WATERMARKING DENGAN METODE WAVELET PADA OBJEK CITRA DIGITAL

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat
Untuk Memperoleh Gelar sarjana
Jurusan Teknik Informatika

oleh:

Nama : Firmansyah
No. Mahasiswa : 03 523 205

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2007

LEMBAR PENGESAHAN PEMBIMBING

WATERMARKING DENGAN METODE WAVELET PADA OBJEK CITRA DIGITAL

TUGAS AKHIR

oleh:

Nama : **Firmansyah**
No. Mahasiswa : **03 523 205**

Yogyakarta, April 2007

Pembimbing,

Taufiq Hidayat, S.T., MCS.

LEMBAR PENGESAHAN PENGUJI

WATERMARKING DENGAN METODE WAVELET PADA OBJEK CITRA DIGITAL

TUGAS AKHIR

oleh:

Nama : Firmansyah

No. Mahasiswa : 03 523 205

Telah Dipertahankan di Depan Sidang Penguji Sebagai Salah Satu Syarat untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, April 2007

Tim Penguji

Taufiq Hidayat, S.T.,MCS.
Ketua

Syarif Hidayat, S.Kom.
Anggota I

Hendrik, S.T.
Anggota II

Mengetahui,
Ketua Jurusan Teknik Informatika
Universitas Islam Indonesia

Yudi Prayudi, S.Si., M.Kom.

Karya kecil ini Aku persembahkan dengan segenap cinta untuk:

Ayah Bundaku tercinta. Terima kasih atas kasih sayang, doa, nasehat, semangat, motivasi, dan pengorbanan yang tidak akan pernah terbalaskan.

Ketiga saudaraku. Terima kasih atas cinta dan support dari kalian.

Soulmateku, terima kasih telah menjadi teman setia disetiap langkah

Sahabat-sahabatku. Terima kasih atas persahabatan yang indah selama ini.

MOTTO

You can if you think you can

Do all the goods you can. All the best you can. In all times you can. In all places you can. For all the creatures you can

Niscaya Allah akan meninggikan orang-orang yang beriman di antara kamu dan orang-orang yang diberi ilmu pengetahuan beberapa derajat (Al-Qur'an, Surat Al-Mujadilah:11)

Kemuliaan seseorang adalah agamanya, harga dirinya (kehormatannya) adalah akalnya, sedangkan ketinggian kedudukannya adalah akhlaknya (HR. Ahmad dan Al-Hakim)

Jenius adalah 1 % inspirasi dan 99 % keringat. Tidak ada yang dapat menggantikan kerja keras. Keberuntungan adalah sesuatu yang terjadi ketika kesempatan bertemu dengan kesiapan

Langkah pertama mencapai keberhasilan adalah melakukan suatu perkerjaan kecil dengan sebaik-baiknya dan dengan cara yang benar, hingga keberhasilan dapat tercapai. Setelah itu lakukanlah pada hal-hal yang lebih besar

Agar dapat membahagiakan seseorang, isilah tangannya dengan kerja, hatinya dengan kasih sayang, pikirannya dengan tujuan, ingatannya dengan ilmu yang bermanfaat, masa depannya dengan harapan, dan perutnya dengan makanan

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Puji dan syukur saya panjatkan kehadirat Allah SWT, karena atas berkat rahmat dan hidayah-Nya lah saya dapat menyelesaikan penulisan karya ilmiah ini. Shalawat dan salam semoga senantiasa tercurah kepada junjungan kita Nabi Muhammad SAW, teladan kebaikan dan kemuliaan sepanjang jaman dan seluruh alam.

Tidak lupa juga saya ingin mengucapkan terima kasih yang sebesar-besarnya kepada pihak-pihak yang telah membantu penyusunan karya ilmiah ini, yang antara lain:

1. Bpk. Fathul Wahid, S.T., M.Sc., selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
2. Bpk. Yudi Prayudi, S.Si., M.Kom., selaku Ketua Jurusan Teknik Informatika Universitas Islam Indonesia.
3. Bpk. Taufiq Hidayat, S.T., M.CS., selaku dosen pembimbing. Terima kasih telah mengarahkan dan membimbing saya dengan penuh kesabaran selama penulisan tugas akhir ini.
4. Orangtua-ku tercinta, H. Dalin Muhammad dan Hj. Suharti beserta ketiga saudaraku. Kepada mereka saya ucapkan terima kasih yang setulus-tulusnya, sedalam-dalamnya dan seikhlas-ikhlasnya. Ya Allah, limpahkan-lah rahmat-Mu kepada mereka mereka, di dunia ini dan di akhirat nanti. Amin.
5. Keluarga besar Pak Cik Arpi Gusmadi, S.T., terima kasih atas dukungan yang tiada henti.
6. Keluarga besar Devy Yurista Nurdiantika, S.T., thanks untuk cinta, do'a dan hal-hal yang kalian lakukan untuk mendorongku menjadi lebih baik dari hari ke hari.

Saya menyadari bahwa karya ilmiah ini masih jauh dari sempurna. Oleh karena itu, segala kritik dan saran yang sifatnya membangun akan selalu saya harapkan.

Demikianlah yang dapat saya sampaikan. Semoga karya ilmiah ini dapat bermanfaat bagi penulis khususnya dan dapat dijadikan referensi untuk melakukan penelitian sejenis, demi kemajuan ilmu pengetahuan umumnya.

Yogyakarta, April 2007

Penulis,

Firmansyah

SARI

Tugas akhir ini bertujuan untuk merancang dan membuat aplikasi *watermarking* sebagai teknik untuk menyembunyikan informasi ke dalam sebuah file citra digital.

Metode *wavelet* digunakan untuk menentukan frekuensi sinyal citra digital yang bisa dijadikan tempat untuk menyisipkan informasi. Fungsi skala dan fungsi *wavelet* digunakan sebagai koefisien matrik filter dalam proses transformasi. Hasil perkalian matrik filter dengan matrik citra akan menghasilkan matrik transformasi dengan empat band frekuensi, yaitu: HH (*High column-High row*), HL (*High column- Low row*), LH (*Low column-High row*) dan LL (*Low column-Low row*). Komponen LL merupakan komponen frekuensi yang paling rendah dan menyatakan konfigurasi objek. Sedangkan tiga komponen lainnya (HH, HL, LH) menyatakan detail objek. Informasi (*watermark*) akan disisipkan pada komponen frekuensi HH karena komponen ini merupakan komponen yang paling tidak sigifikan jika dibandingkan dengan ke 3 komponen frekuensi lainya. .

Hasil pengujian tugas akhir ini menunjukkan bahwa, informasi (*watermark*) dapat disisipkan dengan baik pada file citra digital tanpa mengubah tampilan citra asli secara signifikan. Pengujian dalam tugas akhir ini dilakukan sebanyak dua kali pada file objek citra digital yang berbeda, dan didapatkan hasil sesuai dengan yang diharapkan

Kata kunci : *Watermarking, Wavelet,* citra digital.

DAFTAR ISI

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan pesat pada teknologi digital membuat pengembangan pola perlindungan media *digital* terhadap pembajakan menjadi suatu hal yang penting. Representasi digital suatu media, di satu sisi mempermudah akses serta berpotensi meningkatkan protabilitas, efisiensi dan keakuratan informasi yang dikemukan. Namun di sisi lain, juga memudahkan terjadinya pembajakan. Pembajakan meliputi akses ilegal terhadap data yang dikirim melalui jaringan, modifikasi isi data serta penyebarluasan salinan-salinan yang tidak sah.

Transfer data melalui jaringan memang diakui memudahkan pekerjaan dan memiliki efisiensi yang tinggi. Namun demikian tetap saja ada kekurangan. Transfer data semacam itu sangat rentan terhadap penyadapan yang dilakukan oleh pihak-pihak ketiga. Jika data yang ditransfer tidak bersifat rahasia mungkin hal ini tidak begitu diperhatikan. Namun jika kerahasiaan data yang ditransfer sangat penting maka diperlukan perlindungan atau proteksi khusus pada data tesebut dari pengaksesan oleh pihak-pihak ketiga. Maka dari itu dikembangkanlah berbagai teknik proteksi data, termasuk di dalamnya adalah *watermarking.*

Watermarking adalah suatu teknik penyisipan informasi ke dalam data multimedia, dimana informasi tersebut dapat berupa data-data teks, video, audio ataupun citra yang mengidentifikasikan kepemilikan suatu pihak. *Watermarking* sendiri

merupakan penerapan dari ilmu Steganografi yaitu ilmu yang mempelajari bagaimana menyembunyikan data ke dalam data yang lain. Dengan menggunakan metode ini diharapkan informasi yang mendandakan kepemilikan dapat disisipkan pada data yang akan di transfer tanpa diketahui keberadaanya. Sehingga apabila terjadi pembajakan maka pemilik data asli dapat menunjukan bahwa data yang dimiliki oleh pihak ketiga tersebut bukan merupakan data asli.

Banyak metode yang dapat digunakan dalam pembuatan *watermarking* pada sebuah data digital dan salah satunya adalah dengan menggunakan teknik transformasi. Namun dalam penelitian tugas akhir ini penulis menggunakan metode *wavelet* yang merupakan bagian dari teknik transformasi dalam membuat *watermark* pada objek citra *digital.*

1.2 Rumusan Masalah

Bagaimana membangun sebuah perangkat lunak aplikasi untuk menyisipkan informasi (*watermark*) ke dalam file citra digital dengan menggunakan metode transformasi *wavelet*, agar file citra digital tersebut dapat terjaga keasliannya.

1.3 Batasan Masalah

Batasan-batasan masalah yang penulis buat dalam pembuatan tugas akhir ini adalah:

1. Skripsi ini menitik beratkan pada pembuatan sebuah aplikasi *watermarking* dengan menggunakan metode *wavelet* (transformasi *wavelet* Haar).

2. File citra sampul dan file citra *watermark* adalah file citra digital berjenis bitmap dengan format BMP 24 bit.
3. Ukuran dari file citra *watermark* harus lebih kecil dari file citra sampul. Pada penulisan tugas akhir ini, ukuran file citra *watermark* dibatasi yaitu 128x128 pixel dan ukuran file citra sampul 800 x 600.
4. Proses ekstrak *watermark* (*decode*) hanya sebatas menampilkan watermark yang ada dalam citra sampul, tidak untuk memisahkan watermark dengan citra sampul.
5. File citra ter-*watermark* tidak dapat dikembalikan lagi kebentuk asal setelah disisipi file citra *watermark*.

1.4 Tujuan Penelitian

Tujuan yang ingin dicapai adalah membuat aplikasi yang mampu:

1. Menjabarkan dan menjelaskan tentang dasar-dasar *watermarking* pada file citra, teknik yang digunakan serta aplikasinya.
2. Menyisipkan informasi (*watermark*) ke dalam suatu objek citra digital.
3. Mengecek keaslian dari objek citra digital setelah disisipi *watermark* yang menggunakan metode transformasi *wavelet*.

1.5 Manfaat Penelitian

Manfaat yang ingin dicapai dalam pembuatan tugas akhir ini adalah:

1. Bagi penulis, dapat memahami tentang teknik *watermarking* dengan menggunakan metode tansformasi *wavelet* Haar sebagai salah satu teknik yang dapat digunakan dalam pengamanan data.

2. Bagi pengguna citra digital aplikasi ini dapat digunakan untuk menjaga keaslian data citra digital yang dimiliki.
3. Bagi kalangan akademis skirpsi ini di harapkan dapat memberikan wawasan tentang penggunaan metode tansformasi *wavelet* khususnya transformasi wavelet Haar.

1.6 Metodologi Penelitian

Metodologi yang digunakan dalam pembuatan tugas akhir ini adalah:

1. Pengumpulan data melalui studi literatur dengan mempelajari materi yang diperoleh berkaitan dengan topik tugas akhir melalui media seperti buku, artikel, tulisan-tulisan pada situs internet maupun media informasi lainnya, serta berkonsultasi dengan narasumber yang lebih mendalami topik yang berkaitan.
2. Pengembangan perangkat lunak, yaitu:
   a. Analisis kebutuhan yang digunakan untuk mengetahui dan menerjemahkan pemasalahan dan kebutuhan perangkat lunak. Metode analisis yang akan digunakan adalah metode analisis dengan pendekatan terstuktur yang lengkap dengan teknik yang dibutuhkan dalam pengembanagan sistem.
   b. Perancangan sistem, terdiri dari beberapa rancangan yaitu perencanaan antarmuka (*interface*) untuk memberikan gambaran tentang perangkat lunak yang dibuat serta pemodelan sistem. Serta implementasi hasil perancangan dan analisi kerja yang dilakukan untuk menguji aplikasi yang di kembangkan.

1.7 Sistematis Penulisan

Untuk memudahkan penulisan tugas akhir agar lebih terperinci, maka dibuat sistematika penulisan sebagai berikut:

Bab I Pendahuluan

Berisi Latar belakang, rumusan maslah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan.

Bab II Landasan Teori

Berisi teori tentang watermarking dan metode-metode yang dapat digunakan untuk membuat aplikasi watermarking tersebut.

Bab III Metodologi

Memuat uraian tentang metode perancangan perangkat lunak yang dipakai dalam sistem yang akan dibuat dan hasil perancangan perangkat lunak yang merupakan terjemahan perangkat lunak yang meliputi pembuatan diagram alir (*flowchar*).

Bab IV Hasil dan Pembahasan

Memuat uraian tentang pengujian dan analisi kinerja perangkat lunak dan asumsi-asumsi yang dipakai seta batasan lain yang dibuat dan ditemui selama pengembangan perangkat lunak. Selain itu memuat dokumen implementasi perangkat lunak, serta dokumentasi hasil pengujian terhadap perangkat lunak yang telah dibuat, meliputi penanganan kesalahan dan pembahasan pengujian pada proses-proses yang ada.

Bab V Penutup

Bagian ini memuat kesimpulan dari proses pengembangan perangkat lunak, baik pada tahap analisis kebutuhan perangkat lunak, perancangan implentasi dan terutama pada analisis kinerja perangkat lunak. Selain itu pada bagian ini juga memuat tentang saran yang perlu diperhatikan untuk pengembangan perangkat lunak tersebut.

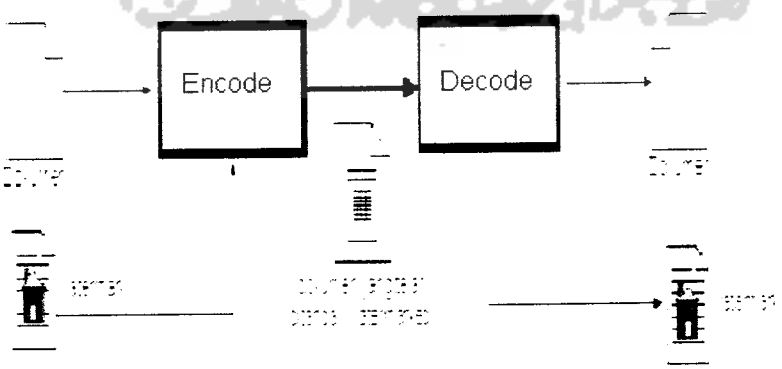
BAB II

LANDASAN TEORI

2.1 *Watermarking*

Secara umum *watermarking* dapat diartikan sebagai suatu teknik untuk menyisipkan informasi ke dalam data multimedia. Informasi yang dapat disisipkan tersebut dapat berupa data citra, audio, ataupun video yang menggambarkan kepemilikan suatu pihak. Informasi yang disisipkan tersebut disebut *watermark*.

Watermarking merupakan ilmu pengembangan dari steganografi, yaitu ilmu yang digunakan untuk menyembunyikan pesan kedalam suatu media. Seiring dengan berkembangnya dunia teknologi digital maka ilmu steganografi tersebut juga mengalami perkembangan dalam media penyimpannya.Dalam *watermarking*, *watermark* dapat dianggap sebagai sidik digital yang bersifat pribadi (rahasia), dengan kata lain *watermark* tersebut hanya diketahui oleh orang-orang yang berhak untuk mengetahuinya. Menurut [DAV02] proses *watermarking* dapat digambarkan dengan gambar 2.1:



Gambar 2.1 Prinsip Kerja *Watermarking*

Teknik *watermarking* ini memanfaatkan kekurangan sistem indera manusia seperti mata dan telinga. Dengan adanya kekurangan inilah, metoda *watermarking* ini dapat diterapkan pada berbagai media digital. Jadi dengan kata lain *watermarking* dapat didefenisikan sebagai suatu cara untuk penyembunyian atau penanaman data/informasi tertentu (baik hanya berupa catatan umum maupun rahasia) ke dalam suatu data digital lainnya, tetapi tidak diketahui kehadirannya oleh indera manusia (indera penglihatan atau indera pendengaran) dan mampu menghadapi proses-proses pengolahan sinyal digital sampai pada tahap tertentu [SUH00].

2.1.1 Karakteristik Digital *Watermarking*

Watermark suatu meteri digital harus memiliki sejumlah karakteristik, antara lain:

1. *Imperceptible*, yaitu memberikan karakteristik *watermark* agar sebisa mungkin harus tidak dapat terlihat atau berbeda dengan dokumen aslinya. Hal ini dimaksudkan untuk tidak merubah status dokumen yang bernilai tinggi secara hukum maupun komersial.
2. *Robustness*, yaitu karakteristik ini tergantung aplikasi dari *watermark* itu sendiri. Apabila digunakan sebagai identifikasi kepemilikan (*copyright*), *watermark* harus memiliki ketahanan terhadap berbagai macam modifikasi yang mungkin bisa dilakukan untuk merubah/menghilangkan *copyright*. Jika digunakan untuk otentikasi *content*, *watermark* sebisa mungkin bersifat *fragile*, sehingga apabila isinya telah mengalami perubahan, maka watermark akan mengalami perubahan/rusak, sehingga dapat terdeteksi adanya usaha modifikasi terhadap isi.
3. *Security*, yaitu teknik *watermark* harus dapat mencegah usaha-usaha untuk mendeteksi dan memodifikasi informasi *watermark* yang disisipkan ke dalam dokumen. Kunci *watermark* menjamin hanya orang yang berhak saja yang dapat

melakukan hal tersebut. Namun aspek ini tidak dapat mencegah siapapun untuk membaca dokumen yang bersangkutan.

2.1.2 Manfaat Aplikasi *Watermarking*

Watermarking sebagai suatu teknik penyembunyian data pada data digital lain dapat dimanfaatkan untuk berbagai tujuan [BEN96] seperti:

1. *Tamper-proofing*, yaitu *watermarking* digunakan sebagai alat untuk mengidentifikasikan atau alat indikator yang menunjukkan data digital *(host)* telah mengalami perubahan dari aslinya.
2. *Feature location*, yaitu penggunaan metoda *watermarking* sebagai alat untuk identifikasikan isi dari data digital pada lokasi-lokasi tertentu, seperti contohnya penamaan objek tertentu dari beberapa objek yang lain pada suatu citra digital.
3. *Annotation/caption*, yaitu *watermarking* hanya digunakan sebagai keterangan tentang data digital itu sendiri.
4. *Copyright-Labeling*, yaitu *watermarking* digunakan sebagai metoda untuk penyembunyikan label hak cipta pada data digital sebagai bukti otentik kepemilikan karya digital tersebut.

2.1.3 Teknik *Watermarking* Pada Objek Citra Digital

Ada beberapa teknik yang bisa digunakan untuk membangun sebuah aplikasi *watermarking*, diantaranya adalah:

1. Least Significant Bit (LSB)

Cara paling umum untuk menyembunyikan pesan adalah dengan memanfaatkan *Least-Significant Bit* (LSB). Walaupun banyak kekurangan pada metode ini tetapi

kemudahan implementasinya membuat metode ini tetap digunakan sampai sekarang. Metode ini membutuhkan syarat, yaitu jika dilakukan kompresi pada stego harus digunakan format *lossless compression* karena metode ini menggunakan bit-bit pada setiap *pixel* pada image. Jika digunakan format *lossy compression* pesan rahasia yang disembunyikan dapat hilang.

Jika digunakan image 24 bit color sebagai *cover*, sebuah bit dari masing-masing komponen *Red, Green* dan *Blue* dapat digunakan sehingga 3 bit dapat disimpan pada setiap *pixel*. Sebuah image 800 x 600 piksel dapat digunakan untuk menyembunyikan 1.440.000 bit (180.000 bytes) data rahasia.

Misalnya, di bawah ini terdapat 3 *pixel* dari image 24 bit color :

(00100111 11101001 11001000)

(00100111 11001000 11101001)

(11001000 00100111 11101001)

jika diinginkan untuk menyembunyikan karakter A (10000001b) dihasilkan :

(00100111 11101000 11001000)

(00100110 11001000 11101000)

(11001000 00100111 11101001)

dapat dilihat bahwa hanya 3 bit saja yang perlu diubah untuk menyembunyikan karakter A ini. Perubahan pada LSB ini akan terlalu kecil untuk terdeteksi oleh mata manusia sehingga pesan dapat disembunyikan secara efektif.

Jika digunakan image 8 bit color sebagai cover hanya 1 bit saja dari setiap *pixel* warna yang dapat dimodifikasi sehingga pemilihan image harus dilakukan dengan sangat hati-hati karena perubahan LSB dapat menyebabkan terjadinya perubahan warna yang

ditampilkan pada citra. Akan lebih baik jika image berupa image *grayscale* karena perubahan warnanya akan lebih sulit dideteksi oleh mata manusia.

Proses ekstraksi pesan dapat dengan mudah dilakukan dengan mengekstrak LSB dari masing-masing *pixel* pada stego secara berurutan dan menuliskan nya ke *output* file yang akan berisi pesan tersebut.

Kekurangan dari metode modifikasi LSB ini adalah bahwa metode ini membutuhkan "tempat penyimpanan" yang relatif besar. Kekurangan lain adalah bahwa stego yang dihasilkan tidak dapat dikompress dengan format *lossy compression.*

2. Masking dan Filtering

Teknik masking dan filtering ini biasanya dibatasi pada image 24 bit color atau image *grayscale*. Hal ini dapat dilakukan misalnya dengan memodifikasi *luminance* beberapa bagian dari image. Walaupun metode ini akan mengubah tampilan dari image, dimungkinkan untuk melakukannya dengan cara tertentu sehingga mata manusia tidak melihat perbedaannya. Karena metode ini menggunakan aspek image yang memang terlihat langsung, metode ini akan lebih *robust* terhadap kompresi (terutama *lossy compression*), *cropping*, dan beberapa image processing lain, bila dibandingkan dengan metode modifikasi LSB.

3. Transformation

Metode transformasi merupakan metode yang lebih kompleks untuk menyembunyikan pesan pada image. Metode ini dapat dibagi menjadi dua teknik lagi yaitu:

a) Discrete Cosine Transformation (DCT)

DCT digunakan, terutama pada kompresi JPEG, untuk mengubah sebuah sinyal menjadi komponen dasarnya. DCT pertama kali diperkenalkan oleh Ahmed, Natarajan

dan Rao pada tahun 1974 dalam makalah yang berjudul " *on image processing and a discrete consine transform*" [WAT94].

DCT digolongkan menjadi 2 yaitu:

1. *Discrete Cosine Transformation* dimensi satu (1-D DCT)

Discrete Cosine Transformation dimensi satu (1-D DCT) biasa digunakan untuk pengolahan sinyal seperti gelombang audio. DCT jenis ini dapat dirumuskan dengan:

$$S(u) = \sqrt{1/n}\ C(u) \sum_{x=0}^{n-1} S(x) \cos\frac{(2x+1)u\pi}{2n} \quad \text{..... (2.1)}$$

dimana n dan u adalah bilangan real dan merupakan variabel,

dan $C(u) = \begin{cases} 2^{1/2}, & untuk \quad u = 0 \\ 1, & untuk \quad lainya \end{cases}$,dengan dengan $u = 0,\ldots,n-1$.

2. *Discrete Cosine Transformation* dimensi dua (2-D DCT)

Discrete Cosine Transformation dimensi dua (2-D DCT) merupakan jenis transformasi untuk pengolahan sinyal berbentuk citra, karana transformasi berjenis ini adalah transformasi berbentuk matrik. DCT dimensi dua dapat dirumuskan dengan:

$$S(u,v) = \frac{2}{\sqrt{nm}} C(u)\ C(v) \sum_{y=0}^{m-1} \sum_{x=0}^{n-1} S(x,y) \cos\frac{(2x+1)u\pi}{2\pi} \quad \text{..... (2.2)}$$

dimana n dan m adalah ordo matrik yang merupakan basis fungsi, dengan $u=0,\ldots,m-1$. Rumus DCT dimensi dua ini biasa disebut dengan *Forward Discrete Consine Transformation.*

b) Wavelet

Wavelet adalah fungsi matematika yang memotong-motong data menjadi komponen frekuensi dan skala yang berbeda-beda, sedangkan Transformasi *Wavelet* adalah dekomposisi suatu sinyal dengan keluarga dari basis ortonormal real $\psi a,b(x)$ yang diperoleh melalui translasi dan dilasi sebuah fungsi kernel $\psi(x)$.

Menurut [HEN06] *wavelet* adalah salah satu cara kompresi data yang cocok digunakan untuk kompresi image, audio, dan video. Tujuannya adalah untuk menyimpan data dalam "ruang" yang sekecil mungkin dalam sebuah file. *Wavelet* juga merupakan sebuah basis. Basis *wavelet* berasal dari sebuah fungsi penskalaan atau dikatakan juga sebuah *scaling function*. *Scaling function* memiliki sifat yaitu dapat disusun dari sejumlah salinan dirinya yang telah didilasikan, ditranslasikan dan diskalakan, dan dengan adanya *Scaling function wavelet* dapat di bentuk dalam bermacam-macam jenisnya. Berdasarkan *scaling function* inilah basis wavelet memiliki nama yang berbeda-beda, diantaranya:

a. *Wavelet* Haar memiliki *scaling function* dengan koefisien $c_0 = c_1 = 1$.
b. *Wavelet* Daubechies dengan 4 koefisien (DB4) memiliki *scaling function* dengan koefisien $c_0 = (1+\sqrt{3})/4$, $c_1 = (3+\sqrt{3})/4$, $c_2 = (3-\sqrt{3})/4$, $c_3 = (1-\sqrt{3})/4$
c. *Wavelet* B-Spline kubik memiliki *scalilng function* dengan koefisien $c_0 = 1/8$, $c_1 = 4/8$, $c_2 = 6/8$, $c_3 = 4/8$, $c_4 = 1/8$.

2.2 Transformasi *Wavelet* Haar

Karena metode yang digunakan dalam membuat aplikasi *watermarking* ini adalah metode transformasi wavelet Haar, maka penulis akan sedikit membahas tentang metode transformasi *wavelet* Haar tersebut.

Transformasi *wavelet* Haar pertama kali diperkenalkan pada tahun 1909 oleh Alfred Haar dan merupakan tranformasi *wavelet* yang paling sederhana. Transformasi ini menggunakan fungsi skala atau biasa disebut *Father Wavelet* ($\phi(x)$) dan fungsi *wavelet* atau biasa disebut *Mather Wavelet* ($\psi(x)$). Fungsi skala dapat dinyatakan dengan:

$$\phi(x) = \begin{cases} 1, & 0 \le x < 1 \\ 0, & otherwise \end{cases}$$

sedangkan fungsi *wavelet* dapat dinyatakan dengan:

$$\psi(x) \begin{cases} 1, & 0 \le x < 0{,}5 \\ -1, & 0{,}5 \le x < 1 \\ 0, & otherwise \end{cases}$$

dengan menggunakan fungsi skala dan fungsi wavelet, maka suatu fungsi ($F(x)$) dapat dinyatakan dengan menggunakan rumus:

$$F(x) = \sum_{k=0}^{\infty}\sum_{j=0}^{\infty} C_{j,k} \quad \phi(2^j x - k) + \sum_{k=-\infty}^{\infty}\sum_{j=0}^{\infty} d_{j,k} \quad \psi(2^j x - k) \qquad \text{.....(2.3)}$$

atau yang lebih umum

$$F(x) = \sum_{k=0}^{2^j-1}\sum_{j=0}^{\infty} C_{j,k} \quad \phi_k^j(x) + \sum_{k=0}^{2^j-1}\sum_{j=0}^{\infty} d_{j,k}\psi_k^j(x) \qquad \text{.....(2.4)}$$

dimana, :

$$\phi_k^j(x) = \phi(2^j x - k); \quad \psi_k^j(x) = \psi(2^j x - k)$$

$$j = 0,1,...; \quad k = 0,1,...2^j - 1$$

$C_{j,k}$ dan $d_{j,k}$ adalah koefisien-koefisien *wavelet* yang akan dihitung. Indek j menyatakan skala, sedangkan k menyatakan translasi.

Selain dinyatakan sebagai bentuk fungsi, transformasi *wavelet* Haar dapat juga dinyatakan dalam bentuk matrik. Dimana matrik tersebut memiliki 2 koefisien yaitu C_0 dan C_1, sehingga matriknya berbentuk $\begin{bmatrix} C_0 & C_1 \\ C_1 & -C_0 \end{bmatrix}$ dimana koefisien $C_0 = 1$, $C_1 = 1$.

Transformasi *wavelet* Haar dari suatu fungsi di implementasikan dengan melakukan filter *low pass* dan *high pass*. Untuk suatu sinyal B_t maka transformasi

wavelet Haar dari suatu sinyal di dapat dengan mengalikan matrik basis Haar (filter) dengan sinyal B_t :

$$\begin{bmatrix} m_0 \\ w_0 \\ m_1 \\ w_1 \\ m_2 \\ w_2 \\ m_3 \\ w_3 \end{bmatrix} = \begin{bmatrix} c_0 & c_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ c_0 & -c_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_0 & c_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_0 & -c_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_0 & c_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_0 & -c_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_0 & c_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_0 & -c_1 \end{bmatrix} \begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \\ B_4 \\ B_5 \\ B_6 \\ B_7 \end{bmatrix} \quad \text{.....(2.5)}$$

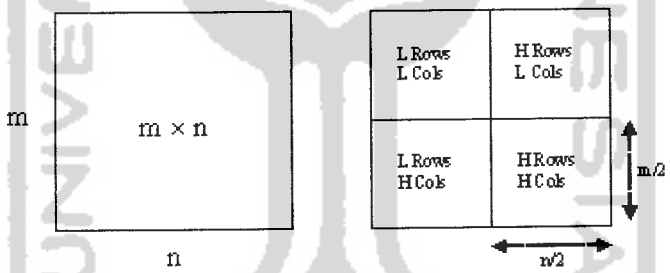
dimana m_t, w_t merupakan koefisien-koefisien hasil transformasi dengan m_t menyatakan koefisien berfrekuensi rendah dan w_t menyatakan koefisien berfrekuensi tinggi.

2.3 *Watermarking* Dengan Metode Transformasi *Wavelet* Haar

Watermarking berbasis *wavelet* adalah suatu ide baru dalam penerapan *wavelet*. Beberapa usulan telah dikemukakan beberapa peneliti dibidang steganografi untuk menerapkan transformasi yang satu ini kedalam *watermarking*. Sebagai catatan, beberapa transformasi selain *wavelet* sudah ada yang dipakai untuk *watermarking* diantaranya *fast Fourier Transform* (FFT) dan *Discrete Consine Transform* (DCT). Seperti diketahui *wavelet* adalah penerapan dari teori transformasi, dengan menggunakan metode ini pesan (*watermark*) dapat di sembunyikan pada objek citra digital dengan mengkompresi data yang bertujuan untuk memberikan ruang yang sekecil mungkin agar dapat disisipkan pesan (*watermark*). Seperti sudah dijelaskan sebelumnya, dampak dari pengkompresian data tersebut akan menghilangkan sebagian kecil dari data, dimana data tersebut akan sedikit mengalami perubahan yang tidak terlau signifikan atau bahkan tidak tampak jika dipandang dengan mata telanjang.

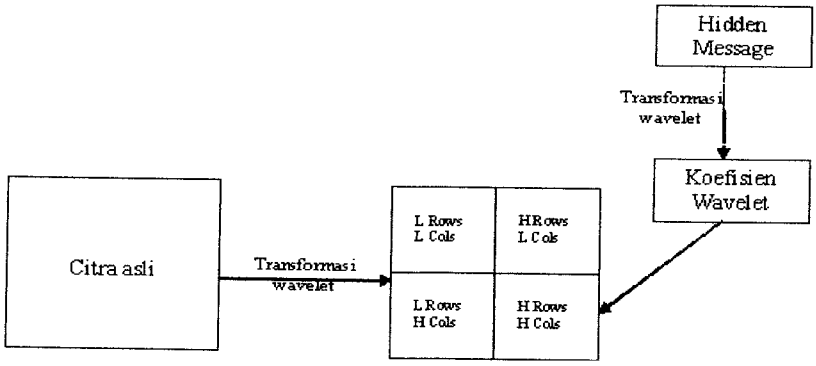
2.3.1 Penerapan transformasi *Wavelet* dalam aplikasi *Watermarking*

Penerapan transformasi *wavelet* dalam pembuatan *watermarking* akan mengubah komponen citra dari domain spasial ke domain frekuensi. Proses dasarnya meliputi *lowpass filter* (L[n]) dan *Highpass filter* (H[n]). Mula-mula proses dilakukan sepanjang baris dari citra, sehingga setiap baris dari citra akan terbagi menjadi bagian L (*low*) yang berfrekuensi rendah dan bagian H (*high*) yang berfrekuensi tinggi. Kemudian proses dilanjutkan sepanjang kolom dari citra sehingga setiap kolom dari citra akan terbagi menjadi bagian L (*low*) yang berfrekuensi rendah dan bagian H (high) yang berfrekuensi tinggi. Pada akhirnya citra akan terbagi menjadi 4 band frekuensi, yaitu :LL,HL,LH,HH, seperti terlihat pada gambar berikut, dimana L menyatakan frekuensi rendah dan H menyatakan frekuensi tinggi.



Gambar 2.2 citra asli (kiri) citra hasik transformasi (kanan)

Komponen LL merupakan komponen frekuensi yang paling rendah yang menyatakan konfigurasi objek. Sedangkan 3 kompnen lainya (LH,HL,HH) menyatakan detail objek. Komponen HH merupakan komponen yang paling tidak signifikan jika di bandingkan dengan ke 3 komponen frekuensi yang lain. Karena itu informasi dari pesan yang akan di sembunyikan disisipkan pada komponen ini, sepeti terlihat pada gambar dibawah ini.



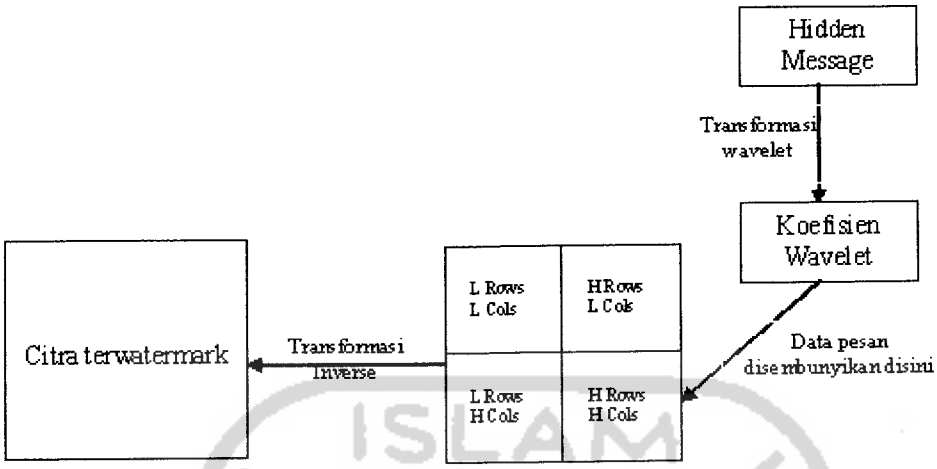
Gambar 2.3 Proses Penyisipan pesan ke dalam File Citra

Perubahan pada komponen HH tidak akan mempengaruhi atau memberi perubahan yang berarti pada citra.

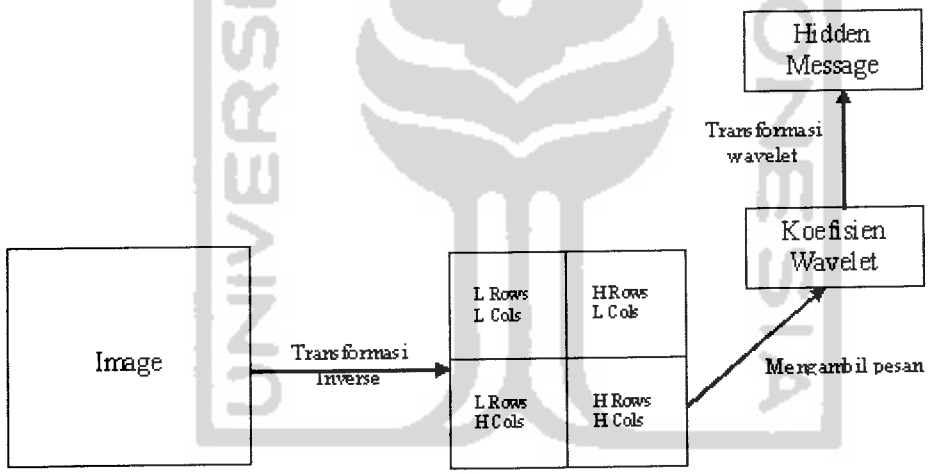
Setelah tanda disisipkan, maka perlu di lakukan teransformasi inversnya untuk mendapatkan citra ber-*watermark* (citra yang telah disisipi pesan). Transformasi invers dapat diperoleh dengan mengalikan invers dari matrik filter dengan koefisien-koefisien *lowpass/highpass* yang bersesuaian, sehingga akan diperoleh vektor sinyal B semula. Invers dari matrik filter adalah matrik transposenya sendiri.

2.3.2 *Extracting data*

Setelah pesan di sisipkan, maka diperoleh citra ber-*watermark* dengan pesan rahasia di dalamnya. Untuk mengetahui model pesan tersebut, maka data pesan tersebut perlu diextrak atau dikodekan. Mula-mula citra ber-*watermark* ditransformasi wavelet-kan lagi, sehingga diperoleh 4 komponen frekuensi LL,HL,LH dan HH. Pada komponen HH data pesan dapat diambil karena sebelumnya pesan disisipi pada bagian ini. Setelah diperoleh data pesan, kemudian dilakukan transformasi inversenya sehingga akan di peroleh citra pesan yang asli.



Gambar 2.4 Proses tranformasi Invers



Gambar 2.5 Proses pendeteksian *watermark*

2.4 Kelebihan dan Kekurangan Transformasi *Wavelet*

Keuntungan penggunaan teori transformasi *wavelet* dalam pembuatan aplikasi *watermarking* adalah kemampuannya yang lebih baik dalam merepresentasikan daerah transien. Artinya, elemen dari data yang transien akan direpresentasikan dalam jumlah informasi yang lebih kecil dari pada yang terjadi pada transformasi lain. Sedangkan

kerugiannya adalah kurang baik digunakan pada data yang bersifat periodik dan *smooth*, selain itu dengan menggunakan metode *wavelet* dalam penyisipan informasi dapat menghilangkan informasi tertentu dalam objek citra yang akan dipasang *watermark*.

2.5 Jenis-jenis *Data Carrier* Digital

Pada *digital watermarking*, terdapat beberapa jenis *digital carrier* atau media digital yang dapat digunakan untuk menyembunyikan pesan, diantaranya file citra, file audio, teks video dan transmisi. Pada file citra pesan dapat disembunyikan dengan melakukan manipulasi pada properti citra seperti luminasi, kontras dan komponen warna citra tersebut. Pada file audio, suatu gema atau gaung *(echo)* yang berukuran kecil dan *slight delay* dapat dimasukan kedalam suatu suara *(sound)* dengan amplitudo yang lebih besar. Suatu pesan juga dapat disembunyikan pada suatu dokumen dengan memanipulasi posisi garis atau kata-kata dari dokumen tersebut.

Jika menggunakan citra sebagai media *(data carrier)*, maka selanjutnya dalam proses *watermarking* data carrier disebut sebagai citra sampul *(cover image)*. Sebuah citra sampul *(cover image)* dimodifikasi dengan cara menyisipkan suatu informasi rahasia (*hidden message)*. Citra yang telah dimodifikasi tersebut dinamakan image ter-*watermark (watermark image)*. *Watermark* yang telah disisipkan tersebut dapat kembali di dapatkan dengan melakukan proses ekstraksi pada citra yang telah ter-*watermak*. Sehingga apabila setelah melalui proses ekstraksi *watermark* yang ada didalam citra sampul tersebut mengalami kerusakan atau hilang, maka dapat diambil kesimpulan bahwa citra terwatermark tersebut telah mengalami serangan.

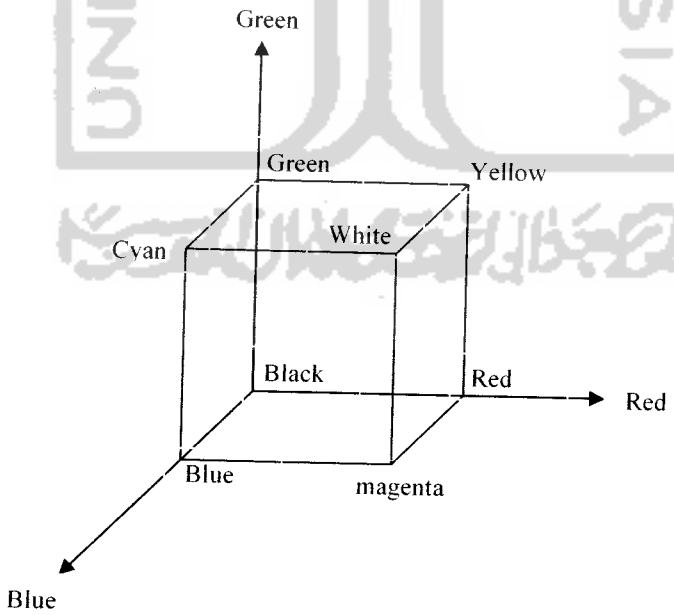
2.6 Struktur Citra dan Proses Pada Citra

Citra digital merupakan media yang sangat umum digunakan sebagai media (*data carrier*) dalam *watermarking*. Sebuah citra digital dapat diperoleh dengan menggunakan kamera *digital*, *scanner* atau peralatan lainya. Sebuah citra *digital* dapat dideskripsikan sebagai deretan nilai yang merepresentasikan intensitas cahaya dari sebuah titik, intensitas cahaya atau jarak warna ini disebut sebagai *pixel*. Dalam *pixel-pixel* ini dapat dilihat data *raster* dari sebuah citra, dan dengan *pixel-pixel* ini juga dapat dinyatakan ukuran suatu citra. Misalnya sebuah citra dengan ukuran 640 x 480 pixel terdiri dari 307.200 pixel [JOH01]. Citra dengan ukuran besar merupakan citra yang paling disukai dalam penerapan teknik watermarking karena mempunyai ruang yang besar untuk menyembunyikan informasi (*watermark*). Kualitas terbaik dari suatu citra *watermark* biasanya dibuat dengan menggunakan citra bitmap 24-bit sebagai citra sampul (*cover image*). Setiap *byte* menunjukan satu warna dari tiga komponen warna yang ada dan setiap nilai dari tiga byte data tersebut seluruhnya mendeskripsikan warna dan nilai luminansi dari suatu *pixel*. *Pixel-pixel* pada citra dinyatakan sebagai koordinat x dan y dimana x dan y merupaka nilai integer

Setiap *pixel* dari citra dapat terdiri dari 24-bit atau 8-bit data. Sebuah citra 24-bit akan mempunyai 2^{24} = 16777216 kompinasi warna [JOH01]. Setiap 24 bit dari citra 24-bit terdiri dari 3 byte yang masing-masing merepresentasikan warna merah (*red*), hijau (*green*) dan biru (*blue*) tau disingkat RGB. Sebuah warna dapat dibuat dengan mencampurkan antara ketiga warna tersebut dengan proposi warna yang berbeda. Dengan menggunakan model RGB, nilai dari fungsi *f(x,y)* merupakan suatu vektor dengan tiga komponen yang salaing berhibungan yaitu merah (R), hijau (G), biru (B). Hal ini dapat dipandang sebagai sebuah garis ortogonal yang memperlihatkan 3 dimensi ruang warna.

Setiap nilai dari *f(x,y)* merupakan sebuah titik yang terdapat pada kubus warna yang diperlihatkan pada gambar 2.2. ketiga komponen secara bormal dikuantisasi menggunakan 8 bit. Sebuah citra yang dibuat dengan menggunakan ketiga komponen ini dideskripsikan sebagai sebuah citra berwarna 24-bit.

Setiap *Pixel* bisa mempunyai nilai 0 sampai 255 yang merepresentasikan intensitas dari warna. Warna yang paling gelap akan mempunyai nilai 0 dan warna yang paling terang akan bernilai 255. sebagai contoh, sebuah pixel tersusun atas tiga byte data yang dapat disajikan sebagai berikut : 11111111 00000000 00000000. Delapan bit pertama merepresentasikan warnah merah, delapan bit kedua merepresentasikan warna hijau dan delapan bit ke tiga merepresentasikan warna biru. Nilai-nilai bit ini akan menghasilkan suatau pixel dengan warna merah, karena pada byte merah menunjukkan nilai maksimum (11111111) sedangkan pada byte hijau dan biru menunjukkan nilai minimum (00000000).



Gambar 2.6 Kubus Warna RGB

2.7 Warna

Sebuah citra digital tidak hanya terdiri dari warna hitam dan putih saja, tetapi bisa juga terdiri dari berbagai macam warna. Suatu citra berwarna tentu akan terlihat lebih menarik dibanding citra yang hanya berwarna hitam dan putih, karena suatu citra yang berwarna akan lebih memberikan informasi visual yang akan lebih banyak dibandingkan dengan sebuah citra dengan derajat keabuan.

Warna merupakan reaksi yang dirasakan oleh sistem visual mata manusia terhadap perubahan panjang gelombang cahaya dan setiap warna memiliki panjang gelombang tersendiri. Dari spektrum warna kita ketahui bahwa merah memiliki panjang gelombang paling besar, sedangkan warna violet memiliki panjang gelombang paling rendah. Dari berkas cahaya dengan kumpulan panjang yang berbeda-beda akan menimbulkan reaksi yang lain.

Dalam mempelajari warna, suatu warna biasanya terdiri dari 3 komponen warna dasar yang memberikan spektrum paling lebar seperti merah (*red*), hijau (*green*) dan biru (*blue*). Dengan ketiga warna dasar tersebut (yang biasa disingkat RGB) dapat dibentuk berbagai macam warna, tetapi sesungguhnya pencampuran ketiga warna ini tidak dapat mencakup keseluruhan daerah warna.

Untuk pembakuan sistem CIE memberikan standar panjang gelombang ketiga warna daras tersebut, yaitu:

- Merah = 700 nm
- Hijau = 546,1 nm
- Biru = 435,8 nm

Selain cara mempresentasikan ketiga warna dasar tersebut, terdapat pula cara lain untuk mendeskripsikan sifat-sifat warna dari suatu objek. Cara lain tersebut yakni dengan

mempresentasikan ke 3 atribut dasar seperti *brighness, hue* dan *saturation. Brighness* mempresentasikan luminasi yang dilihat, *hue* menyatakan derajat kemerahan, kehijauan dan kebiruan sedangkan saturation menunjukan aspek persepsiyang berubah-ubah bila cahaya putih ditambahkan pada caraya monokromatik.

BAB III

METODOLOGI

3.1 Metode Analisis Kebutuhan

Analisis kebutuhan merupakan tahap awal dalam perancangan perangkat lunak yang mempengaruhi proses fungsi dalam implementasi. Kesalahan dan kekurangan dalam tahap ini akan mengakibatkan kesalahan dalam perancangan perangkat lunak, sehingga perangkat lunak tidak dapat berjalan sebagaimana mestinya.

Metode yang dipakai untuk menganalisis kebutuhan guna mendukung implementasi aplikasi ini adalah metode analisis terstuktur (*structured analyze*). Metode ini melakukan pendekatan pendekatan secara terstruktur dengan mendefenisiskan kebutuhan sistem dari kebutuhan input, proses dan output.

3.2 Hasil Analisis Kebutuhan

Kebutuhan yang digunakan dalam membangun sitem aplikasi ini adalah kebutuhan yang meliputi kebutuhan masukan (*input*), kebutuhan keluaran (*output*), kebutuhan perangkat lunak (*software*), kebutuhan perangkat keras (*hardware*), kebutuhan antarmuka (*interface*).

3.2.1 Kebutuhan input (masukan)

Input yang dibutuhkan untuk aplikasi ini adalah:

1. File citra ber*form*at BMP 24-bit sebagai data sampul (*cover image*) yang akan diberikan *watermarking*.

2. File citra *watermark* ber*form*at BMP 24-bit.
3. File citra yang ter-*watermark*.

3.2.2 Kebutuhan output (keluaran)

Output yang diharapkan adalah:

1. File citra digital yang telah disisipi (ditempeli) *watermark*.
2. *Watermark* yang bersifat todak terlihat (*invisible*) yang menjadi bukti keaslian data citra digital tersebut.
3. In*form*asi yang terdapat di dalam *watermark*.

3.2.3 Kebutuhan perangkat lunak

Perangkat lunak yang dibutuhkan dalam pembuatan aplikasi ini adalah:

1. Microsoft Windows XP Profesional sebagai sistem operasi (SO) yang digunakan dalam mengimplementasikan perangkat lunak.
2. Borland Delphi 7.0 yang digunakan dalam membangun aplikasi ini.

3.2.4 Kebutuhan perangkat keras

Spesifikasi perangkat keras yang dibutuhkan untuk membangun aplikasi *watermarking* ini adalah:

1. Komputer dengan prosesor Pentium III 500 MHz atau yang lebih tinggi.
2. 64 Mb RAM atau lebih tinggi.
3. 350 Mb Hardisk space
4. Monitor SVGA atau VGA.
5. Mouse.

6. Keybord.
7. Printer

3.2.5 Kebutuhan antarmuka

Kebutuhan antarmuka yang akan dibangun pada sistem ini adalah kebutuhan antarmuka yang berorientasi WINDOWS, di mana kebutuhan antarmuka tersebut akan lebih memudahkan pengguna untuk menyisipkan *watermark* pada media citra digital. Kebutuhan antarmuka pada sistem ini akan dibangun dengan menggunakan *software* Borland Delphi 7.0, dimana sistem atau aplikasi ini terdiri dari dua *form* utama sebagai antarmuka untuk menempelkan in*form*asi (*watermark*) dan mendeteksi atau membaca in*form*asi (*watermark*) pada citra digital.

Rancangan sistem atau aplikasi yang dibangun diharapkan dapat membatu *user* dalam mengamankan data citra digital yang dimiliki sehingga data citra digital tersebut dapat terjaga keasliannya.

3.3 Metode Perancangan Perangkat Lunak

Metode perancangan sistem yang digunakan dalam pembuatan aplikasi *watermarking* dengan menggunakan metode *wavelet* ini adalah metode perancangan beraliran data dengan memakai alat-alat pengembangan sistem berupa *flowchart*. *Flowchart* pada dasarnya merupakan konsep perancangan yang mudah pada sistem program terstuktur.

3.4 Proses Perancangan Perangkat Lunak

Dalam proses perancangan dan pembuatan aplikasi perangkat lunak ini terdapat beberapa tahap yanag dapat dibedakan menjadi 3 tahap yaitu:

1. Tahap *pemrograman visual*

   Pada tahap ini yang dilakukan adalah merancang *form* dan komponen-komponen yang diperlukan dalam penggunaan program. Perancangan tersebut diperoleh dari paket-paket komponen yang tersedia di Borland Delphi 7.0.

2. Tahap penulisan kode

   Pada tahap ini proses yang dilaukan adalah penulisan kode-kode berdasarkan pemrograman Delphi yang akan dijalankan dengan prosedure untuk selanjutnya diletakan pada kontrol-kontrol (*object*) yang dipakai untuk menjalankan proses tertentu.

3. Tahap *Debugging*

   Pada tahap ini dilakukan pengujian terhadap perangkat lunak yang dibuat dengan meng-*compile* program. Metode pengujian yang digunakan adalah *try* and *error*, dimana setiap langkah yang menghasilkan output diteliti kembali keabsahannya, sehingga hasil yang didapat benar-benar dengan metode yang digunakan.

3.5 Hasil Perancangan Perangkat Lunak

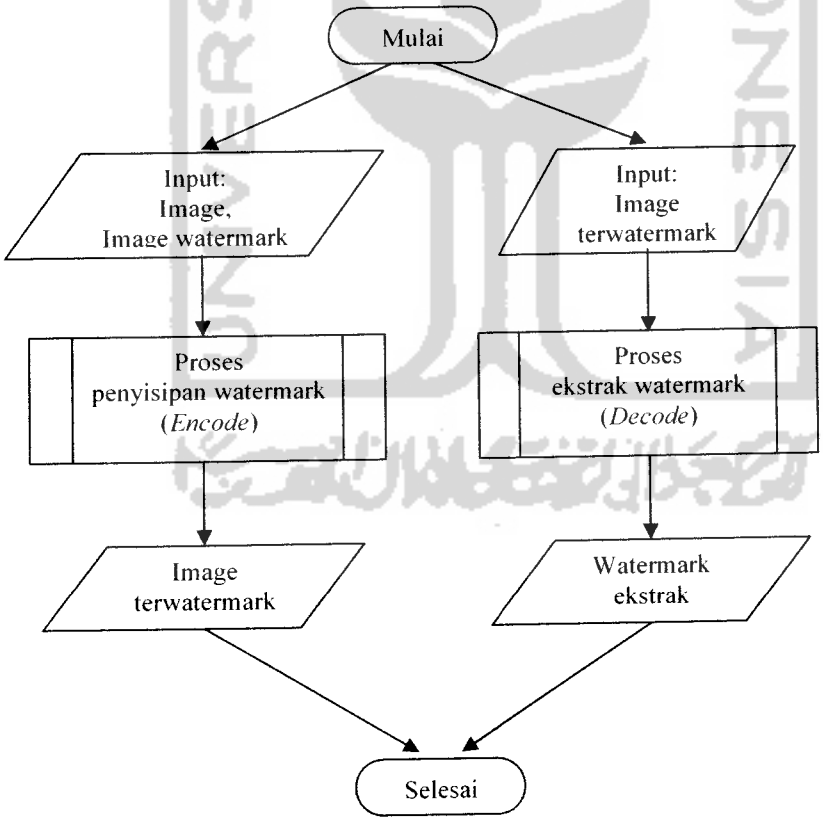
Berdasarkan hasil analisis yang telah dilakukan maka dapat diketahui apa saja yang menjadi kebutuhan perangkat keras, kebutuhan perangkat lunak, antarmuka sistem, serta masukan sistem yang akan diproses menjadi keluaran sistem melalui metode-metode yang ada, sehingga sistem yang dibuat nanti sesuai dengan apa yang diharapkan.

Dalam mengembangkan proses-proses yang terjadi akan digambarkan dengan *flowchart* sehingga akan terlihat urutan proses yang dilakukan dalam aplikasi ini.

3.5.1 Perancangan Diagram Alir Sistem

Diagram alir sistem merupakan visualisasi langkah kerja dari sistem dalam wujud *flowchart*. Penggunaan *flowchart* memberikan kelebihan karena seluruh bagian sistem dapat ditampilkan mulai dari bagian input sistem, proses-proses yang ada dalam sistem, serta bagian output sistem.penggunaan *flowchart* akan memberikan kemudahan dalam menilai alur kerja dan efektifitas kerja sistem. *Flowchart* terbagi menjadi beberapa bagian yaitu :

1. Flowchart umum sistem

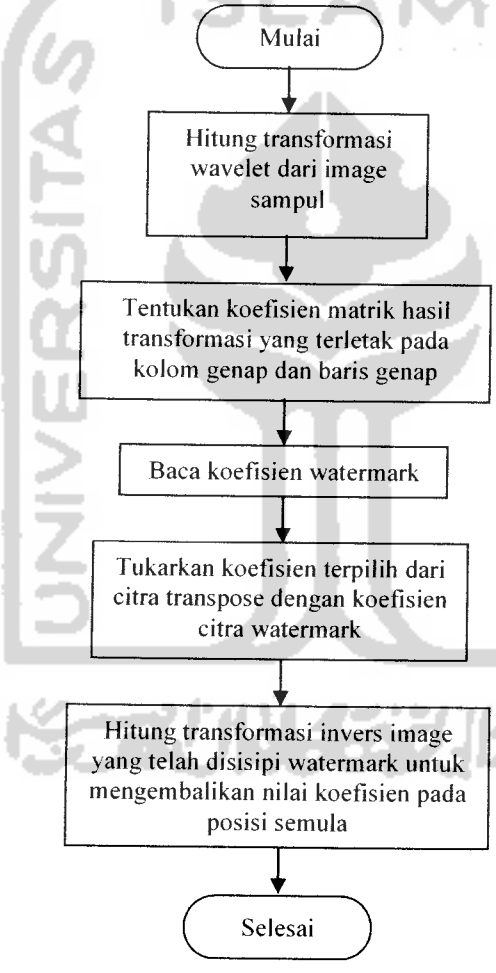


Gambar 3.1 *Flowchart* Umum Sistem

Pada gambar 3.1 menggambarkan sistem aplikasi *watermarking* yang akan dibangun. Dimana sistem tersebut dibagi menjadi dua proses yaitu proses penyisipan *watermark* (*encode*) dan proses pengekstrakan *watermark* (*decode*).

2. Flowchart proses penyisipan *watermark* (*encode*)

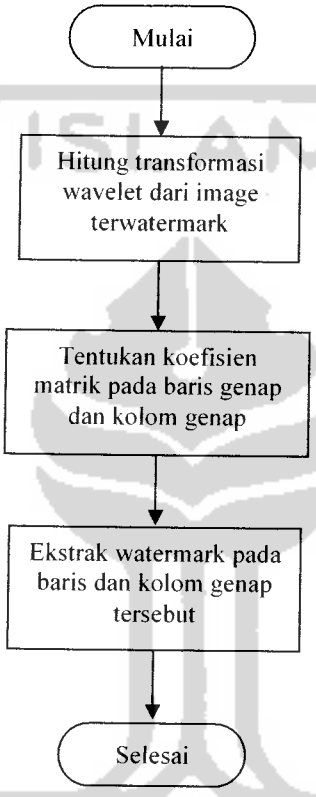
Proses lebih detail dari proses penyisipan *watermark* dapat dilihat pada gambar 3.2 berikut :



Gambar 3.2 Proses Penyisipan *Watermark*

3. Flowchart Proses Ekstrak *Watermark* (*Decode*)

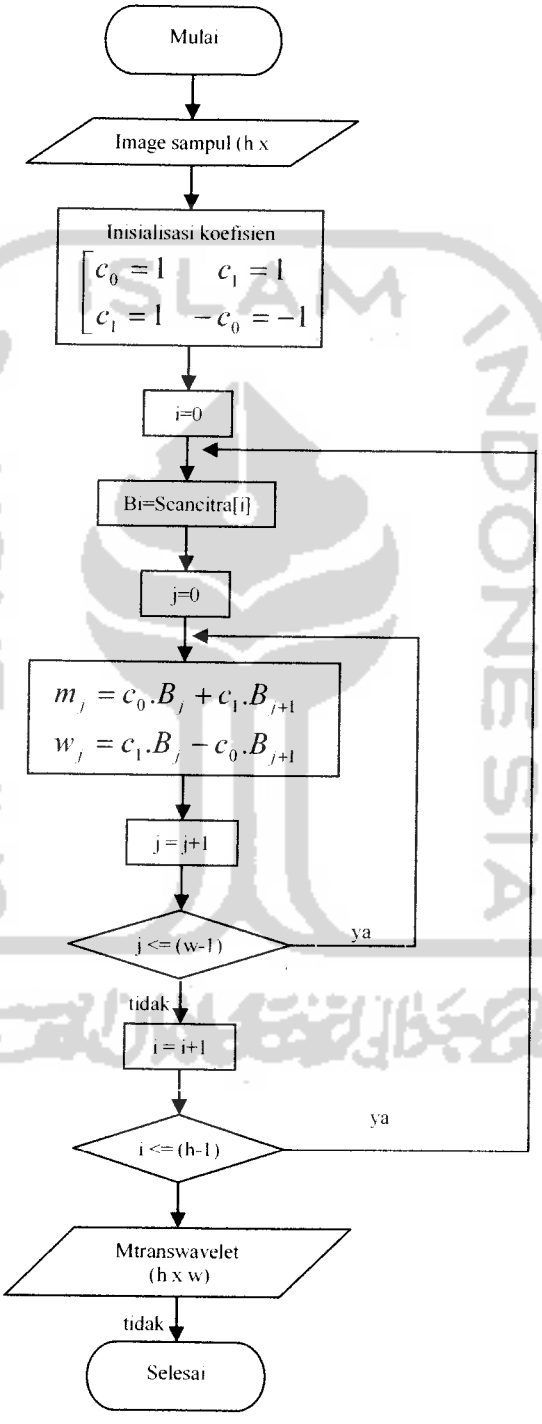
Proses Lebih detail dari proses ekstrak *watermark* dapat dilihat pada gambar 3.3 berikut :



Gambar 3.3 Proses Ekstrak *Watermark*

4. Algoritma Transformasi *Wavelet*

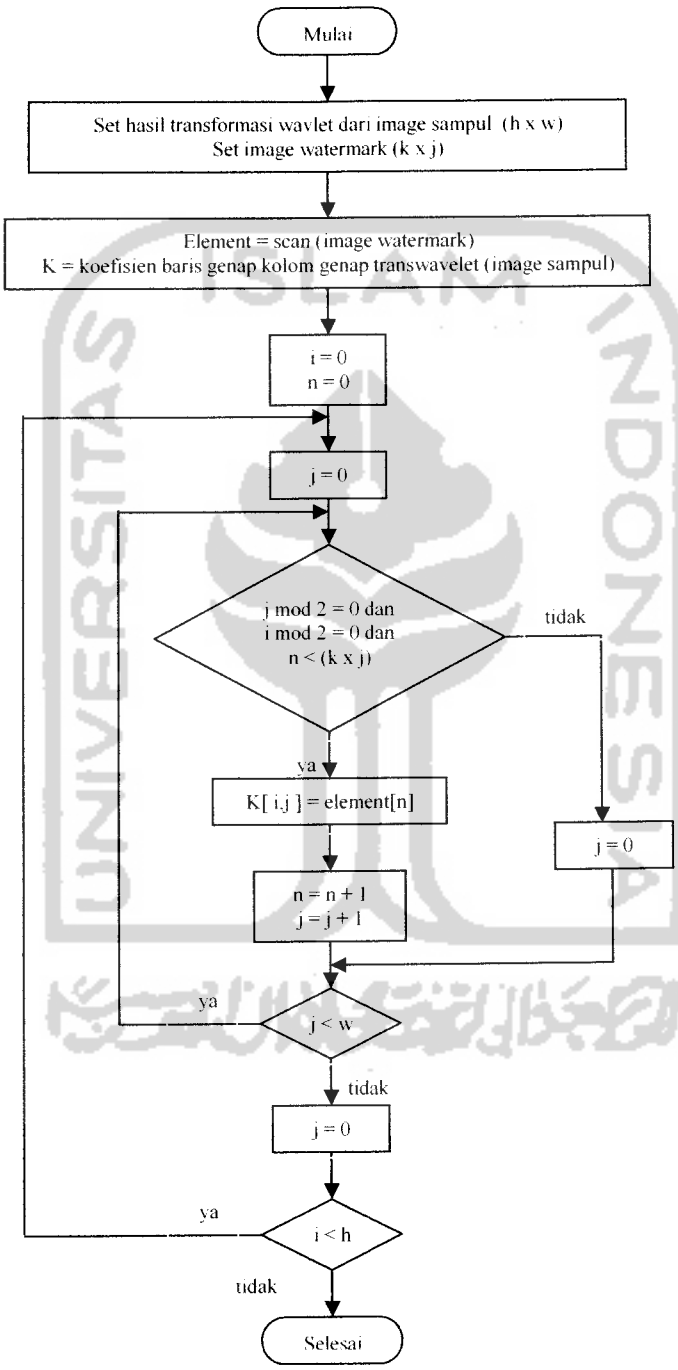
Berikut adalah algoritma transformasi *wavelet* yang penulis buat dalam bentuk *flowchart* :



Gambar 3.4 *Flowchart* Transformasi *Wavelet*

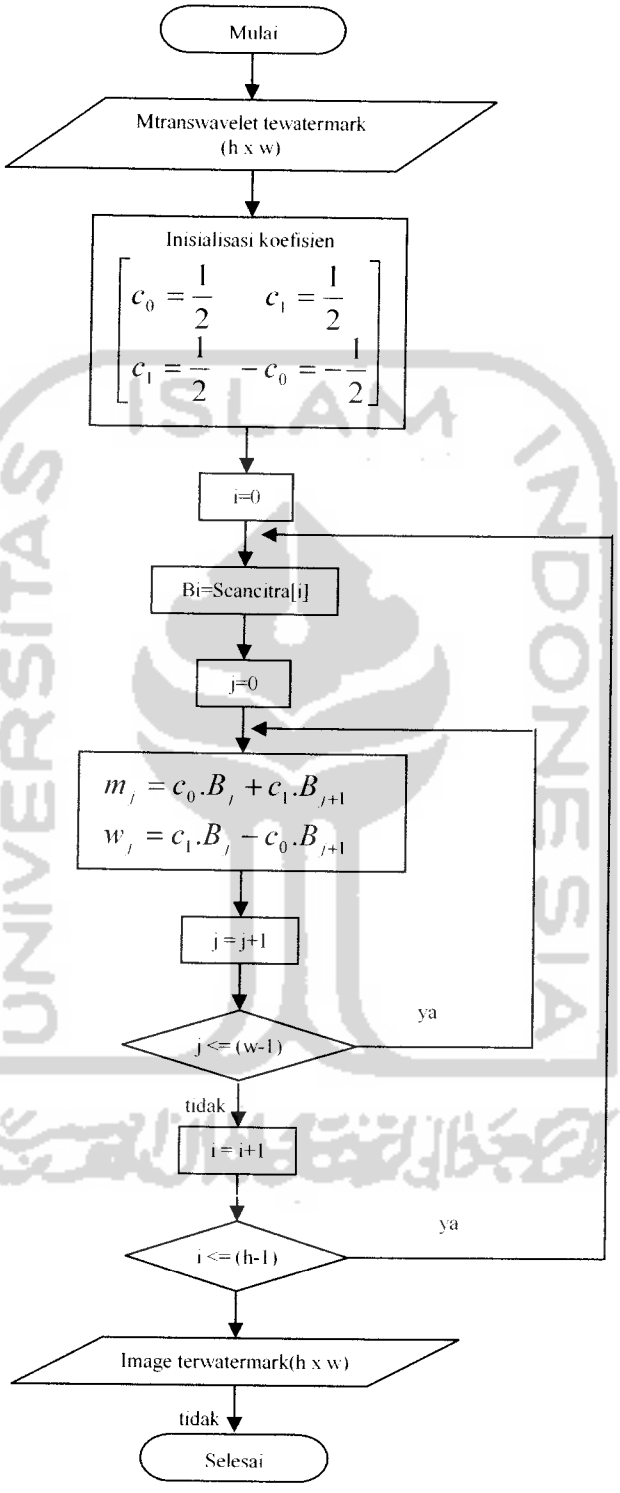
5. Algoritma Penyisispan *Watermark*

Berikut adalah algoritma penyisipan pesan yang penulis buat dalam bentuk *flowchart* :



Gambar 3.5 Algoritma Penyisipan *Watermark*

6. Algoritma Transformasi Invers



Gambar 3.6 Algoritma Transformasi Invers

Sama seperti transformasi *wavelet*, transformasi invers juga dapat di implementasikan dalam bentuk matrik. Nilai invers diperoleh dengan mengalikan koefisien-koefisien hasil transformasi dengan matrik invers dari matrik C (filter). Jika diimplementasikan dalam bentuk matrik, fungsi matrik invers dari transformasi *wavelet* Haar dapat dinyatakan sebagai berikut :

$$\begin{bmatrix} c_0 & c_1 \\ c_1 & -c_0 \end{bmatrix}, \text{ dimana koefisien } c_0 = \frac{1}{2} \text{ dan } c_1 = \frac{1}{2}.$$

7. Algoritma Pengekstrakan *Watermark*

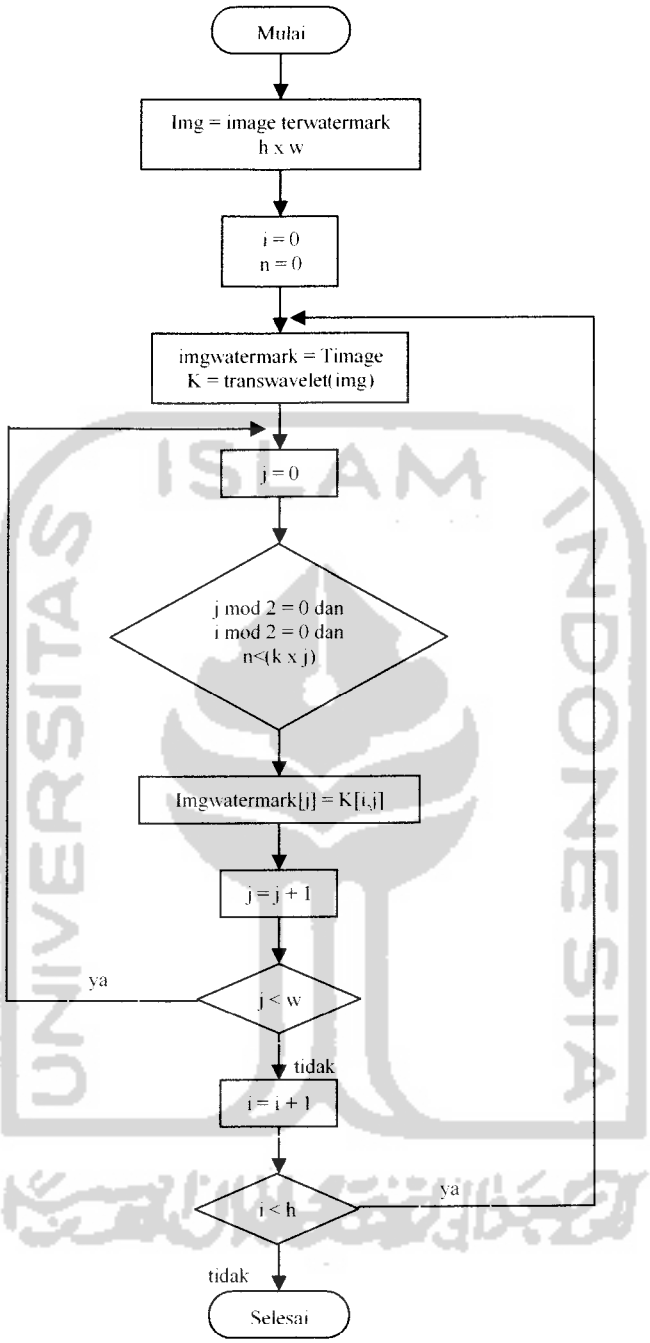
Sebelum mengekstrak pesan, maka citra ter-*watermark* perlu di transformasi *wavelet* terlebih dahulu. Setelah dikakukan transformasi, maka tentukan koefisien yang paling tidak signifikan (*insignificant*), yaitu koefisien-koefisien yang terletak pada baris dan kolom genap. Nilai-nilai dari koefisien tersebut diekstrak kemudian disusun kembali hingga menghasilkan *watermark*.

Rumus yang digunakan pada algoritma mengekstrak pesan ini merupakan kebalikan rumus 3.7, yaitu :

$$\text{Element}_{[i]} = \text{Koef}_{[i,j]} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3.2)$$

dimana Koef merupakan koefisien hasil transformasi wavelet, elemen merupakan elemen dari citra hasil setelah di ekstrak dan i = 0, 1, 2, 3, ...

Algoritma pengekstrakan *watermark* ini dapat dilihar pada gambar 3.7 berikut ini :

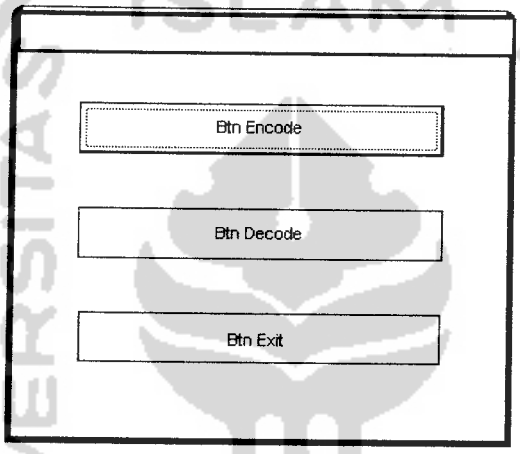


Gambar 3.7 Algoritma Pengekstrakan *Watermark*

3.6 Implementasi Antarmuka (*Interface*)

3.6.1 *Form* menu utama

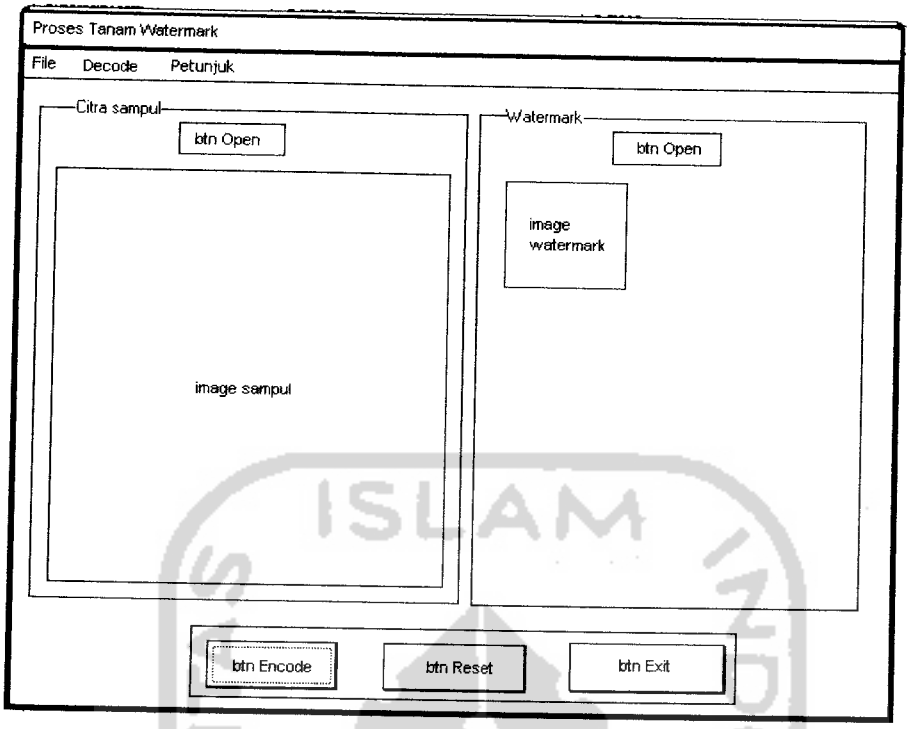
Form menu utama dalam program ini memberikan dua pilihan kepada *user* untuk melakukan proses, yaitu proses penyisipan *watermark* (*encode*), proses pengekstrakan *watermark* (*decode*). Pada menu utama ini terdapat tiga buah *button*, yaitu *button tanam watermark* dan *button ekstrak watermark*. Jika kedua button ini di proses maka *user* akan terhubung dengan ketiga proses tersebut.



Gambar 3.8 Rancangan Tampilan *Form* Menu Utama

3.6.2 *Form* penyisipan *watermark*

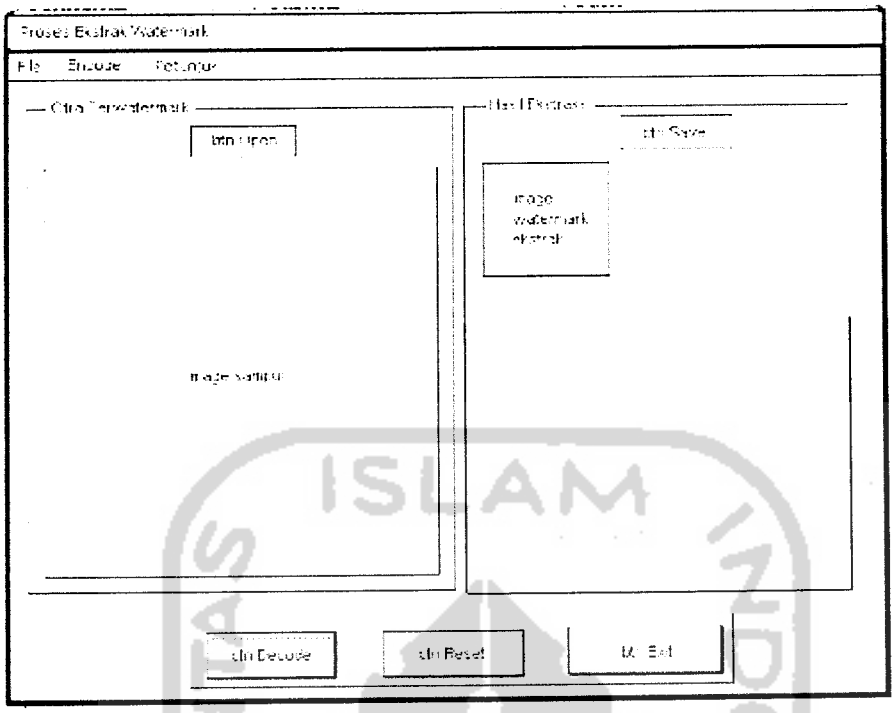
Antarmuka ini dirancang agar pengguna dapat melakukan proses penanaman *watermark* pada media citra digital. Pada *form* ini, jika button open diklik maka file citra dan *watermark* akan muncul pada *panel* yang telah disediakan. Citra sampul dan *watermark* akan diolah pada proses-proses berikutnya seperti proses transformasi *wavelet* dan proses penyisipan pesan. Proses-proses tersebut akan berjalan apabila user mengklik *button encode*. *Button reset* dan *exit* merupakan button tambahan yang bisa digunakan untuk mengulangi proses penanaman *watermark* atau keluar dari *form* tanam *watermark*.



Gambar 3.9 Rancangan *Form* Tanam *Watermark*

3.6.3 *Form* ekstrak *watermark*

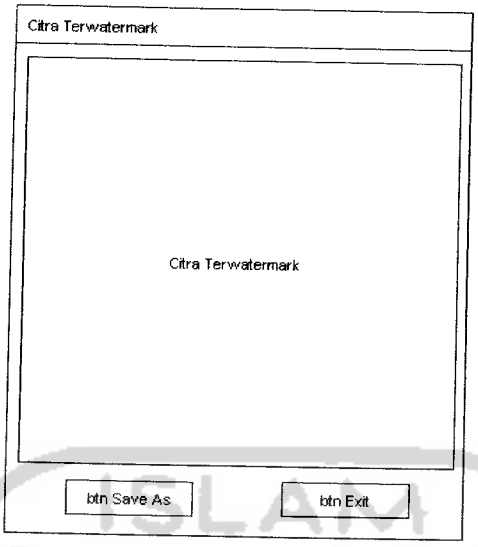
Form proses ekstrak *watermark* ini berfungsi untuk melakukan proses mengekstrak *watermark* dan menampilakan kembali *watermark* yang telah disisipkan kedalam citra. Tampilan utamanya tidak berbeda jauh dengan *form* tanam *watermark*. Pada *form* ekstrak pesan ini *watermark* yang tertanam pada image dapat ditampilkan kembali kedalam *panel* hasil ekstraksi yang mana *watermark* hasil ekstraksi tersebut bisa secara langsung dibandingkan dengan *watermark* asli yang bisa ditampilkan oleh *user* dengan mengeloadnya pada *panel watermark* asli. Proses ekstraksi *watermark* akan berjalan apabila button *decode* diklik. Selain itu pada *form* ini terdapat dua *button* tambahan yaitu *button reset* yang berfungsi untuk mengulangi proses dari awal dan button eksit yang berfungsi untuk keluar dari *form* ekstrak *watermark*.



Gambar 3.10 Rancangan *Form* Ekstrak *Watermark*

3.6.4 *Form* Hasil *Encode*

Form hasil *encode* ini berfungsi untuk menampilkan citra yang sudah disisipi pesan. *Form* ini akan muncul apabila proses dari penanaman *watermark* telah selesai. Pada *form* ini, citra yang sudah ter-*watermark* dapat di simpan oleh *user* dengan mengklik *button Save As*. Jika user ingin keluar dari *form* maka *user* dapat mengklik *button exit*.



Gambar 3.11 Rancangan *Form* Hasil *Encode*

3.7 Batasan Implementasi

Dalam pembuatan program aplikasi *watermarking* ini, implementasi dibatasi pada proses pemasukan data (file citra, *watermark*, citra ter-*watermark*), penanaman *watermark* pada image, dan pengekstrakan *watermark*.

Bahasa dan kompilator yang digunakan dapam pembuatan aplikasi *watermarking* dibangun dengan menggunakan prangkat lunak Borland Delphi 7.0 dan dijalankan pada Sistem Operasi berbasis windows. Bahasa pemrograman Delphi dipilih memiliki komponen yang berbasis visual dan non visual, disamping itu juga merupakan program yang dapat dioperasikan pada keadaan yang sebenarnya.

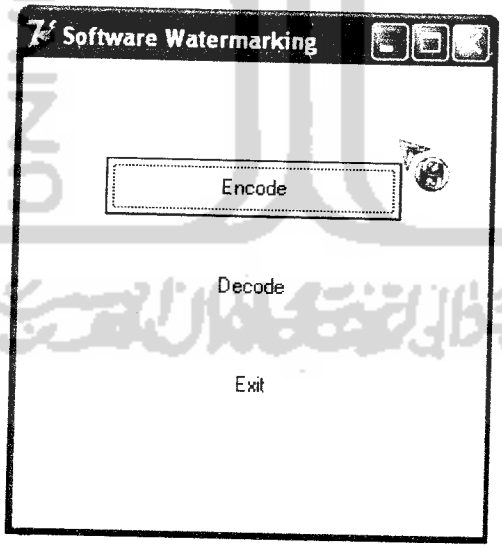
3.8 Hasil Implementasi

3.8.1 Implementasi Antarmuka

Antarmuka merupakan sarana untuk menghubungkan pemakai (*user*) dengan program. Antarmuka dari program *watermarking* berbasis transformasi *wavelet* ini diimplementasikan dengan komponen *form* Borland Delphi 7.0.

1. Antarmuka *Form* Utama

Dalam *form* utama terdapat dua button utama yang akan menghubungkan pengguna (*user*) dengan kedua proses utama program. *Button* yang pertama adalah *button encode*, yang akan mengantarkan pengguna (*user*) ke proses penyisipan *watermark* (*encode*). Sedangkan *button decode* akan mengantarkan pengguna (*user*) menuju proses pengekstrakan pesan (*decode*). Selain itu pada *form* utama ini juga terdapat *button exit* yang merupakan *button* tambahan yang bisa digunkan *user* untuk keluar dari *form* ini. Antarmuka pemakai dari menu utama ini dapat dilihat dalam gambar 3.12.



Gamabar 3.12 Antarmuka *Form* Utama

2. Antarmuka Proses Tanam *Watermark* (*encode*)

Pada Antarmuka proses *encode* ini terdapat dua *button open* untuk menampilkan citra sampul (*cover image*) dan *watermark*. Terdapat *mainMenu* yang terdiri dari 3 *submenu*, yaitu *submenu* file, *decode* dan petunjuk.

a. Submenu *File*

Submenu File terdiri dari:

1. Submenu *open*

   Submenu *open* digunakan untuk menampilkan citra sampul dan *watermark*. Submenu ini dibagi menjadi dua bagian yaitu submenu *open* citra sampul dan submenu *open watermark*.

2. Submenu *reset*

   Submenu *reset* digunakan untuk mereset tampilan citra dan *watermark* yang ada pada *form encode*. Dengan menggunakan submenu ini maka tampilan *form encode* akan seperti tampilan awal saat *form* ini dibuka untuk pertama kalinya.

3. Submenu *exit*

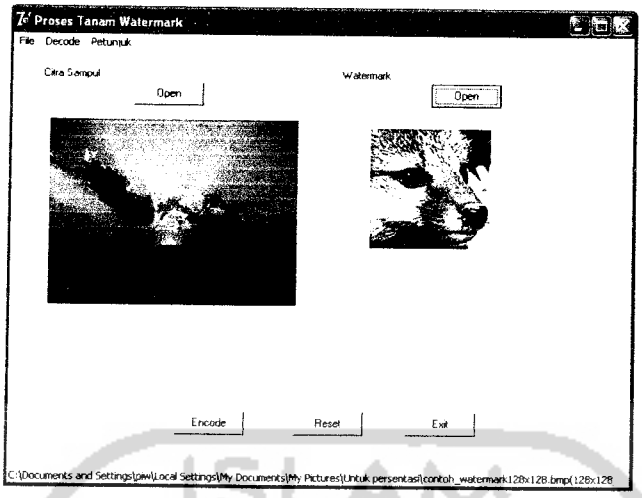
   Digunakan untuk keluar dari program.

b. Decode

Submenu ini digunakan untuk memanggil dan menampilkan *form decode*.

c. Petunjuk

Subemenu petunjuk berisi tentang informasi penggunaan *form encode*.

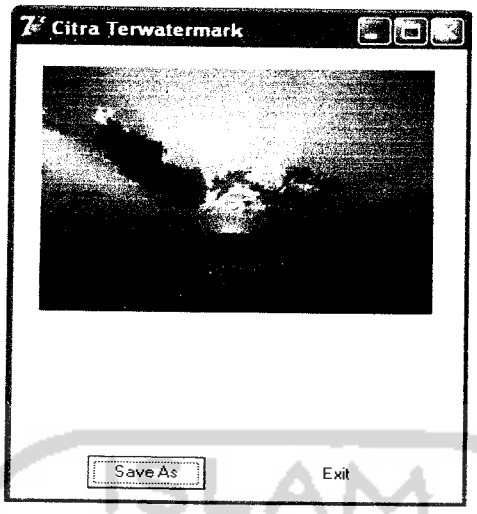


Gambar 3.13 Antarmuka Proses Tanam *watermark* (*encode*)

Selain mainMenu, di dalam *form* ini juga terdapat *button encode, reset* dan *exit*. *Button encode* berfungsi untuk memanggil submodul penyisipan pesan, sedangkan button reset adalah untuk mengembalikan *form* pada keadaan semula dan button exit digunakan untuk keluar dari *form encode*.

3. Antarmuka Hasil *Encode*

Setelah proses penanaman *watermark* (*encode*) telah selesai dijalankan maka program akan menampilkan *form* hasil dari proses *encode* tersebut. Antarmuka hasil *encode* ini memiliki dua button yaitu *button Save As* yang berfungsi untuk menyimpan citra yang telah ter*watermark* dan *button Exit* yang digunakan untuk menutup *form* hasil *encode*. Antarmuka hasil *encode* dapat dilihat pada gambar 3.14 berikut:



Gambar 3.14 Antarmuka Hasil *Encode*

4. Antarmuka Proses Pengekstrakan *Watermark* (*Decode*)

Pada antarmuka proses *decode* ini terdapat dua buah *button open* yang masing-masing berfungsi untuk menampilkan citra yang sudah ter*watermark* dan *watermark* asli. Seperti pada *form* proses *encode*, pada *form* ini juga terdapat mainMenu yang terdiri dari 3 submenu, yaitu submenu file, *encode* dan petunjuk.

a. Submenu File

Submenu File terdiri dari:

1. Submenu *open*

   Submenu open digunakan untuk menampilkan citra ter*watermark* dan *watermark* asli. Submenu ini dibagi menjadi dua bagian yaitu submenu open citra ter*watermark* dan submenu open *watermark* asli.

2. Submenu *reset*

   Submenu *reset* digunakan untuk mereset tampilan citra dan *watermark* yang ada pada *form decode*. Dengan menggunakan submenu ini maka

tampilan *form encode* akan seperti tampilan awal saat *form* ini dibuka untuk pertama kalinya.

3. Submenu *exit*

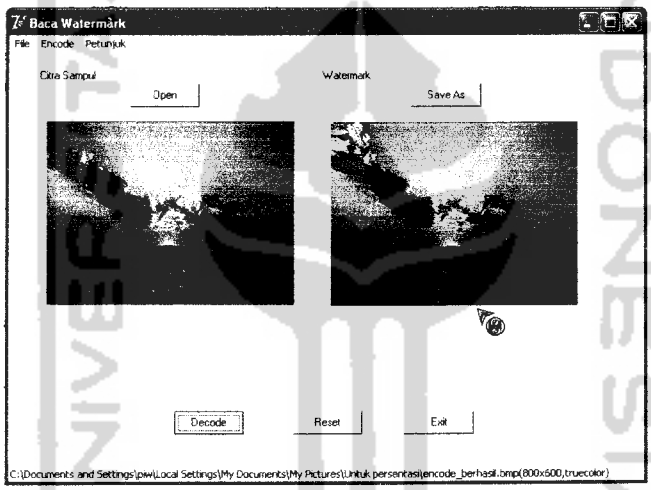
   Digunakan untuk keluar dari program.

b. Encode

Submenu ini digunakan untuk memanggil dan menampilkan *form encode*.

c. Petunjuk

Submenu petunjuk berisi tentang informasi penggunaan *form decode*.



Gambar 3.15. Antarmuka Ekstrak *Watermark*

Pada proses *decode* submodul dari proses pengekstrakan akan dipanggil dengan mengklik *button decode*, sedangkan fungsi *button reset* dan *exit* sama dengan fungsi yang ada pada *form encode* yaitu untuk mereset *form* kembali pada keadaan awal dan untuk keluar dari program. *Watermark* yang telah berhasil di ekstrak akan ditampilkan pada panel *watermark* ekatrak yang mana *watermark* ekstrak tersebut dapat disimpan dengan mengklik *button Save As*.

3.8.2 Implementasi Prosedur

Implementasi dari prosedur aplikasi *watermarking* dengan menggunakan metode wavelet pada objek citra digital yaitu meliputi prosedur penanaman *watermark* (*encode*) dan pengekstrakan *watermark* (*decode*).

1. Prosedur *Encode*

Prosedur *encode* adalah prosedur yang menitik beratkan pada penukaran elemen *watermark* dengan elemen citra sampul yang terpilih. Prosedur *encode* ini merupakan kesatuan dari beberapa proses yaitu, proses pembacaan nilai koefisien citra sampul dan *watermark*, proses transformasi wavelet untuk mengambil koefisien citra sampul terpilih dan proses transformasi invers untuk mengembalikan koefisien-koefisien citra sampul pada posisi semula. Untuk lebih jelasnya proseduer *encode* ini akan di bagi kedalam beberapa proses seperti yang telah disebutkan diatas.

a. Proses pembacaan nilai koefisien

Proses pembacaan nilai koefisien citra ini dilakukan dengan menggunakan fungsi *scanline* dan karena citra yang digunakan adalah citra *true color* maka komponen *red*, *green*, *blue* akan di simbolkan dengan variabel Ri untuk *red*, Gi untuk *green* dan Bi untuk *blue*.

```
for i:=0 to high-1 do
begin
 Pc:=image.Picture.Bitmap.Scanline[i];
 for j:=0 to width-1 do
 begin
   Bi[i,j]:=Pc[3*j];
   Gi[i,j]:=Pc[3*j+1];
   Ri[i,j]:=Pc[3*j+2];
 end;
end;
```

b. Proses transformasi wavelet

Proses transformasi wavelet dilakukan sebanyak dua kali, yaitu transformasi sepanjang baris dan transformasi sepanjang kolom, dimana untuk transformasi sepanjang baris variabel hasil transformasi yang digunakan adalah Rr untuk komponen *red*, Gr untuk komponen *green* dan Br untuk komponen *blue*. Sedangkan untuk transformasi sepanjang kolom, variabel hasil trans*form*si yang digunakan adalah Ro untuk komponen *red*, Go untuk komponen *green* dan Bo untuk komponen *blue*.

1). Proses transformasi wavelet sepanjang baris citra

```
for i:=0 to high-1 do
 for j:=0 to (width div 2)-1 do
 begin
    Rr[i,j*2]      :=Ri[i,j*2]+Ri[i,j*2+1];
    Rr[i,j*2+1]    :=Ri[i,j*2]-Ri[i,j*2+1];
    Gr[i,j*2]      :=Gi[i,j*2]+Gi[i,j*2+1];
    Gr[i,j*2+1]    :=Gi[i,j*2]-Gi[i,j*2+1];
    Br[i,j*2]      :=Bi[i,j*2]+Bi[i,j*2+1];
    Br[i,j*2+1]    :=Bi[i,j*2]-Bi[i,j*2+1];
 end;
```

2). Proses transformasi wavelet sepanjang kolom citra

```
for i:=0 to w-1 do
 for j:=0 to (high div 2)-1 do
 begin
      Ro[j*2,i]      :=Rr[j*2,i]+Rr[j*2+1,i];
      Ro[j*2+1,i]    :=Rr[j*2,i]-Rr[j*2+1,i];
      Go[j*2,i]      :=Gr[j*2,i]+Gr[j*2+1,i];
      Go[j*2+1,i]    :=Gr[j*2,i]-Gr[j*2+1,i];
      Bo[j*2,i]      :=Br[j*2,i]+Br[j*2+1,i];
      Bo[j*2+1,i]    :=Br[j*2,i]-Br[j*2+1,i];
end;
```

c. Proses penyisipan elemen *watermark*

Pada proses ini eleman *watermark* yang sudah dibaca dengan fungsi scanline akan ditukarkan dengan elemen citra hasil tras*form*asi wavelet pada baris genap dan kolom genap. Pada proses ini Ro adalah elemen *red* untuk citra transformasi, Go adalah elemen *green* untuk citra transformasi dan Bo adalah elemen *blue* untuk citra transformasi. Sedangkan untuk citra *watermark* Wr adalah elemen *red*, Wg adalah elemen *green* dan Bo adalah elemen *blue*.

```
i:=0;
j:=0;
for m:=0 to (high_watermark)-1 do
begin
        for n:=0 to width_watermark-1 do
        if (i<high-1) and (j<width-1) then
    begin
        Ro[i*2+1,j*2+1]:=Wr[m,n]/4;
        Bo[i*2+1,j*2+1]:=Wb[m,n]/4;
        Go[i*2+1,j*2+1]:=Wg[m,n]/4;
        j:=j+1;
    end;
j:=0;
i:=i+1;
end;
```

d. Proses transformasi invers

Seperti di jelaskan sebelumnya transformasi invers bertujuan untuk mengembalikan elemen citra pada posisi semula setelah dilakukan proses penukaran elemen dengan citra *watermark*. Transformasi ini dilakuakan sebanyak dua kali juga, namun perbedaannya dengan transformasi wavelet, transformasi ini dikakukan sepanjang kolom dulu baru sepanjang baris citra. Pada proses ini variabel yang digunakan sama seperti transformasi wavelet yaitu Rr, Gr dan Br untuk hasil transformasi elemen red, green dan blue. Sedangkan variabel untuk transformasi sepanjang kolom adalah Ro, Go dan Bo untuk komponen elemen red, green dan blue.

a) Proses transformasi invers sepanjang kolom

```
for i:=0 to w-1 do
    for j:=0 to h3-1 do
    begin
      Rr[j*2,i]  :=0.5*Ro[j*2,i]+0.5*Ro[j*2+1,i];
      Rr[j*2+1,i]:=0.5*Ro[j*2,i]-0.5*Ro[j*2+1,i];
      Gr[j*2,i]  :=0.5*Go[j*2,i]+0.5*Go[j*2+1,i];
      Gr[j*2+1,i]:=0.5*Go[j*2,i]-0.5*Go[j*2+1,i];
      Br[j*2,i]  :=0.5*Bo[j*2,i]+0.5*Bo[j*2+1,i];
      Br[j*2+1,i]:=0.5*Bo[j*2,i]-0.5*Bo[j*2+1,i];
    end;
```

b) Proses transformasi invers sepanjang baris

```
for i:=0 to h-1 do
    for j:=0 to w3-1 do
    begin
      Ro[i,j*2]  :=0.5*Rr[i,j*2]+0.5*Rr[i,j*2+1];
      Ro[i,j*2+1]:=0.5*Rr[i,j*2]-0.5*Rr[i,j*2+1];
      Go[i,j*2]  :=0.5*Gr[i,j*2]+0.5*Gr[i,j*2+1];
      Go[i,j*2+1]:=0.5*Gr[i,j*2]-0.5*Gr[i,j*2+1];
      Bo[i,j*2]  :=0.5*Br[i,j*2]+0.5*Br[i,j*2+1];
      Bo[i,j*2+1]:=0.5*Br[i,j*2]-0.5*Br[i,j*2+1];
    end;
```

2. Prosedure *Decode*

Prosedure *decode* adalah prosedur yang menitik beratkan pada pendeteksian elemen *watermark* yang ada pada citra hasil *encode*. Prosedur dilakukan agar nilai koefisien citra *watermark* dapat dibaca dan disususun kembali sehingga sesuai dengan gambar dari citra *watermark* aslinya. Implementasi dari proses ini meliputi proses transformasi wavelet dan proses pendeteksian *watermark*. Implementasi transformasi wavelet yang dilakukan pada proses *decode* sama dengan implementasi yang dilakukan pada proses *encode*. Variabel yang digunakan pada proses pendeteksian *watermark* adalah variabel yang sama digunakan pada proses penyisispan *watermark*.

```
for m:=0 to (high div 2)-1 do
 begin
  for n:=0 to (width div 2)-1 do
   if (i<high-1) and (j<width-1) then
   begin
    Wr[m,n]:=Ro[i*2+1,j*2+1]*4;
    Wb[m,n]:=Bo[i*2+1,j*2+1]*4;
    Wg[m,n]:=Go[i*2+1,j*2+1]*4;
   j:=j+1;
  end;
  j:=0;
  i:=i+1;
end;
```

BAB IV

HASIL DAN PEMBAHASAN

4.1 Pengujian Program

Pada tahap analisis kinerja perangkat lunak dijelaskan tentang pengujian aplikasi *watermarking* dengan menggunakan metode wavelet pada objek citra digital. Pengujian dilakukan dengan kompleks dan diharapkan dapat diketahui kekurangan-kekurangan dari sistem untuk kemudian diperbaiki sehingga kesalahan dari sistem dapat diminimalisasi atau bahkan dihilangkan. Pengujian sistem ini dilakukan untuk mendapatkan hasil yang akurat.

4.2 Analisis Kinerja Sistem

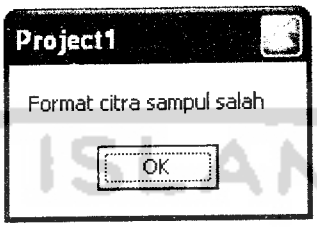
4.2.1 Penanganan kesalahan

Perangkat lunak ini dibuat cukup bersifat *user friendly* atau dengan kata lain mudah untuk di mengerti oleh pengguna. Jika terdapat kesalahan-kesalahan pemasukan data maka sistem akan memberikan tanggapan (*feedback*) kepada pengguna berupa jendela dialog (*Messagebox*). Tipe-tipe kesalahan tersebut dapat dilihat pada penjelasan berikut:

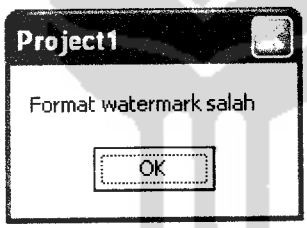
1. Penanganan kesalahan *input* format gambar

Penanganan kesalahan *input* format gambar ini dilakukan untuk menangkap *error* yang terjadi ketika data yang dimasukan tidak sesuai dengan format yang seharusnya. Contoh penanganan kesalahan *input* data terdapat pada form tanam *watermark* (*encode*)

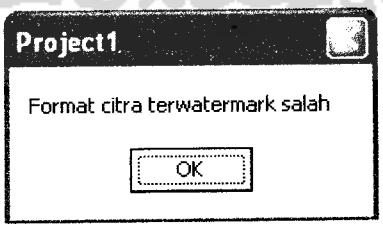
dan form ekstrak *watermark* (*decode*). Jika image yang di*input*kan tidak berformat BMP 24-bit, maka akan muncul *messagebox* seperti gambar 4.1, 4.2 dan 4.3



Gambar 4.1 Tampilan Jendela Dialog Jika Format Citra Sampul Bukan BMP 24-bit



Gambar 4.2 Tampilan Jendela Dialog Jika Format Citra *Watermark* Bukan BMP 24-bit



Gambar 4.3 Tampilan Jendela Dialog Jika Format Citra Ter*watermark* Bukan BMP 24-bit

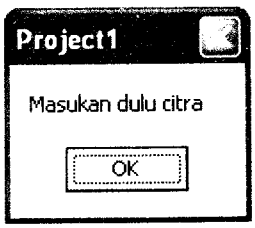
2. Penanganan kesalahan *input* ukuran *watermark*

Penanganan kesalahan *input* ukuran *watermark* ini dilakukan untuk menangkap *error* yang terjadi ketika ukuran *watermark* melebihi batasan ukuran yang disediakan. Contoh penanganan kesalahan *input* ini terdapat pada *form* tanam *watermark* (*encode*). Jika ukuran *watermark* melebihi batasan yang ditentukan, maka akan muncul *messagebox* seperti pada gambar 4.4

Gambar 4.4 Tampilan Jendela Dialog Jika Ukuran *Watermark* Melebihi Batas yang Ditentukan

3. Penanganan kesalahan proses jika citra belum di*input*kan

Penanganan kesalahan proses jika citra belum di*input*kan dilakukan untuk menangkap *error* yang terjadi ketika *button encode* atau *decode* di klik sedangkan citra sampul atau *watermark* belum dimasukan. Contoh penanganan kesalahan tersebut dapat dilihat pada gambar 4.5



Gambar 4.5 Tampilan Jendela Dialog Jika Citra Belum Dimasukan

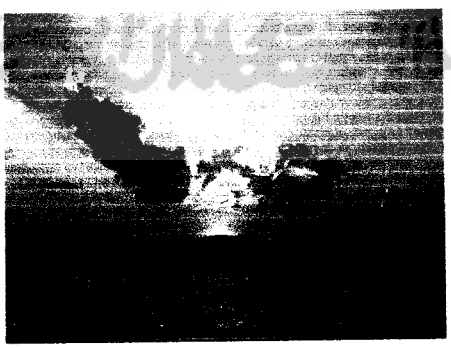
4.2.2 Pengujian dan analisis

Pada tahap pengujian dan analisis program ini, dilakukan perbandingan antara kebenaran masukan serta kesesuaian program dengan kebutuhan sistem. Pengujian dan analisis tersebut dilakukan pada dua tahap. Tahap pertama adalah pengujian dan analisis proses tanam *watermark* (*encode*). Pengujian dan analisis pada proses tanam *watermark* (*encode*) ini menggunakan beberapa file citra sampul dan file citra *watermark*. Proses kedua adalah proses ekstrak *watermark* (*decode*). Dalam proses ini analisis dan pengujiannya akan menggunakan file citra ter*watermark* dari proses *encode* sebelumnya.

1. Proses Tanam *Watermark* (*Encode*)

Pada proses penanaman *watermark* (*encode*) penulis akan melakukan pengujian dengan menggunakan dua contoh citra sampul yang memiliki nilai koefisien warna yang berbeda. Sebelum melakukan proses *encode*, terlebih dahulu dilakukan penyeragaman terhadap citra-citra *input*an tersebut. Dalam hal ini penulis menggunakan ukuran 800x600 untuk citra sampul dan 128x128 untuk citra *watermark* dengan format citra adalah BMP 24-bit.

a. Pengujian pertama



Gambar 4.6 Citra Sampul Pertama

Gambar 4.7 Citra *Watermark* Pertama

Hasil proses setelah dilakukan penanaman *watermark* (*encode*)

Gambar 4.8 Citra Ter*watermark* Pertama

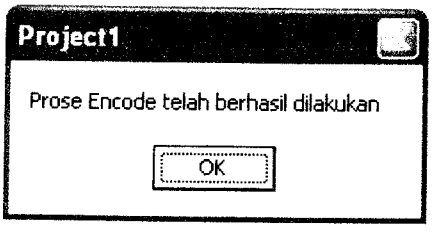
b. Pengujian kedua

Gambar 4.9 Citra Sampul Kedua

Gambar 4.10 Citra *Watermark* Kedua

Gambar 4.11 Citra Ter*watermark* Kedua

Pada form tanam *watermark* (*encode*), apabila *button encode* di klik maka sistem akan melakukan proses penanaman *watermark* kedalam image sampul. Dalam hal ini apabila proses telah berhasil dilakukan maka sistem akan menampilkan *messagebox* seperti pada gambar 4.12

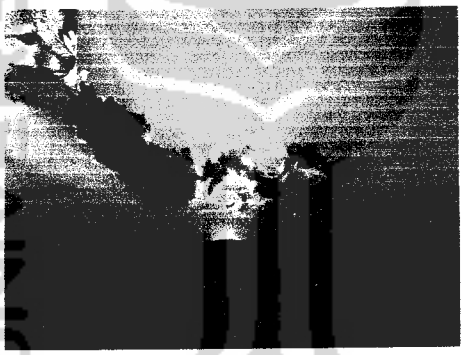


Gambar 4.12 Tampilan Jendela Dialog Jika Proses *Encode* Telah Berhasil Dilakukan

Secara kasat mata citra sampul dan citra ter*watermark* tidak memiliki perbedaan, namun jika dilakukan proses *decode* maka akan nampak *watermark* yang ada pada citra ter*watermark*.

2. Proses Estrak *Watermark* (*Decode*)

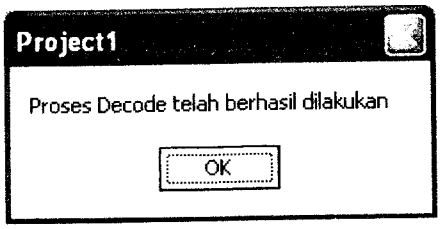
Pada bagian ini citra yang telah disisipi *watermark* akan diproses, sehingga *watermark* yang ada pada citra dapat diketahui dan ditampilkan. Gambar 4.13 dan 4.14 menerangkan hasil dari proses *decode* citra ter*watermark* pada pengujian pertama dan kedua.



Gambar 4.13 Citra Hasil *Decode* Pengujian Pertama

Gambar 4.14 Citra Hasil *Decode* Pengujian Kedua

Bila proses *decode* telah berhasil dilakukan, maka sistem akan memunculkan *messagebox* seperti pada gambar 4.15



Gambar 4.15 Tampilan Jendela Dialog Jika Proses *Decode* Telah Berhasil Dilakukan

4.3 Analisis Ketahanan *watermark*

Telah dijelaskan sebelumnya, salah satu dari unsur *watermarking* adalah ketahanan (*robustness*). Sebuah aplikasi *watermarking* yang baik harus tingkat ketahaan yang tinggi agar *watermark* yang ada pada sebuah citra tidak mudah rusak. Dalam penulisan tugas akhir ini, penulis akan mencontohkan tiga proses penyerangan pada citra ter*watermark* untuk membuktikan tingkat ketahanan dari aplikasi *watermarking* dengan menggunakan metode wavelet, yaitu rotasi, penghapusan dan pembalikan elemen warna. Pengujian dilakukan terhadap citra hasil *encode* pada proses pertama.

4.3.1 Rotasi

Pada bagian ini citra ter*watermark* akan di rotasi sebanyak 180^0 seperti terlihat pada gambar 4.16

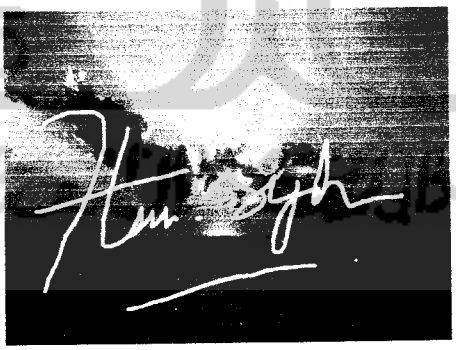
Gambar 4.16 Rotasi 180^0

Setelah dilakukan proses *decode* pada citra diatas maka hasilnya dapat dilihat pada gambar 4.17

Gambar 4.17 *Decode* Pada Citra Terotasi 180^0

4.3.2 Penghapusan

Pada bagian ini, citra ter*watermark* akan mengalami penghapusan pada sebagian elemen warnanya, seperti terlihat pada gambar 4.18



Gambar 4.18 Penghapusan Elemen Citra

Setelah dilakukan proses *decode* pada citra diatas, maka hasilnya dapat dilihat pada gambar 4.19

Gambar 4.19 *Decode* Pada Citra yang Tehapus Elemennya

4.3.3 Pembalikan Elemen Warna

Pada bagian ini, citra ter*watermark* akan mengelami penyerangan berupa perubahan warna seperti terlihat pada gambar 4.20

Gambar 4.20 Pembalikan Elemen Warna Citra

Setelah dilakukan proses *decode* pada citra diatas , maka *watermark* yang tersimpan dalam citra akan tampak seperti gambar 4.21

Gambar 4.21 *Decode* Pada Citra yang Telah Berubah Warna

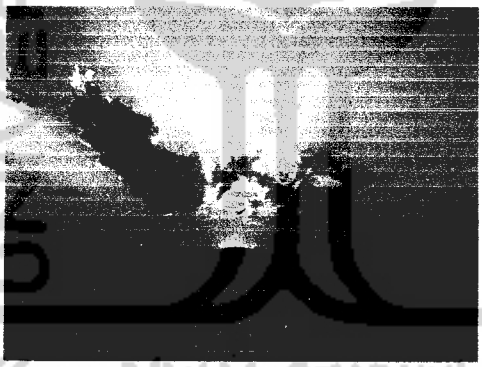
4.4 Perbandingan Citra

Perbandingan citra ini bertujuan untuk melihat perubahan yang terjadi pada citra sampul dan citra *watermark* setelah mengalami proses *encode* dan *decode*. Perbandingan ini dilakukan terhadap citra hasil *encode* dan *decode* pada pengujian pertama.

4.4.1 Perbandingan citra sampul

Setelah mengalami proses *encode*, citra sampul akan mengalami perubahan nilai koefisien pada bit-bit yang tidak signifikan, yang mengakibatkan perubahan citra itu sendiri, hal ini dikarenakan adanya citra *watermark* yang telah disisipkan di bit-bit tersebut. Namun secara kasat mata perbedaan tersebut sangat sulit untuk diketahui. Untuk lebih jelasnya mari perhatikan gambar 4.22 dan 4.23

Gambar 4.22 Citra Sampul Asli



Gambar 4.23 Gambar Citra Sampul Setelah Proses *Encode*

4.4.2 Perbandingan citra *watermark*

Citra *watermark* yang disisipkan pada citra sampul akan mengalami perubahan pada nilai-nilai koefisiennya, hal ini disebabkan proses transformasi wavelet Haar yang mengubah beberapa nilai koefisien dari citra ter*watermark*. Namun hal ini merupakan suatu kewajaran yang akan terjadi jika menggunakan transformasi wavelet dalam proses penyisipan *watermark* kedalam citra sampul. Untuk lebih jelasnya perhatikan gambar 4.24 dan 4.25

Gambar 4.24 *Watermark* Asli

Gambar 4.25 *Watermark* Ekstrak

4.5 Analisis Kinerja

4.5.1 Proses Tanam *Watermark* (*Encode*)

Proses *encode* dalam aplikasi *watermarking* dengan menggunakan metode wavelet ini sudah berjalan dengan baik, hal ini terbukti dengan tingkat kesamaan yang tinggi antara file citra asli (sebelum disisipi *watermark*) dengan file citra ter*watermark* (setelah disisipi *watermark*). Tingkat kesamaan tersebut dapat dilihat dari perubahan elemen warna yang tidak terlalu mencolok dan ukuran dimensi citra yang tidak berubah setelah dilakukan proses *encode*.

4.5.2 Proses Ekstrak *Watermark* (*Decode*)

Proses *decode* dalam aplikasi *watermarking* dengan menggunakan metode wavelet ini sudah berjalan dengan baik, hal ini terbukti dengan terbacanya file citra *watermark* yang ada pada file citra sampul setelah dilakukan proses *encode*. Namun elemen warna pada file citra *watermark* hasil ekstraksi sedikit mengalami perubahan. Perubahan ini merupakan suatu kewajaran yang merupakan dampak langsung dari proses transformasi wavelet. Selain pengujian dengan file citra ter*watermark* asli, analisis juga dilakukan pada file cita ter*watermark* yang sudah mengalami penyerangan (pengeditan). Dalam hal ini terbukti bahwa file citra *watermark* yang berada pada file citra sampul masih dapat dideteksi keberadaannya untuk beberapa serangan tertentu, seperti penghapusan dan pembalikan elemen warna. Namun untuk serangan seperti rotasi, file cita *watermark* sudah tidak dapat terbaca lagi.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang dapat penulis setelah melakukan penelitian tentang pembuatan aplikasi *watermarking* berbasis wavelet adalah :

1. Transformasi Wavelet dapat digunakan sebagai metode untuk membuat aplikasi *watermarking*.
2. Implementasi program penanaman *watermark* (*encode*) pada koefisien baris genap dan kolom genap pada citra hasil transformasi wavelet menyebabkan *watermark* tidak tampak oleh indra penglihatan manusia.
3. Program pengekstrakan pesan (*decode*) yang dibuat telah berhasil mengekstrak citra pesan yang disisipkan.

5.2 Saran

Secara umum program yang dibuat masih belum sempurna, karena itu beberapa saran yang ingin disampaikan, yaitu :

1. Karena aplikasi ini hanya menggunakan citra berformat BMP 24-bit, maka perlu dikembangkan penelitian yang mengimlpementasikan aplikasi *watermarking* berbasis wavelet Haar untuk citra berformat lain.
2. Perlu dilakukan pengembangan lebih lanjut terhadap aplikasi yang dibangun agar sorfware aplikasi *watermarking* ini bisa lebih sempurna.

DAFTAR PUSTAKA

[AND03] Andino masalino. "***Pengantar Steganografi***", http:// www.ilmukomputer.com kuliah_umum.html, diakses pada 22 November 2006.

[ASN01] Asnawi, Choerun 2001. Implementasi Steganografi Berbasis wavelet Haar pada File Citra. *skripsi*, tidak diterbitkan. Yogyakarta: Fakultas Matematika dan Ilmu Pengetahuan alam, Universitas Gaja Mada.

[BEN96] Bender, D. Gruhl, N. Morimoto, A. Lu 1996, Techniques for data hiding, IBM System Journal, Vol. 35.

[DAV02] David Andriyano 2002, Watermark Beramplitudo Tinggi pada Sinyal Suara, *Tesis,* Tidak diterbitkan. Bandung: Magister Bidang Khusus Teknologi Informasi, Program Studi Telekomunikasi, Program Pasca Sarjana, Institut Teknologi Bandung.

[JOH01] Johnson, N.F, 2001,***"Introduction To Steganography : Hidden Information"***, http://www.crazytrain.com/monkeyboy/rude_johnson_gmu2001_stegob.pdf, diakses pada 15 Januari 2007.

[PRA02] Prayudi, Yudi."***Metode Watermarking Ganda Sebagai Teknik Pengamanan Citra Digital"***. Program Studi Informatika, Program Pasca Sarjana, Institut Teknologi Sepuluh November, Surabaya, 2002.

[SET05] Setiawan, Agus 2005 Implementasi watermarking pada citra menggunakan discrite consine Transform. *skripsi*, tidak diterbitkan. Yogyakarta: Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Gaja Mada.

[SUH00] Suhono H. Supangkat, Kuspriyanto, Juanda 2000. ***"Watermarking sebagai Teknik Penyembunyian Label Hak Cipta pada Data Digital"***. Bandung: Departemen Teknik Elektro, Institut Teknologi Bandung.

[SUL05] Sulianti, A 2005. Pengenalan Wajah Pada Citra Digital Dengan Menggunakan Wavelet Doubechies, *skripsi*, tidak diterbitkan. Yogyakarta: Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Gajah Mada.

[WAT94] Watson,A., B., 1994, ***"Image Compression Using the Discrete Consine Transform"***. Mathematic journal, 4(1), pp 81-88, http://vision.arc.nasa.gov/publication/mathjournal94.pdf, diakses pada 15 Januari 2007.