SKRIPSI

PENGENALAN SUARA DENGAN METODE *DYNAMIC TIME WARPING* UNTUK MENJALANKAN PROGRAM APLIKASI USER

Disusun Oleh :

ABDI SOFYANDI

NIM : 04.12.645

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
SEPTEMBER 2008

LEMBAR PERSETUJUAN

PENGENALAN SUARA DENGAN METODE *DYNAMIC TIME WARPING* UNTUK MENJALANKAN PROGRAM APLIKASI USER

SKRIPSI

Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Komputer & Informatika Strata Satu (S-1)

Disusun Oleh :

ABDI SOFYANDI
NIM : 04.12.645

Diperiksa dan Disetujui

Dosen Pembimbing I

I Komang Somawirata, ST, MT
NIP.Y.103 0100 361

Dosen Pembimbing II

M.Ashar, ST, MT
NIP.P.103 0500 408

Mengetahui
Ketua Jurusan Teknik Elektro S-1

Ir. F. Yudi Limpraptono, MT
NIP.Y.103 9500 274

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER & INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2008

LEMBAR PERSEMBAHAN

Inti tujuan hidupku telah aku hadapi dengan penuh semangat dan kerja keras, untuk mendapatkan sebenarnya hidup yang selama ini aku harapakan. Di jenjang inilah aku mendapatkan arti sebenarnya kerasnya menjalani kehidupan ini, tapi dengan dukungan dari lingkungan dan orang-orang terdekat, semuanya dapat aku lalui dengan lancar. Yang dulunya aku tak sama sekali banyak mengerti masalah arti hidup, sekarang aku bisa mengerti segalanya. Mulai dari dengan peraihan prestasi, kreativitas dan banyak lainnya. Tak lupa aku bersyukur kepada ALLAH S.W.T yang slama ini dengan petunjuknya aku bisa bertawakal dalam jalani semuanya.

Banyak hal yang bisa aku perbuat untuk membuat hidup ini lebih berarti, dan ini tidak lepas dari pengaruh banyak pihak dimana mungkin aku tidak akan bisa membalas jasa dan pengorbanan yang telah diberikan kepadaku. Terima kasih buat semuanya, hanya ini yang bisa aku persembahkan buat kalian. Terimalah ucapan persembahanku ini

- Yang maha segalanya ALLAH S.W.T yang telah memberikan makna dalam hal yang baik dan yang buruk.
- Kepada bapak, mamak, adik yenny dan rio beserta semua keluargaku yang tercinta, terima kasih atas dukungan moral & financial yang telah support abdi selama ini, semua itu sangat berarti buat abdi tuk jalani hidup dimasa akan datang.
- Bapak Dosenku Ashar,ST,MT , terima kasih pak atas bimbingan dan semuanya, bapaklah yang membuat aku mengerti bagaimana cara menghadapi masalah. Bapak telah mengajarkan saya berkerja sama dengan Tim serta dalam pencapain prestasi.
- Hai Tim ASAH, {Agit (Sogit), Sahru (Sao), Herru (Tonjang)}, sobatku.....aku gak tau harus mau bilang apa sama kalian, tanpa kalian aku mungkin bukan apa-apa, kalian sangat berperan dalam kehidupan ku beberapa tahun ini dalam jalani jenjang ini. kalian

mengajarkan banyak hal dari mulai yang memarahi aku trus diwaktu kita tertawa bersama.... Sobatku aku gak mau Tim kita ini sampai disini aja. Aku sangat cinta dengan Tim kita ini harapanku adalah Tim kita akan menjadi yang terbaik dan solid serta bisa mewujudkan cita-cita kita, perjuangan belum berakhir, ini adalah awal perjuangan sesungguhnya sobatku.....!!!

- Pujaan hatiku, Ruri.. Terima kasih sudah sabar menemaniku dan mendukungku dalam menyelesaikan semua ini, memberikan inspirasi dan tujuan hidupku, aku sangat berharap kamu juga bisa temani aku dalam menghadapi kehidupan aku yang masih panjang ini...amin ya Allah....!!!
- Kawan-kawan aku LAB Pemrograman Komputer & Multimedia angkatan 04,05,06,07 terima kasih ya atas semuanya...!!!
- Halo teman-teman aku seperjuangan Teknik Informatika 04' ITN Malang, terima kasih ya sudah mau kenal denganku.
- Kepada Seluruh civitas Dosen & Karyawan ITN Malang.
- Tak lupa pula pada keluarga besar Kos Satria dan Kos VB-16 di Malang, terima kasih atas diterimanya aku ditempat kalian.

PENGENALAN SUARA DENGAN METODE *DYNAMIC TIME WARPING* UNTUK MENJALANKAN PROGRAM APLIKASI USER

Abdi Sofyandi
04.12.645
Jurusan Teknik Elektro,Konsentrasi Teknik Komputer & Informatika,
Fakultas Teknologi Industri, Institut Teknologi Nasional, Jl. Karanglo KM.
2, Tasik Madu Malang, abdy_50fl4n@yahoo.com.
Dosen Pembimbing : I. I Komang Somawirata, ST, MT
II. M.Ashar, ST, MT

Kata Kunci : *Speech recognition, Sampling, frame blocking, windowing, FFT, cepstrum.*

Proses pengenalan suara (Speech Recognition) merupakan suatu teknik terapan dari proses pengolahan sinyal digital yang telah banyak digunakan untuk berbagai macam aplikasi, misalnya saja teknologi di bidang komputerisasi dan telekomunikasi sudah tidak hanya mampu menyediakan layanan pengiriman data text saja tetapi juga sudah mampu melayani pengiriman data dengan menggunakan suara, bahkan saat ini sudah banyak teknologi aplikasi yang menggunakan suara sebagai password untuk *login* dan lain-lain. Skripsi ini mencoba untuk membuat aplikasi terapan lain yang dapat memanfaatkan suara manusia yaitu membuat perintah untuk menjalankan aplikasi user komputer dengan menggunakan suara yang teknologinya sering disebut sebagai speech processing atau speech recognition.

Skripsi ini pada dasarnya memanfaatkan sinyal suara sebagai inputan, sinyal tersebut akan diekstrak oleh sistem sehingga diperoleh parameter-paremeter standart dari masing-masing sinyal suara yang masuk. Proses ekstrak sendiri terdiri dari proses sampling, frame blocking, windowing, FFT, dan cepstrum. Sinyal hasil ekstrak tersebut disimpan sebagai data standart (data referensi), sehingga pada saat ada sinyal/suara baru yang masuk, maka sistem akan memprosesnya (ekstraksi) dan akan membandingkan persamaan sinyal baru tersebut dengan sinyal standart yang sudah ada di database. Data dengan error yang terkecil diasumsikan mempunyai nilai yang sama dengan sinyal standart schingga diijinkan untuk mengakses sistem dan bisa menjalakan aplikasi user komputer yang disedikan oleh sistem.

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa yang telah memberikan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Aplikasi Pengenalan Suara Untuk Menjalankan Program Aplikasi User" ini dengan lancar. Skripsi ini merupakan persyaratan kelulusan Studi di Jurusan Teknik Elektro S-1 Konsentrasi Teknik Komputer & Informatika ITN Malang dan untuk mencapai gelar Sarjana Teknik.

Keberhasilan peyelesaian laporan skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Untuk itu penyusun menyampaikan terima kasih kepada :

1. Bapak Ir. F. Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S-1.
2. Bapak I Komang Somawirata, ST, MT selaku Dosen Pembimbing I
3. Bapak M.Ashar, ST, MT selaku Dosen Pembimbing II dan Ka. Laboratorium Pemrograman Komputer & Multimedia.
4. Ayah dan Ibu serta saudara-saudara kami yang telah memberikan do'a restu, dorongan, semangat, dan biaya.
5. Rekan-rekan Instruktur di Laboratorium Pemrograman Komputer & Multimedia.
6. Semua yang telah membantu dalam penyelesaian penyusunan skripsi ini.

Penyusun telah berusaha semaksimal mungkin dan menyadari sepenuhnya akan keterbatasan pengetahuan dalam menyelasaikan laporan ini. Untuk itu

penyusun mengharapkan saran dan kritik yang membangun dari pembaca demi kesempurnaan laporan ini.

Harapan penyusun semoga laporan skripsi ini memberikan manfaat bagi perkembangan ilmu pengetahuan dan pembaca.

Malang, September 2008

Penyusun

DAFTAR ISI

DAFTAR GAMBAR

BAB II TINJAUAN PUSTAKA

BAB III PERANCANGAN DAN ANALISA SISTEM

BAB IV IMPLEMENTASI DAN PENGUJIAN

DAFTAR TABEL

BAB IV IMPLEMENTASI DAN PENGUJIAN

BAB I

PENDAHULUAN

Sekarang ini perkembangan teknologi komputer dan telekomunikasi sudah sangat modern tidak terasa perkembangannya terus maju dalam waktu yang cepat. Sehingga sudah hampir semua kegiatan manusia dikerjakan oleh teknologi, terutama dibidang teknologi robotik. Robotik dijaman modern sekarang ini ada yang tingkah-lakunya sudah seperti manusia, robot dapat berkomunikasi layaknya manusia dengan manusia.

1.1 Latar Belakang

Dalam beberapa tahun terakhir ini dunia telekomunikasi maupun teknologi komputer mengalami perkembangan yang sangat pesat. Salah satunya adalah penelitian di bidang suara. Dewasa ini sinyal informasi yang dikirim tidak hanya berupa data text saja tetapi juga suara manusia. Banyak sekali teknologi saat ini yang memanfaatkan teknologi dalam bidang suara, salah satunya adalah robot. Sehingga gerakan-gerakan robot dapat diatur dengan suara manusia. Masih banyak persoalan yang terjadi ketika suara dimanfaatkan oleh sebuah sistem karena setiap orang memiliki ciri suara yang berbeda-beda. Suara merupakan modal utama yang dimiliki manusia untuk berkomunikasi dengan orang lain. Dengan suara manusia dapat memberikan informasi maupun perintah. Salah satu teknologi yang memanfaatkan suara adalah proses login atau password. Untuk mengatasi hal tersebut digunakan proses pengenalan suara yang dikeluarkan oleh manusia.

Pada dasarnya suara manusia dapat digunakan sebagai sumber informasi untuk menyatakan ide, keinginan dan perasaan kepada objek. Dengan itu objek akan mengerti apa yang diinginkan dan menjalakannya sesuai dengan intruksi yang diperintahkan. Untuk menciptakan suatu intruksi pengenalan suara yang dapat dimengerti, maka skripsi ini akan membuat sebuah sistem atau aplikasi yang memanfaatkan teknologi pengenalan suara *(Speech Recognition)*. Sistem ini diharapkan akan mengenali suara dari *Microphone* dan kemudian hasil dari pengenalan suara tersebut digunakan sebagai perintah untuk membuka aplikasi user.

1.2 Perumusan Masalah

Perumusan masalah yang diambil dalam pembuatan skripsi ini adalah

a) Bagaimana sebuah inputan berupa suara bisa menjadi sebuah instruksi untuk dapat menjalankan program aplikasi user sehingga dapat menggantikan instruksi dari aktifitas melalui *hardware*.
b) Penyesuaian sinyal suara yang masuk dengan sinyal yang ada pada database sehingga sinyal dapat dikenali sebagai instruksi untuk membuka program aplikasi.

1.3 Tujuan

Menfaatkan sinyal wicara agar dapat dikenali sebagai instruksi untuk menjalankan suatu program aplikasi user.

1.4 Batasan Masalah

Adapun batasan-batasan masalah yang diambil pada pembuatan Software aplikasi ini sebagai berikut:

- Bahasa pemrograman yang digunakan adalah bahasa C dengan didukung dengan Tcl/Snack2.2 pada Windows.
- Sinyal diasumsikan ideal, yaitu sinyal tidak bernoise.
- Feature aplikasi komputer yang bisa diakses hanya feature-feature yang terdaftar di database.
- Hanya bisa mengakses satu aplikasi dalam waktu yang sama.

1.5 Metodologi Penelitian

Skripsi ini dilakukan melalui beberapa tahap yaitu sebagai berikut:

1. Studi literatur

   Studi literatur yang dilakukan untuk terlaksananya skripsi ini yaitu tinjauan dari berbagai macam pustaka untuk mempelajari teori-teori yang terkait dan berhubungan dengan permasalahan melalui media Referensi Buku, dan Literatur dari Internet.

2. Perancangan

   Merancang sistem dengan melalui banyak metode yang digunakan, sesuai dengan pengelolahan sinyal suara untuk dapat dikenali oleh bahasa pemrograman.

   ❖ Perekaman suara sebagai sample

   Perekaman suara dari 5 pola tipe suara, perekaman dilakukan dengan berulang-ulang dengan kata yang berbeda. Kata-kata yang di rekam

merupakan feature yang ada pada komputer seperti : Ms Word, Excel, Power Point, Explorer dan Notepad.

- Pembuatan data standart

  Pembuatan data standart dari sinyal-sinyal suara yang telah diambil/direkam sebagai sinyal standart dari feature-feature aplikasi komputer yang akan digunakan.

- Proses Penyesuaian (*matching*)

  Penyesuain dan pengambilan data dari user pada database sehingga pada saat ada sinyal *independent* (sinyal baru) yang masuk dapat dicari nilai errornya. Data dengan nilai *error* terkecil diasumsikan mempunyai tipikal suara yang sama dengan sinyal suara standart dan akan diijinkan untuk melakukan akses pada komputer untuk membuka suatu program aplikasi

3. Pembuatan program

   Dalam sistem yang talah direncanakan maka bahasa pemrograman yang digunakan adalah bahasa C/C++ didukung dengan Tcl dan Snack sebagai pengelola suara.

4. Implementasi dan pengujian

   Implementasi dan pengujian dilakukan setelah sistem yang dirancang telah tercapai dengan mengasumsikan error yang terjadi.

1.6 Sistematika Penulisan

Pada penulisan skripsi ini terdiri atas lima bagian. Secara singkat, kelima bagian tersebut adalah sebagai berikut :

BAB I PENDAHULUAN

Pada bab ini akan diterangkan secara singkat mengenai latar belakang, tujuan dan manfaat, batasan masalah, dan sistematika pembahasan.

BAB II TEORI PENUNJANG

Dijelaskan mengenai teori-teori yang berkaitan dengan bahasan skripsi ini dan menunjang terselesaikannya skripsi ini.

BAB III PERENCANAAN DAN PEMBUATAN

Dalam bab ini akan dijelaskan mengenai tahap-tahap perencanaan dan pengolahan sinyal wicara serta pembuatan data base dari feature-feature aplikasi.

BAB IV PENGUJIAN DAN ANALISA

Bab ini berisi tentang analisa program yang digunakan dalam pengolahan sinyal wicara.

BAB VI PENUTUP

Bab ini berisi tentang kesimpulan yang diambil dari hasil analisa serta saran-saran yang diharapkan dapat memberikan pengembangan dan penyempurnaan skripsi ini.

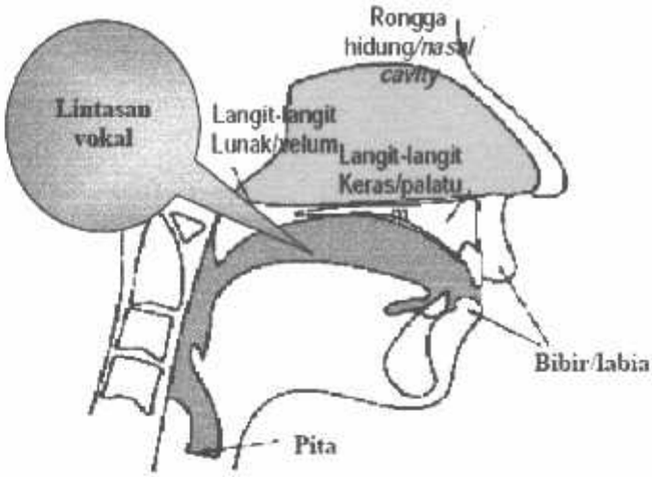
BAB II

TEORI PENUNJANG

Pada bab ini akan dijelaskan mengenai teori-teori yang berkaitan dengan bahasan skripsi ini untuk menunjang skripsi yang telah dirancang. Mulai dari sinyal data suara yang masuk, pemrosesan sinyal dan pengolahan data suara untuk menjalankan suatu intruksi.

2.1 Sistem Pengolahan Suara

Manusia menggunakan suara sebagai sumber informasi untuk mengkomunikasikan ide, keinginan, dan perasaannya kepada orang lain. Organ tubuh yang berpengaruh dalam proses produksi wicara adalah paru-paru, tenggorokan (*trachea*), larinks, farinks, rongga hidung (*nasal cavity*), dan rongga mulut (*oral cavity*). Pembangkitan sinyal suara terletak pada bentuk lintasan vokalnya (*vocal tract*). Lintasan vocal tersebut terdiri atas: dibawah katub tenggorokan (*laryngeal pharynx*), antara langit-langit lunak katub tenggorokan (*oral pharynk*), di atas velum dan diujung depan rongga hidung (*nasal pharynx*), dan rongga hidung (*nasal cavity*), seperti ditunjukkan pada gambar di bawah ini:

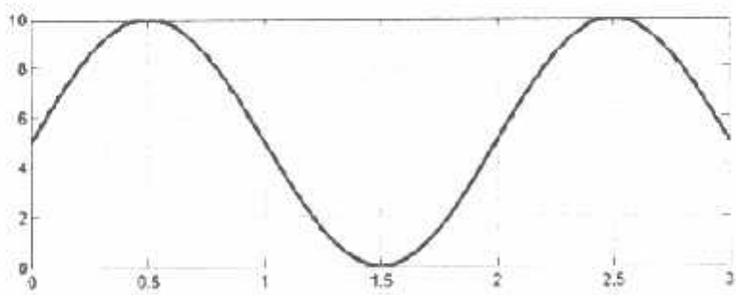


Gambar 2.1 Skema diagram organ penghasil sinyal suara

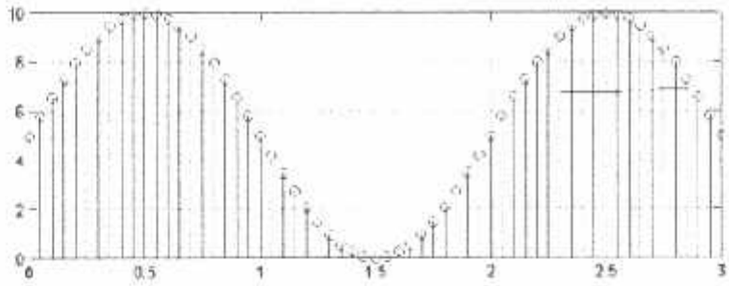
2.1.1 Proses Sampling

Sinyal suara merupakan sinyal yang tidak terbatas dalam domain waktu (*infinite time interval*). Suara manusia akan menghasilkan sinyal analog yang terus kontinyu. Untuk keperluan pemrosesan dalam transformasi fourier maka sinyal wicara harus dibentuk dalam potonganpotongan waktu yang terbatas (*finite time interval*). Karena itu sinyal yang ada dipotong-potong dalam slot-slot *interval* waktu tertentu. Berdasarkan pada teori *sampling Nyquist*[8], maka syarat dari frekuensi sampling adalah minimal dua kali frekuensi sinyal.

Berikut adalah gambar dari sinyal sinus dengan sinyal sinus tersampling.



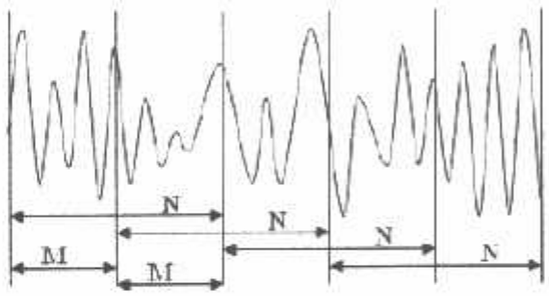
Gambar 2.2 Bentuk sinyal sinus



Gambar 2.3 Bentuk sinyal sinus yang telah disampling

2.1.2 *Frame Blocking*

Pada langkah ini, sinyal suara diblok menjadi potongan-potongan sinyal yang disebut dengan frame, didalam frame-frame dengan N sample dan digeser sebesar M sample dimana $N \approx 2 \times M$. seperti ditunjukkan pada gambar 2.4 Sehingga didapatkan nilai dari sinyal yang baru adalah sebagai berikut:

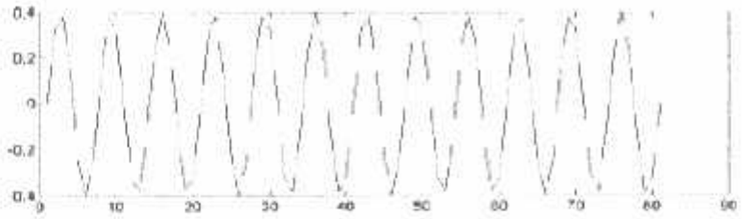


Gambar 2.4 Bentuk sinyal yang diframe blocking

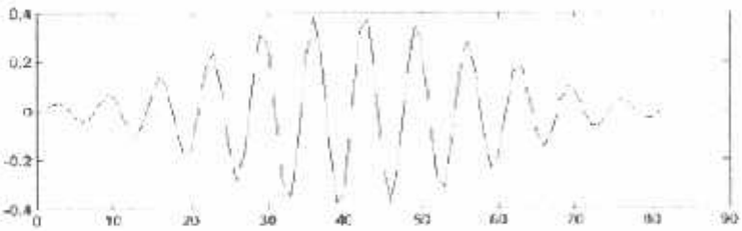
2.1.3 *Windowing*

Pada langkah ini dilakukan fungsi *weighting* pada setiap frame-frame yang telah dibentuk pada proses *frame blocking*. *Windowing* ini diperlukan untuk mengurangi efek diskontinuitas dari potongan – potongan Sinyal[3]. Dalam hal ini *windowing* yang digunakan yaitu hamming windowing.

$$Wham(n) = \{ 0.52 - 0.46 \cos[2\pi n / (N-1)] \quad 0 \le n \le N-1$$



Gambar 2.5 Bentuk sinyal murni



Gambar 2.6 Bentuk sinyal yang telah diwindow

2.1.4 FFT (*Fast Fourier Transform*)

Transformasi fourier adalah suatu metode yang sangat efisien untuk menyelesaikan transformasi fourier diskrit yang banyak dipakai untuk keperluan analisa sinyal seperti pemfilteran, analisa korelasi, dan analisa spektrum. Fast fourier Transformation atau transformasi Fourier cepat, merupakan proses lanjutan dari DFT (*Diskrit Fourier Transformation*). Transformasi Fourier ini dilakukan untuk mentransformaikan sinyal dari domain waktu ke domain frekuensi[2].

FFT adalah bentuk khusus dari persamaan integral fourier :

$$H = \int h(t)\, e^{-jwt}\, dt$$

Dengan mengubah variable-variabel, waktu (t), frekuensi (w) kedalam bentuk diskrit diperoleh transformasi fourier diskrit (DFT) yang persamaanya adalah:

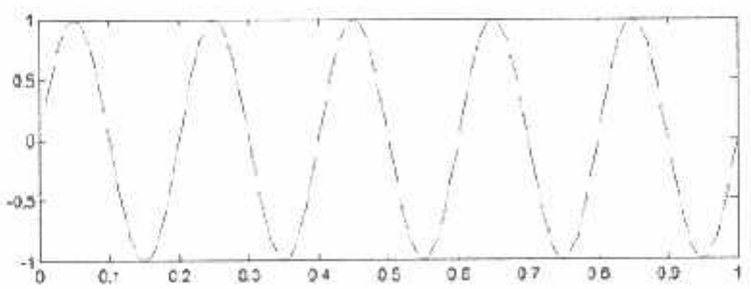
$$H(kwo) = \sum_{n=0}^{N-1} h(nT)\, e^{-jkwonT}$$

Disederhanakan dengan T=1 sampel waktu N=sample frekuensi k sehingga menjadi :

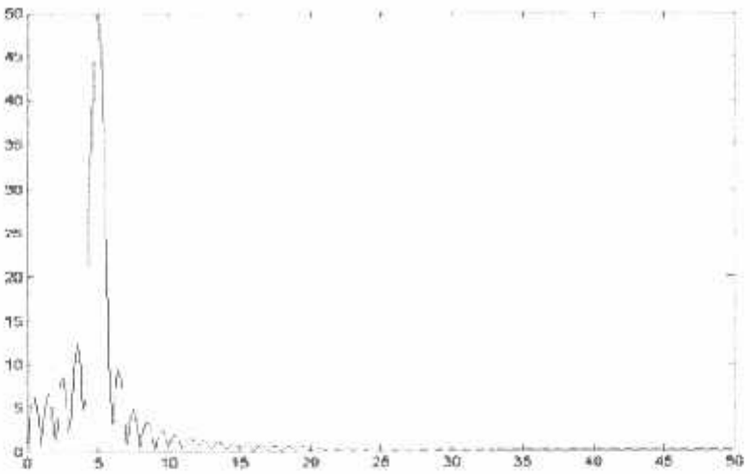
$$H(k) = \sum h(n)e$$

Dengan k : 0,1,2,…,N-1

FFT dilakukan dengan membagi N buah titik pada transformasi fourier diskrit menjadi 2, masing-masing (N/2) titik transformasi. Proses memecah menjadi 2 bagian ini diteruskan dengan membagi (N/2) titik menjadi (N/4) dan seterusnya hingga diperoleh titik minimum[5]. Pemakaian *FFT* karena untuk penghitungan komputasi yang lebih cepat dan mampu mereduksi jumlah perkalian dari N2 menjadi NlogN perkalian[3]. FFT yang digunakan memakai 256 point dan arena hasil FFT simetris, maka keluaran FFT tersebut hanya diambil sebanyak 128 data. Hasil dari proses FFT akan diperolch titik-titik sinyal yang simetris sehingga data yang dimbil hanya setengah dari data keseluruhan yang selanjutnya akan diolah di *LPC*.



Gambar 2.7 Bentuk sinyal dalam domain waktu



Gambar 2.8 Bentuk sinyal dalam domain frekuensi

2.1.5 Kepstrum

Cepstrum adalah fourier transform dari logaritma autospektrum. Berguna untuk membatasi periodiksitas pada autospektrum. *Cepstrum* juga berfungsi untuk menggandakan domain frekuensi dan konvolusi pada domain waktu.

Bentuk matematis dari proses cepstrum bisa dipisahkan menjadi dua variabel. Jika kita mengambil bentuk log dari magnitude kedua variabel tersebut, maka diperoleh rumus:

$$\text{Log } |X(\omega)| = \log |G(\omega)| + \log |H(\omega)|$$

Menghitung invers dari Fourier Transform dengan rumus dia atas, menghasilkan *quefrency*, yang mempunyai bentuk matematis sebagai berikut:

$$F^{-1} \log|X(\omega)| = F^{-1} \log|G(\omega)| + F^{-1} \log|H(\omega)|$$

Quefrency adalah x-axis dari cepstrum yang merupakan suatu unit dalam domain waktu.

2.1.6 DTW (*Dynamic Time Warping*)

Dynamic Time Warping (DTW) merupakan cara untuk membandingkan pola wicara dalam menentukan kesamaan jarak antara pola-pola yang berbeda. Dimana pengenalan wicara merupakan proses secara otomatis dalam mengambil dan menentukan informasi linguistik yang disampaikan dengan gelombang wicara menggunakan komputer atau sirkuit elektronik. Informasi linguistik tersebut pada akhirnya akan direpresentasikan dengan deretan waktu vektor-vektor spektral.

Metode normalisasi waktu menggunakan dua buah fungsi warping.

$$\begin{aligned} ix &= \phi x(k) \qquad k = 1.2.3.....T \\ iy &= \phi y(k) \qquad k = 1.2.3.....T \end{aligned} \tag{2.0}$$

2.1.6.1 Dinamik Programming

Dynamic Programming (DP) digunakan untuk memecahkan masalah deteksi pengurutan, kemampuan penggunaan pola pengenalan wicara dan masalah waktu penjajaran dan normalisasi Ada dua tipe masalah penggunaan Dynamic Programming, yaitu :

1. Masalah jalan optimal, dengan menganggap sekumpulan titik mulai dari 1 sampai dengan N, yang dihubungkan dengan pasangan setiap dari titik (i,j). Representasi perpindahan secara langsung dari titik ke 1 ke titik lainnya. Rentetan perpindahan tersebut tidak mempunyai bilangan yang ditetapkan dari transisi satu titik ke titik lainnya, hal ini disebut rentetan keputusan asinkron[7].
2. Masalah keputusan rentetan sinkronisasi, terdapat perbedaan dari satu sinkronisasi yang beraturan dari proses keputusan[7].

Penggunaan teknik *Dynamic Programming* pada *Dynamic Time Warping* untuk pertama kali dilakukan oleh Slutsker (1968), Vintsyuk (1968), dan Volichko (1971) dari Jepang untuk pengenalan wicara. Pada proses ini pembicara mengucapkan kata yang sama dengan durasi berubah setiap waktu dengan ekspansi yang non linier. Oleh karena itu, Dynamic Time Warping adalah tahap yang penting pada proses pengenalan kata. Proses DTW melakukan pengembangan pada sumbu waktu untuk mencocokkan posisi phonem/kata yang sama antara input wicara dan referensi template.

Dynamic Programming secara umum digunakan untuk memecahkan masalah rentetan keputusan, yang dilakukan pada bidang kisi sebagaimana ditampilkan pada gambar 2.12. Sinyal yang dicocokkan adalah A dan B. Di dalam bidang kisi, sinyal A disesuaikan sepanjang sumbu –x dan sinyal B disesuaikan sepanjang sumbu –y [8]. Setiap antar bagian pada bidang ini ditetapkan sebagai sebuah simpul atau titik, dimana titik (i,j) adalah pencocokan frame i untuk sinyal A dengan frame j untuk sinyal B. Misalnya diasumsikan dua rentetan waktu untuk vektor utama yang dibandingkan sebagai berikut:

$$A=a_1,a_2,\ldots,a_I \text{ dan } B=b_1,b_2,\ldots,b_J$$

Bilamana jarak spektral antara dua vektor utama a1 dan b1 direpresentasikan oleh d(c)=d(i,j), penjumlahan jarak dari awal Bilamana jarak spektral antara dua vektor utama a1 dan b1 direpresentasikan oleh d(c)=d(i,j), penjumlahan jarak dari awal sampai akhir dari rentetan sepanjang F dapat direpresentasikan dengan :

$$D(F) = \frac{\sum_{k=1}^{K} d(ck)wk}{\sum_{k=1}^{K} wk} \qquad (2.1)$$

Semakin kecil nilai ini, semakin baik kecocokan antara A dan B. Dimana wk adalah fungsi pembobotan positif terhadap F.

Dengan memperkecil persamaan diatas dengan kondisi berikut :

1. Kondisi monotonisasi dan kontinuitas

   $$0 \le ik - ik-1 \le 1, 0 \le jk - jk-1 \le 1 \qquad (2.2)$$

2. Kondisi perbatasan

   $$i1 = j1 = I, ik = I, jk = I \qquad (2.3)$$

3. Kondisi adjustment window

   $$ik - jk \le r, r = konstan \qquad (2.4)$$

Pendefinisian wk sedemikian rupa sehingga menyebut pada persamaan 2.4 menjadi konstan yang tidak tergantung pada F. Untuk contoh jika:

$wk = (ik - ik-1) + (jk - jk-1)(i0 = j0 = 0)$, wk akan menjadi :

$$\sum_{k=1}^{K} wk = I + J \qquad (2.5)$$

dan persamaan diatas,menjadi :

$$D(F) = \frac{1}{I+J} \sum_{k=1}^{K} d(ck)wk \qquad (2.6)$$

Penurunan pemrograman dinamik pada persamaan-persamaan diatas, penggunaan kondisi untuk F dan formulasi perumusan wk, persamaan 2.0 dapat dituliskan kembali sebagai berikut:

$$g(i,j) = \min \begin{Bmatrix} g(i,j-1)+d(i,j) \\ g(i-1,j-1)+2d(i,j) \\ g(i-1,j)+d(i,j) \end{Bmatrix} \quad (2.7)$$

Oleh karena itu jarak antara dua rentetan waktu A dan B setelah DTW dapat diperoleh sebagai berikut :

Kita set dari kondisi awal untuk **g(1,1)=2d(1,1)** dan **j=1**, dan hitung persamaan 2.7 dengan variasi **I** dalam sepanjang adjusment window. Perhitungan ini diulangi dengan menaikkan **j** sampai **j=1**. Jarak keseluruhan antara dua rentetan waktu A dan B dapat diperoleh dari **g(I,J)/(I+J)**

- **Batasan Normalisasi Waktu**

Supaya proses normalisasi waktu mempunyai arti pada perbedaan batasan dengan berbagai variasi interval waktu, dibutuhkan beberapa batasan terhadap fungsi warping.

- **Batasan Titik Awal dan Akhir**

Bilamana pola wicara yang dibandingkan merupakan tuturan perkata maka harus memenuhi syarat pola mulai dari titik awal sampai titik akhir dari tanda permulaan dan pengakhiran pola wicara. Titik awal dan akhir mempunyai batasan yang membatasi pola wicara harus dipenuhi.

Titik awal $\varphi x(1) = 1$, $\varphi y(1) = 1$

Titik akhir $\varphi x(T) = Tx$, $\varphi y(T) = Ty$

BAB III

PERANCANGAN DAN ANALISA SISTEM

Dalam bab ini akan dibahas tentang perencanaan sistem yang nantinya terwujudnya aplikasi yang diinginkan, mulai dari desain, pengolahan sinyal suara dan pemrograman. Selain itu juga akan membahas tentang pengolahan hasil pembandingan keluaran agar bisa dikenali sebagai perintah untuk membuka aplikasi komputer dan instalasi software yang ada pada skripsi ini menggunakan snack dan tcl/tk serta cara membuat file ekstensi **.dll**

3.1 Spesifikasi Sistem

Sistem yang akan direncanakan menggunakan software dan hardware untuk mendukung proses pembuatan sistem ini. Dengan itu maka suara di inputkan melalui microphone kemudian di proses sehingga dapat menghasilkan suatu intruksi yang dapat digunakan untuk menjalakan suatu perintah. Desain yang dirancang yaitu perekaman suara yang digunakan sebagai data standart sebagai data yang tersimpan didatabase. Dengan data ini maka suara inputan yang digunakan sebagai data baru, data ini akan dibandingkan dengan data standart untuk mendapatkan persamaan. Data yang memiliki tingkat error yang sedikit maka diasumsikan sebagai data yang tepat, kemudian jalankan program aplikasi user.

- **Spesifikasi Software**

Dalam merancang sistem adapun software yang mendukung mulai dari penginputan sinyal suara dan pemrosesan sinyal untuk menjadi suatu intruksi, diantaranya :

1. **Snack**

Snack merupakan software yang digunakan untuk pemrosesan suara yang dipakai sebagai *extensi* dalam suatu bahasa pemrograman. Snack biasanya digunakan bersama-sama dengan bahasa pemrograman script seperti Tcl/Tk, Python, Ruby. Dalam skripsi ini akan menggunakan software snack versi 2.2.2.n.

Snack memiliki banyak perintah dalam hubungannya pada pemrosesan dan memvisualisasikan suara seperi memainkan, merekam, menampilkan obyek suara[1]. Selain itu *Snack* juga menyediakan obyek-obyek suara dengan level yang lebih tinggi dan manajemen penyimpanan yang fleksibel. Selain itu *Snack* dapat menangani sebagian besar format file suara yang ada. dan juga dapat menampilkan visualisasi dari suatu sinyal suara secara real-time.

Snack dapat diperluas dengan perintah-perintah baru yang beroperasi dalam suatu obyek suara. Selain itu juga memungkinkan untuk menambah suatu format file suara dan tipe *filter* yang baru. *Snack* dapat digunakan untuk *array* yang lebar untuk sesuatu yang berhubungan dengan file-file suara dan data audio. Dari satu baris perintah yang mudah, seperti memainkan suatu file suara atau menemukan nilai ekstrim sinyalnya sampai aplikasi-aplikasi analisis suara yang kompleks dengan suatu tampilan grafis (*graphical user interfaces*).

Gambar 3.1 Contoh aplikasi Snack

2. Tcl

Tcl merupakan singkatan dari *Tool Command Language*. Sedangkan *Tk* adalah *Graphical Toolkit* extension dari *Tcl*[4]. Tcl/tk menyediakan bermacam-macam item standar antarmuka GUI untuk memfasilitasi user untuk mmbuat sebuah tampilan atau desain secara cepat dan juga bisa untuk pengembangan aplikasi tingkat tinggi lainnya.

Untuk bahasa pemrograman di Tcl/tk bentuknya sama seperti bahasa pemrograman di C/C++ terutama pada Loop structures,definisi fungsi dan logika penghitungannya. ***Catatan,*** didalam *Tcl* semua data direpresentasikan sebagai string.

Setiap masing – masing perintah *Tcl* mempunyai format perintah seperti ini ***command arg1 arg2 arg3 .....*** *Tcl* akan mengambil setiap kata pada format perintah dan mengevaluasinya. Setelah mengevaluasi setiap kata, kata pertama (*command*) dianggap sebagai nama fungsi dan fungsi dieksekusi berdasarkan argument yang mengikutinya.

3. Ms Visual C++ 6.0

Untuk proses pengelolaan suara maka *snack* dapat ditambahkan dengan perintah baru yang beroperasi pada obyek *sound*. *Snack* juga mungkin untuk menambah format file sound baru dan tipe filternya. Kita juga bisa mendefinisikan beberapa obyek perintah baru yang berhubungan dengan

Snack sound dan mengubah tracknya. Hal ini bisa diwujudkan dengan menggunakan *Snack C-library*. Sebuah contoh ekstensi dapat dicari dalam direktori ext dari distribusi sumber *Snack* bersama dengan informasi bagaimana cara membangun dan menggunakannya baik untuk Unix dan Windows. Kode sumber (*source code*) yang menunjukkan bagaimana *speech recognizer* dapat men-track obyek *Snack sound* dapat ditemukan disini. Dibawah ini adalah sebuah contoh kecil sebuah perintah dengan menambah *Snack*. Kode dan file yang berhubungan dapat dicari dalam direktori ext dari distribusi Snack.

- **Spesifikasi Hardware**

Dalam perancangan serta untuk melakukan suatu implementasi maka dibutuhkan beberapa hardware pendukung agar sistem ini dapat berjalan sesuai dengan hasil yang diinginkan, diantaranya:

1. **Komputer atau Notebook (Laptop)**

   Komputer digunakan sebagai pemrosesan data suara, pada saat perekaman sebagai data suara standart yang disimpan di database. Data suara inputan sebagai data yang dibandingkan dengan data suara standart pada database. Mulai dari pemrosesan data sinyal suara sampai dapat diasumsikan sebagai perintah untuk menjalankan program aplikasi user.

2. **Microphone**

   Microphone merupakan hardware yang sangat penting pada sistem ini karena sebagai inputan data sinyal suara yang akan diproses.

3.2 Desain Sistem

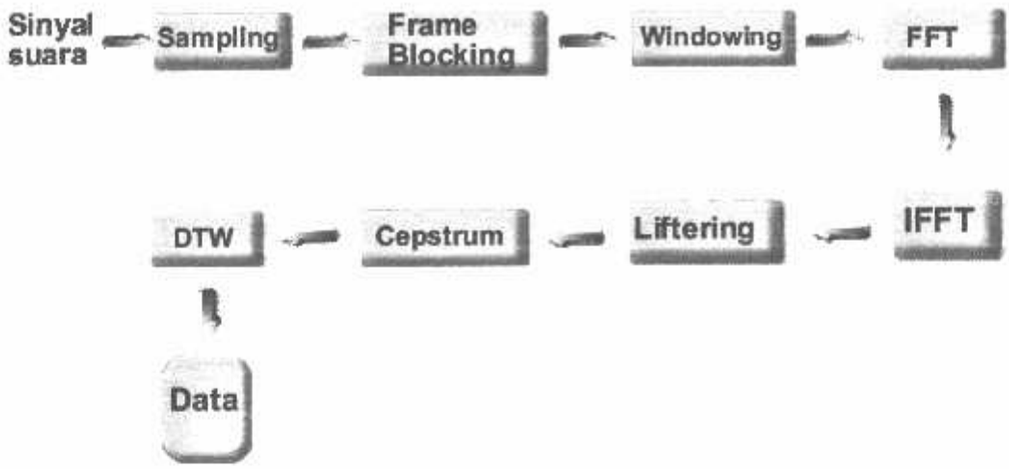
Untuk pembuatan sistem ini maka desain keseluruhannya secara garis besar adalah sebagai berikut:

Gambar 3.2 Skema Blok diagram Utama

Gambar 3.3 Skema Blok diagram Proses

Desian keseluruhan diatas menjelaskan bahwa sebagai inputan yang digunakan sinyal suara, kemudian diproses dikomputer dengan berbagai tahap mulai dari sampling, frame blocking, windowing, FFT, IFFT, liftering, cepstrum dan DTW. Dimana proses ini sama digunakan pada saat perekaman dan pengambilan data suaranya. Hanya yang membedakannya adalah pada saat pemanggilan fungsi parameternya. Adapun blok diagram keseluruhan sebagai berikut:

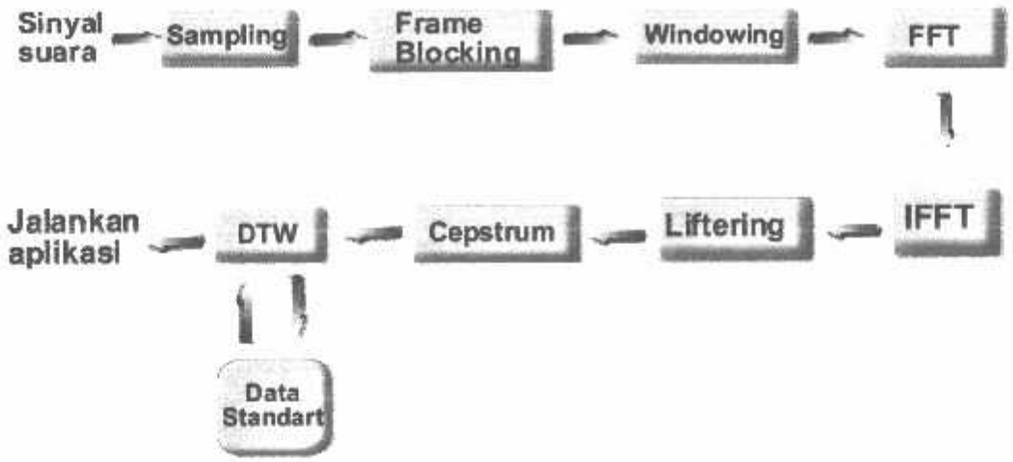
a. Blok diagram perekaman



Gambar 3.4 Skema Blok diagram perekaman suara

Dalam diagram ini akan dijelaskan bagaimana perekaman data suara yang nantinya digunakan sebagai data standart yang diletakkan didalam database. Data suara yang direkam sebanyak 5 pola akurasi suara sebagai perbandingan data suara yang baru untuk membuka aplikasi.

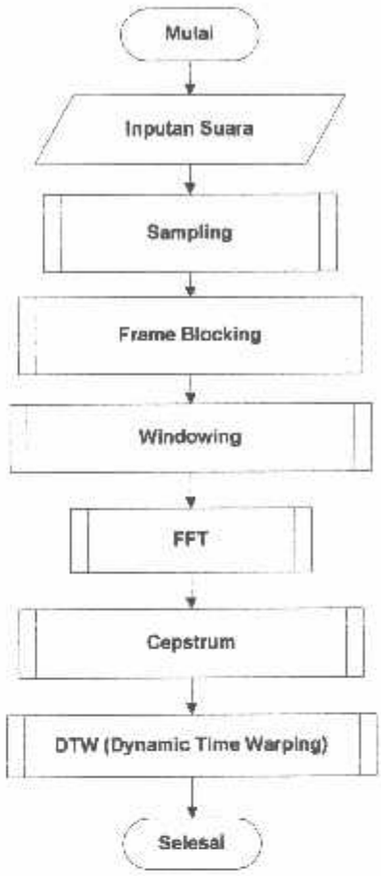
b. Blok diagram pemanggilan data



Gambar 3.5 Skema Blok diagram pengambilan data suara

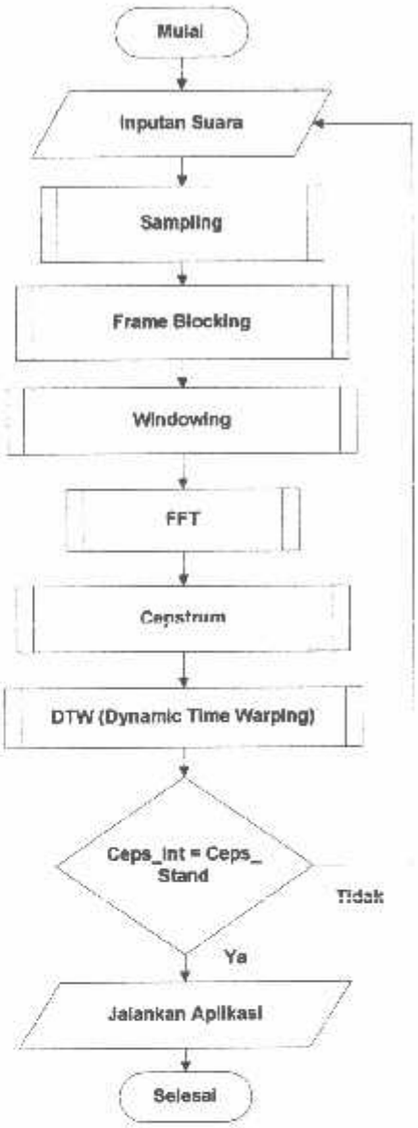
Blok diatas menjelaskan bagaimana prose pemanggilan data, dimana data suara inputan akan dibandingkan dengan data standart. Data input yang memiliki error sedikit maka diasumsikan data yang tepat untuk menjalankan aplikasi.

c. Flowchart Perekaman data



Gambar 3.6 Flowchart Perekaman data

d. Flowchart Pembandingan data



Gambar 3.7 Flowchart Pengambilan data

3.3 Desain Data Input

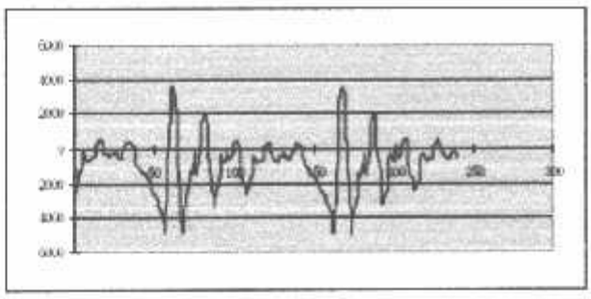
Proses perekaman suara dilakukan dengan menggunakan software buatan sendiri yang berbasis pada Tcl/tk dan Snack. Dengan menggunakan Tcl/tk dapat dibuat tampilan software dimana lingkupnya seperti membuat button, canvas,entry,radio button,check button dan banyak lagi, sedangkan Snack digunakan untuk memproses sinyal suara. Snack memiliki perintah-perintah untuk play, record, process dan memvisualisasi suara.

Pada perekaman suara ini, frekuensi sampling diset pada 12000 Hz. Selain itu juga diset pada MONO channel dan resolusi 16 bit PCM. Hasil perekaman disimpan dalam file .wav. Perekaman pertama digunakan sebagai data standart.

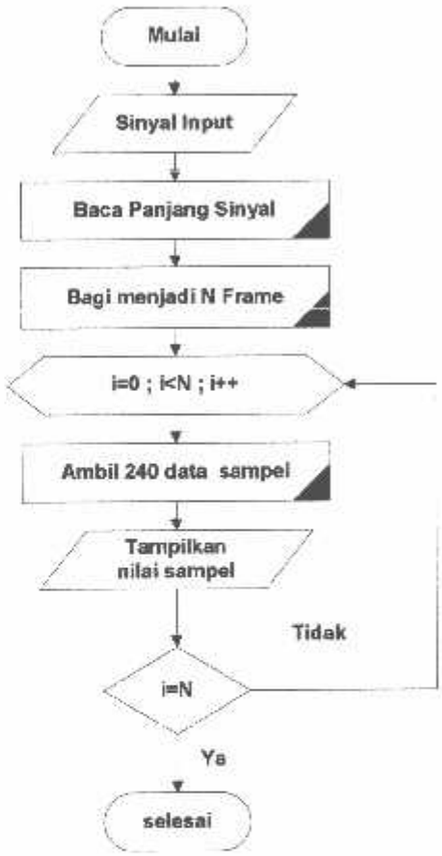
Didalam skripsi ini akan dibuat aplikasi seperti pada gambar dibawah ini. Dimana aplikasi ini merupakan gabungan atau integrasi dari Tcl/tk dan snack.

- **Sampling Sinyal Suara**

Seperti yang telah dijelaskan pada sub bab diatas, disebutkan bahwa frekuensi sampling yang digunakan adalah sebesar 12000 Hz, dimana dalam 1 detik sinyal tersebut terdapat 12000 titik sampling. Sebagai contoh, pada gambar dibawah, mempunyai sinyal sepanjang 20 ms disampling dengan frekuensi sampling sebesar 12000 Hz, akan menghasilkan titik sampling sebanyak 240 titik.



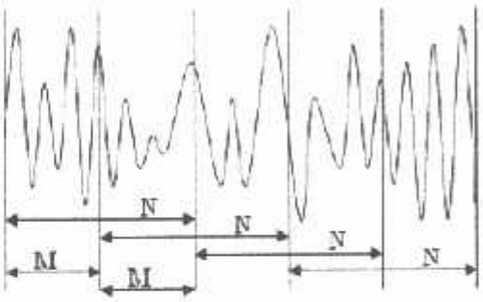
Gambar 3.8 sinyal hasil sampling dengan frekuensi 12000 Hz (I frame)



Gambar 3.9 Flowchart dari proses sampling

- ***Frame Blocking***

Hasil perekaman data suara merupakan sinyal analog yang berada dalam domain waktu yang bersifat variant time, yaitu suatu fungsi yang bergantung waktu. Oleh karena itu sinyal tersebut harus dipotongpotong dalam slot-slot waktu tertentu agar dapat dianggap invariant. Sinyal suara dipotong sepanjang 20 milidetik disetiap pergeseran 10 milidetik. Setiap potongan tersebut disebut frame. Jadi dalam satu frame terdapat 240 sampel dari 12000 sampel yang ada. Potongan frame digambarkan seperti gambar di bawah ini:



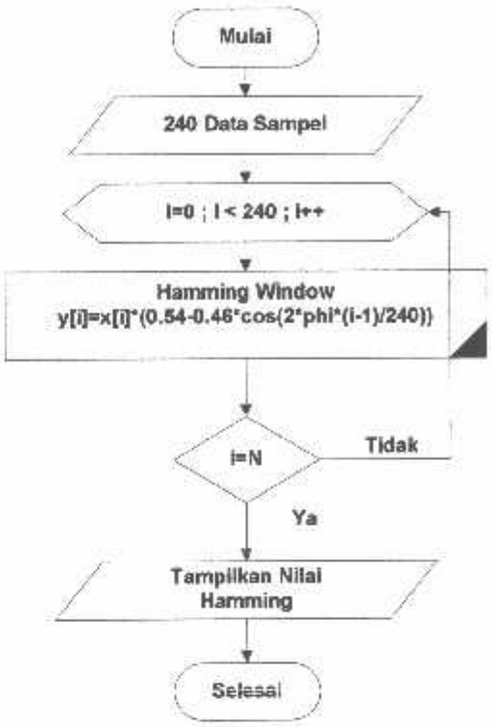
Gambar 3.10 frame blocking sinyal

- ***Windowing***

Windowing yang digunakan pada skripsi ini adalah window hamming. Digunakan windowing ini karena mempunyai hasil yang lebih baik dalam pembatasan sinyal yang akan dianalisa. Persamaan dari window hamming adalah sebagai berikut :

$$y[i] = x[i] * (0.54 - 0.46 * \cos(2.0 * 3.14159265 * (i - 1) / 240))$$

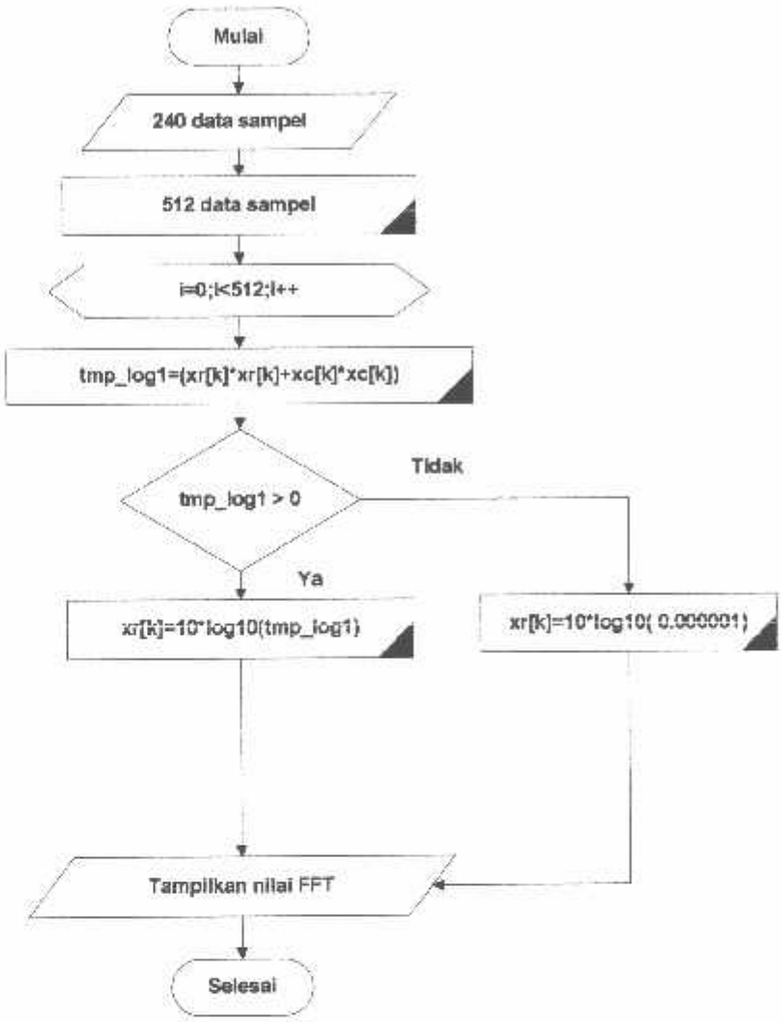
Proses windowing dapat dilihat pada flowchart berikut ini :



Gambar 3.11 Flowchart dari proses windowing

- ***Fast Fourier Transform***

Proses FFT sinyal masukan dapat dilihat dari flowchart berikut ini:

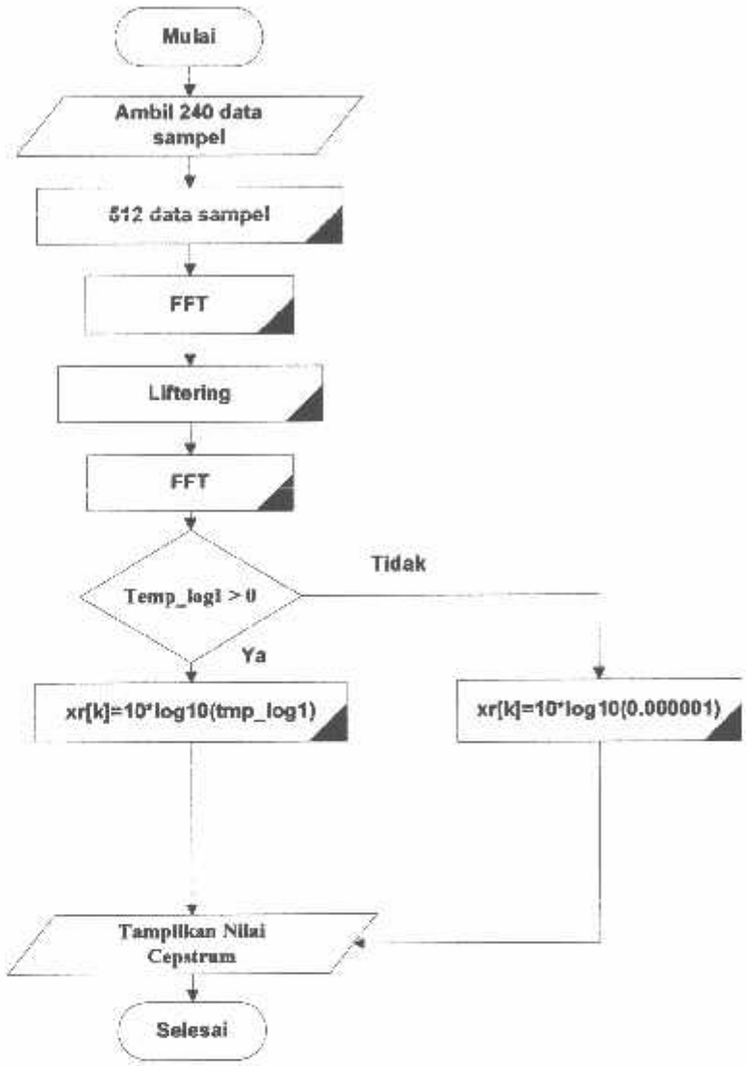


Gambar 3.12 Flowchart dari proses FFT

- ***Cepstrum***

Cepstrum adalah fourier transform dari logaritma autospektrum. Berguna untuk membatasi periodiksitas pada autospektrum. *Cepstrum* juga berfungsi

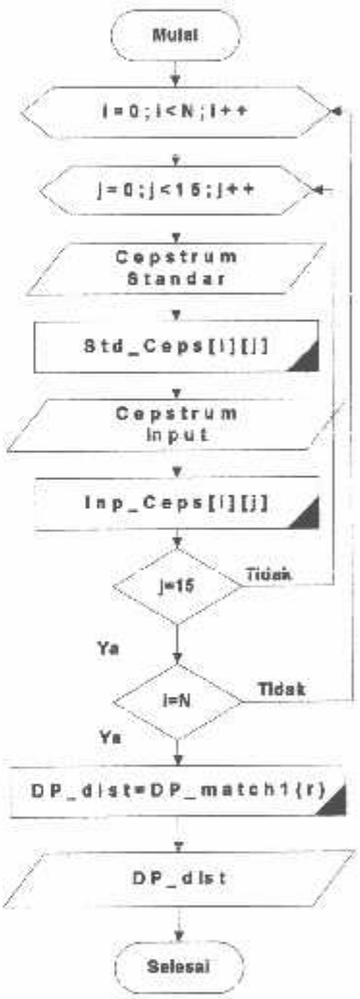
untuk menggandakan domain frekuensi. Alur dari proses *cepstrum* ditunjukkan dalam flowchart dibawah ini:



Gambar 3.13 Flowchart proses Cepstrum

- ***Dynamic Time warping***

Proses yang terjadi pada *Dynamic Time Warping* adalah pengukuran jarak suara antara sinyal standar dan sinyal input. Yang diukur adalah berupa deretan nilai hasil dari *Cepstrum* dalam bentuk kolom dan baris (i,j) yang disimpan dalam satu file. Pada kolom berisi nilai tiap cepstrum, sedangkan baris berisi banyaknya frame. Teknik yang digunakan dinamakan *Dynamic Programming*. Alur kerja dari *Dynamic Programming* ditunjukkan dalam flowchart dibawah ini :

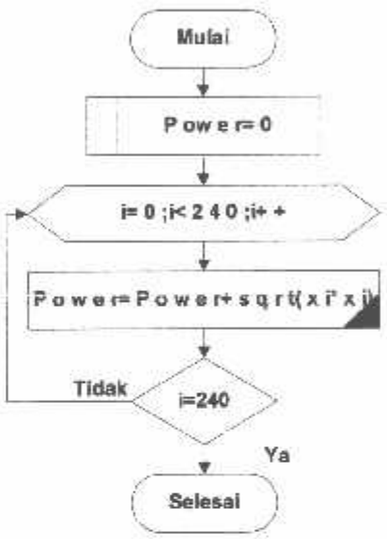


Gambar 3.14 Flowchart proses dynamic programming

- ***Seqmutasi Frame***

Untuk mendapatkan suatu sinyal suara tanpa adanya noise maka sinyal harus benar-benar dipotong tepat pada awal dan akhir suatu sinyal pada setiap frame. Oleh karena itu dipakai power sebagai salah satu cara yang efektif dalam menentukan awal dan akhir suatu sinyal suara.

Dimana tiap frame mempunyai power yang merupakan hasil kuadratisasi dari penyampelan tiap frame yang kemudian hasil keseluruhan diakar, seperti yang terlihat pada diagram alir berikut ini :



Gambar 3.15 Flowchart proses power atau segmutasi frame

Jumlah sampel per frame ditetapkan sebesar 240 sampel. Kemudian dicari power masing-masing antara sinyal utuh (sinyal suara asli) dan sinyal noise kemudian dengan $\mu \geq \mu + 3 \bullet$ s tan *dar deviasi* maka didapatkan awal dan akhir sinyal suara tersebut.

BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

Pada bab ini akan menjelaskan tentang implementasi dan pengujian terhadap kinerja software yang telah dibuat. Dengan demikian akan diketahui tingkat keberhasilan dan tingkat kekurangan dari sistem atau software yang telah dibuat dalam skripsi ini.

Implementasi dan Pengujian yang dilakukan meliputi :

1. Implementasi dan pengujian sistem pengolahan sinyal suara
2. Implementasi dan pengujian hasil DTW programming
3. Implementasi dan pengujian perintah membuka aplikasi komputer

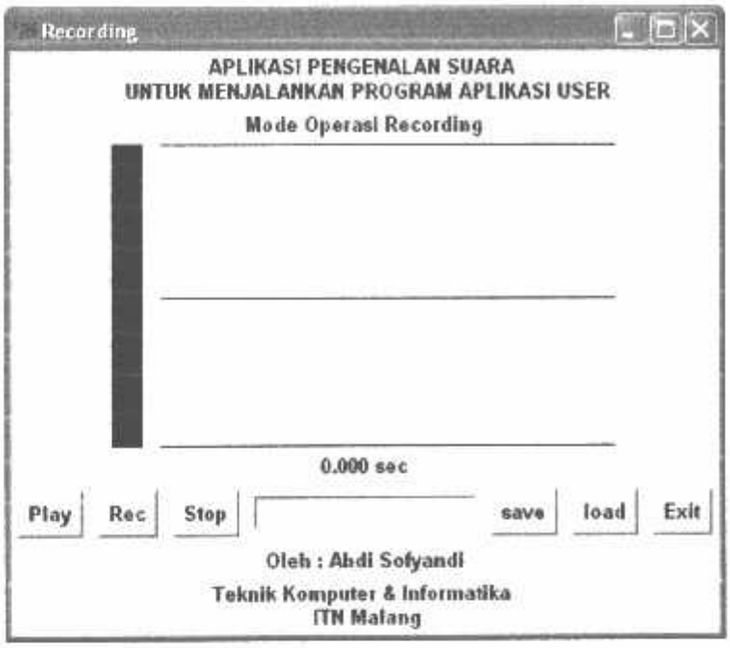
4.1 Implementasi dan Pengujian Sistem Pengolahan Sinyal Suara

Tujuan diadakan pengujian dan analisa untuk pengolahan sinyal suara adalah untuk mendapatkan ciri atau parameter dari sinyal suara tersebut.

Dimana tahap pengolahan sinyal suara sebagai berikut : perekaman suara, sampling, frame blocking ,windowing ,FFT (*Fast Fourier Transform)*, Cepstrum dan terakhir proses DTW (*Dynamic Time Warping*).

4.1.1 Perekaman Suara

Pada proses perekaman ini digunakan software perekaman suara buatan sendiri yang berbasis pada Snack dan Tcl/tk. Dimana tampilan software buatan seperti pada gambar dibawah ini .



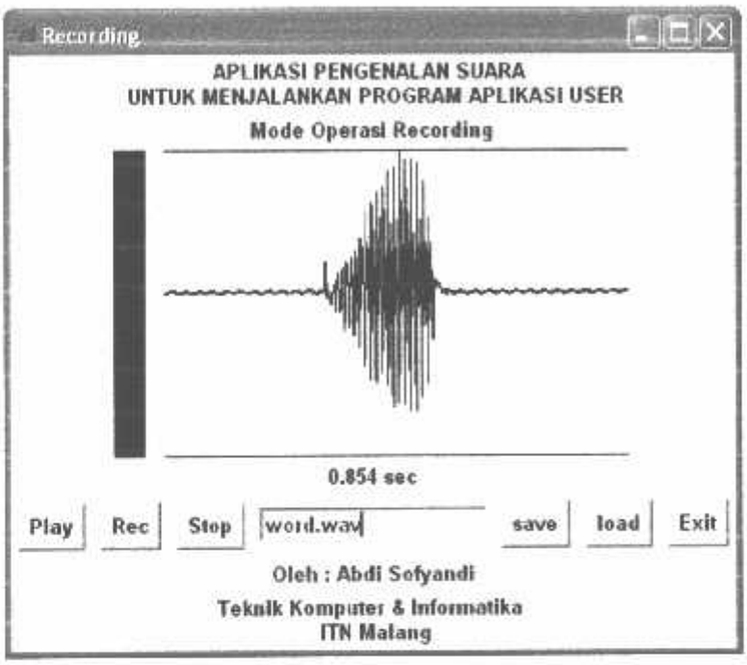
Gambar 4.1 Tampilan Software untuk Mode Perekaman

Pada gambar diatas ada empat fungsi tombol, dimana fungsi tombol tersebut memiliki fungsi yang berbeda yaitu :

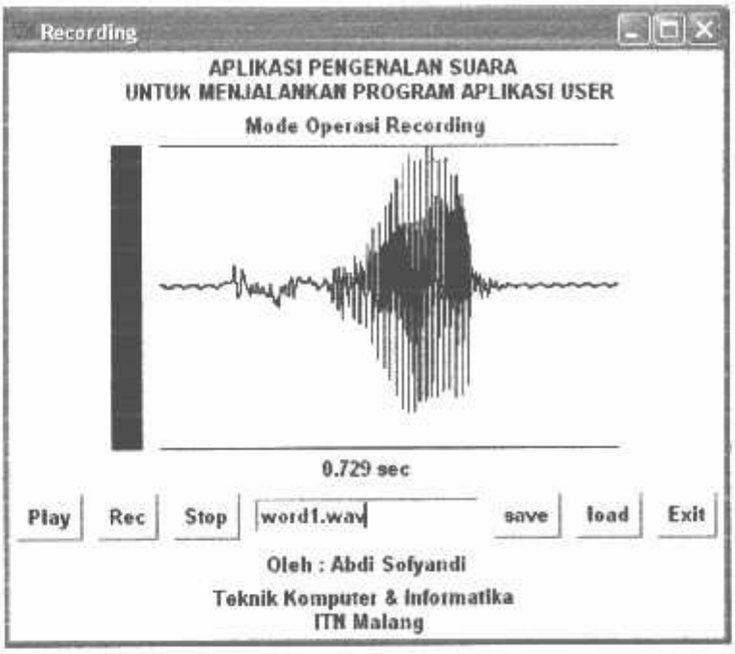
1. Tombol **Play** berfungsi untuk menyuarakan hasil suara yang telah terekam oleh software
2. Tombol **Rec** berfungsi untuk merekam suara yang diinputkan oleh penyuara melalui *Microphone.*
3. Tombol **Stop** berfungsi untuk menghentikan proses perekaman
4. Tombol **Save** berfungsi untuk menyimpan hasil perekaman suara.

   Catatan** : pada saat sebelum menekan tombol save terlebih dahulu harus mengisi nama file suara di kotak enrty. Dalam penulisan nama file harus berakhiran* ***.wav
5. Tombol **load** berfungsi untuk membaca file .wav yang sudah ada.

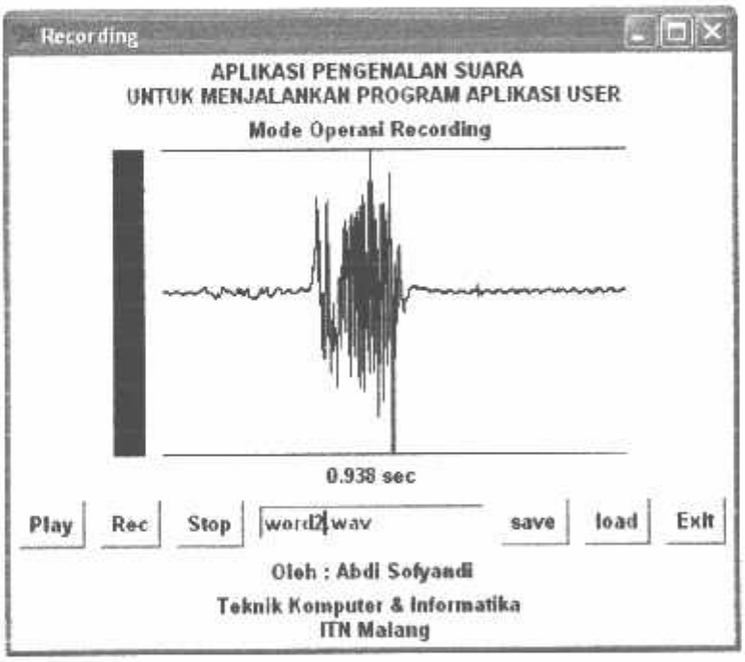
Sebelum perekaman dalam program tcl tersebut diinisialisasikan frekuensi sampling yang digunakan pada 12000 Hz, pada MONO channel dengan resolusi 16 bit PCM.



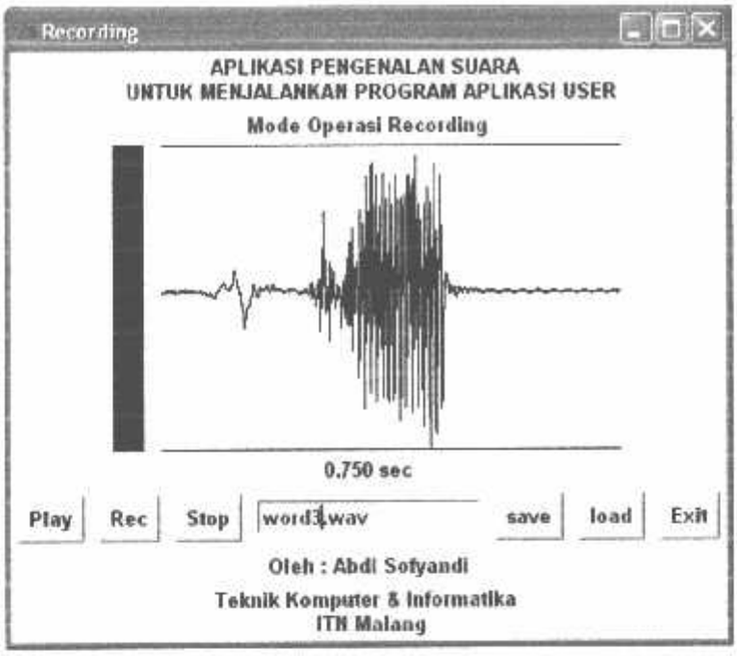
Gambar 4.2 Perekaman kata" word" pada Mode Perekaman



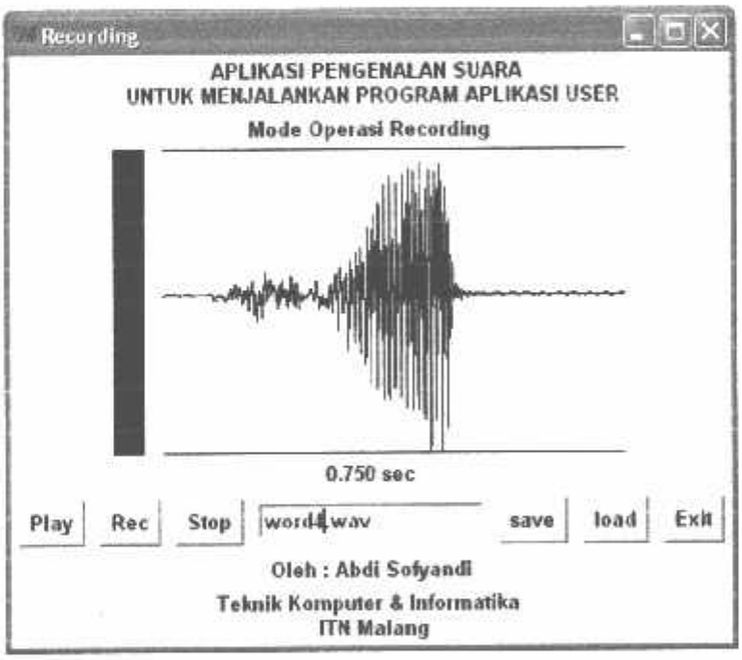
Gambar 4.3 Perekaman kata "word1" pada Mode Perekaman



Gambar 4.4 Perekaman kata" word2" pada Mode Perekaman



Gambar 4.5 Perekaman kata "word3" pada Mode Perekaman

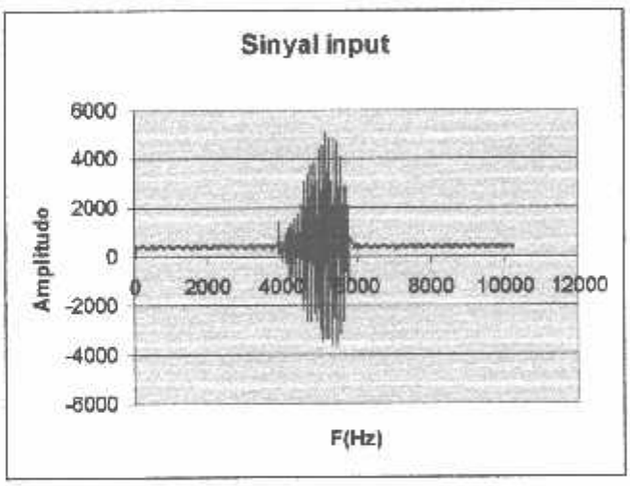


Gambar 4.6 Perekaman kata "word4" pada Mode Perekaman

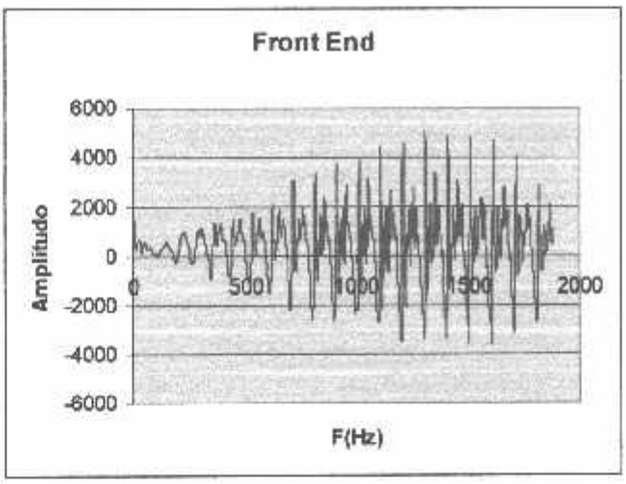
Jika tombol Record ditekan maka program akan merekam kata "word " (contoh). Pada tampilan canvas dari Mode Perekaman akan secara otomatis akan menggambarkan bentuk dari sinyal suara yang telah direkam. Hasil dari proses perekaman kata "word" dapat dilihat pada gambar 4.2 diatas.

4.1.2 Proses *Front End*

Sinyal yang masuk dari hasil perekaman merupakan sinyal yang masih terhubung dengan noise dan masih memiliki *tail* baik di awal sinyal maupun akhir sinyal dan merupakan sinyal yang bersifat variant time. Pada proses front end ini, sinyal tail-tail dan sinyal-sinyal noise dipotong dan diambil sinyal muminya saja sahingga di dapatkan hasil sebagai berikut:



Gambar4.7 Sinyal Input kata word

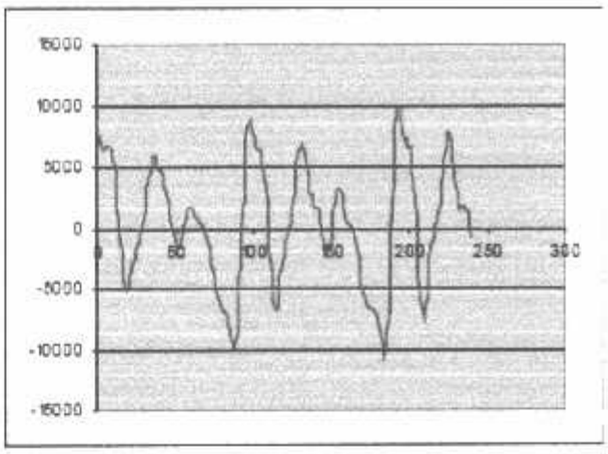


Gambar4.8 Hasil front-end kata word

Dengan frekuensi sampling sebesar 12000 Hz, maka didapatkan bentuk sinyal yang hampir mirip dengan sinyal analognya, karena sinyal suara manusia memiliki jarak frekuensi antara 300 sampai 6000 Hz, dimana syarat Nyquist minimal ***fsampling*** $\geq$ ***2 x fsinyal*** telah terpenuhi.

4.1.3 Proses *Frame Blocking*

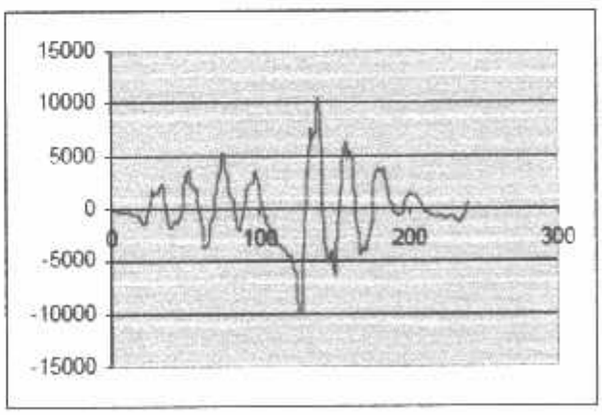
Pada proses ini dilakukan pemotongan sinyal dalam slot-slot tertentu agar dianggap invariant. Pada proyek akhir ini sinyal suara dipotong sepanjang 20 milidetik disetiap pergeseran 10 milidetik. Setiap potongan tersebut disebut frame. Jadi dalam satu frame terdapat 240 sampel dari 12000 sampel yang ada. Hasil nilai dari proses ini adalah sebagai berikut:



Gambar4.9 Hasil frame ke-1 kata word

4.1.4 Proses *Windowing*

Setelah proses frame blocking, sinyal diproses windowing untuk mengurangi efek diskontinuitas ketika sinyal ditransformasikan ke domain frekuensi. Proses windowing dilakukan tiap-tiap subband yang terdiri 240 data sample dan digeser setiap setengah subband yaitu 120 sample . Karena adanya pergeseran inilah kemungkinan puncak-puncak yang mestinya terambil menjadi terpotong dapat terjadi. Data hasil Windowing untuk kata "word" ditunjukkan di bawah ini .

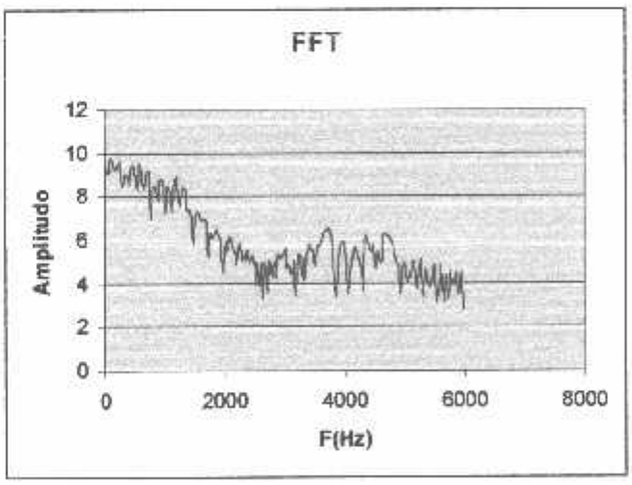


Gambar 4.10 Hasil kata word yang di windowing

Berdasarkan data di atas maka dapat dikatakan bahwa window hamming menyebabkan sinyal yang disampel lebih halus. Hal ini membuktikan bahwa fungsi dari windowing untuk mengurangi efek discontinuitas pada ujung-ujung frame adalah benar.

4.1.5 FFT (*Fast Fourier Tramsform*)

Pada proses ini sinyal yang sebelumnya berada dalam domain waktu akan dirubah dalam domain frekuensi. Setiap sinyal yang berasal dari alam merupakan sinyal analog yang bila diolah harus dirubah dalam bentuk sinyal digital. Dan pengolahan dalam digital merupakan pengolahan dalam bentuk diskrit. Pada proyek akhir ini sinyal dalam domain waktu akan dirubah dalam domain frekuensi dengan 512 titik. Karena hasil yang diperoleh berupa hasil dari fungsi konvolusi maka hanya akan diambil 256 titik saja yang akan diolah dalam proses selanjutnya. Sedangkan 256 sisanya tidak dipergunakan karena berupa pencerminan saja. Hasil dari Fast Fourier Transform (FFT) dari kata "word" dalam bentuk teks adalah :

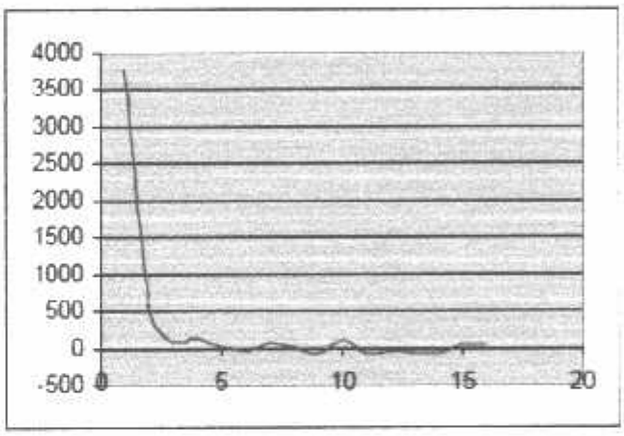


Gambar 4.11 Hasil FFT kata word

4.1.6 *Liftering*

Pengujian selanjutnya setelah proses FFT adalah liftering. Sebelum proses *liftering* dilakukan hasil dari FFT di invers terlebih dahulu. Hasil dari IFFT (Invers Fast Fourier Transform) diliftering dengan cara memproscsnya kembali dengan Fast Fourier Transform (FFT) yang bertujuan untuk mendapatkan hasil yang sebenarnya. Pada *liftering* ini data yang diambil adalah 16 data saja tiap framenya yang bisa mewakili semua data yang telah terolah dalam FFT.

Hasil liftering dari kata "word" dalam bentuk text adalah sebagai berikut:



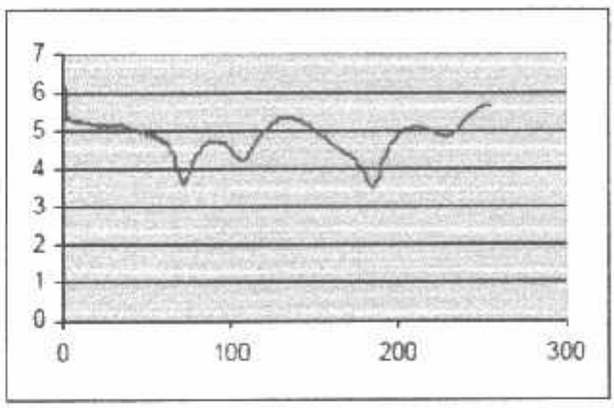
Gambar 4.12 Hasil Liftering kata word

4.1.7 *Cepstrum*

Cepstrum pada dasarnya sama dengan FFT, hanya saja hasil dari *cepstrum* harus melewati beberapa proses, seperti yang telah dijelaskan di atas yaitu dari hasil FFT harus di invers dulu untuk mendapatkan nilai lifternya dan untuk mendapatkan nilai *cepstrum* maka nilai lifter tersebut harus diproses dengan

FFT kembali, hasil dari proses FFT kedua inilah yang disebut sebagai nilai *cepstrum*.

Hasil dari nilai cepstrum untuk kata "word" adalah sebagai berikut:



Gambar 4.13 Hasil Cepstrum kata "word"

4.1.8 *Dynamic Time Warping*

Pengujian terakhir dari proses pengolahan sinyal suara adalah membandingkan sinyal hasil cepstrum antara data input dan data standarnya.

Selanjutnya menentukan data sebagai data standar dan data sebagai data masukan. Pada sistem ini digunakan kata standart sesuai dengan feature yang ada pada windows dan kata masukan untuk masing-masing pengujian dengan kata standar sebanyak 3 (tiga) pola itonasi suara. Adapun kata yang digunakan baik sebagai standar dan sebagai kata yang diuji adalah kata word.

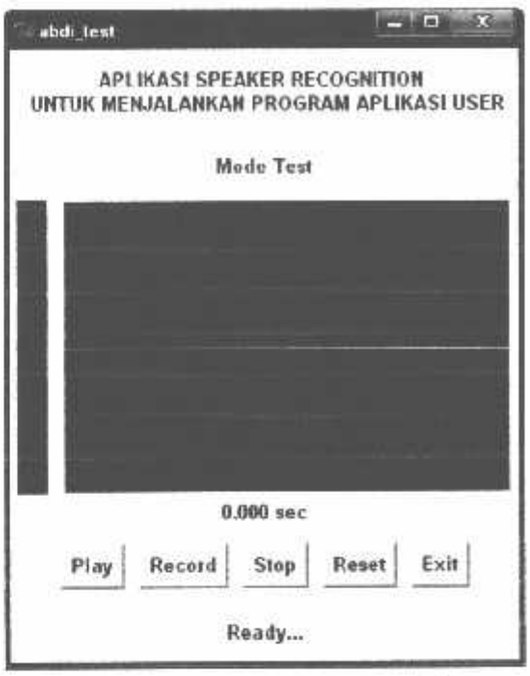
Tabel 4.1 Pengujian kata "word"

sampel	sinyal_in	Font-End	Frame	Hamm	FFT	Lift	Ceps
0	0	352	6928	0	23.4375	3778.07	6.0941
1	0	445	7587	-30.533	46.875	461.627	6.04011
2	0	247	7017	15.2247	70.3125	59.3654	5.25199
3	0	172	6483	49.3978	93.75	142.131	5.3976
4	-1	159	6677	49.4935	117.188	7.5938	5.2572
5	0	111	6626	49.408	140.625	-54.139	5.27093
6	-1	123	6603	61.5247	164.063	64.9838	5.2617
7	0	82	6587	51.8898	187.5	8.60539	5.23693
8	-1	-46	6124	48.3283	210.938	-63.724	5.26125
9	0	-138	5824	77.3181	234.375	96.7661	5.22409
10	-1	-239	5363	93.9732	257.813	-80.378	5.25262
11	0	-327	3987	90.1866	281.25	-46.215	5.2127
.....	.....	.....	.....	.....	.....	.....	.....

4.1.9 Pengujian data standar

Selanjutnya menentukan data sebagai data standar dan data sebagai data masukan. Kata yang dimasukan untuk masing-masing pengujian dengan kata standar sebanyak 1 (satu) kali untuk masing masing penguji yang terdiri dari 3 orang. Adapun kata yang digunakan baik sebagai standar dan sebagai kata yang diuji adalah word, excel, notepad, explorer dan powerpoint.

Pengujian ini akan dilakukan dengan memasukkan nilai treshold yang berbeda-beda yaitu antara 0.5-1.00. Dengan membandingkan jarak sinyal input dengan standart, berikut adalah data jarak sinyal input dengan standart suara dari hasil pengujian tersebut.



Gambar 4.14 Form pengujian data

❖ Suara standart dengan nilai **treshold 0.5**

Tabel 4.2 Jarak pada pengujian kata "word".

Suara ke:	abdi	roni	puji
1	0.402032	0.857657	0.355623
2	0.625204	0.674103	0.623154
3	0.679829	0.471361	0.562314
4	0.455639	0.811666	0.789230
5	0.502216	0.465993	0.652136
6	0.366117	0.693642	0.565465
7	0.828707	0.613618	0.653495
8	0.455639	0.811666	0.789230
9	0.502216	0.465993	0.652136
10	0.640016	0.493328	0.452362
porsentase	40%	50%	40%

Tabel 4.3 Jarak pada pengujian kata "Excel".

Suara ke:	abdi	roni	puji
1	1.084726	0.499191	0.562323
2	0.331097	0.608052	0.423601
3	0.618141	0.612070	0.502113
4	0.565658	0.705373	0.623015
5	0.408482	0.710687	0.895200
6	0.564966	0.473298	0.490265
7	0.693995	1.037979	0.362150
8	0.306762	0.568546	0.536521
9	0.454501	1.379393	0.623562
10	1.500334	0.373234	0.956321
porsentase	40%	30%	30%

Tabel 4.4 Jarak pada pengujian kata "Notepad".

Suara ke:	abdi	roni	puji
1	0.473096	0.282904	1.142524
2	0.976241	0.734190	0.445642
3	0.431315	0.292313	0.570354
4	0.382843	0.881118	0.378434
5	0.425822	1.006238	0.522486
6	1.031584	0.808692	0.721542
7	0.756306	0.374646	0.506601
8	0.834122	0.957582	0.519245
9	0.614320	0.599411	0.462775
10	0.731581	0.497464	0.504801
porsentase	40%	30%	30%

Tabel 4.5 Jarak pada pengujian kata "Explorer".

Suara ke:	abdi	roni	puji
1	0.733117	0.373895	0.359613
2	1.062475	0.719992	0.789505
3	0.447972	1.110336	0.458156
4	0.356438	1.075092	0.759761
5	0.582094	0.473389	0.903223
6	0.416891	0.620962	0.746770
7	0.920269	0.670783	0.544832
8	0.476052	0.417504	0.992121
9	1.020647	0.559310	0.659764
10	1.338251	0.677228	0.303338
porsentase	40%	30%	40%

Tabel 4.6 Jarak pada pengujian kata "Powerpoint".

Suara ke:	abdi	roni	puji
1	0.567609	0.640133	1.165613
2	0.386238	1.090954	1.024504
3	0.811958	0.567868	0.598036
4	0.653763	0.525063	0.523182
5	0.484151	0.424793	0.938832
6	0.726070	0.822214	0.402991
7	0.701970	0.292685	0.683848
8	0.495797	0.830497	0.495709
9	0.960524	1.306671	0.483144
10	0.613275	0.771774	0.774637
porsentase	50%	30%	40%

Dari tabel diatas prosentase keberhasilan user dalam mengakses system pada saat treshold bernilai 0.5 sangat rendah sekali yaitu di bawah 50%, hal ini dikarenakan nilai treshold yang digunakan bernilai kecil sehingga setiap hasil perbandingan antara sinyal input dan sinyal standart yang bernilai di atas 0.5 tidak dikenali oleh sistem dan tidak diijinkan untuk mengakses sistem.

❖ Suara standart dengan nilai **treshold 0.75**

Tabel 4.7 Jarak pada pengujian kata "word".

Suara ke:	abdi	roni	puji
1	0.321853	0.356019	0.638512
2	1.116625	0.351050	1.303766
3	1.025868	1.434222	0.457077
4	0.327594	0.891989	1.069751
5	0.719588	0.741637	0.586932
6	0.737725	0.638346	0.812839
7	0.671984	1.053837	0.729664
8	1.027802	0.824474	0.938512
9	0.855454	0.648367	1.047253
10	0.670201	1.202910	0.957070
porsentase	60%	50%	50%

Tabel 4.8 Jarak pada pengujian kata "Excel".

Suara ke:	abdi	roni	puji
1	1.050225	0.907005	0.499683
2	0.571556	0.512550	0.514072
3	0.605751	0.488500	0.759769
4	0.728106	0.973695	0.903220
5	1.257095	0.756083	0.746774
6	0.440550	0.623928	0.544831
7	0.744542	0.811146	0.802310
8	1.352514	0.928509	0.802310
9	0.432747	0.482005	0.992124
10	0.812394	0.301685	0.992123
porsentase	50%	60%	50%

Tabel 4.9 Jarak pada pengujian kata "Notepad".

Suara ke:	abdi	roni	puji
1	1.070112	0.665533	0.518623
2	0.796343	0.994377	0.773358
3	0.522910	0.770163	0.742602
4	0.816332	0.839625	1.165712
5	0.592678	0.835088	0.988789
6	0.319814	0.780470	0.445635
7	0.266231	0.416075	0.570305
8	0.589232	0.571501	1.025610
9	1.037097	0.445879	0.558613
10	0.932478	0.552566	0.899540
porsentase	50%	50%	60%

Tabel 4.10 Jarak pada pengujian kata "Explorer".

Suara ke:	abdi	roni	puji
1	0.628323	0.553302	0.825278
2	0.560893	1.043320	0.421986
3	0.530785	1.915225	1.129902
4	1.657600	0.819783	0.701200
5	0.428186	0.603269	0.340110
6	0.310154	0.396780	0.511306
7	0.609623	0.580673	0.462397
8	0.934510	0.565934	1.218053
9	1.613130	0.867827	1.253870
10	0.703120	1.313216	1.251031
porsentase	60%	60%	50%

Tabel 4.11 Jarak pada pengujian kata "Powerpoint".

Suara ke:	abdi	roni	puji
1	0.572840	0.403081	0.398036
2	0.742211	1.143555	0.523182
3	0.931708	0.412857	0.938832
4	0.765890	1.172596	0.402991
5	0.677475	0.762043	1.683848
6	0.541399	0.644781	1.495709
7	0.863835	0.763554	0.483144
8	1.004245	0.591836	0.774637
9	0.587960	0.669542	0.855071
10	0.985545	1.067886	0.802310
porsentase	60%	50%	50%

Dari tabel diatas tingkat keberhasilan pengoperasian sistem dengan nilai treshold 0.75 lebih tinggi dari treshold 0.5 yaitu antara 50% - 60%, hal ini dikarenakan pada saat nilai treshold 0.75, nilai-nilai hasil perbandingan input dan standart yang nilainya lebih kecil dan sama dengan 0.75 dikenali sebagai perintah untuk membuka aplikasi komputer.

❖ Suara standart dengan nilai **treshold 1.00**

Tabel 4.12 Jarak pada pengujian kata "word".

Suara ke:	abdi	roni	puji
1	0.632351	0.738303	0.783443
2	0.658493	0.363753	0.638515
3	0.707568	1.149365	1.212831
4	0.698505	0.295644	1.417794
5	0.803695	0.420352	0.977773
6	0.581589	0.976898	0.656552
7	0.948622	0.996951	0.632130
8	0.785540	1.434223	0.561235
9	0.993250	1.208140	1.325612
10	0.451566	0.639162	0.123410
porsentase	90%	70%	80%

Tabel 4.13 Jarak pada pengujian kata "Excel".

Suara ke:	abdi	roni	puji
1	1.154720	0.774807	1.260733
2	0.374839	0.850162	0.514072
3	0.984751	1.214072	0.467792
4	0.551194	0.744542	0.475527
5	1.171132	0.825825	1.273615
6	0.761714	1.025635	0.634481
7	0.824642	1.156894	1.563510
8	0.573924	0.965342	1.123566
9	0.710074	0.815532	0.563265
10	0.937720	0.868256	0.986952
porsentase	80%	70%	70%

Tabel 4.14 Jarak pada pengujian kata "Notepad".

Suara ke:	abdi	roni	puji
1	0.497260	1.151231	0.341841
2	0.695609	0.880629	1.078443
3	1.109406	0.933641	0.735385
4	0.282290	0.446647	1.149444
5	0.781762	1.292541	0.979266
6	0.587943	0.452909	0.323545
7	1.269976	0.636494	0.418621
8	0.864635	0.568449	0.748105
9	0.836743	1.300329	1.273987
10	0.447962	1.353104	1.015011
porsentase	80%	70%	70%

Tabel 4.15 Jarak pada pengujian kata "Explorer".

Suara ke:	abdi	roni	puji
1	0.737516	0.532001	0.520512
2	0.505684	0.623533	0.963914
3	0.813788	0.365231	1.343788
4	1.019756	1.563320	1.158152
5	0.866355	1.265132	0.575354
6	1.145940	0.563212	0.903224
7	0.885336	0.653212	0.468742
8	0.357721	0.984210	1.230508
9	0.874638	0.566154	1.259613
10	1.265350	1.344420	0.789501
porsentase	90%	70%	70%

Tabel 4.16 Jarak pada pengujian kata "Powerpoint".

Suara ke:	abdi	roni	puji
1	0.768462	0.987147	0.523182
2	1.559721	1.796772	0.938832
3	1.408969	0.611678	0.499683
4	0.730940	0.634481	1.380911
5	0.391230	1.334747	0.514072
6	0.995094	0.997589	1.378485
7	0.834841	1.015175	0.988742
8	0.806950	0.819342	1.142593
9	0.512806	0.618379	1.245604
10	1.365357	1.459107	0.570375
porsentase	80%	80%	80%

Dari tabel diatas prosentase dari nilai treshold=1.00 memiliki prosentase akurasi yang paling bagus yaitu hampir mendekati 100%. Untuk nilai treshold di atas 1.00, pada saat diinputkan kata sembarang akan tetap mengenali dan memproses inputan tersebut sebagai sebuah perintah sehingga tingkat akurasinya buruk.

Dari table 4.2 - 4.16 diatas dapat disimpulkan :

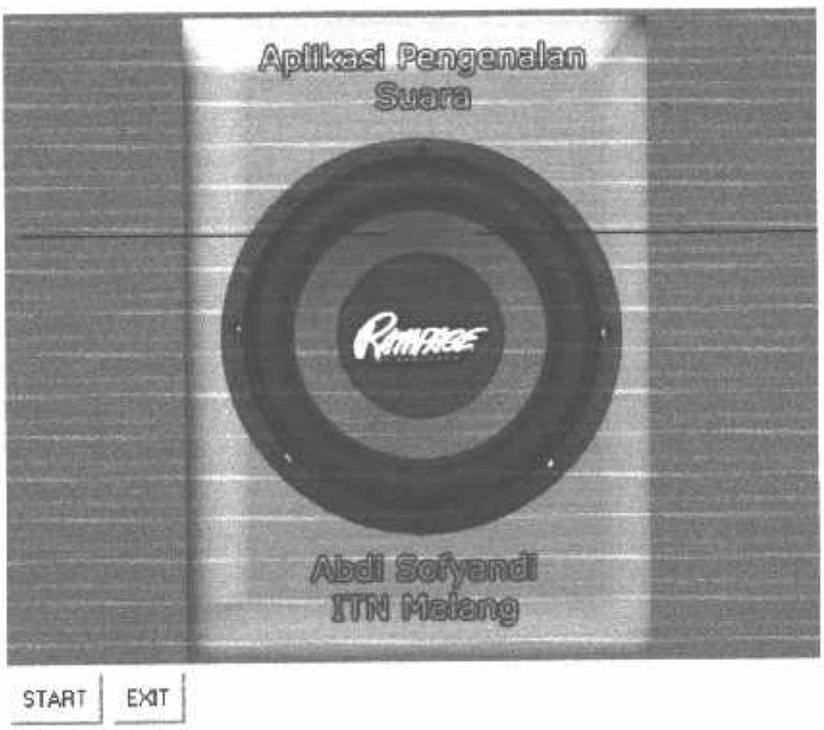
- Nilai treshold antara 0.5 – 1.00 memiliki tingkat prosentasi keakuratan yang berbeda-beda, untuk treshold=0.5, prosentase ketepatan sistem dalan mengeksekusi adalah dibawah 50%, untuk treshold=0.75, prosentase ketepatan sistem dalam mengeksekusi adalah sekitar 50% - 60% dan akurasi terbaik didapat pada nilai treshold sebesar 1.00, pada nilai prosentase keberhasilan sistem dalam mengeksekusi data hampir mendekati 100%. Sedangkan untuk nilai di atas 1.00 tingkat akurasinya buruk sebab setiap inputan (sembarang inputan) tetap diproses oleh sistem dan dikenali sebagai perintah eksekusi. Dengan kata lain semakin kecil nilai treshold maka semakin kecil nilai batas toleransi untuk nilai hasil pematchingan dan semakin besar nilai treshold maka semakin besar pula nilai toleransi untuk hasil pematchingan pada sistem.

- Dengan menggunakan Dynamic time warping dan nilai minimal serta sorting, pengukuran jarak yang terkecil digunakan sebagai penunjuk perintah yang benar.

4.2 Pengujian Sistem

Pengujian sistem dilakukan pada komputer yang sudah terinstal Tcl dan Snack untuk menerima sinyal suara dari user. Dari pengolahan sinyal suara didapatkan ciri-ciri dari user. Pengukuran jarak dan matching terjadi pada proses DTW.

Proses dari berjalannya sistem adalah sebagai berikut:



Gambar 4.14 Tampilan Software Penguji

- User menginputkan sinyal suara melalui micropnone.

  Penginputan data dilakukan dengan menggunakan microphone dan dilakukan di tempat yang tidak banyak gangguan suara lain, sehingga sinyal yang terdapat masih banyak ditumpangi oleh noise.

- Sistem akan mengolah/mengekstrak sinyal suara yang masuk sehingga didapatkan parameter-parameter yang diperlukan.

  Sinyal suara dari user akan diolah dan diproses sesuai dengan program yang telah dibuat, pertama-tama yaitu mendeteksi sinyal yang masuk dan mencari nilai dari sinyal tesebut sehingga pada saat digambarkan data yang diperoleh bisa mempunyai bentuk yang sama dengan sinyal aslinya.

  Setelah didapat data dari sinyal asli maka selanjutnya program akan mencari sinyal suara murni dan memotong tail-tail dari sinyal. Sinyal yang telah dipotong tail-tailnya itu kemudian di potong menjadi beberapa frame kemudian di windowing untuk mengurangi efek discontinuitasnya.

  Sinyal hasil windowing masih dalam domain waktu sehingga untuk mendapatkan sinyal dalam domain frekuensi harus di proses dengan FFT (Fast Fourier Transform). Hasil FFT tersebut kemudian akan di liftering untuk didapatkan suatu nilai yang bisa mewakili 256 data, dalam proyek akhir ini digunakan 16 data. Hasil dari liftering ini akan menjadi inputan pada proses cepstrum. Nilai–nilai dari cepstrum ini yang akan dimatchingkan dengan data standart pada database.

- Sinyal hasil akan disamakan dengan sinyal standart yang ada.

  Sinyal input yang telah diproses akan dimatchingkan dengan data yang telah tersimpan pada database dengan menggunakan metode *Dynamic*

Time Wrapping. Pada proses DTW ini, yang dimatchingkan adalah nilai power dari masing-masing frame sinyal standart dan sinyal input.

- Sinyal dengan nilai error terendah (hasil pematcingan) akan diterima sebagai perintah untuk menjalankan aplikasi komputer.

Sinyal dengan nilai error matching terendah diasumsikan sebagai sinyal yang sama dengan sinyal standart yang terdekat dengan nilainya.

Semua program untuk memproses sinyal mulai dari pembacaan sinyal awal sampai proses DTW dikerjakan oleh visual C yang hasilnya dieksport ke tcl/tk schingga yang berfungsi sebagai eksekutor adalah tcl. Pada tcl inilah disetting frekuensi sampling dan channel dari sinyal yang masuk.

Eksekusi dari perintah atau sinyal yang telah diperintahkan oleh user dikerjakan oleh eksekutor pada program tcl/tk. Perintah eksekusi apliksi program pada tcl menggunakan perintah " exec filename.exe".

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan pada hasil pengujian dan analisa terhadap hasil yang didapatkan, maka dapat diambil suatu kesimpulan yaitu :

1. Akurasi suara setiap manusia sangat berbeda-beda, dalam pengujian tingkat keberhasilan dalam pemanggilan data masih rendah. Data yang berhasil diasumsikan kurang-lebih 70%, apalagi data inputan sangat banyak noise.
2. Penyesuaian data masih belum sempurna, sehingga pembandingan cepstrum inputan dengan cepstrum standart masih sangat sulit disamakan.
3. Nilai treshold mempengaruhi berhasil tidaknya user untuk mengeksekusi program, treshold dengan nilai 1,00 adalah nilai treshold terbaik yang bisa digunakan untuk megeksekusi program.

5.2 Saran

Mengingat masih banyaknya hal-hal yang belum dapat diimplementasikan pada skripsi ini, maka kami mempertimbangkan beberapa saran untuk perbaikan-perbaikan proyek akhir kami ini dalam hal :

1. Tingkat pengurangan noise pada waktu perekaman suara yang semakin baik.
2. Desain tampilan software yang semakin baik. Dalam hal ini penulis menyarankan untuk menggunakan visual Tcl.

3. Otomatis sistem, sehingga pengguna tidak perlu lagi menggerakkan mouse untuk memberikan perintah.
4. Waktu pemrosesan sinyal suara untuk mendapatkan ciri atau parameter dipersingkat dengan membuat algoritma sistem yang lebih baik.
5. Pengambilan sample yang lebih banyak lagi untuk tiap dependent speaker agar bisa didapatkan hasil yang lebih akurat.

DAFTAR PUSTAKA

[1] Snack url: http://wiki.tcl.tk/2647

[2] Fast Fourier Transform url:
http://en.wikipedia.org/wiki/Fast_Fourier_transform

[3] Windowing System url : http://en.wikipedia.org/wiki/Windowing_system

[4] Marshall T. Rose. " Tcl/Tk A Developer's Guide" , San Francisco, 2003.

[5] Introduction Tcl url: http://en.wikipedia.org/wiki/Tcl

[6] Dynamic Time Warping url:
http://en.wikipedia.org/wiki/Dynamic_time_warping

[7] Tcl-Snack url:
http://rpmfind.net/linux/RPM/mandriva/devel/cooker/ppc/media/contrib/release/tcl-snack-2.2.10-3mdk.ppc.html

[8] Tanudja Harlianto. "Pengolahan sinyal digital dan system pemrosesan sinyal", Andi Yogyakarta, 2007

[9] Ekstraksi Ciri Sinyal Wicara url.
http://www.eepis-its.edu/~tribudi/LN_SIP_Prak/rev_01_Speech_prak_5_Matlab.pdf

[10] Speech Recognition With Windows XP url:
http://www.microsoft.com/windowsxp/using/setup/expert/moskowitz_02september23.mspx

LAMPIRAN

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama Mahasiswa	: Abdi Sofyandi
NIM	: 0412645
Jurusan	: Teknik Elektro S1
Konsentrasi	: Teknik Komputer dan Informatika
Judul Skripsi	: Pengenalan Suara Dengan Metode *Dynamic Time Warping* Untuk Menjalankan Program Aplikasi User

Dipertahankan dihadapan tim penguji skripsi jenjang Strata Satu (S-1) pada:

Hari : Jumat

Tanggal : 26 September 2008

Dengan Nilai : 79,8 (B^+)

PANITIA UJIAN SKRIPSI

KETUA

Ir. Mochtar Asroni, MSME
NIP.Y. 1018100036

SEKRETARIS

Ir. F. Yudi Limpraptono, MT
NIP.Y. 1039500274

ANGGOTA PENGUJI

PENGUJI I

Yan Watequlis, ST
NIP. P. 1030400403

PENGUJI II

Ir. F. Yudi Limpraptono, MT
NIP.Y. 1039500274

Formulir Perbaikan Skripsi

Dalam pelaksanaan ujian skripsi jenjang Strata satu (S-1) Jurusan Teknik Elektro konsentrasi Teknik Komputer dan Informatika, maka perlu adanya perbaikan skripsi untuk Mahasiswa :

Nama : Abdi Sofyandi
NIM : 04.12.645
Jurusan : Teknik Elektro S-1
Konsentrasi : Teknik Komputer dan Informatika
Masa Bimbingan : 1 Agustus 2008 s/d 1 Januari 2009
Judul Skripsi : Pengenalan Suara Dengan Metode *Dynamic Time Warping* Untuk Menjalankan Program Aplikasi User.

Tanggal	Uraian	Paraf
Penguji I 26-Sep-08	1. Perbaikan Program.	
Penguji II 26-Sep-08	1. Bab II berjudul Teori Penunjang. 2. Perbaikan Program	

Mengetahui,

Dosen Pembimbing I

I Komang Somawirata, ST, MT
NIP.Y. 1030100361

Dosen Pembimbing II

M. Ashar, ST, MT
NIP. P.1030500408

Dosen Penguji,

PENGUJI I

Yan Watequlis, ST
NIP. P. 1030400403

PENGUJI II

Ir. F. Yudi Limpraptono, MT
NIP.Y. 1039500274

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK

N) (PERSERO) MALANG
ANK NIAGA MALANG

Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting) Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

Malang, 11 Agt, 2008

Nomor : ITN-231/I.TA/2/08
Lampiran : -
Perihal : BIMBINGAN SKRIPSI

Kepada : Yth. Sdr. I KOMANG SOMAWIRATA, ST, MT
Dosen Institut Teknologi Nasional Malang

Dosen Pembimbing
Jurusan Teknik Elektro S-1
di
Malang

Dengan hormat
Sesuai dengan permohonan dan persetujuan dalam Proposal Skripsi
Untuk Mahasiswa :

Nama	: ABDI SOFYANDI
Nim	: 0412645
Fakultas	: Teknologi Industri
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Komputer & Informatika

Maka dengan ini pembimbingan tersebut kami serahkan sepenuhnya kepada Saudara/i selama masa waktu (enam) 6 bulan, terhitung mulai tanggal :

1 Agustus 2008 s/d 1 Februari 2009

Sebagai satu syarat untuk menempuh ujian Sarjana Teknik,
Jurusan Teknik Elektro S-1
Demikian agar maklum dan atas perhatian serta bantuannya kami sampaikan terima kasih

Ketua Jurusan
Teknik Elektro S-1

Ir. F. Yudi Limpraptono, MT
Nip. Y. 1039500274

Tembusan Kepada Yth :
1. Mahasiswa Yang Bersangkutan
2. Arsip

Form. S 4a

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK

(PERSERO) MALANG
NIAGA MALANG

Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting) Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

Malang, 11 Agt, 2008

omor : ITN-232/I.TA/2/08
ampiran : -
erihal : BIMBINGAN SKRIPSI

epada : Yth. Sdr. **M. ASHAR, ST, MT**
Dosen Institut Teknologi Nasional Malang

Dosen Pembimbing
Jurusan Teknik Elektro S-1
di
Malang

Dengan hormat
Sesuai dengan permohonan dan persetujuan dalam Proposal Skripsi
Untuk Mahasiswa :

Nama	: ABDI SOFYANDI
Nim	: 0412645
Fakultas	: Teknologi Industri
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Komputer & Informatika

Maka dengan ini pembimbingan tersebut kami serahkan sepenuhnya kepada Saudara/i selama masa waktu (enam) 6 bulan, terhitung mulai tanggal :

1 Agustus 2008 s/d 1 Februari 2009

Sebagai satu syarat untuk menempuh ujian Sarjana Teknik,
Jurusan Teknik Elektro S-1
Demikian agar maklum dan atas perhatian serta bantuannya kami sampaikan terima kasih

Ketua Jurusan
Teknik Elektro S-1

Ir. F. Yudi Limpraptono, MT
Nip. Y. 1039500274

Tembusan Kepada Yth :
1. Mahasiswa Yang Bersangkutan
2. Arsip

Form. S 4a

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

FORMULIR BIMBINGAN SKRIPSI

Nama : **ABDI SOFYANDI**
Nim : **04.12.645**
Masa Bimbingan : **1 AGUSTUS 2008 s/d 1 JANUARI 2009**
Judul Skripsi : **APLIKASI PENGENALAN SUARA UNTUK MENJALANKAN PROGRAM APLIKASI USER**

No.	Tanggal	Uraian	Paraf Pembimbing
1.	29/08	Pengajuan judul skripsi	
2.	9/10	Konsul bab I & bab II	
3.	16/10	Konsultasi bab II	
4.	9/11	Konsultasi perbaikan bab II & bab III	
5.	03/12	Acc maju skripsi	
6.			
7.			
8.			
9.			
10.			

Malang,
Dosen Pembimbing

I Komang Somawirata, ST.,MT.
NIP : 1030100361

Form S-4B

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

FORMULIR BIMBINGAN SKRIPSI

Nama : ABDI SOFYANDI
Nim : 04.12.645
Masa Bimbingan : 1 AGUSTUS 2008 s/d 1 JANUARI 2009
Judul Skripsi : APLIKASI PENGENALAN SUARA UNTUK MENJALANKAN PROGRAM APLIKASI USER

No.	Tanggal	Uraian	Paraf Pembimbing
1.		Bab I, II	
2.		Perancangan system	
3.		Bab III (Revisi)	
4.		Sinyal FFT (Revisi)	
5.		Frame blocking	
6.		Bab IV (Revisi)	
7.		Pengujian database	
8.		Pengujian speech	
9.		Makalah hasil	
10.		Bab V (Revisi)	

Malang,
Dosen Pembimbing

M. ASHAR, ST.MT
NIP. 1030500408

Form S-4B

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T Elektronika / T.Infokom, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : Abdi Setyawan
N I M : 04 12 645
Perbaikan meliputi :

Ujian ulang lagi

Malang, 26/9 2008

(TAN ...)

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T. Elektronika / T.Infokom, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : ABDI
N I M : 0412 645
Perbaikan meliputi :

1. Bab II bagian Teori Penunjang.
2. Perbaiki program.

Malang, 200

( )

➢ **Program abdi.tcl**

```
#!/bin/sh
# the next line restarts using wish \
exec wish8.4 "$0" "$@"

load lib\\libglobal

package require snack

#wm geometry . 450x380+50+50
wm geometry . 450x380+85+80
wm overrideredirect . 1

snack::sound u
snack::sound v
snack::sound w
snack::sound x
snack::sound z -channels 2
snack::sound snd

u configure -rate 12000 -channels MONO -encoding LIN16
image create photo test -file abdi1.gif

#GUI
canvas .c -height 350 -width 450 -bg #0000bb
pack [frame .a1]
pack [frame .a4]


button .brecord -relief raised -activebackground darkblue -activeforeground white
-disabledforeground grey -text "START" -command {
u record:z record
after 1500 tick
}

label .value -textvariable IntValue -fg white
button .bexit -relief raised -activebackground darkblue -activeforeground white -
disabledforeground grey -text "EXIT" -command exit -cursor pirate
pack .c
        .c create image 100 5 -image test -anchor nw
        .c create waveform  10 75 -sound u -height 100 -width 200 -tags sect
        .c create waveform  260 75 -sound snd -height 100 -width 200 -tags sect

pack .brecord .bexit .value -side left -padx 3.2 -pady 1

proc tick {} {
   global u
```

```
    u stop
    u input
    u dtw
    w length 1000
    w signal
    w play
    z stop
    Announce

}

proc SetValue value {
global IntValue
set IntValue [6000000000000000]
}

bind .  {destroy .}
bind .  {destroy .}
focus .

proc Announce {} {
        set data [read [open data\\hasil\\out.txt]]

        if { $data == 0 } {
                .a4 configure -text "Status ready"
        }
        if { $data == 1 } {
                exec WINWORD.exe &
        }
        if { $data == 2 } {
                exec WINWORD.exe &
        }
        if { $data == 3 } {
                exec WINWORD.exe &
        }
        if { $data == 4 } {
                exec WINWORD.exe &
        }
        if { $data == 5 } {
                exec WINWORD.exe &
        }
        if { $data == 6 } {
                exec WINWORD.exe &
        }
        if { $data == 7 } {
                exec WINWORD.exe &
        }
        if { $data == 8 } {
```

```
        exec WINWORD.exe &
}
if { $data == 9 } {
        exec WINWORD.exe &
}
if { $data == 10 } {
        exec WINWORD.exe &
}
if { $data == 11 } {
        exec WINWORD.exe &
}
if { $data == 12 } {
        exec WINWORD.exe &
}
if { $data == 13 } {
        exec WINWORD.exe &
}
if { $data == 14 } {
        exec WINWORD.exe &
}
if { $data == 15 } {
        exec WINWORD.exe &
}
if { $data == 16 } {
        exec WINWORD.exe &
}
if { $data == 17 } {
        exec WINWORD.exe &
}
if { $data == 18 } {
        exec WINWORD.exe &
}
if { $data == 19 } {
        exec WINWORD.exe &
}
if { $data == 20 } {
        exec WINWORD.exe &
}
if { $data == 21 } {
        exec EXCEL.exe &
}
if { $data == 22 } {
        exec EXCEL.exe &
}
if { $data == 23 } {
        exec EXCEL.exe &
}
if { $data == 24 } {
```

```
        exec EXCEL.exe &
}
if { $data == 25 } {
        exec EXCEL.exe &
}
if { $data == 26 } {
        exec EXCEL.exe &
}
if { $data == 27 } {
        exec EXCEL.exe &
}
if { $data == 28 } {
        exec EXCEL.exe &
}
if { $data == 29 } {
        exec EXCEL.exe &
}
if { $data == 30 } {
        exec EXCEL.exe &
}
if { $data == 31 } {
        exec EXCEL.exe &
}
if { $data == 32 } {
        exec EXCEL.exe &
}
if { $data == 33 } {
        exec EXCEL.exe &
}
if { $data == 34 } {
        exec EXCEL.exe &
}
if { $data == 35 } {
        exec EXCEL.exe &
}
if { $data == 36 } {
        exec EXCEL.exe &
}
if { $data == 37 } {
        exec EXCEL.exe &
}
if { $data == 38 } {
        exec EXCEL.exe &
}
if { $data == 39 } {
        exec EXCEL.exe &
}
if { $data == 40 } {
```

```
        exec EXCEL.exe &
}
if { $data == 41 } {
        exec POWERPNT.exe &
}
if { $data == 42 } {
        exec POWERPNT.exe &
}
if { $data == 43 } {
        exec POWERPNT.exe &
}
if { $data == 44 } {
        exec POWERPNT.exe &
}
if { $data == 45 } {
        exec POWERPNT.exe &
}
if { $data == 46 } {
        exec POWERPNT.exe &
}
if { $data == 47 } {
        exec POWERPNT.exe &
}
if { $data == 48 } {
        exec POWERPNT.exe &
}
if { $data == 49 } {
        exec POWERPNT.exe &
}
if { $data == 50 } {
        exec POWERPNT.exe &
}
if { $data == 51 } {
        exec POWERPNT.exe &
}
if { $data == 52 } {
        exec POWERPNT.exe &
}
if { $data == 53 } {
        exec POWERPNT.exe &
}
if { $data == 54 } {
        exec POWERPNT.exe &
}
if { $data == 55 } {
        exec POWERPNT.exe &
}
if { $data == 56 } {
```

```
        exec POWERPNT.exe &
}
if { $data == 57 } {
        exec POWERPNT.exe &
}
if { $data == 58 } {
        exec POWERPNT.exe &
}
if { $data == 59 } {
        exec POWERPNT.exe &
}
if { $data == 60 } {
        exec POWERPNT.exe &
}
if { $data == 61 } {
        exec explorer &
}
if { $data == 62 } {

        exec explorer &
}
if { $data == 63 } {

exec explorer &
}
if { $data == 64 } {

exec explorer &
}
if { $data == 65 } {

exec explorer &
}
if { $data == 66 } {

exec explorer &
}
if { $data == 67 } {

exec explorer &
}
if { $data == 68 } {

exec explorer &
}
if { $data == 69 } {

exec explorer &
```

```
}
if { $data == 70 } {

exec explorer &
}
if { $data == 71 } {

exec explorer &
}
if { $data == 72 } {

exec explorer &
}
if { $data == 73 } {

exec explorer &
}
if { $data == 74 } {

exec explorer &
}
if { $data == 75 } {

exec explorer &
}
if { $data == 76 } {

exec explorer &
}
if { $data == 77 } {

exec explorer &
}
if { $data == 78 } {

exec explorer &
}
if { $data == 79 } {

exec explorer &
}
if { $data == 80 } {

exec explorer &
}
if { $data == 81 } {

exec notepad &
```

```
}
if { $data == 82 } {

exec notepad &
}
if { $data == 83 } {

        exec notepad &
}
if { $data == 84 } {

        exec notepad &
}
if { $data == 85 } {

        exec notepad &
}
if { $data == 86 } {

        exec notepad &
}
if { $data == 87 } {

        exec notepad &
}
if { $data == 88 } {

        exec notepad &
}
if { $data == 89 } {

        exec notepad &
}
if { $data == 90 } {

        exec notepad &
}
if { $data == 91 } {

        exec notepad &
}
if { $data == 92 } {

        exec notepad &
}
if { $data == 93 } {

        exec notepad &
```

```
        }
        if { $data == 94 } {

                exec notepad &
        }
        if { $data == 95 } {

                exec notepad &
        }
        if { $data == 96 } {

                exec notepad &
        }
        if { $data == 97 } {

                exec notepad &
        }
        if { $data == 98 } {

                exec notepad &
        }
        if { $data == 99 } {

                exec notepad &
        }
        if { $data == 100 } {

                exec notepad &
        }

}
```

- **Program abdi_record.tcl**

```
#!/bin/sh
# the next line restarts using wish \
exec wish8.4 "$0" "$@"

package require -exact snack 2.2
package require -exact snack 2.2

load lib\\libglobal

snack::sound z -channels 2
```

```
snack::sound x -file sound.wav -debug 0
snack::sound y
x configure -rate 12000 -encoding LIN16 -channels MONO
y configure -rate 12000 -encoding LIN16 -channels MONO

set width 600
set height 200
set pps 300
set color black
set frame 1

option add *font {Helvetica 10 bold}

pack [frame .f1]
pack [frame .f2]
pack [frame .f3]
pack [frame .f4]
pack [frame .f5]
pack [label .f1.l1 -text "APLIKASI SPEECH RECOGNITION UNTUK
MENJALANKAN\n PROGRAM APLIKASI USER"]
pack [label .f1.l2 -text "Mode Operasi Recording"]


canvas .f1.c -bg black -width 300 -height 200
snack::levelMeter .f1.lm -width 20 -length 200 -orient vertical -oncolor red
pack .f1.lm .f1.c -side left -padx 5

pack [label .f2.time -text "0.000 sec" -width 10]

button .f3.b1 -text Play -command {y play}
button .f3.b2 -text Rec -command {y record;z record}
button .f3.b3 -text Stop -command {y stop;z stop;y input;y dtbase}
button .f3.b4 -text save -command {y write data_wav\\$filename.wav;y input}
button .f3.b5 -text load -command {y read data_wav\\$filename.wav;y input}
button .f3.b6 -text Exit -command exit
entry .f3.entry -width 20 -textvariable filename
pack .f3.b1 .f3.b2 .f3.b3 .f3.entry .f3.b4 .f3.b5 .f3.b6 -side left -padx 5 -pady 5

label .f4.1 -text "Oleh Abdi Sofyandi"
label .f4.2 -text "Teknik Komputer & Informatika \n ITN Malang"
pack .f4.1 .f4.2

after 1 Update

proc Update {} {
        set l [z max -start 0 -end -1 -channel 0]
        .f2.time config -text [format "%.3f sec" [y length -unit seconds]]
        z length 0
```

```
        .f1.lm configure -level $l
        after 50 Update
}
.f1.c itemconf wave -pixelspersecond 300 -width 300
.f1.c create waveform 152 102 -anchor c -sound y -height $height -tags wave -
debug 0 -zerolevel 0 -frame $frame -fill white
```

- **Program global.c**

```
include <math.h>
#include <windows.h>
#include <stdio.h>
#include <string.h>
#include "snack.h"

#define     beki    9
#define     nlpc    256
#define     inv         1.0f
#define     inv2    -1.0f

static float G[300][300];
static float st_CEP[300][16];
static float in_CEP[300][16];

static int st_blk,in_blk;

float DP_match1(int);
float distance(int,int);
//void trace();

int Signal(ClientData cdata, Tcl_Interp *interp, int objc, Tcl_Obj *CONST
objv[])
{
        Sound *sound;
        int i;

        sound = Snack_GetSound(interp, Tcl_GetStringFromObj(objv[0],
NULL));
        for(i=0;i<Snack_GetLength(sound);i++)
        {
                if((i/10)%2)
                {
                        Snack_SetSample(sound,0,i,10000);
                }
                else
                {
                        Snack_SetSample(sound,0,i,-10000);
                }
```

```
	}

	return TCL_OK;
}

int Reset(ClientData cdata, Tcl_Interp *interp, int objc, Tcl_Obj *CONST objv[])
{
	FILE *freset;

	freset=fopen("data\\hasil\\out.txt","w");
	fprintf(freset,"%d\n",0);

	fclose(freset);

	return TCL_OK;
}

int front_end_clip(xx,yy,jml_sin)
float *xx,*yy;
int jml_sin;
{
	int i,mulai,akhir,jml_sin_new;
	float jml=0,mean,dev,jumdev=0,sd,batas;

	for(i=0;i<jml_sin;i++)
		jml+=xx[i];
	mean=jml/jml_sin;

	for(i=0;i<jml_sin;i++)
	{
		dev=(float)fabs(xx[i]-mean);
		jumdev+=dev*dev;
	}

	sd=(float)sqrt(jumdev/jml_sin);
	batas=mean+3*sd;

	for(i=0;i<jml_sin;i++)
	{
		if(xx[i]>=batas)
		{
			mulai=i;
			break;
		}
	}

	for(i=jml_sin;i>0;i--)
	{
```

```
            if(xx[i]>=batas)
            {
                akhir=i;
                break;
            }
        }

        jml_sin_new=0;
        for(i=mulai;i<akhir;i++)
        {
            yy[jml_sin_new]=xx[i];
            jml_sin_new++;
        }
        return(jml_sin_new);
}

FILE
*fsignal,*finfo,*fframe,*fsignalframe,*fhamm,*ffft,*flift,*flog,*fceps,*fspec;
Sound *sound;

double temp_log1;
float x[30000],y[30000],temp[30000];
float real[512],imag[512];
float ms,spd;
int j,k,frame,frame_no,signal_length,signal_length_new,start,orde;

void hammr();
void cal_fft();

int Input(ClientData cdata, Tcl_Interp *interp, int objc, Tcl_Obj *CONST objv[])
{
        fsignal=fopen("data\\input\\sinyal_in.txt","w");
        finfo=fopen("data\\input\\info_in.txt","w");
        fframe=fopen("data\\input\\frame_in.txt","w");
        fsignalframe=fopen("data\\input\\signalframe_in.txt","w");
        fhamm=fopen("data\\input\\hamm_in.txt","w");
        ffft=fopen("data\\input\\fft_in.txt","w");
        flift=fopen("data\\input\\lift_in.txt","w");
        flog=fopen("data\\input\\log_in.txt","w");
        fspec=fopen("data\\input\\spec_in.txt","w");
        fceps=fopen("data\\input\\ceps_in.txt","w");

        /////// Get the sound structure for this sound ///////

        sound = Snack_GetSound(interp, Tcl_GetStringFromObj(objv[0],
NULL));

        signal_length = Snack_GetLength(sound);
```

```
for (k=0;k<signal_length;k++)
{
        x[k]=(float)Snack_GetSample(sound, 0, k);
        fprintf(fsignal,"%f\n",x[k]);
}

signal_length_new = front_end_clip(x,&y,signal_length);
for (k=0;k<signal_length_new;k++)
        fprintf(finfo,"%f\n",y[k]);

frame_no = signal_length_new/120;
fprintf(fframe,"%d\n",frame_no);

start=0;
//      ms=0;
//      spd=(float)20/240;

for(frame=0;frame<frame_no-1;frame++)
{
        start=(frame)*120;
        for(k=0;k<240;k++)
        {
                temp[k]=y[start];
                fprintf(fsignalframe,"%f\n",temp[k]);
                start++;
        }
}

for(frame=0;frame<frame_no-1;frame++)
{
        for (k=0;k<240;k++)
        {
                real[k]=temp[k];
                imag[k]=0;
        }

        ////// hamming window //////
        ////////////////////////////

        hammr(real,imag,nlpc);

        for(k=0;k<256;k++)
                {
                        real[j]=0;
                        imag[j]=0;
                }
                //fprintf(fhamm,"%f\t%f\n",ms=ms+spd,real[k]);
```

```
fprintf(fhamm,"%f\t%f\n",real[k]);


////// FFT //////

cal_fft(real,imag,beki,inv);

////// Mutlak Log //////

for(k=0;k<512;k++)
{
        temp_log1=(double)(real[k]*real[k]+imag[k]*imag[k]);
        if(temp_log1>0)
                real[k]=(float)log10(temp_log1);
        else
                real[k]=(float)log10(0.000001);
        imag[k]=0;

        fprintf(ffft,"%f\t%f\n",k*6000.f/256.f,real[k]);
}

////// IFFT //////

cal_fft(real,imag,beki,inv2);

/*      //////Liftering//////

for(k=16;k<512-16;k++)
{
        real[k]=0.0;
        imag[k]=0.0;
}*/

for(k=0;k<256;k++)
        fprintf(flift,"%f\n",real[k]);

////// Mutlak Log (Data Sample yg diambil) //////

for(k=0;k<16;k++)
{
        temp_log1=(double)(real[k]*real[k]+imag[k]*imag[k]);
        if(temp_log1>0)
                real[k]=(float)log10(temp_log1);
        else
                real[k]=(float)log10(0.000001);
        imag[k]=0;

        fprintf(flog,"%f\n",real[k]);
```

```
            }

            ////////Jika orde =256 adalah spektrum/////////
            //Jika orde antara 10 sampai 30 adalah kepstrum//

//          orde=16; //////

            ////// Spectrum //////

            cal_fft(real,imag,beki,inv);

/*          for(k=0;k<orde;k++)
            {
                temp_log1=(double)(real[k]*real[k]+imag[k]*imag[k]);
                if(temp_log1>0)
                    real[k]=(float)log10(temp_log1);
                else
                    real[k]=(float)log10(0.000001);
                imag[k]=0;

                fprintf(fspec,"%f\t%f\n",k*6000.f/256.f,real[k]);
            }*/

            ////// Cepstrum //////

            for(k=orde;k<512-orde;k++)
            {
                real[k]=0.0;
                imag[k]=0.0;
            }

            cal_fft(real,imag,beki,inv);

            for(k=0;k<256;k++)
            {
                temp_log1=(double)(real[k]*real[k]+imag[k]*imag[k]);
                if(temp_log1>0)
                    real[k]=(float)log10(temp_log1);
                else
                    real[k]=(float)log10(0.000001);
                imag[k]=0;

                fprintf(fceps,"%f\t%f\n",k*6000.f/256.f,real[k]);
            }
        }

        fclose(fsignal);
        fclose(finfo);
```

```
        fclose(fframe);
        fclose(fsignalframe);
        fclose(fhamm);
        fclose(ffft);
        fclose(flift);
        fclose(flog);
        fclose(fspec);
        fclose(fceps);

        return TCL_OK;
        //return TCL_OK;
}


FILE
*fsignal,*finfo,*fframe,*fsignalframe,*fhamm,*ffft,*flift,*flog,*fceps,*spec;
Sound *sound;

double temp_log1;
float x[30000],y[30000],temp[30000];
float real[512],imag[512];
float ms,spd;
int k,frame,frame_no,signal_length,signal_length_new,start,orde;

void hammr();
void cal_fft();

int dtbase(ClientData cdata, Tcl_Interp *interp, int objc, Tcl_Obj *CONST
objv[])
{
fsignal=fopen("data\\standart\\sinyal_x.txt","w");
        finfo=fopen("data\\standart\\info_x.txt","w");
        fframe=fopen("data\\standart\\frame_x.txt","w");
        fsignalframe=fopen("data\\standart\\signalframe_x.txt","w");
        fhamm=fopen("data\\standart\\hamm_x.txt","w");
        ffft=fopen("data\\standart\\fft_x.txt","w");
        flift=fopen("data\\standart\\lift_x.txt","w");
        flog=fopen("data\\standart\\log_x.txt","w");
        fspec=fopen("data\\standart\\spec_x.txt","w");
        fceps=fopen("data\\standart\\ceps_x.txt","w");


        sound = Snack_GetSound(interp, Tcl_GetStringFromObj(objv[0],
NULL));

        signal_length = Snack_GetLength(sound);

        for (k=0;k<signal_length;k++)
```

```
{
        x[k]=(float)Snack_GetSample(sound, 0, k);
        fprintf(fsignal,"%f\n",x[k]);
}

signal_length_new = front_end_clip(x,&y,signal_length);
for (k=0;k<signal_length_new;k++)
        fprintf(finfo,"%f\n",y[k]);

frame_no = signal_length_new/120;
fprintf(fframe,"%d\n",frame_no);

start=0;
ms=0;
spd=(float)20/240;

for(frame=0;frame<frame_no-1;frame++)
{
        start=(frame)*120;
        for(k=0;k<240;k++)
        {
                temp[k]=y[start];
                fprintf(fsignalframe,"%f\n",temp[k]);
                start++;
        }
}

for(frame=0;frame<frame_no-1;frame++)
{
        for (k=0;k<240;k++)
        {
                real[k]=temp[k];
                imag[k]=0;
        }

        ////// hamming window //////
        ////////////////////////////

        hammr(real,imag,nlpc);

        for(k=0;k<256;k++)
                fprintf(fhamm,"%f\t%f\n",ms=ms+spd,real[k]);

        ////// FFT //////

        cal_fft(real,imag,bcki,inv);

        ////// Mutlak Log //////
```

```
for(k=0;k<512;k++)
{
        temp_log1=(double)(real[k]*real[k]+imag[k]*imag[k]);
        if(temp_log1>0)
                real[k]=(float)log10(temp_log1);
        else
                real[k]=(float)log10(0.000001);
        imag[k]=0;

        fprintf(ffft,"%f\t%f\n",k*6000.f/256.f,real[k]);
}

////// IFFT //////

cal_fft(real,imag,beki,inv2);

///////Liftering//////

for(k=16;k<512-16;k++)
{
        real[k]=0.0;
        imag[k]=0.0;
}

for(k=0;k<256;k++)
        fprintf(flift,"%f\n",real[k]);

////// Mutlak Log (Data Sample yg diambil) //////

for(k=0;k<16;k++)
{
        temp_log1=(double)(real[k]*real[k]+imag[k]*imag[k]);
        if(temp_log1>0)
                real[k]=(float)log10(temp_log1);
        else
                real[k]=(float)log10(0.000001);
        imag[k]=0;

        fprintf(flog,"%f\n",real[k]);
}

/////////Jika orde =256 adalah spektrum/////////
//Jika orde antara 10 sampai 30 adalah kepstrum//

/*      orde=256; //////

////// Spectrum //////
```

```
        cal_fft(real,imag,beki,inv);

        for(k=0;k<orde;k++)
        {
                temp_log1=(double)(real[k]*real[k]+imag[k]*imag[k]);
                if(temp_log1>0)
                        real[k]=(float)log10(temp_log1);
                else
                        real[k]=(float)log10(0.000001);
                imag[k]=0;

                fprintf(fspec,"%f\t%f\n",k*6000.f/256.f,real[k]);
        }*/

        ////// Cepstrum //////

        for(k=orde;k<512-orde;k++)
        {
                real[k]=0.0;
                imag[k]=0.0;
        }

        cal_fft(real,imag,beki,inv);

        for(k=0;k<512;k++)
        {
                temp_log1=(double)(real[k]*real[k]+imag[k]*imag[k]);
                if(temp_log1>0)
                        real[k]=(float)log10(temp_log1);
                else
                        real[k]=(float)log10(0.000001);
                imag[k]=0;

                fprintf(fceps,"%f\t%f\n",k*6000.f/256.f,real[k]);
        }
}

fclose(fsignal);
fclose(finfo);
fclose(fframe);
fclose(fsignalframe);
fclose(fhamm);
fclose(ffft);
fclose(flift);
fclose(flog);
fclose(fspec);
fclose(fceps);
```

```
        return TCL_OK;
}
/***** Menghitung jarak antara sinyal input dan standar word 1 *****/

        fframe=fopen("data\\standart\\frame_w1.txt","r");
        fscanf(fframe,"%d\n",&st_blk);
        fclose(fframe);

        for(i=0;i<st_blk;i++)
        {
                for(j=0;j<16;j++)
                {
                        fscanf(fstd1,"%f\t%f\n",&in_1,&std_data_1);
                        st_CEP[i][j]=std_data_1;
                }
        }

        r=7;

        DP_dist=DP_match1(r);
        hasil[1]=DP_dist;
//      trace();

        /***** Menghitung jarak antara sinyal input dan standar word 2 *****/

        fframe=fopen("data\\standart\\frame_w2.txt","r");
        fscanf(fframe,"%d\n",&st_blk);
        fclose(fframe);

        for(i=0;i<st_blk;i++)
        {
                for(j=0;j<16;j++)
                {
                        fscanf(fstd2,"%f\t%f\n",&in_2,&std_data_2);
                        st_CEP[i][j]=std_data_2;
                }
        }

        r=7;

        DP_dist=DP_match1(r);
        hasil[2]=DP_dist;
//      trace();

        /***** Menghitung jarak antara sinyal input dan standar word 3 *****/

        fframe=fopen("data\\standart\\frame_w3.txt","r");
```

```
fscanf(fframe,"%d\n",&st_blk);
fclose(fframe);

for(i=0;i<st_blk;i++)
{
        for(j=0;j<16;j++)
        {
                fscanf(fstd3,"%f\t%f\n",&in_3,&std_data_3);
                st_CEP[i][j]=std_data_3;
        }
}

r=7;

DP_dist=DP_match1(r);
hasil[3]=DP_dist;
//      trace();

/***** Menghitung jarak antara sinyal input dan standar word 4 *****/

fframe=fopen("data\\standart\\frame_w4.txt","r");
fscanf(fframe,"%d\n",&st_blk);
fclose(fframe);

for(i=0;i<st_blk;i++)
{
        for(j=0;j<16;j++)
        {
                fscanf(fstd4,"%f\t%f\n",&in_4,&std_data_4);
                st_CEP[i][j]=std_data_4;
        }
}

r=7;

DP_dist=DP_match1(r);
hasil[4]=DP_dist;
//      trace();

/***** Menghitung jarak antara sinyal input dan standar word 5 *****/

fframe=fopen("data\\standart\\frame_w5.txt","r");
fscanf(fframe,"%d\n",&st_blk);
fclose(fframe);

for(i=0;i<st_blk;i++)
{
        for(j=0;j<16;j++)
```

```
                {
                        fscanf(fstd5,"%f\t%f\n",&in_5,&std_data_5);
                        st_CEP[i][j]=std_data_5;
                }
        }

        r=7;

        DP_dist=DP_match1(r);
        hasil[5]=DP_dist;
//      trace();

/*
        Initialize the square package and create a new sound command 'global'.
        The syntax is: sndName global
*/

EXPORT(int, Global_Init)(Tcl_Interp *interp)
{
        #ifdef USE_TCL_STUBS
        if (Tcl_InitStubs(interp, "8", 0) == NULL) {
                return TCL_ERROR;
        }
        #endif

        #ifdef USE_SNACK_STUBS
        if (Snack_InitStubs(interp, "2", 0) == NULL) {
                return TCL_ERROR;
        }
        #endif

        if (Tcl_PkgProvide(interp, "global", "1.0") != TCL_OK) {
                return TCL_ERROR;
        }

        Snack_AddSubCmd(SNACK_SOUND_CMD, "input",(Snack_CmdProc
*)Input,NULL);
        Snack_AddSubCmd(SNACK_SOUND_CMD, "dtbase",(Snack_CmdProc
* )dtbase,NULL);
        Snack_AddSubCmd(SNACK_SOUND_CMD, "dtw",(Snack_CmdProc *)
Dtw,NULL);
        Snack_AddSubCmd(SNACK_SOUND_CMD, "reset",(Snack_CmdProc
*) Reset,NULL);
        Snack_AddSubCmd(SNACK_SOUND_CMD, "signal",(Snack_CmdProc
*) Signal,NULL);

        return TCL_OK;
}
```

```
EXPORT(int, Global_SafeInit)(Tcl_Interp *interp)
{
        return Global  Init(interp);
}

#ifdef __cplusplus
#endif

/*void hammr(x,y,n)
float *x,*y;
int n;
{
        int i;

        --x;
        --y;

        for(i=0;i<=n;++i)
        {
                x[i]=x[i]*(float)(0.54-0.46*cos(2.0*3.141592654*(i)/(float)n));
                y[i]=y[i]*(float)(0.54-0.46*cos(2.0*3.141592654*(i)/(float)n));
        }
}*/

void hammr(x,y,n)
float *x,*y;
  int n;
{
  int i,N;

  float pi=3.141592654,b;

  N=(int)pow(2,n);

  for(i=1;i<=N;i++)
  {
      b=0.54-0.46*cos(2.0*pi*(double)i/((double)N-1.0));
      x[i]=x[i]*b;
      y[i]=y[i]*b;
  }
}


void cal_fft(x,y,l,mode)
float *x,*y,mode;
int l;
{
```

```
int np,lmx,lo,lix,lm,li,j1,j2,nv2,npm1,i,j,k;
float scl,arg,c,s,t1,t2;

for(i=0;i<pow(2,l);i++)
        y[i]=0;

--x;
--y;

/* ======= radix-2 fft ======= */

np=(int)pow(2.0,(float)(l));
lmx=np;
scl=(float)(6.283185303/(float)np);

for(lo=1;lo<=l;++lo)
{
        lix=lmx;
        lmx=(int)(lmx/2.0);
        arg=0.0;
        for(lm=1;lm<=lmx;++lm)
        {
                c=(float)cos(arg);
                s=(float)(mode*sin(arg));
                arg=arg+scl;
                for (li=lix; lix<0 ? li>=np : li<=np; li+=lix)
                {
                        j1=li-lix+lm;
                        j2=j1+lmx;
                        t1=x[j1]-x[j2];
                        t2=y[j1]-y[j2];
                        x[j1]=x[j1]+x[j2];
                        y[j1]=y[j1]+y[j2];
                        x[j2]=c*t1+s*t2;
                        y[j2]=c*t2-s*t1;
                }
        }
        scl=(float)(2.0*scl);
}
```

```
        /* ====== bit reversal ====== */

        j=1;
        nv2=(int)(np/2.0);
        npm1=np-1;
        for(i=1;i<=npm1;++i)
        {
                if(i>=j)
                        goto L30;
                t1=x[j];
                t2=y[j];
                x[j]=x[i];
                y[j]=y[i];
                x[i]=t1;
                y[i]=t2;

                L30:
                k=nv2;

                L40:
                if(k>=j)
                        goto L50;

                j-=k;
                k=(int)(k/2.0);
                goto L40;

                L50:
                j=j+k;
        }
}

float DP_match1(r)
int r;
{
        int i,j;
        int I,J,up,dp;
        float dist,g,g0,g1,g2,g3,a;
        float DP_mdist=0.0;

        for(i=0;i<in_blk;i++)
        {
                for(j=0;j<st_blk;j++)
                {
                        G[i][j] = (float)1.0e+30;
                }
        }
        I=in_blk-1;
```

```
        J=st_blk-1;
        a=(float)st_blk/(float)in_blk;
        dist=distance(0,0);

        G[0][0]=(float)(2.0*dist);
        dist=0.0;

        for(i=0;i<=I;i++)
        {
                up=(int)(a*i+r);
                if(up>J)
                        up=J;
                dp=(int)(a*i-r);
                if(dp<0)
                        dp=0;

                for(j=dp;j<=up;j++)
                {
                        if(i==0 && j==0)
                                j++;
                        g0=(float)1.0e+30;
                        g1=(float)1.0e+30;
                        g2=(float)1.0e+30;

                        dist = distance(i,j);
                        if(j-1>=0)
                                g0 = (float)(G[i][j-1]+dist);
                        if(i-1>=0 && j-1>=0)
                                g1 = (float)(G[i-1][j-1]+2.0*dist);
                        if(i-1>=0)
                                g2 = (float)(G[i-1][j]+dist);

                        g3 = (g0<g1) ? g0:g1;
                        g = (g2<g3) ? g2:g3;
                        G[i][j] = g;
                }
        }

        DP_mdist = G[I][J]/(st_blk+in_blk);
        return(DP_mdist);
}

float distance(ab_t,ab_r)
int ab_t,ab_r;
{
        int i;
        float a,kyori;
```

```
        a=0.0;
        kyori=0.0;

        for(i=0;i<16;i++)
        {
                a=(in_CEP[ab_t][i]-st_CEP[ab_r][i]);
                kyori+=a*a;
        }
        return(kyori);
}
```