Algoritma Dynamic Programming dalam Pengenalan Suara

Alexander Sukono - 13513023
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
alexander.sukono@gmail.com

***Abstrak*— Pertama kali dikenalkan pada tahun awal 1970an, algoritma *dynamic programming* telah menjadi algoritma yang populer dalam bidang pengenalan suara. Makalah ini menjelaskan tentang bagaimana dengan algoritma *Dynamic Programming* dapat mengenali suara. Jenis teknik algoritma *Dynamic Programming* yang dipakai adalah teknik algoritma *Dynamic Time Warping* (DTW) yang akan dijelaskan kemudian. Dalam suatu sistem pengenalan suara membutuhkan perbandingan antara sinyal masuk dari kata dan berbagai kata-kata di kamus. Masalah ini dapat diselesaikan dengan sangkil menggunakan algoritma DTW yang membandingkan secara dinamis dengan tujuan korespondensi dari kata-kata yang terkait.**

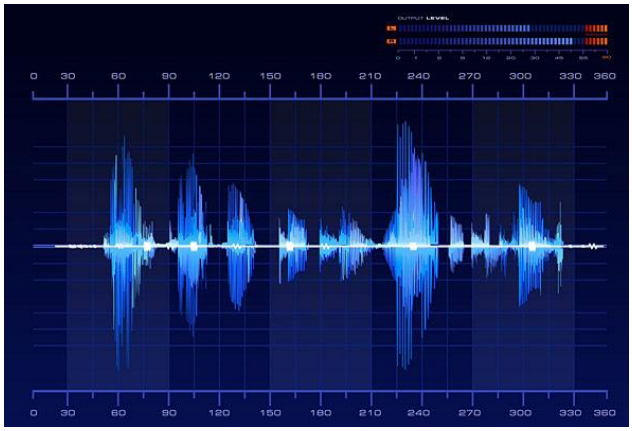
Kata Kunci*— *Dynamic Programming, Speech Recognition,* Pengenalan Suara, *Dynamic Time Warping

I. PENDAHULUAN

Pengenalan suara atau *Automatic Speech Recognition* adalah suatu pengembangan teknik dan sistem yang memungkinkan komputer untuk menerima masukan berupa kata yang diucapkan oleh pembicara. Teknologi ini memungkinkan suatu perangkat untuk mengenali dan memahami kata-kata yang terucap dengan cara digitalisasi kata dan mencocokkan sinyal digital tersebut dengan suatu pola yang tersimpan dalam suatu kamus atau perangkat tertentu.

Pengenalan suara punya sejarah panjang dengan pasang surut inovasi yang utama. Sejak tahun 1940, perusahaan *American Telephone and Telegraph Company* (AT&T) sudah mulai mengembangkan suatu perangkat teknologi yang dapat mengidentifikasi kata yang diucapkan oleh manusia. Lalu sekitar tahun1960-an, para peneliti dari perusahaan tersebut berhasil membuat suatu alat atau perangkat yang dapat mengidentifikasi kata-kata yang terpisah, dan setelah itu pada tahun 1970-an mereka berhasil membuat perangkat yang dapat mengidentifikasi kata-kata yang kontinu. Alat pengenal suara atau ucapan kemudian menjadi sangat fungsional sejak tahun 1980-an dan sekarang sudah menjadi salah satu hal yang tidak asing bagi masyarakat. Teknologi pengenalan suara ini masih terus dikembangkan dan terus ditingkatkan kemangkusannya hingga sekarang.

Aplikasi dari alat pengenal suara dapat ditemukan dalam berbagai bidang seperti di bidang kesehatan contohnya pada aplikasi antarmuka suara pengguna atau *Voice User Interface* (VUI) yang menggunakan teknologi pengenal ucapan dimana pengendalian saklar lampu tidak perlu dilakukan secara manual dengan menggerakkan saklar tetapi cukup dengan mengeluarkan perintah dalam bentuk ucapan atau kata-kata sebagai saklarnya. Metode ini membantu manusia yang secara fisik tidak dapat menggerakkan saklar. Lalu, di bidang militer yaitu pengatur lalu lintas udara atau yang dikenal dengan *Air Traffic Controllers (ATC)* yang dipakai oleh para pilot



Gambar 1 : Gambaran sinyal suara yang akan dibandingkan dengan kata yang terdapat pada sistem [4]

untuk mendapatkan keterangan dan situasi mengenai keadaan lalu-lintas udara seperti radar, cuaca, dan navigasi. Hal ini sangat berguna untuk tidak menyulitkan pilot bila harus berkomunikasi dan menyesuaikan navigasi tertentu dengan terlebih dahulu menekan tombol tertentu. Setelah itu, di bidang komunikasi yaitu komando suara atau *Voice Command* yang terdapat di komputer atau di ponsel pintar terkini. Pengguna tinggal menyesuaikan intonasi dan tata bahasa sehingga komputer dapat membandingkannya dengan daftar perintah yang sudah tersedia dan komando suara akan menampilkan tampilan verifikasi apakah perintah tersebut

sesuai yang diinginkan pengguna atau tidak.

II. DASAR TEORI

Dasar ilmu yang dipakai dalam makalah ini adalah pengenalan ucapan atau *Speech Recognition*, *Dynamic Programming* dan *Dynamic Time Warping* (DWT).

2.1 *Speech Recognition*

Pengenalan suara atau *Speech Recognition* adalah suatu pengembangan teknik dan sistem yang memungkinkan komputer untuk menerima masukan berupa kata yang diucapkan oleh pembicara. Berdasarkan kemampuan dalam mengenal kata yang diucapkan, terdapat 5 jenis pengenalan kata terurut dari yang tertua sampai terbaru yaitu :

1. Kata-kata yang terisolasi
   Proses pengidentifikasian atau pengenalan kata yang hanya dapat mengenal kata yang diucapkan jika kata tersebut memiliki selang waktu pengucapan antar kata
2. Kata-kata yang berhubungan
   Proses pengidentifikasian kata yang mirip dengan kata-kata yang terisolasi, akan tetapi membutuhkan selang waktu pengucapan antar kata yang lebih singkat
3. Kata-kata yang berkelanjutan
   Proses pengidentifikasian kata yang sudah lebih mutakhir karena dapat mengenal kata-kata yang diucapkan secara berkesinambungan dengan selang waktu yang sangat sedikit atau tanpa selang waktu. Proses pengenalan suara ini sangat rumit karena membutuhkan metode khusus untuk membedakan kata-kata yang diucapkan tanpa selang waktu. Pengguna dapat melafalkan kata-kata secara natural
4. Kata-kata spontan
   Proses pengidentifikasian kata yang dapat mengenal kata-kata yang diucapkan secara spontan tanpa jeda waktu antar kata
5. Verifikasi atau identifikasi suara
   Proses pengidentifikasian kata yang tidak hanya mampu mengenal kata, tapi juga mengidentifikasi siapa yang berbicara.

2.2. *Dynamic Programming*

Program Dinamis (*dynamic programming*) adalah metode pemecahan masalah dengan cara menguraikan solusi menjadi sekumpulan tahapan (*stage*) sedemikian sehingga solusi dari persoalan dapat dipandang dari serangkaian keputusan yang saling berkaitan.

Istilah program dinamis muncul karena perhitungan solusi menggunakan tabel yang ukurannya dapat berubah. Kata program tidak berhubungan dengan *source code*, tetapi perencanaan. Karakteristik penyelesaian persoalan dengan program dimanis adalah terdapat sejumlah berhingga pilihan yang mungkin, solusi dibangun secara bertahap, solusi pada setiap tahap dibangun dari solusi tahap sebelumnya, lalu kita menggunakan persyaratan optimasi dan kendala untuk membatasi sejumlah pilihan yang harus dipertimbangkan pada suatu tahap.

Perbedaan program dinamis dengan algoritma lain seperti *Greedy* adalah lebih dari satu rangkaian keputusan yang dipertimbangkan, sedangkan *Greedy* hanya satu rangkaian keputusan yang dihasilkan.

Pada program dinamis, rangkaian keputusan yang optimal dibuat dengan menggunakan prinsip optimalitas. Prinsip optimalitas sendiri adalah jika solusi total optimal, maka bagian solusi sampai tahap ke-k juga optimal. Atau dengan kata lain, jika kita bekerja dari tahap ke *k* ke *k*+1, kita dapat menggunakan hasil optimal dari tahap *k* tanpa harus kembali ke tahap awal. Ongkos pada tahap *k*+1 = (ongkos yang dihasilkan pada tahap *k*) + (ongkos dari tahap *k* ke tahap *k*+1). Ada macam-macam persoalan program dinamis, persoalan tersebut dapat diselesaikan dengan program dinamis jika persoalan dapat dibagi menjadi beberapa tahap, yang pada setiap tahap hanya diambil satu keputusan, masing-masing tahap terdiri dari sejumlah status yang berhubungan dengan tahap tersebut, hasil dari keputusan yang diambil pada setiap tahap ditransformasikan dari status yang bersangkutan ke status berikutnya pada tahap berikutnya, ongkos pada suatu tahap meningkat secara teratur dengan bertambahnya jumlah tahapan, ongkos pada suatu tahap bergantung pada ongkos tahap-tahap yang sudah berjalan dan ongkos pada tahap tersebut, keputusan terbaik pada suatu tahap bersifat independen terhadap keputusan yang dilakukan pada tahap sebelumnya, adanya hubungan rekursif yang mengidentifikasikan keputusan terbaik untuk setiap status pada tahap *k* memberikan keputusan terbaik untuk setiap status pada tahap *k*+1, prinsip optimalitas berlaku pada persoalan tersebut.

Algoritma program dinamis digunakan untuk optimisasi misalnya, untuk mencari jalur terpendek antara 2 titik, atau cara tercepat untuk mengkali banyak matriks. Algoritma program dinamis akan mengecek *sub*-masalah yang sudah terselsaikan sebelumnya dan menggabungkannya dengan solusinya untuk memberikan solusi terbaik dari masalah yang diberikan. Ada banyak alternatif, seperti algoritma *Greedy*, yang menyeleksi pilihan optimal lokal untuk setiap percabangan di jalan. Pilihan optimal lokal adalah pilihan yang tidak bagus, karena tidak menjamin mendapatkan solusi optimal. Misalnya, katakanlah seseorang harus berpindah dari titik A ke titik B secepat mungkin. Algoritma program dinamis akan mencarikan jalan terpendek yang dekat dengan A, dan menggunakan solusinya untuk akhirnya mendapatkan jalur terpendek ke B. Sebaliknya, algoritma *Greedy* akan membawa orang itu ke jalan terpendek yang ada di tiap persimpangan. Seperti yang dibayangkan, strategi ini mungkin tidak akan membawanya dengan waktu sampai yang paling cepat.

Ada dua pendekatan yang digunakan dalam program dinamis yaitu program dinamis maju (*forward* atau *up-down*) dan program dinamis mundur (*backward* atau *bottom-up*).

Berikut langkah-langkah pengembangan algoritma

program dinamis :

1. Karakteristikkan struktur solusi optimal
2. Definisikan secara rekursif nilai solusi optimal
3. Hitung nilai solusi optimal secara maju atau mundur.
4. Konstruksi solusi optimal

Ada berbagai contoh persoalan yang diselesaikan dengan program dinamis dan aplikasinya yaitu untuk mencari lintasan terpendek, penganggaran modal, TSP (*Travelling Salesman Problem*), algoritma CYK (Cocke-Younger-Kasami) untuk menentukan apakah suatu string bisa dibuat dengan masukan *context-free grammar*.

2.3 *Dynamic Time Warping*

Dynamic Time Warping (DTW) adalah sebuah algotima aplikasi dari *Dynamic Programming* untuk mengukur kesamaan antara dua pola yang bersifat sementara yang mungkin bervariasi waktu atau kecepatannya. Contohnya, kesamaan dalam pola berjalan dapat diidentifikasi menggunakan DTW, bahkan jika satu orang berjalan lebih cepat daripada yang lain, atau jika ada percepatan dan perlambatan selama pengamatan. DTW sudah digunakan untuk pola sementara dari video, suara dan data grafik. Semua data yang dapat diubah ke dalam bentuk pola linear dapat dianalisis dengan DTW. Aplikasi yang terkenal adalah *Automatic Speech Recognition.*

Algoritma ini mengkalkulasi nilai jalur melengkung (*warping path*) dari kedua deret yang ingin dibandingkan dan jarak antara mereka. Asumsikan kita mempunyai dua deret numerik sebagai berikut $(a_1,a_2, ..., a_n)$ *dan* $(b_1, b_2, ..., b_m)$. Dapat kita lihat, panjang dari dua deret dapat berbeda. Algoritma memulai dengan kalkulasi jarak lokal antara elemen dari dua deret dengan menggunakan tipe jarak yang berbeda. Metode yang paling sering digunakan untuk kalkulasi jarak adalah rumus jarak Euclidian. Yang menghasilkan matriks jarak dengan *n* baris dan *m* kolom sebagai berikut :

$$d_{ij} = |a_{ij} - b_{ij}| \text{ , } i = 1 \text{ s.d. } n, j = \text{ s.d } m \text{ .}$$

Memulai dengan matriks jarak lokal, lalu matriks jarak minimal antara deret dapat ditentukan menggunakan algoritma *dynamic programming* dengan kriteria optimasi berikut :

$$a_{ij} = d_{ij} + min(a_{i-1,j-1}, a_{i-1,j}, a_{i,j-1})$$

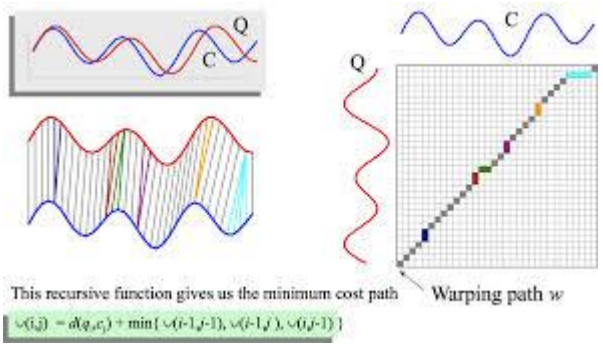
dimana a_{ij} adalah jarak minimal antara deret bagian $(a_1,a_2, ..., a_i)$ *dan* $(b_1, b_2, ..., b_j)$. Sebuah jalan melengkung (*warping path*) adalah jalan melalui matriks jarak minimal dari elemen a_{11} ke elemen a_{nm} yang terdiri dari a_{ij} elemen yang membentuk jarak a_{nm}. Biaya global *warp* dari dua deret dengan di bawah ini :

$$\mathrm{GC} = \frac{1}{p}\sum_{i=1}^{p} W_i$$

dimana W_i adalah elemen-elemen yang dimiliki oleh jalur melengkung, dan p adalah jumlah dari mereka. Kalkulasi dibuat untuk dua deret yang ditunjukkan oleh tabel di bawah ini berikut dengan jalur melengkungnya yang disorot.

	2	*3*	*7*	*7*	*8*	*8*	*6*	*2*	*5*
4	2	2	3.5	5	7	8	8	9	9
7	4.5	4	2	2	2.5	3	3.5	6	7
7	5.5	6	2	2	2.5	3	3.5	6	7
8	7	8.5	2.5	2.5	2	2	3	6	7.5
2	4	4.5	5	5	5	5	5	3	4.5
2	4	4.5	7	7.5	8	8	7	3	4.5

Tabel 1 : Jalur melengkung (warping path)



Gambar 2 : Ilustrasi perbandingan gelombang dengan warp path[6]

Ada 3 kondisi yang diharuskan di algoritma DTW yang memastikan mereka konvergen dengan cepat :

1. monoton : jalur tidak pernah kembali, yang berarti indeks i dan j digunakan untuk menyebrang deret tidak pernah berkurang.
2. kontinuitas : jalur maju bertahap, indeks i dan j naik dengan maksimum 1 per tahap
3. batasan : jalur mulai di pojok kiri atas akan berakhir di pojok kaan bawah, dan jika mulai di pojok kiri bawah akan berakhir di pojok kanan atas.

Contoh dari implementasi jalur melengkung menggunakan bahasa pemrograman Java ditunjukkan di bawah ini :

```
public static void dtw(double s1[],double
s2[],double dm[][], Stack<Double> w)
// s1,s2 = deret, dm = jarak minimal
// w = jalur melengkung
{
    int n=s1.length,m=s2.length;
    double d[][]=new double[n][m];
    //matriks jarak euclidian
    for(int i=0;i<n;i++)
        for(intj=0;j<m;j++)
            d[i][j]=Math.abs(s1[i]-s2[j]);

    //menentukan jarak minimal
    dm[0][0]=d[0][0];

    for(int i=1;i<n;i++)
        dm[i][0]=d[i][0]+dm[i-1][0];

    for(int j=1;j<m;j++)
        dm[0][j]=d[0][j]+dm[0][j-1];

    for(int i=1;i<n;i++)
        for(int j=1;j<m;j++)
            if(dm[i-1][j-1]<=dm[i-1][j])
                if(dm[i-1][j-1]<=dm[i][j-
```

```
                    1])
                        dm[i][j]=d[i][j]+dm[i
                        -1][j-1];
                    else
                        dm[i][j]=d[i][j]+dm[i
                        ][j-1];
                else
                    if(dm[i-1][j]<=dm[i][j-
                    1])
                        dm[i][j]=d[i][j]+dm[i
                        -1][j];
                    else
                        dm[i][j]=d[i][j]+dm[i
                        ][j-1];

    int i=n-1,j=m-1;

    // menentukan jalur melengkung
    w.push(new Double(dm[i][j]));
    do{
        if(i>0&&j>0)
            if(dm[i-1][j-1]<=dm[i-1][j])
                if(dm[i-1][j-1]<=dm[i][j-
                1])
                {
                    i--;
                    j--;
                }
                else
                    j--;
            else
                if(dm[i-1][j]<=dm[i][j-1])
                    i--;
                else
                    j--;
        else if(i==0)
             j--;
        else
             i--;

        w.push(new Double(dm[i][j]));
    } while(i!=0||j!=0);
}
```

Seperti algoritma *dynamic programming* yang lain, DTW memiliki kompleksitas polinomial yaitu $O(n^2)$. Ketika deret memiliki sangat banyak jumlah elemen, setidaknya ada dua perhitungan yang membebani yaitu:

- mengingat banyak jumlah matriks yang besar
- kalkulasi banyak jumlah jarak

Ada peningkatan dalam algoritma DTW yang mengatasi dua masalah di atas yaitu : *FastDTW* (*Fast Dynamic Time Warping)*. Solusi yang ditawarkan terdiri dari membagi matriks-matriks jarak ke 2,4,8,16, dan seterusnya matriks ke dimensi yang lebih kecil dengan proses membagi dua secara terus menerus. Dengan cara ini, proses kalkulasi jarak dapat berkurang dan jalur melengkung dapat digabungkan dengan cara menggabungkan jalur melengkung yang terkalkulasi untuk matriks kecil yang terbagi-terbagi. Solusi membagi-bagi dan menggabungkan ini menggunakan metode *Divide & Conquer* yang sudah diajarkan pada saat perkuliahan.

III. METODE ANALISIS ALGORITMA DTW DALAM PENGENALAN SUARA

Pada bagian ini, akan dibahas mengenai analisis algoritma DTW dalam pengenalan suara atau *Speech Recognition*

3.1 Analisis Sinyal Suara

Suara berjalan melalui lingkungan sebagai gelombang longitudinal dengan sebuah kecepatan yang bergantung pada densitas lingkungannya. Cara atau jalan termudah untuk menggambarkannya adalah dengan grafik sinusoidal. Grafik menampilkan variasi dari tekanan udara yang bergantung dengan waktu.

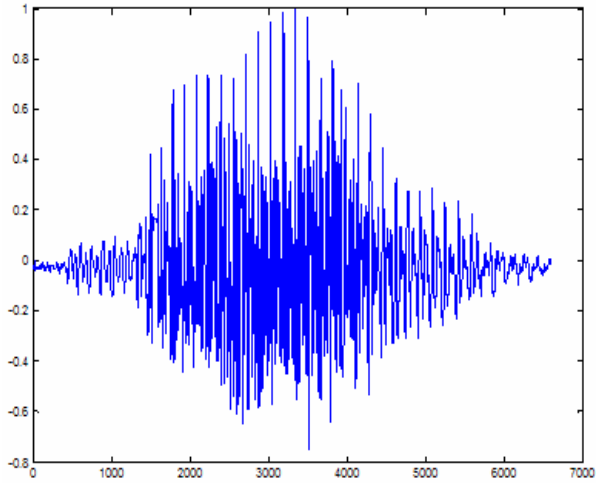
Bentuk dari gelombang suara bergantung pada 3 faktor : amplitudo, frekuensi dan fase. Amplitudo adalah perpindahan grafik sinusoidal di atas dan di bawah sumbu sementara yaitu y=0. Pengukuran amplitudo menggunakan desibel sangat penting karena merupakan representasi langsung bagaimana volume suara dirasakan oleh kita. Frekuensi adalah banyak siklus sinusoid yang dibuat tiap detik. Setiap siklus terdiri dari sebuah osilasi dimulai dengan garis sedang, lalu mencapai maksimum, setelah itu mencapai minimum dan kembali lagi ke garis sedang atau tengah. Frekuensi diukur dalam siklus per detik atau Hertz (Hz). Kebalikan dari frekuensi adalah periode. Faktor yang terakhir adalah fase. Fase mengukur posisi dari awal kurva sinusidal. Fase tidak bisa dirasakan oleh indra manusia, tetapi manusia dapat mendeteksi perubahan relatif fase antara dua sinyal. Pada kenyataannya, dengan cara ini sistem saraf manusia merasakan lokasi sebuah suara dengan menghitung perbedaan fase yang dirasakan oleh telinga.

Untuk mengkonversi sebuah sinyal suara ke kurva sinusoidal kita menggunakan Teorema Fourier yang mengatakan gelombang kompleks periodik dapat dipecah ke bentuk kurva sinusoidal dengan mempunyai frekuensi, amplitudo, dan fase yang berbeda. Proses ini dinamakan analisis Fourier dan menghasilkan sebuah himpunan amplitudo, fase, dan frekuensi dari setiap gelombang sinusoidal. Menambahkan semua kurva sinusoidal ini kita akan mendapatkan gelombang suara yang asli. Frekuensi atau fase bersamaan dengan amplitudo dinamakan spektrum. Sinyal periodik menunjukkan model waktu rekursif yang berhubungan ke laju vibrasi pertama dari sinyal yang dinamakan frekuensi fundamental. Sebuah spektrum menunjukan frekuensi dari deret pendek dalam sebuah suara, dan jika kita ingin menganalisis perubahan yang bergantung pada waktu, kita butuh suatu cara untuk menunjukannya. Cara ini dinamakan dengan sebuah spektogram.

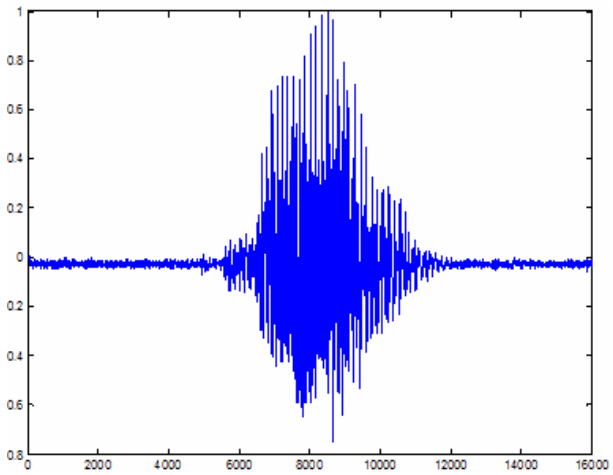
Sebuah spektogram adalah diagram dua dimensi yang terdiri dari frekuensi dan waktu yang mempunyai warna berbeda dimana gelap adalah kuat, dan terang adalah lemah yang menunjukan intensitas amplitudo. Spektogram mempunyai peran besar dalam pengenalan suara dan para ahli dapat menentukan banyak detil hanya dengan melihat spektrogram dari sebuah suara.

3.2 Teknik Isolasi Suara

Teknik pengidentifikasian sekarang ini dapat secara akurat mengidentifikasi poin mulai dan akhir dari kata yang diucapkan dalan aliran bunyi atau *audio stream*, berdasarkan pemrosesan sinyal bergantung pada waktu. Menentukan titik mulai dan akhir adalah masalah yang mudah jika rekaman suara dilakukan dalam kondisi yang ideal. Dalam kasus ini perbandingan sinyal dengan bunyi



Gambar 3 : Sinyal suara sebelum diisolasi[6]



Gambar 4 : Sinyal suara sesudah diisolasi[6]

berisik sangat besar karena sangat mudah untuk menentukan lokasi dalam aliran yang berisi sinyal yang benar dengan menganalisis sampel. Dalam kondisi nyata tidak semudah itu, bunyi latar belakang mempunyai intensitas yang signifikan dan dapat menganggu proses isolasi dari kata dalam aliran. Oleh karena itu, ada sebuah algoritma bernama Rabiner-Lamel yang berfungsi menghilangkan suara latar belakang tersebut. Algoritma ini menganggap titik mulai dimana energi mengesampingkan level yang superior dan laju nilai positif dan negatif tidak mengesampingkan level yang sudah ditentukan dianggap sebagai titik mulai area suara atau titik di mana tidak diam. Algoritma Rabiner-Lamel secara detil tidak dibahas dalam makalah ini. Pengisolasian suara dapat dilihat pada contoh gambar di bawah :

3.3 Identifikasi Suara Menggunakan Algoritma DTW

Sesudah melalui proses isolasi, identifikasi suara dapat dilakukan dengan perbandingan langsung dari bentuk numerik sebuah sinyal atau dengan perbandingan spektogram sinyal. Proses perbandingan dalam kedua kasus harus berimbang untuk kedua panjang yang berbeda dari deret-deret dan sifat alami non-linear dari suara. Algoritma DTW berhasil mengatasi masalah-masalah tersebut dengan mencari jalur melengkung yang berkorespondensi dengan jarak optimal antara dua deret yang mempunyai panjang yang berbeda. Ada beberapa ciri khas ketika algoritma ini diaplikasikan ke dua kasus di bawah ini :

1. Perbandingan langsung bentuk numerik atau sinyal. Dalam kasus ini, untuk setiap deret numerik, sebuah deret baru terbentuk, deret yang mempunyai dimensi lebih kecil. Deret numerik dapat mempunyai ribuan nilai numerik, sementara subderetnya ratusan. Mengurangi nilai numerik dapat dilakukan dengan menghilangkan mereka yang berada di antara titik ekstrim. Proses mengurangi panjang deret numerik ini harus tidak mengubah bentuknya. Tampaknya, proses ini mengurangi kepresisian dalam pengenalan. Akan tetapi, kepresisian nyatanya kembali lagi ke banyaknya jumlah kata yang terdapat dalam kamus sistem.
2. Representasi sinyal spektogam dan mengaplikasikan algoritma DTW untuk membandingkannya. Metode terdiri dari membagi sinyal numerik ke beberapa jendela atau interval yang akan tumpang tindih. Untuk setiap jendela, interval nilai riil, frekuensi suara akan dikalkulasi dan disimpan ke dalam suatu matrix. Parameter akan sama setiap operasi kalkulasi panjang jendela dan panjang transformasi Fourier. Transformasi Fourier secara simetris berhubungan ke tengah dan sejumlah bilangan komplek dari pertengahan kedua yang adalah bilangan kompleks konjugasi dari bilangan-bilangan yang tidak simetris dari pertengahan pertama. Karena kejadian ini, hanya nilai dari pertengahan pertama yang dapat disimpan, jadi spektogram akan menjadi matriks bilangan kompleks, panjang barisnya sama dengan setengah dari panjang transformasi Fourier dan panjang kolomnya bergantung pada panjang suara. Setelah itu, algoritma DTW dapat dijalankan untuk membandingkan kedua gelombang suara.

IV. KESIMPULAN

Makalah ini menjelaskan tentang aplikasi algoritma DTW untuk membandingkan kedua pola sinyal suara untuk dipakai pada alat pengenalan suara. DTW berguna untuk mengenali suara yang sudah terisolasi dengan data kamus dalam jumlah sedikit. Menggunakan *dynamic programming* memastikan kompleksitas algoritmanya $O(n^2 v)$, dimana n adalah panjang dari pola sinyal dan v adalah banyak kata di dalam kamus. Tetapi juga ada kelemahan dari DTW yaitu makin banyak jumlah kamus yang memastikan bahwa kenaikan rata-rata sukses proses

pengenalan suara mengakibatkan waktu pemrosesan yang makin lama, jadi harus memilih antara ketepatan atau kecepatan. Akan tetapi, DTW tetap menjadi algoritma yang mudah dipakai dan sangat cocok untuk aplikasi yang hanya mengenali kata sederhana seperti aplikasi untuk komputer mobil, sistem keamanan, dan lain-lain

V. PENGAKUAN

Alexander Sukono, sebagai penulis dari makalah ini, ingin menyampaikan ucapan syukur yang setinggi-tingginya kepada Tuhan YME atas berkat-Nya sehingga saya dapat menyelesaikan makalah ini. Dan juga ingin memberikan ucapan terimakasih yang sebesarnya kepada Dr. Nur Ulfa Maulidevi, S.T., M.Sc dan Dr.Ir. Rinaldi Munir, M.T. atas bimbingannya sehingga penulis dapat mengerti konsep-konsep Strategi Algoritma, dan semua orang yang telah memberikan dukungan kepada penulis dalam berbagai bentuk untuk menyelesaikan makalah ini.

REFERENSI

[1] Munir, Rinaldi (2005), Diktat Kuliah IF 2251 Strategi Algoritmik

[2] http://www.metode-algoritma.com/2013/06/pengenalan-suara-voice-speech.html, diakses pada tanggal 2 Mei 2015 pukul 20:24 WIB

[3] Al-Naymat, G., Chawla, S., & Taheri, J. (2012). SparseDTW: A Novel Approach to Speed up Dynamic Time Warping

[4] http://web.science.mq.edu.au/~cassidy/comp449/html/ch11s02.html, diakses pada tanggal 2 Mei 2015 pukul 23:10 WIB

[5] https://www.topcoder.com/community/data-science/data-science-tutorials/dynamic-programming-from-novice-to-advanced/, diakses pada tanggal 3 Mei 2015 pukul 10:35 WIB

[6] Felix, Titus (2008), Dynamic Programming Algorithms in Speech Recognition

[7] http://www.datanami.com/2014/05/30/deep-neural-networks-power-big-gains-speech-recognition/, diakses pada tanggal 3 Mei 2015 pukul 11:01 WIB

[8] http://www.andrew.cmu.edu/user/mmohta/15418Project/finalreport.html, diakses pada tanggal 3 Mei 2015 pukul 12:44 WIB

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 4 Mei 2015

Alexander Sukono - 13513023