STEGANOGRAFI PADA CITRA *DIGITAL* MENGGUNAKAN METODE *DISCRETE WAVELET TRANSFORM*

SKRIPSI

KONSENTRASI TEKNIK REKAYASA KOMPUTER

Diajukan Untuk Memenuhi Persyaratan
Memperoleh Gelar Sarjana Teknik

OLEH:
DZIKRU ROHMATUL IZA
NIM. 0810633046 - 63

KEMENTERIAN PENDIDIKAN NASIONAL
UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
JURUSAN TEKNIK ELEKTRO
MALANG
2013

PENGANTAR

Alhamdulillah, segenap puji dan syukur penulis panjatkan kepada Allah SWT. yang telah melimpahkan rahmat, hidayah, dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul "Steganografi Pada Citra *Digital* Menggunakan Metode *Discrete Wavelet Transform*" yang diajukan untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik.

Penulis ingin mengucapkan terimakasih yang sebesar-besarnya kepada berbagai pihak yang telah membantu dan mendukung dalam penyelesaian skripsi ini, yaitu :

1. Keluarga di rumah : Ibu, Bapak, Mas Ishlahul dan Adik Rizky.
2. Seluruh Dosen di TEUB.
3. Bapak Waru Djurianto ST.,MT. selaku KKDK Rekayasa Komputer TEUB.
4. Pembimbing Skripsi Bapak Ir. Muhammad Aswin, MT. dan Bapak Ali Mustofa, ST.,MT.
5. Seluruh karyawan di TEUB.
6. Dhea Candra Dewi dan keluarga.
7. Teman-teman angkatan 2008 "Concordes".
8. Irfan Tri W., Hendry Asfahany, teman-teman Candisari, dan semua pihak yang telah membantu penulisan skripsi ini yang tidak dapat disebutkan satu per satu.

Dalam penulisan skripsi ini, penulis menyadari adanya kekurangan dan ketidaksempurnaan. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun untuk kelengkapan dan kesempurnaan skripsi ini. Penulis berharap semoga skripsi ini dapat bermanfaat bagi pembaca dan dapat memberikan sumbangan pemikiran khususnya mahasiswa TEUB.

Malang, Juli 2013

Penulis

DAFTAR ISI

PENGANTAR ..... **i**

DAFTAR ISI ..... **ii**

DAFTAR TABEL ..... **vi**

DAFTAR GAMBAR ..... **viii**

DAFTAR LAMPIRAN ..... **ix**

ABSTRAK ..... **x**

BAB I **1**

PENDAHULUAN **1**

1.1 Latar Belakang. 2

1.2 Rumusan Masalah 3

1.3 Batasan Masalah 3

1.4 Tujuan Penulisan 3

1.5 Manfaat 3

1.6 Sistematika Penulisan 4

BAB II **5**

TINJAUAN PUSTAKA **5**

2.1 Citra 5

2.1.1 Citra *Digital* 5

2.1.2 Jenis-Jenis Citra *Digital* 6

2.1.3 Pembentukan*e* Citra *Digital* 7

2.1.4 Warna dan Ruang Warna 8

2.1.4.1 RGB (*Red Green Blue*) 9

2.1.4.2 YcbCr (*Luminance – Chrominance*) 9
2.2 *Discrete Wavelet Transform (DWT)* 10
2.2.1 *Domain* dalam Transformasi Sinyal 10
2.2.2 *Wavelet* 11
2.2.3 Transformasi *Wavelet* 12
2.2.4 *Discrete Wavelet Transform (DWT)* 13
2.3 Steganografi 17
2.3.1 Sejarah Steganografi 17
2.3.2 Metode Steganografi 18
2.3.3 Tujuan Steganografi 19
2.3.4 Kriteria Steganografi 19
2.3.5 *Digital Image* Steganografi 20
2.3.6 Steganografi menggunakn *DWT* 21
2.4 Kualitas Citra 21
2.4.1 *Mean Square Error* 21
2.4.2 *Peak Signal Noise to Ratio* 22

BAB III 23
METODE PENELITIAN 23
3.1 Studi Literatur 24
3.2 Analisa Kebutuhan 24
3.3 Konfigurasi Sistem 24
3.4 Langkah Kerja Sistem 25
3.4.1 Dekomposisi Citra Digital yang Akan Disisipi Pesan 25
3.4.2 Proses Penyisipan Pesan pada Citra 26

3.4.2 Ekstraksi Pesan……..…….. 26
3.5 Pengujian Sistem 27
3.6 Kesimpulan dan Saran 27

BAB IV 28
PERANCANGAN DAN IMPLEMENTASI……..... 28
4.1 Perancangan Secara Umum 28
4.1.1 Blok Sistem Diagram 28
4.1.2 Cara Kerja Aplikasi 29
4.2 Perancangan Perangkat Lunak 29
4.2.1 Perancangan Preprocess 31
4.2.2 Perancangan Proses 33
4.2.2.1 Perancangan Proses Transformasi Pixel ke YCbCr 34
4.2.2.2 Perancangan Proses DWT 35
4.2.2.3 Perancangan Filtering Koefisien DWT 36
4.2.2.4 Perancangan Penyisipan Pesan 37
4.2.2.5 Perancangan Ekstraksi Pesan 38
4.3 Implementasi Sistem 39
4.4 Lingkungan Implementasi 39
4.5. Implementasi Interface 40
4.5.1 Implementasi Antarmuka Tampilan Awal Penyisipan Pesan 40
4.5.2 Implementasi Antarmuka Tampilan Awal Ekstraksi Pesan 41
4.5.3 Implementasi Antarmukaa Penyisipan dan Ekstraksi Pesan 41

BAB V **43**

PENGUJIAN **43**

5.1 Pengujian Fungsionalitas Sistem 43

5.2 Pengujian Kompleksitas Algoritma 53

5.3 Pengujian Kualitas Citra *Stego-image* 57

5.4 Analisis Faktor Kegagalan 65

5.5 Kesimpulan Hasil Pengujian 65

BAB VI **66**

PENUTUP **66**

6.1 Kesimpulan 66

6.2 Saran 66

DAFTAR PUSTAKA **67**

LAMPIRAN **68**

DAFTAR TABEL

Tabel 5.1 Hasil Pengujian Penyisipan Pesan dalam Nilai *PSNR* 57

Tabel 5.2 Hasil Pengujian Perubahan Citra Secara Kasat Mata dan Tingkat Keberhasilan Ekstraksi Pesan 58

Tabel 5.3 Hasil Pengujian Tingkat Keberhasilan Ekstraksi Pesan 1 dan 2 59

Tabel 5.4 Hasil Pengujian Tingkat Keberhasilan Ekstraksi Pesan 3 dan 4 60

Tabel 5.5 Hasil Pengujian Perubahan Citra Secara Kasat Mata 1 62

Tabel 5.6 Hasil Pengujian Perubahan Citra Secara Kasat Mata 2 63

DAFTAR GAMBAR

BAB II

Gambar 2.1 Citra *Digital* 6

Gambar 2.2 *Binary Image* 6

Gambar 2.3 *Greyscale Image* 6

Gambar 2.4 *Color Image* 7

Gambar 2.5 (a) Tingkat Kecerahan Yang *Continue*, (b) Tingkat Kecerahan Setelah Mengalami Kuantisasi 16 Tingkatan Diskrit 8

Gambar 2.6 Ruang Warna *RGB* 9

Gambar 2.7 Ruang Warna *YCbCr* 9

Gambar 2.8 (a) Gelombang (*wave*), (b) *Wavelet* 12

Gambar 2.9 Keluarga Wavelet 13

Gambar 2.10 Dekomposisi *Wavelet* Dengan Fungsi Dasar x[n] f = 0 ~ π 16

Gambar 2.11 Diagram Sistem Steganografi 20

Gambar 2.12 Steganografi Menggunakan *Discrete Wavelet Transform* 21

BAB III

Gambar 3.1 Diagram Alir Metode Penelitian 23

Gambar 3.2 Blok Diagram Sistem Penyisipan Pesan 24

Gambar 3.3 Blok Diagram Sistem Ekstraksi Pesan 25

Gambar 3.4 Proses Dekomposisi 25

BAB IV

Gambar 4.1 Diagram Blok Sistem Secara Keseluruhan 28

Gambar 4.2 Detail Desain Aplikasi Penyisipan Pesan 30

Gambar 4.3 Detail Desain Aplikasi Ekstraksi Pesan 31

Gambar 4.4 *Flowchart Load* Citra 32
Gambar 4.5 *Flowchart Load* Pesan 33
Gambar 4.6 Flowchart Transformasi Pixel ke YCbCr 34
Gambar 4.7 Flowchart Proses *DWT* 35
Gambar 4.8 *Flowchart Filtering* Koefisien *DWT* 36
Gambar 4.9 *Flowchart* Penyisipan Pesan 37
Gambar 4.10 *Flowchart* Ekstraksi Pesan 38
Gambar 4.11 Tampilan Awal Penyisipan Pesan 40
Gambar 4.12 Tampilan Awal Ekstraksi Pesan 41
Gambar 4.13 Tampilan Antarmuka Penyisipan Pesan 42
Gambar 4.!4 Tampilan Antarmuka Ekstraksi PEsan 42

BAB V

Gambar 5.1 Proses Load Cover-Image 44
Gambar 5.2 Proses Load Pesan 45
Gambar 5.3 Preview Cover-image dan Pesan 46
Gambar 5.4 Proses Penyisipan Pesan 47
Gambar 5.5 Proses Simpan File Stego-image 48
Gambar 5.6 Proses Load Citra Stego-image 49
Gambar 5.7 Preview Citra 50
Gambar 5.8 Proses Ekstraksi Pesan 51
Gambar 5.9 Proses Simpan Pesan 52

DAFTAR LAMPIRAN

Lampiran 1. Proses DWT 68
Lampiran 2. Proses Filtering..... 69
Lampiran 3. Proses Penyisipan Pesan..... 72
Lampiran 4. Proses Ekstraksi Pesan 74
Lampiran 5. Proses IDWT 76

ABSTRAK

DZIKRU ROHMATUL IZA, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Brawijaya, 2013, "***Steganografi Pada Citra Digital Menggunakan Metode Discrete Wavelet Transform***", Dosen Pembimbing: **Ir. Muhammad Aswin, MT**. dan **Ali Mustofa,ST.,MT.**

Kemudahan pertukaran informasi melalui internet memiliki sisi positif dan negatif. Sisi positif dari kemudahan perukaran pesan adalah dengan cepatnya seseorang mengetahui berbagai informasi yang ada di dunia. Sedangkan sisi negatifnya adalah jika tidak ada sistem keamanan yang baik, saat pesan yang ingin disampaikan bersifat rahasia, maka akan sangat mudah bagi orang lain yang tidak berkepentingan mengetahui pesan yang disampaikan.

Steganografi merupakan salah satu solusi untuk melindungi pesan yang bersifat rahasia yang disampaikan melalui internet. Untuk melakukan steganografi, media digital berupa citra merupakan *cover-object* yang baik, karena banyak dijumpai di internet dan banyak sekali pertukaran citra *digital* melalui internet, sehingga tidak mengundang kecurigaan akan adanya pesan rahasia yang disispkan . *Discrete Wavelet Transform* (*DWT*) merupakan salah satu metode yang dapat digunakan dalam teknik steganografi dalam *domain transform*. Resolusi dari *cover-image* dan ukuran pesan yang disisipkan sangat mempengaruhi kualitas citra *stego-image*.

Kata Kunci : Steganografi, *Discrete Wavelet Transform*, Citra *Digital*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Informasi merupakan sebuah kebutuhan bagi manusia dalam kehidupan sehari-hari. Semakin hari kebutuhan manusia akan informasi terus meningkat. Seiring dengan meningkatnya kebutuhan tersebut, media penyalur informasi menjadi sangat beragam diantaranya media cetak, media elektronik, dan media *digital.*

Pesatnya perkembangan teknologi informasi menyebabkan penggunaan media *digital* semakin banyak digunakan. Penyampaian informasi melalui media *digital* dipilih karena efisiensi waktu pengiriman yang sangat cepat dan penggunaannya yang semakin mudah. Permasalahan muncul ketika seseorang ingin mengirimkan informasi yang bersifat rahasia, namun keamanan pada media internet sangatlah minim, sehingga informasi yang ingin disampaikan rentan sekali teradap pencurian.

Untuk mengatasi permasalahan keamanan tersebut dapat menggunakan teknik *kriptografi*, yaitu teknik pengenkripsian pesan. Mengubah isi informasi asli (*plaintext*) menjadi sebuah informasi acak (*chipertext*). Namun teknik *kriptografi* ini dapat menimbulkan kecurigaan karena pesan acak yang dikirimkan tidak memiliki makna secara kasat mata, sehingga pihak lain yang merasa curiga dapat merusak pesan sehingga penerima tidak akan memperoleh pesan asli yang dimaksudkan.

Untuk menjawab masalah dari teknik *kriptografi* tersebut, digunakan teknik penyembunyian pesan yang lain, yaitu steganografi. Teknik steganografi tidak mengubah pesan rahasia yang ingin dikirimkan, melainkan menyisipkan pada media lain (*cover object*) yang umum digunakan dalam kehidupan sehari-hari. Pesan yang dikirimkan melalui media baru yang telah dsisipi pesan rahasia (*stego-object*) tidak akan mengundang kecurigaan orang lain, karena perbedaannya tidak dapat dilihat secara kasat mata.

Media yang paling mudah dimanfaatkan untuk steganografi adalah berkas-berkas multimedia. Berkas multimedia sangat umum ditemui di dunia internet sehingga orang tidak akan menyadari jika ada pesan rahasia yang disisipkan pada

berkas tersebut. Berkas multimedia memiliki bermacam-macam jenis, akan tetapi yang paling sering kita jumpai adalah berkas dalam bentuk citra.

Salah satu jenis media yang sering kita jumpai dalam dunia internet adalah file citra digital dengan format *bitmap*. Format ini sangat sering digunakan karena memiliki kualitas yang baik karena dapat menampilkan gambar yang bersifat lebih natural baik dalam bentuk dan warna.

Discrete Wavelet Transform (DWT) merupakan salah satu metode yang banyak digunakan untuk memanipulasi data pada *domain transform*. *DWT* membagi sebuah dimensi sinyal menjadi dua bagian, biasanya bagian dengan frekuensi tinggi dan frekuensi rendah, yang disebut dengan dekomposisi. Sebuah sinyal dilewatkan melalui *highpass filter* untuk menganalisis frekuensi tinggi, dan dilewatkan melalui *lowpass filter* untuk menganalisis frekuensi rendah.

Keluaran dari *highpass filter* dan *lowpass filter* ini menghasilkan koefisien *DWT*, dengan menggunakan koefisien ini citra asli dapat direkonstruksi. Proses rekonstruksi ini disebut *Iverse Discrete Wavelet Transform* (IDWT). Secara umum penyisipan pesan ke dalam citra dilakukan dengan cara membandingkan koefisien *DWT* dari dekomposisi citra, dimana koefisien yang memiliki nilai terbesar adalah tempat yang paling signifikan untuk menyisipkan. Dengan melalui proses tersebut pihak lain yang tidak berkepentingan akan sulit untuk mengetahui adanya pesan yang disisipkan dan memecahkan pesan rahasia tersebut.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan secara jelas diatas, maka rumusan masalah ditekankan pada:

1. Bagaimana cara melakukan penyembunyian pesan ke dalam citra *digital* menunakan *Discrete Wavelet Transform (DWT)*.
2. Bagaimana cara melakukan ekstrasi pesan tersembunyi pada sebuah citra *digital* yang telah disisipkan pesan.
3. Bagaimana membandingkan kualitas citra *digital* yang telah tersisipi pesan rahasia dengan citra digital yang belum tersisipi pesan rahasia.

1.3 Batasan Masalah

Beberapa hal yang menjadi batasan masalah dalam pembuatan program ini antara lain:

1. Perangkat lunak tidak menangani manipilasi citra setelah citra tersebut disisipkan pesan.
2. Media yang digunakan sebagai media penyisipan pesan adalah citra *digital* dalam format *bitmap*.
3. Informasi yang disisipkan berupa text.

1.4 Tujuan Penulisan

Tujuan yang ingin dicapai dari pelaksanaan tugas akhir ini adalah :

1. Memahami teknik penyembunyian pesan pada citra *digital* menggunakan *Discrete Wavelet Transform (DWT)*.
2. Memahami teknik ekstraksi pesan tersembunyi pada sebuah citra *digital* yang telah disisipkan pesan.
3. Melakukan pengujian terhadap perbandingan kualitas antara citra *digital* yang telah disisipi pesan rahasia dengan citra *digital* yang belum tersisipi pesan rahasia.

1.5 Manfaat

Diharapkan manfaat yang dapat diperoleh melalui pengerjaan skripsi ini adalah :

a) Bagi Penyusun

1. Diharapkan dapat merancang aplikasi steganografi pada citra *digital* menggunakan *Discrete Wavelet Transform (DWT)*. menggunakan bahasa pemrograman C#.
2. Memperoleh pemahaman mengenai kelebihan serta kekurangan perangkat lunak yang telah diperbuat.
3. Menambah wawasan ilmu pengetahuan yang telah dipelajari sebelumnya, serta sebaai pelatuhan berpikir kritis dalam menyelesaikan masalah yang dihadapi.

b) Bagi Pengguna

1. Memberikan kemudahan pengguna dalam melakukan penyembunyian pesan rahasia pada citra, agar selanjutnya memiliki tingkat keamanan yang tinggi akan pesan yang ingin disampaikan.

1.6 Sistematika Penulisan

Sistematika penulisan laporan skripsi ini adalah sebagai berikut :

BAB I Pendahuluan

Menjelaskan latar belakang, rumusan masalah, tujuan dan sistematika penulisan skripsi

BAB II Dasar Teori

Menjelaskan kajian pustaka dan dasar teori yang digunakan

BAB III Metodologi

Menjelaskan metode yang digunakan dalam pengerjaan skripsi

BAB IV Perancangan dan Implementasi

Menjelaskan langkah langkah perancangan perangkat lunak beserta implementasinya.

BAB V Pengujian

Menjelaskan langkah-langkah pengujian dari perangkat lunak yang telah dibuat dan membahas hasil pengujiannya.

BAB VI Kesimpulan dan Saran

Berisi Kesimpulan dan Saran

BAB II
TINJAUAN PUSTAKA

2.1 Citra

Citra adalah gambar pada bidang dwimatra (dua dimensi). Ditinjau dari sudut pandang matematis, citra merupaka fungsi menerus (*continue*) dari intensitas cahaya pada bidang dwimatra. Sumber cahaya menerangi objek, kemudian objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pantulan cahaya ini ditangkap oleh alat-alat optik, misalnya mata pada manusia, kamera, pemindai (*scanner*), dan sebagainya, sehingga bayangan objek yang disebut citra tersebut terekam.

Citra dibagi menjadi dua jenis, yaitu :

1. Citra diam (*still image*) yaitu citra tunggal yang tidak bergerak.
2. Citra bergerak (*moving image*) yaitu rangkaian dari citra diam yang ditampilkan secara berurutan sehingga menghasilkan kesan pada mata sebagai gambar bergerak.

Citra yang dimaksud dalam keseluruhan tugas akhir ini adalah citra diam (*still image*). Selanjutnya citra diam cukup disebut sebagai citra.

2.1.1 Citra *Digital*

Citra *digital* adalah suatu matriks yang terdiri dari baris dan kolom dimana setiap pasang indeks baris dan kolom menyatakan suatu titik pada citra. Nilai dari setiap matriks menyatakan nilai kecerahan titik tersebut. Titik-titik tersebut dinamakan sebagai elemen citra atau piksel.

Citra *digital* dapat di definisikan sebagai fungsi dua variabel, *f(x,y)*, dimana *x* dan *y* adalah koordinat spasial dan nilai *f(x,y)* adalah intensitas citra pada koordinat tersebut. Teknologi dasar untuk menciptakan dan menampilkan warna pada citra *digital* berdasarkan penelitian bahwa sebuah warna merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau, dan biru *(Red, Green, Blue –RGB)*

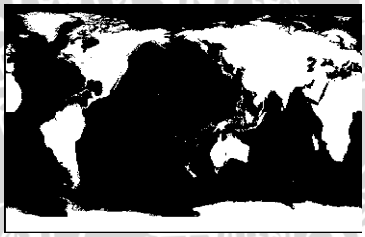
Gambar 2.1 Citra *Digital*

2.1.2 Jenia-Jenis Citra *Digital*

Citra *digital* dapat dikelompokkan menjadi tiga macam, yaitu :

1. *Binary Image*

   Binary image merupakan citra *digital* yang hanya memiliki dua kemungkinan nilai pada setiap piksel yang diwakilinya, yang pada umumnya 0 untuk menyatakan warna hitam dan 1 untuk menyatakan warna putih.



Gambar 2.2 *Binary image*

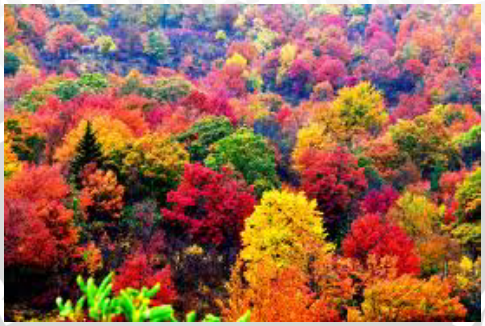
2. *Grayscale Image*

   Grayscale image adalah citra yang memiliki warna abu-abu, hitam, dan putih. Pada umumnya *grayscale* image ini memiliki 8 bit warna. *Grayscale* image ini lebih dikenal sebagai gambar hitam putih.

Gambar 2.3 *Greyscale Image*

3. *Color Image*

Color image adalah citra digital yang mengandung informasi warna pada setiap pikselnya.



Gambar 2.4 *Color Image*

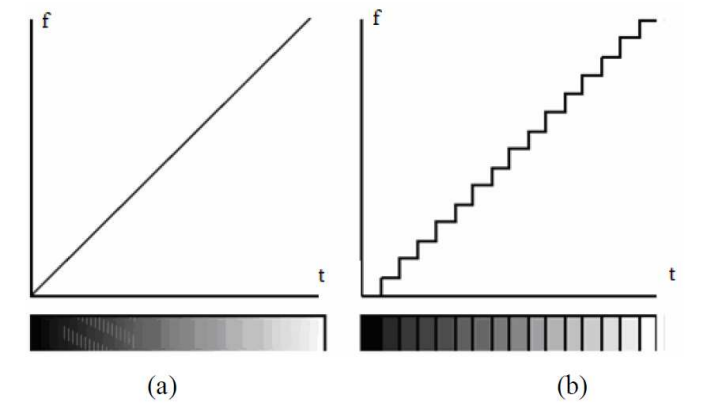
2.1.3 Pembentukan Citra *Digital*

Komputer merupakan alat yang beroperasi dalam sistem *digital* yang menggunakan *bit* atau *byte* dalam pengukuran datanya, dan yang terpenting dalam sistem *digital* adalah sifatnya yang diskrit, bukan *continue*. Hal ini berlawanan dengan citra digital yang merupakan representasi citra asal yang bersifat *continue*. Untuk mengubah citra yang bersifat *continue* ke dalam bentuk digital diperkukan sebuah cara, dalam hal ini komputer menggunakan sistem bilangan biner.

Dengan menggunakan sistem biner, citra dapat diproses dalam komputer dengan mengekstrak informasi citra analog asli dan mengirimkannya ke komputer dalam bentuk biner. Proses ini disebut dengan *digitasi* yang dapat dilakukan oleh alat seperti kamera *digital* dan *scanner*. Kedua alat ini selain dapat mengambil citra, juga dapat bertindak seagai alat *input* (masukan) bagi komputer. Alat penangkap citra *digital* ini dapat menyediakan aliran data biner bagi komputer yang didapatkan dari pembacaan kecerahan pada sebuah citra asli dalam interval sumbu *x* dan sumbu *y*.

Citra *digital* merupakan citra yang tersusun dari piksel diskrit dari tingkat kecerahan dan warna yang telah terkuantisasi. Pada dasarnya citra yang memiliki warna dan tingkat kecerahan yang *continue* perlu diubah dalam bentuk informasi warna, tingkat kecerahan, dan sebagainya yang bersifat diskrit untuk dapat menjadi sebuah citra *digital*. Pada Gambar dibawah ini akan diperlihatkan kurva tingkat

kecerahan yang *continue* dengan nilai hitam dan putih yang tidak terbatas (a) dan kurva tingkat kecerahan setelah mengalami kuantisasi dalam 16 tingkatan diskrit (b).



Gambar 2.5 (a) Tingkat kecerahan yang *continue*, (b) tingkatan kecerahan setelah mengalami kuantisasi 16 tingkatan diskrit.

Tingkatan kecerahan pada Gambar 2.5 (a) yang bersifat *continue* dapat diubah menjadi tingkat kecerahan seperti Gambar 2.5 (b), sumbu *f* merupakan ukuran frekuensi, dan sumbu t merupakan waktu**,** dengan pembacaan tingkat kecerahan menggunakan interval tertentu pada sumbu *x* dan *y* seperti yang telah disebutkan di atas. Pembagian seperti pada bagian tingkat kecerahan ini juga berlaku untuk warna agar nilai warna dapat menjadi diskrit.

2.1.4 Warna dan Ruang Warna

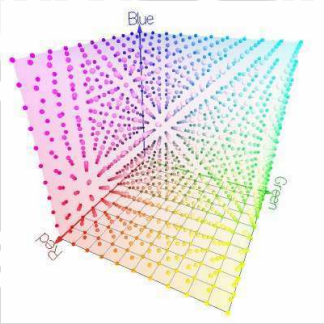
Warna merupakan hasil persepsi dari warna cahaya dalam *spectrum* wilayah yang terlihat oleh retina mata, dengan panjang gelombang antara 400n/m sampai dengan 700n/m.

Ruang warna atau yang sering juga disebut sebagai model warna merupakan sebuah cara atau metode untuk menentukan, membuat, dan memvisualisasikan warna. Dalam skripsi ini, penulis akan membahas dua ruang warna yang akan digunakan untuk aplikasi steganografi. Dua ruang warna tersebut adalah sebagai berikut (Rafael C. Gonzalez, 2002) :

1. *RGB* (*Red, Green, Blue*)
2. *YCbCr* (*Luminance – Chrominance*)

2.1.4.1 *RGB* (*Red Green Blue*)

Citra berwarna umumnya memiliki ruang warna *RGB*. Ruang warna *RGB* dapat divisualisasikan sebagai kubus, dengan tiga sumbunya yang mewakili komponen warna merah (*red*) R, hijau (*green*) G, dan biru (*blue*) B.



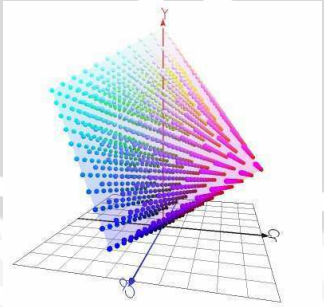
Gambar 2.6. Ruang Warna *RGB*

(http://www.couleur.org/, 2009)

RGB sering digunakan dalam sebagian besar aplikasi komputer, karena dengan ruang warna ini tidak diperlukan transformasi untuk menampilkan informasi pada layar monitor. Hal tersebut juga menyababkan *RGB* banyak dimanfaatkan sebagai ruang warna dasar bagi sebagian besar aplikasi.

2.1.4.2 *YCbCr* (*Luminance – Chrominance*)

YCbCr merupakan standart internasional bagi pengkodean digital gambar televisi yang didefinisikan di *CCIR Recommended* 601. Y merupakan komponen *luminance*, Cb dan Cr adalah komponen *chrominance*.



Gambar 2.7. Ruang Warna *YCbCr*

(http://www.couleur.org/, 2009)

YCbCr dapat diperoleh dari *RGB* dengan menggunakan persamaan sebagai berikut :

$$Y = 0.229R + 0.587G + 0.114B \quad (2.1)$$

$$Cb = -0.1687R – 0.3313G + 0.5B + 128 \quad (2.2)$$

$$Cr = 0.5R – 0.418 – 0.0813B + 128 \quad (2.3)$$

Sedangkan untuk konversi *YCbCr* kedalam *RGB* dapat diperoleh dengan menggunakan persamaan berikut :

$$R = Y + 1.402 (Cr – 128) \quad (2.4)$$

$$G = Y – 0.34414 (Cb – 128) – 0.71414 (Cr – 128) \quad (2.5)$$

$$B = Y + 1.772 (Cb – 128) \quad (2.6)$$

2.2 Discrete Wavelet Transform

Transformasi *wavelet* adalah transformasi matematika yang digunakan untuk menganalisis sinyal bergerak. Sinyal bergerak ini dianalisis unruk didapatkan informasi spektrum frekuensi dan waktunya secara bersamaan. Salah satu jenis pengembangan transformasi *wavelet* adalah *Discrete Wavelet Transform (DWT).*

2.2.1 *Domain* dalam Transformasi Sinyal

Bentuk mentah dari penggambaran waktu dan amplitudo disebut dengan sinyal. Penggambaran dengan waktu dan amplitudo yang dikategorikan dalam *domain* waktu sering kali perlu ditransformasikan dalam domain lain untuk analisis dan pemrosesan sinyal. *Domain* lain selain domain waktu misalnya domain frekuensi dan domain waktu-frekuensi. Dengan adanya transformasi sinyal ini maka informasi yang kemungkinan masih tersimpan di dalam sinyal asal dapat diidentifikasi. Informasi di dalam sinyal ini dapat ditampilkan melalui transformasi dengan cara mendapatkan spektrumnya. Spektrum yang diperoleh dari sebuah sinyal dapat berupa frekuensi atau waktu tergantung dari jenis transformasi yang digunakan.

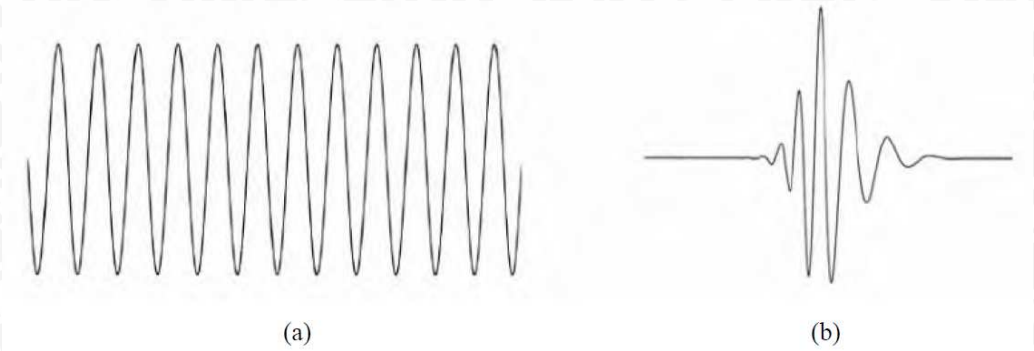
Sinyal sendiri dibagi dua jenis, yaitu sinyal tidak bergerak (*stationary signals*) dan sinyal bergerak (*non-stationary signals*). Citra dan suara merupakan salah satu contoh dari sinyal yang dapat bergerak. Contoh lain dari jenis sinyal bergerak adalah sinyal dalam bidang biologi seperti *electrodiogram* dan *electromyograpgy*.

Untuk mendapatkan informasi dari sinyal tidak bergerak, khususnya sinyal dengan representasi frekuensi, dapat digunakan transformasi *Fourier*. Karena sinyal ini tidak bergerak, maka hanya perlu mendapatkan spektrum frekuensi sebuah sinyal saja, agar informasi dari sinyal tersebut ditampilkan.

Berbeda dengan sinyal tidak bergerak, untuk mendapatkan informasi dari sinyal bergerak perlu sebuah transformasi yang bisa mendapatkan spektrum frekuensi dengan keterangan waktunya. Dalam transformasi *Fourier*, spektrum frekuensi dari sebuah sinyal bisa didapatkan, namun transformasi ini tidak member tahu kapan terjadinya frekuensi sinyal tersebut. Sehingga transformasi *Fourier* hanya cocok untuk jenis sinyal tidak bergerak. Untuk itulah diperlukan transformasi lain untuk menampilkan informasi dari jenis sinyal bergerak ini, transformasi *wavelet* adalah salah satunya. Transformasi ini bisa mendapatkan spektrum frekuensi dan waktu secara bersamaan, sehingga sinyal bergerak khususnya sinyal dengan representasi waktu-frekuensi bisa diproses menggunakan transformasi ini.

2.2.2 *Wavelet*

Gelombang (*wave*) adalah sebuah fungsi yang bergerak naik turun ruang dan waktu secara periodik. Sedangkan *wavelet* merupakan gelombang yang dibatasi atau terlokalisasi, atau dapat dikatakan sebagai gelombang pendek. *Wavelet* ini menkonsentrasikan energinya dalam ruang dan waktu sehingga cocok untuk menganalisis sinyal yang sifatnya sementara saja.



Gambar 2.8. (a) Gelombang (*wave*), (b) *Wavelet*

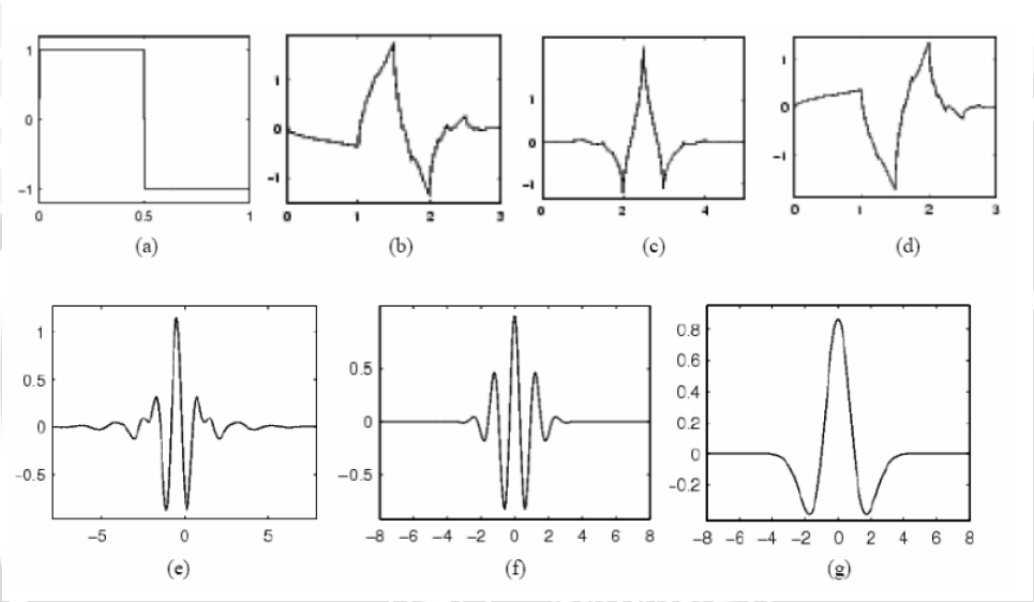
Wavelet pertama kali digunakan dalam analisis dan pemrosesan sinyal *digital* dari sinyal gempa bumi. Penggunaan *wavelet* pada saat ini sudah semakin berkembang dengan munculnya area sains terpisah yang berhubungan dengan analisis *wavelet* dan teori transformasi *wavelet*. Dengan munculnya area sains ini *wavelet* mulai digunakan secara luas dalam filterisasi dan pemrosesan data, pengenalan citra, sintesis dan pemrosesan berbagai variasi sinyal, kompresi dan pemrosesan citra, dll.

2.2.3 Transformasi *Wavelet (Wavelet Transform)*

Transformasi sinyal merupakan bentuk lain dari penggambaran sinyal yang tidak mengubah isi informasi dalam sinyal tersebut. Transformasi *wavelet* (*wavelet transform*) menyediakan penggambaran frekuensi waktu dari sinyal. Pada awalnya, transformasi *wavelet* digunakan untuk menganalisis sinyal bergerak (*non-stationary signals*). Sinyal bergerak ini dianalisis dalam transformasi *wavelet* dengan menggunakan teknik *multi-resolution analysis*. Secara umum teknik *multi-resolutiom analysis* adalah teknik yang digunakan untuk menganalisis frekuensi dengan cara frekuensi yang berbeda dianalisis menggunakan resolusi yang berbeda. Resolusi dari sinyal merupakan ukruan jumlah informasi di dalam sinyal yang data berubah melalui operasi filterisasi.

Transformasi *wavelet* memiliki dua seri dalam pengembangannya yaitu *Continous Wavelet Transform (CWT)* dan *Discrete Wavelet Transform (DWT).* Semua fungsi yang digunakan dalam transformasi *CWT* dan *DWT* diturunkan dari *mother wavelet* melalui translasi/pergeseran dan penskalaan/kompresi. *Mother wavelet*

merupakan fungsi dasar yang digunakan dalam transformasi *wavelet*. Karena *mother wavelet* menghasilkan semua fungsi dasar yang digunakan dalam transformasi melalui translasi dan penskalaan, maka *mother wavele*t juga akan menentukan karakteristik dari transformasi *wavelet* yang dihasilkan. Oleh karena itu, perlu pencatatan secara teliti terhadap penerapan *wavelet* dan pemilihan yang tepat terhadap *mother wavelet* harus dilakukan agar dapat menggunakan transformasi *wavelet* secara efisien. Fungsi-fungsi yang termasuk di dalam keluarga *wavelet* dipaparkan sebagai berikut.



Gambar 2.9 Keluarga *wavelet* (a)Haar, (b)Daubechies, (c)Coiflet, (d)Symlet, (e)Meyer, (f)Morlet, (g)Maxican Hat.

Seri pengembangan kedua dari transformasi *wavelet* adalah *Discrete Wavelet Transform (DWT)*. Seri pengembangan ini merupakan seri *CWT* yang didiskritkan. Dengan pendiskritan *CWT* ini maka perhitungan dalam *CWT* dapat dibantu menggunakan komputer.

2.2.4 *Discrete Wavelet Transform (DWT)*

Ide dasar *DWT* sama seperti *continuos wavelet transform* (*CWT*). Representasi waktu – skala dari sinyal digital diperoleh menggunakan teknik filtering. *Recall* dari *CWT* adalah korelasi antara *wavelet* pada perbedaan skala dan sinyal pada frekuensi digunakan sebagai ukuran persamaan. *CWT* dihitung dengan mengalikan dari sinyal,

dan mengintegrasikan semua waktu. Dalam kasus diskrit, filter dari perbedaan jalan pintas frekuensi digunakan untuk menganalisa sinyal pada perbedaan skala. Sinyal dilewatkan melalui *highpass filter* untuk menganalisa frekuensi tinggi, dan sinyal dilewatkan melalui *lowpass filter* untuk menganalisa frekuensi rendah. Analisis frekuensi yang berbeda dengan menggunakan resolusi yang berbeda inilah yang disebut dengan *multiresolution – analysis* (Polikar, Robi, 1998).

Prosedur dimulai dengan melewatkan sinyal (*sesuence*) melewati setengah kumpulan *lowpass filter* digital dengan respon dorongan h[n]. Filterisasi sinyal dapat disamakan dengan operasi matematika dari sinyal dengan respon dorongan dari filter. Operasi dalam waktu diskrit dirumuskan seperti dibawah ini (Polikar,Robi,1998) :

$$x[n]*h[n] = \sum_{k=-\infty}^{\infty} x[k].h[n-k] \qquad (2.7)$$

Pembagian sinyal melalui menjadi frekuensi tinggi dan frekuensi rendah dalam proses filterisasi *highpass filter* dan *lowpass* filter disebut sebagai dekomposisi. Proses dekomposisi dimulai dengan melewatkan sinyal asal melewati *highpass filter* dan *lowpass filter*. Misalkan sinyal asal ini memiliki rentang frekuensi dari 0 sampai dengan π rad/s. Dalam melewati *highpass filter* dan *lowpass filter* ini, rentang frekuensi di-*subsampling* menjadi dua, sehingga rentang frekuensi tertinggi pada masing – masing *subsample* menjadi $\pi/2$ rad/s. Setelah filterisasi setengah dari *sample* atau salah satu *subsample* dapat dieliminasi berdasarkan aturan Nyquist. Proses dekomposisi ini dapat melalui satu atau lebih tingkatan. Dekomposisi satu tingkat ditulis dengan persamaan sebagai berikut :

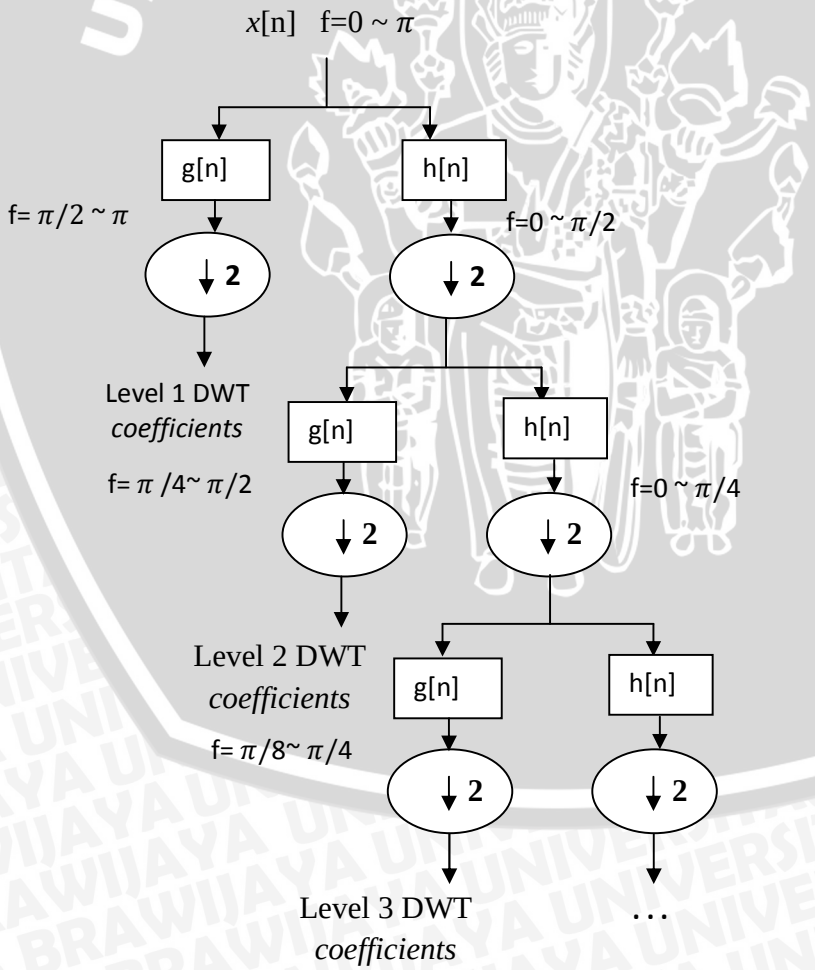
$$y_{\text{tinggi}}[k] = \sum_n x[n]h[2k-n] \qquad (2.8)$$

$$y_{\text{rendah}}[k] = \sum_n x[n]g[2k-n] \qquad (2.9)$$

$y_{\text{tinggi}}[k]$ dan $y_{\text{rendah}}[k]$ merupakan hasil dari *highpass filter* dan *lowpass filter* dan sebagai koefisien DWT dimana $y_{\text{tinggi}}[k]$ adalah detil dari informasi sinyal dan $y_{\text{rendah}}[k]$ adalah taksiran dasar dari fungsi penskalaan, $x[n]$ merupakan sinyal asal, h[n] adalah

highpass filter dan g[n] adalah *lowpass filter*. Untuk dekomposisi lebih dari satu tingkat, persamaan 2.8 dan 2.9 dapat digunakan pada masing–masing tingkatan.

Dekomposisi ini membagi dua waktu resolusi sejak setengah *sample* sekarang digolongkan seluruh sinyal. Akan tetapi, operasi menggandakan resolusi frekuensi, semenjak kumpulan frekuensi dari sinyal sekarang mempunyai rentang hanya setengah dari frekuensi sebelumnya, secara efektif mengurangi ketidakpastian dalam frekuensi dengan setengah. Prosedur diatas, yang juga dikenal sebagai pengkodean *subband*, dapat diulang untuk dekomposisi lebih lanjut. Pada setiap tingkat, penyaringan dan *subsampling* akan menghasilkan setengah jumlah *sample* (dan karena setengah waktu resolusi) dan setengah rentang band frekuensi (dan karena itu disebut resolusi frekeunsi *double*). Proses dekomposisi dapat dijelaskan seperti gambar dibawan ini (Polikar, Robi, 1998) :



Gambar 2.10 Dekomposisi *Wavelet* Dengan Fungsi Dasar x[n] f = 0 ~ π

Proses rekonstruksi diawali dengan menggabungkan koefisien *DWT* dari yang berada pada akhir dekomposisi dengan sebelumnya meng-*upsample* oleh 2 (2) melalui *highpass filter* dan *lowpass filter*. Proses rekonstruksi ini sepenuhnya merupakan kebalikan dari proses dekomposisi. Sehingga persamaan rekonstruksi pada masing – masing tingkatan dapat ditulis sebagai berikut :

$$x[n] = \sum_k (y_{\text{tinggi}}[k]\, h[-n+2k]) + (y_{\text{low}}[k]\, g[-n+2k]) \qquad (2.10)$$

Secara umum penyisipan pesan dilakukan dengan cara memodifikasi koefisien pada rentang frekuensi LL, LH, HL, atau HH yang merupakan rentang frekensi hasil dekomposisi citra menggunakan *wavelet*. Data pesan ini dapat dianggap sebagai rangkaian bilangan *w* dengan panjang *L*, yang disisipkan pada koefisien rentang yang dipilih *f* .

Algoritma umum penyisipan pesan pada koefisien adalah :

$$f' = f + \alpha .w(k) , k = 1,\ldots L \qquad (2.11)$$

Dimana α merupakan kekuatan penyisipan yang mengontrol tingkat kekuatan penyisipan pesan dan *f*' adalah koefisien sinyal asal yang telah dimodifikasi.

2.3 Steganografi

Pengertian steganografi dilihat dari segi bahasa terdiri dari dua kata, yaitu *steganos* dan *graptos* yang diambil dari bahasa yunani. *Steganos* memiliki arti menyembunyikan, sedangkan *graptos* memiliki arti tulisan. Secara umum steganografi adalah teknik penyisipan pesan kedalam media.

Walaupun steganografi dapat dikatakan memiliki hubungan yang erat dengan kriptografi, tapi metode ini sangat berbeda dengan kriptografi. Kriptografi mengacak pesan sehingga tidak dapat dimengerti, sedangkan steganografi menyembunyikan pesan sehingga tidak terlihat. Pesan yang terdapat pada *chipertext* mungkin akan mengalami kecurigaan, sedangkan pesan yang disisipkan dengan steganografi tidak. Kedua teknik ini pada umumnya dikombinasikan dalam pemakaiannya untuk mendapatkan metode pengiriman pesan rahasasia yang sulit dilacak. Pertama pesan

dienkripsi, kemudian *chipertext* disembunyikan dengan cara steganografi pada media yang tampak tidak mencurigakan.

2.3.1 Sejarah Steganografi

Steganografi sudah digunakan sejak dahulu kala, yang membedakan dengan teknologi masa kini yaitu media yang digunakan. Menurut catatn sejarah, teknik steganografi sudah diterapkan sejak tahun 440 SM. Berikut akan dipaparkan beberapa contoh dari steganografi yang dilakukan pada masa lampau :

a. Raja Darius dari Susa mencukur habis kepala salah satu tawanannya, dan menulis pesan di atas kepala tawanannya tersebut. Kemudian saat rambutnya tumbuh kembali, tawanan tersebut dikirim ke menantunya, Aristogoras di Miletus tanpa dapat terdeteksi tentang adanya pesan yang ingin disampaikan.
b. Seorang prajurit Yunani yaitu Demaratus hendak mengirimkan pesan ke Sparta yang berisi peringatan bahwa Xerxes, raja Persia, hendak menyerang Yunani. Teknik yang digunakan adalah dengan menggunakan meja yang telah diukir kemudian diberi lapisan lilin untuk menutupi pesan tersebut, dengan demikian pesan dalam meja dapat disampaikan tanpa menimbulkan kecuriaan oleh para penjaga.
c. Romawi menggunakan tinta yang tak terlihat yang berasal dari sari alami seperti susu dan jus buah. Tinta tersebut kemudian dituliskan di atas kertas. Pada akhirnya pesan yang dituliskan dapat dibaca dengan cara memanaskan kertas dengan menggunakan api di bawahnya.

Pada masa kini, steganografi lebih banyak dilakukan pada media digital, seperti teks, gambar, audio, dan video. Steganografi lebih dikenal menekankan kepada bagaimana cara penyisipan pesan tanpa banyak mempengaruhi kualitas media yang digunakan.

2.3.2 Metode Steganografi

Ada dua teknik steganografi yang dapat diterapkan pada citra *digital*. Pertama adalah teknik yang bekerja pada *domain spatial.* Teknik pada *domain spatial* sering disebut dengan teknik substitusi. Substitusi dilakukan sedemikian rupa agar indra manusia tidak dapat membedakan adanya pesan yang disispkan pada media penyisipan pesan. Salah satu metode yang terkenal dalam *domain spatial* adalah metode *Least Significant Byte (LSB).* Kedua adalah teknik yang bekerja pada *domain transform.* Teknik pada ranah transform memfokuskan penyisipan pesan ke dalam frekuensi dari *cover-file.* Salah satu metode yang bekerja dalam *domain transform* adalah *Discrete Wavelet Transform (DWT).*

Steganografi memiliki dua buah proses, yaitu penyisipan dan ekstraksi pesan. Proses penyisipan pesan pada steganografi membutuhkan dua buah masukan, yaitu pesan yang ingin disembunyikan dan media penyisipan. Hasil dari proses ini disebut dengan *stego-object,* yaitu suatu media yang mempunyai kemiripan dengan media penyisipan yang telah terdapat pesan tersembunyi di dalamnya.

2.3.3 Tujuan Steganografi

Penggunaan steganografi anatra lain bertujuan untuk menyamarkan eksistensi (keberadaan) data rahasia, sehingga sulit di deteksi, dan melindungi hak cipta suatu produk. Steganografi dapat dipandang sebagai kelanjutan kriptografi. Jika pada kriptografi, data yang disandikan (*chipertext*) tetap tersedia, maka dengan steganografi *chipertext* dapat disembunyikan sehingga pihak ketiga tidak mengetahui keberadaannya. Data rahasia yang disembunyikan dapat diekstraksi kembali persis sama seperti keadaan aslinya.

2.3.4 Kriteria Steganografi

Penyembunyian data rahasia ke dalam citra digital akan mengubah kualitas citra tersebut. Criteria yang harus diperhatikan dalam penyembunyian data adalah :

1. *Fidelity*

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

2. *Robustness*

   Data yang disembunyikan harus tahan (*robust*) terhadapp berbagai operasi manipulasi yang dilakukan terhadap citra penampung.

3. *Recovery*

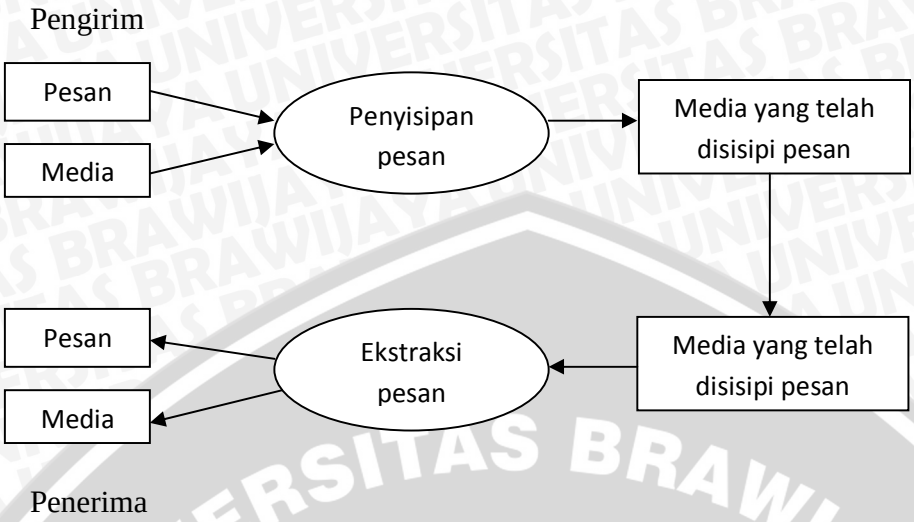
   Data yang disembunyikan harus dapat diungkapkan kembali (*reveal).* Karena tujuan dari steganografi adalah penyembunyian data, maka sewaktu-waktu data rahasia di dalam citra penampung harus dapat diambil kembali untuk digunakan lebih lanjut.

2.3.5 *Digital Image* Steganografi

Steganografi pada masa kini sering dikatakan sebagai *digital steganografi*. *Digital steganografi* menggunakan media *digital* sebagai wadah untuk menampung pesan rahasia. Media tersebut dapat berupa dokumen citra, suara, teks, maupun video. Terdapat beberapa istilah yang berkaitan dengan *digital steganografi,* yaitu :

1. Pesan rahasia *(message*)
2. Media untuk menyembunyikan pesan (*cover-file*)
3. Media yang telah disispi pesan rahasia (*stego-file*)

Pesan akan disisipkan ke dalam media oleh pengirim data dalam proses penyisipan data dengan menggunakan sebuah kunci, kemudian media yang telah disisipi pesan akan diterima oleh penerima dan diekstraksi menggunakn kunci yang sama dengan kunci pengirim, untuk mendapatkan pesan yan gsam dengan pesan yang ingin disampaikan oleh pengirim. Proses penyisipan pesan pada media dapat dilihat pada gambar di bawah ini.

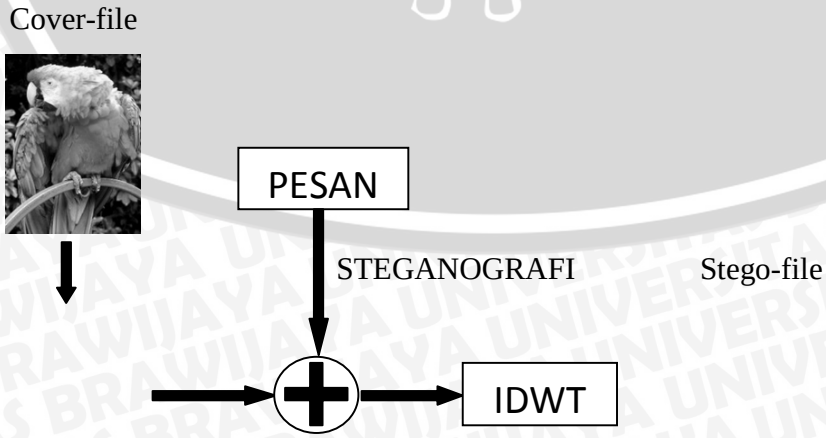


Gambar 2.11 Diagram Sistem Steganografi

2.3.6 Steganografi menggunakan DWT

Discrete Wavelet Transform (DWT) merupakan salah satu kakas yang banyak digunakan dalam teknik penyembunyian pesan dengan *domain* transform. Steganografi yang berbasis *wavelet* adalah pendekatan yang popular karena kekuatannya dalam melawan serangan.

Citra *digital* sebelumnya didekomposisi menggunakan *DWT* untuk dapat menyisipkan pesan, selanjutnya dijalankan proses *IDWT* untuk membentuk citra yang telah tersisipi pesan (*stego-file*). Dibawah ini akan dijelaskan proses steganografi menggunakan *Discrete Wavelet Transform (DWT).*



Gambar 2.12 Steganografi Menggunakan *Discrete Wavelet Transform* (http://anoa5.files.wordpress.com/2010/05/2/jpg)

2.4 Kualitas Citra

Penghitungan kualitas citra pada Tugas Akhir in diilakukan dengan dua cara, yaitu menghitung *Mean Square Error (MSE)* dan *Peak Signal-to-Noise Ratio (PSNR)* sebagai pembanding kualitas citra hasil rekosntruksi dengan citra asal.

2.4.1 *Mean Square Error* (*MSE*)

MSE merupakan salah satu cara untuk mengukur jumlah perbedaan antara nilai perkiraan dengan nilai yang sebenarnya. *MSE* mengukur rata – rata wilayah kesalahan (*error*). *MSE* (*Mean Square Error*) merupakan sigma dari jumlah kesalahan (*error*) antara citra hasil kompresi dan citra asli. Perhitungan nilai *MSE* dari citra digital berukuran NxM, dilakukan sesuai dengan rumus :

$$MSE = \frac{1}{MN}\sum_{Y=1}^{M}\sum_{X=1}^{N}\left[I(x,y) - I'(x,y)\right]^2$$

I(x,y) : nilai piksel di citra asli

I’(x,y) : nilai piksel pada citra hasil steganografi

M,N : adalah dimensi citra

2.4.2 *Peak Signal Noise to Ratio (PSNR)*

Pada proses penyisipan pesan, terjadi perunahan informasi atau adanya penurunan kualitas informasi. Informasi yang hilang seharusnya seminimal mungkin sehingga kualitas hasil steganografi bagus. *PSNR* merupakan salah satu cara untuk mengukur tingkat kesalahan akibat hilangnya informasi. *PSNR* memiliki satuan *decibel* (dB), semakin besar nilai *PSNR* semakin bagus kualitas hasil kompresi.

Perhitungan nilai *PSNR* dari citra digital berukuran N x M, dilakukan sesuai dengan persamaan berikut :

$$PSNR = 20xLog10\left(\frac{255}{\sqrt{\frac{1}{MN}\sum_{Y=1}^{M}\sum_{X=1}^{N}\left[I(x,y) - I'(x,y)\right]^2}}\right)$$

I(x,y) : adalah nilai piksel di citra asli

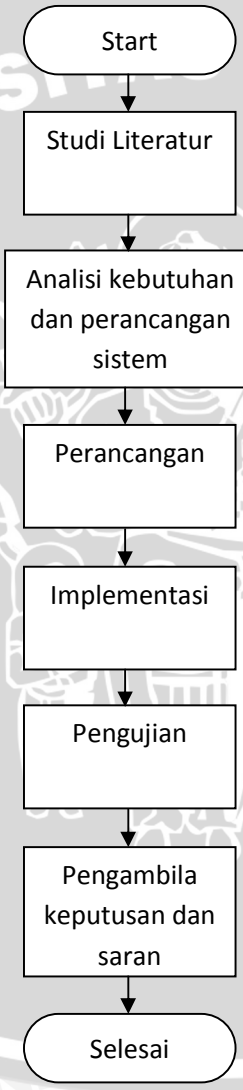
I'(x,y) : adalah nilai piksel pada citra hasil steganografi

M,N : adalah dimensi citra

BAB III

METODE PENELITIAN

Dalam penyusunan skripsi ini, dirancang suatu perangkat aplikasi steganografi pada citra *digital* dengan metode *Discrete Wavelet Transform*. Metode penelitian yang digunakan pada penyusunan skripsi ini dapat digambarka ndalam bentuk diagram seperti Gambar 3.1



Gambar 3.1 Diagram Alir Metodologi Penelitian

3.1 Studi Literatur

Studi literatur menjelaskan dasar teori yang digunakan untuk menunjang penulisan skripsi. Teori-teori pendukung tersebut meliputi:

1. Membaca dan mempelajari buku – buku yang berhubungan dengan pengolahan citra *digital*.
1. Mempelajari metode *Discrete Wavelet Transform.*
2. Mempelajari metode steganografi pada citra *digital*.
3. Mempelajari teknik – teknik dasar pemrograman dengan menggunakan *Microsoft Visual Studio .NET*

3.2 Analisa Kebutuhan

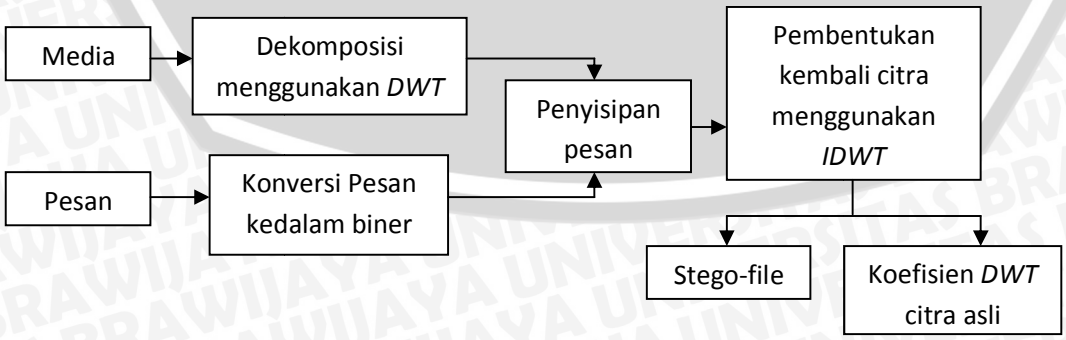
Perangkat yang digunakan untuk menunjang pembuatan aplikasi *steganografi* pada citra terkompresi *JPEG* dengan metode *Direct Wavelet Transform,* antara lain:

a. Perangkat Keras :
   1. *PC* atau *Laptop*

b. Perangkat Lunak :
   1. *Operating system* Windows 7
   2. *Microsoft visual studio .NET C#*

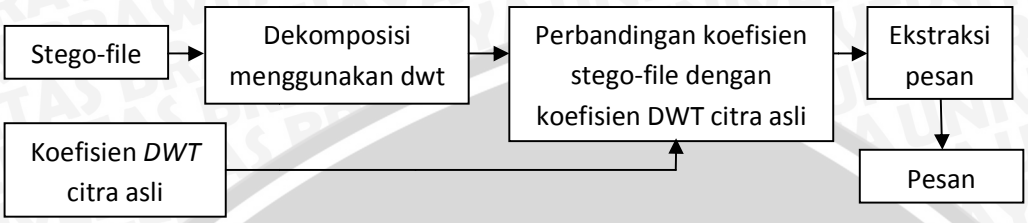
3.3 Konfigurasi Sistem

Proses steganografi pada citra digital menggunakan metode *Discrete Wavelet Transform* dapat digambarkan dengan model pada Gambar 3.2.



Gambar 3.2 Blok Diagram Sistem Penyisipan Pesan

Proses ekstraksi pesan pada citra digital menggunakan metode *Discrete Wavelet Transform* dapat digambarkan dengan model pada Gambar 3.3.



Gambar 3.3 Blok Diagram Sistem Ekstraksi Pesan

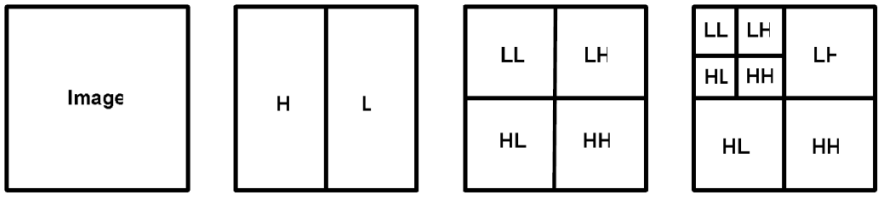
3.4 Langkah Kerja Sistem

Langkah kerja sistem yang akan dibuat dapat dikelompokkan menjadi tiga bagian, yaitu dekomposisi citra digital yang akan disisipi pesan, proses penyisipan pesan, dan ekstraksi pesan.

3.4.1 Dekomposisi Citra *Digital* yang Akan Disisipi Pesan

Dekomposisi citra Digital yang akan disisipi pesan merupakan langkah pertama yang harus dilakukan dalam proses steganografi. Dekomposisi ini menggunaan *Discrete Wavelet Transform (DWT)*. Proses dekomposisi citra dijelaskan sebagai berikut :

1. Dekomposisi citra berdasarkan *Discrete Wavelet Transform (DWT)* sehingga menghasilkan rentang frekuensi LL, LH, HL, dan HH.
2. Dekomposisi citra satu tingkat sehingga menghasilkan LL_1, LH_1, HL_1, dan HH_1.
3. Dekomposisi terhadap citra *digital* dilakukan dua tingkat sehingga LL_1 menjadi LL_2, LH_2, HL_2, dan HH_2.



Gambar 3.4 Proses Dekomposisi

3.4.2 Proses Penyisipan Pesan pada Citra

Penyisipan pesan pada proses steganografi diawali dengan mengubah pesan dalam bentuk biner, selanjutnya proses penyisipan pesan dijelaskan sebagai berikut :

1. Setelah citra melalui proses dekomposisi dalam *l* tingkatan *DWT*, pesan disisipkan ke dalam rentang frekuensi LH_l atau HL_l (l={2,3,4}).
2. Mencari koefisien terbesar f_{LH} dari rentang frekuensi LH atau koefisien terbesar f_{HL} dari rentang frekuensi HL.
3. Menyisipkan pesan ke dalam rentang frekuensi LH atau HL.

   $f'_{LH}(m,n) = f_{LH}(m,n) + \alpha.w(m,n)$; $m,n = 1,...,L$ (III-1)

   $f'_{HL}(m,n) = f_{HL}(m,n) + \alpha.w(m,n)$; $m,n = 1,...,L$ (III-2)

   Dimana $f_{LH}(m,n)$ merupakan koefisien teresar yang dipilih dan $f'_{LH}(m,n)$ merupakan koefisien yang dimodifikasi pada posisi (m,n) untuk rentang frekuensi HL. $f_{HL}(m,n)$ merupakan koefisien yang terbesar yang dipilih dan f_{HL}. $f'_{HL}(m,n)$ merupakan koefisien yang dimodifikasi pada posisi (m,n) untuk rentang frekuensi HL. α merupakan kekuatan penyisipan yang mengontrol tingkat kekuatan penyisipan pesan dan f' merupakan koefisien sinyal asal yang telah dimodifikasi. Penyisipan pesan pada citra *digital* menggunakan *DWT* ini dilakukan pada koefisien rentang frekuensi (koefisien *DWT*) sebelum direkonstruksi menggunakan IDWT.

3.4.3 Ekstraksi Pesan

1. Citra Stego-file didekomposisi dalam dua tingkatan DWT.
2. Memilih koefisien Stego-file dari rentang frekuensi LH dan HL yaitu f_{LH} dan f_{HL}.
3. Mencari koefisien citra Stego-file dari rentang frekuensi LH dan HL yaitu f_{LH} dan f_{HL}.
4. Membandingkan koefisien stego-file hasil dari proses DWT dengan koefisien DWT citra asli.
5. Mengekstraksi pesan yang ingin disampaikan.

3.5 Pengujian Sistem

Pengujian dilakukan untuk menjamin dan memastikan bahwa sistem yang telah dirancang memiliki tingkat kesalahan yang kecil. Untuk mengetahui apakah sistem bekerja dengan baik dan sesuai dengan perancangan, maka diperlukan serangkain pengujian. Pengujian yang dilakukan adalah sebagai berikut :

1. Pengujian fungsionalitas sistem.
2. Pengujian kompleksitas algoritma.
3. Pengujian kualitas citra *stego-image*.

3.6 Kesimpulan dan Saran

Pada tahap ini diambil kesimpulan dari hasil pengujian dan analisis terhadap sistem yang dibangun. Tahap yang paling akhir adalah pengambilan saran terhadap penelitian yang selanjutnya sehingga dapat menyempurnakan kekurangan yang ada dan mengembangkan pada tingkat pokok kajian yang lebih lanjut.

BAB IV

PERANCANGAN DAN IMPLEMENTASI

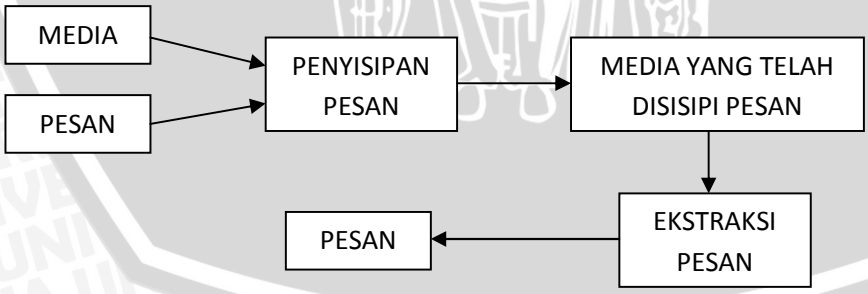
Bab ini akan menjelaskan perancangan dan implementasi aplikasi steganografi yang terdiri dari beberapa tahap. Tahap pertama merupakan tahap preproses dimana proses ini menyiapkan citra yang akan diolah, tahap kedua merupakan proses penyisipan pesan ke dalam citra *digital*, dan tahap ketiga adalah proses ekstraksi pesan.

4.1 Perancangan Secara Umum

Tahap awal yang dibutuhkan adalah membuat perancangan aplikasi secara global dimana tahap awal ini berfungsi sebagai acuan dalam proses pembuatan aplikasi yang akan dibuat. Perancangan ini diawali dengan pendefinisian kegiatan pelaku atau *user* dalam menggunakan program steganografi dengan menggunakan teknik pengolahan citra, serta perangkat yang digunakan meliputi blok sistem diagram dan cara kerja aplikasi.

4.1.1 Blok Diagram Sistem

Pada sistem steganografi terdiri dari beberapa langkah yang dapat digambarkan menjadi blok diagram dengan model seperti Gambar 4.1.



Gambar 4.1 Diagram Blok Sistem Secara Keseluruhan

Sumber : Perancangan

Fungsi masing-masing bagian dalam diagram blok ini adalah senagai berikut :

1. Pesan dan media penampung pesan digunakan sebagai *input* sistem.
2. Melakukan *image processing* penyisipan pesan kedalam media.
3. Proses penyisipan pesan menghasilkan *stego-image*.
4. Melakukan proses ekstraksi pesan dari *stego-image*.

4.1.2 Cara Kerja Aplikasi

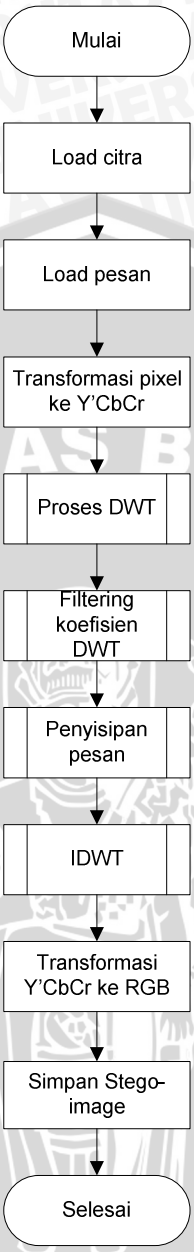
Aplikasi steganografi pada citra digital menggunakan metode *Discrete Wavelet Transform* memiliki cara kerja yang dimulai dari pengambilan citra (*cover-file*) yang akan digunakan sebagai media penyisipan pesan dan pesan rahasia yang ingin disampaikan, kemudian dilakukan proses pengolahan citra penyisipan pesan ke dalam citra sehingga menghasilkan citra yang disisipi pesan (*stego-file*), setelah itu dilakukan proses pengolahan citra mengekstraksi pesan dari *stego-file* sehingga menghasilkan pesan rahasia yang ingin disampaikan.

4.2 Perancangan Perangkat Lunak

Pada bagian perancanan ini perangkat lunak yang akan dibuat menggunakan bahasa pemrograman *Microsoft Visual Studio C#.NET 2010* dan sistem yang digunakan untuk membangun perangkat lunak ini dirancang dengan spesifikasi mampu melakukan hal-hal berikut :

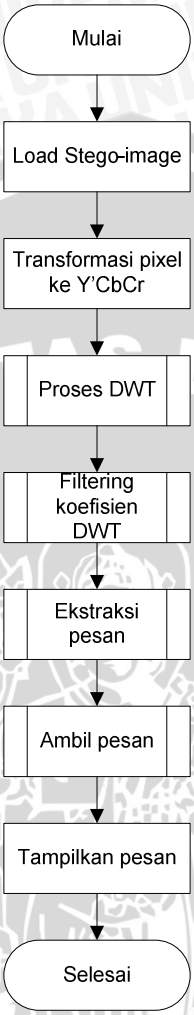
1. Mengakses data citra yang telah tersimpan di dalam *harddisk* komputer.
2. Mengakses pesan yang akan disisipkan kedalam citra.
3. Melakukan proses transformasi dari pixel ke YCbCr.
4. Melakukan proses DWT.
5. Melakukan proses *filtering* koefisien *DWT*.
6. Melakukan proses penyisipan pesan.
7. Melakukan proses ekstraksi pesan.

Sedangkan untuk detail desain aplikasi secara umum akan ditunjukkan pada gambar 4.2 dan 4.3.



Gambar 4.2 Detail Desain Aplikasi Penyisipan Pesan

Sumber : Perancangan

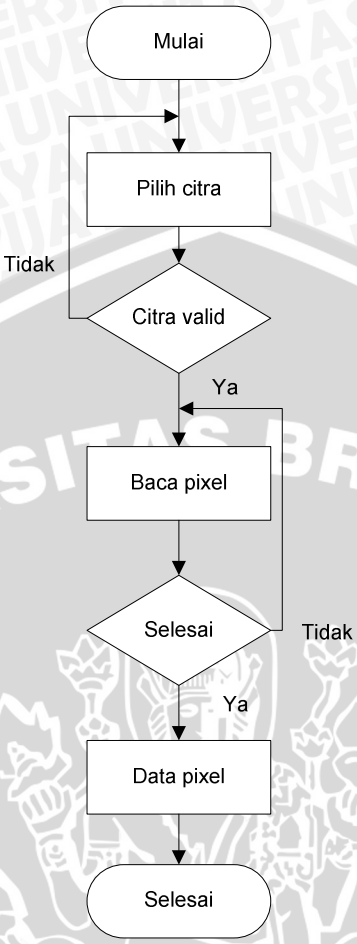


Gambar 4.3 Detail Desain Aplikasi Ekstraksi Pesan

Sumber : Perancangan

4.2.1 Perancangan *Preprocess*

Pada *preprocess* terdapat dua langkah yaitu sub *preprocess load* citra dan *load* pesan. *Flowchart* perancangan preprocess dapat dilihat pada Gambar 4.4 dan 4.5.

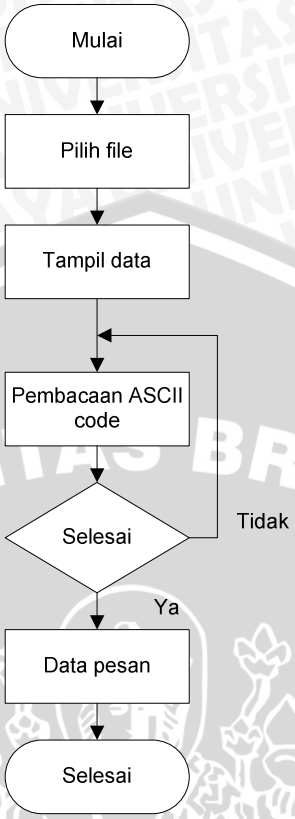


Gambar 4.4 *Flowchart Load* Citra

Sumber : Perancangan

Penjelasan *flowchart* sebagai berikut :

1. Memilih citra sebagai masukan.
2. Validasi citra apakah sesuai dengan format yang diinginkan, jika tidak kembali ke pemilian citra.
3. Menampilkan *preview* dari citra yang sudah dipilih dan memiliki format *bmp*.
4. Membaca pixel dari citra sampai data *pixel* pada citra terbaca secara keseluruhan.
5. Setelah semua proses diatas berlangsung akan diperoleh data *pixel*.



Gambar 4.5 *Flowchart Load* Pesan

Sumber : Perancangan

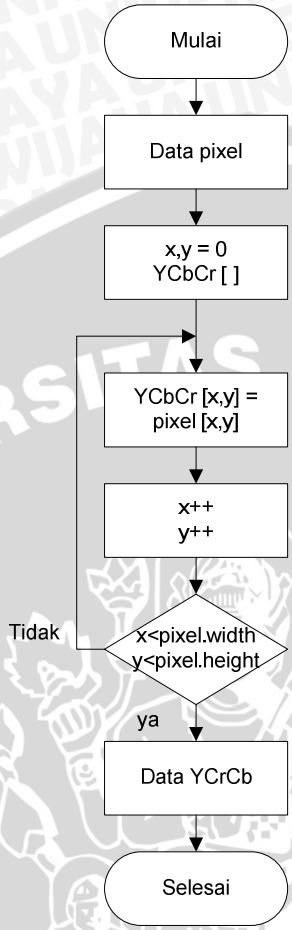
Penjelasan *flowchart* sebagai berikut :

1. Memilih pesan sebagai masukan.
2. Menampilkan *review* dari pesan yang telah dipilih.
3. Pembacaan kode ASCII pada pesan yang dipilih.
4. Setelah semua proses diatas selesai akan diperoleh data pesan.

4.2.2 Perancangan Proses

Setelah citra melewati *preprocess* maka dilanjutkan dengan pemrosesan citra untuk melakukan steganografi pada citra. Proses yang dilakukan pada tahap ini antara lain transformasi pixel ke Y'CbCr, dekomposisi citra, *filtering* koefisen *DWT*, penyisipan pesan, dan ekstraksi pesan.

4.2.2.1 Perancangan Proses Transformasi *Pixel* ke Y'CbCr



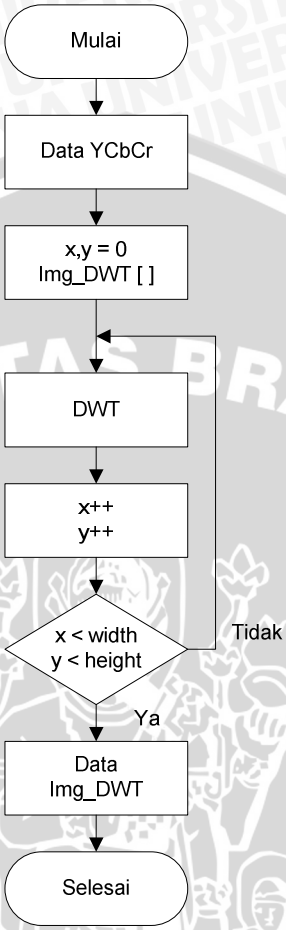
Gambar 4.6 *Flowchart* Transformasi *Pixel* ke YCbCr

Sumber : Perancangan

Penjelasan *flowchart* sebagai berikut :

1. Mengambil data *pixel* dari citra yang telah dipilih.
2. Deklarasi variabel Y'CbCr.
3. Transformasikan dari *pixel* RGB menjadi Y'CbCr.
4. Lakukan iterasi berdasarkan panjang x lebar dari citra.
5. Setelah semua proses diatas selesai dilakukan kemudian data Y'CbCr akan diperoleh.

4.2.2.2 Perancangan proses *DWT*



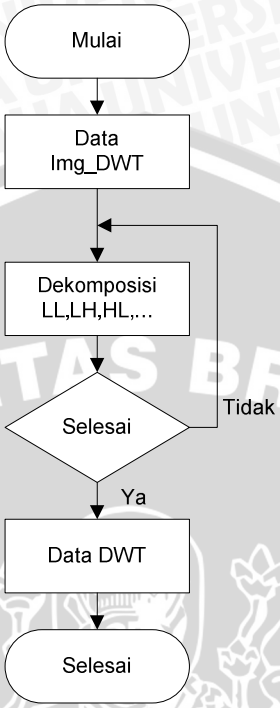
Gambar 4.7 *Flowchart* Proses *DWT*

Sember : Perancangan

Penjelasan *flowchart* sebagai berikut :

1. Menambil data Y'CbCr hasil transformasi dari pixel.
2. Deklarasi img_DWT.
3. Transformasikan dari *domain* spasial Y'CbCr ke dalam *domain* frekuensi img_DWT.
4. Lakukan iterasi berdasarkan panjang x lebar dari citra.
5. Setelah semua proses diatas selesai dilakukan data img_DWT akan diperoleh.

4.2.2.3 Perancangan *Filtering* koefisien *DWT*



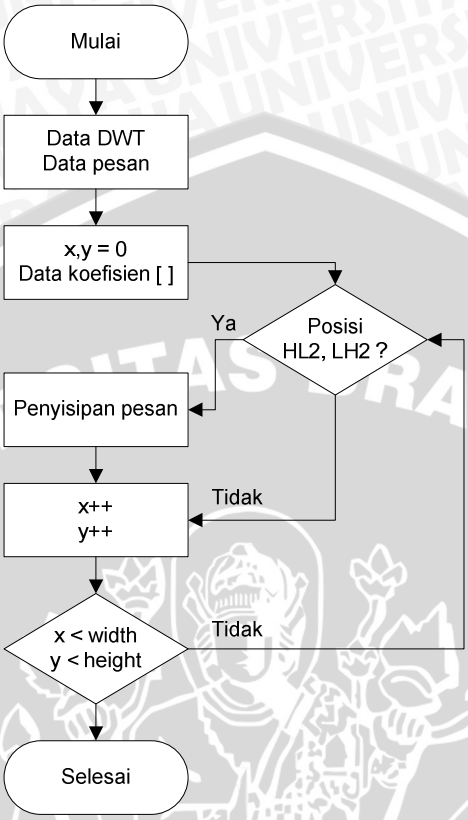
Gambar 4.8 *Flowchart Filtering* Koefisien *DWT*

Sember : Perancangan

Penjelasan *flowchart* sebagai berikut :

1. Ambil data img_DWT.
2. Dekomposisi data img_DWT menggunakan *highpass filter* dan *lowpass filter*.
3. *Filtering* mengelompokkan hasil dekomposisi menjadi empat kelompok frekuensi yaitu LL, HL, LH, dan HH.
4. Diperoleh data *DWT* dalam rentang empat kelompok frekuensi yang kemudian akan dilakukan proses penyisipan pesan pada frekuensi HL, dam LH.

4.2.2.4 Perancangan Penyisipan Pesan



Gambar 4.9 *Flowchart* Penyisipan Pesan

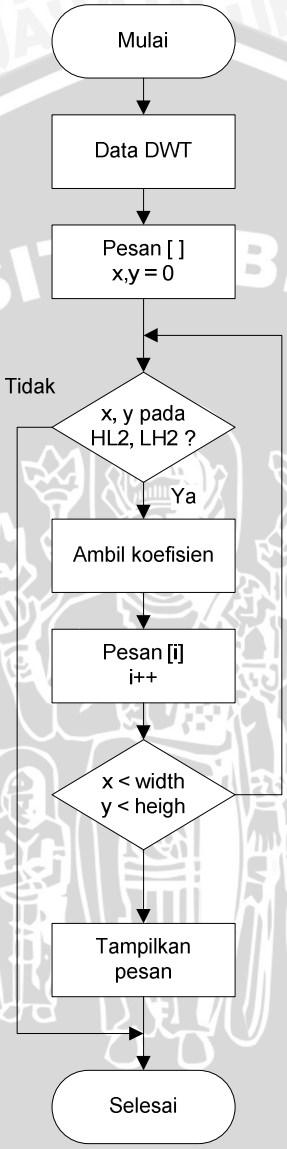
Sumber : Perancangan

Penjelasan *flowchart* sebagai berikut :

1. Mengambil data *DWT* yang telah di filtering menjadi empat kelompok frekuensi.
2. Mengambil data pesan yang ingin disisipkan.
3. Deklarasi data koefisien yang selanjutnya akan digunkan sebagai tempat penyisipan pesan.
4. Jika posisi data koefisien terdapat pada rentan gfrekuensi HL2 dan LH2 maka dilakukan proses penambahan pesan pada koefisien tersebut. Jika tidak, lakukan iterasi tanpa penambahan pesan.

5. Lakukan proses penyisipan pesan sesuai dengan panjang x lebar dari koefisien data.

4.2.2.5 Perancangan Ekstraksi Pesan



Gambar 4.10 *Flowchart* Ekstraksi Pesan

Sumber : Perancangan

Penjelasan *flowchart* sebaai berikut :

1. Mengambil koefisien data *DWT* yang telah disisipi pesan yang berasal dari stego-image.
2. Deklarasi pesan [ ] sebagai tempat penampungan hasil ekstraksi pesan yang diambill dari koefisien *DWT*.
3. Jika x,y ada pada rentang frekuensi HL2 dab LH2 maka dilakukan proses pengambilan pesan dengan cara membandingkan koefisien citra asli denan koefisien citra hasil dari stegi-image.
4. Lakukan proses pembacaan pesan sampai dengan panjang x lebar koefisien citra.
5. Tampilksn pesan hasil dari ekstraksi.

4.3 Implementasi Sistem

Setelah tahap perancangan maka untuk selanjutnya adalah tahap implementasi. Tahap ini merupakan proses transfrmasi dari proses perancangan yang telah dibuat sebelumnya dimana hasil perancangan ditransformasikan kedalam bentuk bahasa pemrograman (*coding*) yang sesuai dengan sintaks yang ada dalam bahasa pemrograman *C#.NET*.

4.4 Lingkungan Implementasi

Aplikasi dibuat menggunakan *Microsoft Visual Studio C#*. Sistem diimplementasikan dengan menggunakan spesifikasi sebagai berikut :

1. Perangkat Keras (Laptop)
   Spesifikasi :
   Processor : Intel core i5 M520 2,40Ghz
   Memory : 2 GB
   VGA : Nvidia NVS 3100M

2. Perangkat Lunak

Sistem Operasi : *Microsoft Windows 7*

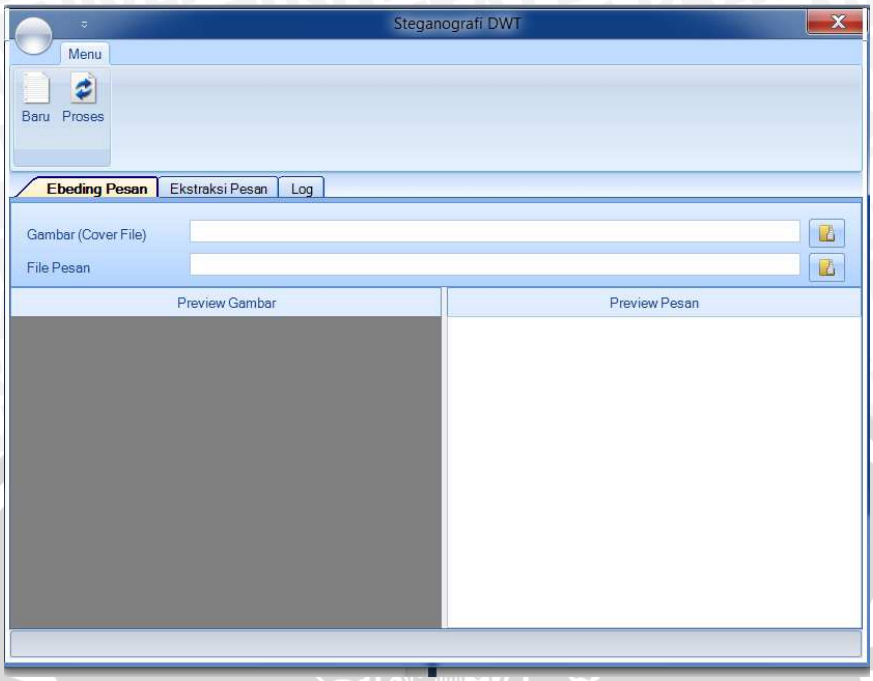
Bahasa Pemrograman : *Microsoft Visual Studio C#.NET 2010*

4.5 Implementasi *Interface* (Antarmuka)

Program steganografi pada citra menggunakan metode *Discrete Wavelet Transform* ini didesain dengan sederhana dan melakukan proses penyisipan pesan sesuai dengan langkah-langkah yang berurutan. Hal ini bertujuan agar pemakai tidak mengalami kesulitan dalam pengoperasian program ini. Pada tampilan utama program ini memiliki 8 *button* yang memiliki fungsi yang berbeda-beda. Selain itu pada tampilan utama program terdapat dua *picture box* yang berguna untuk menampilkan citra masukan dan pesan.

4.5.1 Implementasi Antarmuka Tampilan Awal Penyisipan Pesan

Pada tampilan antarmuka awal penyisipan pesan terdapat dua buah masukan, yaitu gambar (*cover-file*) dan pesan yang ingin disisipkan, dua menu utama yaitu baru dan proses. Selain itu juga terdapat dua *picture box* yang berguna untuk menampilkan citra (*cover-file*) dan pesan.

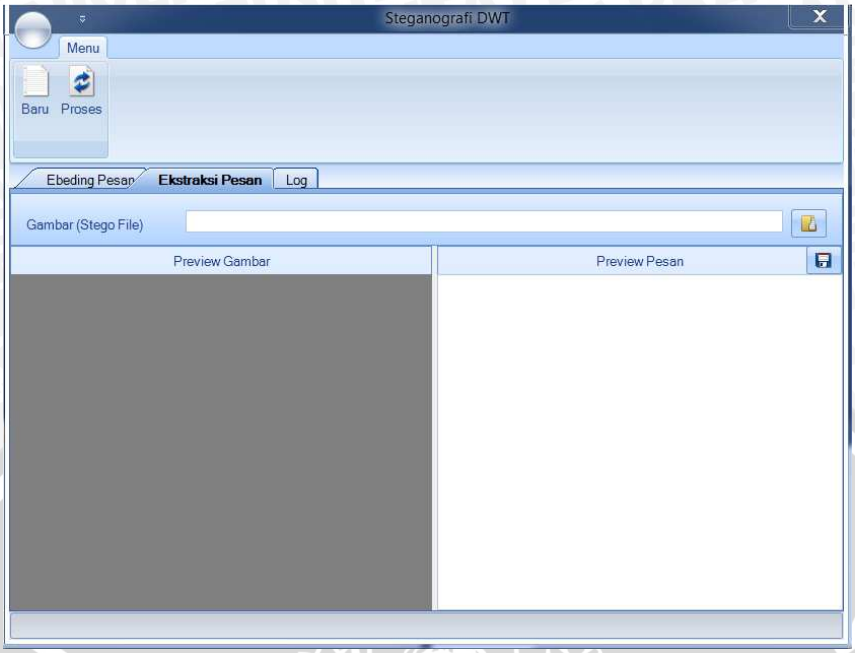


Gambar 4.11 Tampilan Awal Penyisipan Pesan

Sumber : Perancangan

4.5.2 Implementasi Antarmuka Tampilan Awal Ekstraksi Pesan

Pada tampilan antarmuka awal ekstraksi pesan terdapat dua menu utama yaitu baru dan proses. Pada tampilan ini juga memiliki dua buah masukan, yaitu Gambar (*stego-file*) dan Koefisien *cover-image*. Selain itu terdapat pula dua *picture box* yang berfungsi untuk menampilkan *preview* gambar yang dipilih dan pesan hasil ekstraksi pesan.

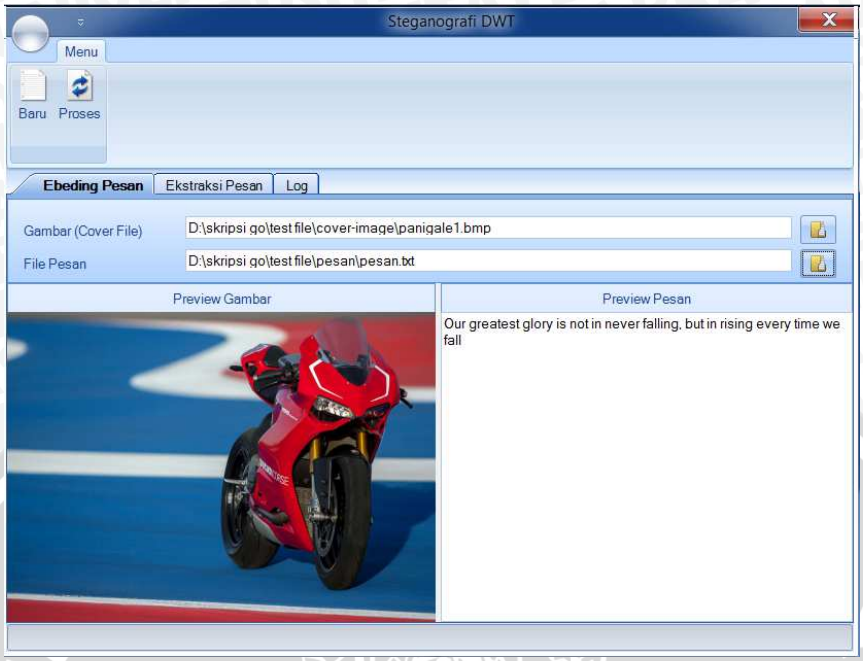


Gambar 4.12 Tampilan Awal Ekstraksi Pesan

Sumber : Perancangan

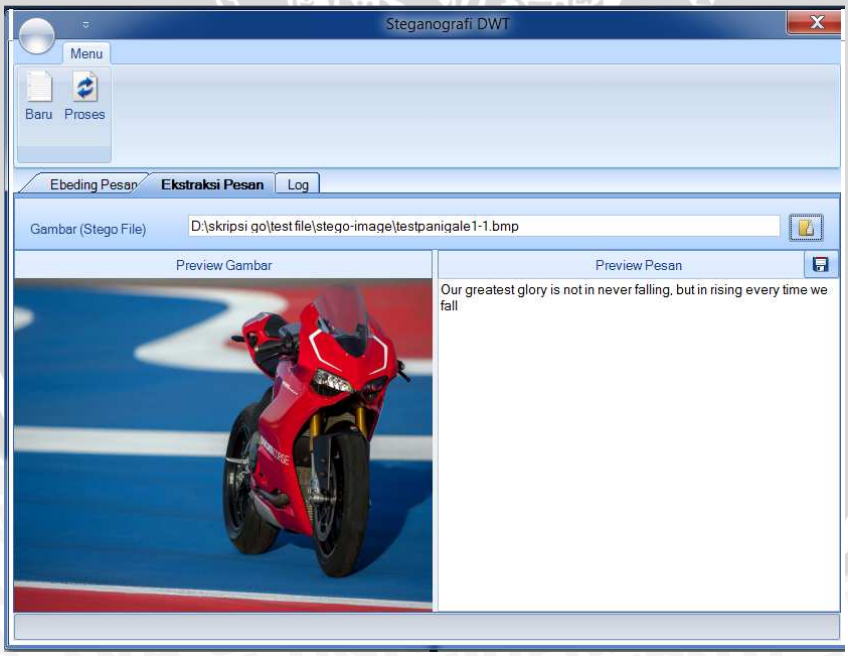
4.5.3 Implementasi Antarmuka Penyisipan dan Ekstraksi Pesan

Untuk memilih masukan berupa gambar dan pesan yang ingin disampaikan digunakan *button* buka direktori yang terletak disebalah kanan dari jenis masukan yang diinginkan, setelah button tersebut dipilih maka akan keluar tampilan baru yang berupa direktori *file* tempat dimana kita memilih masukan. Tampilan dari proses penyisipan pesan dan ekstraksi pesan dapat dilihat pada Gambar 4.13 dan 4.14.



Gambar 4.13 Tampilan Antarmuka Penyisipan Pesan

Sumber : Perancangan



Gambar 4.14 Tampilan Antarmuka Ekstraksi Pesan

Sumber : Perancangan

BAB V

PENGUJIAN

Untuk mengetahui sistem bekerja dengan baik dan sesuai dengan perancangan, maka diperlukan serangkaian pengujian. Pengujian yang dilakukan dalam bab ini adalah sebagai berikut :

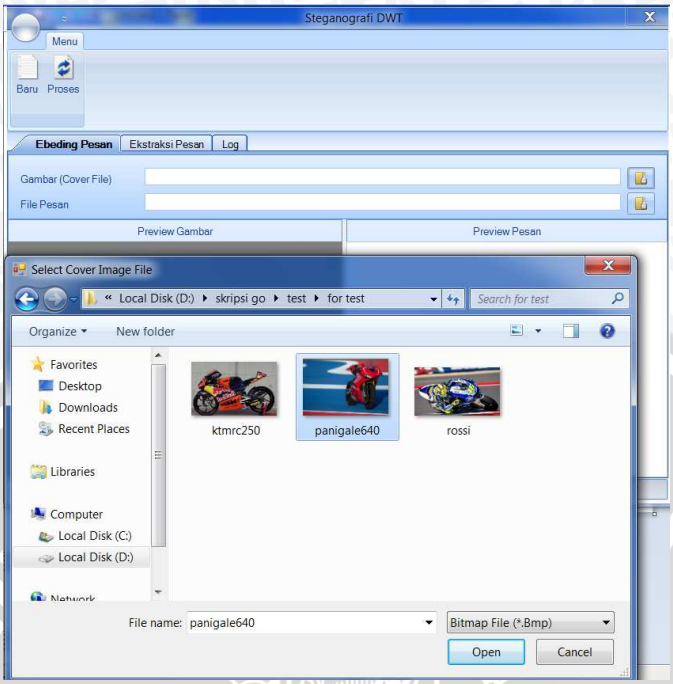
1. Pengujian fungsionalitas aplikasi.
2. Pengujian kompleksitas algoritma.
3. Pengujian kualitas citra *stego-image*.
4. Analisis faktor kegagalan.
5. Kesimpulan hasil pengujian.

5.1 Pengujian Fungsionalitas Aplikasi

Pengujian fungsionalitas perangkat lunak ini dilakukan untuk memastikan keseluruhan sistem berjalan sebagaimana mestinya. Pengujian ini dilakukan secara berurutan, dimulai dengan menjalankan aplikasi dan selanjutnya dijelaskan sebagai berikut :

1. Penyisipan pesan kedalam *cover-image*.
   a. *Load* Gambar

      Proses *load* gambar dimulai dengan menekan *button* direktori yang terletak di sebelah kanan kolom *input cover file*, kemudian akan muncul sebuah jendela yang berisi direktori dimana kita akan mengambil *cover-image* yang akan digunakan unuk menyisipkan pesan.

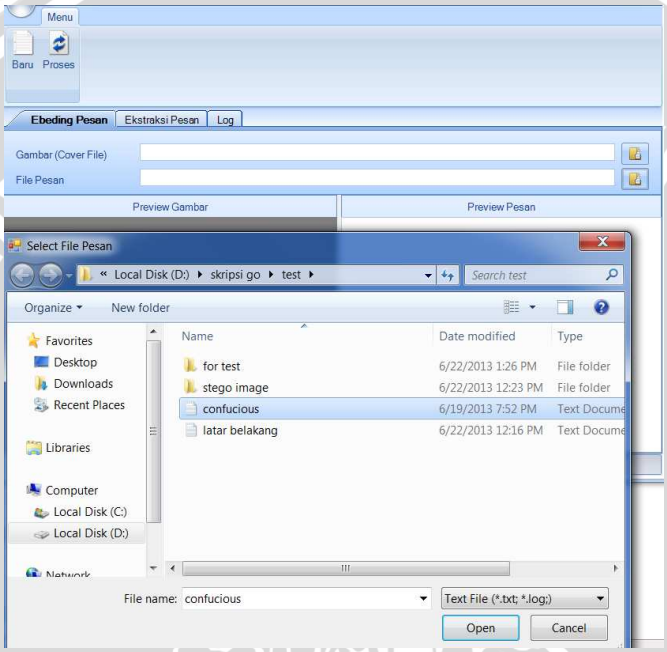


Gambar 5.1. Proses *Load Cover-image*

Sumber : Pengujian

b. *Load* Pesan

Proses *load* pesan dimulai dengan menekan *button* direktori yang terletak di sebelah kanan kolom *input file* pesan, kemudian akan muncul jendela baru yang berisi direktori dimanan kita akan mengambil pesan yang ingin disampaikan.

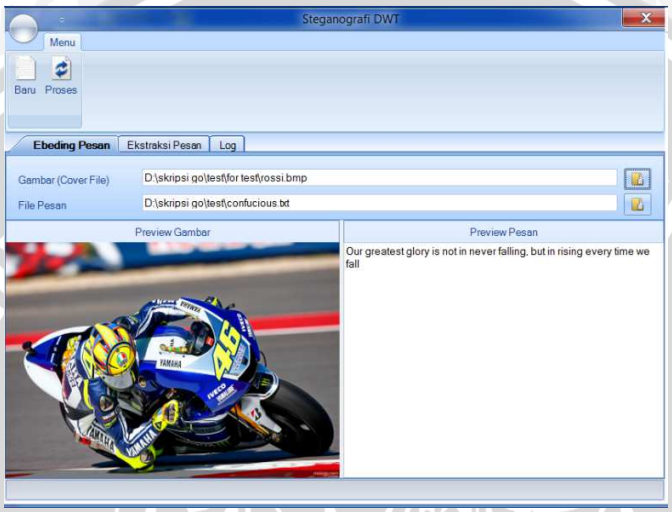


Gambar 5.2. Proses *Load* Pesan

Sumber : Pengujian

c. *Preview* Gambar dan Pesan

Proses selanjutnya setelah memilih *cover-image* dan pesan adalah *preview* gambar dan *preview* pesan yang ditampilkan berdampingan dibawah kolom *input file* pesan.

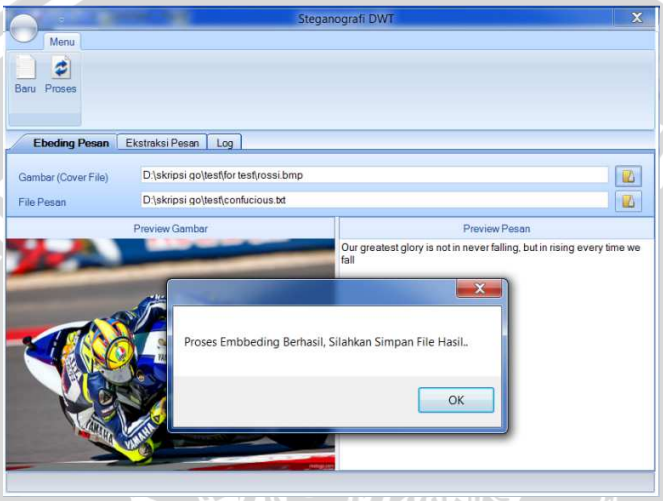


Gambar 5.3. *Preview Cover-image* dan pesan

Sumber : Pengujian

d. Proses Penyisipan Pesan

Proses penyisipan pesan dilakukan dengan cara memilih *button* proses yang terletak di sebelah kiri atas. Setelah proses penyisipan pesan selesai dilakukan, akan muncul jendela baru yang berisi pemberitahuan bahwa pesan telah berhasil disispkan dan pilihan untuk menyimpan *stego-image*.

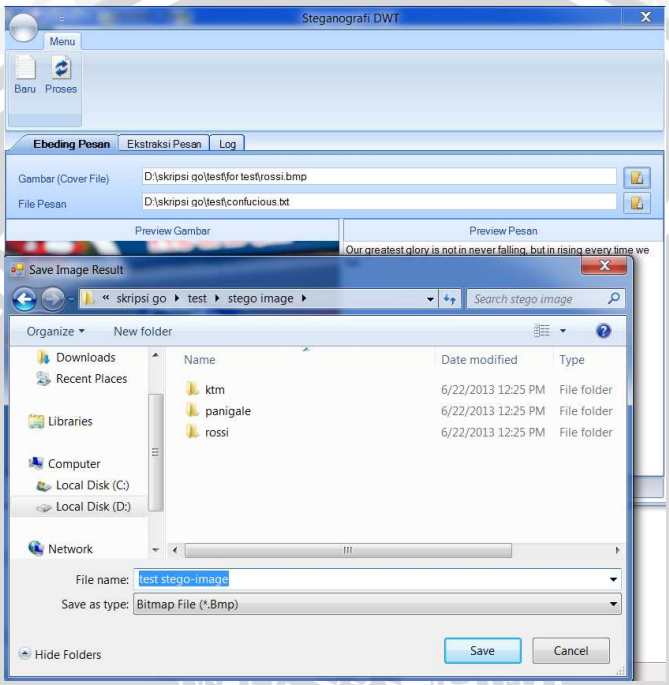


Gambar 5.4. Proses Penyisipan Pesan

Sumber : Pengujan

e. Simpan *File Stego-image*

Proses selanjutnya adalah menyimpan *stego-image* hasil dari penyisipan pesan yang akan ditampilkan dalam jendela baru yang berisi direktori dimana kita akan menyimpan *stego-image* tersebut.



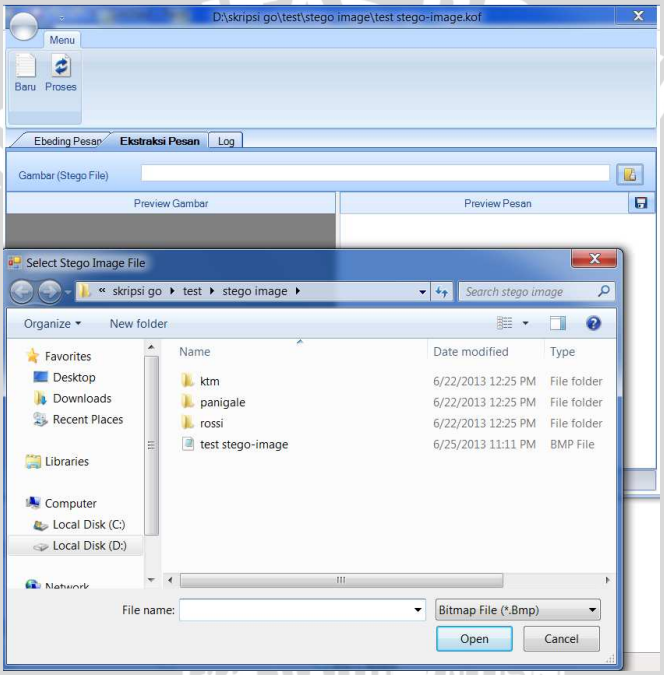
Gambar 5.5 Proses Simpan File Stego-image

Sumber : Pengujian

2. Ekstraksi pesan dari *stego-image*.

   a. *Load* Citra *Stego-image*

   Proses ekstraksi pesan dimulai dengan menekan *button* ekstraksi pesan yang berada di bawah *button* proses. Selanjutnya memilih *file stego-image* yang akan diekstraksi dengan menekan *open button* direktori yang berada di sebelah kanan kolom *input file stego-image*.

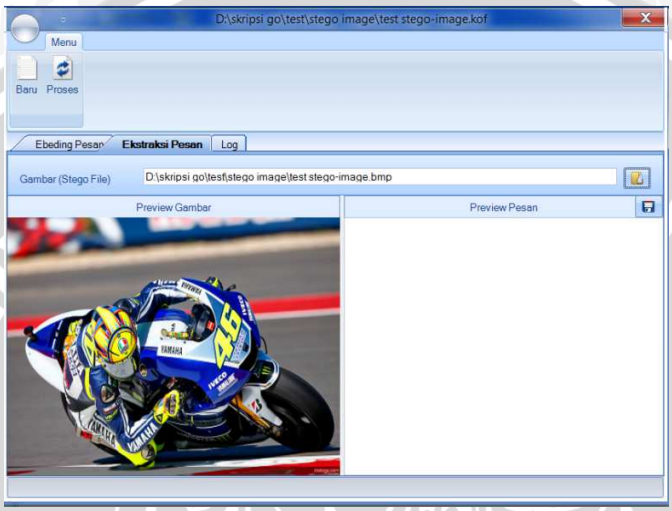


Gambar 5.6. Proses Load Citra Setego-image

Sumber : Pengujian

b. *Preview* Citra

Setelah memilih *file stego-image* akan ditampilkan *preview* dari *file* tersebut yang ditampilkan pada kolom *preview* gambar yang terletak di sebelah kiri bawah antarmuka aplikasi.

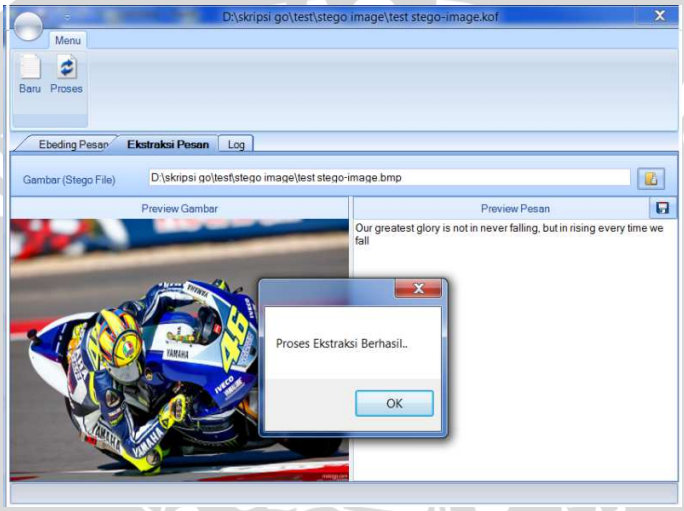


Gambar 5.7. *Preview* Citra

Sumber : Pengujian

c. Proses Ekstraksi dan Preview Pesan

Proses ekstraksi pesan dilakukan dengan menekan *button* proses yang ada di sebelah kiri atas antarmuka aplikasi. Setelah itu akan muncul jendela baru yang berisi notifikasi bahwa proses ekstraksi berhasil dilakukan, pesan akan di tampilkan pada kolom *preview* pesan yang terletak disebelah kanan bawah antarmuka aplikasi.

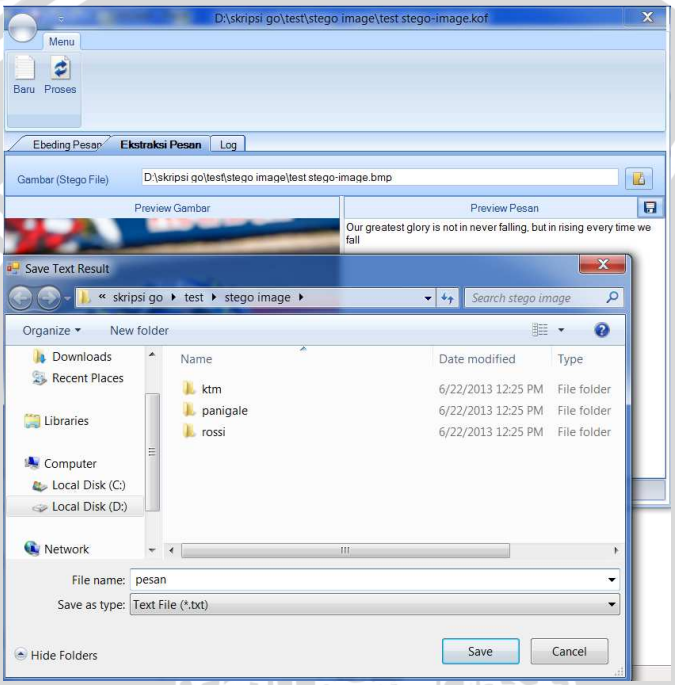


Gambar 5.8. Proses Ekstraksi Pesan

Sumber : Pengujian

d. Simpan Pesan

Proses selanjutnya adalah menyimpan pesan yang telah di ekstraksi dengan memilih *button* simpan pesan yang terletak disebelah kanan dibawah *open button* direktori file stego-image. Kemudian akan muncul jendela baru yang berisi direktori tempat dimana kita akan menyimpan pesan.



Gambar 5.9. Proses Simpan Pesan

Sumber : Pengujian

5.2 Pengujian Kompleksitas Algoritma

a. Algoritma Proses DWT

```
//metod proses DWT
 public void doDWT()
 {

   imageDWT = new YCrCb[Gambar.Height, Gambar.Width];
   double Yv, Cr, Cb;
   for (int y = 0; y < Gambar.Height; ++y)
   {
      for (int x = 0; x < Gambar.Width; ++x)
      {

        // statement dwt

        //init pixel dwt baru
        imageDWT[y, x] = new YCrCb(Yv,Cr,Cb);
       }
    }
 }
```

Kompleksitas algoritma = O (N^2)

b. Algoritma Filtering DWT

```
double LLv=0, LHv=0, HLv=0, HHv=0;
for (int y = 0; y < Gambar.Height/2; ++y)
{
    for (int x = 0; x < Gambar.Width/2; ++x)
    {
      LLv = 0;
      LHv = 0;
      HLv = 0;
      HHv = 0;

      for (int i =0 ;i<1;i++){
      // perhitungan posii

    }
     aLL.Add(LLv);
     aLH.Add(LHv);
     aHL.Add(HLv);
      aHH.Add(HHv);
     }
  }
```

Kompleksitas algoritma = O (N^3)

c. Algoritma Penyisipan Pesan

```
double Yv, Cr, Cb;
int currIndex=0;
FileInfo fi = new FileInfo(nama);
DirectoryInfo di = fi.Directory;
fi = new FileInfo(nama);
if (fi.Exists == true)
    fi.Delete();


    for (int y = 0; y < Gambar.Height; ++y)
    {
       for (int x = 0; x < Gambar.Width; ++x)
       {
          if (currIndex < dataMsg.Length)
          {

            if ( (x > Gambar.Width / 2 && (y > 0 && y < Gambar.Height/2)) ||
            (y > Gambar.Height / 2 && (x > 0 && x < Gambar.Width / 2)) ||
            ((x > Gambar.Width / 4 && x < Gambar.Width / 2) && (y > 0 && y <
            Gambar.Height / 4)) ||
            ((x > 0 && x < Gambar.Width / 4) && (y > Gambar.Height / 4 && y <
            Gambar.Height / 2)) )

            {

            string strKof = Convert.ToString(dataMsg[currIndex]/alpha)+"|";
            Console.Write(imageDWT[y, x].GetY() + ":" + imageDWT[y,x].
            GetCr() + ":" + imageDWT[y, x].GetCb());
            Yv = imageDWT[y, x].GetY() + (dataMsg[currIndex] * alpha);

            if (currIndex + 1 < dataMsg.Length)
            {

               Cr = (double)imageDWT[y, x].GetCr() + (dataMsg[currIndex + 1]   *
               alpha);
               strKof += Convert.ToString(dataMsg[currIndex + 1] / alpha) + "|";
            }

            else
            {

               Cr = imageDWT[y, x].GetCr();
               strKof += Convert.ToString(alpha) + "|";

            }

            if (currIndex + 2 < dataMsg.Length)
            {

              Cb = (double)imageDWT[y, x].GetCb() + (dataMsg[currIndex + 2]
               * alpha);
               strKof += Convert.ToString(dataMsg[currIndex + 2] / alpha);
            }
```

```
            else
            {

              Cb = imageDWT[y, x].GetCb();
              strKof += (alpha)  ;
            }

              imageDWT[y, x] = new YCrCb(Yv, Cr, Cb);
              Console.WriteLine(" -->>> " + imageDWT[y, x].GetY() + ":" +
              imageDWT[y, x].GetCr() + ":" + imageDWT[y, x].GetCb());
              sw.WriteLine(strKof );
              currIndex += 3;
            }
          }
        }
    }
```

Kompleksitas algoritma = O (N) + (N^3)

d. Algoritma Ekstraksi pesan

```
//metod ekstrak
dataMsg = new byte[1000];
for (int y = 0; y < dataMsg.Length; ++y)
{
   dataMsg[y] = 0;
}
msgResult = new byte[dataMsg.Length];
sbResult = new StringBuilder();
for (int y = 0; y < Gambar.Height; ++y)
{
   for (int x = 0; x < Gambar.Width; ++x)
   {
      if ( sR.Peek() >= 0)
      {

         //periksa range tinggi dan rendah
         if ((x > Gambar.Width / 2 && (y > 0 && y < Gambar.Height / 2)) ||    (y
         > Gambar.Height / 2 && (x > 0 && x < Gambar.Width / 2)) ||
         ((x > Gambar.Width / 4 && x < Gambar.Width / 2) && (y > 0 && y <
         Gambar.Height / 4)) ||
         ((x > 0 && x < Gambar.Width / 4) && (y > Gambar.Height / 4 && y <
         Gambar.Height / 2)))

         {
            //mengambil isi dataMsg
            string datas = sR.ReadLine();
            string[] Arrhasil = datas.Split('|');
            double yx , cr, cb;
            yx = double.Parse(Arrhasil[0]);
            cr = double.Parse(Arrhasil[1]);
            cb = double.Parse(Arrhasil[2]);
```

```
Console.WriteLine("  --------> " + imageDWT[y, x].GetY() + ":" +
imageDWT[y, x].GetCr() + ":" + imageDWT[y, x].GetCb());
```

```
            double tmpVal = 0;
            if (imageDWT[y, x].GetY() - (yx/alpha) < Double.MaxValue  )

            {
              tmpVal = Math.Round((imageDWT[y, x].GetY() - yx) / alpha);
              sbResult.Append((char) Math.Round(yx * alpha));
            }

            if (imageDWT[y, x].GetCr() - cr < Double.MaxValue)
            {
              tmpVal = Math.Round((imageDWT[y, x].GetCr() - cr) / alpha);
              sbResult.Append((char)Math.Round(cr * alpha));
            }

            if (imageDWT[y, x].GetCb() - cb < Double.MaxValue)
            {
              tmpVal = Math.Round((imageDWT[y, x].GetCb() - cb) / alpha);
              sbResult.Append((char)Math.Round(cb * alpha));
            }
          }
        }
    }
}
```

Kompleksitas algoritma = O (N) + (N^3)

e. Algoritma IDWT

```
//metode invers dwt

  double Yv=0, Cr=0, Cb=0;
  for (int y = 0; y < Gambar.Height; ++y)
  {
     for (int x = 0; x < Gambar.Width; ++x)
     {
        //memeriksa nilai frekuensi
        {
            //mengisi dengan nilai awal cover
        }
        else
        {
           //mengisi dengan fungsi invers
        }
            imageDWT[y, x] = new YCrCb(Yv, Cr, Cb);
        }
     }
```

Kompleksitas algoritma = O (N^2)

Jumlah kompleksitas algoritma adalah :

O (N^2) + O (N^3) + (O (N) + (N^3)) + (O (N) + (N^3)) + O (N^2) = 2N + 2(N^2) + 3(N^3)

5.3 Pengujian Kualitas Citra *Stego-image*

5.3.1 Pengujian Kualitas Citra Berdasarkan Nilai *PSNR*

Pengujian yang pertama dilakukan terhadap sebuah citra yang memiliki resolusi 586 x 371 *pixel* dan berukuran sebesar 637 KB. Citra akan diuji dengan menyisipkan lima buah pesan berbeda yang memiliki ukuran masing-masing 30 *bytes*, 50 *bytes*, 70 *bytes*, 90 *bytes* dan 110 *bytes*. Pengujian ini dilakukan untuk mengetahui pengaruh ukuran pesan yang disisipkan dengan tingkat kualitas citra hasil steganografi.

Untuk melakukan analisis pengujian pengaruh jumlah pesan terhadap kualitas citra, tiap gambar hasil penyisipan pesan (*stego-image*) dibandingkan dengan citra asli (*cover-image*) berdasarkan perhitungan tingkat *PSNR*. Dibawah ini adalah hasil dari pengujain kualitas citra :

Tabel 5.1 Hasil Pengujian Penyisipan Pesan dalam Nilai *PSNR*.

No.	Citra	Resolusi (px)	Ukuran (KB)	Pesan (karakter)	Ukuran (*bytes*)	PSNR (dB)
1	KTMRC	586 x 371	637	30	30	99,112
2	KTMRC	586 x 371	637	50	50	98,476
3	KTMRC	586 x 371	637	70	70	95,985
4	KTMRC	586 x 371	637	90	90	91,960
5	KTMRC	586 x 371	637	110	110	91,558

Sumber : Pengujian

Dari hasil pengujian yang dilakukan pada sebuah citra yang telah disisipi masing pesan sebesar 30 *bytes*, 50 *bytes*, 70 *bytes*, 90 *bytes* dan 110 *bytes*, diperoleh hasil yang nilai *PSNR* dari masing-masing stego-image berbeda untuk setiap jumlah pesan yang disisipkan. Dari table 5.1 terlihat bahwa jumlah karakter yang disisipkan pada citra uji berpengaruh terhadap nilai *PSNR* yang dihasilkan. Semakin banyak

karakter yang disisipkan maka semakin berkurang kualitas citra *stego-image* seiring dengan menurunnya nilai *PSNR*.

5.3.2 Tingkat Keberhasilan Ekstraksi Pesan dab Pengujian Perubahan Citra Secara Kasat Mata

Pengujian yang kedua dilakukan terhadap enam buah citra yang berbeda yang masing-masing memiliki resolusi 800 x 450 px, 640 x 426 px, 586 x 371 px, 480 x 270 px, 586 x 371 px dan 480 x 270 px. Setiap gambar disisipkan sebuah pesan yang memiliki nilai yang berbeda antara satu gambar dengan gambar yang lain, masing-masing pesan memiliki ukuran 110 *bytes*, 90 *bytes*, 70 *bytes*, 50 *bytes* dan 3,34 KB.

Pengujian yang kedua ini dilakukan untuk mengetahui tingkat keberhasilan ekstraksi pesan dari *stego-image*, serta untuk mengetahui adanya perubahan citra *stego-image* hasil dari steganografi secara kasat mata. Dibawah ini adalah hasil dari pengujian kedua :

Tabel 5.2 Hasil Pengujian Perubahan Citra Secara Kasat Mata dan Tingkat Keberhasilan Ekstraksi Pesan.

No.	Citra	Resolusi (px)	Ukuran pesan awal	Ukuran pesan akhir	Perubahan citra secara kasat mata	Status ekstraksi
1	MM99	800 x 450	110 *bytes*	110 *bytes*	Tidak	Berhasil
2	Panigale	640 x 426	90 *bytes*	90 *bytes*	Tidak	Berhasil
3	KTMRC	586 x 371	70 *bytes*	70 *bytes*	Tidak	Berhasil
4	Rossi	480 x 270	50 *bytes*	50 *bytes*	Tidak	Berhasil
5	KTM2	586 x 371	3,34 KB	3.34 KB	Ya	Tidak Berhasil, Pesan Terpotong
6	Rossi46	480 x 270	3,34 KB	3.34 KB	Ya	Tidak Berhasil, Pesan Terpotong

Sumber : Pengujian

Tabel 5.3 Hasil Pengujian Tingkat Keberhasilan Ekstraksi Pesan 1 dan 2

No.	Citra	Isi Pesan
1	MM99 *cover-image* 	Pesan Awal : ini adalah pesan rahasia yang memilki jumlah sebanyak seratus sepuluh karakter, dimohon menjaga kerahasiaannya
	MM99 *stego-image* 	Pesan Ekstraksi : ini adalah pesan rahasia yang memilki jumlah sebanyak seratus sepuluh karakter, dimohon menjaga kerahasiaannya
2	Panigale *cover-image* 	Pesan Awal : ini adalah pesan yang rahasia dan kita harus menyimpan rahasia ini dengan sangat hati-hati
	Panigale *stego-image*	Pesan Ekstraksi : ini adalah pesan yang rahasia dan kita harus menyimpan rahasia ini dengan sangat hati-hati

Sumber : Pengujian

Tabel 5.4 Hasil Pengujian Tingkat Keberhasilan Ekstraksi Pesan 3 dan 4

No.	Citra	Isi Pesan
3	KTMRC *cover-image* 	Pesan Awal : ini adalah pesan yang rahasia dan tidak boleh tersebar ke ruang publik
	KTMRC *stego-image* 	Pesan Ekstraksi : ini adalah pesan yang rahasia dan tidak boleh tersebar ke ruang public
4	Rossi *cover-image* 	Pesan Awal : ini adalah pesan yang bersifat sangat amat rahasia

	Rossi *stego-image*  	Pesan Ekstraksi : ini adalah pesan yang bersifat sangat amat rahasia

Sumber : Pengujian

Dari hasil pengujian tingkat keberhasilan ekstraksi pesan yang disajikan pada tabel 5.2, 5.3 dan 5.4 terlihat bahwa pesan berhasil diekstraksi tanpa ada kerusakan pada pesan hasil ektraksi dari *stego-image*. Dari keseluruhan citra yang diuji memiliki ukuran resolusi yang berbeda, serta ukuran pesan yang berbeda-beda pula pada setiap gambar.

Pada citra percobaan nomor satu sampai dengan empat yang disajikan dalam tabel 5.2, menunjukkan bahwa citra *stego-image* tidak mengalami perubahan kualitas citra secara signifikan, perubahan yang tidak nampak dikarenakan ukuran pesan yang disisipkan relatif kecil, yaitu antara 50 *bytes* sampai dengan 110 *bytes*.

Saat ukuran pesan yang disisipkan berukuran relatif kecil, maka perubahan kualitas citra *stego-image* tidak dapat terlihat secara kasat mata, sebaliknya saat ukuran pesan yang disisipkan besar maka akan terdapat perubahan kualitas citra yang dapat diamati secara kasat mata, seperti pada citra nomor lima dan enam yang disajikan pada tabel 5.2.

Perubahan secara kasat mata pada citra nomor lima dan enam lebih lanjut akan diperlihatkan pada tabel 5.5 dan 5.6 yang membandingkan perubahan kualitas pada citra sebelum disisipi pesan (*cover-image*) dan citra yang telah disisipi pesan (*stego-image)*.

Tabel 5.5 Hasil Pengujian Perubahan Citra Secara Kasat Mata 1

No.	Citra
1	KTM2 *cover-image* 
	KTM2 *stego-image* 

Sumber : Pengujian

Tabel 5.6 Hasil Pengujian Perubahan Citra Secara Kasat Mata 2

No.	Citra
2	Rossi2 *cover-image*  Rossi2 *stego-image* 

Sumber : Pengujian

Dari hasil pengujian perubahan citra secara kasat mata yang disajikan dalam tabel 5.5. dan 5.6, terlihat bahwa terjadi perbedaan yang signifikan antara *cover-image*

dan *stego-image*. Perbedaan tersebut nampak terlihat jelas diamati dengan kasat mata, perbedaan tersebut ditandai dengan panah berwarna kuning. Perbedaan yang terlihat mencolok tersebut disebabkan oleh jumlah pesan yang terlalu besar, sehingga mempengaruhi *pixel image* pada *stego-image* hasil dari proses steganografi.

Selain ukuran pesan yang disisipkan, besarnya resolusi gambar yang digunakan sebagai *cover-image* juga mempengaruhi kualitas citra *stego-image*. Dengan ukuran pesan yang sama, pada gambar yang disajikan dalam tabel 5.5 dan 5.6 terlihat bahwa hasil dari perubahan kualitas citra terlihat berbeda, perbedaan perubahan kualitas citra tersebut terjadi dikarenakan ukuran resolusi citra yang dijadikan sebagai *cover-image* berbeda.

5.3.4 Pengujian Batas Maksimal Pesan yang Bisa Disisipkan kedalam Citra

Pengujian batas maksimal pesan dilakukan untuk mengetahui berapa kapasitas maksimal dari pesan yang dapat disisipkan kedalam *cover-image*, pada pengujian ini tidak memperhatikan terjadinya penurunan kualitas citra maupun kerusakan pada *stego-image* hasil steganografi.

Pengujian dilakukan pada citra rossi46 yang memiliki resolusi 480 x 270 pixel dan berukuran sebesar 379 *KB* kemudian diuji dengan memasukkan beberap pesan yang memiliki urkuran berbeda-beda.

Dari beberapa percobaan yang dilakukan untuk menguji kapasitas maksimal pesan yang dapat dimasukkan ke dalam citra rossi46, pesan maksimal yang bisa disisipkan sebesar 205 *KB*, saat pesan yang dimasukkan melebihi 205 *KB* maka aplikasi tidak dapat menyisipkan pesan. Sehingga jumlah kapasaitas pesan maksimal yang dapat disisipkan menggunakan aplikasi ini adalah ¾ dari kapasitas *cover-image*.

5.4 Analisis Faktor Kegagalan

Pada kenyataannya, aplikasi steganografi menggunakan *DWT* tidak selalu berjalan dengan optimal. Hal ini dipengaruhi oleh beberapa faktor yang diantara sebagai berikut :

1. Resolusi citra yang digunakan sebagai *cover-image* terlalu kecil, sehingga jika terjadi perubahan pada *pixel image* akan terlihat dengan jelas pada *stego-image* hasil dari penyisipan pesan.
2. Pesan yang disispkan terlalu besar yang menyebabkan perubahan *pixel image* pada *stego-image*, sehingga perubahan pada *stego-image* terlihat dengan jelas.
3. Citra yang memiliki sebaran frekuensi rendah, sehingga jika terjadi perubahan *pixel image* akan mudah merubah terlihat pada stego-image.

5.5 Kesimpulan Hasil Pengujian

Berdasarkan hasil pengujan yang dilakukan, memberikan kesimpulan bahwa secara fungsional sistem sudah dapat menghasilkan *output* sesuai dengan yang diharapkan apabila resolusi citra *cover-image* cukup untuk disispi pesan sehingga perubahan pada *stego-image* tidak terlihat jelas.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, pengujian dan analisis sistem maka dapat diambil kesimpulan sebagai berikut :

1. Proses penyisipan pesan diawali dengan *load cover-image* dan *load* pesan, kemudian dilakukan proses transformasi *RGB* ke *YCbCr*, selanjutnya proses *DWT* dan *filtering* koefisien *DWT*, setelah itu dilakukan penyisipan pesan yang diakhiri dengan mentransformasikan kembali *YCbCr* ke *RGB* yang kemudian terbentuk *stego-image*.
2. Proses ekstraksi pesan diawali dengan *load stego-image*, kemudian transformasi *RGB* ke *YCbCr*, selanjutnya proses *DWT* dan *filtering* koefisien *DWT*, setelah itu melakukan perbandingan antara koefisien *DWT cover-image* dengan *stego-image* yang selanjutnya pesan dapat diekstraksi.
3. Resolusi citra yang digunakan sebagai *cover-image* harus ideal dengan ukuran pesan yang akan disisipkan, jika tidak maka akan terlihat perubahan citra *stego-image* yang terlihat secara kasat mata.
4. Terjadinya sedikit perubahan *pixel* yang terlihat secara kasat mata pada *stego-image* tidak menyebabkan terjadinya kerusakan pesan hasil ekstraksi dari *stego-image*.

6.2 Saran

Dalam perancangan dan pembuatan aplikasi steganografi menggunakan *DWT* ini masih terdapat kekurangan dan kelemahan, oleh karena itu masih diperlukan adanya penyempurnaan dalam rangka pengembangan ke depan. Adapun hal yang dapat dikembangkan ke depan adalah pemilihan format file *cover-image* yang lebih banyak.

DAFTAR PUSTAKA

1. Al-Naima, Fawzi. 2010. *A Modified High Capacity Image Steganography Technique Based on Wavelet Transform*. Iraq: IAJIT.
2. Burger, Wilhelm, dkk. 2008. *Digital Image Processing*. New York: Spinger.
3. Burrus, C. Sidney, Gopinath, Ramesh A., Guo, Haitao. (1998). *Introduction to Wavelets and Wavelet Transforms: A Primer*. Prentice-Hall, New Jersey.
4. Chen Po-Yuech, Lin Hung-Ju. 2006. *A DWT Based Approach for Image Steganography*. Taiwan : IJACE.
5. Cole, Eric. 2003. *Hiding in Plainsight : Steganography and the Art of Cover Communication*. Canada: Wiley Publishing.Inc.
6. Gonzalez, Rafael C., Woods, R.E., 2002. *Digital Image Processing*. New Jersey : Prentice-Hall, Inc., Upper Saddle River.
7. Hayes, Monson H. (1999). *Digital Signal Processing*. McGraw-Hill, New York.
8. Munir, Rinaldi. 2006. *Pengolaan Citra Digital*. Bandung: Informatika.
9. Putra, Darma. 2010. *Pengolahan Citra Digital*. Yogyakarta: Andi.
10. Robi Polikar, *'The Story of Wavelets'*, *In physics and Modern topics in Mechanical and Electrical Engineering*, References Scientific and Eng. Society Press.
11. Polikar, R. (1998). *The Wavelet Tutorial*. *http://www.public.ee-.iastate.edu/*, Durham Computation Center, Iowa State University, 1996.
12. http://anoa5.files.wordpress.com/2010/05/2/jpg
13. http://www.couleur.org/

Lampiran 1. Proses DWT

```
//metod proses DWT
public void doDWT()
{
    //isi log
    parent.tLog.Items.Add(DateTime.Now.ToString()+" : Initializing DWT Starting"
);
        imageDWT = new YCrCb[Gambar.Height, Gambar.Width];
        double Yv, Cr, Cb;
        for (int y = 0; y < Gambar.Height; ++y)
        {
            for (int x = 0; x < Gambar.Width; ++x)
            {

                Yv = Math.Pow(2, -y / 2) * imageYCbCr[y, x].GetY() * Omega *
(Math.Pow(2, -y) * (imageYCbCr[y, x].GetY() - x));
                Cr = Math.Pow(2, -y / 2) * imageYCbCr[y, x].GetCr() * Omega *
(Math.Pow(2, -y) * (imageYCbCr[y, x].GetCr() - x));
                Cb = Math.Pow(2, -y / 2) * imageYCbCr[y, x].GetCb() * Omega *
(Math.Pow(2, -y) * (imageYCbCr[y, x].GetCb() - x));

                //init pixel dwt baru
                imageDWT[y, x] = new YCrCb(Yv,Cr,Cb);
            }
        }
        parent.tLog.Items.Add(DateTime.Now.ToString() + " : Initializing DWT
Complete");

}
```

Lampiran 2. Proses *Filtering*

```
public  void Filtering()
{
    //init array batas frekuensi tinggi dan rendah
    ArrayList aLL = new ArrayList();
    ArrayList aLH = new ArrayList();
    ArrayList aHL = new ArrayList();
    ArrayList aHH = new ArrayList();
    //isi log
    parent.tLog.Items.Add(DateTime.Now.ToString() + " : Getting Range HL, LH");
    double LLv=0, LHv=0, HLv=0, HHv=0;
    for (int y = 0; y < Gambar.Height/2; ++y)
    {
        for (int x = 0; x < Gambar.Width/2; ++x)
        {
            LLv = 0;
            LHv = 0;
            HLv = 0;
            HHv = 0;

            for (int i =0 ;i<1;i++){
                // perhitungan posii
                LLv = LLv + (imageDWT[2*y+i, 2*x+i].GetY())/4;
                LHv = LHv + ((imageDWT[2 * y, 2 * x + i].GetY()) / 4) -
((imageDWT[2 * y + 1, 2 * x + i].GetY()) / 4);
                HLv = HLv + ((imageDWT[2 * y + i, 2 * x ].GetY()) / 4) -
((imageDWT[2 * y + i, 2 * x + 1].GetY()) / 4);
                HHv = HHv + ((imageDWT[2 * y   , 2 * x].GetY()    +  imageDWT[2 *
y + 1, 2 * x + 1].GetY() - imageDWT[2 * y   , 2 * x + 1].GetY()    +  imageDWT[2
* y + i, 2 * x].GetY() )/4);

            }
            aLL.Add(LLv);
            aLH.Add(LHv);
            aHL.Add(HLv);
            aHH.Add(HHv);
        }
    }
    aLL.Sort();
    aLH.Sort();
    aHL.Sort();
    aHH.Sort();

    //mendapatkan range frekuensi
    LL = (double) aLL[0];
    HL = (double) aHL[aHL.Count - 1];
    LH = (double) aLH[0];
    HH = (double) aHH[0];
    aLL.Clear();
    aLH.Clear();
    aHL.Clear();
    aHH.Clear();

    for (int y = 0; y < Gambar.Height / 2; ++y)
    {
        for (int x = 0; x < Gambar.Width / 2; ++x)
```

```
        {
            LLv = 0;
            LHv = 0;
            HLv = 0;
            HHv = 0;

            for (int i = 0; i < 1; i++)
            {
                LLv = LLv + (imageDWT[2 * y + i, 2 * x + i].GetCr()) / 4;
                LHv = LHv + ((imageDWT[2 * y, 2 * x + i].GetCr()) / 4) -
((imageDWT[2 * y + 1, 2 * x + i].GetCr()) / 4);
                HLv = HLv + ((imageDWT[2 * y + i, 2 * x].GetCr()) / 4) -
((imageDWT[2 * y + i, 2 * x + 1].GetCr()) / 4);
                HHv = HHv + ((imageDWT[2 * y, 2 * x].GetCr() + imageDWT[2 * y +
1, 2 * x + 1].GetCr() - imageDWT[2 * y, 2 * x + 1].GetCr() + imageDWT[2 * y + i,
2 * x].GetCr()) / 4);

            }
            aLL.Add(LLv);
            aLH.Add(LHv);
            aHL.Add(HLv);
            aHH.Add(HHv);
        }
    }
    aLL.Sort();
    aLH.Sort();
    aHL.Sort();
    aHH.Sort();
    LL1 = (double)aLL[0];
    HL1 = (double)aHL[aHL.Count - 1];
    LH1 = (double)aLH[0];
    HH1 = (double)aHH[aHH.Count - 1];

    aLL.Clear();
    aLH.Clear();
    aHL.Clear();
    aHH.Clear();

    for (int y = 0; y < Gambar.Height / 2; ++y)
    {
        for (int x = 0; x < Gambar.Width / 2; ++x)
        {
            LLv = 0;
            LHv = 0;
            HLv = 0;
            HHv = 0;

            for (int i = 0; i < 1; i++)
            {
                LLv = LLv + (imageDWT[2 * y + i, 2 * x + i].GetCb()) / 4;
                LHv = LHv + ((imageDWT[2 * y, 2 * x + i].GetCb()) / 4) -
((imageDWT[2 * y + 1, 2 * x + i].GetCb()) / 4);
                HLv = HLv + ((imageDWT[2 * y + i, 2 * x].GetCb()) / 4) -
((imageDWT[2 * y + i, 2 * x + 1].GetCb()) / 4);
                HHv = HHv + ((imageDWT[2 * y, 2 * x].GetCb() + imageDWT[2 * y +
1, 2 * x + 1].GetCb() - imageDWT[2 * y, 2 * x + 1].GetCb() + imageDWT[2 * y + i,
2 * x].GetCb()) / 4);
```

```
                }
                aLL.Add(LLv);
                aLH.Add(LHv);
                aHL.Add(HLv);
                aHH.Add(HHv);
            }
        }
        aLL.Sort();
        aLH.Sort();
        aHL.Sort();
        aHH.Sort();
        LL2 = (double)aLL[0];
        HL2 = (double)aHL[aHL.Count - 1];
        LH2 = (double)aLH[0];
        HH2 = (double)aHH[aHH.Count - 1];
    }
```

Lampiran 3. Proses Penyisipan Pesan

```
//metod embbed data

public  bool embbed()
{
    double Yv, Cr, Cb;
    int currIndex=0;
    FileInfo fi = new FileInfo(nama);
    DirectoryInfo di = fi.Directory;
    fi = new FileInfo(nama);
    if (fi.Exists == true)
        fi.Delete();
    parent.tLog.Items.Add(DateTime.Now.ToString() + " : Starting Embbed Data");
    StreamWriter sw = new StreamWriter(nama, true, Encoding.ASCII);

    for (int y = 0; y < Gambar.Height; ++y)
    {
        for (int x = 0; x < Gambar.Width; ++x)
        {
            if (currIndex < dataMsg.Length)
            {
                // periksa filter tinggi dan rendah
                if ( (x > Gambar.Width / 2 && (y > 0 && y < Gambar.Height / 2))
|| (y > Gambar.Height / 2 && (x > 0 && x < Gambar.Width / 2)) ||
((x > Gambar.Width / 4 && x < Gambar.Width / 2) && (y > 0 && y < Gambar.Height /
4)) ||
((x > 0 && x < Gambar.Width / 4) && (y > Gambar.Height / 4 && y < Gambar.Height
/ 2)) )
                {
                    string strKof =
Convert.ToString(dataMsg[currIndex]/alpha)+"|";
                    Console.Write(imageDWT[y, x].GetY() + ":" + imageDWT[y,
x].GetCr() + ":" + imageDWT[y, x].GetCb());
                    Yv = imageDWT[y, x].GetY() + (dataMsg[currIndex] * alpha);
                    if (currIndex + 1 < dataMsg.Length)
                    {
                    Cr = (double)imageDWT[y, x].GetCr() + (dataMsg[currIndex +
1] * alpha);
                    strKof += Convert.ToString(dataMsg[currIndex + 1] / alpha) +
"|";
                    }
                    else
                    {
                        Cr = imageDWT[y, x].GetCr();
                        strKof += Convert.ToString(alpha) + "|";
                    }
                    if (currIndex + 2 < dataMsg.Length)
                    {
                        Cb = (double)imageDWT[y, x].GetCb() + (dataMsg[currIndex
+ 2] * alpha);
                        strKof += Convert.ToString(dataMsg[currIndex + 2] /
alpha);
                    }
                    else
                    {
                        Cb = imageDWT[y, x].GetCb();
```

```
                        strKof += (alpha)  ;
                    }
                    imageDWT[y, x] = new YCrCb(Yv, Cr, Cb);
                    Console.WriteLine(" -->>> " + imageDWT[y, x].GetY() + ":" +
imageDWT[y, x].GetCr() + ":" + imageDWT[y, x].GetCb());
                    sw.WriteLine(strKof );
                    currIndex += 3;
                }
            }
        }
    }
    sw.Close();

    parent.tLog.Items.Add(DateTime.Now.ToString() +" : Emmbeding data
complete");

    if (currIndex < dataMsg.Length)
    {
        return false;
    }
    else
    {
        return true;
    }
}
```

Lampiran 4. Proses Ekstraksi Pesan

```
//metod ekstrak
StringBuilder sbResult = new StringBuilder();
public bool extract()
{

    parent.tLog.Items.Add(DateTime.Now.ToString() + " : Starting Ekstract
Data");
    StreamReader sR = new StreamReader(nama,  Encoding.ASCII,true);
    dataMsg = new byte[1000];
    for (int y = 0; y < dataMsg.Length; ++y)
    {
        dataMsg[y] = 0;
    }
    msgResult = new byte[dataMsg.Length];
    sbResult = new StringBuilder();
    for (int y = 0; y < Gambar.Height; ++y)
    {
        for (int x = 0; x < Gambar.Width; ++x)
        {
            if ( sR.Peek() >= 0)
            {
                //periksa range tinggi dan rendah

                if ((x > Gambar.Width / 2 && (y > 0 && y < Gambar.Height / 2))
|| (y > Gambar.Height / 2 && (x > 0 && x < Gambar.Width / 2)) ||
((x > Gambar.Width / 4 && x < Gambar.Width / 2) && (y > 0 && y < Gambar.Height /
4)) ||
((x > 0 && x < Gambar.Width / 4) && (y > Gambar.Height / 4 && y < Gambar.Height
/ 2)))
                {
                    //mengambil isi dataMsg
                    string datas = sR.ReadLine();
                    string[] Arrhasil = datas.Split('|');
                    double yx , cr, cb;
                    yx = double.Parse(Arrhasil[0]);
                    cr = double.Parse(Arrhasil[1]);
                    cb = double.Parse(Arrhasil[2]);

                    Console.WriteLine("  --------> " + imageDWT[y, x].GetY() +
":" + imageDWT[y, x].GetCr() + ":" + imageDWT[y, x].GetCb());

                    double tmpVal = 0;
                    if (imageDWT[y, x].GetY() - (yx/alpha) < Double.MaxValue  )
                    {
                      tmpVal = Math.Round((imageDWT[y, x].GetY() - yx) / alpha);

                      sbResult.Append((char) Math.Round(yx * alpha));
                    }
                    if (imageDWT[y, x].GetCr() - cr < Double.MaxValue)
                    {
                        tmpVal = Math.Round((imageDWT[y, x].GetCr() - cr) /
alpha);

                        sbResult.Append((char)Math.Round(cr * alpha));
                    }
                    if (imageDWT[y, x].GetCb() - cb < Double.MaxValue)
```

```
                    {
                        tmpVal = Math.Round((imageDWT[y, x].GetCb() - cb) /
alpha);
                        sbResult.Append((char)Math.Round(cb * alpha));
                    }
                }
            }
        }
    }
    Console.WriteLine("Hasil : " + sbResult.ToString());
    sR.Close();
    parent.tLog.Items.Add(DateTime.Now.ToString() + " : Complete Extract data");
    return true;
}
public string GetResult()
{
   parent.tLog.Items.Add(DateTime.Now.ToString() + " : Getting Result
Messages");
   string res = "";
   for (int i = 0; i < msgResult.Length; i++)
   {
      res +=  (char) msgResult[i];
   }
   return sbResult.ToString();
}
```

Lampiran 5. Proses IDWT

```
//metode invers dwt
public void IDWT()
{
    parent.tLog.Items.Add(DateTime.Now.ToString() + " : Starting IDWT");
    double Yv=0, Cr=0, Cb=0;
    for (int y = 0; y < Gambar.Height; ++y)
    {
        for (int x = 0; x < Gambar.Width; ++x)
        {
            //memeriksa nilai frekuensi
            if (imageDWT[y, x].GetY() == 0 || imageDWT[y, x].GetCr() == 0 ||
imageDWT[y, x].GetCb() == 0)
            {
                //mengisi dengan nilai awal cover
                Yv = imageYCbCr[y, x].GetY();
                Cr = imageYCbCr[y, x].GetCr();
                Cb = imageYCbCr[y, x].GetCb();
            }
            else
            {
                //mengisi dengan fungsi invers
                Yv = imageDWT[y, x].GetY() / (Omega * (Math.Pow(2, -y) *
(imageYCbCr[y, x].GetY() - x)) * Math.Pow(2, -y / 2));
                Cr = imageDWT[y, x].GetCr() / (Omega * (Math.Pow(2, -y) *
(imageYCbCr[y, x].GetCr() - x)) * Math.Pow(2, -y / 2));
                Cb = imageDWT[y, x].GetCb() / (Omega * (Math.Pow(2, -y) *
(imageYCbCr[y, x].GetCb() - x)) * Math.Pow(2, -y / 2));
            }

                imageDWT[y, x] = new YCrCb(Yv, Cr, Cb);
            }

    }
    parent.tLog.Items.Add(DateTime.Now.ToString() + " : IDWT Complete");
}
```