DAFTAR SINGKATAN

BER	*Bit Error Rate*
CER	*Character Error Rate*
SNR	*Signal to Noise Rasio*
FFT	*Fast Fourier Tansform*
DWT	*Discrete Wavelet Transform*
DCT	*Discrete Cosine Transform*
LPF	*Low-pass Filter*
MOS	*Mean Opinion Score*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi dan internet yang sangat pesat elah berikan banyak manfaat, salah satunya yaitu dengan kemudahan yang didapat dalam mengakses berbagai informasi dalam format digital, berupa gambar, audio, dan video. Seharusnya hal tersebut memiliki banyak nilai positif, tetapi ada saja beberapa oknum yang memanfaatkan hal tersebut untuk melakukan hal melanggar hukum yaitu berupa duplikasi atau sering disebut pembajakan hak cipta (*copyright piracy*), tentu hal tersebut merugikan bagi pemilik hak cipta seperti lagu atau musik yang dapat dengan mudah digandakan oleh oknum yang tidak bertanggung jawab, maka dari itu dibutuhkan suatu teknik untuk melindungi karya dari hak cipta tersebut, dari ketiga media berupa gambar, audio maupun video, riset ini fokus terhadap media audio atau suara.

Audio Watermarking merupakan teknik yang akan digunakan dalam perlindungan hak cipta (*copyright piracy*) dalam media suara tersebut, dengan proses penyisipan data informasi ke *host file*, yang bertujuan untuk membuktikan atau menandai suatu kepemilikan, *watermarking* dilakukan sebaik mungkin sehingga informasi yang disisipkan tidak merusak *host file* atau karya yang ingin disisipkan tersebut, dan juga tidak dapat diketahui oleh indera pendengaran manusia pada umumnya.

Riset ini akan menguji suatu ketahanan *audio watermarking* pada pembajakan data berupa musik atau lagu, dengan cara disisipkan suatu informasi (teks) menggunakan *software* MATLAB denngan metode *frequency masking* di mana penyisipan setiap *frame* berada di domain frekuensi, disimpan dengan format awal ".mp3" yang kemudian diubah atau *convert* menjadi ".wav" begitu pula dari ".wav" menjadi ".wav", setelah itu hasil format audio tersebut akan dilihat hasil analisanya dari dua poin, yang pertama yaitu perbandingan dari suara sebelum dan sesudah proses *watermarking*, yang kedua yaitu analisa dari data BER, CER, dan data yang disisipkan apakah masih utuh atau tidak.

1.2 Tujuan

Tujuan penelitian ini adalah :

1) dapat merancang penyisipan data dengan metode *frequency masking* pada sistem *audio watermarking*.
2) menguji kualitas suara dan kualitas *audio watermarking* setelah diubah format *file audio*.
3) menganalisa ketahanan data yang disisipkan.
4) menganalisa nilai BER dan CER.

1.3 Rumusan Masalah

Berdasarkan deskripsi latar belakang dan penelitian terkait, maka dapat dirumuskan beberapa masalah di tugas akhir ini, yaitu :

1) Bagaimana implementasi metode *frequency masking* dalam proses *embedding* dan proses *extraction*, untuk mendapatkan informasi data yang disisipkan.
2) Bagaimana kinerja dari sistem *audio watermarking* yang dirancang.
3) Bagaimana ketahanan, dan tingkat akurasi *audio watermarking* yang dirancang.
4) Bagaimana hasil BER dan CER.

1.4 Batasan Masalah

Beberapa asumsi dan batasan masalah pada tugas akhir ini, yaitu :

1) Perancangan sistem dilakukan dengan menggunakan aplikasi MATLAB.
2) Jumlah *file audio* yang digunakan ada 5 *host file*.
3) Data yang disisipkan berupa teks.
4) Metode *frequency masking* digunakan pada proses *embedding* dan ekstraksi.
5) Proses mengubah format audio menggunakan “.mp3”, dan “.wav”.
6) Analisa pertama pada perubahan hasil suara sebelum dan sesudah diubah formatnya, kedua pada ketahanan data setelah di *watermark*.

1.5 Penelitian Terkait

Pada penelitian [13] dilakukan analisis ketahanan di domain frekuensi pada ambient mode dengan menggunakan metode frequency masking. Simulasi menunjukkan bahwa menggunakan metode frequency masking pada uji ketahanan rekaman suara atau ambient mode, belum mencapai hasil yang maksimal, hal tersebut karena nilai BER minimum pada pengujian ambient yang didapatkan yaitu 0.4375 dan pada beberapa

pengujian ada yang tidak berhasil didapatkan kembali informasi awal yang disisipkan. Tingkat BER dan CER minimum dari *file audio watermarking* sudah mencapai nilai yang sempurna atau sangat baik, yaitu 0 sebagai syarat *quality audio watermarking* (tidak ada informasi yang disisipkan hilang saat diekstraksi kembali) dan syarat *audibility* yang baik (efek *watermark* tidak terasa pada pendengaran).

1.6 Metode penelitian

Metodologi dalam proses penyelesaian penelitian ini terdiri dari beberapa tahapan, yaitu:

1. Tahap identifikasi dasar penelitian

   Pada tahap ini dilakukan identifikasi dasar-dasar penelitian, mempelajari konsep dan teori tentang *audio watermarking* dan metode *frequency masking* yang dapat membantu proses perancangan.
2. Tahap analisa masalah

   Menganalisa permasalahan yang akan didapat berdasarkan data yang digunakan dan didiskusikan dengan pembimbing untuk selanjutnya mencari solusi dari masalah yang ditemukan.
3. Tahap perancangan sistem

   Membuat perancangan sistem *audio watermarking* dengan metode *frequency masking* yang sebelumnya sudah didiskusikan dan direncanakan dengan pembimbing sebelum menuju tahap implementasi.
4. Tahap implementasi rancangan sistem

   Pada tahap ini, dilakukan implementasi dari rancangan sistem yang sudah didiskusikan dengan dosen pembimbing di tahap sebelumnya, meliputi proses *embedding* (menyisipkan data teks ke dalam *file audio*).
5. Tahap pengujian ketahanan

   Setelah dilakukan implementasi sistem, dilakukan pengujian terhadap *file audio* yang sudah disisipkan data teks atau di*watermark*, yaitu dengan mengubah format *file audio*, yang kemudian nanti diekstraksi kembali data teks yang disisipkan.
6. Tahap analisa hasil

   Pada tahap ini, dilakukan analisa dari hasil yang didapatkan setelah *file audio* diekstrak, parameter-parameter yang dilihat yaitu berupa kualitas data teks yang diterima, berikut nilai BER, dan CER.

7. Tahap membuat kesimpulan

   Tahap akhir dari semua tahap-tahap sebelumnya didapatkan hasil keseluruhan, maka ditarik kesimpulan terhadap tugas akhir yang dilakukan.

1.7 Sistematika Penulisan

Berikut adalah sistematika penulisan pada tugas akhir ini :

BAB I : PENDAHULUAN

Bab ini berisikan tentang latar belakang, tujuan, rumusan masalah, batasan masalah, dan metode penelitian yang digunakan pada tugas akhir ini.

BAB II : DASAR TEORI

Bab ini menjelaskan mengenai teori-teori yang mendukung dan mendasari pengerjaan tugas akhir ini, yaitu teori dasar mengenai *watermarking*,proses *watermarking*, sinyal audio, FFT, dan *frequency masking*.

BAB III : PERANCANGAN DAN IMPLEMENTASI SISTEM

Bab ini berisi pembahasan tentang langkah-langkah perancangan kerja sistem, *embedding*, *extraction*, serta pengujian secara subjektif dan objektif.

BAB IV : PENGUJIAN SISTEM DAN ANALISIS

Bab ini berisi pembahasan dari hasil pengujian dan analisis yang telah dilakukan, implementasi, dan aplikasi untuk model penyisipan dan pemisahan data informasi.

BAB V : KESIMPULAN DAN SARAN

Bab ini berisikan kesimpulan yang dapat diambil dari pengujian sistem dan analisis yang telah dibahas sebelumnya dan saran-saran yang dapat memperbaiki tugas akhir ini untuk penelitian selanjutnya.

BAB II
DASAR TEORI

2.1 Watermarking

Watermark merupakan suatu teknik yang menyerupai *steganography* yaitu ilmu yang mempelajari tentang penyembunyian suatu data pada data lain, dan pada *digital watermarking* data yang disisipkan ke dalam suatu media berupa sinyal digital, tujuan utama dari sistem *watermarking* yaitu untuk mencapai ketahanan terhadap serangan yang bersifat merusak atau menghilangkan informasi yang disisipkan, khusus untuk perlindungan hak cipta, berbeda dengan tujuan dari *steganography* yang justru menyisipkan informasi sebanyak mungkin dan untuk mengkomunikasikan informasi digital, dan juga tidak harus memiliki ketahanan terhadap serangan yang bersifat merusak, oleh karena itu jika tujuan utama untuk perlindungan data digital, metode *digital watermarking* lebih layak digunakan, ada beberapa syarat karakteristik yang harus dipenuhi untuk dapat mengoptimalkan system *digital watermarking*, berikut sedikit dari banyak karakteristik yang harus dipenuhi, yaitu [1] :

1) *Percetual transparancy* : kebutuhan utama dari *watermarking*, informasi yang telah disisipkan seharusnya tidak menurunkan kualitas dari *host-data* yang disisipkan, tidak boleh terdengar oleh telinga atau terlihat oleh mata (*imperceptible*).
2) *Robustnes* (ketahanan) : informasi yang disisipkan harus mempunyai ketahanan terhadap beberapa jenis serangan pada *watermark*, dan informasi tersebut harus dapat diterima kembali di *decoder* dengan benar.
3) *Security* (keamanan) : informasi yang disisipkan sulit untuk dilepaskan dari *host-data* bahkan setelah mengalami berbagai macam serangan.
4) *Data rate* / *Payload* / *Bit rate* : *watermark* harus mampu memverifikasi dan menyediakan bukti yang terpercaya untuk membuktikan suatu produk [2].

dan memiliki tujuan penggunaan yang paling sering dilakukan antara lain :

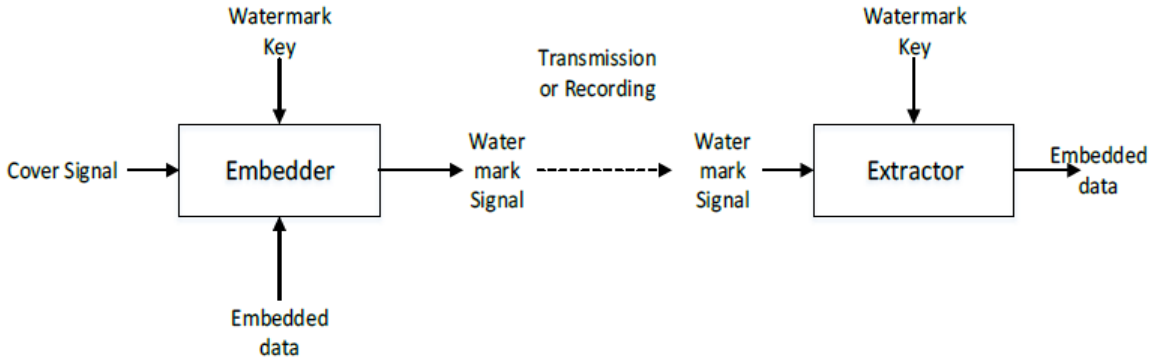
1) *Copyright protection* / *proof of ownership* : untuk membuktikan hak cipta sesuatu.

2) *Tamper detection* : untuk mendeteksi apakah perusakan pada data asli (pemalsuan).
3) *Copy protection* : untuk mengontrol hak peng-*copy* an suatu data dan mencgah pembajakan ilegal.
4) *Finger printing* : untuk melacak file yang di-*copy* secara ilegal.
5) *Broadcast monitoring* : untuk memantau atau memonitoring suatu sinyal yang di *broadcast*.
6) *Information carrier* : *watermark* digunakan sebagai pembawa informasi.

2.1.1 Proses *watermarking*

Saat melakukan *watermarking* terdiri dari 2 jenis proses, yaitu :

1) *Embedding process* : proses penyisipan data informasi.
2) *Extraction process* : proses ekstraksi atau pengambilan data informasi yang telah disisipkan pada saat embedding.



Gambar 2.1 Proses *Watermarking* [3]

Pada gambar 2.1 di atas bisa dilihat dua proses yang dilakukan pada *watermarking*, pada proses *embedder* (penyisipan) terdapat 3 inti, yaitu *embedded data* sebagai informasi yang akan disisipkan, *cover signal* sebagai *host-data* yang akan disisipkan informasi, dan *watermark key* yang berfungsi sebagia kunci parameter utama pada algoritma penyembunyian data untuk menyembunyikan informasi di *cover signal*.

Setelah proses *embedder* dilakukan, sebuah keluaran berupa sinyal yang telah disisipi informasi menggunakan *watermark key*, atau biasa disebut juga sebagai *watermark signal*. *Watermark signal* ini kemudian ditransmisikan sehingga sampai pada penerima, pada penerima akan dilakukan proses ekstraksi atau pengambilan data dengan menggunakan *watermark key* untuk pengambilan data sehingga didapatkan kembali informasi yang telah disisipkan tersebut. Informasi yang dihasilkan dari proses ekstraksi

ini bergantung pada karakteristik *watermarking* yang sebelumnya sudah dijelaskan agar mendapatkan hasil informasi yang sesuatu dengan informasi yang dikirimkan [3].

2.1.2 Metode Audio Watermarking

Metode *audio watermarking* terbagi menjadi dua domain, yaitu :

1) Domain waktu

Metode ini bekerja dengan cara mengubah data audio dalam domain waktu yang akan disisipkan *watermark*, contohnya dengan mengubah LSB (*Least Significant Bit*) dari data tersebut, secara umum metode ini rentan terhadap proses kompresi, transmisi dan *encoding*. Metode yang termasuk dalam teknik algoritma ini yaitu :

a) *Compressed-domain watermarking* : pada teknik ini hanya representasi data yang terkompresi yang diberi *watermark*, saat data di *uncompressed* maka *watermark* tidak lagi tersedia.

b) *Bit dithering* : *watermark* disisipkan pada tiap LSB, baik pada representasi data terkompresi atau tidak, teknik ini membuat derau pada sinyal.

c) Modulasi amplitudo : cara ini membuat setiap puncak sinyal dimodifikasi agar jatuh ke dalam pita-pita amplitudo yang telah ditentukan.

d) *Echo hiding* : dalam metode ini salinan-salinan terputus-putus dari sinyal dicampur sinyal asli dengan rentang waktu yang cukup kecil, rentang waktu ini cukup kecil sehingga amplitudo salinannya cukup kecil sehingga tidak terdengar.

2) Domain frekuensi

Metode ini bekerja dengan cara mengubah kontek spektral dalam domain frekuensi dari sinyal seperti membuang komponen frekuensi tertentu atau menambahkan data sebagai derau dengan amplitudo rendah sehingga tidak terdengar, metode yang termasuk dalam teknik ini yaitu :

a) *Phase coding* : bekerja berdasarkan karakteristik sistem pendengaran manusia (*Human Auditory System*) yang mengabaikan suara yang lebih lemah jika dua suara itu datang bersamaan, secara garis besar data *watermark* dibuat menjadi derau dengan amplitudo

yang lebih lemah dibandingkan amplitudo data audio lalu digabungkan.

b) *Frequency masking* : metode ini memanfaatkan kelemahan pendengaran manusia yang tidak dapat mendengar pada frekuensi tertentu.

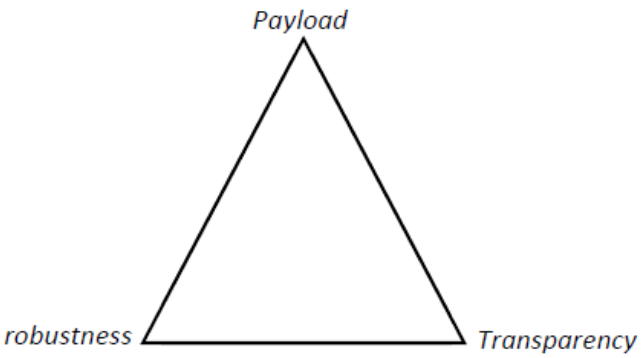
c) Modifikasi pita frekuensi : informasi *watermark* ditambahkan dengan cara membuang atau menyisipkan ke dalam pita-pita (*band*) spektral tertentu.

d) Penyebaran spektrum : metode ini diadopsi dari teknik penyebaran spektrum dalam telekomunikasi.

2.1.3 Trade-off pada watermarking [4]

Berdasarkan pada parameter-parameter karakteristik yang sudah disebutkan sebelumnya, terdapat suatu *trade-off* antara masing-masing karakteristik, akan tetapi *trade-off* terbesar terdapat pada parameter *payload* dan *robustness* yang biasa dikenal *transparency*. Ketika diinginkan *robustness* yang tinggi, maka jumlah data yang disisipkan juga akan mengurang dan data yang akan disisipkan akan semakin tidak transparan. Ketika diinginkan transparansi yang tinggi, maka *robustness* juga akan semakin mengurang termasuk jumlah data yang disisipkan. Oleh karena itu pada penentuan antara parameter tersebut, sangat perlu diperhatikan nilai-nilai yang akan diperlukan agar diinginkan hasil yang cukup memuaskan.

Penentuan *trade-off* tersebut dapat digambarkan seperti gambar berikut ini :



Gambar 2.2 *Trade-off watermarking* [4]

2.2 Sinyal audio [4]

Sinyal audio adalah gelombang suara yang memiliki bentuk gelombang longitudinal, dan dikategorikan dalam sinyal analog, sedangkan komputer tidak dapat

memproses sinyal analog. Komputer hanya mampu mengenal sinyal *digital*, oleh karena itu diperlukan sebuah proses untuk merubah sinyal audio menjadi *digital* agar dapat diproses komputer.

Sinyal audio terdiri dari *telephone quality speech*, *wideband speech*, dan *wideband audio*. Rentang frekuensi untuk setiap sinyal audio ini adalah :

1) 30 Hz – 3400 Hz untuk *telephone quality speech*
2) 50 Hz – 7000 Hz untuk *wideband speech*
3) 20 Hz – 20000 Hz untuk *wideband audio* dengan kualitas yang tinggi

Pendengaran yang dimiliki oleh manusia dapat mendeteksi atau mendengar suara dengan frekuensi yang bervariasi antara 20 Hz – 20000 Hz, suara ini disebut dengan *audiosonic* atau dikenal dengan audio, gelombang suara pada batas frekuensi tersebut disebut dengan sinyal akustik. Akustik merupakan cabang fisika yang mempelajari bunyi. Level tekanan suara (volume suara) dihitung dalan *desibel* (dB), yaitu perhitungan rasio antara titik referensi yang dipilih dalam skala logaritmik dan level yang benar-benar dialami. Keras lemahnya bunyi atau tinggi rendahnya gelombang disebut dengan amplitudo. Bunyi mulai dapat merusak telinga jika tingkat volumenya lebih besar dari 85 dB dan pada ukuran 130 dB akan mampu membuat hancur gendang telinga.

Frekuensi adalah banyaknya gelombang yang mempunyai pola yang sama yang berulang pada interval tertentu selama 1 detik. Berdasarkan frekuensi, suara atau bunyi dibagi atas :

1) *Infrasound* yaitu suara pada rentang frekuensi 0 Hz – 2 Hz
2) *Audiosound* yaitu suara pada rentang frekuensi 20 Hz – 20 kHz
3) *Ultrasound* yaitu suara pada rentang frekuensi 20 kHz – 1 GHz
4) *Hypersound* yaitu suara pada rentang frekuensi 1 GHz – 10 THz

2.3 FFT (*Fast Fourier Transform*) [5]

Pada dasarnya fungsi dari FFT itu sendiri yaitu merubah sinyal dari domain waktu ke domain frekuensi, untuk mengoperasikan *fast fourier transform*, dibutuhkan transformasi *fourier aperiodic* dalam waktu diskrit. Persamaan tersebut digunakan dalam DFT dan IDFT. Karena merupakan penyempurnaan dari FFT, maka *property* tersebut tetap akan dimiliki. Formula pada DFT dan IDFT didefinisikan sebagai :

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \text{ dengan k} = 0,1,2,3, \text{N} - 1 \qquad (2.1)$$

$$x(\mathrm{n}) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)W_N^{-kn} \text{ dengan } n = 0.1, \dots, N \quad (2.2)$$

$$W_n = e^{-j2\pi/N} \quad (2.3)$$

Terdapat banyak algoritma FFT yang dapat digunakan. Salah satu pendekatannya adalah dengan menggunakan *Divide and Conquer radix* 2, yang artinya membagi tiap proses menjadi dua bagian, untuk dapat menggunakan FFT, perlu diketahui dahulu prinsip kerja DFT karena FFT menggunakan DFT dalam pengomputasiannya. Untuk mendapatkan hasil yang lebih optimal, penggunaan *direct computing* mulai dialihkan ke penggunaan algoritma *divide and conquer*.

Misalnya, terdapat N buah komputasi DFT, komputasi tersebut dapat dianggap sebagai hasil dari perkalian dua buah bilangan, L dan M, maka N = LM. Sejumlah data tersebut dapat dipresentasikan sebagai N kolom array 1 dimensi dengan panjang L, yang isinya x(i).

Tabel 2.1 Tabel x(i) [5]

x(0)	x(1)	x(2)	x(3)	x(4)	x(5)	x(6)	x(N-1)

Kemudian, data tersebut dipindahkan ke dalam matrix berukuran LM, seperti berikut :

Tabel 2.2 Tabel Matrix LM [5]

m / 1	0	1	M-1
0	x(0,0)	x(0,1)	...
1	...	...	...
N-1	...	...	...

Jika ingin mengambil index yang bersesuaian dengan index pada array 1 dimensi, maka digunakan rumus dibawah ini :

$n = Ml + m$ untuk memilih berdasarkan kolom dan $n = l + mL$ untuk memilih berdasarkan baris, namun keduanya sebenarnya sama saja. Jika data tetap dipetakan pada matrix seperti diatas, maka untuk pengomputasian dapat digunakan persamaan berikut :

$$X(p,q) = \sum_{m=0}^{M-1}\sum_{l=0}^{L-1} x(l,m)W_n^{(Mp+q)(mL+l)} \quad (2.4)$$

Langkah pengomputasiannya adalah sebagai berikut :

1) Komputasikan sejumlah M DFT dengan persamaan

$$F(l,q) = \sum_{n=0}^{M-1} x(l,m) W_M^{mq} \text{ dengan } 0 \leq q \leq M-1 \quad (2.5)$$

2) Kemudian, komputasikan matriks baru G(1,q) yang merupakan hasil perkalian antara F(1,q) dengan W_N^{lq}, dan dapat dirumuskan menjadi

$$G(l,q) = W_N^{lq} F(l,q) \quad (2.6)$$

3) Kemudian kalkulasikan N DFT

$$X(p,q) = \sum_{l=0}^{L-1} G(l,q) W_L^{lp} \quad (2.7)$$

Dari sekian banyak data yang dihasilkan pada tahap DFT diatas, kita bisa memilih frekuensi mana yang akan kita pertukarkan nilainya, yang dalam hal ini nilai dalam bit yang merepresentasikan suara pada frekuensi F.

Pada metode penghitungan DFT dilakukan dengan direct computation. Idenya adalah dengan mengubah komponen kompleks kedalam persamaan trigonometri dibawah ini :

$$X_R(k) = \sum_{n=0}^{N-1} [x_R(n) \cos \frac{2\pi kn}{N} + x_1(n) \sin \frac{2\pi kn}{N} \quad (2.8)$$

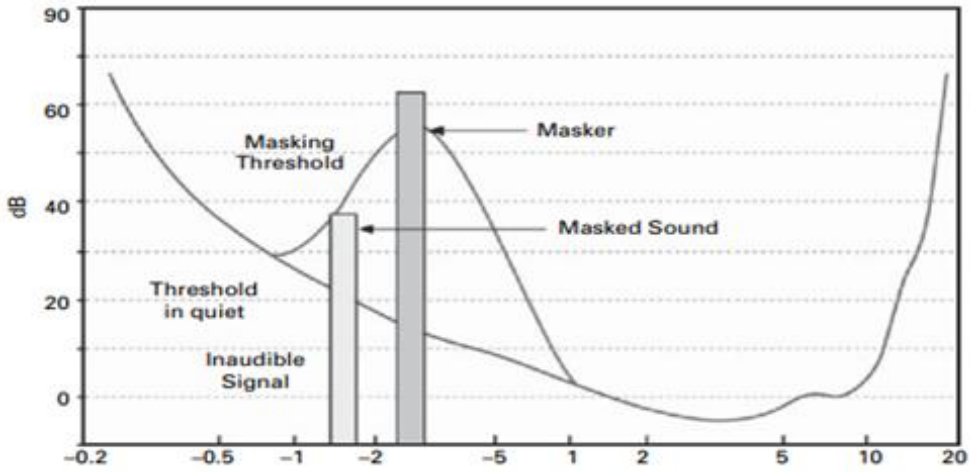
$$X_l(k) = -\sum_{n=0}^{N-1} [x_R(n) \sin \frac{2\pi kn}{N} + x_1(n) \cos \frac{2\pi kn}{N} \quad (2.9)$$

2.4 *Frequency Masking* [6]

Berdasarkan domain penyisipannya, teknik *watermarking audio* dapat dibagi menjadi dua kelompok, yaitu *temporal watermarking* dan *spectral watermarking*. *Temporal watermarking* melakukan penyisipan data pada *audio host* dalam domain waktu, sedangkan *spectral watermarking* terlebih dulu melakukan proses transformasi dari domain waktu ke dalam domain frekuensi, sehingga penyisipannya dilakukan pada elemen-elemen frekuensi.

Spectral watermarking melibatkan proses transformasi frekuensi seperti DWT (*Discrete Wavelet Transform*) atau DCT (*Discrete Cosine Transform*) kepada *audio host* untuk memperoleh komponen *audio* dalam domain frekuensi, kemudian menyisipkan sinyal *watermark* ke dalam *audio host* tersebut, dan selanjutnya melakukan invers transformasi frekuensi untuk mendapatkan *audio* yang telah diberi *watermark* (*audio watermarked*). Proses *spectral watermarking* dapat digambarkan sebagai berikut.

Frequency masking merupakan metode yang memanfaatkan kelamahan pendengaran manusia yang tidak dapat mendengar pada frekuensi tertentu. *Masking model* yang digunakan adalah model yang didefinisikan pada ISOMPEG *Audio Psychoacoustic Model*. Fenomena frekuensi domain dimana sinyal *low level* dapat dibuat tidak terdengar merupakan masking terbesar yang dalam *critical band* dan *masking* ini efektif untuk sebuah *lesser degree* dalam *neighboring bands* [7].



Gambar 2.3 *Frequency masking method* pada suatu sinyal audio [7]

BAB III
PEMODELAN SISTEM

Hasil dari sistem yang akan dibuat yaitu algoritma yang dapat menyisipkan informasi ke dalam suatu *file audio* dengan menggunakan *audio watermarking signal*. *File audio* yang sudah di *watermark* menggunakan metode *frequency masking* ini kemudian di uji ketahanannya dengan mengubah format audio menggunakan *online-convert*, lalu dibandingkan hasil sebelum diubah format dengan yang sudah diubah formatnya (apakah ada perbedaan yang jelas pada suara atau tidak),setelah itu diambil kembali data informasi yang sudah disisipkan sebelumnya, dan juga di cek nilai BER dan CER.

3.1 Kebutuhan Perangkat

3.1.1 Spesifikasi perangkat keras

Spesifikasi perangkat keras yang diperlukan dalam membuat sistem *audio watermarking* ini antara lain :

a) 1 buah PC dengan spesifikasi :
   - Processor Intel Core i5
   - RAM 4 Gb
   - Harddisk 500 Gb

b) 1 buah speaker stereo

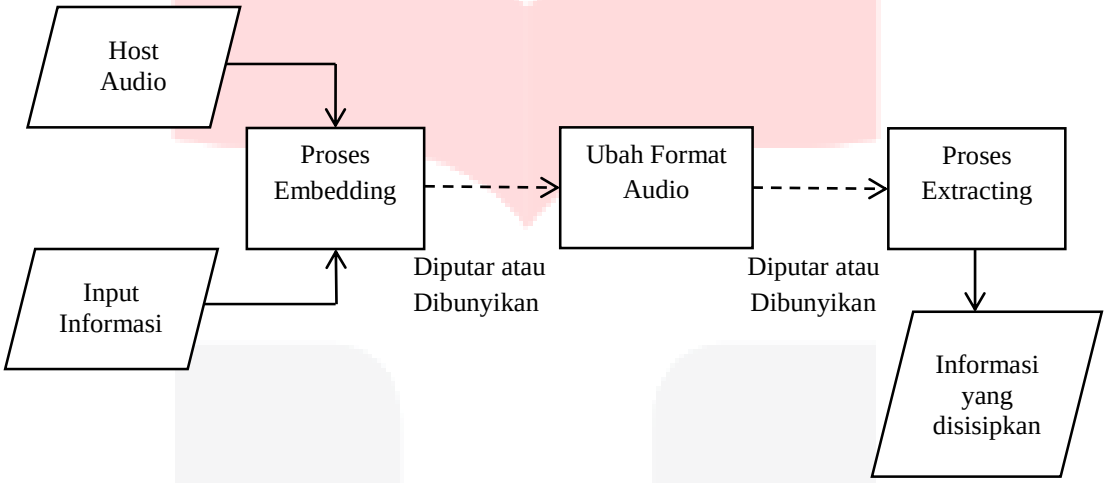
3.1.2 Spesifikasi perangkat lunak

Spesifikasi perangkat lunak yang digunakan untuk membuat sistem *audio watermarking* ini antara lain :

a) Sistem Operasi Microsoft Windows 7 sebagai *platform* untuk membuat sistem *audio watermarking*.
b) *Software* MATLAB 2014 digunakan untuk merancang dan mengimplementasikan keseluruhan sistem *watermark.*
c) VLC sebagai alat untuk memutar audio.
d) "*online-convert*" sebagai pengubah format audio secara online.
e) Microsoft Office 2013 yang digunakan untuk mengolah dan menganalisa data, serta pembuatan laporan.

3.2 Deskripsi perancangan kerja

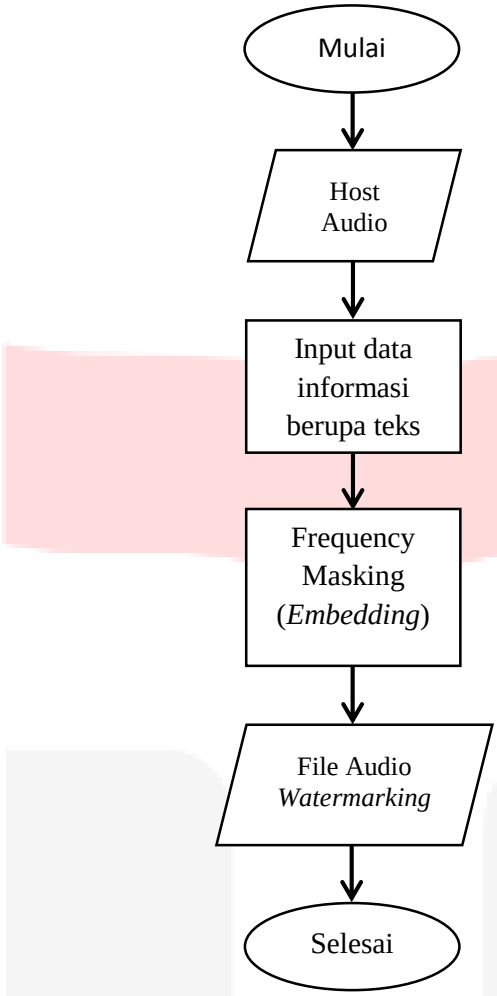
Perancangan *audio watermarking* ini memiliki dua tahap yang penting, yaitu *embedding* (proses penyisipan) yang menggunakan metode *frequency masking*, data yang disisipkan berupa *text* pada audio sehingga audio tersebut terdapat informasi di dalamnya dan *extracting* (proses pengambilan informasi) dimana audio yang telah disisipkan data tadi dipisahkan kembali antara suara asli dengan informasi yang disisipkan sebelumnya. Tahapan-tahapan yang dilalui secara umum dapat dilihat pada blok diagram yang terlihat pada gambar 3.1 :



Gambar 3.1 Skenario pengujian sistem

Pemilihan *host* yang akan digunakan pada proses *watermarking* yaitu file audio, yang nantinya akan dilakukan proses penyisipan informasi (*embedding*), setelah mendapatkan audio yang sudah diberikan *watermark*, lalu akan dilakukan pengecekan nilai SNR nya, dan dibunyikan menggunakan VLC *media player* yang didengarkan melalui *speaker*, setelah itu diubah format audio ke format yang lain dengan menggunakan *online-convert*, dari ".mp3" menjadi ".wav" begitu pula ".wav" menjadi ".mp3". jika semua uji ketahanan sudah dilakukan, proses berikutnya yaitu pemisahan (*extraction*) untuk mendapatkan kembali data informasi yang tadi disisipkan dan diubah format beserta pengecekan nilai BER dan CER nya.

3.2.1 Proses penyisipan (*Embedding*)



Gambar 3.2 Diagram alur proses *Embedding*

a) Host Audio : file audio yang digunakan berdurasi 10 detik, dengan format ".wav".
b) Input data : data informasi yang digunakan berupa teks.
c) *Embedding* : metode *frequency masking* digunakan, dengan mengubah teknik penyisipan dari domain waktu menjadi domain frekuensi menggunakan FFT.
d) File audio *watermarking* : yaitu file yang sudah di-*watermark.*

Pada tahap penyisipan (*embedding*) menggunakan metode *frequency masking* memiliki diagram alur seperti di atas, dalam proses *frequency masking* sinyal audio asli yang berdurasi 10 detik akan di-*framing* terlebih dahulu menjadi 500 *frame* dengan panjang tiap *frame* berdurasi 20 ms, tiap *frame* berisikan data sebanyak 885 bit yang akan diproses dengan FFT untuk diubah dari domain waktu ke domain frekuensi

sebanyak jumlah *frame* dan sebanyak data tiap *frame*, lalu informasi yang disisipkan berupa teks yang sudah dikonversi menjadi biner, tiap bit informasi akan disisipkan secara berurutan ke tiap *frame.* Namun sebelum proses tersebut, dari tiap *frame* akan dicari letak data yang memiliki nilai respon magnitude paling maksimum, dan setelah data dari nilai respon magnitude yang paling maksimum tersebut, akan digantikan dengan bit informasi yang akan disisipkan, lalu dikalikan dengan variabel alpha. Proses tersebut berlaku untuk nilai bit 1 pada informasi yang disisipkan. Untuk nilai bit 0 pada informasi yang disisipkan, maka hanya akan dilakukan proses FFT saja pada *frame* tertentu yang akan menggantikan bit 0 pada informasi yang disisipkan (*watermark signal)*, dalam proses FFT dapat dirumuskan sebagai berikut :

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{-j2\pi km/N} \tag{3.1}$$

setelah memilih *host audio*, dilakukan proses pemasukan teks informasi yang berfungsi utama pada proses *embedding* ini. Setelah itu dilakukan proses *frequency masking* dan penghitungan BER (*Bit Error Rate*) pada *file audio* yang sudah disisipkan informasi atau disebut juga sebagai *watermark signal*.

Dalam proses *watermark*,dapat dirumuskan sebagai berikut [7]

$$y(n) = x(n) + \propto w(n) \tag{3.2}$$

Dimana *y*(*n*) adalah sinyal audio yang sudah diberi *watermark*, $x(n)$ sinyal asli dan $\propto w(n)$ adalah sinyal watermark itu sendiri. Untuk menampilkan pendeteksian *watermark* dengan *correlation*, hasil akhir nilai *d* akan menjadi :

$$d = dx + dw$$

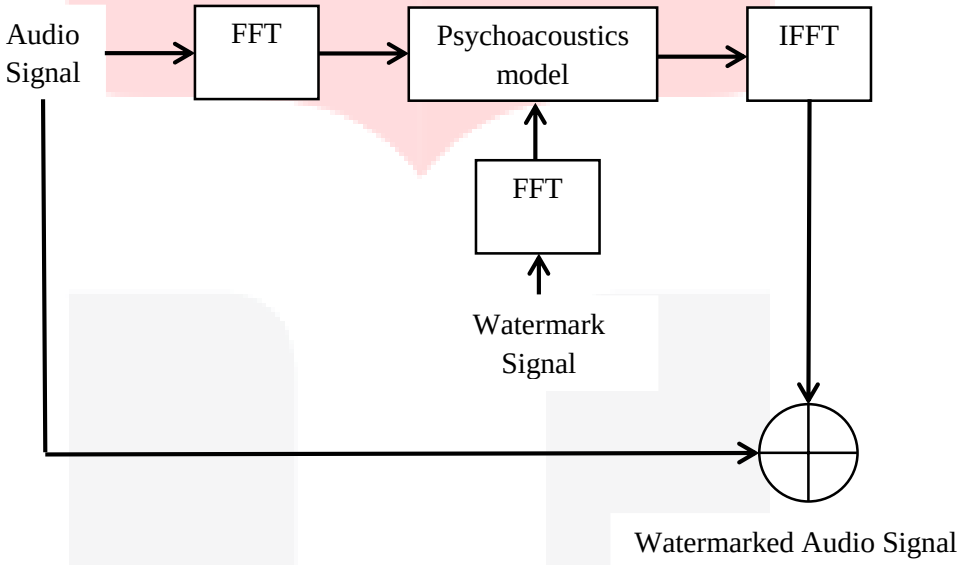
$$= \frac{1}{N}\sum_i x(i)w(i) + \frac{1}{N}\sum_i W^2(i) \tag{3.3}$$

Dari (3.3) jelas bahwa d_w adalah energi dari pada *watermark w*(*n*) dan nilai *dx* yang diharapkan adalah 0. Berdasarkan teorema central limit, jika N cukup besar dan jika kontribusi dari jumlahnya independen, *dx* merupakan distribusi *Gaussian*. Jika ambang untuk deteksi *watermark* diset T = $\frac{dx}{2}$, probabilitas dari *false negative* P- (*watermark ada*, tapi *detector* menjawab tidak ada) akan sama dengan probabilitas *false positive* P+ (*watermark* tidak ada tapi *detector* menjawab ada) [7]:

$$P_+ = P_- = \frac{1}{2} er\ f\ c\left(\frac{dw}{\sqrt{8\sigma_x}}\right) \tag{3.4}$$

Dimana *erfc*() adalah fungsi *complementary error*, dari (3.4) kita akan menggambar kondisi preliminary berikut untuk mengurangi probabilitas *false*.

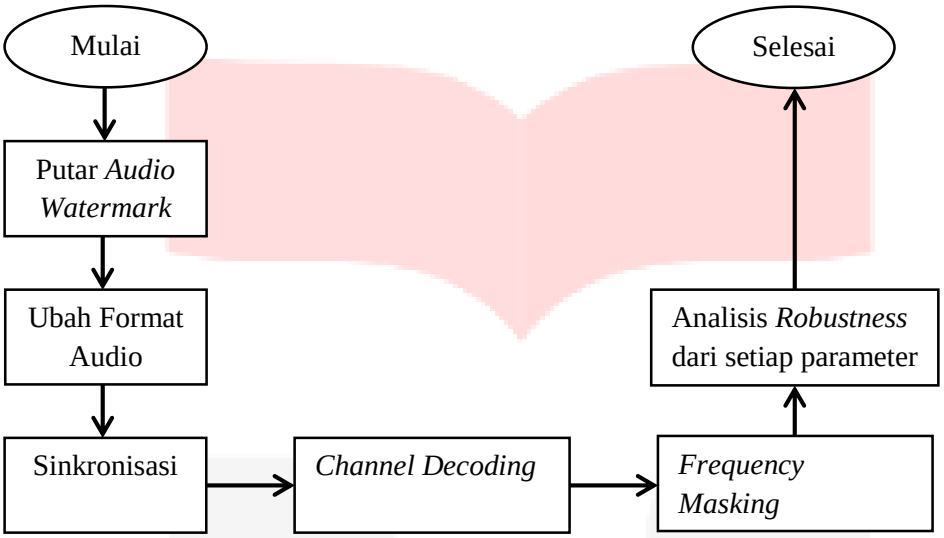
1) Probabilitas *false positive* dan *negative* menurun ketika kekuatan *watermark* d_w yang akan disisipkan meningkat. Kekuatan ini terbatas dan tergantung pada perceptual karakteristik dari sinyal audio.
2) Probabilitas *false positive* dan *negative* menurun ketika standar deviasi meningkat, ini dapat diambil dengan mengaplikasikan *whitening* atau *decorrelation* sebelum *correlaton* dalam sinyal. Proses penyisipan dan pendeteksian *watermark* tersebut dapat dilihat prosesnya dalam gambar 3.3 :



Gambar 3.3 Proses *watermarking* dengan metode *frequency masking* [8]

3.2.2 Proses pemisahan (*Extraction*)

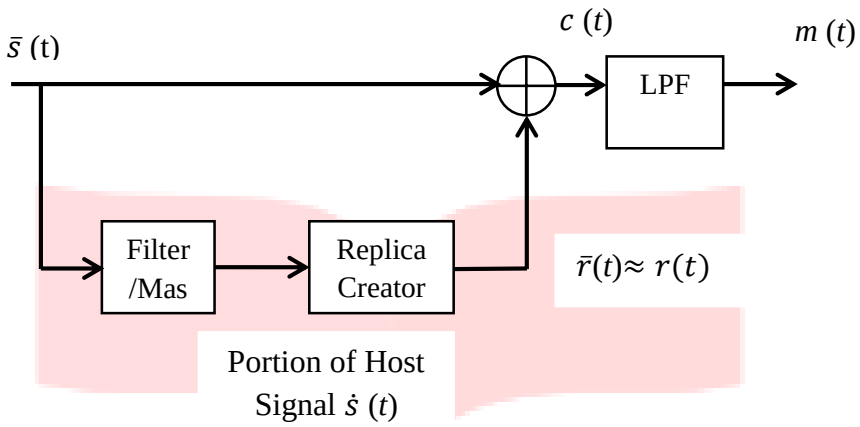
Pada *extracting* atau pengambilan data ini, audio yang berhasil disisipkan informasi atau disebut sinyal *watermark* akan diputar dan didengarkan. Setelah itu diubah formatnya dengan menggunakan *online-convert* dari ".mp3" menjadi ".wav" dan dari ".wav" menjadi ".mp3". Diagram alur dari proses ini dapat dilihat pada gambar 3.4:



Gambar 3.4 Diagram alur proses *Extraction*

a) Putar *audio watermark* : untuk mendengarkan kembali kualitas suara file audio yang sudah di-*watermark*.
b) Ubah format audio : proses mengubah format dari ".wav" menjadi ".mp3" dan juga dari ".mp3" menjadi ".wav".
c) Sinkronisasi : sinkronisasi dilakukan untuk membandingkan file audio asli dengan yang sudah diberi *watermark*.
d) *Channel decoding* dan *frequency masking* : proses ini dilakukan untuk mendapatkan kembali data informasi yang berupa teks, untuk dibandingkan keutuhan teks yang sudah disisipkan pada proses *embedding*.
e) Analisis *robustness* : dalam tahap ini dilakukan proses analisa untuk melihat nilai BER dan CER dari file audio yang sudah di-*watermark*.

Setelah dilakukan proses ubah format audio dengan *online-convert*, audio yang dihasilkan akan masuk ke dalam proses sinkronisasi terlebih dahulu, setelah itu dilakukan ekstraksi untuk mendapatkan informasi yang disisipkan ditahap *embedding*.



Gambar 3.5 Proses ekstraksi [3]

Sinyal *watermark* yang telah diubah formatnya akan difilter untuk pemisahan sinyal replikanya. Setelah itu sinyal replika tersebut akan dimasukkan kembali ke dalam *replica creator* sebelum nantinya difilter kembali menggunakan *Low-pass Filter* (LPF) untuk mendapatkan kembali informasi yang telah disisipkan. Setelah melalui LPF maka dilakukan pengecekan nilai BER untuk mengetahui apakah informasi yang didapat dalam kualitas baik atau buruk (seperti kondisi awal atau tidak).

3.3 Penguji sistem

Penelitian ini dilakukan pengujian dengan dua cara, yaitu Subjektif dan Objektif pada audio yang ter-*watermark.*

3.3.1 Pengujian secara subjektif

Pengujian subjektif berdasarkan pada karakteristik *auditory* manusia. Pengujian ini umumnya dilakukan dengan mengukur *Mean Opinion Score* (MOS) berdasarkan kriteria pada tabel berikut :

Tabel 3.1 Kriteria penilaian MOS [10]

Score	Kualitas Audio	Level Distorsi
1	Bad (Buruk)	*Watermark* mengganggu sekali, audio tidak dapat terdengar

2	Poor (Kurang)	*Watermark* mengganggu sekali, audio bisa terdengar
3	Fair (Cukup)	*Watermark* terasa dan sedikit mengganggu
4	Good (Baik)	*Watermark* terasa sedikit tapi tidak mengganggu
5	Excellent (Amat Baik)	*Watermark* tidak terasa

Proses segmentasi Pada beat detection sinyal hasil segmentasi di olah seperti di jelaskan untuk mencari titik titik ekstrim. Beat detection memungkinkan kita untuk melakukan penyisipan atau tidak, Dengan menggunakan titik-titik ekstrim setiap frame untuk pemungutan suara mayoritas untuk satu BIT ,kita mengambil setiap frame, menganalisanya dengan EMD

3.3.2 Pengujian secara objektif

Pengujian ini dilakukan berdasarkan pada hasil penyisipan, dan diukur dengan tiga parameter, yaitu :

1) *Signal to Noise Ratio* (SNR)

Signal to Noise Ratio adalah nilai yang menyatakan tingkat *noise* atas audio yang telah disisipkan pesan. SNR digunakan untuk mengukur kualitas audio secara obyektif sesuai standar audio yang ada, dan batas minimum SNR dikatakan bagus untuk audio > 20 dB [9]

$$SNR = 10\ log_{10}\left[\frac{\sum_{i=0}^{N-1} f^2(n)}{\sum_{i=0}^{N-1}(g(n)-f(n))^2}\right] \quad (3.5)$$

2) *Bit Error Rate* (BER)

Bit Error Rate merupakan parameter pengukuran obyektif juga yang dipakai untuk mengukur ketepatan data hasil ekstraksi pesan tersembunyi yang disisipkan pada file audio dengan cara menghitung presentase bit yang salah dari hasil ekstraksi dengan bit keseluruhan sebelum dilakukan penyisipan, secara matematis, dapat dihitung dengan [10]:

$$BER = \frac{jumlah\ bit\ error}{jumlah\ bit\ seluruhnya} \times 100\ \% \quad (3.6)$$

3) *Character Error Rate* (CER)

Character Error Rate merupakan parameter pengujian yang dipakai untuk mengukur tingkat akurasi data hasil proses ekstraksi dari pesan yang telah disisipkan pada *host audio* dengan menghitung presentase perbandingan jumlah karakter yang salah satu *error* dari hasil ekstraksi dengan jumlah

karakter keseluruhan sebelum dilakukan ekstraksi, secara matematis, dapat dihitung dengan [11]:

$$CER = \frac{jumlah\ karakter\ yang\ salah}{jumlah\ karakter\ keseluruhan} \times 100\ \% \quad (3.7)$$

4) *TestBed*

TestBed adalah sekumpulan tes yang dilakukan secara mandiri untuk mengetes kualitas ketahanan audio yang dihasilkan. *TestBed* didasarkan dari ISO/IEC TR 21000-11 PAT dan SDMI pada tahun 1999, Parameter *item* serangan yang digunakan pada masing-masing tes dapat dilihat pada tabel berikut :

Tabel 3.2 *Tesbed item parameter* [10]

Item#	Type of Signal Processing	Description
1	Low pass filter	< 6K, by Cool Edit Pro
2	Bans pass filter	100 – 6k Hz, 12dB/oct., 2nd order Butterworth filter, by Cool Edit Pro
3	D/A,A/D	D/A.A/D converting twice
4	Stereo to Mono	Cool Edit Pro
5	Noise addition	Adding white and pink noise with constant level of 20dB lower than total averaged music power, by Cool Edit Pro
6	Changing the sample rate	44.1K -> 22.05K, by Cool Edit Pro
7	Time Scale modification	Pitch-invariant time scaling : +/-4%, by Cool Edit Pro
8	Linear speed change	+/- 10%, by Cool Edit Pro
9	Pitch shifting	+/- 4%, by Cool Edit Pro
10	Multi-band equalization	10-band graphic equalizer with the characteristics listed below: Freq.[Hz]: 31 62 125 250 500 1K 2K 4K 8K 16K Gain[dB]: -6 +6 -6 +6 -6 +6 -6 +6 -6 +6 by Cool Edit Pro

11	Echo addition	Maximum delay: 100ms Feedback coefficient: around 0.3 by Cool Edit Pro
12	Cropping	30sec and 60sec, by Cool Edit Pro
13	MPEG-1	Layer 2, 3(128K)
14	MPEG-2	ISO/IEC 13818-7, AAC(128K)
15	MPEG-4	ISO/IEC 14496-3, LC-AAC, HE-AAC(128K)
16	Dolby	AC3(128K), E(128K)

BAB IV
ANALISA DAN PENGUJIAN SISTEM

4.1 Analisa masalah

Sebagian besar produk digital seperti gambar, suara, ataupun video merupakan suatu karya seseorang yang dilindungi hak cipta, banyak dan mudahnya penggandaan atau duplikasi suatu media, membuat pemegang hak cipta dirugikan. Berbagai cara telah dilakukan demi menjaga karya dan keaslian suatu produk tersebut, jika media gambar dilindungi dengan menggunakan teknik *image processing*, sedangkan suara menggunakan teknik *audio watermarking*, fokus terhadap *audio watermarking*. *Audio watermarking* itu sendiri merupakan suatu teknik penyisipan atau penyembunyian suatu informasi ke dalam suatu *file audio*, dan dengan menggunakan metode *frequency masking*, dimana metode tersebut memanfaatkan kelemahan pendengaran manusia yang tidak dapat mendengar pada frekuensi tertentu, yang diharapkan dapat memberikan ketahanan terhadap *file audio* yang telah disisipkan.

Frequency masking ini pun dibagi menjadi dua kelompok, yaitu *temporal watermarking* dan *spectral watermarking*. *Temporal watermarking* itu melakukan penyisipan data pada *audio host* dalam domain waktu, sedangkan *spectral watermarking* harus melakukan proses transformasi terlebih dahulu dari domain waktu ke domain frekuensi, sehingga nanti penyisipan dilakukan pada elemen-elemen frekuensi, untuk merubah dari domain waktu ke domain frekuensi membutuhkan metode FFT (*Fast Fourier Transform*).

Dalam proses *watermarking* pun harus memenuhi karakteristik yang telah ditentukan, beberapa karakteristik itu, yaitu :

1) *Percetual Transparancy* : yang merupakan kebutuhan utama dari *watermarking*, dimana informasi yang telah disisipkan seharusnya tidak menurunkan kualitas dari *host-data* yang disisipkan. Dan tidak boleh terdengar oleh telinga.
2) *Robustness* : informasi yang disisipkan harus mempunyai ketahanan terhadap beberapa jenis serangan pada *watermark*, dan informasi tersebut harus dapat diterima kembali di *decoder* dengan benar.

3) *Data Rate/Payload/Bit Rate* : *watermark* harus mampu mem-verifikasi dan menyediakan bukti yang terpercaya untuk membuktikan keaslian suatu produk.

Dari sistem, metode, dan karakteristik di atas, akan diujikan ketahanannya dengan cara mengubah format *file audio* pada saat *file audio* telah disisipkan data informasi, mengubah format audio itu sendiri pun menggunakan *online-convert* dari format pertama hasil *embedding* ".mp3" lalu menjadi ".wav" begitu pula ".wav" menjadi ".mp3", maka tahap berikutnya yaitu dengan mendengarkan hasil *file audio* yang sudah di *watermark*, bertujuan untuk mengukur standar suara yang dihasilkan. Apakah berpengaruh terhadap kualitas audio atau tidak. Tahap itu merupakan penilaian secara subjektif, sedangkan secara objektif itu pada saat pengambilan data informasi (*extraction*) dengan menghitung BER, CER, dan juga dilihat hasil data yang disisipkan, apakah kembali dalam hasil yang utuh atau tidak.

4.2 Pengujian sistem

Pengujian sistem ini dilakukan dengan menggunakan perangkat keras dan perangkat lunak dengan spesifikasi sebagai berikut :

a) 1 buah PC dengan spesifikasi (perangkat keras) :
   - Processor Intel Core i5
   - RAM 4 Gb
   - Harddisk 500 Gb
b) 1 buah Speaker Stereo
c) Sistem Operasi Microsoft Windows 7 (perangkat lunak)
d) MATLAB 2014
e) *online-convert*

Pengukuran hasil *watermark* dilakukan dengan cara objektif dan subjektif pada beberapa pengujian, yaitu :

a) Mengukur hasil penyisipan pesan yang sama ke dalam 5 *file audio* yang berbeda jenis nya berdasarkan nilai SNR.

b) Mengukur hasil dari salah satu *host audio* yang telah disisipkan beberapa teks dengan jumlah karakter minimal dan maksimal yang bisa disisipkan berdasarkan nilai SNR.
c) Mengukur hasil tingkat ketahanan dari salah satu *file audio* yang telah dilakukan pengubahan format audio dengan menggunakan *online-convert* berdasarkan nilai BER, CER, dan keutuhan data informasi yang telah disisipkan.
d) Mengukur kualitas suara atau *file audio* dari hasil sebelum di *watermark* dan setelah di *watermark* dari setiap jenis *file audio* yang berbeda-beda.

Penilaian dilakukan pada 5 jenis *file audio* yang berbeda dengan sebagai audio host yang telah disisipkan data informasi berupa teks. Berikut ini 5 jenis *file audio* :

Tabel 4.1 Berkas audio dan jenisnya

Audio	Jenis Audio
Jazz_first.wav	Lagu dengan nada pelan
Metal_first.wav	Lagu dengan nada keras
Slow_first.wav	Lagu dengan nada pelan
Vocal_first.wav	Suara pembicaraan
Movie_first.wav	Suara film

4.2.1 Analisa secara objektif

Pengujian pada tahap ini dilakukan terhadap 5 *file host audio* yang nantinya akan disisipkan informasi rahasia berupa teks dengan kata “Telkom University 1st”, pengujian secara objektif ini pertama dilakukan pada kondisi normal tanpa tambahan gangguan apapun, kedua dilakukan pada salah satu *file audio* yang disisipkan dengan jumlah karakter maksimal yang dapat disisipkan, ketiga itu diuji dengan mengubah format menggunakan *online-convert* dari “.mp3” menjadi “.wav” begitu pula “.wav” menjadi “.mp3” dan diambil kembali data informasi yang telah disisipkan (*extraction*)

kemudian pengukuran dilakukan berdasarkan parameter nilai BER, CER, serta utuh atau tidaknya data informasi yang telah disisipkan sebelumnya.

4.2.1.1 Analisa perbandingan nilai SNR dengan jenis *file audio* yang berbeda

Berdasarkan hasil pengujian sistem audio watermarking yang dilakukan, dengan menyisipkan data informasi berupa teks "Telkom University 1st", dan hasil dari proses tersebut sebagai berikut :

Tabel 4.2 Perbandingan nilai SNR dengan *file audio* yang berbeda

Jenis Audio	SNR setelah proses *Embedding* (dB)	Tingkat Akurasi SNR
Jazz	25.4001	Baik
Metal	26.6613	Baik
Slow	26.8452	Baik
Vocal	28.3236	Baik
Movie	23.6165	Baik

Pada tabel 4.2 dapat dilihat bahwa nilai SNR setelah dilakukannya proses penyisipan (*embedding*) diperoleh hasil SNR yang berbeda-beda setiap jenis *file audio*, jenis *file audio* Vocal_first.wav memperoleh nilai SNR yang baling besar dibandingkan dengan jenis audio yang lainnya, hal berikut dikarenakan pada jenis *file audio* Vocal_first.wav tersebut memiliki *range* frekuensi yang sangat baik untuk menyisipkan pesan informasi, sehingga metode *frequency masking* dapat menyisipkan pesan informasi di frekuensi tertentu yang sulit untuk didengar pada telinga manusia. Dengan demikian metode *frequency masking* pada *file audio* Vocal_first.wav memiliki noise/serangan yang sedikit pada saat proses penyisipan (*embedding*) dibandingkan dengan jenis *file audio* yang lainnya.

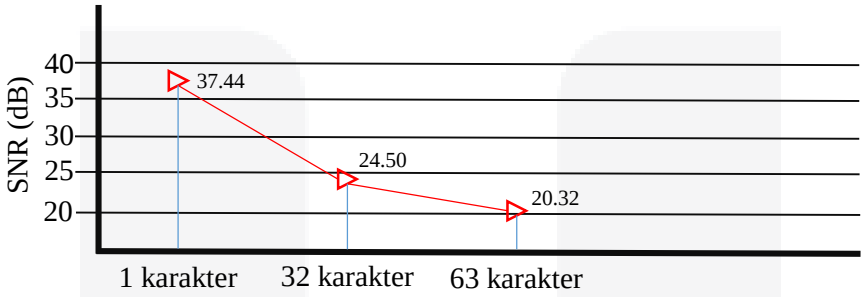
4.2.1.2 Analisa perbandingan nilai SNR terhadap jumlah karakter

Berdasarkan hasil pengujian sistem *audio watermarking* yang dilakukan terhadap salah satu jenis *file audio* yaitu Movie_first.wav dengan menyisipkan jumlah karakter

minimal dan maksimal dari 1 karakter sampai dengan jumlah karakter yang paling banyak, dengan parameter pengujian tersebut, maka didapatkan hasil sebagai berikut :

Tabel 4.3 Perbandingan nilai SNR terhadap jumlah karakter

Jenis Audio	Jumlah Karakter	Hasil SNR (dB)	Tingkat Akurasi SNR
Movie	Karakter Minimum		
	1 karakter	37.4456	Baik
	Karakter Maksimum		
	63 karakter	20.3290	Cukup Baik



Gambar 4.1 Perbandingan nilai SNR terhadap jumlah karakter

Pada tabel 4.3 berupa hasil perbandingan antara nilai SNR pada *audio watermarking* terhadap jumlah karakter minimum dan maksimum yang dapat disisipkan, dari hasil tersebut dapat disimpulkan bahwa semakin banyak jumlah karakter yang disisipkan (teks), maka semakin kecil pula nilai SNR dari *file audio watermarking* tersebut, dari *file audio* Movie dengan panjang audio 10 detik, hal itu dikarenakan semakin banyak jumlah karakter yang disisipkan, maka semakin banyak pula bit yang disisipkan pada setiap frekuensi menggunakan metode ini, dengan begitu tingkat *noise* atau serangan pun akan semakin besar dan mengakibatkan nilai SNR yang semakin kecil pula.

4.2.2 Analisa ketahanan data informasi terhadap perubahan format

Berdasarkan hasil pengujian sistem *audio watermarking* yang dilakukan terhadap salah satu jenis *file audio* yaitu Vocal_first.wav dengan menyisipkan data informasi berupa "Telkom University 1st" dan mengubah formatnya menjadi Vocal_first.mp3, dengan menggunakan *online-convert*, maka didapatkan hasil sebagai berikut :

Tabel 4.4 Ketahanan data informasi terhadap perubahan format (*online-convert*)

Jenis Audio	Proses format yang diubah	Sebelum ubah format menggunakan *converter*		Setelah ubah format menggunakan *converter*		
		Data yang disisipkan (*embedding*)	Nilai SNR (dB)	Data yang diekstrak (*extracting*)	Nilai BER	Nilai CER
Vocal_first,wav	.mp3 menjadi .wav	Telkom University 1st	28.3236	Telkom University 1st	0	0
Vocal_first,wav	.wav menjadi .mp3	Telkom University 1st	28.3236	û?ÿ¿«	0.5357	1

Pada tabel 4.4 didapatkan hasil sebagai berikut, data informasi yang sebelumnya disisipkan (*embedding*) dapat kembali secara utuh dan nilai BER dan CER nya pun bisa dikatakan layak dalam uji ketahanan, hal tersebut dikarenakan pada proses konversi menggunakan *online-convert*, saat mengubah dari ".Mp3" menuju ".Wav", *file audio* akan mengalami peningkatan kualitas suara serta tidak adanya data yang dikompres, namun sebaliknya saat mengubah dari ".Wav" menuju ".Mp3", maka *file audio* akan mengalami penurunan kualitas suara, dan berpengaruh pada hasil *converter* karena adanya pengompressan audio, pada tabel 4.4 dapat dilihat data yang disisipkan tidak kembali secara utuh, dengan nilai BER : 0.5357 dan CER : 1.

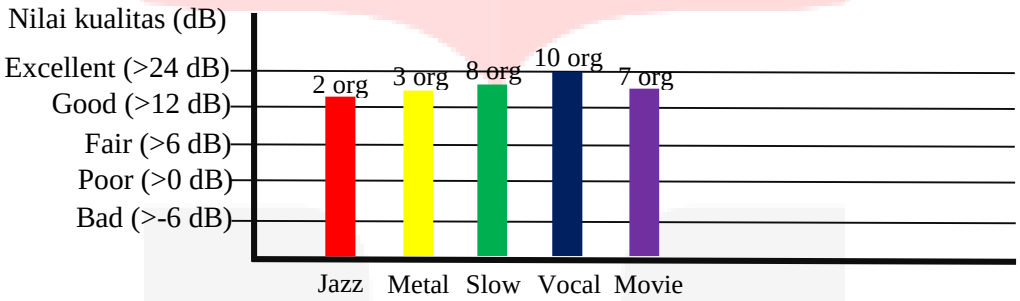
4.2.3 Analisa secara subjektif

Pengujian pada tahap ini, digunakan sebagai metode yang dikenal dengan sebutan *Mean Opinion Score* (MOS). MOS itu sendiri pun dilakukan dengan melakukan

pengumpulan data atau *survei* kepada 30 orang untuk membandingkan audio yang sudah di *watermark* dengan audio yang belum di *watermark*, dengan data informasi yang disisipkan kali ini yaitu "Telkom University 1st", dan didapatkan hasil dari *survei* sebagai berikut :

Tabel 4.5 Kriteria penilaian MOS dalam (dB)

SNR (dB)		Level Distorsi
Excellent	➢ 24 dB	*Watermark* tidak terasa
Good	➢ 12 dB	*Watermark* terasa sedikit, tapi tidak mengganggu
Fair	➢ 6 dB	*Watermark* terasa dan sedikit mengganggu
Poor	➢ 0 dB	*Watermark* mengganggu sekali, audio bisa terdengar
Bad	➢ -6 dB	*Watermark* mengganggu sekali, audio tidak dapat terdengar



Gambar 4.2 Hasil MOS dengan perbandingan nilai kualitas suara dan SNR

Berdasarkan gambar 4.1 diatas, hampir semua responden berpendapat bahwa *audio watermarking* yang dihasilkan oleh *file audio* Vocal memiliki kualitas suara amat baik, dan *watermark* tidak terasa. Hal tersebut dibuktikan saat pengujian secara objektif yang menghasilkan nilai SNR yang lebih baik diantara *file audio* yang lain, yaitu 28.3236 dB.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian sistem *audio watermarking* sebelumnya, dapat disimpulkan bahwa :

1. Penyisipan jumlah karakter pada salah satu *file audio*, sangat berpengaruh terhadap nilai SNR, dari hasil pengujian *file audio* Movie memiliki batasan karakter maksimal yang dapat disisipkan yaitu 63 karakter, dengan nilai minimum SNR mencapai 20.3290, hal tersebut terjadi karena setiap banyaknya karakter (teks) yang disisipkan, maka semakin banyak pula bit yang disisipkan pada setiap frekuensi, dengan begitu akan berbanding terbalik dengan tingkat *noise* atau serangan, karena mengakibatkan nilai SNR yang semakin kecil pula.
2. Metode *Frequency Masking* dapat mencapai hasil *audio watermarking* yang sangat baik dan maksimal, hal tersebut dilihat dari nilai BER dan CER pada pengujian sistem tersebut dengan mengubah format menggunakan *online-convert* dari ".Mp3" menjadi ".Wav" didapatkan hasil BER : 0, dan CER : 0. Begitupun dengan data yang disisipkan berupa "Telkom University 1st" dapat kembali dengan utuh. Hal tersebut dikarenakan pada proses konversi dari .Mp3 menuju .Wav, mengalami peningkatan kualitas suara dan tidak adanya data yang dikompresi.
3. Data MOS menunjukkan sebagian besar penilaian *file audio* Vocal memiliki kualitas suara yang lebih baik dibandingkan dengan yang lain.

5.2 Saran

Tugas akhir ini dilakukan untuk menganalisa penerapan teknik *frequency masking* pada *audio watermarking* dan diuji ketahanannya pada perubahan format menggunakan *online-convert*, penelitian ini dibuat menggunakan MATLAB 2014, dan memiliki hasil yang baik dan layak, untuk penelitian selanjutnya diharapkan memiliki parameter pengukuran lain, antara lain penulis menyarankan :

1) Penelitian selanjutnya, gunakan penelitian dengan metoda berbeda, contoh : *Echo Hiding, Replica Modulation, Direct Spread Spectrum, Less Significant Bit*, dan lain-lain untuk perbandingan dan perubahan nilai SNR, BER, dan CER yang dihasilkan.
2) Panjang data informasi (teks), memiliki panjang dan karakter yang berbeda-beda (unik).
3) Informasi yang disisipkan berupa media lain, contoh : Gambar, Audio, atau Video. Agar terlihat pengaruh dan perbedaan yang signifikan ketika disisipkan pada *Host Audio*.
4) Penelitian selanjutnya, dapat menggunakan format audio selain jenis ".Mp3" dan ".Wav", seperti : ".Mp4", "WMA", "AAC", atau "FLAC". Agar terlihat perbandingannya antar format audio.
5) Penelitian selanjutnya, bisa ditambahkan kode-kode lain untuk dapat memperkecil nilai BER dan CER atau memperbagus kualitas *audio watermarking* dan menambahkan akurasi lebih baik dari sebelumnya namun tetap menjaga *audibility*.
6) Penelitian selanjutnya, bisa melakukan uji ketahanan dengan cara lain, seperti pemotongan *file audio*.
7) Nilai MOS yang baik, didapatkan dengan memberikan kuisioner kepada orang yang mengerti tentang audio dan kualitas suara.

DAFTAR PUSTAKA

[1] Hargtung, F., Kutter, M. 1998. "*Multimedia watermarking techninques*", Proc. IEEE, Vol. 86, 1079-1107.

[2] Chauhan, S., Rizvi, S. 2013. "*A Survey: Digital Audio Watermarking Techniques and Applications*", 4th International Conference on Computer and Communication Technologies, 185-192.

[3] Petrovic, R. 2001. "*Audio signal watermarking based on replica modulation*", 5th International Conference TELSIKS'01, 227-234.

[4] Julius, A.S. 2012 "*Analisis Watermark pada File Audio Berbasis Metode Phase Coding".*

[5] Otniel. 2010/2011. "*Digital Audio Watermarking dengan Fast Fourier Transform".* Bandung.

[6] Mirko Luca Lobina., Luigi Atzori., & Davide Mula. "*Masking Models and Watermarking: A Discussion on Methods and Effectiveness*" University of Cagliari, Italy., University Chair of Juridicual Informatics at LUISS Guido Carli, Italy.

[7] Herianto "*Novel Digital Audio Watermarking*" Jurusan Teknik Informatika ITB, Bandung 40132, Email: if14077@students.if.itb.ac.id

[8] DongHwan Shin. "*Introduction to Audio Watermarking".* Seoul.

[9] Eargle, John. (2005). *Handbook of Recording Engineering.* Springer

[10] Hussein, S. 2015. "*Analisa Audio Watermarking Berbasis Metode Phase Coding pada Ambient Mode"*

[11] Andini, N., Farda, E., Hidayat, B. 2014. "*Implementasi Metode Hidden Markov Model untuk Deteksi Tulisan Tangan"*

[12] Marimoto, N., Bender, W., Gruhl, D., & Lu, A. 1996 . *Techniques for Data Hiding*. IBM Systems Journal vol.35 (3-4), 313-336.

[13] Nurbani Yusuf (2015)"*Analisis Ketahanan Audio Watermarking di Domain Frekuensi pada Ambient Mode dengan Menggunakan Frequency Masking Method*" Jurusan Teknik Telekomunikasi Telkom University.

LAMPIRAN A

DATA HASIL PENGUJIAN

“Pengujian Mean Opinion Score (MOS)”

ANALISIS KETAHANAN AUDIO WATERMARKING PADA FORMAT AUDIO MENGGUNAKAN METODE FREQUENCY MASKING

Perbandingan antara file audio sebelum di watermark dan sesudah di watermark, dengan file audio yang berbeda-beda

* Wajib

1) Jazz_first.wav (https://goo.gl/ucSZ6g) vs Jazz_TelUniv1st.mp3 (https://goo.gl/7TJnqp) *

- Bad (Buruk) , Watermark mengganggu sekali, audio tidak dapat terdengar
- Poor (Kurang) , Watermark mengganggu sekali, audio bisa didengar
- Fair (Cukup) , Watermark terasa dan sedikit mengganggu
- Good (Baik) , Watermark terasa sedikit tapi tidak mengganggu
- Excellent (Amat Baik) , Watermark tidak terasa

2) Metal_first.wav (https://goo.gl/PrVv9J) vs Metal_TelUniv1st.mp3 (https://goo.gl/dAZ2Fd) *

- Bad (Buruk) , Watermark mengganggu sekali, audio tidak dapat terdengar
- Poor (Kurang) , Watermark mengganggu sekali, audio bisa didengar
- Fair (Cukup) , Watermark terasa dan sedikit mengganggu
- Good (Baik) , Watermark terasa sedikit tapi tidak mengganggu
- Excellent (Amat Baik) , Watermark tidak terasa

3) Slow_first.wav (https://goo.gl/EyVV1n) vs Slow_TelUniv1st.mp3 (https://goo.gl/iH4lW3) *

- Bad (Buruk) , Watermark mengganggu sekali, audio tidak dapat terdengar
- Poor (Kurang) , Watermark mengganggu sekali, audio bisa didengar
- Fair (Cukup) , Watermark terasa dan sedikit mengganggu
- Good (Baik) , Watermark terasa sedikit tapi tidak mengganggu
- Excellent (Amat Baik) , Watermark tidak terasa

4) Vocal_first.wav (https://goo.gl/kXefDB) vs Vocal_TelUniv1st.mp3 (https://goo.gl/SbpDzS) *

- Bad (Buruk) , Watermark mengganggu sekali, audio tidak dapat terdengar
- Poor (Kurang) , Watermark mengganggu sekali, audio bisa didengar
- Fair (Cukup) , Watermark terasa dan sedikit mengganggu
- Good (Baik) , Watermark terasa sedikit tapi tidak mengganggu
- Excellent (Amat Baik) , Watermark tidak terasa

5) Movie_first.wav (https://goo.gl/OLMdDn) vs Movie_TelUniv1st.mp3 (https://goo.gl/5LNjYI) *

- Bad (Buruk) , Watermark mengganggu sekali, audio tidak dapat terdengar
- Poor (Kurang) , Watermark mengganggu sekali, audio bisa didengar
- Fair (Cukup) , Watermark terasa dan sedikit mengganggu
- Good (Baik) , Watermark terasa sedikit tapi tidak mengganggu
- Excellent (Amat Baik) , Watermark tidak terasa

Nilai	Jazz	Metal	Slow	Vocal	Movie
Excellent (> 24 dB)	1	2	5	7	2
Good (> 12 dB)	1	1	3	3	5
Fair (> 6 dB)	-	-	-	-	-
Poor (> 0 dB)	-	-	-	-	-
Bad (> -6 dB)	-	-	-	-	-
Jumlah	2	3	8	10	7
Rata-rata	6,66	10	26,66	33,33	23,33

LAMPIRAN B
KODE PROGRAM

```
function varargout = embedding(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @embedding_OpeningFcn, ...
                   'gui_OutputFcn',  @embedding_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
function embedding_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

guidata(hObject, handles);
function varargout = embedding_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function browser_Callback(hObject, eventdata, handles)
 [name,path]=uigetfile('*.mp3;*');
global host fs nama;
nama=name;
[host,fs]=wavread([path,nama]);
set(handles.hostname,'String',nama);
axes(handles.axes1);plot(host);
function player_Callback(hObject, eventdata, handles)
global host fs suara;
status=get(hObject,'String');
suara=(audioplayer(host,fs));
if strcmp(status,'Play')
    play(suara);
    set(hObject,'String','Stop');
else set(hObject,'String','Play');
    stop(suara);
end

function teks_Callback(hObject, eventdata, handles)
function teks_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function process_Callback(hObject, eventdata, handles)
global host fs out pjg bitdataawal pesan
pesan=get(handles.teks,'String');
bitdataawal=reshape(de2bi(double(pesan),8)',1,[]);
watermark=bitdataawal;
pjg=length(watermark);
alpha=0;
x=host;%read the wave film
```

```
x=x';
% seg1=reshape(x,885);
seg1=enframe(x,885); % each frame is 20ms, 500 frames in total
% seg1=buffer(x(1,:),length(x)/500);
% seg1=seg1';
index=1;
 for i=1:1:size(seg1,1);
     if index<=length(watermark)
    if watermark(1,i)==1
        temp(i,:)=fft2(seg1(i,:));
        [r,c]=find(temp(i,:)==max(temp(i,:)));
        if c(1)+1<=885 && c(1)>=1
          temp(i,c(1)+1)=temp(i,c(1)+1)*alpha;
        elseif c(1)+1>885
          temp(i,c(1)-1)=temp(i,c(1)-1)*alpha;
        end
    else if watermark(1,i)==0
          temp(i,:)=fft2(seg1(i,:));
       end
    end
    index=index+1;
     else
        temp(i,:)=fft2(seg1(i,:));
     end

 end
index=1;
for i=1:1:size(seg1);
    temp1(i,:)=ifft2(temp(i,:));
end
temp1=real(temp1');
out=temp1(:);
out=mapminmax(out);
for i=1:1:length(out)% the following two loops avoid data overflow
    if out(i)>=1
    out(i)=0.99996948242188;
%     out(i)=1;
    end
end
for i=1:1:length(out)
    if out(i)<=-1
out(i)=-0.99996948242188;
% out(i)=-1;
    end
end

axes(handles.axes11);plot(out);

y=x(1,1:length(out))';
snr=10*log10(sum(y.^2)/sum((y-out).^2));
set(handles.snrtarget,'String',num2str(snr));
function player2_Callback(hObject, eventdata, handles)
global out fs host suara2
suara2=audioplayer(out,fs)
status=get(hObject,'String');
if status=='Play'
    play(suara2);
    set(hObject,'String','Stop');
else
```

```
    stop(suara2);
    set(hObject,'String','Play');
end

function snrtarget_Callback(hObject, eventdata, handles)
function snrtarget_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function save_Callback(hObject, eventdata, handles)
global out fs nama
[namafile,path]=uiputfile('*.mp3;*','audio tersisipi',[cd,'\Hasil
Watermarking\',get(handles.snrtarget,'String'),nama]);
wavwrite(out,fs,strcat(path,namafile));
function pushbutton7_Callback(hObject, eventdata, handles)
function extract_Callback(hObject, eventdata, handles)
function edit5_Callback(hObject, eventdata, handles)
function edit5_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function pushbutton9_Callback(hObject, eventdata, handles)
function browser_emb_Callback(hObject, eventdata, handles)
global waudio fs
[namafile,path]=uigetfile('*.wav;*.mp3;*.mp4');
[waudio,fs]=audioread([path,namafile]);
axes(handles.axes4);plot(waudio);
set(handles.Judul,'String',namafile);
function extr_Callback(hObject, eventdata, handles)
global waudio laudio pjg bitdataawal pesan
z=0.7;
x=laudio';
y=waudio'';
seg1=enframe(x,885); % divided into 500 frames, each frame 2ms
seg2=enframe(y,885);
for i=1:1:size(seg1,1)
    tempx(i,:)=fft2(seg1(i,:));
    tempy(i,:)=fft2(seg2(i,:));

    [r,c]=find(tempx(i,:)==max(tempx(i,:)));
    if c(1)+1<=885 && c(1)>=1
        v(i)=tempy(i,c(1)+1)/tempx(i,c(1)+1);
    elseif c(1)+1>=885
        v(i)=tempy(i,c(1)-1)/tempx(i,c(1)-1);
    end
 end

 for i=1:1:size(seg1,1)
%         if v(i)>0.4 && v(i)<0.9; % threshold value(dpt bit sebanyak frame)
    %     if v(i)>0.3 && v(i)<0.7;
     if v(i)<z;
% if v(i)>=z;
         res(i)=1;
%         else
%             if v(i)<z;
    else if v(i)>=z;

            res(i)=0;
```

```
        end
    end
end
bit_hasil=res(1:pjg);
watermark=char(bi2de(reshape(bit_hasil,8,[])'));
set(handles.hasil,'String',watermark');

errornya=symerr(bitdataawal,bit_hasil);
BER=errornya/length(bitdataawal);
set(handles.hasilber,'String',num2str(BER));


errornya=0;
for k=1:length(watermark)
   if watermark(k)~= pesan(k)
      errornya=errornya+1;
   end
end
CER=errornya/k;
CER
num2str(CER)
set(handles.cerhasil,'String',num2str(CER));
function hasil_Callback(hObject, eventdata, handles)
function hasil_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function browser_ori_Callback(hObject, eventdata, handles)
global laudio fs
[namafile,path]=uigetfile('*.wav;*.mp3;*.mp4');
[laudio,fs]=audioread([path,namafile]);
axes(handles.axes5);plot(laudio);
set(handles.Judul1,'String',namafile);
function axes1_CreateFcn(hObject, eventdata, handles);
```