BAB 3
METODE PENELITIAN

3.1 ALAT DAN BAHAN

3.1.1 *Dataset*

Penelitian ini dilakukan dengan menganalisis sinyal suara jantung yang telah mengalami proses *denoising*. Data sinyal suara jantung yang akan digunakan dalam proses analisis merupakan data sekunder yaitu berupa *dataset*. *Dataset* ini merupakan data yang berisi rekaman suara jantung yang akan digunakan sebagai bahan dalam pengambilan data sinyal suara jantung. Sinyal suara jantung yang didapatkan dari *dataset* inilah yang digunakan dalam proses analisis sinyal suara jantung sebelum dan sesudah mengalami proses *denoising*. *Dataset* sinyal suara jantung didapatkan dari alamat berikut *https://physionet.org/challenge/2016/*. Sinyal suara jantung ini memiliki frekuensi sampling sebesar 2000 Hz dan terdiri dari jantung normal dan abnormal. Sinyal suara jantung yang digunakan pada penelitian ini adalah sinyal suara jantung normal. Jumlah data sinyal PCG yang digunakan sebesar 30 data sinyal PCG normal.

3.1.2 Perangkat Keras (*Hardware*)

Perangkat keras yang digunakan dalam proses penelitian ini yaitu sebuah PC (*Personal Computer*) dengan spesifikasi sebagai berikut :

1. Intel Celeron N400.
2. *Windows* 10 (64-*Bit*).
3. RAM 4 GB.
4. Memory Harddisk 1 TB.

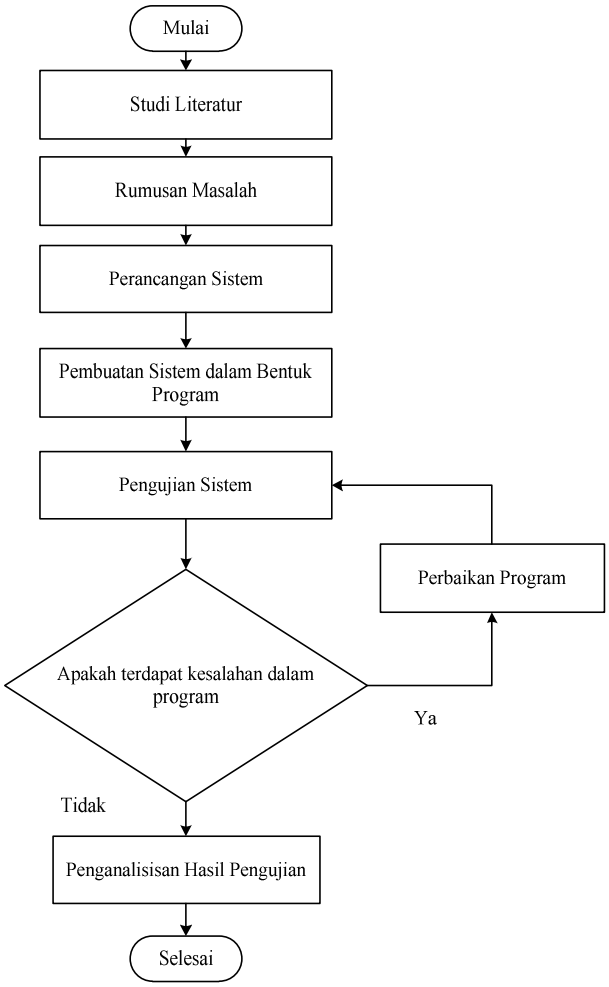
3.1.3 Perangkat Lunak (*Software*)

Software yang akan digunakan dalam pembuatan sistem *denoising* ini adalah Matlab R2018a. *Software* Matlab ini banyak digunakan dalam penelitian di

bidang pengembangan sistem, desain sistem, dan lain-lain. Matlab merupakan *Software* yang akan digunakan untuk proses pemisahan antara sinyal PCG yang masih mengandung *noise* AWGN menjadi sinyal PCG yang tidak mengandung *noise*. Pemisahan *noise* AWGN dari sinyal PCG menggunakan metode TWD dan FHD.

3.2 ALUR PENELITIAN

Alur penelitian merupakan salah satu komponen penting untuk memulai sebuah penelitian hingga berakhirnya penelitian. Pada alur penelitian ini berisikan langkah-langkah yang menjelaskan setiap kegiatan penelitian yang akan dilakukan. Berikut ini merupakan diagram alur penelitian yang disajikan dalam bentuk *flowchart*, yaitu :



Gambar 3.1 Alur Penelitian

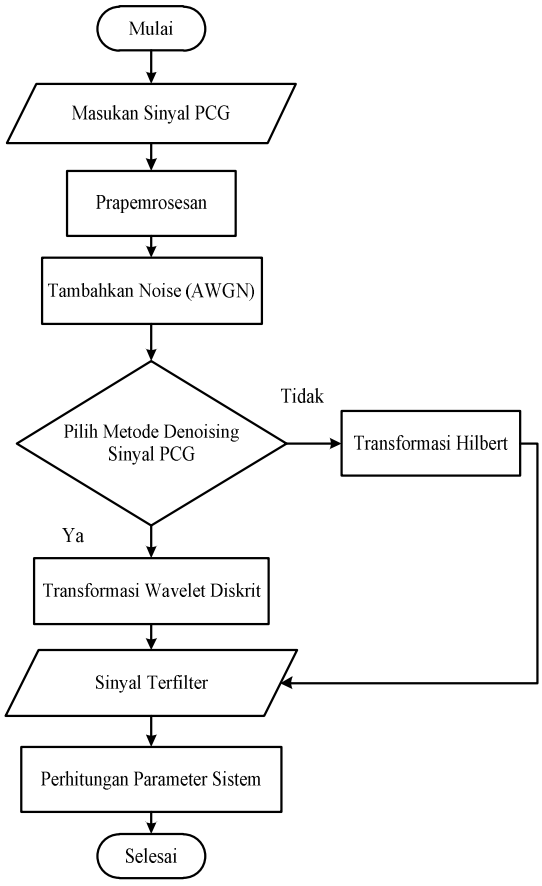
Pada Gambar 3.1 terlihat bahwasannya penelitian yang akan dilakukan dalam tugas akhir ini dimulai dari perancangan sistem. Perancangan sistem merupakan suatu gambaran sistem yang akan digunakan untuk melakukan pengolahan data. Sistem yang akan dibuat merupakan sistem dalam bentuk kode program yang ditulis pada sebuah *software* Matlab R2018a. Rancangan sistem yang telah selesai dibuat dalam bentuk program akan dilakukan pengujian terhadap sistem tersebut. Tujuan dari pengujian ini yaitu untuk mengetahui kemampuan sistem sesungguhnya dari sistem yang telah dibuat. Apabila terjadi kesalahan pada saat pengujian sistem maka langkah yang akan dilakukan yaitu melakukan perbaikan terhadap sistem dengan melakukan pengecekan kode program yang telah dibuat untuk memastikan bahwa kode program yang ditulis sudah sesuai dengan aturan yang semestinya. Apabila sistem tidak ada kesalahan saat sedang dilakukan pengujian maka langkah selanjutnya yaitu melakukan analisis terhadap data yang telah diolah oleh sistem.

3.3 RANCANGAN SISTEM

Pada penelitian ini akan digambarkan mengenai rancangan sistem yang terdiri dari proses prapemrosesan sampai dengan proses *denoising noise* AWGN pada Sinyal PCG dengan menggunakan metode TWD dan FHD. Rancangan sistem ini merupakan tahap penggambaran dari sistem yang akan dibuat dalam bentuk kode program pada *software* Matlab R2018a. Rancangan sistem ini terdiri dari bentuk *flowchart* dan kode program. Pada bentuk *flowchart* akan memberikan gambaran racangan sistem secara umum dan secara spesifik. Rancangan sistem secara spesifik menjelaskan teknik *denoising noise* AWGN dengan menggunakan metode transformasi TWD dan FHD. Penulisan kode program ini merupakan bentuk implementasi dari sebuah rumus matematika yang kemudian diterjemahkan menjadi kode program dengan menggunakan *software* Matlab R2018a. Penulisan kode program ini tidak hanya menggunakan rumus matematika, tetapi menggunakan fungsi *toolbox* yang mana telah disediakan oleh *software* Matlab R2018a. Berikut ini merupakan bentuk rancangan sistem yang akan dibuat, yaitu :

1. Rancangan Sistem Secara Umum

Rancangan sistem ini memberikan gambaran secara menyeluruh mengenai tahap-tahap *denoising noise* AWGN dengan menggunakan metode TWD dan FHD. Berikut ini merupakan rancangan sistem secara umum yang mana dibuat dalam bentuk flowcahart, yaitu :



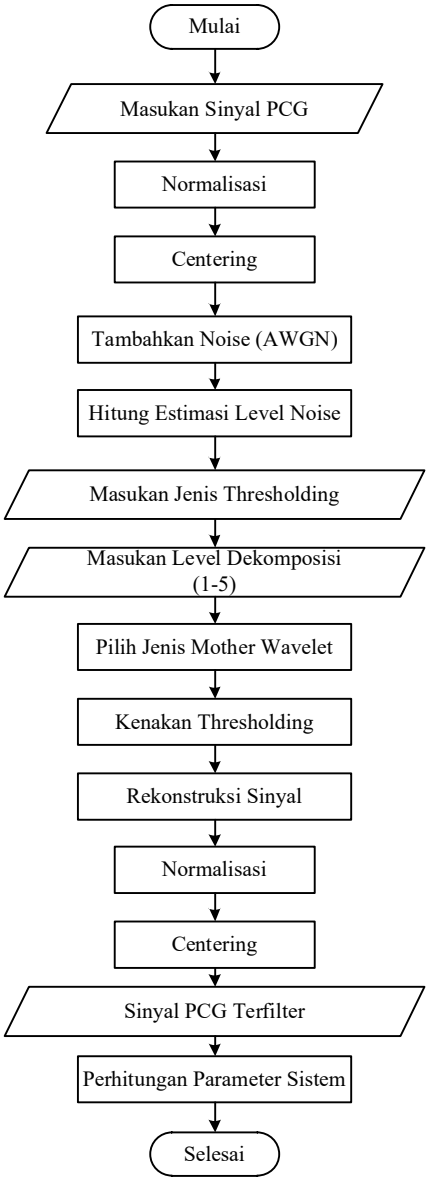
Gambar 3.2 Rancangan Sistem

Pada Gambar 3.2 merupakan rancangan sistem bagian pertama, yang mana rancangan sistem tersebut dimulai dengan memasukan data. Data yang akan dimasukan merupakan data rakaman Sinyal PCG. Sinyal PCG yang belum ditambahkan *noise* AWGN ini akan dilakukan tahap prapemrosesan terlebih dahulu yang terdiri dari normalisasi dan *centering*. Apabila tahap prapemrosesan telah selesai maka tahap selanjutnya yaitu penambahan AWGN. Apabila telah ditambahkan AWGN maka tahap selanjutnya yaitu menentukan metode mana yang akan digunakan untuk melakukan penghilangan *noise* AWGN yang ada pada

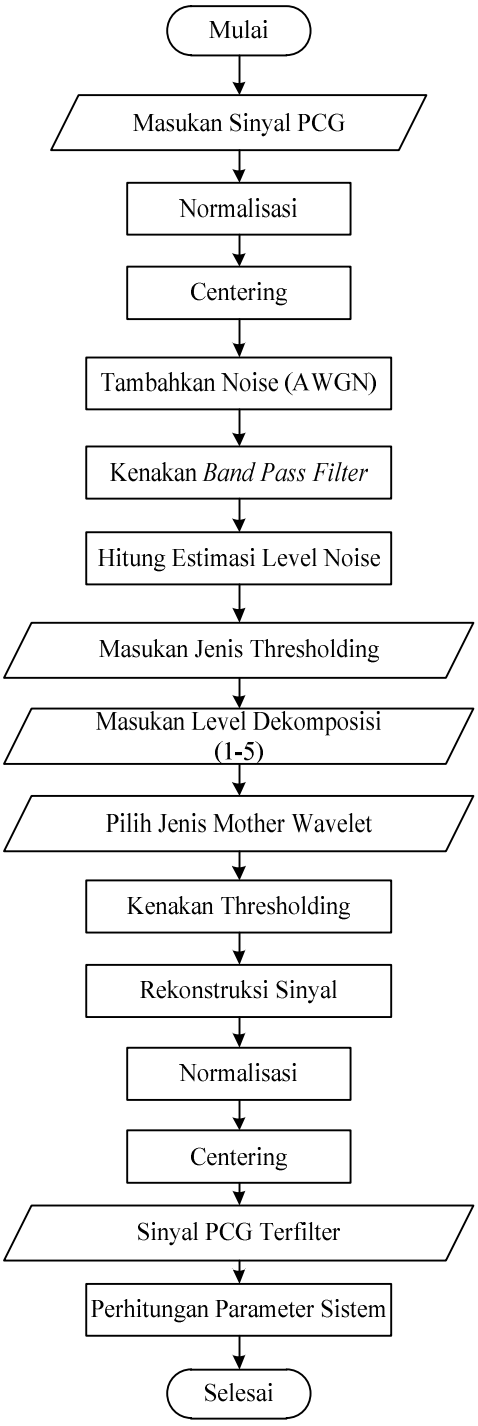
Sinyal PCG. Jika pemilihan metode telah dilakukan maka proses selanjutnya adalah proses penghilagan *noise* AWGN dengan menggunakan metode yang telah dipilih. Tahap selanjutnya yaitu proses penghitungan parameter sistem sebelum dan sesudah proses penghilangan *noise*.

2. Rancangan Sistem Wavelet *Denoising*

Berikut ini merupakan racangan sistem yang mana dibuat dalam bentuk *flowchart* dengan menggunakan metode TWD, yaitu :



Gambar 3.3 Rancangan sistem TWD tanpa BPF



Gambar 3.4 Rancangan sistem TWD dengan BPF

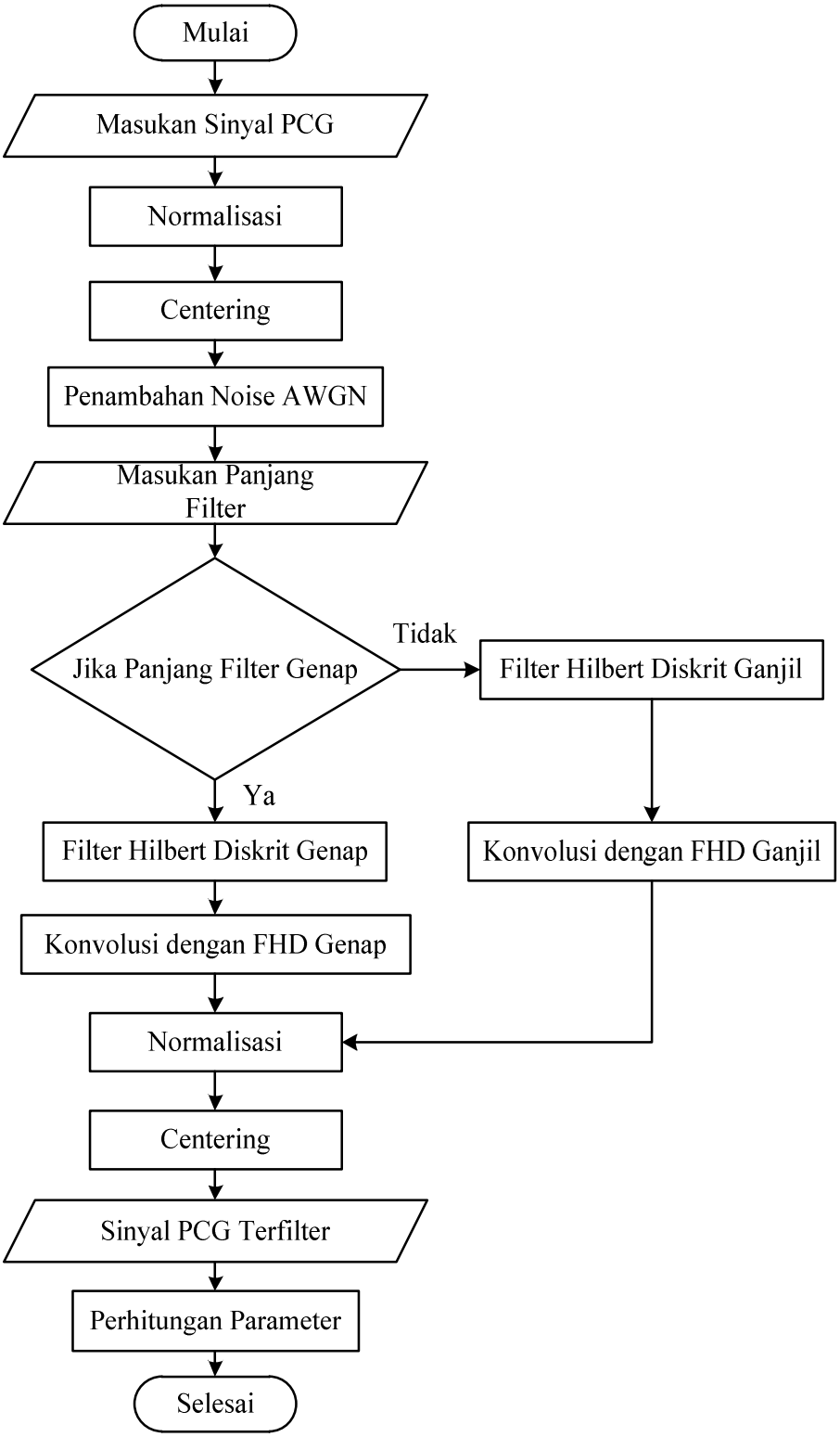
Pada Gambar 3.3 merupakan tahap penghilangan *noise* AWGN pada Sinyal PCG dengan mengguakan metode TWD. Pada metode ini Sinyal PCG harus dinormalisasi dan *centering* terlebih dahulu sebelum ditambahkan *noise*

AWGN. Sinyal PCG yang telah ditambahkan *noise* AWGN akan dihitung level estimasi *noise*-nya dan diikuti dengan penentuan jenis *thresholding*nya yaitu *hard*, *soft*, atau *adaptive*. Masing-masing *thresholding* ini memiliki perhitungan masing-masing sehingga akan menghasilkan nilai *threshold* yang berbeda-beda untuk proses penghilangan *noise* AWGN. Apabila pada tahap ini telah selesai dilakukan, maka tahap selanjutnya adalah pemilihan jenis mother wavelet yang akan digunakan untuk proses penghilangan *noise* AWGN. Setelah dikenakan koefisien wavelet yang didapat dengan menggunakan *threshold* yang telah didapat sebelumnya untuk prose pemfilteran. Proses selanjutya yaitu proses rekonstruksi Sinyal PCG yang telah difilter. Sinyal PCG yang telah difilter akan dikenakan proses normalisasi dan *centering*. Tahap terakhir yaitu perhitungan parameter sistem.

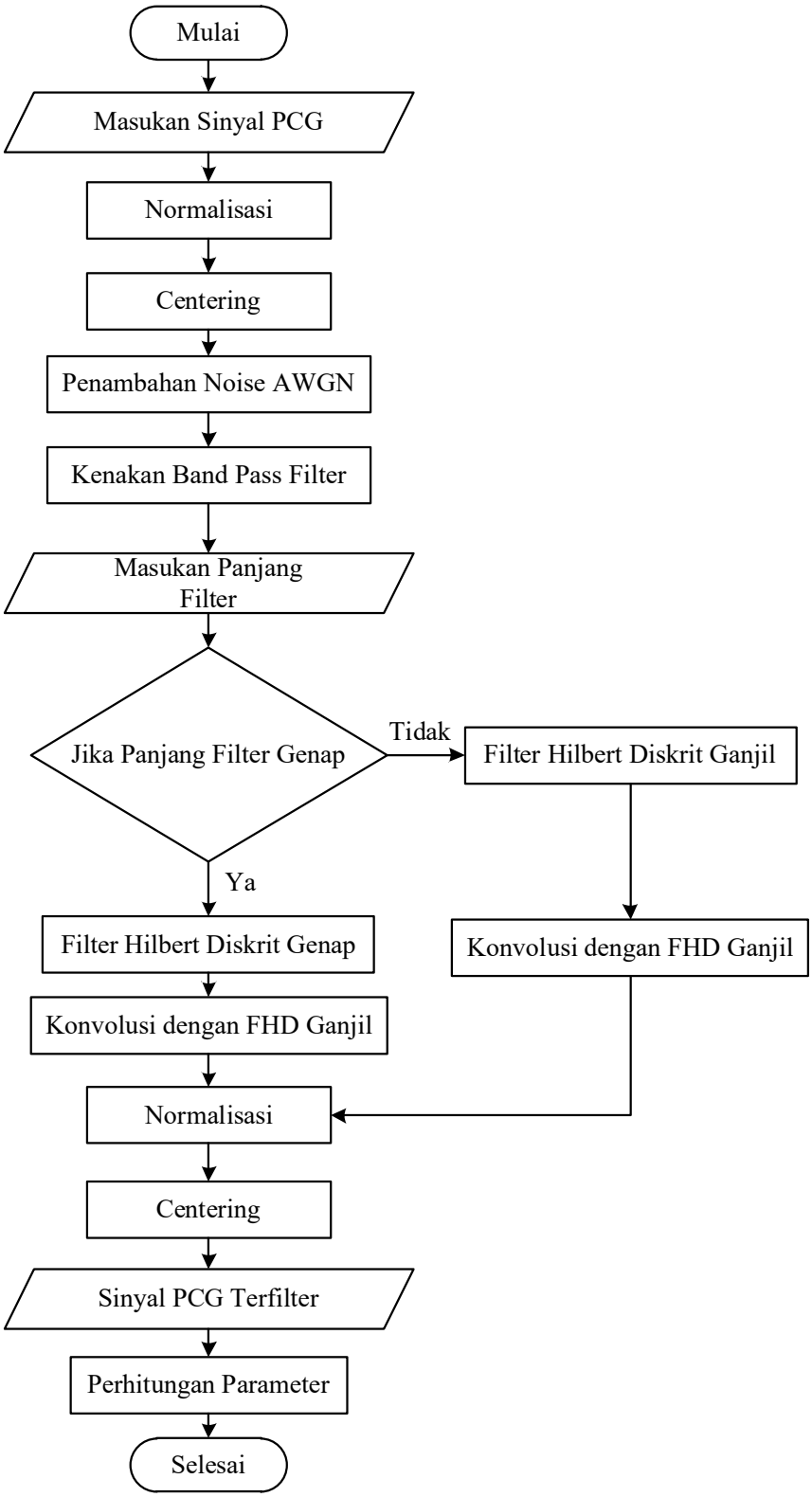
Pada Gambar 3.4 merupakan rnacangan sistem dimana ada penambahan BPF sebelum proses *denoising* dengan menggunakan TWD. Hal ini dikarenakan Sinyal PCG yang digunakan tidak memiliki Sinyal *Ground Truth*, sehingga *noise* yang ada pada Sinyal PCG tidak bisa diketahui nilainya. Oleh karena itu dibuatlah dua rancangan sistem untuk membandingkan nilai parameter yang nantinya akan dihasilkan, baik yang tidak menggunakan BPF ataupun yang menggunakan BPF.

3. Rancangan Sistem Diskrit Hilbert Filter

Pada Gambar 3.5 merupakan *flowchart* untuk proses penghilangan *noise* AWGN yang ada pada Sinyal PCG. Pada Gambar 3.5 Sinyal PCG akan dinormalisasi dan di-*centering* terlebih dahulu sebelum ditambahkan *noise* AWGN. Langkah selanjutnya yaitu menentukan panjang filter yang akan digunakan dari FHD yang terdiri dari panjang filter genap dan ganjil. Apabila panjang filter telah selesai dipilih, maka proses selanjutnya yaitu melakukan proses *filtering*. Sinyal yang telah difilter akan dilakukan proses normalisasi dan *centering* seperti pada tahap sebelumnya. Setelah itu akan dilakukan perhitungan parameter sistem dan proses penghilangan *noise* AWGN pada sinyal PCG telah selesai dilakukan. Berikut ini merupakan bentuk racangan sistem yang mana dibuat dalam bentuk *flowchart* dengan menggunakan metode FHD, baik yang tidak menggunakan BPF ataupun yang menggunakan BPF yaitu :



Gambar 3.5 Rancangan sistem FHD tanpa BPF



Gambar 3.6 Rancangan Sistem FHD dengan BPF

Pada Gambar 3.6 merupakan rnacangan sistem dimana ada penambahan BPF sebelum proses *denoising* dengan menggunakan FHD. Hal ini dikarenakan Sinyal PCG yang digunakan tidak memiliki Sinyal *Ground Truth*, sehingga *noise* yang ada pada Sinyal PCG tidak bisa diketahui nilainya. Oleh karena itu dibuatlah dua rancangan sistem untuk membandingkan nilai parameter yang nantinya akan dihasilkan, baik yang tidak menggunakan BPF ataupun yang menggunakan BPF.

3.3.1 Masukan Sinyal *Phonocardiogram*

Berikut merupakan tahap memasukan Sinyal PCG ke dalam sistem yang telah dibuat dengan menggunakan *software* Matlab R2018a, yaitu :

```
[fname, pname] = uigetfile('*.wav', 'Choose an fPCG data');
% import data ke matlab000
dname = fullfile(pname, fname);
[data, fs] = audioread(dname);
fprintf('File: %s\n', fname);
```

3.3.2 Prapemrosesan

Pada tahap prapemrosesan ini terdiri dari dua tahap yaitu tahap normalisasi dan *centering*. Tahap prapemrosesan merupakan salah satu tahap yang penting sebelum melakukan pengolahan Sinyal PCG.

3.3.2.1 Normalisasi

Tujuan dari normalisasi ini yaitu agar setiap elemen dalam vektor suara bisa diskalakan ke dalam kisaran antara −1.0 dan +1.0 [22]. Sehingga amplitudo pada sebelum dan sesudah proses penghapusan *noise* tidak ada yang berbeda. Berikut ini merupakan persamaan matematis yang digunakan untuk melakukan proses normalisasi pada tahap prapemrosesan, yaitu :

$$s_{norm}(t) = \frac{s(t)}{max(|s(t)|)} \quad \textbf{(3.1)}$$

Dimana,

$s(t)$ = Sinyal PCG

$s_{norm}(t)$ = Sinyal Hasil Normalisasi

Berikut ini merupakan penerapan rumus menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
% normalisasi data mentah agar berada pada -1 hingga +1 volt
data = data ./ max(abs(data));
```

3.3.2.2 *Centering*

Centering mengubah semua konsentrasi menjadi fluktuasi disekitar nol sebagai gantinya yaitu disekitar rata-rata data. Dengan ini, ia menyesuaikan perbedaan di antara data tinggi dan rendah yang berlimpah. Oleh karena itu digunakan untuk fokus pada bagian data yang berfluktuasi, dan hanya menyisakan variasi yang relevan. Tujuan dari *centering* ini yaitu untuk menghapus *offset* yang ada pada sauatu data [23]. Berikut ini merupakan persamaan matematis untuk melakukan *centering*, yaitu :

$$\tilde{x}_{ij} = x_{ij} - \bar{x}_{\iota} \qquad \textbf{(3.2)}$$

Dimana,

$\tilde{x}_{ij}$ = Sinyal Hasil *Centering*

x_{ij} = Sinyal PCG Asli

$\bar{x}_{\iota}$ = Rata-rata Sinyal PCG Asli

Berikut ini merupakan penerapan rumus menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
% centering
data = data - mean(data(:));
```

3.3.3 Penambahan AWGN

AWGN merupakan salah satu metode untuk menambahkan *noise* pada sebuah sinyal secara distribusi dan merata ke dalam semua frekuensi. Sinyal PCG sebelum masuk tahap *denoising* dengan menggunakan TWD dan FHD akan ditambahkan AWGN terlebih dahulu. Penambahan AWGN pada Sinyal PCG ini bertujuan untuk menurunkan kualitas Sinyal PCG dan digunakan untuk mengukur seberapa baik sistem yang dibuat dapat menghilangkan AWGN yang telah bercampur dengan Sinyal PCG. Berikut ini merupakan penerapan rumus dari persamaan 2.5 menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
% berikan noise  acak N(0,1)
% datan = wgn(length(data), 1, 0)';
datan = awgn(data, 10, 'Measured');
datax = data + datan;
```

3.3.4 Penambahan *Band Pass Filter*

Pada penelitian ini akan ada empat percobaan yang dilakukan yaitu percobaan tanpa menggunakan BPF dan menggunakan BPF, baik untuk TWD maupun untuk FHD. Penambahan BPF ini digunakan karena Sinyal PCG yang digunakan tidak memiliki Sinyal *Ground Truth*. Hal ini menyebabkan nilai *noise* yang ada pada Sinyal PCG tidak bisa diketahui dengan pasti. Berikut merupakan kode program yang digunakan untuk BPF pada matlab dengan rentang frekuensi PCG sebesar 10 – 250 Hz pada Matlab R2018a, yaitu :

```
bpFilt = designfilt('bandpassiir','FilterOrder',10, ...
'HalfPowerFrequency1',10,'HalfPowerFrequency2',250, ...
'SampleRate',fs);
filtdata = filtfilt(bpFilt, datan);
datax = filtdata;
```

3.3.5 *Denoising* Transformasi Wavelet Diskrit

Ada beberapa tahapan yang perlu dilakukan dalam melakukan *denoising* dengan menggunakan metode TWD, yaitu :

3.3.5.1 Estimasi Level *Noise*

Metode *thresholding* bisa dikelompokan menjadi dua kategori yaitu global *thresholding* dan bergantung level *thresholding*. Hal ini berarti bahwa pemilihan sebuah nilai tunggal untuk *thresholding* λ untuk diterapkan secara global untuk semua empirical koefisien wavelet, yang mana dapat diartikan bahwa memungkinkan perbedaan nilai λ_j yang dipilih untuk setiap level resolusi j. Sebuah *thresholding* membutuhkan membutuhkan sebuah estimasi dari sebuah *noise* level σ. Biasanya standar deviasi dari nilai data secara jelas tidak bagus untuk sebuah estimasi, kecuali kalau yang mendasari fungsi S-nya adalah masuk

akal. Berdasarkan Donoho dan Johnstone (1994) estimasi σ dalam domain wavelet dan disarankan bahwa sebuah estimasi yang kuat yang mana berdasarkan hanya pada koefisien wavelet pada level resolusi yang paling baik. Alasannya yaitu berdasarkan pertimbangan hanya pada level yang paling baik yang koefisien wavelet empirikalnya cenderung berhubungan yang mana sebagian besar terdiri dari *noise*. Berdasarkan pada level estimasi yang diusulkan oleh Donoho dan Johnstone (1994) dapat dituliskan secara matematis sebagia berikut [10]. Berikut ini merupakan penerapan rumus pada persamaan 2.9 menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
lvl0 = 1;
mot = 'db1';
[C, L] = wavedec(x, lvl0, mot);
cD1 = detcoef(C, L, lvl0);
xs = median(abs(cD1));
nLevel  = xs / 0.6745;
```

3.3.5.2 Aturan *Shrinkage*

Aturan *Shrinkage* menentukan untuk proses perhitungan nilai *threshold*, yang mana nilai *threshold* dinotasikan dengan simbol λ. Aturan lain dalam perhitungan *threshold* independen yaitu dari fungsi *Shrinkage* ketika yang lain memperoleh *threshold* yang berbeda untuk fungsi *threshold* yang berbeda. Aturan yang lainnya menganggap bahwa sebuah unit skala *noise* , $\sigma = 1$, sedangkan yang lainnya tidak [10].

1) Universal

Berikut ini merupakan penerapan rumus sebagaimana persamaan 2.12 menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
n = length(x);                % Panjang Sinyal PCG
th = sqrt(2 * log(n));
th = th * nLevel ;            % nLevel  merupakan Estimasi Level  Noise
```

2) SURE

Berikut ini merupakan penerapan rumus pada persamaan 2.19 menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
a = sort(abs(x)) .^ 2;                  % x merupakan Sinyal PCG
b = cumsum(a);
n = length(x);                          % n merupakan panjang Sinyal PCG
c = linspace(n - 1, 0, n);
s = b + c .* a;
risk = (n - (2 .* (1 : n)) + s) / n;
[~, ibest] = min(risk);
th = sqrt(a(ibest));
th = th * nLevel ;                      %nLevel  merupakan Estimasi Level  Noise
```

3.3.5.3 *Thresholding*

Ada tiga jenis fungsi *thresholding* yang akan digunakan yaitu *hard*, *soft*, dan *adaptive thresholding*. Berikut merupakan fungsi *thresholding* yang akan digunakan, yaitu :

1) *Hard Threshold*

Pada *Hard Threshold* menggunakan aturan universal untuk menacari nilai *threshold*-nya. Proses *filtering* dengan menggunakan *hard threshold* dapat dituliskan secara matematis sebagaimana persamaan 2.21. Berikut ini merupakan penerapan rumus pada persamaan 2.21 menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
xf = x .* (abs(x) > th);
```

2) *Soft Threshold*

Pada *Soft Threshold* menggunakan aturan universal untuk menacari nilai *threshold*nya. Proses *filtering* dengan menggunakan *soft threshold* dapat dituliskan secara matematis sebagaimana persamaan 2.22. Berikut ini merupakan penerapan rumus pada persamaan 2.22 menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
temp = abs(x) - th;
temp = (temp + abs(temp))/2;
xf = sign(x) .* temp;
```

3) *Adaptive Threshold*

Pada *Adaptive Threshold* menggunakan aturan SURE untuk menacari nilai *threshold*nya. Proses *filtering* dengan menggunakan SURE *threshold* dapat dituliskan secara matematis sebagaimana persamaan 2.23. Berikut ini merupakan penerapan rumus pada persamaan 2.23 menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
xf = x .* (abs(x) > th);
```

3.3.5.4 Rekonstruksi Transformasi Wavelet Diskrit

Rekosntruksi digunakan untuk menggabungkan sinyal yang telah di-*thresholding* baik menggunakan *hard*, *soft*, ataupun *adaptive*. Rekostruksi ini adalah tahap terakhir dari *denoising* Sinyal PCG dengan menggunakan TWD. Berikut adalah kode program yang digunakan untuk proses rekonstruksi dengan menggunakan TWD, yaitu :

```
% rekonstruksi
dden = waverec(xf, L, mot{1});
 % normalisasi ke -1 - +1 volt
dden = dden ./ max(abs(dden));
 % centering
dden = dden - mean(dden);
```

3.3.6 *Denoising* Filter Hilbert Diskrit

Proses *denoising* dengan menggunakan FHD dilakukan dengan menggunakan panjang filter yang berbeda-beda. Ada dua jenis panjang filter yang digunakan untuk proses *denoising* dengan menggunakan FHD yaitu panjang filter genap dan ganjil. Berikut merupakan Firter Hilbert Diskrit yang akan digunakan, yaitu :

3.3.6.1 Filter Hilbert Diskrit Genap

Dikatakan sebagai FHD Genap karena panjang filter yang digunakan untuk proses *filtering* Sinyal PCG yang terkena *noise* AWGN adalah bernilai genap. Berikut merupakan penulisan secara matematis dari FHD Genap, yaitu [24]:

$$\mathrm{h}(i) = \frac{2}{N}\sum_{k=0}^{N-1}\sin(w) = \frac{2}{N}sin^2\left(\frac{\pi i}{2}\right)\cot\left(\frac{\pi i}{N}\right) \quad (3.3)$$
$$i = 0,1,\dots,N-1, w = \frac{2\pi ki}{N}$$

Dimana,

$\mathrm{h}(i)$ = Respon impuls diskrit Hilbert filter

N = Panjang FHD genap

w = Koefisien Detail

Berikut ini merupakan penerapan rumus menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
x1 = 0:N-1;
hi1 = (2/N)*sin((pi*x1)/2).^2.*cot((pi*x1)/N);
hi1 = hi1(2 : end).';
```

3.3.6.2 Filter Hilbert Diskrit Ganjil

Dikatakan sebagai FHD Ganjil karena panjang filter yang digunakan untuk proses *filtering* Sinyal PCG yang terkena *noise* AWGN adalah bernilai ganjil. Berikut merupakan penulisan secara matematis dari FHD Ganjil, yaitu [24]:

$$\mathrm{h}(i) = \frac{1}{N}\left[1 - \frac{cos(\pi i)}{cos\left(\frac{\pi i}{N}\right)}cot\left(\frac{\pi i}{N}\right)\right] \quad (3.4)$$
$$i = 0,1,\dots,N-1$$

Dimana,

$\mathrm{h}(i)$ = Respon impuls diskrit Hilbert filter

N = Panjang FHD ganjil

Berikut ini merupakan penerapan rumus menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
x3 = 0:N-1; % Menghasilkan nilai awal Inf atau NaN
a1 = cos(pi*x3);
a2 = cos((pi*x3/N));
a3 = cot((pi*x3)/N);
hi1 = (1/N)*(1-((a1./a2).*a3));
hi1 = hi1(2 : end).';
```

3.3.6.3 Konvolusi Diskrit Hilbert Filter

FHD yang telah dibuat baik yang panjang filternya ganjil maupun genap akan dikonvolusikan dengan Sinyal PCG yang terkena *noise* AWGN. Hasil konvolusi antara Sinyal PCG yang terkena AWGN dengan FHD inilah yang disebut sebagai Sinyal PCG terfilter. Berikut merupakan bentuk matematis dari konvolusi Sinyal PCG yang terkena *noise* AWGN dengan FHD, yaitu [24]:

$$v(i) = -x(i) \otimes h(i) \quad i = 0,1,\dots,N-1 \tag{3.5}$$

Dimana,

$v(i)$ = Konvolusi

$x(i)$ = Sinyal Asli

$h(i)$ = Respon impuls diskrit Hilbert filter

Berikut ini merupakan penerapan rumus menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
dden1 = conv(-datax, hi1, 'same');    % Konvolusi Filter Hilbert Genap
dden2 = conv(-datax, hi1, 'same');    % Konvolusi Filter Hilbert Ganjil
```

3.4 PARAMETER PENELITIAN

Parameter penelitian yang akan digunakan ada dua macam yaitu parameter pengujian dan parameter sistem. Berikut merupakan parameter-parameter yang akan digunakan dalam penelitian ini, yaitu :

3.4.1 Parameter Pengujian

Parameter pengujian adalah parameter yang digunakan untuk menguji sistem yang telah dibuat untuk proses *denoising* Sinyal PCG dengan

menggunakan metode TWD dan FHD. Berikut merupakan parameter pengujian yang akan digunakan, yaitu :

3.4.1.1 Parameter Transformasi Wavelet Diskrit

Berikut ini merupakan parameter pengujian yang akan digunakan dalam *denoising* AWGN pada Sinyal PCG dengan menggunakan metode TWD, yaitu :

Tabel 3.1 Parameter Pengujian TWD

No	Jenis Mather Wavelet	Level Dekomposisi	*Thresholding*		
1	db10, db20, db30	1-5	*Hard*	*Soft*	*Adaptive*
2	sym6, sym7, sym8	1-5	*Hard*	*Soft*	*Adaptive*
3	coif3, coif4, coif5	1-5	*Hard*	*Soft*	*Adaptive*

3.4.1.2 Parameter Filter Hilbert Diskrit

Berikut ini merupakan parameter pengujian yang akan digunakan dalam *denoising noise* AWGN pada Sinyal PCG dengan menggunakan metode Diskrit Hilbert Filter, yaitu :

Tabel 3.2 Parameter Pengujian FHD Genap

No	Panjang Filter	Kelipatan		
1	Tahap 1	2, 4, ...,20	22, 24, …,40	42, 44, …,60
2	Tahap 2	2, 6, …, 38	42, 46, …, 78	82, 86, …, 118
3	Tahap 3	2, 8, …, 56	62, 68, …, 116	122, 128, …, 176

Tabel 3.3 Parameter Pengujian FHD Ganjil

No	Panjang Filter	Kelipatan		
1	Tahap 1	1, 3, …,19	21, 23, …,39	41, 43, …,59
2	Tahap 2	1, 5, …, 37	41, 45, …, 77	81, 85, …, 117
3	Tahap 3	1, 7, ..., 55	61, 67, …, 115	121, 127, …, 175

3.4.2 Parameter Sistem

Parameter sistem yang akan digunakan adalah SNR dan MSE, parameter sistem ini berfungsi untuk mengetahui kinerja dari sistem yang telah dibuat untuk

proses *denoising*. Berikut merupakan parameter sistem yang akan digunakan, yaitu :

3.4.2.1 Signal Noise to Ratio

SNR digunakan untuk mengetahui kinerja dari sistem yang dibuat. Pengukuran kinerja sistem dilakukan dengan cara membandingkan SNR sebelum dan sesudah dilakukan proses *denoising* pada Sinyal PCG. SNR ini biasanya merupakan ukuran yang paling penting daripada kekuatan *noise*, ketika mempertimbangkan bahwa kekuatan sinyal untuk *denoising noise* dapat bervariasi. Semakin besar nilai SNR yang dihasilkan maka kualitas sinyal akan semakin bagus dan sebaliknya. Secara matematis bentuk persamaan dari SNR, yaitu [8][13] :

$$SNRbefore = 10log_{10}\left(\frac{(\bar{x}(n))^2}{(\bar{x}'(n))^2}\right) \quad \textbf{(3.6)}$$

Dimana,

$\bar{x}(n)$ = Rata-rata nilai Sinyal PCG Asli

$\bar{x}'(n)$ = Nilai rata-rata Sinyal PCG yang ditambahkan AWGN

$$SNRafter = 20log_{10}\left(\frac{Peak\ to\ Peak\ (PCG)}{4.std(PCG)}\right) \quad \textbf{(3.7)}$$

Dimana,

Peak to Peak (PCG) = Mengukur nilai amplitudo maximum-minus minimum S1 Sinyal PCG

std(PCG) = Mengukur *noise* pada Sinyal PCG

Berikut ini merupakan penerapan rumus menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
% SNR Sebelum Denoising tanpa toolbox
snr_before = mean(data.^2) / mean(datax.^2);
SNR1 = 10 * log10(snr_before);
% SNR Sebelum Denoising dengan toolbox
% SNR1 = snr(data, datax);
% SNR Sesudah Denoising
p2p = peak2peak(dden);
```

```
stdp = std(dden);
cpr = p2p ./ (4 * stdp);
SNR2 = abs(20 * log10(cpr));
fprintf(', SNR1 : %.2f', SNR1(id, ix));
fprintf(', SNR2 : %.2f]\n', SNR2(id, ix));
```

3.4.2.2 Mean Square Error

MSE adalah parameter yang digunakan untuk mengetahui tingkat *error* dari proses *denoising* yang telah dilakukan. Jika nilai MSE semakin mendekati 0 maka itu menandakan bahwa sistem yang telah dibuat memiliki kehandalan yang bagus. Secara matematis MSE dapat dituliskan sebagai berikut :

$$MSE = \frac{\sum_{i=1}^{n}(s - s_e)^2}{n} \quad \textbf{(3.8)}$$

Dimana,

s = Sinyal Asli

s_e = Sinyal hasil *denoising*

n = Panjang sinyal

Berikut ini merupakan penerapan rumus menjadi kode program dengan menggunakan *software* Matlab R2018a, yaitu :

```
MSE = (mean((data - dden).^2))*100;
fprintf('[MSE : %.2f%%', MSE);
```