Jurnal CoreIT, Vol.1, No.2, Desember 2015
ISSN: 2460-738X (Cetak)

Aplikasi Pencarian *Chord* dalam Membantu Penciptaan Lagu Menggunakan Algoritma *Fast Fourier Transform (FFT)* dan Metode Klasifikasi *K-Nearest Neighbor (KNN)*

Niky Fetra[1], Muhammad Irsyad[2]
[1,2]Teknik Informatika UIN Sultan Syarif Kasim Riau
Jl. H.R. Soebrantas no. 155 KM. 18 Simpang Baru, Pekanbaru 28293
nikyfetra@gmail.com[1], irsyadtech@uin-suska.ac.id[2]

***Abstrak* – Musik merupakan salah satu hiburan yang menjadi kesukaan mayoritas manusia. Musik yang lengkap terdiri dari suara vokal dan musik pengiring yang berasal dari suara alat musik. Sebagian alat musik dimainkan menggunakan *chord*. Dalam sebuah musik, suara vokal dan *chord* harus selaras. Keselarasan ditemukan saat nada dasar suara vokal sama dengan nada *chord* walaupun berada pada oktaf yang berbeda. Namun, tidak semua orang mampu menemukan keselarasan antara irama suara vokal dan *chord*. Untuk mengatasi masalah ini, dibangun sebuah aplikasi komputer yang mampu melakukan pencarian *chord* dalam membantu penciptaan lagu. Suara masukan merupakan suara vokal tanpa musik yang dapat dimasukan dengan cara direkam pada aplikasi atau dipilih dari computer berupa file dengan ekstensi **.WAV*. Proses yang dilakukan oleh aplikasi adalah pemotongan suara vokal berdasarkan birama dan tempo pilihan pengguna dan mencari *chord* yang selaras dengan suara vokal masukan. Dalam menemukan keselarasan, aplikasi akan mencari frekuensi suara masukan menggunakan algoritma *fast fourier transform (FFT)* dan melakukan klasifikasi *chord* menggunakan algoritma *K-Nearest Neighbor (K-NN)*. Hasil pengujian proses pencarian *chord* saran dari aplikasi mencapai akurasi terbesar 82,35% pada pengujian suara vokal tanpa musik dan 22,22% pada pengujian suara yang diiringi musik.**

Kata kunci – Chord, Fast Fourier Transform (FFT), Frekuensi, K-Nearest Neighbor (K- NN), Musik, Penciptaan Lagu, Oktaf, Vokal

PENDAHULUAN

A. Latar Belakang

Musik adalah penghayatan isi hati manusia yang diungkapkan dalam bentuk bunyi yang teratur dalam melodi atau ritme serta mempunyai unsur atau keselarasan yang indah[1]. Musik tidak hanya bermain dengan alunan nada dan terfokus pada satu sumber suara. Musik yang indah adalah kolaborasi antara beberapa sumber suara.

Sebagian besar orang- orang yang memiliki hobi di bidang seni musik memiliki keinginan untuk menciptakan lagu sendiri. Namun, tidak semua orang yang memiliki hobi di bidang seni musik mampu menghasilkan keselarasan antara irama suara dan irama dari alat *musik*. Sebagian besar alat musik menggunakan kunci yang biasa disebut *chord*. Dalam menciptakan lagu, *chord* akan membantu untuk menyelaraskan tinggi nada alat musik dan tinggi irama suara. Untuk menyelaraskan tinggi nada alat musik dan vokal, dapat dilakukan pembandingan antara frekuensi signal suara vokal dan frekuensi signal suara *chord*.

Dalam pengolahan frekuensi signal suara, dapat dgunakan algoritma *Fast Fourier Transform (FFT)*. Signal suara yang ditransformasikan menggunakan algoritma *FFT* akan menghasilkan signal dalam domain frekuensi. Hasil ekstraksi ini dapat digunakan dalam penentuan *chord*. Banyak algoritma yang dapat digunakan dalam penentuan *chord* ini. Salah satunya adalah algoritma *K-Nearest Neighbor (KNN)*. Pada sistem, pembentuk kelas menggunakan metode KNN, semua data testing dapat dikembalikan dengan benar, baik untuk data testing yang sama persis dengan data trainingnya maupun data testing yang berbeda dengan data testingnya[2].

Berbagai penelitian pencarian *chord* menggunakan algoritma FFT telah banyak dilakukan sebelumnya seperti Aplikasi Pengenalan dan Pembelajaran *Chord* Gitar Berbasis Web[3] dan aplikasi Pengkonversi Nada - Nada Instrumen Tunggal Menjadi *Chord* Menggunakan Metode Pitch Class Profile[4]. Pada penelitian-penelitian sebelumnya, belum ada ditemukan aplikasi yang mampu mengenali *chord* dari masukan suara vokal yang dapat digunakan dalam membantu penciptaan lagu. Oleh karena itu, penelitian yang akan dilakukan ini bertujuan untuk menghasilkan aplikasi pencarian *chord* dari masukan suara vokal.

B. Rumusan Masalah

Adapun rumusan masalah pada penelitian ini adalah sebagai berikut :

1. Bagaimana membangun aplikasi yang dapat membantu dalam menciptakan lagu?
2. Bagaimana cara membangun aplikasi yang dapat menghasilkan keluaran *chord* berdasarkan masukan suara vokal?
3. Bagaimana mengimplementasikan algoritam *Fast Fourier Transform* dan metode klasifikasi *K-Nearest Neighbor (KNN)* dalam penentuan *chord*?

C. Batasan Masalah

Batasan masalah pada penelitian ini bertujuan agar pembahasan tidak meluas dan dapat terfokus pada bidang kajiannya, maka batasan masalahnya sebagai berikut:

1. Aplikasi yang akan dibangun hanya memproses masukan birama 4/4 dan 2/4.
2. Aplikasi yang dibangun hanya menghasilkan *chord mayor* dan *chord minor*.

D. Tujuan Penelitian

Tujuan dari penelitian ini adalah membangun aplikasi pencarian *chord* berdasarkan suara vokal yang dapat membantu dalam penciptaan lagu dengan menggunakan algoritma *Fast Fourier Transform* dalam transformasi signal suara dan metode klasifikasi *K-Nearest Neighbor (KNN)* dalam penentuan *chord.*

LANDASAN TEORI

A. Audio/Suara

Audio (suara) merupakan fenomena fisik yang dihasilkan oleh getaran suatu benda yang berupa sinyal analog dengan amplitudo yang berubah secara kontinyu terhadap waktu yang disebut frekuensi[5]. Audio berhubungan dengan sinyal. Sinyal adalah fenomena dari lingkungan yang terukur atau terkuantisasi, sementara sistem merupakan bagian dari lingkungan yang menghubungkan sinyal dengan sinyal lainnya atau dengan kata lain merespon sinyal masuk dengan menghasilkan sinyal keluaran[6].

Audio memiliki frekuensi yaitu jumlah getaran yang terjadi dalam satu detik. Frekuensi dilambangkan dengan *f* dan memiliki satuan *Hertz (Hz)* atau bisa juga dengan satuan *Cycle Per Second (cps)*. Pada pengolahan audio digital, terdapat *Bit depth* dan *sampling rate* sebagai penentu kualitas sebuah suara digital. Semakin tinggi nilai *bit depth* dan *sampling rate*, maka semakin baik kualitas suara yang dihasilkan. *Bit depth* adalah nilai resulusi suara atau jumlah tingkatan level suara. *Sampel rate* merupakan nilai signal audio yang diambil dalam satu detik. Setiap file memiliki format tersendiri. Untuk file audio, ada beberapa format yang biasa digunakan seperti *Mp3, AAC, AC3, AMR, AU,WAV*, dan lain lain. Dari berbagai format ini, format *WAV* adalah format yang paling sering digunakan dalam pengolahan audio

B. Musik

Musik merupakan salah satu media penyampaian pesan sekaligus media untuk hiburan. Musik yang indah dapat memberikan ketenangan bagi pendengarnya. Dalam permainan musik, terdapat tempo dan birama yang harus diperhatikan. Tempo merupakan kecepatan sebuah lagu dengan perubahan kecepatannya dalam musik. Tempo atau kecepatan dalam dalam lagu dapat diukur menggunakan *Metronome.* Tempo didefinisikan dalam satuan MM. Sebagai contoh adalah tempo 100 MM yang berarti kecepatan metronum maelzell dalam 1 menit sama dengan 100 titinada seperempat.

Tempo dapat dikelompokkan menjadi tempo lambat, tempo sedang, dan tempo cepat. Berikut akan dijelaskan pengelompokan tempo.

1. Tempo Lambat
   Beberapa istilah tanda tempo lambat :
   a. *Grave* (MM = 40): lambat sekali dan Khidmad.
   b. *Larghissimo* (MM = 44): lebih lambat dari *largho.*
   c. *Largho* (MM = 46): lambat sekali dan lebar.
   d. *Lento* (MM = 52): lambat dengan rasa derita.
   e. *Adagio* (MM = 56): lambat dengan kesungguhan.
   f. *Larghetto* (MM = 60): lebih cepat dari *largho*.
   g. *Adagietto* (MM = 66): lebih cepat dari *adagio.*
2. Tempo Sedang
   Beberapa istilah tanda tempo sedang :
   a. *Andante* (MM = 72): seperti orang berjalan biasa.
   b. *Andantino* (MM = 80): seperti orang berjalan agak cepat.
   c. *Maestoso* (MM = 88): khidmad dan agung.
   d. *Moderato* (MM = 96): sedang.
3. Tempo Cepat
   Beberapa istilah tanda tempo cepat.
   a. *Allegretto* (MM = 108): lebih lambat dari allegro.
   b. *Animato* (MM = 120): riang gembira.
   c. *Marcia* (MM = 126): seperti orang berbaris.

d. *Allegro* (MM = 132): cepat dan hidup.
e. *Assai* (MM = 144): lebih cepat.
f. *Vivaee* (MM = 160): cepat dan garang.
g. *Presto* (MM = 184): sangat cepat.
h. *Prestissimo* (MM = 208): lebih cepat dari presto.

Birama merupakan ketukan-ketukan yang datangnya berulang dan teratur dalam waktu yang sama pada sebuah lagu. Birama biasa ditandakan sengan angka pecahan seperti 4/4, 2/4 dan lain-lain. Pembilang menunjukkan jumlah bagian ketukan (jumlah pukulan) dalam tiap-tiap birama. Sedangkan penyebut menunjukkan nilai nada dalam setiap ketukan.

C. Chord

Chord adalah gabungan dari beberapa nada yang menghasilkan perpaduan nada sehingga dapat digunakan sebagai pengiring lagu. Sebagian besar alat musik modern menggunakan *chord* untuk mengiringi lagu. Ada berbagai jenis *chord*, diantaranya adalah *chord mayor dan chord minor* yang merupakan *chord* dasar dalam permainan musik. Sedangkan jenis *chord* lain seperti *chord diminished*, *chord augmented*, *chord minor 6*, *chord mayor 7*, dan *chord suspended* digunakan untuk memperindah atau mengubah kualitas sebuah lagu[7].

Chord tidak lepas dari tangga nada. Tangga nada merupakan kumpulan dari nada nada yang harmonis dengan aturan tertentu yang mendasarinya

Tangga nada kromatik adalah kumpulan dari semua nada musik. Adapun susunan tangga nada dalam satu otaf adalah C1 - C#1 - D1 - D#1 - E1 - F1 - F#1 - G1 - G#1 - A1 - A#1 - B1 - C2 dan seterusnya.

Nilai frekuensi nada dalam oktaf tertentu adalah dua kali nilai frekuensi pada oktaf sebelumnya. Sebagai contoh nada A1=55, nada A2=110, nada A3=220, dan seterusnya.

D. *Fast Fourier Transform* (FFT)

Algoritma *Fast Fourier Transform* (FFT) adalah suatu algoritma yang digunakan untuk mentransformasikan sinyal suara menjadi sinyal frekuensi, artinya proses perekaman suara disimpan dalam bentuk digital berupa gelombang spectrum suara berbasis frekuensi[8]. *FFT* dapat digunakan untuk menghitung *Descret Fourier Transform (DFT)* dengan cepat. *DFT* dapat didefenisikan sebagai berikut:

$$f(x) = \sum_{n=0}^{N-1} x(n) e^{-i\left(\frac{2\pi}{N}\right)mn} \quad (1)$$

Dimana :
n = indeks dalam domain waktu (0, 1, ., N-1)
m = indeks dalam domain frekuensi (0, 1, ..., N-1)
Berdasarkan persamaan euler yang menyatakan :

$$e^{\pm ix} = cos(x) \pm i\, sin(x) \quad (2)$$

Maka persamaan 1 dapat ditulis sebagai berikut :

$$f(x) = \sum_{n=0}^{N-1} \left[x(n) \cos\left(\frac{2\pi mn}{N}\right) - i\, x(n) \sin\left(\frac{2\pi mn}{N}\right) \right] \quad (3)$$

Dalam hal ini, i merupakan imaginer. Hal ini menyebabkan hasil dari persamaan diatas terdiri dari unsure real dan imaginer. Untuk mendapatkan *Spectrum Fourier* (|f(x)|) dapat diselesaikan dengan persamaan berikut:

$$|f(x)| = \sqrt{Real^2(x) + Im^2(x)} \quad (4)$$

E. K-Nearest Neighbor (KNN)

Algoritma *K-Nearest Neighbor* (KNN) adalah salah satu algoritma pengklasifikasian data. Klasifikasi merupakan suatu metode untuk mengelompokkan sebuah objek ke dalam kelompok atau kelas tertentu[2]. Berikut adalah langkah-langkah dalam metode klasifikasi K-NN:

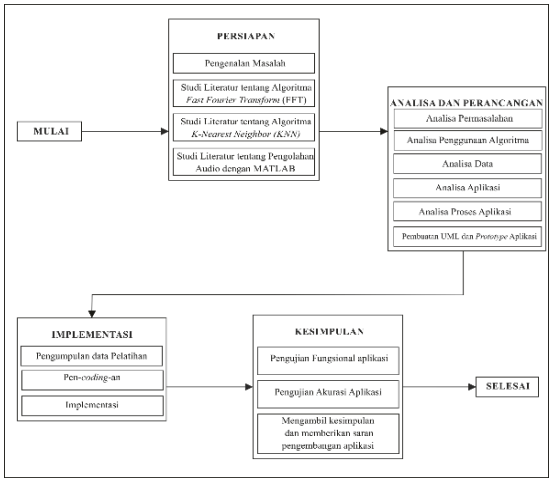
1. Menghitung jarak antara data baru ke setiap data pelatihan dengan persamaan berikut:

$$di = \sqrt{\sum_{i=1}^{p} (x_{2i} - x_{1i})^2} \quad (5)$$

Dimana:
x_1 = sampel data ;
p = dimensi data;
x_2 = data uji;
i = variable data.
d = jarak;

2. Menentukan k labeled data yang mempunyai jarak paling minimal
3. Klasifikasi data baru ke dalam labeled data yang mayoritas

METODOLOGI PENELITIAN



Gambar 1. Tahapan penelitian

ANALISA DAN PERANCANGAN

A. Analisis Data

Data Pelatihan

Data pelatihan diperoleh dengan merekam suara *strumming* gitar. Suara yang menjadi data pelatihan adalah suara *chord* A, A#, Am, A#m, B, Bm, C, C#, Cm, C#m, D,D#, Dm, D#m, E, Em, F, F#, Fm, F#m, G, G#, Gm, dan G#m. Suara *chord* direkam pada birama 2/4 dan 4/4 dengan tempo 40 MM, 44 MM, 46 MM, 52 MM, 56 MM, 60 MM, 66 MM, 72 MM, 80 MM, 88 MM, 96 MM, 108 MM, 120 MM, 126 MM, 132 MM, 144 MM, 160 MM, 184 MM, dan 208 MM. Jadi, total suara yang direkam sebagai data latih adalah 24 x 2 x 19 = 912 data suara.

Semua nada akan ditentukan frekuensi dasarnya dengan melakukan transformasi suara ke dalam domain frekuensi menggunakan algoritma *FFT*. Untuk menentukan nada dasar, maka disimpan tiga frekuensi dengan magnetudo yang tertinggi.

Data pelatihan ditambah dengan kelipatan frekuensi data yang telah direkam. Frekuensi yang ditambah adalah:

a. 2 kali frekuensi suara rekaman,
b. 1/2 kali frekuensi suara rekaman, dan
c. 1/4 kali frekuensi suara rekaman

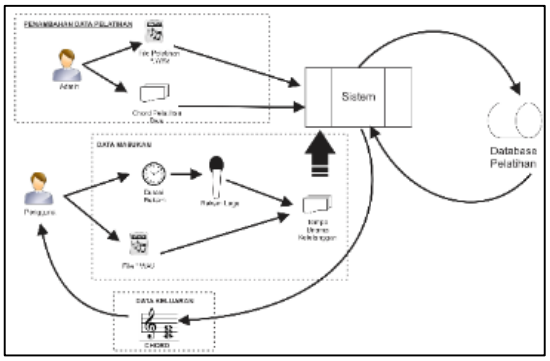
Jumlah data yang ditemukan adalah sebanyak 3648 data dengan 24 macam chord sebagai kelas pada data pelatihan. Frekuensi tertinggi dari semua data adalah 4400 hz dan frekuensi terendah dari semua data adalah 54 hz. Rentang yang didapatkan sudah memenuhi rentang suara yang mampu dicapai manusia.

Data Uji

Data uji merupakan data yang akan diuji pada sistem. Pada aplikasi pencarian chord data uji yang digunakan adalah file suara dengan ekstensi *.WAV atau file suara yang direkam langsung pada aplikasi. Suara yang dimasukkan akan dilakukan pencarian frekuensi tertinggi dengan algoritma FFT.

B. Analisis Aplikasi

Gambaran umum dari apliaksi yang dibangun dapat dilihat pada Gambar 2:



Gambar 2. Gambaran Umum Aplikasi

C. Pengolahan Data Pelatihan

Data Masukan

Data masukan dalam penambahan data pelatihan adalah file suara *chord* baru dan *chord* baru.

a. File suara *chord*
   File suara *chord* adalah file baru yang akan menjadi data pelatihan. File suara ini dapat dipilih admin dari komputer dengan ekstensi *.WAV.
b. Chord
   Chord adalah pilihan kelas baru yang dipilih admin dari file suara yang telah ditambahkan.

Proses

Berikut akan dijelaskan proses yang terjadi pada penambahan data pelatihan:

a. Pencarian frekuensi dengan magnetudo tertinggi, dilakukan menggunakan algoritma *Fast Fourier Transform (FFT)*.
b. Pencarian frekuensi pelatihan, meliputi:
   1) Frekuensi suara pelatihan,
   2) 2 x frekuensi suara pelatihan,
   3) 1/2 kali frekuensi suara pelatihan,
   4) 1/4 kali frekuensi suara pelatihan

Data keluaran

Data keluaran dari proses penambahan data pelatihan adalah frekuensi suara pelatihan baru dengan *chord* sebagai kelas data. Data keluaran ini akan disimpan ke dalam database pelatihan.

D. Pencarian *Chord*

Data Masukan

Terdapat empat data masukan yang akan dipilih pengguna dalam menjalankan aplikasi yaitu:

a. Suara vokal lagu.
   Data suara vokal dapat dimasukkan dengan merekam secara langsung pada aplikasi atau memilih file dengan ekstensi **.WAV* dari komputer.

b. Birama.
   Pilihan birama yang dapat digunakan yaitu birama 2/4 dan 4/4.
c. Tempo
   Tempo yang dapat dipilih pengguna terdapat 19 jenis yaitu 40 MM (lambat), 44 MM (lambat), 46 MM (lambat), 52 MM (lambat), 56 MM (lambat), 60 MM (lambat), 66 MM (lambat), 72 MM (sedang), 80 MM (sedang), 88 MM (sedang), 96 MM (sedang), 108 MM (cepat), 120 MM (cepat), 126 MM (cepat), 132 MM (cepat), 144 MM (cepat), 160 MM (cepat), 184 MM (cepat), dan 208 MM (cepat).
d. Ketetanggaan
   Pengguna dapat memasukkan nilai ketetanggaan dengan bilangan bulat dari 1-10.

Proses

Proses yang dilakikan pada aplikasi pecarian *chord* dalam membantu penciptaan lagu adalah sebagai berikut :

a. Penghitungan durasi dalam satu *bar*. Durasi dimaksudkan untuk mencari lamanya sebuah *chord* dimainkan sebelum pindah ke *chord* berikutnya dalam sebuah lagu. Pencarian durasi ini dilakukan dengan menghitung keterkaitan antara tempo dan birama. Sebagai contoh, durasi satu *bar* pada tempo 40 MM dan birama 2/4 dapat dihitung dengan :

$$durasi = \frac{60\ detik\ .2}{40} = 3\ detik$$

b. Pemotongan lagu, proses ini akan memotong suara vokal yang telah dimasukkan sehingga menghasilkan *n* buah potongan lagu. Pada langkah selanjutnya akan dilakukan pengulangan sebanyak *n* kali.
c. Pencarian frekuensi, Proses ini dilakukan menggunakan algoritma *FFT*.
d. Pencarian *chord*, proses pencarian *chord* dilakukan menggunakan algoritma *K-NN* berdasarkan data pelatihan yang telah disimpan pada sistem.

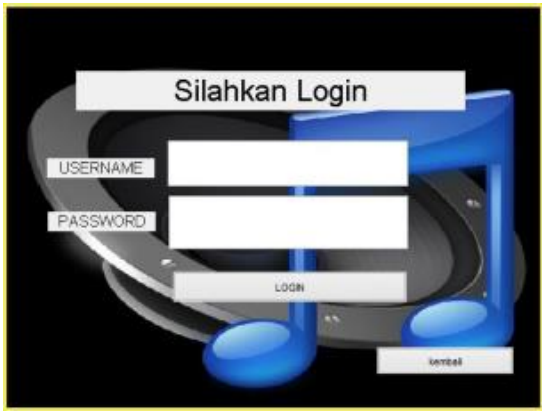
Data Keluaran

Setelah semua proses selesai dilakukan, maka akan ditemukan hasil akhir dari aplikasi. Hasil akhir dari aplikasi adalah chord saran dari aplikasi yang dapat digunakan pengguna untuk mengiringi lagu yang telah dimasukkan sebelumnya.

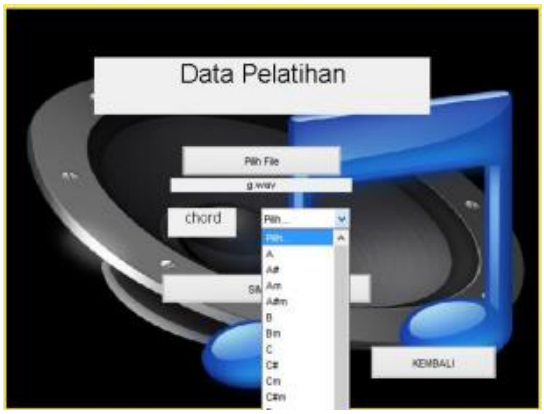
IMPLEMENTASI DAN PENGUJIAN

Hasil implementasi dari aplikasi yang dikembangkan dapat dilihat pada Gambar 3 sampai dengan Gambar 7..

Gambar 3 Tampilan Halaman Utama Aplikasi

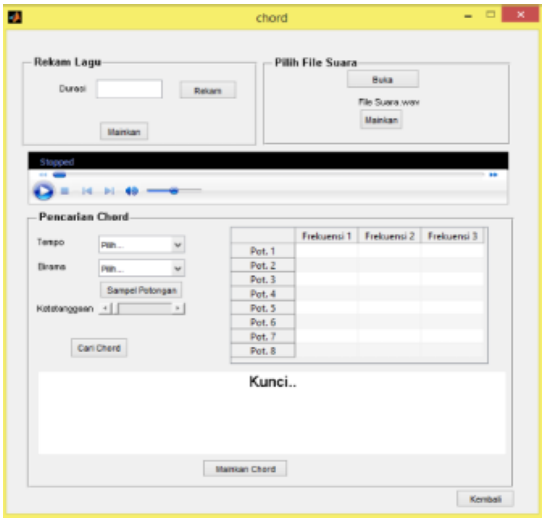


Gambar 4 Tampilan Halaman Login



Gambar 5 Tampilan Halaman Tambah Data Pelatihan

Gambar 6 Tampilan Halaman Tutorial



Gambar 7 Tampilan Halaman Pencarian *Chord*

Pengujian Fungsional Aplikasi

Pengujian fungsional aplikasi meliputi pengujian algoritma dan pengujian dengan metode *Black Box.*

Pengujian algoritma dilakukan dengan membandingkan hasil perhitungan manual dan hasil perhitungan dengan aplikasi. Dari hasil pengujian ini, dapat diketahui perhitungan manual dan perhitungan pada aplikasi menghasilkan keluaran yang sama. Pada pengujian blackbox semua fitur aplikasi dapat berjalan dengan baik sesuai dengan yang diharapkan.

Pengujian Akurasi Aplikasi

Pada pengujian ini, dilakukan dua kali pencarian *chord* yaitu pencarian *hord* lagu tanpa musik dan *chord* lagu dengan musik.

1. Lagu tanpa iringan musik

   Pencarian yang pertama adalah pencarian *chord* dari sebuah lagu ciptaan Krisna Balagita. Berikut adalah hasil pengujian akurasi pada lagu pertama :

Tabel 1. Hasil pengujian akurasi lagu pertama

Pengujian ke	***Chord* Benar**	***Chord* Salah**	**Akurasi**
1	35	16	68.63 %
2	35	16	68.63 %
3	35	16	68.63 %
4	36	15	70.59 %
5	39	12	76.47 %
6	41	10	80.39 %
7	42	9	82.35 %
8	41	10	80.39 %
9	36	15	70.59 %
10	36	15	70.59 %

2. Lagu dengan iringan musik

   Lagu yang digunakan pada pengujian ini adalah lagu yang berjudul "Untuk Kita Renungkan" ciptaan Ebiet G Ade. Berikut adalah hasil pengujian akurasi pada lagu kedua:

Tabel 2. Hasil pengujian akurasi lagu kedua

Pengujian ke	***Chord* Benar**	***Chord* Salah**	**Akurasi**
1	9	54	14.28 %
2	9	54	14.28 %
3	8	55	12.70 %
4	8	55	12.70 %
5	5	58	7.94 %
6	8	55	12.70 %
7	14	49	22.22 %
8	11	52	17.46 %
9	12	51	19.04 %
10	11	52	17.46 %

KESIMPULAN DAN SARAN

Setelah dilakukan tahapan dalam merancang dan membangun aplikasi pencarian *chord* dalam membantu penciptaan lagu, dapat diambil kesimpulan sebagai berikut :

1. Aplikasi pencarian *chord* dalam membantu penciptaan lagu dapat memberikan saran *chord* dari masukan suara vokal yang dapat dipilih dari komputer dengan ekstensi **.WAV* atau suara vokal yang di rekam langsung oleh aplikasi.
2. Tahapan pencarian *chord* meliputi pemotongan signal suara berdasarkan tempo serta birama pilihan pengguna, penecarian frekuensi setiap potongan signal suara dengan algoritma *Fast Fourier Transform (FFT),* dan pencarian *chord* masing masing potongan menggunakan algoritma *K-Nearest Neighbor (K-NN).*
3. Hasil pencarian *chord* suara vokal tanpa musik

menghasilkan akurasi yang lebih baik dibandingkan dengan pencarian *chord* lagu dengan musik dimana tingkat akurasi aplikasi dalam mencari *chord* sebuah lagu tanpa musik mencapai 82,35% menggunakan 7 ketetanggaan terdekat dan tingkat pencarian *chord* sebuah lagu dengan musik hanya mencapai 22,22% pada 7 ketetanggaan terdekat.

4. Berdasarkan hasil saran *chord* suara vokal tanpa musik dari ke-sepuluh percobaan menggunakan ketetanggaan yang berbeda, dapat ditarik kesimpulan saran *chord* yang dapat digunakan untuk mengiringi suara vokal masukan.

Adapun saran penulis dalam penelitian ini untuk pengembangan selanjutnya adalah sebagai berikut :

1. Aplikasi selanjutnya dapat diterapkan pada perangkat mobile
2. Aplikasi mampu menenukan tempo dan birama secara otomatis
3. Aplikasi dapat menghasilkan keluaran semua jenis *chord*

REFERENSI

[1] Widhyatama,Sila. Pola Imbal Gamelan Bali Dalam Kelompok Musik Perkusi Cooperland Di Kota Semarang.Semarang:Universitas Negeri Semarang. 2012

[2] Syafitri, Nesi. Perbandingan motode K-Nearest Neighbor (KNN) dan Metode Nearest Cluster Classifier (NCC) dalam Pengklasifikasian Kualitas Batik Tulis.Padang: STMIK Padang. 2010

[3] Herdayandi , Purwangga Bagja. *Aplikasi Pengenalan dan Pembelajaran Akord Gitar Berbasis Web*. Bandung: Universitas Komputer Indonesia.2011

[4] Gaffar, Imam, dkk.Aplikasi Pengkonversi Nada-Nada Instrumen Tunggal Menjadi *Chord* Menggunakan Metode Pitch Class Profile.Semarang:Universitas Diponegoro Semarang. 2012

[5] Nurasyiah.Perancangan Aplikasi Kompresi File Audio dengan Algoritma Aritmetic Coding. Medan: STMIK Budidarma Medan.2013

[6] Hanggarsari ,Prativi Nugraheni, dkk. Simulasi Sistem Pengacakan Sinyal Suara Secara Realtime Berbasis Fast Fourier Transform (FFT). Lampung: Universitas Lampung.2012

[7] Sijabat, Davit Wasty. Simulasi Pengenalan *Chord* Terisolasi Berbasiskan Speaker Dependent Dengan Metode Hidden Markov Model. Depok: Universitas Indonesia.2009

[8] Adler, John, dkk. *Identifikasi Suara dengan MATLAB sebagai Aplikasi Jaringan Syaraf Tiruan*. Bandung: Universitas Komputer Indonesia. 2013