**Hak cipta dan penggunaan kembali:**

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

**Copyright and reuse:**

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

Team project ©2017
Dony Pratidana S. Hum | Bima Agus Setyawan S. IIP

# BAB III

# ANALISIS DAN PERANCANGAN SISTEM

## 3.1. Metode Penelitian

Metode penelitian yang dilakukan adalah metode perancangan aplikasi untuk membuat komposer *beat* musik berbasis *voice recorder* dengan fitur reduksi *noise* menggunakan transformasi *wavelet*. Pertama peneliti merancang algoritma dan *interface* dari aplikasi. Peneliti melakukan uji coba dengan mengambil sampel beberapa suara hasil rekaman. Setelah itu menentukan cara *denoising* dan jenis *wavelet* yang ingin digunakan.

Selain untuk membuat *beat* musik, aplikasi ini juga dapat menyimpan dan memuat *beat* yang sudah direkam, sehingga dapat dipakai untuk keperluan lainnya. Ada juga fitur lainnya yaitu pengaturan tempo, jumlah *loop* yang diinginkan, fitur rekam suara, pengaturan *sequence*, fungsi *save* untuk merekam *beat* musik, fungsi *load* untuk memuat *beat* musik yang sudah dibuat sebelumnya, dan fungsi *play* untuk memainkan *beat* musik secara langsung. Dalam penelitian ini juga dianalisis sinyal gelombang rekaman audio hingga menghasilkan gelombang sinyal rekaman audio baru yang mempunyai kualitas suara lebih baik. Selain itu penelitian ini dapat melakukan perbaikan kualitas rekaman audio dengan mengurangi *noise* yang meggunakan transformasi *wavelet*, sehingga dapat dianalisa jenis *wavelet* yang dapat mereduksi *noise* dengan baik. Analisa respon audio juga dilakukan melalui proses *equalizer*.



Pengembangan komposer...,Imanuel,FTI UMN,2017

## 3.2. Instrumen Penelitian

Perangkat – perangkat yang digunakan untuk melakukan penelitian ini adalah sebuah komputer dengan spesifikasi sebagai berikut:

Perangkat keras (Personal Computer - Laptop):

- *Motherboard*: ASUSTek COMPUTER INC. K46CB
- Prosesor Intel® Core™ i5-3337U CPU @1.80 GHz
- *Memory* 4GB DDR3
- *Hard Disk Drive* 750GB
- *Microphone*

Perangkat lunak:

- Windows 7 Professional 64-bit *Operating System*
- *Software* MATLAB versi R2016b

## 3.3. Perancangan Aplikasi *Beat Composer*

Perancangan aplikasi *voice recorder beat music composer* adalah sebagai berikut:

### 3.3.1. Perancangan Konsep Aplikasi

Aplikasi ini berguna untuk merancang suatu *beat* musik, dengan perekaman suara sebagai fitur utama. Langkah awal ialah perekaman suara dengan menggunakan *microphone*, kemudian menentukan *sequence* yang diinginkan. Satu *sequence* berjumlah 4 ketukan. Terdapat 4 *sequence* dalam satu baris perekaman, yang berarti ada 16 ketukan yang dapat dimainkan. Setiap ketukan dapat dipilih atau tidak tergantung keinginan *user*.



Pengembangan komposer...,Imanuel,FTI UMN,2017

Gambar 3.1. Rancangan *interface* salah satu baris *record* dan *sequence*

Rekaman suara terdiri dari beberapa suara yang berbeda-beda, sehingga ketika suara-suara tersebut sudah direkam maka akan disinkronisasikan secara otomatis ketika tombol *play* dijalankan. Program akan berjalan berdasarkan *sequence* yang telah ditentukan sebelumnya oleh *user*. *Beat* musik akan berjalan berdasarkan tiap ketukan dari kiri ke kanan. *Sequence* dipilih dengan menekan tombol ketukan yang diinginkan. Hasil suara pada tiap baris rekaman dijalankan bersamaan.

Hasil audio musik yang telah dirancang dapat langsung disimpan menjadi *file* .mat dan .wav. *File* .mat dapat memuat ulang informasi perekeman *beat* ke dalam *sequence*, sedangkan .wav tidak dapat memuat ulang perekaman *beat* ke dalam *sequence*. Tetapi selain itu, *user* juga dapat menggunakan *equalizer* untuk analisa respon audio. Fitur reduksi *noise* juga dapat dipilih sebagai opsi jika hasil audio yang dihasilkan memiliki *noise* yang cukup mengganggu.

### 3.3.2. Perancangan Fitur Aplikasi

Fitur aplikasi dirancang untuk menghasilkan *beat* musik yang optimal dan dapat digunakan untuk keperluan lainnya. Adapun beberapa fitur yang terdapat pada aplikasi.

- Tempo, merupakan ukuran kecepatan dalam irama lagu. Tempo



Pengembangan komposer...,Imanuel,FTI UMN,2017

mempengaruhi cepat atau lambatnya irama *sequence* yang sudah ditentukan. Tempo dapat diatur oleh *user* dengan parameter angka. Semakin kecil angka tempo, semakin lambat *beat*nya. Semakin besar angka tempo, semakin cepat *beat*nya. Satuan tempo yang diinput adalah *beats per minute* (bpm).

- *Loop*, menentukan berapa kali *sequence* ingin dimainkan. *User* dapat menentukan *loop* dengan parameter angka, dengan angka minimal *loop* yaitu 4 (sesuai dengan maksimal jumlah *sequence*) Fitur ini mendukung lama atau tidaknya audio dijalankan. Semakin tinggi nilai *loop* yang dimasukkan, semakin lama durasi sebuah *beat* musik. Sebaliknya, semakin kecil nilai *loop* yang dimasukkan maka durasi *beat* musik yang dimainkan semakin pendek.
- *Save* (.mat), untuk menyimpan *beat* yang sudah ditentukan. Hasil dari *beat* yang sudah dipilih disimpan dalam bentuk matriks (.mat), sehingga dapat dimuat ulang (fitur *load*).
- *Save* (.wav), untuk menyimpan audio dari *beat* yang telah ditentukan sehingga dapat langsung dimainkan di luar program utama. *File* .wav tidak dapat dimuat ulang melalui fitur *load* pada GUI *beat composer*. Selain itu *file* .wav berguna untuk analisa *denoising* menggunakan transformasi *wavelet*.
- *Load*, untuk memuat *beat* musik yang sudah disimpan sebelumnya. *Sequence* dan ketukan yang telah dibuat sebelumnya akan langsung termuat ulang. Fitur ini juga membantu *user* untuk mengubah ulang *beat* musik, sehingga ketika sudah tepat, dapat disimpan ulang



Pengembangan komposer...,Imanuel,FTI UMN,2017

melalui tombol *save*.

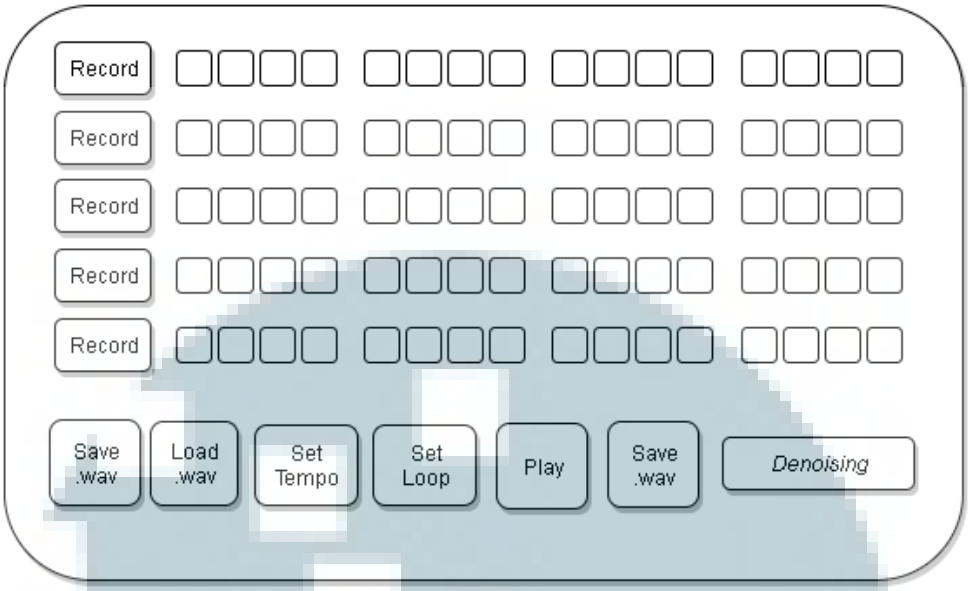
- *Play*, untuk memainkan secara langsung *beat* musik yang sudah ditentukan. Fitur ini dapat dijalankan langsung tanpa harus menyimpan *beat* musik terlebih dahulu, sehingga *user* dapat mendengarkan musik secara langsung.
- *Denoising*, fitur ini berguna untuk mereduksi *noise* dan menganalisa spektrum dari sebelum direduksi dan setelah direduksi. *File* yang dianalisa harus dengan format .wav. *File* audio yang dianalisa dapat ditambahkan *noise* secara manual dan dapat dipilih jenis-jenis *wavelet* untuk reduksi *noise*. Tampilan analisa ini tidak dijadikan satu dengan program *beat composer*.
- *Equalizer*, fitur ini untuk mengubah respon audio. Fungsi ini untuk mengontrol suara rendah sampai tinggi dengan cara mengatur *low pass filter*, *band pass filter*, dan *high pass filter* sehingga dapat dilihat karakteristik audio setelah meggunakan *equalizer*. Dalam penelitian ini, *equalizer* digunakan hanya untuk menganalisa hasil respon audio di domain frekuensi.

### 3.3.3. Perancangan *Graphical User Interface* (GUI) Aplikasi

Perancangan *interface* dilakukan di MATLAB dengan menggunakan editor GUI. Rancangan *interface voice recorder beat composer* seperti berikut.



Pengembangan komposer...,Imanuel,FTI UMN,2017




Gambar 3.2. Rancangan GUI Aplikasi *Beat Composer*

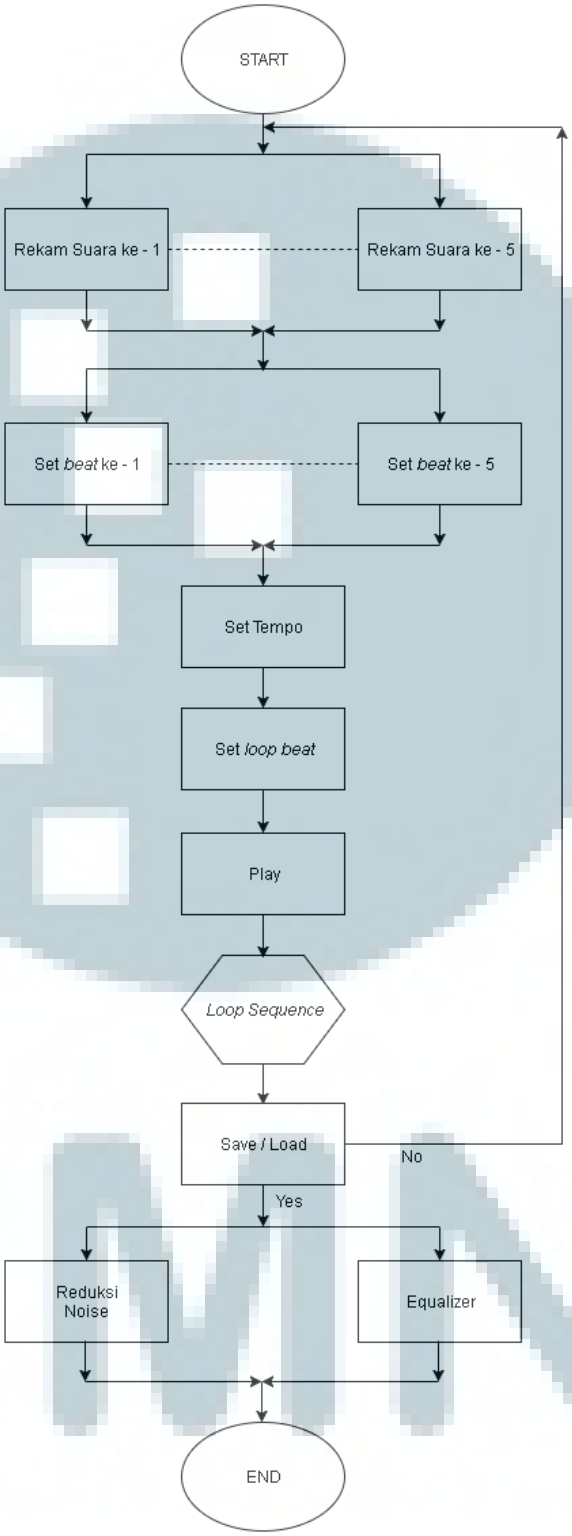
Tampilan GUI memiliki fitur – fitur yang telah dijelaskan sebelumnya. Fitur *denoising* tidak muncul pada tampilan yang sama, melainkan muncul di *window* lain. Tipe *action* pada setiap fitur yaitu,

Tabel 3.1 Tipe *Action* Fitur Aplikasi

| Fitur | Tipe *action* |
|---|---|
| *Record* | *Toogle button* |
| *Sequence* | *Toogle button* |
| *Save* .wav | *Push button* |
| *Load* .wav | *Push button* |
| *Set Tempo* | *Edit string* |
| *Set Loop* | *Edit string* |
| *Play* | *Push button* |
| *Save* .wav | *Push button* |
| *Denoising* | *Push button* |

### 3.3.4. Langkah Kerja Aplikasi

Cara kerja aplikasi *beat composer* dijelaskan lewat *flow chart diagram*.




Gambar 3.3 *Flow Chart Diagram* Aplikasi *Beat Composer*

Pertama, *user* membuka *file* program untuk membuka aplikasi di MATLAB. GUI akan muncul dan langkah awal *user* akan dihadapkan pada GUI pada Gambar 3.2. Langkah awal untuk menjalankan aplikasi adalah menentukan jumlah perekaman suara. Perekaman suara minimal satu kali dan untuk menghasilkan musik yang lebih bagus dilakukan minimal tiga sampai lima kali perekaman suara, dengan maksimal lima kali perekaman. Dalam program ini, hasil perekaman suara tergantung kualitas *microphone* dan kondisi lingkungan saat perekaman.

Kemudian *user* menentukan *beat* dari tiap *sequence* yang ditampilkan. Penentuan *beat* bebas dan tidak ada batasan tertentu. *User* dibebaskan untuk bereksplorasi sampai *beat* dirasa cukup bagus. Setiap *sequence* dimasukkan ke dalam *cell* matriks yang terdapat *row* dan *columns*. *Rows* berisikan nilai suara dari jumlah suara yang telah direkam, sedangkan *columns* berisikan nilai *sequence* yang dari jumlah *sequence* yang telah dipilih.

*User* dapat menentukan tempo dan jumlah *loop* yang diinginkan dengan menentukan angka tempo dan *loop*. Musik dapat didengarkan secara langsung melalui tombol *play*. Informasi pada GUI berguna untuk menampilkan apakah *user* sudah menentukan *beat* musik atau belom. *Beat* pada *sequence* akan melakukan *looping* secara terus menerus sampai batas maksimal input *loop* yang telah ditentukan. Setelah dirasa bagus, *user* dapat memilih tombol *save* untuk menyimpan *beat* musik yang telah diatur sebelumnya. Jika ingin memuatnya kembali dapat memilih fitur *load*. Output musik yang dihasilkan dalam bentuk *file* WAV, sehingga dapat dimainkan di luar aplikasi.



Pengembangan komposer...,Imanuel,FTI UMN,2017

Perancangan *code* algorima pada aplikasi ini dibagi menjadi 6 fungsi yang berbeda-beda untuk memudahkan dalam pengerjaan dan pengamatan. Fungsi-fungsi tersebut dapat dilihat pada Tabel 3.2.

Tabel 3.2. Fungsi dan Keterangan Aplikasi *beat composer*

| Fungsi | Keterangan |
|---|---|
| btmenuGUI | Merancang desain GUI, deklarasi variabel global, menentukan jumlah perekaman dan jumlah sequence, menentukan posisi window, memanggil 5 fungsi lainnya (loadbt, savebt, seqbt, recbt, play) |
| loadbt | Memuat ulang .mat dengan acuan nama dan *path file* dengan informasi *cell* matriks *sequence*, jumlah rekaman, tempo, dan *loops*. |
| savebt | Menyimpan .mat atau .wav. dengan acuan nama dan *path file* dengan informasi *cell* matriks *sequence*, jumlah rekaman, tempo, dan *loops*. |
| seqbt | Menentukan langkah *button sequence* ketika dipilih atau tidak dipilih oleh *user* |
| recbt | Menentukan kondisi ketika direkam atau tidak, menyimpan sementara hasil rekaman ke dalam variabel |
| play | Menjalankan *sequence*, tempo, dan *loops* dengan keluaran audio. |



Pengembangan komposer...,Imanuel,FTI UMN,2017

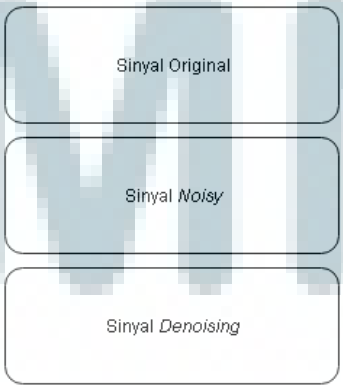
## 3.4. Perancangan *Denoising*

### 3.4.1. Perancangan Analisa

Perancangan analisa *denoising* dilakukan dengan membandingkan tiga buah sinyal yang berbeda yaitu,

1) Sinyal audio sebelum direduksi, yaitu sinyal audio original yang berasal dari hasil *beat composer*.
2) Sinyal audio yang ditambahkan *noise*. Audio original ditambahkan nilai SNR *noise* dari 5 dB sampai 20 dB untuk membandingkan *noise* paling rendah hingga *noise* paling tinggi.
3) Sinyal audio yang telah mengalami proses reduksi *noise*. Hasil dari audio yang telah direduksi karena proses transformasi jenis *wavelet* yang dipilih. Jenis *wavelet* yang dipilih yaitu Haar, Daubechies2, Daubechies6, Daubechies10, Symlets2, Symlets5, dan Symlets8.

Ketiga sinyal audio tersebut terdapat pada spektrum yang berbeda-beda yang dipilih oleh *user* dengan memuat *file* .wav. Dari hasil pengamatan *user* dapat melihat dan mengamati perbedaan ketiga sinyal tersebut.




Gambar 3.4. Tampilan Analisa Sinyal



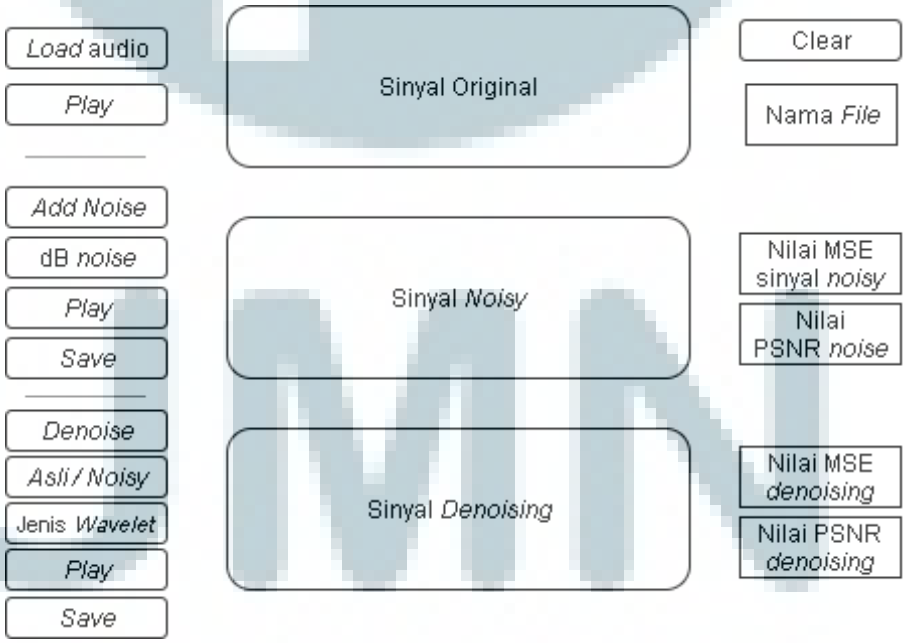
Pengembangan komposer...,Imanuel,FTI UMN,2017

Selain pada pengamatan sinyal, juga dilakukan analisa terhadap nilai *Mean Squared Error* (MSE) dan *Peak Signal-to-noise Ratio* (PSNR) sehingga memiliki analisa yang pasti untuk diamati. PSNR memiliki satuan dB.

Nilai MSE dan PSNR terdapat pada bagian sinyal yang ditambahkan *noise* dan sinyal dari hasil reduksi. Nilai MSE dan PSNR pada bagian *noise* membandingkan sinyal original dengan sinyal *noisy*, sedangkan nilai MSE dan PSNR pada bagian setelah direduksi, membandingkan sinyal original dengan sinyal setelah direduksi.

### 3.4.2. Perancangan Fitur & *Graphical User Interface* (GUI) Analisa

Perancangan *interface* dilakukan di MATLAB dengan menggunakan editor GUI. Rancangan analisa adalah sebagai berikut.




Gambar 3.5. Tampilan GUI Analisa *Denoising*



Pengembangan komposer...,Imanuel,FTI UMN,2017

GUI pada Gambar 3.5 merupakan perancangan analisa untu ketiga audio yang berbeda. Penjelasan dari tiap bagian audio pada GUI adalah sebagai berikut

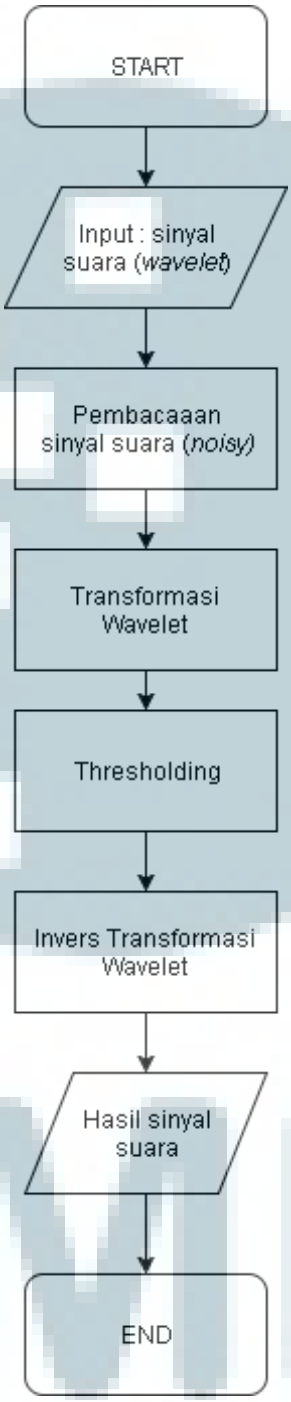
1) Tampilan sinyal original dapat dimuat dengan memilih *file* .wav yang berasal dari hasil *beat composer*. User dapat mendengar audio terlebih dahulu untuk mengecek hasil suara. Nama *file* yang dimuat akan terlihat di bagian kanan atas
2) Tampilan sinyal *noisy* melambangkan sinyal original yang disisipkan *noise*. Nilai *noise* dapat dipilih dalam *action popup*, yaitu mulai dari 5 dB, 10 dB, 15 dB, 20 dB, yang merupakan nilai *Signal-to-noise Ratio* (SNR). *Noise* akan langsung ditambahkan dan dimunculkan ketika *user* memilih tombol *Add Noise.* Audio hasil penambahan *noise* dapat didengarkan (tombol *Play*) dan disimpan (tombol *Save*). Nilai MSE dan PSNR akan langsung muncul ketika *noise* ditambahkan.
3) Tampilan sinyal reduksi *noise* menampilkan sinyal setelah *noise* direduksi dari sinyal original. Hasil sinyal reduksi berbeda-beda tergantung dari pemilihan jenis *wavelet*. Terdapat 6 jenis *wavelet* yang dapat dipilih. Hasil sinyal reduksi akan langsung muncul ketika *user* memilih tombol *Denoise*. *User* dapat memilih mereduksi *noise* dari sinyal original atau dari sinyal yang ditambahkan *noise* secara manual. Audio hasil reduksi dapat didengarkan (tombol *Play*) dan disimpan (tombol *Save*). Nilai MSE dan PSNR akan langsung muncul ketika *denoising* dipilih.



Pengembangan komposer...,Imanuel,FTI UMN,2017

### 3.4.3. Langkah Kerja Aplikasi

Cara kerja analisa *denoising* dijelaskan lewat *flow chart diagram.*




Gambar 3.6. *Flow chart denoising* dengan menggunakan Transformasi *Wavelet*



Pengembangan komposer...,Imanuel,FTI UMN,2017

Sinyal suara yang dimuat pada saat analisa dapat mengalami adanya kebisingan kecil maupun besar. Penambahan *noise* secara manual yang dapat diatur dapat mempermudah analisa oleh *user*. Nilai *noise* dapat dipilih, yaitu mulai dari 5 dB, 10 dB, 15 dB, 20 dB. Nilai – nilai ini merupakan *Signal-to-noise Ratio* (SNR), sehingga semakin besar angka SNR *noise* maka semakin kecil tingkat kebisingannya, sebaliknya semakin kecil angka SNR *noise* maka semakin besar tingkat kebisingannya.

Dalam reduksi *noise*, langkah pertama adalah input sinyal suara dari audio yang telah ditambahkan *noise*. Setelah itu dilakukan pembacaan *file* suara. Kemudian dilakukan proses transformasi *wavelet* diskrit pada matriks dan pengubahan sinyal suara ke koefisien *wavelet* untuk dapat dilakukan proses selanjutnya. Proses reduksi *noise* melibatkan *threshold* yang artinya melewatkan koefisien tersebut ke suatu ambang batas yang telah ditentukan, sehingga koefisien yang tidak sesuai ambang tersebut tidak dipakai. Sinyal suara yang yang terdapat *noise* akan berkurang *noise*nya dari sinyal suara sebelumnya. Setelah itu langkah terakhir adalah invers transformasi *wavelet* yang bertujuan untuk mengubah koefisien *wavelet* yang telah di *treshold* ke sinyal asalnya. Transformasi *wavelet* ini berperan sehingga sinyal dapat dibaca sebagai sinyal suara yang dapat di*plot* juga dapat didengarkan dengan *output* .wav. Blok diagram reduksi *noise* dapat dilihat pada Gambar 3.7.

Gambar 3.7. Blok Diagram Reduksi *Noise*



Pengembangan komposer...,Imanuel,FTI UMN,2017

Langkah-langkah *denoising* lebih lengkapnya dalam transformasi *wavelet* seperti yang ditunjukkan pada *flow chart diagram* pada Gambar 3.6 dalam penelitian ini adalah sebagai berikut:

1. *Load* sinyal suara yang original.
2. *Noise* ditambahkan ke sinyal suara original menggunakan White Gaussian *noise* dan menghasilkan sinyal suara yang memiliki *noise*. Jika sinyal suara original ingin langsung dilakukan proses *denoising*, maka tidak terjadi penambahan *noise* secara manual.
3. Dekomposisi dilakukan menggunakan transformasi *wavelet*.
4. Pemakaian jenis – jenis *wavelet*.
5. Penerapan *thresholding* ke dalam koefisien *noise*.
6. Rekonstruksi menggunakan transformasi *wavelet* balik
7. Mendapatkan sinyal suara yang telah direduksi dan parameter yang berbeda.

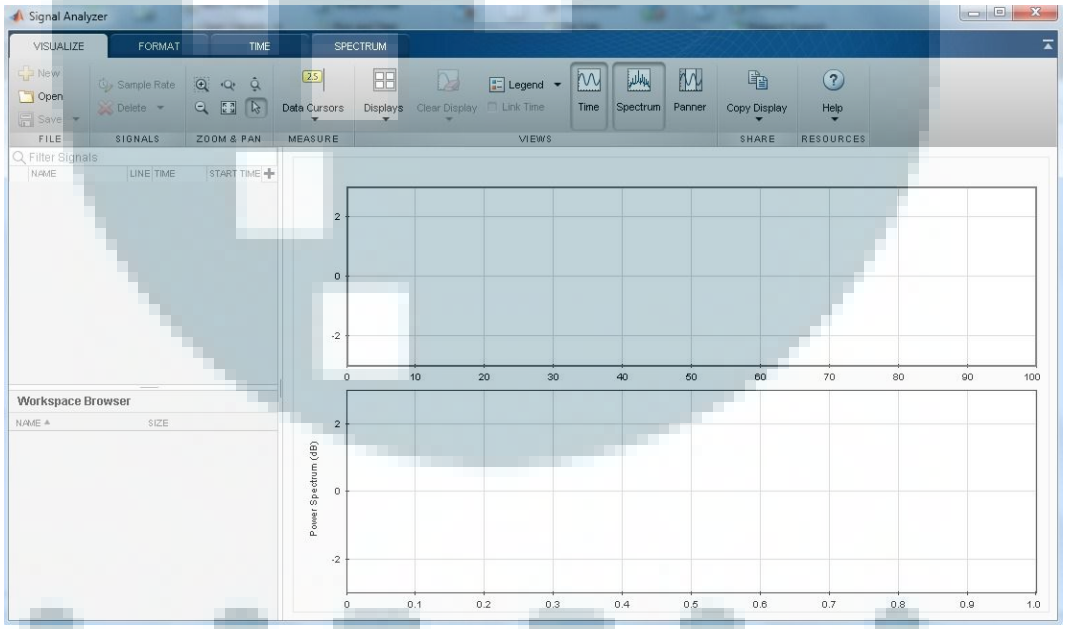
Hasil dari sinyal reduksi *noise* dianalisa dari setiap jenis *wavelet* dan setiap nilai SNR *noise* yang dipilih, sehingga dilakukan analisa pada satu audio dan analisa secara berulang-ulang sehingga didapatkan perbandingan dan nilai MSE serta PSNR. Perbandingan ini yang dapat ditarik kesimpulan jenis *wavelet* apa yang paling baik dipilih untuk mereduksi *noise*. Hasil audio juga didengarkan oleh peneliti untuk memeriksa *noise* sudah berkurang atau belum.

Analisa untuk mengetahui sinyal suara dalam frekuensi domain setelah proses *denoising* dan *equalizer* dapat dilihat melalui *Signal Analyzer*



Pengembangan komposer...,Imanuel,FTI UMN,2017

pada Matlab. *Equalizer* tidak dibuat oleh penulis dan memakai program *equalizer* yang sudah ada, sehingga hanya mengamati respon sinyal suara setelah diproses melalui *equalizer*, dengan mengubah filter yaitu, *low pass filter*, *band pass filter*, dan *high pass filter*. Hasil dari perubahan di *equalizer* dapat menghasilkan output .wav yang berbeda dari sebelumnya, sehingga dapat di analisa *file* sebelum diproses melalui *signal analyzer* dan *file* setelah diproses melalui *signal analyzer*. Signal *Analyzer* dapat melihat sinyal suara dalam domain waktu dan frekuensi dengan mengambil variabel data suara pada *file* audio .wav.




Gambar 3.8. Tampilan *Signal Analyzer*