BAB II

LANDASAN TEORI

2.1 Nada (Pitch), Frekuensi, dan Persepsi Manusia

Gelombang suara, seperti gelombang lainnya, tercipta karena adanya suatu objek yang melewati suatu medium sehingga menyebabkan medium tersebut bergetar (Wolfe, 2001), seperti pita suara manusia, string gitar, ataupun getaran diafragma pada *speaker* televisi. Setiap gelombang memiliki frekuensi. Frekuensi dari suatu gelombang menunjukkan seberapa sering suatu medium bergetar. Frekuensi diukur berdasarkan jumlah getaran bolak-balik per unit waktu, dengan satuan Hertz (Wolfe, 2001).

Telinga manusia merupakan detektor sensitif, yang mampu mendeteksi fluktuasi tekanan udara yang mengenai gendang telinga (Wolfe, 2001). Telinga manusia mampu mendeteksi gelombang suara dengan berbagai frekuensi, yang berkisar antara 20 Hz sampai 20000 Hz. Segala suara dengan frekuensi di bawah 20 Hz disebut infrasonik dan yang berada di atas 20000 Hz disebut ultrasonik.

Sensasi ketika mendengarkan suatu frekuensi disebut nada (Wolfe, 2001). Dengan kata lain nada itu tidak nyata, nada merupakan sebuah persepsi manusia yang menggambarkan tinggi atau rendahnya suatu nada (Klemens, 2014). Nada tinggi merupakan persepsi untuk gelombang dengan frekuensi tinggi dan nada rendah merupakan persepsi untuk gelombang dengan frekuensi rendah.

2.2 MIDI Note Numbering

Musical Instrument Digital Interface, atau MIDI, adalah sebuah protokol standar industri musik yang mendefinisikan sistem untuk penomoran not. Sebuah nomor not MIDI dapat dihitung menggunakan rumus 2.1 (Mcleod, 2008).

$$n = 69 + 12 \log_2 (f/440) \quad \text{...Rumus 2.1}$$

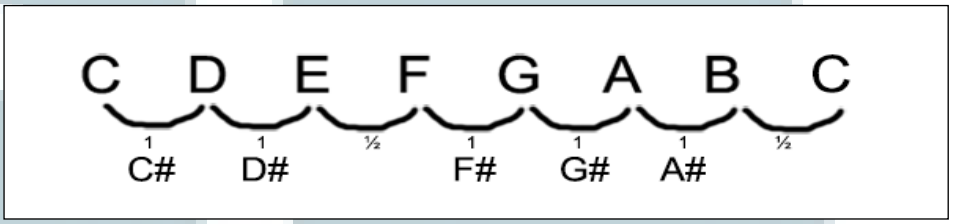
dimana f adalah frekuensi dan n adalah hasil nomor not MIDI. Angka 69 melambangkan not A_4 pada frekuensi 440Hz. Nilai n dapat berupa bilangan *real* yang melambangkan sebuah nada di antara nada regular. Tabel 2.1 menunjukkan beberapa konversi frekuensi menjadi nomor not MIDI (Mcleod, 2008).

Tabel 2.1 Tabel Konversi Nada

Nada	MIDI *note number*	Frekuensi (Hz)
A0	21	27.50
A♯0	22	29.14
B0	23	30.87
C1	24	32.70
C♯1	25	34.65
D1	26	36.71
D♯1	27	38.89
E1	28	41.20
F1	29	43.65
F♯1	30	46.25
G1	31	49.00
G♯1	32	51.91
A1	33	55.00
A♯1	34	58.27
B1	35	61.74
C2	36	65.41
C♯2	37	69.30
D2	38	73.42
D#2	39	77.78
E2	40	82.41
F2	41	87.31
F#2	42	92.50
G2	43	98.00
G#2	44	103.83
A2	45	110.00
A#2	46	116.54

2.3 Tangga Nada Kromatik

Kumpulan dari semua nada dalam musik disebut sebagai tangga nada kromatik (Prasetio, 2014). Terdapat 12 macam notasi nada, 7 di antaranya dinamakan menggunakan 7 huruf pertama dari abjad, yaitu dari A sampai G. Kelima nada lainnya diberi nama dengan menempatkan tanda kres (#) setelah notasi nada (Prasetio, 2014). Berikut susunan tangga nada kromatik beserta jaraknya.



Gambar 2.1 Tangga Nada Kromatik

2.4 Oktaf

Oktaf adalah interval antara dua buah not dengan perbandingan frekuensi 2 : 1 (Mcleod, 2014). Satu oktaf terdiri dari satu tangga nada kromatik dan akan berulang untuk oktaf selanjutnya. Penamaan oktaf diletakkan di belakang suatu notasi nada, seperti C1, D1, E1, C2, dan seterusnya.

2.5 Pitch Detection Algorithm

Pitch Detection Algorithm merupakan suatu algoritma yang dirancang untuk memperkirakan suatu frekuensi dasar dari suatu sinyal (Mcleod, 2008). *Pitch Detection Algorithm* digunakan dalam berbagai konteks, seperti sistem pertunjukan musik, pencarian informasi musik, pembuatan aplikasi yang berhubungan dengan audio, dan lain sebagainnya. *Pitch Detection Algorithm* dapat dilakukan menggunakan domain waktu, frekuensi, ataupun gabungan keduanya.

Pitch Detection Algorithm berbasis domain waktu mengolah data dalam bentuk mentah yang dibaca langsung dari *sound card.* Secara umum *Pitch Detection Algorithm* berbasis domain waktu memperkirakan periode dari sebuah sinyal, kemudian membalikkan nilai tersebut untuk mendapatkan frekuensi dari sinyal tersebut (Mcleod, 2008). Contoh dari *Pitch Detection Algorithm* berbasis domain waktu adalah *zero-crossing, autocorrelation, square difference function, average magnitude difference function,* dan *mcleod pitch method.*

Pitch Detection Algorithm berbasis domain frekuensi tidak menggunakan sinyal mentah secara langsung, melainkan memproses data domain waktu dan mengubahnya menjadi domain frekuensi. Hal ini dilakukan menggunakan transformasi *Fourier.* Contoh dari *Pitch Detection Algorithm* berbasis domain frekuensi adalah *Spectrum Peak Methods, Phase Vocoder,* dan *Harmonic Product Spectrum* (Mcleod, 2008).

2.6 McLeod Pitch Method

Metode pendeteksi nada ini dikemukakan oleh Philip McLeod dalam tulisannya yang berjudul *A Smarter Way To Find Pitch* (2003). Metode ini merupakan kombinasi dari autokorelasi, *normalized square difference function,* dan *peak picking algorithm* pada *Pitch Detection Algorithm* berbasis domain waktu.

Autokorelasi adalah sebuah fungsi untuk menemukan pola berulang dengan cara menghitung korelasi antara suatu sinyal dengan *phased version* dari sinyal itu sendiri. *Phase* adalah suatu variabel. Fungsi autokorelasi dilakukan

untuk semua perbedaan fase yang mungkin. Rumus 2.2 menunjukkan notasi matematik dari fungsi autokorelasi (Broenink,2011).

$$r't(\tau) = \sum_{j=t}^{W-\tau} X_j X_{j+\tau} \quad \text{...Rumus 2.2}$$

dimana r'$t(\tau)$ adalah fungsi autokorelasi dari *lag* τ dihitung mulai waktu t, W adalah ukuran *window*, dan *X* adalah sinyal diskrit.

Square Difference Function (SDF) adalah fungsi untuk menghitung perbedaan antara suatu sinyal dengan *phased version* dari sinyal itu sendiri dengan mengambil perbedaan kuadrat untuk setiap nilai. Rumus 2.3 menunjukkan notasi matematik dari fungsi SDF (Broenink,2011).

$$d't(\tau) = \sum_{j=t}^{W-\tau} (X_j - X_{j+\tau})^2 \quad \text{...Rumus 2.3}$$

Penjabaran dari *square difference function* akan menghasilkan fungsi autokorelasi di dalamnya, seperti yang ditunjukkan pada rumus 2.4 (Broenink,2011).

$$d't(\tau) = \sum_{j=t}^{W-\tau} X_j^2 + X_{j+\tau}^2 - 2X_j X_{j+\tau} \quad \text{...Rumus 2.4}$$

dan apabila bagian pertama didefinisikan sebagai (Broenink,2011)

$$m't(\tau) = \sum_{j=t}^{W-\tau} (X_j^2 + X_{j+\tau}^2) \quad \text{...Rumus 2.5}$$

maka dapat didefinisikan bahwa (Broenink,2011)

$$d't(\tau) = m't(\tau) - 2r't(\tau) \quad \text{...Rumus 2.6}$$

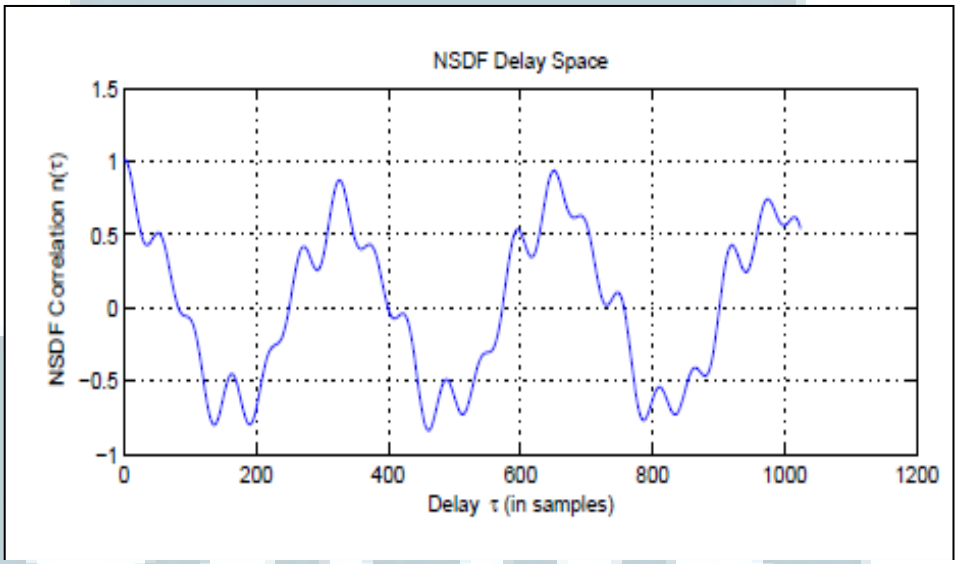
Setelah *square difference function* dihitung pada waktu t, pokok permasalahan adalah menentukan τ mana yang sesuai dengan nada, biasanya adalah *local minimum.* Namun menurut Mcleod, sulit untuk memutuskan *local minimum* tanpa adanya pemahaman tentang rentang nilai. Mcleod berhasil

mendefinisikan rentang nilai tersebut dengan melakukan normalisasi pada *square difference function.* Rumus 2.7 merupakan notasi matematika dari *square difference function* yang telah dinormalisasi (Broenink,2011).

$$n't(\tau) = 1 - \frac{m't(\tau) - 2r't(\tau)}{m't(\tau)}$$

$$n't(\tau) = \frac{2r't(\tau)}{m't(\tau)} \quad \text{...Rumus 2.7}$$

Hasil dari *square difference function* yang telah dinormalisasi adalah bilangan dengan rentang nilai antara -1 sampai dengan 1, karena berdasarkan rumus 2.6, disimpulkan bahwa panjang maksimal yang mungkin dari 2*rt*(τ) adalah m*t*(τ). Hasil 1 pada *square difference function* menunjukkan berkorelasi sempurna, 0 tidak berkorelasi, dan -1 berkorelasi negatif. Gambar 2.2 menunjukkan contoh hasil *square difference function.*



Gambar 2.2 Contoh Hasil *Normalized Square Function*
(sumber : Mcleod, 2008)

Setelah mendapatkan seluruh nilai korelasi sampai dengan nilai τ, akan dilakukan pencarian periode dengan mengggunakan algoritma *peak picking*. Pertama, semua *key maxima* (titik puncak dari suatu gelombang yang *zero-*

crossing) akan dicari. Berdasarkan contoh pada gambar 2.2, terdapat tiga buah *key maxima* yaitu pada τ 325, 650, dan 980. Selanjutnya, untuk mendapatkan nilai x dan y yang lebih akurat dari ketiga titik itu digunakan *Parabolic Interpolation*, yaitu persamaan yang membandingkan nilai dari titik tertinggi dengan nilai dari kedua titik yang bersebelahan. Dari *key maxima* tersebut, akan dipilih titik tertinggi (Nmax). Selanjutnya, akan dilakukan proses penentuan *threshold* yang bernilai Nmax * k (bilangan *real* antara 0.8 – 1.0). Setelah itu, akan dilakukan penentuan periode yang merupakan *key maxima* terdekat yang nilainya lebih tinggi dibandingkan *threshold.* Setelah mendapatkan periode, frekuensi dapat ditentukan dengan menggunakan rumus 2.8.

Frekuensi = Sample rate / periode ...Rumus 2.8

2.7 Parabolic Interpolation

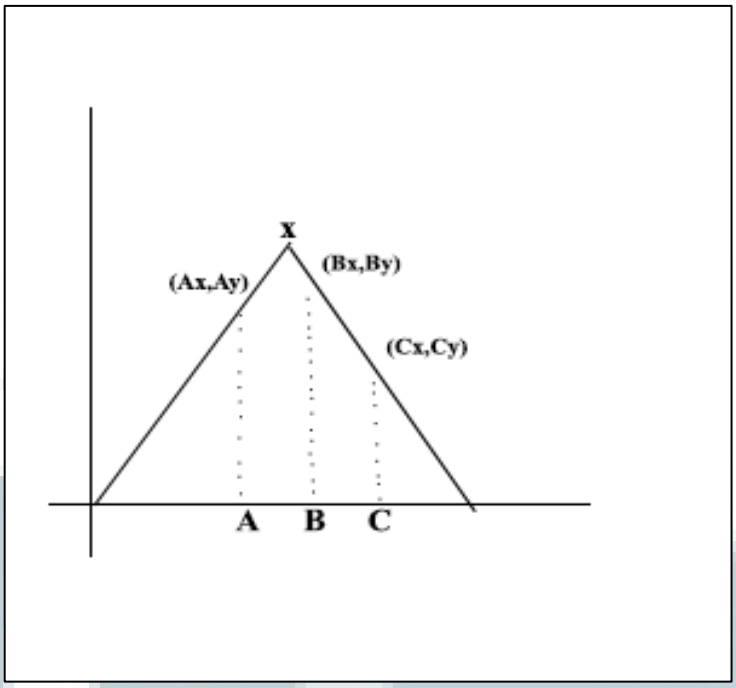
Parabolic interpolation merupakan suatu metode untuk mencari titik maksimum atau minimum dari suatu fungsi yang dijalankan dengan jarak yang sama. *Parabolic interpolation* membandingkan titik tertinggi dari fungsi tersebut dengan kedua titik yang bersebelahan. Awalnya *bottom* dihitung dengan rumus 2.9 (Mcleod,2008).

Bottom =Ay + Cy – 2By ...Rumus 2.9

Apabila *bottom* bernilai 0 , maka koordinat titik B telah tepat di tengah. Jika tidak, maka nilai x dan y dapat dicari dengan menggunakan rumus 2.10 dan 2.11 (Mcleod,2008).

x = Bx + (Ay – Cy) / (2 * *bottom*) ...Rumus 2.10

$y = By – (Ay-Cy)^2 / (8 * bottom)$ ...Rumus 2.11



Gambar 2.3 *Parabolic Interpolation*

Gambar 2.3 menunjukkan contoh dari *parabolic interpolation*. Tujuan dari *parabolic interpolation* adalah untuk mencari titik x, yang merupakan titik maksimum sebenarnya pada grafik ini. *Parabolic interpolation* dilakukan dengan cara membandingkan nilai dari titik maksimum awal dengan dengan kedua titik yang bersebelahan dengan titik maksimum awal tersebut (Mcleod,2008). Pada gambar 2.3, titik (Bx,By) merupakan titik maksimum awal yang didapat dari suatu fungsi yang dijalankan dengan jarak yang terlalu besar. Titik tersebut kemudian akan dibandingkan dengan kedua titik yang bersebelahan dengannya, yaitu titik (Ax,Ay) dan (Cx,Cy).