ALGORITMA WATERMARKING MENGGUNAKAN PENDEKATAN *FAST FOURIER TRANSFORM* DAN *DISCRETE WAVELET TRANSFORM*

SKRIPSI

MUHAMMAD SUDIN NUR

H 111 13 024

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2017

ALGORITMA WATERMARKING MENGGUNAKAN PENDEKATAN *FAST FOURIER TRANSFORM* DAN *DISCRETE WAVELET TRANSFORM*

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains
pada Program Studi Matematika Departemen Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Hasanuddin

MUHAMMAD SUDIN NUR

H 111 13 024

PROGRAM STUDI MATEMATIKA
DEPARTEMEN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS HASANUDDIN
MAKASSAR
2017

LEMBAR KEOTENTIKAN

Saya yang bertanda tangan di bawah ini menyatakan dengan sesungguh-sungguhnya bahwa skripsi yang saya buat dengan judul :

ALGORITMA WATERMARKING MENGGUNAKAN PENDEKATAN *FAST FOURIER TRANSFORM* DAN *DISCRETE WAVELET TRANSFORM*

Adalah benar-benar hasil karya dan pekerjaan saya sendiri, bukan hasil plagiat dan belum pernah dipublikasikan sebelumnya.

Makassar, 13 November 2017

MUHAMMAD SUDIN NUR

NIM : H 111 13 024

iii

ALGORITMA WATERMARKING MENGGUNAKAN PENDEKATAN *FAST FOURIER TRANSFORM* DAN *DISCRETE WAVELET TRANSFORM*

MUHAMMAD SUDIN NUR

H 111 13 024

Disetujui oleh :

Pembimbing Utama	Pembimbing Pertama
Dr. Hendra, S.Si, M.Kom	Dr. Eng. Armin Lawi, M. Eng
NIP. 19760102 200212 1001	NIP. 19720423 199512 1001

Pada Tanggal : 13 November 2017

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :

Nama : MUHAMMAD SUDIN NUR

NIM : H111 13 024

Program Studi : MATEMATIKA

Judul Skripsi :Algoritma Watermarking Menggunakan Pendekatan *Fast Fourier Transform* dan *Discrete Wavelet Transform*

Telah berhasil dipertahankan dihadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Sains pada Program Studi Matematika Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

DEWAN PENGUJI

		Tanda Tangan
1. Ketua	: **Dra. Nur Erawaty, M.Si**	(.....)
2. Sekretaris	: **Prof. Dr. Eng Mawardi Bahri, M.Si**	(.....)
3. Anggota	: **Dr. Erna Tri Herdiani, M.Si**	(.....)
4. Anggota	: **Dr. Hendra, S.Si, M.Kom**	(.....)
5. Anggota	: **Dr. Eng. Armin Lawi, M. Eng.**	(.....)

Ditetapkan di : Makassar

Tanggal : 13 November 2017

KATA PENGANTAR

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ

Segala puji hanya milik Dzat yang Maha Mulia, Alloh Subhanahu Wa Ta'ala, yang ditangan-Nya tergenggam nyawa seluruh makhluk semesta alam, yang Maha kekal sebelum sesuatunya ada, dan akan tetap kekal setelah segala sesuatunya tiada. Shalawat dan salam kepada Baginda Rosulullah Shollallohu Alaihi Wasallam, kekasih Alloh juga para ahlul bait dan para sahabat-sahabat beliau yang senantiasa kita rindukan perjumpaan dengannya.

Hanya dengan taufik dan hidayah-Nya penulis dapat menyelesaikan skripsi yang berjudul "Algoritma Watermarking Menggunakan Pendekatan *Fast Fourier Transform* dan *Discrete Wavelet Transform"*. Penyusunan tugas akhir ini tentunya tidak lepas dari bantuan berbagai pihak baik moril maupun materil. Oleh karena itu, penulis menyampaikan ucapan terima kasih yang tulus dan penghargaan yang tak terhingga kepada Ayahanda dan Ibunda tercinta **La Ode Samaria** dan **Wa Ono** yang telah menjadi penyemangat kepada penulis untuk menggapai cita-cita. Untuk saudaraku **Rahman Nuwirno** terima kasih banyak atas motivasi yang telah diberikan kepada penulis.

Penghargaan yang tulus dan ucapan terima kasih dengan penuh keikhlasan juga penulis ucapkan kepada :

1. Ibu **Rektor Universitas Hasanuddin** beserta jajarannya, Bapak **Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam** beserta jajarannya dan semua pihak birokrasi atas ilmu dan kemudahan-kemudahan yang diberikan, baik dibidang akademik maupun dibidang kemahasiswaan.
2. Bapak **Prof. Amir Kamal Amir, M.Sc.** selaku Ketua Departemen Matematika FMIPA UNHAS, **Segenap dosen pengajar** yang telah memberikan ilmu kepada penulis, dan Terima kasih juga untuk segenap jajaran Pegawai Akademik Departemen Matematika atas bantuannya dalam pengurusan akademik selama ini.

3. Bapak **Dr. Hendra, S.Si, M.Kom** selaku pembimbing utama dan Bapak **Dr. Eng. Armin Lawi, M. Eng.** selaku pembimbing pertama. Terima kasih atas segala ilmu, dan nasehat yang diberikan, serta kesabaran dalam mengarahkan penulis menyelesaikan tugas akhir ini. Terima kasih telah meluangkan waktu untuk membimbing penulis.
4. Ibu **Dra. Nur Erawaty, M.Si** selaku Ketua Tim Penguji dalam penulisan tugas akhir ini. Terima kasih atas segala masukan dan motivasi yang diberikan kepada penulis selama menjalani pendidikan di Jurusan Matematika.
5. Bapak **Prof. Dr. Eng Mawardi Bahri, M.Si** selaku Sekertaris Tim Penguji. Terima kasih telah memberikan kritikan yang membangun dalam penyempurnaan penyusunan tugas akhir ini serta segala *support* dan waktu yang telah diberikan kepada penulis.
6. Ibu **Dr. Erna Tri Herdian, M.Si** selaku Anggota Tim Penguji sekaligus dosen Penasehat Akademik penulis. Terima kasih atas segala koreksi dan saran yang diberikan dalam penyusunan tugas akhir ini.
7. Murobbiku kanda Hidayatullah Wirasanda. Terima kasih atas segala ilmu dan dukungan morilnya .
8. Keluargaku tercinta di kampus **Binomial 2013** kalian luar biasa telah memberikan banyak pelajaran dan sejarah dalam hidupku banyak cobaan yang telah dilewatkan selama menjadi MABA, PANITIA dan PENGURUS.
9. Kanda-kanda di **Himpunan Mahasiswa Matematika** FMIPA Unhas. Bangga rasanya bisa bergabung bersama HIMATIKA. “Bravo HIMATIKA!”.
10. Adik – adikku dikampus yaitu : **Transpose 2014** , **Simetris 2015** , serta **Algoritma 2016** , yang selalu memberiku semangat dan membuatku belajar bagaimana caranya jadi pengader yang baik .
11. Saudara – saudaraku **MIPA 2013** , serta pengurus **BEM FMIPA** Unhas periode 2016 / 2017 , **Maperwa FMIPA** Unhas periode 2017 / 2018. Kalian luar biasa , kalian mengajarkanku apa artinya kebersamaan .

Serta kepada seluruh pihak yang tidak dapat penulis sebutkan satu persatu, terima kasih untuk semuanya. Semoga apa yang telah dituliskan oleh penulis pada skripsi ini dapat bermanfaat bagi sesama.

Makassar, 13 November 2107

Penulis

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini :

Nama : Muhammad Sudin Nur

NIM : H 111 13 024

Program Studi : Matematika

Jurusan : Matematika

Fakultas : Matematika dan Ilmu Pengetahuan Alam

Jenis Karya : Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*) atas karya ilmiah saya yang berjudul :

"ALGORITMA WATERMARKING MENGGUNAKAN PENDEKATAN *FAST FOURIER TRANSFORM* DAN *DISCRETE WAVELET TRANSFORM*"

Beserta perangkat yang ada (jika diperlukan). Terkait dengan hal di atas, maka pihak universitas berhak menyimpan, mengalih-media/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di Makassar Pada tanggal 13 November 2017

Yang menyatakan

(Muhammad Sudin Nur)

ABSTRAK

Perkembangan teknologi jaringan yang begitu pesat menyebabkan data multimedia seperti gambar, audio, dan video lebih mudah untuk didistribusikan ataupun disalin secara legal maupun illegal. Pengaksesan hak cipta oleh pengguna yang tidak bertanggung jawab dalam mendistribusikan informasi digital sangat merugikan bagi pemilik hak cipta, Untuk itu dibutuhkan suatu metode *watermarking* dalam melindungi informasi hak cipta diantaranya adalah metode *Fast Fourier Transform* (FFT) dan *Discrete Wavelet Transform* (DWT). Dalam penelitian ini dilakukan penggabungan metode *Fast Fourier Transform* (FFT) dan *Discrete Wavelet Transform* (DWT), dengan cara menyisipkan gambar *watermark* ke dalam citra asli untuk mendapatkan citra ter-*watermark*. Hasil simulasi menunjukkan bahwa untuk citra asli yang di *watermark* pada dekomposisi tingkat tiga menunjukkan hasil yang paling baik dengan masing-masing nilai PSNR 29.3374 db, 38.9447 db, 38.9447 db, 23.2427 db, 38.9399 db dan masing-masing nilai NC hasil ekstraksi 0.974343, 0.965329, 0.966879, 1, 0.984348. Pengujian citra ter-*watermark* terhadap beberapa serangan yakni *Compression, Salt* dan *Pepper, Gausion, Speckle, Resizing,* dan *Rotation* menghasilkan penurunan kualitas citra hasil ekstraksi akan tetapi data penyisip masih bisa dilihat secara kasat mata.

Kata kunci : *Watermarking, Discrete Wavelet Transform, Fast Fourier Transform, Compression, Salt* dan *Pepper, Gausion, Speckle, Resizing, Rotation* , PSNR, NC.

ABSTRACT

The rapid development of network technology causes multimedia data such as images, audio, and video easier to be distributed or copied legally or illegally. Accessing copyright by irresponsible users in distributing digital information is very detrimental to copyright owners. Therefore, it needed a method of watermarking in protecting copyright information such as *Fast Fourier Transform* (*FFT*) and *Discrete Wavelet Transform* (*DWT*) method. This research is done by combine of *Fast Fourier Transform* (*FFT*) and *Discrete Wavelet Transform* (*DWT*) method and embedding the watermark image into the original image to research the watermarked image. The simulation results show that for the original watermarked image of the third level decomposition shows the best result with each PSNR value 29.3374 db, 38.9447 db, 38.9447 db, 23.2427 db, 38.9399 db and each NC value of extraction result 0.974343, 0.965329, 0.966879, 1, 0.984348. Testing of watermarked images of several attacks namely Compression, Salt and Pepper, Gaussian, Speckle, Resizing, and Rotation resulting in decreased image quality of extraction results but the insert data can still be seen by naked eye.

Keywords: *Watermarking, Discrete Wavelet Transform, Fast Fourier Transform, Compression, Salt and Pepper, Gausion, Speckle, Resizing, Rotation, PSNR, NC.*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kemajuan teknologi di zaman modern saat ini memberikan kemudahan bagi pengguna dalam mengakses lebih mudah berbagai macam media informasi yang disediakan. Pengaksesan informasi yang disediakan yakni dalam bentuk digital seperti gambar, audio, video lebih mudah untuk didistribusikan ataupun disalin secara legal maupun illegal. Pengaksesan informasi oleh pengguna yang tidak bertanggung jawab dalam mendistribusikan informasi digital tersebut sangat merugikan bagi pemilik informasi, Untuk itu dibutuhkan suatu metode dalam melindungi informasi digital tersebut.

Watermarking digital merupakan metode yang relevan untuk saat ini, karena dengan metode ini memungkinkan data digital yang asli dapat diketahui. *watermarking* digital merupakan cabang ilmu yang mempelajari bagaimana teknik menyembunyikan data atau informasi yang bersifat rahasia ke dalam media informasi yang lainnya. *Watermarking* digital adalah proses menyembunyikan informasi menjadi konten multimedia digital sehingga informasi dapat diekstraksi atau terdeteksi untuk berbagai keperluan termasuk pencegahan dan pengendalian dalam mengcopy suatu data digital (Sugiono, 2008).

Dalam penulisan ini diperkenalkan algoritma *watermarking* citra pada domain transformasi yakni menggunakan pendekatan *FFT (Fast Fourier Transform)* dan *DWT (Discrete Wavelet Transform).* Dalam metode ini pada awalnya citra yang *watermarking* akan di transformasikan menjadi discrete. Citra awal didekomposisi melalui transformasi *DWT,* kemudian akan dipilih sub-level wavelet pada tingkat tertentu. Sub-level yang terpilih akan ditransformasi menjadi Transformasi Fourier. Informasi *watermarking* yang disisipkan akan di tempatkan pada posisi yang sesuai, Setelah itu citra ditransformasikan menggunakan *IFFT* dan *IDWT* untuk mendapatkan citra *watermark* serta hasil *ekstraksi* nya.

Berdasarkan hal tersebut, Penulis tertarik dalam membuat suatu penulisan mengenai "*Algoritma Watermarking menggunakan pendekatan Fast Fourier Transform dan Discrete Wavelet Transform*".

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, maka dapat dirumuskan masalah yang akan di bahas dalam penulisan ini yaitu :

1. Bagaimana teknik penyisipan *watermark* ke citra host menggunakan metode *FFT* dan *DWT*?
2. Bagaimana kualitas *watermarking* yang dihasilkan pada penggunaan sub level 1,2 & 3 dengan metode *watermaking DWT* ?
3. Bagaimana kualitas hasil ekstraksi *watermarking* dengan berbagai jenis serangan pada ketiga sub level yang digunakan ?

1.3 Batasan Masalah

Dalam penelitian ini penulis membuat batasan masalah yaitu:

1. File citra penampung (*host*) adalah citra *grayscale.*
2. File citra penyisip (*embed*) adalah citra *monokrom* (hitam putih)
3. File citra penampung dan citra penyisip berformat .jpg
4. Jenis Serangan yang digunakan yaitu *Compression, Salt* dan *Pepper, Gausian, Speckle, Resizing,* dan *Rotation.*

1.4 Tujuan Penelitian

1. Menguraikan proses penyisipan *watermark* ke citra host menggunakan metode *FFT* dan *DWT*.
2. Menganalisis kualitas hasil *watermarking* pada sub level 1,2,dan 3 pada *DWT*.
3. Menganalisis Kualitas hasil ekstraksi *watermark* setelah dilakukan pengujian dengan beberapa serangan.

1.5 Manfaat Penelitian

Manfaat Penulisan ini adalah:

1. Memahami teknik penyisipan berformat gambar atau logo kedalam suatu file citra dengan menggunakan metode *FFT* dan *DWT*.
2. Mengetahui ketahanan *watermark* terhadap beberapa serangan atau tanpa serangan yang diujikan dengan melihat nilai PSNR dan NC.

BAB II

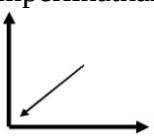
TINJAUAN PUSTAKA

II.1 Citra Digital

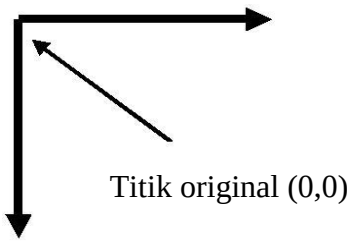
Citra digital adalah gambar dua dimensi yang dapat ditampilkan pada layar monitor komputer sebagai himpunan berhingga (diskrit) nilai digital yang disebut dengan piksel (*picture elements*) (Fahmi, 2007).

Suatu citra adalah fungsi intensitas 2 dimensi f(*x*,*y*), dimana *x* dan *y* adalah koordinat spasial dan f pada titik (*x*,*y*) merupakan tingkat kecerahan (*brightness*) suatu citra pada suatu titik. Suatu citra diperoleh dari penangkapan kekuatan sinar yang dipantulkan oleh objek. Citra digital tersusun atas sejumlah berhingga elemen, masing-masing memiliki lokasi dan nilai atau intensitas tertentu. Elemen-elemen ini disebut elemen gambar, elemen citra, dan juga piksel yang dinyatakan dalam bilangan bulat. Tingkat ketajaman atau resolusi warna pada citra digital tergantung pada jumlah ”bit” yang digunakan oleh komputer untuk merepresentasikan setiap piksel tersebut. Tipe yang sering digunakan untuk merepresentasikan citra adalah ”8-bit citra” (256 *colors* (0 untuk hitam - 255 untuk putih)), tetapi dengan kemajuan teknologi perangkat keras grafik, kemampuan tampilan citra di komputer hingga 32 bit (232.000 warna) (Sitorus, 2008).

Piksel (0,0) terletak pada sudut kiri atas pada citra, indeks *x* begerak ke kanan dan indeks *y* bergerak ke bawah. Konvensi ini dipakai merujuk pada cara penulisan larik yang digunakan dalam pemrograman komputer. Letak titik origin pada koordinat grafik citra dan koordinat pada grafik matematika terdapat perbedaan. Hal yang berlawanan untuk arah vertikal berlaku pada kenyataan dan juga pada sistem grafik dalam matematika yang sudah lebih dulu dikenal. Gambar II.1 dan II.2 berikut memperlihatkan perbedaan kedua sistem ini.



Gambar II.1 Koordinat pada grafik matematika (Sitorus, 2008)



Gambar II.2 Koordinat pada citra (Sitorus, 2008)

II.1.1 Format file citra

Format file menentukan bagaimana informasi data direpresentasikan dalam suatu file. Informasi tersebut meliputi ada tidaknya kompresi, program aplikasi (*feature*) yang didukung (*support*), penggunaan enkripsi, dan lain-lain. Tiap format file memiliki kelebihan dan kelemahan masing-masing (Supangkat, 2000).

Menurut Supangkat (2000), dalam sistem operasi Windows, format file dapat dibedakan dari namanya yaitu diakhiri titik dan diikuti dengan tiga atau empat huruf terakhir sebagai penanda format. Untuk file citra (*image*), format yang umum digunakan adalah :

1. *bmp* (*Windows Bitmap*).

   Merupakan representasi dari citra grafis yang terdiri dari susunan titik-titik yang tersimpan di memori komputer. Format ini dikembangkan oleh *Microsoft* dan nilai setiap titik diawali oleh satu bit data untuk gambar hitam putih atau lebih bagi gambar berwarna. Format *bmp* menggunakan kompresi tipe *lossless*, berarti tidak ada data yang dibuang selama proses kompresi.

2. *gif* (*Graphics Interchange Format*)

   Merupakan format gambar yang mampu menayangkan maksimum 256 warna dan mendukung warna transparan dan animasi sederhana. Format ini mengkompresi gambar dengan sifat *lossless*, berarti terdapat data yang hilang selama proses kompresi.

3. *jpg* / *jpeg* (*Joint Photographic Experts Group*)

   Format ini mampu menayangkan warna dengan kedalaman 24 bit *true color* dan menggunakan kompresi tipe *lossy*. Kualitas *jpeg* bisa bervariasi tergantung setting kompresiyang digunakan.

4. *png* (*Portable Network Graphics*)

   Merupakan salah satu format penyimpanan citra dengan kompresi tipe *lossless*. Format *png* diperkenalkan untuk menggantikan format *gif* dan umumnya dipakai untuk citra *web*.

5. *tiff* (*Tagged Image File Format*)

   Merupakan format yang sering digunakan, mendukung citra *compressed* berbagai metode dan *uncompressed* .

II.1.2 Klasifikasi Citra Digital

Berdasarkan cara penyimpanan atau pembentukannya, citra digital dibagi menjadi 2 jenis, yaitu :

1. Gambar Bitmap, yaitu gambar yang terbentuk dari sekumpulan titik penyusun gambar (piksel). Gambar bitmap dipengaruhi oleh banyaknya piksel, sehingga semakin banyak jumlah piksel maka kualitas gambar semakin baik dan halus, begitu pula sebaliknya. Gambar bitmap biasanya diperoleh dari scanner, kamera digital, kamera *handphone*, dan sebagainya (Sitorus, 2008).
2 Gambar vektor, yaitu gambar yang terbentuk dari garis, kurva, dan bidang yang masing-masing merupakan suatu formulasi matematik. Jika gambar vektor diperbesar, maka kualitas gambarnya masih tetap baik dan tidak berubah. Gambar vektor biasanya dibuat dengan menggunakan aplikasi – aplikasi gambar vektor seperti *Corel Draw*, *Adobe Illustrator*, *MacromediaFreehand*, dan sebagainya (Sitorus, 2008).

Sedangkan berdasarkan warna-warna penyusunnya, menurut Fahmi (2007) citra digital dapat dibagi menjadi tiga macam, yaitu :

1 Citra biner, yaitu citra yang hanya terdiri atas dua warna, yaitu hitam dan putih. Oleh karena itu, setiap piksel pada citra biner cukup

direpresentasikan dengan 1 bit.

2 Citra *grayscale*, yaitu citra yang nilai piksel-nya merepresentasikan derajat keabuan atau intensitas warna putih. Nilai intensitas paling rendah merepresentasikan warna hitam dan nilai intensitas paling tinggi merepresentasikan warna putih. Pada umumnya citra *grayscale* memiliki kedalaman piksel 8 bit (256 derajat keabuan), tetapi ada juga citra *grayscale* yang kedalaman piksel-nya bukan 8 bit, misalnya 16 bit untuk penggunaan yang memerlukan ketelitian tinggi.

3 Citra berwarna, yaitu citra yang nilai piksel-nya merepresentasikan warna tertentu. Banyaknya warna yang mungkin digunakan bergantung kepada kedalaman piksel citra yang bersangkutan. Citra berwarna direpresentasikan dalam beberapa kanal (*channel*) yang menyatakan komponen-komponen warna penyusunnya. Banyaknya kanal yang digunakan bergantung pada model warna yang digunakan pada citra tersebut. Contoh model warna yang biasa digunakan pada citra digital adalah RGB dan YCbCr.

Dari klasifikasi citra tersebut, maka penelitian dilakukan pada gambar bitmap karena citra digital termasuk gambar bitmap dan terbentuk dari piksel – piksel yang akan dimanfaatkan dalam teknik penyisipan *watermark*.

II.2 *Watermarking*

Watermarking merupakan sebuah proses penambahan kode secara permanen ke dalam citra digital. Penyisipan kode ini harus memiliki ketahanan (*robustness*) yang cukup baik dari berbagai manipulasi, seperti pengubahan, transformasi, kompresi, maupun enkripsi. Kode yang disisipkan juga tidak merusak citra digital sehingga citra digital terlihat seperti aslinya. *Watermarking* dapat juga merupakan cara untuk menyisipkan *watermark* kedalam media yang ingin dilindungi hak ciptanya. *Watermarking* merupakan proses penanaman *watermark*. Digital *Watermarking* merupakan cara yang digunakan untuk menyisipkan informasi atau *watermark* pada suatu dokumen digital. Dari defenisi-definisi diatas dapat penulis simpulkan bahwa *watermarking* merupakan cara

untuk menyisipkan *watermark* atau proses penambahan kode secara permanen ke dalam citra digital yang ingin dilindungi hak ciptanya dengan tidak merusak citra aslinya dan tahan terhadap serangan (Sugiono, 2008).

Watermark merupakan sebuah pola atau kode atau data tertentu yang membawa informasi tertentu sesuai dengan tujuannya dan sengaja ditanamkan secara permanen kedalam data media induknya. *Watermark* dalam citra digital tersebut tidak dapat diketahui keberadaannya oleh pihak lain yang tidak mengetahui rahasia skema penyisipan *watermark*. *Watermark* tersebut juga tidak dapat diidentifikasi dan dihilangkan. Penggunaan *watermarking* sangat diperlukan untuk melindungi karya intelektual digital seperti gambar, teks, musik, video, dan termasuk perangkat lunak. Penggandaan atas produk digital yang dilakukan oleh pihak-pihak yang tidak bertanggung jawab semakin merajalela tanpa ada ikatan hukum yang pasti sehingga merugikan pemegang hak cipta akan produk digital tersebut. Oleh karena itu, penyisipan *watermark* memiliki peran yang cukup signifikan untuk mencegah terjadinya penggandaan terhadap produk digital (Sugiono, 2008).

Label *watermark* adalah sesuatu data atau informasi yang akan dimasukkan kedalam data digital yang ingin dilakukan proses *watermarking*. Ada 2 jenis label *watermark* yang dapat digunakan:

1. Teks biasa

   Label *watermark* dari teks biasanya menggunakan nilai-nilai ASCII dari masing-masing karakter dalam teks yang kemudian dipecahkan atas bit per bit. Kelemahan dari label ini adalah kesalahan pada satu bit saja akan menghasilkan hasil yang berbeda dari teks sebenarnya.

2. Citra atau suara

   Berbeda dengan teks, kesalahan pada beberapa bit masih dapat memberikan persepsi yang sama dengan aslinya, baik oleh pendengaran maupun penglihatan kita.

Oleh karena itu, penyisipan logo sebagai label *watermark* dirasakan lebih efektif dibandingkan teks, citra, ataupun suara karena selain tidak sensitif terhadap kesalahan bit, ukuran *file* juga tidak terlalu besar. Logo yang dipakai berupa logo

biner atau hitam putih karena komputasi yang dibutuhkan tidak terlalu rumit namun tetap menjamin visualisasi yang cukup baik (Sugiono, 2008).

II.2.1 Klasifikasi *Watermarking*

Menurut Agrawal (2013), setiap aplikasi watermarking mempunyai ciri tersendiri. Ada beberapa persyaratan yang harus dipenuhi pada teknik watermarking :

1. *Perceptual Transparency*

   Sebagian besar teknik watermarking harus menanamkan watermark yang tidak terlalu mempengaruhi kualitas data host. Prosedur watermarking seharusnya membuat teknik dimana manusia tidak dapat membedakan antara data asli dan data yang telah disisipkan watermark. Tetapi dalam watermarking modifikasi kecilpun pada data asli bisa dibedakan apabila dibandingkan dengan data asli. Karena pengguna data watermark biasanya tidak memiliki akses ke data asli, sehingga mereka tidak dapat melakukan perbandingan. Oleh karena itu, mungkin cukup bahwa modifikasi dalam data tidak diketahui selama data tidak dibandingkan dengan data asli.

2. *Robustness* (kekokohan)

   Sangat diharapkan bahwa data watermark selalu ada pada host, bahkan jika kualitas data host diturunkan. Contohnya kompresi teknik lossy, filtering, re-sampling, kompresi data digital-analog (D/A) dan analog-digital (A/D).

Berdasarkan kenampakan, *Watermarking* dapat dikelompokkan dalam beberapa kategori

1. *Visible Watermarking*

   Pada *visible watermarking* ini, *watermark* yang disisipkan pada suatu media terlihat dengan jelas. *Watermark* biasanya berbentuk logo atau teks baik transparan atau tidak yang diletakkan tidak mengganggu atau menutupi media asal. Jenis *watermarking* ini biasanya diterapkan pada media yang memang dimaksudkan untuk disebar secara umum bersama dengan identitas pemilik asal media tersebut.

2. *Invisible Watermarking*

   Sesuai namanya, *watermark* pada *invisible watermarking* yang disisipkan pada media tidak lagi dapat dipersepsi dengan indera. Namun, keberadaannya tetap dapat dideteksi. Penerapan teknik *invisible watermarking* ini lebih sulit dari pada teknik yang digunakan pada *visible watermarking*.

Selain itu, *watermark* juga dikategorikan berdasarkan kekuatan *watermark* yang ada pada media. Adapun kategorinya yaitu

1. *Fragile Image Watermarking*

   Fragile image watermarking merupakan jenis *watermark* yang ditujukan untukmenyisipkan label kepemilikan media digital. Pada *fragile watermarking* ini, *watermark* mudah sekali berubah atau bahkan hilang jika dilakukan perubahanterhadap media digital. Dengan begitu, media digital sudah tidak lagi memiliki *watermark* yang asli. *Fragile image watermarking* ini biasanya digunakan agardapat diketahui apakah suatu *image* sudah berubah atau masih sesuai aslinya. Jenis *watermark* inilah yang banyak diterapkan pada suatu media digital.

2. *Robust Image Watermarking*

   Robust image watermarking adalah teknik penggunaan *watermark* yangditujukan untuk menjaga integritas atau orisinalitas media digital. *Watermark* yang disisipkan pada media akan sangat sulit sekali dihapuskan atau dibuang. Dengan *Robust Image*, proses penggandaan media digital yang tidak memiliki izin dapat dihalangi. Kebanyakan aplikasi dari *robust watermarking* ini bukan pada sebuah media digital, melainkan pada sistem proteksi CD atau DVD. (Aliwa. 2009)

Untuk memperlihatkan apakah dalam suatu citra terdapat *watermark* atau tidak, maka digunakan pendekatan nilai PSNR *(Peak Signal to Noise Ratio),* yaitu ukuran distorsi yang timbul pada citra dikarenakan adanya *watermark* di dadalamnya. Pengukuran *fidelity* dapat dihitung dengan menghitung nilai MSE *(Mean Squared Error)* dan PNSR *(Peak Signal to Noise Ratio).* MSE dan PSNR dapat dihitung dengan menggunakan persamaan (2.1) dan (2.2). Pada persamaan

(2.1), I (x,y) adalah nilai *gray* level citra asli di posisi (x,y), I' adalah nilai derajat keabuan citra yang telah diberi *watermark* di posisi (x,y), X dan Y adalah ukuran panjang dan lebar. Pada persamaan (2.2), m adalah nilai maksimum yang mungkin dimiliki oleh sebuah piksel. Sebagai contoh, untuk data citra 8 bit, nilai maksimumnya adalah 255 (Fahmi, 2007).

$$MSE = \frac{1}{XY} \sum_{x=1}^{X} \sum_{y=1}^{Y} [I(x,y) - I'(x,y)]^2 \qquad (2.1)$$

$$PSNR = 10 * log_{10} \frac{m^2}{MSE} \qquad (2.2)$$

Perhitungan PSNR untuk citra warna 24 bit sama dengan citra 8 bit, kecuali MSE dijumlahkan dari tiap komponen dibagi 3 yang ditunjukkan pada persamaan berikut:

$$PSNR = 10 * log_{10} \left(\frac{255^2}{\frac{MSE(R)+MSE(G)+MSE(B)}{3}} \right) \quad , \qquad (2.3)$$

dengan,

MSE(R) adalah MSE untuk komponen warna *Red* (merah)

MSE(G) adalah untuk komponen warna *Green* (hijau)

MSE(B) adalah untuk komponen warna *Blue* (biru).

Nilai PSNR yang tinggi adalah lebih baik karena berarti rasio sinyal terhadap noise juga tinggi. Untuk mengukur kemiripan label *watermark* asli dan label *watermark* hasil ekstraksi secara kuantitatif digunakan *Normalized Cross Correlation* (NC) yang didefinisikan pada rumus :

$$NC = \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{M-1} W(i,j)W'(i,j)}{\sum_{i=0}^{M-1} \sum_{j=0}^{M-1} [W(i,j)]^2} \quad , \qquad (2.4)$$

dengan ,

M x M adalah ukuran dari label *watermark*

W' adalah label *watermark* hasil ekstraksi

W adalah label *watermark* asli.

II.3 Serangan *Watermark*

Menurut Sianipar (2013), serangan dapat menyebabkan terjadinya perubahan bit pada citra ter-*watermark*, sehingga apabila dilakukan ekstraksi, maka *watermark* yang telah dipisahkan akan tampak sangat berbeda dibandingkan dengan *watermark* yang disisipkan pada awalnya. Ada beberapa serangan yang biasa dilakukan pada citra ter-*watermark* seperti Compression ,Salt & Pepper, Gaussion, Speckle, *Scaling, Rotation*.

1. *Compression*

   Serangan ini juga merupakan serangan yang sering dilakukan secara tidak sengaja. Kompresi sering dilakukan pada *file* multimedia seperti audio, video, dan citra. *Watermark* yang disisipkan biasanya lebih tahan terhadap kompresi yang memliki *domain* sama dengan *domain* yang dipakai pada saat *watermarking*. Misalnya citra yang disisipi *watermark* menggunakan *DCT* (*Discrete Cosine Transform*) lebih tahan terhadap kompresi JPEG daripada citra yang disisipi *watermark* dalam *domain* spasial atau citra yang disispi *watermark* menggunakan *DWT (Discrete Wavelet Transform)* lebih kuat terhadap kompresi JPEG2000.

2. Salt & Pepper

   Serangan ini disebut pula dengan serangan impuls, serangan shot, atau serangan biner. Serangan ini bias disebabkan oleh gangguan tiba-tiba dalam sinyal citra, penampakannya berupa warna hitam atau putih yang secara acak menyebar. Untuk mendemonstrasikan serangan tipe ini, pertama tama akan dibangkitkan suatu citra abu-abu, yang di mulai dengan citra berwarna.

3. Gaussian

   Merupakan serangan putih yang diidealkan, yang disebabkan oleh fluktuasi acak dalam sinyal. Serangan Gaussian merupakan serangan putih yang terdistribusi secara normal. Jika suatu citra

dipresentasiakn sebagai *I*, dan serangan Gaussian oleh *N*, maka suatu citra yang diserang dapat di modelkan dengan menambahkan keduanya.

4. Speckle

Speckle dapat di modelkan oleh nilai-nilai acak yang dikalikan dengan nilai-nilai piksel. oleh karena itu, serangan ini juga disebut dengan serangan multiplikatif. Serangan speckle merupakan masalah utama pada aplikasi-aplikasi radar.

II.4 *Fast Fourier Transform* (*FFT*)

Pada tahun 1960, J. W. Cooley dan J. W. Tukey, berhasil merumuskan suatu teknik perhitungan algoritma Fourier Transform yang efisien. Teknik perhitungan algoritma ini dikenal dengan *Fast Fourier Transform* atau lebih populer dengan istilah *FFT* yang diperkenalkan oleh J.S.Bendat dan A.G.Piersol pada 1986 (Putra, 2010).

Fast Fourier Transform dalam bahasa indonesia adalah *Transformasi Fourier* Cepat adalah sumber dari suatu algoritma untuk menghitung *Discrete Fourier Transform* (transformasi fourier diskri tatau DFT) dengan cepat, efisien dan inversnya. Fast *Fourier Transform (FFT)* diterapkan dalam beragam bidang dari pengolahan sinyal digital dan memecahkan persamaan diferensial parsial menjadi algoritma-algoritma untuk penggandaan bilangan integer dalam jumlah banyak. Ada pun kelas dasar dari algoritma *FFT* yaitu decimation in time (DIT) dan decimation in frequency (DIF). Garis besar dari kata Fast diartikan karena formulasi *FFT* jauh lebih cepat dibandingkan dengan metode perhitungan algoritma *Fourier Transform* sebelumnya (Putra, 2010).

Metode *FFT* memerlukan sekitar 10000 operasi algoritma matematika untuk data dengan 1000 observasi, 100 kali lebih cepat dibandingan dengan metode sebelumnya. Penemuan *FFT* dan perkembangan personal komputer, teknik *FFT* dalam proses analisa data menjadi populer, dan merupakan salah satu

metode baku dalam analisa data. Satu bentuk transformasi yang umum digunakan untuk merubah sinyal dari domain waktu ke domain frekuensi adalah *Transformasi Fourier* (Putra, 2010).

Persamaan dari bentuk sinyal kontinu *x(t)* :

$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt \tag{2.5}$$

Fast Fourier Transform (FFT) adalah algoritma transformasi yang mengubah citra dari domain spasial menjadi domain frekuensi. Fast Fourier Transform adalah Salah satu metode pengolah gambar penting yang digunakan untuk menguraikan citra menjadi komponen sinus dan kosinus. Output dari transformasi mewakili citra dalam domain frekuensi, sedangkan gambar input adalah domain spasial yang setara. Pada citra domain frekuensi, setiap titik mewakili frekuensi tertentu yang terdapat dalam citra domain spasial. *FFT* digunakan dalam berbagai aplikasi, seperti penyaringan gambar, rekonstruksi citra, kompresi gambar, analisis citra dan watermarking gambar. *FFT* 2-D adalah perpanjangan langsung dari kasus 1-D dapat dituliskan dengan persamaan sbb

(Putra, 2010) :

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{2.6}$$

Dimana,

M x N = ukuran panjang dan lebar dari label *watermark*

j = Bilangan imajiner

u,v = titik koordinat dari citra

Invers transform dapat didefinisikan :

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{2.7}$$

Contoh Proses menggunakan *FFT* pada citra :

Misalkan matriks A citra berukuran 4x4 dengan nilai diambil dari citra.

$$A = \begin{bmatrix} 100 & 98 & 97 & 97 \\ 107 & 105 & 103 & 101 \\ 112 & 109 & 107 & 105 \\ 116 & 114 & 112 & 109 \end{bmatrix}$$

Dengan menggunakan persamaan (2.6) :

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

$$F(u,v) = \frac{1}{16} \sum_{x=0}^{3} \sum_{y=0}^{3} f(x,y) e^{-j2\pi\left(\frac{ux}{4}+\frac{vy}{4}\right)}, misalkan\ \alpha = \left(\frac{ux}{4}+\frac{vy}{4}\right)$$

$$F(u,v) = \frac{1}{16} \sum_{x=0}^{3} \sum_{y=0}^{3} f(x,y) e^{-j2\pi\alpha}$$

Menghitung nilai FFT entri-entri matriks A

1. Koordinat (0,0)

$$F(0,0) = \frac{1}{16} \sum_{x=0}^{3} \sum_{y=0}^{3} f(x,y) e^{-j2\pi\alpha}$$

$$= \frac{1}{16} \sum_{x=0}^{3} \sum_{y=0}^{3} f(x,y) e^{0}$$

$$= \frac{1}{16} (f(0,0) + f(0,1) + f(0,2) + f(0,3) + f(1,0) + f(1,1)$$
$$+ f(1,2) + f(1,3) + f(2,0) + f(2,1) + f(2,2) + f(2,3)$$
$$+ f(3,0) + f(3,1) + f(3,2) + f(3,3))$$

$$= \frac{1}{16} (100 + 98 + 97 + 97 + 107 + 105 + 103 + 101 + 112$$
$$+109 + 107 + 105 + 116 + 114 + 112 + 109) = 105{,}75$$

2. Koordinat (0,1)

$$F(0,1) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$

$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{y}{4}\right)}$$

$$= f(0,0) + f(0,1)e^{-j2\pi\left(\frac{1}{4}\right)} + f(0,2)e^{-j2\pi\left(\frac{2}{4}\right)} + f(0,3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(1,0) + f(1,1)e^{-j2\pi\left(\frac{1}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{2}{4}\right)} + f(1,3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(2,0) + f(2,1)e^{-j2\pi\left(\frac{1}{4}\right)} + f(2,2)e^{-j2\pi\left(\frac{2}{4}\right)} + f(2,3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(3,0) + f(3,1)e^{-j2\pi\left(\frac{1}{4}\right)} + f(3,2)e^{-j2\pi\left(\frac{2}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$= 100 + (-0{,}07 - 107j) + (111{,}99 + 0{,}14j) + (0{,}22 + 115{,}99)$$
$$+98 + (-0.07 - 105j) + (-108{,}99 + 0{,}14j) + (0{,}23$$
$$+113{,}99) + 97 + (-0{,}07 - 103j) + (106{,}99 + 0{,}14j)$$
$$+(0{,}21 + 111{,}99j) + 97 + (-0{,}07 - 101j) + (-104{,}99$$
$$+0{,}13j) + (0{,}21 + 108.99)$$
$$= \frac{-40{,}40 + 35{,}54j}{16}$$

3. Koordinat (0,2)

$$F(0,2) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$

$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{2y}{4}\right)}$$

$$= f(0,0) + f(0,1)e^{-j2\pi\left(\frac{2}{4}\right)} + f(0,2)e^{-j2\pi\left(\frac{4}{4}\right)} + f(0,3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(1,0) + f(1,1)e^{-j2\pi\left(\frac{2}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{4}{4}\right)} + f(1,3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(2,0) + f(2,1)e^{-j2\pi\left(\frac{2}{4}\right)} + f(2,2)e^{-j2\pi\left(\frac{4}{4}\right)} + f(2,3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(3,0) + f(3,1)e^{-j2\pi\left(\frac{2}{4}\right)} + f(3,2)e^{-j2\pi\left(\frac{4}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$= 100 + (-106{,}99 + 0{,}14j) + (111{,}99 - 1{,}28j) + (-115{,}99$$

$$+0{,}44j) + 98 + (-104{,}99 + 0{,}13j) + (108{,}99 - 0{,}28j)$$
$$+(-113{,}99 + 0{,}43j) + 97 + (-102{,}99 + 0{,}13j) + (106{,}99$$
$$-0{,}27j) + (-111{,}99 + 0{,}43j) + 97 + (-100{,}99 + 0{,}13j)$$
$$+(104{,}99 - 0{,}27j) + (-108{,}99 + 0{,}41j)$$
$$= \frac{-41{,}99 + 1{,}14j}{16}$$

4. Koordinat (0,3)

$$F(0{,}3) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{3y}{4}\right)}$$
$$= f(0{,}0) + f(0{,}1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(0{,}2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(0{,}3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(1{,}0) + f(1{,}1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(1{,}2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(1{,}3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(2{,}0) + f(2{,}1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(2{,}2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(2{,}3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(3{,}0) + f(3{,}1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(3{,}2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(3{,}3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$= 100 + (0{,}20 + 106{,}99j) + (-111{,}99 + 0{,}43j) + (-0{,}67$$
$$-115{,}99j) + 98 + (0{,}19 + 104{,}99j) + (108 - 0{,}28j)$$
$$+(-113{,}99 + 0{,}43j) + 97 + (-102{,}99 + 0{,}13j) + (106{,}99$$
$$+0{,}41j) + (-0{,}64 - 111{,}99j) + 97 + (0{,}19 + 100{,}99j)$$
$$+(-104 + 0{,}39j) + (-0{,}62 - 108{,}99j) = \frac{-42{,}77 - 33{,}35j}{16}$$

5. Koordinat (1,0)

$$F(1{,}0) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{x}{4}\right)}$$
$$= f(0{,}0) + f(0{,}1) + f(0{,}2) + f(0{,}3) + f(1{,}0)e^{-j2\pi\left(\frac{1}{4}\right)}$$

$$+f(1,1)e^{-j2\pi\left(\frac{1}{4}\right)}+f(1,2)e^{-j2\pi\left(\frac{1}{4}\right)}+f(1,3)e^{-j2\pi\left(\frac{1}{4}\right)}$$
$$+f(2,0)e^{-j2\pi\left(\frac{2}{4}\right)}+f(2,1)e^{-j2\pi\left(\frac{2}{4}\right)}+f(2,2)e^{-j2\pi\left(\frac{2}{4}\right)}$$
$$+f(2,3)e^{-j2\pi\left(\frac{2}{4}\right)}+f(3,0)e^{-j2\pi\left(\frac{3}{4}\right)}+f(3,1)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(3,2)e^{-j2\pi\left(\frac{3}{4}\right)}+f(3,3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$=100+107+112+116+(-0{,}06+98j)+(-0{,}06+105j)$$
$$+(-0{,}06+109j)+(-0{,}07-114j)+(-96{,}99+0{,}12j)$$
$$+(-102{,}99+0{,}13j)+(-106{,}99+0{,}14j)+(-111{,}99$$
$$+0{,}14j)+(0{,}18+96{,}99j)+(0{,}19+100{,}99j)+(0{,}19$$
$$+104{,}99j)+(0{,}21+108{,}99j)$$
$$=\frac{16{,}51-13{,}47j}{16}$$

6. Koordinat (1,1)

$$F(1,1)=\frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3}f(x,y)e^{-j2\pi\alpha}$$
$$=\frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3}f(x,y)e^{-j2\pi\left(\frac{x}{4}+\frac{y}{4}\right)}$$
$$=f(0,0)+f(0,1)e^{-j2\pi\left(\frac{1}{4}\right)}+f(0,2)e^{-j2\pi\left(\frac{2}{4}\right)}+f(0,3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(1,0)e^{-j2\pi\left(\frac{1}{4}\right)}+f(1,1)e^{-j2\pi\left(\frac{2}{4}\right)}+f(1,2)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(1,3)e^{-j2\pi\left(\frac{4}{4}\right)}+f(2,0)e^{-j2\pi\left(\frac{2}{4}\right)}+f(2,1)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(2,2)e^{-j2\pi\left(\frac{4}{4}\right)}+f(2,3)e^{-j2\pi\left(\frac{5}{4}\right)}+f(3,0)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(3,1)e^{-j2\pi\left(\frac{4}{4}\right)}+f(3,2)e^{-j2\pi\left(\frac{5}{4}\right)}+f(3,3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$=100+(-0.07-107j)+(-111{,}99+0{,}14j)+(0{,}22$$
$$+115{,}99j)+(-0{,}06-98j)+(-104{,}99+0{,}13j)+(0{,}21$$
$$+108{,}99j)+(113{,}99-0{,}29j)+(-96{,}99+0{,}12j)+(0{,}19$$
$$+102{,}99j)+(106{,}99-0{,}27j)+(-0{,}35-111{,}99j)+(0{,}18$$
$$+96{,}99j)+(100{,}99-0{,}26j)+(-0{,}33-104{,}99j)$$

$$+(-108{,}99 + 0{,}41j)$$
$$= \frac{-1{,}01 + 2{,}99j}{16}$$

7. Koordinat (1,2)

$$F(1{,}2) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{x}{4}+\frac{2y}{4}\right)}$$
$$= f(0{,}0) + f(0{,}1)e^{-j2\pi\left(\frac{2}{4}\right)} + f(0{,}2)e^{-j2\pi\left(\frac{4}{4}\right)} + f(0{,}3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(1{,}0)e^{-j2\pi\left(\frac{1}{4}\right)} + f(1{,}1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(1{,}2)e^{-j2\pi\left(\frac{5}{4}\right)}$$
$$+f(1{,}3)e^{-j2\pi\left(\frac{7}{4}\right)} + f(2{,}0)e^{-j2\pi\left(\frac{2}{4}\right)} + f(2{,}1)e^{-j2\pi\left(\frac{4}{4}\right)}$$
$$+f(2{,}2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(2{,}3)e^{-j2\pi\left(\frac{8}{4}\right)} + f(3{,}0)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(3{,}1)e^{-j2\pi\left(\frac{5}{4}\right)} + f(3{,}2)e^{-j2\pi\left(\frac{7}{4}\right)} + f(3{,}3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$= 100 + (-106{,}99 + 0{,}14j) + (111{,}99 - 0{,}28j) + (-115{,}99$$
$$+0{,}44j) + (-0{,}06 - 98j) + (0{,}19 + 104{,}99j) + (-0{,}35$$
$$-108{,}99j) + (0{,}50 + 113{,}99j) + (-96{,}99 + 0{,}12j)$$
$$+(102{,}99 - 0{,}26j) + (-106{,}99 + 0{,}41j) + (111{,}99 - 0{,}57j)$$
$$+(0{,}18 + 96{,}99j) + (-0{,}32 - 100{,}99j) + (0{,}47 + 104{,}99j)$$
$$+(-0{,}62 - 108{,}99j)$$
$$= \frac{0{,}01 + 3{,}99j}{16}$$

8. Koordinat (1,3)

$$F(1{,}3) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{x}{4}+\frac{3y}{4}\right)}$$

$$= f(0,0) + f(0,1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(0,2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(0,3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(1,0)e^{-j2\pi\left(\frac{1}{4}\right)} + f(1,1)e^{-j2\pi\left(\frac{4}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{7}{4}\right)}$$
$$+f(1,3)e^{-j2\pi\left(\frac{10}{4}\right)} + f(2,0)e^{-j2\pi\left(\frac{2}{4}\right)} + f(2,1)e^{-j2\pi\left(\frac{5}{4}\right)}$$
$$+f(2,2)e^{-j2\pi\left(\frac{8}{4}\right)} + f(2,3)e^{-j2\pi\left(\frac{11}{4}\right)} + f(3,0)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(3,1)e^{-j2\pi\left(\frac{6}{4}\right)} + f(3,2)e^{-j2\pi\left(\frac{9}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{12}{4}\right)}$$
$$= 100 + (0{,}20 + 106{,}99j) + (-111{,}99 + 0{,}43j) + (-0{,}66$$
$$-115, j) + (-0{,}06 - 98j) + (104{,}99 - 0{,}27j) + (0{,}48$$
$$-108{,}99j) + (-113{,}99 + 0{,}72j) + (-96{,}99 + 0{,}12j)$$
$$+(-0{,}33 - 102{,}99j) + (106{,}99 - 0{,}54j) + (0{,}78 + 111{,}99j)$$
$$+(0{,}18 + 96{,}99j) + (-100{,}99 + 0{,}38j) + (-0{,}59 - 104{,}99j)$$
$$+(108{,}99 - 0{,}83j)$$
$$= \frac{-2{,}99 + 3{,}02j}{16}$$

9. Koordinat (2,0)

$$F(2,0) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{2x}{4}\right)}$$
$$= f(0,0) + f(0,1) + f(0,2) + f(0,3) + f(1,0)e^{-j2\pi\left(\frac{2}{4}\right)}$$
$$+f(1,1)e^{-j2\pi\left(\frac{2}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{2}{4}\right)} + f(1,3)e^{-j2\pi\left(\frac{2}{4}\right)}$$
$$+f(2,0)e^{-j2\pi\left(\frac{4}{4}\right)} + f(2,1)e^{-j2\pi\left(\frac{4}{4}\right)} + f(2,2)e^{-j2\pi\left(\frac{4}{4}\right)}$$
$$+f(2,3)e^{-j2\pi\left(\frac{4}{4}\right)} + f(3,0)e^{-j2\pi\left(\frac{6}{4}\right)} + f(3,1)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(3,2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$= 100 + 107 + 112 + 116 + (-97{,}99 + 0{,}12j) + (-104{,}99$$
$$+0{,}13j) + (-108{,}99 + 0{,}13j) + (-113{,}99 + 0{,}14j) + (96{,}99$$
$$-0{,}25j) + (102{,}99 - 0{,}26j) + (106{,}99 - 0{,}27j) + (111{,}99$$
$$-0{,}28j) + (-96{,}99 + 0{,}37j) + (-100{,}99 + 0{,}38j)$$

$$+(-104{,}99 + 0{,}39j) + (-108{,}99 + 0{,}41j)$$
$$= \frac{16 + 1{,}04j}{16}$$

10. Koordinat (2,1)

$$F(2,1) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{2x}{4}+\frac{y}{4}\right)}$$
$$= f(0,0) + f(0,1)e^{-j2\pi\left(\frac{1}{4}\right)} + f(0,2)e^{-j2\pi\left(\frac{2}{4}\right)} + f(0,3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(1,0)e^{-j2\pi\left(\frac{2}{4}\right)} + f(1,1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{4}{4}\right)}$$
$$+f(1,3)e^{-j2\pi\left(\frac{5}{4}\right)} + f(2,0)e^{-j2\pi\left(\frac{4}{4}\right)} + f(2,1)e^{-j2\pi\left(\frac{5}{4}\right)}$$
$$+f(2,2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(2,3)e^{-j2\pi\left(\frac{7}{4}\right)} + f(3,0)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(3,1)e^{-j2\pi\left(\frac{7}{4}\right)} + f(3,2)e^{-j2\pi\left(\frac{8}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$= 100 + (-0{,}07 - 107j) + (-111{,}99 + 0{,}14j) + (0{,}22$$
$$+115{,}99j) + (-97{,}99 + 0{,}12j) + (0{,}19 + 104{,}99j)$$
$$+(108{,}99 - 0{,}28j) + (-0{,}36 - 113{,}99j) + (96{,}99 - 0{,}25j)$$
$$+(-0{,}33 - 102{,}99j) + (-106{,}99 + 0{,}41j) + (0{,}49$$
$$+111{,}99j) + (-96{,}99 + 0{,}37j) + (-104{,}99 - 0{,}53j)$$
$$+(104{,}99 - 0{,}53j) + (-0{,}62 - 108{,}99j)$$
$$= \frac{-3{,}01 + 0{,}99j}{16}$$

11. Koordinat (2,2)

$$F(2,2) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{2x}{4}+\frac{2y}{4}\right)}$$

$$= f(0,0) + f(0,1)e^{-j2\pi\left(\frac{2}{4}\right)} + f(0,2)e^{-j2\pi\left(\frac{4}{4}\right)} + f(0,3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(1,0)e^{-j2\pi\left(\frac{2}{4}\right)} + f(1,1)e^{-j2\pi\left(\frac{4}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(1,3)e^{-j2\pi\left(\frac{8}{4}\right)} + f(2,0)e^{-j2\pi\left(\frac{4}{4}\right)} + f(2,1)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(2,2)e^{-j2\pi\left(\frac{8}{4}\right)} + f(2,3)e^{-j2\pi\left(\frac{10}{4}\right)} + f(3,0)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(3,1)e^{-j2\pi\left(\frac{8}{4}\right)} + f(3,2)e^{-j2\pi\left(\frac{10}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{12}{4}\right)}$$
$$= 100 + (-106{,}99 + 0{,}13j) + (111{,}99 - 0{,}28j) + (-115{,}99$$
$$+0{,}44j) + (-97{,}99 + 0{,}12j) + (104{,}99 - 0{,}27j)$$
$$+(-108{,}99 + 0{,}41j) + (113{,}99 - 0{,}58j) + (96{,}99 - 0{,}25j)$$
$$+(-102{,}99 + 0{,}39j) + (106{,}99 - 0{,}54j) + (-111{,}99$$
$$+0{,}71j) + (-96{,}99 + 0{,}37j) + (100{,}99 - 0{,}51j) + (-104{,}99$$
$$+0{,}66j) + (108{,}99 - 0{,}83j)$$
$$= \frac{-2 - 0{,}01j}{16}$$

12. Koordinat (2,3)

$$F(2{,}3) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{2x}{4}+\frac{3y}{4}\right)}$$
$$= f(0,0) + f(0,1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(0,2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(0,3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(1,0)e^{-j2\pi\left(\frac{2}{4}\right)} + f(1,1)e^{-j2\pi\left(\frac{5}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{8}{4}\right)}$$
$$+f(1,3)e^{-j2\pi\left(\frac{11}{4}\right)} + f(2,0)e^{-j2\pi\left(\frac{4}{4}\right)} + f(2,1)e^{-j2\pi\left(\frac{7}{4}\right)}$$
$$+f(2,2)e^{-j2\pi\left(\frac{10}{4}\right)} + f(2,3)e^{-j2\pi\left(\frac{13}{4}\right)} + f(3,0)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(3,1)e^{-j2\pi\left(\frac{9}{4}\right)} + f(3,2)e^{-j2\pi\left(\frac{12}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{15}{4}\right)}$$
$$= 100 + (0{,}20 + +106{,}99j) + (-111{,}99 + 0{,}45j) + (-0{,}66$$
$$-115{,}99j) + (-97{,}99 + 0{,}12j) + (-0{,}33 - 104{,}99j)$$
$$+(108{,}99 + 0{,}55j) + (0{,}79 + 113{,}99j) + (96{,}99 - 0{,}25j)$$

$$+(0{,}46+102{,}99j)+(-106{,}99+0{,}68j)+(-0{,}92$$
$$-111{,}99j)+(-96{,}99+0{,}37j)+(-0{,}57-100{,}99j)$$
$$+(104{,}99-0{,}79j)+(1{,}03+108{,}99j)$$
$$=\frac{-3-1j}{16}$$

13. Koordinat (3,0)

$$F(3{,}0)=\frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3}f(x,y)e^{-j2\pi\alpha}$$
$$=\frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3}f(x,y)e^{-j2\pi\left(\frac{3x}{4}\right)}$$
$$=f(0{,}0)+f(0{,}1)+f(0{,}2)+f(0{,}3)+f(1{,}0)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(1{,}1)e^{-j2\pi\left(\frac{3}{4}\right)}+f(1{,}2)e^{-j2\pi\left(\frac{3}{4}\right)}+f(1{,}3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(2{,}0)e^{-j2\pi\left(\frac{6}{4}\right)}+f(2{,}1)e^{-j2\pi\left(\frac{6}{4}\right)}+f(2{,}2)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(2{,}3)e^{-j2\pi\left(\frac{6}{4}\right)}+f(3{,}0)e^{-j2\pi\left(\frac{9}{4}\right)}+f(3{,}1)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(3{,}2)e^{-j2\pi\left(\frac{9}{4}\right)}+f(3{,}3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$=100+107+112+116+(0{,}19+97{,}99j)+(0{,}19$$
$$+104{,}99j)+(0{,}21+108{,}99j)+(0{,}22+113{,}99j)$$
$$+(-96{,}99+0{,}37j)+(-102{,}99+0{,}39j)+(-106{,}99$$
$$+0{,}41j)+(-111{,}99+0{,}43j)+(-0{,}55-96{,}99j)+(-0{,}57$$
$$-100{,}99j)+(-0{,}59-104{,}99j)+(0{,}62-108{,}99j)$$
$$=\frac{14{,}47+15{,}59j}{16}$$

14. Koordinat (3,1)

$$F(3{,}1)=\frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3}f(x,y)e^{-j2\pi\alpha}$$
$$=\frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3}f(x,y)e^{-j2\pi\left(\frac{3x}{4}+\frac{y}{4}\right)}$$

$$= f(0,0) + f(0,1)e^{-j2\pi\left(\frac{1}{4}\right)} + f(0,2)e^{-j2\pi\left(\frac{2}{4}\right)} + f(0,3)e^{-j2\pi\left(\frac{3}{4}\right)}$$
$$+f(1,0)e^{-j2\pi\left(\frac{3}{4}\right)} + f(1,1)e^{-j2\pi\left(\frac{4}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{5}{4}\right)}$$
$$+f(1,3)e^{-j2\pi\left(\frac{6}{4}\right)} + f(2,0)e^{-j2\pi\left(\frac{6}{4}\right)} + f(2,1)e^{-j2\pi\left(\frac{7}{4}\right)}$$
$$+f(2,2)e^{-j2\pi\left(\frac{8}{4}\right)} + f(2,3)e^{-j2\pi\left(\frac{9}{4}\right)} + f(3,0)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(3,1)e^{-j2\pi\left(\frac{10}{4}\right)} + f(3,2)e^{-j2\pi\left(\frac{11}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{12}{4}\right)}$$
$$= 100 + (-0{,}07 - 107j) + (-111{,}99 + 0{,}14j) + (0{,}22$$
$$+115{,}99j) + (0{,}19 + 97{,}99j) + (104{,}99 - 0{,}27j) + (-0{,}35$$
$$+108{,}99j) + (-113{,}99 + 0{,}43j) + (-96{,}99 + 0{,}37j) + (0{,}46$$
$$+102{,}99j) + (106{,}99 - 0{,}54j) + (-0{,}64 - 111{,}99j) + (0{,}55$$
$$-96{,}99j) + (-100{,}99 + 0{,}64j) + (0{,}73 + 104{,}99j)$$
$$+(108{,}99 - 0{,}82j)$$
$$= \frac{-3{,}00 - 3{,}05j}{16}$$

15. Koordinat (3,2)

$$F(3,2) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{3x}{4}+\frac{2y}{4}\right)}$$
$$= f(0,0) + f(0,1)e^{-j2\pi\left(\frac{2}{4}\right)} + f(0,2)e^{-j2\pi\left(\frac{4}{4}\right)} + f(0,3)e^{-j2\pi\left(\frac{6}{4}\right)}$$
$$+f(1,0)e^{-j2\pi\left(\frac{3}{4}\right)} + f(1,1)e^{-j2\pi\left(\frac{5}{4}\right)} + f(1,2)e^{-j2\pi\left(\frac{7}{4}\right)}$$
$$+f(1,3)e^{-j2\pi\left(\frac{9}{4}\right)} + f(2,0)e^{-j2\pi\left(\frac{6}{4}\right)} + f(2,1)e^{-j2\pi\left(\frac{8}{4}\right)}$$
$$+f(2,2)e^{-j2\pi\left(\frac{10}{4}\right)} + f(2,3)e^{-j2\pi\left(\frac{12}{4}\right)} + f(3,0)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(3,1)e^{-j2\pi\left(\frac{11}{4}\right)} + f(3,2)e^{-j2\pi\left(\frac{13}{4}\right)} + f(3,3)e^{-j2\pi\left(\frac{15}{4}\right)}$$
$$= 100 + (-106{,}99 + 0{,}14j) + (111{,}99 - 0{,}28j) + (-115{,}99$$
$$+0{,}44j) + (0{,}19 + 97{,}99j) + (-0{,}33 - 104{,}99j) + (0{,}48$$

$$+108{,}99j) + (-0{,}65 - 113{,}99j) + (96{,}99 - 0{,}37j)$$
$$+(102{,}99 - 0{,}52j) + (-106{,}99 + 0{,}68j) + (111{,}99 + 0{,}85j)$$
$$+(-0{,}55 - 96{,}99j) + (0{,}70 + 100{,}99j) + (-0{,}86 - 104{,}99j)$$
$$+(1{,}03 + 108{,}99j)$$
$$= \frac{0{,}01 - 4{,}04j}{16}$$

16. Koordinat (3,3)

$$F(3{,}3) = \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\alpha}$$
$$= \frac{1}{16}\sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)e^{-j2\pi\left(\frac{3x}{4}+\frac{3y}{4}\right)}$$
$$= f(0{,}0) + f(0{,}1)e^{-j2\pi\left(\frac{3}{4}\right)} + f(0{,}2)e^{-j2\pi\left(\frac{6}{4}\right)} + f(0{,}3)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(1{,}0)e^{-j2\pi\left(\frac{3}{4}\right)} + f(1{,}1)e^{-j2\pi\left(\frac{6}{4}\right)} + f(1{,}2)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(1{,}3)e^{-j2\pi\left(\frac{12}{4}\right)} + f(2{,}0)e^{-j2\pi\left(\frac{6}{4}\right)} + f(2{,}1)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(2{,}2)e^{-j2\pi\left(\frac{12}{4}\right)} + f(2{,}3)e^{-j2\pi\left(\frac{15}{4}\right)} + f(3{,}0)e^{-j2\pi\left(\frac{9}{4}\right)}$$
$$+f(3{,}1)e^{-j2\pi\left(\frac{12}{4}\right)} + f(3{,}2)e^{-j2\pi\left(\frac{15}{4}\right)} + f(3{,}3)e^{-j2\pi\left(\frac{18}{4}\right)}$$
$$= 100 + (0{,}20 + +106{,}99j) + (-111{,}99 + 0{,}43j) + (-0{,}66$$
$$-115{,}99j) + (0{,}19 + 97{,}99j) + (-104{,}99 + 0{,}39j)$$
$$+(-0{,}62 - 108{,}99j) + (113{,}99 - 0{,}86j) + (-96{,}99 + 0{,}37j)$$
$$+(-0{,}59 - 102{,}99j) + (106{,}99 - 0{,}81j) + (1{,}06 + 111{,}99j)$$
$$+(-0{,}55 - 96{,}99j) + (100{,}99 - 0{,}77j) + (0{,}99 + 104{,}99j)$$
$$+(-108{,}99 + 1{,}24j)$$
$$= \frac{-0{,}973 - 3{,}01j}{16}$$

Setelah melakukan perhitungan manual, maka dibentuk kembali matriks hasil perhitungan sebagai berikut:

$$B = \begin{bmatrix} 105{,}75 & \frac{16{,}51 - 13{,}47j}{16} & \frac{16 + 1{,}04j}{16} & \frac{14{,}47 + 15{,}59j}{16} \\ \frac{-40{,}40 + 35{,}54j}{16} & \frac{-1{,}01 + 2{,}99j}{16} & \frac{-3{,}01 + 0{,}99j}{16} & \frac{-3{,}00 - 3{,}05j}{16} \\ \frac{-41{,}99 + 1{,}14j}{16} & \frac{0{,}01 + 3{,}99j}{16} & \frac{-2 - 0{,}01j}{16} & \frac{0{,}01 - 4{,}04j}{16} \\ \frac{-42{,}77 - 33{,}35j}{16} & \frac{-2{,}99 + 3{,}02j}{16} & \frac{-3 - 1j}{16} & \frac{-0{,}973 - 3{,}01j}{16} \end{bmatrix}$$

II.5 *Wavelet Transform*

Wavelet transform berkorespondensi dengan dekomposisi dari fungsi kuadrat integral $s\,(x) \in L^2(R)$ dalam fungsi skala dan pergeseran dari $\psi_{k,l}(t)$.

$$\psi_{k,\,l}\,(t) = k^{-\frac{1}{2}}\psi\,(\tfrac{t-1}{k}) \tag{2.8}$$

Fungsi $\psi\,(x)$ disebut fungsi *wavelet* yang menunjukkan sifat bandpass. Koefisien *wavelet* $d_{k,l}$ adalah

$$d_{k,\,l} = \frac{l}{\sqrt{k}}\int_{-\infty}^{\infty} s\,(x)\,\psi^*\left(\frac{x-l}{k}\right)dx \tag{2.9}$$

dengan $k \in R^+, l \in R$ dan $*$ menunjukkan fungsi konjugat kompleks.

Transformasi *wavelet* diskrit (DWT) 1-D sinyal $s(t)$ dalam proses dekomposisi dari fungsi skala lowpass $\phi(t)$ berubah serta dilatasi versi fungsi prototype bandpass *wavelet* $\psi(t)$. Fungsi khusus dari $\phi(t)$ dan $\psi(t)$ sebagai berikut:

$$\psi_{j,\,k}\,(t) = 2^{-\frac{j}{2}}\,\psi(2^{-j}\,t - k) \tag{2.10}$$

$$\phi_{j,\,k}\,(t) = 2^{-\frac{j}{2}}\,\phi(2^{-j}\,t - k) \tag{2.11}$$

untuk $j,k \in$ z, yang membentuk basis ortonormal, maka diperoleh :

$$z(t) = \sum_{j=j0}^{\infty}\sum_k u_{j,k}\phi_{j,k}(t) + \sum_{j=j0}^{\infty}\sum_k \omega_{j,k}\psi_{j,k}\,(t) \tag{2.12}$$

dengan

$$u_{j,k} = \int s(t)\, \phi^*_{j,k}(t)\, dt \text{ dan } \omega_{j,k} = \int s(t)\, \psi^*_{j,k}(t)\, dt \qquad (2.13)$$

tinjau bentuk transformasi Fourier berikut :

$$F(u) = \int_{-\infty}^{\infty} f(x)\, e^{-j2\pi xu}\, dx \text{ dan } \quad f(x) = \int_{-\infty}^{\infty} F(u)\, e^{-j2\pi xu}\, du \qquad (2.14)$$

Jadi $f(x)$ yang diwakili disini sebagai kombinasi linear dari fungsi dasar basis adalah $e^{j\omega x}$. Transformasi *wavelet* di sisi lain yang mewakili *f*(x) atau *f*(t) sebagai kombinasi linear dapat di lihat dari persamaan berikut :

$$\psi_{k,l}(t) = 2^{-\frac{k}{2}}\, \psi(2^{-k}\, t - l) \qquad (2.15)$$

Dimana $\psi_{k,l}(t)$ disebut dengan *mother wavelet*. Parameter k dan l adalah bilangan bulat yang menghasilkan basis yang berfungsi sebagai variasi melebar dan bergeser pada *mother wavelet*. Parameter k adalah frekuensi dan l adalah waktu. Oleh karena itu dengan k dan l, akan memiliki frekuensi yang berbeda dan waktu atau ruang yang berbeda (Tuakia, 2013).

II.5.1 *Discreate Wavelet Transform (DWT)*

Discrete Wavelet Transform (DWT) secara umum merupakan dekomposisi citra pada frekuensi *subband* citra tersebut. Komponen *subband* transformasi *wavelet* dihasilkan dengan cara penurunan level dekomposisi. Dalam transformasi *wavelet* diskrit, penggambaran sebuah skala waktu sinyal digital didapatkan dengan menggunakan teknik filterisasi digital. Secara garis besar proses dalam teknik ini adalah dengan melewatkan sinyal yang akan dianalisis pada *filter* dengan frekuensi dan skala yg berbeda (Tuakia, 2013).

DWT dari sinyal x dapat melalui serangkaian filter. Pertama sampel melalui lowpass filter dengan respon impuls g yang mengakibatkan 2 konvolusi :

$$y(n) = (x * g)(n) = \sum_{k=-\infty}^{\infty} x(k) g(n-k) \qquad (2.16)$$

Implementasi transformasi *wavelet* diskrit dapat dilakukan dengan cara melewatkan sinyal ke dalam dua filterisasi DWT yaitu *highpass filter* (HPF) dan *lowpass filter* (LPF), dimana HPF digunakan untuk menganalisis frekuensi tinggi

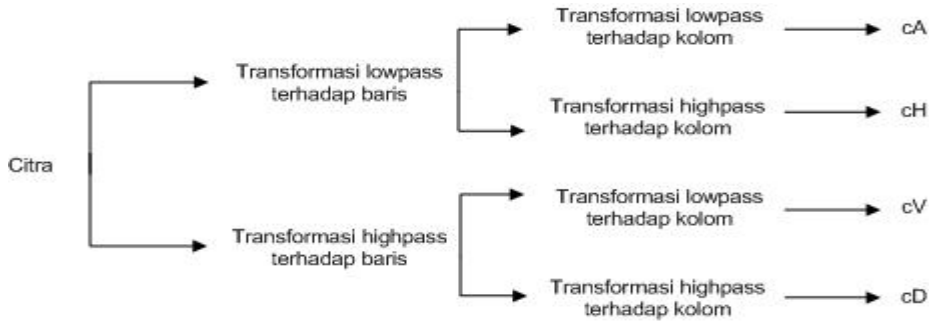
dan LPF digunakan untuk menganalisis frekuensi rendah. Analisis terhadap frekuensi dilakukan dengan cara menggunakan resolusi yang di hasilkan setelah sinyal melewati filterisasi. Analisis frekuensi yang berbeda inilah yang disebut dengan multi-resolution analysis (Tuakia, 2013).

Proses dekomposisi ini dapat melalui satu atau lebih tingkatan. Dekomposisi satu tingkat ditulis pada persamaan (2.17) dan (2.18) :

$$y_{tinggi}(k) = \sum_{k=-\infty}^{\infty} x(n)h(2k-n) \qquad (2.17)$$

$$y_{rendah}(k) = \sum_{k=-\infty}^{\infty} x(n)g(2k-n) \qquad (2.18)$$

Nilai $y_{tinggi}(k)$ dan $y_{rendah}(k)$ merupakan hasil dari *highpass filter* dan *lowpass filter*, $x(n)$ merupakan sinyal asal, $h(n)$ adalah *highpass filter*, dan $g(n)$ adalah *lowpass filter*. Untuk dekomposisi lebih dari satu tingkat, dapat digunakan pada masing masing tingkatan. Adapaun dekomposisi tingkatan yang dimaksud dapat di lihat pada gambar II.3.



Gambar II.3 Dekomposisi *Wavelet* Diskrit (Tiwari, 2015).

Dengan menggunakan koefisien DWT maka dapat dilakukan proses *Inverse Discrete Wavelet Transform* (IDWT) untuk merekonstruksi menjadi sinyal asal melalui persamaan berikut :

$$x(n) = \sum_{k=-\infty}^{\infty} y_{tinggi}(k)h(2k-n) + y_{rendah}(k)g(2k-n) \qquad (2.19)$$

Transformasi *wavelet* diskrit tiga dimensi digambarkan pada gambar II.4 berikut:

<table>
<tr><td>LL3</td><td>HL3</td><td rowspan="2">HL2</td><td rowspan="3">HL1</td></tr>
<tr><td>LH3</td><td>HH3</td></tr>
<tr><td colspan="2">LH2</td><td>HH2</td></tr>
<tr><td colspan="3">LH1</td><td>HH1</td></tr>
</table>

Gambar II.4 Transformasi *Wavelet* Diskrit pada Citra 2D. (Tiwari, 2015)

Contoh Proses menggunakan metode *DWT* pada citra :

Misalkan matriks A citra berukuran 4x4 dengan nilai diambil dari citra.

$$A = \begin{bmatrix} 130 & 113 & 86 & 29 \\ 148 & 133 & 67 & 32 \\ 157 & 131 & 68 & 39 \\ 167 & 138 & 70 & 29 \end{bmatrix}$$

Langkah – langkah dekomposisi wavelet haar terhadap potongan citra di atas adalah

1. Filter dekomposisi LH, yaitu

$$LH = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

2. Untuk setiap kolom, kalikan kolom tersebut dengan matrik dekomposisi
   Untuk kolom pertama :

$$LH1 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \text{ x } \begin{bmatrix} 130 \\ 148 \\ 157 \\ 167 \end{bmatrix} = \begin{bmatrix} 139 \\ -9 \\ 162 \\ -5 \end{bmatrix}$$

Untuk kolom kedua:

$$LH2 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \text{ x } \begin{vmatrix} 113 \\ 133 \\ 131 \\ 138 \end{vmatrix} = \begin{vmatrix} 123 \\ -10 \\ 134 \\ -3{,}5 \end{vmatrix}$$

Untuk kolom ketiga:

$$LH3 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \text{ x } \begin{vmatrix} 86 \\ 67 \\ 68 \\ 70 \end{vmatrix} = \begin{vmatrix} 76{,}5 \\ 9{,}5 \\ 69 \\ -1 \end{vmatrix}$$

Untuk kolom keempat:

$$LH4 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \text{ x } \begin{vmatrix} 29 \\ 32 \\ 39 \\ 29 \end{vmatrix} = \begin{vmatrix} 30{,}5 \\ -1{,}5 \\ 34 \\ 5 \end{vmatrix}$$

Maka hasil dekomposisi perbaris ini menghasilkan matriks LH5 sebagai berikut:

$$LH5 = \begin{bmatrix} 139 & 123 & 76{,}5 & 30{,}5 \\ -9 & -10 & 9{,}5 & -1{,}5 \\ 162 & 134 & 69 & 34 \\ -5 & -3{,}5 & -1 & 5 \end{bmatrix}$$

$$LH6 = \begin{bmatrix} 139 & 123 & 76{,}5 & 30{,}5 \\ 162 & 134 & 69 & 34 \\ -9 & -10 & 9{,}5 & -1{,}5 \\ -5 & -3{,}5 & -1 & 5 \end{bmatrix}$$

Kemudian mengalikan kembali matriks dekomposisi LH dengan matriks LH6 maka hasilnya

$$LH7 = \begin{bmatrix} 131 & 148 & -9,5 & -4,25 \\ 8 & 14 & 0,5 & -0,75 \\ 53,5 & 51 & 4 & 2 \\ 23 & 17,5 & 5,5 & -3 \end{bmatrix}$$

$$LH8 = \begin{bmatrix} 131 & -9,5 & 148 & -4,25 \\ 8 & 0,5 & 14 & -0,75 \\ 53,5 & 4 & 51 & 2 \\ 23 & 5,5 & 17,5 & -3 \end{bmatrix}$$

Maka dekomposisi wavelet Haar 1 level terhadap citra sebagai berikut:

$$LH9 = \begin{bmatrix} 131 & -9,5 & 148 & -4,25 \\ 8 & 0,5 & 14 & -0,75 \\ 53,5 & 4 & 51 & 2 \\ 23 & 5,5 & 17,5 & -3 \end{bmatrix}$$

Keterangan:
- warna biru adalah bagian aproksimasi
- warna merah adalah bagian detail horizontal
- warna orange adalah bagian detail vertikal
- warna hijau adalah bagian detail diagonal

BAB III

METODOLOGI PENELITIAN

III.1 Algoritma Penyisipan menggunakan *FFT*

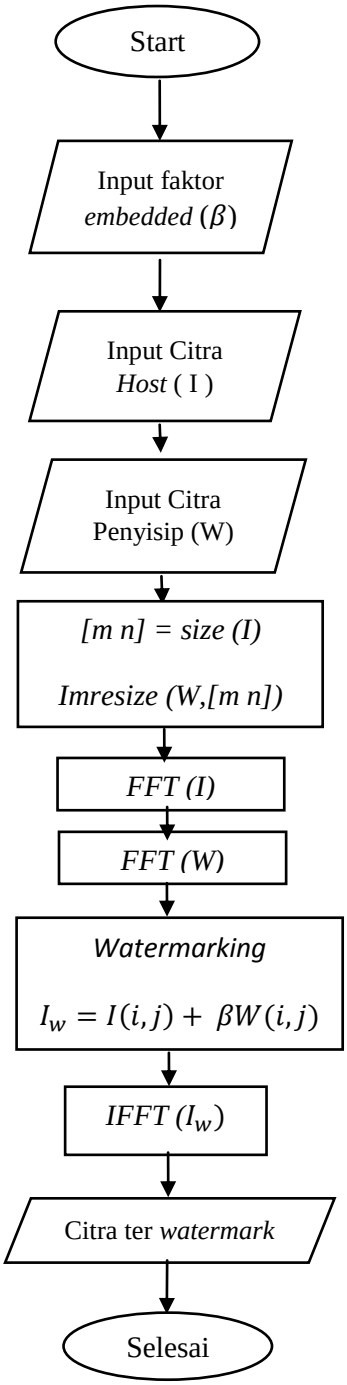
Prosedur penyisipan *watermark* pada citra digital (*file host*) dengan metode *FFT* dijelaskan sebagai berikut :

1. Menentukan faktor *embedded* (β).
2. Menentukan citra digital yang akan dijadikan citra *host (I)* dan citra penyisip (W).
3. Mengubah ukuran citra penyisip (W) sesuai ukuran citra *host (I)*.
4. Menerapkan *FFT* pada citra *host (I)* dan citra penyisip (W)
5. *Watermarking* dengan menggunakan persamaan :

$$I_w(i,j) = I(i,j) + \beta W(i,j)$$

6. Lakukan *Inverse FFT* pada *watermarked image* untuk menghasilkan citra ter *watermark.*

Proses penyisipan *watermark* pada citra digital dengan metode *FFT* dapat dilihat pada *flowchart* di bawah ini yang ditunjukkan oleh gambar III.1.



Gambar III.1 Alur kerja proses penyisipan citra *watermark* pada citra *host* dengan mengunakan metode *FFT*

III. 2 Algoritma Ekstraksi Menggunakan *FFT*

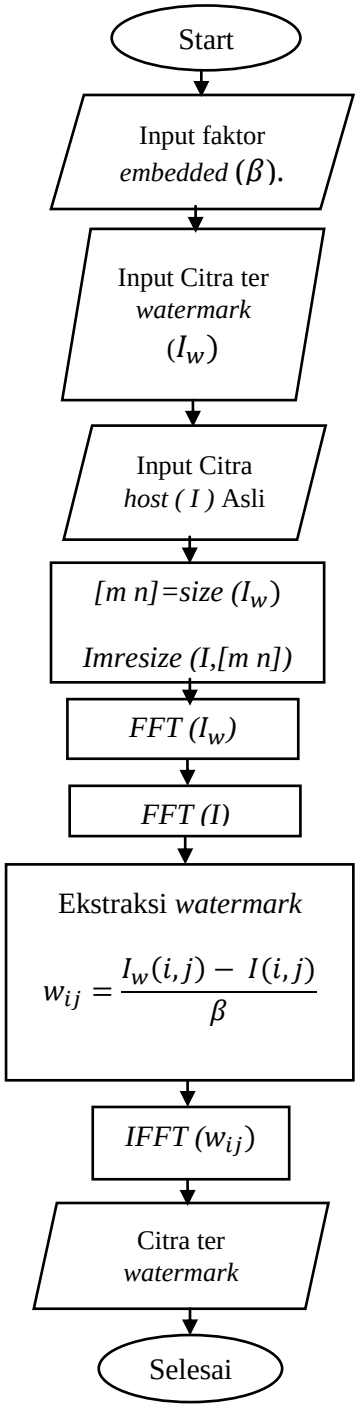
Proses Ekstraksi *watermark* pada citra digital (*file host*) dengan metode *FFT* dijelaskan sebagai berikut :

1. Menentukan faktor *embedded* (β).
2. Input citra ter *watermark* (I_w) dan citra *host (I).*
3. Mengubah ukuran citra *host* (I) sesuai dengan ukuran citra ter - *watermark* (I_w)
4. Menerapkan *FFT* pada citra ter *watermark* (I_w) dan citra *host (I).*
5. Ekstraksi dengan menggunakan persamaan:

   $$w_{ij} = \frac{I_w(i,j) - I(i,j)}{\beta}$$

6. Menerapkan *IFFT* pada Citra *watermark* untuk menghasilkan citra ter *watermark.*

Proses ekstraksi *watermark* pada citra digital (*file host*) dengan metode *FFT* dapat dilihat pada *flowchart* di bawah ini yang ditunjukkan oleh gambar III.2.



Gambar III.2 Alur kerja proses *ekstraksi* citra *watermark* pada citra ter *watermark* dengan mengunakan metode *FFT*

III. 3 Algoritma Penyisipan Menggunakan *DWT-FFT*

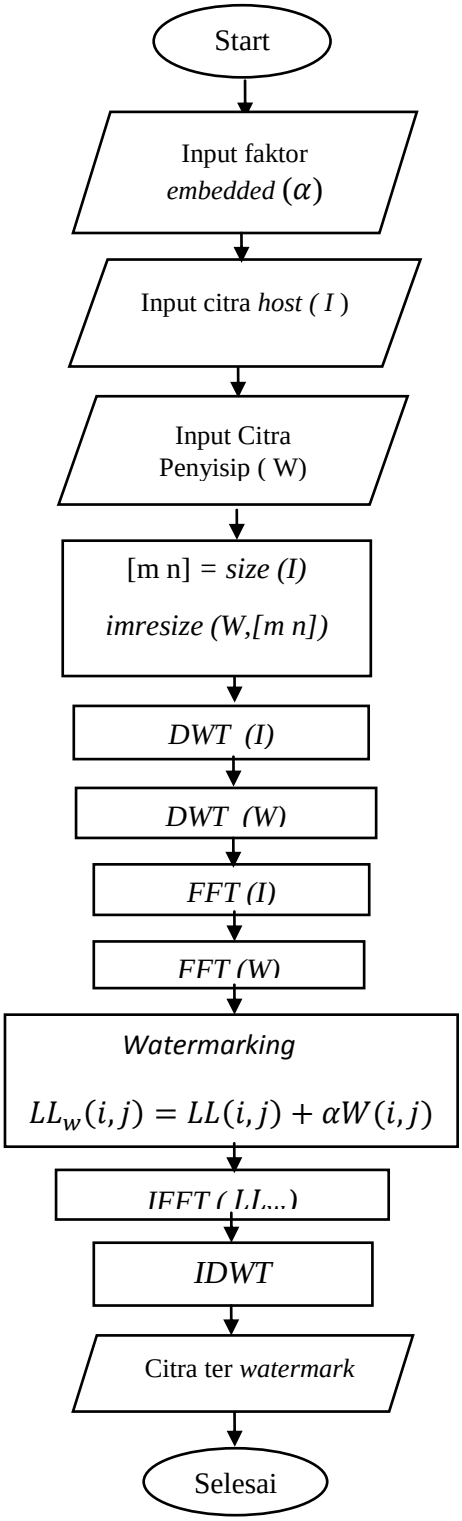
Proses penyisipan *watermark* pada citra digital (*file host*) dengan metode *DWT-FFT* dijelaskan sebagai berikut :

1. Menentukan faktor *embedded* (α).
2. Menentukan citra digital yang akan dijadikan *file host (I)* dan citra penyisip (W).
3. Mengubah ukuran citra penyisip (W) sesuai ukuran citra *host (I)*.
4. Dekomposisi (bagi) file gambar atau citra digital menjadi 4 (empat) bagian *file host (I)* dan citra penyisip (W) dengan menggunakan *Discrete Wavelet Transform (DWT)* .
5. Pilih salah satu *sub level* sebagai tempat penyisipan *watermark* pada *file host (I)* dan citra penyisip (W).
6. Terapkan *FFT* pada pada *sub level* yang dipilih.
7. *Watermarking* dengan menggunakan persamaan :

$$LL_w(i,j) = LL(i,j) + \alpha W(i,j)$$

8. Lakukan *Inverse FFT* pada *watermarking image.*
9. Lakukan proses *Invers DWT* untuk merokonstruksi *watermarked image*.

Proses penyisipan *watermark* pada citra digital (*file host*) dengan metode *DWT-FFT* dapat dilihat pada *flowchart* di bawah ini yang ditunjukan oleh gambar III.3.



Gambar III.3 Alur kerja proses penyisipan citra *watermark* pada citra *host* dengan mengunakan metode *DWT-FFT*

III. 4 Algoritma Ekstraksi Menggunakan *DWT-FFT*

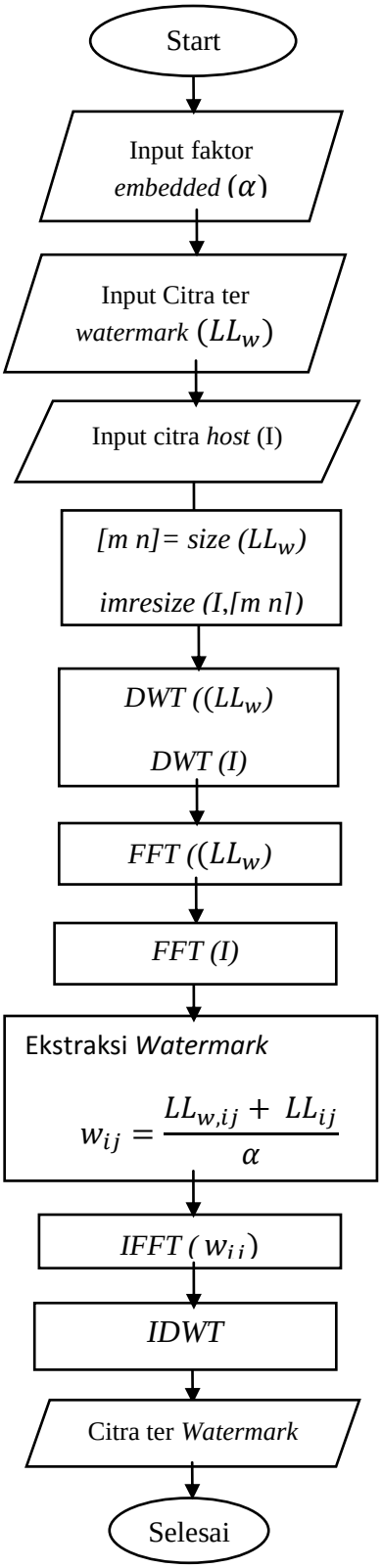
Proses ekstraksi *watermark* dari citra digital (*file host*) untuk metode *DWT-FFT* dijelaskan sebagai berikut :

1. Menentukan faktor *embedded* (α).
2. Input citra ter *watermark* (LL_w) *dan* citra *host (I).*
3. Mengubah ukuran citra *host* (I) sesuai dengan ukuran citra ter -*watermark* (LL_w)
4. Menerapakan metode *DWT* pada *watermarked image* (LL_w) *dan* citra *host (I)* untuk mendapatkan sub level.
5. Pilih koefisien detail approkmasi (LL) pada *image* (LL_w) *dan* citra *host (I)* .
6. Terapkan *FFT* pada detail approkmasi (LL).
7. Ekstraksi *watermark* dengan menggunkan algoritma ekstraksi yaitu dengan mengurangi matriks *watermarked image* dengan *file host* kemudian dibagi dengan factor α;

$$w_{ij} = \frac{LL_{w,ij} + LL_{ij}}{\alpha}$$

8. Menerapkan *IFFT* pada *watermarking image.*
9. Dari *watermark image* yang berupa fungsi lakukan proses *IDWT* untuk rekonstruksi citra ter *watermark*.

Proses ekstraksi *watermark* dari citra digital (*file host*) untuk metode *DWT-FFT* dapat dilihat pada *flowchart* di bawah ini yang ditunjukan pada gambar III.4.



Gambar III.4 Alur kerja proses *ekstraksi* citra *watermark* pada citra ter *watermark* dengan mengunakan metode *DWT-FFT*

BAB IV

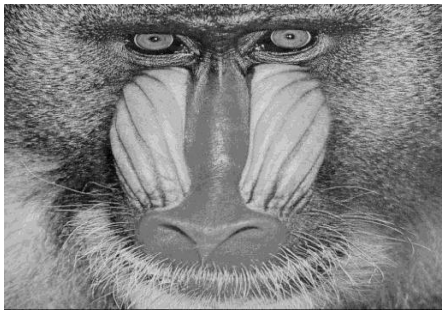
HASIL DAN PEMBAHASAN

IV.1 Citra Asli dan Citra *Watermark*

Pada penulisan skripsi ini, citra asli yang digunakan adalah citra *grayscale*. Citra asli yang digunakan juga berbeda ukuran, dimana citra asli antara lain berdimensi 960x453 (Gambar 4.1), 512x512 (Gambar 4.2) , 512x512 (Gambar 4.3), 493x356 (Gambar 4.4) dan 752x600 (Gambar 4.5). Berikut adalah citra asli yang digunakan dalam penelitian ini.

Gambar 4.1 Car

Gambar 4.2 Lena



Gambar 4.3 Baboon

Gambar 4.4 Butterfly

Gambar 4.5 starry_night

Adapun gambar 4.6 dengan ukuran dimensi 629x629 merupakan citra *watermark* pada penelitian kali ini.

Gambar 4.6 *Watermark*

IV. 2 Penyisipan dan Ekstraksi Citra *Watermark*

Proses penyisipan *watermark* ke dalam citra asli menggunakan metode *Fast Fourier Transform* dan *Discreate Wavelet Transform* (*level* 1, 2 dan 3). Pada penelitian ini dilakukan beberapa percobaan *watermarking* dimana dari beberapa hasil percobaan *watermarking* akan dibandingkan untuk melihat kualitas citra setelah disisipi citra *watermark* dengan melihat nilai PSNR dan NC. Dalam menentukan proses *watermarking* terbaik pada citra, diperlukan pengaturan nilai α dan β yang mewakili nilai 0 dan 1 dalam citra *watermark* biner. Dalam penentuan α dan β diambil interval nilai untuk α dan β adalah 0.01. Nilai α dan β adalah nilai pengali citra asli sebagai perwakilan dari nilai pada citra watermark

biner. Adapun kualitas citra setelah disisipkan citra *watermark* dapat di lihat pada tabel 4.1.

Tabel 4.1 Perbandingan Nilai PSNR dan NC tanpa *attack*

Metode *Watermarking*	**Citra *Watermarking***	**PSNR**	**NC**
FFT	Car	38.9538	0.938668
	Lena	38.9114	0.930603
	Baboon	38.9114	0.931659
	Butterfly	38.9457	0.994481
	starry_night	38.9116	0.947454
DWT-FFT (Level 1)	Car	29.3358	0.849446
	Lena	38.9174	0.86818
	Baboon	38.9174	0.888803
	Butterfly	23.2418	0.94549
	starry_night	38.9142	0.912632
DWT-FFT (Level 2)	Car	29.336	0.972758
	Lena	38.9279	0.961469
	Baboon	39.9279	0.964326
	Butterfly	23.2421	1
	starry_night	38.9237	0.983128

DWT-FFT (Level 3)	Car	29.3374	0.974343
	Lena	38.9447	0.965329
	Baboon	38.9447	0.966879
	Butterfly	23.2427	1
	starry_night	38.9399	0.984348

Pada table 4.1 hasil proses *watermarking* dapat dilihat dari nilai PSNR, dari metode yang digunakan yakni metode *FFT* dan metode *DWT-FFT* kualitas hasil *watermarking* yang lebih baik dapat dilihat pada metode *DWT-FFT* karena pada metode *DWT-FFT* mempresentasikan nilai PSNR semakin besar seiring dengan menaikan penggunaan level 1,2 dan 3, Walaupun terdapat dua hasil *watermarking* yang mengalami penurunan kualitas yakni pada citra *watermarking* Car dan Butterfly masing-masing menunjukan nilai PSNR sebesar 29.3358 dan 23.2418 pada *level* 1 yang sebelumnya pada nilai PSNR dengan metode *FFT* sebesar 38.9538 dan 38.9457 begitu pula pada level 2 dan 3. Hal ini menunjukan bahwa citra asli Car dan Butterfly memiliki kualitas citra terbesar dan paling baik pada metode *FFT* dibandingkan dengan metode *DWT-FFT* dan untuk citra yang lain yakni citra asli Lena, Baboon dan Starry_night memiliki kualitas paling baik pada metode *DWT-FFT*. Semakin naik *level* yang digunakan maka semakin besar pula nilai PSNR yang dihasilkan, dimana kualitas nilai PSNR yang lebih baik di tunjukan pada *level* 3 yakni 38.9447 (Lena), 38.9447 (Baboon), dan 38.9399 (Starry_night).

Hasil proses Ekstraksi *watermarking* dapat dilihat dari nilai NC, dari metode yang digunakan yakni metode *FFT* dan metode *DWT-FFT* kualitas yang lebih baik ditunjukan pada metode *DWT-FFT* level 3. Dari nilai NC ini mengindikasikan bahwa kemiripan citra antara *watermark* ekstrak dengan *watermark* asli memiliki tingkat kemiripan tidak jauh berbeda meskipun informasi pada *watermark* ekstrak ada yang tidak sempurna.

IV. 3 Ketahanan Citra ter-*Watermark*

IV.3.1 *Salt & Pepper*

Noise *Salt & Pepper* yang di uji pada citra akan nampak seperti titik-titik. Untuk citra RGB titik-titik muncul dalam tiga warna yakni merah (*red*), hijau (*green*) dan biru (*blue*), sedangkan pada citra *Grayscale noise* akan muncul dalam dua warna yakni hitam (*black*) dan putih (*white*). *Noise* ini memberikan efek "*on* dan *off*" pada piksel. Pada Matlab kita dapat mengatur konstanta noise. Konstanta berupa angka numerik non gaussian dengan range 0 sampai dengan 1. Makin besar konstantanya citra akan semakin kabur, sebaliknya makin kecil konstantanya efek pada citra makin tidak terlihat (Hamsir, 2017). Nilai default untuk konstanta *noise* ini adalah 0.001. Dalam percobaan ini, *noise salt & pepper* akan di uji pada citra hasil *watermarking* dimensi 960x453, 512x512, 493x356 dan 752x600 pixel. Berikut ini pada tabel 4.2 yang menunjukan nilai PSNR dan NC sebelum dan sesudah penambahan *noise Salt & Pepper* pada citra.

Tabel 4.2 Hasil nilai PSNR dan NC sebelum dan sesudah penambahan *noise salt & pepper* pada citra

Metode *Watermarking*	**Citra *Watermarking***	**Sebelum**		**Sesudah**	
		PSNR	**NC**	**PSNR**	**NC**
FFT	Car	38.9538	0.938668	33.1544	0.938246
	Lena	38.9114	0.930603	33.9645	0.930153
	Baboon	38.9114	0.931659	33.9015	0.931227
	Butterfly	38.9457	0.994481	33.3506	0.994022
	starry_night	38.9116	0.947454	33.4917	0.947406

DWT-FFT (Level 1)	Car	29.3358	0.849446	28.4309	0.848886
	Lena	38.9174	0.86818	33.5093	0.956356
	Baboon	38.9174	0.888803	33.9612	0.888123
	Butterfly	23.2418	0.94549	23.0557	0.94516
	starry_night	38.9142	0.912632	33.6551	0.912221
DWT-FFT (Level 2)	Car	29.336	0.972758	28.5298	0.96852
	Lena	38.9279	0.961469	33.639	0.953756
	Baboon	39.9279	0.964326	33.7285	0.956866
	Butterfly	23.2421	1	23.0031	1
	starry_night	38.9237	0.983128	33.4906	0.977672
DWT-FFT (Level 3)	Car	29.3374	0.974343	28.4702	0.961976
	Lena	38.9447	0.965329	33.6993	0.962597
	Baboon	38.9447	0.966879	34.0692	0.965507
	Butterfly	23.2427	1	23.0332	1
	starry_night	38.9399	0.984348	33.5165	0.982188

Pada tabel 4.2 dapat dilihat perbedaan nilai PSNR dan NC setiap citra ter-*watermark* sebelum dan sesudah diberikan *noise Salt & Pepper*. Nilai PSNR pada citra yang telah diberikan *noise* lebih kecil dibandingkan sebelum diberikan *noise*, sehingga mengakibatkan kualitas citra ter-*watermark* menjadi lebih rendah dibandingkan sebelum diberikan *noise*. Sedangkan untuk citra *watermarking* pada hasil ekstraksi setelah penambahan *noise* memiliki nilai NC lebih rendah dibandingkan sebelum diberikan *noise* walaupun terdapat salah satu citra

watermarking yang mempertahankan nilai NC nya, sehingga citra *watermark* mengalami kerusakan.

Dari hasil analasis nilai PSNR dari dua metode *watermarking* diatas setelah dilakukan penambahan *noise Salt & Pepper*, kualitas citra *watermaking* yang baik terlihat pada metode *FFT*. Akan tetapi dengan melihat satu metode yakni metode *watermarking DWT-FFT* dengan menggunakan tiga level, Kualitas citra *watermarking* yang baik ditunjukan pada level tiga. Sedangkan dari hasil analisis nilai NC dari dua metode diatas, kualitas citra *watermarking* hasil ekstraksi yang baik terlihat pada metode *DWT-FFT* yakni pada level tiga.

Berikut ini adalah salah satu gambar dari contoh citra ter-*watermark* yang diberi *noise salt & pepper* dan hasil ekstraksi citra ter-*watermark* yang disisipkan.

(a) (b)

Gambar 4.7 Citra *watermarking* (a) dan Hasil ekstraksi (b) (*FFT)*

(a) (b)

Gambar 4.8 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 1)


(a) (b)

Gambar 4.9 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 2)


(a) (b)

Gambar 4.10 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 3)

IV.3.2 *Rotation*

Rotasi merupakan suatu transformasi geometri dengan memindahkan nilai piksel dari posisi awal menuju posisi akhir yang ditentukan melalui nilai variable rotasi sebesar θ° terhadap sudut θ° atau garis horizontal dari citra (Hamsir, 2017).

Jika citra $B(x', y')$ adalah hasil rotasi sebesar θ dari citra $A(x, y)$ yang tumpuannya berada di titik pusat citra $A(\frac{a}{2}, \frac{b}{2})$, maka rotasi citra dari A ke :

$$B[x'][y'] = B\left[\left(x - \frac{a}{2}\right) cos\delta + \left(y - \frac{b}{2}\right) sin\delta\right]\left[-\left(x - \frac{a}{2}\right) sin\delta + \left(y - \frac{b}{2}\right) cos\delta\right]$$

Berikut ini pada tabel 4.3 yang menunjukan nilai PSNR dan NC setelah citra diberikan serangan rotasi.

Tabel 4.3 Hasil sebelum dan sesudah citra dilakukan serangan *rotation*

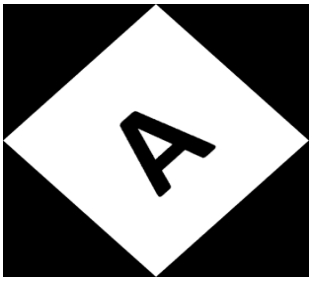
Metode *Watermarking*	**Citra *Watermarking***	**Sebelum**		**Sesudah**	
		PSNR	**NC**	**PSNR**	**NC**
FFT	Car	38.9538	0.938668	42.5865	0.939365
	Lena	38.9114	0.930603	41.9401	0.931468
	Baboon	38.9114	0.931659	41.9404	0.93244
	Butterfly	38.9457	0.994481	42.0903	0.972429
	starry_night	38.9116	0.947454	41.9917	0.944723
DWT-FFT (Level 1)	Car	29.3358	0.849446	32.1565	0.816855
	Lena	38.9174	0.86818	41.946	0.860581
	Baboon	38.9174	0.888803	41.9462	0.876152
	Butterfly	23.2418	0.94549	28.4354	0.842589
	starry_night	38.9142	0.912632	41.9945	0.891705
DWT-FFT (Level 2)	Car	29.336	0.972758	32.157	0.927957
	Lena	38.9279	0.961469	41.9565	0.972256
	Baboon	39.9279	0.964326	41.9568	0.976983
	Butterfly	23.2421	1	28.4363	0.899851
	starry_night	38.9237	0.983128	42.0038	0.994017
DWT-FFT (Level 3)	Car	29.3374	0.974343	32.158	0.927092
	Lena	38.9447	0.965329	41.9734	0.977973

	Baboon	38.9447	0.966879	41.9735	0.980156
	Butterfly	23.2427	1	28.4369	0.910249
	starry_night	38.9399	0.984348	42.0201	0.994279

Pada tabel 4.3 dapat dilihat perbedaan nilai PSNR dan NC setiap citra ter-*watermark* sebelum dan sesudah dilakukan serangan rotasi. Nilai PSNR pada citra *watermarking* yang telah diberikan serangan rotasi dengan metode *watermaking FFT* lebih besar dibandingkan sebelum diberikan serangan, sehingga kualitas citra ter-*watermark* lebih baik dibandingkan sebelum diberikan serangan rotasi. Hal ini pula ditunjukan untuk nilai PSNR pada citra *watermarking* yang telah diberikan serangan rotasi dengan metode *watermarking DWT-FFT* lebih besar dibandingkan sebelum diberikan serangan rotasi, sehingga kualitas citra *watermarking* lebih baik dibandingkan sebelum diberikan serangan rotasi. Untuk citra *watermarking* pada hasil ekstraksi memiliki nilai NC lebih baik dengan menggunkan metode *DWT-FFT* dibandingkan dengan metode *FFT* setelah diberikan serangan, sehingga kualitas citra hasil ekstraksi *watermarking* pada metode *DWT-FFT* lebih baik dibandingkan dengan metode *FFT*.

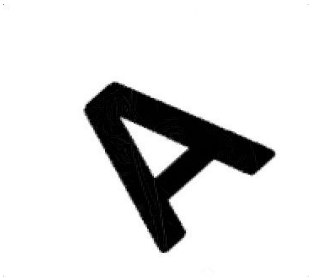
Dari hasil analasis nilai PSNR dari dua metode *watermarking* diatas sesudah diberikan serangan rotasi, kualitas citra *watermarking* yang baik terlihat pada metode *FFT*. Akan tetapi dengan melihat satu metode yakni metode *watermarking DWT-FFT* dengan menggunakan tiga level, kualitas citra *watermarking* yang baik ditunjukan pada level tiga. Sedangkan dari hasil analisis nilai NC dari dua metode diatas, kualitas citra *watermarking* hasil ekstraksi yang baik terlihat pada metode *DWT-FFT* yakni pada level tiga.

Berikut ini adalah salah satu gambar dari contoh citra ter-*watermark* yang diberi serangan rotasi dan hasil ekstraksi citra ter-*watermark* yang disisipkan.

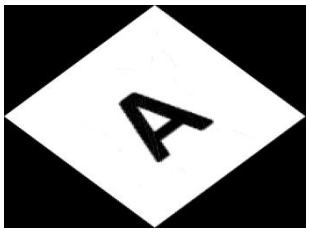
(a) (b)

Gambar 4.11 Citra *watermarking* (a) dan Hasil ekstraksi (b) *FFT*

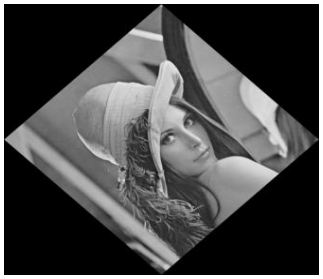
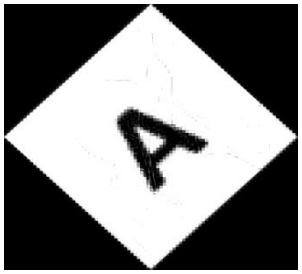
(a) (b)

Gambar 4.12 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 1)

(a) (b)

Gambar 4.13 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 2)

(a) (b)

Gambar 4.14 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 3)

IV.3.3 *Speckle*

Speckle merupakan noise ganda. Noise ini ditambahkan pada citra menggunakan persamaan $J = I + n * I$, dimana n terdistribusi random seragam dengan mean 0 dan variance *V*. *V* adalah konstanta non negative yang besarnya dapat berubah-ubah. Makin besar nilai V maka citra akan semakiin kabur (Putra,2010). Nilai default untuk konstanta *noise* ini adalah 0.0001. Dalam percobaan ini, *noise speckle* akan di uji pada citra hasil *watermarking* dimensi 960x453, 512x512, 493x356 dan 752x600 pixel. Berikut ini pada tabel 4.4 yang menunjukan nilai PSNR dan NC sebelum dan sesudah penambahan *noise Speckle* pada citra.

Tabel 4.4 Hasil sesudah penambahan *noise speckle* pada citra

Metode *Watermarking*	**Citra *Watermarking***	**Sebelum**		**Sesudah**	
		PSNR	**NC**	**PSNR**	**NC**
FFT	Car	38.9538	0.938668	37.5489	0.863699
	Lena	38.9114	0.930603	38.0043	0.924473
	Baboon	38.9114	0.931659	37.9748	0.928119
	Butterfly	38.9457	0.994481	38.0786	0.966675
	starry_night	38.9116	0.947454	38.3104	0.956359
DWT-FFT (Level 1)	Car	29.3358	0.849446	29.1924	0.898584
	Lena	38.9174	0.86818	38.0072	0.908802
	Baboon	38.9174	0.888803	37.9692	0.913654
	Butterfly	23.2418	0.94549	23.2233	0.950077
	starry_night	38.9142	0.912632	38.313	0.927108

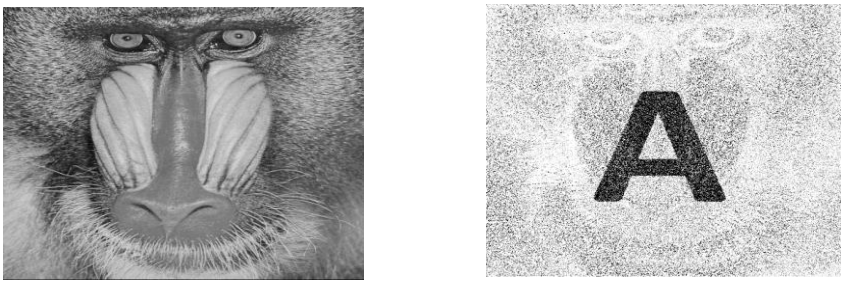
DWT-FFT (Level 2)	Car	29.336	0.972758	29.1902	0.976777
	Lena	38.9279	0.961469	38.0153	0.96503
	Baboon	39.9279	0.964326	37.9839	0.966427
	Butterfly	23.2421	1	23.2248	1
	starry_night	38.9237	0.983128	38.3155	0.984019
DWT-FFT (Level 3)	Car	29.3374	0.974343	29.1923	0.976127
	Lena	38.9447	0.965329	38.0437	0.965368
	Baboon	38.9447	0.966879	38.0126	0.970204
	Butterfly	23.2427	1	23.2246	1
	starry_night	38.9399	0.984348	38.3294	0.984295

Pada tabel 4.4 dapat dilihat perbedaan nilai PSNR dan NC setiap citra ter-*watermark* sebelum dan sesudah diberikan *noise speckle*. Nilai PSNR pada citra yang telah diberikan *noise* lebih kecil dibandingkan sebelum diberikan *noise*, sehingga mengakibatkan kualitas citra ter-*watermark* menjadi lebih rendah dibandingkan sebelum diberikan *noise*. Sedangkan untuk citra *watermarking* pada hasil ekstraksi setelah penambahan *noise* rata-rata memiliki nilai NC lebih tinggi dibandingkan sebelum diberikan *noise* sehingga kemiripan citra antara *watermark* ekstrak dengan *watermark* asli memiliki tingkat kemiripan tidak jauh berbeda

Dari hasil analasis nilai PSNR dari dua metode *watermarking* diatas setelah dilakukan penambahan *noise speckle*, kualitas citra *watermaking* yang baik terlihat pada metode *FFT*. Akan tetapi dengan melihat satu metode yakni metode *watermarking DWT-FFT* dengan menggunakan tiga level, Kualitas citra *watermarking* yang baik ditunjukan pada level tiga. Sedangkan dari hasil analisis

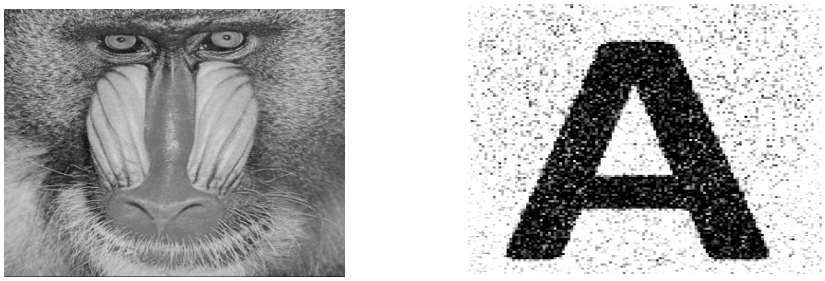
nilai NC dari dua metode diatas, kualitas citra *watermarking* hasil ekstraksi yang baik terlihat pada metode *DWT-FFT* yakni pada level tiga.

Berikut ini adalah salah satu gambar dari contoh citra ter-*watermark* yang diberi *noise speckle* dan hasil ekstraksi citra ter-*watermark* yang disisipkan.



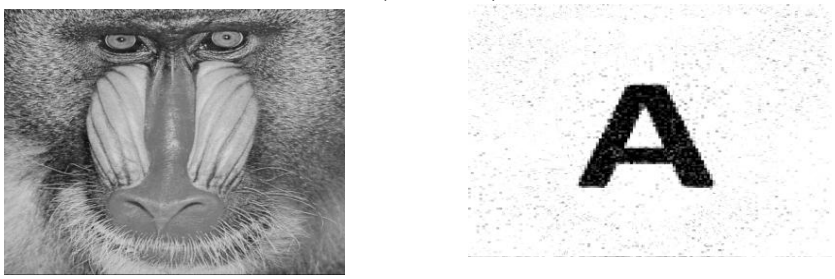
(a) (b)

Gambar 4.15 Citra *watermarking* (a) dan Hasil ekstraksi (b) *FFT*



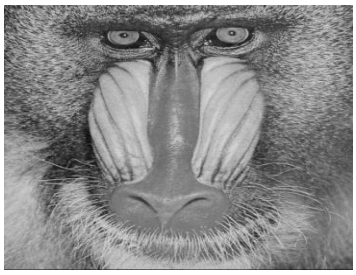
(a) (b)

Gambar 4.16 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 1)

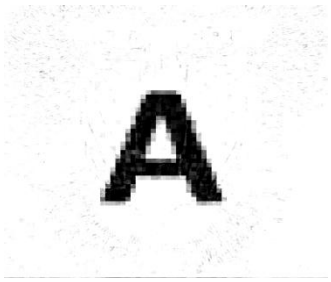


(a) (b)

Gambar 4.17 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 2)



(a)



(b)

Gambar 4.18 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 3)

IV.3.4 *Gaussian*

Disebut juga *Gaussian White Noise*. Untuk menambahkan noise ini pada Matlab memerlukan input tambahan berupa rata-rata dan variasi. Rata-rata dan variasi merupakan suatu konstanta real. Nilai gaussian positif maupun negative. Makin besar rata-rata dan variasinya maka citra akan semakin kabur, sebaliknya makin kecil konstantanya efek pada citra makin tidak terlihat. Nilai default adalah 0 untuk mean dan 0.01 untuk variance. Disebut white noise karena pada saat nilai rata-rata dan variasinya besar maka citra seolah-olah hanya terlihat seperti citra putih saja (Putra,2010). Nilai default untuk konstanta *noise* ini adalah 0.0001. Dalam percobaan ini, *noise 54aussian* akan di uji pada citra hasil *watermarking* dimensi 960x453, 512x512, 493x356 dan 752x600 pixel.

Berikut ini pada tabel 4.5 yang menunjukan nilai PSNR dan NC sebelum dan sesudah penambahan *noise gaussian* pada citra.

Tabel 4.5 Hasil sesudah penambahan *noise gaussian* pada citra

Metode *Watermarking*	**Citra *Watermarking***	**Sebelum**		**Sesudah**	
		PSNR	**NC**	**PSNR**	**NC**
FFT	Car	38.9538	0.938668	36.4364	0.812493
	Lena	38.9114	0.930603	36.3798	0.806892

	Baboon	38.9114	0.931659	36.3915	0.808999
	Butterfly	38.9457	0.994481	36.432	0.85767
	starry_night	38.9116	0.947454	39.3957	0.822638
DWT-FFT (Level 1)	Car	29.3358	0.849446	29.0464	0.974449
	Lena	38.9174	0.86818	36.4023	0.953965
	Baboon	38.9174	0.888803	36.395	0.95637
	Butterfly	23.2418	0.94549	23.1695	1
	starry_night	38.9142	0.912632	36.3996	0.972312
DWT-FFT (Level 2)	Car	29.336	0.972758	29.0447	0.972758
	Lena	38.9279	0.961469	36.3911	0.966714
	Baboon	39.9279	0.964326	36.3994	0.967158
	Butterfly	23.2421	1	23.169	1
	starry_night	38.9237	0.983128	36.3991	0.986149
DWT-FFT (Level 3)	Car	29.3374	0.974343	29.0468	0.979136
	Lena	38.9447	0.965329	36.408	0.966645
	Baboon	38.9447	0.966879	36.4299	0.967307
	Butterfly	23.2427	1	23.1717	1
	starry_night	38.9399	0.984348	36.4083	0.985414

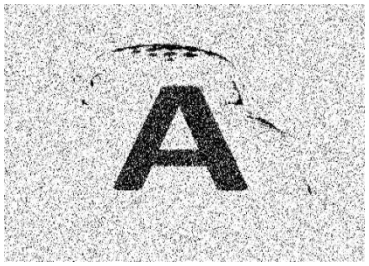
Pada tabel 4.5 dapat dilihat perbedaan nilai PSNR dan NC setiap citra ter-*watermark* sebelum dan sesudah diberikan *noise gaussian*. Nilai PSNR pada citra yang telah diberikan *noise* lebih kecil dibandingkan sebelum diberikan *noise*, sehingga mengakibatkan kualitas citra ter-*watermark* menjadi lebih rendah dibandingkan sebelum diberikan *noise*. Sedangkan untuk citra *watermarking* pada hasil ekstraksi setelah penambahan *noise* rata-rata memiliki nilai NC lebih rendah dibandingkan sebelum diberikan *noise* sehingga kemiripan citra antara *watermark* ekstrak dengan *watermark* asli memiliki tingkat kemiripan berbeda.

Dari hasil analasis nilai PSNR dari dua metode *watermarking* diatas setelah dilakukan penambahan *noise gaussian*, kualitas citra *watermarking* yang baik terlihat pada metode *FFT*. Akan tetapi dengan melihat satu metode yakni metode *watermarking DWT-FFT* dengan menggunakan tiga level, Kualitas citra *watermaking* yang baik ditunjukan pada level tiga. Sedangkan dari hasil analisis nilai NC dari dua metode diatas, kualitas citra *watermarking* hasil ekstraksi yang baik terlihat pada metode *DWT-FFT* yakni pada level tiga.

Berikut ini adalah salah satu gambar dari contoh citra ter-*watermark* yang diberi *noise Gaussian* dan hasil ekstraksi citra ter-*watermark* yang disisipkan.

(a) (b)

Gambar 4.19 Citra *watermarking* (a) dan Hasil ekstraksi (b) *FFT*

(a) (b)

Gambar 4.20 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 1)


(a) (b)

Gambar 4.21 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 2)


(a) (b)

Gambar 4.22 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 3)

IV.3.5 *Compression*

Image compression atau yang disebut juga kompresi citra adalah proses untuk meminimalisasi jumlah bit yang merepresentasikan suatu citra sehingga ukuran data citra menjadi lebih kecil. Pada dasarnya teknik kompresi citra digunakan pada proses transmisi data (*data transmission*) dan penyimpanan data (*data storage*). Teknik kompresi JPEG biasanya digunakan untuk foto atau citra di website. JPEG menggunakan kompresi tipe lossy. Kualitas JPEG 2000 bisa

bervariasi tergantung setting kompresi yang digunakan.. Tingkat kompresi yang baik untuk JPEG adalah 10:1-20:1 untuk citra foto, 30:1-50:1 untuk citra web, dan 60:1-100:1 untuk kualitas rendah seperti citra untuk ponsel (Putra, 2010).

Berikut ini pada tabel 4.6 yang menunjukan nilai PSNR dan NC sebelum dan sesudah dilakukan serangan *compression* pada citra.

Tabel 4.6 Hasil sebelum dan sesudah dilakukan serangan *compression* pada citra

Metode *Watermarking*	**Citra *Watermarking***	**Sebelum**		**Sesudah**	
		PSNR	**NC**	**PSNR**	**NC**
FFT	Car	38.9538	0.938668	38.461	0.938211
	Lena	38.9114	0.930603	38.4247	0.941978
	Baboon	38.9114	0.931659	36.8478	0.844137
	Butterfly	38.9457	0.994481	37.4507	0.965589
	starry_night	38.9116	0.947454	38.8834	0.94726
DWT-FFT (Level 1)	Car	29.3358	0.849446	29.2046	0.980018
	Lena	38.9174	0.86818	38.4326	0.964222
	Baboon	38.9174	0.888803	36.8525	0.965706
	Butterfly	23.2418	0.94549	23.2297	1
	starry_night	38.9142	0.912632	38.8861	0.982374
DWT-FFT (Level 2)	Car	29.336	0.972758	29.2045	0.972758
	Lena	38.9279	0.961469	38.4406	0.965574
	Baboon	39.9279	0.964326	36.8583	0.9675

	Butterfly	23.2421	1	23.2299	1
	starry_night	38.9237	0.983128	38.8963	0.983539
DWT-FFT (Level 3)	Car	29.3374	0.974343	29.2056	0.976874
	Lena	38.9447	0.965329	38.4575	0.965832
	Baboon	38.9447	0.966879	36.8697	0.967591
	Butterfly	23.2427	1	23.2304	1
	starry_night	38.9399	0.984348	38.9132	0.983966

Pada tabel 4.6 dapat dilihat perbedaan nilai PSNR dan NC setiap citra ter-*watermark* sebelum dan sesudah dilakukan serangan *compression*. Nilai PSNR pada citra yang telah dikompresi lebih kecil dibandingkan sebelum diberikan *noise*, sehingga mengakibatkan kualitas citra ter-*watermark* menjadi lebih rendah dibandingkan sebelum dilakukan kompresi . Sedangkan untuk citra *watermarking* pada hasil ekstraksi setelah dilakukan kompresi rata-rata memiliki nilai NC lebih rendah dibandingkan sebelum dilakukan kompresi sehingga kemiripan citra antara *watermark* ekstrak dengan *watermark* asli memiliki tingkat kemiripan berbeda.

Dari hasil analasis nilai PSNR dari dua metode *watermarking* diatas setelah dilakukan kompresi, kualitas citra *watermarking* yang baik terlihat pada metode *FFT*. Akan tetapi dengan melihat satu metode yakni metode *watermarking DWT-FFT* dengan menggunakan tiga level, Kualitas citra *watermarking* yang baik ditunjukan pada level tiga. Sedangkan dari hasil analisis nilai NC dari dua metode diatas, kualitas citra *watermarking* hasil ekstraksi yang baik terlihat pada metode *DWT-FFT* yakni pada level tiga.

Berikut ini adalah salah satu gambar dari contoh citra ter-*watermark* yang dikompresi dan hasil ekstraksi citra ter-*watermark* yang disisipkan.

(a) (b)

Gambar 4.23 Citra *watermarking* (a) dan Hasil ekstraksi (b) *FFT*

(a) (b)

Gambar 4.24 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 1)

(a) (b)

Gambar 4.25 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 2)

(a) (b)

Gambar 4.26 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 3)

IV.3.6 *Resizing*

Resize gambar / citra adalah mengubah ukuran panjang dan lebar gambar. Nilai default untuk konstanta *resizing* ini adalah 0.5. Dalam percobaan ini, *resizing* akan diuji pada citra hasil *watermarking* dimensi 960x453, 512x512, 493x356 dan 752x600 pixel. Berikut ini pada tabel 4.7 yang menunjukan nilai PSNR dan NC sebelum dan sesudah penambahan diberikan serangan *resizing* pada citra

Tabel 4.7 Hasil sesudah dilakukan serangan *resizing* pada citra

Metode *Watermarking*	**Citra *Watermarking***	**Sebelum**		**Sesudah**	
		PSNR	**NC**	**PSNR**	**NC**
FFT	Car	38.9538	0.938668	38.9622	0.938043
	Lena	38.9114	0.930603	38.9124	0.931915
	Baboon	38.9114	0.931659	38.9119	0.932127
	Butterfly	38.9457	0.994481	38.9684	0.965678
	starry_night	38.9116	0.947454	38.9547	0.937101
DWT-FFT (Level 1)	Car	29.3358	0.849446	28.6093	0.870115
	Lena	38.9174	0.86818	32.7289	0.883017
	Baboon	38.9174	0.888803	23.3198	0.886441
	Butterfly	23.2418	0.94549	21.9562	0.94025
	starry_night	38.9142	0.912632	25.0018	0.915117
DWT-FFT (Level 2)	Car	29.336	0.972758	28.6102	0.980333
	Lena	38.9279	0.961469	32.731	0.974948

	Baboon	39.9279	0.964326	23.3201	0.97638
	Butterfly	23.2421	1	21.9566	1
	starry_night	38.9237	0.983128	25.0022	0.997643
DWT-FFT (Level 3)	Car	29.3374	0.974343	28.6111	0.984683
	Lena	38.9447	0.965329	32.7361	0.979247
	Baboon	38.9447	0.966879	23.3206	0.981239
	Butterfly	23.2427	1	21.9569	1
	starry_night	38.9399	0.984348	25.0028	1

Pada tabel 4.7 dapat dilihat perbedaan nilai PSNR dan NC setiap citra ter-*watermark* sebelum dan sesudah dilakukan serangan *resizing*. Nilai PSNR pada citra *watermarking* yang telah diberikan serangan *resizing* dengan metode *watermaking FFT* lebih besar dibandingkan sebelum diberikan serangan, sehingga kualitas citra ter-*watermark* lebih baik dibandingkan sebelum diberikan serangan. Sedangkan nilai PSNR pada citra *watermarking* yang telah diberikan serangan dengan metode *watermarking DWT-FFT* lebih kecil dibandingkan sebelum diberikan serangan, sehingga kualitas citra *watermarking* kurang baik dibandingkan sebelum diberikan serangan. Untuk citra *watermarking* pada hasil ekstraksi setelah dilakukan serangan memiliki nilai NC lebih baik dengan menggunkan metode *FFT* dibandingkan dengan metode *DWT-FFT* sebelum diberikan serangan, sehingga kualitas citra *watermarking* pada metode *FFT* lebih baik sedangkan dengan metode *DWT-FFT* mengalami kerusakan.

Dari hasil analasis nilai PSNR dari dua metode *watermarking* diatas sesudah diberikan serangan *resizing*, kualitas citra *watermarking* yang baik terlihat pada metode *FFT*. Akan tetapi dengan melihat satu metode yakni metode *watermarking DWT-FFT* dengan menggunakan tiga level, Kualitas citra

watermarking yang baik ditunjukan pada level tiga. Sedangkan dari hasil analisis nilai NC dari dua metode diatas, kualitas citra *watermarking* hasil ekstraksi yang baik terlihat pada metode *DWT-FFT* yakni pada level tiga.

Berikut ini adalah salah satu gambar dari contoh citra ter-*watermark* yang diberi serangan *resizing* dan hasil ekstraksi citra ter-*watermark* yang disisipkan.

(a) (b)

Gambar 4.27 Citra *watermarking* (a) dan Hasil ekstraksi (b) *FFT*

(a) (b)

Gambar 4.28 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 1)

(a) (b)

Gambar 4.29 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 2)

(a)

(b)

Gambar 4.30 Citra *watermarking* (a) dan Hasil ekstraksi (b) *DWT-FFT* (Level 3)

BAB V

KESIMPULAN DAN SARAN

V.1 Kesimpulan

Berdasarkan hasil penelitian dari beberapa citra asli dan citra *watermark* yang dipilih, adapun kesimpulan yang dapat diambil dari penelitian ini sebagai berikut :

1. Proses penyisipan citra *watermark* ke citra *host* dengan menggunakan metode *FFT* dan *DWT-FFT* digunakan persamaan brikut :
   - Metode *FFT*

     $$I_w(i,j) = I(i,j) + \beta W(i,j)$$

   - Metode *DWT-FFT*

     $$LL_w(i,j) = LL(i,j) + \alpha W(i,j)$$

2. Kualitas hasil *watermarking* pada sub level 1, 2, dan 3 pada metode *DWT-FFT* sebelum diberikan serangan yakni *Compression, Salt* dan *Pepper, Gausion, Speckle, Resizing,* dan *Rotation* dapat dilihat dengan parameter nilai PSNR. Dari hasil analisis bahwa sub level 3 memberikan kualitas yang lebih baik dibandingkan dengan sub level 1 dan 2, hal ini menunjukkan bahwa perbandingan kualitas citra *watermarking* tidak jauh berbeda dengan citra sebelum disisipkan sehingga citra asli masih jelas dibaca.
3. Kualitas hasil ekstraksi *watermark* setelah dilakukan dengan beberapa serangan yakni *Compression, Salt* dan *Pepper, Gausion, Speckle, Resizing,* dan *Rotation* pada metode *FFT* dan *DWT-FFT* dapat dilihat dengan parameter nilai NC. Dari hasil analisis dari kedua metode tersebut kualitas hasil ekstraksi yang baik terlihat dengan menggunakan metode *DWT-FFT* pada sub level 3. Hal ini menunjukkan bahwa kemiripan citra hasil ekstraksi tidak jauh berbeda dengan citra *watermark* sehingga citra *watermark* masih jelas dibaca.

V.2 Saran

Penelitian mengenai algoritma *watermarking* menggunakan metode *FFT* dan *DWT-FFT* dapat dikembangkan lebih lanjut. Penelitian lain yang dapat dikembangkan antara lain mengganti dengan penggabungan metode yang lain, mengganti dengan beberapa *mother wavelet* serta melakukan beberapa serangan yang berbeda dan menambah sub-level metode *DWT-FFT*.

DAFTAR PUSTAKA

Agrawal D., Gupta V., Mehta G. 2013. *Skripsi Digital Watermarking Technique using Discrete Cosine Transform*. Departement of Electronic & Communication, India.

Aliwa, M.B., Tobely T.A., Fahmy M.M., Nasr M.S.,Aziz M.H.A. 2009. *Skripsi Robust Digital Watermarking Based Falling-off-Boundary in Corners Board-MSB-6 Grayscale Images.* Faculty of Engineering-TANTA University, Mesir.

Fahmi. 2007. *Skripsi Studi dan Implementasi Watermarking Citra Digital DenganMenggunakan Fungsi Hash*. Institut Teknologi Bandung, Bandung.

Hamsir, Nurul Haj. 2017. *Watermarking Image Menggunakan Discrete Wavelet Transform*. Makassar : Universitas Hasanuddin.

Putra, Darma. (2010). *Pengolahan Citra Digital*. Andi Offset, Yogyakarta.

Sianipar, R.H. 2013. *Pemrograman MATLAB dalam contoh dan terapan.* INFORMATIKA Bandung, Bandung.

Supangkat, K. J. (2000). *Skripsi Watermarking Sebagai Teknik Penyembunyian Hak Cipta pada Data Digital.* Institute Teknologi Bandung, Bandung.

Sitorus, P.P. 2008. *Skripsi Watermarking pada Citra Digital Menggunakan Discrete Cosine Transform.* Universitas Sumatera Utara, Medan.

Sugiono J.P, Setiawan Y. 2008. *Skripsi Watermarking Pada File Audio PCM WAVE Dengan Metode Echo Data Hiding*. Konferensi Nasional Sistem dan Informatika 2008; Nopember 15, 2008, Bali.

Tiwari, A. (2015). *Digital Watermarking Analysis Using DCT And DWT*. International Journal of Emerging Technology and Innovative Engineering Volume I, Issue 6, June 2015 (ISSN: 2394 – 6598).

Tuakia, N. (2013). *Skripsi Implementasi Watermarking Pada Citra Medis Menggunakan Metode Discrete Wavelet Transform*. Universitas Brawijaya, Malang.

LAMPIRAN

Lampiran I

Program *Watermarking* gui dengan metode *FFT* pada matlab R2013a

```
function varargout = Penyisip(varargin)
% PENYISIP MATLAB code for Penyisip.fig
%      PENYISIP, by itself, creates a new PENYISIP or raises the
existing
%      singleton*.
%
%      H = PENYISIP returns the handle to a new PENYISIP or the
handle to
%      the existing singleton*.
%
%      PENYISIP('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in PENYISIP.M with the given input
arguments.
%
%      PENYISIP('Property','Value',...) creates a new PENYISIP or
raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before Penyisip_OpeningFcn gets called.
An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Penyisip_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Penyisip

% Last Modified by GUIDE v2.5 26-Oct-2016 11:53:45

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Penyisip_OpeningFcn, ...
                   'gui_OutputFcn',  @Penyisip_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Penyisip is made visible.
function Penyisip_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Penyisip (see VARARGIN)

% Choose default command line output for Penyisip
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Penyisip wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Penyisip_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
msgbox('Terimah kasih telah menggunakan Program FFT')
pause(3)
close();
close();


% --- Executes on button press in pushbutton3.
```

```
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Gambar Host
gambar1=imread(gambar);
gambar1=rgb2gray(gambar1);
axes(handles.axes1);
imshow(gambar1);imwrite(gambar1,'Citra Host.jpg');
setappdata(0,'gambar1',gambar1)

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Citra Penyisip
gambar2=imread(gambar);
gambar2=rgb2gray(gambar2);
axes(handles.axes2);
imshow(gambar2);
setappdata(0,'gambar2',gambar2)

% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% M=453;
% N=960;
a= getappdata(0,'gambar1');
b= getappdata(0,'gambar2');
[M N]=size(a);
% z=imresize(a,[M N]);
e=imresize(b,[M N]);
f=fft2(a);
g=fft2(e);
h= f + (0.01*g);
i=ifft2(h);

axes(handles.axes3);
imshow(uint8(i));
imwrite(uint8(i),'Watermarking FFT.bmp');
```

Lampiran 2

Program Ekstraksi gui dengan metode *FFT* pada matlab R2013a

```
function varargout = Pengekstrak(varargin)
% PENGEKSTRAK MATLAB code for Pengekstrak.fig
%      PENGEKSTRAK, by itself, creates a new PENGEKSTRAK or raises
the existing
%      singleton*.
%
%      H = PENGEKSTRAK returns the handle to a new PENGEKSTRAK or
the handle to
%      the existing singleton*.
%
%      PENGEKSTRAK('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in PENGEKSTRAK.M with the given
input arguments.
%
%      PENGEKSTRAK('Property','Value',...) creates a new
PENGEKSTRAK or raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before Pengekstrak_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Pengekstrak_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Pengekstrak

% Last Modified by GUIDE v2.5 14-Aug-2017 18:17:09

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Pengekstrak_OpeningFcn, ...
                   'gui_OutputFcn',  @Pengekstrak_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Pengekstrak is made visible.
function Pengekstrak_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Pengekstrak (see VARARGIN)

% Choose default command line output for Pengekstrak
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Pengekstrak wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Pengekstrak_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
msgbox('Terimah kasih telah menggunakan Program Ekstraksi FFT')
pause(3)
close();
close();

% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.bmp'); %Pilih Gambar Yang Sudah Terwatermark
gambar3 = imread (gambar);
axes(handles.axes6);
imshow(gambar3);
setappdata(0,'gambar3',gambar3)

% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)


O = getappdata(0,'gambar4');
p=fft2(O);
C = getappdata(0,'gambar3');
w=fft2(C);

y=(w-p)/0.01;
r=ifft2(y);
axes(handles.axes5);
imshow (uint8(r));title('Hasil Ekstraksi FFT ');
imwrite(uint8(r),'Ekstraksi FFT.bmp');


% --- Executes on button press in pushbutton10.
function pushbutton10_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton10 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Gambar asli
gambar4 = imread (gambar);
gambar4 =rgb2gray(gambar4);
axes(handles.axes4);
imshow(gambar4);
setappdata(0,'gambar4',gambar4)


% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton11 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Gambar asli
gambar4 = imread (gambar);
gambar4 =rgb2gray(gambar4);
gambar4 =imrotate(gambar4,45,'bilinear');
axes(handles.axes4);
imshow(gambar4);
setappdata(0,'gambar4',gambar4)
```

```
% --- Executes on button press in pushbutton12.
function pushbutton12_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton12 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Gambar asli
gambar4 = imread (gambar);
gambar4 =rgb2gray(gambar4);
gambar4 =imresize(gambar4,0.5);
% gambar4 =imcrop(gambar4,[80 80 453 960]);
axes(handles.axes4);
imshow(gambar4);
setappdata(0,'gambar4',gambar4)
```

Lampiran 3

Program *Watermarking* gui dengan metode *FFT-DWT* pada matlab R2013a

```
function varargout = penyisipan(varargin)
% PENYISIPAN M-file for penyisipan.fig
%      PENYISIPAN, by itself, creates a new PENYISIPAN or raises
the existing
%      singleton*.
%
%      H = PENYISIPAN returns the handle to a new PENYISIPAN or
the handle to
%      the existing singleton*.
%
%      PENYISIPAN('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in PENYISIPAN.M with the given
input arguments.
%
%      PENYISIPAN('Property','Value',...) creates a new PENYISIPAN
or raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before penyisipan_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to penyisipan_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help penyisipan

% Last Modified by GUIDE v2.5 14-Aug-2017 19:31:35

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @penyisipan_OpeningFcn, ...
                   'gui_OutputFcn',  @penyisipan_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before penyisipan is made visible.
function penyisipan_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to penyisipan (see VARARGIN)

% Choose default command line output for penyisipan
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes penyisipan wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = penyisipan_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Gambar Host
gambar1=imread(gambar);
gambar1=rgb2gray(gambar1);
axes(handles.axes1);
imshow(gambar1);
setappdata(0,'gambar1',gambar1)


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Citra Penyisip
gambar2=imread(gambar);
gambar2=rgb2gray(gambar2);
axes(handles.axes2);
imshow(gambar2);
setappdata(0,'gambar2',gambar2)


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% ALgoritma Penyisipan DWT+FFT sub level 1
a= getappdata(0,'gambar1');
[h_LL,h_HL,h_LH,h_HH]=dwt2(a,'haar');
aa=h_LL;
e=fft2(aa);

b= getappdata(0,'gambar2');
[m n]=size(a);
f=imresize(b,[m n]);
[w_LL,w_HL,w_LH,w_HH]=dwt2(f,'haar');
ff=w_LL;
g=fft2(ff);

W=e+(0.01*g);
h=ifft2(W);
i=idwt2(h,h_HL,h_LH,h_HH,'haar');

axes(handles.axes3);
imshow(uint8(i));title('Watermaking DWT-FFT');
imwrite(uint8(i),'Watermarking DWT-FFT.bmp');

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
msgbox('Terimah kasih telah menggunakan Program Watermaking DWT-
FFT')
pause(3)
close();
close();


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% ALgoritma Penyisipan DWT+FFT sub level 2
a= getappdata(0,'gambar1');
[h_LL1,h_LH1,h_HL1,h_HH1]=dwt2(a,'haar');
%aa=h_LL;
[Rh_LL1  Ch_LL1]=size(h_LL1);
[h_LL2,h_LH2,h_HL2,h_HH2]=dwt2(h_LL1,'haar');
% aaa=h_LL2;
[Rh_LL2  Ch_LL2]=size(h_LL2);
e=fft2(h_LL2);

b= getappdata(0,'gambar2');
[m n]=size(a);
f=imresize(b,[m n]);
[w_LL1,w_LH1,w_HL1,w_HH1]=dwt2(f,'haar');
% ff=w_LL;
[Rw_LL1  Cw_LL1]=size(w_LL1);
[w_LL2,w_LH2,w_HL2,w_HH2]=dwt2(w_LL1,'haar');
% fff=w_LL2;
[Rw_LL2 Cw_LL2]=size(w_LL2);
g=fft2(w_LL2);

W=e+(0.01*g);
h=ifft2(W);
i=idwt2(h,h_LH2,h_HL2,h_HH2,'haar',[Rh_LL1,Ch_LL1]);
j=idwt2(i,h_LH1,h_HL1,h_HH1,'haar');
axes(handles.axes3);
imshow(uint8(j));title('Watermaking DWT-FFT2');
imwrite(uint8(j),'Watermarking DWT-FFT2.bmp');

% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% ALgoritma Penyisipan DWT+FFT sub level 3
a= getappdata(0,'gambar1');
[h_LL1,h_LH1,h_HL1,h_HH1]=dwt2(a,'haar');
% aa=h_LL;
[Rh_LL1  Ch_LL1]=size(h_LL1);
[h_LL2,h_LH2,h_HL2,h_HH2]=dwt2(h_LL1,'haar');
% aaa=h_LL2;
[Rh_LL2  Ch_LL2]=size(h_LL2);
[h_LL3,h_LH3,h_HL3,h_HH3]=dwt2(h_LL2,'haar');
[Rh_LL3  Ch_LL3]=size(h_LL3);
e=fft2(h_LL3);

b= getappdata(0,'gambar2');
[m n]=size(a);
f=imresize(b,[m n]);
[w_LL1,w_LH1,w_HL1,w_HH1]=dwt2(f,'haar');
% ff=w_LL;
[Rw_LL1  Cw_LL1]=size(w_LL1);
```

```
[w_LL2,w_LH2,w_HL2,w_HH2]=dwt2(w_LL1,'haar');
% fff=w_LL2;
[Rw_LL2  Cw_LL2]=size(w_LL2);
[w_LL3,w_LH3,w_HL3,w_HH3]=dwt2(w_LL2,'haar');
[Rw_LL3  Cw_LL3]=size(w_LL3);
g=fft2(w_LL3);

W=e+(0.01*g);
h=ifft2(W);
i=idwt2(h,h_LH3,h_HL3,h_HH3,'haar',[Rh_LL2,Ch_LL2]);
j=idwt2(i,h_LH2,h_HL2,h_HH2,'haar',[Rh_LL1,Ch_LL1]);
k=idwt2(j,h_LH1,h_HL1,h_HH1,'haar');
axes(handles.axes3);
imshow(uint8(k));title('Watermaking DWT-FFT3');
imwrite(uint8(k),'Watermarking DWT-FFT3.bmp');
```

Lampiran 4

Program Ekstraksi gui dengan metode *FFT-DWT* pada matlab R2013a

```
function varargout = pengekstrakan(varargin)
% PENGEKSTRAKAN M-file for pengekstrakan.fig
%      PENGEKSTRAKAN, by itself, creates a new PENGEKSTRAKAN or
raises the existing
%      singleton*.
%
%      H = PENGEKSTRAKAN returns the handle to a new PENGEKSTRAKAN
or the handle to
%      the existing singleton*.
%
%      PENGEKSTRAKAN('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in PENGEKSTRAKAN.M with the given
input arguments.
%
%      PENGEKSTRAKAN('Property','Value',...) creates a new
PENGEKSTRAKAN or raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before pengekstrakan_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to pengekstrakan_OpeningFcn
via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help pengekstrakan

% Last Modified by GUIDE v2.5 27-Aug-2017 09:05:45

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @pengekstrakan_OpeningFcn,
...
                   'gui_OutputFcn',  @pengekstrakan_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
```

```
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before pengekstrakan is made visible.
function pengekstrakan_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to pengekstrakan (see
VARARGIN)

% Choose default command line output for pengekstrakan
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes pengekstrakan wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = pengekstrakan_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.bmp'); %Pilih Gambar Yang Sudah Terwatermark
gambar3 = imread (gambar);
axes(handles.axes1);
imshow(gambar3);
setappdata(0,'gambar3',gambar3)


% --- Executes on button press in pushbutton2.
```

```
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Gambar asli
gambar4 = imread (gambar);
gambar4 = rgb2gray(gambar4);
axes(handles.axes2);
imshow(gambar4);
setappdata(0,'gambar4',gambar4)

% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% ALgoritma Ekstraksi DWT+FFT sub level 1
a = getappdata(0,'gambar3');
[e_LL1,e_HL1,e_LH1,e_HH1]=dwt2(a,'haar');
[Re_LL1  Ce_LL1]=size(e_LL1);
j=e_LL1;
l=fft2(j);

c= getappdata(0,'gambar4');
[h_LL1,h_HL1,h_LH1,h_HH1]=dwt2(c,'haar');
[Rh_LL1  Ch_LL1]=size(h_LL1);
aa=h_LL1;
p=fft2(aa);

m=(l-p)/0.01;
n=ifft2(m);
o=idwt2(n,h_HL1,h_LH1,h_HH1,'haar',[Rh_LL1,Ch_LL1]);
axes(handles.axes3);
imshow (uint8(o));title('Ekstraksi ');
imwrite(uint8(o),'EkstraksiDWT-FFT1.bmp');

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
msgbox('Terimah kasih telah menggunakan Program Watermaking DWT-
FFT')
pause(3)
close();
close();


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% ALgoritma Ekstraksi DWT+FFT sub level 2
a = getappdata(0,'gambar3');
[e_LL1,e_LH1,e_HL1,e_HH1]=dwt2(a,'haar');
% j=e_LL;
[Re_LL1  Ce_LL1]=size(e_LL1);
[e_LL2,e_LH2,e_HL2,e_HH2]=dwt2(e_LL1,'haar');
% jj=e_LL2;
[Re_LL2  Ce_LL2]=size(e_LL2);
l=fft2(e_LL2);

c= getappdata(0,'gambar4');
[h_LL1,h_LH1,h_HL1,h_HH1]=dwt2(c,'haar');
% aa=h_LL;
[Rh_LL1  Ch_LL1]=size(h_LL1);
[h_LL2,h_LH2,h_HL2,h_HH2]=dwt2(h_LL1,'haar');
% aaa=h_LL2;
[Rh_LL2  Ch_LL2]=size(h_LL2);
p=fft2(h_LL2);


m=(l-p)/0.01;
n=ifft2(m);
o=idwt2(n,h_LH2,h_HL2,h_HH2,'haar',[Rh_LL1,Ch_LL1]);
p=idwt2(o,h_LH1,h_HL1,h_HH1,'haar');
axes(handles.axes3);
imshow (uint8(p));title('EkstraksiDWT-FFT2 ');
imwrite(uint8(p),'EkstraksiDWT-FFT2.bmp');

% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% ALgoritma Ekstraksi DWT+FFT sub level 3
a = getappdata(0,'gambar3');
[e_LL1,e_LH1,e_HL1,e_HH1]=dwt2(a,'haar');
% j=e_LL;
[Re_LL1  Ce_LL1]=size(e_LL1);
[e_LL2,e_LH2,e_HL2,e_HH2]=dwt2(e_LL1,'haar');
% jj=e_LL2;
[Re_LL2  Ce_LL2]=size(e_LL2);
[e_LL3,e_LH3,e_HL3,e_HH3]=dwt2(e_LL2,'haar');
% jjj=e_LL3;
[Re_LL3  Ce_LL3]=size(e_LL3);
l=fft2(e_LL3);

c= getappdata(0,'gambar4');
[h_LL1,h_LH1,h_HL1,h_HH1]=dwt2(c,'haar');
% aa=h_LL;
[Rh_LL1  Ch_LL1]=size(h_LL1);
[h_LL2,h_LH2,h_HL2,h_HH2]=dwt2(h_LL1,'haar');
```

```
% aaa=h_LL2;
[Rh_LL2  Ch_LL2]=size(h_LL2);
[h_LL3,h_LH3,h_HL3,h_HH3]=dwt2(h_LL2,'haar');
% aaaa=h_LL3;
[Rh_LL3  Ch_LL3]=size(h_LL3);
p=fft2(h_LL3);

m=(1-p)/0.01;
n=ifft2(m);
o=idwt2(n,h_LH3,h_HL3,h_HH3,'haar',[Rh_LL2,Ch_LL2]);
p=idwt2(o,h_LH2,h_HL2,h_HH2,'haar',[Rh_LL1,Ch_LL1]);
q=idwt2(p,h_LH1,h_HL1,h_HH1,'haar');

axes(handles.axes3);
imshow (uint8(q));title('Ekstraksi ');
imwrite(uint8(q),'EkstraksiDWT-FFT3.bmp');


% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Gambar asli
gambar4 = imread (gambar);
gambar4 =rgb2gray(gambar4);
gambar4 =imrotate(gambar4,45,'bilinear');
axes(handles.axes2);
imshow(gambar4);
setappdata(0,'gambar4',gambar4)

% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
gambar = uigetfile('*.jpg'); %Pilih Gambar asli
gambar4 = imread (gambar);
gambar4 =rgb2gray(gambar4);
gambar4=imresize(gambar4,0.5);
% gambar4 =imcrop(gambar4,[25 25 960 453]);
axes(handles.axes2);
imshow(gambar4);
setappdata(0,'gambar4',gambar4)
```