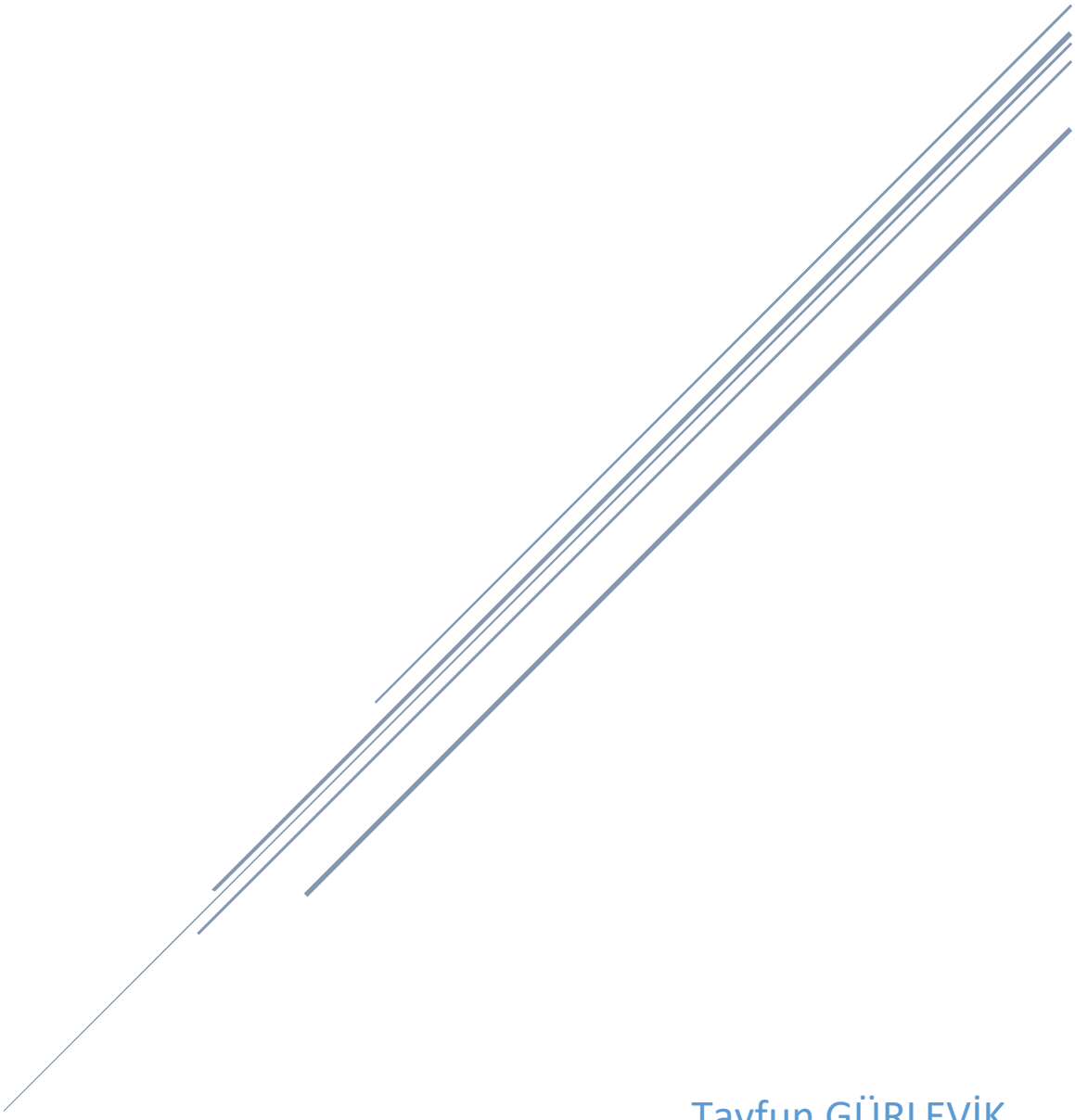
BCA607 HAREKET ANALİZİ SİSTEMLERİ

Final Raporu



Tayfun GÜRLEVİK
N19139647

İçindekiler

1. Kısım 1 1
2. Kısım 2 7

Grafik 1 Markerların yörüngesi 3
Grafik 2 M1, M3, ve M5 markerları ile oluşturulan rigidbody 4
Grafik 3 Rigidbody Roll açısı 6
Grafik 4 Rigidbody Pitch Açısı 6
Grafik 5 Rigidbody Yaw Açısı 7
Grafik 6 Bileşke ivme 8
Grafik 7 Bileşke ivme(Yerçekimsiz) 8
Grafik 8 İvmenin FFT specturumu 9
Grafik 9 Filtrelenmis ivme(Yercekimsiz) 9
Grafik 10 Velocity with drift 10
Grafik 11 Position with drift 10
Grafik 12 Velocity high pass filtered 11
Grafik 13 Position no-drift 11

1. Kısım 1

İlk olarak 13. Haftanın dosyalarında bulunan vicon_all_labelled.txt dosyasının matlab programında kullanbilecek şekilde virgülle ayrılmış formata dönüştürülmesi gerekmektedir. Bu amaçla C# yazılım dili kullanarak ilgili dosyadaki M1,M2,M3,M4 ve M5 isimleri ile etiketlenmiş markerların pozisyon bilgisini başka bir dosyaya yazdıracak program geliştirilmiştir.

Marker bilgilerini virgülle ayrılmış formatta parse eden programın kodu aşağıdaki gibidir:

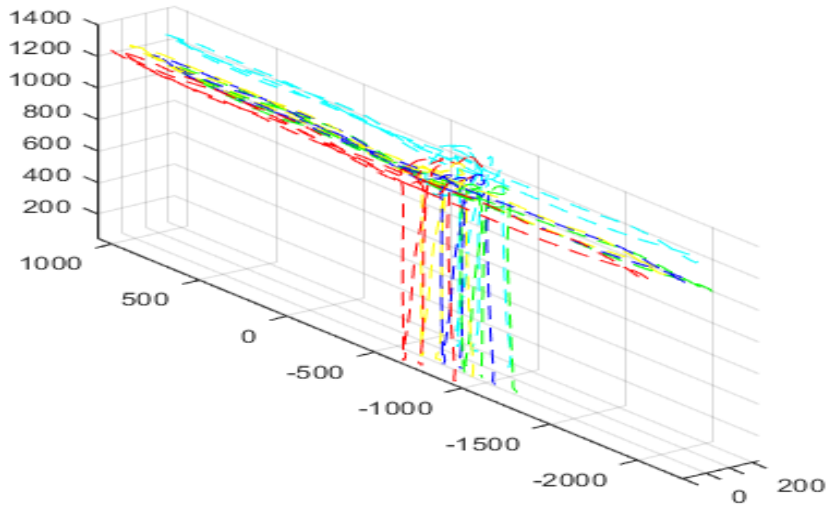
```
using System;
using System.IO;

namespace ViconParser
{
    class Program
    {
        static void Main(string[] args)
        {
            string filepath = "vicon_all_labelled.txt";
            using (StreamReader reader = new StreamReader(filepath))
            {
                using (StreamWriter writer = new StreamWriter("vicon_parsed.txt"))
                {

                    do
                    {
                        string line = reader.ReadLine();
                        if (line.Contains("Frame Number:"))
                        {
                            int frameNumber = int.Parse(line.Split(':')[1].Trim());
                            Console.WriteLine(frameNumber);
                            writer.Write(frameNumber + ",");

                        }
                        if (line.Contains("Markers (5):"))
                        {
                            string marker1Line = reader.ReadLine();
                            int startIndex = marker1Line.IndexOf('(');
                            int endIndex = marker1Line.IndexOf(')');
                            string marker1Pozitions = marker1Line.Substring(startIndex +
1, endIndex - startIndex - 1);
                            Console.WriteLine(marker1Pozitions);
                            string marker2Line = reader.ReadLine();
                            startIndex = marker2Line.IndexOf('(');
                            endIndex = marker2Line.IndexOf(')');
                            string marker2Pozitions = marker2Line.Substring(startIndex +
1, endIndex - startIndex - 1);
                            Console.WriteLine(marker2Pozitions);
                            string marker3Line = reader.ReadLine();
                            startIndex = marker3Line.IndexOf('(');
                            endIndex = marker3Line.IndexOf(')');
                            string marker3Pozitions = marker3Line.Substring(startIndex +
1, endIndex - startIndex - 1);
                            Console.WriteLine(marker3Pozitions);
                            string marker4Line = reader.ReadLine();
                            startIndex = marker4Line.IndexOf('(');
                            endIndex = marker4Line.IndexOf(')');
```

```
                            string marker4Pozitions = marker4Line.Substring(startIndex +
1, endIndex - startIndex - 1);
                            Console.WriteLine(marker4Pozitions);
                            string marker5Line = reader.ReadLine();
                            startIndex = marker5Line.IndexOf('(');
                            endIndex = marker5Line.IndexOf(')');
                            string marker5Pozitions = marker5Line.Substring(startIndex +
1, endIndex - startIndex - 1);
                            Console.WriteLine(marker5Pozitions);
                            writer.WriteLine("{0},{1},{2},{3},{4}", marker1Pozitions,
marker2Pozitions,
                                marker3Pozitions, marker4Pozitions, marker5Pozitions);
                        }
                    } while (!reader.EndOfStream);
                }
            }

        }
    }
}
```

Program dosyası ViconParser.zip içerisinde bulunabilir.

Daha sonra elde edilen virgülle ayrılmış formattaki dosya kullanılarak matlabda ilgili değişkenlere markerların pozisyon bilgileri aktarılmıştır. Elde edilen sonuç Grafik 1'de gösterilmiştir.

```
%% vicon dosyasindan labellarin pozisyonlarini okuma islemi

load 'vicon_parsed.txt';
frames=vicon_parsed(:,1);
M1=vicon_parsed(:,2:4);
M2=vicon_parsed(:,5:7);
M3=vicon_parsed(:,8:10);
M4=vicon_parsed(:,11:13);
M5=vicon_parsed(:,14:16);
%% Yorunge cizdirme islemi
plot3(M1(:,1),M1(:,3),M1(:,2),'--r')
grid on
axis equal
hold on
plot3(M2(:,1),M2(:,3),M2(:,2),'--g')
plot3(M3(:,1),M3(:,3),M3(:,2),'--b')
plot3(M4(:,1),M4(:,3),M4(:,2),'--y')
plot3(M5(:,1),M5(:,3),M5(:,2),'--c')
hold off
```

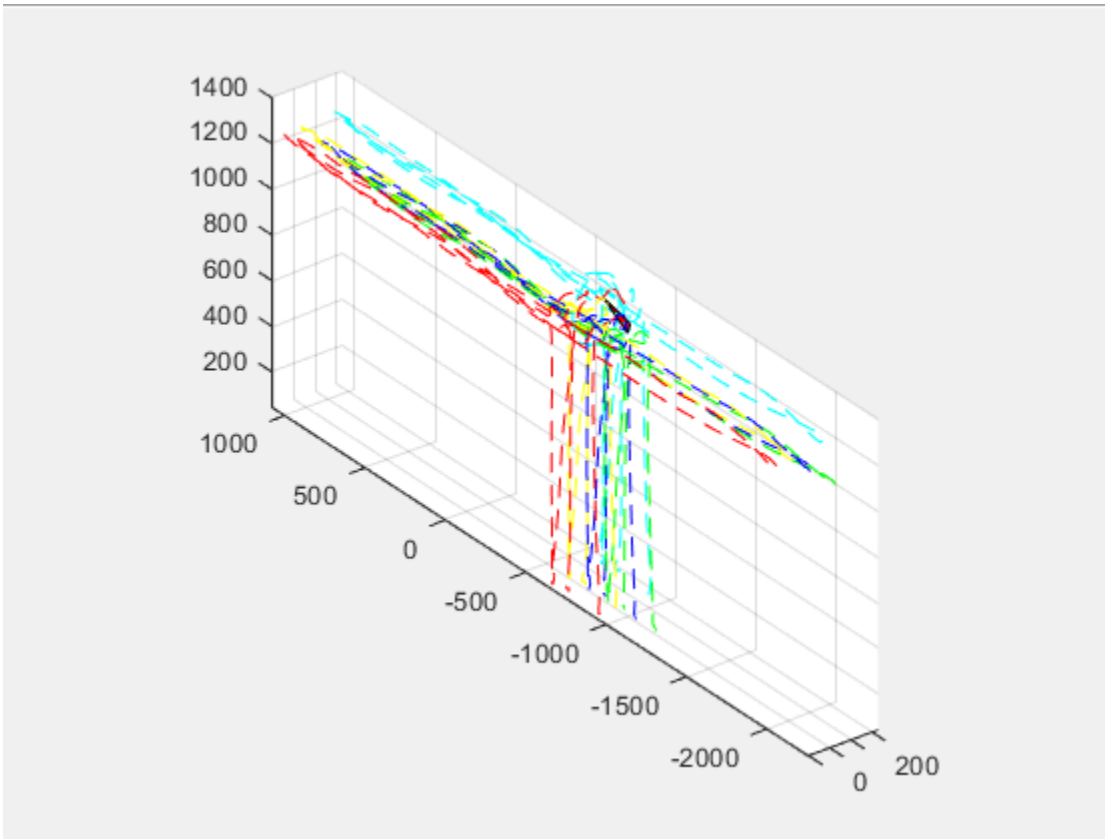


Grafik 1 Markerların yörüngesi

M1, M3, M5 markerlarının pozisyon bilgisi kullanılarak fill3 komutu yardımıyla bir poligon (Grafik 2) oluşturulmuş ve animasyonu final.mp4 dosyasına kaydedilmiştir.

```
%% Video olusturma islemi

video=true;
if video
   aviObj=VideoWriter('final.mp4','MPEG-4');
   aviObj.FrameRate=25;
   open(aviObj);
end

for i=1:length(frames)
    plot3(M1(:,1),M1(:,3),M1(:,2),'--r')
    grid on
    axis equal
    hold on
    plot3(M2(:,1),M2(:,3),M2(:,2),'--g')
    plot3(M3(:,1),M3(:,3),M3(:,2),'--b')
    plot3(M4(:,1),M4(:,3),M4(:,2),'--y')
    plot3(M5(:,1),M5(:,3),M5(:,2),'--c')



    X(1,1)=M1(i,1);
    X(1,2)=M3(i,1);
    X(1,3)=M5(i,1);
    Y(1,1)=M1(i,3);
    Y(1,2)=M3(i,3);
    Y(1,3)=M5(i,3);
    Z(1,1)=M1(i,2);
    Z(1,2)=M3(i,2);
```

```
    Z(1,3)=M5(i,2);


    fill3(X,Y,Z,'red');
    if video
       frame=getframe(gcf);
       writeVideo(aviObj,frame);
    end
    hold off
end

if video
   close(aviObj);
end
```



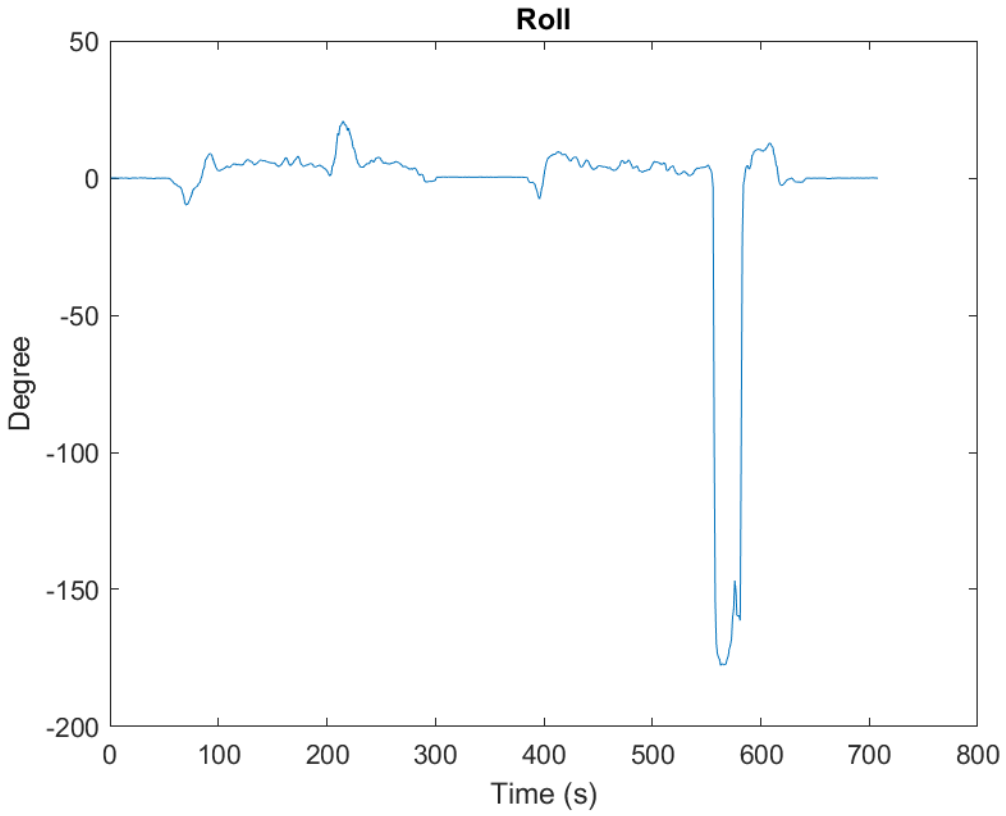
Grafik 2 M1, M3, ve M5 markerları ile oluşturulan rigidbody

M1, M3 ve M5 markerlarının poziyonlarını kullanarak bir rigid body oluşturmak amacıyla P1, P2 ve P3 noktaları tanımlanmış olup bu noktalar vasıtasıyla rigidbodynin yerel koordinat sisteminin birim vektörleri bulunmuştur.

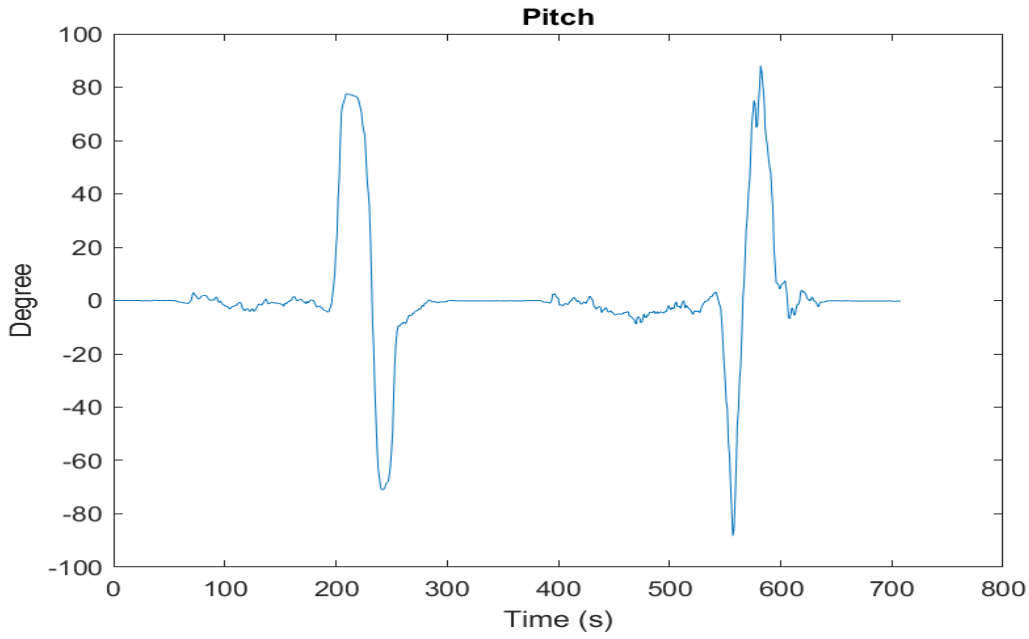
Hafta 10'da için Research Methods.pdf dosyasında anlatılan adımlar takip edilerek, Kalibrasyon çubuğunun hareketsiz durduğu ilk frame kalibrasyon karesi olarak kabul edilerek, CalCS(Kalibrasyon koordinat sistemi) ve bu matrisin tersinin , birim matris ile çarpımıyla da RTM(Rotational Transformation Matrix) hesaplanmıştır.

Her bir kare için PCS (Provisional Coordinate System) hesaplanarak RTM ile çarpımıyla SCS(segment coordinat system) hesaplanmıştır. Rotm2eul komutu yardımıyla SCS için euler açıları hesaplanmış, elde edilen vektörün bileşenleri ile de roll(Grafik 3), pitch(Grafik 4) ve yaw(Grafik 5) değerleri hesaplanarak grafikleri elde edilmiştir.

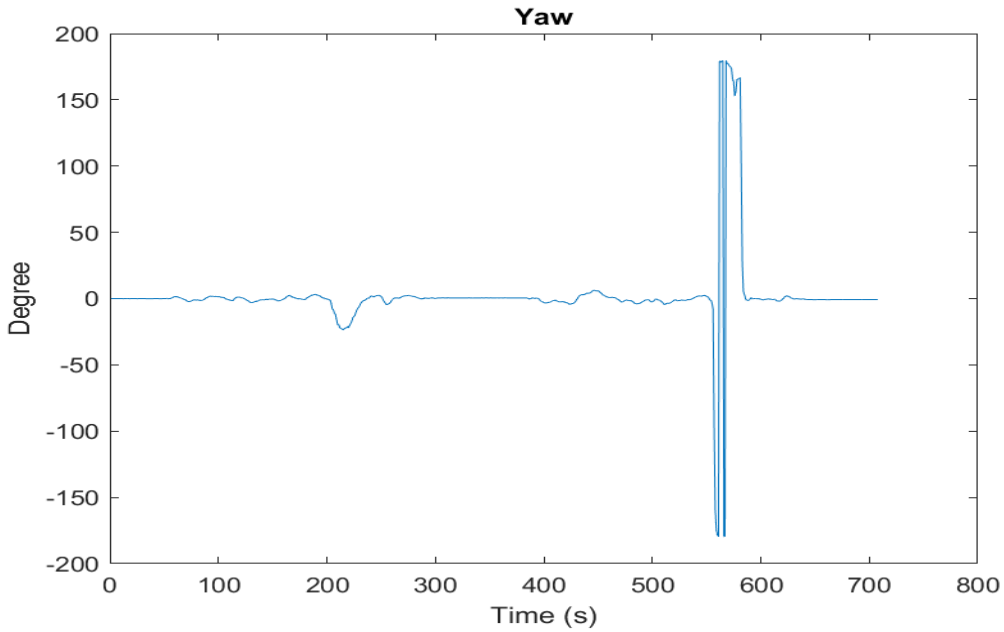
```
%% Roll,pitch ve yaw degerlerinin hesaplanmasi
roll=zeros(1,length(frames));
pitch=zeros(1,length(frames));
yaw=zeros(1,length(frames));
for n=1:length(frames)
    P1=M1(n,:);
    P2=M3(n,:);
    P3=M5(n,:);
    [P1(1,2),P1(1,3)]=deal(P1(1,3),P1(1,2));
    [P2(1,2),P2(1,3)]=deal(P2(1,3),P2(1,2));
    [P3(1,2),P3(1,3)]=deal(P3(1,3),P3(1,2));
    v1=P2-P1;
    v2=cross(P3-P1,v1);
    i=v1/norm(v1);
    j=v2/norm(v2);
    k=cross(i,j);
    PCS=[i(1),j(1),k(1);i(2),j(2),k(2);i(3),j(3),k(3)];
    SCS=PCS*RTM;
    eul=rotm2eul(SCS,'XYZ');
    roll(n)=eul(1);
    pitch(n)=eul(2);
    yaw(n)=eul(3);

end
%% grafiklerin elde edilmesi

plot(1:length(frames),radtodeg(roll));
title('Roll');
xlabel('Time (s)');
ylabel('Degree');
saveas(gcf,'roll.png');
plot(1:length(frames),radtodeg(pitch));
title('Pitch');
xlabel('Time (s)');
ylabel('Degree');
saveas(gcf,'pitch.png');
plot(1:length(frames),radtodeg(yaw));
title('Yaw');
xlabel('Time (s)');
ylabel('Degree');
saveas(gcf,'yaw.png');
```



Grafik 3 Rigidbody Roll açısı



Grafik 4 Rigidbody Pitch Açısı



Grafik 5 Rigidbody Yaw Açısı

2. Kısım 2

a) İlk olarak phoneIMU.mat dosyası matlab alanına yüklenerek a(accelerometer) verileri alınmıştır.

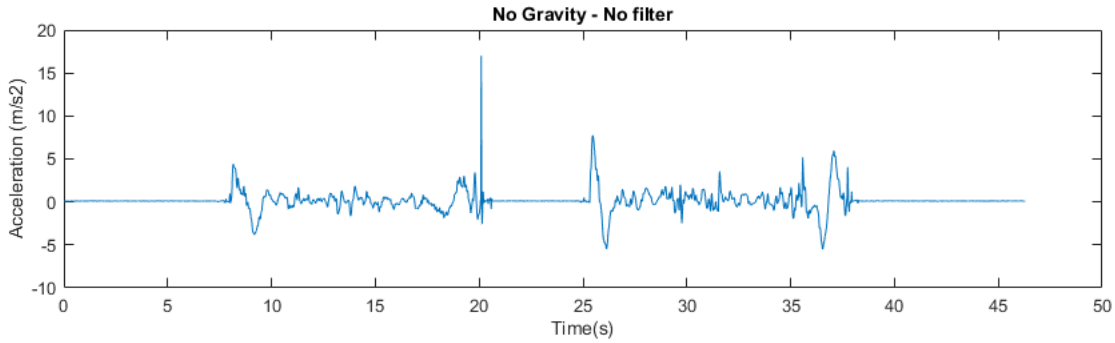
Bileşke ivme,

|a|=|a| = $\sqrt{{a_x}^2 + {a_y}^2 + {a_z}^2}$ formülü ile hesaplanmıştır.

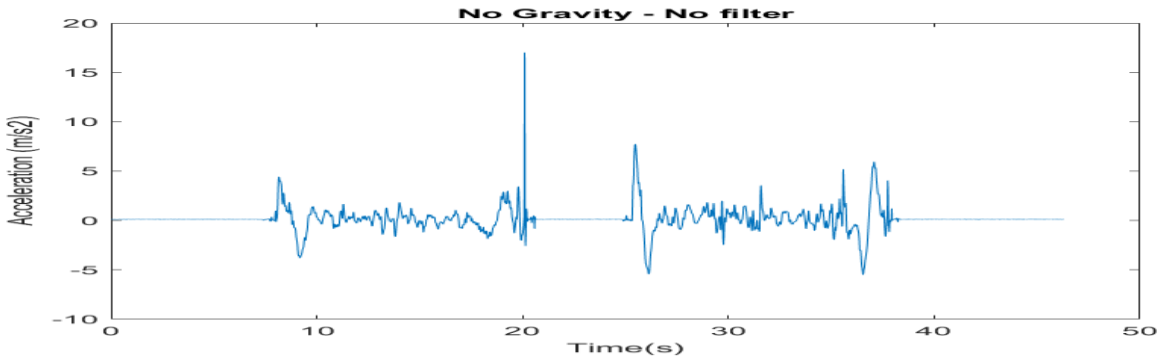
Yerçekimi dünya üzerinde her yerde aynı olmadığından Hacettepe Üniversitesi, Matematik Bölümünün enlem ve boylam değerleri yardımıyla(39.8695973, 32.7367129) yerçekimi https://www.sensorsone.com/local-gravity-calculator/ adresindeki hesaplama aracıyla 9.80148 m/s2 olarak bulunmuştur. Aşağıdaki kod yardımıyla bileşke ivme ve yerçekiminden arındırılmış bileşke ivme hesaplanmış ve grafikleri elde edilmiştir(Grafik 6, Grafik 7).

```
load('hafta13\phoneIMU.mat');
bileske_ivme=sqrt(a(:,1).^2+a(:,2).^2+a(:,3).^2);
%bileske ivme
plot(t_a,bileske_ivme);
title('Raw Magnitude - No filter');
xlabel('Time(s)');
ylabel('Acceleration (m/s2)');
saveas(gcf,'bileske_ivme.png');
g=9.80148;

a_no_g=bileske_ivme-g;

%bileske ivme(yercekimsiz)
plot(t_a,a_no_g);
title('No Gravity - No filter');
xlabel('Time(s)');
ylabel('Acceleration (m/s2)');
```

```
saveas(gcf,'bileske_ivme_yercekimsiz.png');
```



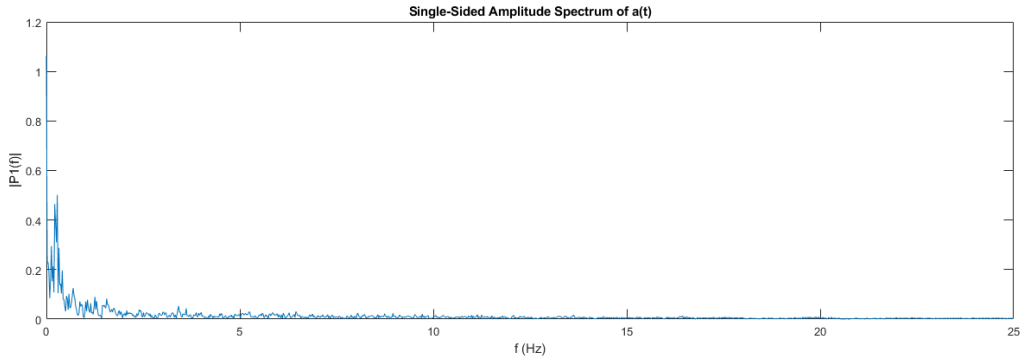
Grafik 6 Bileşke ivme



Grafik 7 Bileşke ivme(Yerçekimsiz)

İvmenin filtrelenmesi amacıyla öncelikle FFT(Fast Fourier transform) analizi yapılmıştır.

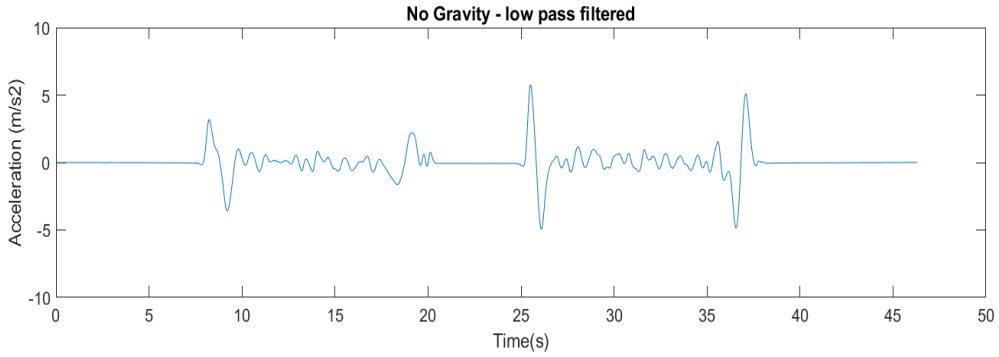
```
%% FFT incelemesi
Fs = 50;            % Sampling frequency
T = 1/Fs;             % Sampling period
L = length(t_a);             % Length of signal
t = t_a;        % Time vector
Y=fft(a);
P2 = abs(Y/L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = Fs*(0:(L/2))/L;
plot(f,P1)
title('Single-Sided Amplitude Spectrum of a(t)')
xlabel('f (Hz)')
ylabel('|P1(f)|')
```



Grafik 8 İvmenin FFT specturumu

Grafik 8'de görüldüğü üzere gürültüyü azaltmak için kullanılacak filtrede cut-off frekansı olarak 2-2.5 Hz arasında bir frekans seçilmesi uygun olacaktır. Low-pass filtreleme yaptıktan sonra çok küçük ivmeleri de ayılamak için cutoff frekansını 0.02 seçerek bir de high-pass filtreden geçirdim.

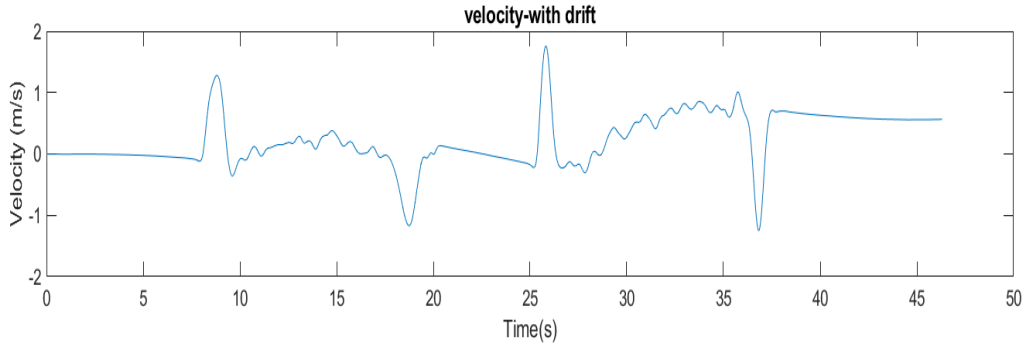
```
%% ivmenin filtrelenmesi

fc=2;
fs=1/0.02;
[c,d]=butter(2,fc/(fs/2),'low');

[e,f]=butter(1,0.02/(fs/2),'high');
filtrelenmis_ivme=filtfilt(c,d,a_no_g);
filtrelenmis_ivme=filtfilt(e,f,filtrelenmis_ivme);
%filtrelenmis yercekimsiz ivme
plot(t_a,filtrelenmis_ivme);
title('No Gravity - low pass filtered');
xlabel('Time(s)');
ylabel('Acceleration (m/s2)');
ylim([-10 10]);
saveas(gcf,'filtrelenmis_ivme.png');
```



Grafik 9 Filtrelenmis ivme(Yercekimsiz)

b) Filtrelenmiş ivmenin integralini alarak drift içeren hız grafiğini çizdirdiğim matlab kodu aşağıdaki gibidir.

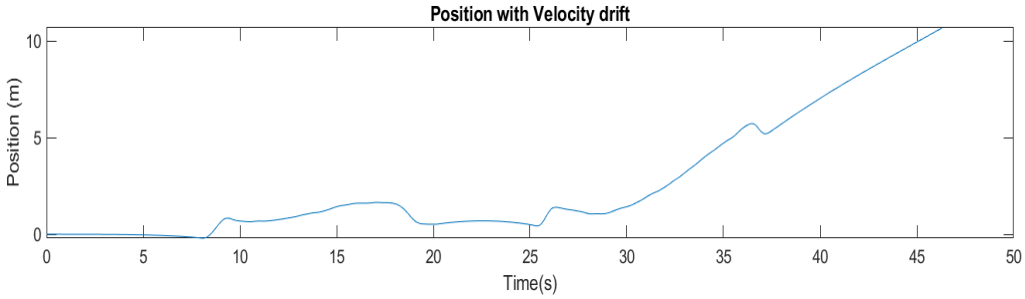
```
%% velocity-with drift
 v = cumtrapz(t_a,filtrelenmis_ivme);    %m/s
 plot(t_a,v);
 title('velocity-with drift');
 xlabel('Time(s)');
 ylabel('Velocity (m/s)');
 ylim([-2 2]);
 saveas(gcf,'velocity-with-drift.png');
```



Grafik 10 Velocity with drift

Elde ettiğim drift içeren hız değerlerinin integrali alınarak pozisyon değerleri hesaplanmıştır. İlgili matlab kodu ve pozisyon grafiği aşağıdaki gibidir.

```
%% position-with drift

  s=cumtrapz(t_a,v);
  plot(t_a,s);
  title('Position with Velocity drift');
 xlabel('Time(s)');
 ylabel('Position (m)');

 saveas(gcf,'position-with-drift.png');
```

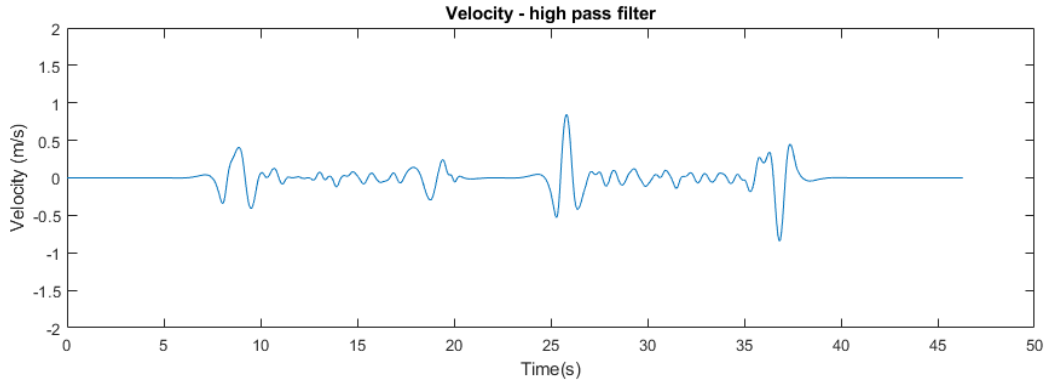


Grafik 11 Position with drift

c) Hızı low pass filtre kullanarak filtrelemek için gerekli matlab kodu ve sonucunda elde edilen grafik aşağıdaki gibidir.

```
%% filter velocity-with drift
fcv=0.5;
[e,f]=butter(2,fcv/(fs/2),'high');
v_filtered=filtfilt(e,f,v);

plot(t_a,v_filtered);
title('Velocity - high pass filter');
```

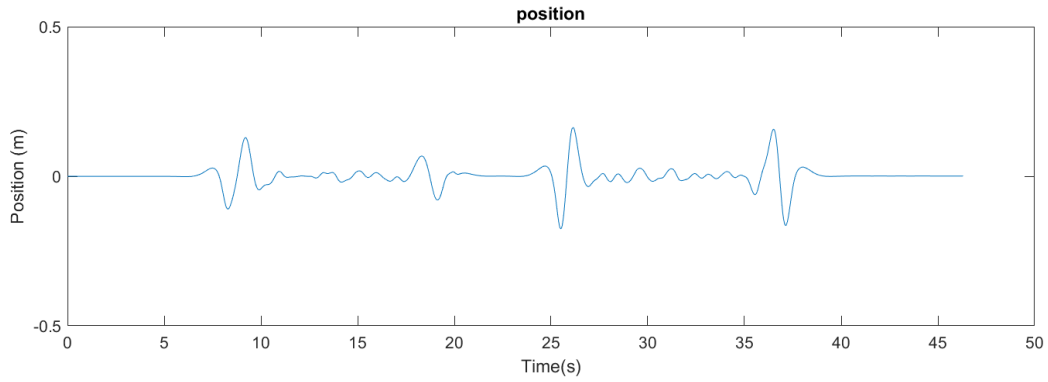
```
xlabel('Time(s)');
ylabel('Velocity (m/s)');
ylim([-2 2]);
saveas(gcf,'Velocity-filtered.png');
```



Grafik 12 Velocity high pass filtered

Filtrelenmiş hız verisinin integrali alınarak pozisyon değerleri elde edilmiştir. Bu işlem için gerekli matlab kodu ve elde edilen grafik aşağıdaki gibidir.

```
%% position no drift
s_filtered=cumtrapz(t_a,v_filtered);
plot(t_a,s_filtered);
title('position');
xlabel('Time(s)');
ylabel('Position (m)');
ylim([-0.5 0.5]);
saveas(gcf,'position.png');
```



Grafik 13 Position no-drift