Pemrosesan Sinyal Biopotensial

Endryantoro, M. Thoriqul Aziz[1]* **Paramita, Kinar Safira Dyah [2]** **Shani, Mercya Salsabillah [3]**

[1]Teknik Biomedis, 081711733002
[2]Teknik Biomedis, 081711733003
[3]Teknik Biomedis, 081711733006
* Corresponding author's Email: m.thoriqul.aziz.endryantoro-2017@fst.unair.ac.id

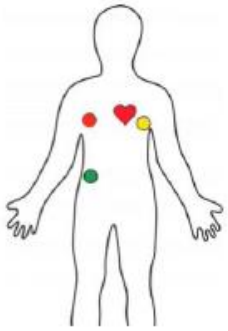
Abstrak: Biopotensial terjadi akibat adanya perbedaan kosentrasi ion dalam tubuh sehingga menyebabkan adanya potensial listrik, salah satunya adalah jantung. Nilai potensial ini dapat disadap menggunakan module mikrokontroler. Dalam penerapannya, sinyal yang didapat selalu tidak sesuai dengan sinyal yang diinginkan akibat dari adanya noise, baik dari noise kelistrikan pada alat maupun noise oleh lingkungan. Sebab itu dibutuhkan sebuah metode filtering untuk mengurangi noise sinyal, sehingga data yang muncul adalah data yang diharapkan. Metode filtering salah satunya menggunakan filter digital FIR dan IIR yang hanya menggunakan persamaan matematika untuk kemudian mengolah hasil rekam data sinyal dalam software Matlab. Pengunaan filter digital lebih mudah diaplikasikan dan tidak membutuhkan banyak perangkat meskipun meskipun filtering dilakukan secara tidak langsung. Dari hasil sinyal dengan noise yang kemudian difilter, dapat dibandingkan hasil gambaran sinyal yang paling sesuai dengan sinyal sebenarnya. Sehingga memunculkan data sinyal yang lebih akurat.

Kata Kunci: Sinyal, Noise, Domain Waktu, Domain Frekuensi, PLI, Motion Artifact, Wandering Baseline, Filter, FIR, IIR.

1. Pendahuluan

Biopotensial adalah sebuah perbedaan kosentrasi ion ion dalam tubuh yang masuk dan keluar sedemikian sehingga terjadi perbedaan potenisal listrik didalam dan diluar sel (Claveland Medical Device Inc, 2006). Salah astu pengukuran biopotensial yaitu menggunakan ECG atau elektrokardiogram yang fungsinya untuk mengukur sinyal kelistrikan jantung.

Sinyal ECG terbentu dari repolarisasi dan depolarisasi atrium dan ventrikel jantung, sehingga dalam satu kali siklus jantung mampu mengahsilkan satu gelombang P-QRS-T. Teknik penyadapan dapat dilakukan dengan menggunakan metode *Einthoven* yang hanya membutuhkan 3 buah eletroda yang dipasang sesuai gambar 1.1



Gambar 1.1 Pemasangan Elektroda ECG

Dalam pengaplikasiannya, pengambilan sinyal jantung kini dapat dilakukan dengan module Arduino sederhana dan dapat langsung diolah dalam computer meski tingkat keakuratan rendah. Pengolahan sinyal jantung secara umum dilakukan didalam domain frekuensi.

Akan tetapi dalam pengambilan sinyal jantung, alat sering kali masih menemukan noise/ ganguan terhadap sinyal yang dibaca. Noise tersebut bisanya diakibatkan oleh frekuensi tegangan listrik PLN atau

biasa disebut dengan Power Line Interference(PLI). Beberapa noise lain yaitu adanya Motion Artifact yang diakibatkan oleh pergerakan sinyal biopotensial yang lain disekitar jantung yang kemudian menggangu frekuensi sadapan sinyal dan juga Wandering Baseline yang disebabkan karena fluktuasi secara tiba tiba listrik dc yang digunakan dalam perangkat.

Untuk mengani noise tersebut, diperlukan sebuah filter untuk memotong atau mengurangi noise yang tidak dibutuhkan dalam sinyal Secara perangkat, filtering dapat dilakukan dengan 2 cara yaitu filter analog dan filter digital. Filter analog mengandalakan skema rangkaian RLC sedemiakian sehingga menghasilkan nilai frekuensi cut off filter tertentu, bergantung dari besar nilai RLC yang digunakan. Sedangkan filter digital dilakukan dengan metode komputasi pada software tertentu dengan menggunakan persamaan matematika filtering. Salah satau software yang sering digunakan adalah Matlab.

Beberapa contoh filter digital sederhana yaitu Finite Impulse Response(FIR) dan Infinite Impulse Response(IIR). Kelebihan penggunaan filter ini yaitu lebih mudah diaplikasikan dan juga terdapat syntax fungsi pada matlab. Rumus matematika FIR dapat direpreentasikan sebagai berikut :

$$y[n] = \sum_{k=0}^{N-1} h[k].x[n-k] \ (Eq.1)$$

Sedeangkan, rumus matematika IIR dapat direpresentasika sebagai berikut:

$$y[n] = \sum_{k=0}^{N-1} b_k.x[n-k] - \sum_{k=1}^{M} a_k.y[n-k] \ (Eq.2)$$

Dari metode filtering tersebut, diharapkan data hasil penyadapan sinyal ECG dapat diminimalisir nilai noise-nya dan juga bertujuan untuk mengetahui filter yang cocok dilihat dari kondisi noise serta dapa mebandingkan antara filter satu dengan yang lain.

2. Bahan dan Metode

Alat yang digunakan adalah mikrokontroler Teensy 3.5 yang kemudian terintegrasi dengan software Arduino IDE yang terisntal dengan modul ECG sederhana yaitu AD8232. Dua alat tersebut kemudain dirangkai dan disambungkan pada laptop yang sudah terisntal software sebelumnya. Kemudian pada bagian sensor dipasang Electroda Ag-Cl sebagai sensor penangkap gelombang biopotensial yaitu pada jantung dengan skema peletakan elektroda seperti pada gambar 1.1. Dari software Arduino IDE kemudian diberikan kode pemrograman untuk memberikan perintah kepada Teensy dan modul ECG untuk mengambil gambaran sinyal jantung. Dari gambaran tersebut, kemudian nilai gelombang sinyal direkam oleh software Tera Term sampai selang waktu 10 detik dengan kondisi resipien jantung stabil.

Dari data yang telah direkam dalam software Tera Term, kemudian data akan disimpan dalam bentuk csv dan dapat dibuka dengan Microsoft Excel. Kemudian buka program Matlab yang sudah terinstal untuk kemudian mengolah sinyal. Pengolahan sinyal pertama kali dilakukan yaitu melakukan ploting terlebih dahulu pada software Matlab dengan kode program tertentu. Setelah itu sinyal kemudian diberikan noise dari pengolahan sinyal pada Matlab yaitu Power Line Interference(PLI), Motion Artifact Noise, dan Wandering Baseline Noise. Hasil sinyal yang sudah diberikan noise kemudian disimpan kembali dalam bentuk csv.

Percobaan berikutnya yang dilakukan yaitu melakukan digital filtering menggunakan filter FIR dan IIR. FIR(Finite Impulse Response) dilakukan dengan menggunakan matematika yang menyesuaikan dengan rumus umum dari filter FIR. Sedangkan filter IIR(Infinite Impulse Response) menggunakan syntax khusus pada matlab. Penentuan penggunaan filter memperhatikan nilai frekuensi yang akan di filter, frekuensi sampling, dan panjang data yang terekam.

3. Data dan Analisis Hasil Pengamatan

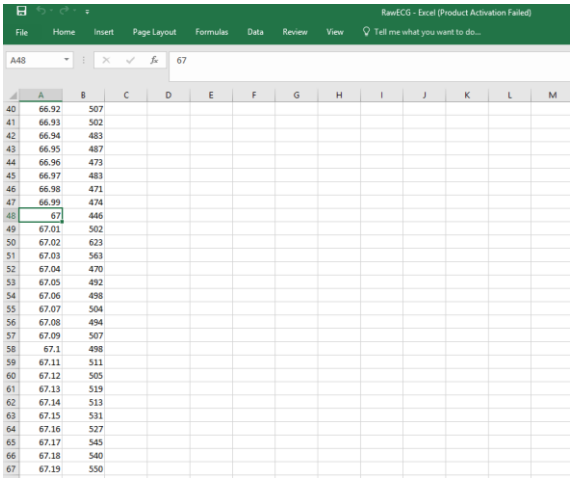
3.1 Pengambilan Data Sinyal ECG

Kode program untuk pengambilan sinyal janutng pada Arduino IDE:

```
void setup() {
    Serial.begin(9600);
}
void loop() {
  int sensorValue =
analogRead(A0);
  float t=millis()/1000.00;
```

```
Serial.print(t);
Serial.print(",");
Serial.println(sensorValue);

delay(10);
```

Data hasil perekaman sinyal jantung dalam bentuk csv:



	A	B
40	66.92	507
41	66.93	502
42	66.94	483
43	66.95	487
44	66.96	473
45	66.97	483
46	66.98	471
47	66.99	474
48	67	446
49	67.01	502
50	67.02	623
51	67.03	563
52	67.04	470
53	67.05	492
54	67.06	498
55	67.07	504
56	67.08	494
57	67.09	507
58	67.1	498
59	67.11	511
60	67.12	505
61	67.13	519
62	67.14	513
63	67.15	531
64	67.16	527
65	67.17	545
66	67.18	540
67	67.19	550

Gambar 3.1 Hasil Rekam Data dalam Format csv

Dari hasil rekaman data diperoleh frekuensi sampling 100 Hz dengan panjang data 6495

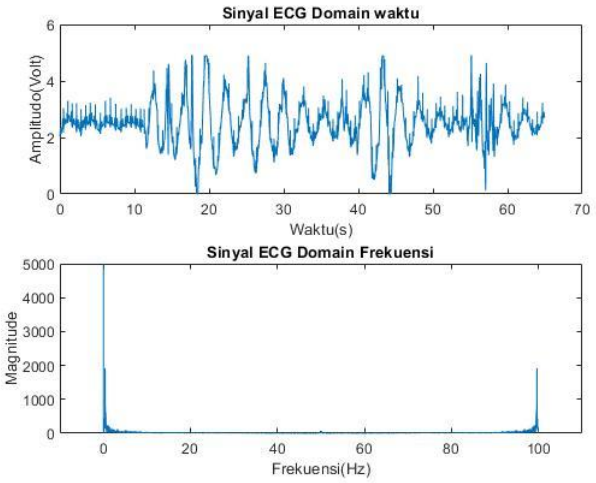
3.2 Penambahan Noise Sinyal ECG dan Penerapan Fast Fourier Transform(FFT)

a) Penerapan FFT

Kode Program dalam penambahan FFT pada raw sinyal pada program Matlab R2018a:

```
RawECG =xlsread('RawECG.csv');
sinyal=RawECG(:,2)/1023.00*5.0;
fs=100; % frekuensi sampling 100
L=length(sinyal); %panjang data
waktu=(0:L-1)/fs;
%-------Plot Waktu---------
subplot(2,1,1)
plot(waktu,sinyal);
title("Sinyal ECG Domain
waktu");
xlabel("Waktu(s)");
ylabel("Amplitudo(Volt)");
%----FFT dan Plot Frekuensi---
frek=(0:L-1)*fs/L;
B=abs(fft(sinyal));
subplot(2,1,2)
plot(frek,B)
title("Sinyal ECG Domain
Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
```

Gambaran grafik sinyal setelah penambahan FFT:



Gambar3.2 Gambaran Sinyal Hasil Rekaman Dalam Domain Waktu dan Frekuensi

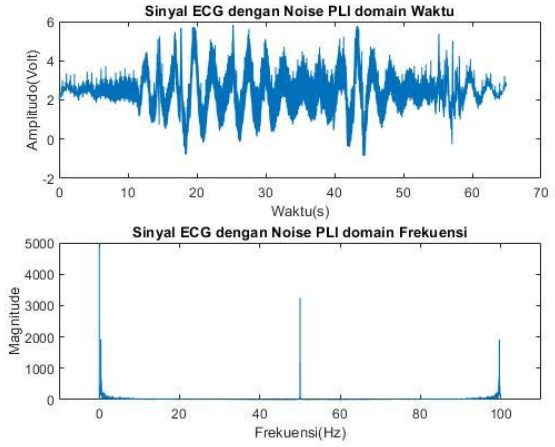
b) Penambahan noise PLI

Kode Program dalam penambahan PLI pada raw sinyal pada program Matlab R2018a:

```
RawECG =xlsread('RawECG.csv');
sinyal=RawECG(:,2)/1023.00*5.0;
fs=100; % frekuensi sampling 100
L=length(sinyal); %panjang data
time=L/fs;
waktu=linspace(0,time,L);
S = sin(2*pi*50*waktu); % noise
pli
ss=transpose(S);%menyamakan
matriks
noise=sinyal+ss;
%---- SINYAL SETELAH NOISE -----
hold on
figure(1)
subplot(2,1,1)
plot(waktu,noise);
title("Sinyal ECG dengan Noise
PLI domain Waktu");
xlabel("Waktu(s)");
ylabel("Amplitudo(Volt)");
%-- SINYAL DOMAIN FREKUENSI--
B=abs(fft(noise));
frek=(0:L-1)*fs/L;
```

```
subplot(2,1,2)
plot(frek,B)
title("Sinyal ECG dengan Noise
PLI domain Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
csvwrite("file_noise_pli.csv",no
ise);%save noise
```

Gambaran grafik perbedaan sinyal setelah ditambah noise:



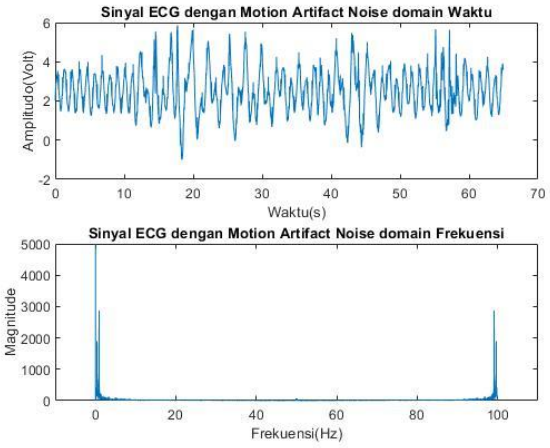
Gambar 3.3 Gambaran Sinyal yang Ditambahkan Noise PLI dalam Domain Waktu dan Frekuensi

c) Penambahan Motion Artifact Noise

Kode Program dalam penambahan Motion Artifact Noise pada raw sinyal pada program Matlab R2018a:

```
RawECG =xlsread('RawECG.csv');
sinyal=RawECG(:,2)/1023.00*5.0;
fs=100; % frekuensi sampling 100
L=length(sinyal); %panjang data
time=L/fs;
waktu=linspace(0,time,L);
rdom=rand(1,1); %noise Motion
Artifact
S = sin(2*pi*rdom*waktu); %noise
Motion Artifac
ss=transpose(S);
noise=sinyal+ss;
%--- SINYAL SETELAH NOISE ------
hold on
figure(1)
subplot(2,1,1)
plot(waktu,noise);
title("Sinyal ECG dengan Motion
Artifact Noise domain Waktu");
xlabel("Waktu(s)");
ylabel("Amplitudo(Volt)");
%------- SINYAL DOMAIN
FREKUENSI-----
B=abs(fft(noise));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,B)
title("Sinyal ECG dengan Motion
Artifact Noise domain
Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
csvwrite("file_noise_MA.csv",noi
se);
```

Gambaran grafik perbedaan sinyal setelah ditambah noise:



Gambar 3.4 Gambaran Sinyal yang Ditambahkan Motion Artifact Noise dalam Domain Waktu dan Frekuensi

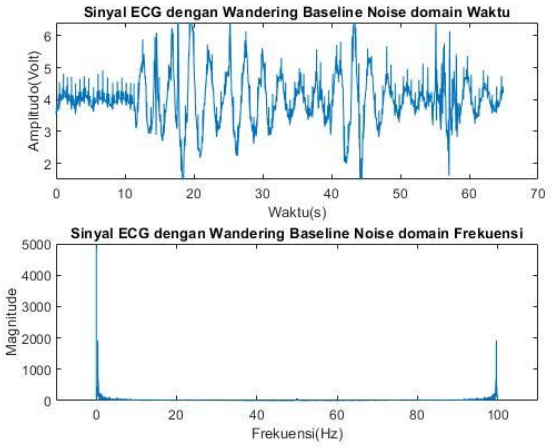
d) Penambahan Wandering Baseline Noise

Kode Program dalam penambahan Wandering Baseline Noise pada raw sinyal pada program Matlab R2018a:

```
RawECG =xlsread('RawECG.csv');
sinyal=RawECG(:,2)/1023.00*5.0;
fs=100; % frekuensi sampling 100
L=length(sinyal); %panjang data
time=L/fs;
waktu=linspace(0,time,L);
ss=1.5; %Noise Wandering
Baseline
```

```
noise=sinyal+ss;
%---- SINYAL SETELAH NOISE -----
hold on
figure(1)
subplot(2,1,1)
plot(waktu,noise);
title("Sinyal ECG dengan
Wandering Baseline Noise domain
Waktu");
xlabel("Waktu(s)");
ylabel("Amplitudo(Volt)");
%------- SINYAL DOMAIN
FREKUENSI----
B=abs(fft(noise));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,B)
title("Sinyal ECG dengan
Wandering Baseline Noise domain
Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
csvwrite("file_noise_MA.csv",noi
se);
```

Gambaran grafik perbedaan sinyal setelah ditambah noise:



Gambar 3.5 Gambaran Sinyal yang Ditambahkan Wandering Baseline Noise dalam Domain Waktu dan Frekuensi

3.3 Penerapan Filter FIR dan IIR

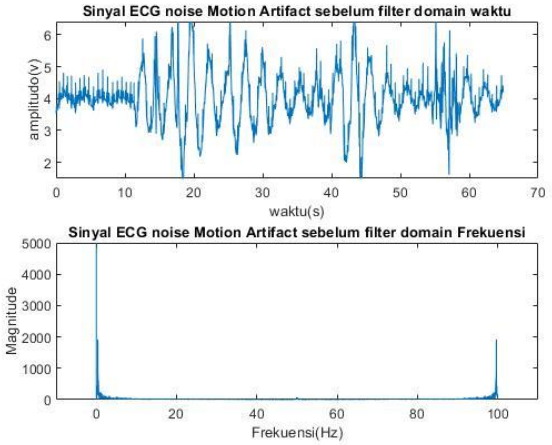
a) Filter FIR Highpass untuk Motion Artifact Noise

Kode Program filter FIR Highpass untuk Motion Artifact Noise pada program Matlab R2018a:

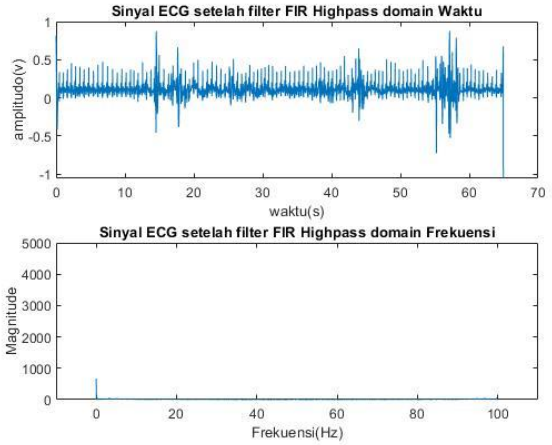
```
sinyal
=xlsread('file_noise_MA.csv');
fs=100; % frekuensi sampling 100
L=length(sinyal); %panjang data
time=L/fs;
waktu=linspace(0,time,L);
fc=3;
N=57; %keoefisien filter belum
jadi
fcn=fc/fs; %normalisasi
for k=1:N
    n=k-(N-1)/2;
    if n==0
        hd(k)=1-(2*fcn);
    else
        hd(k)=-
sin(2*pi*fcn*n)/(pi*n);
    end
end
nn=[1:N];
wn=0.42-
0.5*cos(2*pi*nn/100)+0.08*cos(4*
pi*nn/100);
h=hd.*wn;
hk=conv(sinyal,h,'same');
%-----sebelum----
hold on
figure(1)
subplot(2,1,1)
plot(waktu,sinyal);
title("Sinyal ECG noise Motion
Artifact sebelum filter domain
waktu");
xlabel("waktu(s)");
ylabel("amplitudo(v)");
%-----fft------
B=abs(fft(sinyal));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,B)
title("Sinyal ECG noise Motion
Artifact sebelum filter domain
Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
%----------setelah-----
hold on
figure(2)
subplot(2,1,1)
plot(waktu,hk);
title("Sinyal ECG setelah filter
FIR Highpass domain Waktu");
xlabel("waktu(s)");
ylabel("amplitudo(v)");
```

```
%------fft---------
C=abs(fft(hk));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,C)
title("Sinyal ECG setelah filter
FIR Highpass domain Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
```

Gambaran grafik perbedaan sinyal:



Gambar 3.6 Sinyal Noise Motion Artifact Sebelum Difilter dalam Domain Waktu dan Frekuensi



Gambar 3.7 Sinyal Setelah Difilter Menggunakan FIR Highpass dalam Domain Waktu dan Frekuensi

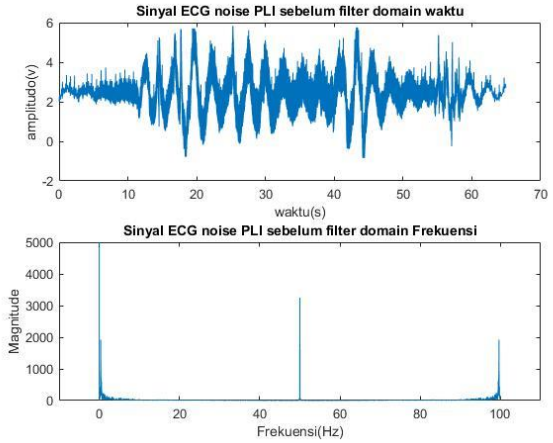
b) Filter FIR Bandstop untuk Power Line Interference(PLI)

Kode Program filter FIR Bandstop untuk Power Line Interference(PLI) pada program Matlab R2018a:

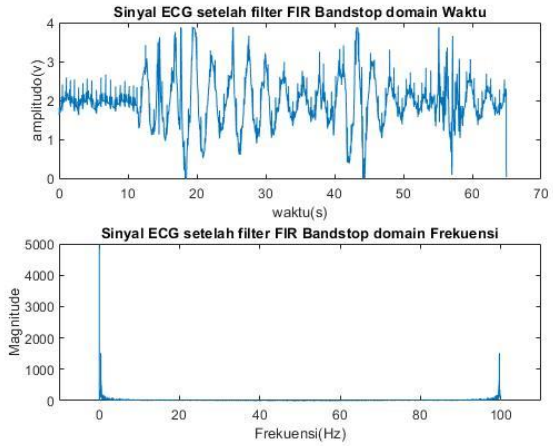
```
sinyal
=xlsread('file_noise_pli.csv');
fs=100; % frekuensi sampling 100
L=length(sinyal); %panjang data
time=L/fs;
waktu=linspace(0,time,L);
fk=48/fs;
fl=50/fs;
N=77; %keoefisien filter belum
jadi
for k=1:N
    n=k-(N-1)/2;
    if n==0
        hd(k)=1-(2*fl-2*fk);
    else

hd(k)=(sin(2*pi*fk*n)/(pi*n))-
(sin(2*pi*fl*n)/(pi*n));
    end
end
nn=[1:N];
wn=0.42-
0.5*cos(2*pi*nn/100)+0.08*cos(4*
pi*nn/100);
h=hd.*wn;
hk=conv(sinyal,h,'same');
%-----sebelum----
hold on
figure(1)
subplot(2,1,1)
plot(waktu,sinyal);
title("Sinyal ECG noise PLI
sebelum filter domain waktu");
xlabel("waktu(s)");
ylabel("amplitudo(v)");
%-----fft------
B=abs(fft(sinyal));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,B)
title("Sinyal ECG noise PLI
sebelum filter domain
Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
%----------setelah-----
hold on
figure(2)
subplot(2,1,1)
plot(waktu,hk);
title("Sinyal ECG setelah filter
FIR Bandstop domain Waktu");
xlabel("waktu(s)");
ylabel("amplitudo(v)");
```

```
%------fft---------
C=abs(fft(hk));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,C)
title("Sinyal ECG setelah filter
FIR Bandstop domain Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
```

Gambaran grafik perbedaan sinyal:



Gambar 3.8 Sinyal Noise PLI Sebelum Difilter dalam Domain Waktu dan Frekuensi



Gambar 3.9 Sinyal Setelah Difilter Menggunakan FIR Bandstop dalam Domain Waktu dan Frekuensi

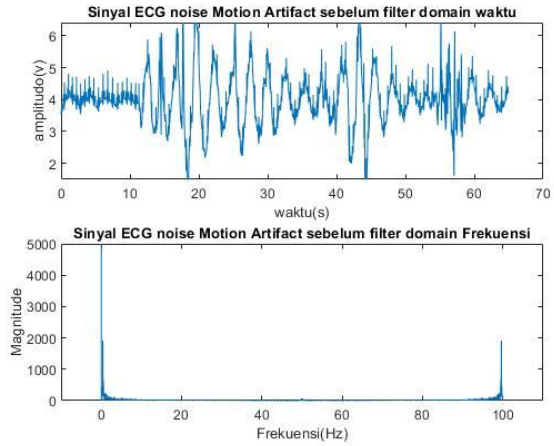
c) Filter IIR Highpass untuk Motion Artifact Noise

Kode Program filter IIR Highpass untuk Motion Artifact Noise pada program Matlab R2018a:

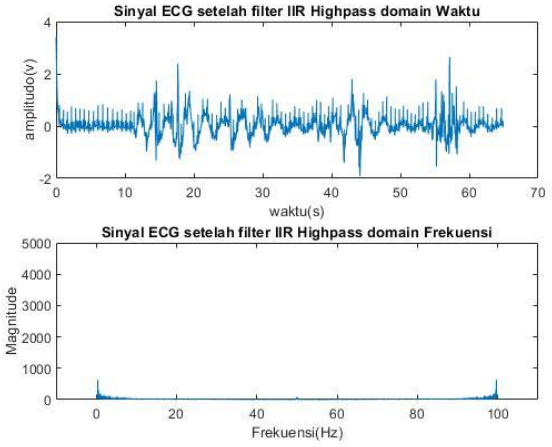
```
sinyal
=xlsread('file_noise_MA.csv');
fs=100; % frekuensi sampling 100
L=length(sinyal); %panjang data
time=L/fs;
waktu=linspace(0,time,L);
fc=3;
N=1;
%------ sebelum------
hold on
figure(1)
subplot(2,1,1)
plot(waktu,sinyal)
title("Sinyal ECG noise Motion
Artifact sebelum filter domain
waktu");
xlabel("waktu(s)");
ylabel("amplitudo(v)");
%-----fft------
B=abs(fft(sinyal));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,B)
title("Sinyal ECG noise Motion
Artifact sebelum filter domain
Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
%--filter IIR Highpass
Highpass=designfilt('highpassiir
','FilterOrder',N,'PassbandFrequ
ency',fc,'PassbandRipple',0.5,'S
ampleRate',fs);
y=filter(Highpass,sinyal);
%-------setelah-----
hold on
figure(2)
subplot(2,1,1)
plot(waktu,y)
title("Sinyal ECG setelah filter
IIR Highpass domain Waktu");
xlabel("waktu(s)");
ylabel("amplitudo(v)");
%------fft---------
C=abs(fft(y));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,C)
title("Sinyal ECG setelah filter
IIR Highpass domain Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
```

```
hold off
```

Gambaran grafik perbedaan sinyal:



Gambar 3.10 Sinyal Noise Motion Artifact Sebelum Difilter dalam Domain Waktu dan Frekuensi



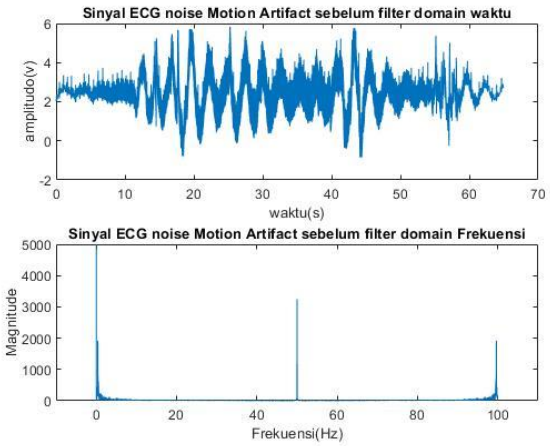
Gambar 3.11 Sinyal Setelah Difilter Menggunakan IIR Highpass dalam Domain Waktu dan Frekuensi

d) Filter IIR Bandstop untuk Power Line Interference(PLI)

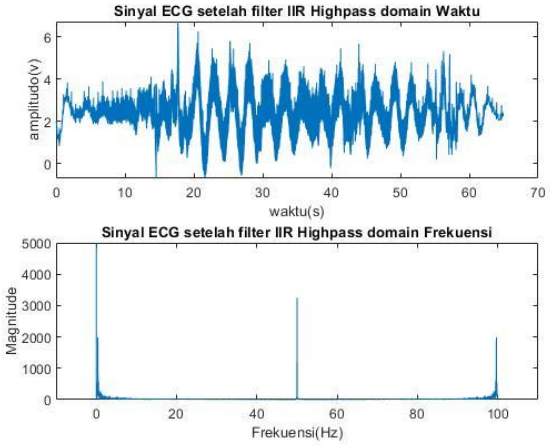
Kode Program filter IIR Bandstop untuk Power Line Interference(PLI) pada program Matlab R2018a:

```
sinyal
=xlsread('file_noise_pli.csv');
fs=100; % frekuensi sampling 100
L=length(sinyal); %panjang data
time=L/fs;
waktu=linspace(0,time,L);
fk=48/fs;
fl=50/fs;
N=58;
%------ sebelum------
hold on
figure(1)
subplot(2,1,1)
plot(waktu,sinyal)
title("Sinyal ECG noise Motion
Artifact sebelum filter domain
waktu");
xlabel("waktu(s)");
ylabel("amplitudo(v)");
%-----fft------
B=abs(fft(sinyal));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,B)
title("Sinyal ECG noise Motion
Artifact sebelum filter domain
Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
%----------filter IIR Highpass
bandstop=designfilt('bandstopiir
','FilterOrder',N,'HalfPowerFreq
uency1',fk,'HalfPowerFrequency2'
,fl,'SampleRate',fs);
y=filter(bandstop,sinyal);
%-------setelah-----
hold on
figure(2)
subplot(2,1,1)
plot(waktu,y)
title("Sinyal ECG setelah filter
IIR Highpass domain Waktu");
xlabel("waktu(s)");
ylabel("amplitudo(v)");
%------fft---------
C=abs(fft(y));
frek=(0:L-1)*fs/L;
subplot(2,1,2)
plot(frek,C)
title("Sinyal ECG setelah filter
IIR Highpass domain Frekuensi");
xlabel("Frekuensi(Hz)");
ylabel("Magnitude");
xlim([-10 110]);
ylim([0 5000]);
hold off
```

Gambaran grafik perbedaan sinyal:



Gambar 3.12 Sinyal Noise PLI Sebelum Difilter dalam Domain Waktu dan Frekuensi



Gambar 3.11 Sinyal Setelah Difilter Menggunakan IIR Bandstop dalam Domain Waktu dan Frekuensi

4. Pembahasan

Pada percobaan pertama yaitu pengambilan data sinyal biopotensial dari jantung menggunakan module ECG AD8232 yang terkoneksi dengan Arduino Teensy. Alasan penggunaan Arduino Teensy yaitu kecepatan dan kemampuan olah data yang baik dari hardware sehingga tidak terdapat data yang hilang. Pada program Arduino IDE, menggunakan kode pemrograman analog read pada example dengan menambahkan variable t sebagai penunjuk waktu yang mana nilainya dibagi 1000 agar waktu yang ditunjukkan sama dengan waktu pengambilan data oleh mikrokontroler. Kemudian data akan mulai disimpan pada Software Tera Term saat software dimulai. Hasil rekaman disimpan dalam bentuk csv dengan 2 kolom yaitu waktu dan besar amplitude sinyal seperti pada gambar 3.1

Dari hasil data yang telah disimpan, kemudian dilakukan pengolahan pada software Matlab dengan menunjukan grafiks sinyal dalam domain waktu dan domain frekuensi. Diperolah hasil seperti pada gambar.. yang menunjuka bahwa pengambilan *raw* sinyal kurang begitu baik karena sinyal terlihat tidak stabil. Sinyal pada gambar 3.2 dalam domain waktunya terlihat stabil pada waktu awal. Dan nilai frekuensi yang sering muncul adalah berkisar dibawah 10 Hz dan di 100 Hz. Artinya pengambilan raw sinyal sudah terjadi noise yang frekuensinya relative kecil yaitu dibawah 10 Hz.

Kemudian pada percobaan berikutnya yaitu raw data tersebut diberikan noise tambahan melewati software Matlab. Penambahan noise PLI dilakukan dengan menambahkan nilai raw sinyal dengan sinyal sinusoidal baru berfrekuensi 50 Hz, sehingga ketika domain sinyal dirubah menjadi domain frekuensi maka akan muncul magnitude sinyal pada frekuensi 50 Hz lebih tinggi dibanding dengan sebelum diberi noise seperti pada gambar 3.3. Penambahan noise berikutnya yaitu motion artifact dengan menggunakan fungsi randn pada matlab sehingga nilai raw sinyak akan ditambahkan dengan sinyal lain yang berfrekuensi tidak lebih dari 1 Hz, ditunjukan dengan magnitude frekuensi rendah hasil FFT(Fast Fourier Transform) lebih tinggi seperti pada gambar 3.4. Noise yang terakhir yaitu Wandering Baseline dimana noise ini disebabkan dari nilai offset tegangan dc demikian sehingga menggeser posisi sinyal. Dapat diamati dari gambar 3.2 dan gambar 3.5 dimana dalam 2 domain waktu tersebut, pada gambar 3.5 memiliki nilai awal sinyal yang lebih tinggi dibanding dengan gambar 3.2 . Dari hasil semua sinyal yang diberikan noise tersebut, kemudian tiap sinyalnya disimpan kembali dalam bentuk csv untuk selanjutnya masuk dalam penggunaan filter digital.

Filter digital yang digunakan adalah filter FIR(Finite Impulse Response) dan IIR(Infinite Impulse Respon). Dalam dua jenis filter tersebut, didalamnya memuat filter frekuensi yaitu lowpass,

highpass, bandpass, bandstop filter. Dalam penentuan jenis filter frekuensi, maka harus mengtahui terlebih dahulu nilai atau besar frekuensi yang akan di filtering yang mana dapat diamati dari gambaran sinyal dalam domain frekuensi. Filter pertama yaitu FIR Highpass untuk noise motion artifact. Pemilihan highpass karena nilai frekuensi motion artifac yaitu berada di rentan 0-1 Hz sedemian sehingga dipilih highpass untuk membuat filter pada batar frekuensi cut off yang ditentukan yaitu 3 Hz dan mengambil yang berada diatasnya. Hasil dari FIR Highpass ini dapat diamati pada gambar 3.7 dimana pada domain frekuensi gambar 3.6 , nilai magnitude frekuensi rendah sangat tinggi, dan kemudian pada domain frekuensi gambar 3.7 lebih rendah. Akibatnya dari kedua gambar tersebut dapat kemudian diamati sinyal dalam domain waktu dimana pada gambar 3.7 lebih bersih dari noise dibanding pada gambar 3.6

Filtering yang kedua yaitu pada sinyal PLI menggunakan FIR Bandstop karena hanya ingin memotong nilai frekuesi sinyal pada angka 50 Hz, maka dilakukan filtering digital dengan memasukan matematika program bandstop FIR dalam Matlab, dan dapat diamati pada gambar 3.8 dan gambar 3.9. Frekuensi 50 hz pada domain frekuesi gambar 3.8 menjadi berkurang pada gambar 3.9 , akibatnya keluaran sinyal pada gambar 3.9 menjadi lebih jelas dan bersih. Penggunaan nilai ordo filter juga berpengaruh dalam tingkat ketajaman filtrasi, dimana semakin tinggi ordo filter maka akan semakin tajam tingkat filtrasi. Jika terlalu tajam maka berakibat pada hilangnya data frekuensi.

Filtering selanjutnya yaitu IIR dengan tipe filter frekuensi sama dengan filter pada FIR dengan menyesuaikan kondisi noise-nya. Akan tetapi pada filter IIR ini digunakan syntax `designfilt` yang terdapat pada matlab sehingga tidak perlu lagi untuk memasukan persamaan matematika pada program. Hasil dari filter IIR highpass pada gambar 3.11 dan dibandingakan dengan gambar 3.10 memiliki tingkat filtering yang realtif tinggi dengan nilai ordo filter sama dengan 1. Jika dibandingkan dengan hasil akhir filter FIR highpass pada gambar 3.7 , maka filter FIR Highpass teramati lebih tajam karena banyak nilai frekuensi rendah yang terpotong. Yang bukan berarti gambaran sinyal domain waktu pada gambar 3.9 lebih baik dari gambar 3.11.

Sedangkan pada filter IIR Bandstop jika diamati hasil gambaran sinyal sebelum dan sesudah filter pada gambar 3.12 dan gambar 3.13, maka dapat diamati bahwa filter tidak bekerja pada sinyal sehingga keluaran filtering masih tetap sama. Dapat diamati lebih jelas yaitu pada domain frekuensi sinyal sesudah dan sebelum filter IIR Bandstop, dimana nilai magnitude frekuensi 50 Hz masih relatif tinggi, jika dibandingkan dengan hasil filter FIR Bandstop pada gambar 3.9 .Kesalahan ini bisa diakibatkan karena kesalahan memberikan kode pemrograman sedemikian sehingga hasil keluaran tidak menunjukkan semestinya.

5. Kesimpulan

Sinyal Biopotensial secara khusus adalah sinyal jantung dapat diamati karena memiliki nilai potensial listrik yang relative tinggi dengan gambaran sinyal stabil. Akan tetapi dalam faktanya, dalam pengambilan sinyal masih terdapat noise yang juga merusak gambaran sinyal. Maka untuk melakukan filtering noise tersebut dapat dilakukan dengan menggunakan filter digital menggunakan software yang salah satunya adalah Matlab dimana hal ini lebih mudah diterapkan karena tidak membutuhkan banyak perangkat jika dibandingkan dengan filter analog. Pemilihan jenis filter juga harus menyesuaikan dengan frekuensi noise sinyal yang akan difilter.

References

[1] John Semmlow.2018.**"Circuits, Signals, and Systems for Bioengineers"**. Elsevier Inc. London, United Kingdom.

[2] Rahmat Maulana Yasin, Abdullah Nur Aziz, Hartono. 2018. **"Rancang Bangun Sistem Kontrol Berbasis Biopotensial Mata(Studi Kasus : Mengontrol Aplikasi Berbasis Android)"**. Fisika FMIPA Unsoed. Diakses pada tanggal 11 September 2019.