BAB III
METODE PENELITIAN

3.1 ALAT DAN BAHAN

Pada bab ini akan dijelaskan mengenai data-data yang digunakan sebagai *input* pada perangkat lunak untuk diproses dan diuji lebih lanjut sehingga menghasilkan *output* data yang diharapkan.

3.1.1 Dataset

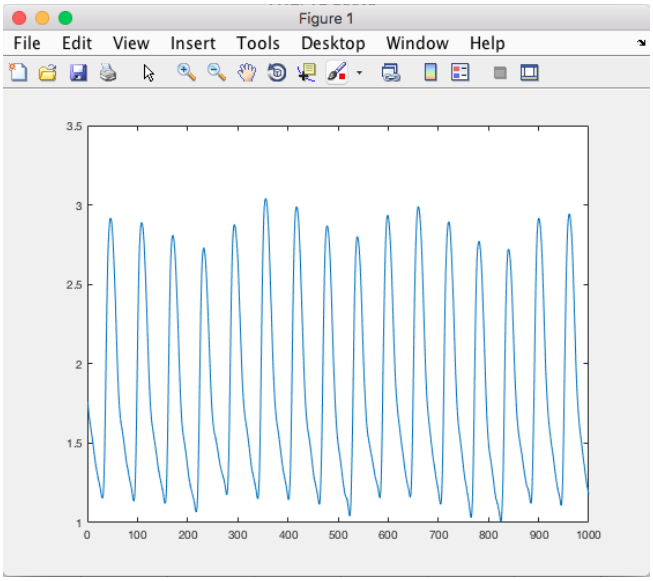
Pada penelitian ini Dataset yang digunakan pada klasifikasi tekanan darah adalah data sinyal ECG, PPG, dan ABP yang diunduh dari situs *University of California-Irvine* (UCI) *Machine Learning Repository*. Dari data yang diunduh, terdapat 61 individu, dimana setiap individu memiliki rekaman sinyal yang berbeda-beda. Dari 61 individu tersebut diambil 60 individu dengan masing-masing satu buah rekaman sinyal. Jadi kami mendapatkan 60 buah sinyal dari 60 individu.

3x1000 double

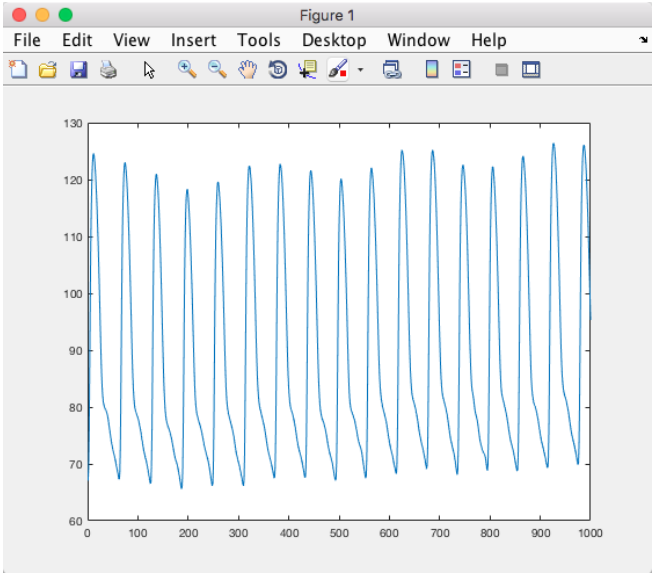
		1	2	3	4	5	6	7	8	9
ECG	1	1.7595	1.7185	1.6843	1.6579	1.6373	1.6158	1.5934	1.5709	1.5494
ABP	2	67.0630	69.3586	75.3665	85.0376	96.2229	105.9429	112.9276	117.6654	120.9868
PPG	3	-0.0606	-0.0753	-0.0704	-0.0352	0.0244	0.0899	0.1549	0.2048	0.2498
	4									
	5									
	6									
	7									
	8									
	9									
	10									

Gambar 3. 1 Data rekaman sinyal pada satu individu.

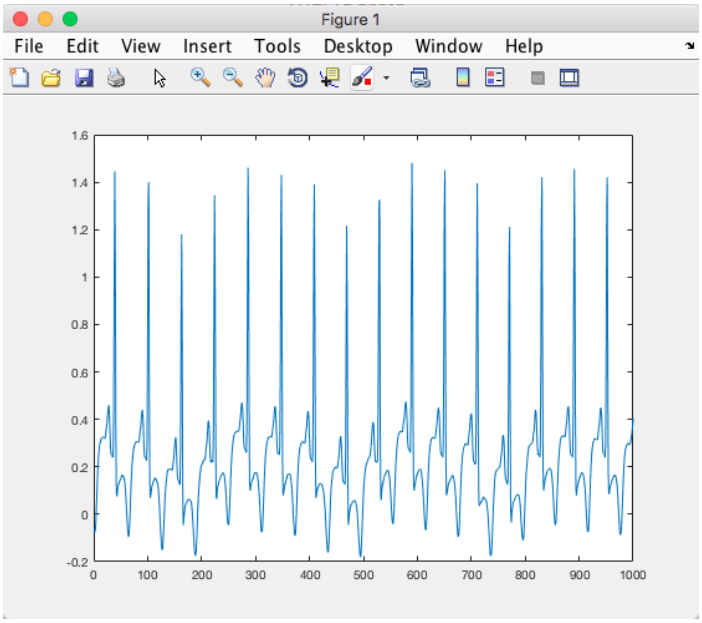
Agar data pada Gambar 3.1 mudah dibaca, maka divisualisasikan menggunakan fungsi plot pada Matlab sehingga dapat terlihat pergerakan naik turun gelombang. Visualisasi sinyal dapat ditunjukkan pada Gambar 3.2 untuk sinyal EKG, Gambar 3.3 untuk tekanan ABP, dan Gambar 3.4 untuk sinyal PPG. Pada Gambar 3.2 dan 3.4, koordinat x mewakili jumlah kolom, dan koordinat y adalah amplitudo. Sedangkan pada Gambar 3.3 koordinat x mewakili jumlah kolom, dan koordinat y adalah tekanan darah.



Gambar 3. 2 Visualisasi Sinyal ECG



Gambar 3. 3 Visualisasi Sinyal Tekanan ABP



Gambar 3. 4 Visualisasi Sinyal PPG

3.1.2 Perangkat

Dalam penelitian Klasifikasi Tekanan Darah Dari Sinyal EKG dan PPG Menggunakan Metode *Wavelet Transform* dan *Random Forest* memerlukan perangkat keras dan perangkat lunak. Berikut jenis perangkat dan spesifikasi ditunjukan pada Tabel 3.1.

Tabel 3. 1 Implementasi Perangkat

Perangkat	Jenis Perangkat	Spesifikasi
Perangkat Keras	Prosesor	Intel Core i5 2,5 GHz
	Memori	8 GB 1333 MHz DDR3
Perangkat Lunak	Sistem Operasi	MacOS High Sierra Versi 10.13.6 (17G66)
	Perangkat Pengembang	MATLAB_2018a
	Perangkat Pembantu	Microsoft Word 2020 Microsoft Excel 2020

3.2 ALUR PENELITIAN

Rancangan penelitian KlasifikasiTekanan Darah Dari Sinyal EKG dan PPG Menggunakan Metode *Wavelet Transform* dan *Random Forest* memiliki 3 proses

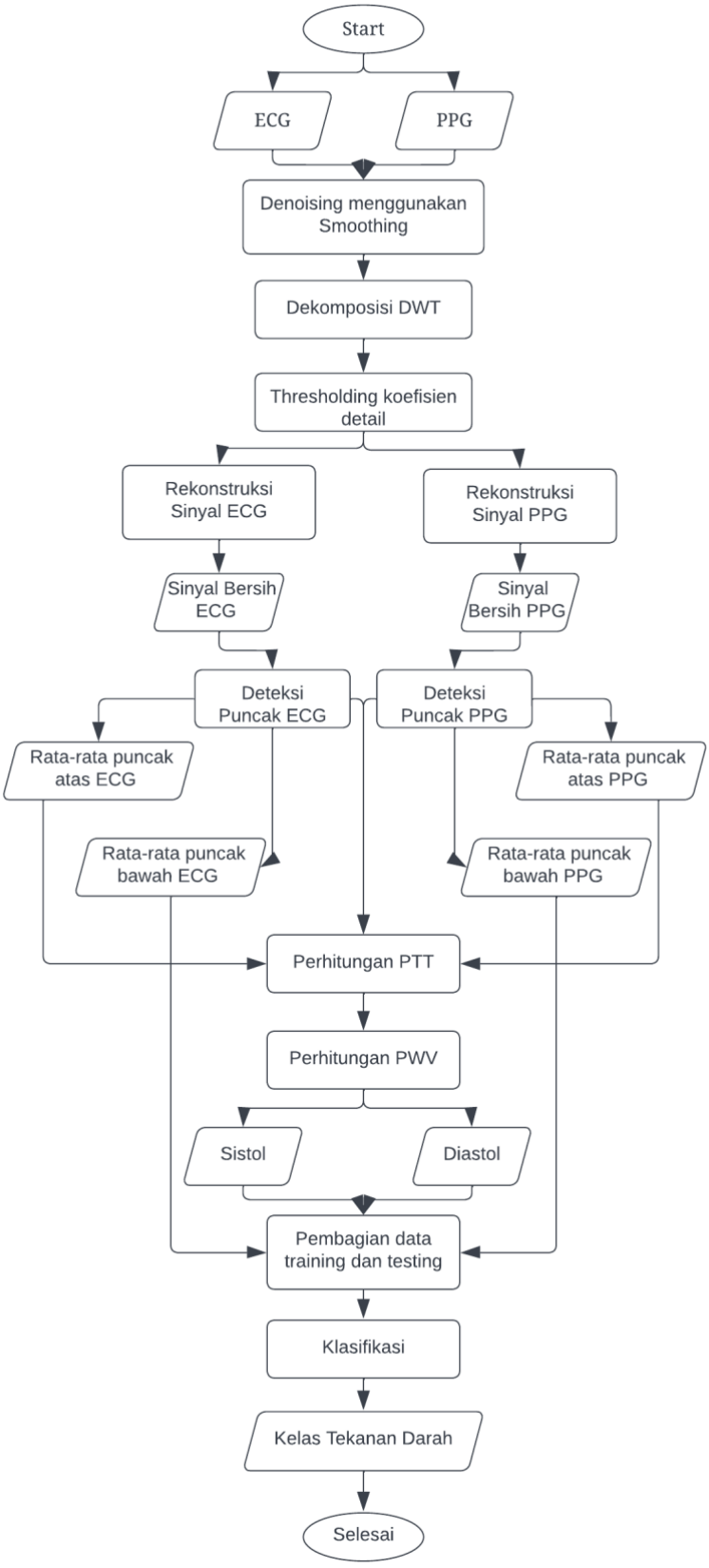
utama yaitu *preprocessing*, ekstraksi dan komputasi fitur, lalu klasifikasi. Dimulai dengan data *preprocessing* yaitu menghilangkan *noise* dari data sinyal ECG dan PPG dengan menggunakan *signal smoothing*. Selanjutnya sinyal tersebut diproses menggunakan *Discrete Wavelet Transform* (DWT). Kemudian direkonstruksi kembali dengan mendapatkan sinyal yang bersih dan tidak ada *noise* setelah proses *preprocessing*.

Setelah sinyal bersih maka bisa dilanjutkan dengan proses ektraksi fitur untuk dilakukan pendeteksian sinyal puncak atas dan bawah dari sinyal EKG serta sinyal puncak atas dan bawah dari PPG. Dengan diketahuinya hasil dari ekstraksi fitur, kita bisa melakukan komputasi perhitungan nilai *Pulse Transit Time* (PTT) hanya dengan memanfaatkan sinyal puncak atas dari sinyal EKG dan PPG. Selanjutnya untuk mencari nilai *Pulse Wave Volecity* (PWV) menggunakan nilai PTT yang sudah didapat. Setelah PWV diketahui, maka dapat menghitung nilai tekanan sistol dan diastolnya.

Selain itu juga dilakukan komputasi penghitungan nilai waktu rata-rata dari puncak atas dan bawah sinyal ECG dan PPG. Terdapat 8 fitur yang dihasilkan dari proses komputasi, yaitu tekanan sistolik, tekanan diastolik, puncak atas EKG, puncak bawah EKG, puncak atas PPG, dan puncak bawah dari PPG. Lalu dilakukan *splitting* atau memisahan data menjadi 2 bagian yaitu data *training* dan *testting*.

Pada pembagian data *training* dan data *testing* dilakukan dengan menggunakan K-*fold Cross Validation* yang merupakan metode yang paling sering digunakan. Dalam metode ini dataset dibagi menjadi beberapa K-partisi secara acak.

Kemudian dilakukan sejumlah K-eksperimen, dimana setiap percobaan menggunakan data partisi ke -K sebagai data *testing* dan menggunakan partisi yang tersisa sebagai data *training*. Setelah melakukan *splitting*, dalam melakukan klasifikasi *Random Forest* yaitu pemilihan *mtry* dan *ntree* yang digunakan untuk mencari tingkat misklasifikasi pada data *training* kemudian data *testing* yang akan digunakan untuk mencari tingkat akurasi. Pada data yang diujikan memiliki 4 kelas yang terdiri dari 8 variabel atau *feature* hasil ekstrtaksi dari sinyal ECG dan PPG setelah melewati *preprocessing* dan komputasi.



Gambar 3. 5 Alur Diagram

3.3 PREPROCESSING

Preprocessing adalah proses mengubah data menjadi format yang sesuai lebih sederhana, lebih efektif, dan sesuai dengan kebutuhan pengguna. indikator yang dapat digunakan sebagai referensi hasil yang lebih akurat, waktu komputasi lebih pendek, juga data menjadi lebih kecil tanpa mengubah informasi yang ada di dalamnya. Beberapa metode *preprocessing* adalah memilih *subset* sampel dari populasi data yang besar, juga *denoising* yaitu menghilangkan *noise* dari data.

3.3.1 *Denoising*

Denoising dilakukan dengan menggunakan *signal smoothing* dengan tujuan untuk mengurangi *noise* pada sinyal sehingga sinyal dengan nilai abnormal dan di luar batas normal akan dihilangkan. Sinyal terputus-putus juga akan dihapus.

Bentuk *smoothing* yang paling sederhana adalah *moving average* dengan meratakan *filter point averaging*. Rata-rata bergerak bekerja dengan mengganti setiap nilai data dengan rata-rata nilai tetangga dengan jumlah poin yang dirata-ratakannya. Nilai rata-rata titik optimal pada percobaan ini adalah 40 untuk sinyal EKG dan 20 untuk sinyal PPG.

```
1. [FileName,PathName] = uigetfile('*.mat','Select the MATLAB data
2. file')
3. load(FileName,'-mat')
4. fs = 125
5. raw_ecg = Parter(1,2001:3000);
6. raw_ppg = Parter(3,2001:3000);
7. ecgsmooth = smooth(raw_ecg,40);
8. ppgsmooth = smooth(raw_ppg,20);
```

Kode Sumber 4. 1 kode masukan dan *smoothing*

Sinyal *input* data berupa EKG dan PPG yang memiliki frekuensi 125 Hz. Pada Kode Sumber 4.1 baris 1-3 merupakan proses pencarian *file* dalam format .mat yang akan digunakan. Kemudian pada baris 5-6 data dipecah menjadi 2 yaitu sinyal EKG dan PPG. Sinyal data EKG terletak pada baris 1 dataset, sedangkan sinyal PPG terletak pada baris 3. Pada baris 7-8, pemulusan dilakukan dengan menggunakan *point averaging* atau *window* jendela 40 untuk sinyal EKG dan 20 untuk sinyal PPG.

3.3.2 Dekomposisi DWT

DWT membagi dimensi sinyal menjadi dua bagian, bagian dengan frekuensi tinggi dan frekuensi rendah, disebut dekomposisi. Sinyal dilewatkan melalui *highpass filter* untuk dianalisis frekuensi tinggi, dan melewati *lowpass filter* untuk dianalisis frekuensi rendah. *Output* dari *highpass filter* dan *lowpass filter* ini menghasilkan Koefisien DWT, dengan menggunakan koefisien ini citra asli dapat direkonstruksi. Jenis *mother wavelet* yang digunakan adalah *Daubechies* 6 (db6). Proses dekomposisi dilakukan sebanyak 8 kali yaitu pada level 8 untuk mendapatkan 8 koefisien detail dan 8 koefisien aproksimasi.

```
1.  [Lo_D,Hi_D,Lo_R,Hi_R] = wfilters('db6');
2.  [C,L] = wavedec(ecgsmooth,8,Lo_D,Hi_D);
3.  [d1,d2,d3,d4,d5,d6,d7,d8]=detcoef(C,L,[1,2,3,4,5,6,7,8]);
4.  A8 = wrcoef('a',C,L,Lo_R,Hi_R,8);
5.  A6 = wrcoef('a',C,L,Lo_R,Hi_R,6);
6.  A7 = wrcoef('a',C,L,Lo_R,Hi_R,7);
7.  A5 = wrcoef('a',C,L,Lo_R,Hi_R,5);
8.  A4 = wrcoef('a',C,L,Lo_R,Hi_R,4);
9.  A3 = wrcoef('a',C,L,Lo_R,Hi_R,3);
10. A2 = wrcoef('a',C,L,Lo_R,Hi_R,2);
11. A1 = wrcoef('a',C,L,Lo_R,Hi_R,1);
12. D1 = wrcoef('d',C,L,Lo_R,Hi_R,1);
13. D2 = wrcoef('d',C,L,Lo_R,Hi_R,2);
14. D3 = wrcoef('d',C,L,Lo_R,Hi_R,3);
15. D4 = wrcoef('d',C,L,Lo_R,Hi_R,4);
16. D5 = wrcoef('d',C,L,Lo_R,Hi_R,5);
17. D6 = wrcoef('d',C,L,Lo_R,Hi_R,6);
18. D7 = wrcoef('d',C,L,Lo_R,Hi_R,7);
19. D8 = wrcoef('d',C,L,Lo_R,Hi_R,8);
```

Kode Sumber 4. 2 Kode Program DWT pada sinyal ECG

Hasil dari *preprocessing* tersebut digunakan sebagai *input* pada proses DWT. Berdasarkan Kode Sumber 4.2, baris 1-3 merupakan dekomposisi DWT. Sinyal tersebut menggunakan 4 filter yang ditunjukkan oleh baris pertama, yaitu Lo_D untuk dekomposisi *low-pass*, Hi_D dekomposisi *high-pass*, Lo_R untuk rekonstruksi *low-pass*, dan Hi_R untuk rekonstruksi *high-pass*. Pada baris kedua sinyal didekomposisi pada level 8 menggunakan filter *low-pass* dan dekomposisi *high-pass*. Setelah dekomposisi, proses selanjutnya adalah mencari koefisien perkiraan dan detail seperti yang ditunjukkan pada baris 4-19. Proses DWT pada PPG sama seperti proses DWT pada ECG, bisa ditunjukan pada Kode Sumber 4.3

```
1.  [Loo_D,Hii_D,Loo_R,Hii_R] = wfilters('db6');
2.  [C,L] = wavedec(ppgsmooth,8,Loo_D,Hii_D);
3.  [d1,d2,d3,d4,d5,d6,d7,d8]=detcoef(C,L,[1,2,3,4,5,6,7,8]);
4.  A_8 = wrcoef('a',C,L,Loo_R,Hii_R,8);
5.  A_6 = wrcoef('a',C,L,Loo_R,Hii_R,6);
```

```
A_7 = wrcoef('a',C,L,Loo_R,Hii_R,7);
A_5 = wrcoef('a',C,L,Loo_R,Hii_R,5);
A_4 = wrcoef('a',C,L,Loo_R,Hii_R,4);
A_3 = wrcoef('a',C,L,Loo_R,Hii_R,3);
A_2 = wrcoef('a',C,L,Loo_R,Hii_R,2);
A_1 = wrcoef('a',C,L,Loo_R,Hii_R,1);
D_1 = wrcoef('d',C,L,Loo_R,Hii_R,1);
D_2 = wrcoef('d',C,L,Loo_R,Hii_R,2);
D_3 = wrcoef('d',C,L,Loo_R,Hii_R,3);
D_4 = wrcoef('d',C,L,Loo_R,Hii_R,4);
D_5 = wrcoef('d',C,L,Loo_R,Hii_R,5);
D_6 = wrcoef('d',C,L,Loo_R,Hii_R,6);
D_7 = wrcoef('d',C,L,Loo_R,Hii_R,7);
D_8 = wrcoef('d',C,L,Loo_R,Hii_R,8);
```

Kode Sumber 4. 3 Kode Program DWT pada sinyal PPG

3.3.3 *Thresholding*

Nilai *threshold* dicari dari masing-masing *subband* koefisien detail dekomposisi *wavelet*. Koefisien detail di-*threshold* karena pada proses DWT dilewatkan ke batas atas filter sehingga menghasilkan sinyal yang ekstrim, dengan peningkatan dan penurunan amplitudo yang signifikan dan sangat rapat. Tujuan *Thresholding* yaitu untuk memberikan ambang batas pada sinyal, sinyal yang berada dibawah ambang batas akan diratakan (di-*set* 0), apabila lebih dari ambang batas akan dipertahankan.

```
tr = 'sqtwolog';
thr_D1 = thselect(D1,tr);
thr_D2 = thselect(D2,tr);
thr_D3 = thselect(D3,tr);
thr_D4 = thselect(D4,tr);
thr_D5 = thselect(D5,tr);
thr_D6 = thselect(D6,tr);
thr_D7 = thselect(D7,tr);
thr_D8 = thselect(D8,tr);
tD1 = wthresh(D1,'h',thr_D1);
tD2 = wthresh(D2,'h',thr_D2);
tD3 = wthresh(D3,'h',thr_D3);
tD4 = wthresh(D4,'h',thr_D4);
tD5 = wthresh(D5,'h',thr_D5);
tD6 = wthresh(D6,'h',thr_D6);
tD7 = wthresh(D7,'h',thr_D7);
tD8 = wthresh(D8,'h',thr_D8);
thr_D1_ = thselect(D_1,tr);
thr_D2_ = thselect(D_2,tr);
thr_D3_ = thselect(D_3,tr);
thr_D4_ = thselect(D_4,tr);
thr_D5_ = thselect(D_5,tr);
thr_D6_ = thselect(D_6,tr);
thr_D7_ = thselect(D_7,tr);
thr_D8_ = thselect(D_8,tr);
tD1_ = wthresh(D1,'h',thr_D1_);
tD2_ = wthresh(D2,'h',thr_D2_);
tD3_ = wthresh(D3,'h',thr_D3_);
```

```
29. tD4_ = wthresh(D4,'h',thr_D4_);
30. tD5_ = wthresh(D5,'h',thr_D5_);
31. tD6_ = wthresh(D6,'h',thr_D6_);
32. tD7_ = wthresh(D7,'h',thr_D7_);
33. tD8_ = wthresh(D8,'h',thr_D8_);
```

Kode Sumber 4. 4 Kode Program *Thresholding*

Setelah dekomposisi, koefisien *detail* di-*threshold* menggunakan *sqtwolog* dan *hard thresholding* untuk mempertahankan sinyal yang memiliki amplitudo besar. Pada *thresholding* untuk sinyal EKG dan PPG memiliki proses yang sama. Besarnya nilai threshold dari sqtwolog adalah akar dari 2 kali panjang log sinyal. Koefisien *detail threshold* dapat dilihat pada Kode Sumber 4.4 baris 2-9. *thselect* adalah fungsi dari matlab yang digunakan untuk proses *denoising* 1 dimensi. Fungsi ini akan mengembalikan nilai *threshold* sesuai dengan aturan yang digunakan, dalam hal ini menggunakan aturan *sqtwolog*. Pada baris 10-17 adalah fungsi dari *hard thresholding*. *wthresh* adalah fungsi matlab yang digunakan untuk melakukan *hard* dan *soft thresholding*. Dalam skripsi ini memakai *hard thresholding* yang digunakan untuk mempertahankan sinyal amplitudo yang tinggi.

3.3.4 Rekonstruksi Sinyal

Proses rekonstruksi koefisien *wavelet* dari sinyal yang ditransformasikan dilakukan dengan proses *up-sampling* dan penyaringan. Proses *up-sampling* adalah memasukkan nilai nol di antara masing-masing koefisien dan membuat panjangnya menjadi dua kali panjang aslinya. Selanjutnya, data di konvolusi dengan set fungsi balik skala dan *wavelet* untuk mendapatkan sinyal yang direkonstruksi.

Rekonstruksi bertujuan untuk membangun sinyal ke bentuk aslinya. Rekonstruksi sinyal dibangun dari aproksimasi dan sinyal *detail* serta rekonstruksi *low-pass filter* dan *high-pass filter* yang diperoleh dari filter DWT *Daubachies* 6. Untuk sinyal EKG dibangun dari koefisien aproksimasi A3-A7 dan detail D3, D4, D5, dan D6. Sedangkan sinyal PPG dibangun dari koefisien aproksimasi A8 dan detail D3, D4, D5, dan D6. Keluaran dari rekonstruksi sinyal adalah sinyal bersih yang akan diproses pada tahap selanjutnya. Sinyal Bersih yang dihasilkan adalah ukuran matriks 1x1000 untuk setiap EKG dan PPG. Kode Sumber 4.5 untuk mendapatkan sinyal yang bersih. Baris 1-2 adalah rekonstruksi sinyal EKG, sedangkan baris 3-4 adalah rekonstruksi sinyal PPG.

```
1. ApECG = A3 - A7;
2. ECSignal = idwt((ApECG + tD3 + tD4 + tD5 + tD6),8,Lo_R,Hi_R);
3. ApPPG = A_8;
4. PPSignal = idwt((ApPPG + tD3_ + tD4_ + tD5_ +
tD6_),8,Loo_R,Hii_R);
```

Kode Sumber 4. 5 Kode Program Rekonstruksi Sinyal ECG dan PPG

3.4 EKSTRAKSI DAN KOMPUTASI FITUR

Setelah mendapatkan sinyal yang bersih, langkah selanjutnya adalah mengekstrak fitur-fitur yang dibutuhkan untuk mendukung proses klasifikasi. Inti dari proses ekstraksi adalah untuk mendeteksi titik puncak atas dan titik bawah dari sinyal EKG dan PPG serta menghitung PTT dan PWV.

3.4.1 Deteksi Sinyal Puncak

Deteksi sinyal puncak teratas dilakukan dengan mempertimbangkan indeks tetangganya. Jika mengalami peningkatan yang tajam dan disertai dengan penurunan yang tajam maka akan terdeteksi sebagai puncaknya. Pada matlab terdapat fungsi *findpeaks* yang dapat digunakan sebagai pendeteksi puncak sinyal. *Findpeaks* adalah metode untuk mendeteksi dan menganalisis ujung atau puncak gelombang sinyal. *Findpeaks* juga dikenal sebagai *find local maxima*, karena metode ini menemukan ujung tertinggi dari gelombang atau area lokal.

Matlab telah menyediakan fungsi khusus yang digunakan untuk mengimplementasikan metode *findpeaks*, yaitu pks dan locs. Fungsi yang digunakan dengan *findpeaks* memiliki parameter data, parameter yang dimaksud adalah parameter berupa gelombang sinyal *input*. Pks adalah jumlah puncak yang terdeteksi, locs adalah lokasi setiap puncak yang terdeteksi. Sedangkan untuk mencari puncak bawah, hanya perlu melakukan invers matriks sinyal.

```
1. oneChannel = ECSignal(1:1000,2)
2. ECSignal_inverted = -ECSignal
3. oneChannel2 = ECSignal_inverted(1:1000,2)
4. ampR_ecg = findpeaks(oneChannel)
5. ampS_ecg = findpeaks(oneChannel2)
```

Kode Sumber 4. 6 Kode Program Deteksi Puncak Atas dan Bawah ECG

```
1. oneChannel3 = PPSignal(1:1000,2)
2. PPGSignal_inverted = -PPSignal
3. oneChannel4 = PPGSignal_inverted(1:1000,2)
4. ampR_ppg = findpeaks(oneChannel3)
5. ampS_ppg = findpeaks(oneChannel4)
```

Kode Sumber 4. 7 Kode Program Deteksi Puncak Atas dan Bawah PPG

3.4.2 Komputasi Fitur

Setelah proses ekstraksi berhasil, langkah selanjutnya adalah menghitung nilai PTT dengan mengurangkan indeks lokasi dari puncak PPG ke puncak ECG kemudian mengubahnya menjadi satuan detik dengan sample rate 125/detik. Ada 1000 sampel data, jadi 1000 data mewakili 8 detik. Setiap sampel mewakili 0,008 detik. Jadi untuk mengetahui nilai PTT, cukup kalikan selisih indeks lokasi dengan 0,008 detik. Nilai PTT tersebut kemudian digunakan untuk menghitung nilai PWV seperti pada persamaan 2.2.

Selanjutnya dilakukan perhitungan seperti persamaan 2.3 untuk mencari nilai sistolik dan diastolik. Selain sistol dan diastol, untuk fitur lain seperti rata-rata puncak atas dan bawah EKG dan PPG juga dihitung. Jadi jumlah atribut atau *feature* yang digunakan untuk proses klasifikasi adalah 8 yaitu sistol, diastolik, puncak atas EKG, puncak bawah EKG, puncak atas EKG, dan puncak bawah PPG. Proses penghitungan PTT, PWV, sistolik, dan diastolik seperti pada Kode Sumber 4.8. Pada baris 1-4, puncak rata-rata sinyal EKG dihitung dan PPG. Proses perhitungan PTT dapat dilihat pada baris 5-19 berdasarkan Persamaan 2.1. Pada baris 17-18 yaitu kode menghitung nilai PWV berdasarkan Persamaan 2.2 dan pada baris 19-21, nilai sistolik dan diastolik dihitung berdasarkan Persamaan 2.3.

```
1.  meanR_ECG = mean(ampR_ecg)
2.  meanS_ECG = mean(ampS_ecg)
3.  meanR_EPPG = mean(ampR_ppg)
4.  meanS_EPPG= mean(ampS_ppg)
5.  for i=1:length(meanR_ECG)
6.  for j=1:length(meanR_EPPG)
7.    if meanR_ECG (i)>meanR_EPPG (j)
8.        Selisih = (meanR_ECG (i) - meanR_EPPG (j))
9.            PTT = Selisih * 0.008
10. else
11. PTT = 0;
12.    end
13. end
14. end
15. disp('PTT=');disp(PTT');
16. PWV = 0.5*1.6/PTT;
17. disp('PWV=');disp(PWV');
18. BPsis = (0.05089855*PWV)+62.5590972;
19. disp('BPsis=');disp(BPsis');
20. BPdis = (0.04940772*PWV)+17.4800472;
21. disp('BPdis=');disp(BPdis');
```

Kode Sumber 4. 8 Kode Program Komputasi Fitur

3.5 KLASIFIKASI *RANDOM FOREST*

Data yang didapat dari proses ekstraksi bisa digunakan untuk klasifikasi menggunakan *random forest* tanpa melakukan normalisasi. Dengan membuat buat *bootstrap* sample atau pengembalian sampel Z dengan *replacement* (pengembalian) dari suatu ukuran N dari kelompok data lalu memilih m (*mtry*) variabel secara random dari variabel, dimana $m \leq p$. Setelah dilakukan pemilhian m secara random, maka pohon ditumbuhkan tanpa *pruning* (pemangkasan).

Proses tersebut dilakukan sebanyak n kali hingga terbentuk suatu *forest* sebanyak n pohon. Proses penentuan suatu kelas dilakukan dengan *aggregating* atau *majority vote*. Setelah terbentuk *forest*, kemudian dicari parameter *mtry* yang optimal sehingga diperoleh nilai misklasifikasi *error* (*Out of Bag Error*) yang stabil dan tingkat kepentingan variabel (*Variabel Importance*). Dengan diperolehnya nilai *mtry* yang optimal bisa langsung dilakukan prediksi pada data testing.