi *byte* yang menyimpan bit pesan dapat diketahui dari bilangan acak yang dibangkitkan oleh *PRNG*. Jika kunci yang digunakan pada waktu ekstraksi sama dengan kunci pada waktu penyisipan, maka bilangan acak yang dibangkitkan juga sama. Dengan demikian, bit-bit data rahasia yang bertaburan di dalam citra dapat dikumpulkan kembali.

METODOLOGI

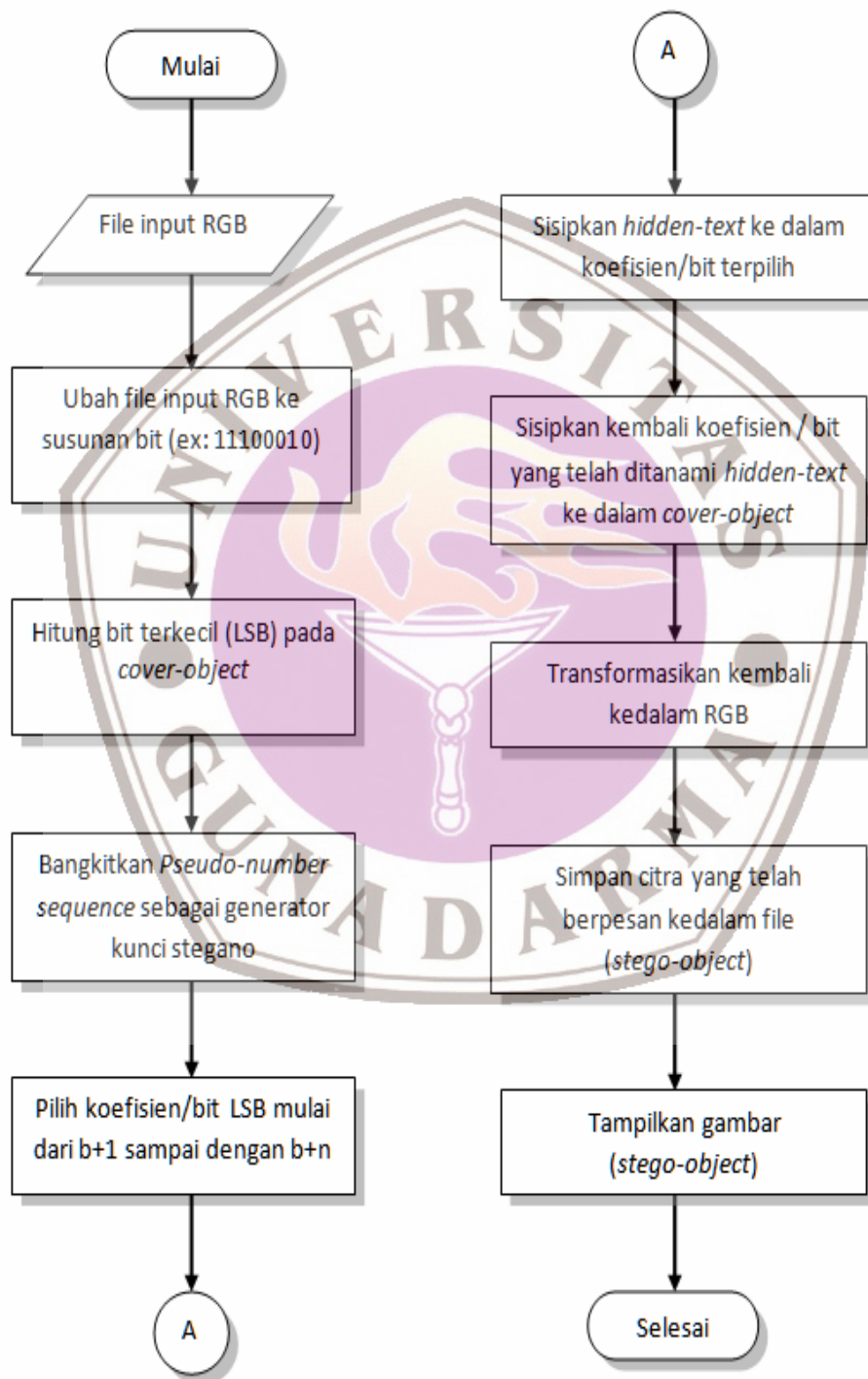
2.1 Rancangan Algoritma LSB pada citra digital

Secara garis besar jalannya aplikasi ini adalah terbagi dua proses utama yaitu hide message atau penyisipan pesan dan extract message atau pendekteksian kembali pesan yang tersembunyi.

Pada proses penyisipan pesan (embedding message) dimulai dengan memilih gambar yang akan dijadikan cover object untuk menyisipkan dan menyembunyikan pesan ke dalam gambar kemudian menentukan key file yang akan digunakan sebagai password dalam proses extract dan menuliskan isi pesan text yang akan disisipkan kedalam gambar. Sedangkan pada proses pendeteksian pesan (extraction message) dimulai dengan memilih file gambar atau covert object yang akan di extract dan memasukan key file, yang hasil'y ekstraksi pesannya dapat disimpan pada satu file tertentu yang dipilih.

Berikut merupakan digram alir atau flowchart yang akan menjelaskan proses embedding message yaitu bagaimana suatu file gambar dapat disisipkan pesan sehingga menghasilkan stego object atau encoder dan proses extraction message yaitu bagaimana mengekstrak pesan dari suatu file gambar stego object agar dapat terbaca kembali pesan yang dienkripsi sebelumnya.

Diagram alir proses embedding message



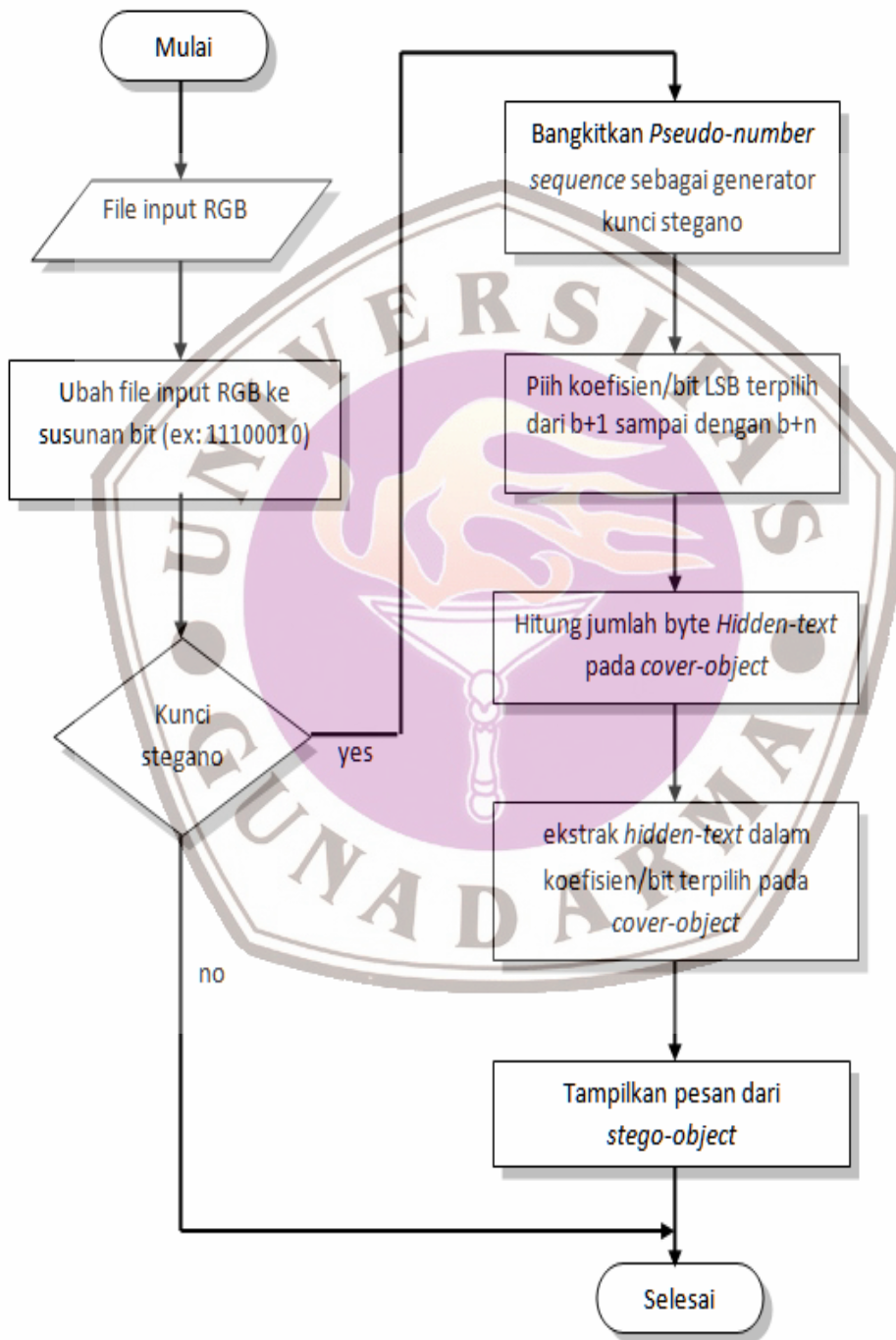
Gambar 3.1 Flowchart Embedding Message (encoder)

Pada gambar 3.1 diatas adalah flowchart proses embedding message kedalam file citra (cover-object) dimulai dengan membaca file citra ke RGB, Seperti kita ketahui untuk file bitmap 24 bit maka setiap pixel (titik) pada gambar tersebut terdiri dari susunan tiga warna merah, hijau dan biru (RGB) yang masing-masing disusun oleh bilangan 8 bit (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-object.

Setelah menentukan bit terkecil dari cover-object yang akan digunakan maka langkah selanjutnya yaitu membangkitkan pseudo-number sequence yang akan digunakan sebagai generator kunci stegano dengan menentukan key yang akan dipakai sebagai password untuk mengenkripsi pesan kedalam cover-object. Proses selanjutnya adalah memilih koefisien bit terpilih mulai dari $b+1$ sampai $b+n$ untuk disisipkan hidden-text kedalamnya,

Selanjutnya adalah setelah memilih koefisien atau bit-bit terpilih maka proses berikutnya adalah menyisipkan hidden-text ke dalamnya koefisien atau bit-bit tersebut sehingga akan dihasilkan koefisien atau bit-bit yang baru yang telah mengandung pesan, dan menyisipkannya kembali kedalam cover-object, yang kemudian koefisien tersebut selanjutnya akan di transformasikan kembali kedalam nilai RGB yang baru dan menyimpan citra yang telah berpesan ke dalam cover-object sehingga diperoleh atau dapat ditampilkan sebuah gambar baru yang telah disisipkan pesan atau stego-object.

Diagram alir proses extraction message



Gambar 3.2 Flowchart Extraction Message (decoder)

Pada gambar 3.2 diatas adalah flowchart proses extraction message dari stego-object menghasilkan hidden-text yang terdapat didalamnya atau untuk mengungkap kembali pesan yang disisipkan kedalam file citra, proses awalnya dimulai dengan membaca file citra ke RGB, dan mengubah file input RGB kedalam format biner. Kemudian langkah selanjutnya adalah memeriksa kunci stegano yang digunakan sebagai password saat mengenkripsi pesan, jika kunci stegano yang dimasukkan benar maka akan beralih ke proses selanjutnya yakni membangkitkan nilai PRNG atau pseudo number generator yang menyimpan bit-bit atau koefisien terpilih yang secara acak berada pada file citra atau stego-object.

Setelah diperoleh koefisien atau bit-bit yang terpilih yang mengandung pesan maka proses ekstraksi akan berjalan dan menghitung jumlah byte hidden-text pada cover-object. Setelah diperoleh byte yang tersembunyi pada cover-object maka proses berikutnya adalah mengekstrak kembali pesan yang tersembunyi (hidden-text) yang terdapat didalamnya sehingga pesan dapat ditampilkan kembali.

2.3 Peak Signal to Noise Ratio (PSNR)

Peak Signal to Noise Ratio (PSNR) adalah perbandingan antara nilai maksimum dari sinyal yang diukur dengan besarnya derau yang berpengaruh pada sinyal tersebut. PSNR biasanya diukur dalam satuan desibel. Pada tugas akhir kali ini, PSNR digunakan untuk mengetahui perbandingan kualitas citra sebelum dan sesudah disisipkan pesan.

Untuk menentukan PSNR, terlebih dahulu harus ditentukan nilai rata-rata kuadrat dari error (MSE - *Mean Square Error*). Perhitungan MSE adalah sebagai berikut :

$$MSE = \frac{1}{mn} \sum_i^m \sum_j^n ||I(i, j) - K(i, j)||^2$$

Dimana :

MSE = Nilai *Mean Square Error* dari citra tersebut

m = panjang citra tersebut (dalam piksel)

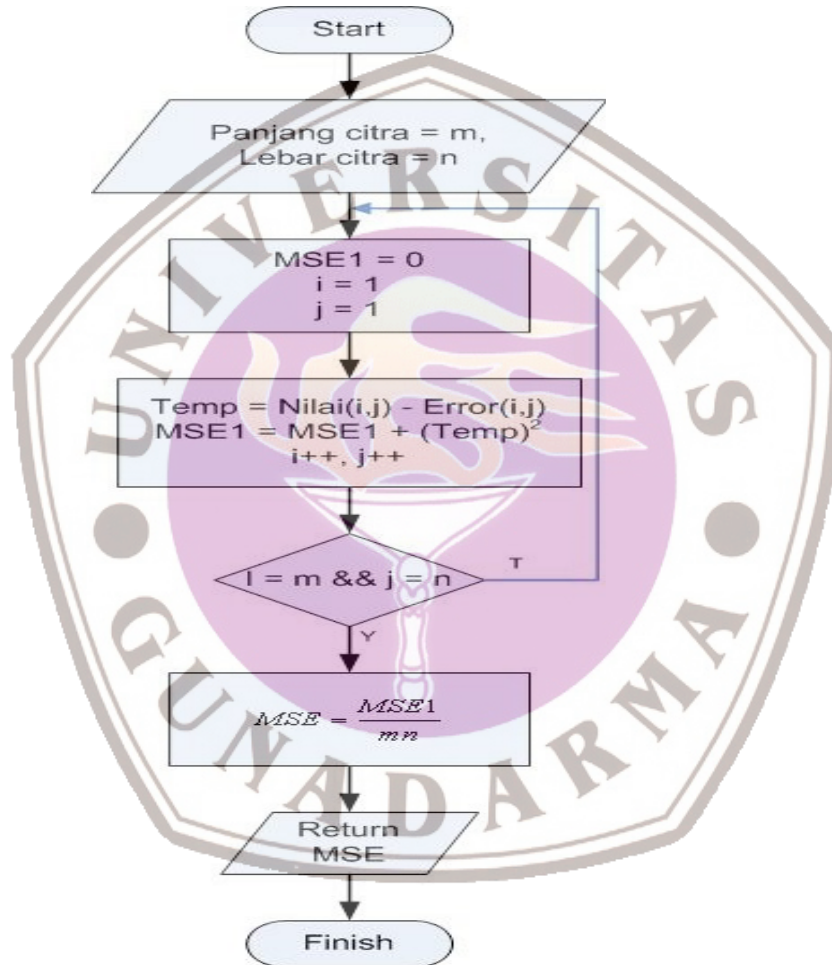
n = lebar citra tersebut (dalam piksel)

(i,j) = koordinat masing-masing piksel

I = nilai *bit* citra pada koordinat i,j

K = nilai derajat keabuan citra pada koordinat i,j

Dari rumus di atas, dapat dibuat diagram alir perhitungan MSE seperti ditunjukkan pada gambar 3.5 di bawah ini :



Gambar 2.5 Flowchart Perhitungan MSE

Sementara nilai PSNR dihitung dari kuadrat nilai maksimum sinyal dibagi dengan MSE. Apabila diinginkan PSNR dalam desibel, maka nilai PSNR akan menjadi sebagai berikut :

$$PSNR = 10 \cdot \log \left(\frac{MAX_I^2}{MSE} \right) = 20 \cdot \log \left(\frac{MAX_I}{\sqrt{MSE}} \right)$$

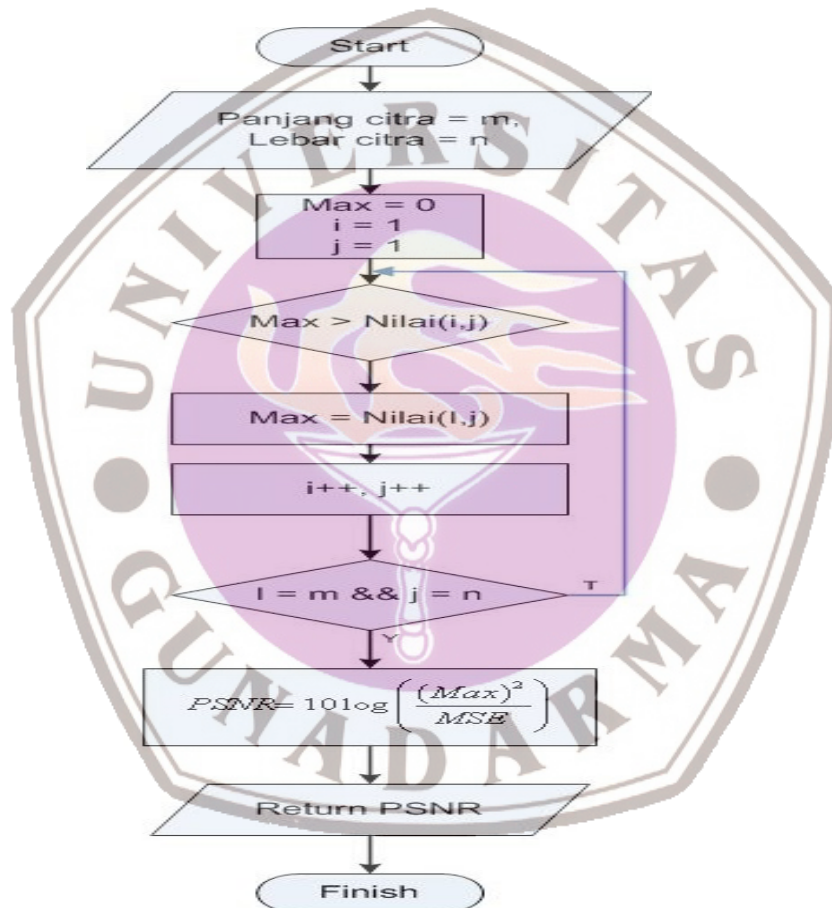
Dimana :

PSNR = nilai PSNR citra (dalam dB)

MAX_i = nilai maksimum piksel

MSE = nilai MSE

Dari rumus perhitungan PSNR di atas, dapat dibuat diagram alir untuk menghitung PSNR seperti ditunjukkan pada gambar 3.6 di bawah ini.



Gambar 2.6 Flowchart Perhitungan PSNR

Dari flowchart PSNR di atas, maka berikut ini adalah penerapan algoritma PSNR yang dipakai pada penulisan ini. Algoritma PSNR yang digunakan dibuat menggunakan bahasa pemrograman matlab mengingat matlab merupakan bahasa pemrograman sangat baik untuk mengolah file citra karena dilengkapi fungsi-fungsi yang memudahkan pemakaiannya. Dibawah ini merupakan program yang digunakan untuk mengetahui nilai PSNR dari setiap file citra sebelum dan sesudah disisipkan pesan.

3.3 Perbandingan kualitas citra dengan PSNR




Dari proses penyisipan pesan ke dalam file citra tentunya akan ada perbedaan kualitas citra sebelum dan sesudah proses penyisipan pesan, untuk mengetahui seberapa besar penurunan kualitas citra maka akan dilakukan perhitungan nilai PSNR seperti yang telah dijelaskan pada bab sebelumnya.

Berikut ini akan dilakukan pengujian terhadap enam buah citra uji yang telah berisikan pesan tersembunyi melalui aplikasi steganography yang telah dibuat. Keenam citra uji ini akan disisipkan sejumlah karakter dengan jumlah yang bervariasi mulai dari 100 karakter hingga 500 karakter, hal ini dimaksudkan untuk mengetahui seberapa besar perubahan yang terjadi pada citra uji yang diukur dengan besarnya perubahan nilai PSNR dari setiap citra uji tersebut.

Citra pengujian memiliki ukuran yang bervariasi, yang diharapkan dapat menunjukkan kemampuan aplikasi steganography yang dibuat terhadap berbagai macam ukuran citra uji.

Tabel 4.2 menunjukkan citra uji yang digunakan beserta deskripsi terhadap citra tersebut yang diperlukan dalam proses pengujian.

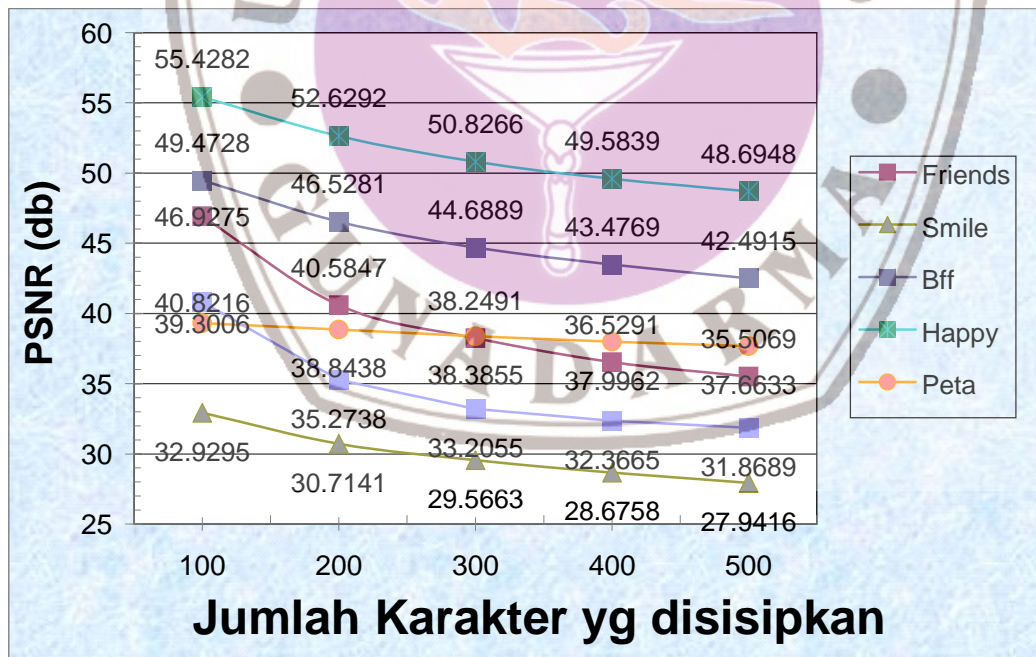
Tabel 3.2 Citra Pengujian

Nama Citra Uji	Gambar Citra	Ukuran Citra (piksel x piksel)	Ukuran Citra Awal (KB)	Ukuran Citra Akhir (KB)
Hold on.bmp		495 x 342	40.8	626
Friend. bmp		604 x 453	63	1064.96
Smile.bmp		400 x 320	42.3	500

Bff.bmp		1600 x 1200	391	7495.68
Peta.bmp		772 x 698	105	2099.2
Happy.bmp		3072 x 2304	534	27648

Jumlah karakter yang disisipkan	Nilai PSNR (db)					
	Citra Happy	Citra Bff	Citra Friends	Citra Hold on	Citra Peta	Citra Smile
100	55,4282	49,4728	46,9275	40,8216	39,3006	32,9295
200	52,6292	46,5281	40,5847	35,2738	38,8438	30,7141
300	50,8266	44,6889	38,2491	33,2055	38,3855	29,5663
400	49,5839	43,4769	36,5291	32,3665	37,9962	28,6758
500	48,6948	42,4915	35,5069	31,8689	37,6633	27,9416

Tabel 3.3 Hasil Pengujian Citra dalam Nilai PSNR (db)



Gambar 3.10 Grafik Perbandingan Nilai PSNR Terhadap Jumlah Karakter yang disisipkan.

Dari hasil percobaan diatas terhadap keenam buah file citra uji yang disisipkan karakter dengan jumlah yang bervariasi serta jumlah karakter yang terus ditambahkan pada setiap pengujiannya yakni dari 100, 200, 300, 400 dan 500 karakter diperoleh hasil nilai desibel dari masing-masing file citra uji yang berbeda untuk setiap jumlah karakter yang disisipkan.

Hasil pengujian nilai PSNR dari setiap file citra yang diuji dapat dilihat dari tabel 4.3 diatas dan grafik perbandingan nilai PSNR terhadap jumlah karakter yang disisipkan dapat pula dilihat pada gambar 4.10 dari grafik tersebut terlihat sangat jelas bahwa jumlah karakter yang disisipkan pada setiap file citra uji berpengaruh terhadap nilai PSNR yang dihasilkan atau dengan kata lain file citra uji yang digunakan mengalami perubahan sesuai dengan jumlah karakter yang disisipkan kedalam file citra sebelumnya. Semakin banyak karakter yang disisipkan maka semakin berkurang pula kualitas citra yang dihasilkan.

Hal ini ditandai dengan berkurangnya nilai PSNR yang dihasilkan oleh masing-masing file citra uji, dimana dari uji coba pengujian penyisipan karakter dengan jumlah karakter yang berbeda-beda, diperoleh hasil PSNR yang semakin berkurang sesuai dengan banyaknya karakter yang disisipkan, seperti pada file citra uji happy.bmp dimana pada penyisipan 100 karakter diperoleh nilai PSNR 55,4282 db dan nilai PSNR yang semakin menurun sesuai dengan banyaknya karakter yang disisipkan sampai dengan 48,6948 db pada penyisipan 500 karakter. Dari hasil uji coba pada keenam file citra uji diperoleh rata-rata penurunan nilai PSNR sebesar 6,7855 db untuk setiap file citra uji.

Besarnya ukuran file citra juga mempengaruhi perolehan nilai PSNR. Nilai PSNR yang dihasilkan dari keenam file citra uji bervariasi sesuai dengan besar file ukuran file citra yang digunakan, seperti yang terlihat pada tabel 4.2 dan tabel 4.3 bahwa nilai PSNR yang dihasilkan semakin berkurang sesuai dengan besar ukuran file citra yang digunakan dengan jumlah penyisipan karakter yang sama, hal ini dapat dilihat pada file citra uji happy.bmp yang memiliki ukuran 3072 x 2304 piksel dengan file citra uji smile.bmp yang memiliki ukuran 400 x 320 piksel dimana pada dengan penyisipan 100

karakter pada file citra happy.bmp diperoleh nilai PSNR 55,4282 db sedangkan pada file citra uji smile.bmp diperoleh nilai PSNR 32,9295 db.

Dari hasil uji coba proses ekstraksi pesan yang terdapat pada file citra uji dalam aplikasi Steganografi ini, pesan atau informasi yang disisipkan pada file citra dapat diperoleh kembali secara utuh atau dengan kata lain pesan yang disisipkan sebelum proses penyisipan dan setelah proses ekstraksi sama tanpa ada perubahan atau gangguan yang menyebabkan isi pesan tidak dapat diperoleh sepenuhnya.

Hal ini membuktikan bahwa pada aplikasi Steganografi yang di buat ini menghasilkan hasil yang cukup baik untuk setiap penyembuyian pesan kedalam file citra uji bergantung dari pemilihan *cover-object* atau file citra yang akan digunakan dan banyaknya karakter yang disisipkan pada file citra, 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.

Kesimpulan

Dari penulisan ini maka dapat disimpulkan bahwa aplikasi Steganografi yang telah dihasilkan dari implementasi algoritma LSB (*Least Significant Bit*) dapat digunakan dengan baik untuk menyembunyikan pesan di dalam pesan sebuah image atau file citra digital sedemikian rupa sehingga orang lain tidak menyadari ada sesuatu di dalam pesan tersebut.

Pada proses ekstraksi, pesan atau informasi yang disisipkan pada file citra uji dalam aplikasi Steganografi ini, dapat diperoleh kembali secara utuh atau dengan kata lain pesan yang disisipkan sebelum proses penyisipan dan setelah proses ekstraksi sama tanpa ada perubahan atau gangguan yang menyebabkan isi pesan tidak dapat diperoleh sepenuhnya.

Hasil pengujian nilai PSNR terhadap image atau file citra digital yang dihasilkan dari aplikasi Steganografi inipun menunjukkan nilai yang cukup baik bergantung pada besar ukuran file citra yang digunakan dan besarnya jumlah karakter yang disisipkan pada file citra tersebut. Semakin besar ukuran file citra yang digunakan maka semakin baik nilai PSNR dalam decibel (db) yang diperoleh di bandingkan dengan file citra yang berukuran lebih kecil dengan jumlah sisipan karakter yang sama. Hal ini menunjukkan bahwa untuk memperoleh file citra yang baik setelah proses penyisipan, dan tidak mengalami perubahan yang cukup berarti dari file citra sebelumnya maka besar ukuran file citra dalam piksel dan banyaknya karakter yang akan disisipkan perlu diperhatikan untuk memperoleh hasil yang baik.

Dengan demikian pesan yang disisipkan kedalam file citra tidak akan menimbulkan kecurigaaan dan menjaga keamanan pesan yang disisipkan dalam file citra digital tersebut.

Saran

Pada aplikasi Steganografi ini, file citra yang dihasilkan setelah proses penyisipan mengalami pengurangan kualitas yang cukup banyak bergantung dari jumlah karakter yang disisipkan, dimana semakin banyak karakter yang disisipkan maka semakin besar pula pengurangan kualitas citra yang diperoleh yang ditandai dengan pengurangan nilai PSNR. Oleh karena itu, untuk meningkatkan kualitas citra dihasilkan maka kedepannya diharapkan dapat dikembangkan suatu aplikasi Steganografi dengan metode lain yang lebih baik agar kualitas citra yang dihasilkan tidak jauh berbeda dengan kualitas citra sebelumnya.



This document was created with Win2PDF available at <http://www.daneprairie.com>.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.