II.3 Teori Pendukung

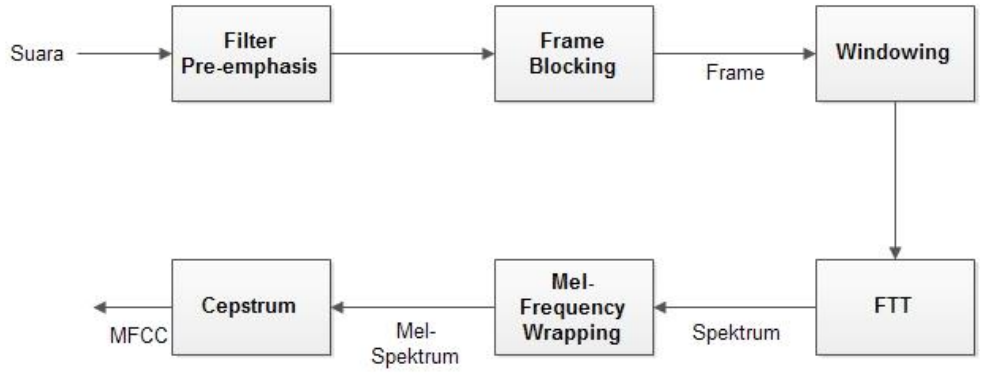
Teori pendukung dibutuhkan untuk mendapatkan pemahaman yang baik mengenai sistem yang akan direalisasikan. Teori pendukung yang dapat memberikan pemahaman tersebut yang dikhususkan untuk sistem yang akan penulis rancang yaitumetode konversi sinyal suara Mel-Frequency Cepstrum Coefficients (MFCC).

II.3.1 Mel-Frequency Cepstrum Coefficients (MFCC)

MFCC merupakan salah satu metode yang mengkonversikan sinyal suara menjadi beberapa paramater. MFCC memiliki beberapa keunggulan, yaitu:

1. Mampu menangkap karakteristik suara yang sangat penting bagi pengantar suara, atau dengan kata lain dapat menangkap informasi-informasi penting yang terkandung dalam sinyal suara.
2. Menghasilkan data seminimal mungkin, tanpa menghilangkan informasi-informasi penting yang dikandungnya.
3. Mereplikasi organ pendengaran manusia dalam melakukan persepsi terhadap sinyal suara.

Ekstraksi ciri menggunakan MFCC juga merupakan adaptasi dari sistem pendengaran manusia, di mana sinyal suara akan difilter secara linear untuk frekuensi rendah (di bawah 1000 Hz) dan secara logaritmik untuk frekuensi tinggi (di atas 1000 Hz). Blok diagram untuk MFCC ditunjukkan pada **Gambar 2.0**.



Gambar 2.0 Blok diagram MFCC

II.3.1.1 Filter *Pre*-Emphasis

Proses pemfilteran sinyal suara diperlukan setelah proses perekaman atau *sampling*. Tujuan dari pemfilteran adalah untuk mendapatkan bentuk spektral frekuensi sinyal suara yang lebih halus. Filter pre-emphasis didasari oleh hubungan *input/output* dalam domain waktu yang dinyatakan dalam persamaan:

$$y(n)=x(n)-ax(n-1)$$

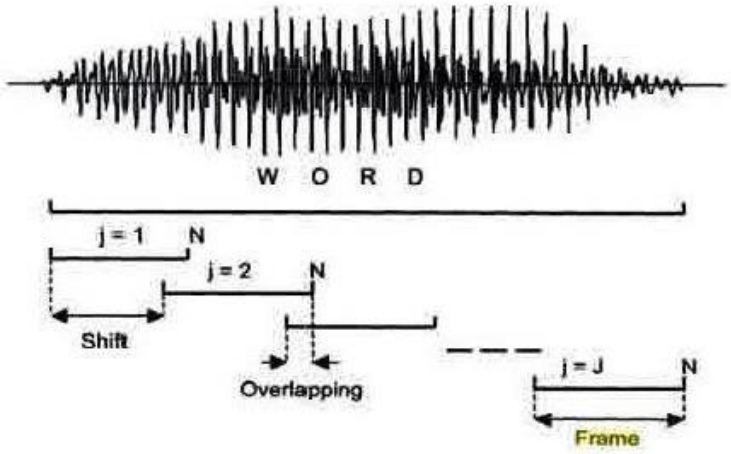
Keterangan :
y = hasil perhitungan sinyal suara
n = nomor urutan sinyal suara
x = masukan sinyal suara
a = konstanta filter pre-emphasis, biasanya bernilai $0,9 \leq a \leq 1,0$

Proses pemfilteran dengan pre-emphasis mempunyai dua keuntungan utama, yaitu:

1. Sinyal suara secara alami memiliki kemiringan spektral yang negatif berkisar 20 dB/decade karena karakter fisik dari pembentukan ucapan. Filter pre-emphasis berfungsi untuk mengimbangi kemiringan alami spektral sebelum dilakukan analisis, sehingga dapat meningkatkan efisiensi dari analisis.
2. Pendengaran manusia lebih sensitif dengan spektrum yang berkisar di atas 1kHz. Filter pre-emphasis menguatkan spektrum di area di atas 1 kHz tersebut.

II.3.1.2 Frame Blocking

Frame Blocking adalah suatu proses di mana sinyal suara dibagi menjadi beberapa potongan yang nantinya dapat memudahkan dalam perhitungan dan analisa suara. Setiap potongan-potongan dari sinyal suara disebut *frame*. Satu *frame* terdiri dari beberapa sampel tergantung tiap detik suara akan disampel dan berapa besar frekuensi *sampling*-nya.



Gambar 2.1 *Frame Blocking*

Berdasarkan pada **Gambar 2.1**, sinyal suara dibagi menjadi beberapa *frame* dan saling bertumpang tindih (*overlapping*). *Overlapping* dilakukan untuk

menghindari hilangnya ciri atau karakteristik suara pada perbatasaan perpotongan setiap frame. Panjang daerah *overlap* yang umum digunakan adalah kurang lebih 30% - 50% dari panjang *frame*.

Frame pada *frame blocking* pada umumnya memiliki panjang antara 10-30ms. Panjang *frame* yang digunakan sangat mempengaruhi keberhasilan dalam analisia spektral. Di satu sisi, ukuran dari *frame* harus sepanjang mungkin untuk dapat menunjukkan resolusi frekuensi yang baik. Tetapi di sisi lain ukuran *frame* juga harus cukup pendek untuk dapat menunjukkan resolusi waktu yang baik.

II.3.1.3 Proses *Windowing*

Proses *framing* dapat menyebabkan terjadinya kebocoran spektral atau *aliasing.* Efek ini dapat terjadi karena rendahnya jumlah *sample rate,* atau karena proses *frame blocking* di mana menyebabkan sinyal menjadi *discontinue.* Untuk mengurangi kemungkinan terjadinya kebocoran spektral, maka hasil dari proses *framing* harus melewati proses *windowing*.

Fungsi *window* yang paling sering digunakan dalam aplikasi *speech recognition* adalah *Hamming Window*. Fungsi ini menghasilkan *sidelobe level* yang tidak terlalu tinggi (kurang lebih -43 dB) selain itu *noise* yang dihasilkan juga tidak telalu besar (kurang lebih 1,36 BINS).

Fungsi *Hamming Window* adalah sebagai berikut:

$$w_{ham}(n) = 0.54 - 0.46 \cos \frac{2\pi n}{M-1}$$

Keterangan:

$W_{ham}(n)$ = keluaran dari proses *Hamming Window*
n = 0, 1, 2, ..., – 1
M = panjang *frame*

II.3.1.4 Fast Fourier Transform (FFT)

FFT merupakan salah satu metode untuk transformasi sinyal suara menjadi sinyal frekuensi. Artinya proses perekaman suara disimpan dalam bentuk digital berupa gelombang spektrum suara berbasis frekuensi. Hasil dari proses FFT menghasilkan pendeteksian gelombang frekuensi domain dalam bentuk diskrit.

FFT merupakan turunan dari persamaan *Discrete Fourier Transform* (DFT) di mana jumlah perhitungan digital pada DFT dapat dikurangi secara signifikan. Sehingga dengan adanya FFT maka perhitungan digital terhadap spektrum-spektrum frekuensi dapat diwujudkan secara sederhana dalam implementasinya. Prinsip dasar FFT adalah menguraikan penghitungan N-titik

DFT menjadi penghitungan DFT dengan ukuran yang lebih kecil dan memanfaatkan periodisitas dan simetri dari bilangan kompleks. FFT dapat dituliskan dalam bentuk sinusoidal sebagai berikut [9]:

$$x(k) = \frac{1}{N}\sum_{n=1}^{N=1} x(n) \cos\left(\frac{2\pi kn}{N}\right) - \quad jSin\left(\frac{2\pi kn}{N}\right)$$

Keterangan:

N = koefisien sinus dan cosinus pada $2\pi kn$
k = indeks dari frekuensi pada frekuensi ke-*N*
n = indeks waktu
x(k) = nilai dari spektrun ke- (domain frekuensi)
x(n) = nilai sinyal pada domain waktu

II.3.1.5 Mel-Frequency *Wrapping*

Mel-Frequency Wrapping biasanya dilakukan menggunakan *Filterbank*. *Filterbank* adalah salah satu bentuk dari filter yang dilakukan dengan tujuan untuk mengetahui ukuran energi dari *frequency band* tertentu dalam sinyal suara. Pada MFCC, *Filterbank* diterapkan dalam domain frekuensi. Berikut ini adalah rumus yang digunakan dalam perhitungan *filterbanks*:

$$y[i] = \sum_{j=1}^{N} S[j]H_i[j]$$

Keterangan :

y[i] = hasil dari *mel-frequency wrapping*
S[j] = *magnitude spectrum* pada frekuensi j
$H_i[j]$ = koefisien *filterbank* pada frekensi ($1 \leq \leq$ M)
M = jumlah channel dalam *filterbank*

II.3.1.6 *Cepstrum*

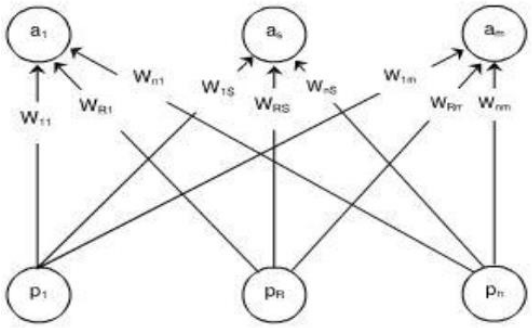
Cepstrum adalah sebutan kebalikan untuk *spectrum*. *Cepstrum* biasa digunakan untuk mendapatkan informasi dari suatu sinyal suara yang diucapkan oleh manusia. Pada langkah terakhir ini, spektrum log mel dikonversi menjadi *cepstrum* menggunakan *Discrete Cosine Transform* (DCT). Hasil dari proses ini dinamakan *Mel-Frequency Cepstrum Coefficients* (MFCC).

MFCC ini adalah hasil alihragam kosinus dari logaritma *short-term power spectrum* yang dinyatakan dalam skala mel-frekuensi. Bila *mel power spectrum coefficients* dinotasikan sebagai Sk, k = 1,2…..K , Minh N. Do mendefinisikan koefisien dari MFCC (ĉn) sebagai:

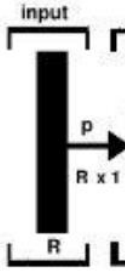
II.3.1.7 *Jaringan* Saraf Tiruan *Linear Vector Quantization* (LVQ)

LVQ *merupakan* salah satu jaringan saraf tiruan dengan pembelajaran terbimbing (*supervised learning*). LVQ mengklasifikasikan *input* secara berkelompok ke dalam kelas yang sudah didefinisikan melalui jaringan yang telah dilatih. Dengan kata lain LVQ mendapatkan n *input* dan mengelompokkan ke dalam m *output*.

Pada dasarnya arsitektur LVQ sama dengan *self-organizing map* kohonen (tanpa struktur topologi yang diasumsikan dengan *layer output*). Sebagai tambahan setiap *layer output* mempunyai kelas yang diketahui.



Gambar 2.2 Arsitektur LVQ sederhana



$$cn = \sum_{k-1}^{K} (\log Sk) \cos\left[n\left(k - \frac{1}{2}\right)\frac{\pi}{K}\right], n = 1,2,\ldots,K$$

Gambar 2.3 Arsitektur LVQ dengan *competitive layer* dan *linear layer*

Pada **Gambar 2.2** menunjukkan arsitektur LVQ sederhana, di mana hanya terdapat *layer input*, bobot, dan *layer output*. Pada **Gambar 2.3** menunjukkan arsitektur LVQ yang terdiri dari sebuah *layer input*, sebuah *competitive layer*, dan sebuah *linear layer*. *Competitive layer* mengklasifikasikan vektor *input* ke dalam

sejumlah *cluster* berdasarkan jarak yang terdapat di antara masing-masing vektor masukannya. Selanjutnya *linear layer* memetakan kelas yang didapatkan oleh *competitive layer* ke dalam kelas yang telah didefinisikan sebelumnya oleh pengguna, dalam layer ini menggunakan fungsi aktivasi *linear* dengan tujuan agar *input* sebanding dengan *outputnya.*

Algoritma LVQ bertujuan akhir mencari nilai bobot yang sesuai untuk mengelompokkan vektor-vektor ke dalam kelas tujuan yang telah diinisialisasi pada saat pembentukan jaringan LVQ. Sedangkan algoritma pengujiannya adalah menghitung nilai *output* (kelas vektor) yang terdekat dengan vektor *input*, atau dapat disamakan dengan proses pengklasifikasian. Algoritma pembelajaran LVQ adalah sebagai berikut:

1. Inisialisasi vektor referensi ; inisialisasi rating pembelajaran α (0)
2. Ketika kondisi berhenti adalah *false*, lakukan langkah 2 sampai 6
3. Untuk setiap *input* pelatihan vektor x lakukan langkah 3-4
4. Temukan j hingga $||p - Wp||$ minimum
5. Perbaharui Ws sebagai berikut :
   Jika T = Cs, maka
   Ws (baru) = Ws (lama) + α[p – Ws (lama)];
   Jika T $\neq$ Cs, maka
   Ws (baru) = Ws (lama) – α[p – Ws(lama)];
6. Kurang rating pelatihan
7. Tes kondisi berhenti, yaitu kondisi yang mungkin menetapkan sebuah jumlah tetap dari iterasi atau *rating* pembelajaran mencapai nilai kecil yang cukup.

Keterangan :

p	: vektor pelatihan (*input*) (p1, ..., pR, ..., pn)
T	: kategori yang tepat atau kelas untuk vektor pelatihan
Ws	: bobot vektor untuk unit *output* ke-s (W11, ..., WR1, ..., Wn1)
Cs	: kategori atau kelas yang ditampilkan oleh unit output ke-s
$\|\|p - Wj\|\|$	: jarak *Euclidean* antara vektor *input* dan bobot vektor untuk *layer output* ke-s.