KONVERSI NADA-NADA AKUSTIK MENJADI CHORD MENGGUNAKAN PITCH CLASS PROFILE

Drs. Miftahul Huda, MT [1], Dwi Kurnia Basuki, S. Si. M. Kom[2]
Fandy Akbar[3] , Febrianzah Junaidy Permana[2]
[1]Dosen Jurusan Telekomunikasi, [2]Dosen Jurusan Teknik Informatika
[3]Mahasiswa Jurusan Teknik Informatika
Jurusan Teknik Informatika
Institut Teknologi Sepuluh Nopember
Kampus ITS Sukolilo, Surabaya 60111, Indonesia
Tel: +62 (31) 594 7280; Fax: +62 (31) 594 6114
E-mail : huda@eepis-its.edu, dwiki@eepis-its.edu
fundye@student.eepis-its.edu, junjp@yahoo.co.id
Homepage : http://www.eepis-its.edu

Abstraksi

Chord recognition atau pengenalan chord adalah sebuah transkripsi dari suara menjadi chord, dimana sebuah masukan yang berupa file audio akan diklasifikasikan sesuai dengan taraf-taraf ketentuan yang berbeda-beda. Chord merupakan beberapa nada yang dibunyikan sehingga menciptakan suatu suara yang harmonis. Chord yang akan dikenali dalam aplikasi ini adalah chord standar yaitu chord mayor dan chord minor. Proses kerja aplikasi diawali dengan proses sampling file audio berformat wave. Langkah selanjutnya adalah membagi data wave menjadi frame-frame (frame blocking). Dari frame-frame tersebut kemudian ditranformasikan dengan Fast Fourier Transform (FFT). Proses dilanjutkan dengan deteksi puncak untuk mengambil nilai-nilai puncak dari FFT. Nilai-nilai deteksi puncak tersebut diklasifikasi berdasarkan frekuensi nadanya melaui Pitch Class Profile (PCP). Dengan teknik PCP maka dapat dideteksi warna dari sebuah chord berdasarkan kelas pitch dari nada. Tahap akhir dari aplikasi adalah mencocokkan hasil PCP input dengan basis data (code book) dan menampilkan hasilnya dalam bentuk bar-bar chord.

Kata Kunci : *Chord, Fast Fourier Transform, Pitch Class Profile*

1. Pendahuluan

Fungsi untuk mengenali *chord* secara otomatis sangat penting dalam beberapa aplikasi, salah satunya adalah untuk belajar musik. *Chord* adalah gabungan beberapa nada yang dibunyikan sehingga menciptakan suara harmonis. Secara garis besar *chord* dikategorikan menjadi beberapa tipe, antara lain *: Chord mayor, minor, diminished, augmented, suspended, seventh, dll[1,2]*.

Pada penelitian ini dibangun sebuah aplikasi pengenalan *chord* secara otomatis yang dapat digunakan untuk membantu seorang pemain musik atau pemula dalam belajar musik atau mencari *chord-chord* dari suatu lagu. Proses pengenalan melibatkan sistem database yang berisi patern – patern dari sebuah chord dalam suatu jenis.

2. Tinjauan Pustaka

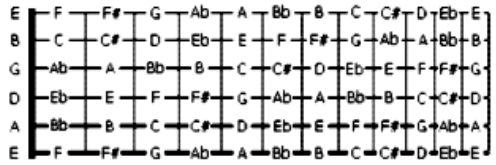
2.1. Tangga Nada

Tangga nada berisikan kumpulan nada-nada yang harmonis. Tangga nada kromatik merupakan kumpulan dari semua nada dalam musik. Karena nada selalu berulang untuk tiap oktaf yang ada, maka istilah 'tangga nada kromatik' sering dipakai untuk ke-12 nada dari tiap oktaf. Tabel berikut ini adalah frekuensi nada pada gitar mulai fret 0 / standart tunning sampai fret 13.

Tabel 1 *Frekuensi Nada Gitar*

Fret	Senar 6	Senar 5	Senar 4	Senar 3	Senar 2	Senar 1
0	2.41	110.00	146.83	196.00	246.94	329.63
1	87.31	116.54	155.56	207.65	261.63	349.2
2	92.50	123.47	164.81	220.00	277.18	369.99
3	98.00	130.81	174.61	233.08	293.66	392.00
4	103.83	138.59	185.00	246.94	311.13	415.30
5	110.00	146.83	196.00	261.63	329.63	440.0
6	116.54	155.56	207.65	277.18	349.23	466.16
7	123.47	164.81	220.00	293.66	369.99	493.88
8	130.81	174.61	233.08	311.13	392.00	523.25
9	138.59	185.00	246.94	329.63	415.30	554.37
10	146.83	196.00	261.63	349.23	440.00	587.33
11	155.56	207.65	277.18	369.99	466.16	622.25
12	164.81	220.00	293.66	392.00	493.88	659.26
13	174.61	233.08	311.13	415.30	523.25	698.46

Tangga nada kromatik untuk gitar dapat dilihat pada gambar berikut ini :



Gambar 1. *Tangga Nada Kromatik Pada Gitar*

2.2. Perekaman

Proses perekaman digunakan untuk mendapatkan database chord yang nantinya akan digunakan sebagai patern dalam proses *matching*. Proses perekaman terdiri atas 2 macam, yaitu :

- Analog : Proses perekaman secara analog lebih rumit karena menggunakan peralatan tape atau kaset plastik dengan kecepatan yang konstan untuk merekam dan memutar ulang.
- Digital : Proses perekaman digital secara mekanik jauh lebih sederhana, tetapi sangat banyak melibatkan elektronika. Sinyal masukan diperbanyak 1000 kali per detik dan setiap potongan akustik masing-masing diberikan angka digitalnya sendiri, yang berisikan nomor 0 dan 1.

Dengan perekaman digital, proses editing maupun pengolahan sinyal dapat lebih mudah dilakukan.

2.3. Pengolahan Sinyal Digital

Pengolalahan sinyal digital adalah pemrosesan sinyal yang mempunyai kaitan dengan penyajian, perubahan bentuk, dan manipulasi dari isi sinyal dan informasi dalam bentuk digital. Pemrosesan sinyal digital menawarkan pengendalian akurasi yang lebih baik. Dalam penelitian ini, pengolahan sinyal digital memiliki beberapa proses yaitu :

2.3.1. Sampling

Sinyal suara merupakan sinyal yang tidak terbatas dalam domain waktu (*infinite time interval*). Sinyal suara akan menghasilkan sinyal analog yang terus kontinyu. Untuk keperluan pemrosesan dalam transformasi fourier maka sinyal audio harus dibentuk dalam potongan-potongan waktu yang terbatas (*finite time interval*).

Karena itu sinyal yang ada dipotong-potong dalam slot-slot interval waktu tertentu. Berdasarkan pada teori sampling Nyquist, maka syarat dari frekuensi sampling adalah minimal dua kali frekuensi sinyal.

2.3.2. Front End Detection

Front-end Detection digunakan untuk menentukan batasan suatu sinyal, dalam hal ini letak sinyal awal dan akhir dari suatu frame sehingga bentuk sinyal asli tidak berubah. Hal ini dilakukan agar didapatkan sinyal suara tanpa noise karena pada proses pengambilan sampel sinyal suara sering terdapat noise yang mengakibatkan perubahan bentuk sinyal asli.[6]

2.3.3. Frame Blocking

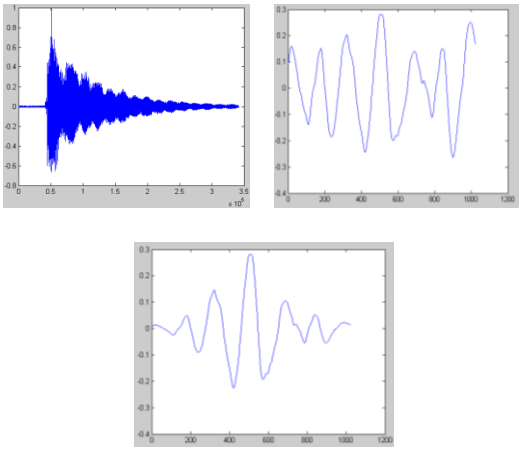
Frame Blocking adalah pembagian sinyal audio menjadi beberapa frame yang nantinya dapat memudahkan dalam perhitungan dan analisa sinyal, satu frame terdiri dari beberapa sampel tergantung tiap berapa detik suara akan disampel dan berapa besar frekuensi samplingnya. Pada proses ini dilakukan pemotongan sinyal dalam slot-slot tertentu agar memenuhi 2 syarat yaitu linear dan time invariant.

Dalam penelitian ini dilakukan pengambilan sample 65 millisecond yang nilainya 2866 data sample tiap frame dari frekuensi sampling 44100Hz.

2.3.4. Windowing

Window penghalus pada setiap frame digunakan untuk untuk mereduksi puncak pada setiap segmen (awal & akhir suatu frame), kita perlu untuk mengaplikasikan suatu window penghalus pada setiap frame. Untuk meningkatkan kualitas penghalusan ini, bisa dilakukan overlapping satu frame dengan yang lain, sehingga dapat membangkitkan suatu feature yang lebih halus sepanjang durasi waktu tersebut.

Jenis windowing ada beberapa macam yaitu Hamming, Hanning, Bartlet, Rectanguler dan Blackman. Dalam penelitian ini digunakan window hanning dengan memanfaatkan komponen DSP lab.



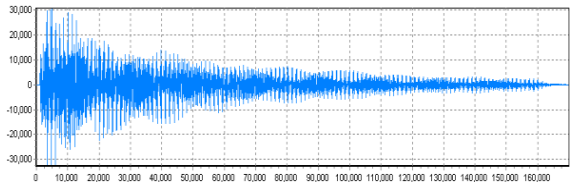
Gambar 2. *Perbedaan sinyal sebelum dan sesudah di hamming pada chord C Major.*

2.3.5. FFT

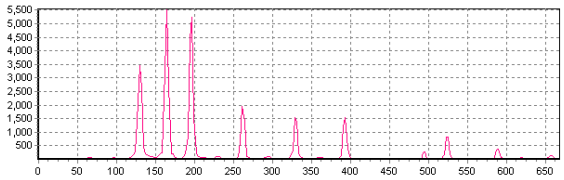
Fast Fourier Transform, adalah suatu algoritma untuk menghitung transformasi fourier diskrit dengan cepat dan efisien. Misalkan "x_0, ..., x_{N-1}" merupakan bilangan kompleks. Transformasi Fourier Diskret didefinisikan oleh rumus:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} nk} \qquad k = 0, \ldots, N-1.$$

Menghitung deret ini secara langsung memerlukan operasi aritmetika sebanyak O(N^2). Sebuah algoritma FFT hanya memerlukan operasi sebanyak O(N log N) untuk menghitung deret yang sama. Secara umum algoritma tersebut tergantung pada pemfaktoran N. Jadi bisa dikatakan bahwa proses FFT lebih cepat dari DFT.



Gambar 3. *Sinyal Chord C Major dalam domain waktu*



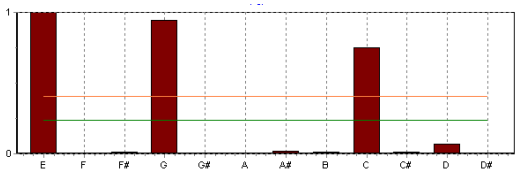
Gambar 4. *Sinyal Chord C Major dalam domain frekuensi*

2.3.6. Pitch Class Profile

Sebuah nada dalam musik mempunyai standard pitch, yang di identifikasi dengan sebuah nama dan satu oktaf (misal C4). Dengan teknik PCP maka dapat di deteksi warna dari sebuah *chord* berdasarkan kelas Pitch dari nada. PCP berawal dari sebuah penyajian frekuensi antara lain Transformasi Fourier, setelah di frekuensi di hitung melalui Transformasi Fourier maka frekuensi dipetakan ke dalam 12 pitch kelas(E, F, F#, G, G#, A, A#, B,C, C#, D, D#) [5]. Untuk menentukan pitch didefinisikan dengan persamaan berikut:

$$Pitch = \frac{12 \times \log\left(\frac{|Hz_{bin}|}{440}\right)}{\log(2)}$$

Keterangan:
Pitch =Pitch dari Nada-nada.
Hzbin = Frekuensi Nada yang didapat.



Gambar 5. *Hasil PCP dari FFT chord C Major*

2.3.7. Matching

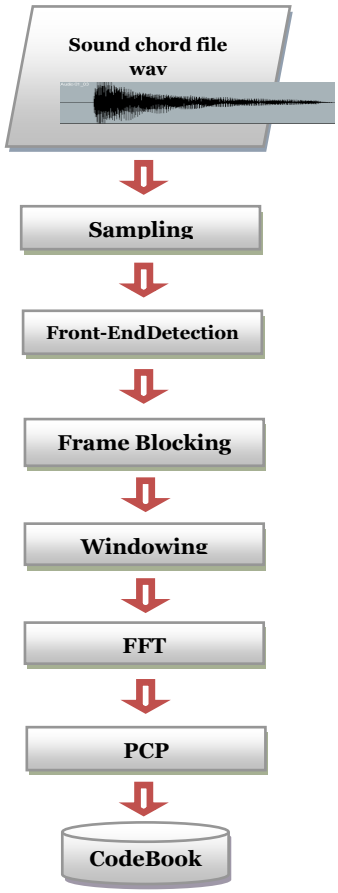
Dalam penelitian ini matching digunakan untuk menentukan hasil *chord*. Dari hasil PCP sebuah input yang kemudian dibandingkan dengan database yang ada, dimana diambil error yang paling terkeci, untuk kemudian hasil ditampilkan pada aplikasi. Metode untuk proses matching ini dikenal dengan sum square error (SSE) dimana nilai dari PCP dibandingkan dengan Database (Codebook) kemudian
dari semua error dibandingkan yang mana terkecil. Persamaan dari SSE adalah :

$$SSE\ (X) = \sum_{k=1}^{m} (X_k - c(X))^2$$

Keterangan :
SSE(X) = Jumlah error input ke X.
X_k = Input PCP nada Ke-k (dari Nada E sampai D#).
c(X) = Nilai PCP pada Codebook ke-X (*Chord* Ke-X).

3. Perancangan Sistem

Perancangan diperlukan agar dapat diperoleh kejelasan dalam mengembangkan aplikasi sehingga kerumitan ketika implementasi sistem dapat diminimalisasi. Berikut ini adalah blok diagram proses pembuatan sistem :



Gambar 6. *Blok diagram pembuatan sistem*

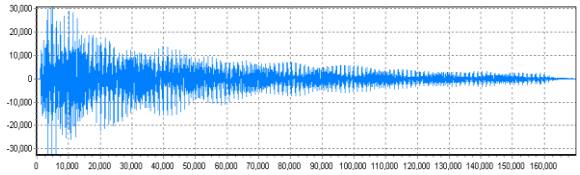
3.1. Perekaman Chord

Input dari sistem yang dibuat adalah berupa file audio berformat wav. File ini didapat dari perekaman dari instrumen gitar. Setiap chord yang akan dijadikan database direkam menggunakan instrumen gitar, dengan setiap chord direkam sebanyak 10 sample. Format perekaman ini disimpan dalam format wav standart atau kualitas CD, yaitu 44100Hz, 16 bit, strereo.

3.2. Sampling

Proses sampling ini adalah proses pembacaan dari input file audio wav. Pembacaan akan disimpan ke dalam array, sehingga file wav dapat diproses melalui penyimpanan ini. Langkah pertama yang harus dilakukan adalah pembuatan tipe data dasar untuk mengambil informasi yang ada pada header file

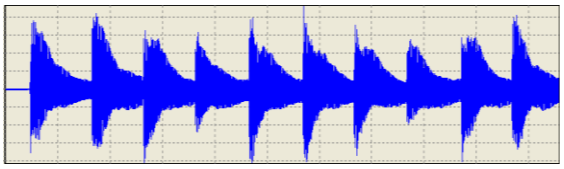
WAV. Setelah membuat tipe data dasar dari file wave maka dilakukan pembacaan file wave.



Gambar 7. *Data wave yang telah dibaca dan ditampilkan dalam Chart.*

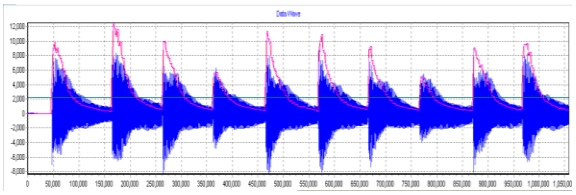
3.3. Front – End Detection

Pada penelitian ini, *front end detection* digunakan untuk mendeteksi ada tidaknya setiap petikan gitar pada sinyal secara keseluruhan. Karena setiap petikan gitar yang direkam akan menghasilkan nilai data wav yang besar dan nilainya akan turun secara komulatif selama tidak terdapat petikan lagi.



Gambar 8. *Sinyal petikan gitar*

Pada tahap ini juga dilakukan proses Frame Blocking sekaligus, karena proses perhitungan power sinyal dilakukan setiap frame. Berikut ini adalah gambar hasil front-end detection :



Gambar 9. *Hasil Front – End Detection*

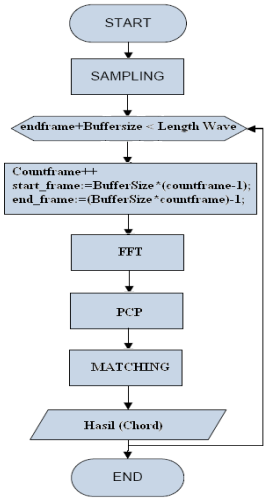
3.4. Frame Blocking

Pada proses ini dilakukan pemotongan sinyal dalam slot-slot tertentu agar memenuhi 2 syarat yaitu linear dan time invariant. Pengambilan sample setiap frame diambil dalam waktu milisecond (ms). Berikut ini rumus yang digunakan untuk pengambilan sample tiap frame :

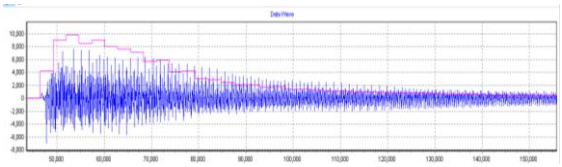
$$SFr = Fs * (t / 1000)$$

Keterangan :
SFr : Sample per Frame
Fs : Frekuensi sample / Sample Rate dari format audio wav
t : waktu pengambilan (ms)

Dibawah ini flowchart algoritma perhitungan Frame Blocking :



Gambar 10. *Flowchart FrameBlocking.*



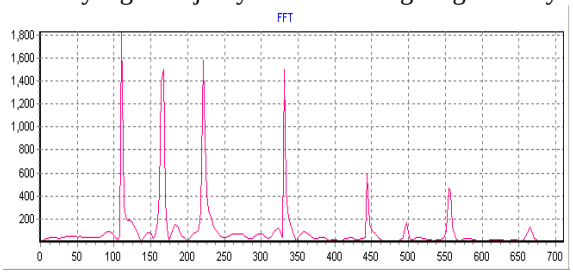
Gambar 11. *Hasil Frame Blocking*

3.5. Windowing

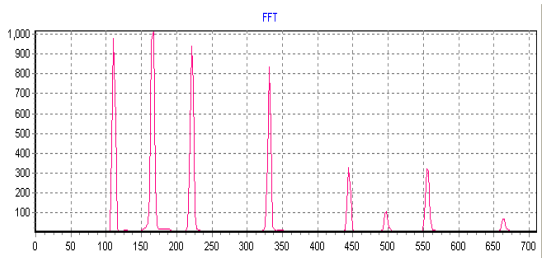
Proses windowing ini dijalankan pada proses FFT, karena fungsi windowing ini jadi satu dalam tool FFT pada komponen DSPlab. Jadi hanya memangil fungsi windowing yang disediakan oleh komponen tersebut.

3.6. FFT

Pada implementasi pembuatan sistem, nilai awal sinyal ini akan menjadi parameter pada fungsi FFT. Untuk prosesnya, mulai titik awal sinyal akan diambil sample sebanyak buffer size dari FFT yang digunakan untuk dimasukkan dalam variable array RealIn yang selanjutnya akan dihitung magnitudenya.



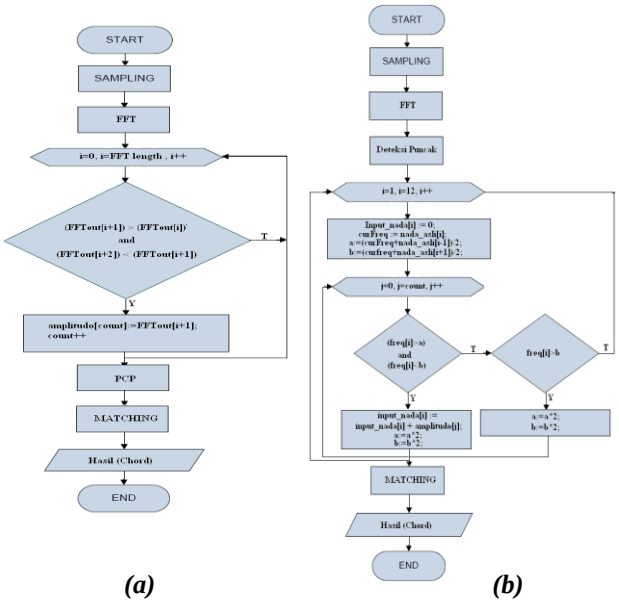
Gambar 12. *Hasil FFT tanpa Windowing*



Gambar 13. *Hasil FFT dengan Windowing*

3.7. PCP

Setiap nilai puncak frekuensi akan dikelompokkan menjadi 12 nada berdasarkan nilai satuan frekuensi nada. Dibawah ini flowchart pembuatan sistem deteksi puncak dan PCP :



(a) ***(b)***

Gambar 14. *(a). Flowchart Deteksi Puncak (b). Flowchart PCP*

Puncak	Frekuensi	Puncak	Frekuensi	Puncak	Frekuensi
1	91.52	12	301.46	23	497.96
2	110.36	13	309.54	24	508.72
3	126.51	14	331.07	25	524.87
4	166.88	15	344.53	26	554.48
5	183.03	16	379.52	27	573.32
6	201.87	17	387.60	28	578.70
7	220.72	18	398.36	29	605.62
8	250.32	19	419.90	30	621.77
9	255.71	20	444.12	31	629.85
10	274.55	21	460.27	32	646.00
11	285.31	22	476.42	33	662.15

Gambar 15. *Hasil Deteksi Puncak*

Setelah diperoleh nilai puncak frekuensi, maka nilai tersebut dikelompokkan berdasarkan satuan frekuensi nada. Dalam mengelompokkan ke dalam frekuensi nada, diperlukan perhitungan jarak awal dan akhir disetiap nada, sebagai range dari nada tersebut. Berikut ini rumus range nada yang dipakai :

Awal nada ke-i = (frek. nada ke-(i-1) + frek. Nada ke-i)/2
Akhir nada ke-i = (frek.nada ke-i + frek. nada ke-(i+1))/2

Contoh untuk nada A :
Awal = (frek. nada G# + frek. nada A)/2
= (103.83 + 110.00)/2
= 213.83/2
= 106.91
Akhir = (frek. nada A + frek. nada A#)/2
= (110.00 + 116.54)/2
= 113.27
Rumus diatas digunakan untuk semua nada. Setelah semua nada memiliki nilai range, maka setiap nilai puncak frekuensi akan dikelompokkan berdasarkan range – range nada.

3.8. Codebook

Codebook adalah nilai yang akan disimpan kedalam database. Nilai tersebut merupakan nilai hasil rata – rata PCP tiap sinyal petikan gitar yang terdeteksi (Front – End Detection).

Nada	Nilai PCP
E	0.63
F	0.00
F#	0.00
G	0.00
G#	0.00
A	1.00
A#	0.00
B	0.02
C	0.00
C#	0.06
D	0.00
D#	0.00

Gambar 16. *Tabel nilai rata – rata PCP*

3.9. Perancangan Database

Perancangan database dilakukan dengan membuat tabel untuk menyimpan nilai frekuensi hasil akhir pengolahan / pemprosesan file audio, yaitu nilai rata – rata PCP. Nilai frekuensi yang akan disimpan ke dalam database ini, akan digunakan pula dalam proses matching pada pembuatan aplikasi digital musik mentor.

Field	Type	Collation	Attributes	Null	Default	Extra
chord_id	int(255)			No		auto_increment
chord_name	varchar(10)	latin1_swedish_ci		No		
e_tone	float			No		
f_tone	float			No		
f_kres_tone	float			No		
g_tone	float			No		
g_kres_tone	float			No		
a_tone	float			No		
a_kres_tone	float			No		
b_tone	float			No		
c_tone	float			No		
c_kres_tone	float			No		
d_tone	float			No		
d_kres_tone	float			No		

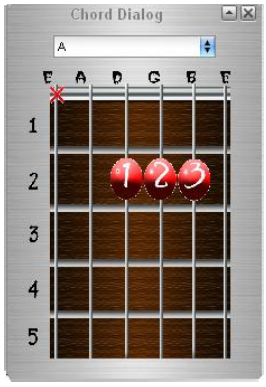
Gambar 19. *Stuktur tabel database*

4. Implementasi

Aplikasi ini memiliki beberapa antarmuka yang dikaitkan dengan proses yang dilakukan dalam pengenalan chord, diantaranya adalah :

Gambar 20. *Hasil Matching*

Pada gambar 20 hasil matching ditampilkan pada scrollbox, garis berwarna hitam menunjukan panjang dari file wave dan Label 'A' menunjukan *Chord* A Major.



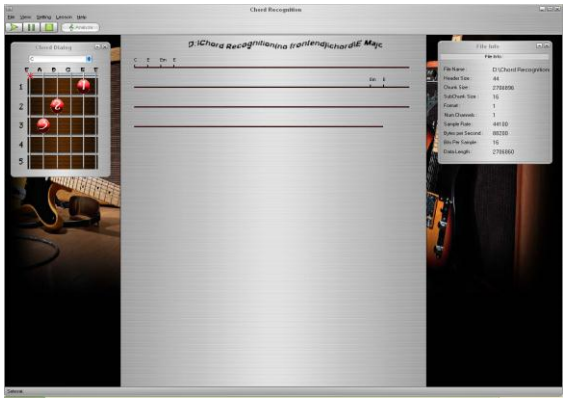
Gambar 21. *Hasil tampilan pada Chord Dialog.*

Pada gambar 21 hasil matching juga ditampilkan pada *chord* dialog yang fungsinya untuk menampilkan bentuk *chord* yang sedang dimainkan. Pada gambar diatas combo box pada *chord* dialog bertuliskan huruf A hal ini menunjukan bahwa *chord* yang dimainkan adalah *chord* A Major.

Pada pengujian kali ini dilakukan percobaan mendeteksi . file audio yang digunakan database dan beberapa file lagu. Percobaan dilakukan dengan membandingkan sistem yang menggunakan front-end detection dan yang tidak menggunakan front-end detection.

Percobaan I

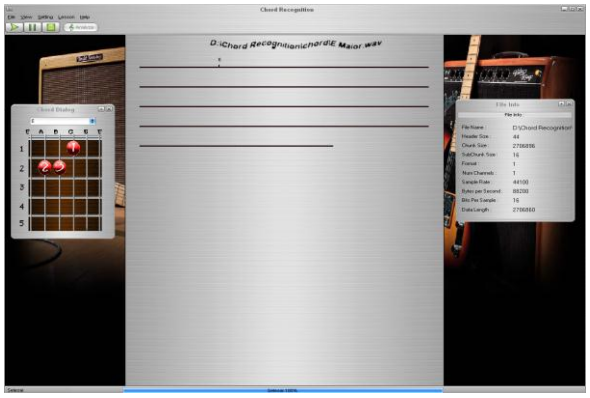
Pengujian berikut ini akan coba membandingkan sistem yang menggunakan front-end dan sistem yang tidak menggunakan front-end. Percobaan dimulai dari pengenalan *chord* dari file wave yang dijadikan codebook dimana terdapat 24 file wav, berikut ini percobaan dengan sistem tanpa menggunakan front-end detection.



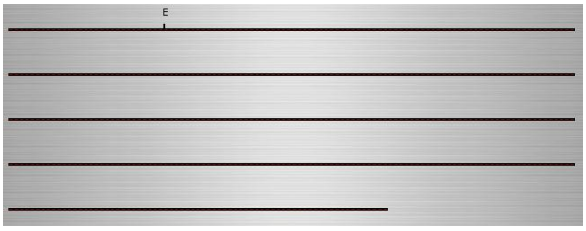
Gambar 22. *Hasil pengenalan Chord E Major tanpa Front-end Detection*

Percobaan II

Pengujian berikut ini menggunakan metode sebelumnya yaitu menggunakan front-end detection. Percobaan dilakukan dengan menggunakan file yang sama.



Gambar 23. *Hasil pengenalan Chord E Major Sistem Dengan front-end detection*



Gambar 24. *Hasil pengenalan Chord E Major Sistem Dengan front-end detection*

Pada gambar diatas hasil pengenanalan dilakukan dengan menggunakan metode front-end pada *Chord* E Major dari database 'E Major.wav' dimana *Chord* E Major dibunyikan 10 kali. Dari hasil informasi yang ditampilkan tidak terdapat *error* yang muncul.

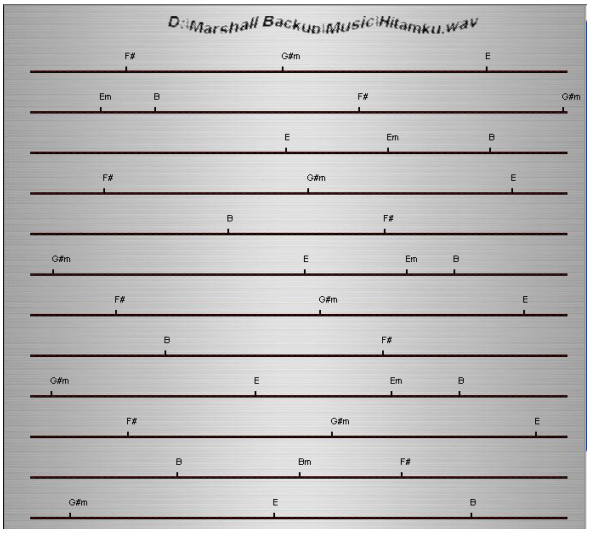
Tabel 2. *Hasil Pengenalan Chord dari File Wave Database dengan front-end detection*

No	**File Wave**	**Dikenali**	***Error***
1	E Major	10	0
2	E minor	10	0
3	F Major	10	1
4	F minor	10	0
5	F# Major	10	0
6	F# minor	10	1
7	G Major	10	0
8	G minor	10	1
9	G# Major	10	0
10	G# minor	10	1
11	A	10	0
12	A minor	10	1
13	A# Major	10	0
14	A# minor	10	0
15	B Major	10	0
16	B minor	10	0
17	C Major	10	0
18	C minor	10	1
19	C# Major	10	0
20	C# minor	10	0
21	D	10	0
22	D minor	10	0
23	D# Major	10	0
24	D# minor	10	0

Pada tabel 2 diatas percobaan dilakukan dengan metode menggunakan front-end detection hasi akurasi *chord* yang dikenali mencapai 95% hal ini dikarenakan file input wave yang digunakan menggunakan single instrument saja. Jadi dapat disimpulkan bahwa untuk kasus single instrument metode front-end detection dapat dikatakan berhasil.

Percobaan III

Pada percobaan Ketiga ini akan dilakukan ujicoba dengan input beberapa lagu yang telah di rekam. Dimana masih membandingkan metode dengan front-end detection dan tanpa front-end detection. Berikut ini percobaan dengan sistem yang menggunakan front-end Hasilnya ditampilkan pada gambar berikut.



Gambar 25. *Hasil analisa Lagu Andra-Hitamku sistem dengan Front End detection.*

Gambar diatas adalah hasil analisa program dari lagu Andra&the Backbone–Hitamku single instrument dengan sistem yang menggunakan front-end detection. Dari hasil yang di tampilkan terdapat beberapa *error* yaitu muncul *chord* E minor dan B minor yang seharusnya E Major dan B Major hal ini dikarenakan *chord-chord* perpindahan masih belum dibuat codebooknya. Untuk lagu ini akurasi yang didapat kan adalah 80%.

Tabel 3. *Tabel Hasil Perhitungan Error dengan metode front-end.*

No	***Chord* Benar**	***Chord* Hasil program**	***Error***
1	F# Major	F# Major	0
2	G# minor	G# minor	0
3	E Major	E Major,E Minor,	0.5
4	B Major	B Major, B Minor	0.5

5. Kesimpulan

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

1. Aplikasi ini mengasilkan informasi – informasi *chord* dari input sebuah lagu dimana informasi yang disajikan lebih cepat daripada pencarian secara manual oleh pemain musik.
2. Dengan adanya Quiz maka sangat membantu kemampuan para pemain musik dalam mengenali *chord.*
3. Tingkat keberhasilan sistem yang menggunakan metode front-end detection untuk kasus lagu single instrument akurasi mencapai 80% - 95%, sedangkan untuk lagu lebih dari satu instrument metode front-end detection tidak cocok dikarenakan banyak chord yang tidak terdeteksi.

4. Untuk sistem yang tidak menggunakan metode front-end detection tingkat keberhasilan dalam lagu single instrument mencapai 60% - 80% dan untuk lagu lebih dari satu instrument metode ini dapat mengenali *chord* dengan akurasi yang hampir sama.

DAFTAR PUSTAKA

[1],Artikel tentang "Pengenalan Chord", http://ratdix.wordpress.com, 2008.

[2],Artikel tentang"Akord", http://id.wikipedia.org/wiki/Akord..

[3],Chuan Ching-Hua, Kevin Zhu, "Transforming Audio into Guitar Tab Scores", The University of Southern California, 2004.

[4],Zhang Xinglin, David Gerhard, "Chord Recognition Using Instrument Voicing Constraint"Dept. of Computer Science, Dept. of Music Univeristy of Regina, 2008

[5],Artikel tentang"Tentang Matlab", http://theguh05.blogspot.com/, 2008.

[6],Sumi Kouhei, Katsutoshi Itoyama, Kazuyoshi Yoshii, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno, "Automatic Chord Recognition Based On Probabilistic Integration Of Chord Transition And Bass Pitch Estimation", Dept. of Intelligence Science and Technology National Institute of Advanced Industrial Graduate School of Informatics, Kyoto University Science and Technology (AIST) Japan,2008

[7],Artikel dan Tutorial tentang "Front-End Detection", http://agusslamet.wordpress.com/2008/09/front-end-detection/, 2008.

[8] Musik Tutorial (part-1) – Tangga Nada, http://Silentman13.wordprees.com

[9] Cabral Giordano, Pachet François, Briot Pierre Jean *," Recognizing Chords with EDS: Part One"*, Laboratoire d' informatique de Paris 6,France, 2005.

[10] Sumi Kouhei, Katsutoshi Itoyama, Kazuyoshi Yoshii, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno, *"Automatic Chord Recognition Based On Probabilistic Integration Of Chord Transition And Bass Pitch Estimation"*, Dept. of Intelligence Science and Technology National Institute of Advanced Industrial Graduate School of Informatics, Kyoto University Science and Technology (AIST) Japan,2008

[11] Artikel tentang "Code Gear Delphi", http://id.wikipedia.org/wiki/CodeGear_delphi.

[12] Artikel tentang "Komponen DSPLab", http://www.teworks.com

[13] Artikel tentang "Delphi MySql", http://lina84.wordpress.com/2008/04/06/koneksi-database-delphi-mysql/.

[14] Artikel tentang "Wave", http://en.wikipedia.org/wiki/wav.

[15] Artikel tentang "Load & Display WAVE File (.WAV)", http://pebbie.wordpress.com/2007/08/08/load-display-wave-file-wav/, 2007.

[16] Artikel tentang "Kelebihan Pemrosesan Sinyal Digital" ,http://agfi.staff.ugm.ac.id/blog/index.php/2008/11/kelebihan-pemrosesan-sinyal-digital/.

[17] Solichin Achmad, "Sekilas Tentang MySQL", Fak. Teknologi Informasi, Univ. Budi Luhur.

[18] Artikel "Teori Recording" http://melly-kustina.blogspot.com/2009/03/teori-recording.html.