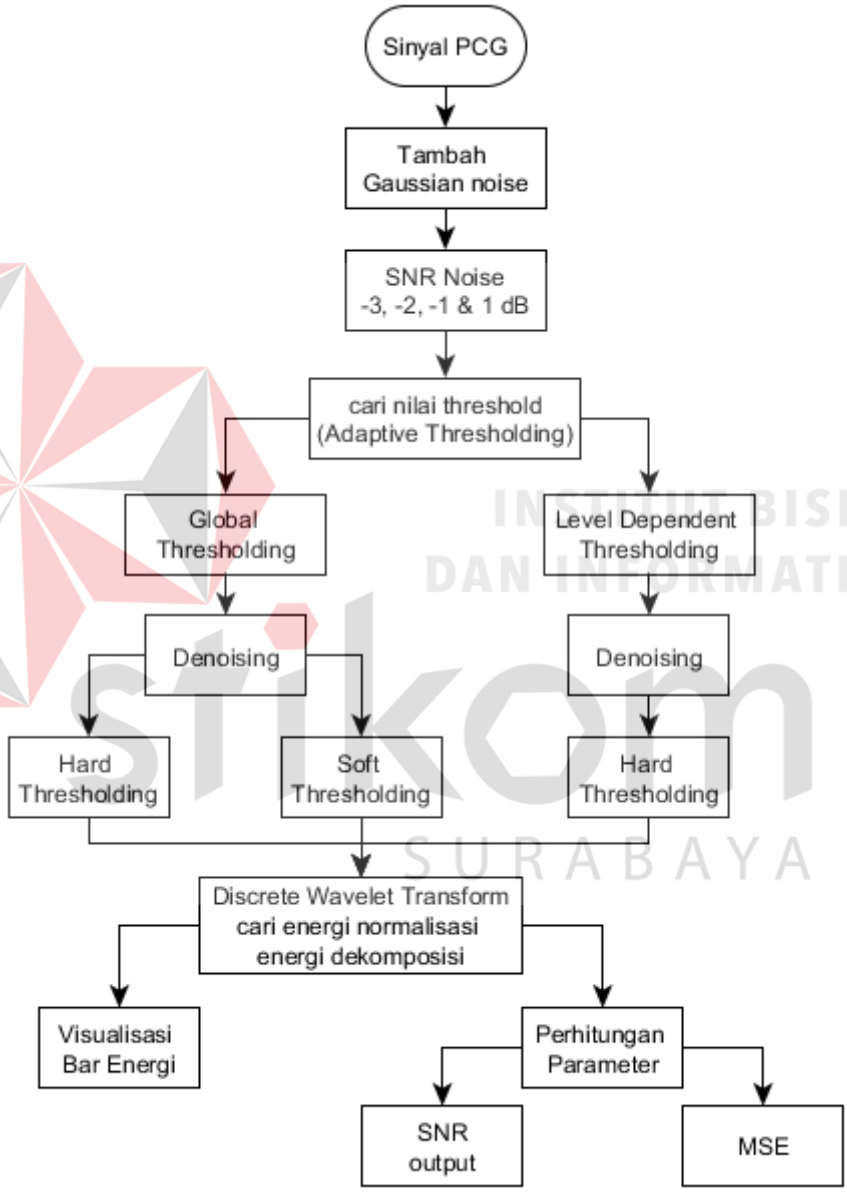
BAB III

METODE PENELITIAN DAN PERANCANGAN SISTEM

3.1 Metode Penelitian



Gambar 3.1 Diagram Blok Rancangan Penelitian.

Metode penelitian yang digunakan meliputi studi kepustakaan, pembuatan program, dan analisis. Studi kepustakaan dilakukan untuk mencari teori atau informasi dari buku, jurnal, dan artikel-artikel yang berkaitan dengan *Denoising*, *Signal to Noise ratio* (SNR), *Mean Square Error* (MSE), dekomposisi sinyal PCG, Dari informasi studi kepustakaan yang diperoleh, maka dilakukan pembuatan program pada matlab untuk membantu analisis.

Penelitian ini menggunakan data *database* yang didapatkan dari Michigan University (‘*normal heart sound*’) licensi yang diberikan berupa *freeware*. Berdasarkan blok diagram pada Gambar 3.1, dijelaskan bahwa sinyal PCG yang masuk akan di tambahkan *Gaussian noise* sebesar -3,-2,-1 dan 1 dB, kemudian mencari nilai *threshold* secara adaptif dengan menerapkan *threshold rules*. Terdapat 2 metode *threshold rules* yang digunakan yaitu *global thresholding* dan *level dependent thresholding*. Proses selanjutnya setelah nilai *threshold* didapatkan akan melakukan proses *Denoising* untuk menghapus sinyal yang tidak diperlukan dari sinyal suara jantung (yang dianggap sebagai *noise*). Pada penelitian ini proses *global thresholding* menggunakan metode *denoising hard* dan *soft thresholding* sedangkan proses *level dependent thresholding* menggunakan metode *denoising hard thresholding*. Setelah sinyal yang tidak diperlukan dihapus maka akan di dekomposisi kedalam bentuk gelombang yang disebut *Disccrete Wavelet Transform*, yang mana sinyal tersebut akan di pecah menjadi sinyal yang berfrekuensi tinggi (*aproksimasi*) dan sinyal yang berfrekuensi rendah (*detail*). Parameter yang digunakan pada proses ini antara lain sinyal PCG dari subyek, frekuensi sampling, *Mother Wavelet*, dan tingkat

dekomposisi. Dari hasil dekomposisi akan didapatkan beberapa sinyal *detail* dan sinyal *aproksimasi* yang terakhir sehingga akan dihitung energi dekomposisi yang telah dinormalisasi, setelah didapatkan energi normalisasi proses selanjutnya memvisualisasikanya dalam bentuk grafik bar. Hasil perhitungan parameter SNR dan MSE akan dianalisis untuk menguji seberapa besar keberhasilan dari *denoising* dengan *adaptive thresholding*.

3.1.1 Sinyal PCG

Sinyal PCG merupakan sinyal suara jantung yang diambil dari *database* Michigan University, licensi yang diberikan adalah *freeware* atau gratis. Sinyal suara jantung yang digunakan merupakan sinyal yang bersih dari *noise*. Penggunaan sinyal bersih ini digunakan untuk pembanding keberhasilan proses *denoising* antara sinyal yang bersih yang ditambahkan *noise* dan sinyal yang telah di*denoising*.

3.1.2 *Noise*

Data sinyal suara jantung persiklus diambil dari Michigan University merupakan data yang tanpa *noise*, kemudian melakukan penambahan *noise* ke sinyal jantung berupa *guassian noise* sebesar -3, -2 , -1 dan 1 dB. Pembangkitan *Gaussian noise* ini dengan menggunakan fungsi ‘randn’ pada matlab. Contoh program pembangkitan *Gaussian noise* bisa dilihat pada gambar 3.2.

```
3 -     db = x;%input SNR
4 -     psignal = var(signal);
5 -     pnoise = (psignal/(10^(db/10)));
6 -     noise=pnoise*randn(size(signal));
7       %tambah sinyal+noise
8 -     sinyalnoise = signal+noise;
```

Gambar 3.2 Cuplikan Program pembangkitan *Gaussian noise*

Proses penambahan *noise* ini digunakan untuk melanjutkan ke tahapan pencarian nilai *threshold* secara adaptif dan *denoising* sinyal yang telah tercampur oleh *Gaussian noise* dengan nilai SNR *noise* yang berbeda beda.

3.1.3 *Adaptive Thresholding*

Pada penelitian yang dilakukan mencari nilai *threshold* dengan karakteristik sinyal input dengan penerapan *thresholding rules*, terdapat 2 metode yaitu *global thresholding* dan *level dependent thresholding*. Pencarian nilai *threshold* dengan *global thresholding* menggunakan karakteristik dari panjang data dari sinyal input. Sedangkan *level dependent thresholding* mencari nilai *threshold* berdasarkan level resolusi / level dekomposisi. Pada penelitian ini level resolusi yang digunakan adalah 10 level dikarenakan sinyal yang diolah memiliki frekuensi sampling sebesar 8KHz. Pada penelitian ini penggunaan metode *level dependent thresholding* digunakan *function* pada matlab. Cuplikan program proses pemanggilan *function* dapat dilihat pada Gambar 3.3.

```
13      %THRESHOLD RULES
14      %'rigrsure' uses the principle of Stein's Unbiased Risk.
15      %'heursure' is an heuristic variant of the first option.
16      %'sqtwolog' for the universal threshold sqrt(2*log(length(X)))
17      %'minimaxi' for minimax thresholding
18 -    thrules = 'rigrsure';

16 -    xd = wden(sinyalnoise,thrules,thrmetods,'sln',level,w);
```

Gambar 3.3 *function level dependent thresholding*

Pada matlab terdapat *function* yang digunakan untuk memanggil *level dependent thresholding* yaitu ‘wden’. Pada function ini memiliki parameter input sinyalnoise untuk sinyal masukan, thrules untuk metode *level dependent thresholding*, level untuk tingkatan level, dan ‘w’ untuk Mother Wavelet. Penggunaan *function* ini telah mengacu pada rumus *level dependent thresholding* menggunakan prinsip *Stein Unbiased Risk Estimator* (SURE) pada suatu *level* resolusi.

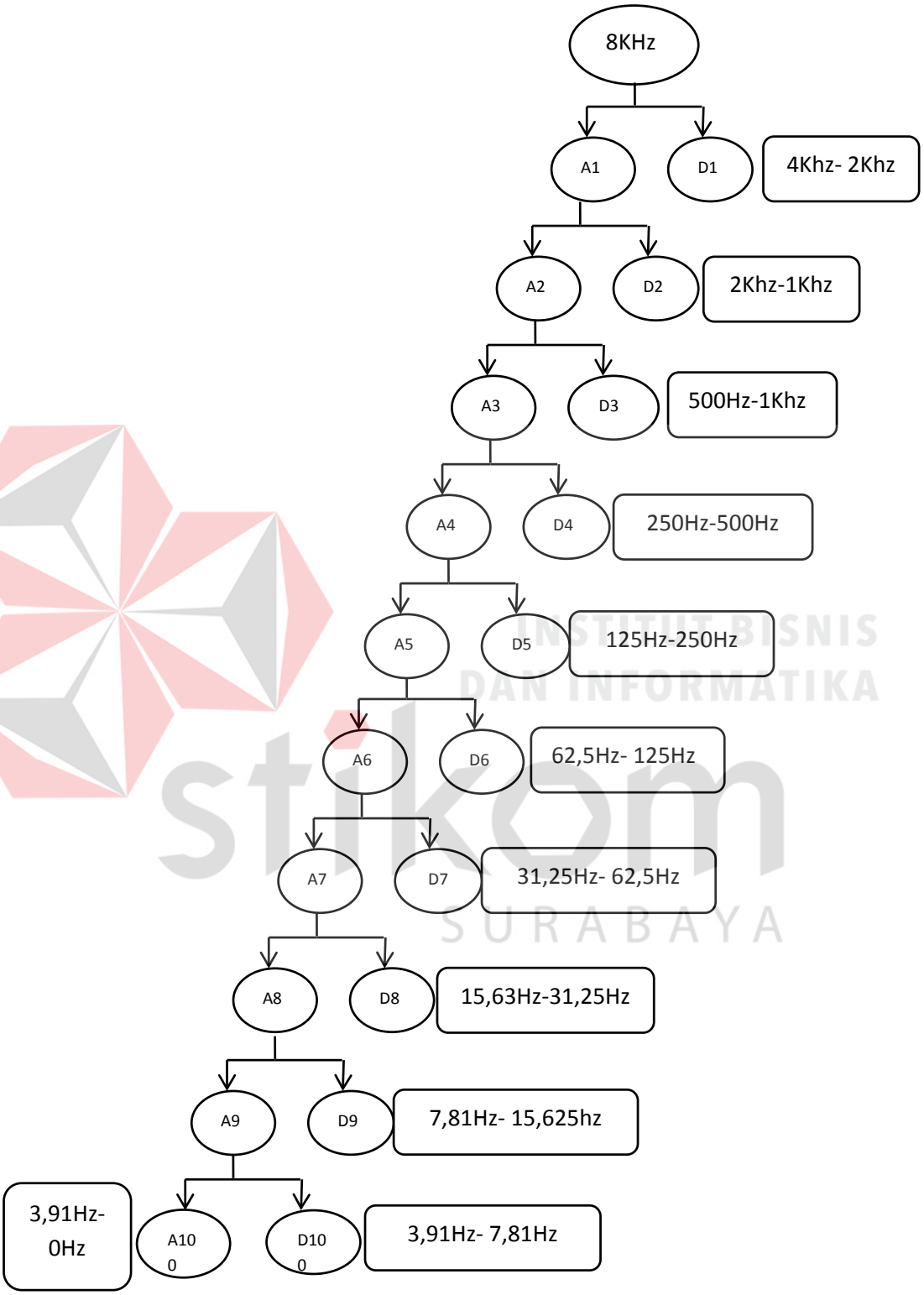
3.1.4 *Denoising*

Data sinyal PCG yang telah tercampur *noise* akan dicari nilai *threshold* dengan metode *threshold rules* yang ada. Nilai *threshold* digunakan untuk perbandingan pada setiap koefisien *wavelet*. *Denoising* digunakan untuk menghapus data sinyal yang tidak diperlukan dengan membandingkan nilai *threshold* dan setiap koefisen *wavelet*, yang telah ditambaakan *guassian noise*. Setelah itu proses *denoising* menggunakan metode *soft thresholding* atau *hard thresholding*. Metode *Denoising* yang digunakan adalah *soft tresholding*, dimana metode ini akan membuat nilai yang berada antara *threshold* –T< X < T menjadi perlahan menuju 0, sedangkan nilai yang lebih dari T telah diubah untuk mendekati axis X. Sedangkan pada metode *hard thresholding* membuat nilai

yang berada antara *threshold* –T< X < T menjadi langsung menuju 0. *Denoising* pada penelitian ini dilakukan secara *adaptive* karena *threshold* yang didapatkan dari karakteristik dari sinyal input.

3.1.5 *Discrete Wavelet Transform*

Discrete wavelet transform digunakan untuk mendekomposisikan sinyal masukan PCG ke dalam bentuk gelombang seusai dengan *Mother Wavelet* yang digunakan, dekomposisi dilakukan dengan memisahkan sinyal masukan ke dalam frekuensi rendah dan frekuensi tinggi, hasil dari dekomposisi adalah komponen *approximation* yang merupakan *scaling function* (*lowpass filter*) dan komponen *detail* yang merupakan *Wavelet function* (Sundararajan, 2015). Level dekomposisi ditetapkan berdasarkan frekuensi sampling yang digunakan. (Venkatta, 2015). Penelitian ini dipengaruhi beberapa parameter yaitu sinyal PCG dari setiap subyek, frekuensi sampling, *Mother Wavelet*, dan level dekomposisi. Sinyal PCG akan didekomposisikan menjadi A yang merupakan *approksimasi* dan D yang merupakan *detail*, serta akan didekomposisikan sesuai dengan frekuensi sampling 8Khz akan didekomposisikan sebanyak 10 tingkat yang dapat dilihat pada Gambar 3.4.



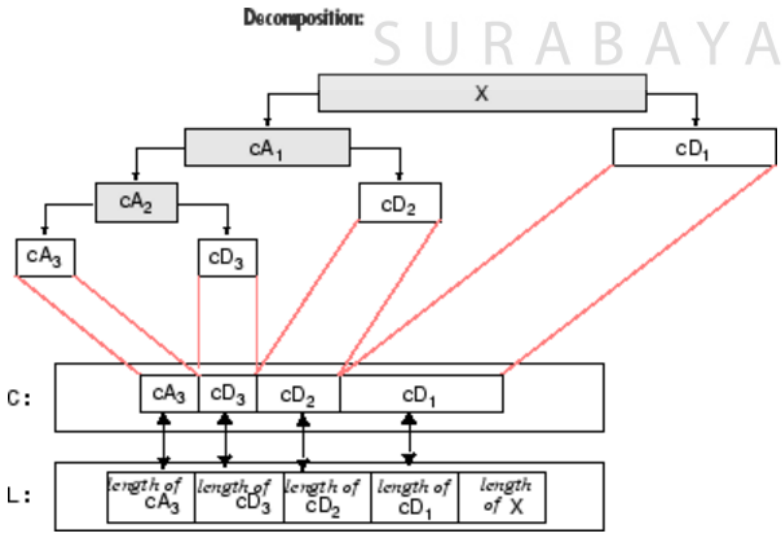
Gambar 3.4 Dekomposisi 10 Tingkat Dengan Frekuensi Sampling 8khz.

Analisis transformasi *Wavelet* diskrit dilakukan dengan mendekomposisi sinyal PCG menggunakan Matlab, untuk mendekomposisi sinyal satu dimensi maka digunakan fungsi `wavedec`, cuplikan program proses dekomposisi dapat dilihat pada Gambar 3.5.

```
%baca sinyal
sinyal = audioread('nfile.wav');
%jenis wavelet
w = 'db5';
%tingkatan level
L = 10;
%dekomposisi
[C,L] = wavedec(sinyal,L,w);
```

Gambar 3.5 Cuplikan Program Proses Dekomposisi.

Fungsi `wavedec('x',N,'Wname')` pada matlab memiliki parameter input x untuk sinyal masukan, N untuk tingkat level, dan Wname untuk *Mother Wavelet*, sedangkan parameter outputnya adalah hasil dekomposisi dan panjang data dari setiap komponen dapat dilihat pada Gambar 3.6.



Gambar 3.6 Dekompisi *Wavelet* Diskrit 1D. (Matlab)

3.1.6 Energi Dekomposisi dan Normalisasi Energi Dekomposisi

Energi Dekomposisi digunakan untuk mengetahui ciri atau pola sinyal PCG dengan yang lainya, pada penelitian ini digunakan dekomposisi 10 level untuk 8Khz. Berdasarkan level maka dapat dihitung energi dekomposisinya pada setiap komponen detail dan approksimasi terakhir. Energi dekomposisi rerata pada sinyal detail dapat dihitung dengan persamaan sebagai berikut:

- Energi dekomposisi rerata level 10

$E_{Di}= \frac{\sum(Di(k))^2}{jumlah\ cuplik\ Di}$, K= 1,2,……. Panjang D_i (3.1)

i = 1,2,…. N=10

- Energi dekomposisi rerata level 15

$E_{Di}= \frac{\sum(Di(k))^2}{jumlah\ cuplik\ Di}$, K= 1,2,……. Panjang D_i (3.2)

i = 1,2,…. N=15

Energi dekomposisi rerata sinyal aproksimasi dihitung dengan persamaan sebagi berikut:

- Energi dekomposisi rerata level 10

$E_{A10}= \frac{\sum(A_{10}(k))^2}{jumlah\ cuplik\ A_{10}}$, K= 1,2,…….Jumlah cuplik A_{10} (3.3)

Setelah energi didekomposisi rerata dihitung maka akan dilakukan normalisasi energi agar nilai energi berada diantara nilai 0 dan 1. Energi normalisasi dihitung dengan persamaan sebagai berikut:

- Normalisasi Energi dekomposisi level 10

$$EN_j = \frac{E_{Di}}{maks(E_{Di},\ E_{A\,10})} \quad (3.4)$$

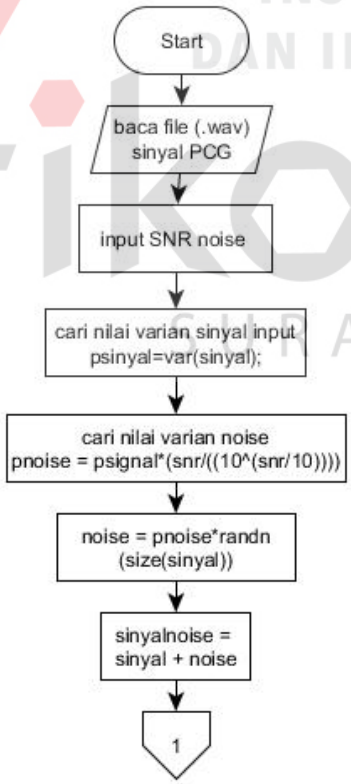
EN_j = Energi rerata normalisasi pada dekomposisi ke –j (j= 1,2,3…N=10)

E_{Di} = Energi rerata sinyal detail ke- I (i= 1,2,3….N=10)

E_{A10} = Energi rerata sinyal aproksimasi A_{10}

3.2 Flowchart Program pembangkitan *Guassian Noise*

Flowchart program untuk membangkitkan *guassian noise* kemudian di tambahkan pada sinyal tanpa *noise*, sebagai berikut :



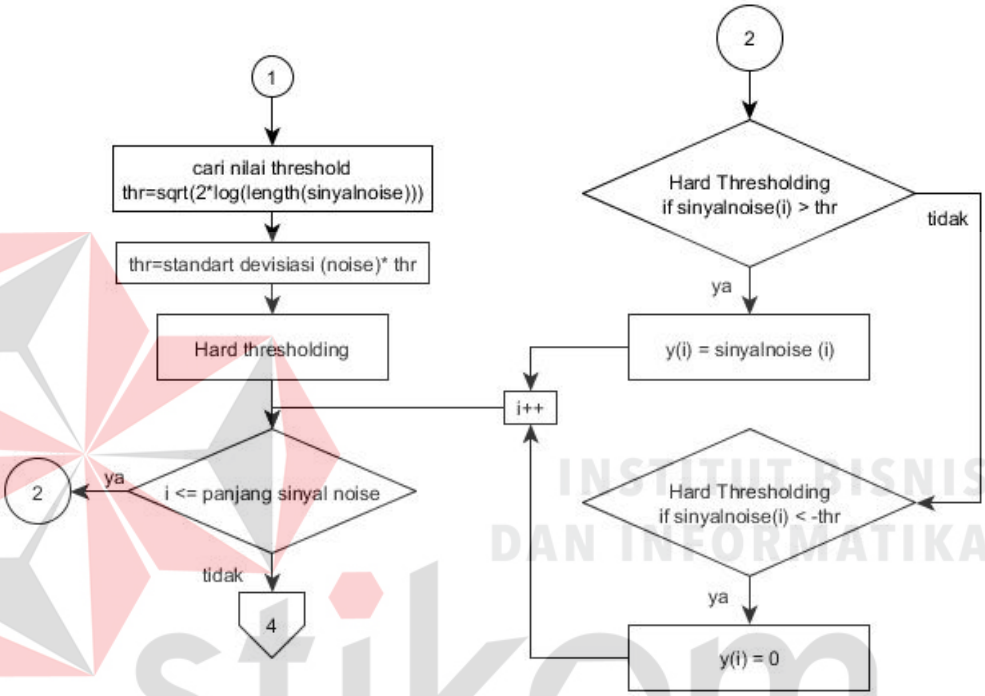
Gambar 3.7 *Flowchart* Program pembangkitan *guassian noise* dan penambahan *noise* ke sinyal tanpa *noise*

Gambar 3.7 merupakan gambar *flowchart* dari program pembangkitan guassian noise dan penambahan noise ke sinyal tanpa noise yang akan dijelaskan sebagai berikut :

1. Membaca file input berupa audio dari sinyal suara jantung (.wav)
2. Menginputkan SNR *noise*, di gunakan untuk membangkitkan *guassian noise*.
3. Menghitung varian dari sinyal tanpa noise
4. Mencari nilai varian dari noise dengan menggunakan rumus SNR yang telah di anti Log kan.
5. Membangkitkan Guassian dengan mengkalikan varian dari noise dengan fungsi randn (size(sinyal)). Size sinyal digunakan untuk menyamakan panjang noise dangan panjang sinyal tanpa noise.
6. Menjumlahkan sinyal tanpa noise dan sinyal noise.
7. Sinyal dan noise berhasil di campur.

3.3 Flowchart Program *denoising* dengan metode *Hard Thresholding*

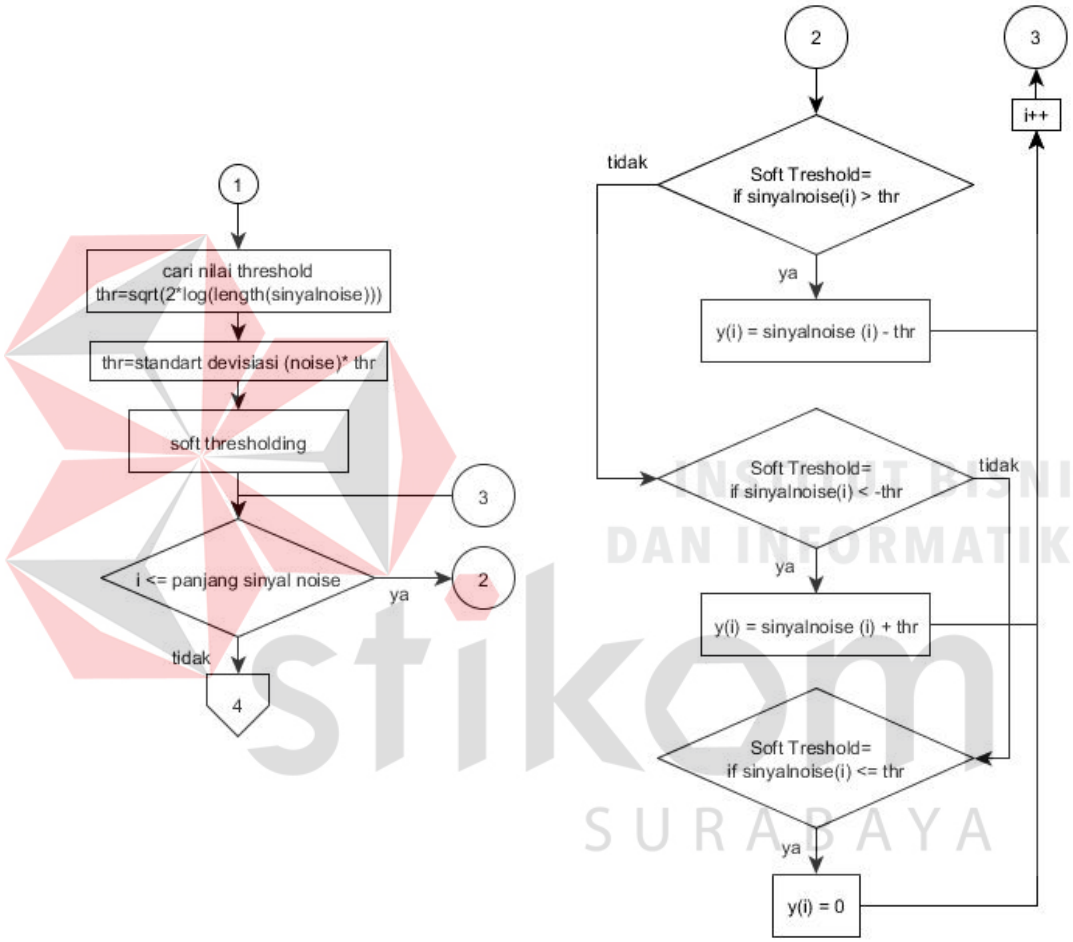
Flowchart program untuk mencari nilai *threshold* kemudian di *denoising* dengan metode *hard thresholding*, sebagai berikut :



Gambar 3.8 *Flowchart* Program Perhitungan nilai *threshold* dan *denoising* menggunakan *hard thresholding* (Lanjutan Gambar 3.7).

3.4 Flowchart Program *denoising* dengan metode *Soft Thresholding*

Flowchart program untuk mencari nilai *threshold* kemudian di *denoising* dengan metode *soft thresholding*, sebagai berikut :

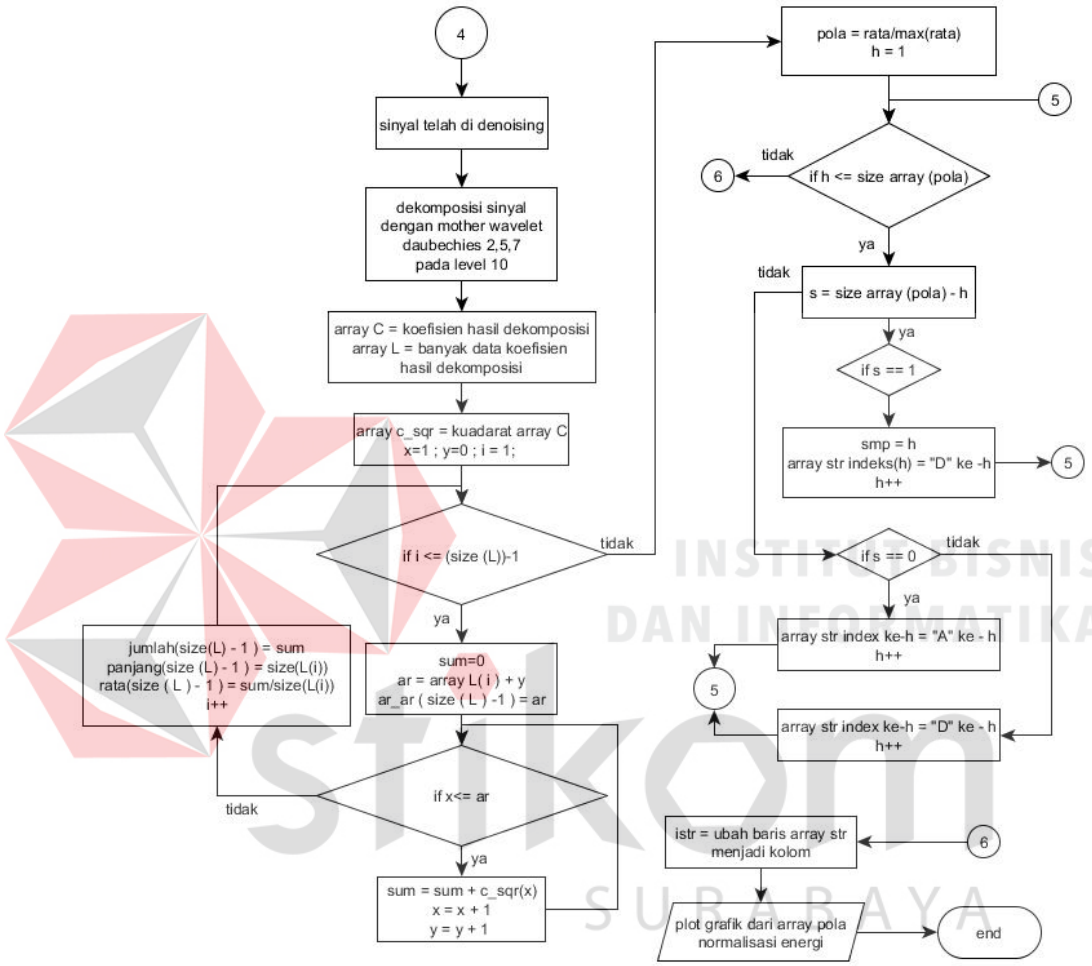


Gambar 3.9 *Flowchart* Program Perhitungan nilai *threshold* dan *denoising* menggunakan *soft thresholding* (Lanjutan Gambar 3.8).

Gambar 3.8 dan gambar 3.9 merupakan gambar *flowchart* dari program Perhitungan nilai *threshold* dan *denoising* menggunakan *soft* dan *hard thresholding* yang akan dijelaskan sebagai berikut:

1. Mencari nilai *threshold* dengan metode *global thresholding* dan level *dependent thresholding.*
2. Mencari *standart devisiasi* / nilai estimator dari *noise*
3. Mengkalikan hasil *threshold* dengan hasil *standart devisiasi* dari *noise*
4. Nilai *threshold* telah di dapatkan
5. Melakukan perulangan sepanjang data dari sinyal
6. Melakukan pembandingan nilai *threshold* dengan koefisien setiap *wavelet*.
7. Jika *noised signal* lebih dari *threshold* maka signal akan dirubah perlahan menuju ke 0 untuk *soft thresholding*.
8. Jika *noised signal* lebih dari *threshold* maka signal akan dirubah langsung menuju ke 0 untuk *hard thresholding*.
9. Jika *noised signal* kurang dari nilai *threshold* maka sinyal dibiarkan karna tidak di sebagai anggap *noise*.

3.5 *Flowchart* Program Energi Dekomposisi dan Normalisasi Energi



Gambar 3.10 *Flowchart* Program Perhitungan Energi Dekomposisi dan Normalisasi Energi (Lanjutan Gambar 3.7 dan gambar 3.8 atau gambar 3.9).

Gambar 3.10 merupakan gambar *flowchart* dari program perhitungan energi dan normalisasi energi yang akan dijelaskan sebagai berikut :

1. Program membaca data rekaman PCG dengan format audio.

2. Program menyimpan data rekaman ke dalam bentuk array dan mendekomposisikanya dengan transformasi *Wavelet* descrete menggunkanan berbagai *Mother Wavelet* dan berbagai orde ke dalam bentuk data-data integer yang tersimpan pada array C, selain itu hasil dekomposisi juga menyimpan banyak data hasil yang disimpan pada array L.

3. Semua elemen pada array C di kuadratkan.

4. Pemecahan data pada array C yang di sesuaikan dengan nilai dari setiap index dari array L yang merupakan array untuk menampung banyak data pada setiap detail dan aproksimasi hasil dekomposisi.

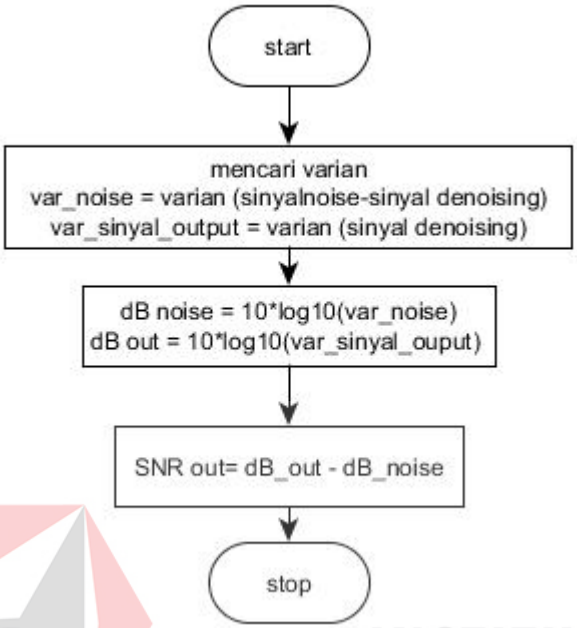
5. Data yang sudah dipecah pada array C dijumlahkan sesuai dengan tingkat detail dan aproksimasinya untuk mendapatkan nilai energi pada setiap tingkatan detail dan aproksimasi.

6. Data pada array C yang telah di jumlah usesuai dengan tingkat detail dan aproksimasi untuk mendapatkan nilai energi akan di hitung nilai rata-ratanya.

7. Menghitung normalisasi energi dari setiap aproksimasi dan detail dengan membagi nilai rata-rata energi dari setiap tingkatan detail dan aproksimasi dengan nilai rata-rata yang terbesar lalu hasilnya akan disimpan pada array pola.

8. Hasil dari normalisasi energi di visualisasikan dengan grafik bar.

3.6 Flowchart Program perhitungan nilai *Signal to Noise Ratio* (SNR)



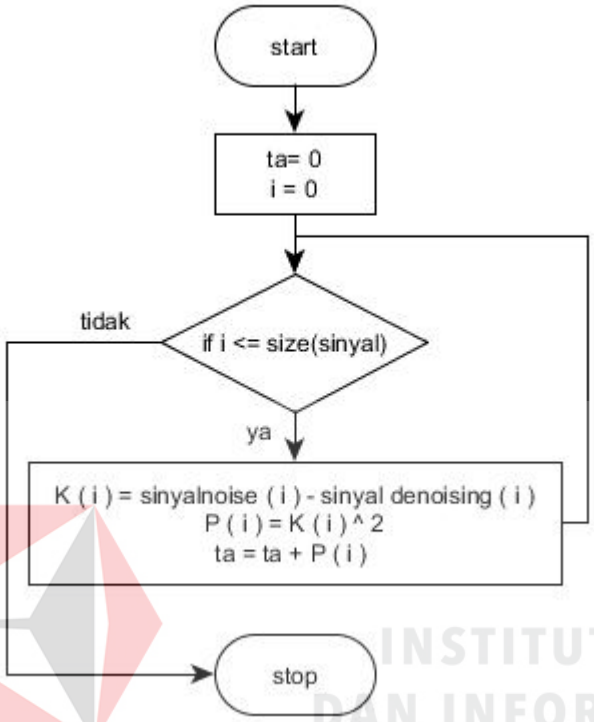
Gambar 3.11 *Flowchart* program untuk mencari nilai SNR

Gambar 3.11 merupakan gambar *flowchart* dari program Perhitungan nilai *Signal to Noise ratio* (SNR) yang akan dijelaskan sebagai berikut:

1. Mencari varian dari noise dan varian dari sinyal yang telah di denoising
2. Mengkalikan rumus SNR dengan varian dari noise dan varian sinyal yang telah di denoising.
3. Mengkurangi hasil SNR sinyal output dengan SNR sinyal noise.

Nilai SNR telah di dapatkan.

3.7 Flowchart Program perhitungan nilai *Mean Square Error* (MSE)



Gambar 3.12 *Flowchart* program untuk mencari nilai MSE

Gambar 3.12 merupakan gambar *flowchart* dari program Perhitungan nilai *Mean Square Error* (MSE) yang akan dijelaskan sebagai berikut :

1. Inisialisasi ta = 0 dan i = 0;
2. Melakukan perulangan selama i kurang dari panjang data sinyal.
3. Mengkurangi array isi array sinyalnoise (indeks ke-i) dengan isi array sinyal denoising (indeks ke-i).
4. Mengkuadratkan hasil pengurangan isi dari tiap array.
5. Menjumlahkan semua hasil kuadrat.
6. Mendapatkan nilai hasil MSE.