PROGRAM *VOICE RECOGNITION* MENGGUNAKAN METODE *FAST FOURIER TRANSFORM* (FFT)

SKRIPSI

Oleh:

ALIYA SALSABILA JULIANANDA

165090301111026

JURUSAN FISIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS BRAWIJAYA

MALANG

2020

PROGRAM *VOICE RECOGNITION* MENGGUNAKANMETODE *FAST FOURIER TRANSFORM* (FFT)

SKRIPSI

Sebagai Salah Satu Syarat Untuk Meraih Gelar
Sarjana Sains dalam Bidang Fisika

Oleh:

ALIYA SALSABILA JULIANANDA

165090301111026

JURUSAN FISIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS BRAWIJAYA

MALANG

2020

LEMBAR PENGESAHAN SKRIPSI

PROGRAM *VOICE RECOGNITION* MENGGUNAKAN METODE *FAST FOURIER TRANSFORM* (FFT)

Oleh:

ALIYA SALSABILA JULIANANDA 165090301111026

Malang, 16 JULI 2020

Pembimbing I	**Pembimbing II**
Dr.rer.nat. Abdurrouf, S.Si., M.Si	**Dr.Eng. Agus Naba, S.Si.,M.T.**
NIP. 197209031994121001	**NIP. 197208061995121001**

Mengetahui,
Ketua Jurusan Fisika
Fakultas MIPA UB

Prof. Dr. Muhammad Nurhuda, Rer. Nat
NIP 196409101990021001

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Aliya Salsabila Juliananda
NIM : 165090301111026
Jurusan : Fisika
Penulis Skripsi Berjudul :

PROGRAM *VOICE RECOGNITION* MENGGUNAKAN METODE *FAST FOURIER TRANSFORM* (FFT)

Dengan ini menyatakan bahwa :

1. **Isi dari skripsi yang saya tulis dan saya buat adalah benar-benar karya saya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam skripsi ini.**
2. **Apabila di kemudian hari ternyata skripsi yang saya tulis terbukti hasil menjiplak, maka saya akan bersedia menanggung segala resiko yang akan saya terima.**

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang, 19 Juni 2020
Yang menyatakan,

Aliya Salsabila Juliananda
165090301111026

PROGRAM *VOICE RECOGNITION* MENGGUNAKAN METODE *FAST FOURIER TRANSFORM* (FFT)

ABSTRAK

Suara merupakan suatu sinyal yang dipengaruhi oleh waktu. Untuk mengubah sinyal tersebut dari domain waktu menjadi domain frekuensi diperlukan transformasi sinyal. Salah satu yang dapat digunakan adalah metode *Fast Fourier T.ransform* (FFT). Proses untuk mengenali suara pembicara dapat didefinisikan sebagi *voice recognition*. Dalam proses ini dilakukan proses *training* sebagai referensi database yang akan dibandingkan dengan suara uji dan juga proses *testing* sebagai suara uji. Penelitian ini dilakukan memakai Matlab dan menggunakan GUI sebagai antarmukanya. Proses pengenalan suara dilakukan dengan mencari indeks nilai maksimum pada sinyal FFT, kemudian sinyal tersebut dianalisis untuk mengetahui ciri suara pengguna. Pada prosesnya, penelitian ini menggunakan nilai *frame blocking* 16, 32, 64, 128, dan 256. Indeks nilai maksimum antara pembicara memiliki rentang yang berbeda pada masing-masing *frame blocking*. Semakin besar nilai *frame blocking* yang digunakan, maka kemungkinan *error* akan semakin kecil. Program ini berhasil dibuat dan dapat mengenal suara dari pengguna. Pada program ini presentase pengenalan suara manusia terbaik adalah 94% saat nilai *frame blocking* 256.

Kata kunci: *Fast Fourier Transform* (FFT), *voice recognition*, Matlab

VOICE RECOGNITION PROGRAM USING FAST FOURIER TRANSFORM METHOD (FFT)

ABSTRACT

Sound is a signal that is influenced by time. To change the signal from the time domain to the frequency domain, signal transformation is needed, one of which can be used is Fast Fourier Transform (FFT) method. The process for recognizing the speaker's voice can be defined as voice recognition. In this process, the training process is done as a database reference that will be compared with the test sound and also the testing process as the test sound. This study using Matlab software and using the GUI as the interface. The voice recognition process is done by finding the maximum value on the FFT signal, then the signal is analyzed to find out the characteristic of user's voice. This study uses frame blocking values of 16, 32 ,64, 128, and 256. Each speaker has a different range of the maximum index on each frame blocking. The bigger value used, the possibility of errors will be smaller. This program was successfully created and can recognize the voice of the user. In this program, the best percentage of human voice recognition is 94% when the frame blocking value is 256.

Kata kunci: Fast Fourier Transform (FFT),voice recognition¸ Matlab

KATA PENGANTAR

Bismillahirahmanirrahim, Alhamdulillahi robbil ‘alamin. Segala puji dan syukur kepada Allah Subhanahu wa Ta’ala, Tuhan semesta alam yang telah mencurahkan karunia dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul “**PROGRAM *VOICE RECOGNITION* DENGAN METODE *FAST FOURIER TRANFORM***” sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Sains dalam bidang Fisika.

Dengan doa dan bantuan tersebut, penyusunan skripsi dapat berjalan dengan baik dan lancar .Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Orang tua penulis, Neno, dan Nena, serta seluruh keluarga penulis .yang senantiasa mendoakan dan memberikan dukungan kepada penulis
2. Dr.rer.nat. Abdurrouf, S.Si., M.Si. sebagai dosen pembimbing pertama dan Dr. Eng. Agus Naba, S.Si, MT., Ph.D., selaku dosen pembimbing kedua yang telah memberikan pengarahan dan masukan kepada penulis selama penyusunan skripsi.
3. Segenap Dosen Jurusan Fisika Universitas Brawijaya yang telah memberikan banyak ilmu dan bantuan selama penulis menempuh pendidikan sarjana.
4. Teman-teman seperjuangan yang selalu mendukung dan membantu penulis, Ni’ma, Masdar, Renald, Adjib, Fariz, Zul, dan Dite.

5. Mochammad Machfudh dan Satrio Wiradinata Riady Boer, kakak penulis yang selalu membantu dan memberikan pengaruh positif selama masa perkuliahan
6. Sahabat-sahabat selama masa perkuliahan yang menyemangati dan menemani selama perkuliahan ini, Irwansyah, Wulan, Bilal, Neni, Jassica, Diva, Restya, Majdi, dan Mona
7. Mecky, Yunda, Rulli, Fiza, dan seluruh keluarga Smanda Malang sebagai teman-teman seperantauan yang memberi canda tawa selama merantau ini
8. Seluruh pihak yang tidak bisa penulis tuliskan satu persatu yang telah membantu penulis baik secara langsung maupun tidak langsung

Penulis menyadari bahwa setiap karya manusia tidak ada yang sempurna, sehingga penulis yakin bahwa dalam penyusunan skripsi ini masih terdapat banyak kekurangan. Penulis sangan mengharapkan kritik dan saran yang membangun dari pembaca demi keberlanjutan riset skripsi ini. Semoga hasil penelitian dalam skripsi ini dapat bermanfaat bagi pembaca

Malang, 19 Juni 2020

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1. Latar Belakang

Manusia memiliki keragaman bentuk suara yang dapat dilihat dari presepsi suara fisik terhadap suara diantaranya adalah frekuensi, jenis suara, pitch, timbre, dan volumenya. Karakteristik suara manusia juga berbeda-beda akibat dari resonansi dalam tenggorokan yang juga berbeda. Dengan melihat dan mendengar secara langsung suara dari lawan pembicara manusia biasanya dapat langsung mengidentifikasi suara seseorang (Bhaskoro & D, 2012).

Suara memiliki rentang frekuensi tertentu. Frekuensi dan intensitas suara di nyatakan dalam satuan Hertz dan desibel (dB). Kedua satuan ini diambil dari nama penemunya, Alexander Graham Bell dan Heinrich Rudolf Hertz. (Adler, Azhar, & Supatmi, 2013)

Suara sendiri merupakan bentuk sinyal yang dipengaruhi oleh waktu. Proses analisis dalam domain waktu sendiri memerlukan analisis cukup panjang dan melibatkan turunan fungsi yang dapat menimbulkan ketidaktelitian hasil. Analisis dapat dilakukan jika sinyal tersebut berbentuk spektrum frekuensi. Untuk mengubah sinyal tersebut dari domain waktu menjadi domain frekuensi diperlukan transformasi sinyal. Salah satu metode yang dapat digunakan adalah *Fast Fourier Transform* (FFT). FFT dapat menunjukkan frekuensi yang terkandung di dalam sinyal dan

menunjukkan jumlah komponen frekuensi dalam sinyal (Syaifuddin & Suryono, 2014).

Berdasarkan latar belakang yang telah diuraikan diatas, penulis mengankat permasalahan dengan judul "Program *Voice Recognition* Menggunakan Metode FFT"

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana menerapkan metode FFT dalam proses pengenalan suara manusia?
2. Bagaimana akurasi metode FFT dalam proses pengenalan suara manusia?

1.3. Batasan Masalah

Permasalahan pada penelitian ini dibatasi oleh beberapa hal, diantaranya:

1. Program yang dikembangkan berbasis MATLAB
2. Metode yang digunakan *Fast Fourier Transform*
3. Input yang dimasukkan berupa suara manusia

1.4. Tujuan Penelitian

Tujuan dari penelitian yang dilakukan yaitu:

1. Untuk menerapkan metode FFT dalam proses pengenalan suara manusia
2. Untuk mengetahui akurasi metode FFT dalam proses pengenalan suara manusia

1.5. Manfaat Penelitian

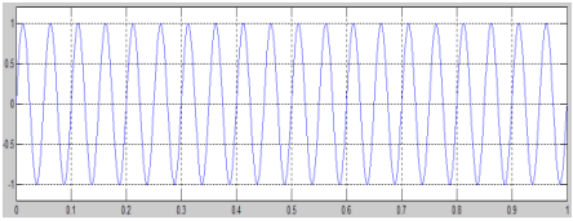
Manfaat yang bisa didapat dari penelitian ini yaitu untuk memberikan pengetahuan kepada pembaca mengenai penerapan metode FFT dalam proses pengenalan suara manusia, sehingga pembaca dapat mengetahui akurasi dari penggunaan metode FFT dalam mengenali suara manusia.

BAB II
TINJAUAN PUSTAKA

2.1. Sinyal

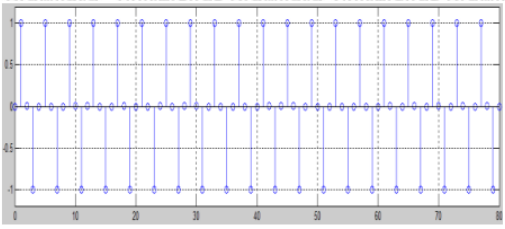
Sinyal adalah sesuatu yang menggambarkan data yang memiliki informasi dalam besaran fisis sesuai dengan perubahan dalam ruang, waktu, atau peubah-ubah bebas lainnya. Sinyal sendiri dapat diklasifikasikan dalam beberapa bentuk, antara lain sinyal dalam bentuk bilangan kompleks dan bilangan nyata (Mustofa, 2018). Beberapa klasifikasi sinyal diantaranya adalah:

1. Sinyal waktu kontinyu, yaitu sinyal yang terdefinisi pada setiap waktu (lihat Gambar 2.1)



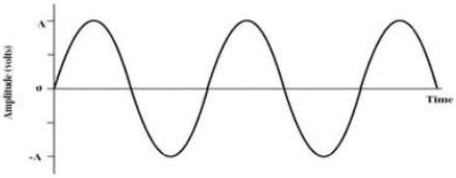
Gambar 2. 1. Sinyal waktu kontinyu

2. Sinyal waktu diskrit, yaitu sinyal yang terdefinisi pada waktu-waktu tertentu seperti ditunjukkan pada Gambar 2.2



Gambar 2. 2. Sinyal waktu diskrit

3. Sinyal analog, yaiitu sinyal waktu kontinyu dengan amplitudo yang kontinyu. Dapat dilihat pada Gambar 2.3



Gambar 2. 3. Sinyal analog

4. Sinyal digital, yaitu sinyal waktu diskrit dengan amplitudo bernilai diskrit (lihat Gambar 2.4)

Gambar 2. 4. Sinyal Digital

2.2. Suara

Suara merupakan gelombang akustik yang sesungguhnya memiliki kasus khusus dari suatu gelombang elastik pada medium udara atau fluida. Suara sendiri memiliki range frekuensi tertentu dan intensitas yang bias ataupun tidak bisa didengar oleh manusia (Adler *et al*., 2013).

Gelombang suara merupakan gelombang yang dihasilkan dari benda yang bergetar getaran ini akan merambat di udara, atau air, ataupun material lainnya. Satu-satunya tempat suara tak dapat merambat adalah ruangan hampa udara. Gelombang suara memiliki lembah dan bukit, satu lembah dan satu bukit akan membentuk satu siklus atau periode. Siklus ini berlangsung

berulang-ulang yang akan membentuk konsep frekuensi. sehingga, frekuensi merupakan jumlah dari sikus yang terjadi dalam satu detik. Satuannya yaitu Hertz. Gelombang suara yang semakin cepat, maka frekuensi semakin tinggi. Frekuensi lebih tinggi diinterpretasikan sebagai jalur yang lebih tinggi. (Sipasulta, Lumenta, & Sompie, 2014).

2.3. Pengenalan Suara

Pengenalan suara merupakan kemampuan komputer untuk membedakan kata-kata yang diucapkan. Pengenalan suara ini berupa program yang dapat mengenali perbendaharaan kata dari kata-kata yang sudah diprogram terlebih dahulu sebelumnya (Shelly, Cashman, & Vermaat, 2007). Proses yang dilakukan untuk mengenali suara pembicara yang dilakukan perangkat didefinisikan sebagai *voice recognition* . Dalam proses ini diperlukan proses referensi dan proses uji. Proses referensi diperlukan sebagai database yang akan dibandingkan dengan suara uji. *Voice recognition* memenuhi dua fungsi, yaitu identifikasi dan verifikasi. Identifikasi berfungsi untuk memecahkan identitas seseorang, sedangkan verifikasi untuk menolak atau menerima identitas yang di klaim(Prayoga, Astuti, & Waluyo, 2019).

2.4. *Sampling*

Sinyal analog dapat diubah dan diproses menjadi bit-bit digital, teknik yang memungkinkan untuk proses ini adalah *sampling*. *Sampling* merupakan proses untuk mendapatkan sinyal diskrit yang berguna untuk mengetahui ciri yang akurat dari

sinyal dan memudahkan proses desimasi (pengurangan jumlah sampling). Proses sampling sendiri dengan cara mengambil nilai-nilai sinyal pada titik-titik diskrit sepanjang variable waktu dari sinyal waktu kontinyu, sehingga didapatkan sinyal waktu diskrit. Jumlah dari titik-titik yang diambil setiap detik dinamakan sebagai *sampling rate*

Menurut teorema Nyquist, *sampling rate* diharuskan lebih besar 2 kali dari frekuensi sinyal aslinya, yaitu

$$fs \geq 2fm \; ; f > 2fm \tag{2. 1}$$

dimana fs merupakan frekuensi *sampling* dan *fm* adalah frekuensi tertinggi dari sinyal (Faradiba, 2017).

2.5. *Preprocessing*

Presprocessing adalah proses awal yang dilakukan untuk memperbaiki kualitas objek. Tujuan dilakukannya proses ini adalah untuk menyeratakan input sinyal suara agar lebih mudah diproses untuk pengenalan suaranya nanti. Dalam proses ini terdapat beberapa tahapan, diantaranya normalisasi, *frame blocking*, dan *windowing*, ekstraksi ciri, FFT, pencarian nilai maksimum, dan teks suara (Sibarani, 2018).

Proses normalisasi dilakukan untuk menyetarakan amlitudo suara terekam mejadi maksimum. Hal ini bertujuan agar efek kuat dan lemahnya suara yang terekam tidak mempengaruhi proses pengenalan suara. Perhitungan matematisnya dapat dilihat sebagai berikut

$$x_{norm}(n) = \frac{x_{input}}{max\ |x_{input}|} \tag{2. 2}$$

Dapat dilihat pada persamaan diatas bahwa normalisasi dilakukan dengan membagi data input yang berupa data sinyal masukann (x_{input}) dengan nilai abosulut maksimum data tersebut ($max\ |x_{input}|$). Dimana hasil datas sinyal normalisasi disimbolkan dengan $x_{norm}(n)$ dengan n adalah panjang sinyal (Prayoga *et al*., 2019).

Setelah proses normalisasi, dilakukan proses prmotongan beberapa bagian sinyal suara. Tujuan pemotongan sinyal ini untuk menghilangkan bagian yang tidak termasuk bagian dari sinyal dan mengurangi cacat sinyal akibar *noise*. Proses pemotongan sinyal ini dilakukan dua kali tahapan. Tahap pertama adalah memotong bagian *silence* atau bagian awal sinyal yang tidak termasuk sinyal suara sebenarnya. Tahapan kedua yaitu memotong bagian transisi sinyal dengan menghilangkan $^1/_4$ bagian dari sinyal (Sibarani, 2018).

Sinyal suara yang sudah dinormalisasi harus diproses secara *short segments* akibat adanya pergeseran artikulasi dari organ produksi vokal. Proses *frame blocking* merupakan proses pembagian suara menjadi beberapa frame. Hal ini berfungsi untuk memudahkan dalam perhitungan dan analisis suara (Prayoga *et al*., 2019).

Sinyal suara yang dipotong-potong saat proses *frame blocking* dapat menyebabkan kesalahan data pada proses *fourier trasnform* proses *windowing* diperlukan untuk mengurangi dikontinuitas dari potongan-potongan sinyal tersebut (Izzah, 2018). Jenis *window* yang digunakan pada penelitian ini adalah *hamming*

window. *Hamming window* biasa digunakan dalam analisa sinyal suara, proses ini dilakukan untuk mengurangi perubahan tidak terduga dan tidak diinginkan pada frekuensi yang terjadi pada *frame* sinyal suara. Persamaan matematis fungsi ini d apat dilihat pada persamaan berikut: (Singh, 2015).

$$w(k) = 0{,}52 - 0{,}46\, cos\frac{2\pi k}{L-1} \quad (2.3)$$

$$x(k) = S_n w(k) \quad (2.4)$$

dimana,

k = bilangan bulat

L = *frame size*

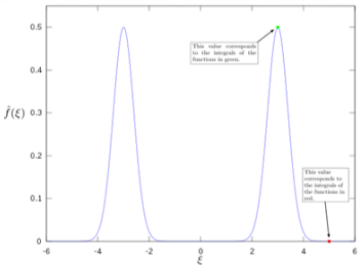
S_n = nilai sampel dari *frame* sinyal

$x(k)$ = nilai sample sinyal hasil *windowing*

$w(k)$ = fungsi *window*

2.6. Transformasi Fourier

Untuk menganalisis sinyal suara biasanya digunakan tranformasi yang disebut FFT. FFT merupakan tranformasi matematika yang digunakan untuk mengubah sinyal dari domain waktu ke domain frekuensi. FFT menggunakan spektrum frekuensi dari sinyal suara sebagai pengganti gelombang. Domain frekuensi ini memberikan informasi lebih lanjut tentang sinyal suara dan karenanya lebih efisien dalam pengenalan suara (Singh, 2015).



Gambar 2. 5. Transformasi Fourier pada fungsi osilasi

Ada dua jenis algoritma FFT yaitu algoritma *Fast Fourier Transform Decimation in Time* (FFT DIT) dan algoritma *Fast Fourier Transform Decimation in Frequency* (FFT DIF). Pada DIT, input disusun dan dikelompokkan menjadi kelompok ganjil dan kelompok genap. Untuk runtun bernomor genap yaitu *x(0), x(2), x(4), ...,x(N-2)* dan runtun bernomor ganjil yaitu *x(1), x(3), x(5),...,x(N-1)*. Kedua runtun berisi N/2-titik. Runtun genap ditandai dengan *x(2k)* dengan *x=0* sampai *k=N/2-1*, sedangkan runtun ganjil menjadi *x(2k-1)*. FFT ini menggunakan persamaan DFT yang dibagi bagian ganjil dan genap. Persamaan (2.5) menunjukkan persamaan DFT, kemudian persamaan ini dibagi menjadi bagian ganjil dan genap yang dapat dilihat pada persamaan (2.6) dan (2.7).

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-\frac{j2\pi nk}{N}} \quad (2.5)$$

karena $W_N^2 = e^{-\frac{j2\pi n2}{N}} = e^{-\frac{j2\pi}{\frac{N}{2}}}$, maka $W_N^2 = W_{\frac{N}{2}}$. Sehingga persamaan (2.5) menjadi:

$$X(k) = \sum_{n=0}^{(\frac{N}{2})-1} x(2n) W_{\frac{N}{2}}^{nk} + W_N^{2nk} \sum_{n=0}^{(\frac{N}{2})-1} x(2n+1) W_{\frac{N}{2}}^{nk} \quad (2.6)$$

$$X\left(k+\frac{N}{2}\right)=\sum_{n=0}^{\left(\frac{N}{2}\right)-1} x(2n)W_{\frac{N}{2}}^{nk}+W_N^{2nk}\sum_{n=0}^{\left(\frac{N}{2}\right)-1} x(2n+1)W_{\frac{N}{2}}^{nk} \quad (2.7)$$

Perhitungan FFt ini memiliki hasil bilangan kompleks, sehingga persamaan ini dapat dinyatakan dalam *magnitude* atau *phase* pada persamaan (2.8) dan (2.9) (Prayoga *et al*., 2019).

$$magnitude\ |X| = \sqrt{R^2+I^2} \quad (2.8)$$

$$phase = tan^{-1}\left(\frac{1}{R}\right) \quad (2.9)$$

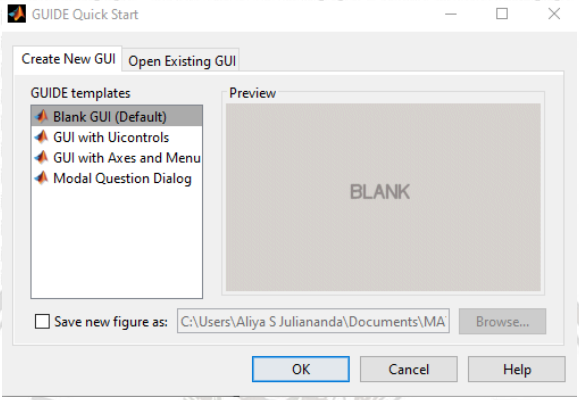
2.7. MATLAB

MATLAB (*Matrix Laboratory)* merupakan sebuah lingkungan komputasi numerikal dan bahasa pemrograman generasi keempat. MATLAB sendiri diciptakan akhir tahun 1970-an oleh Cleve Moler dan kemudian dikembangkan oleh The MathWorks, MATLAB memungkinkan manipulasi matriks, melakukan plot fungsi dan data, implementasi algoritma, pembuatan antarmuka pengguna, dan penggunaan antarmuka dengan program dalam bahasa lainnya (Hidayat, 2017).

Gambar 2. 6. Logo MATLAB

Pada Matlab terdapat fasilitas GUIDE (GUI *builder*). GUI merupakan singkatan *Graphical User Interface*. Pembuatan GUI akan memudahkan pengguna dalam menggunakan program. Ada

dua langkah utama dalam pembuatan GUI, yaitu mendesain tata letaknya dan menulis fungsi *callback* yang dapat melalukan operasi ketika pengguna memilih fitur yang berbeda (Hunt *et al*., 2006)



Gambar 2. 7. Tampilan untuk memulai GUI

BAB III
METODE PENELITIAN

3.1. Waktu dan Tempat Pelaksanaan

Penelitian ini dilaksanakan pada Maret 2019 sampai Mei 2019 di kediaman penulis (Jl. Gajayana). Selain itu, penelitian ini juga berlangsung di Laboraturium Fisika Dasar Jurusan Fisika Universitas Brawijaya, Malang.

3.2. Alat

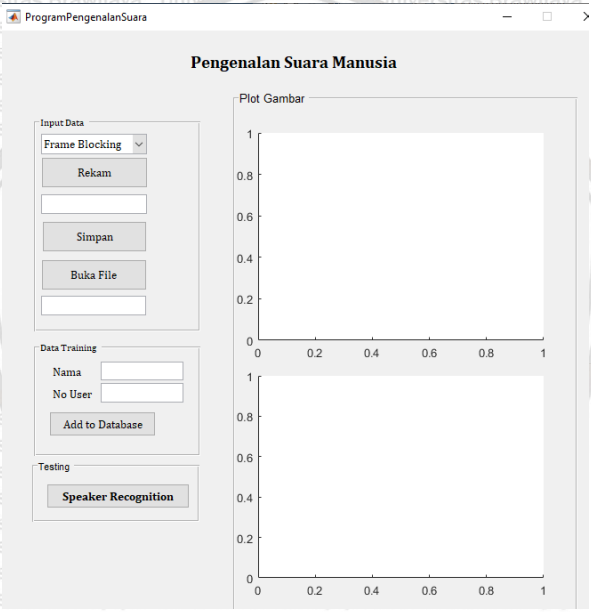
Penelitian ini menggunakan beberapa alat, diantaranya sebuah laptop Lenovo Ideapad 320 dengan spesifikasi berupa prosesor Intel Core i3-6006U @2.00GHz (4CPUs), RAM 4GB dan software MATLAB R2014a. Selain itu juga digunakan sebuah mikrofon sebagai alat bantu proses perekaman suara. Bahan yang digunakan pada penilitian ini adalah suara manusia.

3.3. Tahapan Penelitian

Sebelum memulai penelitian dilakukan instalasi MATLAB. Program pengenalan suara ini dilakukan dengan bantuan beberapa library di MATLAB, serta menggunakan antarmuka dalam MATLAB yaitu *Grap ichal User Interface* (GUI).

Program ini menggunakan GUI sebagai antarmuka. Gambar 3.1 menunjukkan tampilan antarmuka program *voice recogniion*. Program ini dimulai dengan memilih nilai *frame blocking* yang akan digunakan, kemudia untuk input suara dapat digunakan dua cara, yaitu merekam suara dan mengambil file suara yang tersedia. Suara yang diambil dalam format *.wav*. Pengguna juga dapat

menyimpan hasil rekam suara dengan menekan tombol “Simpan”. Plot gambar yang dihasilkan pada program berupa plot perekaman untuk kotak pertama dan plot FFT untuk kotak kedua. Saat pengambilan data pelatihan (*training)*, pengguna terlebih dahulu memasukkan nama dan No User. Sedangkan, saat melakukan *testing*, nantinya akan keluar *message box* output pengenalan suara.

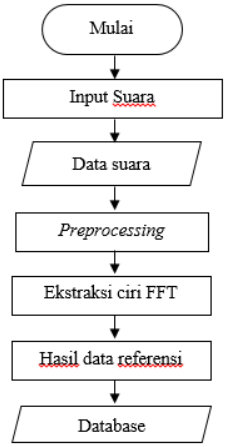


Gambar 3. 1. Rancang tampilan program *voice recognition*

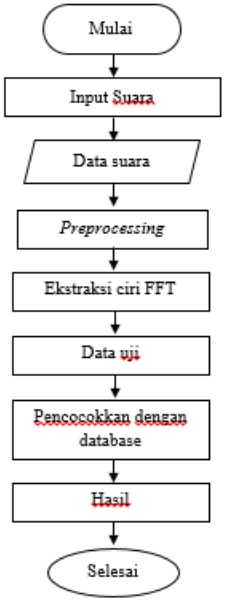
3.3.1. Proses Pengenalan Suara

Proses pengenalan suara ini dimulai dengan proses pelatihan terlebih dahulu sebelum melakukan proses pengujian. Proses pelatihan dilakukan untuk mendapatkan data referensi yang akan disimpan dalam database. Data referensi inilah yang nantinya akan digunakan sebagai pembanding data dari proses pengujian,

sehinga dapat ditentukan suara pengguna. Proses pencarian data referensi ditunjukkan pada Gambar 3.2, sedangkan proses pengujian data ditunjukkan pada Gambar 3.3.



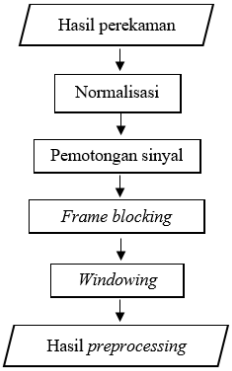
Gambar 3. 2 Diagram alir proses pelatihan



Gambar 3. 3 Diagram alir proses pengujian

Input data pada penelitian ini menggunakan lima pembicara (Bilal, Fariz, Renald, Wulan, dan Zul) Masing-masing pembicara mengucapkan “Selamat Pagi” sebanyak sepuluh kali. Suara yang dihasilkan akan digunakan pada proses berikutnya.

Setelah didapatkan data input, maka dilakukan tahap *preprocessing*. *Preprocessing* merupakan tahapan yang dilakukan untuk menyertakan sinyal suara agar mudah diproses untuk pengenalan suara. Tahapan pada proses ini diantaranya, normalisasi, pemotongan sinyal, *frame blocking*, dan *windowing*.



Gambar 3. 4. Diagram alir subproses *preprocessing*

3.3.2. Analisa Performa

Analisa performa bertujuan untuk mengetahui akurasi dari sistem yag telah dirancang bekerja akurat atau tidak akibat dari metode yang digunakan yaitu FFT. Keakuratan ini juga dilihat dari pengubahan panjang dari FFt dan nilai *frame blocking* yang digunakan. untuk mengetahui tingkat keakuratan dari program yang telah dibuat dapat mengguanakan persamaan berikut:

$$Akurasi = \frac{jumlah\ suara\ yang\ dikenali}{banyaknya\ percobaan} \times 100\% \quad (3.1)$$

BAB IV
HASIL DAN PEMBAHASAN

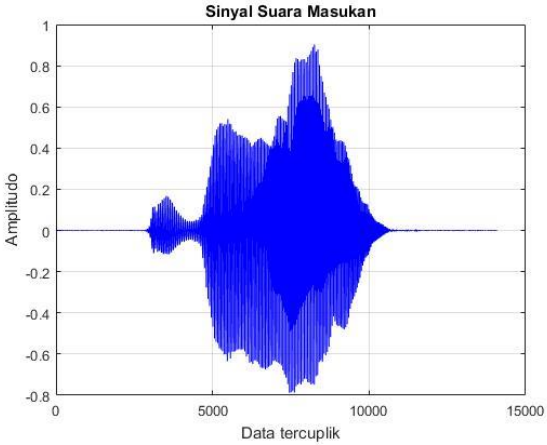
4.1. Preprocessing

Untuk mendapatkan hasil yang baik, sebelum melakukan perhitungan FFT, sinyal suara masukkan terlebih dahulu melalui proses *preprocessing*. *Preprocessing* ini dilakukan untuk memperbaiki kualitas objek. Pada penelitian ini, pengambilan sinyal suara masukan dapa dilakukan cara yaitu perekaman langsung ataupun pengambilan file suara. Untuk perekaman suara melalui proses *sampling* terlebih dahulu, dimana frekuensi sample yang digunakan adalah sebesar 16000 Hz.

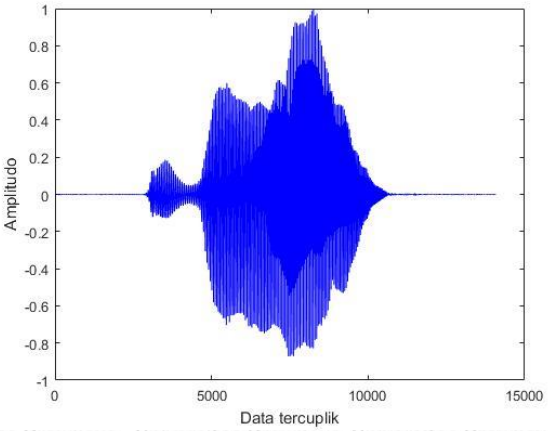
Untuk pengambilan file, format yang digunakan berbentuk *.wav*. Frekuensi sampling yang digunakan pada pengambilan file ini seusai dengan alat perekamnya. Umumnya alat perekam akan bekerja pada kecepatan sampling 44,1 kHz, 48 kHz, 88,2 kHz, atau 96 kHz. Akan tetapi pada kecepatan sekitar 50 kHz keatas tidak memberikan informasi yang signifikan bagi pendengar. Yang paling umum digunakan adalah pada frekuensi 44,1 kHz, karena pada frekuensi ini mampu memberikan frekuensi maksimum hingga 20kHz.

Gambar 4. 1 menunjukkan contoh plot sinyal suara input. Sinyal tersebut kemudian di normalisasi dengan cara membagi nilai maksimum sinyal dengan nilai absolut maksimal suara yang terekam tersebut. Dapat dilihat pada Gambar 4. 2. amplitudo suara terekam sudah setara dan menjadi maksimum. Proses ini

dilakukan agar efek kuata atau lemahnya sinyal suara terekam tidak mempengaruhi proses selanjutnya.



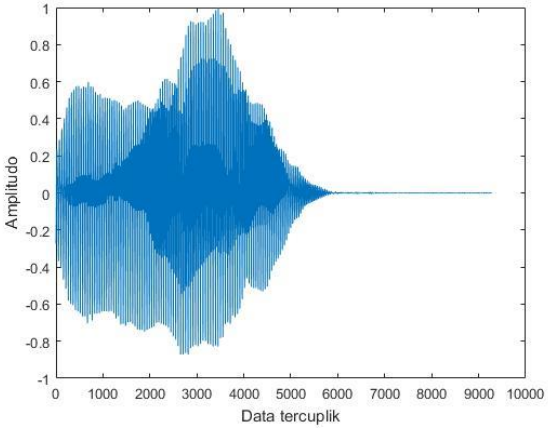
Gambar 4. 1 Plot sinyal suara



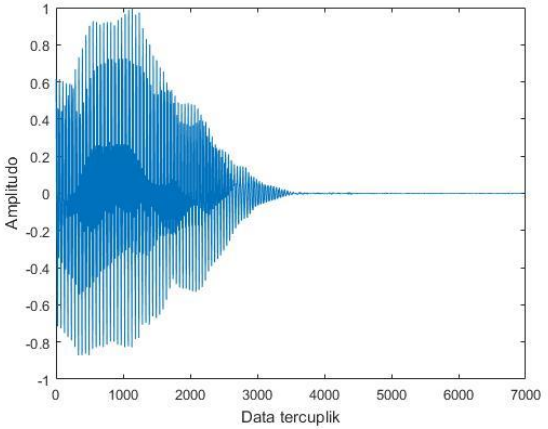
Gambar 4. 2 Plot Sinyal Normalisasi

Sinyal suara yang sudah dinormalisasi kemudian dipotong dalam dua tahapan. Tahapan pertama yaitu pemotongan bagian

silence. Pemotongan ini dilakukan untuk memotongan bagian yang bukan merupakan bagian dari sinya suara manusia. Tahapan ini dapat dilihat hasilnya pada Gambar 4. 3. Tahapan kedua adalah proses pemotongan sinyal transisi. Proses ini dilakukan dengan menghilangkan seperempat bagian sinyal yang ada diawal. Hasil pemotongan dapat dilihat pada Gambar 4. 4



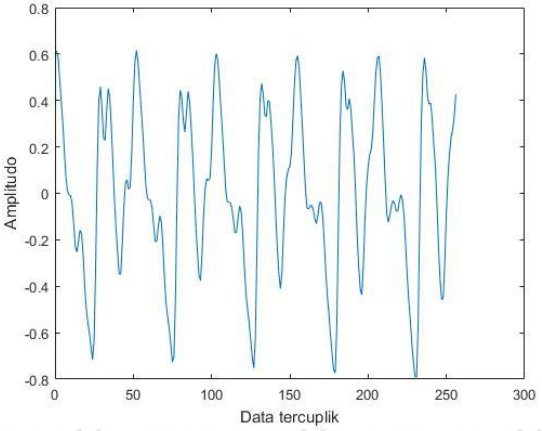
Gambar 4. 3 Plot Hasil Pemotongan Sinyal Suara



Gambar 4. 4 Plot Hasil Pemotongan Sinyal Transisi

Kemudian dilakukan pengambilan data sesuai dengan panjang nilai *frame blocking* yang telah dipilih. Proses ini digunakan untuk memilih data dari keseluruhan data yang terekam dari hasil pemotongan sinyal. Nilai yang dipilih bertujuan untuk mengurangi jumlah data sinyal yang akan diproses.

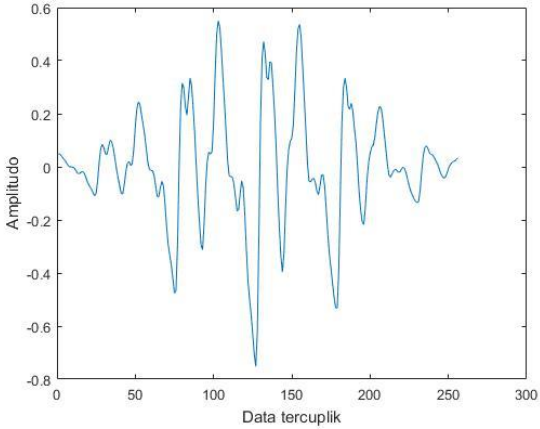
Proses ini diawali dengan menentukan nilai tengah dari *sampling*. Dari titik tengah yang diperoleh akan ditentukan besar data yang diambil untuk proses berikutnya. Hasil dari proses ini dapat dilihat pada Gambar 4. 5.



Gambar 4. 5 Plot hasil *frame blocking*

Suara yang dipotong-potong dapat membuat suara menjadi *discontinue* pada awal dan akhir tiap *frame*. Untuk menghilangkan efek diskontinuitas ini dilakukan proses *windowing*. Proses yang digunakan yaitu proses *hamming*

window yang sudah ada dalam toolbox *signal processing* Matlab. Dalam prosesnya hasil dari *hamming* dikalikan dengan hasil perhitungan *frame blocking*. Hasil proses *windowing* (lihat Gambar 4. 6) inilah nantinya yang akan digunakan pada proses perhitungan sinyal FFT.



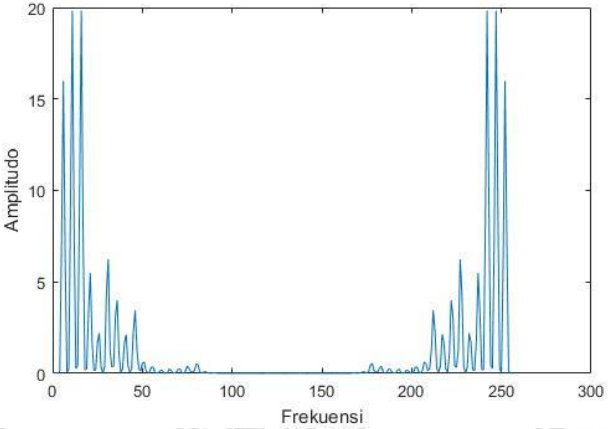
Gambar 4. 6 Plot hasil *windowing*

4.2. Hasil Ektraksi Ciri FFT

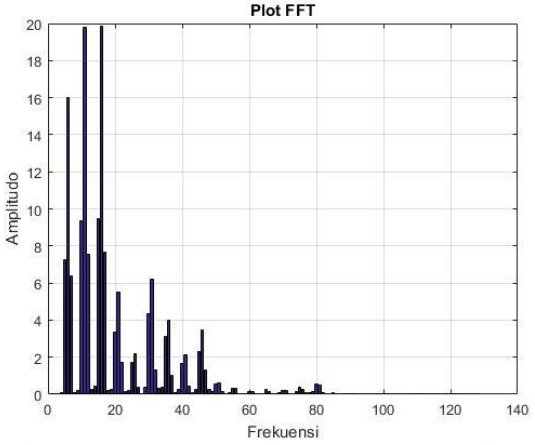
Perhitungan FFT digunakan untuk mencari nilai maksimum yang digunakan sebagai pola pengenalan baik untuk data *training* maupun data *uji*. Hasil dari ekstraksi ciri inilah yang akan dianalisi untuk mengetahui ciri dari suara pembicara. Pada proses pelatihan output yang didapat akan dimasukkan ke dalam database sebagai data referensi. Pada proses pengujian, nilai maksimum dibandingankan dengan data referensi sehingga pemilik suara nantinya dapat dikenali.

Gambar 4. 7 plot merupakan tampilan hasil ekstraksi ciri FFT. Hasil tersebut kemudian dipotong sebanyak setengah ukuran

sinyal yang ditentukan. Hasil pemotongan dilakukan untuk memilih bagian yang akan diproses. Sinyal tersebut kemudia ndiubah dimensinya sebagai *bar* yang ditunjukkan pada Gambar 4. 8. Undakan yang paling tinggi pada plot sinyal dilihat sebagai indeks nilai maksimum pada plot tersebut. Nilai inilah yang menjadi acuan dalam proses pengenalan suara.



Gambar 4. 7 Tampilan plot hasil FFT

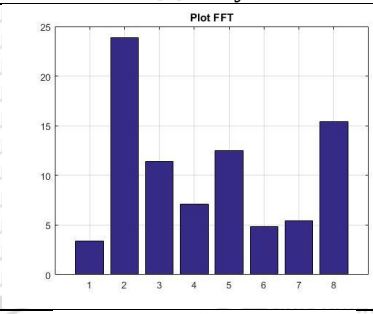
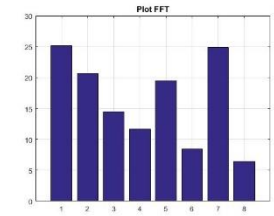
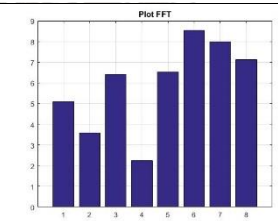
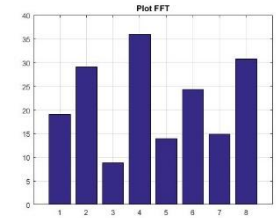
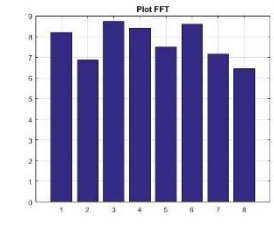


Gambar 4. 8. Tampilan sinyal FFT dari nilai maksimum

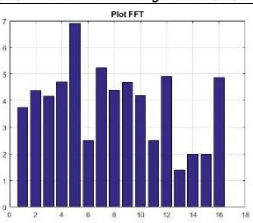
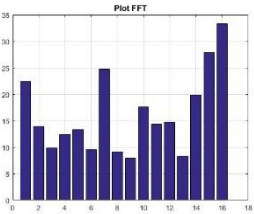
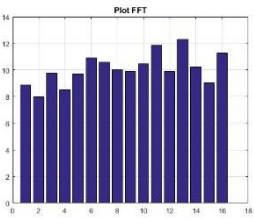
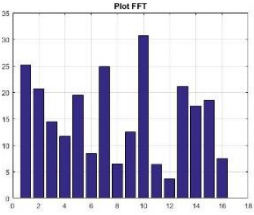
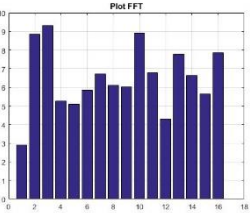
Tabel 4.1 menunjukkan salah satu plot sinyal FFT masing-masing pembicara dengan *frame blocking* 16. Dilihat pada table tersebut rentang nilai maksimum memiliki selisih yang sedikit. Begitupun pada Tabel 4.2 nilai indeks nilai maksimum masih belum terelihat selisih yang signifikan. Pada Tabel 4.3 mulai terlihat selisih yang signifikan. Nilai selisih yang sedikit ini dapat menyebabkan nilai error akan semakin besar. Hal ini disebabkan rentang selisih yang sedikit dapat mengakibatkan kesalahan dalam mengenal suara pembicara.

Nilai indeks maksimum terlihat sangat berbeda pada Tabel Tabel 4.4 dan Tabel 4.5. Rentang selisih yang didapat cukup jauh, Perbedaan rentang inilah yang dijadikan acuan karakter dari masing-masing suara. Semakin besar nilai selisihnya, maka kemungkinan error akan lebih sedikit.

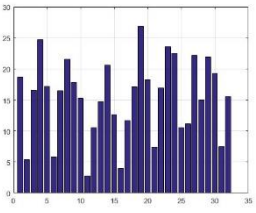
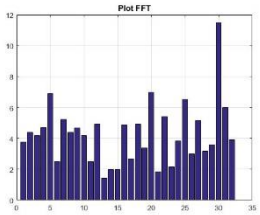
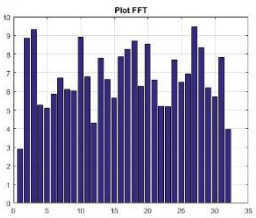
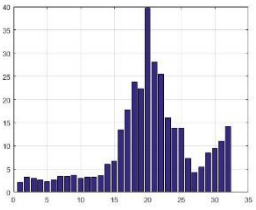
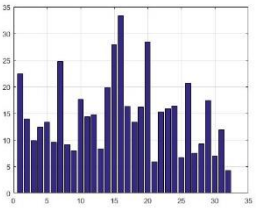
Tabel 4. 1 Plot Sinyal FFT pada *frame blocking* 16

Pembicara	Plot Sinyal
Bilal	
Renald	
Fariz	
Wulan	
Zul	

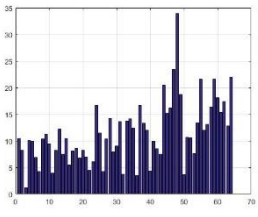
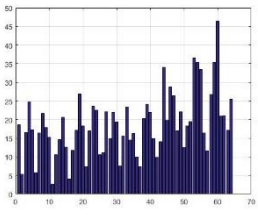
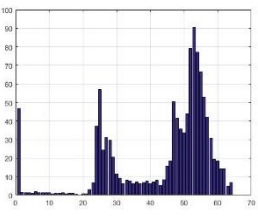
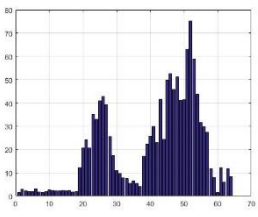
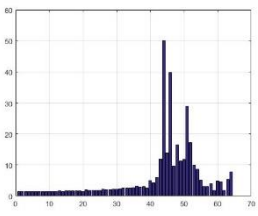
Tabel 4. 2 Plot Sinyal FFT pada *frame blocking* 32

Pembicara	Plot Sinyal
Bilal	
Renald	
Fariz	
Wulan	
Zul	

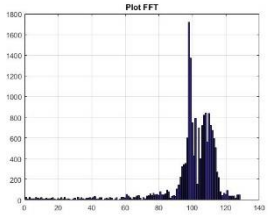
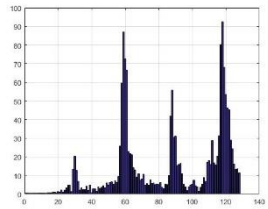
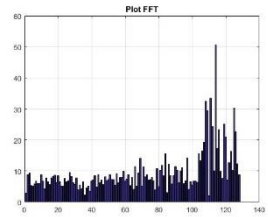
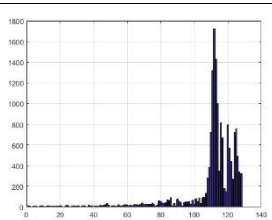
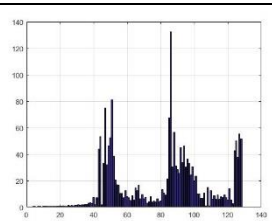
Tabel 4. 3 Plot Sinyal FFT pada *frame blocking* 64

Pembicara	Plot Sinyal
Bilal	
Renald	
Fariz	
Wulan	
Zul	

Tabel 4. 4 Plot Sinyal FFT pada *frame blocking* 128

Pembicara	Plot Sinyal
Bilal	
Renald	
Fariz	
Wulan	
Zul	

Tabel 4. 5 Plot Sinyal FFT *frame blocking 256*

Pembicara	Plot Sinyal
Bilal	
Renald	
Fariz	
Wulan	
Zul	

4.3. Proses pelatihan

Proses pelatihan merupakan proses analisis suara yang hasilnya akan dijadikan acuan untuk mengenali suara. Pada penelitian ini proses pelatihan dilakukan dengan mengambil sepuluh sampel suara untuk setiap pembicara. Setiap sampel dianalisis pada tiap nilai *frame blocking* yang dipilih. Hasil dari proses pelatihan ini dimasukkan kedalam database. Gambar 4.9 menunjukkan database yang telah masuk dalam *workspace* Matlab. Database ini berisikan "No User" dari pemilik suara yang diinisialisasi dengan C dan output indeks nilai maksimum FFT yang diinisialisasi dengan F.

Gambar 4. 9 Database

Database tersebut dibedakan untuk masing-masing *frame blocking* yang dipilih. Oleh karena itu hanya berisikan 50 data untuk setiap database, dimana ada lima pembicara dengan masing-masing sepuluh sampel suara. Jika dibuat dalam bentuk tabel, maka nilai output FFT yang didapat dapat dilihat pada tabel berikut ini.

Tabel 4. 6 Indeks Nilai Maksimum *Frame Blocking 16*

Suara	Pengambilan Ke-									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Bilal	3	2	3	3	3	2	3	3	3	2
Fariz	6	5	6	5	5	6	5	5	5	5
Renald	7	6	7	6	6	6	6	6	6	7
Wulan	4	3	3	3	3	4	3	3	3	4
Zul	1	2	1	1	2	2	1	2	1	2

Tabel 4. 7 Indeks Nilai Maksimum *Frame Blocking 32*

Suara	Pengambilan Ke-									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Bilal	9	8	9	9	9	9	8	9	9	8
Fariz	12	12	12	12	13	14	12	12	13	12
Renald	15	13	14	15	14	16	13	15	15	13
Wulan	10	10	10	11	10	11	10	10	10	11
Zul	7	6	7	8	6	8	7	8	6	7

Tabel 4. 8 Indeks Nilai Maksimum *Frame Blocking 64*

Suara	Pengambilan Ke-									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Bilal	17	18	17	18	18	16	18	18	17	18
Fariz	25	26	25	25	25	26	25	25	25	26
Renald	29	29	28	29	27	29	29	26	29	28
Wulan	23	22	22	24	24	23	22	23	23	23
Zul	15	16	14	15	15	16	15	17	15	15

Jika dilihat pada Tabel 4.6 indeks nilai maksimum antara satu suara dengan suara lainnya memiliki nilai yang tidak jauh berbeda. Bahkan dibeberapa kali pengambilan, nilai yang didapat

beberapa pembicara memiliki nilai yang sama. Seperti Bilal yang indeks nilai maksimumnya sama seperti Wulan dan Zul ataupun Fariz dan Renald yang nilai indeksnya tidak jauh berbeda.

Begitupun pada Tabel 4.7 indeks nilai yang diapat belum memperlihatkan perbedaan yang signifikan. Beberapa pembicara masih berada direntang nilai yang berdekatan. Pada Tabel 4.8 mulai terlihat perbedaan nilainya. Masing-masing suara pada proses pengambilan memiliki rentang yang berbeda, hanya saja selisih yang didapat belum terlalu jauh berbeda. Sehingga kemungkinan error masih cukup besar.

Tabel 4. 9 Indeks Nilai Maksimum *Frame Blocking* 128

Suara	Pengambilan Ke-									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Bilal	49	49	49	47	48	48	49	48	49	49
Fariz	56	57	56	56	56	56	56	57	56	56
Renald	62	60	61	60	62	61	64	60	60	61
Wulan	52	52	53	52	53	53	53	53	52	53
Zul	46	45	45	46	46	46	46	44	45	46

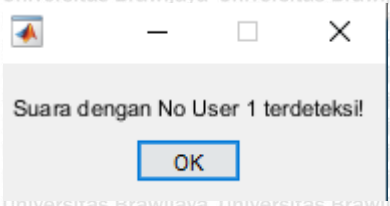
Tabel 4. 10 Indeks Nilai Maksimum *Frame Blocking* 256

Suara	Pengambilan Ke-									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Bilal	92	92	92	91	92	91	91	90	91	90
Fariz	116	118	116	119	116	116	117	116	116	116
Renald	120	123	120	121	121	121	123	121	121	122
Wulan	109	110	109	109	109	108	109	109	110	109
Zul	84	86	84	85	85	84	85	84	86	84

Mulai terlihat perbedaan signifikan pada indeks nilai maksimum pada Tabel 4.9 dimana setiap pembicara memiliki nilai yang berbeda, selisih nilainya juga cukup jauh. Selisih yang lebih besar dapat dilihat pada Tabel 4.10. Pola suara yang didapat terlihat sangat berbeda pada saat *frame blocking* 128 dan 256 dibandingan pada saat *frame blocking* 16, 32, ataupun 64. Hal inilah yang dapat membedakan pemilik suara saat proses pengenalan suara. Perbedaan pola suara yang didapat pembicara satu dan lainnya dapat dijadikan acuan pada proses pengujian.

4.4. Pengujian dan Analisa Performa

Analisa ini dilakukan untuk membuktikan kemampuan program pengenalan suaran ini. Pengujian ini dilakukan pada suara yang telah melakukan proses pelatihan, yaitu suara Bilal, Renald, Fariz, Wulan, dan Zul. Pengambilan data uji ini juga dilakukan sebanyak sepuluh kali masing-masing suara tiap nilai *frame blocking*. Proses yang dilakukan pada pengujian juga sama dengan proses pelatihan. Kemudian, hasil indeks maksimum yang didapat dicocokkan dengan nilai yang ada pada database. Output yang dihasilkan berupa No User pengguna. Pada program ini output berupa *message box* yang dapat dilihat pada Gambar 4.10.



Gambar 4. 10 Tampilan *message box* pengenalan suara

Tabel Tabel 4. 11 Menunjukkan keakuratan dalam proses pengenalan suara. Dapat dilihat pada table tersebut presentase keakuratan dalam pengenalan suara akan semakin besar ketika nilai *frame blocking* semakin tinggi. Sesuai dengan penjelasan sebelumnya bahwa saat *frame blocking* bernilai kecil kemungkinan indeks nilai maksimum yang didapatkan sama karena rentang yang sedikit. Sedangkan saat *frame blocking* semakin besar, indeks nilai maksimum akan lebih terlihat perbedaan yang signifikan. Oleh karena itu, jika dilihat pada tabel tersebut, Renald, Wulan, Zul memiliki jumlah pengenalan yang lebih sedikit dibandingkan dengan Bilal dan Fariz terutama pada *frame blocking* 16, 32, dan 64. Hal ini dikarenakan data Bilal dan Fariz yang telah masuk lebih dahulu dibandingkan Renald, Wulan, dan Zul, sehingga apabila indeks nilai ketiga suara ini sama dengan indeks nilai data yang terlebih dahulu masuk, maka suara tersebut akan dikenali sebagai pemilik suara pada data yang lebih dahulu.

Tabel 4. 11. Hasil pengujian proses pengenalan suara

Frame Blocking	Jumlah Pengenalan Suara					Presentase Pengenalan (%)
	Bilal	Fariz	Renald	Wulan	Zul	
16	10	8	1	3	0	44
32	10	8	2	3	1	48
64	10	9	5	5	2	62
128	10	10	8	10	7	90
256	10	10	9	10	8	94

BAB V
PENUTUP

5.1. Kesimpulan

Kesimpulan yang didapat dari hasil penelitian system pengenalan suara manusia ini adalah:

1. Perancangan sistem pengenalan suara ini dimulai dengan proses *training* suara terlebih dahulu sebagai data referensi dan proses *testing* sebagai data uji. Untuk mendapatkan kedua data ini melalu proses yang sama yaitu perekaman suara, *preprocessing*, dan ekstraksi ciri FFT. Ekstraksi ciri ini dilakukan untuk mendapatkan nilai indeks maksimum yang dicari sebagai hasil *output.*
2. Program pengenalan suara menghasilkan tingkat keakuratan paling baik sebesar 94% yang diperoleh dari sepuluh kali pengambilan suara untuk setiap suara pada saat nilai *frame blocking* 256.

5.2. Saran

Program pengenalan suara ini masih jauh dari kata sempurna diharapkan adanya pengembangan system ini agar menjadi semakin baik. Beberapa saran untuk penelitian selanjutnya adalah dengan menggunakan metode lain yang dapat dibandingkan dengan keakuratan hasil pengenalan suara.

DAFTAR PUSTAKA

Adler, J., Azhar, M., & Supatmi, S. (2013). Identifikasi Suara dengan MATLAB sebagai Aplikasi Jaringan Syaraf Tiruan. *Telekontran*, *1*(1), 16–23.

Ariyanto, E., & H, F. S. (2014). Identifikasi dan Aplikasi Pengenalan Spektrum Bunyi Gamelan Menggunakan Jaringan Syaraf Tiruan Pada Matlab. *Neutrino*, *7*(1), 7–15.

Bhaskoro, S. B., & D, A. R. W. (2012). Aplikasi Pengenalan Gender Menggunakan Suara. *SNATI*, 16–23.

Faradiba. (2017). Pengenalan Pola Sinyal Suara Manusia Menggunakan Metode Back Propagation Neural Network. *EduMatSains*, *2*(1), 1–15.

Hidayat, R. (2017). *MATLAB pada Sistem Pemrosesan Sinyal dan Komunikasi Digital: Simulasi berbagai Aplikasi Teknik*. Malang: Penerbit Gunung Samudera.

Hunt, B. R., Lipsman, R. L., Rosenberg, J. M., Coombes, K. R., Osborn, J. E., & Stuck, G. J. (2006). *A Guide to MATLAB: For Beginners and Experienced Users* (Second). Cambridge: Cambridge University Press.

Izzah, N. (2018). Klastering Suara Berdasarkan Gender Menggunakan Algoritma K-Means Dari Hasil Ekstraksi FFT (Fast Fourier Transform). *Soulmath*, *6*(1), 47–58.

Mustofa, A. (2018). *Pengolahan Sinyal Digital*. Malang: UB Press.

Prayoga, N. F. I., Astuti, Y., & Waluyo, C. B. (2019). Analisis Speaker Recognition menggunakan Metode Dynamic Time Warping

(DTW) Berbasis Matlab. *AVITEC*, *1*(1), 77–85.

Shelly, G. B., Cashman, T. J., & Vermaat, M. E. (2007). *Discovering Computers* (3rd ed.). Jakarta: Penerbit Salemba.

Sibarani, R. A. L. (2018). *Identifikasi Sinyal Suara menggunakan Metode Fast Fourier Transform (*FFT*) Berbasis MATLAB*. Universitas Sumatera Utara.

Singh, N. (2015). Speaker Recognition and Fast Fourier Transform. *International Journal of Advanced Research in Computer Sciens and Software Engineering*, *5*(7), 530–534. https://doi.org/10.13140/RG.2.1.2722.0969

Sipasulta, R. Y., Lumenta, A. S. M., & Sompie, S. R. U. A. (2014). Simulasi Sistem Pengacak Sinyal Dengan Metode FFT (Fast Fourier Transform). *E-Journal Teknik Elektro Dan Komputer*, 1–9.

Syaifuddin, A., & Suryono. (2014). Fast Fourier Transform (FFT) untuk Analisis Sinyal Suara Doppler Ultrasonik. *Youngster Physics Journal*, *3*(3), 181–188.

LAMPIRAN

Kode Program

```
function varargout =
ProgramPengenalanSuara(varargin)
% PROGRAMPENGENALANSUARA MATLAB code for
ProgramPengenalanSuara.fig
%      PROGRAMPENGENALANSUARA, by itself,
creates a new PROGRAMPENGENALANSUARA or raises
the existing
%      singleton*.
%
%      H = PROGRAMPENGENALANSUARA returns the
handle to a new PROGRAMPENGENALANSUARA or the
handle to
%      the existing singleton*.
%
%
PROGRAMPENGENALANSUARA('CALLBACK',hObject,eventD
ata,handles,...) calls the local
%      function named CALLBACK in
PROGRAMPENGENALANSUARA.M with the given input
arguments.
%
%
PROGRAMPENGENALANSUARA('Property','Value',...)
creates a new PROGRAMPENGENALANSUARA or raises
the
%      existing singleton*.  Starting from the
left, property value pairs are
%      applied to the GUI before
ProgramPengenalanSuara_OpeningFcn gets called.
An
%      unrecognized property name or invalid
value makes property application
%      stop.  All inputs are passed to
ProgramPengenalanSuara_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 ProgramPengenalanSuara
% Last Modified by GUIDE v2.5 19-Jun-2020
10:30:28

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',
mfilename, ...
                   'gui_Singleton',
gui_Singleton, ...
                   'gui_OpeningFcn',
@ProgramPengenalanSuara_OpeningFcn, ...
                   'gui_OutputFcn',
@ProgramPengenalanSuara_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
ProgramPengenalanSuara is made visible.
function
ProgramPengenalanSuara_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
ProgramPengenalanSuara (see VARARGIN)

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

% Update handles structure
guidata(hObject, handles);

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


% --- Outputs from this function are returned to
the command line.
function varargout =
ProgramPengenalanSuara_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)
nama=get(handles.edit2,'String');
uno=str2num(char(get(handles.edit1,'String')));

y = handles.y;

%%normalisasi
y1=y/max(y);
% figure(1);
% plot(y1);

%%batas potong
b0=0.3;

%%pemotongan sinyal
b1=find(y1>b0 | y1<-b0);
y1(1:b1(1))=[];
% figure(2)
% plot(y1)

bts=floor(0.25*length(y1));
y1(1:bts)=[];
% figure(3)
% plot (y1)

%%frame blocking
frame=handles.frame;
y2=y1(1:frame);
% figure(4);
% plot(y2);

%% Windowing
h=hamming(frame);
y3=y2.*h;
% figure(5);
% plot(y3);

%%FFT
y4=abs(fft(y3));
```

```
% figure(6);
% plot(y4);
y5=y4(1:frame/2);
m=max(y5);
f=find(y5==m,1);
axes(handles.axes2)
% figure(7)
plot(y5);
bar(y5);
grid on
title('Plot FFT')

nama=get(handles.edit2,'String');
uno=str2num(char(get(handles.edit1,'String')));
try
    load training_database
    F=[F;f];
    C=[C;uno];
    save training_database F C
catch
    F=f;
    C=uno;
    save training_database F C
end
msgbox('Data disimpan!');

function edit1_Callback(hObject, eventdata,
handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    structure with handles and user
data (see GUIDATA)

% Hints: get(hObject,'String') returns contents
of edit1 as text
%        str2double(get(hObject,'String'))
returns contents of edit1 as a double

% --- Executes during object creation, after
setting all properties.
```

```
function edit1_CreateFcn(hObject, eventdata,
handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    empty - handles not created until
after all CreateFcns called

% Hint: edit controls usually have a white
background on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit2_Callback(hObject, eventdata,
handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    structure with handles and user
data (see GUIDATA)

% Hints: get(hObject,'String') returns contents
of edit2 as text
%        str2double(get(hObject,'String'))
returns contents of edit2 as a double


% --- Executes during object creation, after
setting all properties.
function edit2_CreateFcn(hObject, eventdata,
handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    empty - handles not created until
after all CreateFcns called
```

```
% Hint: edit controls usually have a white
background on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- 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)
%%rekam suara
Fs=4800;
nBits=8;
nChannel=2;
recObj = audiorecorder(Fs,nBits,nChannel);
set(handles.edit3, 'String', 'Mulai...')
recordblocking(recObj, 3);
set(handles.edit3,'String', 'Selesai')
y = getaudiodata(recObj);
%plot rekaman
axes(handles.axes1)
plot(y);
grid on
title('Sinyal Suara Masukan')

handles.y=y;
handles.Fs=Fs;
guidata(hObject,handles);

% % --- 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)
% %%
% y=handles.y;
% Fs=handles.Fs;
% sound(y);

% --- 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)
%% menampilkan menu browse file
[filename,pathname] = uigetfile('*.wav');
% jika ada file yang dipilih maka akan
mengeksekusi perintah di bawah ini
if ~isequal(filename,0)
    % membaca file sinyal suara
    %Fs=7200;
    [y,Fs] =
audioread(fullfile(pathname,filename));
    set(handles.edit4,'String',filename)
    % menampilkan plot sinyal suara pada axes
    axes(handles.axes1);
    plot(y)
    grid on
    title('Sinyal Suara Masukan')
    set(gca,'YLim')
else
    % jika tidak ada file yang dipilih maka akan
kembali
    return
end
handles.y = y;
handles.Fs = Fs;
guidata(hObject, handles)
```

```
function edit3_Callback(hObject, eventdata,
handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    structure with handles and user
data (see GUIDATA)

% Hints: get(hObject,'String') returns contents
of edit3 as text
%        str2double(get(hObject,'String'))
returns contents of edit3 as a double


% --- Executes during object creation, after
setting all properties.
function edit3_CreateFcn(hObject, eventdata,
handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    empty - handles not created until
after all CreateFcns called

% Hint: edit controls usually have a white
background on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit4_Callback(hObject, eventdata,
handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
```

```
% handles    structure with handles and user
data (see GUIDATA)

% Hints: get(hObject,'String') returns contents
of edit4 as text
%        str2double(get(hObject,'String'))
returns contents of edit4 as a double


% --- Executes during object creation, after
setting all properties.
function edit4_CreateFcn(hObject, eventdata,
handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    empty - handles not created until
after all CreateFcns called

% Hint: edit controls usually have a white
background on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in
popupmenu1.
function popupmenu1_Callback(hObject, eventdata,
handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    structure with handles and user
data (see GUIDATA)
indeks=get(handles.popupmenu1,'Value');
switch indeks
    case 2
        frame_blocking=16;
    case 3
        frame_blocking=32;
```

```
    case 4
        frame_blocking=64;
    case 5
        frame_blocking=128;
    case 6
        frame_blocking=256;
end
handles.frame=frame_blocking;
guidata(hObject,handles);

% Hints: contents =
cellstr(get(hObject,'String')) returns
popupmenu1 contents as cell array
%        contents{get(hObject,'Value')} returns
selected item from popupmenu1


% --- Executes during object creation, after
setting all properties.
function popupmenu1_CreateFcn(hObject,
eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a
future version of MATLAB
% handles    empty - handles not created until
after all CreateFcns called

% Hint: popupmenu controls usually have a white
background on Windows.
%       See ISPC and COMPUTER.
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- 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)
y = handles.y;

%%batas potong
b0=0.3;
%%normalisasi
y1=y/max(y);

%%pemotongan sinyal
b1=find(y1>b0 | y1<-b0);
y1(1:b1(1))=[];

bts=floor(0.25*length(y1));
y1(1:bts)=[];

%%frame blocking
frame=handles.frame;
y2=y1(1:frame);

%% Windowing
h=hamming(frame);
y3=y2.*h;

%%FFT
y4=abs(fft(y3));
y5=y4(1:frame/2);
axes(handles.axes2)
plot(y5);
bar(y5);
m=max(y5);
f=find(y5==m,1);

load training_database
D=[];
for (i=1:size(F,1))
    d=sum(abs(F(i)-f));
    D=[D d];
end
sm=inf;
```

```
ind=-1;
for (i=1:length(D))
    if (D(i)<sm)
        sm=D(i);
        ind=i;
    end
end
detected_class=C(ind);
msgbox(strcat(['Suara dengan No User
',num2str(detected_class),' terdeteksi!']),


% --- 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)
%% menampilkan menu save file
[filename, pathname] = uiputfile('*.wav');
% jika ada file yang disimpan maka akan
mengeksekusi perintah di bawah ini
if ~isequal(filename,0)
    % membaca variabel Fs dan myRecording yang
ada di lokasi handles
    Fs = handles.Fs;
    y = handles.y;
    % menyimpan file sinyal suara
    audiowrite(fullfile(pathname,filename),y,Fs)
else
    % jika tidak ada file yang disimpan maka
akan kembali
    return
end
```