

**ANALISIS DAN PERANCANGAN APLIKASI
STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN
METODE LSB (*LEAST SIGNIFICANT BIT*)**



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
SYARIF HIDAYATULLAH
JAKARTA
2010 M/ 1431 H**

**ANALISIS DAN PERANCANGAN APLIKASI
STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN
METODE LSB (*LEAST SIGNIFICANT BIT*)**



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
SYARIF HIDAYATULLAH
JAKARTA
2010 M/ 1431 H**

**ANALISIS DAN PERANCANGAN APLIKASI
STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN
METODE LSB (*LEAST SIGNIFICANT BIT*)**

ADIRIA

204091002554

Skripsi

Diajukan Untuk Memenuhi Syarat Kelulusan Sarjana (S1)

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
SYARIF HIDAYATULLAH
JAKARTA
2010 M/ 1431 H**

**ANALISIS DAN PERANCANGAN APLIKASI STEGANOGRAFI PADA
CITRA DIGITAL MENGGUNAKAN METODE LSB (*LEAST
SIGNIFICANT BIT*)**

Skripsi

Sebagai salah satu syarat untuk memperoleh gelar sarjana komputer

Pada Fakultas Sains dan Teknologi

Universitas Islam Negeri Syarif Hidayatullah Jakarta

Oleh :

Adiria

204091002554

Pembimbing I

Menyetujui,

Pembimbing II

Arini, MT

NIP.197601312009012001

Qurrotul Aini, MT

NIP.197303252009012001

Mengetahui :

Ketua Program Studi Teknik Informatika

Yusuf Durrachman, M.Sc
NIP.197105222006041002

PENGESAHAN UJIAN

Skripsi ini berjudul **“ANALISIS DAN PERANCANGAN APLIKASI STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN METODE LSB (*LEAST SIGNIFICANT BIT*)”**, telah diuji dan dinyatakan LULUS dalam sidang Munaqosyah Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta. Pada hari Senin skripsi ini telah diterima sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata Satu (S1) Program Studi Teknik Informatika.

Jakarta, 5 April 2010

Penguji I

Penguji II

Victor Amrizal

Imam M.Shofi

Dekan
Fakultas Sains dan Teknologi

Ketua Program
Studi Teknik Informatika

Dr. Syopiansyah Jaya Putra, M.Sis
NIP. 150317956

Yusuf Durrachman, M.Sc
NIP. 197105222006041002



PERNYATAAN

DENGAN INI SAYA MENYATAKAN BAHWA SKRIPSI INI BENAR – BENAR HASIL KARYA SENDIRI YANG BELUM PERNAH DIAJUKAN SEBAGAI SKRIPSI ATAU KARYA ILMIAH PADA PERGURUAN TINGGI ATAU LEMBAGA MANAPUN.



Jakarta, 5 April 2010

Adiria

204091002554

ABSTRAK

Adiria, Analisis Dan Perancangan Aplikasi Steganografi pada Citra Digital Menggunakan Metode LSB (*Least Significant Bit*). Dibimbing oleh Ibu **Arini, MT** dan **Qurrotul Aini, MT**.

Di seluruh dunia, internet (*interconnection network*) sudah berkembang menjadi salah satu media komunikasi data yang sangat populer. Dengan berbagai teknik pengambilan informasi secara ilegal yang berkembang, banyak orang yang mencoba untuk mengakses informasi yang bukan haknya. Berbagai macam teknik untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak telah banyak dilakukan dalam upaya mengamankan suatu data penting. Steganografi adalah salah satu teknik menyembunyikan *file* pesan agar bagi orang awam tidak menyadari keberadaan dari *file* pesan yang disembunyikan. Teknik ini sering digunakan untuk menghindari kecurigaan orang dan menghindari keinginan orang untuk mengetahui isi *file* pesan tersebut. Citra Digital adalah salah satu media yang paling umum dikenal oleh masyarakat. Penelitian ini bertujuan menganalisis dan merancang suatu aplikasi steganografi sebagai salah satu teknik pengamanan file pesan. Pada teknik steganografi ini digunakan metode LSB (*Least Significant Bit*) yaitu menyembunyikan *byte-byte* data atau informasi pada bit terakhir pada citra digital, sehingga tidak terjadi perubahan ukuran dan bentuk terhadap citra digital secara kasat mata, data atau informasi pun dapat dikembalikan seperti semula tanpa ada perubahan ukuran dan bentuknya.

Kata Kunci : Aplikasi, Steganografi, Citra Digital, Steganografi, LSB

KATA PENGANTAR

بسم الله الرحمن الرحيم

Segala puji syukur penulis panjatkan kehadiran Allah SWT, sang pemilik sifat rahman dan rahiim yang telah melimpahkan segala rahmat dan inayahNya, sehingga penulis dapat menyelesaikan skripsi ini. Shalawat dan salam selalu tercurahkan kepada junjungan baginda besar Nabi Muhammad SAW.

Sebagai bentuk penghargaan yang tidak terlukiskan, penulis menuangkan dalam bentuk ucapan terima kasih sebesar-besarnya kepada:

1. DR. Syopiansyah Jaya Putra, M.Sis, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta.
2. Bapak Yusuf Durrachman, M.Sc, selaku Ketua Program Studi Teknik Informatika dan Ibu Viva Arifin, MMSI, selaku Sekretaris Program Studi Teknik Informatika.
3. Ibu Arini, MT dan Ibu Qurrotul Aini, MT selaku dosen pembimbing yang telah memberikan waktu dan perhatiannya dalam penyusunan skripsi ini.
4. Ayahanda Masri dan Ibunda Marlina yang senantiasa memberikan dukungan baik moral, materil, dan spiritual selama menyelesaikan skripsi ini *“mom dad this is for you”*.
5. Adik–adikku tersayang yaitu Wahyudi *“rajin belajar & cepet dewasa pikiranya ya..”* dan Ariyo Saputra *“hai jagoan kecilku cepet gede ya..”*.
Saudara-saudara sepupuku : Ristamansyah *“makasih da”*, Usni *“makasih*

bang”, Syadriansyah “thanx sob”, Putra, Eki, Elo, Teta, K wis, K widia, k Eni, Reza & Ncim “*makasih udah boleh numpang ngeprint selama skripsi, kapan2 numpang lagi ya hehe..*” serta saudara– saudara penulis lainnya yang telah mendukung dan memberikan doanya. Semoga kita selalu dalam hangatnya ikatan keluarga yang kokoh, kuat dan abadi serta saling membantu satu sama lain.

6. Ari Kristianto, Mirwan Nurjaya, Muhammad Agus Syarifuddin, Muhammad Qadhavi, Muhammad Syah Reza, Rahmat Mulya, Setiajid, Wangsa Dipraja, Yayan Pebriana, Yazidanyastuti, Fila Anggraini, Personal Motivator si nenk “*makasih ya buat supportnya*”, Personal Tour Guide si aa “*makasih buat jalan2 dan refresingnya*”, Anton Budiwan “*makasih minjem kamarnya buat ngetik*”, dan teman-teman di kosan “*god bless you all my friends*”.
7. Seluruh rekan-rekan yang tidak dapat penulis sebutkan satu persatu yang secara tidak langsung membantu dan memberikan semangat sehingga penulisan skripsi ini dapat berjalan dengan lancar.

Akhir kata hanya kepada Allah jualah penulis memanjatkan doa, semoga Allah memberikan balasan berupa amal yang berlipat kepada mereka dan semoga skripsi ini dapat bermanfaat dan memberikan kontribusi bagi semua pihak. Amiin.

Jakarta, 5 April 2010

Adiria
204091002554

DAFTAR ISI

Halaman Judul	i
Halaman Keterangan Judul	ii
Lembar Pengesahan Pembimbing	iii
Lembar Pengesahan Ujian	iv
Lembar Pernyataan	v
Abstrak	vi
Kata Pengantar	vii
Daftar Isi	ix
Daftar Gambar	xiii
Daftar Tabel	xvi
Daftar Istilah	xvii
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Identifikasi Masalah	2
1.3 Rumusan Masalah	3
1.4 Batasan Masalah	3
1.5 Tujuan Penelitian	4
1.6 Manfaat Penelitian	4
1.7 Metode Penelitian	5
1.7.1 Metode Pengumpulan Data Dan Informasi	5

1.7.2 Metode Pengembangan Sistem	6
1.8 Sistematika Penulisan	7
BAB II LANDASAN TEORI.....	8
2.1 Pengertian Analisis	8
2.2 Pengertian Perancangan	8
2.3 Steganografi	9
2.3.1 Sejarah Steganografi	9
2.3.2 Pengertian Steganografi	10
2.3.3 Metode Steganografi	14
2.4 LSB (<i>Least Significant Bit</i>)	17
2.5 ASCII	20
2.6 Multimedia	23
2.7 Citra Digital	24
2.7.1 Pengertian Citra Digital	24
2.7.2 Pengolahan Citra (<i>Image Processing</i>)	25
2.7.3 Perbandingan <i>File</i> Gambar BMP (Bitmap) dengan JPG, GIF, atau PNG	28
2.8 <i>Delphi</i>	30
2.8.1 Pengertian <i>Delphi</i>	30
2.8.2 Kelebihan <i>Delphi</i>	31
2.9 Perancangan Program	32
2.9.1 Model–Model Pengembangan Sistem	32
2.9.2 Diagram Alur (<i>Flowchart</i>)	34
2.9.3 <i>State Transition Diagram</i> (STD)	36

2.10 Konsep RAD	38
2.10.1 Definisi RAD	38
2.10.2 Tahapan-Tahapan RAD	38
2.10.3 Keunggulan RAD	40
BAB III METODE PENELITIAN	42
3.1 Metode Pengumpulan Data	42
3.2 Metode Pengembangan Sistem	43
3.2.1 Fase Perencanaan Syarat-Syarat	44
3.2.2 Proses Desain RAD (<i>RAD Design Workshop</i>).....	44
3.2.3 Fase Implementasi (<i>Implementation Phase</i>)	45
BAB IV ANALISIS DAN PERANCANGAN	47
4.1 Fase Perencanaan Syarat-Syarat	47
4.1.1 Analisis Kebutuhan	47
4.1.2 Menentukan Tujuan	48
4.1.3 Menentukan Syarat-Syarat	48
4.2 Fase Desain RAD	49
4.2.1 Perancangan Proses	49
4.2.2 <i>Flowchart</i> Aplikasi Steganografi	50
4.2.3 Perancangan Antarmuka	53
4.2.4 STD (<i>State Transition Diagram</i>)	58
4.2.5 Konstruksi	59
4.3 Fase Implementasi	60
4.3.1 Instalasi Aplikasi	60

4.3.2	Analisis Penyembunyian <i>File</i>	62
4.3.3	Analisis Format <i>File</i> Apa Saja yang Dapat Disisipkan ke <i>File Image</i>	62
4.3.4	Analisis Penyisipan dan Ekstraksi <i>File</i> Pesan terhadap Tiga <i>File Image</i> yang Berbeda	64
4.3.5	Analisis Penyisipan dan Ekstraksi <i>File Image</i> dan <i>File</i> Pesan	66
4.3.6	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 2 Bit Terakhir	67
4.3.7	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 3 Bit Terakhir	70
4.3.8	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 4 Bit Terakhir	72
4.3.9	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 5 Bit Terakhir	74
4.3.10	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 6 Bit Terakhir	76
4.3.11	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 7 Bit Terakhir	78
4.3.12	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 8 Bit Terakhir	80
4.3.13	Analisis Ukuran <i>File</i> Pesan Terhadap <i>File Image</i>	82
4.4	Perbandingan Steganografi Sejenis	83
BAB V	PENUTUP	86
5.1	Kesimpulan	86
5.2	Saran	87
DAFTAR PUSTAKA	88
LAMPIRAN	91

DAFTAR GAMBAR

Gambar 2.1	Perbedaan Pesan yang Disembunyikan.....	11
Gambar 2.2	Diagram Penyisipan dan Ekstraksi Pesan	13
Gambar 2.3	Diagram Sistem Steganografi.....	14
Gambar 2.4	<i>Spread Spectrum Method</i>	16
Gambar 2.5	<i>Most Significant Bit</i> (MSB) dan <i>Least Significant Bit</i> (LSB)	17
Gambar 2.6	Proses <i>File Image</i> Menjadi Kumpulan <i>Pixel-Pixel</i>).....	17
Gambar 2.7	Nilai-Nilai <i>Pixel</i> Dalam Suatu <i>File Image</i>	18
Gambar 2.8	Koordinat Spasial dan Nilai $f(x,y)$	25
Gambar 2.9	Bagan Pengolahan Citra	25
Gambar 2.10	Tampilan <i>Form</i> Awal <i>Delphi</i> 2007 Win32	31
Gambar 2.11	Simbol State	37
Gambar 2.12	Simbol <i>Transition State</i>	37
Gambar 2.13	Simbol Kondisi dan Aksi	37
Gambar 2.14	Tahapan-Tahapan RAD.....	40
Gambar 3.1	Skema Sistem Model RAD	43
Gambar 3.2	Ilustrasi Metode Penelitian Pengembangan Aplikasi Steganografi pada Citra <i>Digital</i>	46
Gambar 4.1	Proses Penyisipan dan Ekstraksi Pesan.....	50
Gambar 4.2	<i>Flowchart</i> Proses Penyisipan Pesan Rahasia	51
Gambar 4.3	<i>Flowchart</i> Proses Ekstraksi Pesan Rahasia	52

Gambar 4.4	<i>Layout Form Induk</i>	54
Gambar 4.5	Rancangan <i>Form</i> Utama <i>Tab Encoding</i>	56
Gambar 4.6	Rancangan <i>Form</i> Utama <i>Tab Decoding</i>	57
Gambar 4.7	Perancangan <i>Form About</i>	58
Gambar 4.8	<i>State Transition Diagram</i> Aplikasi Steganografi pada Citra <i>Digital</i>	59
Gambar 4.9	<i>File Image</i> Sebelum dan Sesudah Disisipkan <i>File</i> Pesan	66
Gambar 4.10	<i>File</i> Pesan Rahasia Sebelum Disisipkan ke <i>File Image</i>	67
Gambar 4.11	<i>File</i> Pesan yang Diambil Kembali dari <i>File Image</i>	67
Gambar 4.12	Perbedaan Kualitas <i>File Image</i> Jika Digunakan 2 Bit Terakhir.....	68
Gambar 4.13	Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 2 Bit Terakhir.....	69
Gambar 4.14	Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 2 Bit Terakhir.....	69
Gambar 4.15	Perbedaan Kualitas <i>File Image</i> Jika Digunakan 3 Bit Terakhir.....	70
Gambar 4.16	Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 3 Bit Terakhir.....	71
Gambar 4.17	Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 3 Bit Terakhir.....	71
Gambar 4.18	Perbedaan Kualitas <i>File Image</i> Jika Digunakan 4 Bit Terakhir.....	72
Gambar 4.19	Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 4 Bit Terakhir.....	73
Gambar 4.20	Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 4 Bit Terakhir.....	73
Gambar 4.21	Perbedaan Kualitas <i>File Image</i> Jika Digunakan 5 Bit Terakhir.....	74
Gambar 4.22	Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 5 Bit Terakhir.....	75

Gambar 4.23 Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 5 Bit Terakhir.....	75
Gambar 4.24 Perbedaan Kualitas <i>File Image</i> Jika Digunakan 6 Bit Terakhir.....	76
Gambar 4.25 Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 6 Bit Terakhir.....	77
Gambar 4.26 Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 6 Bit Terakhir.....	77
Gambar 4.27 Perbedaan Kualitas <i>File Image</i> Jika Digunakan 7 Bit Terakhir.....	78
Gambar 4.28 Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 7 Bit Terakhir.....	79
Gambar 4.29 Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 7 Bit Terakhir.....	79
Gambar 4.30 Perbedaan Kualitas <i>File Image</i> Jika Digunakan 8 Bit Terakhir.....	80
Gambar 4.31 Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 8 Bit Terakhir.....	81
Gambar 4.32 Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 8 Bit Terakhir.....	81

DAFTAR TABEL

Tabel 2.1 Keuntungan dan Kelemahan Metode LSB	19
Tabel 2.2 ASCII	21
Tabel 2.3 Beberapa Metode dan Perbedaannya	32
Tabel 2.4 Simbol-simbol <i>Flowchart</i>	34
Tabel 4.1 Tabel Uji Penyembunyian <i>File</i>	62
Tabel 4.2 Tabel Uji Format <i>File</i> Apa Saja yang Dapat Disisipkan ke <i>File Image</i>	63
Tabel 4.3 Tabel Uji <i>File</i> Output dari 3 <i>File</i> Pesan Berbeda, <i>File Image</i> “Monalisa.bmp”	65
Tabel 4.4 Tabel Uji <i>File</i> Output dari 3 <i>File</i> Pesan Berbeda, <i>File Image</i> “Lena.bmp”	65
Tabel 4.5 Tabel Uji <i>File</i> Output dari 3 <i>File</i> Pesan Berbeda, <i>File Image</i> “Fruits.bmp”	65
Tabel 4.6 Uji Ukuran <i>File</i> Pesan Rahasia Terhadap <i>File Image</i>	82
Tabel 4.7 Perbandingan Steganografi Sejenis 1.....	83
Tabel 4.8 Perbandingan Steganografi Sejenis 2.....	84

DAFTAR ISTILAH

ASCII (American Standart Code For Information Interchange):

Standar huruf dan tanda baca untuk komputer. ASCII merupakan kode berupa karakter 8 bit berbentuk angka 1 dan 0 untuk mewakili karakter-karakter alpha numerik.

Bit:

Berasal dari kata *binary* digit. Merupakan satuan terkecil data komputer yang hanya merupakan angka 0 dan 1. Semua data bisa ditulis dalam bit.

Bit Depth:

Jumlah bit yang digunakan untuk mempresentasikan tiap titik dalam representasi citra grafis. Makin besar jumlah bit yang digunakan untuk mempresentasikan suatu titik, semakin banyak warna dan atau bayangan abu-abu yang dapat dibuat.

Bitmap:

Sebuah *image* grafis yang disusun dari *pixel-pixel* dan dikonversikan ke dalam bit. Biasa digunakan dalam Microsoft Windows.

Byte:

Kumpulan delapan buah bit, yang membentuk satu data bermakna, misalnya huruf A, B, atau angka 3,4 dan lain-lain.

Color Depth:

Color depth merujuk pada jumlah warna yang ditampilkan di monitor oleh *video card*. Semakin banyak warna yang digunakan, semakin realistis tampilan yang dapat dilihat. Seperti gambar yang didapat dari foto, merubah *color depth* komputer dapat meningkatkan kualitas gambar, namun dapat juga tidak apabila foto itu sendiri terbatas pada jumlah warna tertentu.

Encrypt:

Penerjemahan data menjadi kode rahasia. Enkripsi adalah cara yang paling efektif untuk memperoleh pengamanan data. Untuk membaca *file* yang di-*enkrip*, kita harus mempunyai akses terhadap kata sandi yang memungkinkan kita men-*dekrip* pesan tersebut.

Decrypt:

Mengubah kembali hasil enkripsi ke bentuk aslinya sehingga informasi tersebut dapat dibaca.

Error: Istilah untuk menunjukan bahwa terdapat suatu penyimpangan dalam *software* atau kerusakan *hardware*.

Pixel:

Picture element, elemen terkecil citra digital yang bisa dilihat mata. Sensor citra secara fisik (dua dimensi) dibuat dari rangkaian ribuan sel yang peka cahaya. Tiap sel disebut *pixel*, bagi monitor atau *display* komputer, *pixel* adalah titik-titik cahaya yang membentuk suatu objek di layar komputer.



BAB I

PENDAHULUAN

1.1 Latar Belakang

Di seluruh dunia, internet (*interconnection network*) sudah berkembang menjadi salah satu media komunikasi data yang sangat populer. Kemudahan dalam penggunaan dan fasilitas yang lengkap merupakan keunggulan yang dimiliki oleh internet dan bukan rahasia umum di kalangan masyarakat pengguna internet pada saat sekarang ini. Namun, seiring dengan berkembangnya media internet dan aplikasi yang menggunakan internet semakin bertambah pula kejahatan dalam sistem informasi. Dengan berbagai teknik pengambilan informasi secara ilegal yang berkembang, banyak yang mencoba untuk mengakses informasi yang bukan haknya. Untuk itu, sejalan dengan berkembangnya media internet yang sangat cepat, harus juga diikuti dengan perkembangan pengamanan dalam sistem informasi yang berada dalam media internet tersebut.

Berbagai macam teknik yang digunakan untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak telah banyak dilakukan dalam upaya mengamankan suatu data penting dengan menggunakan sistem kriptografi yang melakukan enkripsi sebelum data penting tersebut ditransmisikan. Tindakan pengamanan menggunakan cara tersebut ternyata dianggap belum cukup dalam mengamankan suatu data karena adanya peningkatan kemampuan komputasi. Berbeda dengan teknik kriptografi, steganografi menyembunyikan pesan rahasia

agar bagi orang awam tidak menyadari keberadaan dari pesan yang disembunyikan. Teknik ini sering digunakan untuk menghindari kecurigaan orang dan menghindari keinginan orang untuk mengetahui isi pesan rahasia tersebut. Caranya dengan menyembunyikan informasi rahasia di dalam suatu wadah penampung informasi dengan sedemikian rupa sehingga keberadaan informasi rahasia yang ditempelkan tidak terlihat. Wadah penampung informasi tersebut dapat berbentuk berbagai jenis *file* multimedia *digital* seperti teks, citra, audio, video. Dalam tugas akhir ini, peneliti membuat analisis dan merancang aplikasi steganografi merupakan solusi dari permasalahan tersebut. Dengan penggunaan teknik ini, data informasi dapat kita sembunyikan di dalam media digital yang kita punya.

1.2 Identifikasi Masalah

Berbagai macam teknik yang digunakan untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak telah banyak dilakukan dalam upaya mengamankan suatu data penting dengan menggunakan sistem kriptografi yang melakukan enkripsi sebelum data penting tersebut ditransmisikan. Tetapi cara ini juga menimbulkan rasa keingintahuan seseorang untuk memecahkan rahasia tersebut, sehingga terciptalah *software-software* yang mampu memecahkan enkripsi dalam pesan, sehingga keamanan informasi tidak selalu terjamin.

Dengan steganografi penulis memanfaatkan kelemahan indera manusia, sehingga informasi yang ingin kita sampaikan tidak jatuh pada orang-orang yang

tidak berhak mengaksesnya, tanpa menimbulkan rasa keingintahuan seseorang terhadap informasi tersebut.

1.3 Rumusan Masalah

Dari uraian latar belakang di atas, maka dapat dirumuskan masalah pada tugas akhir berikut:

1. Bagaimana cara menyembunyikan *file* ke dalam media citra *digital*, sehingga *file* benar-benar terjaga keamanannya dengan menggunakan teknik steganografi sebagai suatu solusi pengamanan pesan?
2. Bagaimana metode LSB dapat digunakan sebagai salah satu metode steganografi dalam penyembunyian *file* ke dalam citra digital?

1.4 Batasan Masalah

Batasan masalah dalam tugas akhir ini mencakup:

1. Format *file* citra *digital* yang dapat digunakan untuk menyimpan pesan adalah 24bit (*true color*) berformat *.bmp.
2. Format *file* citra digital yang dihasilkan dari program steganografi ini adalah *.bmp.
3. Teknik steganografi yang digunakan hanya dapat menyimpan pesan berupa format *.txt, *.doc, *.xls, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav.
4. Metode steganografi yang digunakan adalah LSB (*Least Significant Bit*).
5. *Tools* yang digunakan adalah Delphi 2007 Win 32.

1.5 Tujuan Penelitian

Tujuan yang hendak dicapai dalam tugas akhir ini adalah:

1. Memberikan informasi bagaimana teknik steganografi dapat diterapkan di dalam *file* citra digital.
2. Implementasi penyembunyian *file* ke dalam citra digital dengan memberikan satu alternatif metode steganografi.
3. Memanipulasi citra digital yang di dalamnya terdapat *file* sehingga *file* tersebut tidak dapat diketahui keberadaannya dan secara kasat mata tidak terjadi perubahan pada citra *digital* hasil manipulasi.

1.6 Manfaat Penulisan

Manfaat yang akan didapat dari penulisan skripsi dalam pembuatan aplikasi steganografi ini adalah:

a. Bagi Penulis

1. Memenuhi tugas akhir sebagai syarat untuk menyelesaikan studi Strata 1 (S-1) Teknik Informatika.
2. Mengetahui seberapa besar kemampuan mengimplementasikan pengetahuan mengenai steganografi pada sebuah aplikasi.
3. Hasil penelitian diharapkan dapat digunakan untuk mengembangkan ilmu komputer khususnya dalam bidang steganografi.
4. Sebagai bahan pertimbangan bagi seseorang dalam mengamankan *filenya*.

b. Bagi Universitas

1. Mengetahui kemampuan mahasiswa dalam menguasai materi teori yang telah diperoleh selama masa kuliah.
2. Mengetahui kemampuan mahasiswa dalam menerapkan ilmunya dan sebagai bahan evaluasi.

c. Bagi Masyarakat

1. Sebagai salah satu solusi dalam hal mengamankan *file* mereka dari orang-orang yang tidak berhak melihatnya.
2. Sebagai salah satu bahan pertimbangan bagi mereka yang ingin mengamankan *file* rahasia mereka agar terjaga kerahasiaannya.

1.7 Metode Penelitian

1.7.1 Metode Pengumpulan Data dan Informasi

a. Studi Pustaka

Dengan metode ini, penulis mendapatkan informasi yang berkaitan dengan steganografi melalui buku-buku referensi dan melalui berbagai situs di internet yang berkaitan, sehingga penulis dapat mengumpulkan data dan informasi yang diperlukan.

b. Studi Literatur

Penulis mencoba mencari perbandingan dengan studi sejenis dari beberapa penulisan di beberapa karya ilmiah, seperti jurnal dan skripsi.

1.7.2 Metode Pengembangan Sistem

Dalam menyusun tugas akhir ini penulis menggunakan metode pengembangan sistem *Rapid Application Development* (RAD). Ada tiga fase luas pada RAD yang mengajak *user* maupun *analyst* dalam merencanakan, mendesain, dan mengimplementasi suatu sistem.

1. Fase Kebutuhan Perencanaan (*Requirement Planning Phase*)

Menentukan tujuan dan syarat dari informasi.

2. Proses Desain RAD (*RAD Design Workshop*)

Perancangan proses yang akan terjadi dalam sistem, perancangan *input* dan *output interface*, serta pengkodean terhadap rancangan rancangan yang telah didefinisikan.

3. Fase Implementasi (*Implementation Phase*)

Pada tahapan ini dilakukan pengujian terhadap sistem dan melakukan pengenalan terhadap sistem (Kendall, 2005).

1.8 Sistematika Penulisan

Dalam penyusunan skripsi ini penulis menyajikan tulisan ini terdiri atas:

BAB I PENDAHULUAN

Bab ini terdiri dari latar belakang masalah, identifikasi masalah, rumusan masalah, batasan masalah, tujuan, manfaat, metode penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini berisi uraian tentang landasan teori yang diperlukan dalam analisis dan perancangan aplikasi steganografi pada citra digital menggunakan metode LSB (*Least Significant Bit*).

BAB III METODE PENELITIAN

Bab ini menguraikan secara rinci metode yang digunakan dalam analisis dan perancangan aplikasi.

BAB IV ANALISIS DAN PERANCANGAN

Bab ini membahas mengenai analisis, perancangan dan pengujian sistem.

BAB V PENUTUP

Bab ini berisi kesimpulan dari seluruh bab dan saran-saran untuk pengembangan sistem lebih lanjut.



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.

BAB II

LANDASAN TEORI

2.1 Pengertian Analisis

Analisis berkaitan dengan pemahaman dan pemodelan aplikasi serta domain dimana aplikasi beroperasi. Masukan awal fase analisis adalah pernyataan masalah yang mendeskripsikan masalah yang ingin diselesaikan dan menyediakan pandangan konseptual terhadap sistem yang diusulkan

Sebutan lengkap analisis adalah analisis kebutuhan perangkat lunak (*software requirement analysis*). Analisis adalah mendaftarkan apa-apa yang harus dipenuhi oleh sistem perangkat lunak, bukan mengenai bagaimana sistem perangkat lunak melakukannya (Hariyanto, 2004).

2.2 Pengertian Perancangan

Perancangan merupakan penghubung antara spesifikasi kebutuhan dan implementasi. Perancangan merupakan rekayasa representasi yang berarti terhadap sesuatu yang hendak dibangun. Hasil perancangan harus dapat ditelusuri sampai ke spesifikasi kebutuhan dan dapat diukur kualitasnya berdasar kriteria-kriteria rancangan yang bagus. Perancangan menekankan pada solusi logic mengenai cara sistem memenuhi kebutuhan (Hariyanto, 2004).

2.3 Steganografi

2.3.1 Sejarah Steganografi

Usia steganografi hampir setara usia kriptografi. Steganografi sudah dikenal oleh bangsa Yunani sejak lama. Herodatus, seorang penguasa Yunani, mengirimkan pesan rahasia menggunakan kepala budak atau prajurit sebagai media. Caranya, rambut budak dibotaki, lalu pesan rahasia ditulis pada kulit kepala budak. Setelah rambut-rambut budak tumbuh cukup banyak (yang berarti menutupi pesan rahasia), budak tersebut dikirim ke tempat tujuan pesan untuk membawa pesan rahasia di kepalanya. Di tempat penerima kepala budak dibotaki kembali untuk membaca pesan yang tersembunyi di balik rambutnya. Pesan tersebut berisi peringatan tentang invasi dari Bangsa Persia.

Bangsa Romawi mengenal steganografi dengan menggunakan tinta tak-nampak (*invisible ink*) untuk menulis pesan. Tinta tersebut dibuat dari campuran sari buah, susu, dan cuka. Jika tinta digunakan untuk menulis maka tulisannya tidak tampak. Tulisan di atas kertas dapat dibaca dengan cara memanaskan kertas tersebut (Munir, 2006).

Pada abad 20, steganografi benar-benar mengalami perkembangan. Selama berlangsung perang Boer, Lord Boden Powell (pendiri gerakan kepanduan) yang bertugas untuk membuat tanda posisi sasaran dari basis artileri tentara Boer, untuk alasan keamanan, Boden Powell menggambar peta-peta posisi musuh pada sayap kupu-kupu agar gambar-gambar peta sasaran tersebut terkamuflase (<http://ilmukomputer.org>).

Selama Perang dunia II, agen-agen spionase juga menggunakan steganografi untuk mengirim pesan. Caranya dengan menggunakan titik-titik yang sangat kecil sehingga keberadaanya tidak dapat dibedakan pada tulisan biasa yang diketik.

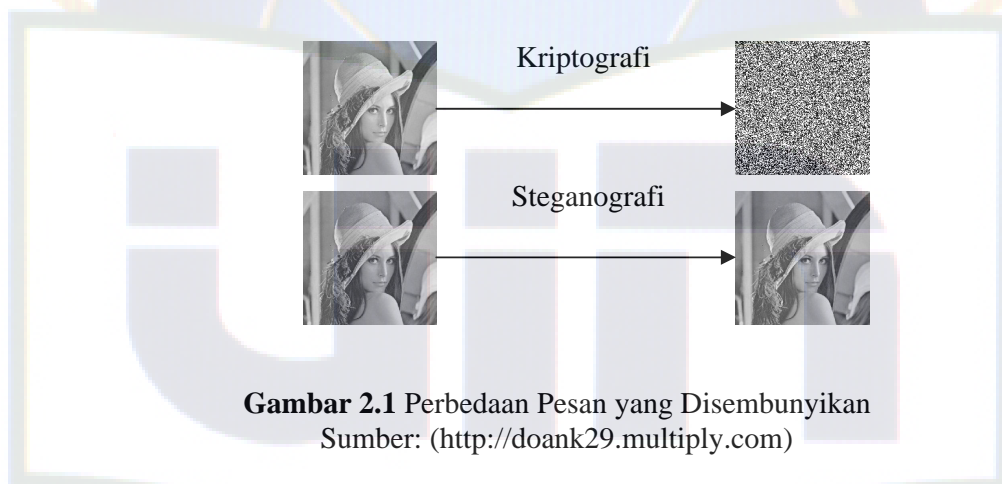
Saat ini steganografi sudah banyak diimplementasikan pada media digital. Steganografi digital menggunakan media digital sebagai penampung, seperti citra digital, video digital, atau audio. Informasi yang disembunyikan juga berbentuk digital seperti teks, citra, data audio, atau data video. Steganografi digital dapat dipakai di negara-negara yang menerapkan sensor ketat terhadap informasi atau di negara di mana enkripsi pesan terlarang. Pada negara-negara seperti itu informasi rahasia dapat disembunyikan dengan menggunakan steganografi (Munir, 2006)

2.3.2 Pengertian Steganografi

Steganografi berasal dari Bahasa Yunani, yaitu “steganos” yang artinya "tulisan tersembunyi (*covered writing*)" dan “graphos” yang berarti tulisan. Steganografi adalah ilmu dan seni menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui (Munir, 2006:). Sedangkan menurut Doni Ariyus, steganografi merupakan cabang ilmu yang mempelajari tentang bagaimana menyembunyikan suatu informasi “rahasia” di dalam informasi lainnya (Ariyus, 2006)

Steganografi membutuhkan dua properti yaitu media penampung dan pesan rahasia. Media penampung yang umum digunakan adalah gambar, suara, video, atau teks. Pesan yang disembunyikan dapat berupa sebuah artikel, gambar, daftar barang, kode barang, atau pesan lain (Munir, 2006). Steganografi berbeda dengan

kriptografi, perbedaannya terletak pada bagaimana proses penyembunyian data dan hasil akhir dari proses tersebut. Kriptografi melakukan proses pengacakan data aslinya sehingga menghasilkan data terenkripsi yang benar-benar acak dan berbeda dengan aslinya, sedangkan steganografi menyembunyikan dalam data lain yang akan ditumpanginya tanpa mengubah data yang ditumpanginya tersebut, sehingga data yang ditumpanginya sebelum dan setelah proses penyembunyian hampir sama (Munir, 2006). Perbedaan kriptografi dan steganografi dapat diilustrasikan pada Gambar 2.1.



Gambar 2.1 Perbedaan Pesan yang Disembunyikan
Sumber: (<http://doank29.multiply.com>)

Tujuan steganografi adalah untuk menghindari kecurigaan (*conspicuous*) sedangkan kriptografi menyembunyikan *isi* (*content*) pesan agar pesan tidak dapat dibaca (<http://haryanto.staff.gunadarma.ac.id>).

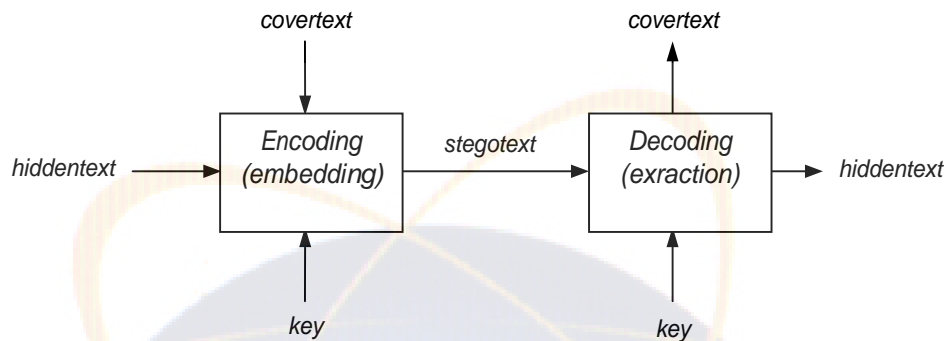
Steganografi memanfaatkan kelemahan indera manusia seperti indera pendengaran dan indera penglihatan. Dengan adanya kelemahan ini steganografi dapat diterapkan di berbagai media *digital*. Hasil keluaran *file* yang telah disisipi pesan mempunyai persepsi bentuk yang sama dengan *file* aslinya. Penggunaan

komputer diperlukan untuk mengetahui keberadaan pesan yang tersembunyi dalam *file digital*.

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 *digital*, baik *hiddentext* maupun *coverttext* dapat berupa teks, citra, audio, maupun video. Jadi, kita dapat menyembunyikan pesan berupa kode program di dalam sebuah citra, atau video, dan kita juga dapat menyembunyikan gambar rahasia di dalam citra lain atau di dalam sebuah berkas musik *mp3*. Penyisipan pesan ke dalam media *coverttext* dinamakan *encoding*, sedangkan ekstraksi pesan dari *stegotext* dinamakan *decoding*. Kedua proses ini mungkin memerlukan kunci rahasia (yang dinamakan *stegokey*) agar hanya pihak yang berhak saja yang dapat melakukan penyisipan pesan dan ekstraksi pesan (Munir, 2006).

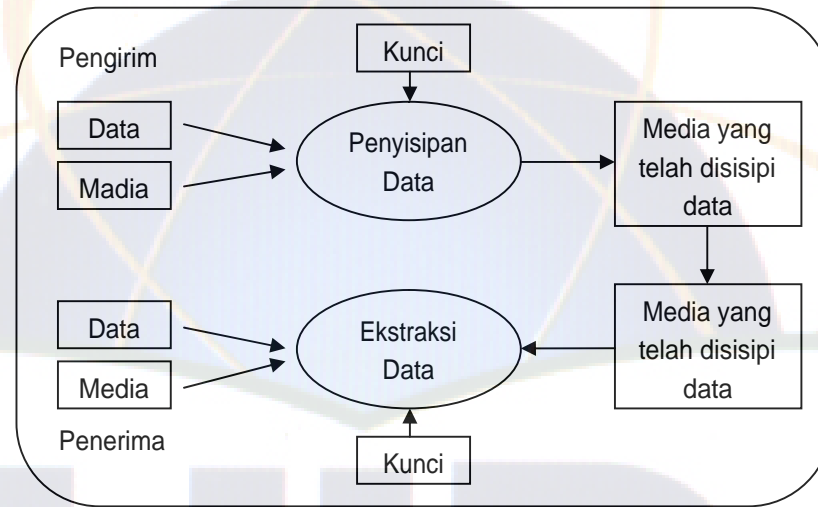


Gambar 2.2 Diagram Penyisipan dan Ekstraksi Pesan
Sumber: (Munir, 2006)

Penyembunyian pesan rahasia ke dalam media penampung pasti mengubah kualitas media tersebut. Kriteria yang harus diperhatikan dalam penyembunyian pesan adalah:

1. *Imperceptibility*. Keberadaan pesan rahasia tidak dapat dipersepsikan oleh indrawi. Misalnya, jika *coverttext* berupa citra, maka penyisipan pesan membuat citra *stegotext* sukar dibedakan oleh mata dengan citra *coverttext*-nya. Jika *coverttext* berupa audio (misalnya berkas mp3, wav, midi, dan sebagainya), maka indra telinga tidak dapat mendeteksi perubahan pada audio *stegotext*-nya.
2. *Fidelity*. Mutu media penampung tidak berubah banyak akibat penyisipan. Perubahan tersebut tidak dapat dipersepsi oleh indrawi. Misalnya, jika *coverttext* berupa citra, maka penyisipan pesan membuat citra *stegotext* sukar dibedakan oleh mata dengan citra *coverttext*-nya. Jika *coverttext* berupa audio (misalnya berkas mp3, wav, midi, dan sebagainya), maka audio *stegotext* tidak rusak dan indra telinga tidak dapat mendeteksi perubahan tersebut.

3. *Recovery*. Pesan yang disembunyikan harus dapat rekonstruksi kembali (*reveal*). Karena tujuan steganografi adalah *data hiding*, maka sewaktu-waktu pesan rahasia di dalam *stegotext* harus dapat diambil kembali untuk digunakan lebih lanjut (Munir, 2006).



Gambar 2.3 Diagram Sistem Steganografi
Sumber: (<http://doank29.multiply.com>)

2.3.3 Metode Steganografi

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

1. Ranah spasial (waktu) (*spasial/time domain*)

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

2. Ranah *transform* (*transform domain*)

Teknik ini memodifikasi langsung hasil transformasi frekuensi sinyal. Contoh metode yang tergolong ke dalam teknik ranah frekuensi adalah *spread spectrum* (Munir, 2006).

Sedangkan ada empat jenis metode Steganografi, yaitu:

1. *Algorithms and Transformation*

Algoritma *compression* (kompresi) 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*).

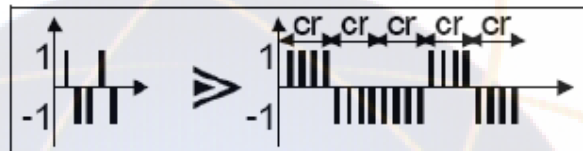
2. *Redundant Pattern Encoding*

Redundant Pattern Encoding adalah menggambar pesan kecil pada kebanyakan gambar. Keuntungan dari metode ini adalah dapat bertahan dari *cropping* (kegagalan), kerugiannya yaitu tidak dapat menggambar pesan yang lebih besar.

3. *Spread Spectrum method*

Spread Spectrum steganografi terpecah-pecah sebagai pesan yang diacak (*encrypt*) 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)

(<http://www.students.if.itb.ac.id>). Contoh dari penyebaran bit-bit informasi dapat dilihat pada Gambar 2.4. Faktor pengali dilambangkan dengan cr yang bernilai skalar. Panjang bit-bit hasil penyebaran ini menjadi cr kali panjang bit-bit awal.



Gambar 2.4 *Spread Spectrum method*
Sumber: (<http://www.students.if.itb.ac.id>)

4. *Least Significant Bit (LSB)*

Metode yang digunakan untuk menyembunyikan pesan pada media digital tersebut berbeda-beda. Contohnya pada *file image* pesan dapat disembunyikan dengan menggunakan cara menyisipkannya pada bit rendah atau bit yang paling kanan (LSB) pada data *pixel* yang menyusun *file* tersebut. Seperti kita ketahui untuk *file bitmap* 24 bit maka setiap *pixel* (titik) pada gambar tersebut terdiri dari susunan tiga warna yaitu 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 (<http://doank29.multiply.com>).

Gambar 2.6 Proses *File Image* Menjadi Kumpulan *Pixel-Pixel*

Pada dasarnya sebuah gambar bitmap merupakan kumpulan dari titik-titik yang disebut *pixel*. *Pixel-pixel* disetiap gambar mempunyai nilai berbeda-beda.

R:181 G: 69 B: 85	R:180 G: 69 B: 85	R:177 G: 71 B: 85	R:176 G: 71 B: 85	R:176 G: 71 B: 85	R:176 G: 70 B: 84	R:178 G: 67 B: 83	R:178 G: 66 B: 82	R:176 G: 72 B: 83	R:175 G: 71 B: 82
R:181 G: 69 B: 85	R:180 G: 69 B: 85	R:177 G: 71 B: 85	R:176 G: 71 B: 85	R:175 G: 70 B: 84	R:175 G: 69 B: 83	R:177 G: 66 B: 82	R:177 G: 65 B: 81	R:174 G: 70 B: 81	R:176 G: 72 B: 83
R:179 G: 67 B: 83	R:178 G: 67 B: 83	R:175 G: 69 B: 83	R:174 G: 69 B: 83	R:174 G: 69 B: 83	R:174 G: 68 B: 82	R:176 G: 65 B: 81	R:176 G: 64 B: 80	R:171 G: 69 B: 80	R:175 G: 73 B: 84
R:177 G: 65 B: 81	R:176 G: 65 B: 81	R:174 G: 68 B: 82	R:174 G: 69 B: 83	R:174 G: 69 B: 83	R:174 G: 68 B: 82	R:177 G: 66 B: 82	R:177 G: 65 B: 81	R:171 G: 71 B: 81	R:175 G: 73 B: 84
R:176 G: 64 B: 80	R:176 G: 65 B: 81	R:174 G: 68 B: 82	R:174 G: 69 B: 83	R:175 G: 70 B: 84	R:176 G: 70 B: 84	R:179 G: 68 B: 84	R:180 G: 68 B: 84	R:175 G: 75 B: 85	R:173 G: 73 B: 83
R:176 G: 64 B: 80	R:176 G: 65 B: 81	R:175 G: 69 B: 83	R:175 G: 70 B: 84	R:176 G: 71 B: 85	R:178 G: 72 B: 86	R:181 G: 70 B: 86	R:182 G: 70 B: 86	R:178 G: 78 B: 88	R:173 G: 73 B: 83
R:170 G: 64 B: 78	R:172 G: 66 B: 80	R:173 G: 67 B: 81	R:171 G: 65 B: 79	R:170 G: 64 B: 78	R:171 G: 65 B: 79	R:176 G: 70 B: 84	R:180 G: 74 B: 88	R:178 G: 72 B: 82	R:179 G: 73 B: 83
R:173 G: 67 B: 81	R:175 G: 69 B: 83	R:176 G: 70 B: 84	R:175 G: 69 B: 83	R:173 G: 67 B: 81	R:174 G: 68 B: 82	R:177 G: 71 B: 85	R:180 G: 74 B: 88	R:173 G: 67 B: 77	R:174 G: 68 B: 78
R:172 G: 66 B: 80	R:174 G: 68 B: 82	R:175 G: 69 B: 83	R:175 G: 69 B: 83	R:174 G: 68 B: 82	R:173 G: 67 B: 81	R:174 G: 68 B: 82	R:175 G: 69 B: 83	R:170 G: 64 B: 76	R:172 G: 66 B: 78
R:173 G: 67 B: 81	R:174 G: 68 B: 82	R:176 G: 70 B: 84	R:177 G: 71 B: 85	R:175 G: 69 B: 83	R:174 G: 68 B: 82	R:172 G: 66 B: 80	R:171 G: 65 B: 79	R:172 G: 66 B: 78	R:173 G: 67 B: 79

Gambar 2.7 Nilai-Nilai *Pixel* Dalam Suatu *File Image*

Misalkan diambil nilai dari beberapa pixel pada gambar di atas, dimana nilai pixel dikonversikan dahulu ke dalam biner untuk menyisipkan sebuah karakter “R” = 01010010 di mana 01010010 adalah kode biner untuk 82 yang merupakan kode ASCII karakter “R”.

(00100111	11101001	11001000)
(00100111	11001000	11101001)
(11001000	00100111	11101001)

Segmen citra sebelum disisipkan

(00100110	11101001	11001000)
(00100111	11001000	11101000)
(11001001	00100110	11101001)

Segmen citra sesudah disisipkan “R”

Terlihat di atas perubahan pada contoh segmen data citra yang terdapat pada bit-bit yang paling kanan, setelah disisipkan ‘01010010’ sebagai data yang disembunyikan, bahwa perubahan bit hanya terjadi pada sisi yang paling kanan dari 8 bit yang ada.

Steganografi dengan metode LSB juga hanya mampu menyimpan informasi dengan ukuran yang sangat terbatas. Misalnya suatu citra 24-bit (R=8-bit, G=8-bit, B=8-bit) digunakan sebagai wadah untuk menyimpan data berukuran 100 bit, jika masing-masing komponen warnanya (RGB) digunakan satu *pixel* untuk menyimpan informasi rahasia tersebut, maka setiap *pixel*nya disimpan 3 bit informasi, sehingga setidaknya dibutuhkan citra wadah berukuran 34 *pixel* atau setara $34 \times 3 \times 8 = 816$ bit (8 kali lipat). Jadi suatu citra 24-bit jika digunakan untuk menyimpan informasi rahasia hanya mampu menampung informasi maksimum berukuran $1/8$ dari ukuran citra penampung tersebut (<http://webmail.informatika.org>).

Tabel 2.1 Keuntungan dan Kelemahan Metode LSB

No.	Keuntungan	Kelemahan
1.	Mudah diimplementasikan.	Tidak tahan terhadap pengubahan (modifikasi) terhadap <i>cover object</i> .
2.	Proses <i>encoding</i> cepat.	Mudah dihapus karena lokasi penyisipan diketahui (bit LSB).

3.	Mengeliminasi tingkat kecurigaan seseorang.	Besar pesan sangat tergantung dari media yang dipergunakan.
----	---	---

Sumber: (<http://haryanto.staff.gunadarma.ac.id>)

2.5 ASCII

Kode Standar Amerika untuk Pertukaran Informasi atau ASCII (*American Standard Code for Information Interchange*) merupakan suatu standar internasional dalam kode huruf dan simbol seperti *Hex* dan *Unicode* tetapi ASCII lebih bersifat universal, contohnya 124 adalah untuk karakter "|", selalu digunakan oleh komputer dan alat komunikasi lain untuk menunjukkan teks. Kode ASCII sebenarnya memiliki komposisi bilangan biner sebanyak 8 bit. Dimulai dari 00000000 hingga 11111111. Total kombinasi yang dihasilkan sebanyak 256, dimulai dari kode 0 hingga 255 dalam sistem bilangan Desimal (www.bobbemer.com).

Tabel 2.2 ASCII

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	\$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(72	48	H	104	68	h
9	09	Horizontal tab	41	29)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	
29	1D	Group separator	61	3D	=	93	5D]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	Ł	224	E0	α
129	81	ù	161	A1	í	193	C1	ł	225	E1	β
130	82	é	162	A2	ó	194	C2	Ť	226	E2	Γ
131	83	â	163	A3	ú	195	C3	ţ	227	E3	π
132	84	ä	164	A4	ñ	196	C4	—	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	†	229	E5	σ
134	86	å	166	A6	ª	198	C6	‡	230	E6	μ
135	87	ç	167	A7	º	199	C7	‡	231	E7	ι
136	88	ê	168	A8	¿	200	C8	Ł	232	E8	Φ
137	89	ë	169	A9	ƒ	201	C9	Ŧ	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	Ł	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	Ŧ	235	EB	δ
140	8C	î	172	AC	¾	204	CC	‡	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	=	237	ED	ø
142	8E	Ä	174	AE	«	206	CE	‡	238	EE	ε
143	8F	Å	175	AF	»	207	CF	Ł	239	EF	Π
144	90	É	176	B0	█	208	DO	Ł	240	FO	≡
145	91	æ	177	B1	█	209	D1	Ŧ	241	F1	±
146	92	Æ	178	B2	█	210	D2	π	242	F2	≥
147	93	ô	179	B3		211	D3	Ł	243	F3	≤
148	94	ö	180	B4	†	212	D4	Ł	244	F4	[
149	95	ò	181	B5	‡	213	D5	Ŧ	245	F5]
150	96	û	182	B6	‡	214	D6	π	246	F6	÷
151	97	ù	183	B7	π	215	D7	‡	247	F7	≈
152	98	ÿ	184	B8	Ŧ	216	D8	‡	248	F8	°
153	99	Ö	185	B9	‡	217	D9	Ŧ	249	F9	•
154	9A	Ü	186	BA	‡	218	DA	Ŧ	250	FA	·
155	9B	ø	187	BB	Ŧ	219	DB	█	251	FB	√
156	9C	£	188	BC	Ł	220	DC	█	252	FC	π
157	9D	¥	189	BD	Ł	221	DD	█	253	FD	ε
158	9E	€	190	BE	Ŧ	222	DE	█	254	FE	■
159	9F	f	191	BF	Ŧ	223	DF	█	255	FF	□

Sumber: (<http://www.google.co.id/>)

2.6 Multimedia

Multimedia adalah suatu istilah umum yang sering digunakan untuk menjelaskan penyebaran informasi, aplikasi, presentasi dan dokumen lain yang menggunakan gabungan kombinasi dari teks, grafik, animasi, dan video (Anderleigh, 1996).

Menurut (Hoffstetter, 2001), multimedia adalah penggunaan komputer untuk menyajikan dan mengkombinasikan teks, grafik, audio dan video dengan alat bantu (*tool*) yang memungkinkan pengguna (*user*) untuk melayani, berinteraksi, menciptakan dan berkomunikasi.

Multimedia juga dapat didefinisikan sebagai gabungan beberapa elemen, yaitu:

1. Elemen Teks

Terdiri dari huruf, nomor, dll. Aplikasi dari elemen teks adalah *Word Processing*, meliputi; *Microsoft word*, *Notepad*, dan *Office org*.

2. Elemen Grafik

Terdiri dari objek yang berupa garis- garis, kotak, bulatan, *shading*, *fill colours*. Aplikasi dari elemen ini adalah Draw Program, meliputi; *Corel draw*, *Adobe illustrator*, dan *Adobe flash*.

3. Elemen Gambar (*image*)

Terdiri dari gambar statik hasil kombinasi banyak *pixel*. Aplikasi elemen ini adalah Paint Program, meliputi; *Adobe photoshop*, *Scanner maching*, dan *ms. Paint*.

4. Elemen Audio

Terdiri dari *Sound* (suara) Seperti musik player, dll. Aplikasi elemen ini adalah Recording, meliputi; *Cooledit pro 2.0* dan komponen Player, meliputi; *Winamp, Jet audio, Real player*, dll.

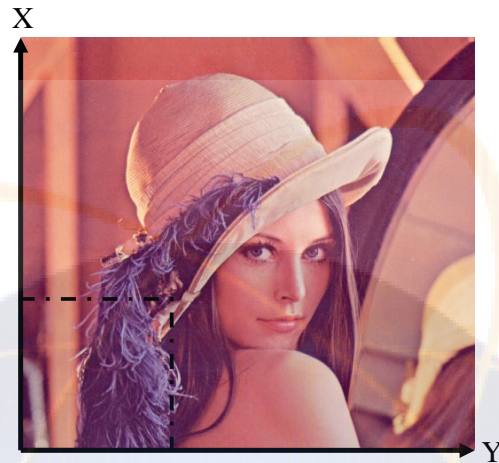
5. Elemen Visual

Terdiri dari susunan gambar yang digerakkan. Aplikasi dari elemen ini adalah Video Editing, meliputi; *Adobe premiere, Canopus edius*, dan *Pinnacle studios* serta Animasi, meliputi; *Adobe after effect, Adobe potoshop, Adobe Flash* (<http://ilmucerdas.wordpress.com>).

2.7 Citra Digital

2.7.1 Pengertian Citra Digital

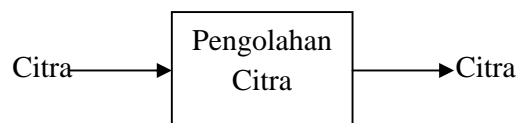
Citra adalah kumpulan *pixel-pixel* yang disusun dalam larik dua dimensi. *Pixel* (0,0) terletak pada sudut kiri atas pada citra, indeks X bergerak ke kanan dan indeks Y bergerak ke bawah (Ahmad, 2005). Citra digital dapat diartikan juga sebagai fungsi dua variabel, $f(x,y)$, di mana x dan y adalah koordinat spasial dan nilai $f(x,y)$ adalah intensitas citra pada koordinat tersebut, hal tersebut diilustrasikan pada Gambar 2.8. Teknologi dasar untuk menciptakan dan menampilkan warna pada citra digital berdasarkan pada penelitian bahwa sebuah warna merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau, dan biru (*Red, Green, Blue RGB*) (<http://www.ittelkom.ac.id>).



Gambar 2.8 Koordinat Spasial dan Nilai $f(x,y)$
 Sumber: (www.jcatki.no-ip.org)

2.7.2 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, maksudnya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas lebih baik dari pada citra masukan (Munir, 2004). Di dalam steganografi citra keluaran yang dimaksud adalah *stegotext* atau *stego-object*.



Gambar 2.9 Bagan Pengolahan citra
 Sumber: (Munir, 2004)

Secara umum tahapan pengolahan citra digital meliputi akusisi citra, peningkatan kualitas citra, segmentasi citra, representasi dan uraian, pengenalan dan interpretasi.

1. Akusisi citra

Pengambilan data dapat dilakukan dengan menggunakan berbagai media seperti kamera analog, kamera digital, handycamp, scanner, optical reader dan sebagainya. Citra yang dihasilkan belum tentu data digital, sehingga perlu didigitalisasi.

2. Peningkatan kualitas citra

Pada tahap ini dikenal dengan *pre-processing* dimana dalam meningkatkan kualitas citra dapat meningkatkan kemungkinan dalam keberhasilan pada tahap pengolahan citra digital berikutnya.

3. Segmentasi citra

Segmentasi bertujuan untuk memilih dan mengisolasi (memisahkan) suatu objek dari keseluruhan citra. Segmentasi terdiri dari *downsampling*, penapisan dan deteksi tepian. Tahap *downsampling* merupakan proses untuk menurunkan jumlah *pixel* dan menghilangkan sebagian informasi dari citra. Dengan resolusi citra yang tetap, *downsampling* menghasilkan ukuran citra yang lebih kecil. Tahap segmentasi selanjutnya adalah penapisan dengan filter median, hal ini dilakukan untuk menghilangkan derau yang biasanya muncul pada frekuensi tinggi pada spektrum citra. Pada penapisan dengan filter median, gray level citra pada setiap *pixel* digantikan dengan nilai median dari gray level pada *pixel* yang terdapat pada window filter. Tahap yang terakhir pada proses segmentasi yaitu deteksi tepian.

Pendekatan algoritma Canny dilakukan berdasarkan konvolusi fungsi citra dengan operator Gaussian dan turunan-turunannya. Pendeteksi tepi ini dirancang untuk merepresentasikan sebuah tepian yang ideal, dengan ketebalan yang diinginkan. Secara umum, proses segmentasi sangat penting dan secara langsung akan menentukan keakurasian sistem dalam proses identifikasi iris mata.

4. Representasi dan Uraian

Representasi mengacu pada data konversi dari hasil segmentasi ke bentuk yang lebih sesuai untuk proses pengolahan pada komputer. Keputusan pertama yang harus sudah dihasilkan pada tahap ini adalah data yang akan diproses dalam batasan-batasan atau daerah yang lengkap. Batas representasi digunakan ketika penekanannya pada karakteristik bentuk luar, dan area representasi digunakan ketika penekanannya pada karakteristik dalam, sebagai contoh tekstur. Setelah data telah direpresentasikan ke bentuk tipe yang lebih sesuai, tahap selanjutnya adalah menguraikan data.

5. Pengenalan dan Interpretasi

Pengenalan pola tidak hanya bertujuan untuk mendapatkan citra dengan suatu kualitas tertentu, tetapi juga untuk mengklasifikasikan bermacam-macam citra. Dari sejumlah citra diolah sehingga citra dengan ciri yang sama akan dikelompokkan pada suatu kelompok tertentu. Interpretasi meliputi penekanan dalam mengartikan objek yang dikenali (<http://www.itelkom.ac.id>).

2.7.3 Perbandingan *File* Gambar BMP (Bitmap) dengan JPG, GIF, atau PNG

Tipe *file* BMP umum digunakan pada sistem operasi Windows dan OS/2. Kelebihan tipe *file* BMP adalah dapat dibuka oleh hampir semua program pengolah gambar. Baik *file* BMP yang terkompresi maupun tidak terkompresi, *file* BMP memiliki ukuran yang jauh lebih besar daripada tipe-tipe yang lain.

File BMP cocok digunakan untuk:

- *desktop background* di *windows*.
- sebagai gambar sementara yang mau diedit ulang tanpa menurunkan kualitasnya.

File BMP tidak cocok digunakan untuk:

- web atau blog, perlu dikonversi menjadi JPG, GIF, atau PNG.
- disimpan di *harddisk/flashdisk* tanpa di ZIP/RAR, kecuali *space* tidak masalah bagi Anda.

Tipe *file* JPG sangat sering digunakan untuk *web* atau *blog*. *File* JPG menggunakan teknik kompresi yang menyebabkan kualitas gambar turun (*lossy compression*). Setiap kali menyimpan ke tipe JPG dari tipe lain, ukuran gambar biasanya mengecil, tetapi kualitasnya turun dan tidak dapat dikembalikan lagi. Ukuran *file* BMP dapat turun menjadi sepersepuluhnya setelah dikonversi menjadi JPG. Meskipun dengan penurunan kualitas gambar, pada gambar-gambar tertentu (misalnya pemandangan), penurunan kualitas gambar hampir tidak terlihat mata.

File JPG cocok digunakan untuk:

- gambar yang memiliki banyak warna, misalnya foto wajah dan pemandangan.

- gambar yang memiliki gradien, misalnya perubahan warna yang perlahan-lahan dari merah ke biru.

File JPG tidak cocok digunakan untuk:

- gambar yang hanya memiliki warna sedikit seperti kartun atau komik.
- gambar yang memerlukan ketegasan garis seperti logo.

Tipe *file* GIF memungkinkan penambahan warna transparan dan dapat digunakan untuk membuat animasi sederhana, tetapi saat ini standar GIF hanya maksimal 256 warna saja. *File* ini menggunakan kompresi yang tidak menghilangkan data (*lossles compression*) tetapi penurunan jumlah warna menjadi 256 sering membuat gambar yang kaya warna seperti pemandangan menjadi tidak realistis.

File GIF cocok digunakan untuk:

- gambar dengan jumlah warna sedikit (dibawah 256).
- gambar yang memerlukan perbedaan warna yang tegas seperti logo tanpa gradien.
- gambar animasi sederhana seperti banner-banner iklan, header, dan sebagainya.

- print shoot (hasil dari print screen) dari program-program *simple* dengan jumlah warna sedikit.

File GIF tidak cocok digunakan untuk:

- gambar yang memiliki banyak warna seperti pemandangan.
- gambar yang di dalamnya terdapat warna gradien atau semburat.

Tipe *file* PNG merupakan solusi kompresi yang powerfull dengan warna yang lebih banyak (24 bit RGB + alpha). Berbeda dengan JPG yang menggunakan teknik kompresi yang menghilangkan data, *file* PNG menggunakan kompresi yang tidak menghilangkan data (lossles compression). Kelebihan *file* PNG adalah adanya warna transparan dan alpha. Warna alpha memungkinkan sebuah gambar transparan, tetapi gambar tersebut masih dapat dilihat mata seperti samar-samar atau bening. *File* PNG dapat diatur jumlah warnanya 64 bit (*true color* + alpha) sampai *indexed color* 1 bit. Dengan jumlah warna yang sama, kompresi *file* PNG lebih baik daripada GIF, tetapi memiliki ukuran *file* yang lebih besar daripada JPG. Kekurangan tipe PNG adalah belum populer sehingga sebagian browser tidak mendukungnya.

File PNG cocok digunakan untuk:

- gambar yang memiliki warna banyak.
- gambar yang mau diedit ulang tanpa menurunkan kualitas.

File PNG tidak cocok digunakan untuk:

- gambar yang jika dikompres dengan JPG hampir-hampir tidak terlihat penurunan kualitasnya (<http://www.tutorialgratis.net>).

2.8 DELPHI

2.8.1 Pengertian DELPHI

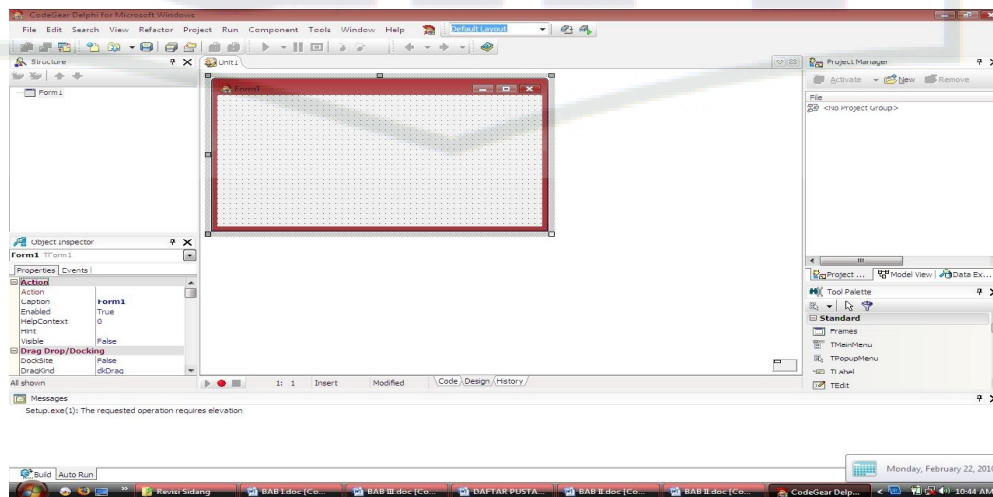
Bahasa pemrograman *Delphi* yang termasuk dalam salah satu bahasa pemrograman visual adalah generasi lanjut pemograman Pascal. Adapun, rilis (versi Delphi pertama) adalah tahun 1995, kemudian berlanjut sampai rilis ketujuh

pada tahun 2002 dan kini rilis terbarunya adalah Delphi 8 dan 2005. Pemrograman Delphi sendiri dibuat oleh Borland International Corporation dan berjalan di atas platform (sistem operasi) *Windows*, sedangkan sebagai pengetahuan yang berjalan di atas platform (sistem operasi) *Linux* adalah *Kylix*, yang merupakan saudara kembar pemrograman Delphi (Malik, 2006).

Borland Delphi atau lebih sering disebut dengan Delphi merupakan salah satu produk Borland yang sudah cukup terkenal. Kemampuannya untuk membuat aplikasi-aplikasi, baik aplikasi *database* maupun *non-database* sudah tidak perlu diragukan lagi, didukung dengan fasilitas yang cukup lengkap (Prasetyo, 2004).

2.8.2 Kelebihan Delphi

Kadang penulis berpikir, mengapa tidak memakai bahasa pemrograman visual lain (*Microsoft Visual Basic*, *C++ Builder*, atau lainnya)? Jawabannya karena objek dasar Delphi adalah bahasa pemrograman Pascal di mana kita sudah mengetahui bersama bahwa pemrograman Pascal sudah dikenal oleh kalangan masyarakat Indonesia, dengan survei dan berbagai sumber (Prasetyo, 2004).



Gambar 2.10 Tampilan *Form* Awal *Delphi* 2007 Win32

2.9 Perancangan Program

Di dalam merancang suatu program, dapat menggunakan beberapa alat bantu yaitu, metode pengembangan sistem, *flowchart*, dan *state transition diagram* (STD).

2.9.1 Model-model Pengembangan Sistem

Dalam sebuah perancangan perangkat lunak diperlukan model-model proses atau paradigma rekayasa perangkat lunak berdasarkan sifat aplikasi proyeknya, metode dan alat bantu yang dipakai, dan kontrol serta penyampaian yang dibutuhkan. Ada beberapa model dari proses perangkat lunak, yaitu: Model *Sekuensial Linear*, Model Prototipe, Model RAD (*Rapid Application Development*), Model Evolusioner, dan Model Formal. Untuk menyelesaikan masalah di dalam sebuah sistem harus dilakukan penggabungan strategi pengembangan yang melingkupi lapisan proses, metode, dan alat-alat bantu serta fase-fase generik (Pressman, 2002). Pada Table 2.3 dijelaskan beberapa metode dan perbedaannya.

Table 2.3 Beberapa Metode dan Perbedaannya

Metode	Kelebihan	Kekurangan	Penggunaan secara umum
<i>Sequensial Linier</i> (waterfall)	Metode ini baik digunakan untuk kebutuhan yang sudah diketahui dengan baik.	Iterasi yang sering terjadi menyebabkan masalah baru. bagi pelanggan sulit menentukan kebutuhan secara eksplisit dan harus sabar karena memakan waktu yang lama.	<i>Waterfall</i> bekerja dengan baik pada proyek skala kecil.

<i>Prototype</i>	Metode ini cukup efektif dengan mendapatkan kebutuhan dan aturan yang jelas dan pelanggan bisa langsung melihat sistem yang sebenarnya.	Pengembang kadang-kadang membuat implementasi sembarang, karena ingin working version selesai dengan cepat.	Prototyping dapat bekerja dengan baik jika ada kerjasama yang baik antara pengembang dengan pengguna
RAD	Metode ini lebih cepat dari <i>waterfall</i> jika kebutuhan dan batasan proyek sudah diketahui dengan baik. Dan bisa untuk dimodularisasi.	Karena proyek dipecah menjadi beberapa bagian, maka dibutuhkan banyak orang untuk membentuk suatu tim. Karena komponen-komponen yang sudah ada, fasilitas-fasilitas pada tiap komponen belum tentu digunakan seluruhnya sehingga kualitas program bisa menurun.	RAD cocok untuk aplikasi yang tidak mempunyai resiko teknis yang tinggi. RAD cocok untuk proyek yang memiliki SDM yang baik dan sudah berpengalaman.
<i>Iterative</i>	Fase desain, pengkodean, pengujian lebih cepat.	butuh waktu yang banyak untuk menganalisis dan terlalu banyak langkah yang dibutuhkan model	hanya cocok untuk <i>software</i> berskala besar
Spiral	Model ini digunakan untuk sistem skala besar.membutuhkan Konsiderasi langsung terhadap resiko teknis, sehingga dapat mengurangi terjadinya resiko yang lebih besar.	Resiko utama tidak ditemukan, maka masalah bisa muncul kemudian. Sehingga membutuhkan kemampuan manajemen dan perkiraan resiko (<i>risk assessment</i>) yang cukup tinggi.	Hanya cocok untuk <i>software</i> skala besar.

Sumber: (Pressman, 2002)

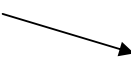

2.9.2 Diagram Alur (*Flowchart*)



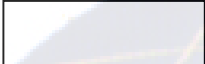
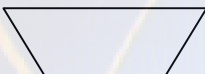
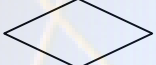





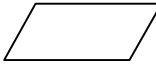
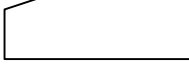
Komputer membutuhkan hal-hal yang terperinci, maka bahasa pemrograman bukan merupakan alat yang bisa dikatakan baik untuk merancang sebuah algoritma awal. Alat yang banyak dipakai untuk membuat algoritma adalah diagram alir. Diagram alur dapat menunjukkan secara jelas arus pengendalian algoritma, yakni bagaimana rangkaian pelaksanaan kegiatan. Suatu diagram alir memberikan gambaran dua dimensi berupa simbol-simbol grafis.

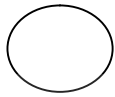
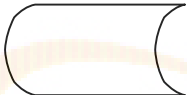


Flowchart program merupakan bagan alir yang menggambarkan urutan logika dari suatu prosedur pemecahan masalah. Tujuan utama dari penggunaan *flowchart* adalah untuk menggambarkan suatu tahapan penyelesaian masalah secara sederhana, terurai, rapi dan jelas dengan menggunakan simbol-simbol standar.

Dalam *flowchart* dikenal dua macam bentuk, yaitu Sistem *Flowchart* dan Program *Flowchart*. Sistem *Flowchart* menggambarkan tahapan proses dari suatu sistem, termasuk sistem multimedia. Sedangkan Program *Flowchart* menggambarkan urutan-urutan intruksi dari suatu program komputer (Al-Bahra' 2005). Simbol-simbol *flowchart* terlihat pada Tabel 2.4

Tabel 2.4 Simbol-simbol *Flowchart*

A. SIMBOL ARUS DAN ARAH	
SIMBOL	KEGUNAAN
 Simbol Arus/ Flow	Untuk menyatakan jalannya arus suatu proses.
 Simbol <i>Communication Link</i>	Untuk menyatakan bahwa adanya transisi suatu data/ informasi dari satu lokasi ke lokasi lainnya.
	Untuk menyatakan sambungan dari satu

Simbol		Connector	proses ke proses lainnya dalam halaman/ lembar yang sama.
		Simbol <i>Off-Line Connector</i>	Untuk menyatakan sambungan dari satu proses ke proses lainnya dalam halaman yang berbeda.
B. SIMBOL PROSES			
SIMBOL		KEGUNAAN	
		Simbol Proses/ <i>Offline Connector</i>	Untuk menyatakan sambungan dari suatu proses ke proses lainnya dalam halaman yang berbeda.
		Simbol Manual	Untuk menyatakan suatu tindakan (proses) yang tidak dilakukan oleh komputer (manual).
		Simbol <i>Decission/ Logika</i>	Untuk menunjukkan suatu kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban, ya/ tidak.
		Simbol <i>Predefined Process</i>	Untuk menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal.
		Simbol Terminal	Untuk menyatakan permulaan atau akhir suatu program.
		Simbol <i>Keying Operation</i>	Untuk menyatakan segala jenis operasi yang diproses menggunakan suatu mesin yang mempunyai <i>keyboard</i> .
		Simbol <i>off-line storage</i>	Untuk menunjukkan bahwa data dalam symbol ini akan disimpan ke suatu media tertentu.
		Simbol Manual Input	Untuk pemasukan data secara manual <i>on-line keyboard</i> .
C. SIMBOL INPUT-OUTPUT			
SIMBOL		KEGUNAAN	
		Simbol Input-Output	Simbol yang menyatakan proses input dan output tanpa tergantung dengan jenis peralatannya.
		Simbol <i>Punched Card</i>	Simbol yang menyatakan input berasal dari kartu atau output ditulis ke kartu.

 Simbol <i>Magnetic Tape Unit</i>	Simbol yang menyatakan input berasal dari pita magnetik atau output disimpan ke pita magnetic.
 Simbol <i>Disk Storage</i>	Untuk menyatakan input berasal dari disk atau output disimpan ke disk.
 Simbol <i>Document</i>	Untuk menyatakan input berasal dari kartu atau output disimpan ke pita magnetic.
 Simbol <i>Display</i>	Untuk menyatakan peralatan output yang digunakan yaitu layar (video computer).

Sumber: (Al-Bahra, 2005)

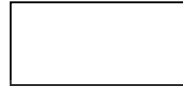
2.9.3 State Transition Diagram (STD)

Menunjukkan bagaimana sistem bertingkah laku sebagai akibat dari kejadian *eksternal*. *State Transition Diagram* (Diagram transisi keadaan) merupakan suatu modeling tool yang menggambarkan *Time Depend Behavior* dari suatu sistem (Al-Bahra, 2005).

Pada mulanya model *State Transition Diagram* ini hanya digunakan untuk menggambarkan suatu sistem yang bersifat *real time*. Ada dua cara kerja sistem ini yaitu pasif dan aktif. STD ini hanya digunakan untuk menuliskan urutan dan pergantian dari layar yang dapat terjadi, ketika pengguna sistem berada pada terminal.

1. Keadaan Sistem (*State*)

Setiap kotak mewakili suatu keadaan dimana sistem mungkin berada didalamnya. Seperti terlihat pada Gambar 2.11 *state* disimbolkan dengan simbol segi empat.



Gambar 2.11 Simbol State

Sumber: (Al-Bahra, 2005)

2. Perubahan Sistem (*Transition State*)

Simbol ini digunakan untuk menghubungkan satu keadaan dengan keadaan lain. Simbol ini digunakan jika sistem memiliki transisi dalam perilakunya. Gambar 2.12 menunjukkan *transition state*.

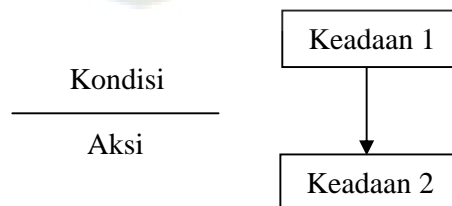


Gambar 2.12 Simbol *Transition State*

Sumber: (Al-Bahra, 2005)

3. Kondisi dan Aksi

Untuk melengkapi STD, dibutuhkan dua hal tambahan: yaitu kondisi sebelum keadaan berubah dan aksi dari pemakai untuk mengubah keadaan. Gambar 2.13 adalah ilustrasi dari kondisi dan aksi yang ditampilkan di sebelah anak panah yang menghubungkan dua keadaan (Al-Bahra, 2005).



Gambar 2.13 Simbol Kondisi dan Aksi

Sumber: (Al-Bahra, 2005)

2.10 Konsep RAD

2.10.1 Definisi RAD

RAD (*Rapid Application Development*) adalah sistem yang menggunakan teknik terstruktur, prototyping, dan JAD (*Joint Application Development*) untuk mengembangkan sistem secara cepat (Whitten, *et al.*2007).

Teknik terstruktur adalah sebuah teknik desain sistem yang menguraikan proses–proses sistem menjadi komponen–komponen yang dapat dikelola (Whitten, *et al.*2007).

Prototyping adalah teknik untuk membangun dengan cepat sebuah model sistem informasi yang fungsional tapi tidak lengkap dengan menggunakan peralatan pengembangan aplikasi (Whitten, *et al.*2007).

JAD (*joint application development*) adalah sebuah teknik yang melengkapi analisis sistem dan teknik desain lain dengan cara menekankan *participative development* diantara *system owner*, *users*, *designers*, dan *builder* (Whitten, *et al.*2007).

2.10.2 Tahapan – Tahapan RAD

Pengembangan sistem dalam penelitian ini menggunakan model RAD (*Rapid Application Development*), yaitu: (Kendall, 2005).

1. Tahap perencanaan syarat–syarat.

Dalam fase ini, pengguna dan penganalisis bertemu untuk mengidentifikasi tujuan–tujuan aplikasi atau sistem serta untuk mengidentifikasi syarat–syarat informasi yang ditimbulkan dari tujuan–tujuan tersebut. Fase ini memerlukan peran aktif mendalam

dari kedua kelompok tersebut, tidak hanya menunjukkan proposal atau dokumen. Selain itu, juga melibatkan pengguna dari beberapa level yang berada dalam organisasi. Orientasi dalam fase ini ialah menyelesaikan problem–problem perusahaan. Meskipun teknologi informasi dan sistem bisa mengarahkan sebagian sistem yang diajukan, fokusnya akan selalu tetap pada upaya pencapaian tujuan–tujuan perusahaan.

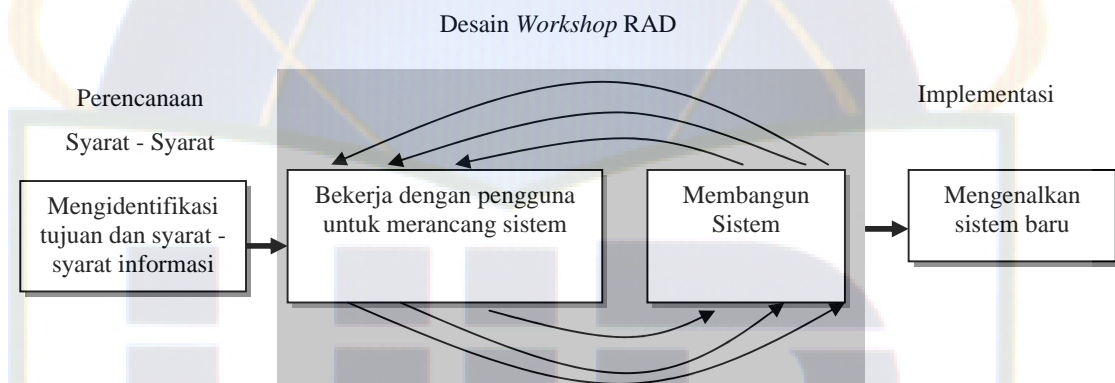
2. Tahap desain *workshop* RAD.

Fase ini adalah fase untuk merancang dan memperbaiki yang bisa digambarkan sebagai *workshop*. Saat membayangkan sebuah *workshop*, partisipasinya sangat intens, tidak pasif, dan biasanya bertahan. Biasanya para partisipan duduk mengitari meja bulat atau di kursi–kursi yang diatur membentuk huruf U dilengkapi meja sehingga masing–masing dapat melihat satu sama lain.

Selama *workshop* desain RAD, pengguna merespons *working prototype* yang ada dan penganalisis memperbaiki modul–modul yang dirancang berdasarkan respons pengguna. Format *workshop* sangat mengagumkan dan mampu memberi dorongan, dan jika pengguna atau penganalisis yang berpengalaman, tidak diragukan lagi bahwa usaha kreatif ini dapat mendorong pengembangan sampai pada tingkat terakselerasi.

3. Tahap implementasi.

Dalam Gambar 2.14 ditunjukkan bahwa dapat terlihat, penganalisis bekerja dengan para pengguna secara intens selama *workshop* untuk merancang aspek-aspek bisnis dan nonteknis dari perusahaan. Segera sesudah aspek-aspek ini disetujui dan sistem – sistem dibangun dan disaring, sistem-sistem baru atau bagian dari sistem diuji coba dan kemudian diperkenalkan kepada organisasi (Kendall, 2005).



Gambar 2.14 Tahapan-Tahapan RAD
Sumber: (Kendall, 2005)

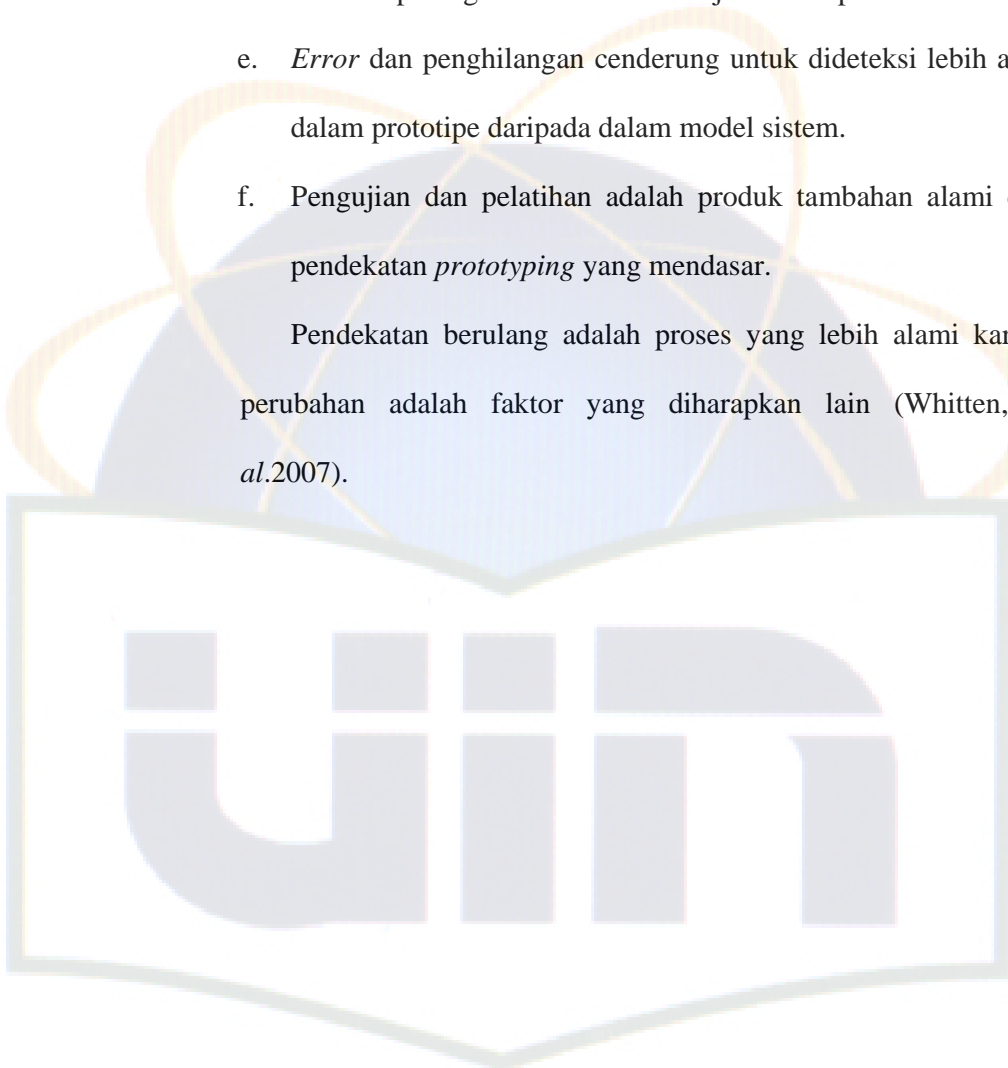
2.10.3 Keunggulan RAD

Kelebihan menggunakan metode RAD adalah:

- Berguna untuk proyek-proyek tempat persyaratan-persyaratan pengguna tidak pasti dan tidak tepat.
- Mendorong pengguna aktif dan partisipasi manajemen.
- Proyek-proyek memiliki visibilitas dan dukungan lebih tinggi karena keterlibatan pengguna yang ekstensif selama proses.

- d. Para pengguna dan manajemen melihat solusi–solusi yang berbasis perangkat lunak dan bekerja lebih cepat.
- e. *Error* dan penghilangan cenderung untuk dideteksi lebih awal dalam prototipe daripada dalam model sistem.
- f. Pengujian dan pelatihan adalah produk tambahan alami dari pendekatan *prototyping* yang mendasar.

Pendekatan berulang adalah proses yang lebih alami karena perubahan adalah faktor yang diharapkan lain (Whitten, *et al.*2007).





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.

BAB III

METODE PENELITIAN

Pada penyusunan skripsi ini, diperlukan data-data informasi sebagai bahan yang dapat mendukung kebenaran materi uraian pembahasan. Untuk menyelesaikan masalah yang ada dalam sebuah perancangan perangkat lunak ada beberapa tahap yang harus dilakukan. Dalam bab ini menguraikan tentang metode pengumpulan data dan metode pengembangan sistem yang digunakan peneliti. Metode penelitian yang digunakan adalah:

3.1 Metode Pengumpulan Data

1. Studi Pustaka

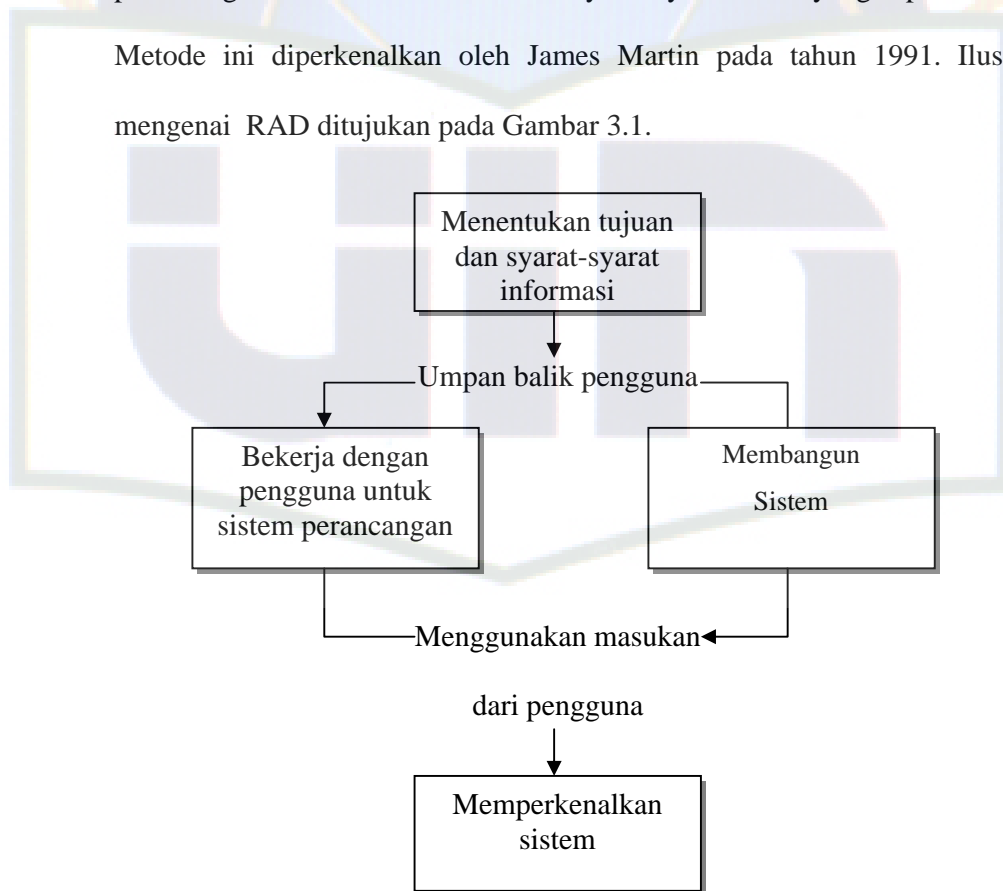
Dengan metode ini, penulis mendapatkan informasi apa saja yang berkaitan dengan steganografi melalui buku-buku referensi dan mencari melalui berbagai situs-situs di internet yang berkaitan, sehingga penulis dapat mengumpulkan data dan informasi yang diinginkan.

2. Studi Literatur

Penulis mencoba mencari perbandingan dengan studi sejenis dari beberapa penulisan di beberapa karya ilmiah, seperti jurnal dan skripsi.

3.2 Metode Pengembangan Sistem

Dalam menyusun tugas akhir ini penulis menggunakan metodologi pengembangan sistem *Rapid Application Development* (RAD). Model Rapid Application Development adalah pendekatan berorientasi objek yang digunakan terhadap pengembangan sistem yang mencakup suatu metode pengembangan perangkat-perangkat lunak. Model RAD adalah proses pengembangan perangkat lunak sekuensial linear yang menekankan siklus pengembangan yang pendek. Hal ini akan mempersingkat waktu dalam perancangan dan berusaha memenuhi syarat-syarat bisnis yang cepat berubah. Metode ini diperkenalkan oleh James Martin pada tahun 1991. Ilustrasi mengenai RAD ditujukan pada Gambar 3.1.



Gambar 3.1 Skema Sistem Model RAD

Dari Gambar 3.1. tersebut terlihat bahwa metode pengembangan sistem *Rapid Application Design* (RAD) terdiri dari tiga tahapan yaitu perencanaan syarat-syarat, desain *workshop* RAD dan implementasi.

3.2.1 Fase Perencanaan Syarat-Syarat

Pada fase penulis melakukan analisis kebutuhan dengan melakukan perbandingan terhadap beberapa aplikasi steganografi yang sudah ada, serta memutuskan fungsi apa yang harus difiturkan oleh aplikasi tersebut. Hasil analisis digunakan untuk mengidentifikasi tujuan dan syarat-syarat untuk mencapai tujuan, yaitu membuat aplikasi steganografi pada citra *digital* untuk memberikan solusi pengamanan pada pengiriman pesan. Untuk memudahkan maka harus didefinisikan sebagai berikut:

1. Apa saja yang menjadi input
2. Bagaimana proses digambarkan dalam alur dan basis aturannya.
3. Apa yang menjadi output atau hasilnya.

3.2.2 Proses Desain RAD (*RAD Design Workshop*)

Pada fase ini penulis melakukan perancangan proses dan perancangan antarmuka dari aplikasi, yaitu:

1. Perancangan Proses

Pada tahap ini akan dilakukan perancangan, evaluasi dan memperbaiki sistem sesuai dengan kebutuhan. Agar sistem yang sedang dibuat dapat dimanfaatkan secara optimal. Perancangan proses pada aplikasi ini digambarkan oleh *flowchart*. Pada fase ini juga dilakukan

pengkodean terhadap rancangan-rancangan yang telah didefinisikan.

Pengkodean yang dilakukan menggunakan *tools Delphi 2007 win32*.

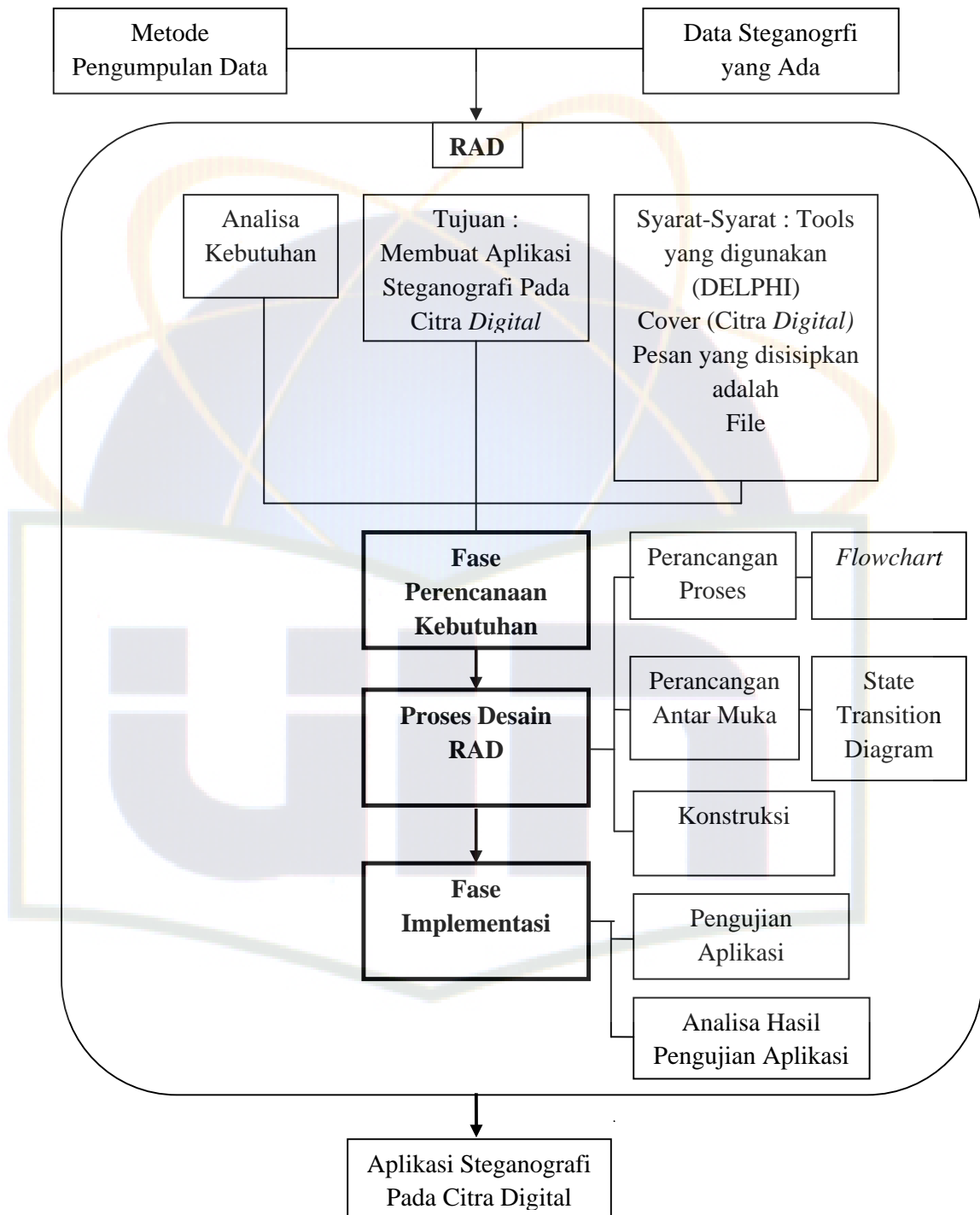
2. Perancangan Antar Muka Pemakai (*User Interface*)

Pada tahap ini akan dilakukan perancangan antar muka pemakai yang memberikan fasilitas komunikasi antar pemakai dan sistem, memberikan berbagai fasilitas informasi dan berbagai keterangan yang bertujuan untuk membantu mengarahkan alur penelusuran masalah sampai ditemukan solusi. Perancangan antar muka pemakai (*user interface*) pada aplikasi ini digambarkan oleh *state transition diagram*.

3.2.3 Fase Implementasi (*Implementation Phase*)

Pada fase ini penulis melakukan pengujian dan analisa terhadap aplikasi. Pengujian dilakukan dengan cara membandingkan kualitas dan besar data citra *digital* sebelum dan sesudah proses penyisipan pesan, serta seberapa besar tingkat keamanannya sehingga informasi rahasia benar– benar terjaga kerahasiannya dengan menggunakan teknik steganografi sebagai suatu solusi pengamanan pesan.

Skema dari metodologi penelitian yang dilakukan dalam pengembangan aplikasi steganografi pada citra *digital* ini ditunjukkan pada Gambar 3.2.



Gambar 3.2 Ilustrasi Metode Penelitian Pengembangan Aplikasi Steganografi pada Citra Digital.



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.

BAB IV

ANALISIS DAN PERANCANGAN

Dalam menyusun tugas akhir ini penulis menggunakan metode pengembangan sistem *Rapid Application Development* (RAD). Terdapat tiga tahapan yaitu perencanaan syarat-syarat, desain *workshop* RAD dan implementasi.

4.1 Fase Perencanaan Syarat – Syarat

Perencanaan syarat-syarat terdiri atas analisis kebutuhan, tujuan dan syarat-syarat. Proses ini dilakukan untuk mengetahui apa saja syarat-syarat dan kebutuhan yang dibutuhkan dalam menganalisis untuk tujuan dari perancangan aplikasi ini.

4.1.1 Analisis Kebutuhan

Pada tahap ini dilakukan suatu perbandingan dari beberapa aplikasi steganografi yang sudah ada. Perbandingan dilakukan untuk mengidentifikasi tujuan-tujuan aplikasi serta untuk mengetahui syarat-syarat informasi yang ditimbulkan dari tujuan-tujuan tersebut. Perbandingan dilakukan dengan mempertimbangkan kemudahan penggunaan dan fitur masing-masing aplikasi steganografi yang pernah ada. Orientasi dalam tahap ini adalah bagaimana cara menyelesaikan masalah yang akan terjadi dalam mencapai tujuan perancangan aplikasi steganografi yang akan dilakukan.

4.1.2 Menentukan Tujuan

Tujuan dari perancangan ini adalah membuat suatu aplikasi steganografi menggunakan file dalam bentuk citra digital/ gambar berformat *.bmp, sebagai media penampung untuk menyimpan file serta untuk memberikan salah satu solusi dalam pengamanan file.

4.1.3 Menentukan syarat-syarat

Syarat-syarat untuk mencapai tujuan dalam pengembangan aplikasi steganografi pada citra digital terdiri dari perangkat lunak dan perangkat yang spesifikasinya adalah sebagai berikut:

Perangkat Lunak :

1. *Windows Vista Black Edition 2009*
2. *Delphi 2007 win32*
3. *Inno Setup 5*
4. *Matlab R2008b*

Perangkat Keras :

1. *Processor intel Core 2 duo*
2. *Harddisk 250GB*
3. *RAM 2GB*
4. *LCD widescreen display dengan resolusi 1280x800 pixel*
5. *DVD RW*

Media :

1. *File Image *.bmp*
2. *File Pesan *.txt, *.doc, *.xsl, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav.*

4.2 Fase Desain RAD

Desain RAD terdiri atas perancangan proses, dan perancangan antar muka. Perancangan proses dilakukan untuk merancang alur proses di dalam program sedangkan perancangan antar muka dilakukan untuk mempermudah pengguna menggunakan hasil dari aplikasi ini.

4.2.1 Perancangan Proses

Tahap perancangan proses dilakukan untuk perancangan, evaluasi dan memperbaiki sistem sesuai dengan kebutuhan, agar sistem yang sedang di buat dapat dimanfaatkan secara optimal. Aplikasi ini menggunakan metode LSB (*Least Significant Bit*), untuk teknik steganografi dalam aplikasi ini karena ukuran pesan yang dapat disisipkan ke media penampungnya relatif besar dengan mengganti setiap LSB dari media penampungnya. Selain itu metode LSB adalah metode yang paling mudah diaplikasikan dibandingkan metode steganografi yang lainnya.

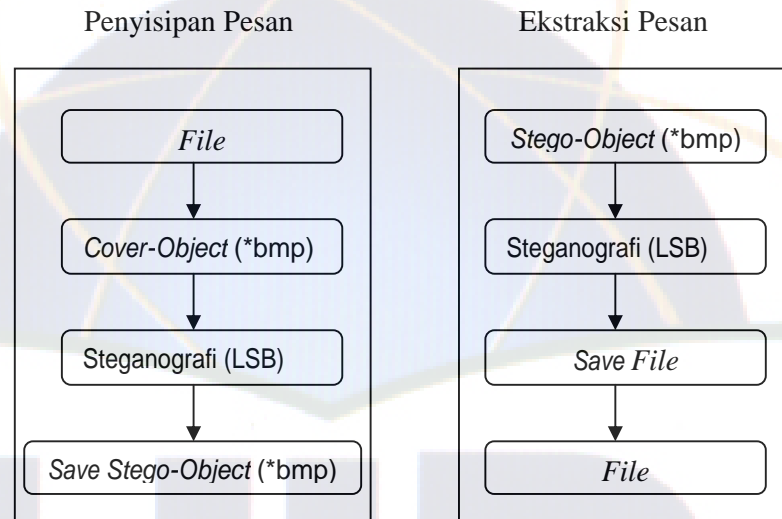
Pada penyembunyian pesan, pesan dapat di ambil/ *copy* yang berupa *file* seperti *.txt, *.doc, *.xsl, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav.

kemudian disisipkan menggunakan algoritma LSB ke media penampungnya.

Sehingga menghasilkan *stego-object* berupa *file *.bmp* yang telah disisipi pesan.

Tahapan selanjutnya adalah proses ekstraksi. Pada tahap ini atau *file*

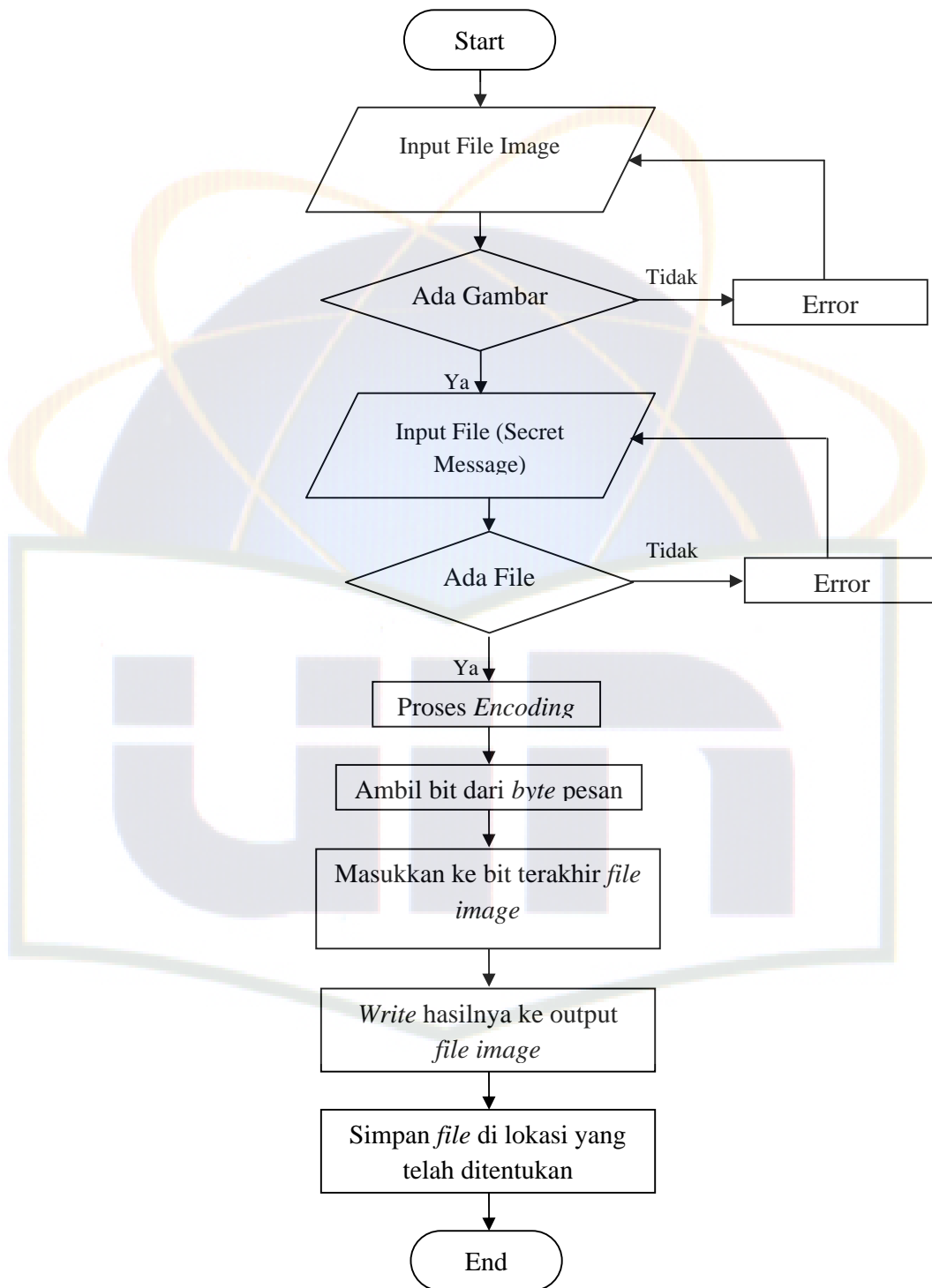
disembunyikan dipisahkan dari media penampungnya menggunakan teknik steganografi dengan metode LSB, sehingga menjadi *file* pesan yang dapat dibaca. Untuk memperjelas gambaran proses penyembunyian dan ekstraksi *file* pesan dapat dilihat pada Gambar 4.1.



Gambar 4.1 Proses Penyisipan dan Ekstraksi *File* Pesan

4.2.2 Flowchart Aplikasi Steganografi

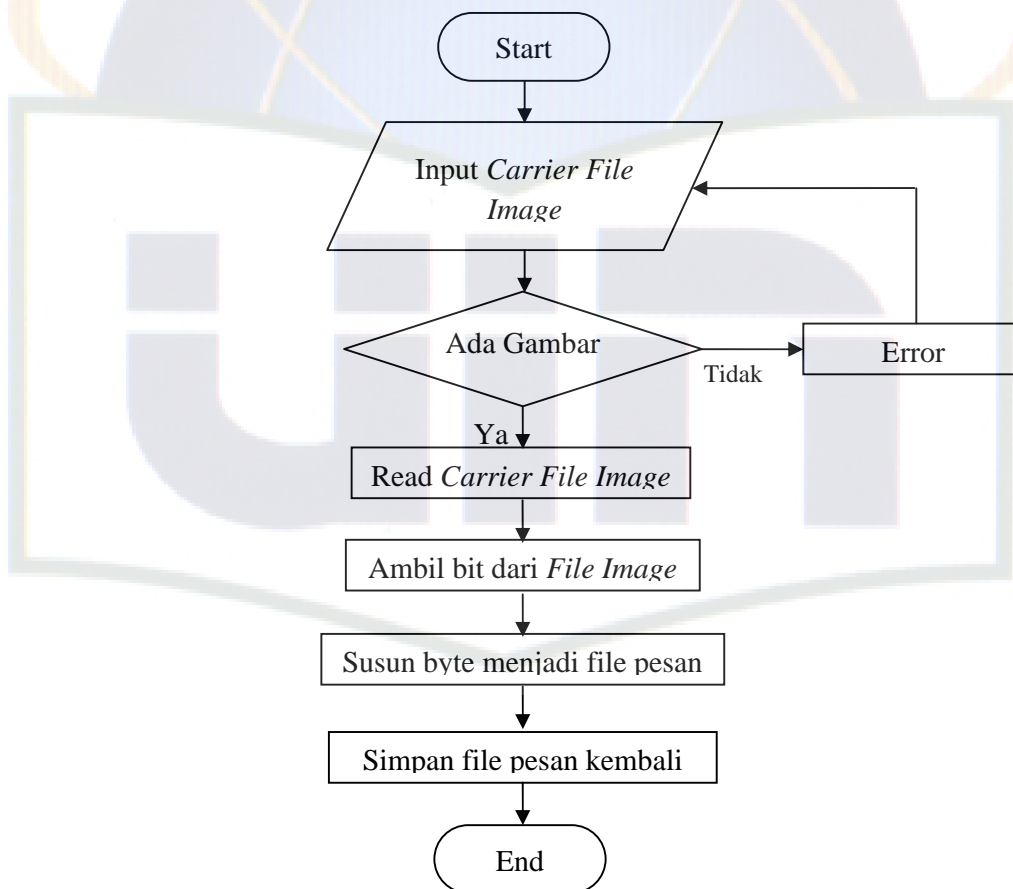
Pada tahap ini akan digambarkan alur proses penyisipan dan ekstraksi pesan rahasia dengan menggunakan *flowchart*.



Gambar 4.2 Flowchart Proses Penyisipan Pesan Rahasia

Gambar 4.2 menjelaskan informasi apa saja yang harus diinput untuk memulai proses penyisipan pesan rahasia. Informasi yang diinput haruslah lengkap, apabila tidak lengkap maka akan muncul error untuk mengingatkannya.

Setelah semua informasi terisi lengkap, program akan memanggil steganografi dengan menggunakan algoritma LSB untuk menyisipkan pesan di tiap LSB yang berada pada *File image*. Setelah semua proses selesai maka terbentuklah *file *.bmp* yang telah disisipkan pesan rahasia.



Gambar 4.3 Flowchart Proses Ekstraksi Pesan Rahasia

Gambar 4.3 menjelaskan informasi apa saja yang harus diinput untuk memulai proses ekstraksi pesan rahasia. Informasi haruslah lengkap, apabila tidak lengkap akan muncul *error* untuk mengingatkannya.







Setelah semua informasi terisi lengkap program steganografi akan menggunakan algoritma LSB untuk mengambil kembali *byte* pesan dari LSB *file image*, sehingga pesan rahasia dapat dibaca kembali. Setelah pesan rahasia dapat dibaca, maka proses ekstraksi pesan telah selesai.




4.2.3 Perancangan Antar Muka

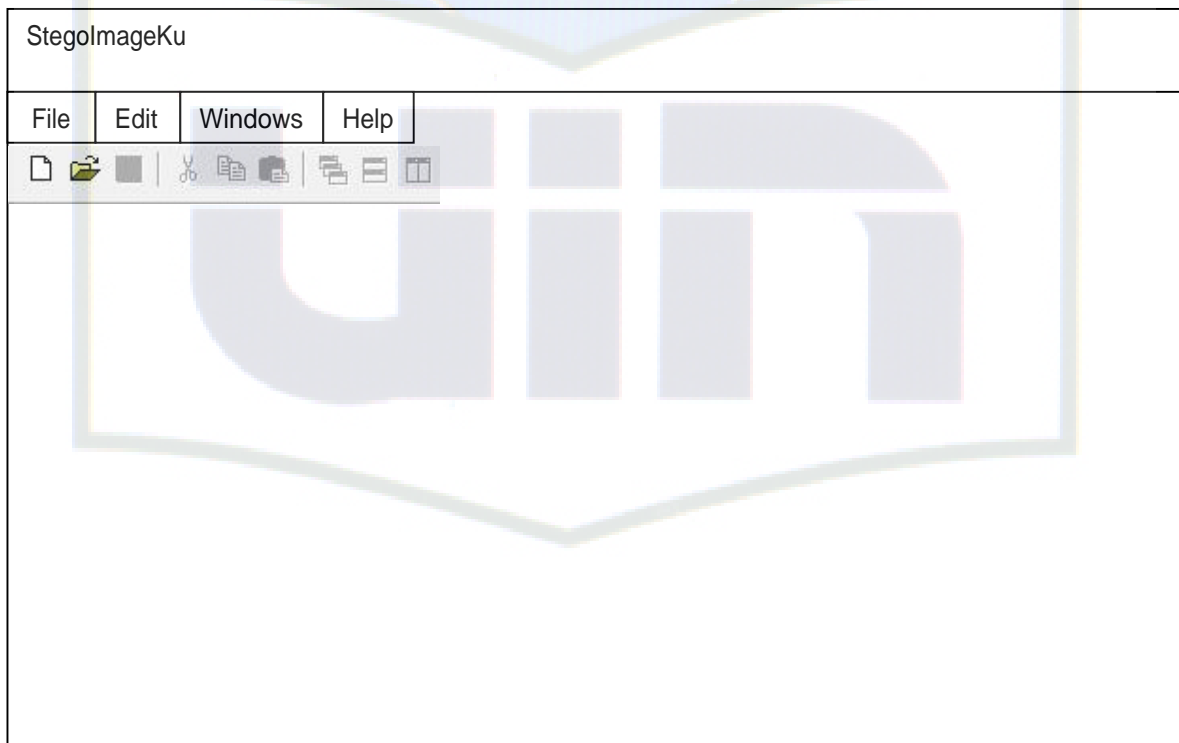
Dalam perancangan antar muka aplikasi steganografi pada citra digital ini, dibuat *form* tampilan yang akan ditampilkan dan dijelaskan fungsi *form* beserta fitur-fiturnya.

a. Perancangan *Form* induk

Form induk merupakan tampilan utama dalam program StegoImageKu yang didalamnya terdapat fitur sebagai berikut:

1.  Tombol *New*: Berfungsi untuk membuka *form Encoding & Decoding* yang baru.
2.  Tombol *Open*: Berfungsi untuk membuka *file*
3.  Tombol *Save*: Berfungsi untuk menyimpan *file*
4.  Tombol *Cut*: Berfungsi untuk memotong *file*
5.  Tombol *Copy*: Berfungsi untuk menggandakan *file*
6.  Tombol *Paste*: Berfungsi untuk memindahkan *file* setelah dicopy atau dicut

7.  Tombol *Cascade* : Berfungsi untuk menampilkan *form Encoding & Decoding* secara berurutan jika lebih dari 1 *form*.
8.  Tombol *Tile Horizontally* : Berfungsi untuk menampilkan *form Encoding & Decoding* secara horizontal jika lebih dari 1 *form*.
9.  Tombol *Tile Vertically* : Berfungsi untuk menampilkan *form Encoding & Decoding* secara vertical jika lebih dari 1 *form*.



Gambar 4.4 *Layout Form Induk*

b. Perancangan *Form* Utama Tab *Encoding*

Form utama Tab *Encoding* merupakan tampilan kedua dalam program StegoImageKu yang didalamnya terdapat fitur sebagai berikut:

1. Tombol *Browse*: Berfungsi untuk mencari letak lokasi direktori *File* yang akan disteganografi.
2. Tombol *Open*: Berfungsi untuk membuka *file image* yang akan dijadikan media penampung *file*.
3. Tombol *Encrypt*: Berfungsi untuk melakukan proses steganografi dengan metode LSB (*Least Significant Bit*).
4. Tombol *Clear*: Berfungsi untuk membatalkan membersihkan form.
5. *ScrollBar* + *Imagefile*: Berfungsi untuk letak *file image* yang Ingin dibuka dan disisipi file.
6. *EditBrowse* : Berfungsi untuk menampilkan letak direktori *file* Yang akan disisipkan.
7. *EditBit* : Berfungsi untuk menampilkan berapa bit yang Digunakan untuk proses penyisipan *file* kedalam *file image* yang menjadi media penampung.
8. *UpDown* : Berfungsi untuk menaikkan/ menurunkan penggunaan bit pada saat proses penyisipan *file* kedalam *file image*.

The screenshot shows the 'StegoImageKu' application window. At the top, there are two tabs: 'Encoding' (which is active) and 'Decoding'. Below the tabs, there is a text input field for a file path, followed by a 'Browse' button. Underneath, there is a label 'Bits per channel' followed by a dropdown menu showing the value '1' and a note '(Only for encryption)'. Below this, there are three buttons: 'Encrypt', 'Open', and 'Clear'. On the right side of the window, there is a large dashed rectangular box labeled 'File Image'.

Gambar 4.5 Rancangan *Form Utama Tab Encoding*

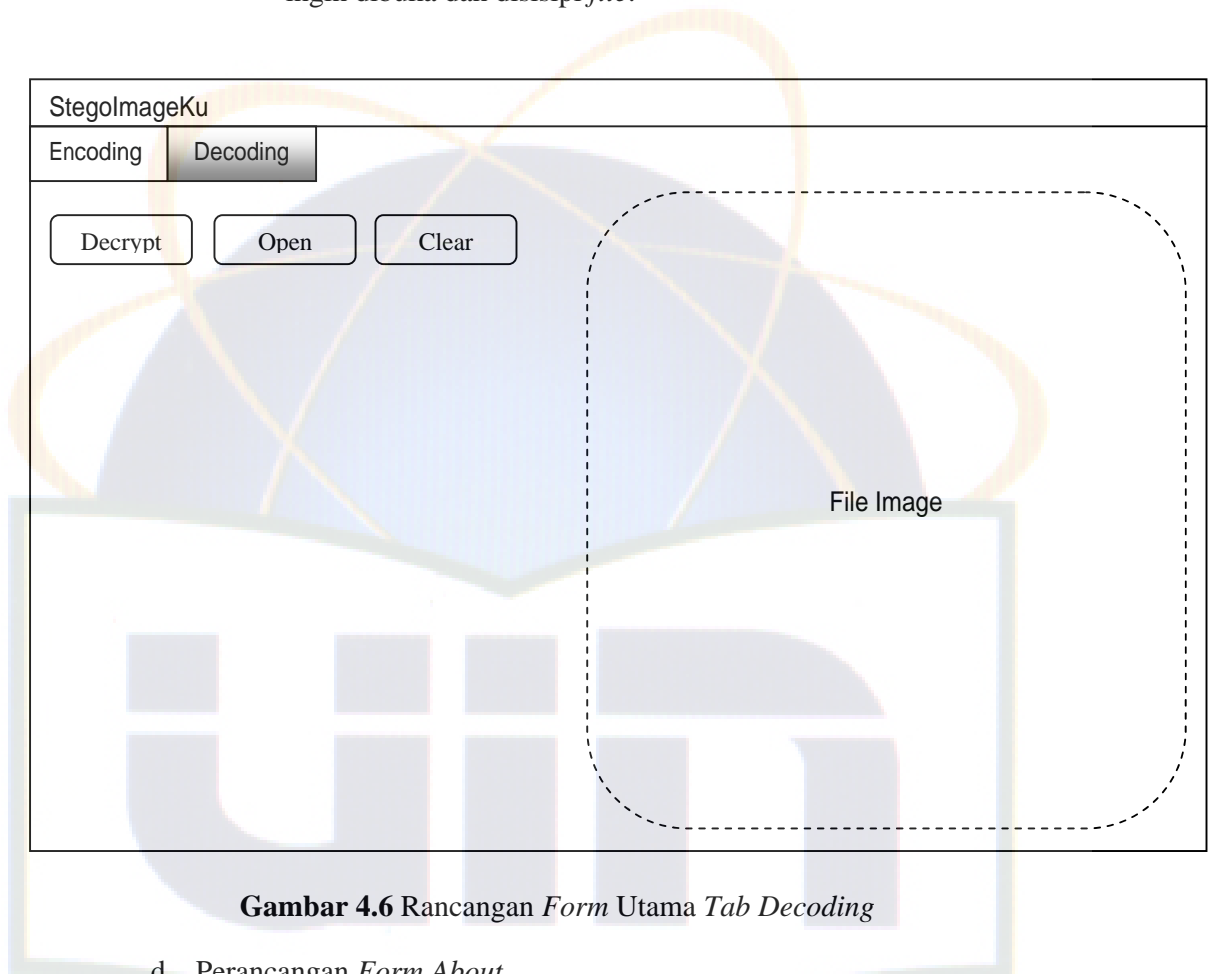
c. Perancangan *Form Utama Tab Decoding*

Form utama Tab Decoding merupakan tampilan kedua dalam program

StegoImageKu yang didalamnya terdapat fitur sebagai berikut:

1. Tombol *Open*: Berfungsi untuk membuka *file image* yang akan dijadikan media penampung *file*.
2. Tombol *Decrypt*: Berfungsi untuk melakukan proses ekstraksi *file* dari file image sebagai penampung.
3. Tombol *Clear*: Berfungsi untuk membatalkan membersihkan *form*.
4. Tombol *Exit*: Berfungsi untuk keluar dari aplikasi.

5. *ScrollBar + Imagefile*: Berfungsi untuk letak *file image* yang ingin dibuka dan disisipi *file*.

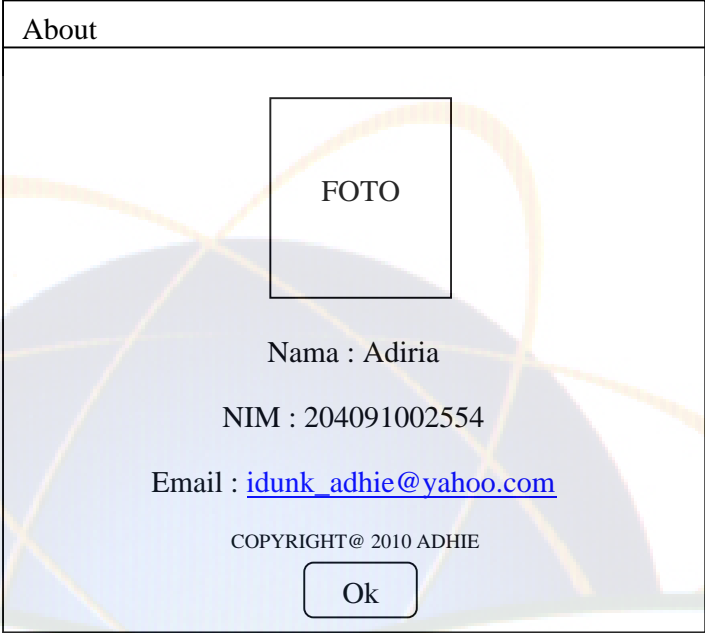


The screenshot shows the 'StegoImageKu' application window. At the top, there are two tabs: 'Encoding' and 'Decoding', with 'Decoding' being the active tab. Below the tabs, there are three buttons: 'Decrypt', 'Open', and 'Clear'. To the right of these buttons is a large, empty rectangular area with a dashed border, labeled 'File Image' at its bottom right corner. The background of the application window features a faint, stylized logo of a building or structure.

Gambar 4.6 Rancangan *Form Utama Tab Decoding*

d. Perancangan *Form About*

Form about ini berfungsi sebagai *form* untuk menampilkan biodata diri.



The image shows a screenshot of a software application window titled "About". Inside the window, there is a light blue background with a faint watermark of a globe and a book. The form contains the following elements:

- A rectangular box labeled "FOTO" for a profile picture.
- Text: "Nama : Adiria"
- Text: "NIM : 204091002554"
- Text: "Email : idunk_adhie@yahoo.com"
- Text: "COPYRIGHT@ 2010 ADHIE"
- An "Ok" button at the bottom.

Gambar 4.7 Perancangan *Form About*

4.2.4 STD (*State Transition Diagram*)

Gambar 4.8 adalah gambaran *State Transition Diagram* dari aplikasi steganografi yang ingin dirancang pembuatannya.

proses encoding dan decoding file yang akan dienkrpsi dan dekripsi dengan meng-*input* file *image* sebagai media penampungnya dan file apa saja untuk disisipkan ke dalam *file image* tersebut. “Uprocess.pas” sebagai fungsi untuk melakukan proses penyisipan *file* ke dalam *file image*, “Ubit.pas” sebagai fungsi untuk membaca bit-bit yang berada di dalam *file image*, ”about.pas” sebagai *form about* yang berfungsi untuk menampilkan biodata penulis dan “Unit4.pas” sebagai *splash screen*.

4.3 Fase Implementasi

Dalam tahapan ini akan dilakukan pengujian dan analisis pengujian terhadap aplikasi StegoImage ini yang bertujuan untuk mengetahui tingkat keberhasilan dari aplikasi dalam mencapai hasil dan tujuan yang diinginkan.

4.3.1 Instalasi Aplikasi

Program instalasi aplikasi StegoImage ini menggunakan *software* Inno Setup 5. Dalam penggunaannya, *software* ini sangat mudah digunakan. Apabila pengguna tidak bisa membuat *script/ coding* secara langsung, program ini menyediakan menu “create a new script file using the script wizard”, jadi pengguna hanya mengikuti alur dari aplikasi dan selanjutnya aplikasi yang membuatkan script untuk proses instalasinya.

Di bawah ini adalah langkah-langkah instalasi aplikasi StegoImage ke dalam komputer pengguna:

1. Pilih program StegoImage Setup.exe, kemudian akan ada tampilan pembuka dari proses instalasi ini menggunakan bahasa inggris. Pilih

tombol *next* untuk melanjutkan instalasi, atau *cancel* untuk membatalkan instalasi.

2. Jika user menekan tombol *next* maka akan muncul halaman *license agreement*, yaitu *form* persetujuan yang menanyakan kesetujuan *user* atas semua konsekuensi jika menggunakan program ini. Apabila user memilih “I accept the agreement” yang berarti setuju maka proses instalasi akan berlanjut ke tahap selanjutnya. Apabila pengguna memilih “I don’t accept the agreement” yang berarti tidak setuju maka proses instalasi berhenti.
3. Tampilan untuk menentukan lokasi aplikasi StegoImage disimpan, akan langsung mengarah ke C:\Program Files\StegoImageKu, tetapi *user* dapat mengubah lokasi dengan cara menekan tombol *browse*. Di halaman ini juga terdapat informasi ukuran program yang akan diinstal.
4. Tampilan untuk menyimpan *shortcut folder* StegoImageKu di *start menu*.
5. Tampilan untuk menampilkan *shortcut* aplikasi StegoImage di *desktop*.
6. Tampilan proses instalasi, apabila informasi yang dibutuhkan sudah lengkap maka program siap untuk diinstal.
7. Tampilan informasi bahwa program telah selesai diinstal, dan aplikasi StegoImage akan muncul sebagai tanda bahwa proses instalasi telah berhasil.

4.3.2 Analisis Penyembunyian File

Analisis ini dilakukan bertujuan untuk mengetahui apakah *file image* dapat menampung *file* pesan tanpa adanya perubahan ukuran pada *file image*. Kemudian apakah *file* pesan tersebut dapat diambil kembali seperti semula tanpa adanya perubahan.

Tabel 4.1 Tabel Uji Penyembunyian File

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Monalisa.bmp	419454	Serbu.txt	265	MonalisaStego1.bmp	419454	Serbu.txt	265
Monalisa.bmp	419454	Gadis.txt	10977	MonalisaStego2.bmp	419454	Gadis.txt	10977
Monalisa.bmp	419454	Daftar Isi.doc	52224 52431,75	MonalisaStego3.bmp	419454	Daftar Isi.doc	52224

Dari hasil Table 4.1 terlihat bahwa *file image* dapat menampung *file* pesan tanpa adanya perubahan ukuran pada *file image*. Dan *file* pesan yang telah disisipi ke dalam *file image* dapat diambil kembali tanpa adanya perubahan pada *file* pesan. Ini dikarenakan *file* pesan yang disisipkan mengalami proses steganografi yang menggunakan metode LSB (Least Significant Bit), dimana file pesan disisipkan pada bit terakhir pada *file image*.

4.3.3 Analisis Format File Apa Saja yang Dapat Disisipkan ke File Image

Analisis ini dilakukan bertujuan untuk mengetahui jenis format apa saja yang dapat ditampung atau disisipi ke dalam *file image* tanpa adanya perubahan pada *file image* dan *file* pesan sebelum maupun setelah proses steganografi.

Tabel 4.2 Tabel Uji Format *File* Apa Saja yang Dapat Disisipkan ke *File Image*

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Pesawat.bmp	1440054	Index.ppt	163840	Pesawat1.bmp	1440054	Index1.ppt	163840
Pesawat.bmp	1440054	19SC096.JPG	33848	Pesawat2.bmp	1440054	19SC0961.JPG	33848
Pesawat.bmp	1440054	AbstrakHB.pdf	65536	Pesawat3.bmp	1440054	AbstrakHB.pdf	65536
Pesawat.bmp	1440054	Acctspay.mdb	102400	Pesawat4.bmp	1440054	Acctspay.mdb	102400
Pesawat.bmp	1440054	Biodata.doc	31744	Pesawat5.bmp	1440054	Biodata.doc	31744
Pesawat.bmp	1440054	browsesub.php	7290	Pesawat6.bmp	1440054	browsesub.php	7290
Pesawat.bmp	1440054	CA_CHING.WAV	18502	Pesawat7.bmp	1440054	CA_CHING.WAV	18502
Pesawat.bmp	1440054	coding.txt	21566	Pesawat8.bmp	1440054	coding.txt	21566
Pesawat.bmp	1440054	salary.xls	58880	Pesawat9.bmp	1440054	salary.xls	58880
Pesawat.bmp	1440054	web.htm	31051	Pesawat10.bmp	1440054	web.htm	31051

Dari hasil Table 4.2 terlihat bahwa *file image* dapat menampung *file* pesan berupa format apa saja tanpa adanya perubahan pada *file image*. Dan *file* pesan yang telah disisipi kedalam *file image* dapat diambil kembali tanpa adanya perubahan pada *file* pesan dengan syarat format *file* pesan sebelum maupun sesudah proses steganografi harus sama. Ini dikarenakan *file* pesan yang disisipkan mengalami proses steganografi yang menggunakan metode LSB (*Least Significant Bit*), dimana *byte-byte file* pesan disisipkan pada bit terakhir pada *file image*.

4.3.4 Analisis Penyisipan dan Ekstraksi *File* Pesan terhadap Tiga *File Image* yang Berbeda

Analisis ini dilakukan untuk membuktikan *file* pesan dapat disisipkan ke dalam *file* image yang berbeda, dengan syarat berformat *.bmp 24 bit (*true color*). Analisis dapat dilihat pada Tabel 4.3, 4.4 dan 4.5.





4.3.5 Analisis Penyisipan dan Ekstraksi *File image* dan *File Pesan*

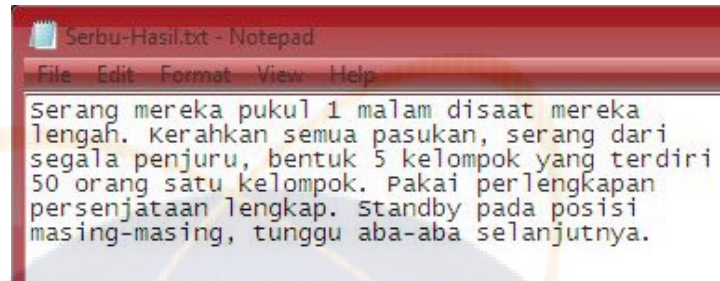
Setelah melakukan proses penyisipan dan ekstraksi *file image*, maka untuk melihat apakah hasil dari penyisipan dan ekstraksi *file image* telah berhasil seperti yang akan dilakukan langkah-langkah berikut:

1. Buka lokasi tempat *file image* yang telah disisipkan pesan rahasia.
Perhatikan *file image* yang asli dan *file image* yang telah disisipkan *file* pesan tidak ada bedanya. Jadi *user* harus mengingat letak lokasi dan nama *file image*-nya dengan pasti.
2. *File Image* yang telah disisipkan pesan rahasia masih dapat dilihat dengan baik selayaknya *file image* asli oleh user, tanpa disadari bahwa adanya *file* pesan yang telah disisipkan ke dalamnya.

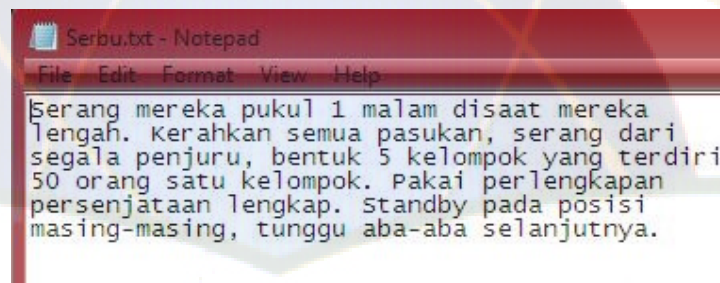


Gambar 4.9 *File Image* Sebelum dan Sesudah Disisipkan *File Pesan*

1. Untuk membuka *file* pesan yang telah diambil dari *file image*, *user* dapat langsung membacanya dengan membuka *file* pesan tersebut yang sebelumnya di-*save* dari aplikasi StegoImageKu.



Gambar 4.10 File Pesan Rahasia Sebelum Disisipkan ke File Image



Gambar 4.11 File Pesan yang Diambil Kembali dari File Image

Dari hasil pengamatan di atas tampak tidak adanya perubahan *file* pesan sebelum dan sesudah proses penyisipan pesan ke dalam *file* image baik bentuk maupun ukurannya.

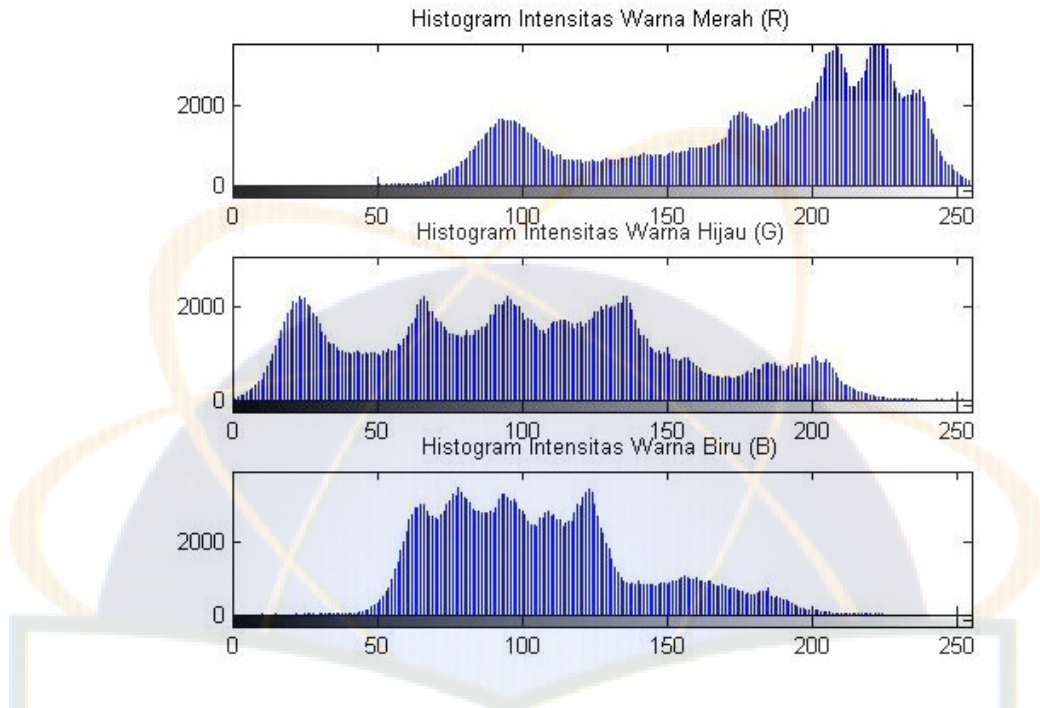
4.3.6 Analisis File Pesan Dan File Image Jika Digunakan 2 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat penurunan intensitas serta kualitas *file* image terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 2 bit terakhir pada *file* image tersebut.

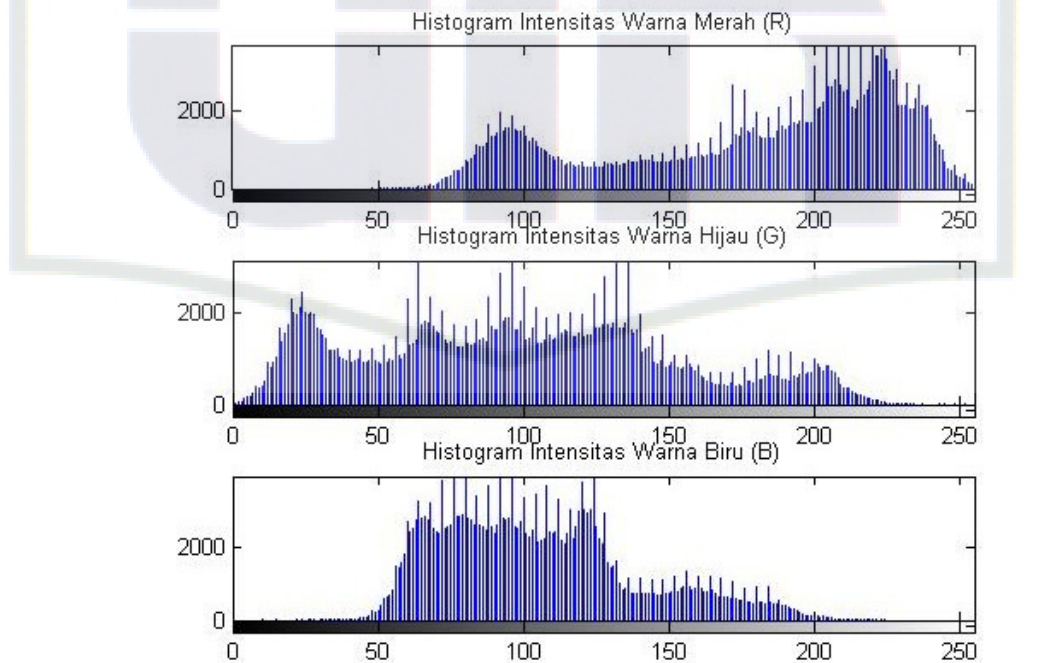


Gambar 4.12 Perbedaan Kualitas *File Image* Jika Digunakan 2 Bit Terakhir

Dari Gambar 4.12 terlihat bahwa *file image* tidak mengalami perubahan baik gambar maupun ukuran, jika menggunakan 2 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 2 bit terakhir dinyatakan berhasil. Proses steganografi menggunakan 2 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.13 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 2 Bit Terakhir.



Gambar 4.14 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 2 Bit Terakhir.

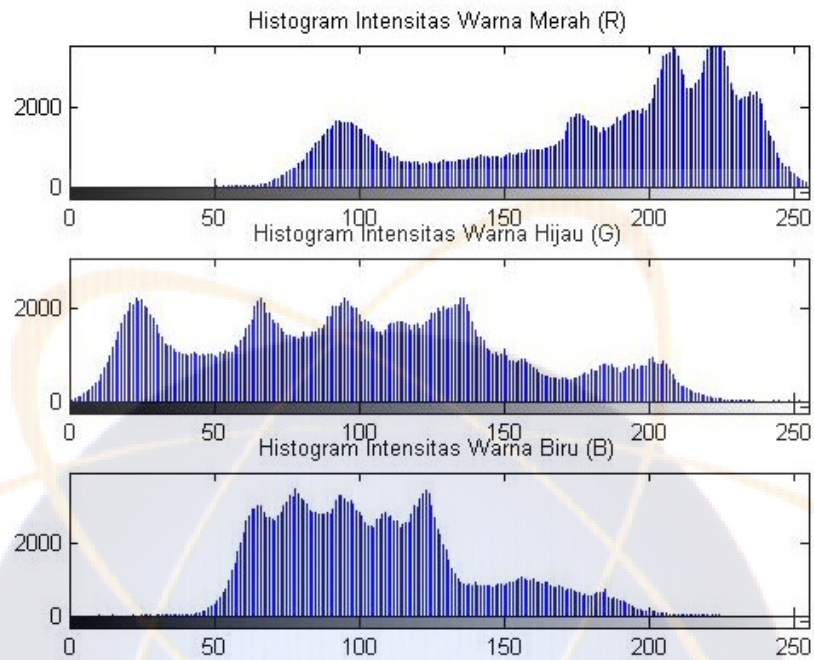
4.3.7 Analisis File Pesan Dan File Image Jika Digunakan 3 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 3 Bit terakhir pada *file image* tersebut.

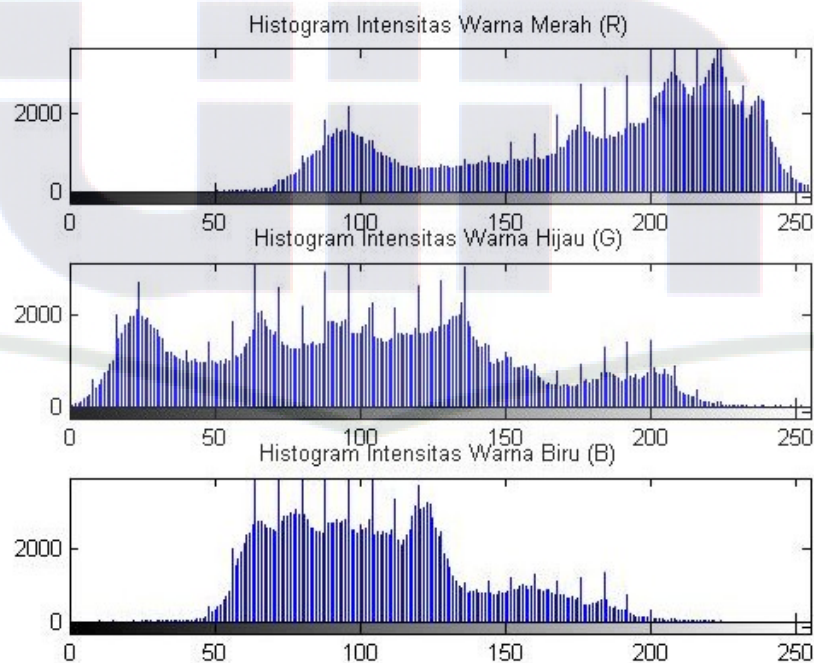


Gambar 4.15 Perbedaan Kualitas *File Image* Jika Digunakan 3 Bit Terakhir

Dari Gambar 4.15 terlihat bahwa *file image* tidak mengalami perubahan baik gambar maupun ukuran, jika menggunakan 3 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 3 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit dan 2 bit terakhir. Proses steganografi menggunakan 3 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit dan 2 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.16 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 3 Bit Terakhir



Gambar 4.17 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 3 Bit Terakhir

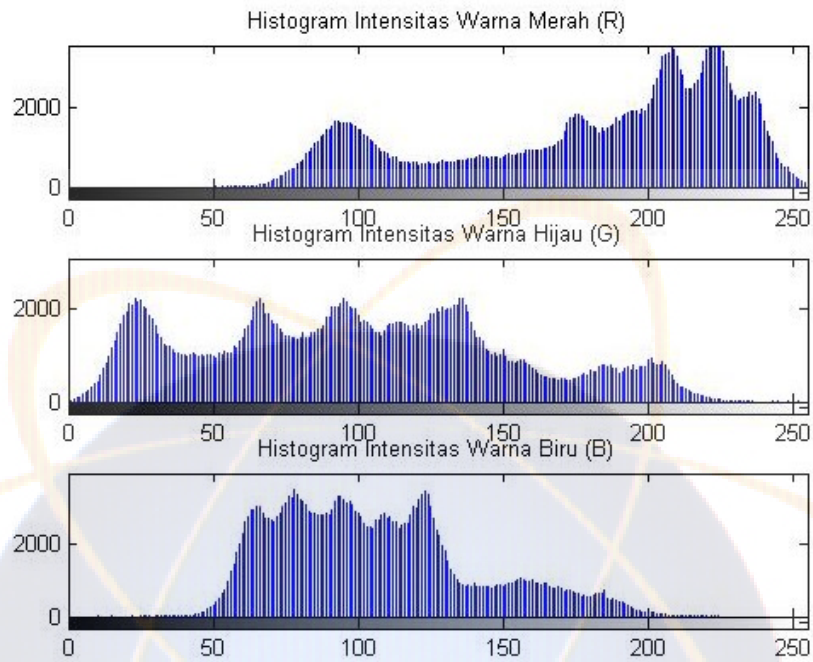
4.3.8 Analisis File Pesan Dan File Image Jika Digunakan 4 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 4 Bit terakhir pada *file image* tersebut.

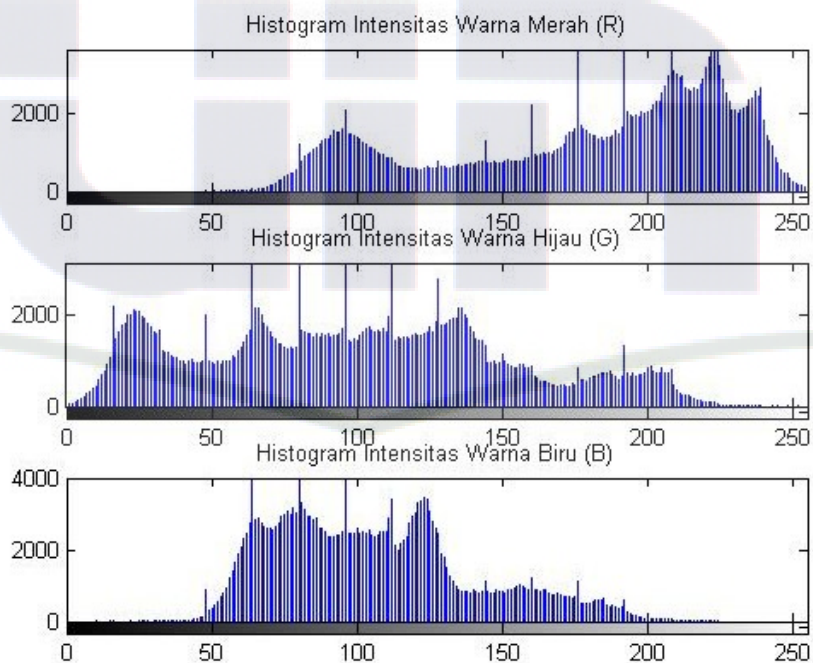


Gambar 4.18 Perbedaan Kualitas *File Image* Jika Digunakan 4 Bit Terakhir

Dari Gambar 4.18 terlihat bahwa *file image* tidak mengalami perubahan baik gambar maupun ukuran, jika menggunakan 4 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 4 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit dan 3 bit terakhir. Proses steganografi menggunakan 4 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit dan 3bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.19 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 4 Bit Terakhir.



Gambar 4.20 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 4 Bit Terakhir.

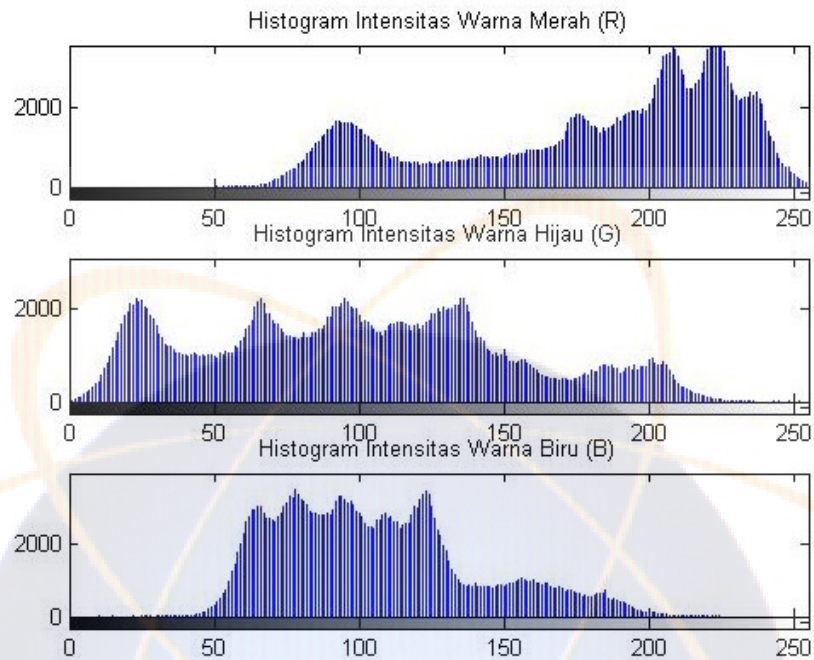
4.3.9 Analisis File Pesan Dan File Image Jika Digunakan 5 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 5 Bit terakhir pada *file image* tersebut.

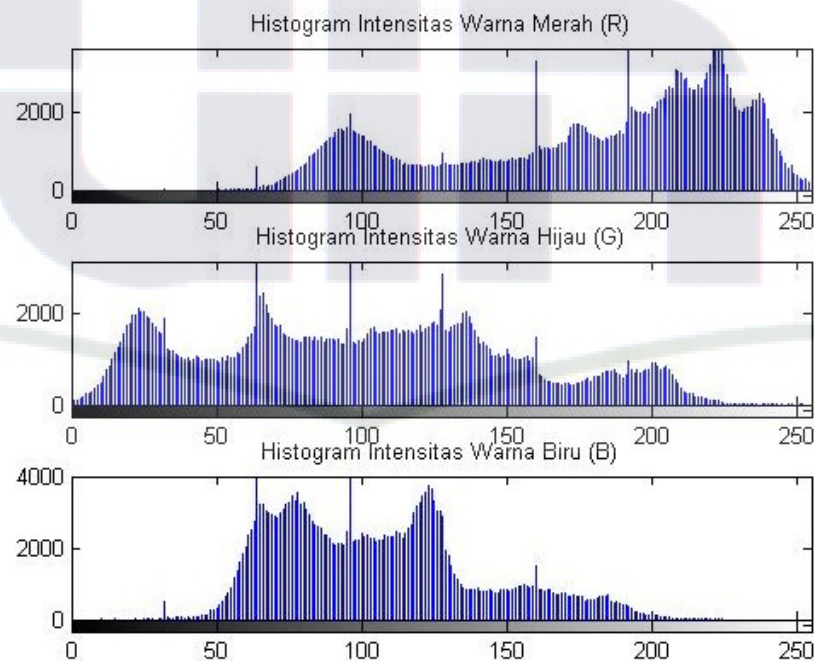


Gambar 4.21 Perbedaan Kualitas *File Image* Jika Digunakan 5 Bit Terakhir

Dari Gambar 4. terlihat bahwa *file image* tidak mengalami perubahan ukuran, jika menggunakan 5 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 5 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit, 3 bit dan 4 bit terakhir. Pada penggunaan 5 bit terakhir mulai terjadi sedikit perubahan gambar, karena terjadi penurunan intensitas warna pada 5 bit terakhir di setiap *pixel*. Proses steganografi menggunakan 5 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit, 3 bit dan 4 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.22 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 5 Bit Terakhir.



Gambar 4.23 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 5 Bit Terakhir.

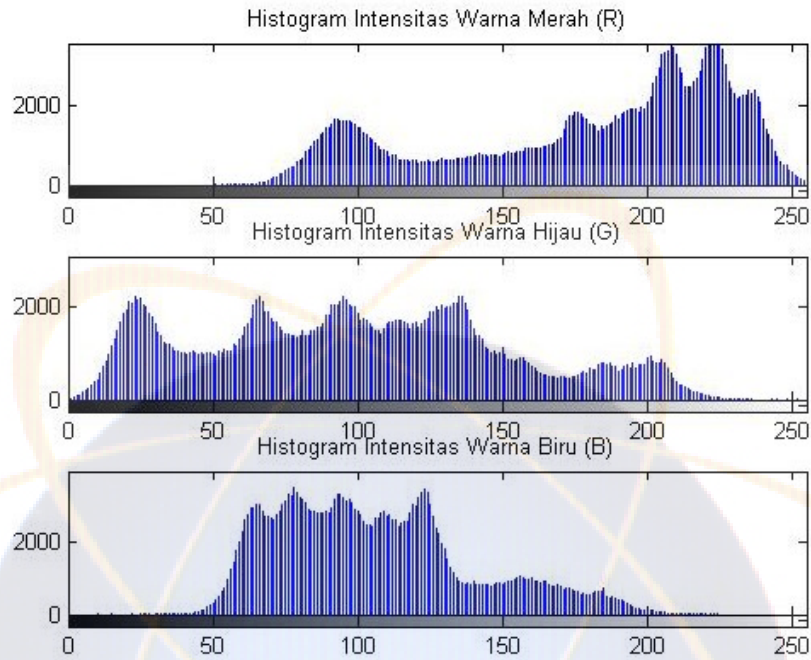
4.3.10 Analisis *File Pesan* Dan *File Image* Jika Digunakan 6 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 6 Bit terakhir pada *file image* tersebut.

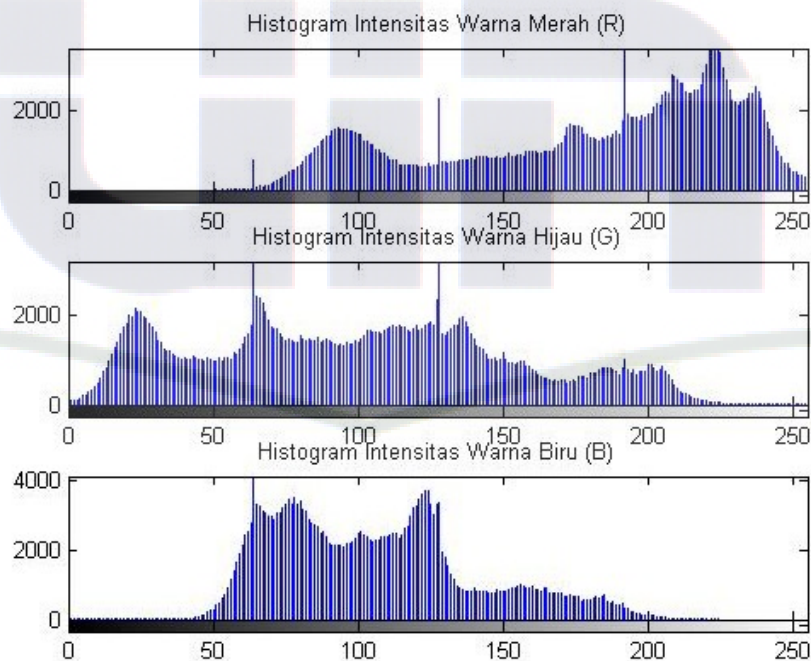


Gambar 4.24 Perbedaan Kualitas *File Image* Jika Digunakan 6 Bit Terakhir

Dari Gambar 4. terlihat bahwa *file image* tidak mengalami perubahan ukuran, jika menggunakan 6 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 6 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit, 3 bit, 4 bit dan 5 bit terakhir. Pada penggunaan 6 bit terakhir mulai terjadi perubahan gambar lebih besar dibandingkan dengan 5 bit, karena terjadi penurunan intensitas warna lebih besar dari pada 5 bit terakhir di setiap *pixel*-nya. Proses steganografi menggunakan 5 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit, 3 bit, 4 bit dan 5 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.25 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 6 Bit Terakhir.



Gambar 4.26 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 6 Bit Terakhir.

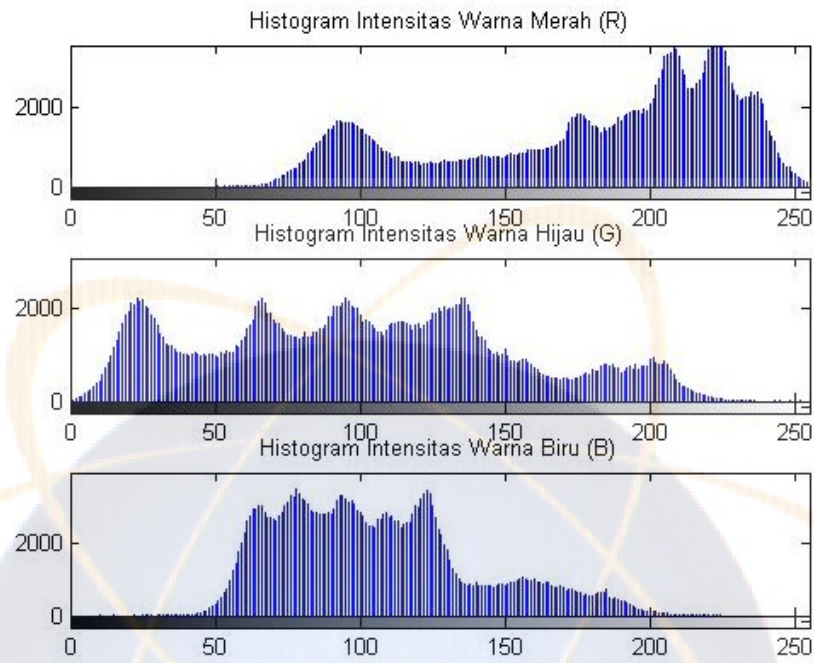
4.3.11 Analisis *File Pesan* Dan *File Image* Jika Digunakan 7 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 7 Bit terakhir pada *file image* tersebut.

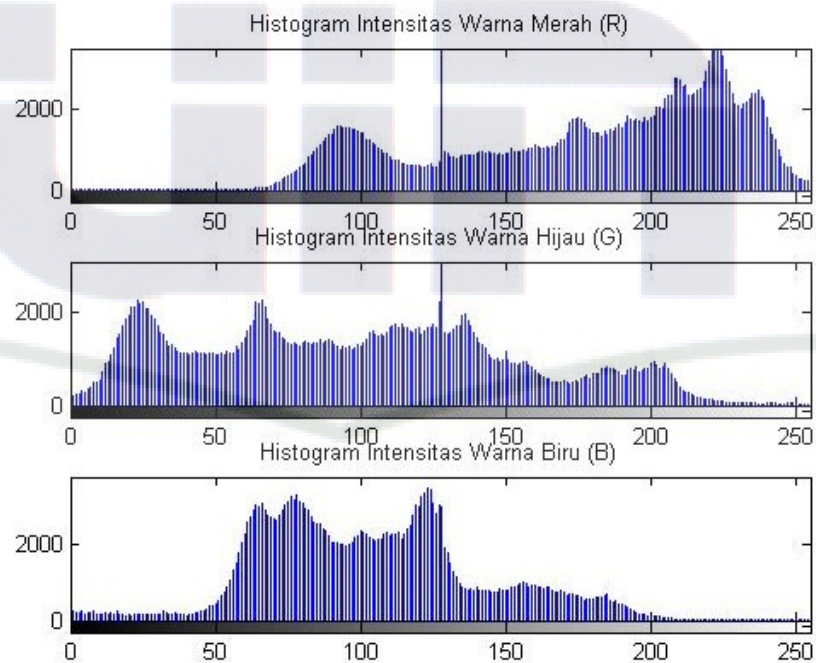


Gambar 4.27 Perbedaan Kualitas *File Image* Jika Digunakan 7 Bit Terakhir

Dari Gambar 4.27 terlihat bahwa *file image* tidak mengalami perubahan ukuran, jika menggunakan 7 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 7 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit, 3 bit, 4 bit, 5 bit dan 6 bit terakhir. Pada penggunaan 7 bit terakhir terjadi perubahan gambar lebih besar dibandingkan dengan 5 bit dan 6 bit, karena terjadi penurunan intensitas warna lebih besar dari pada 5 bit dan 6 bit terakhir di setiap *pixel*-nya. Proses steganografi menggunakan 7 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit, 3 bit, 4 bit, 5 bit dan 6 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.28 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 7 Bit Terakhir.



Gambar 4.29 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 7 Bit Terakhir.

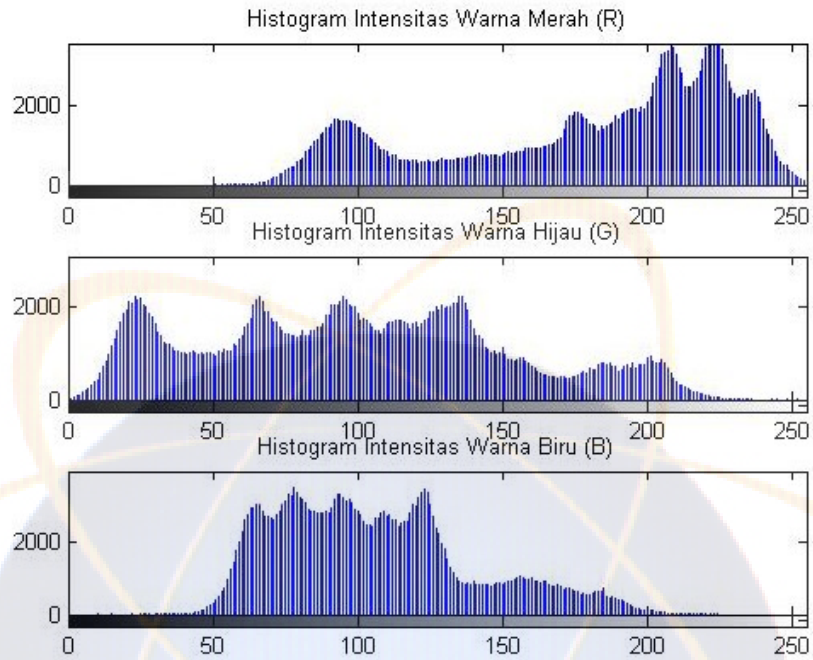
4.3.12 Analisis *File Pesan* Dan *File Image* Jika Digunakan 8 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 8 Bit terakhir pada *file image* tersebut.

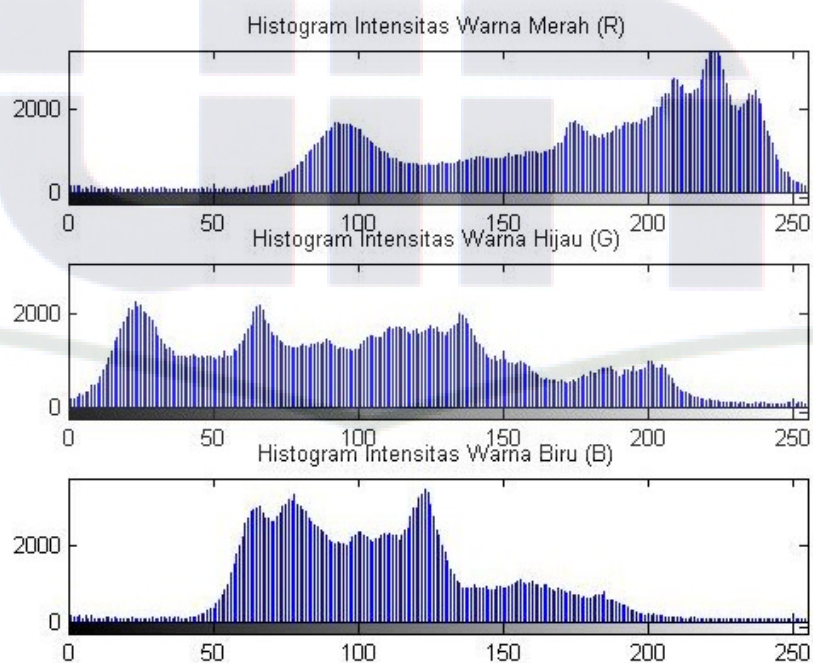


Gambar 4.30 Perbedaan Kualitas *File Image* Jika Digunakan 8 Bit Terakhir

Dari Gambar 4. terlihat bahwa *file image* tidak mengalami perubahan ukuran, jika menggunakan 8 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 8 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit, 3 bit, 4 bit, 5 bit, 6 bit dan 7 bit terakhir. Pada penggunaan 8 bit terakhir terjadi perubahan gambar lebih besar dibandingkan dengan 5 bit, 6 bit dan 7 bit karena terjadi penurunan intensitas warna lebih besar dari pada 5 bit, 6 bit, dan 7 bit terakhir di setiap *pixel*-nya. Proses steganografi menggunakan 8 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit, 3 bit, 4 bit, 5 bit, 6 bit dan 7 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.31 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 8 Bit Terakhir.



Gambar 4.32 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 8 Bit Terakhir.

4.3.13 Analisis Ukuran *File* Pesan Terhadap *File Image*

Analisis ini bertujuan untuk mengetahui batasan ukuran *file* pesan yang dapat disisipkan ke dalam *file image*.

Tabel 4.6 Tabel uji Ukuran *File* Pesan Rahasia Terhadap *File Image*

File Image	Size (byte)	Secret File	Size (byte)	Status Penyisipan
Lena.bmp	786486	Serbu.txt	265	Berhasil
Lena.bmp	786486	Tabel4.doc	44544	Berhasil
Lena.bmp	786486	Bab IV.doc	323584	Tidak

Dari Table 4.3 terlihat bahwa proses penyisipan hanya dapat dilakukan apabila byte yang tersedia di dalam *file image* lebih besar dari pada jumlah *byte file* pesan. Suatu citra 24-bit jika digunakan untuk menyimpan informasi rahasia mampu menampung informasi maksimum berukuran $\frac{1}{8}$ dari ukuran citra penampung tersebut. Jadi semakin besar *file image* maka jumlah *byte* yang disediakan untuk menampung pesan rahasia akan lebih banyak.

4.4 Perbandingan Steganografi Sejenis

Penulis membandingkan dengan steganografi yang sejenis yang telah ada sebelumnya dari berbagai skripsi dari beberapa perguruan tinggi. Ini bertujuan untuk dijadikan bahan perbandingan dengan sistem yang akan penulis buat, bukan bermaksud menentukan baik atau buruknya.

Tabel 4.7 Perbandingan Steganografi Sejenis 1

Judul	Penulis	Tahun	Universitas	Perbedaan
ANALISIS DAN IMPLEMENTASI STEGANOGRAFI DAN KRIPTOGRAFI PADA FILE AUDIO	YUNIAR NURSYAMSIAH SETIORINI 103091029623	2008	Universitas Islam Negeri Syarif Hidayatullah Jakarta	<ol style="list-style-type: none"> 1. Skripsi di atas menggunakan <i>file audio</i> sebagai media penampung <i>file</i>-nya, sedangkan penulis menggunakan Citra Digital untuk penampungan <i>file</i>-nya. 2. Skripsi di atas memakai metode pengembangan sistem SDLC (<i>System Development Life Cycle</i>) sedangkan penulis memakai metode pengembangan sistem RAD (<i>Rapid Application Development</i>). 3. Skripsi diatas hanya fokus terhadap 1 bit terakhir pada file audionya, sedangkan penulis mengembangkan hingga 8 bit dapat digunakan. 4. Skripsi diatas memakai C# dalam pembuatan aplikasinya, sedangkan penulis menggunakan <i>Delphi</i>.

				5. Skripsi diatas hanya dapat beberapa format <i>file</i> yang dapat disisipkan, sedangkan penulis hampir semua format <i>file</i> dapat disisipkan.
--	--	--	--	--

Tabel 4.8 Perbandingan Steganografi Sejenis 2

Judul	Penulis	Tahun	Universitas	Perbedaan
PEMBUATAN APLIKASI STEGANOGRAPHY PADA FILE AUDIO	Hardi Wijaya 26402169	2007	Universitas Kristen Petra Surabaya	<ol style="list-style-type: none"> 1. Skripsi diatas memakai metode pengembangan sistem SDLC (<i>System Development Life Cycle</i>) sedangkan penulis memakai metode pengembangan sistem RAD (<i>Rapid Application Development</i>). 2. Skripsi diatas memakai Visual Basic dalam pembuatan aplikasinya, sedangkan penulis menggunakan Delphi. 3. Skripsi diatas hanya dapat beberapa format <i>file</i> yang dapat disisipkan, sedangkan penulis hampir semua format <i>file</i> dapat disisipkan. 4. Skripsi diatas menggunakan <i>file audio</i> sebagai media penampung <i>file</i>-nya, sedangkan penulis menggunakan Citra Digital untuk penampungan filenya 5. Skripsi diatas hanya fokus terhadap 1 bit terakhir pada file

				audionya, sedangkan penulis mengembangkan hingga 8 bit dapat digunakan.
--	--	--	--	---





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.

BAB V

PENUTUP

5.1 Kesimpulan

Dari hasil pengujian sistem yang dilakukan pada bab sebelumnya, maka dapat disimpulkan ke dalam beberapa hal antara lain:

1. Aplikasi StegoImage berhasil mengimplementasikan teknik steganografi yang menggunakan algoritma LSB dalam mengamankan pesan. Hal ini dibuktikan melalui hasil uji analisa pada Tabel 4.1 bahwa *file* dapat disisipkan ke dalam *file image* dan diambil kembali dari *file image* tersebut.
2. Proses penyisipan *file* tidak terpaku pada satu format *file* tertentu saja, tapi dapat dilakukan pada *file* berformat *.txt, *.doc, *.xls, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav hal ini dibuktikan oleh uji analisi pada Tabel 4.2.
3. Berdasarkan hasil uji analisis, penyisipan *file* ke dalam *file image* mempengaruhi kualitas warna pada *file image* tersebut. Dengan adanya perubahan intensitas warna antara *file image* asli dan *file image* yang sudah disisipkan pesan, hal itu dapat dilihat pada gambar-gambar histogram pada bab IV.
4. Berdasarkan hasil uji analisis pada Tabel 4.1 dapat dinyatakan bahwa *file* pesan Masukan dan hasil keluaran memiliki jumlah *byte* yang sama persis, di mana artinya penyisipan *file* pesan tidak mempengaruhi besar ukuran *file* pesan awal maupun akhir.
5. Berdasarkan hasil uji analisis keseluruhan pada bab IV dapat disimpulkan bahwa algoritma LSB tidak hanya terpaku pada 1 bit terakhir saja sebagai tempat

penyembunyian data, tapi dapat dikembangkan hingga 8 bit.

5.2 Saran

1. Aplikasi StegoImageKu ini dapat diimplementasikan di instansi yang membutuhkan pengamanan *file* seperti bank, kantor pemerintahan, militer dan sebagainya.
2. Aplikasi StegoImageKu hanya *file image* sebagai media penampung, diharapkan dapat dikembangkan sehingga dapat menggunakan *file* teks, audio, video dan lain-lain sebagai media penampungnya.
3. Aplikasi StegoImage masih dikembangkan untuk perangkat *computer desktop*.

Akan lebih praktis apabila dapat dikembangkan lebih lanjut untuk dapat digunakan dalam lingkungan perangkat keras *mobile* seperti telepon genggam.



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.

DAFTAR PUSTAKA

Ahmad, Usman. (2005). *Pengolahan Citra Digital & Teknik Pemogramannya*. Yogyakarta : Graha Ilmu.

Ariyus, Doni. (2006). *Kriptografi Keamanan Data dan Komunikasi*. Yogyakarta : Graha Ilmu.

Andleigh, Prabhat K., Thakrar, Kiran. (1996). *Multimedia System Design*. New Jersey : Prentice Hall International Edition.

doank29.multiply.com/journal/item/3

haryanto.staff.gunadarma.ac.id/Downloads/files/7272/9.Steganografi.ppt

ilmucerdas.wordpress.com/2008/08/02/pengertian-multimedia/

ilmukomputer.org/2006/08/25/aplikasi-steganografi-dengan-borland-delphi/

Kenneth E. Kendall & Julie E Kendall. (2005). *System Analysis and Design*. Sixth Editon. New Jersey : Pearson Education.

Ladjamuddin, Al Bahra Bin. (2005). *Analisis dan Desain Sistem Informasi*. Jakarta : Graha Ilmu.

Malik, Jaja Jamaludin. (2006). *Kumpulan Latihan Pemograman Delphi*. Yogyakarta : Andi.

Marvin Ch, Wijaya, & Agus Priyono. (2007). *Pengolahan Citra Digital Menggunakan MatLAB Image Processing Toolbox*. Bandung: Informatika.

Munir, Rinaldi. (2006). *Kriptografi*. Bandung: Informatika.

Munir, Rinaldi. (2004). *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Bandung : Informatika.

NN. (2002). *11 Kamus Besar Bahasa Indonesia*. Jakarta : Balai Pustaka.

Prasetyo, Didik Dwi. (2004). *Aplikasi Database Client/Server Menggunakan Delphi dan MySQL*. Jakarta: Elex Media Komputindo.

Pressman, Roger S. (2002). *Rekayasa Perangkat Lunak*, Edisi 1. Yogyakarta: Andi.

webmail.informatika.org/~rinaldi/Kriptografi/2007-2008/Makalah1/MakalahIF5054-2007-A-077.pdf

Whitten, Jeffrey, dan Lonnie Bentley. (2007). *Sistem Analisis dan Metode Desain*. Edisi Kelima. Yogyakarta : Penerbit Andi.

www.ascii-code.com/

www.cert.or.id/~budi/courses/ec7010/dikmenjur/taufik-report.pdf

www.e-smartschool.com/pnk/002/PNK0020019.asp

www.informatika.org/~rinaldi/Matdis/2008-2009/Makalah2008/Makalah0809-056.pdf

www.informatika.org/~rinaldi/Kriptografi/Steganografi%20dan%20Watermarking.pdf

www.itelkom.ac.id/library/index.php?view=article&catid=15%3Apemrosesan-

sinyal&id=344%3Acitra-digital&option=com_content&Itemid=15

www.jcatki.no-ip.org:8080/SDL_image/demos/lena.jpg

www.bobbemer.com

www.itelkom.ac.id/library/index.php?view=article&catid=15%3Apemroses

www.tutorialgratis.net%20Tips%20Memilih%20Format%20GambarNet.htm

Hariyanto, Bambang. (2004). *Rekayasa Sistem Berorientasi Objek*. Bandung :
Informatika.





```

program StegoImageKu;
uses
  Forms,
  Main in 'Main.pas' {MainForm},
  ChildWin in 'ChildWin.pas' {MDIChild},
  about in 'about.pas' {AboutBox},
  uProcess in 'uProcess.pas',
  uBits in 'uBits.pas',
  Unit4 in 'Unit4.pas' {Splash};
{$R *.RES}
begin
  Splash := TSplash.Create(Application); //3 baris berikut
  Splash.Show;                          //ditambah secara manual
  Splash.Update;
  while Splash.Timer1.Enabled do
    Application.ProcessMessages;
  Application.Initialize;
  Application.CreateForm(TMainForm, MainForm);
  Application.CreateForm(TAboutBox, AboutBox);
  Application.CreateForm(TSplash, Splash);
  Splash.Hide;                          //2 baris berikut ditambah secara
  Splash.Free; // menghapus form splash screen dr memory
  Application.Run;
end.

```

```

unit Unit4;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, ComCtrls, jpeg;
type
  TSplash = class(TForm)
    Timer1: TTimer;
    Image1: TImage;
    ProgressBar: TProgressBar;
    procedure Timer1Timer(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Splash: TSplash;
implementation
uses Main;
{$R *.dfm}
procedure TSplash.FormCreate(Sender: TObject);
var
  i : integer;
begin
  BorderWidth := 0;
  Show;
  Update;
  i := 7500;
  ProgressBar.Max := 0;
  while i <> 0 do
    begin
      if i = 7500 then
        begin
          ProgressBar.Max := i;
          ProgressBar.Min := 0;
          ProgressBar.Step := 1;
        end;
      if ProgressBar.Max <> 0 then
        ProgressBar.StepIt;
      dec(i);
    end;
  end;
end;

```

```

    update;
end;
procedure TSplash.Timer1Timer(Sender: TObject);
begin
    Timer1.Enabled:=false;
end;
end.

```

```

unit MAIN;
interface
uses Windows, SysUtils, Classes, Graphics, Forms, Controls, Menus,
    StdCtrls, Dialogs, Buttons, Messages, ExtCtrls, ComCtrls, StdActns,
    ActnList, ToolWin, ImgList;

```

```

type
TMainForm = class(TForm)
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    FileNewItem: TMenuItem;
    FileOpenItem: TMenuItem;
    FileCloseItem: TMenuItem;
    Window1: TMenuItem;
    Help1: TMenuItem;
    N1: TMenuItem;
    FileExitItem: TMenuItem;
    WindowCascadeItem: TMenuItem;
    WindowTileItem: TMenuItem;
    WindowArrangeItem: TMenuItem;
    HelpAboutItem: TMenuItem;
    OpenDialog: TOpenDialog;
    FileSaveItem: TMenuItem;
    FileSaveAsItem: TMenuItem;
    Edit1: TMenuItem;
    CutItem: TMenuItem;
    CopyItem: TMenuItem;
    PasteItem: TMenuItem;
    WindowMinimizeItem: TMenuItem;
    StatusBar: TStatusBar;
    ActionList1: TActionList;
    EditCut1: TEditCut;
    EditCopy1: TEditCopy;
    EditPaste1: TEditPaste;
    FileNew1: TAction;
    FileSave1: TAction;
    FileExit1: TAction;
    FileOpen1: TAction;
    FileSaveAs1: TAction;
    WindowCascade1: TWindowCascade;
    WindowTileHorizontal1: TWindowTileHorizontal;
    WindowArrangeAll1: TWindowArrange;
    WindowMinimizeAll1: TWindowMinimizeAll;
    HelpAbout1: TAction;
    FileClose1: TWindowClose;
    WindowTileVertical1: TWindowTileVertical;
    WindowTileItem2: TMenuItem;
    ToolBar2: TToolBar;
    ToolButton1: TToolButton;
    ToolButton2: TToolButton;
    ToolButton3: TToolButton;
    ToolButton4: TToolButton;
    ToolButton5: TToolButton;
    ToolButton6: TToolButton;
    ToolButton9: TToolButton;
    ToolButton7: TToolButton;
    ToolButton8: TToolButton;
    ToolButton10: TToolButton;
    ToolButton11: TToolButton;
    ImageList1: TImageList;
    HowDoesItWork1: TMenuItem;

```

```

procedure FileNew1Execute(Sender: TObject);
procedure FileOpen1Execute(Sender: TObject);
procedure HelpAbout1Execute(Sender: TObject);
procedure FileExit1Execute(Sender: TObject);
procedure HowDoesItWork1Click(Sender: TObject);
private
  { Private declarations }
  procedure CreateMDIChild(const Name: string);
public
  { Public declarations }
end;
var
  MainForm: TMainForm;
implementation
{$R *.dfm}
uses CHILDWIN, about;
procedure TMainForm.CreateMDIChild(const Name: string);
var
  Child: TMDIChild;
begin
  { create a new MDI child window }
  Child := TMDIChild.Create(Application);
  Child.Caption := Name;
  //if FileExists(Name) then Child.Memo1.Lines.LoadFromFile(Name);
end;
procedure TMainForm.FileNew1Execute(Sender: TObject);
begin
  CreateMDIChild(' + IntToStr(MDIChildCount + 1));
end;
procedure TMainForm.FileOpen1Execute(Sender: TObject);
begin
  if OpenFileDialog.Execute then
    CreateMDIChild(OpenDialog.FileName);
end;
procedure TMainForm.HelpAbout1Execute(Sender: TObject);
begin
  AboutBox.ShowModal;
end;
procedure TMainForm.HowDoesItWork1Click(Sender: TObject);
begin
  ShowMessage(
    'In a 24-bit bitmap, each pixel is made up of (surprise, surprise) a 24-bit number. Each number is composed of three 8-bit
    '+
    'numbers (the R, G and B channels). These are the intensity of the Red, Green and Blue colors that create the final color of
    the pixel.' + #13#10#13#10+
    'To hide something inside the image, we will replace the Least Significant Bit (this is, the "rightmost" bit) of each 8-bit
    channel '+
    'of every pixel, with the bits of the file we want to hide.' + #13#10#13#10 +
    'The image will lose some quality because now the colors of the pixels are not the same, but it will go unnoticed to the
    human eye.' + #13#10#13#10 +
    'Obviously, since we are storing only 3 bits per pixel, the image must have a phenomenal size to accommodate just a tiny
    little file.' +
    'For example, if we want to hide 1 MB of data, we need an image with 2,796,203 pixels, which would have a size of
    something like 2,200 x 1,320 pixels. And that is a 8.3 MB file... :O' + #13#10#13#10 +
    'We could replace the 2, 3 or 4 rightmost bits of every channel in order to increase the amount of data we can hide, but the
    quality could decrease considerably.' + #13#10#13#10 +
    'Try 8 bits per channel and you'll see what I'm talking about (amazingly, 7 normally produces intelligible -although very
    ugly- images).');
end;
procedure TMainForm.FileExit1Execute(Sender: TObject);
begin
  Close;
end;
end.

```

unit CHILDWIN;

```

interface
uses Windows, Messages, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,
    ComCtrls, ExtCtrls, ExtDlgs, Dialogs, Buttons, jpeg;
type
  TMDIChild = class(TForm)
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    OpenPictureDialog1: TOpenPictureDialog;
    SavePictureDialog1: TSavePictureDialog;
    OpenTextFileDialog1: TOpenTextFileDialog;
    SaveTextFileDialog1: TSaveTextFileDialog;
    OpenFileDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    GroupBox3: TGroupBox;
    Edit1: TEdit;
    btnBrowse: TBitBtn;
    GroupBox4: TGroupBox;
    lblNote: TLabel;
    lblBPC: TLabel;
    udBitsPerChannel: TUpDown;
    edtBPC: TEdit;
    GroupBox5: TGroupBox;
    btnEncrypt: TBitBtn;
    btnClear: TBitBtn;
    btnOpenEncode: TBitBtn;
    pb1: TProgressBar;
    GroupBox1: TGroupBox;
    ScrollBox1: TScrollBox;
    Image1: TImage;
    GroupBox7: TGroupBox;
    BitBtn6: TBitBtn;
    btnDecrypt: TBitBtn;
    btnDecode: TBitBtn;
    GroupBox2: TGroupBox;
    ScrollBox2: TScrollBox;
    Image2: TImage;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormCreate(Sender: TObject);
    procedure btnBrowseClick(Sender: TObject);
    procedure btnEncryptClick(Sender: TObject);
    procedure btnOpenEncodeClick(Sender: TObject);
    procedure btnDecryptClick(Sender: TObject);
    procedure btnDecodeClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
  var
    MDIChild: TMDIChild;
implementation
uses MAIN, uProcess, about;
{$R *.dfm}
procedure TMDIChild.btnBrowseClick(Sender: TObject);
begin
  if OpenFileDialog1.Execute then begin
    Edit1.Text := OpenFileDialog1.FileName;
  end;
end;
procedure TMDIChild.btnClearClick(Sender: TObject);
begin
  image1.Picture.Bitmap:=nil;
  Edit1.Clear;
end;
procedure TMDIChild.btnDecodeClick(Sender: TObject);
var
  NamaFile : string;

```



```

begin
try
    OpenPictureDialog1.Filter := 'Bitmaps (*.bmp)|*.bmp';
    OpenPictureDialog1.DefaultExt := '*.bmp';
if OpenPictureDialog1.Execute then
    begin
        NamaFile := OpenPictureDialog1.FileName;
        Image2.Picture.LoadFromFile(NamaFile);
    end;
finally
    end;
end;
procedure TMDIChild.btnDecryptClick(Sender: TObject);
begin
if SaveDialog1.Execute then
    try
        btnDecrypt.Enabled:= False;
        lblBPC.Enabled:= False;
        lblNote.Enabled:= False;
        edtBPC.Enabled:= False;
        udBitsPerChannel.Enabled:= False;
        pb1.Show;
        Decrypt(OpenPictureDialog1.FileName, SaveDialog1.FileName, pb1);
        MessageBox( Application.Handle, PChar('Done!' + #13#10#13#10 + 'The file "' +
            SaveDialog1.FileName + '" was created.'),
            'StegaImage', MB_OK + MB_ICONINFORMATION);
    finally
        pb1.Hide;
        btnDecrypt.Enabled:= True;
        lblBPC.Enabled:= True;
        lblNote.Enabled:= True;
        edtBPC.Enabled:= True;
        udBitsPerChannel.Enabled:= True;
    end;
except
    on e: Exception do
        MessageBox(Handle, PChar('An error has occurred.' + #13#10#13#10 + e.Message), 'StegaImageKu', MB_OK +
MB_ICONSTOP);
    end;
end;
procedure TMDIChild.btnEncryptClick(Sender: TObject);
begin
try
    if Trim(Edit1.Text) = '' then begin
        MessageDlg('Content harus diisi', mtWarning, [mbOK], 0);
        Exit;
    end;
if SavePictureDialog1.Execute then
    try
        try
            Image1.Picture.Bitmap:=nil;
            btnEncrypt.Enabled:= False;
            lblBPC.Enabled:= False;
            lblNote.Enabled:= False;
            edtBPC.Enabled:= False;
            udBitsPerChannel.Enabled:= False;
            btnBrowse.Enabled:= False;
            btnOpenEncode.Enabled:= False;
            btnClear.Enabled:= False;
            Edit1.Clear;
            pb1.Show;
            Encrypt(OpenDialog1.FileName, OpenPictureDialog1.FileName, SavePictureDialog1.FileName,
            udBitsPerChannel.Position, pb1);
            MessageBox( Application.Handle, PChar('Done!' + #13#10#13#10 + 'The file "' +
                SavePictureDialog1.FileName + '" was created.'),
                'StegaImageKu', MB_OK + MB_ICONINFORMATION);
        finally
            pb1.Hide;

```

```

    btnEncrypt.Enabled:= True;
    lblBPC.Enabled:= True;
    lblNote.Enabled:= True;
    edtBPC.Enabled:= True;
    udBitsPerChannel.Enabled:= True;
    btnBrowse.Enabled:= True;
    btnOpenEncode.Enabled:= True;
    btnClear.Enabled:= True;
end;
except
on e: Exception do
    MessageBox(Handle, PChar('An error has occurred.' + #13#10#13#10 + e.Message), 'StegaImageKu', MB_OK +
MB_ICONSTOP);
end;
finally
end;
end;
end;
procedure TMDIChild.btnOpenEncodeClick(Sender: TObject);
var
    NamaFile : string;
begin
    try
        OpenPictureDialog1.Filter := 'Bitmaps (*.bmp)|*.bmp';
        OpenPictureDialog1.DefaultExt := '*.bmp';
        if OpenPictureDialog1.Execute then
            begin
                NamaFile := OpenPictureDialog1.FileName;
                Image1.Picture.LoadFromFile(NamaFile);
            end;
        finally
            end;
        end;
    end;
    procedure TMDIChild.FormClose(Sender: TObject; var Action: TCloseAction);
    begin
        Action := caFree;
    end;
    procedure TMDIChild.FormCreate(Sender: TObject);
    begin
        try
        except
            on E: Exception do begin
                MessageDlg(E.Message, mtError, [mbOK], 0);
            end;
        end;
        end;
    end.
end.

```

```

unit uProcess;
interface
uses ComCtrls, SysUtils;
procedure Encrypt(const SourceFile, SourceBitmap, Destination: string; BitsPerChannel: LongInt; ProgressBar:
TProgressBar);
procedure Decrypt(const SourceFile, DestFile: string; ProgressBar: TProgressBar);
implementation
uses Windows, Graphics, Classes, uBits, Forms;
// One byte for every channel of the RGB trio
type pRGBArray= ^TRGBArray;
    TRGBArray= array [1..3] of Byte;
    procedure ProcessEncrypt(Bitmap: TBitmap; Source: TFileStream; Destination: string; BPC: LongInt; ProgressBar:
TProgressBar);
var SourceIndex, SourceSize: LongInt;
    BitIndex, PixelBitIndex: LongInt;
    SourceByte: Byte;
    PixelsRow: pRGBArray;
    RGBIndex: Integer;
    PixelsRowMax, PixelsRowIndex, CurrentRow: Integer;
// A procedure inside another is quite ugly, but we avoid passing a lot of parameters when we call it
    procedure CheckNextPixel;

```

```

begin
  if (RGBIndex <= 3) and (PixelBitIndex + 1 < BPC) then // We're OK, go for the next bit
    Inc(PixelBitIndex)
  else if RGBIndex < 3 then // Switch to next RGB channel
    begin
      Inc(RGBIndex);
      PixelBitIndex:= 0;
    end
  else if RGBIndex = 3 then // Load next pixel
    begin
      PixelBitIndex:= 0;
      RGBIndex:= 1;
      if PixelsRowIndex = PixelsRowMax then // We used all pixels in this row
        begin
          Inc(CurrentRow);
          PixelsRow:= Bitmap.ScanLine[CurrentRow];
          PixelsRowIndex:= 1;
        end
      else // We still have pixels left in this row
        begin
          Inc(PixelsRowIndex);
          Inc(PixelsRow); // Increment the pointer so it points to the next pixel
        end;
      end;
    end;
  end;
begin { ProcessEncrypt }
  {-- STORE THE BITS PER CHANNEL VALUE--}
  PixelsRow:= Bitmap.ScanLine[0];
  // We use 2 bits in the B channel to bring the count to 4 bits, so we can store BPC = 8, value that usually brings
  // devastation to the image quality...
  SetBitAt(PixelsRow^[1], 0, GetBitAt(BPC, 0));
  SetBitAt(PixelsRow^[1], 1, GetBitAt(BPC, 1));
  SetBitAt(PixelsRow^[2], 0, GetBitAt(BPC, 2));
  SetBitAt(PixelsRow^[3], 0, GetBitAt(BPC, 3));
  // Initialize
  PixelsRowMax:= Bitmap.Width;
  CurrentRow:= 0;
  PixelBitIndex:= 0;
  PixelsRowIndex:= 2;
  RGBIndex:= 1;
  Inc(PixelsRow);
  {-- STORE THE ACTUAL LENGTH OF THE DATA --}
  SourceSize:= Source.Size;
  for BitIndex:= 0 to SizeOf(SourceSize) * 8 - 1 do
    begin
      SetBitAt(PixelsRow^[RGBIndex], PixelBitIndex, GetBitAt(SourceSize, BitIndex));
      CheckNextPixel;
    end;
  {-- STORE THE DATA --}
  // Copy the bits from every byte in the source stream onto the bits in the pixels
  Source.Seek(0, soFromBeginning);
  for SourceIndex:= 0 to SourceSize - 1 do
    begin
      if (SourceIndex + 1) mod 10 = 0 then
        begin
          ProgressBar.StepIt;
          Application.ProcessMessages;
        end;
      Source.Read(SourceByte, 1);
      for BitIndex:= 0 to 7 do
        begin
          SetBitAt(PixelsRow^[RGBIndex], PixelBitIndex, GetBitAt(SourceByte, BitIndex));
          CheckNextPixel;
        end;
      end; // for SourceIndex
    end;
  end;
procedure Encrypt(const SourceFile, SourceBitmap, Destination: string; BitsPerChannel: LongInt; ProgressBar:
TProgressBar);
var Bitmap: TBitmap;

```

```

    sSource: TFileStream;
begin
    ProgressBar.Position:= 0;
    ProgressBar.Step:= 1;
    Bitmap:= TBitmap.Create;
    sSource:= nil;
    try
        Bitmap.LoadFromFile(SourceBitmap);
        if Bitmap.PixelFormat <> pf24bit then
            raise Exception.Create('The image must have a 24-bit depth.');
```

sSource:= TFileStream.Create(SourceFile, fmOpenRead);

if sSource.Size = 0 then

raise Exception.Create('The source file is 0 bytes. There's nothing to hide.');

if sSource.Size * 8 + SizeOf(LongInt) * 8 + 1 > Bitmap.Width * Bitmap.Height * 3 * BitsPerChannel then

raise Exception.Create('The image is not big enough to accommodate the file.');

// + 1: The BitsPerChannel value is stored in the first pixel

// SizeOf(LongInt) * 8: we'll store the amount of actual encrypted data in the first pixels after BitsPerChannel

ProgressBar.Max:= sSource.Size div 10;

ProcessEncrypt(Bitmap, sSource, Destination, BitsPerChannel, ProgressBar);

Bitmap.SaveToFile(Destination);

finally

Bitmap.Free;

if Assigned(sSource) then sSource.Free;

end;

end;

{-----}

```

procedure ProcessDecrypt(Bitmap: TBitmap; Destination: TFileStream; ProgressBar: TProgressBar);
var DataSize, dataIndex: LongInt;
    Data, BitIndex: Byte;
    PixelsRow: pRGBArray;
    PixelsRowMax, PixelsRowIndex, CurrentRow, MaxRows: Integer;
    PixelBitIndex: LongInt;
    RGBIndex: Integer;
    BPC: LongInt;
// A procedure inside another (and basically the same as the one in ProcessEncrypt) is quite ugly, but we avoid passing
// a lot of parameters when we call it
procedure CheckNextPixel;
begin
    if (RGBIndex <= 3) and (PixelBitIndex + 1 < BPC) then // We're OK, go for the next bit
        Inc(PixelBitIndex)
    else if RGBIndex < 3 then // Switch to next RGB channel
        begin
            Inc(RGBIndex);
            PixelBitIndex:= 0;
        end
    else if RGBIndex = 3 then // Load next pixel
        begin
            PixelBitIndex:= 0;
            RGBIndex:= 1;
            if PixelsRowIndex = PixelsRowMax then // We used all pixels in this row
                begin
                    Inc(CurrentRow);
                    if CurrentRow > MaxRows then
                        raise Exception.Create('The end of the image was reached while trying to read the hidden information.' + #13#10+
                            'This is probably caused by an image that doesn't contain any hidden data.');
```

PixelsRow:= Bitmap.ScanLine[CurrentRow];

PixelsRowIndex:= 1;

end

else // We still have pixels left in this row

begin

Inc(PixelsRowIndex);

Inc(PixelsRow); // Increment the pointer so it points to the next pixel

end;

end;

end;

begin {ProcessDecrypt}

{-- GET THE BITS PER CHANNEL VALUE --}

```

PixelsRow:= Bitmap.ScanLine[0];
BPC:= 0;
SetBitAt(BPC, 0, GetBitAt(PixelsRow^[1], 0));
SetBitAt(BPC, 1, GetBitAt(PixelsRow^[1], 1));
SetBitAt(BPC, 2, GetBitAt(PixelsRow^[2], 0));
SetBitAt(BPC, 3, GetBitAt(PixelsRow^[3], 0));
if (BPC < 1 ) or (BPC > 8) then
  raise Exception.Create('The BitsPerChannel value is not in the range 1-8.' + #13#10 +
    'This is probably caused by an image that doesn't contain any hidden data.');
```

// Initialize

```

PixelsRowMax:= Bitmap.Width;
MaxRows:= Bitmap.Height - 1;
CurrentRow:= 0;
PixelBitIndex:= 0;
PixelsRowIndex:= 2;
RGBIndex:= 1;
Inc(PixelsRow);
{-- GET THE LENGTH OF HIDDEN DATA --}
for BitIndex:= 0 to SizeOf(DataSize) * 8 - 1 do
  begin
    SetBitAt(DataSize, BitIndex, GetBitAt(PixelsRow^[RGBIndex], PixelBitIndex));
    CheckNextPixel;
  end;

if DataSize <= 0 then
  raise Exception.Create('The stored size of the hidden data is not correct.' + #13#10 +
    'This is probably caused by an image that doesn't contain any hidden data.');
```

ProgressBar.Max:= DataSize div 10;

{-- EXTRACT THE ACTUAL DATA --}

```

for dataIndex:= 1 to DataSize do
  begin
    if dataIndex mod 10 = 0 then
      begin
        ProgressBar.StepIt;
        Application.ProcessMessages;
      end;
    for BitIndex:= 0 to 7 do
      begin
        SetBitAt(Data, BitIndex, GetBitAt(PixelsRow^[RGBIndex], PixelBitIndex));
        CheckNextPixel;
      end;
    Destination.Write(Data, 1);
  end; //for dataIndex
end;

procedure Decrypt(const SourceFile, DestFile: string; ProgressBar: TProgressBar);
var Bitmap: TBitmap;
    Destination: TFileStream;
begin
  ProgressBar.Position:= 0;
  ProgressBar.Step:= 1;
  Bitmap:= TBitmap.Create;
  Destination:= nil;
  try
    try
      if FileExists(DestFile) then DeleteFile(PAnsiChar(DestFile));
      Destination:= TFileStream.Create(DestFile, fmCreate);
      Bitmap.LoadFromFile(SourceFile);
      if Bitmap.PixelFormat <> pf24bit then
        raise Exception.Create('The image doesn't have a 24-bit depth. It surely hasn't been created by this program.');
```

ProcessDecrypt(Bitmap, Destination, ProgressBar);

```

  finally
    Bitmap.Free;
    if Assigned(Destination) then Destination.Free;
  end;
except
  if FileExists(DestFile) then
    DeleteFile(PChar(DestFile));
  raise;
end;
```

```
end;
end.
```

```
unit uBits;
interface
procedure SetBitAt(var Variable: LongInt; Position: Byte; Value: Boolean); overload;
procedure SetBitAt(var Variable: Byte; Position: Byte; Value: Boolean); overload;
function GetBitAt(Variable: LongInt; Position: Byte): Boolean;
implementation
procedure SetBitAt(var Variable: LongInt; Position: Byte; Value: Boolean);
begin
  if Value then
    Variable:= Variable or (1 shl Position)
  else
    Variable:= Variable and ((1 shl Position) xor $FFFFFFF);
end;
procedure SetBitAt(var Variable: Byte; Position: Byte; Value: Boolean);
begin
  if Value then
    Variable:= Variable or (1 shl Position)
  else
    Variable:= Variable and ((1 shl Position) xor $FF);
end;
function GetBitAt(Variable: LongInt; Position: Byte): Boolean;
begin
  if Variable and (1 shl Position) <> 0 then
    Result:= True
  else
    Result:= False;
end;
end.
```

```
unit about;
interface
uses Windows, Classes, Graphics, Forms, Controls, StdCtrls,
  Buttons, ExtCtrls, jpeg;
type
  TAboutBox = class(TForm)
    Panel1: TPanel;
    OKButton: TButton;
    ProgramIcon: TImage;
    ProductName: TLabel;
    Version: TLabel;
    Copyright: TLabel;
    Comments: TLabel;
    Label1: TLabel;
    Label2: TLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  AboutBox: TAboutBox;
implementation
{$R *.dfm}
end.
```

Analisis dan Perancangan Aplikasi Steganografi pada Citra Digital Menggunakan Metode LSB (*Least Significant Bit*)

Adiria^a, Arini, MT^b dan Qurrotul Aini, MT^c

^aMahasiswa Fakultas Sains dan Teknologi
Universitas Islam Negeri Syarif Hidayatullah Jakarta
Tel : (021) 90498539
e-mail : [idunk_adhie@yahoo.com](mailto:adunk_adhie@yahoo.com)

^bDosen Fakultas Sains dan Teknologi
Universitas Islam Negeri Syarif Hidayatullah Jakarta
Tel : 081317545464
e-mail : arinizoel@com

^cDosen Fakultas Sains dan Teknologi
Universitas Islam Negeri Syarif Hidayatullah Jakarta
Tel : 081519694744
e-mail : Atafamily@yahoo.com

ABSTRAK

Di seluruh dunia, internet (*interconnection network*) sudah berkembang menjadi salah satu media komunikasi data yang sangat populer. Dengan berbagai teknik pengambilan informasi secara ilegal yang berkembang, banyak orang yang mencoba untuk mengakses informasi yang bukan haknya. Berbagai macam teknik untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak telah banyak dilakukan dalam upaya mengamankan suatu data penting. Steganografi adalah salah satu teknik menyembunyikan file pesan agar bagi orang awam tidak menyadari keberadaan dari file pesan yang disembunyikan. Teknik ini sering digunakan untuk menghindari kecurigaan orang dan menghindari keinginan orang untuk mengetahui isi file pesan tersebut. Citra Digital adalah salah satu media yang paling umum dikenal oleh masyarakat. Penelitian ini bertujuan menganalisis dan merancang suatu aplikasi steganografi sebagai salah satu teknik pengamanan file pesan. Pada teknik steganografi ini digunakan metode LSB (*Least Significant Bit*) yaitu menyembunyikan byte-byte data atau informasi pada bit terakhir pada citra digital, sehingga tidak terjadi perubahan ukuran dan bentuk terhadap citra digital secara kasat mata, data atau informasi pun dapat dikembalikan seperti semula tanpa ada perubahan ukuran dan bentuknya.

Kata Kunci: Aplikasi, Steganografi, Citra Digital, Steganografi, LSB

1. PENDAHULUAN

Di seluruh dunia, internet (*interconnection network*) sudah berkembang menjadi salah satu media komunikasi data yang sangat populer. Kemudahan dalam penggunaan dan fasilitas yang lengkap merupakan keunggulan yang dimiliki oleh internet dan bukan rahasia umum di kalangan masyarakat pengguna internet pada saat sekarang ini. Namun, seiring dengan berkembangnya media internet dan aplikasi yang menggunakan internet semakin bertambah pula kejahatan dalam sistem informasi. Dengan berbagai teknik pengambilan informasi secara ilegal yang berkembang, banyak yang mencoba untuk mengakses informasi yang

bukan haknya. Untuk itu, sejalan dengan berkembangnya media internet yang sangat cepat, harus juga diikuti dengan perkembangan pengamanan dalam sistem informasi yang berada dalam media internet tersebut.

Berbagai macam teknik yang digunakan untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak telah banyak dilakukan dalam upaya mengamankan suatu data penting dengan menggunakan sistem kriptografi yang melakukan enkripsi sebelum data penting tersebut ditransmisikan. Tindakan pengamanan menggunakan cara tersebut ternyata dianggap belum cukup dalam mengamankan suatu data karena adanya peningkatan kemampuan

komputasi. Berbeda dengan teknik kriptografi, steganografi menyembunyikan pesan rahasia agar bagi orang awam tidak menyadari keberadaan dari pesan yang disembunyikan. Teknik ini sering digunakan untuk menghindari kecurigaan orang dan menghindari keinginan orang untuk mengetahui isi pesan rahasia tersebut. Caranya dengan menyembunyikan informasi rahasia di dalam suatu wadah penampung informasi dengan sedemikian rupa sehingga keberadaan informasi rahasia yang ditempelkan tidak terlihat. Wadah penampung informasi tersebut dapat berbentuk berbagai jenis *file* multimedia *digital* seperti teks, citra, audio, video. Dalam tugas akhir ini, peneliti membuat analisis dan merancang aplikasi steganografi merupakan solusi dari permasalahan tersebut. Dengan penggunaan teknik ini, data informasi dapat kita sembunyikan di dalam media digital yang kita punya.

2. LANDASAN TEORI

2.1 Pengertian Analisis

Analisis berkaitan dengan pemahaman dan pemodelan aplikasi serta domain dimana aplikasi beroperasi. Masukan awal fase analisis adalah pernyataan masalah yang mendeskripsikan masalah yang ingin diselesaikan dan menyediakan pandangan konseptual terhadap sistem yang diusulkan. Sebutan lengkap analisis adalah analisis kebutuhan perangkat lunak (*software requirement analysis*). Analisis adalah mendaftarkan apa-apa yang harus dipenuhi oleh sistem perangkat lunak, bukan mengenai bagaimana sistem perangkat lunak melakukannya (Hariyanto, 2004).

2.2 Pengertian Perancangan

Perancangan merupakan penghubung antara spesifikasi kebutuhan dan implementasi. Perancangan merupakan rekayasa representasi yang berarti terhadap sesuatu yang hendak dibangun. Hasil perancangan harus dapat ditelusuri sampai ke spesifikasi kebutuhan dan dapat diukur kualitasnya berdasar kriteria-kriteria rancangan yang bagus. Perancangan menekankan pada solusi logis mengenai cara sistem memenuhi kebutuhan (Hariyanto, 2004).

2.3 Steganografi

2.3.1 Sejarah Steganografi

Usia steganografi hampir setara usia kriptografi. Steganografi sudah dikenal oleh bangsa Yunani sejak lama. Herodatus, seorang penguasa Yunani, mengirimkan pesan rahasia menggunakan kepala budak atau prajurit sebagai media. Caranya, rambut budak dibotaki, lalu pesan rahasia ditulis pada kulit kepala budak. Setelah rambut-rambut budak tumbuh cukup banyak (yang berarti menutupi pesan rahasia), budak tersebut dikirim ke tempat tujuan pesan untuk membawa pesan rahasia di kepalanya. Di tempat penerima kepala budak dibotaki kembali untuk membaca pesan yang tersembunyi di balik rambutnya. Pesan tersebut berisi peringatan tentang invasi dari Bangsa Persia.

Bangsa Romawi mengenal steganografi dengan menggunakan tinta tak-nampak (*invisible ink*) untuk menulis pesan. Tinta tersebut dibuat dari campuran sari buah, susu, dan cuka. Jika tinta digunakan untuk menulis maka tulisannya tidak tampak. Tulisan di atas kertas dapat dibaca dengan cara memanaskan kertas tersebut (Munir, 2006).

2.3.2 Pengertian Steganografi

Steganografi berasal dari Bahasa Yunani, yaitu “steganos” yang artinya “tulisan tersembunyi (*covered writing*)” dan “graphos” yang berarti tulisan. Steganografi adalah ilmu dan seni menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui (Munir, 2006:). Sedangkan menurut Doni Ariyus, steganografi merupakan cabang ilmu yang mempelajari tentang bagaimana menyembunyikan suatu informasi “rahasia” di dalam informasi lainnya (Ariyus, 2006).

2.3.2 Metode Steganografi

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

1. Ranah spasial (*waktu*) (*spasial/time domain*), Teknik ini memodifikasi langsung nilai *byte* dari *coverttext* (nilai *byte* dapat merepresentasikan intensitas/warna *pixel* atau amplitudo). Contoh metode yang tergolong ke dalam teknik ranah spasial adalah metode *LSB*.
2. Ranah *transform* (*transform domain*)

Teknik ini memodifikasi langsung hasil transformasi frekuensi sinyal. Contoh metode yang tergolong ke dalam teknik ranah frekuensi adalah *spread spectrum* (Munir, 2006).

Sedangkan ada empat jenis metode Steganografi, yaitu:

1. *Algorithms and Transformation*

Algoritma *compression* (kompresi) 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*).

2. *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 menggambar pesan yang lebih besar.

3. *Spread Spectrum method*

Spread Spectrum steganografi terpecah sebagai pesan yang diacak (*encrypt*) 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) (<http://www.students.if.itb.ac.id>). Contoh dari penyebaran bit-bit informasi dapat dilihat pada Gambar 2.4. Faktor pengali dilambangkan dengan *cr* yang bernilai skalar. Panjang bit-bit hasil penyebaran ini menjadi *cr* kali panjang bit-bit awal.

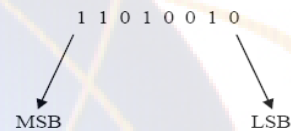
4. *Least Significant Bit* (LSB)

Metode yang digunakan untuk menyembunyikan pesan pada media digital tersebut berbeda-beda. Contohnya pada *file image* pesan dapat disembunyikan dengan menggunakan cara menyisipkannya pada bit rendah atau bit yang paling kanan (LSB) pada data *pixel* yang menyusun *file* tersebut. Seperti kita ketahui untuk *file bitmap* 24 bit maka setiap *pixel* (titik) pada gambar tersebut terdiri dari susunan tiga warna

yaitu 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 (<http://doank29.multiply.com>).

2.4 Metode LSB (*Least Significant Bit*)

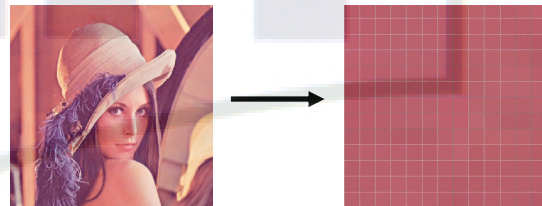
Pada susunan bit di dalam sebuah *byte* (1 *byte* = 8 bit), ada bit yang paling berarti (*most significant bit* atau MSB) dan bit yang paling kurang berarti (*least significant bit* atau LSB) (<http://www.itelkom.ac.id>). Terlihat pada Gambar 2.5. yang menggambarkan contoh nilai biner pada 8 bit.



Gambar 2.5 *most significant bit* (MSB) dan *least significant bit* (LSB)

Sumber: (<http://doank29.multiply.com>)

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. Misalkan *byte* tersebut menyatakan warna merah, maka perubahan satu bit LSB tidak mengubah warna merah tersebut secara berarti, dan mata manusia tidak dapat membedakan perubahan yang sangat kecil itu. Misalkan ada sebuah gambar dengan nilai *pixel*nya sebagai berikut :



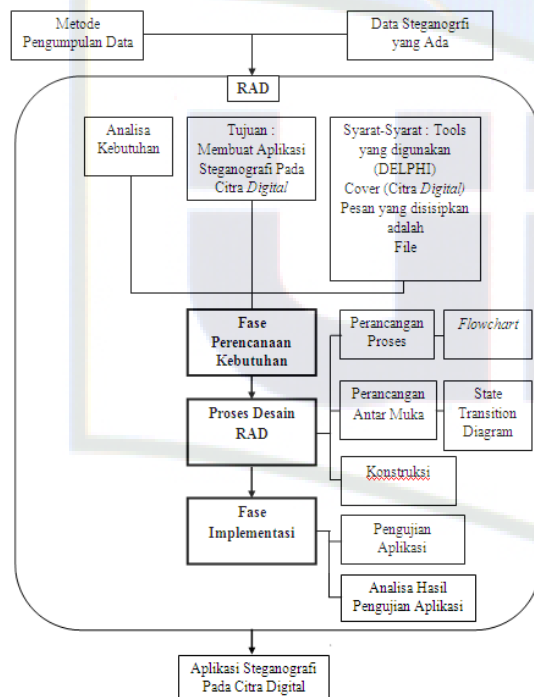
Gambar 2.6 Proses *File Image* Menjadi Kumpulan *Pixel-Pixel*

Pada dasarnya sebuah gambar *bitmap* merupakan kumpulan dari titik-titik yang disebut *pixel*. *Pixel-pixel* disetiap gambar mempunyai nilai berbeda-beda.



Gambar 3.1 Skema Sistem Model RAD

sistem *Rapid Application Design* (RAD) terdiri dari tiga tahapan yaitu perencanaan syarat-syarat, desain *workshop* RAD dan implementasi.



Gambar 3.2 Ilustrasi Metode Penelitian Pengembangan Aplikasi Steganografi pada Citra Digital.

4. ANALISIS DAN PERANCANGAN

4.1 Fase Perencanaan Syarat – Syarat

Perencanaan syarat-syarat terdiri atas analisis kebutuhan, tujuan dan syarat-syarat. Proses ini dilakukan untuk mengetahui apa saja syarat-syarat dan kebutuhan yang dibutuhkan dalam menganalisis untuk tujuan dari perancangan aplikasi ini.

4.1.1 Analisis Kebutuhan

Pada tahap ini dilakukan suatu perbandingan dari beberapa aplikasi steganografi yang sudah ada. Perbandingan dilakukan untuk tujuan-tujuan aplikasi serta untuk mengetahui syarat-syarat informasi yang ditimbulkan dari tujuan-tujuan tersebut. Perbandingan dilakukan dengan mempertimbangkan kemudahan penggunaan dan fitur masing-masing aplikasi steganografi yang pernah ada. Orientasi dalam tahap ini adalah bagaimana cara menyelesaikan masalah yang akan terjadi dalam mencapai tujuan perancangan aplikasi steganografi yang akan dilakukan.

4.1.2 Menentukan Tujuan

Tujuan dari perancangan ini adalah membuat suatu aplikasi steganografi menggunakan file dalam bentuk citra digital/gambar berformat *.bmp, sebagai media penampung untuk menyimpan file serta untuk memberikan salah satu solusi dalam pengamanan file.

4.1.3 Menentukan syarat-syarat

Syarat-syarat untuk mencapai tujuan dalam pengembangan aplikasi steganografi pada citra digital terdiri dari perangkat lunak dan perangkat yang spesifikasinya adalah sebagai berikut:

Perangkat Lunak :

1. Windows Vista Black Edition 2009
2. Delphi 2007 win32
3. Inno Setup 5
4. Matlab R2008b

Perangkat Keras :

1. Processor intel Core 2 duo
2. Harddisk 250GB
3. RAM 2GB

4. LCD *widescreen display* dengan resolusi 1280x800 *pixel*
5. DVD RW

Media :

1. *File Image *.bmp*
2. *File Pesan *.txt, *.doc, *.xsl, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav.*

4.2 Fase Desain RAD

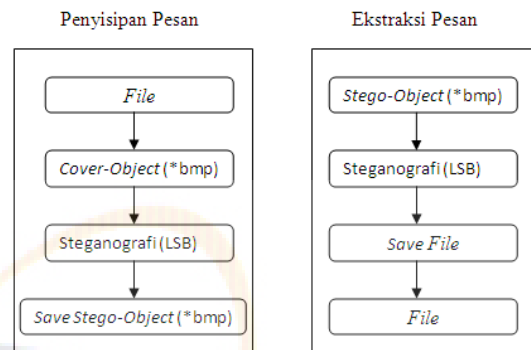
Desain RAD terdiri atas perancangan proses, dan perancangan antar muka. Perancangan proses dilakukan untuk merancang alur proses di dalam program sedangkan perancangan antar muka dilakukan untuk mempermudah pengguna menggunakan hasil dari aplikasi ini.

4.2.1 Perancangan Proses

Tahap perancangan proses dilakukan untuk perancangan, evaluasi dan memperbaiki sistem sesuai dengan kebutuhan, agar sistem yang sedang di buat dapat dimanfaatkan secara optimal. Aplikasi ini menggunakan metode LSB (*Least Significant Bit*), untuk teknik steganografi dalam aplikasi ini karena ukuran pesan yang dapat disisipkan ke media penampungnya relatif besar dengan mengganti setiap LSB dari media penampungnya. Selain itu metode LSB adalah metode yang paling mudah diaplikasikan dibandingkan metode steganografi yang lainnya.

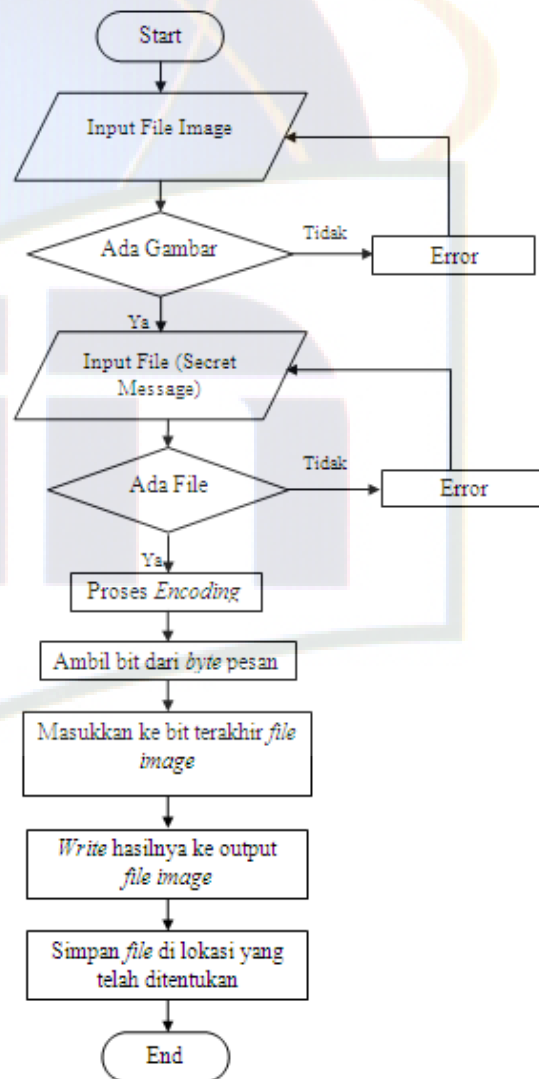
Pada penyembunyian pesan, pesan dapat di ambil/ *copy* yang berupa *file* seperti *.txt, *.doc, *.xsl, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav. kemudian disisipkan menggunakan algoritma LSB ke media penampungnya. Sehingga menghasilkan *stego-object* berupa *file *.bmp* yang telah disisipi pesan. Tahapan selanjutnya adalah proses ekstraksi. Pada tahap ini atau *file* disembunyikan dipisahkan dari media penampungnya menggunakan teknik steganografi dengan metode LSB, sehingga menjadi *file* pesan yang dapat dibaca.

Untuk memperjelas gambaran proses penyembunyian dan ekstraksi *file* pesan dapat dilihat pada Gambar 4.1.

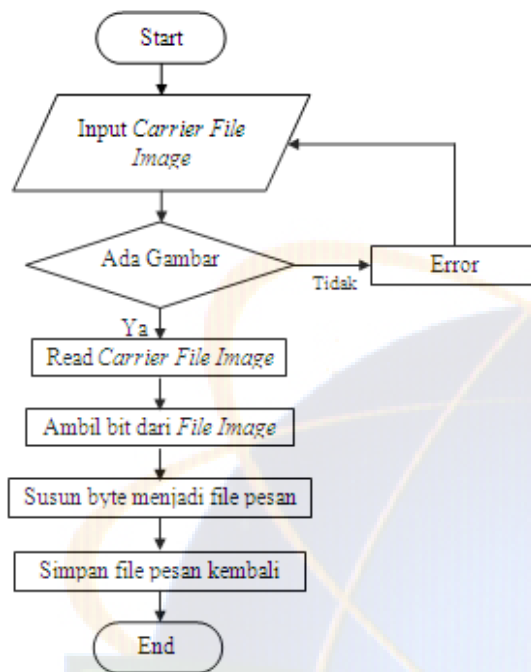


Gambar 4.1 Proses Penyisipan dan Ekstraksi *File* Pesan

4.2.2 Flowchart Aplikasi Steganografi

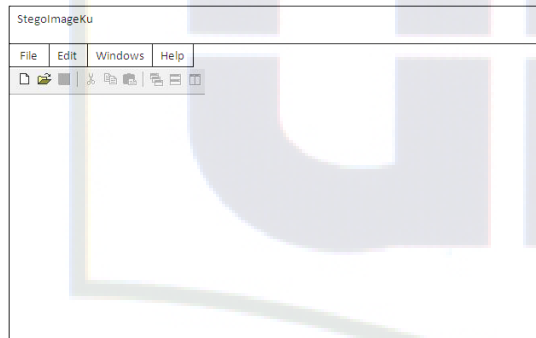


Gambar 4.2 Flowchart Proses Penyisipan Pesan Rahasia



Gambar 4.3 Flowchart Proses Ekstraksi Pesan Rahasia

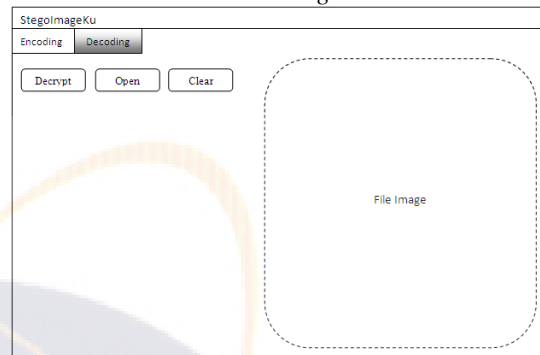
4.2.3 Perancangan Antar Muka



Gambar 4.4 Layout Form Induk



Gambar 4.5 Rancangan Form Utama Tab Encoding

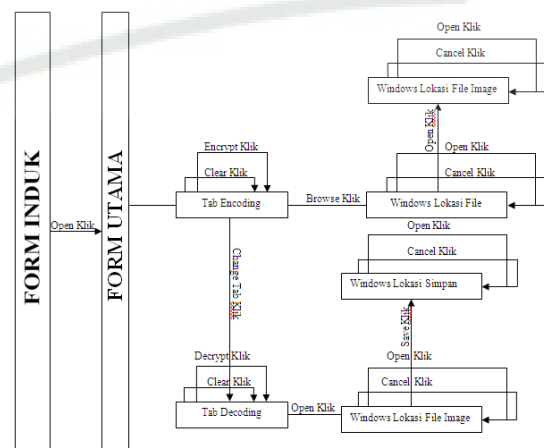


Gambar 4.6 Rancangan Form Utama Tab Decoding



Gambar 4.7 Perancangan Form About

4.2.4 STD (State Transition Diagram)



Gambar 4.8 State Transition Diagram Aplikasi Steganografi pada Citra Digital

4.3 Fase Implementasi

Dalam tahapan ini akan dilakukan pengujian dan analisis pengujian terhadap aplikasi StegoImage ini yang bertujuan untuk mengetahui tingkat keberhasilan dari aplikasi dalam mencapai hasil dan tujuan yang diinginkan.

4.3.1 Instalasi Aplikasi

Program instalasi aplikasi StegoImage ini menggunakan *software* Inno Setup 5. Dalam penggunaannya, *software* ini sangat mudah digunakan. Apabila pengguna tidak bisa membuat *script/ coding* secara langsung, program ini menyediakan menu “create a new script file using the script wizard”, jadi pengguna hanya mengikuti alur dari aplikasi dan selanjutnya aplikasi yang membuatkan script untuk proses instalasinya. Di bawah ini adalah langkah-langkah instalasi aplikasi StegoImage ke dalam komputer pengguna:

1. Pilih program StegoImage Setup.exe, kemudian akan ada tampilan pembuka dari proses instalasi ini menggunakan bahasa Inggris. Pilih tombol *next* untuk melanjutkan instalasi, atau *cancel* untuk membatalkan instalasi.
2. Jika user menekan tombol *next* maka akan muncul halaman *license agreement*, yaitu *form* persetujuan yang menanyakan kesetujuan *user* atas semua konsekuensi jika menggunakan program ini. Apabila user memilih “I accept the agreement” yang berarti setuju maka proses instalasi akan berlanjut ke tahap selanjutnya. Apabila pengguna memilih “I don’t accept the agreement” yang berarti tidak setuju maka proses instalasi berhenti.
3. Tampilan untuk menentukan lokasi aplikasi StegoImage disimpan, akan langsung mengarah ke C:\Program Files\StegoImageKu, tetapi *user* dapat mengubah lokasi dengan cara menekan tombol *browse*. Di halaman ini juga terdapat informasi ukuran program yang akan diinstal.
4. Tampilan untuk menyimpan *shortcut folder* StegoImageKu di *start menu*.
5. Tampilan untuk menampilkan *shortcut* aplikasi StegoImage di *desktop*.

6. Tampilan proses instalasi, apabila informasi yang dibutuhkan sudah lengkap maka program siap untuk diinstal.
7. Tampilan informasi bahwa program telah selesai diinstal, dan aplikasi StegoImage akan muncul sebagai tanda bahwa proses instalasi telah berhasil.

4.3.2 Analisis Penyembunyian File

Tabel 4.1 Tabel Uji Penyembunyian File

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Monalisa.bmp	419454	Serbu.txt	265	MonalisaStego1.bmp	419454	Serbu.txt	265
Monalisa.bmp	419454	Gadis.txt	10977	MonalisaStego2.bmp	419454	Gadis.txt	10977
Monalisa.bmp	419454	Daftar Isi.doc	52224 52431,75	MonalisaStego3.bmp	419454	Daftar Isi.doc	52224

4.3.3 Analisis Format File Apa Saja yang Dapat Disisipkan ke File Image

Tabel 4.2 Tabel Uji Format File Apa Saja yang Dapat Disisipkan ke File Image

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Pesawat.bmp	1440054	Index.ppt	163840	Pesawat1.bmp	1440054	Index1.ppt	163840
Pesawat.bmp	1440054	19SC096.JPG	33848	Pesawat2.bmp	1440054	19SC0961.JPG	33848
Pesawat.bmp	1440054	AbstrakHB.pdf	65536	Pesawat3.bmp	1440054	AbstrakHB.pdf	65536
Pesawat.bmp	1440054	Acctspay.mdb	102400	Pesawat4.bmp	1440054	Acctspay.mdb	102400
Pesawat.bmp	1440054	Biodata.doc	31744	Pesawat5.bmp	1440054	Biodata.doc	31744
Pesawat.bmp	1440054	browsesub.php	7290	Pesawat6.bmp	1440054	browsesub.php	7290
Pesawat.bmp	1440054	CA_CHING.WAV	18502	Pesawat7.bmp	1440054	CA_CHING.WAV	18502
Pesawat.bmp	1440054	coding.txt	21566	Pesawat8.bmp	1440054	coding.txt	21566
Pesawat.bmp	1440054	salary.xls	58880	Pesawat9.bmp	1440054	salary.xls	58880
Pesawat.bmp	1440054	web.htm	31051	Pesawat10.bmp	1440054	web.htm	31051

4.3.4 Analisis Penyisipan dan Ekstraksi File Pesan terhadap Tiga File Image yang Berbeda

Tabel 4.3 Tabel Uji File Output dari 3 File Pesan Berbeda, File Image “Monalisa.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Monalisa.bmp	419454	Serbu.txt	265	MonalisaStego1.bmp	419454	Serbu-Hasil.txt	265
Monalisa.bmp	419454	Tabel4.doc	44544	MonalisaStego2.bmp	419454	Tabel4.doc	44544
Monalisa.bmp	419454	Gadis.txt	10977	MonalisaStego3.bmp	419454	Gadis-Hasil.txt	10977

Tabel 4.4 Tabel Uji *File Output* dari 3 *File* Pesan Berbeda, *File Image* “Lena.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Lena.bmp	786486	Serbu.txt	265	LenaStego1.bmp	786486	Serbu-Hasil.txt	265
Lena.bmp	786486	Tabel4.doc	44544	LenaStego2.bmp	786486	Tabel4.doc	44544
Lena.bmp	786486	Gadis.txt	10977	LenaStego3.bmp	786486	Gadis-Hasil.txt	10977

Tabel 4.5 Tabel Uji *File Output* dari 3 *File* Pesan Berbeda, *File Image* “Fruits.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Fruits.bmp	499554	Serbu.txt	265	FruitsStego1.bmp	499554	Serbu-Hasil.txt	265
Fruits.bmp	499554	Tabel4.doc	44544	FruitsStego2.bmp	499554	Tabel4.doc	44544
Fruits.bmp	499554	Gadis.txt	10977	FruitsStego3.bmp	499554	Gadis-Hasil.txt	10977

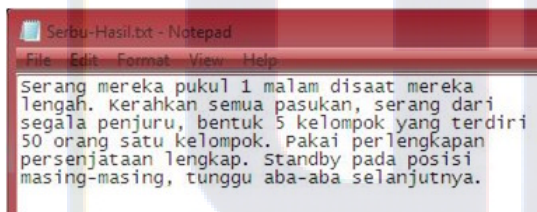


Gambar 4.12 Perbedaan Kualitas *File Image* Jika Digunakan 2 Bit Terakhir

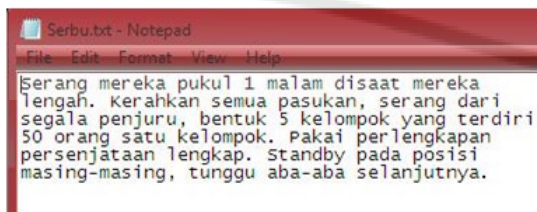
4.3.5 Analisis Penyisipan dan Ekstraksi *File image* dan *File* Pesan



Gambar 4.9 *File Image* Sebelum dan Sesudah Disisipkan *File* Pesan

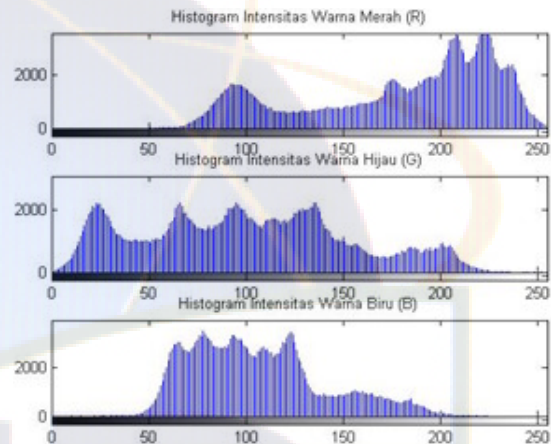


Gambar 4.10 *File* Pesan Rahasia Sebelum Disisipkan ke *File Image*

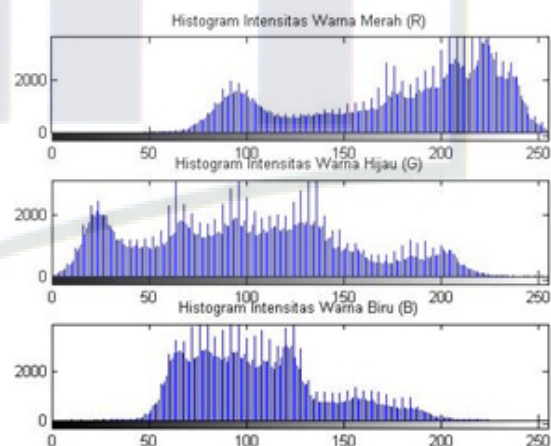


Gambar 4.11 *File* Pesan yang Diambil Kembali dari *File Image*

4.3.6 Analisis *File* Pesan Dan *File Image* Jika Digunakan 2 Bit Terakhir



Gambar 4.13 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 2 Bit Terakhir.

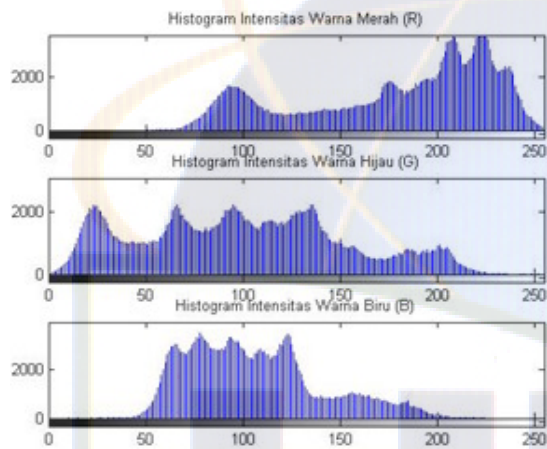


Gambar 4.14 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 2 Bit Terakhir.

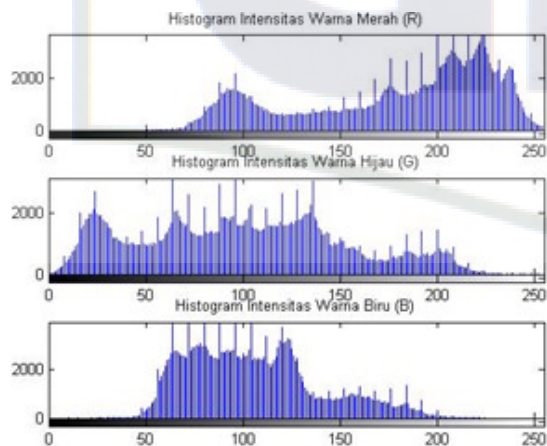
4.3.7 Analisis *File Pesan* Dan *File Image* Jika Digunakan 3 Bit Terakhir



Gambar 4.15 Perbedaan Kualitas *File Image* Jika Digunakan 3 Bit Terakhir



Gambar 4.16 Histogram Intensitas Warna Sebelum Penyisipan *File Pesan* Dengan 3 Bit Terakhir

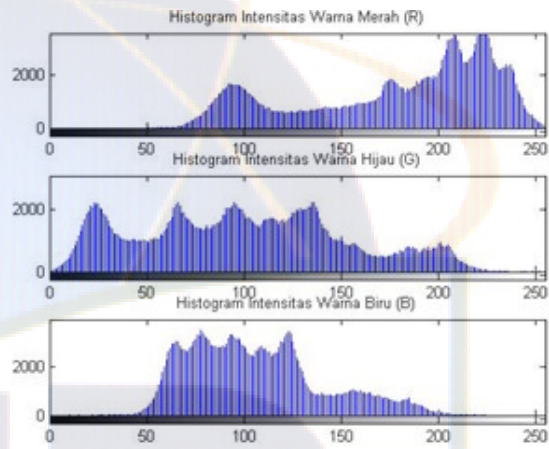


Gambar 4.17 Histogram Intensitas Warna Sesudah Penyisipan *File Pesan* Dengan 3 Bit Terakhir

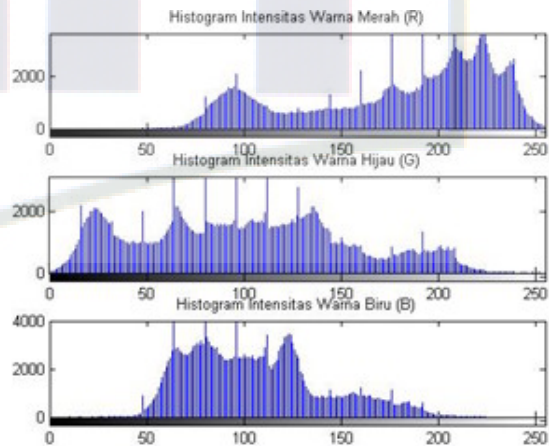
4.3.8 Analisis *File Pesan* Dan *File Image* Jika Digunakan 4 Bit Terakhir



Gambar 4.18 Perbedaan Kualitas *File Image* Jika Digunakan 4 Bit Terakhir



Gambar 4.19 Histogram Intensitas Warna Sebelum Penyisipan *File Pesan* Dengan 4 Bit Terakhir.

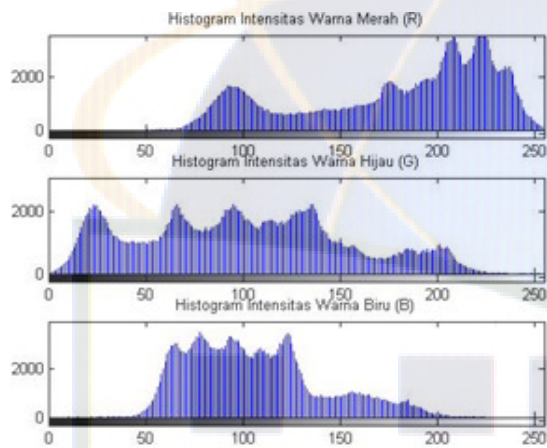


Gambar 4.20 Histogram Intensitas Warna Sesudah Penyisipan *File Pesan* Dengan 4 Bit Terakhir.

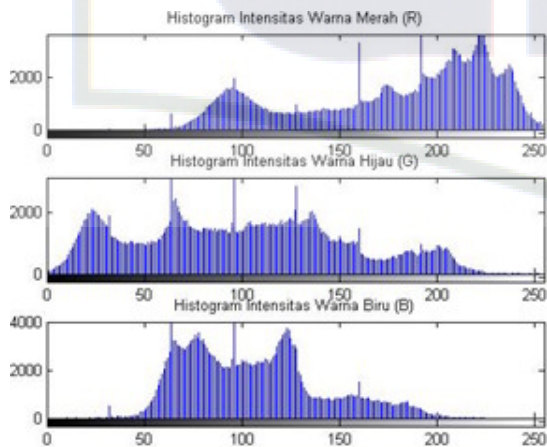
4.3.9 Analisis *File Pesan* Dan *File Image* Jika Digunakan 5 Bit Terakhir



Gambar 4.21 Perbedaan Kualitas *File Image* Jika Digunakan 5 Bit Terakhir



Gambar 4.22 Histogram Intensitas Warna Sebelum Penyisipan *File Pesan* Dengan 5 Bit Terakhir.

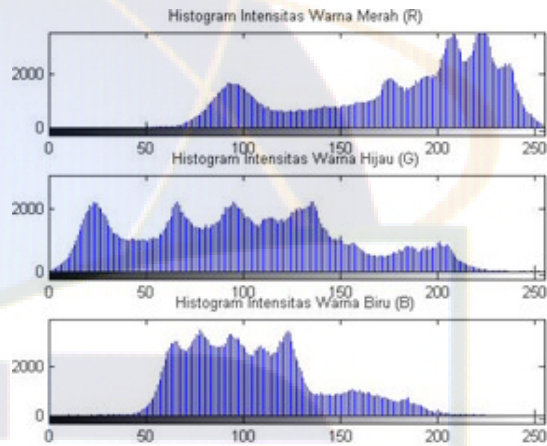


Gambar 4.23 Histogram Intensitas Warna Sebelum Penyisipan *File Pesan* Dengan 5 Bit Terakhir.

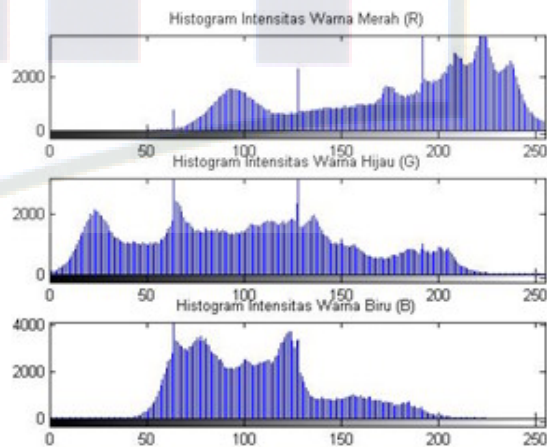
4.3.10 Analisis *File Pesan* Dan *File Image* Jika Digunakan 6 Bit Terakhir



Gambar 4.24 Perbedaan Kualitas *File Image* Jika Digunakan 6 Bit Terakhir



Gambar 4.25 Histogram Intensitas Warna Sebelum Penyisipan *File Pesan* Dengan 6 Bit Terakhir.

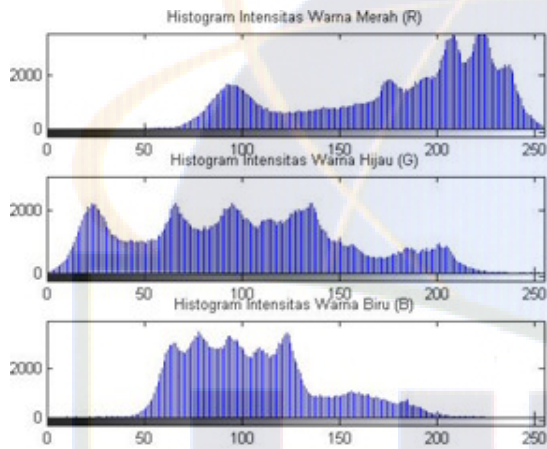


Gambar 4.26 Histogram Intensitas Warna Sesudah Penyisipan *File Pesan* Dengan 6 Bit Terakhir.

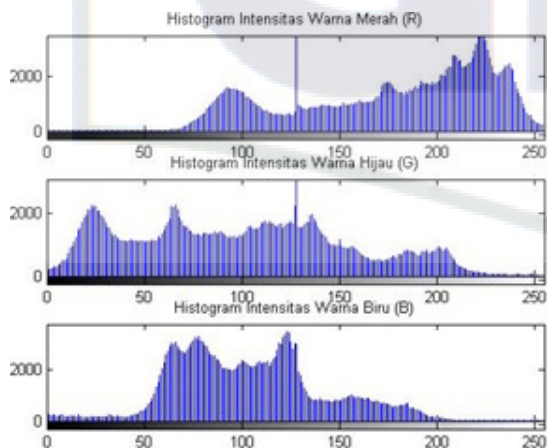
4.3.11 Analisis *File Pesan* Dan *File Image* Jika Digunakan 7 Bit Terakhir



Gambar 4.27 Perbedaan Kualitas *File Image* Jika Digunakan 7 Bit Terakhir

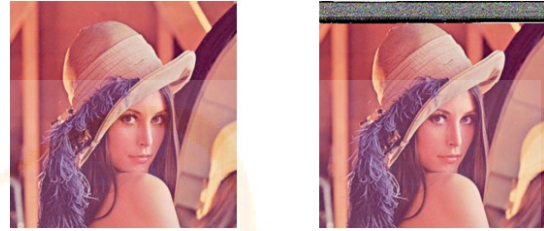


Gambar 4.28 Histogram Intensitas Warna Sebelum Penyisipan *File Pesan* Dengan 7 Bit Terakhir.

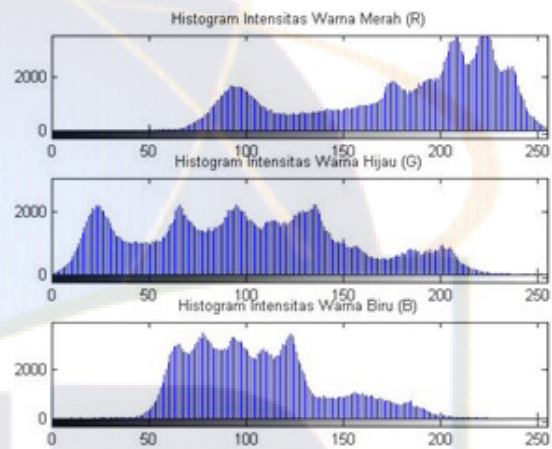


Gambar 4.29 Histogram Intensitas Warna Sesudah Penyisipan *File Pesan* Dengan 7 Bit Terakhir.

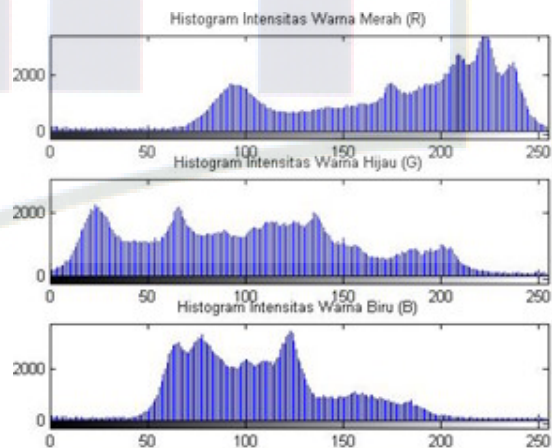
4.3.12 Analisis *File Pesan* Dan *File Image* Jika Digunakan 8 Bit Terakhir



Gambar 4.30 Perbedaan Kualitas *File Image* Jika Digunakan 8 Bit Terakhir



Gambar 4.31 Histogram Intensitas Warna Sebelum Penyisipan *File Pesan* Dengan 8 Bit Terakhir.



Gambar 4.32 Histogram Intensitas Warna Sesudah Penyisipan *File Pesan* Dengan 8 Bit Terakhir.

4.3.13 Analisis Ukuran File Pesan Terhadap File Image

Tabel 4.6 Tabel uji Ukuran File Pesan Rahasia Terhadap File Image

File Image	Size (byte)	Secret File	Size (byte)	Status Penyisipan
Lena.bmp	786486	Serbu.txt	265	Berhasil
Lena.bmp	786486	Tabel4.doc	44544	Berhasil
Lena.bmp	786486	Bab IV.doc	323584	Tidak

4.4 Perbandingan Steganografi Sejenis

Tabel 4.7 Perbandingan Steganografi Sejenis 1

Judul	Penulis	Tahun	Universitas	Perbedaan
ANALISIS DAN IMPLEMENTASI STEGANOGRAFI DAN KRIPTOGRAFI PADA FILE AUDIO	YUNIAR NURSYAMSIH SETTORINI 103091029623	2008	Universitas Islam Negeri Syarif Hidayatullah Jakarta	<ol style="list-style-type: none"> 1. Skripsi di atas menggunakan file audio sebagai media penampung file-nya, sedangkan penulis menggunakan Citra Digital untuk penampungan file-nya. 2. Skripsi di atas memakai metode pengembangan sistem SDLC (System Development Life Cycle) sedangkan penulis memakai metode pengembangan sistem RAD (Rapid Application Development). 3. Skripsi diatas hanya fokus terhadap 1 bit terakhir pada file audionya, sedangkan penulis mengembangkan hingga 8 bit dapat digunakan 4. Skripsi diatas memakai C# dalam pembuatan aplikasinya, sedangkan penulis menggunakan Delphi. 5. Skripsi diatas hanya dapat beberapa format file yang dapat disisipkan, sedangkan penulis hampir semua format file dapat disisipkan.

Tabel 4.8 Perbandingan Steganografi Sejenis 2

Judul	Penulis	Tahun	Universitas	Perbedaan
PEMBUATAN APLIKASI STEGANOGRAPHY PADA FILE AUDIO	Hardi Wijaya 26402169	2007	Universitas Kristen Petra Surabaya	<ol style="list-style-type: none"> 1. Skripsi diatas memakai metode pengembangan sistem SDLC (System Development Life Cycle) sedangkan penulis memakai metode pengembangan sistem RAD (Rapid Application Development). 2. Skripsi diatas memakai Visual Basic dalam pembuatan aplikasinya, sedangkan penulis menggunakan Delphi. 3. Skripsi diatas hanya dapat beberapa format file yang dapat disisipkan, sedangkan penulis hampir semua format file dapat disisipkan. 4. Skripsi diatas menggunakan file audio sebagai media penampung file-nya, sedangkan penulis menggunakan Citra Digital untuk penampungan filenya 5. Skripsi diatas hanya fokus terhadap 1 bit terakhir pada file

5. PENUTUP

5.1 Kesimpulan

Dari hasil pengujian sistem yang dilakukan pada bab sebelumnya, maka dapat disimpulkan ke dalam beberapa hal antara lain:

1. Aplikasi StegoImage berhasil mengimplementasikan teknik steganografi yang menggunakan algoritma LSB dalam mengamankan pesan. Hal ini dibuktikan melalui hasil uji analisa pada Tabel 4.1 bahwa file dapat disisipkan ke dalam file image dan diambil kembali dari file image tersebut.
2. Proses penyisipan file tidak terpaku pada satu format file tertentu saja, tapi dapat dilakukan pada file berformat *.txt, *.doc, *.xls, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav hal ini dibuktikan oleh uji analisis pada Tabel 4.2.
3. Berdasarkan hasil uji analisis, penyisipan file ke dalam file image mempengaruhi kualitas warna pada file image tersebut. Dengan adanya perubahan intensitas warna antara file image asli dan file image yang sudah disisipkan pesan, hal itu dapat dilihat pada gambar-gambar histogram pada bab IV.
4. Berdasarkan hasil uji analisis pada Tabel 4.1 dapat dinyatakan bahwa file pesan Masukan dan hasil keluaran memiliki jumlah byte yang sama persis, di mana artinya penyisipan file pesan tidak mempengaruhi besar ukuran file pesan awal maupun akhir.
5. Berdasarkan hasil uji analisis keseluruhan pada bab IV dapat disimpulkan bahwa algoritma LSB tidak hanya terpaku pada 1 bit terakhir saja sebagai tempat penyembunyian data, tapi dapat dikembangkan hingga 8 bit.

5.2 Saran

1. Aplikasi StegoImageKu ini dapat diimplementasikan di instansi yang membutuhkan pengamanan file seperti bank, kantor pemerintahan, militer dan sebagainya.
2. Aplikasi StegoImageKu hanya file

image sebagai media penampung, diharapkan dapat dikembangkan sehingga dapat menggunakan *file* teks, audio, video dan lain-lain sebagai media penampungnya.

3. Aplikasi StegoImage masih dikembangkan untuk perangkat *computer desktop*. Akan lebih praktis apabila dapat dikembangkan lebih lanjut untuk dapat digunakan dalam lingkungan perangkat keras *mobile* seperti telepon genggam.

REFERENSI

- Ahmad, Usman. (2005). *Pengolahan Citra Digital & Teknik Pemogramannya*. Yogyakarta : Graha Ilmu.
- Ariyus, Doni. (2006). *Kriptografi Keamanan Data dan Komunikasi*. Yogyakarta : Graha Ilmu.
- Andleigh, Prabhat K., Thakrar, Kiran. (1996). *Multimedia System Design*. New Jersey : Prentice Hall International Edition.
- doank29.multiply.com/journal/item/3
- haryanto.staff.gunadarma.ac.id/Downloads/files/7272/9.Steganografi.ppt
- ilmucerdas.wordpress.com/2008/08/02/pengertian-multimedia/
- ilmukomputer.org/2006/08/25/aplikasi-steganografi-dengan-borland-delphi/
- Kenneth E. Kendall & Julie E Kendall. (2005). *System Analysis and Design*. Sixth Editon. New Jersey : Pearson Education.
- Ladjamuddin, Al Bahra Bin. (2005). *Analisis dan Desain Sistem Informasi*. Jakarta : Graha Ilmu.
- Malik, Jaja Jamaludin. (2006). *Kumpulan Latihan Pemograman Delphi*. Yogyakarta : Andi.
- Marvin Ch, Wijaya, & Agus Prijono. (2007). *Pengolahan Citra Digital Menggunakan MatLAB Image Processing Toolbox*. Bandung: Informatika.
- Munir, Rinaldi. (2006). *Kriptografi*. Bandung: Informatika.
- Munir, Rinaldi. (2004). *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Bandung : Informatika.
- NN. (2002). *11 Kamus Besar Bahasa Indonesia*. Jakarta : Balai Pustaka.
- Prasetyo, Didik Dwi. (2004). *Aplikasi Database Client/Server Menggunakan Delphi dan MySQL*. Jakarta: Elex Media Komputindo.
- Pressman, Roger S. (2002). *Rekayasa Perangkat Lunak*, Edisi 1. Yogyakarta: Andi.
- webmail.informatika.org/~rinaldi/Kriptografi/2007-2008/Makalah1/MakalahIF5054-2007-A-077.pdf
- Whitten, Jeffrey, dan Lonnie Bentley. (2007). *Sistem Analisis dan Metode Desain*. Edisi Kelima. Yogyakarta : Penerbit Andi.
- www.ascii-code.com/
- www.cert.or.id/~budi/courses/ec7010/dikmenjur/taufik-report.pdf
- www.e-smartschool.com/pnk/002/PNK0020019.asp
- webmail.informatika.org/~rinaldi/Matdis/2008-2009/Makalah2008/Makalah0809-056.pdf
- webmail.informatika.org/~rinaldi/Kriptografi/Steganografi%20dan%20Watermarking.pdf
- www.itelkom.ac.id/library/index.php?view=article&catid=15%3Apemrosesan-sinyal&id=344%3Acitra-digital&option=com_content&Itemid=15
- www.jcatki.no-ip.org:8080/SDL_image/demos/lena.jpg
- www.bobbemer.com
- www.itelkom.ac.id/library/index.php?view=article&catid=15%3Apemrosesan
- www.tutorialgratis.net%20Tips%20Memilih%20Format%20GambarNet.htm
- Hariyanto, Bambang. (2004). *Rekayasa Sistem Berorientasi Objek*. Bandung : Informatika.



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.

**ANALISIS DAN PERANCANGAN APLIKASI
STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN
METODE LSB (*LEAST SIGNIFICANT BIT*)**



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
SYARIF HIDAYATULLAH
JAKARTA
2010 M/ 1431 H**

**ANALISIS DAN PERANCANGAN APLIKASI
STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN
METODE LSB (*LEAST SIGNIFICANT BIT*)**



**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
SYARIF HIDAYATULLAH
JAKARTA
2010 M/ 1431 H**

**ANALISIS DAN PERANCANGAN APLIKASI
STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN
METODE LSB (*LEAST SIGNIFICANT BIT*)**

ADIRIA

204091002554

Skripsi

Diajukan Untuk Memenuhi Syarat Kelulusan Sarjana (S1)

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI
SYARIF HIDAYATULLAH
JAKARTA
2010 M/ 1431 H**

**ANALISIS DAN PERANCANGAN APLIKASI STEGANOGRAFI PADA
CITRA DIGITAL MENGGUNAKAN METODE LSB (*LEAST
SIGNIFICANT BIT*)**

Skripsi

Sebagai salah satu syarat untuk memperoleh gelar sarjana komputer

Pada Fakultas Sains dan Teknologi

Universitas Islam Negeri Syarif Hidayatullah Jakarta

Oleh :

Adiria

204091002554

Pembimbing I

Menyetujui,

Pembimbing II

Arini, MT

NIP.197601312009012001

Qurrotul Aini, MT

NIP.197303252009012001

Mengetahui :

Ketua Program Studi Teknik Informatika

Yusuf Durrachman, M.Sc

NIP.197105222006041002

PENGESAHAN UJIAN

Skripsi ini berjudul “**ANALISIS DAN PERANCANGAN APLIKASI STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN METODE LSB (*LEAST SIGNIFICANT BIT*)**”, telah diuji dan dinyatakan LULUS dalam sidang Munaqosyah Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta. Pada hari Senin skripsi ini telah diterima sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata Satu (S1) Program Studi Teknik Informatika.

Jakarta, 5 April 2010

Penguji I

Penguji II

Victor Amrizal

Imam M.Shofi

Dekan
Fakultas Sains dan Teknologi

Ketua Program
Studi Teknik Informatika

Dr. Svopiansyah Jaya Putra, M.Sis
NIP. 150317956

Yusuf Durrachman, M.Sc
NIP. 197105222006041002

PERNYATAAN

DENGAN INI SAYA MENYATAKAN BAHWA SKRIPSI INI BENAR –
BENAR HASIL KARYA SENDIRI YANG BELUM PERNAH DIAJUKAN
SEBAGAI SKRIPSI ATAU KARYA ILMIAH PADA PERGURUAN TINGGI
ATAU LEMBAGA MANAPUN.

Jakarta, 5 April 2010

Adiria

204091002554

ABSTRAK

Adiria, Analisis Dan Perancangan Aplikasi Steganografi pada Citra Digital Menggunakan Metode LSB (*Least Significant Bit*). Dibimbing oleh Ibu **Arini, MT** dan **Qurrotul Aini, MT**.

Di seluruh dunia, internet (*interconnection network*) sudah berkembang menjadi salah satu media komunikasi data yang sangat populer. Dengan berbagai teknik pengambilan informasi secara ilegal yang berkembang, banyak orang yang mencoba untuk mengakses informasi yang bukan haknya. Berbagai macam teknik untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak telah banyak dilakukan dalam upaya mengamankan suatu data penting. Steganografi adalah salah satu teknik menyembunyikan *file* pesan agar bagi orang awam tidak menyadari keberadaan dari *file* pesan yang disembunyikan. Teknik ini sering digunakan untuk menghindari kecurigaan orang dan menghindari keinginan orang untuk mengetahui isi *file* pesan tersebut. Citra Digital adalah salah satu media yang paling umum dikenal oleh masyarakat. Penelitian ini bertujuan menganalisis dan merancang suatu aplikasi steganografi sebagai salah satu teknik pengamanan file pesan. Pada teknik steganografi ini digunakan metode LSB (*Least Significant Bit*) yaitu menyembunyikan *byte-byte* data atau informasi pada bit terakhir pada citra digital, sehingga tidak terjadi perubahan ukuran dan bentuk terhadap citra digital secara kasat mata, data atau informasi pun dapat dikembalikan seperti semula tanpa ada perubahan ukuran dan bentuknya.

Kata Kunci : Aplikasi, Steganografi, Citra Digital, Steganografi, LSB

KATA PENGANTAR

بسم الله الرحمن الرحيم

Segala puji syukur penulis panjatkan kehadiran Allah SWT, sang pemilik sifat rahman dan rahiim yang telah melimpahkan segala rahmat dan inayahNya, sehingga penulis dapat menyelesaikan skripsi ini. Shalawat dan salam selalu tercurahkan kepada junjungan baginda besar Nabi Muhammad SAW.

Sebagai bentuk penghargaan yang tidak terlukiskan, penulis menuangkan dalam bentuk ucapan terima kasih sebesar-besarnya kepada:

1. DR. Syopiansyah Jaya Putra, M.Sis, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta.
2. Bapak Yusuf Durrachman, M.Sc, selaku Ketua Program Studi Teknik Informatika dan Ibu Viva Arifin, MMSI, selaku Sekretaris Program Studi Teknik Informatika.
3. Ibu Arini, MT dan Ibu Qurrotul Aini, MT selaku dosen pembimbing yang telah memberikan waktu dan perhatiannya dalam penyusunan skripsi ini.
4. Ayahanda Masri dan Ibunda Marlina yang senantiasa memberikan dukungan baik moral, materil, dan spiritual selama menyelesaikan skripsi ini *“mom dad this is for you”*.
5. Adik-adikku tersayang yaitu Wahyudi *“rajin belajar & cepet dewasa pikiranya ya..”* dan Ariyo Saputra *“hai jagoan kecilku cepet gede ya..”*.
Saudara-saudara sepupuku : Ristamansyah *“makasih da”*, Usni *“makasih*

bang”, Syadriansyah “thanx sob”, Putra, Eki, Elo, Teta, K wis, K widia, k Eni, Reza & Ncim “*makasih udah boleh numpang ngeprint selama skripsi, kapan2 numpang lagi ya hehe..*” serta saudara– saudara penulis lainnya yang telah mendukung dan memberikan doanya. Semoga kita selalu dalam hangatnya ikatan keluarga yang kokoh, kuat dan abadi serta saling membantu satu sama lain.

6. Ari Kristianto, Mirwan Nurjaya, Muhammad Agus Syarifuddin, Muhammad Qadhavi, Muhammad Syah Reza, Rahmat Mulya, Setiajid, Wangsa Dipraja, Yayan Pebriana, Yazidanyastuti, Fila Anggraini, Personal Motivator si nenk “*makasih ya buat supportnya*”, Personal Tour Guide si aa “*makasih buat jalan2 dan refresingnya*”, Anton Budiwan “*makasih minjemin kamarnya buat ngetik*”, dan teman-teman di kosan “*god bless you all my friends*”.
7. Seluruh rekan-rekan yang tidak dapat penulis sebutkan satu persatu yang secara tidak langsung membantu dan memberikan semangat sehingga penulisan skripsi ini dapat berjalan dengan lancar.

Akhir kata hanya kepada Allah jualah penulis memanjatkan doa, semoga Allah memberikan balasan berupa amal yang berlipat kepada mereka dan semoga skripsi ini dapat bermanfaat dan memberikan kontribusi bagi semua pihak. Amiin.

Jakarta, 5 April 2010

Adiria
204091002554

DAFTAR ISI

Halaman Judul	i
Halaman Keterangan Judul	ii
Lembar Pengesahan Pembimbing	iii
Lembar Pengesahan Ujian	iv
Lembar Pernyataan	v
Abstrak	vi
Kata Pengantar	vii
Daftar Isi	ix
Daftar Gambar	xiii
Daftar Tabel	xvi
Daftar Istilah	xvii
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Identifikasi Masalah	2
1.3 Rumusan Masalah	3
1.4 Batasan Masalah	3
1.5 Tujuan Penelitian	4
1.6 Manfaat Penelitian	4
1.7 Metode Penelitian	5
1.7.1 Metode Pengumpulan Data Dan Informasi	5

1.7.2 Metode Pengembangan Sistem	6
1.8 Sistematika Penulisan	7
BAB II LANDASAN TEORI	8
2.1 Pengertian Analisis	8
2.2 Pengertian Perancangan	8
2.3 Steganografi	9
2.3.1 Sejarah Steganografi	9
2.3.2 Pengertian Steganografi	10
2.3.3 Metode Steganografi	14
2.4 LSB (<i>Least Significant Bit</i>)	17
2.5 ASCII	20
2.6 Multimedia	23
2.7 Citra Digital	24
2.7.1 Pengertian Citra Digital	24
2.7.2 Pengolahan Citra (<i>Image Processing</i>)	25
2.7.3 Perbandingan <i>File</i> Gambar BMP (Bitmap) dengan JPG, GIF, atau PNG	28
2.8 <i>Delphi</i>	30
2.8.1 Pengertian <i>Delphi</i>	30
2.8.2 Kelebihan <i>Delphi</i>	31
2.9 Perancangan Program	32
2.9.1 Model–Model Pengembangan Sistem	32
2.9.2 Diagram Alur (<i>Flowchart</i>)	34
2.9.3 <i>State Transition Diagram</i> (STD)	36

2.10 Konsep RAD	38
-----------------------	----

2.10.1 Definisi RAD	38
---------------------------	----

2.10.2 Tahapan-Tahapan RAD	38
----------------------------------	----

2.10.3 Keunggulan RAD	40
-----------------------------	----

BAB III METODE PENELITIAN	42
--	-----------

3.1 Metode Pengumpulan Data	42
-----------------------------------	----

3.2 Metode Pengembangan Sistem	43
--------------------------------------	----

3.2.1 Fase Perencanaan Syarat-Syarat	44
--	----

3.2.2 Proses Desain RAD (<i>RAD Design Workshop</i>)	44
--	----

3.2.3 Fase Implementasi (<i>Implementation Phase</i>)	45
---	----

BAB IV ANALISIS DAN PERANCANGAN	47
--	-----------

4.1 Fase Perencanaan Syarat-Syarat	47
--	----

4.1.1 Analisis Kebutuhan	47
--------------------------------	----

4.1.2 Menentukan Tujuan	48
-------------------------------	----

4.1.3 Menentukan Syarat-Syarat	48
--------------------------------------	----

4.2 Fase Desain RAD	49
---------------------------	----

4.2.1 Perancangan Proses	49
--------------------------------	----

4.2.2 <i>Flowchart</i> Aplikasi Steganografi	50
--	----

4.2.3 Perancangan Antarmuka	53
-----------------------------------	----

4.2.4 STD (<i>State Transition Diagram</i>)	58
---	----

4.2.5 Konstruksi	59
------------------------	----

4.3 Fase Implementasi	60
-----------------------------	----

4.3.1 Instalasi Aplikasi	60
--------------------------------	----

4.3.2	Analisis Penyembunyian <i>File</i>	62
4.3.3	Analisis Format <i>File</i> Apa Saja yang Dapat Disisipkan ke <i>File Image</i>	62
4.3.4	Analisis Penyisipan dan Ekstraksi <i>File</i> Pesan terhadap Tiga <i>File Image</i> yang Berbeda	64
4.3.5	Analisis Penyisipan dan Ekstraksi <i>File Image</i> dan <i>File</i> Pesan	66
4.3.6	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 2 Bit Terakhir	67
4.3.7	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 3 Bit Terakhir	70
4.3.8	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 4 Bit Terakhir	72
4.3.9	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 5 Bit Terakhir	74
4.3.10	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 6 Bit Terakhir	76
4.3.11	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 7 Bit Terakhir	78
4.3.12	Analisis <i>File</i> Pesan dan <i>File Image</i> Jika Digunakan 8 Bit Terakhir	80
4.3.13	Analisis Ukuran <i>File</i> Pesan Terhadap <i>File Image</i>	82
4.4	Perbandingan Steganografi Sejenis	83
BAB V	PENUTUP	86
5.1	Kesimpulan	86
5.2	Saran	87
DAFTAR PUSTAKA	88
LAMPIRAN	91

DAFTAR GAMBAR

Gambar 2.1	Perbedaan Pesan yang Disembunyikan	11
Gambar 2.2	Diagram Penyisipan dan Ekstraksi Pesan.....	13
Gambar 2.3	Diagram Sistem Steganografi.....	14
Gambar 2.4	<i>Spread Spectrum Method</i>	16
Gambar 2.5	<i>Most Significant Bit</i> (MSB) dan <i>Least Significant Bit</i> (LSB)	17
Gambar 2.6	Proses <i>File Image</i> Menjadi Kumpulan <i>Pixel-Pixel</i>).....	17
Gambar 2.7	Nilai-Nilai <i>Pixel</i> Dalam Suatu <i>File Image</i>	18
Gambar 2.8	Koordinat Spasial dan Nilai $f(x,y)$	25
Gambar 2.9	Bagan Pengolahan Citra	25
Gambar 2.10	Tampilan <i>Form</i> Awal <i>Delphi</i> 2007 Win32.....	31
Gambar 2.11	Simbol State.....	37
Gambar 2.12	Simbol <i>Transition State</i>	37
Gambar 2.13	Simbol Kondisi dan Aksi.....	37
Gambar 2.14	Tahapan-Tahapan RAD.....	40
Gambar 3.1	Skema Sistem Model RAD	43
Gambar 3.2	Ilustrasi Metode Penelitian Pengembangan Aplikasi Steganografi pada Citra <i>Digital</i>	46
Gambar 4.1	Proses Penyisipan dan Ekstraksi Pesan	50
Gambar 4.2	<i>Flowchart</i> Proses Penyisipan Pesan Rahasia.....	51
Gambar 4.3	<i>Flowchart</i> Proses Ekstraksi Pesan Rahasia	52

Gambar 4.4	<i>Layout Form Induk</i>	54
Gambar 4.5	Rancangan <i>Form Utama Tab Encoding</i>	56
Gambar 4.6	Rancangan <i>Form Utama Tab Decoding</i>	57
Gambar 4.7	Perancangan <i>Form About</i>	58
Gambar 4.8	<i>State Transition Diagram</i> Aplikasi Steganografi pada <i>Citra Digital</i>	59
Gambar 4.9	<i>File Image</i> Sebelum dan Sesudah Disisipkan <i>File Pesan</i>	66
Gambar 4.10	<i>File Pesan</i> Rahasia Sebelum Disisipkan ke <i>File Image</i>	67
Gambar 4.11	<i>File Pesan</i> yang Diambil Kembali dari <i>File Image</i>	67
Gambar 4.12	Perbedaan Kualitas <i>File Image</i> Jika Digunakan 2 Bit Terakhir	68
Gambar 4.13	Histogram Intensitas Warna Sebelum Penyisipan <i>File Pesan</i> dengan 2 Bit Terakhir	69
Gambar 4.14	Histogram Intensitas Warna Sesudah Penyisipan <i>File Pesan</i> dengan 2 Bit Terakhir	69
Gambar 4.15	Perbedaan Kualitas <i>File Image</i> Jika Digunakan 3 Bit Terakhir	70
Gambar 4.16	Histogram Intensitas Warna Sebelum Penyisipan <i>File Pesan</i> dengan 3 Bit Terakhir	71
Gambar 4.17	Histogram Intensitas Warna Sesudah Penyisipan <i>File Pesan</i> dengan 3 Bit Terakhir	71
Gambar 4.18	Perbedaan Kualitas <i>File Image</i> Jika Digunakan 4 Bit Terakhir	72
Gambar 4.19	Histogram Intensitas Warna Sebelum Penyisipan <i>File Pesan</i> dengan 4 Bit Terakhir	73
Gambar 4.20	Histogram Intensitas Warna Sesudah Penyisipan <i>File Pesan</i> dengan 4 Bit Terakhir	73
Gambar 4.21	Perbedaan Kualitas <i>File Image</i> Jika Digunakan 5 Bit Terakhir	74
Gambar 4.22	Histogram Intensitas Warna Sebelum Penyisipan <i>File Pesan</i> dengan 5 Bit Terakhir	75

Gambar 4.23 Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 5 Bit Terakhir	75
Gambar 4.24 Perbedaan Kualitas <i>File Image</i> Jika Digunakan 6 Bit Terakhir	76
Gambar 4.25 Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 6 Bit Terakhir	77
Gambar 4.26 Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 6 Bit Terakhir	77
Gambar 4.27 Perbedaan Kualitas <i>File Image</i> Jika Digunakan 7 Bit Terakhir	78
Gambar 4.28 Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 7 Bit Terakhir	79
Gambar 4.29 Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 7 Bit Terakhir	79
Gambar 4.30 Perbedaan Kualitas <i>File Image</i> Jika Digunakan 8 Bit Terakhir	80
Gambar 4.31 Histogram Intensitas Warna Sebelum Penyisipan <i>File</i> Pesan dengan 8 Bit Terakhir	81
Gambar 4.32 Histogram Intensitas Warna Sesudah Penyisipan <i>File</i> Pesan dengan 8 Bit Terakhir	81

DAFTAR TABEL

Tabel 2.1 Keuntungan dan Kelemahan Metode LSB	19
Tabel 2.2 ASCII	21
Tabel 2.3 Beberapa Metode dan Perbedaannya	32
Tabel 2.4 Simbol-simbol <i>Flowchart</i>	34
Tabel 4.1 Tabel Uji Penyembunyian <i>File</i>	62
Tabel 4.2 Tabel Uji Format <i>File</i> Apa Saja yang Dapat Disisipkan ke <i>File Image</i>	63
Tabel 4.3 Tabel Uji <i>File</i> Output dari 3 <i>File</i> Pesan Berbeda, <i>File Image</i> “Monalisa.bmp”	65
Tabel 4.4 Tabel Uji <i>File</i> Output dari 3 <i>File</i> Pesan Berbeda, <i>File Image</i> “Lena.bmp”	65
Tabel 4.5 Tabel Uji <i>File</i> Output dari 3 <i>File</i> Pesan Berbeda, <i>File Image</i> “Fruits.bmp”	65
Tabel 4.6 Uji Ukuran <i>File</i> Pesan Rahasia Terhadap <i>File Image</i>	82
Tabel 4.7 Perbandingan Steganografi Sejenis 1	83
Tabel 4.8 Perbandingan Steganografi Sejenis 2	84

DAFTAR ISTILAH

ASCII (American Standart Code For Information Interchange):

Standar huruf dan tanda baca untuk komputer. ASCII merupakan kode berupa karakter 8 bit berbentuk angka 1 dan 0 untuk mewakili karakter-karakter alpha numerik.

Bit:

Berasal dari kata *binary* digit. Merupakan satuan terkecil data komputer yang hanya merupakan angka 0 dan 1. Semua data bisa ditulis dalam bit.

Bit Depth:

Jumlah bit yang digunakan untuk mempresentasikan tiap titik dalam representasi citra grafis. Makin besar jumlah bit yang digunakan untuk mempresentasikan suatu titik, semakin banyak warna dan atau bayangan abu-abu yang dapat dibuat.

Bitmap:

Sebuah *image* grafis yang disusun dari *pixel-pixel* dan dikonversikan ke dalam bit. Biasa digunakan dalam Microsoft Windows.

Byte:

Kumpulan delapan bit, yang membentuk satu data bermakna, misalnya huruf A, B, atau angka 3,4 dan lain-lain.

Color Depth:

Color depth merujuk pada jumlah warna yang ditampilkan di monitor oleh *video card*. Semakin banyak warna yang digunakan, semakin realistis tampilan yang dapat dilihat. Seperti gambar yang didapat dari foto, merubah *color depth* komputer dapat meningkatkan kualitas gambar, namun dapat juga tidak apabila foto itu sendiri terbatas pada jumlah warna tertentu.

Encrypt:

Penerjemahan data menjadi kode rahasia. Enkripsi adalah cara yang paling efektif untuk memperoleh pengamanan data. Untuk membaca *file* yang di-*enkrip*, kita harus mempunyai akses terhadap kata sandi yang memungkinkan kita men-*dekrip* pesan tersebut.

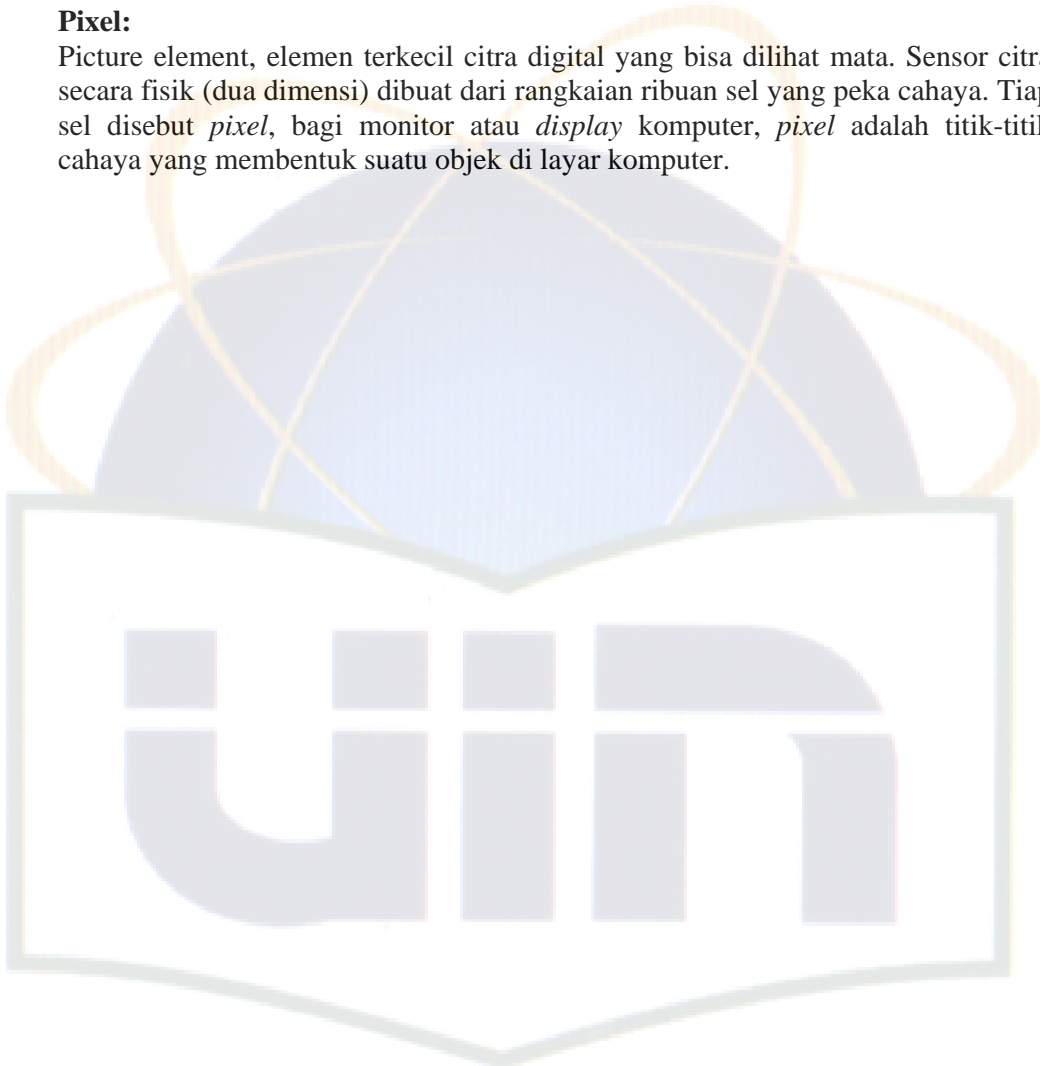
Decrypt:

Mengubah kembali hasil enkripsi ke bentuk aslinya sehingga informasi tersebut dapat dibaca.

Error: Istilah untuk menunjukan bahwa terdapat suatu penyimpangan dalam *software* atau kerusakan *hardware*.

Pixel:

Picture element, elemen terkecil citra digital yang bisa dilihat mata. Sensor citra secara fisik (dua dimensi) dibuat dari rangkaian ribuan sel yang peka cahaya. Tiap sel disebut *pixel*, bagi monitor atau *display* komputer, *pixel* adalah titik-titik cahaya yang membentuk suatu objek di layar komputer.



BAB I

PENDAHULUAN

1.1 Latar Belakang

Di seluruh dunia, internet (*interconnection network*) sudah berkembang menjadi salah satu media komunikasi data yang sangat populer. Kemudahan dalam penggunaan dan fasilitas yang lengkap merupakan keunggulan yang dimiliki oleh internet dan bukan rahasia umum di kalangan masyarakat pengguna internet pada saat sekarang ini. Namun, seiring dengan berkembangnya media internet dan aplikasi yang menggunakan internet semakin bertambah pula kejahatan dalam sistem informasi. Dengan berbagai teknik pengambilan informasi secara ilegal yang berkembang, banyak yang mencoba untuk mengakses informasi yang bukan haknya. Untuk itu, sejalan dengan berkembangnya media internet yang sangat cepat, harus juga diikuti dengan perkembangan pengamanan dalam sistem informasi yang berada dalam media internet tersebut.

Berbagai macam teknik yang digunakan untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak telah banyak dilakukan dalam upaya mengamankan suatu data penting dengan menggunakan sistem kriptografi yang melakukan enkripsi sebelum data penting tersebut ditransmisikan. Tindakan pengamanan menggunakan cara tersebut ternyata dianggap belum cukup dalam mengamankan suatu data karena adanya peningkatan kemampuan komputasi. Berbeda dengan teknik kriptografi, steganografi menyembunyikan pesan rahasia

agar bagi orang awam tidak menyadari keberadaan dari pesan yang disembunyikan. Teknik ini sering digunakan untuk menghindari kecurigaan orang dan menghindari keinginan orang untuk mengetahui isi pesan rahasia tersebut. Caranya dengan menyembunyikan informasi rahasia di dalam suatu wadah penampung informasi dengan sedemikian rupa sehingga keberadaan informasi rahasia yang ditempelkan tidak terlihat. Wadah penampung informasi tersebut dapat berbentuk berbagai jenis *file* multimedia *digital* seperti teks, citra, audio, video. Dalam tugas akhir ini, peneliti membuat analisis dan merancang aplikasi steganografi merupakan solusi dari permasalahan tersebut. Dengan penggunaan teknik ini, data informasi dapat kita sembunyikan di dalam media digital yang kita punya.

1.2 Identifikasi Masalah

Berbagai macam teknik yang digunakan untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak telah banyak dilakukan dalam upaya mengamankan suatu data penting dengan menggunakan sistem kriptografi yang melakukan enkripsi sebelum data penting tersebut ditransmisikan. Tetapi cara ini juga menimbulkan rasa keingintahuan seseorang untuk memecahkan rahasia tersebut, sehingga terciptalah *software-software* yang mampu memecahkan enkripsi dalam pesan, sehingga keamanan informasi tidak selalu terjamin.

Dengan steganografi penulis memanfaatkan kelemahan indera manusia, sehingga informasi yang ingin kita sampaikan tidak jatuh pada orang-orang yang

tidak berhak mengaksesnya, tanpa menimbulkan rasa keingintahuan seseorang terhadap informasi tersebut.

1.3 Rumusan Masalah

Dari uraian latar belakang di atas, maka dapat dirumuskan masalah pada tugas akhir berikut:

1. Bagaimana cara menyembunyikan *file* ke dalam media citra *digital*, sehingga *file* benar-benar terjaga keamanannya dengan menggunakan teknik steganografi sebagai suatu solusi pengamanan pesan?
2. Bagaimana metode LSB dapat digunakan sebagai salah satu metode steganografi dalam penyembunyian *file* ke dalam citra digital?

1.4 Batasan Masalah

Batasan masalah dalam tugas akhir ini mencakup:

1. Format *file* citra *digital* yang dapat digunakan untuk menyimpan pesan adalah 24bit (*true color*) berformat *.bmp.
2. Format *file* citra digital yang dihasilkan dari program steganografi ini adalah *.bmp.
3. Teknik steganografi yang digunakan hanya dapat menyimpan pesan berupa format *.txt, *.doc, *.xsl, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav.
4. Metode steganografi yang digunakan adalah LSB (*Least Significant Bit*).
5. *Tools* yang digunakan adalah Delphi 2007 Win 32.

1.5 Tujuan Penelitian

Tujuan yang hendak dicapai dalam tugas akhir ini adalah:

1. Memberikan informasi bagaimana teknik steganografi dapat diterapkan di dalam *file* citra digital.
2. Implementasi penyembunyian *file* ke dalam citra digital dengan memberikan satu alternatif metode steganografi.
3. Memanipulasi citra digital yang di dalamnya terdapat *file* sehingga *file* tersebut tidak dapat diketahui keberadaannya dan secara kasat mata tidak terjadi perubahan pada citra *digital* hasil manipulasi.

1.6 Manfaat Penulisan

Manfaat yang akan didapat dari penulisan skripsi dalam pembuatan aplikasi steganografi ini adalah:

a. Bagi Penulis

1. Memenuhi tugas akhir sebagai syarat untuk menyelesaikan studi Strata 1 (S-1) Teknik Informatika.
2. Mengetahui seberapa besar kemampuan mengimplementasikan pengetahuan mengenai steganografi pada sebuah aplikasi.
3. Hasil penelitian diharapkan dapat digunakan untuk mengembangkan ilmu komputer khususnya dalam bidang steganografi.
4. Sebagai bahan pertimbangan bagi seseorang dalam mengamankan *filenya*.

b. Bagi Universitas

1. Mengetahui kemampuan mahasiswa dalam menguasai materi teori yang telah diperoleh selama masa kuliah.
2. Mengetahui kemampuan mahasiswa dalam menerapkan ilmunya dan sebagai bahan evaluasi.

c. Bagi Masyarakat

1. Sebagai salah satu solusi dalam hal mengamankan *file* mereka dari orang-orang yang tidak berhak melihatnya.
2. Sebagai salah satu bahan pertimbangan bagi mereka yang ingin mengamankan *file* rahasia mereka agar terjaga kerahasiaannya.

1.7 Metode Penelitian

1.7.1 Metode Pengumpulan Data dan Informasi

a. Studi Pustaka

Dengan metode ini, penulis mendapatkan informasi yang berkaitan dengan steganografi melalui buku-buku referensi dan melalui berbagai situs di internet yang berkaitan, sehingga penulis dapat mengumpulkan data dan informasi yang diperlukan.

b. Studi Literatur

Penulis mencoba mencari perbandingan dengan studi sejenis dari beberapa penulisan di beberapa karya ilmiah, seperti jurnal dan skripsi.

1.7.2 Metode Pengembangan Sistem

Dalam menyusun tugas akhir ini penulis menggunakan metode pengembangan sistem *Rapid Application Development* (RAD). Ada tiga fase luas pada RAD yang mengajak *user* maupun *analyst* dalam merencanakan, mendesain, dan mengimplementasi suatu sistem.

1. Fase Kebutuhan Perencanaan (*Requirement Planning Phase*)

Menentukan tujuan dan syarat dari informasi.

2. Proses Desain RAD (*RAD Design Workshop*)

Perancangan proses yang akan terjadi dalam sistem, perancangan *input* dan *output interface*, serta pengkodean terhadap rancangan rancangan yang telah didefinisikan.

3. Fase Implementasi (*Implementation Phase*)

Pada tahapan ini dilakukan pengujian terhadap sistem dan melakukan pengenalan terhadap sistem (Kendall, 2005).

1.8 Sistematika Penulisan

Dalam penyusunan skripsi ini penulis menyajikan tulisan ini terdiri atas:

BAB I PENDAHULUAN

Bab ini terdiri dari latar belakang masalah, identifikasi masalah, rumusan masalah, batasan masalah, tujuan, manfaat, metode penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini berisi uraian tentang landasan teori yang diperlukan dalam analisis dan perancangan aplikasi steganografi pada citra digital menggunakan metode LSB (*Least Significant Bit*).

BAB III METODE PENELITIAN

Bab ini menguraikan secara rinci metode yang digunakan dalam analisis dan perancangan aplikasi.

BAB IV ANALISIS DAN PERANCANGAN

Bab ini membahas mengenai analisis, perancangan dan pengujian sistem.

BAB V PENUTUP

Bab ini berisi kesimpulan dari seluruh bab dan saran-saran untuk pengembangan sistem lebih lanjut.

BAB II

LANDASAN TEORI

2.1 Pengertian Analisis

Analisis berkaitan dengan pemahaman dan pemodelan aplikasi serta domain dimana aplikasi beroperasi. Masukan awal fase analisis adalah pernyataan masalah yang mendeskripsikan masalah yang ingin diselesaikan dan menyediakan pandangan konseptual terhadap sistem yang diusulkan

Sebutan lengkap analisis adalah analisis kebutuhan perangkat lunak (*software requirement analysis*). Analisis adalah mendaftarkan apa-apa yang harus dipenuhi oleh sistem perangkat lunak, bukan mengenai bagaimana sistem perangkat lunak melakukannya (Hariyanto, 2004).

2.2 Pengertian Perancangan

Perancangan merupakan penghubung antara spesifikasi kebutuhan dan implementasi. Perancangan merupakan rekayasa representasi yang berarti terhadap sesuatu yang hendak dibangun. Hasil perancangan harus dapat ditelusuri sampai ke spesifikasi kebutuhan dan dapat diukur kualitasnya berdasar kriteria-kriteria rancangan yang bagus. Perancangan menekankan pada solusi logic mengenai cara sistem memenuhi kebutuhan (Hariyanto, 2004).

2.3 Steganografi

2.3.1 Sejarah Steganografi

Usia steganografi hampir setara usia kriptografi. Steganografi sudah dikenal oleh bangsa Yunani sejak lama. Herodatus, seorang penguasa Yunani, mengirimkan pesan rahasia menggunakan kepala budak atau prajurit sebagai media. Caranya, rambut budak dibotaki, lalu pesan rahasia ditulis pada kulit kepala budak. Setelah rambut-rambut budak tumbuh cukup banyak (yang berarti menutupi pesan rahasia), budak tersebut dikirim ke tempat tujuan pesan untuk membawa pesan rahasia di kepalanya. Di tempat penerima kepala budak dibotaki kembali untuk membaca pesan yang tersembunyi di balik rambutnya. Pesan tersebut berisi peringatan tentang invasi dari Bangsa Persia.

Bangsa Romawi mengenal steganografi dengan menggunakan tinta tak-nampak (*invisible ink*) untuk menulis pesan. Tinta tersebut dibuat dari campuran sari buah, susu, dan cuka. Jika tinta digunakan untuk menulis maka tulisannya tidak tampak. Tulisan di atas kertas dapat dibaca dengan cara memanaskan kertas tersebut (Munir, 2006).

Pada abad 20, steganografi benar-benar mengalami perkembangan. Selama berlangsung perang Boer, Lord Baden Powell (pendiri gerakan kepanduan) yang bertugas untuk membuat tanda posisi sasaran dari basis artileri tentara Boer, untuk alasan keamanan, Baden Powell menggambar peta-peta posisi musuh pada sayap kupu-kupu agar gambar-gambar peta sasaran tersebut terkamufase (<http://ilmukomputer.org>).

Selama Perang dunia II, agen-agen spionase juga menggunakan steganografi untuk mengirim pesan. Caranya dengan menggunakan titik-titik yang sangat kecil sehingga keberadaanya tidak dapat dibedakan pada tulisan biasa yang diketik.

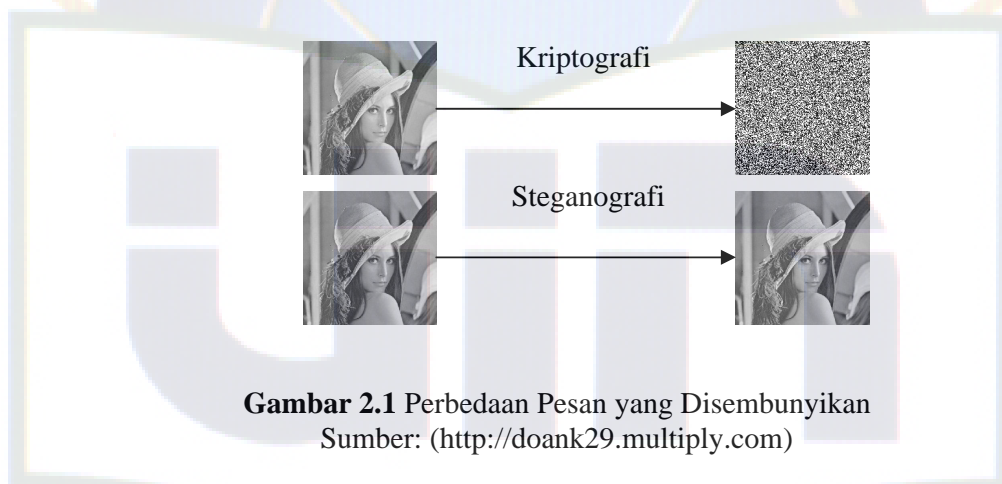
Saat ini steganografi sudah banyak diimplementasikan pada media digital. Steganografi digital menggunakan media digital sebagai penampung, seperti citra digital, video digital, atau audio. Informasi yang disembunyikan juga berbentuk digital seperti teks, citra, data audio, atau data video. Steganografi digital dapat dipakai di negara-negara yang menerapkan sensor ketat terhadap informasi atau di negara di mana enkripsi pesan terlarang. Pada negara-negara seperti itu informasi rahasia dapat disembunyikan dengan menggunakan steganografi (Munir, 2006)

2.3.2 Pengertian Steganografi

Steganografi berasal dari Bahasa Yunani, yaitu “steganos” yang artinya "tulisan tersembunyi (*covered writing*)" dan “graphos” yang berarti tulisan. Steganografi adalah ilmu dan seni menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui (Munir, 2006:). Sedangkan menurut Doni Ariyus, steganografi merupakan cabang ilmu yang mempelajari tentang bagaimana menyembunyikan suatu informasi “rahasia” di dalam informasi lainnya (Ariyus, 2006)

Steganografi membutuhkan dua properti yaitu media penampung dan pesan rahasia. Media penampung yang umum digunakan adalah gambar, suara, video, atau teks. Pesan yang disembunyikan dapat berupa sebuah artikel, gambar, daftar barang, kode barang, atau pesan lain (Munir, 2006). Steganografi berbeda dengan

kriptografi, perbedaannya terletak pada bagaimana proses penyembunyian data dan hasil akhir dari proses tersebut. Kriptografi melakukan proses pengacakan data aslinya sehingga menghasilkan data terenkripsi yang benar-benar acak dan berbeda dengan aslinya, sedangkan steganografi menyembunyikan dalam data lain yang akan ditumpanginya tanpa mengubah data yang ditumpanginya tersebut, sehingga data yang ditumpanginya sebelum dan setelah proses penyembunyian hampir sama (Munir, 2006). Perbedaan kriptografi dan steganografi dapat diilustrasikan pada Gambar 2.1.



Tujuan steganografi adalah untuk menghindari kecurigaan (*conspicuous*) sedangkan kriptografi menyembunyikan *isi (content)* pesan agar pesan tidak dapat dibaca (<http://haryanto.staff.gunadarma.ac.id>).

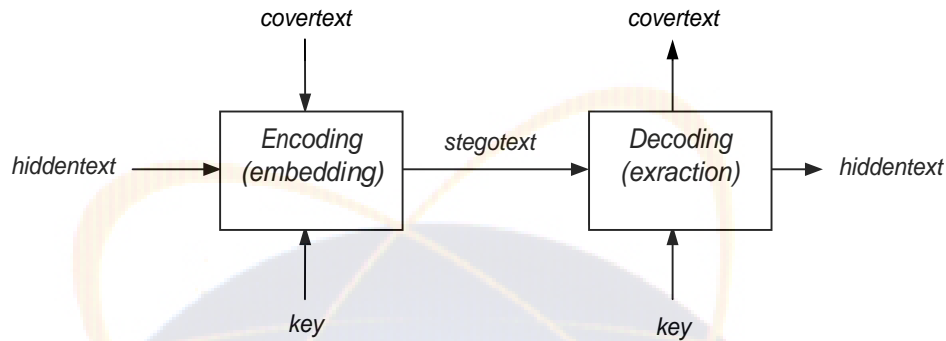
Steganografi memanfaatkan kelemahan indera manusia seperti indera pendengaran dan indera penglihatan. Dengan adanya kelemahan ini steganografi dapat diterapkan di berbagai media *digital*. Hasil keluaran *file* yang telah disisipi pesan mempunyai persepsi bentuk yang sama dengan *file* aslinya. Penggunaan

komputer diperlukan untuk mengetahui keberadaan pesan yang tersembunyi dalam *file digital*.

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 *digital*, baik *hiddentext* maupun *coverttext* dapat berupa teks, citra, audio, maupun video. Jadi, kita dapat menyembunyikan pesan berupa kode program di dalam sebuah citra, atau video, dan kita juga dapat menyembunyikan gambar rahasia di dalam citra lain atau di dalam sebuah berkas musik *mp3*. Penyisipan pesan ke dalam media *coverttext* dinamakan *encoding*, sedangkan ekstraksi pesan dari *stegotext* dinamakan *decoding*. Kedua proses ini mungkin memerlukan kunci rahasia (yang dinamakan *stegokey*) agar hanya pihak yang berhak saja yang dapat melakukan penyisipan pesan dan ekstraksi pesan (Munir, 2006).

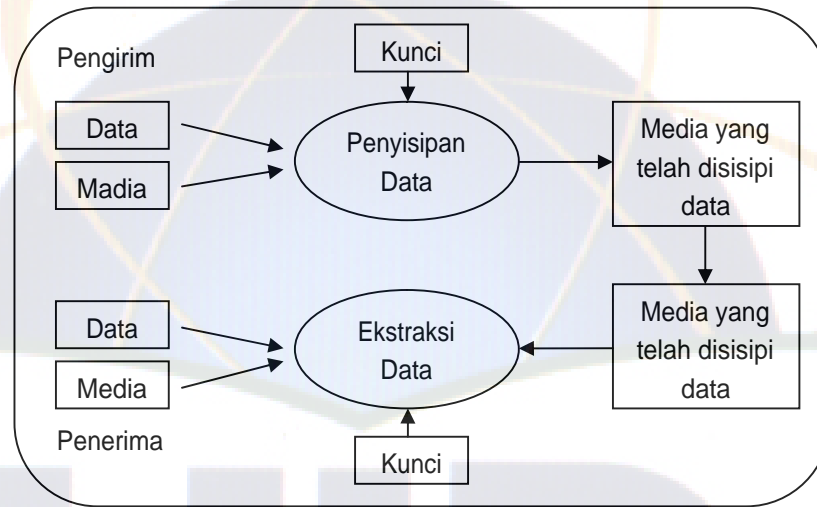


Gambar 2.2 Diagram Penyisipan dan Ekstraksi Pesan
Sumber: (Munir, 2006)

Penyembunyian pesan rahasia ke dalam media penampung pasti mengubah kualitas media tersebut. Kriteria yang harus diperhatikan dalam penyembunyian pesan adalah:

1. *Imperceptibility*. Keberadaan pesan rahasia tidak dapat dipersepsikan oleh indrawi. Misalnya, jika *covertext* berupa citra, maka penyisipan pesan membuat citra *stegotext* sukar dibedakan oleh mata dengan citra *covertext*-nya. Jika *covertext* berupa audio (misalnya berkas mp3, wav, midi, dan sebagainya), maka indra telinga tidak dapat mendeteksi perubahan pada audio *stegotext*-nya.
2. *Fidelity*. Mutu media penampung tidak berubah banyak akibat penyisipan. Perubahan tersebut tidak dapat dipersepsi oleh indrawi. Misalnya, jika *covertext* berupa citra, maka penyisipan pesan membuat citra *stegotext* sukar dibedakan oleh mata dengan citra *covertext*-nya. Jika *covertext* berupa audio (misalnya berkas mp3, wav, midi, dan sebagainya), maka audio *stegotext* tidak rusak dan indra telinga tidak dapat mendeteksi perubahan tersebut.

3. *Recovery*. Pesan yang disembunyikan harus dapat rekonstruksi kembali (*reveal*). Karena tujuan steganografi adalah *data hiding*, maka sewaktu-waktu pesan rahasia di dalam *stegotext* harus dapat diambil kembali untuk digunakan lebih lanjut (Munir, 2006).



Gambar 2.3 Diagram Sistem Steganografi
Sumber: (<http://doank29.multiply.com>)

2.3.3 Metode Steganografi

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

1. Ranah spasial (waktu) (*spasial/time domain*)

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

2. Ranah *transform* (*transform domain*)

Teknik ini memodifikasi langsung hasil transformasi frekuensi sinyal. Contoh metode yang tergolong ke dalam teknik ranah frekuensi adalah *spread spectrum* (Munir, 2006).

Sedangkan ada empat jenis metode Steganografi, yaitu:

1. *Algorithms and Transformation*

Algoritma *compression* (kompresi) 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*).

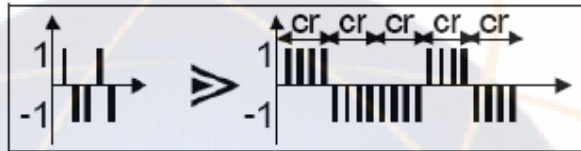
2. *Redundant Pattern Encoding*

Redundant Pattern Encoding adalah menggambar pesan kecil pada kebanyakan gambar. Keuntungan dari metode ini adalah dapat bertahan dari *cropping* (kegagalan), kerugiannya yaitu tidak dapat menggambar pesan yang lebih besar.

3. *Spread Spectrum method*

Spread Spectrum steganografi terpecah-pecah sebagai pesan yang diacak (*encrypt*) 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)

(<http://www.students.if.itb.ac.id>). Contoh dari penyebaran bit-bit informasi dapat dilihat pada Gambar 2.4. Faktor pengali dilambangkan dengan cr yang bernilai skalar. Panjang bit-bit hasil penyebaran ini menjadi cr kali panjang bit-bit awal.



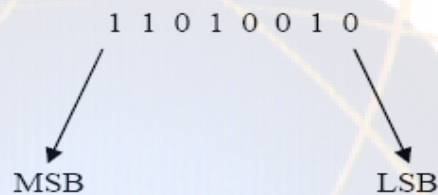
Gambar 2.4 *Spread Spectrum method*
Sumber: (<http://www.students.if.itb.ac.id>)

4. *Least Significant Bit (LSB)*

Metode yang digunakan untuk menyembunyikan pesan pada media digital tersebut berbeda-beda. Contohnya pada *file image* pesan dapat disembunyikan dengan menggunakan cara menyisipkannya pada bit rendah atau bit yang paling kanan (LSB) pada data *pixel* yang menyusun *file* tersebut. Seperti kita ketahui untuk *file bitmap* 24 bit maka setiap *pixel* (titik) pada gambar tersebut terdiri dari susunan tiga warna yaitu 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 (<http://doank29.multiply.com>).

2.4 LSB (*Least Significant Bit*)

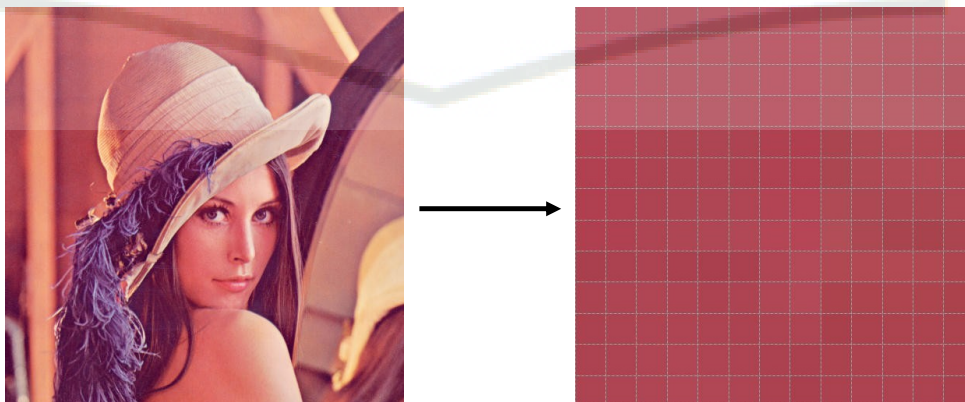
Pada susunan bit di dalam sebuah *byte* (1 *byte* = 8 bit), ada bit yang paling berarti (*most significant bit* atau MSB) dan bit yang paling kurang berarti (*least significant bit* atau LSB) (<http://www.ittelkom.ac.id>). Terlihat pada Gambar 2.5. yang menggambarkan contoh nilai biner pada 8 bit.



Gambar 2.5 *most significant bit* (MSB) dan *least significant bit* (LSB)

Sumber: (<http://doank29.multiply.com>)

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. Misalkan *byte* tersebut menyatakan warna merah, maka perubahan satu bit LSB tidak mengubah warna merah tersebut secara berarti, dan mata manusia tidak dapat membedakan perubahan yang sangat kecil itu. Misalkan ada sebuah gambar dengan nilai pexelnya sebagai berikut :



Gambar 2.6 Proses *File Image* Menjadi Kumpulan *Pixel-Pixel*

Pada dasarnya sebuah gambar bitmap merupakan kumpulan dari titik-titik yang disebut *pixel*. *Pixel-pixel* disetiap gambar mempunyai nilai berbeda-beda.

R:181 G: 69 B: 85	R:180 G: 69 B: 85	R:177 G: 71 B: 85	R:176 G: 71 B: 85	R:176 G: 71 B: 85	R:176 G: 70 B: 84	R:178 G: 67 B: 83	R:178 G: 66 B: 82	R:176 G: 72 B: 83	R:175 G: 71 B: 82
R:181 G: 69 B: 85	R:180 G: 69 B: 85	R:177 G: 71 B: 85	R:176 G: 71 B: 85	R:175 G: 70 B: 84	R:175 G: 69 B: 83	R:177 G: 66 B: 82	R:177 G: 65 B: 81	R:174 G: 70 B: 81	R:176 G: 72 B: 83
R:179 G: 67 B: 83	R:178 G: 67 B: 83	R:175 G: 69 B: 83	R:174 G: 69 B: 83	R:174 G: 69 B: 83	R:174 G: 68 B: 82	R:176 G: 65 B: 81	R:176 G: 64 B: 80	R:171 G: 69 B: 80	R:175 G: 73 B: 84
R:177 G: 65 B: 81	R:176 G: 65 B: 81	R:174 G: 68 B: 82	R:174 G: 69 B: 83	R:174 G: 69 B: 83	R:174 G: 68 B: 82	R:177 G: 66 B: 82	R:177 G: 65 B: 81	R:171 G: 71 B: 81	R:175 G: 73 B: 84
R:176 G: 64 B: 80	R:176 G: 65 B: 81	R:174 G: 68 B: 82	R:174 G: 69 B: 83	R:175 G: 70 B: 84	R:176 G: 70 B: 84	R:179 G: 68 B: 84	R:180 G: 68 B: 84	R:175 G: 75 B: 85	R:173 G: 73 B: 83
R:176 G: 64 B: 80	R:176 G: 65 B: 81	R:175 G: 69 B: 83	R:175 G: 70 B: 84	R:176 G: 71 B: 85	R:178 G: 72 B: 86	R:181 G: 70 B: 86	R:182 G: 70 B: 86	R:178 G: 78 B: 88	R:173 G: 73 B: 83
R:170 G: 64 B: 78	R:172 G: 66 B: 80	R:173 G: 67 B: 81	R:171 G: 65 B: 79	R:170 G: 64 B: 78	R:171 G: 65 B: 79	R:176 G: 70 B: 84	R:180 G: 74 B: 88	R:178 G: 72 B: 82	R:179 G: 73 B: 83
R:173 G: 67 B: 81	R:175 G: 69 B: 83	R:176 G: 70 B: 84	R:175 G: 69 B: 83	R:173 G: 67 B: 81	R:174 G: 68 B: 82	R:177 G: 71 B: 85	R:180 G: 74 B: 88	R:173 G: 67 B: 77	R:174 G: 68 B: 78
R:172 G: 66 B: 80	R:174 G: 68 B: 82	R:175 G: 69 B: 83	R:175 G: 69 B: 83	R:174 G: 68 B: 82	R:173 G: 67 B: 81	R:174 G: 68 B: 82	R:175 G: 69 B: 83	R:170 G: 64 B: 76	R:172 G: 66 B: 78
R:173 G: 67 B: 81	R:174 G: 68 B: 82	R:176 G: 70 B: 84	R:177 G: 71 B: 85	R:175 G: 69 B: 83	R:174 G: 68 B: 82	R:172 G: 66 B: 80	R:171 G: 65 B: 79	R:172 G: 66 B: 78	R:173 G: 67 B: 79

Gambar 2.7 Nilai-Nilai *Pixel* Dalam Suatu *File Image*

Misalkan diambil nilai dari beberapa pixel pada gambar di atas, dimana nilai pixel dikonversikan dahulu ke dalam biner untuk menyisipkan sebuah karakter “R” = 01010010 di mana 01010010 adalah kode biner untuk 82 yang merupakan kode ASCII karakter “R”.

(00100111	11101001	11001000)
(00100111	11001000	11101001)
(11001000	00100111	11101001)

Segmen citra sebelum disisipkan

(00100110	11101001	11001000)
(00100111	11001000	11101000)
(11001001	00100110	11101001)

Segmen citra sesudah disisipkan “R”

Terlihat di atas perubahan pada contoh segmen data citra yang terdapat pada bit-bit yang paling kanan, setelah disisipkan ‘01010010’ sebagai data yang disembunyikan, bahwa perubahan bit hanya terjadi pada sisi yang paling kanan dari 8 bit yang ada.

Steganografi dengan metode LSB juga hanya mampu menyimpan informasi dengan ukuran yang sangat terbatas. Misalnya suatu citra 24-bit (R=8-bit, G=8-bit, B=8-bit) digunakan sebagai wadah untuk menyimpan data berukuran 100 bit, jika masing-masing komponen warnanya (RGB) digunakan satu *pixel* untuk menyimpan informasi rahasia tersebut, maka setiap *pixel*nya disimpan 3 bit informasi, sehingga setidaknya dibutuhkan citra wadah berukuran 34 *pixel* atau setara $34 \times 3 \times 8 = 816$ bit (8 kali lipat). Jadi suatu citra 24-bit jika digunakan untuk menyimpan informasi rahasia hanya mampu menampung informasi maksimum berukuran $1/8$ dari ukuran citra penampung tersebut (<http://webmail.informatika.org>).

Tabel 2.1 Keuntungan dan Kelemahan Metode LSB

No.	Keuntungan	Kelemahan
1.	Mudah diimplementasikan.	Tidak tahan terhadap pengubahan (modifikasi) terhadap <i>cover object</i> .
2.	Proses <i>encoding</i> cepat.	Mudah dihapus karena lokasi penyisipan diketahui (bit LSB).

3.	Mengeliminasi tingkat kecurigaan seseorang.	Besar pesan sangat tergantung dari media yang dipergunakan.
----	---	---

Sumber: (<http://haryanto.staff.gunadarma.ac.id>)

2.5 ASCII

Kode Standar Amerika untuk Pertukaran Informasi atau ASCII (*American Standard Code for Information Interchange*) merupakan suatu standar internasional dalam kode huruf dan simbol seperti *Hex* dan *Unicode* tetapi ASCII lebih bersifat universal, contohnya 124 adalah untuk karakter "|", selalu digunakan oleh komputer dan alat komunikasi lain untuk menunjukkan teks. Kode ASCII sebenarnya memiliki komposisi bilangan biner sebanyak 8 bit. Dimulai dari 00000000 hingga 11111111. Total kombinasi yang dihasilkan sebanyak 256, dimulai dari kode 0 hingga 255 dalam sistem bilangan Desimal (www.bobbemer.com).

Tabel 2.2 ASCII

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	\$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(72	48	H	104	68	h
9	09	Horizontal tab	41	29)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	
29	1D	Group separator	61	3D	=	93	5D]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	Ł	224	E0	α
129	81	ü	161	A1	í	193	C1	ł	225	E1	β
130	82	é	162	A2	ó	194	C2	ŧ	226	E2	Γ
131	83	â	163	A3	ú	195	C3	ţ	227	E3	π
132	84	ä	164	A4	ñ	196	C4	—	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	†	229	E5	σ
134	86	å	166	A6	ª	198	C6	‡	230	E6	μ
135	87	ç	167	A7	º	199	C7	‡	231	E7	ι
136	88	ê	168	A8	¿	200	C8	Ł	232	E8	Φ
137	89	ë	169	A9	ƒ	201	C9	ŕ	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	Ł	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	ŕ	235	EB	δ
140	8C	î	172	AC	¾	204	CC	‡	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	=	237	ED	∞
142	8E	Ä	174	AE	«	206	CE	‡	238	EE	ε
143	8F	Å	175	AF	»	207	CF	Ł	239	EF	Π
144	90	É	176	B0	▒	208	DO	Ł	240	FO	≡
145	91	æ	177	B1	▒	209	D1	ŕ	241	F1	±
146	92	Æ	178	B2	▒	210	D2	π	242	F2	≥
147	93	ô	179	B3		211	D3	Ł	243	F3	≤
148	94	ö	180	B4	†	212	D4	Ł	244	F4	[
149	95	ò	181	B5	‡	213	D5	ŕ	245	F5]
150	96	û	182	B6	‡	214	D6	π	246	F6	÷
151	97	ù	183	B7	π	215	D7	‡	247	F7	≈
152	98	ÿ	184	B8	ŕ	216	D8	‡	248	F8	°
153	99	Ö	185	B9	‡	217	D9	ŕ	249	F9	•
154	9A	Ü	186	BA	‡	218	DA	ŕ	250	FA	·
155	9B	ø	187	BB	ŕ	219	DB	▒	251	FB	√
156	9C	£	188	BC	▒	220	DC	▒	252	FC	¤
157	9D	¥	189	BD	▒	221	DD	▒	253	FD	£
158	9E	€	190	BE	ŕ	222	DE	▒	254	FE	■
159	9F	ƒ	191	BF	ŕ	223	DF	▒	255	FF	□

Sumber: (<http://www.google.co.id/>)

2.6 Multimedia

Multimedia adalah suatu istilah umum yang sering digunakan untuk menjelaskan penyebaran informasi, aplikasi, presentasi dan dokumen lain yang menggunakan gabungan kombinasi dari teks, grafik, animasi, dan video (Anderleigh, 1996).

Menurut (Hoffstetter, 2001), multimedia adalah penggunaan komputer untuk menyajikan dan mengkombinasikan teks, grafik, audio dan video dengan alat bantu (*tool*) yang memungkinkan pengguna (*user*) untuk melayani, berinteraksi, menciptakan dan berkomunikasi.

Multimedia juga dapat didefinisikan sebagai gabungan beberapa elemen, yaitu:

1. Elemen Teks

Terdiri dari huruf, nomor, dll. Aplikasi dari elemen teks adalah *Word Processing*, meliputi; *Microsoft word*, *Notepad*, dan *Office org*.

2. Elemen Grafik

Terdiri dari objek yang berupa garis- garis, kotak, bulatan, *shading*, *fill colours*. Aplikasi dari elemen ini adalah Draw Program, meliputi; *Corel draw*, *Adobe illustrator*, dan *Adobe flash*.

3. Elemen Gambar (*image*)

Terdiri dari gambar statik hasil kombinasi banyak *pixel*. Aplikasi elemen ini adalah Paint Program, meliputi; *Adobe photoshop*, *Scanner maching*, dan *ms. Paint*.

4. Elemen Audio

Terdiri dari *Sound* (suara) Seperti musik player, dll. Aplikasi elemen ini adalah Recording, meliputi; *Cooledit pro 2.0* dan komponen Player, meliputi; *Winamp, Jet audio, Real player*, dll.

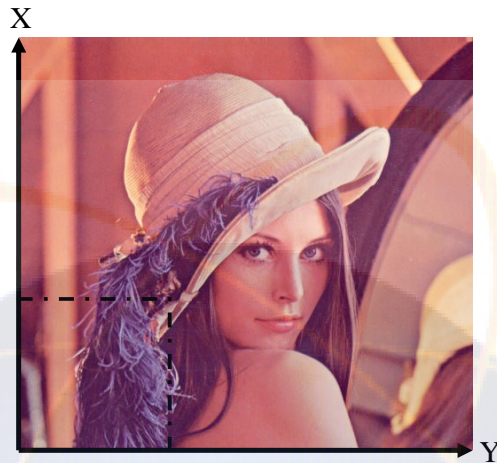
5. Elemen Visual

Terdiri dari susunan gambar yang digerakkan. Aplikasi dari elemen ini adalah Video Editing, meliputi; *Adobe premiere, Canopus edius*, dan *Pinnacle studios* serta Animasi, meliputi; *Adobe after effect, Adobe potoshop, Adobe Flash* (<http://ilmucerdas.wordpress.com>).

2.7 Citra Digital

2.7.1 Pengertian Citra Digital

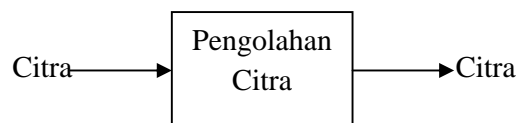
Citra adalah kumpulan *pixel-pixel* yang disusun dalam larik dua dimensi. *Pixel* (0,0) terletak pada sudut kiri atas pada citra, indeks X bergerak ke kanan dan indeks Y bergerak ke bawah (Ahmad, 2005). Citra digital dapat diartikan juga sebagai fungsi dua variabel, $f(x,y)$, di mana x dan y adalah koordinat spasial dan nilai $f(x,y)$ adalah intensitas citra pada koordinat tersebut, hal tersebut diilustrasikan pada Gambar 2.8. Teknologi dasar untuk menciptakan dan menampilkan warna pada citra digital berdasarkan pada penelitian bahwa sebuah warna merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau, dan biru (*Red, Green, Blue RGB*) (<http://www.itttelkom.ac.id>).



Gambar 2.8 Koordinat Spasial dan Nilai $f(x,y)$
 Sumber: (www.jcatki.no-ip.org)

2.7.2 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, maksudnya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas lebih baik dari pada citra masukan (Munir, 2004). Di dalam steganografi citra keluaran yang dimaksud adalah *stegotext* atau *stego-object*.



Gambar 2.9 Bagan Pengolahan citra
 Sumber: (Munir, 2004)

Secara umum tahapan pengolahan citra digital meliputi akuisisi citra, peningkatan kualitas citra, segmentasi citra, representasi dan uraian, pengenalan dan interpretasi.

1. Akuisisi citra

Pengambilan data dapat dilakukan dengan menggunakan berbagai media seperti kamera analog, kamera digital, handycamp, scanner, optical reader dan sebagainya. Citra yang dihasilkan belum tentu data digital, sehingga perlu didigitalisasi.

2. Peningkatan kualitas citra

Pada tahap ini dikenal dengan *pre-processing* dimana dalam meningkatkan kualitas citra dapat meningkatkan kemungkinan dalam keberhasilan pada tahap pengolahan citra digital berikutnya.

3. Segmentasi citra

Segmentasi bertujuan untuk memilih dan mengisolasi (memisahkan) suatu objek dari keseluruhan citra. Segmentasi terdiri dari *downsampling*, penapisan dan deteksi tepian. Tahap *downsampling* merupakan proses untuk menurunkan jumlah *pixel* dan menghilangkan sebagian informasi dari citra. Dengan resolusi citra yang tetap, *downsampling* menghasilkan ukuran citra yang lebih kecil. Tahap segmentasi selanjutnya adalah penapisan dengan filter median, hal ini dilakukan untuk menghilangkan derau yang biasanya muncul pada frekuensi tinggi pada spektrum citra. Pada penapisan dengan filter median, gray level citra pada setiap *pixel* digantikan dengan nilai median dari gray level pada *pixel* yang terdapat pada window filter. Tahap yang terakhir pada proses segmentasi yaitu deteksi tepian.

Pendekatan algoritma Canny dilakukan berdasarkan konvolusi fungsi citra dengan operator Gaussian dan turunan-turunannya. Pendeteksi tepi ini dirancang untuk merepresentasikan sebuah tepian yang ideal, dengan ketebalan yang diinginkan. Secara umum, proses segmentasi sangat penting dan secara langsung akan menentukan keakurasian sistem dalam proses identifikasi iris mata.

4. Representasi dan Uraian

Representasi mengacu pada data konversi dari hasil segmentasi ke bentuk yang lebih sesuai untuk proses pengolahan pada komputer. Keputusan pertama yang harus sudah dihasilkan pada tahap ini adalah data yang akan diproses dalam batasan-batasan atau daerah yang lengkap. Batas representasi digunakan ketika penekanannya pada karakteristik bentuk luar, dan area representasi digunakan ketika penekanannya pada karakteristik dalam, sebagai contoh tekstur. Setelah data telah direpresentasikan ke bentuk tipe yang lebih sesuai, tahap selanjutnya adalah menguraikan data.

5. Pengenalan dan Interpretasi

Pengenalan pola tidak hanya bertujuan untuk mendapatkan citra dengan suatu kualitas tertentu, tetapi juga untuk mengklasifikasikan bermacam-macam citra. Dari sejumlah citra diolah sehingga citra dengan ciri yang sama akan dikelompokkan pada suatu kelompok tertentu. Interpretasi meliputi penekanan dalam mengartikan objek yang dikenali (<http://www.ittelkom.ac.id>).

2.7.3 Perbandingan *File* Gambar BMP (Bitmap) dengan JPG, GIF, atau PNG

Tipe *file* BMP umum digunakan pada sistem operasi Windows dan OS/2. Kelebihan tipe *file* BMP adalah dapat dibuka oleh hampir semua program pengolah gambar. Baik *file* BMP yang terkompresi maupun tidak terkompresi, *file* BMP memiliki ukuran yang jauh lebih besar daripada tipe-tipe yang lain.

File BMP cocok digunakan untuk:

- *desktop background* di *windows*.
- sebagai gambar sementara yang mau diedit ulang tanpa menurunkan kualitasnya.

File BMP tidak cocok digunakan untuk:

- web atau blog, perlu dikonversi menjadi JPG, GIF, atau PNG.
- disimpan di *harddisk/flashdisk* tanpa di ZIP/RAR, kecuali *space* tidak masalah bagi Anda.

Tipe *file* JPG sangat sering digunakan untuk *web* atau *blog*. *File* JPG menggunakan teknik kompresi yang menyebabkan kualitas gambar turun (*lossy compression*). Setiap kali menyimpan ke tipe JPG dari tipe lain, ukuran gambar biasanya mengecil, tetapi kualitasnya turun dan tidak dapat dikembalikan lagi. Ukuran *file* BMP dapat turun menjadi sepersepuluhnya setelah dikonversi menjadi JPG. Meskipun dengan penurunan kualitas gambar, pada gambar-gambar tertentu (misalnya pemandangan), penurunan kualitas gambar hampir tidak terlihat mata.

File JPG cocok digunakan untuk:

- gambar yang memiliki banyak warna, misalnya foto wajah dan pemandangan.

- gambar yang memiliki gradien, misalnya perubahan warna yang perlahan-lahan dari merah ke biru.

File JPG tidak cocok digunakan untuk:

- gambar yang hanya memiliki warna sedikit seperti kartun atau komik.
- gambar yang memerlukan ketegasan garis seperti logo.

Tipe *file* GIF memungkinkan penambahan warna transparan dan dapat digunakan untuk membuat animasi sederhana, tetapi saat ini standar GIF hanya maksimal 256 warna saja. *File* ini menggunakan kompresi yang tidak menghilangkan data (*lossles compression*) tetapi penurunan jumlah warna menjadi 256 sering membuat gambar yang kaya warna seperti pemandangan menjadi tidak realistis.

File GIF cocok digunakan untuk:

- gambar dengan jumlah warna sedikit (dibawah 256).
- gambar yang memerlukan perbedaan warna yang tegas seperti logo tanpa gradien.
- gambar animasi sederhana seperti banner-banner iklan, header, dan sebagainya.

- print shoot (hasil dari print screen) dari program-program *simple* dengan jumlah warna sedikit.

File GIF tidak cocok digunakan untuk:

- gambar yang memiliki banyak warna seperti pemandangan.
- gambar yang di dalamnya terdapat warna gradien atau semburat.

Tipe *file* PNG merupakan solusi kompresi yang powerfull dengan warna yang lebih banyak (24 bit RGB + alpha). Berbeda dengan JPG yang menggunakan teknik kompresi yang menghilangkan data, *file* PNG menggunakan kompresi yang tidak menghilangkan data (lossles compression). Kelebihan *file* PNG adalah adanya warna transparan dan alpha. Warna alpha memungkinkan sebuah gambar transparan, tetapi gambar tersebut masih dapat dilihat mata seperti samar-samar atau bening. *File* PNG dapat diatur jumlah warnanya 64 bit (*true color* + alpha) sampai *indexed color* 1 bit. Dengan jumlah warna yang sama, kompresi *file* PNG lebih baik daripada GIF, tetapi memiliki ukuran *file* yang lebih besar daripada JPG. Kekurangan tipe PNG adalah belum populer sehingga sebagian browser tidak mendukungnya.

File PNG cocok digunakan untuk:

- gambar yang memiliki warna banyak.
- gambar yang mau diedit ulang tanpa menurunkan kualitas.

File PNG tidak cocok digunakan untuk:

- gambar yang jika dikompres dengan JPG hampir-hampir tidak terlihat penurunan kualitasnya (<http://www.tutorialgratis.net>).

2.8 DELPHI

2.8.1 Pengertian DELPHI

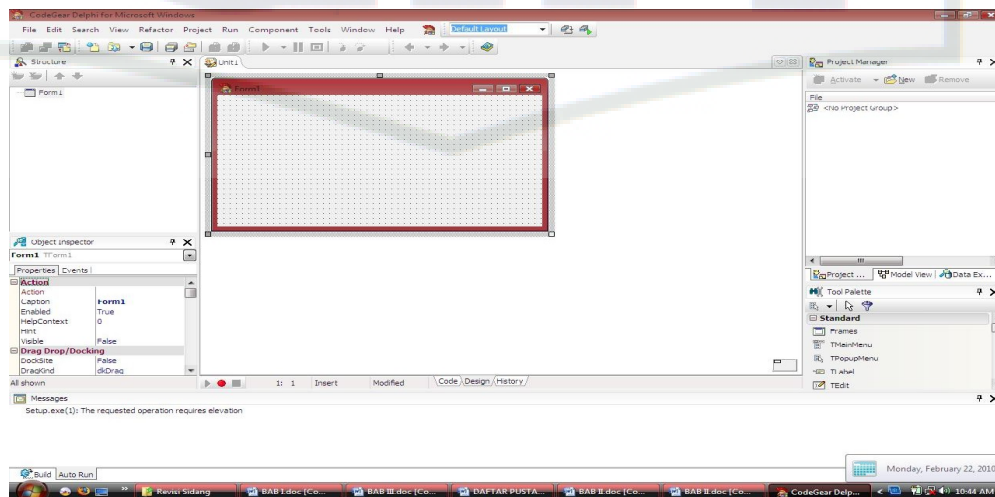
Bahasa pemrograman *Delphi* yang termasuk dalam salah satu bahasa pemrograman visual adalah generasi lanjut pemograman Pascal. Adapun, rilis (versi Delphi pertama) adalah tahun 1995, kemudian berlanjut sampai rilis ketujuh

pada tahun 2002 dan kini rilis terbarunya adalah Delphi 8 dan 2005. Pemrograman Delphi sendiri dibuat oleh Borland International Corporation dan berjalan di atas platform (sistem operasi) *Windows*, sedangkan sebagai pengetahuan yang berjalan di atas platform (sistem operasi) *Linux* adalah *Kylix*, yang merupakan saudara kembar pemrograman Delphi (Malik, 2006).

Borland Delphi atau lebih sering disebut dengan Delphi merupakan salah satu produk Borland yang sudah cukup terkenal. Kemampuannya untuk membuat aplikasi-aplikasi, baik aplikasi *database* maupun *non-database* sudah tidak perlu diragukan lagi, didukung dengan fasilitas yang cukup lengkap (Prasetyo, 2004).

2.8.2 Kelebihan Delphi

Kadang penulis berpikir, mengapa tidak memakai bahasa pemrograman visual lain (*Microsoft Visual Basic*, *C++ Builder*, atau lainnya)? Jawabannya karena objek dasar Delphi adalah bahasa pemrograman Pascal di mana kita sudah mengetahui bersama bahwa pemrograman Pascal sudah dikenal oleh kalangan masyarakat Indonesia, dengan survei dan berbagai sumber (Prasetyo, 2004).



Gambar 2.10 Tampilan *Form* Awal Delphi 2007 Win32

2.9 Perancangan Program

Di dalam merancang suatu program, dapat menggunakan beberapa alat bantu yaitu, metode pengembangan sistem, *flowchart*, dan *state transition diagram* (STD).

2.9.1 Model-model Pengembangan Sistem

Dalam sebuah perancangan perangkat lunak diperlukan model-model proses atau paradigma rekayasa perangkat lunak berdasarkan sifat aplikasi proyeknya, metode dan alat bantu yang dipakai, dan kontrol serta penyampaian yang dibutuhkan. Ada beberapa model dari proses perangkat lunak, yaitu: Model *Sekuensial Linear*, Model Prototipe, Model RAD (*Rapid Application Development*), Model Evolusioner, dan Model Formal. Untuk menyelesaikan masalah di dalam sebuah sistem harus dilakukan penggabungan strategi pengembangan yang melingkupi lapisan proses, metode, dan alat-alat bantu serta fase-fase generik (Pressman, 2002). Pada Table 2.3 dijelaskan beberapa metode dan perbedaannya.

Table 2.3 Beberapa Metode dan Perbedaannya

Metode	Kelebihan	Kekurangan	Penggunaan secara umum
<i>Sequensial Linier</i> (waterfall)	Metode ini baik digunakan untuk kebutuhan yang sudah diketahui dengan baik.	Iterasi yang sering terjadi menyebabkan masalah baru. bagi pelanggan sulit menentukan kebutuhan secara eksplisit dan harus sabar karena memakan waktu yang lama.	<i>Waterfall</i> bekerja dengan baik pada proyek skala kecil.

<i>Prototype</i>	Metode ini cukup efektif dengan mendapatkan kebutuhan dan aturan yang jelas dan pelanggan bisa langsung melihat sistem yang sebenarnya.	Pengembang kadang-kadang membuat implementasi sembarang, karena ingin working version selesai dengan cepat.	Prototyping dapat bekerja dengan baik jika ada kerjasama yang baik antara pengembang dengan pengguna
RAD	Metode ini lebih cepat dari <i>waterfall</i> jika kebutuhan dan batasan proyek sudah diketahui dengan baik. Dan bisa untuk dimodularisasi.	Karena proyek dipecah menjadi beberapa bagian, maka dibutuhkan banyak orang untuk membentuk suatu tim. Karena komponen-komponen yang sudah ada, fasilitas-fasilitas pada tiap komponen belum tentu digunakan seluruhnya sehingga kualitas program bisa menurun.	RAD cocok untuk aplikasi yang tidak mempunyai resiko teknis yang tinggi. RAD cocok untuk proyek yang memiliki SDM yang baik dan sudah berpengalaman.
<i>Iterative</i>	Fase desain, pengkodean, pengujian lebih cepat.	butuh waktu yang banyak untuk menganalisis dan terlalu banyak langkah yang dibutuhkan model	hanya cocok untuk <i>software</i> berskala besar
Spiral	Model ini digunakan untuk sistem skala besar.membutuhkan Konsiderasi langsung terhadap resiko teknis, sehingga dapat mengurangi terjadinya resiko yang lebih besar.	Resiko utama tidak ditemukan, maka masalah bisa muncul kemudian. Sehingga membutuhkan kemampuan manajemen dan perkiraan resiko (<i>risk assessment</i>) yang cukup tinggi.	Hanya cocok untuk <i>software</i> skala besar.

Sumber: (Pressman, 2002)

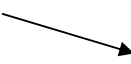

2.9.2 Diagram Alur (*Flowchart*)



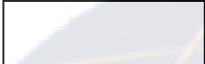
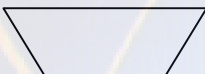
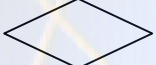





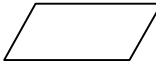
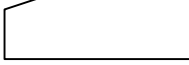
Komputer membutuhkan hal-hal yang terperinci, maka bahasa pemrograman bukan merupakan alat yang bisa dikatakan baik untuk merancang sebuah algoritma awal. Alat yang banyak dipakai untuk membuat algoritma adalah diagram alir. Diagram alur dapat menunjukkan secara jelas arus pengendalian algoritma, yakni bagaimana rangkaian pelaksanaan kegiatan. Suatu diagram alir memberikan gambaran dua dimensi berupa simbol-simbol grafis.

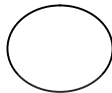



Flowchart program merupakan bagan alir yang menggambarkan urutan logika dari suatu prosedur pemecahan masalah. Tujuan utama dari penggunaan *flowchart* adalah untuk menggambarkan suatu tahapan penyelesaian masalah secara sederhana, terurai, rapi dan jelas dengan menggunakan simbol-simbol standar.

Dalam *flowchart* dikenal dua macam bentuk, yaitu Sistem *Flowchart* dan Program *Flowchart*. Sistem *Flowchart* menggambarkan tahapan proses dari suatu sistem, termasuk sistem multimedia. Sedangkan Program *Flowchart* menggambarkan urutan-urutan intruksi dari suatu program komputer (Al-Bahra' 2005). Simbol-simbol *flowchart* terlihat pada Tabel 2.4

Tabel 2.4 Simbol-simbol *Flowchart*

A. SIMBOL ARUS DAN ARAH	
SIMBOL	KEGUNAAN
 Simbol Arus/ Flow	Untuk menyatakan jalannya arus suatu proses.
 Simbol <i>Communication Link</i>	Untuk menyatakan bahwa adanya transisi suatu data/ informasi dari satu lokasi ke lokasi lainnya.
	Untuk menyatakan sambungan dari satu

Simbol		Connector	proses ke proses lainnya dalam halaman/ lembar yang sama.
		Simbol <i>Off-Line Connector</i>	Untuk menyatakan sambungan dari satu proses ke proses lainnya dalam halaman yang berbeda.
B. SIMBOL PROSES			
SIMBOL		KEGUNAAN	
		Simbol Proses/ <i>Offline Connector</i>	Untuk menyatakan sambungan dari suatu proses ke proses lainnya dalam halaman yang berbeda.
		Simbol Manual	Untuk menyatakan suatu tindakan (proses) yang tidak dilakukan oleh komputer (manual).
		Simbol <i>Decission/ Logika</i>	Untuk menunjukkan suatu kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban, ya/ tidak.
		Simbol <i>Predefined Process</i>	Untuk menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal.
		Simbol Terminal	Untuk menyatakan permulaan atau akhir suatu program.
		Simbol <i>Keying Operation</i>	Untuk menyatakan segala jenis operasi yang diproses menggunakan suatu mesin yang mempunyai <i>keyboard</i> .
		Simbol <i>off-line storage</i>	Untuk menunjukkan bahwa data dalam symbol ini akan disimpan ke suatu media tertentu.
		Simbol Manual Input	Untuk pemasukan data secara manual <i>on-line keyboard</i> .
C. SIMBOL INPUT-OUTPUT			
SIMBOL		KEGUNAAN	
		Simbol Input-Output	Simbol yang menyatakan proses input dan output tanpa tergantung dengan jenis peralatannya.
		Simbol <i>Punched Card</i>	Simbol yang menyatakan input berasal dari kartu atau output ditulis ke kartu.

 Simbol <i>Magnetic Tape Unit</i>	Simbol yang menyatakan input berasal dari pita magnetik atau output disimpan ke pita magnetic.
 Simbol <i>Disk Storage</i>	Untuk menyatakan input berasal dari disk atau output disimpan ke disk.
 Simbol <i>Document</i>	Untuk menyatakan input berasal dari kartu atau output disimpan ke pita magnetic.
 Simbol <i>Display</i>	Untuk menyatakan peralatan output yang digunakan yaitu layar (video computer).

Sumber: (Al-Bahra, 2005)

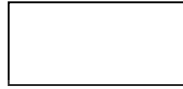
2.9.3 State Transition Diagram (STD)

Menunjukkan bagaimana sistem bertingkah laku sebagai akibat dari kejadian *eksternal*. *State Transition Diagram* (Diagram transisi keadaan) merupakan suatu modeling tool yang menggambarkan *Time Depend Behavior* dari suatu sistem (Al-Bahra, 2005).

Pada mulanya model *State Transition Diagram* ini hanya digunakan untuk menggambarkan suatu sistem yang bersifat *real time*. Ada dua cara kerja sistem ini yaitu pasif dan aktif. STD ini hanya digunakan untuk menuliskan urutan dan pergantian dari layar yang dapat terjadi, ketika pengguna sistem berada pada terminal.

1. Keadaan Sistem (*State*)

Setiap kotak mewakili suatu keadaan dimana sistem mungkin berada didalamnya. Seperti terlihat pada Gambar 2.11 *state* disimbolkan dengan simbol segi empat.



Gambar 2.11 Simbol State
Sumber: (Al-Bahra, 2005)

2. Perubahan Sistem (*Transition State*)

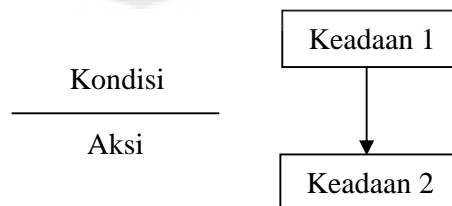
Simbol ini digunakan untuk menghubungkan satu keadaan dengan keadaan lain. Simbol ini digunakan jika sistem memiliki transisi dalam perilakunya. Gambar 2.12 menunjukkan *transition state*.



Gambar 2.12 Simbol *Transition State*
Sumber: (Al-Bahra, 2005)

3. Kondisi dan Aksi

Untuk melengkapi STD, dibutuhkan dua hal tambahan: yaitu kondisi sebelum keadaan berubah dan aksi dari pemakai untuk mengubah keadaan. Gambar 2.13 adalah ilustrasi dari kondisi dan aksi yang ditampilkan di sebelah anak panah yang menghubungkan dua keadaan (Al-Bahra, 2005).



Gambar 2.13 Simbol Kondisi dan Aksi
Sumber: (Al-Bahra, 2005)

2.10 Konsep RAD

2.10.1 Definisi RAD

RAD (*Rapid Application Development*) adalah sistem yang menggunakan teknik terstruktur, prototyping, dan JAD (*Joint Application Development*) untuk mengembangkan sistem secara cepat (Whitten, *et al.*2007).

Teknik terstruktur adalah sebuah teknik desain sistem yang menguraikan proses–proses sistem menjadi komponen–komponen yang dapat dikelola (Whitten, *et al.*2007).

Prototyping adalah teknik untuk membangun dengan cepat sebuah model sistem informasi yang fungsional tapi tidak lengkap dengan menggunakan peralatan pengembangan aplikasi (Whitten, *et al.*2007).

JAD (*joint application development*) adalah sebuah teknik yang melengkapi analisis sistem dan teknik desain lain dengan cara menekankan *participative development* diantara *system owner*, *users*, *designers*, dan *builder* (Whitten, *et al.*2007).

2.10.2 Tahapan – Tahapan RAD

Pengembangan sistem dalam penelitian ini menggunakan model RAD (*Rapid Application Development*), yaitu: (Kendall, 2005).

1. Tahap perencanaan syarat–syarat.

Dalam fase ini, pengguna dan penganalisis bertemu untuk mengidentifikasikan tujuan–tujuan aplikasi atau sistem serta untuk mengidentifikasikan syarat–syarat informasi yang ditimbulkan dari tujuan–tujuan tersebut. Fase ini memerlukan peran aktif mendalam

dari kedua kelompok tersebut, tidak hanya menunjukkan proposal atau dokumen. Selain itu, juga melibatkan pengguna dari beberapa level yang berada dalam organisasi. Orientasi dalam fase ini ialah menyelesaikan problem–problem perusahaan. Meskipun teknologi informasi dan sistem bisa mengarahkan sebagian sistem yang diajukan, fokusnya akan selalu tetap pada upaya pencapaian tujuan–tujuan perusahaan.

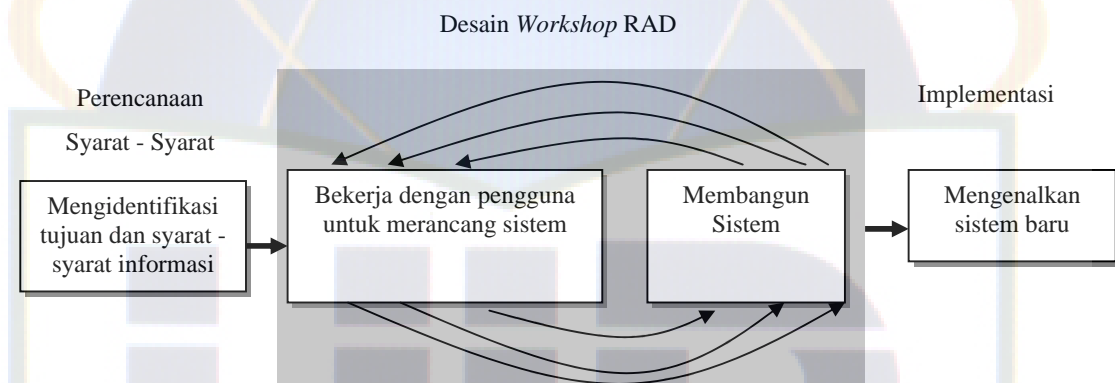
2. Tahap desain *workshop* RAD.

Fase ini adalah fase untuk merancang dan memperbaiki yang bisa digambarkan sebagai *workshop*. Saat membayangkan sebuah *workshop*, partisipasinya sangat intens, tidak pasif, dan biasanya bertahan. Biasanya para partisipan duduk mengitari meja bulat atau di kursi–kursi yang diatur membentuk huruf U dilengkapi meja sehingga masing–masing dapat melihat satu sama lain.

Selama *workshop* desain RAD, pengguna merespons *working prototype* yang ada dan penganalisis memperbaiki modul–modul yang dirancang berdasarkan respons pengguna. Format *workshop* sangat mengagumkan dan mampu memberi dorongan, dan jika pengguna atau penganalisis yang berpengalaman, tidak diragukan lagi bahwa usaha kreatif ini dapat mendorong pengembangan sampai pada tingkat terakselerasi.

3. Tahap implementasi.

Dalam Gambar 2.14 ditunjukkan bahwa dapat terlihat, penganalisis bekerja dengan para pengguna secara intens selama *workshop* untuk merancang aspek-aspek bisnis dan nonteknis dari perusahaan. Segera sesudah aspek-aspek ini disetujui dan sistem – sistem dibangun dan disaring, sistem-sistem baru atau bagian dari sistem diuji coba dan kemudian diperkenalkan kepada organisasi (Kendall, 2005).



Gambar 2.14 Tahapan-Tahapan RAD
Sumber: (Kendall, 2005)

2.10.3 Keunggulan RAD

Kelebihan menggunakan metode RAD adalah:

- Berguna untuk proyek-proyek tempat persyaratan-persyaratan pengguna tidak pasti dan tidak tepat.
- Mendorong pengguna aktif dan partisipasi manajemen.
- Proyek-proyek memiliki visibilitas dan dukungan lebih tinggi karena keterlibatan pengguna yang ekstensif selama proses.

- d. Para pengguna dan manajemen melihat solusi–solusi yang berbasis perangkat lunak dan bekerja lebih cepat.
- e. *Error* dan penghilangan cenderung untuk dideteksi lebih awal dalam prototipe daripada dalam model sistem.
- f. Pengujian dan pelatihan adalah produk tambahan alami dari pendekatan *prototyping* yang mendasar.

Pendekatan berulang adalah proses yang lebih alami karena perubahan adalah faktor yang diharapkan lain (Whitten, *et al.*2007).



BAB III

METODE PENELITIAN

Pada penyusunan skripsi ini, diperlukan data-data informasi sebagai bahan yang dapat mendukung kebenaran materi uraian pembahasan. Untuk menyelesaikan masalah yang ada dalam sebuah perancangan perangkat lunak ada beberapa tahap yang harus dilakukan. Dalam bab ini menguraikan tentang metode pengumpulan data dan metode pengembangan sistem yang digunakan peneliti. Metode penelitian yang digunakan adalah:

1.1 Metode Pengumpulan Data

1. Studi Pustaka

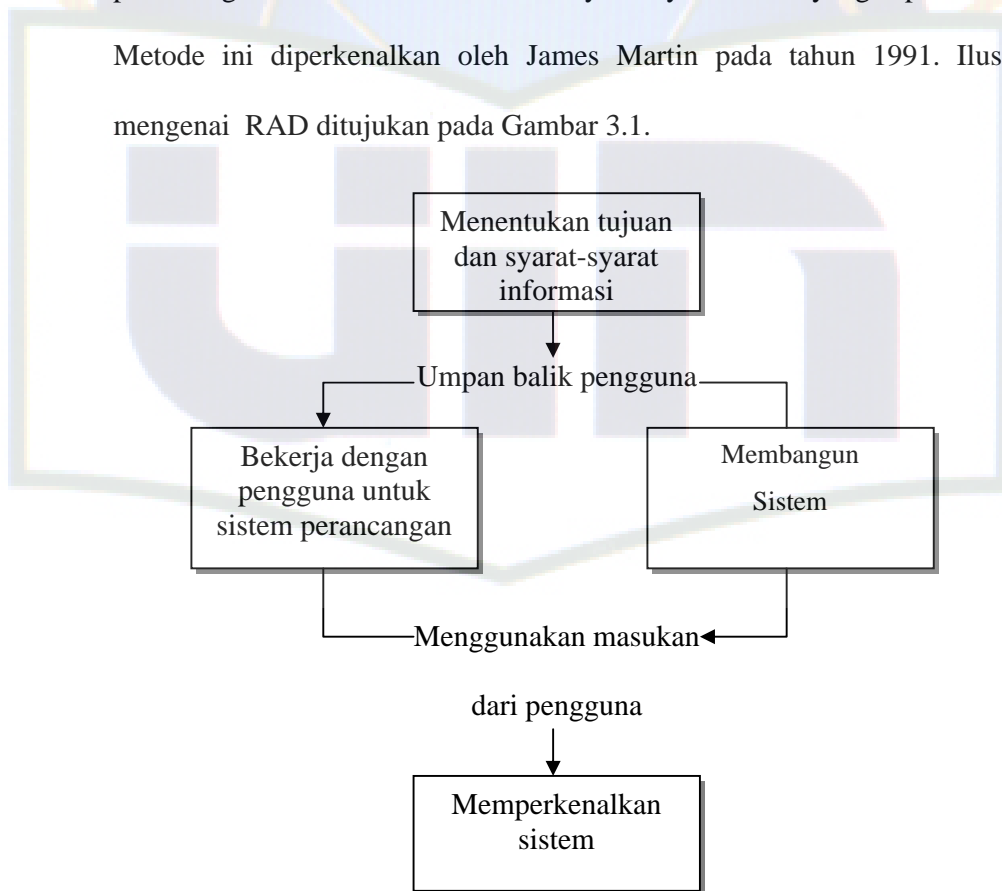
Dengan metode ini, penulis mendapatkan informasi apa saja yang berkaitan dengan steganografi melalui buku-buku referensi dan mencari melalui berbagai situs-situs di internet yang berkaitan, sehingga penulis dapat mengumpulkan data dan informasi yang diinginkan.

2. Studi Literatur

Penulis mencoba mencari perbandingan dengan studi sejenis dari beberapa penulisan di beberapa karya ilmiah, seperti jurnal dan skripsi.

1.2 Metode Pengembangan Sistem

Dalam menyusun tugas akhir ini penulis menggunakan metodologi pengembangan sistem *Rapid Application Development* (RAD). Model Rapid Application Development adalah pendekatan berorientasi objek yang digunakan terhadap pengembangan sistem yang mencakup suatu metode pengembangan perangkat-perangkat lunak. Model RAD adalah proses pengembangan perangkat lunak sekuensial linear yang menekankan siklus pengembangan yang pendek. Hal ini akan mempersingkat waktu dalam perancangan dan berusaha memenuhi syarat-syarat bisnis yang cepat berubah. Metode ini diperkenalkan oleh James Martin pada tahun 1991. Ilustrasi mengenai RAD ditujukan pada Gambar 3.1.



Gambar 3.1 Skema Sistem Model RAD

Dari Gambar 3.1. tersebut terlihat bahwa metode pengembangan sistem *Rapid Application Design* (RAD) terdiri dari tiga tahapan yaitu perencanaan syarat-syarat, desain *workshop* RAD dan implementasi.

3.2.1 Fase Perencanaan Syarat-Syarat

Pada fase penulis melakukan analisis kebutuhan dengan melakukan perbandingan terhadap beberapa aplikasi steganografi yang sudah ada, serta memutuskan fungsi apa yang harus difiturkan oleh aplikasi tersebut. Hasil analisis digunakan untuk mengidentifikasi tujuan dan syarat-syarat untuk mencapai tujuan, yaitu membuat aplikasi steganografi pada citra *digital* untuk memberikan solusi pengamanan pada pengiriman pesan. Untuk memudahkan maka harus didefinisikan sebagai berikut:

1. Apa saja yang menjadi input
2. Bagaimana proses digambarkan dalam alur dan basis aturannya.
3. Apa yang menjadi output atau hasilnya.

3.2.2 Proses Desain RAD (*RAD Design Workshop*)

Pada fase ini penulis melakukan perancangan proses dan perancangan antarmuka dari aplikasi, yaitu:

1. Perancangan Proses

Pada tahap ini akan dilakukan perancangan, evaluasi dan memperbaiki sistem sesuai dengan kebutuhan. Agar sistem yang sedang dibuat dapat dimanfaatkan secara optimal. Perancangan proses pada aplikasi ini digambarkan oleh *flowchart*. Pada fase ini juga dilakukan

pengkodean terhadap rancangan-rancangan yang telah didefinisikan.

Pengkodean yang dilakukan menggunakan *tools Delphi 2007 win32*.

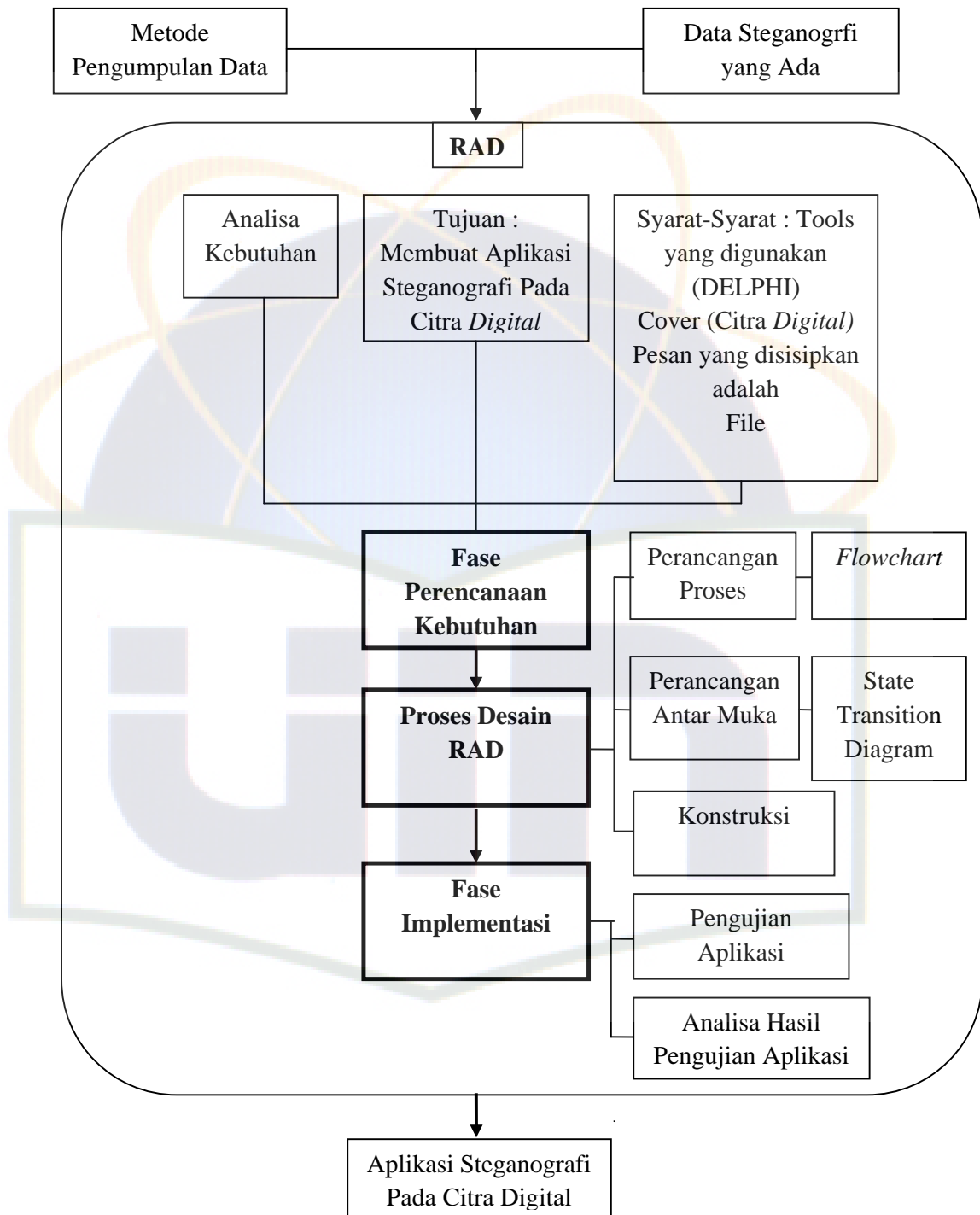
2. Perancangan Antar Muka Pemakai (*User Interface*)

Pada tahap ini akan dilakukan perancangan antar muka pemakai yang memberikan fasilitas komunikasi antar pemakai dan sistem, memberikan berbagai fasilitas informasi dan berbagai keterangan yang bertujuan untuk membantu mengarahkan alur penelusuran masalah sampai ditemukan solusi. Perancangan antar muka pemakai (*user interface*) pada aplikasi ini digambarkan oleh *state transition diagram*.

3.2.3 Fase Implementasi (*Implementation Phase*)

Pada fase ini penulis melakukan pengujian dan analisa terhadap aplikasi. Pengujian dilakukan dengan cara membandingkan kualitas dan besar data citra *digital* sebelum dan sesudah proses penyisipan pesan, serta seberapa besar tingkat keamanannya sehingga informasi rahasia benar– benar terjaga kerahasiannya dengan menggunakan teknik steganografi sebagai suatu solusi pengamanan pesan.

Skema dari metodologi penelitian yang dilakukan dalam pengembangan aplikasi steganografi pada citra *digital* ini ditunjukkan pada Gambar 3.2.



Gambar 3.2 Ilustrasi Metode Penelitian Pengembangan Aplikasi Steganografi pada Citra Digital.

BAB IV

ANALISIS DAN PERANCANGAN

Dalam menyusun tugas akhir ini penulis menggunakan metode pengembangan sistem *Rapid Application Development* (RAD). Terdapat tiga tahapan yaitu perencanaan syarat-syarat, desain *workshop* RAD dan implementasi.

4.1 Fase Perencanaan Syarat – Syarat

Perencanaan syarat-syarat terdiri atas analisis kebutuhan, tujuan dan syarat-syarat. Proses ini dilakukan untuk mengetahui apa saja syarat-syarat dan kebutuhan yang dibutuhkan dalam menganalisis untuk tujuan dari perancangan aplikasi ini.

4.1.1 Analisis Kebutuhan

Pada tahap ini dilakukan suatu perbandingan dari beberapa aplikasi steganografi yang sudah ada. Perbandingan dilakukan untuk mengidentifikasi tujuan-tujuan aplikasi serta untuk mengetahui syarat-syarat informasi yang ditimbulkan dari tujuan-tujuan tersebut. Perbandingan dilakukan dengan mempertimbangkan kemudahan penggunaan dan fitur masing-masing aplikasi steganografi yang pernah ada. Orientasi dalam tahap ini adalah bagaimana cara menyelesaikan masalah yang akan terjadi dalam mencapai tujuan perancangan aplikasi steganografi yang akan dilakukan.

4.1.2 Menentukan Tujuan

Tujuan dari perancangan ini adalah membuat suatu aplikasi steganografi menggunakan file dalam bentuk citra digital/ gambar berformat *.bmp, sebagai media penampung untuk menyimpan file serta untuk memberikan salah satu solusi dalam pengamanan file.

4.1.3 Menentukan syarat-syarat

Syarat-syarat untuk mencapai tujuan dalam pengembangan aplikasi steganografi pada citra digital terdiri dari perangkat lunak dan perangkat yang spesifikasinya adalah sebagai berikut:

Perangkat Lunak :

1. *Windows Vista Black Edition 2009*
2. *Delphi 2007 win32*
3. *Inno Setup 5*
4. *Matlab R2008b*

Perangkat Keras :

1. *Processor intel Core 2 duo*
2. *Harddisk 250GB*
3. *RAM 2GB*
4. *LCD widescreen display dengan resolusi 1280x800 pixel*
5. *DVD RW*

Media :

1. *File Image *.bmp*
2. *File Pesan *.txt, *.doc, *.xsl, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav.*

4.2 Fase Desain RAD

Desain RAD terdiri atas perancangan proses, dan perancangan antar muka. Perancangan proses dilakukan untuk merancang alur proses di dalam program sedangkan perancangan antar muka dilakukan untuk mempermudah pengguna menggunakan hasil dari aplikasi ini.

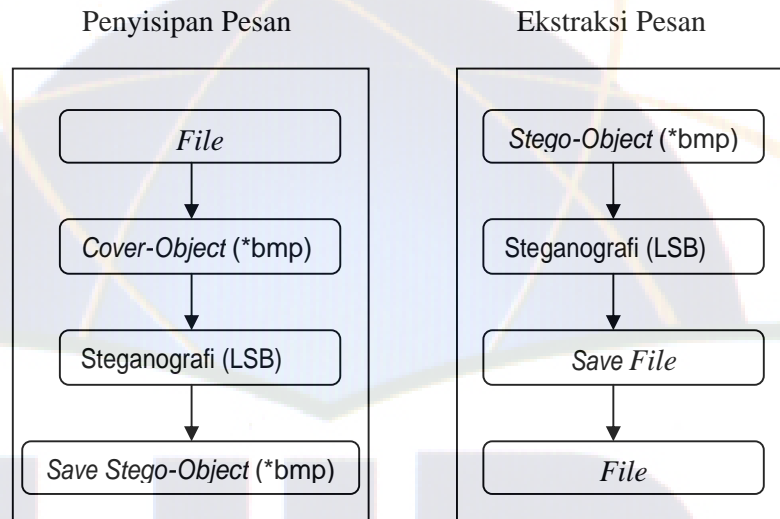
4.2.1 Perancangan Proses

Tahap perancangan proses dilakukan untuk perancangan, evaluasi dan memperbaiki sistem sesuai dengan kebutuhan, agar sistem yang sedang di buat dapat dimanfaatkan secara optimal. Aplikasi ini menggunakan metode LSB (*Least Significant Bit*), untuk teknik steganografi dalam aplikasi ini karena ukuran pesan yang dapat disisipkan ke media penampungnya relatif besar dengan mengganti setiap LSB dari media penampungnya. Selain itu metode LSB adalah metode yang paling mudah diaplikasikan dibandingkan metode steganografi yang lainnya.

Pada penyembunyian pesan, pesan dapat di ambil/ *copy* yang berupa *file* seperti *.txt, *.doc, *.xsl, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav.

kemudian disisipkan menggunakan algoritma LSB ke media penampungnya. Sehingga menghasilkan *stego-object* berupa *file* *.bmp yang telah disisipi pesan. Tahapan selanjutnya adalah proses ekstraksi. Pada tahap ini atau *file*

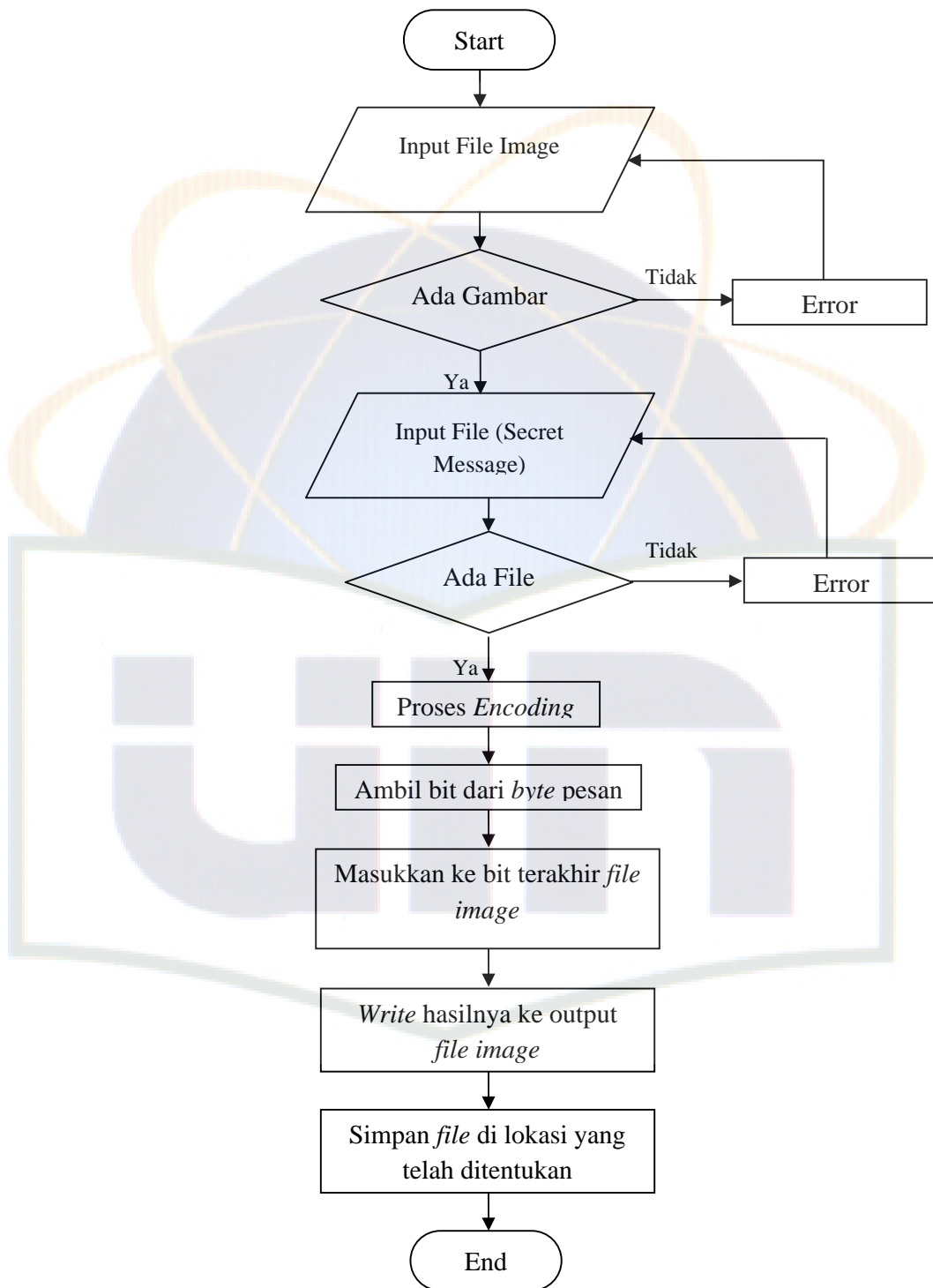
disembunyikan dipisahkan dari media penampungnya menggunakan teknik steganografi dengan metode LSB, sehingga menjadi *file* pesan yang dapat dibaca. Untuk memperjelas gambaran proses penyembunyian dan ekstraksi *file* pesan dapat dilihat pada Gambar 4.1.



Gambar 4.1 Proses Penyisipan dan Ekstraksi *File* Pesan

4.2.2 Flowchart Aplikasi Steganografi

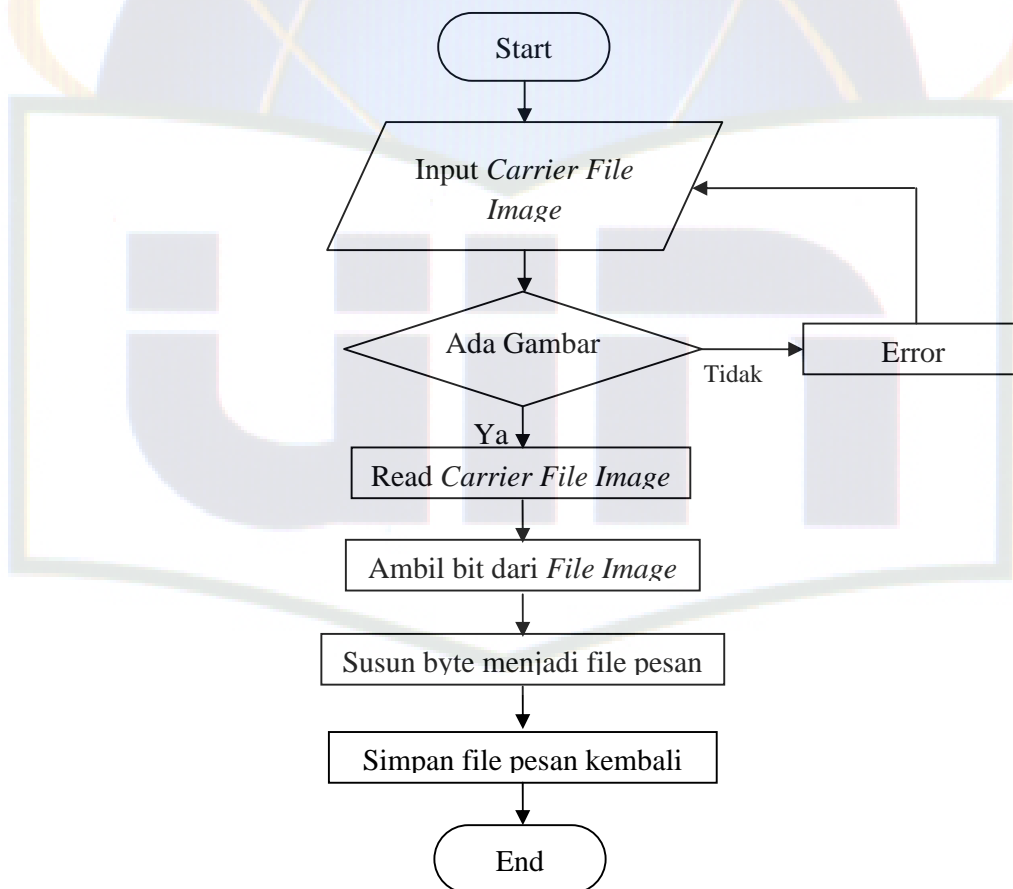
Pada tahap ini akan digambarkan alur proses penyisipan dan ekstraksi pesan rahasia dengan menggunakan *flowchart*.



Gambar 4.2 Flowchart Proses Penyisipan Pesan Rahasia

Gambar 4.2 menjelaskan informasi apa saja yang harus diinput untuk memulai proses penyisipan pesan rahasia. Informasi yang diinput haruslah lengkap, apabila tidak lengkap maka akan muncul error untuk mengingatkannya.

Setelah semua informasi terisi lengkap, program akan memanggil steganografi dengan menggunakan algoritma LSB untuk menyisipkan pesan di tiap LSB yang berada pada *File image*. Setelah semua proses selesai maka terbentuklah *file *.bmp* yang telah disisipkan pesan rahasia.



Gambar 4.3 Flowchart Proses Ekstraksi Pesan Rahasia

Gambar 4.3 menjelaskan informasi apa saja yang harus diinput untuk memulai proses ekstraksi pesan rahasia. Informasi haruslah lengkap, apabila tidak lengkap akan muncul *error* untuk mengingatkannya.







Setelah semua informasi terisi lengkap program steganografi akan menggunakan algoritma LSB untuk mengambil kembali *byte* pesan dari LSB *file image*, sehingga pesan rahasia dapat dibaca kembali. Setelah pesan rahasia dapat dibaca, maka proses ekstraksi pesan telah selesai.




4.2.3 Perancangan Antar Muka

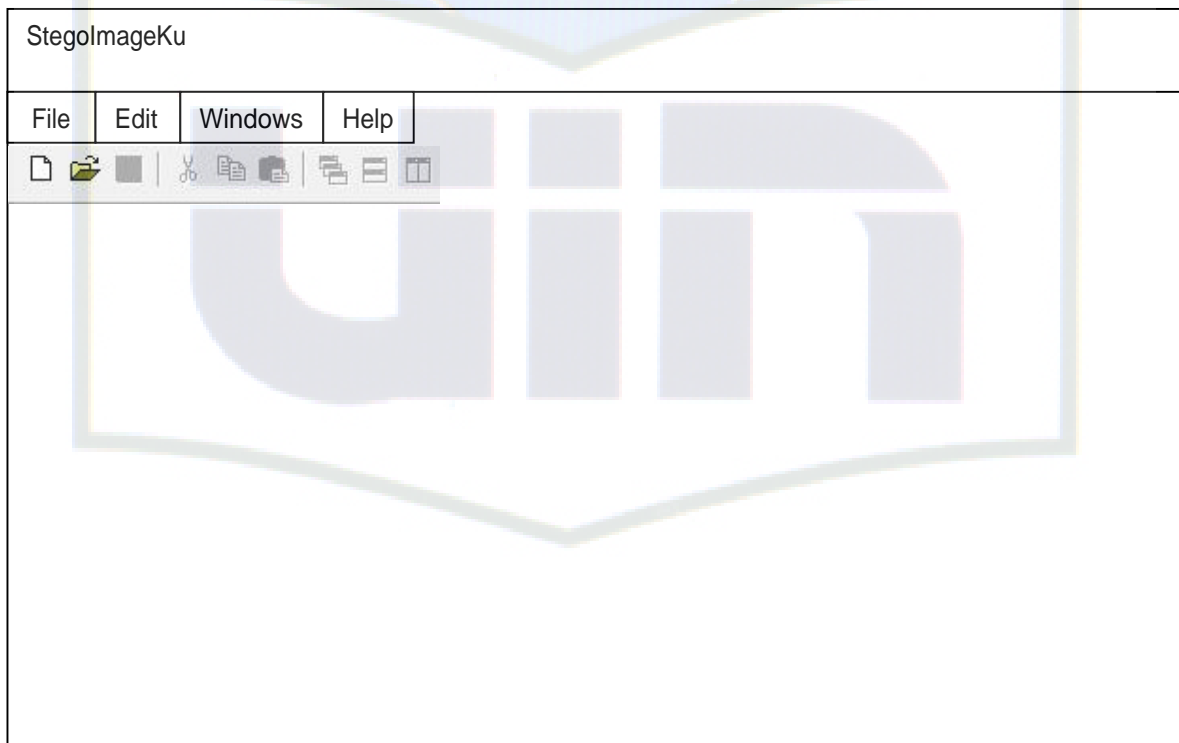
Dalam perancangan antar muka aplikasi steganografi pada citra digital ini, dibuat *form* tampilan yang akan ditampilkan dan dijelaskan fungsi *form* beserta fitur-fiturnya.

a. Perancangan *Form* induk

Form induk merupakan tampilan utama dalam program StegoImageKu yang didalamnya terdapat fitur sebagai berikut:

1.  Tombol *New*: Berfungsi untuk membuka *form Encoding & Decoding* yang baru.
2.  Tombol *Open*: Berfungsi untuk membuka *file*
3.  Tombol *Save*: Berfungsi untuk menyimpan *file*
4.  Tombol *Cut*: Berfungsi untuk memotong *file*
5.  Tombol *Copy*: Berfungsi untuk menggandakan *file*
6.  Tombol *Paste*: Berfungsi untuk memindahkan *file* setelah dicopy atau dicut

7.  Tombol *Cascade* : Berfungsi untuk menampilkan *form Encoding & Decoding* secara berurutan jika lebih dari 1 *form*.
8.  Tombol *Tile Horizontally* : Berfungsi untuk menampilkan *form Encoding & Decoding* secara horizontal jika lebih dari 1 *form*.
9.  Tombol *Tile Vertically* : Berfungsi untuk menampilkan *form Encoding & Decoding* secara vertical jika lebih dari 1 *form*.



Gambar 4.4 *Layout Form Induk*

b. Perancangan *Form* Utama Tab *Encoding*

Form utama Tab *Encoding* merupakan tampilan kedua dalam program StegoImageKu yang didalamnya terdapat fitur sebagai berikut:

1. Tombol *Browse*: Berfungsi untuk mencari letak lokasi direktori *File* yang akan disteganografi.
2. Tombol *Open*: Berfungsi untuk membuka *file image* yang akan dijadikan media penampung *file*.
3. Tombol *Encrypt*: Berfungsi untuk melakukan proses steganografi dengan metode LSB (*Least Significant Bit*).
4. Tombol *Clear*: Berfungsi untuk membatalkan membersihkan form.
5. *ScrollBox* + *Imagefile*: Berfungsi untuk letak *file image* yang Ingin dibuka dan disisipi file.
6. *EditBrowse* : Berfungsi untuk menampilkan letak direktori *file* Yang akan disisipkan.
7. *EditBit* : Berfungsi untuk menampilkan berapa bit yang Digunakan untuk proses penyisipan *file* kedalam *file image* yang menjadi media penampung.
8. *UpDown* : Berfungsi untuk menaikkan/ menurunkan penggunaan bit pada saat proses penyisipan *file* kedalam *file image*.

The image shows a software window titled "StegoImageKu". It has two tabs: "Encoding" (which is active) and "Decoding". In the "Encoding" tab, there is a text input field followed by a "Browse" button. Below this, it says "Bits per channel" followed by a dropdown menu showing "1" and a note "(Only for encryption)". At the bottom of the tab are three buttons: "Encrypt", "Open", and "Clear". On the right side of the window, there is a large dashed rectangular box labeled "File Image".

Gambar 4.5 Rancangan *Form* Utama *Tab Encoding*

c. Perancangan *Form* Utama *Tab Decoding*

Form utama *Tab Decoding* merupakan tampilan kedua dalam program StegoImageKu yang didalamnya terdapat fitur sebagai berikut:

1. Tombol *Open*: Berfungsi untuk membuka *file image* yang akan dijadikan media penampung *file*.
2. Tombol *Decrypt*: Berfungsi untuk melakukan proses ekstraksi *file* dari file image sebagai penampung.
3. Tombol *Clear*: Berfungsi untuk membatalkan membersihkan *form*.
4. Tombol *Exit*: Berfungsi untuk keluar dari aplikasi.

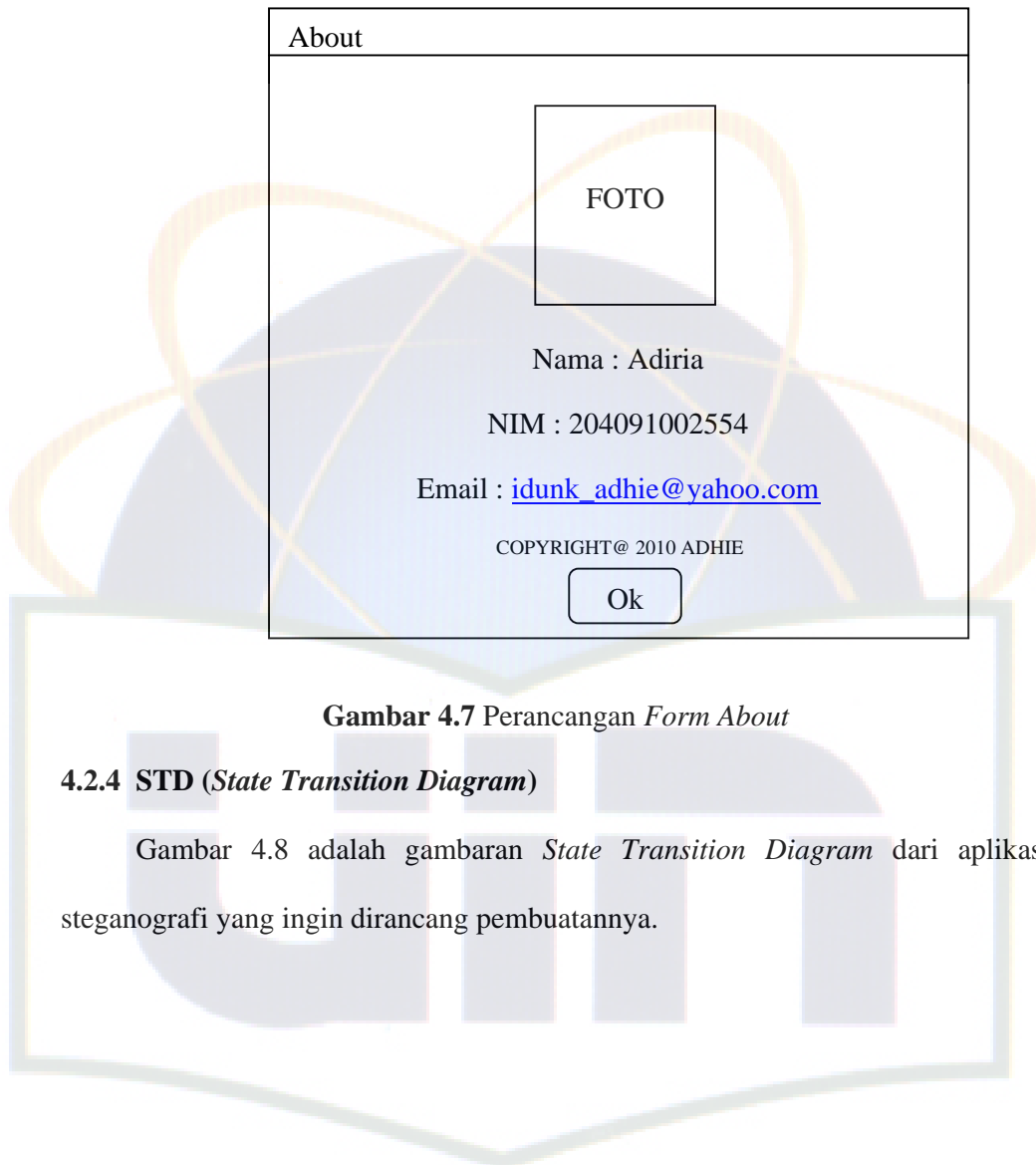
5. *ScrollBar + Imagefile*: Berfungsi untuk letak *file image* yang ingin dibuka dan disisipi *file*.

The image shows a software window titled "StegoImageKu". It has two tabs: "Encoding" and "Decoding", with "Decoding" currently selected. Below the tabs are three buttons: "Decrypt", "Open", and "Clear". To the right of these buttons is a large, empty rectangular area with a dashed border, labeled "File Image". The background of the window features a faint watermark of a blue dome and the letters "UIN".

Gambar 4.6 Rancangan *Form Utama Tab Decoding*

d. Perancangan *Form About*

Form about ini berfungsi sebagai *form* untuk menampilkan biodata diri.



The image shows a screenshot of a web form titled "About". The form is centered on a light gray background. At the top, there is a header bar with the word "About". Below this, there is a rectangular box labeled "FOTO" for a profile picture. Underneath the photo box, the form contains the following text: "Nama : Adiria", "NIM : 204091002554", and "Email : iduntk_adhie@yahoo.com". Below the email field, there is a line of text that reads "COPYRIGHT@ 2010 ADHIE". At the bottom of the form, there is a button labeled "Ok". The entire form is overlaid on a large, faint watermark of a building with a globe in front of it.

About

FOTO

Nama : Adiria

NIM : 204091002554

Email : iduntk_adhie@yahoo.com

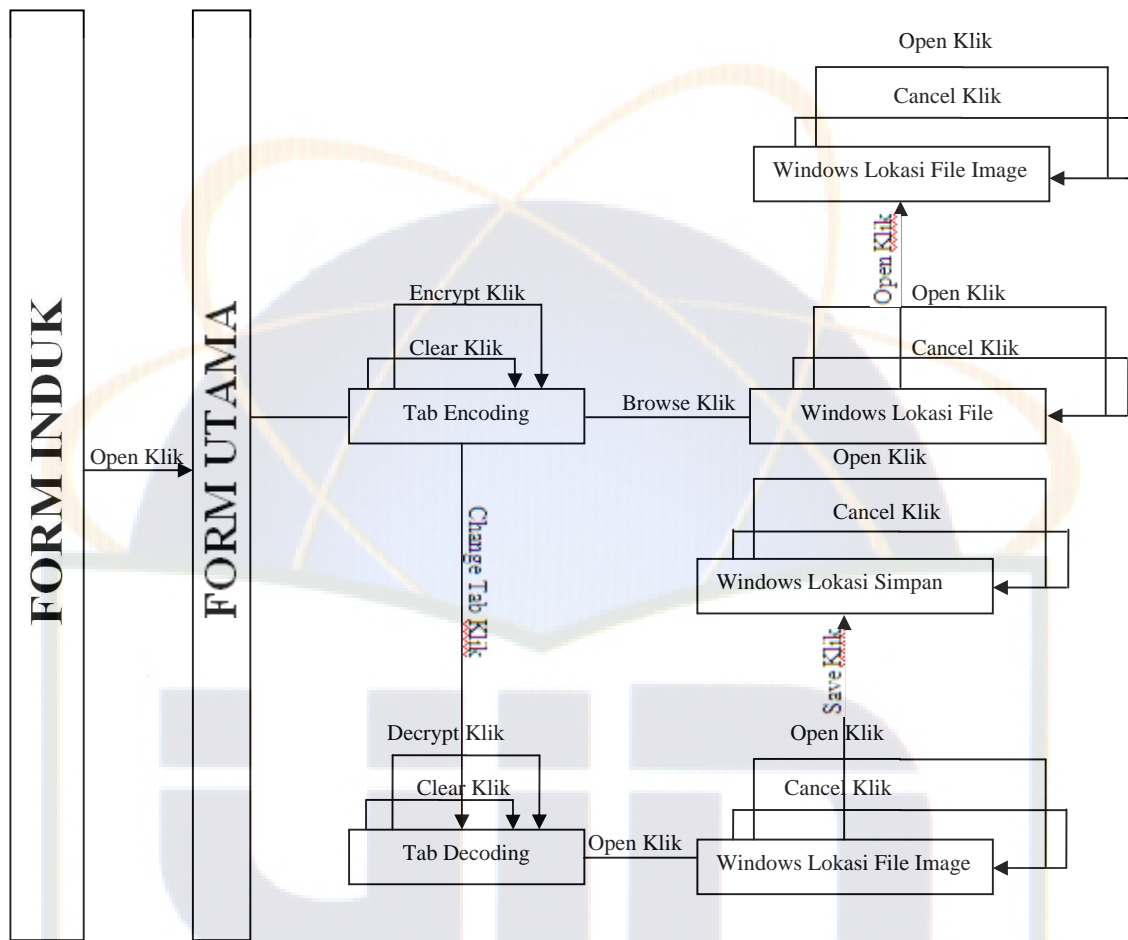
COPYRIGHT@ 2010 ADHIE

Ok

Gambar 4.7 Perancangan *Form About*

4.2.4 STD (*State Transition Diagram*)

Gambar 4.8 adalah gambaran *State Transition Diagram* dari aplikasi steganografi yang ingin dirancang pembuatannya.



Gambar 4.8 State Transition Diagram Aplikasi Steganografi pada Citra Digital

4.2.5 Konstruksi

Pada tahap ini dilakukan pengkodean terhadap rancangan-rancangan yang telah didefinisikan sebelumnya dengan menggunakan Delphi. Untuk melakukan pengkodean pada aplikasi ini, penulis menggunakan *software Delphi 2007 win 32*. Terdapat 6 buah Unit, yaitu “main.pas” sebagai *form* MDI yang berfungsi untuk *form* induk yang mengontrol seluruh fitur-fitur dalam aplikasi StedoImageKu, “ChildWin.pas sebagai sub *form* dari *form* MDI yang berfungsi untuk melakukan

proses encoding dan decoding file yang akan dienkrpsi dan dekripsi dengan meng-*input* file *image* sebagai media penampungnya dan file apa saja untuk disisipkan ke dalam *file image* tersebut. “Uprocess.pas” sebagai fungsi untuk melakukan proses penyisipan *file* ke dalam *file image*, “Ubit.pas” sebagai fungsi untuk membaca bit-bit yang berada di dalam *file image*, ”about.pas” sebagai *form about* yang berfungsi untuk menampilkan biodata penulis dan “Unit4.pas” sebagai *splash screen*.

4.3 Fase Implementasi

Dalam tahapan ini akan dilakukan pengujian dan analisis pengujian terhadap aplikasi StegoImage ini yang bertujuan untuk mengetahui tingkat keberhasilan dari aplikasi dalam mencapai hasil dan tujuan yang diinginkan.

4.3.1 Instalasi Aplikasi

Program instalasi aplikasi StegoImage ini menggunakan *software* Inno Setup 5. Dalam penggunaannya, *software* ini sangat mudah digunakan. Apabila pengguna tidak bisa membuat *script/ coding* secara langsung, program ini menyediakan menu “create a new script file using the script wizard”, jadi pengguna hanya mengikuti alur dari aplikasi dan selanjutnya aplikasi yang membuatkan script untuk proses instalasinya.

Di bawah ini adalah langkah-langkah instalasi aplikasi StegoImage ke dalam komputer pengguna:

1. Pilih program StegoImage Setup.exe, kemudian akan ada tampilan pembuka dari proses instalasi ini menggunakan bahasa inggris. Pilih

tombol *next* untuk melanjutkan instalasi, atau *cancel* untuk membatalkan instalasi.

2. Jika user menekan tombol *next* maka akan muncul halaman *license agreement*, yaitu *form* persetujuan yang menanyakan kesetujuan *user* atas semua konsekuensi jika menggunakan program ini. Apabila user memilih “I accept the agreement” yang berarti setuju maka proses instalasi akan berlanjut ke tahap selanjutnya. Apabila pengguna memilih “I don’t accept the agreement” yang berarti tidak setuju maka proses instalasi berhenti.
3. Tampilan untuk menentukan lokasi aplikasi StegoImage disimpan, akan langsung mengarah ke C:\Program Files\StegoImageKu, tetapi *user* dapat mengubah lokasi dengan cara menekan tombol *browse*. Di halaman ini juga terdapat informasi ukuran program yang akan diinstal.
4. Tampilan untuk menyimpan *shortcut folder* StegoImageKu di *start menu*.
5. Tampilan untuk menampilkan *shortcut* aplikasi StegoImage di *desktop*.
6. Tampilan proses instalasi, apabila informasi yang dibutuhkan sudah lengkap maka program siap untuk diinstal.
7. Tampilan informasi bahwa program telah selesai diinstal, dan aplikasi StegoImage akan muncul sebagai tanda bahwa proses instalasi telah berhasil.

4.3.2 Analisis Penyembunyian File

Analisis ini dilakukan bertujuan untuk mengetahui apakah *file image* dapat menampung *file* pesan tanpa adanya perubahan ukuran pada *file image*. Kemudian apakah *file* pesan tersebut dapat diambil kembali seperti semula tanpa adanya perubahan.

Tabel 4.1 Tabel Uji Penyembunyian File

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Monalisa.bmp	419454	Serbu.txt	265	MonalisaStego1.bmp	419454	Serbu.txt	265
Monalisa.bmp	419454	Gadis.txt	10977	MonalisaStego2.bmp	419454	Gadis.txt	10977
Monalisa.bmp	419454	Daftar Isi.doc	52224 52431,75	MonalisaStego3.bmp	419454	Daftar Isi.doc	52224

Dari hasil Table 4.1 terlihat bahwa *file image* dapat menampung *file* pesan tanpa adanya perubahan ukuran pada *file image*. Dan *file* pesan yang telah disisipi ke dalam *file image* dapat diambil kembali tanpa adanya perubahan pada *file* pesan. Ini dikarenakan *file* pesan yang disisipkan mengalami proses steganografi yang menggunakan metode LSB (Least Significant Bit), dimana file pesan disisipkan pada bit terakhir pada *file image*.

4.3.3 Analisis Format File Apa Saja yang Dapat Disisipkan ke File Image

Analisis ini dilakukan bertujuan untuk mengetahui jenis format apa saja yang dapat ditampung atau disisipi ke dalam *file image* tanpa adanya perubahan pada *file image* dan *file* pesan sebelum maupun setelah proses steganografi.

Tabel 4.2 Tabel Uji Format *File* Apa Saja yang Dapat Disisipkan ke *File Image*

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Pesawat.bmp	1440054	Index.ppt	163840	Pesawat1.bmp	1440054	Index1.ppt	163840
Pesawat.bmp	1440054	19SC096.JPG	33848	Pesawat2.bmp	1440054	19SC0961.JPG	33848
Pesawat.bmp	1440054	AbstrakHB.pdf	65536	Pesawat3.bmp	1440054	AbstrakHB.pdf	65536
Pesawat.bmp	1440054	Acctspay.mdb	102400	Pesawat4.bmp	1440054	Acctspay.mdb	102400
Pesawat.bmp	1440054	Biodata.doc	31744	Pesawat5.bmp	1440054	Biodata.doc	31744
Pesawat.bmp	1440054	browsesub.php	7290	Pesawat6.bmp	1440054	browsesub.php	7290
Pesawat.bmp	1440054	CA_CHING.WAV	18502	Pesawat7.bmp	1440054	CA_CHING.WAV	18502
Pesawat.bmp	1440054	coding.txt	21566	Pesawat8.bmp	1440054	coding.txt	21566
Pesawat.bmp	1440054	salary.xls	58880	Pesawat9.bmp	1440054	salary.xls	58880
Pesawat.bmp	1440054	web.htm	31051	Pesawat10.bmp	1440054	web.htm	31051

Dari hasil Table 4.2 terlihat bahwa *file image* dapat menampung *file* pesan berupa format apa saja tanpa adanya perubahan pada *file image*. Dan *file* pesan yang telah disisipi kedalam *file image* dapat diambil kembali tanpa adanya perubahan pada *file* pesan dengan syarat format *file* pesan sebelum maupun sesudah proses steganografi harus sama. Ini dikarenakan *file* pesan yang disisipkan mengalami proses steganografi yang menggunakan metode LSB (*Least Significant Bit*), dimana *byte-byte file* pesan disisipkan pada bit terakhir pada *file image*.

4.3.4 Analisis Penyisipan dan Ekstraksi *File* Pesan terhadap Tiga *File Image* yang Berbeda

Analisis ini dilakukan untuk membuktikan *file* pesan dapat disisipkan ke dalam *file* image yang berbeda, dengan syarat berformat *.bmp 24 bit (*true color*). Analisis dapat dilihat pada Tabel 4.3, 4.4 dan 4.5.



Tabel 4.3 Tabel Uji File Output dari 3 File Pesan Berbeda, File Image “Monalisa.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Monalisa.bmp	419454	Serbu.txt	265	MonalisaStego1.bmp	419454	Serbu-Hasil.txt	265
Monalisa.bmp	419454	Tabel4.doc	44544	MonalisaStego2.bmp	419454	Tabel4.doc	44544
Monalisa.bmp	419454	Gadis.txt	10977	MonalisaStego3.bmp	419454	Gadis-Hasil.txt	10977

Tabel 4.4 Tabel Uji File Output dari 3 File Pesan Berbeda, File Image “Lena.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Lena.bmp	786486	Serbu.txt	265	LenaStego1.bmp	786486	Serbu-Hasil.txt	265
Lena.bmp	786486	Tabel4.doc	44544	LenaStego2.bmp	786486	Tabel4.doc	44544
Lena.bmp	786486	Gadis.txt	10977	LenaStego3.bmp	786486	Gadis-Hasil.txt	10977

Tabel 4.5 Tabel Uji File Output dari 3 File Pesan Berbeda, File Image “Fruits.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Fruits.bmp	499554	Serbu.txt	265	FruitsStego1.bmp	499554	Serbu-Hasil.txt	265
Fruits.bmp	499554	Tabel4.doc	44544	FruitsStego2.bmp	499554	Tabel4.doc	44544
Fruits.bmp	499554	Gadis.txt	10977	FruitsStego3.bmp	499554	Gadis-Hasil.txt	10977

4.3.5 Analisis Penyisipan dan Ekstraksi *File image* dan *File Pesan*

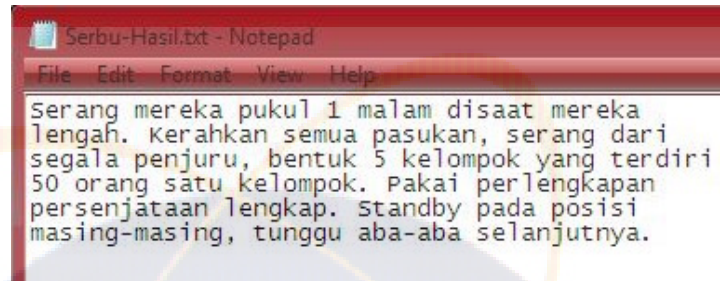
Setelah melakukan proses penyisipan dan ekstraksi *file image*, maka untuk melihat apakah hasil dari penyisipan dan ekstraksi *file image* telah berhasil seperti yang akan dilakukan langkah-langkah berikut:

1. Buka lokasi tempat *file image* yang telah disisipkan pesan rahasia.
Perhatikan *file image* yang asli dan *file image* yang telah disisipkan *file* pesan tidak ada bedanya. Jadi *user* harus mengingat letak lokasi dan nama *file image*-nya dengan pasti.
2. *File Image* yang telah disisipkan pesan rahasia masih dapat dilihat dengan baik selayaknya *file image* asli oleh user, tanpa disadari bahwa adanya *file* pesan yang telah disisipkan ke dalamnya.

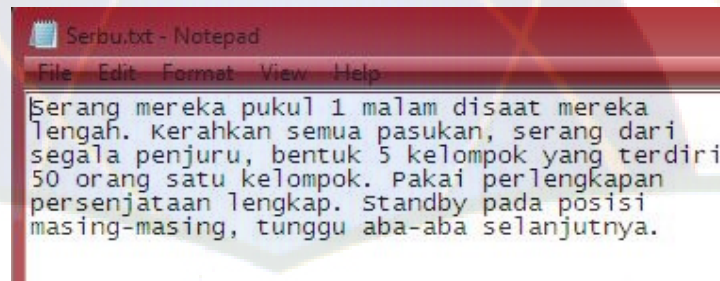


Gambar 4.9 *File Image* Sebelum dan Sesudah Disisipkan *File Pesan*

1. Untuk membuka *file* pesan yang telah diambil dari *file image*, *user* dapat langsung membacanya dengan membuka *file* pesan tersebut yang sebelumnya di-save dari aplikasi StegoImageKu.



Gambar 4.10 File Pesan Rahasia Sebelum Disisipkan ke File Image



Gambar 4.11 File Pesan yang Diambil Kembali dari File Image

Dari hasil pengamatan di atas tampak tidak adanya perubahan *file* pesan sebelum dan sesudah proses penyisipan pesan ke dalam *file* image baik bentuk maupun ukurannya.

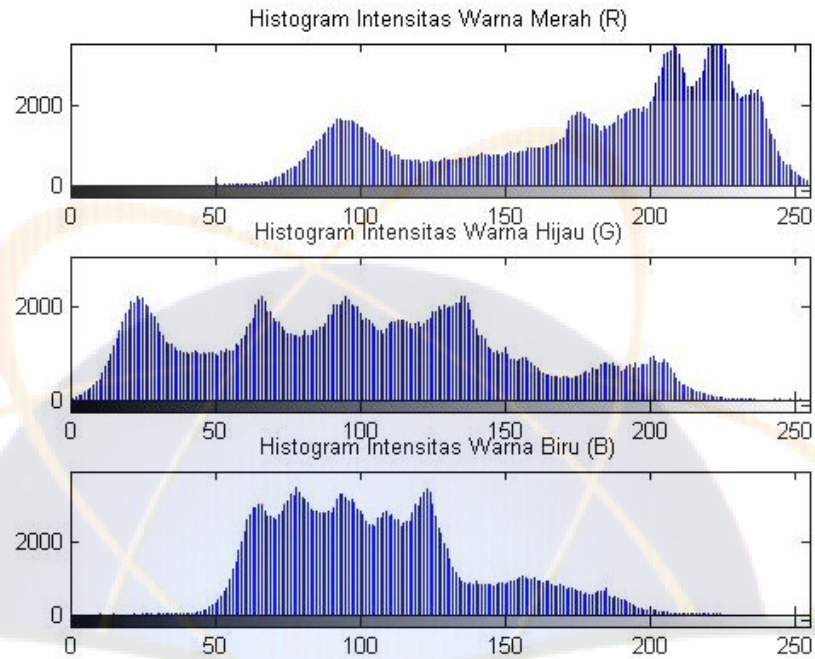
4.3.6 Analisis File Pesan Dan File Image Jika Digunakan 2 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat penurunan intensitas serta kualitas *file* image terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 2 bit terakhir pada *file* image tersebut.

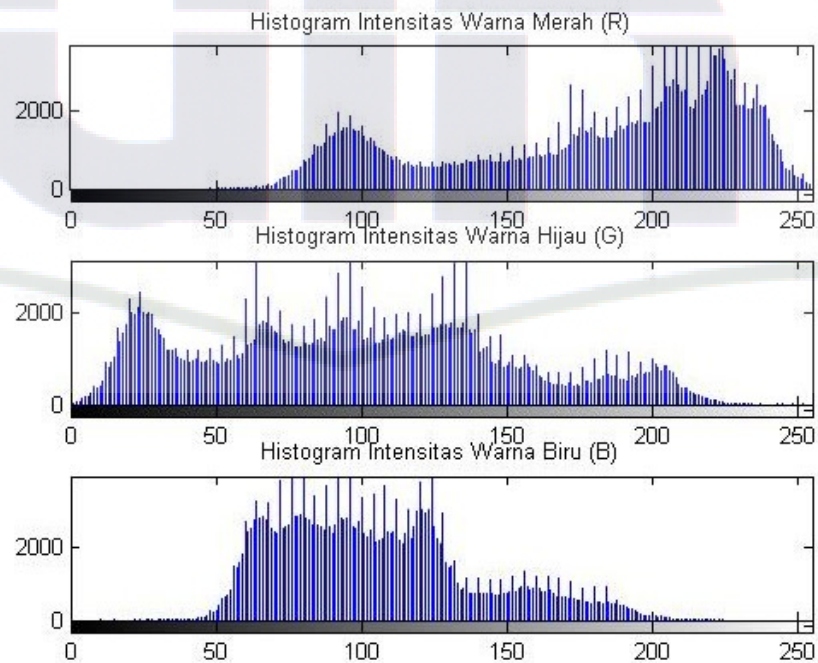


Gambar 4.12 Perbedaan Kualitas *File Image* Jika Digunakan 2 Bit Terakhir

Dari Gambar 4.12 terlihat bahwa *file image* tidak mengalami perubahan baik gambar maupun ukuran, jika menggunakan 2 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 2 bit terakhir dinyatakan berhasil. Proses steganografi menggunakan 2 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.13 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 2 Bit Terakhir.



Gambar 4.14 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 2 Bit Terakhir.

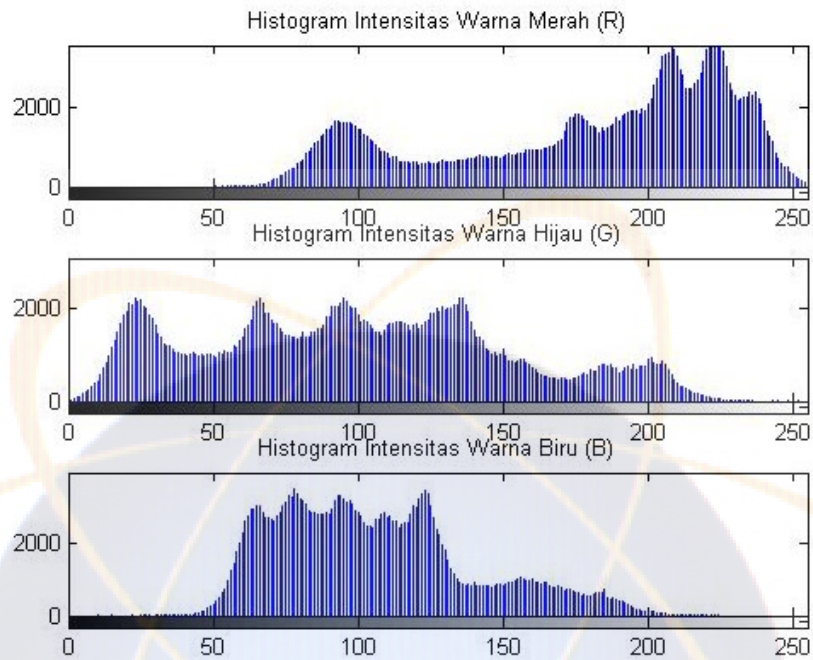
4.3.7 Analisis File Pesan Dan File Image Jika Digunakan 3 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 3 Bit terakhir pada *file image* tersebut.

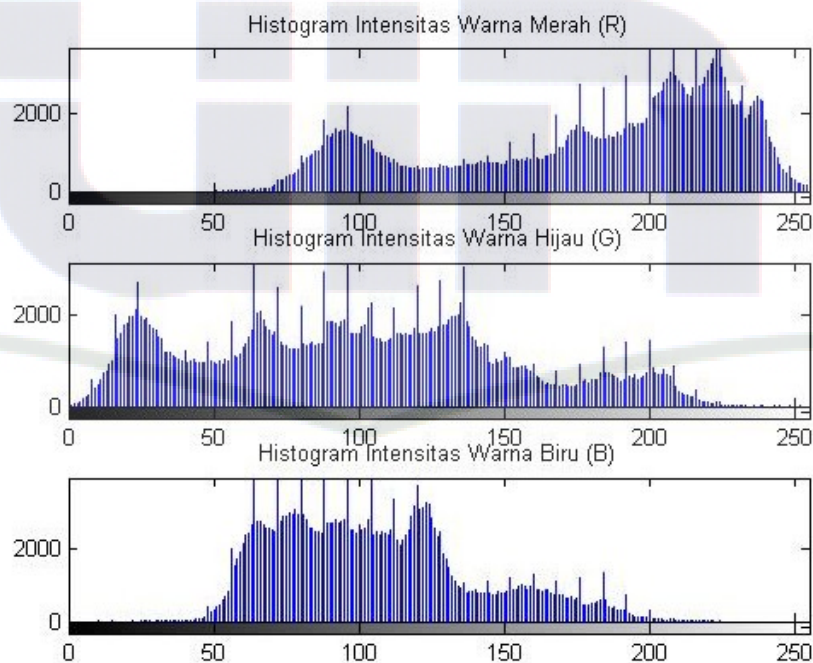


Gambar 4.15 Perbedaan Kualitas *File Image* Jika Digunakan 3 Bit Terakhir

Dari Gambar 4.15 terlihat bahwa *file image* tidak mengalami perubahan baik gambar maupun ukuran, jika menggunakan 3 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 3 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit dan 2 bit terakhir. Proses steganografi menggunakan 3 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit dan 2 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.16 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 3 Bit Terakhir



Gambar 4.17 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 3 Bit Terakhir

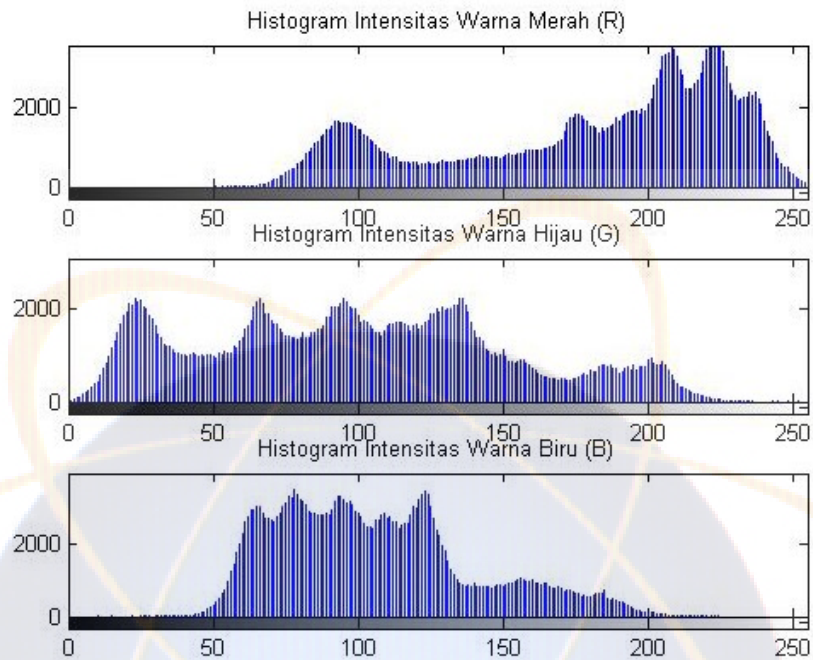
4.3.8 Analisis File Pesan Dan File Image Jika Digunakan 4 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 4 Bit terakhir pada *file image* tersebut.

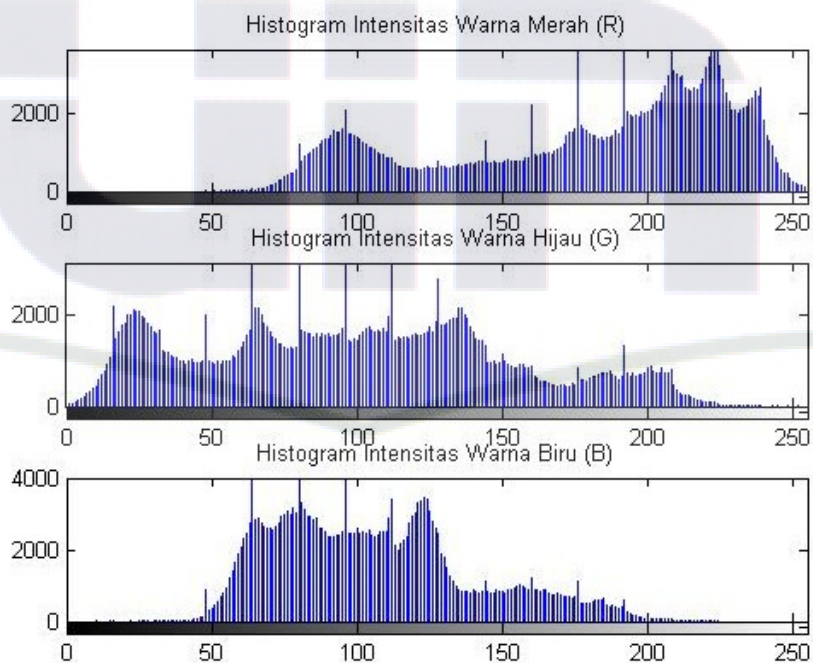


Gambar 4.18 Perbedaan Kualitas *File Image* Jika Digunakan 4 Bit Terakhir

Dari Gambar 4.18 terlihat bahwa *file image* tidak mengalami perubahan baik gambar maupun ukuran, jika menggunakan 4 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 4 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit dan 3 bit terakhir. Proses steganografi menggunakan 4 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit dan 3bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.19 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 4 Bit Terakhir.



Gambar 4.20 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 4 Bit Terakhir.

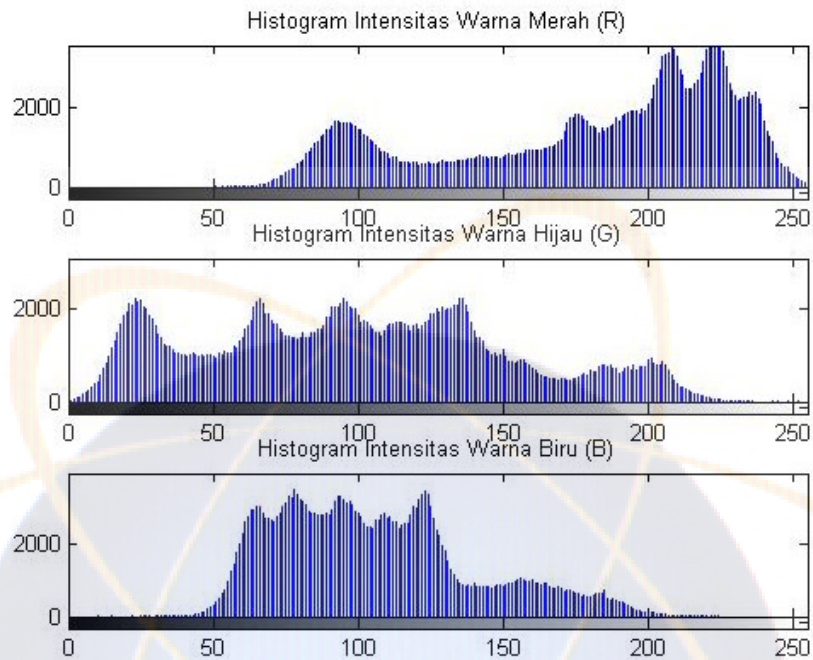
4.3.9 Analisis File Pesan Dan File Image Jika Digunakan 5 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 5 Bit terakhir pada *file image* tersebut.

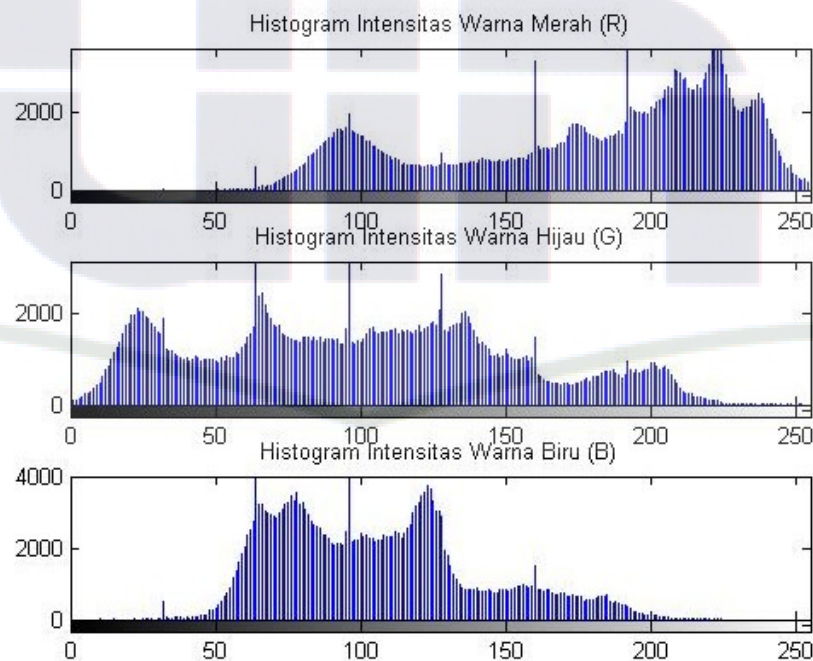


Gambar 4.21 Perbedaan Kualitas *File Image* Jika Digunakan 5 Bit Terakhir

Dari Gambar 4. terlihat bahwa *file image* tidak mengalami perubahan ukuran, jika menggunakan 5 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 5 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit, 3 bit dan 4 bit terakhir. Pada penggunaan 5 bit terakhir mulai terjadi sedikit perubahan gambar, karena terjadi penurunan intensitas warna pada 5 bit terakhir di setiap *pixel*. Proses steganografi menggunakan 5 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit, 3 bit dan 4 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.22 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 5 Bit Terakhir.



Gambar 4.23 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 5 Bit Terakhir.

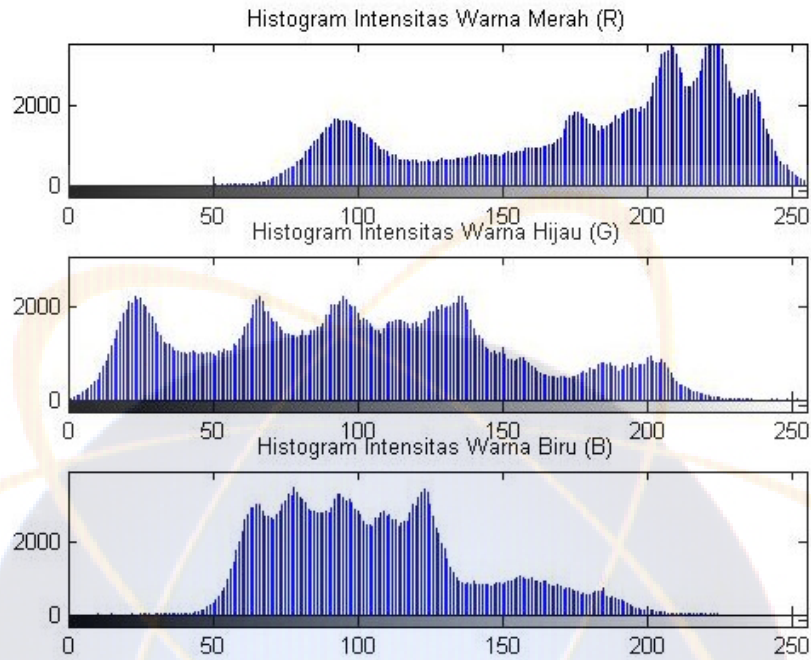
4.3.10 Analisis *File Pesan* Dan *File Image* Jika Digunakan 6 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 6 Bit terakhir pada *file image* tersebut.

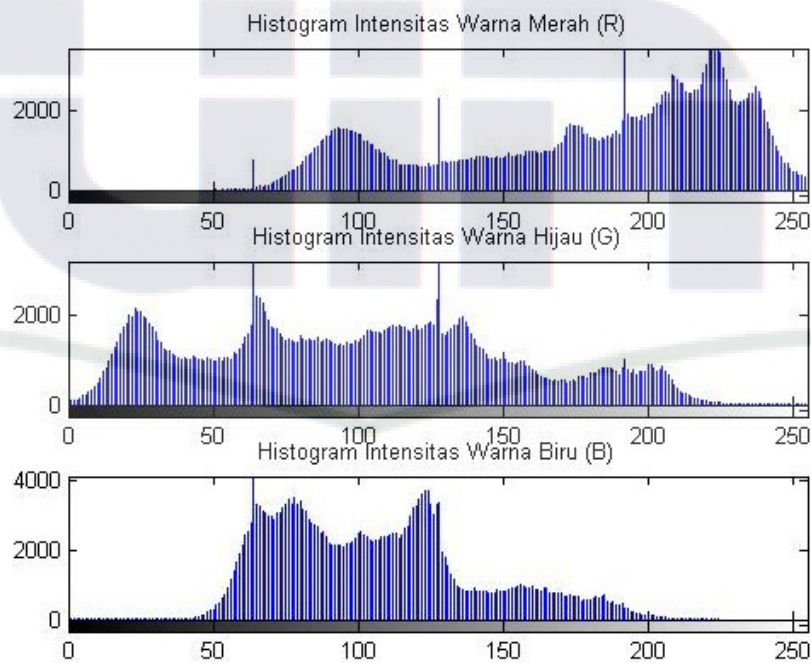


Gambar 4.24 Perbedaan Kualitas *File Image* Jika Digunakan 6 Bit Terakhir

Dari Gambar 4. terlihat bahwa *file image* tidak mengalami perubahan ukuran, jika menggunakan 6 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 6 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit, 3 bit, 4 bit dan 5 bit terakhir. Pada penggunaan 6 bit terakhir mulai terjadi perubahan gambar lebih besar dibandingkan dengan 5 bit, karena terjadi penurunan intensitas warna lebih besar dari pada 5 bit terakhir di setiap *pixel*-nya. Proses steganografi menggunakan 5 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit, 3 bit, 4 bit dan 5 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.25 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 6 Bit Terakhir.



Gambar 4.26 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 6 Bit Terakhir.

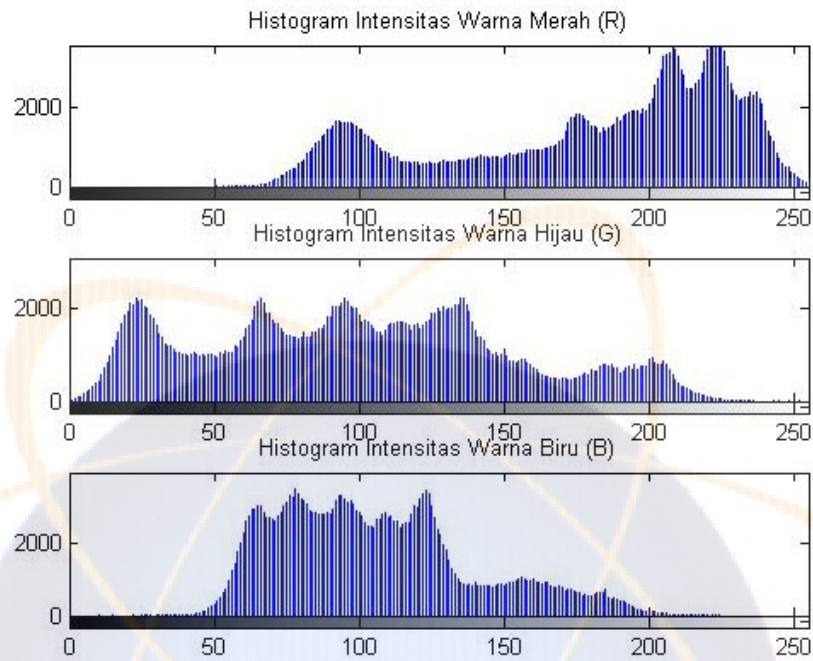
4.3.11 Analisis *File Pesan* Dan *File Image* Jika Digunakan 7 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 7 Bit terakhir pada *file image* tersebut.

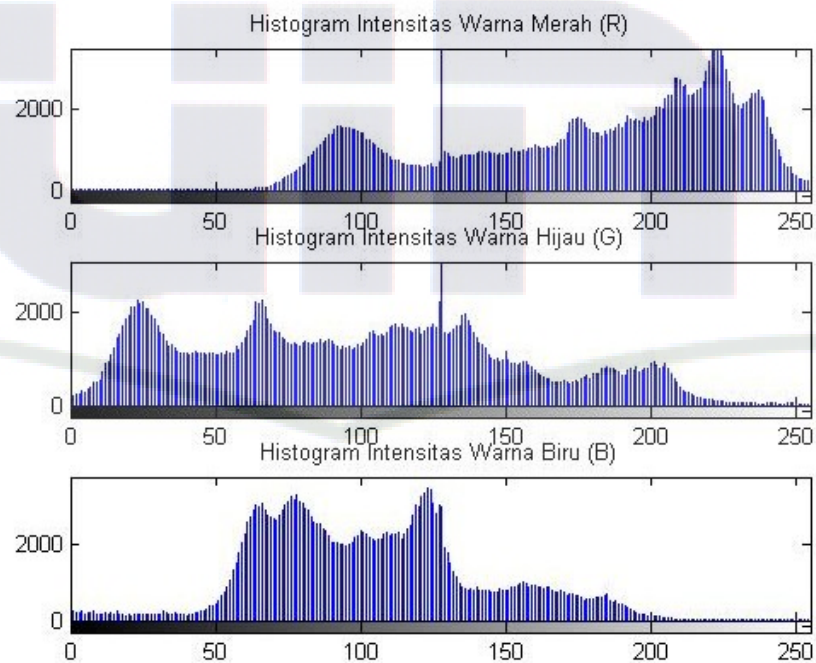


Gambar 4.27 Perbedaan Kualitas *File Image* Jika Digunakan 7 Bit Terakhir

Dari Gambar 4.27 terlihat bahwa *file image* tidak mengalami perubahan ukuran, jika menggunakan 7 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 7 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit, 3 bit, 4 bit, 5 bit dan 6 bit terakhir. Pada penggunaan 7 bit terakhir terjadi perubahan gambar lebih besar dibandingkan dengan 5 bit dan 6 bit, karena terjadi penurunan intensitas warna lebih besar dari pada 5 bit dan 6 bit terakhir di setiap *pixel*-nya. Proses steganografi menggunakan 7 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit, 3 bit, 4 bit, 5 bit dan 6 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.28 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 7 Bit Terakhir.



Gambar 4.29 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 7 Bit Terakhir.

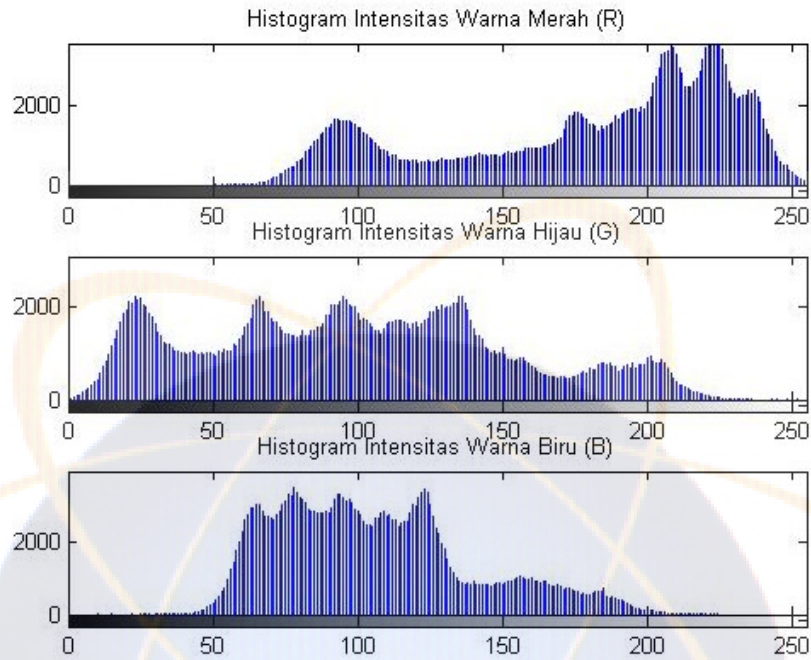
4.3.12 Analisis *File Pesan* Dan *File Image* Jika Digunakan 8 Bit Terakhir

Analisis ini dilakukan bertujuan untuk melihat kualitas *file image* terhadap *file* pesan yang akan disisipkan ke dalamnya, jika menggunakan 8 Bit terakhir pada *file image* tersebut.

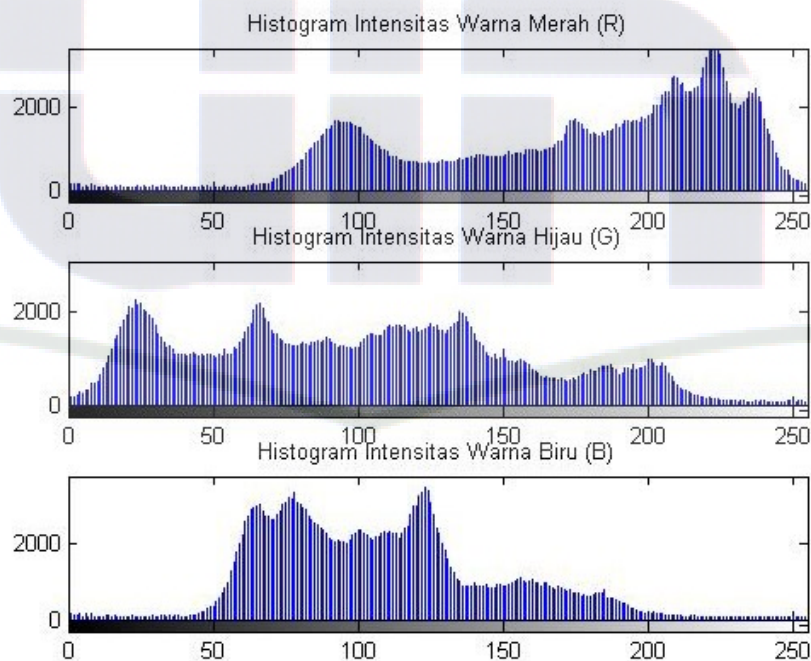


Gambar 4.30 Perbedaan Kualitas *File Image* Jika Digunakan 8 Bit Terakhir

Dari Gambar 4. terlihat bahwa *file image* tidak mengalami perubahan ukuran, jika menggunakan 8 bit terakhir pada *file image* sebagai tempat penampung *byte-byte file* pesan. Hal ini membuktikan proses steganografi menggunakan 8 bit terakhir dinyatakan berhasil, tapi terjadi penurunan intensitas warna lebih besar dibanding menggunakan 1 bit, 2 bit, 3 bit, 4 bit, 5 bit, 6 bit dan 7 bit terakhir. Pada penggunaan 8 bit terakhir terjadi perubahan gambar lebih besar dibandingkan dengan 5 bit, 6 bit dan 7 bit karena terjadi penurunan intensitas warna lebih besar dari pada 5 bit, 6 bit, dan 7 bit terakhir di setiap *pixel*-nya. Proses steganografi menggunakan 8 bit terakhir digunakan, jika *file* pesan terlalu besar sehingga proses steganografi menggunakan 1 bit, 2 bit, 3 bit, 4 bit, 5 bit, 6 bit dan 7 bit terakhir tidak mampu menampung *byte-byte file* pesan.



Gambar 4.31 Histogram Intensitas Warna Sebelum Penyisipan *File* Pesan Dengan 8 Bit Terakhir.



Gambar 4.32 Histogram Intensitas Warna Sesudah Penyisipan *File* Pesan Dengan 8 Bit Terakhir.

4.3.13 Analisis Ukuran *File* Pesan Terhadap *File Image*

Analisis ini bertujuan untuk mengetahui batasan ukuran *file* pesan yang dapat disisipkan ke dalam *file image*.

Tabel 4.6 Tabel uji Ukuran *File* Pesan Rahasia Terhadap *File Image*

File Image	Size (byte)	Secret File	Size (byte)	Status Penyisipan
Lena.bmp	786486	Serbu.txt	265	Berhasil
Lena.bmp	786486	Tabel4.doc	44544	Berhasil
Lena.bmp	786486	Bab IV.doc	323584	Tidak

Dari Table 4.3 terlihat bahwa proses penyisipan hanya dapat dilakukan apabila byte yang tersedia di dalam *file image* lebih besar dari pada jumlah *byte file* pesan. Suatu citra 24-bit jika digunakan untuk menyimpan informasi rahasia mampu menampung informasi maksimum berukuran $\frac{1}{8}$ dari ukuran citra penampung tersebut. Jadi semakin besar *file image* maka jumlah *byte* yang disediakan untuk menampung pesan rahasia akan lebih banyak.

4.4 Perbandingan Steganografi Sejenis

Penulis membandingkan dengan steganografi yang sejenis yang telah ada sebelumnya dari berbagai skripsi dari beberapa perguruan tinggi. Ini bertujuan untuk dijadikan bahan perbandingan dengan sistem yang akan penulis buat, bukan bermaksud menentukan baik atau buruknya.

Tabel 4.7 Perbandingan Steganografi Sejenis 1

Judul	Penulis	Tahun	Universitas	Perbedaan
ANALISIS DAN IMPLEMENTASI STEGANOGRAFI DAN KRIPTOGRAFI PADA FILE AUDIO	YUNIAR NURSYAMSIH SETIORINI 103091029623	2008	Universitas Islam Negeri Syarif Hidayatullah Jakarta	<ol style="list-style-type: none">1. Skripsi di atas menggunakan <i>file audio</i> sebagai media penampung <i>file</i>-nya, sedangkan penulis menggunakan Citra Digital untuk penampungan <i>file</i>-nya.2. Skripsi di atas memakai metode pengembangan sistem SDLC (<i>System Development Life Cycle</i>) sedangkan penulis memakai metode pengembangan sistem RAD (<i>Rapid Application Development</i>).3. Skripsi diatas hanya fokus terhadap 1 bit terakhir pada file audionya, sedangkan penulis mengembangkan hingga 8 bit dapat digunakan.4. Skripsi diatas memakai C# dalam pembuatan aplikasinya, sedangkan penulis menggunakan <i>Delphi</i>.

				5. Skripsi diatas hanya dapat beberapa format <i>file</i> yang dapat disisipkan, sedangkan penulis hampir semua format <i>file</i> dapat disisipkan.
--	--	--	--	--

Tabel 4.8 Perbandingan Steganografi Sejenis 2

Judul	Penulis	Tahun	Universitas	Perbedaan
PEMBUATAN APLIKASI <i>STEGANOGRAPHY</i> PADA <i>FILE AUDIO</i>	Hardi Wijaya 26402169	2007	Universitas Kristen Petra Surabaya	<ol style="list-style-type: none"> 1. Skripsi diatas memakai metode pengembangan sistem SDLC (<i>System Development Life Cycle</i>) sedangkan penulis memakai metode pengembangan sistem RAD (<i>Rapid Application Development</i>). 2. Skripsi diatas memakai Visual Basic dalam pembuatan aplikasinya, sedangkan penulis menggunakan Delphi. 3. Skripsi diatas hanya dapat beberapa format <i>file</i> yang dapat disisipkan, sedangkan penulis hampir semua format <i>file</i> dapat disisipkan. 4. Skripsi diatas menggunakan <i>file audio</i> sebagai media penampung <i>file</i>-nya, sedangkan penulis menggunakan Citra Digital untuk penampungan filenya 5. Skripsi diatas hanya fokus terhadap 1 bit terakhir pada file

				audionya, sedangkan penulis mengembangkan hingga 8 bit dapat digunakan.
--	--	--	--	---



BAB V

PENUTUP

5.1 Kesimpulan

Dari hasil pengujian sistem yang dilakukan pada bab sebelumnya, maka dapat disimpulkan ke dalam beberapa hal antara lain:

1. Aplikasi StegoImage berhasil mengimplementasikan teknik steganografi yang menggunakan algoritma LSB dalam mengamankan pesan. Hal ini dibuktikan melalui hasil uji analisa pada Tabel 4.1 bahwa *file* dapat disisipkan ke dalam *file image* dan diambil kembali dari *file image* tersebut.
2. Proses penyisipan *file* tidak terpaku pada satu format *file* tertentu saja, tapi dapat dilakukan pada *file* berformat *.txt, *.doc, *.xsl, *.ppt, *.mdb, *.pdf, *.php, *.JPG, *.html, *.wav hal ini dibuktikan oleh uji analisi pada Tabel 4.2.
3. Berdasarkan hasil uji analisis, penyisipan *file* ke dalam *file image* mempengaruhi kualitas warna pada *file image* tersebut. Dengan adanya perubahan intensitas warna antara *file image* asli dan *file image* yang sudah disisipkan pesan, hal itu dapat dilihat pada gambar-gambar histogram pada bab IV.
4. Berdasarkan hasil uji analisis pada Tabel 4.1 dapat dinyatakan bahwa *file* pesan masukan dan hasil keluaran memiliki jumlah *byte* yang sama

persis, di mana artinya penyisipan *file* pesan tidak mempengaruhi besar ukuran *file* pesan awal maupun akhir.

5. Berdasarkan hasil uji analisis keseluruhan pada bab IV dapat disimpulkan bahwa algoritma LSB tidak hanya terpaku pada 1 bit terakhir saja sebagai tempat penyembunyian data, tapi dapat dikembangkan hingga 8 bit.

5.2 Saran

1. Aplikasi StegoImageKu ini dapat diimplementasikan di instansi yang membutuhkan pengamanan *file* seperti bank, kantor pemerintahan, militer dan sebagainya.
2. Aplikasi StegoImageKu hanya *file image* sebagai media penampung, diharapkan dapat dikembangkan sehingga dapat menggunakan *file* teks, audio, video dan lain- lain sebagai media penampungnya.
3. Aplikasi StegoImage masih dikembangkan untuk perangkat *computer desktop*. Akan lebih praktis apabila dapat dikembangkan lebih lanjut untuk dapat digunakan dalam lingkungan perangkat keras *mobile* seperti telepon genggam.

DAFTAR PUSTAKA

Ahmad, Usman. (2005). *Pengolahan Citra Digital & Teknik Pemogramannya*. Yogyakarta : Graha Ilmu.

Ariyus, Doni. (2006). *Kriptografi Keamanan Data dan Komunikasi*. Yogyakarta : Graha Ilmu.

Andleigh, Prabhat K., Thakrar, Kiran. (1996). *Multimedia System Design*. New Jersey : Prentice Hall International Edition.

doank29.multiply.com/journal/item/3

haryanto.staff.gunadarma.ac.id/Downloads/files/7272/9.Steganografi.ppt

ilmucerdas.wordpress.com/2008/08/02/pengertian-multimedia/

ilmukomputer.org/2006/08/25/aplikasi-steganografi-dengan-borland-delphi/

Kenneth E. Kendall & Julie E Kendall. (2005). *System Analysis and Design*. Sixth Editon. New Jersey : Pearson Education.

Ladjamuddin, Al Bahra Bin. (2005). *Analisis dan Desain Sistem Informasi*. Jakarta : Graha Ilmu.

Malik, Jaja Jamaludin. (2006). *Kumpulan Latihan Pemograman Delphi*. Yogyakarta : Andi.

Marvin Ch, Wijaya, & Agus Priyono. (2007). *Pengolahan Citra Digital Menggunakan MatLAB Image Processing Toolbox*. Bandung: Informatika.

Munir, Rinaldi. (2006). *Kriptografi*. Bandung: Informatika.

Munir, Rinaldi. (2004). *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Bandung : Informatika.

NN. (2002). *11 Kamus Besar Bahasa Indonesia*. Jakarta : Balai Pustaka.

Prasetyo, Didik Dwi. (2004). *Aplikasi Database Client/Server Menggunakan Delphi dan MySQL*. Jakarta: Elex Media Komputindo.

Pressman, Roger S. (2002). *Rekayasa Perangkat Lunak*, Edisi 1. Yogyakarta: Andi.

webmail.informatika.org/~rinaldi/Kriptografi/2007-2008/Makalah1/MakalahIF5054-2007-A-077.pdf

Whitten, Jeffrey, dan Lonnie Bentley. (2007). *Sistem Analisis dan Metode Desain*. Edisi Kelima. Yogyakarta : Penerbit Andi.

www.ascii-code.com/

www.cert.or.id/~budi/courses/ec7010/dikmenjur/taufik-report.pdf

www.e-smartschool.com/pnk/002/PNK0020019.asp

www.informatika.org/~rinaldi/Matdis/2008-2009/Makalah2008/Makalah0809-056.pdf

www.informatika.org/~rinaldi/Kriptografi/Steganografi%20dan%20Watermarking.pdf

www.ittelkom.ac.id/library/index.php?view=article&catid=15%3Apemrosesan-

sinyal&id=344%3Acitra-digital&option=com_content&Itemid=15

www.jcatki.no-ip.org:8080/SDL_image/demos/lena.jpg

www.bobbemer.com

www.ittelkom.ac.id/library/index.php?view=article&catid=15%3Apemroses

www.tutorialgratis.net%20Tips%20Memilih%20Format%20GambarNet.htm

Hariyanto, Bambang. (2004). *Rekayasa Sistem Berorientasi Objek*. Bandung :
Informatika.





LAMPIRAN


```

program StegoImageKu;
uses
  Forms,
  Main in 'Main.pas' {MainForm},
  ChildWin in 'ChildWin.pas' {MDIChild},
  about in 'about.pas' {AboutBox},
  uProcess in 'uProcess.pas',
  uBits in 'uBits.pas',
  Unit4 in 'Unit4.pas' {Splash};
{$R *.RES}
begin
  Splash := TSplash.Create(Application); //3 baris berikut
  Splash.Show;                          //ditambah secara manual
  Splash.Update;
  while Splash.Timer1.Enabled do
    Application.ProcessMessages;
  Application.Initialize;
  Application.CreateForm(TMainForm, MainForm);
  Application.CreateForm(TAboutBox, AboutBox);
  Application.CreateForm(TSplash, Splash);
  Splash.Hide;                          //2 baris berikut ditambah secara
  Splash.Free; // menghapus form splash screen dr memory
  Application.Run;
end.

unit Unit4;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, ComCtrls, jpeg;
type
  TSplash = class(TForm)
    Timer1: TTimer;
    Image1: TImage;
    ProgressBar: TProgressBar;
    procedure Timer1Timer(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Splash: TSplash;
implementation
uses Main;
{$R *.dfm}
procedure TSplash.FormCreate(Sender: TObject);
var
  i : integer;
begin
  BorderWidth := 0;
  Show;
  Update;
  i := 7500;

```

```

ProgressBar.Max := 0;
while i <> 0 do
begin
  if i = 7500 then
  begin
    ProgressBar.Max := i;
    ProgressBar.Min := 0;
    ProgressBar.Step := 1;
  end;

  if ProgressBar.Max <> 0 then
    ProgressBar.StepIt;
    dec(i);
  end;
  update;
end;
procedure TSplash.Timer1Timer(Sender: TObject);
begin
  Timer1.Enabled:=false;
end;
end.

```

```

unit MAIN;
interface
uses Windows, SysUtils, Classes, Graphics, Forms, Controls, Menus,
  StdCtrls, Dialogs, Buttons, Messages, ExtCtrls, ComCtrls, StdActns,
  ActnList, ToolWin, ImgList;
type

```

```

TMainForm = class(TForm)
  MainMenu1: TMainMenu;
  File1: TMenuItem;
  FileNewItem: TMenuItem;
  FileOpenItem: TMenuItem;
  FileCloseItem: TMenuItem;
  Window1: TMenuItem;
  Help1: TMenuItem;
  N1: TMenuItem;
  FileExitItem: TMenuItem;
  WindowCascadeItem: TMenuItem;
  WindowTileItem: TMenuItem;
  WindowArrangeItem: TMenuItem;
  HelpAboutItem: TMenuItem;
  OpenFileDialog: TOpenDialog;
  FileSaveItem: TMenuItem;
  FileSaveAsItem: TMenuItem;
  Edit1: TMenuItem;
  CutItem: TMenuItem;
  CopyItem: TMenuItem;
  PasteItem: TMenuItem;
  WindowMinimizeItem: TMenuItem;
  StatusBar: TStatusBar;
  ActionList1: TActionList;
  EditCut1: TEditCut;
  EditCopy1: TEditCopy;
  EditPaste1: TEditPaste;
  FileNew1: TAction;
  FileSave1: TAction;
  FileExit1: TAction;
  FileOpen1: TAction;
  FileSaveAs1: TAction;
  WindowCascade1: TWindowCascade;
  WindowTileHorizontal1: TWindowTileHorizontal;
  WindowArrangeAll1: TWindowArrange;
  WindowMinimizeAll1: TWindowMinimizeAll;
  HelpAbout1: TAction;
  FileClose1: TWindowClose;
  WindowTileVertical1: TWindowTileVertical;
  WindowTileItem2: TMenuItem;

```

```

ToolBar2: TToolBar;
ToolButton1: TToolButton;
ToolButton2: TToolButton;
ToolButton3: TToolButton;
ToolButton4: TToolButton;
ToolButton5: TToolButton;
ToolButton6: TToolButton;
ToolButton9: TToolButton;
ToolButton7: TToolButton;
ToolButton8: TToolButton;
ToolButton10: TToolButton;
ToolButton11: TToolButton;
ImageList1: TImageList;
HowDoesItWork1: TMenuItem;
procedure FileNew1Execute(Sender: TObject);
procedure FileOpen1Execute(Sender: TObject);
procedure HelpAbout1Execute(Sender: TObject);
procedure FileExit1Execute(Sender: TObject);
procedure HowDoesItWork1Click(Sender: TObject);
private
{ Private declarations }
procedure CreateMDIChild(const Name: string);
public
{ Public declarations }
end;
var
MainForm: TMainForm;
implementation
{$R *.dfm}
uses CHILDWIN, about;
procedure TMainForm.CreateMDIChild(const Name: string);
var
Child: TMDIChild;
begin
{ create a new MDI child window }
Child := TMDIChild.Create(Application);
Child.Caption := Name;
//if FileExists(Name) then Child.Memo1.Lines.LoadFromFile(Name);
end;
procedure TMainForm.FileNew1Execute(Sender: TObject);
begin
CreateMDIChild(" + IntToStr(MDIChildCount + 1));
end;
procedure TMainForm.FileOpen1Execute(Sender: TObject);
begin
if OpenFileDialog.Execute then
CreateMDIChild(OpenDialog.FileName);
end;
procedure TMainForm.HelpAbout1Execute(Sender: TObject);
begin
AboutBox.ShowModal;
end;
procedure TMainForm.HowDoesItWork1Click(Sender: TObject);
begin
ShowMessage(
'In a 24-bit bitmap, each pixel is made up of (surprise, surprise) a 24-bit number. Each number is composed of three 8-bit
'+
'numbers (the R, G and B channels). These are the intensity of the Red, Green and Blue colors that create the final color of
the pixel.' + #13#10#13#10+
'To hide something inside the image, we will replace the Least Significant Bit (this is, the "rightmost" bit) of each 8-bit
channel '+
'of every pixel, with the bits of the file we want to hide. ' + #13#10#13#10 +
'The image will lose some quality because now the colors of the pixels are not the same, but it will go unnoticed to the
human eye.' + #13#10#13#10 +
'Obviously, since we are storing only 3 bits per pixel, the image must have a phenomenal size to accommodate just a tiny
little file. '+
'For example, if we want to hide 1 MB of data, we need an image with 2,796,203 pixels, which would have a size of
something like 2,200 x 1,320 pixels. And that is a 8.3 MB file... :O' + #13#10#13#10 +

```

'We could replace the 2, 3 or 4 rightmost bits of every channel in order to increase the amount of data we can hide, but the quality could decrease considerably.' + #13#10#13#10 +

'Try 8 bits per channel and you'll see what I'm talking about (amazingly, 7 normally produces intelligible -although very ugly- images).');

```
end;
procedure TMainForm.FileExit1Execute(Sender: TObject);
begin
    Close;
end;
end.
```

```
unit CHILDDWIN;
interface
uses Windows, Messages, SysUtils, Classes, Graphics, Forms, Controls, StdCtrls,
    ComCtrls, ExtCtrls, ExtDlgs, Dialogs, Buttons, jpeg;
```

```
type
    TMDIChild = class(TForm)
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    OpenPictureDialog1: TOpenPictureDialog;
    SavePictureDialog1: TSavePictureDialog;
    OpenTextFileDialog1: TOpenTextFileDialog;
    SaveTextFileDialog1: TSaveTextFileDialog;
    OpenFileDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    GroupBox3: TGroupBox;
    Edit1: TEdit;
    btnBrowse: TBitBtn;
    GroupBox4: TGroupBox;
    lblNote: TLabel;
    lblBPC: TLabel;
    udBitsPerChannel: TUpDown;
    edtBPC: TEdit;
    GroupBox5: TGroupBox;
    btnEncrypt: TBitBtn;
    btnClear: TBitBtn;
    btnOpenEncode: TBitBtn;
    pb1: TProgressBar;
    GroupBox1: TGroupBox;
    ScrollBox1: TScrollBox;
    Image1: TImage;
    GroupBox7: TGroupBox;
    BitBtn6: TBitBtn;
    btnDecrypt: TBitBtn;
    btnDecode: TBitBtn;
    GroupBox2: TGroupBox;
    ScrollBox2: TScrollBox;
    Image2: TImage;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormCreate(Sender: TObject);
    procedure btnBrowseClick(Sender: TObject);
    procedure btnEncryptClick(Sender: TObject);
    procedure btnOpenEncodeClick(Sender: TObject);
    procedure btnDecryptClick(Sender: TObject);
    procedure btnDecodeClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
    private
        { Private declarations }
    public
        { Public declarations }
    end;
    var
        MDIChild: TMDIChild;
implementation
uses MAIN, uProcess, about;
{$R *.dfm}
```

```

procedure TMDIChild.btnBrowseClick(Sender: TObject);
begin
if OpenFileDialog1.Execute then begin
    Edit1.Text := OpenFileDialog1.FileName;
end;
end;
procedure TMDIChild.btnClearClick(Sender: TObject);
begin
    image1.Picture.Bitmap:=nil;
    Edit1.Clear;
end;
procedure TMDIChild.btnDecodeClick(Sender: TObject);
var
    NamaFile : string;
begin
try
    OpenPictureDialog1.Filter := 'Bitmaps (*.bmp)|*.bmp';
    OpenPictureDialog1.DefaultExt := '*.bmp';
if OpenFileDialog1.Execute then
    begin
        NamaFile := OpenPictureDialog1.FileName;
        Image2.Picture.LoadFromFile(NamaFile);
    end;
finally
    end;
end;
procedure TMDIChild.btnDecryptClick(Sender: TObject);
begin
if SaveDialog1.Execute then
    try
        try
            btnDecrypt.Enabled:= False;
            lblBPC.Enabled:= False;
            lblNote.Enabled:= False;
            edtBPC.Enabled:= False;
            udBitsPerChannel.Enabled:= False;
            pb1.Show;
            Decrypt(OpenPictureDialog1.FileName, SaveDialog1.FileName, pb1);
            MessageBox( Application.Handle, PChar('Done!' + #13#10#13#10 + 'The file "' +
                SaveDialog1.FileName + '" was created.'),
                'StegaImage', MB_OK + MB_ICONINFORMATION);
        finally
            pb1.Hide;
            btnDecrypt.Enabled:= True;
            lblBPC.Enabled:= True;
            lblNote.Enabled:= True;
            edtBPC.Enabled:= True;
            udBitsPerChannel.Enabled:= True;
        end;
    except
        on e: Exception do
            MessageBox(Handle, PChar('An error has occurred.' + #13#10#13#10 + e.Message), 'StegaImageKu', MB_OK +
MB_ICONSTOP);
        end;
    end;
end;
procedure TMDIChild.btnEncryptClick(Sender: TObject);
begin
try
    if Trim(Edit1.Text) = '' then begin
        MessageDlg('Content harus diisi',mtWarning,[mbOK],0);
        Exit;
    end;
if SavePictureDialog1.Execute then
    try
        try
            Image1.Picture.Bitmap:=nil;
            btnEncrypt.Enabled:= False;
            lblBPC.Enabled:= False;
            lblNote.Enabled:= False;

```

```

    edtBPC.Enabled:= False;
    udBitsPerChannel.Enabled:= False;
    btnBrowse.Enabled:= False;
    btnOpenEncode.Enabled:= False;
    btnClear.Enabled:= False;
    Edit1.Clear;
    pb1.Show;
    Encrypt(OpenDialog1.FileName, OpenPictureDialog1.FileName, SavePictureDialog1.FileName,
    udBitsPerChannel.Position, pb1);
    MessageBox( Application.Handle, PChar('Done!' + #13#10#13#10 + 'The file "' +
    SavePictureDialog1.FileName + '" was created.'),
    'StegaImageKu', MB_OK + MB_ICONINFORMATION);
finally
    pb1.Hide;
    btnEncrypt.Enabled:= True;
    lblBPC.Enabled:= True;
    lblNote.Enabled:= True;
    edtBPC.Enabled:= True;
    udBitsPerChannel.Enabled:= True;
    btnBrowse.Enabled:= True;
    btnOpenEncode.Enabled:= True;
    btnClear.Enabled:= True;
end;
except
    on e: Exception do
        MessageBox(Handle, PChar('An error has occurred.' + #13#10#13#10 + e.Message), 'StegaImageKu', MB_OK +
        MB_ICONSTOP);
    end;
finally
end;
end;
end;
procedure TMDIChild.btnOpenEncodeClick(Sender: TObject);
var
    NamaFile : string;
begin
    try
        OpenPictureDialog1.Filter := 'Bitmaps (*.bmp)*.bmp';
        OpenPictureDialog1.DefaultExt := '*.bmp';
        if OpenPictureDialog1.Execute then
            begin
                NamaFile := OpenPictureDialog1.FileName;
                Image1.Picture.LoadFromFile(NamaFile);
            end;
        finally
            end;
        end;
    procedure TMDIChild.FormClose(Sender: TObject; var Action: TCloseAction);
    begin
        Action := caFree;
    end;
    procedure TMDIChild.FormCreate(Sender: TObject);
    begin
        try
        except
            on E: Exception do begin
                MessageDlg(E.Message, mtError, [mbOK], 0);
            end;
        end;
        end;
        end;
    end.

unit uProcess;
interface
uses ComCtrls, SysUtils;
procedure Encrypt(const SourceFile, SourceBitmap, Destination: string; BitsPerChannel: LongInt; ProgressBar:
TProgressBar);
procedure Decrypt(const SourceFile, DestFile: string; ProgressBar: TProgressBar);
implementation

```

```

uses Windows, Graphics, Classes, uBits, Forms;
// One byte for every channel of the RGB trio
type pRGBArray= ^TRGBArray;
  TRGBArray= array [1..3] of Byte;
procedure ProcessEncrypt(Bitmap: TBitmap; Source: TFileStream; Destination: string; BPC: LongInt; ProgressBar:
TProgressBar);
var SourceIndex, SourceSize: LongInt;
  BitIndex, PixelBitIndex: LongInt;
  SourceByte: Byte;
  PixelsRow: pRGBArray;
  RGBIndex: Integer;
  PixelsRowMax, PixelsRowIndex, CurrentRow: Integer;
// A procedure inside another is quite ugly, but we avoid passing a lot of parameters when we call it
procedure CheckNextPixel;
begin
  if (RGBIndex <= 3) and (PixelBitIndex + 1 < BPC) then // We're OK, go for the next bit
    Inc(PixelBitIndex)
  else if RGBIndex < 3 then // Switch to next RGB channel
    begin
      Inc(RGBIndex);
      PixelBitIndex:= 0;
    end
  else if RGBIndex = 3 then // Load next pixel
    begin
      PixelBitIndex:= 0;
      RGBIndex:= 1;
      if PixelsRowIndex = PixelsRowMax then // We used all pixels in this row
        begin
          Inc(CurrentRow);
          PixelsRow:= Bitmap.ScanLine[CurrentRow];
          PixelsRowIndex:= 1;
        end
      else // We still have pixels left in this row
        begin
          Inc(PixelsRowIndex);
          Inc(PixelsRow); // Increment the pointer so it points to the next pixel
        end;
    end;
end;
begin {ProcessEncrypt}
  {-- STORE THE BITS PER CHANNEL VALUE--}
  PixelsRow:= Bitmap.ScanLine[0];
  // We use 2 bits in the B channel to bring the count to 4 bits, so we can store BPC = 8, value that usually brings
  // devastation to the image quality...
  SetBitAt(PixelsRow^[1], 0, GetBitAt(BPC, 0));
  SetBitAt(PixelsRow^[1], 1, GetBitAt(BPC, 1));
  SetBitAt(PixelsRow^[2], 0, GetBitAt(BPC, 2));
  SetBitAt(PixelsRow^[3], 0, GetBitAt(BPC, 3));
  // Initialize
  PixelsRowMax:= Bitmap.Width;
  CurrentRow:= 0;
  PixelBitIndex:= 0;
  PixelsRowIndex:= 2;
  RGBIndex:= 1;
  Inc(PixelsRow);

  {-- STORE THE ACTUAL LENGTH OF THE DATA --}
  SourceSize:= Source.Size;
  for BitIndex:= 0 to SizeOf(SourceSize) * 8 - 1 do
    begin
      SetBitAt(PixelsRow^[RGBIndex], PixelBitIndex, GetBitAt(SourceSize, BitIndex));
      CheckNextPixel;
    end;
  {-- STORE THE DATA --}
  // Copy the bits from every byte in the source stream onto the bits in the pixels
  Source.Seek(0, soFromBeginning);
  for SourceIndex:= 0 to SourceSize - 1 do
    begin
      if (SourceIndex + 1) mod 10 = 0 then

```



```

begin
    ProgressBar.StepIt;
    Application.ProcessMessages;
end;
Source.Read(SourceByte, 1);
for BitIndex:= 0 to 7 do
begin
    SetBitAt(PixelsRow^[RGBIndex], PixelBitIndex, GetBitAt(SourceByte, BitIndex));
    CheckNextPixel;
end;
end; // for SourceIndex
end;

procedure Encrypt(const SourceFile, SourceBitmap, Destination: string; BitsPerChannel: LongInt; ProgressBar:
TProgressBar);
var Bitmap: TBitmap;
    sSource: TFileStream;
begin
    ProgressBar.Position:= 0;
    ProgressBar.Step:= 1;
    Bitmap:= TBitmap.Create;
    sSource:= nil;
    try
        Bitmap.LoadFromFile(SourceBitmap);
        if Bitmap.PixelFormat <> pf24bit then
            raise Exception.Create('The image must have a 24-bit depth.');
```



```

        sSource:= TFileStream.Create(SourceFile, fmOpenRead);
        if sSource.Size = 0 then
            raise Exception.Create('The source file is 0 bytes. There's nothing to hide.');
```

```

        if sSource.Size * 8 + SizeOf(LongInt) * 8 + 1 > Bitmap.Width * Bitmap.Height * 3 * BitsPerChannel then
            raise Exception.Create('The image is not big enough to accommodate the file.');
```

```

        // + 1: The BitsPerChannel value is stored in the first pixel
        // SizeOf(LongInt) * 8: we'll store the amount of actual encrypted data in the first pixels after BitsPerChannel
        ProgressBar.Max:= sSource.Size div 10;
        ProcessEncrypt(Bitmap, sSource, Destination, BitsPerChannel, ProgressBar);
        Bitmap.SaveToFile(Destination);
    finally
        Bitmap.Free;
        if Assigned(sSource) then sSource.Free;
    end;
end;

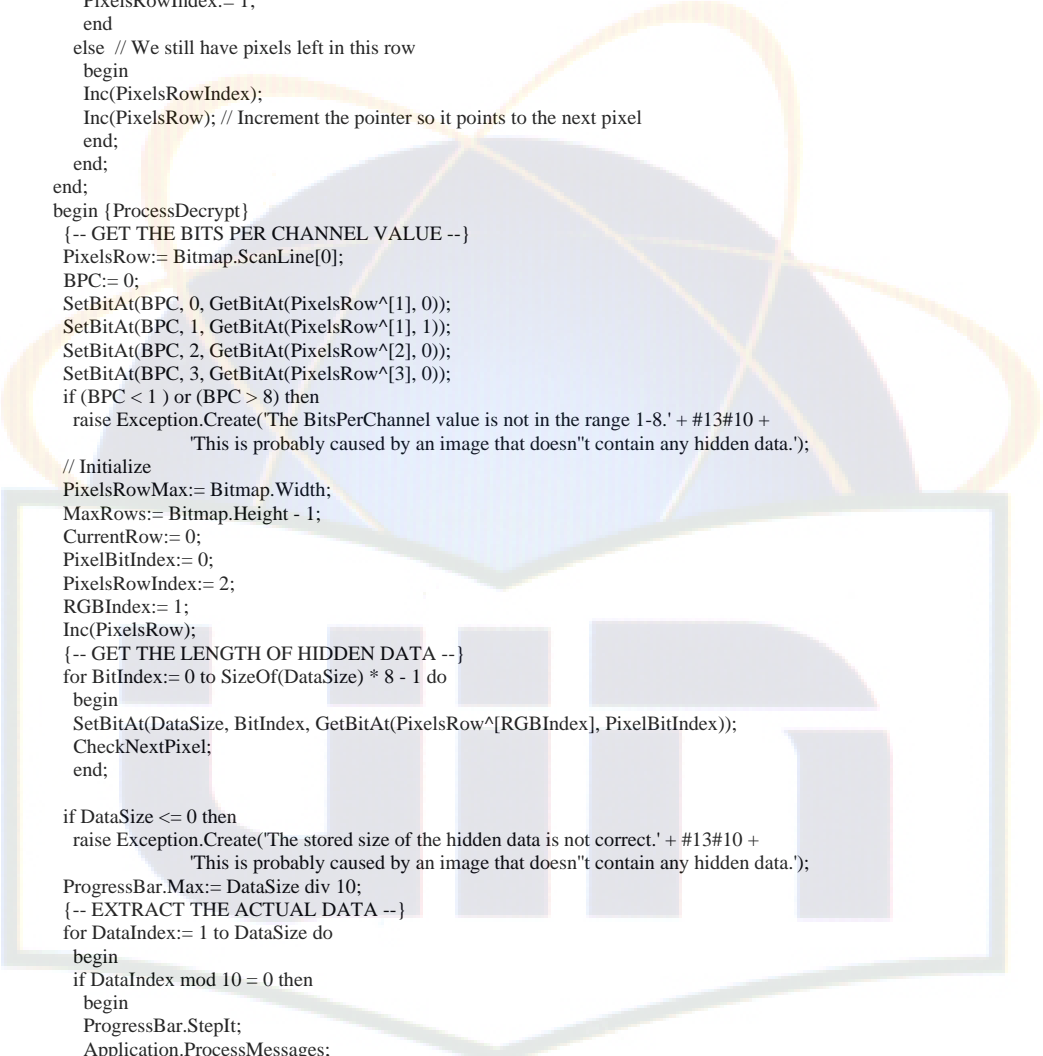
{-----}

procedure ProcessDecrypt(Bitmap: TBitmap; Destination: TFileStream; ProgressBar: TProgressBar);
var DataSize, DataIndex: LongInt;
    Data, BitIndex: Byte;
    PixelsRow: pRGBArray;
    PixelsRowMax, PixelsRowIndex, CurrentRow, MaxRows: Integer;
    PixelBitIndex: LongInt;
    RGBIndex: Integer;
    BPC: LongInt;
// A procedure inside another (and basically the same as the one in ProcessEncrypt) is quite ugly, but we avoid passing
// a lot of parameters when we call it
procedure CheckNextPixel;
begin
    if (RGBIndex <= 3) and (PixelBitIndex + 1 < BPC) then // We're OK, go for the next bit
        Inc(PixelBitIndex)
    else if RGBIndex < 3 then // Switch to next RGB channel
        begin
            Inc(RGBIndex);
            PixelBitIndex:= 0;
        end
    else if RGBIndex = 3 then // Load next pixel
        begin
            PixelBitIndex:= 0;
            RGBIndex:= 1;
            if PixelsRowIndex = PixelsRowMax then // We used all pixels in this row
                begin
                    Inc(CurrentRow);

```

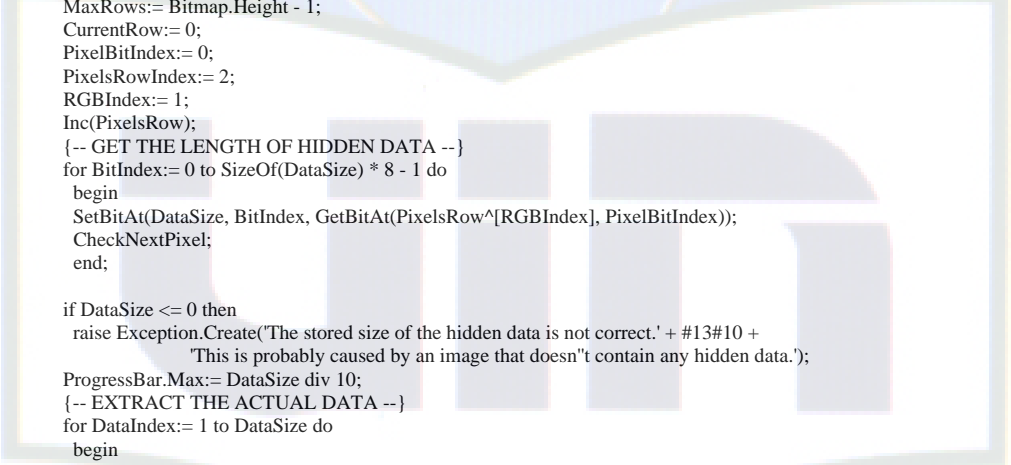
```

    if CurrentRow > MaxRows then
        raise Exception.Create('The end of the image was reached while trying to read the hidden information.' + #13#10 +
            'This is probably caused by an image that doesn''t contain any hidden data.');
```



```

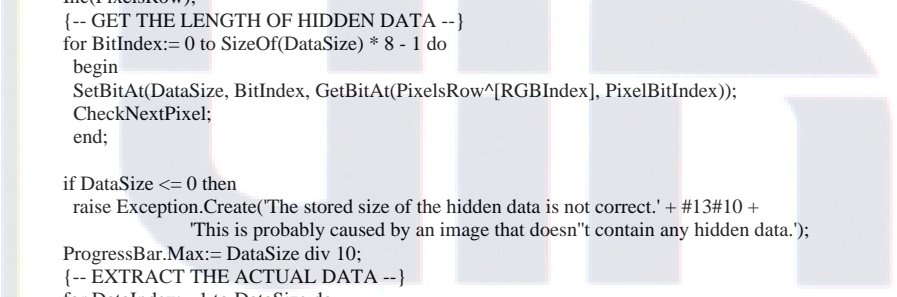
    PixelsRow:= Bitmap.ScanLine[CurrentRow];
    PixelsRowIndex:= 1;
    end
else // We still have pixels left in this row
    begin
        Inc(PixelsRowIndex);
        Inc(PixelsRow); // Increment the pointer so it points to the next pixel
    end;
end;
end;
begin {ProcessDecrypt}
    {-- GET THE BITS PER CHANNEL VALUE --}
    PixelsRow:= Bitmap.ScanLine[0];
    BPC:= 0;
    SetBitAt(BPC, 0, GetBitAt(PixelsRow^[1], 0));
    SetBitAt(BPC, 1, GetBitAt(PixelsRow^[1], 1));
    SetBitAt(BPC, 2, GetBitAt(PixelsRow^[2], 0));
    SetBitAt(BPC, 3, GetBitAt(PixelsRow^[3], 0));
    if (BPC < 1) or (BPC > 8) then
        raise Exception.Create('The BitsPerChannel value is not in the range 1-8.' + #13#10 +
            'This is probably caused by an image that doesn''t contain any hidden data.');
```



```

    // Initialize
    PixelsRowMax:= Bitmap.Width;
    MaxRows:= Bitmap.Height - 1;
    CurrentRow:= 0;
    PixelBitIndex:= 0;
    PixelsRowIndex:= 2;
    RGBIndex:= 1;
    Inc(PixelsRow);
    {-- GET THE LENGTH OF HIDDEN DATA --}
    for BitIndex:= 0 to SizeOf(DataSize) * 8 - 1 do
        begin
            SetBitAt(DataSize, BitIndex, GetBitAt(PixelsRow^[RGBIndex], PixelBitIndex));
            CheckNextPixel;
        end;

    if DataSize <= 0 then
        raise Exception.Create('The stored size of the hidden data is not correct.' + #13#10 +
            'This is probably caused by an image that doesn''t contain any hidden data.');
```



```

    ProgressBar.Max:= DataSize div 10;
    {-- EXTRACT THE ACTUAL DATA --}
    for DataIndex:= 1 to DataSize do
        begin
            if DataIndex mod 10 = 0 then
                begin
                    ProgressBar.StepIt;
                    Application.ProcessMessages;
                end;
            for BitIndex:= 0 to 7 do
                begin
                    SetBitAt(Data, BitIndex, GetBitAt(PixelsRow^[RGBIndex], PixelBitIndex));
                    CheckNextPixel;
                end;
            Destination.Write(Data, 1);
        end; //for DataIndex
    end;
procedure Decrypt(const SourceFile, DestFile: string; ProgressBar: TProgressBar);
var Bitmap: TBitmap;
    Destination: TFileStream;
begin
    ProgressBar.Position:= 0;
    ProgressBar.Step:= 1;
    Bitmap:= TBitmap.Create;
    Destination:= nil;
    try
        try
```

```

    if FileExists(DestFile) then DeleteFile(PAnsiChar(DestFile));
    Destination:= TFileStream.Create(DestFile, fmCreate);
    Bitmap.LoadFromFile(SourceFile);
    if Bitmap.PixelFormat <> pf24bit then
        raise Exception.Create("The image doesn't have a 24-bit depth. It surely hasn't been created by this program.");
    ProcessDecrypt(Bitmap, Destination, ProgressBar);
finally
    Bitmap.Free;
    if Assigned(Destination) then Destination.Free;
end;
except
    if FileExists(DestFile) then
        DeleteFile(PChar(DestFile));
    raise;
end;
end;
end.

```

```

unit uBits;
interface
procedure SetBitAt(var Variable: LongInt; Position: Byte; Value: Boolean); overload;
procedure SetBitAt(var Variable: Byte; Position: Byte; Value: Boolean); overload;
function GetBitAt(Variable: LongInt; Position: Byte): Boolean;
implementation
procedure SetBitAt(var Variable: LongInt; Position: Byte; Value: Boolean);
begin
    if Value then
        Variable:= Variable or (1 shl Position)
    else
        Variable:= Variable and ((1 shl Position) xor $FFFFFFF);
end;
procedure SetBitAt(var Variable: Byte; Position: Byte; Value: Boolean);
begin
    if Value then
        Variable:= Variable or (1 shl Position)
    else
        Variable:= Variable and ((1 shl Position) xor $FF);
end;
function GetBitAt(Variable: LongInt; Position: Byte): Boolean;
begin
    if Variable and (1 shl Position) <> 0 then
        Result:= True
    else
        Result:= False;
end;
end.

```

```

unit about;
interface
uses Windows, Classes, Graphics, Forms, Controls, StdCtrls,
    Buttons, ExtCtrls, jpeg;
type
TAboutBox = class(TForm)
    Panel1: TPanel;
    OKButton: TButton;
    ProgramIcon: TImage;
    ProductName: TLabel;
    Version: TLabel;
    Copyright: TLabel;
    Comments: TLabel;
    Label1: TLabel;
    Label2: TLabel;
private
    { Private declarations }
public
    { Public declarations }
end;

```

```
var  
  AboutBox: TAboutBox;  
implementation  
{SR *.dfm}  
end.
```





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.

Tabel 4.3 Tabel Uji *File Output* dari 3 *File* Pesan Berbeda, *File Image* “Monalisa.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Monalisa.bmp	419454	Serbu.txt	265	MonalisaStego1.bmp	419454	Serbu-Hasil.txt	265
Monalisa.bmp	419454	Tabel4.doc	44544	MonalisaStego2.bmp	419454	Tabel4.doc	44544
Monalisa.bmp	419454	Gadis.txt	10977	MonalisaStego3.bmp	419454	Gadis-Hasil.txt	10977

Tabel 4.4 Tabel Uji *File Output* dari 3 *File* Pesan Berbeda, *File Image* “Lena.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Lena.bmp	786486	Serbu.txt	265	LenaStego1.bmp	786486	Serbu-Hasil.txt	265
Lena.bmp	786486	Tabel4.doc	44544	LenaStego2.bmp	786486	Tabel4.doc	44544
Lena.bmp	786486	Gadis.txt	10977	LenaStego3.bmp	786486	Gadis-Hasil.txt	10977

Tabel 4.5 Tabel Uji *File Output* dari 3 *File* Pesan Berbeda, *File Image* “Fruits.bmp”

File Image	Size (byte)	Secret File	Size (byte)	Output	Size (byte)	Retrieved	Size (byte)
Fruits.bmp	499554	Serbu.txt	265	FruitsStego1.bmp	499554	Serbu-Hasil.txt	265
Fruits.bmp	499554	Tabel4.doc	44544	FruitsStego2.bmp	499554	Tabel4.doc	44544
Fruits.bmp	499554	Gadis.txt	10977	FruitsStego3.bmp	499554	Gadis-Hasil.txt	10977