KAYZER ROKET TAKIMI

TEKNOFEST-2022 ROKET YARIŞMASI

ÖTR AŞAMASI

UÇUŞ BENZETİMİ RAPORU

İçindekiler

Kinematik ve Dinamik Denklemler 3
Atmosfer Modeli 4
Motor Modeli 6
Aerodinamik Model 9
Benzetim Yapısı 12
Benzetimin Doğrulanması 20
Benzetim Sonuçları 22
REFERANSLAR 24

Kinematik ve Dinamik Denklemler

Roket uçuş benzetim yapılırken kinematik ve dinamik denklemler kullanılarak ivme, hız, konum ve uçuş yolu açısı bulunmuştur. Kullanılan formüller ve bu formüllerin neyi bulduğu aşağıda belirtilmiştir. Tüm parametreler nerdeyse birbiriyle bağlantılı olduğu için, denklemler tasarlanan Matlab koduna göre olacaktır.

Euler-Cromer yöntemi kullanılarak yükseklik, açı, hız, menzil değerleri bulunmuştur. Kullanılan formüller aşağıdadır.

$$\vec{F}_{net} = \vec{F}_{thrust} - \vec{F}_{drag} - \vec{F}_G$$

$$m\frac{d^2z}{dt^2} = F_T(t) - F_D(z, \dot{z}^2) - F_G(z, t)$$

$$\vec{F}_{net} = \vec{F}_{thrust} - \vec{F}_{drag} - \vec{F}_G$$

$$m\frac{d^2z}{dt^2} = F_T(t) - F_D(z, \dot{z}^2) - F_G(z, t)$$

$$g(z) = \frac{GM_E}{(z+R_E)^2}$$

Denklemdeki G, yerçekimsel sabittir ve 6.674e-11 olarak alınmıştır. M_E dünyanın kütlesidir ve 5.972e24 olarak alınmıştır. Z ise roketimizin yüksekliğini temsil etmektedir. R_E ise Dünya'nın yarıçapıdır ve 6,371,000 m olarak alınmıştır.

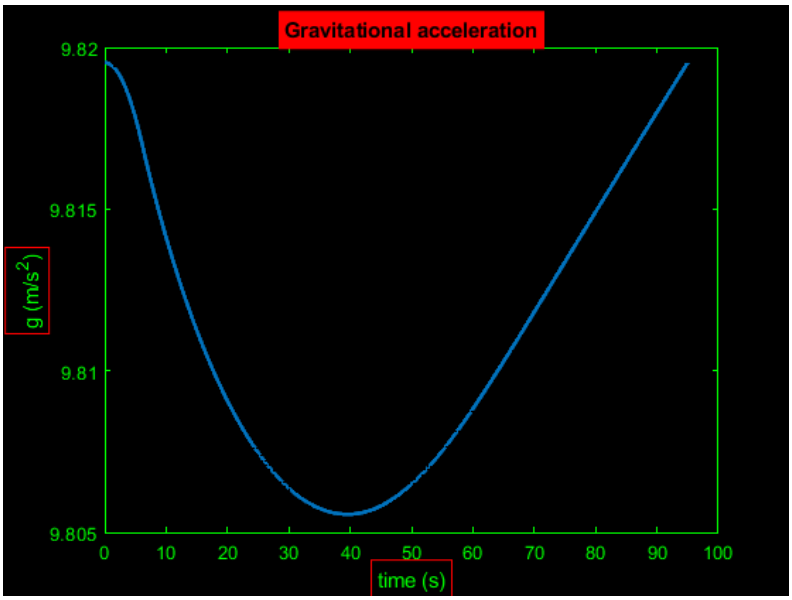
Zamana bağlı kütle ve yerçekimi hesaplanmıştır. Kütle zamanla azaldığından ve yerçekimi yükseklikle azaldığından, yerçekimi kuvveti zamana ve yüksekliğe bağlıdır.

Öncelikle yerçekimini bulmak için aşağıdaki formül kullanılmıştır.

$$g(z) = \frac{GM_E}{(z+R_E)^2}$$

Denklemdeki G, yerçekimsel sabittir ve 6.674e-11 olarak alınmıştır. M_E dünyanın kütlesidir ve 5.972e24 olarak alınmıştır. Z ise roketimizin yüksekliğini temsil etmektedir. R_E ise Dünya'nın yarıçapıdır ve 6,371,000 m olarak alınmıştır.

Zamana bağlı yerçekimi grafiği aşağıdadır.



Görüldüğü üzere yükseklik arttıkça yerçekimi ivmesi azalmıştır.

Atmosfer Modeli

Yerçekimi ivmesinin uçuş benzetim zamanına göre bulunmasından sonra, hava yoğunluğunun zamana göre değişimi hesaplanmıştır. Ek olarak zamana göre sıcaklık ve basınç değerleri hesaplanmıştır.

Öncelikle hava yoğunluğunun, sıcaklığın ve basıncın formülü aşağıda verilmiştir.

$$\rho(z) = \frac{M \cdot T(z)}{R \cdot P(z)}$$

$$T(z) = T_0 - Lz$$

$$P(z) = P_0 \left(\frac{T_0}{T_0 - Lz}\right)^{\frac{g_0 M}{RL}}$$

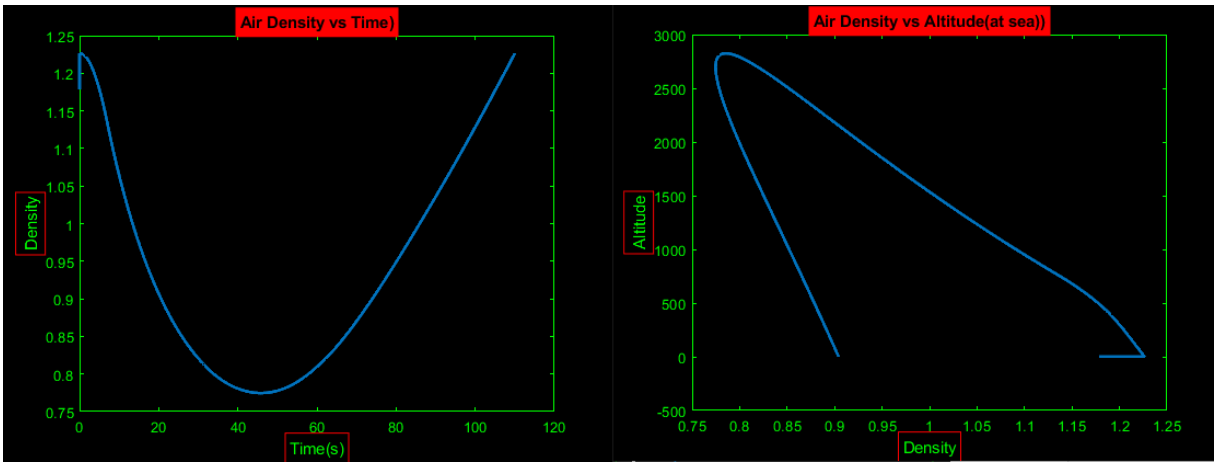
Ek olarak, roketimiz 86 km'den az irtifaya ulaştığı için hesaba dahil edilmeyen ama kodda kullandığımız, 86 km ve üzeri irtifa için kullanılan formül aşağıdadır.

$$\rho(z) = Az^4 + Bz^3 + Cz^2 + Dz + E$$

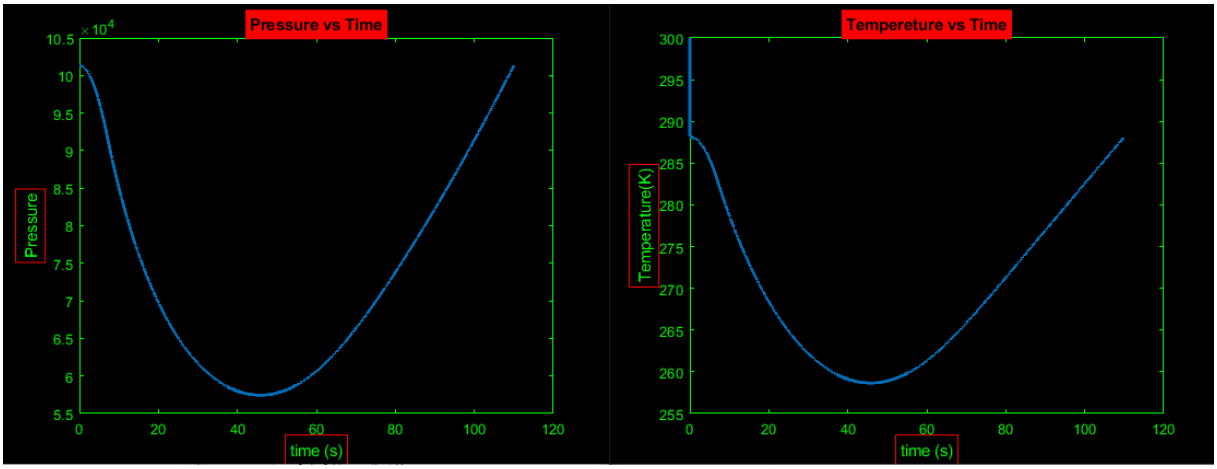
$$P(z) = Az^4 + Bz^3 + Cz^2 + Dz + E$$

Buradaki parametrelerin değerler 2. Kaynak referans alınarak kullanılmıştır.

Hava yoğunluğu zaman grafiği ve hava yoğunluğu deniz seviyesi yüksekliği aşağıda verilmiştir.



Ardından, uçuş benzetimde zamana göre basınç ve sıcaklık değişim grafiği aşağıdadır.



Ardından, ses hızı, mach sayısı ve drag katsayısı hesaplanmıştır. Bu parametreleri bulmak için The Prandtl-Glauert Rule denklemleri kullanılmıştır. Kullanılan formüller aşağıdadır.

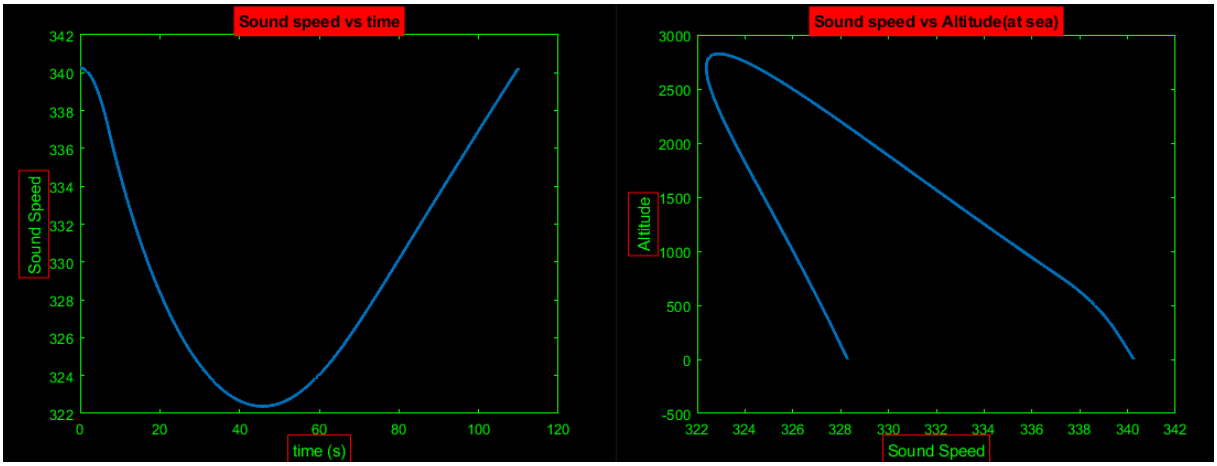
$$c_d = \frac{c_{do}}{\beta}$$

$$\beta = \sqrt{1 - M_\infty^2}$$

$$M_\infty(v, z) = \frac{v_\infty}{a_\infty}$$

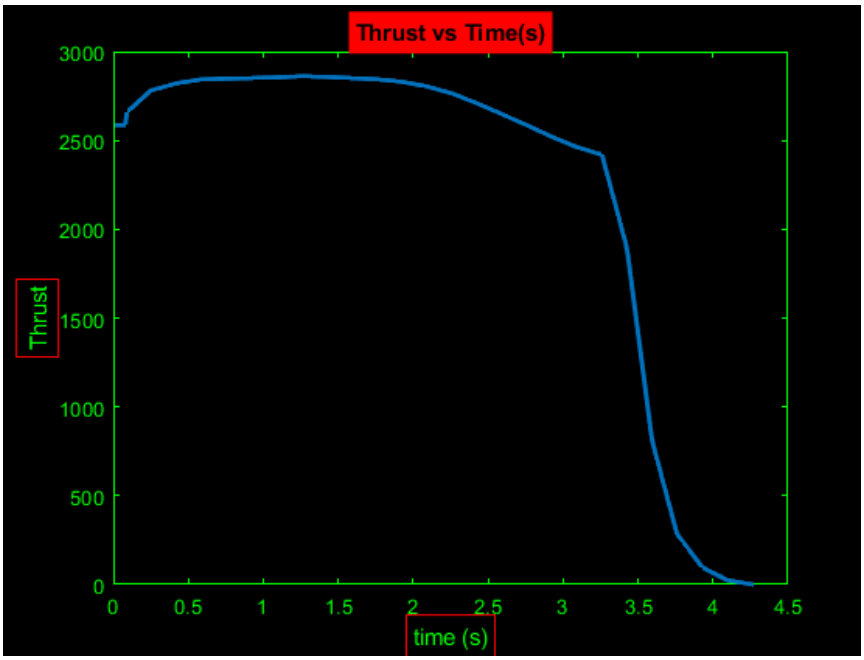
$$a_\infty(z) = \sqrt{\gamma R_{sp} T(z)}$$

İlk önce zamana ve deniz seviyesi yüksekliğine bağlı ses hızı değerleri bulunmuştur. Grafikler aşağıdadır.

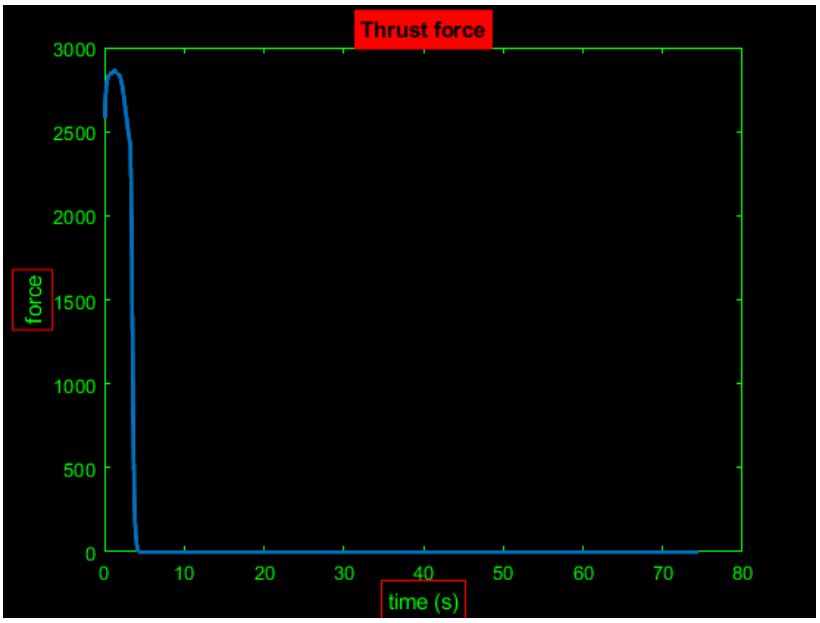


Motor Modeli

İtki kuvveti grafiği için gerekli veriler, paylaşılan itki zaman grafiğinden sağlanmıştır. M2020 motorunun zamana bağlı itki kuvveti grafiği aşağıdadır.



Uçuş benzetim zamanına bağlı interpolasyon kullanılarak elde edilen itki kuvveti grafiği aşağıdadır.

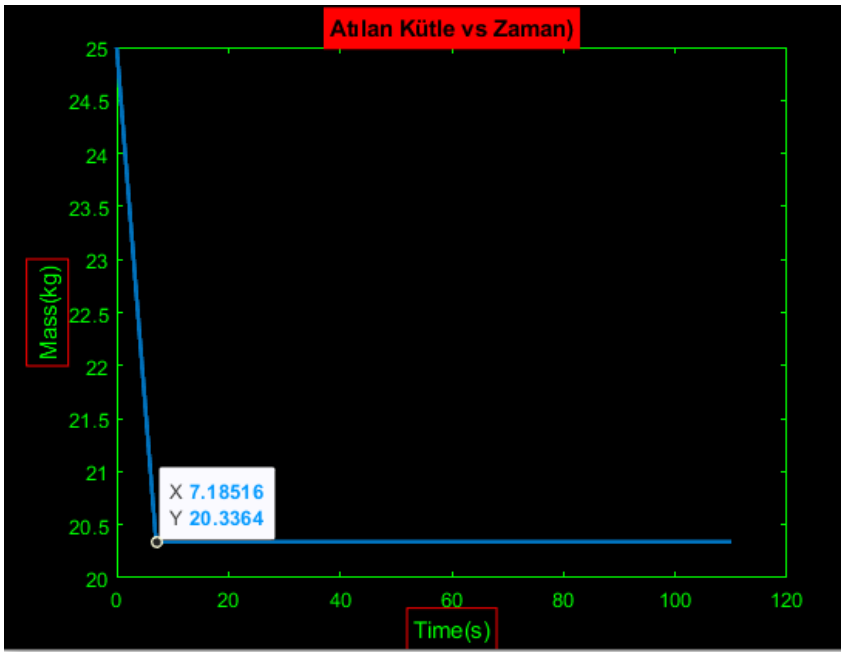


Veriler kullanılarak istenen parametrelere ulaşılmıştır. Kütle zamanla azalmaktadır. Bu yüzden zamana bağlı kütle hesaplandı. Kullanılan formül aşağıdadır. Bu formül Dm Tsiolkovsky Rocket denklemidir.

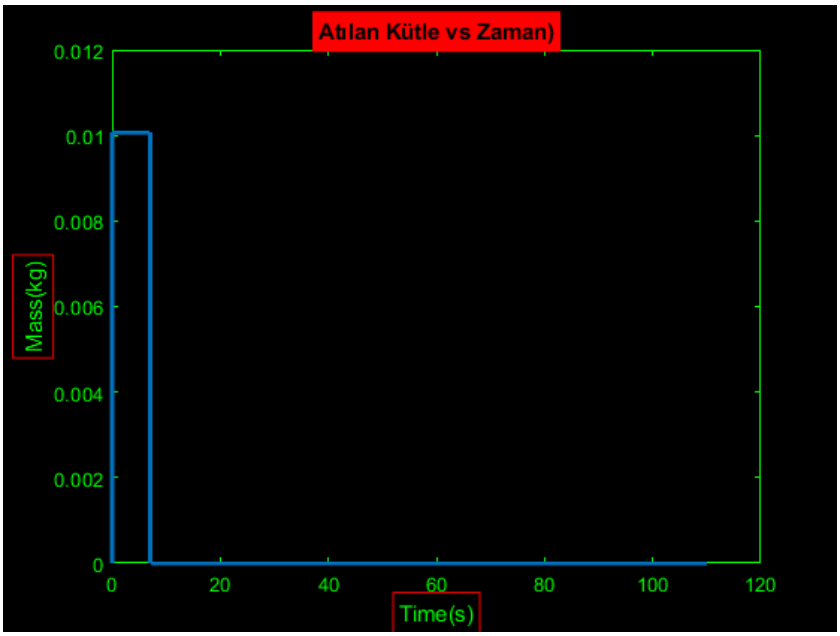
$$\dot{m} = \frac{F_{thrust}}{g_0 I_{sp}}$$

$$m(t) = m_0 - \dot{m}t$$

Buradaki F_{thrust} itki kuvvetimizdir. G0, yerçekimi ivmemizdir. I_{sp} ise özgül itkimizdir. Yani ilk formül dm dediğimiz zamana bağlı kütle azalımını vermektedir. İlk roket kütlemiz 25'tir. Başlangıç yakıt kütlemiz 4.659 kg'dır. Yani sonda kalan kütlemiz 20.341 kg olmalıdır. Yazdığımız kodun çıktısı olarak verdiği kütle zaman grafiği aşağıdadır.

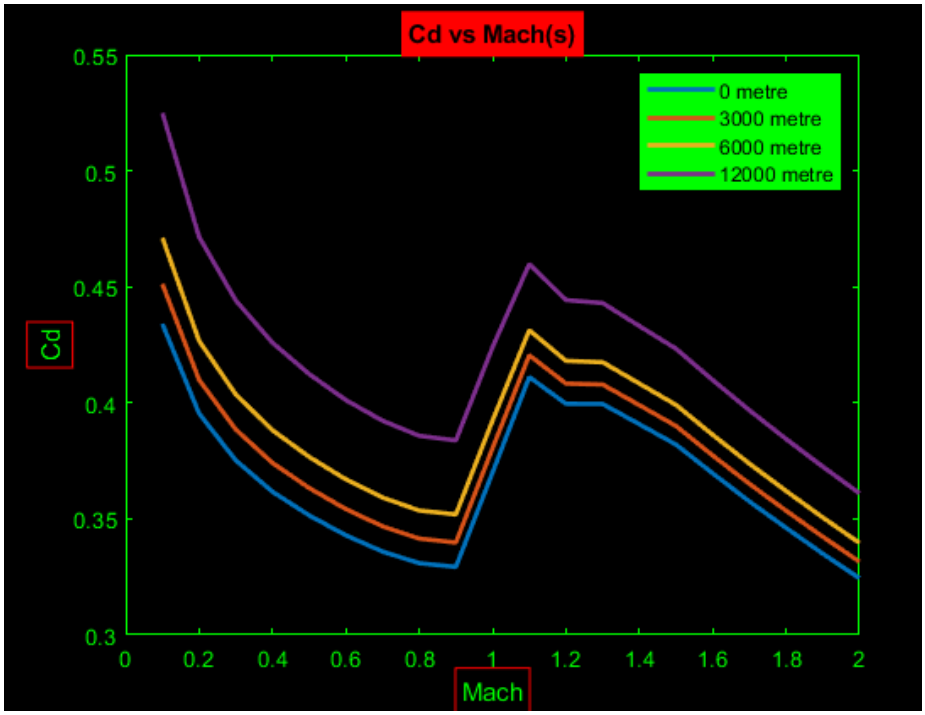


Görüldüğü üzere değerler eşleşmiştir. Ek olarak atılan kütle zaman grafiği aşağıdadır.

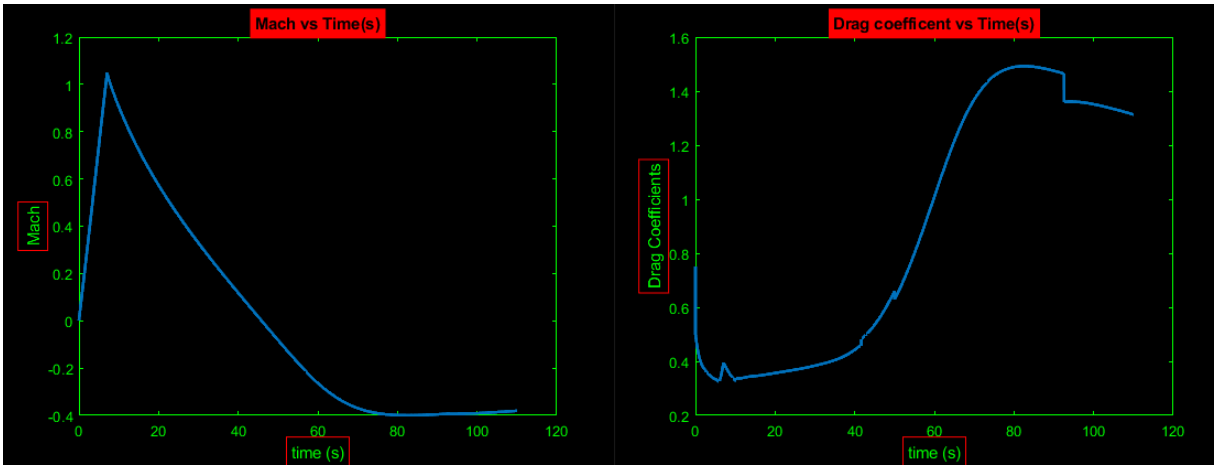


Aerodinamik Model

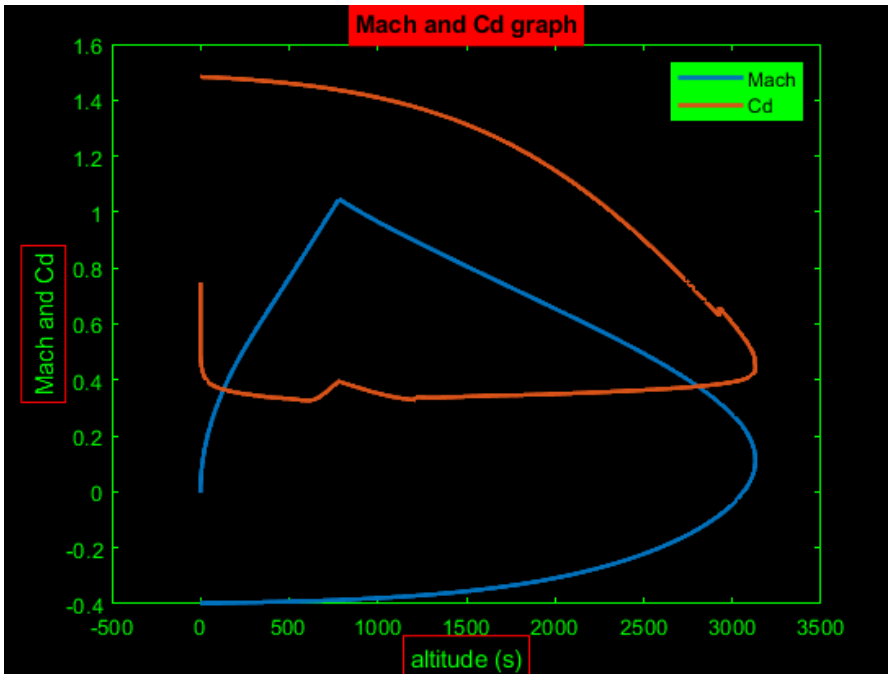
Öncelikle verile excell dosyasındaki Mach-Cd verilerini kullanarak yüksekliğe bağlı Cd/Mach grafiği bulunmuştur. Grafik aşağıdadır.



Ardından, Mach sayısının ve drag katsayısının zamana göre değişimi bulunmuştur. Bunun için, ek olarak bize verilen yüksekliğe bağlı cd Mach verileri kullanılarak tasarlanan interpolasyon fonksiyonu uygulanmıştır.

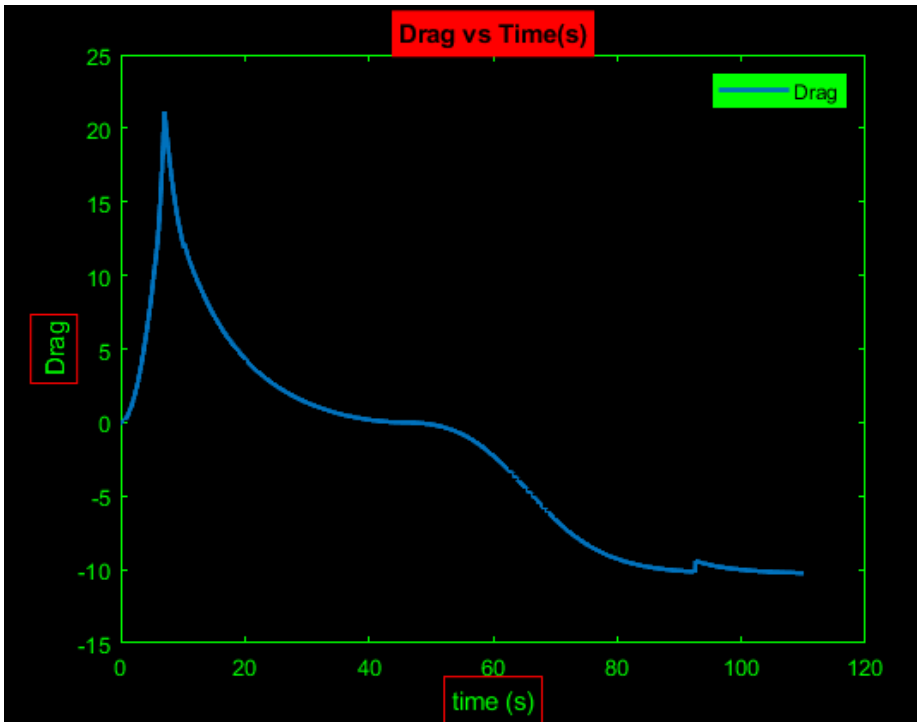


Ardından yüksekliğe bağlı Cd ve Mach verileri tek bir grafikte çizdirilmiştir.



Ardından, atmosferik sürüklenme bulunmuştur. Bu değeri bulmak için kullanılan formül aşağıdadır.

$$F_{drag} = \frac{1}{2}\rho(z)v^2 C_D A$$



Command Window

```
>>  fprintf('Maximum Maxh sayısı:  %d \n ',max(MACH));
Maximum Maxh sayısı:  1.046685e+00
```

Command Window

```
 >> fprintf('Tepe noktası z değeri: %d \n ',max(y))
fprintf('Tepe noktası x değeri:  %d \n ',tepe_x)
Tepe noktası z değeri: 3.129814e+03
 Tepe noktası x değeri:  1.587708e+03
```

Command Window

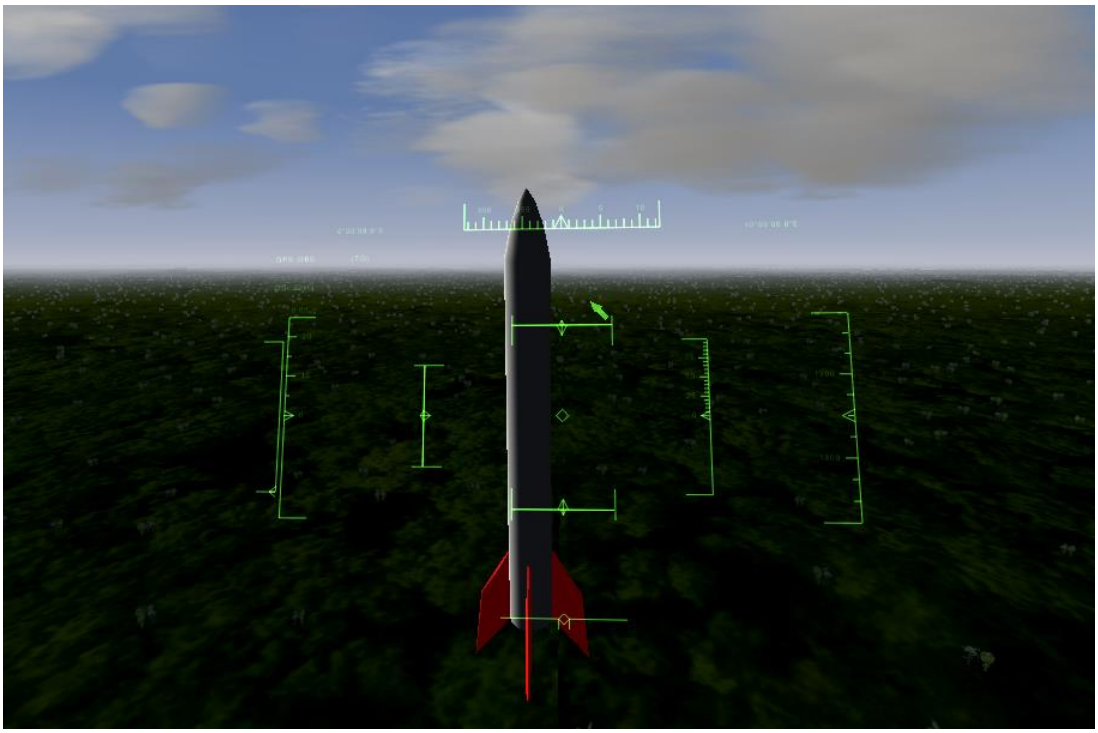
```
Tepe noktası Hız bileşke:  6.310712e+01
```

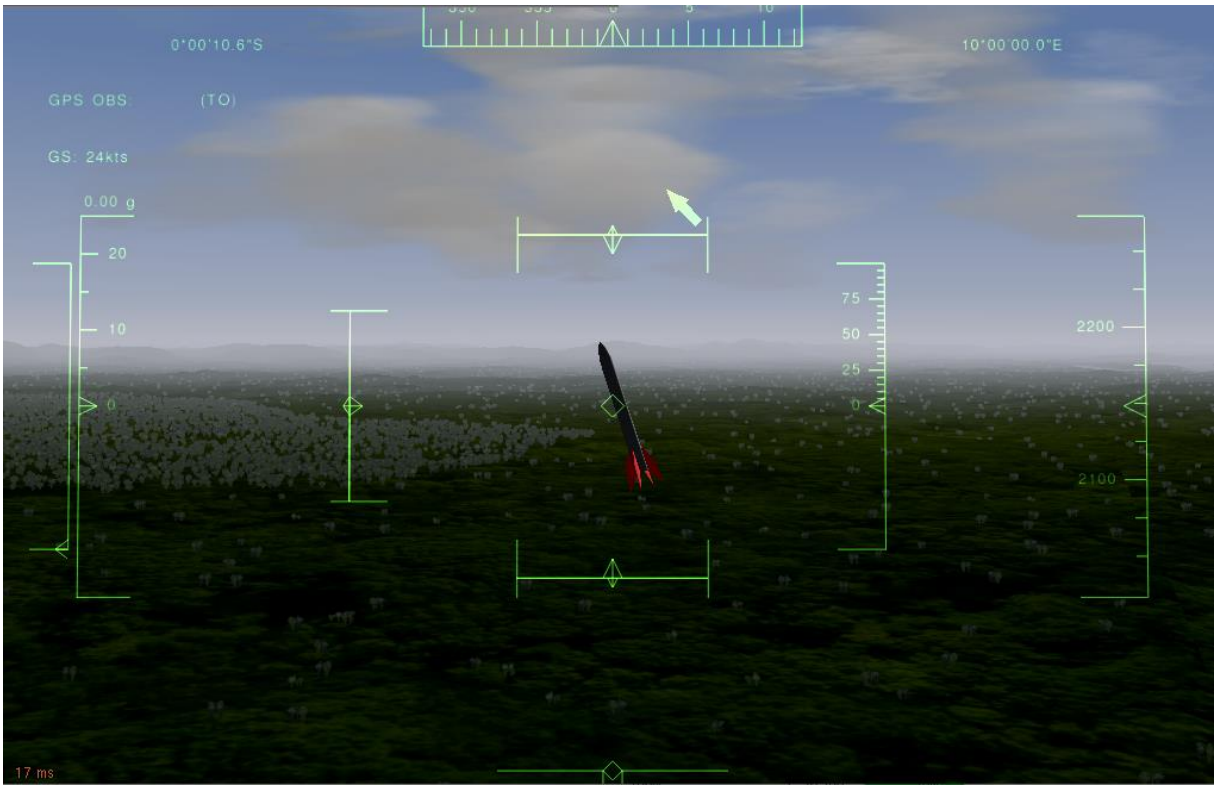
Command Window

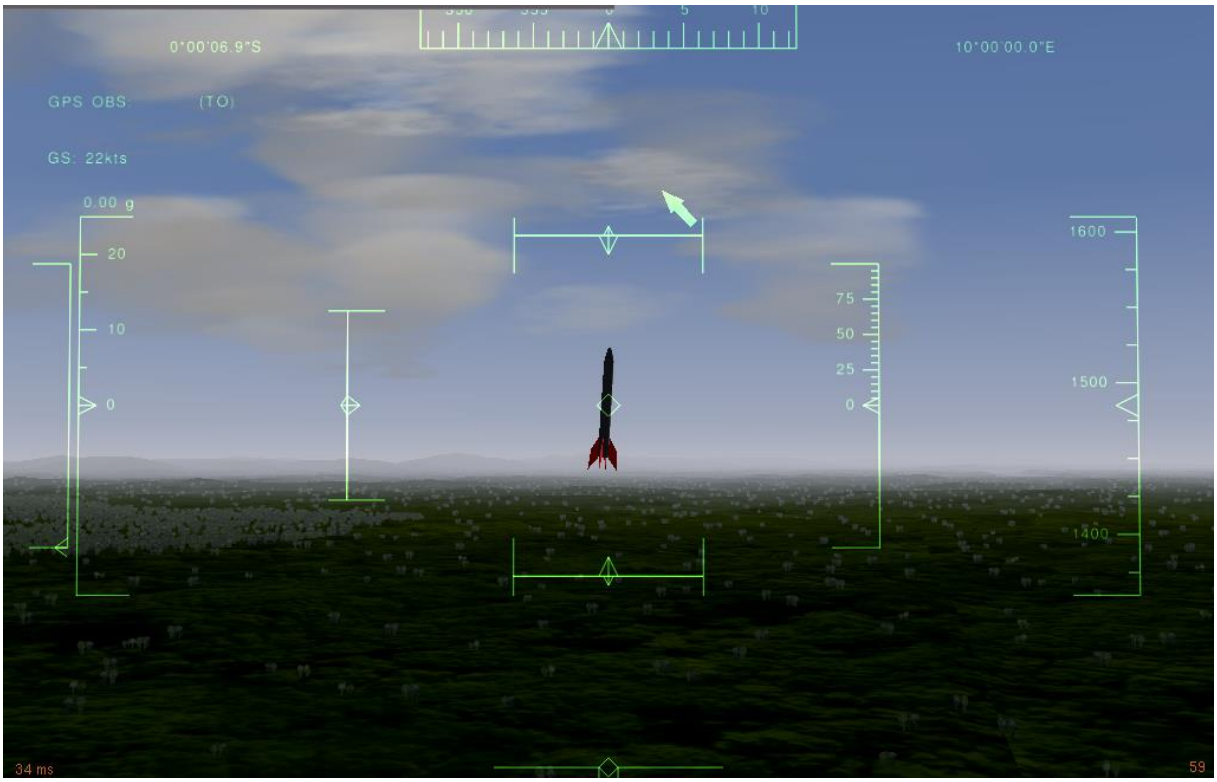
```
>>  fprintf('Tepe noktası zamanı(s):  %d \n ',t_2(ind));
Tepe noktası zamanı(s):  4.861364e+01
```

Benzetim Yapısı

İlk olarak Matlab Scripts de benzetim tamamlanmıştır.Ek olarak Flight Gear ve Matlab Simulink kullanarak elde ettiğim verilerle roketin gerçeğe yakın uçuş simülasyonu yaptık. Simülasyona ait fotoğraflar aşağıdadır.







Matlab Scripts kodları aşağıdadır.

```
clc
clear all
close all
cd_0=load('cd_0.txt');
cd_0=[cd_0 0.*cd_0(:,1)];
cd_3000=load('cd_3000.txt');
cd_3000=[cd_3000 3000.*ones(20,1)];
cd_6000=load('cd_6000.txt');
cd_6000=[cd_6000 6000.*ones(20,1)];
cd_12000=load('cd_12000.txt');
cd_12000=[cd_12000 12000.*ones(20,1)];
itki=load('itki.txt');
t1=0.01;
% index=1;

%% Universal constants
G = 6.674e-11; % gravitational constant
M_earth = 5.972e24; % mass of the Earth
mol = 0.029; % molar mass of air
R = 8.314; % gas constant
P0 = 101325; % standard pressure (Pa), sea level
T0 = 300; % Samsun temperature (K), sea level
rho0 = (mol*P0)/(R*T0); % air density, sea level
g0 = 9.81; % gravitational acceleration, sea level
MACH=0;
%% Local constants
% specifications for Falcon 1e
FT = 2070.11; % rocket thrust, in Newtons
FX = 2587.12;
FY = 2587.12;
C0 = 0.75; % drag coefficient, see notes
 CD=C0;
d = .134; % rocket diameter, in meters
A = pi*(d/2)^2; % rocket cross-sectional area
L = 2.748; % rocket length, in meters
m0 = 27.511; % initial mass, in kg ("wet mass")
empty = 25-4.349; % mass when fuel is expended, in kg ("dry mass")
Isp = 209.5; % specific impulse, in seconds
dm = FT/(g0*Isp); % mass flow rate, dm/dt
```

```
theta=85;
vx=2*cosd(theta);
vy=2*sind(theta);
%% Euler-Cromer Method
dt = 0.01; % time step
z0 =0; % intial altitude
v0 = 2; % initial velocity
v = v0;
z = z0;
V = v;
Z = z;
m = m0;
Rho=rho0;
T=T0;
P=P0;
TT=T0;
PP=P0;
M = m;
Thrust = FT/m;
Drag = 0;
g=9.8195;
grav=g;
nextstage=0;
tmax =110 ;
CS=340.26;
for t=0.01:dt:tmax

    t_deger=t;
 g = (G*M_earth)/((z+6371000)^2); % gravitational acceleration, g(z)
 m = m - (dm*dt); % changing mass, m(t)
 %% density finding
 mol = 0.029;
 R = 8.314;
 h = z/1000; % h, altitude in km

 if h <= 11 % pressure and temperature values by altitude
 T = 288.15 - 6.5*h; %6.5 is temperature lapse rate
 P = 101325*((288.15/(288.15-6.5*h))^(34.1632/-6.5));
 elseif 11 < h && h <= 20
 T = 216.65;
 P = 22632.06*exp(-34.1632*(h-11)/216.65);
 elseif 20 < h && h <= 32
 T = 196.65 + 0.001*z;
 P = 5474.889 * ((216.65/(216.65+(h-20)))^(34.1632));
 elseif 32 < h && h <= 47
 T = 139.05 + 2.8*h;
 P = 868.0187 * ((228.65/(228.65+2.8*(h-32)))^(34.1632/2.8));
 elseif 47 < h && h <= 51
 T = 270.65;
 P = 110.9063 * exp(-34.1632*(h-47)/270.65);
 elseif 51 < h && h <= 71
 T = 413.45 - 2.8*h;
 P = 66.93887*((270.65/(270.65-2.8*(h-51)))^(34.1632/-2.8));
 else %71 < h && h <= 86
 T = 356.65 - 2.0*h;
 P = 3.956420*((214.65/(214.65-2*(h-71)))^(34.1632/-2));
 end
 rho = (mol*P)/(R*T);

 if 86 < h && h <= 91
 P = exp(-4.22012E-08*h^5 + 2.13489E-05*h^4 - 4.26388E-03*h^3 + 0.421404*h^2 - 20.8270*h + 416.225);
 rho = exp(7.5691E-08*h^5 - 3.76113E-05*h^4 + 0.0074765*h^3 - 0.743012*h^2 + 36.7280*h - 729.346 );
 T = 186.8673;
 elseif 91 < h && h <= 100
 P = exp(-4.22012E-08*h^5 + 2.13489E-05*h^4 - 4.26388E-03*h^3 + 0.421404*h^2 - 20.8270*h + 416.225);
 rho = exp(7.5691E-08*h^5 - 3.76113E-05*h^4 + 0.0074765*h^3 - 0.743012*h^2 + 36.7280*h - 729.346 );
 T = 263.1905-76.3232*sqrt(1 - ((h-91)/-19.9429)^2);
 elseif 100 < h && h <= 110
 P = exp(-4.22012E-08*h^5 + 2.13489E-05*h^4 - 4.26388E-03*h^3 - 0.421404*h^2 - 20.8270*h + 416.225);
 rho = exp(7.5691E-08*h^5 - 3.76113E-05*h^4 + 0.0074765*h^3 - 0.743012*h^2 + 36.7280*h - 729.346 );
 T = 263.1905-76.3232*sqrt(1 - ((h-91)/-19.9429)^2);
 elseif 110 < h && h <= 120
 rho = exp(-8.854164E-05*h^3 + 0.03373254*h^2 - 4.390837*h + 176.5294);
 P = 0;
```

```
 T = 240 + 12*(h-110);
 elseif 120 < h && h <= 150
 P = 0;
 rho = exp(3.661771E-07*h^4 - 2.154344E-04*h^3 + 0.04809214*h^2 - 4.884744*h + 172.3597);
 T = 1000 - 640*exp(-0.01875*(h-120)*(6356.766 + 120)/(6356.766+h));
 elseif 150 < h %&& h <= 200
 P = 0;
 rho = 02.0763e-09;
 T = 1000 - 640*exp(-0.01875*(h-120)*(6356.766 + 120)/(6356.766+h));
 end

%%  finding cd and drag
%  Cd = CD(v,temp,C0);
temp=T;
press=P;
 cs = sqrt(1.4*287*T); % sound speed as function of temperature
 Mach = v/cs; % Mach number


%  if Mach < 1
%  Cd = C0/sqrt(1-Mach^2); % Prandtl-Glauert Rule

difference_0=abs(z-0);
difference_3000=abs(z-3000);
difference_6000=abs(z-6000);
difference_12000=abs(z-12000);
difference_all=[difference_0 difference_3000 difference_6000 difference_12000];
siralama=sort(difference_all);
if siralama(1)==difference_0
 new_cd=spline(cd_0(:,2),cd_0(:,1),Mach);
elseif siralama(1)==difference_3000
 new_cd=spline(cd_3000(:,2),cd_3000(:,1),Mach);
elseif siralama(1)==difference_6000
 new_cd=spline(cd_6000(:,2),cd_6000(:,1),Mach);
elseif siralama(1)==difference_12000
 new_cd=spline(cd_12000(:,2),cd_12000(:,1),Mach);
end
Cd=new_cd;


%%
        thrust = FT/m;
% index=index+1
%    if t1*100>423d
%       thrust=0;
%  else
%       ind=find(t1==itki(:,1));
%         newww=itki(index,2);
%   end
%     thrust=newww;
 drag = 0.5*rho*(v^2)*Cd*A/m;
 if v < 0 % flip drag force vector if rocket falls
 drag = drag*-1;
 end
 Launch_Rod_Length = 1;                  % Length of launch rod (m)

    if z <= Launch_Rod_Length       % Launch rod normal force
        Fn = m*g*cosd(theta);
    else
        Fn = 0;                             % No longer on launch rod
    end


 v = v + (thrust - drag - g)*dt; % new velocity
 z = z + v*dt; % new altitude
 V = [V,v];
 Z = [Z,z];
 M = [M,m];
 CS= [CS,cs];
%  FX=[FX,fx];
%  FX=[FY,fy];
%  fx_arary(t1*100)=fx;
 grav = [grav,g];
 Thrust = [Thrust,thrust];
```

```
 Drag = [Drag,drag];
 Rho=[Rho,rho];
 T=[T,temp];
 P=[P,press];
 MACH=[MACH,Mach];
 CD=[CD,Cd];
TT=[TT,T];
PP=[PP,P];
 t1=t;

 if z < 0 % rocket crashes or fails to launch
 break
 elseif m < empty % rocket runs out of fuel, mass becomes stable
 FT = 0;
 dm = 0;
 end

end


t = linspace(0,tmax,length(Z));
%%
Theta(1)=85;
Vx(1)=2*cosd(Theta(1));
Vy(1)=2*sind(Theta(1));
Vx_deneme(1)=2*cosd(Theta(1));
 x(1)=0;
  y(1)=0;
  Distance_x(1)=0;
  Distance_y(1)=0;
  new_thrust(1)=itki(1,2);
  mass(1)=25;
  dmm(1)=dm;
 for i=2:length(V)
     if t(i)<4.301
%      new_thrust(i)=itki(i,2);
   new_thrust(i)=spline(itki(:,1),itki(:,2),t(i));
     else
     new_thrust(i)=0;
     end

     dmm(i) = new_thrust(i)/(grav(i)*Isp); % mass flow rate, dm/dt
      mass(i) = mass(i-1)-(dmm(i)*dt); % changing mass, m(t)

    if Z(i) <= Launch_Rod_Length       % Launch rod normal force
        Fn(i) = M(i)*grav(i)*cosd(Theta(1));
    else
        Fn(i) = 0;                                % No longer on launch rod
    end
 Fx(i)= new_thrust(i)*cosd(Theta(i-1))-Drag(i)*cosd(Theta(i-1))-Fn(i)*sind(Theta(i-1)); % Sum x forces
    Fy(i)= new_thrust(i)*sind(Theta(i-1))-(M(i)*grav(i))-...
        Drag(i)*sind(Theta(i-1))+Fn(i)*cosd(Theta(i-1));    % Sum y forces

    % Acceleration calculations
    Ax(i)= Fx(i)/M(i);                      % Net accel in x direction
    Ay(i)= Fy(i)/M(i);                      % Net accel in y direction

% %      v = v + (thrust - drag - g)*dt; % new velocity
% %  z = z + v*dt; % new altitude
    Vx_deneme(i)=Vx(i-1)+Fx(i)*dt;
    % Velocity calculations
    Vx(i)= Vx(i-1)+Ax(i)*dt;                % Velocity in x direction
%     v = Vy(n) + (Thrust(n) - Drag(n) - g(n))*Delta; % new velocity
    Vy(i)= Vy(i-1)+Ay(i)*dt;                % Velocity in y direction
%       Vy(i)=Vy(i-1) + (new_thrust(i) - Drag(i) - grav(i))*dt; % new velocity
    % Position calculations
    x(i)= x(i-1)+Vx(i)*dt;                  % Position in x direction
    y(i)= y(i-1)+Vy(i)*dt;                  % Position in y direction

    % Distance calculations
    Distance_x(i) = Distance_x(i-1)+abs(Vx(i)*dt);      % Distance in x
    Distance_y(i) = Distance_y(i-1)+abs(Vy(i)*dt);      % Distance in y
    Distance(i) = (Distance_x(i)^2+Distance_y(i)^2)^(1/2); % Total distance
```

```
    % Rocket angle calculation
    Theta(i)= atand(V(i)/Vx(i));      % Angle defined by velocity vector

     if y(i) < 0 % rocket crashes or fails to launch
 break
 end
 end


 %% Plot the trajectory

line = zeros(1,size(t,2));

plot(t(1:length(y)),y/1000)
title('Rocket altitude')
ylim([0,1.5*max(Z)/1000])
xlabel('time (s)')
ylabel('altitude (km)')

%% x
t_2=linspace(0,tmax,length(x));
createfigure1(t_2,x)
title('Rocket horizontal position')
% ylim([1.5*min(Vy),1.5*max(Vy)])
xlabel('time (s)')
ylabel('position (m)')
%% y
t_2=linspace(0,tmax,length(y));
createfigure1(t_2,y)
title('Rocket vertical position')
% ylim([1.5*min(Vy),1.5*max(Vy)])
xlabel('time (s)')
ylabel('position (m)')

%% VY
t_2=linspace(0,tmax,length(Vy));
createfigure1(t_2,Vy)
title('Rocket vertical velocity')
% ylim([1.5*min(Vy),1.5*max(Vy)])
xlabel('time (s)')
ylabel('velocity (m/s)')

%% Vx
t_2=linspace(0,tmax,length(Vx));
createfigure1(t_2,Vx)
title('Rocket horizontal velocity')
% ylim([1.5*min(Vy),1.5*max(Vy)])
xlabel('time (s)')
ylabel('velocity (m/s)')

%% AX
t_2=linspace(0,tmax,length(Ax));
createfigure1(t_2,Ax)
hold on;
plot(t_2,0*Ay,'--w')
title('Rocket horizontal acceleration')
% ylim([1.5*min(Vy),1.5*max(Vy)])
xlabel('time (s)')
ylabel('acceleration (m/s^2)')
%% Ay
t_2=linspace(0,tmax,length(Ay));
createfigure1(t_2,Ay)
title('Rocket vertical acceleration')
% ylim([1.5*min(Vy),1.5*max(Vy)])
hold on;
plot(t_2,0*Ay,'--w')
xlabel('time (s)')
ylabel('acceleration (m/s^2)')

%% Plot the forces
createfigure1(t(1:length(new_thrust)),new_thrust);
title('Thrust force')
xlabel('time (s)')
```

```
ylabel('force')
% ylim([-
0.5*max(new_thrust.*M(1:length(new_thrust))/1000),1.5*max(new_thrust.*M(1:length(new_thrust))/1000)])
% figure
% plot(t,Drag.*M,t,line,'--k')
% title('Drag force')
% xlabel('time (s)')
% ylabel('force (N)')

%% itki

createfigure1(itki(:,1),itki(:,2));
title('Thrust vs Time(s)')
xlabel('time (s)')
ylabel('Thrust')


%% drag

createfigure1(t, Drag)
title('Drag vs Time(s)')
xlabel('time (s)')
ylabel('Drag')
legend('Drag');


%% gravity
createfigure1(t, grav)
title('Gravitational acceleration')
xlabel('time (s)')
ylabel('g (m/s^2)')

%%  cd ve mach grafiği
createfigure1(y,MACH(1:length(y)));
hold on;
plot(y,CD(1:length(y)),'LineWidth',2);
title('Mach and Cd graph')
xlabel('altitude (s)')
ylabel('Mach and Cd');
legend('Mach','Cd');

%%   mach zaman grafiği

createfigure1(t,MACH);
title('Mach vs Time(s)')
xlabel('time (s)')
ylabel('Mach');

%%   CD zaman grafiği

createfigure1(t,CD);
title('Drag coefficent vs Time(s)')
xlabel('time (s)')
ylabel('Drag Coefficients');
% %%   CD mach zaman grafiği
%
% createfigure1(MACH);
% title('Cd vs Mach(s)')
% xlabel('Cd')
% ylabel('Mach');


%% farklı yükseklik mach cd grafiği
createfigure1(cd_0(:,2),cd_0(:,1));
title('Cd vs Mach(s)')
ylabel('Cd')
xlabel('Mach');
hold on
plot(cd_3000(:,2),cd_3000(:,1),'LineWidth',2)
hold on
plot(cd_6000(:,2),cd_6000(:,1),'LineWidth',2);
hold on
plot(cd_12000(:,2),cd_12000(:,1),'LineWidth',2);
legend('0 metre','3000 metre','6000 metre','12000 metre');
```

```
%%  cs yükseklik ve zaman grafiği

createfigure1(CS(1:length(y)),y)
title('Sound speed vs Altitude(at sea)')
ylabel('Altitude')
xlabel('Sound Speed');


createfigure1(t,CS)
title('Sound speed vs time')
xlabel('time (s)')
ylabel('Sound Speed');

%% sıcaklık ve basınç grafiği

t_2=linspace(0,tmax,length(TT));
%sıcaklık
createfigure1(t_2,TT)
title('Tempereture vs Time')
xlabel('time (s)')
ylabel('Temperature(K)');

%BASINÇ
t_3=linspace(0,tmax,length(PP));
createfigure1(t_3, PP)
title('Pressure vs Time')
xlabel('time (s)')
ylabel('Pressure');


%%
% KÜTLE GRAFİĞİ
atilan_kutle(1)=0;
for j=2:length(M);
atilan_kutle(j)=M(j-1)-M(j);
end
createfigure1(t, atilan_kutle)
title('Atılan Kütle vs Zaman)')
xlabel('Time(s)')
ylabel('Mass(kg)');

createfigure1(t, M)
title('Atılan Kütle vs Zaman)')
xlabel('Time(s)')
ylabel('Mass(kg)');
%% AIR DENSİTY

createfigure1(t, Rho)
title('Air Density vs Time)')
xlabel('Time(s)')
ylabel('Density');

createfigure1(Rho(1:length(y)),y)
title('Air Density vs Altitude(at sea))')
xlabel('Density')
ylabel('Altitude');

ylabel('Mass(kg)');
%% Yükseklik-menzil grafiği

createfigure1(x, y)
title('Yükseklik vs Menzil')
xlabel('Menzil')
ylabel('Yükseklik');

%% Uçuş yolu açısı

createfigure1(t_2,Theta)
title('Uçuş yolu Açısı vs Zaman')
ylabel('Açı')
```

```
xlabel('Zaman');

%% yazdırma
% kütle 27511 g
%rakım 970 olması lazım
%yakıt kütlesi 4349

ind=find(y==max(y));
tepe_x=x(ind);

fprintf('Tepe noktası z değeri: %d \n ',max(y))
fprintf('Tepe noktası x değeri:  %d \n ',tepe_x)
minumum=min(abs(t_2(ind)-t))
index_2=find(t==t_2(ind)+minumum)
tepe_mach=MACH(ind);
fprintf('Tepe noktası Mach sayısı:  %d \n ',tepe_mach);
fprintf('Tepe noktası Hız bileşke:  %d \n ',Vx(ind));

 fprintf('Maximum Maxh sayısı:  %d \n ',max(MACH));
 fprintf('Tepe noktası zamanı(s):  %d \n ',t(ind));
  fprintf('Maximum dinamik basınç(s):  %d \n ',max(PP));
```

Benzetimin Doğrulanması

Bize verilen girdi değerlerimizi kullandık. Bize verilen girdi değerleri aşağıdadır.

	Değer
Pozisyon [m]	[0, 0, 0]
Hız (bileşke) [m/s]	2
Uçuş Yolu Açısı [derece]	85

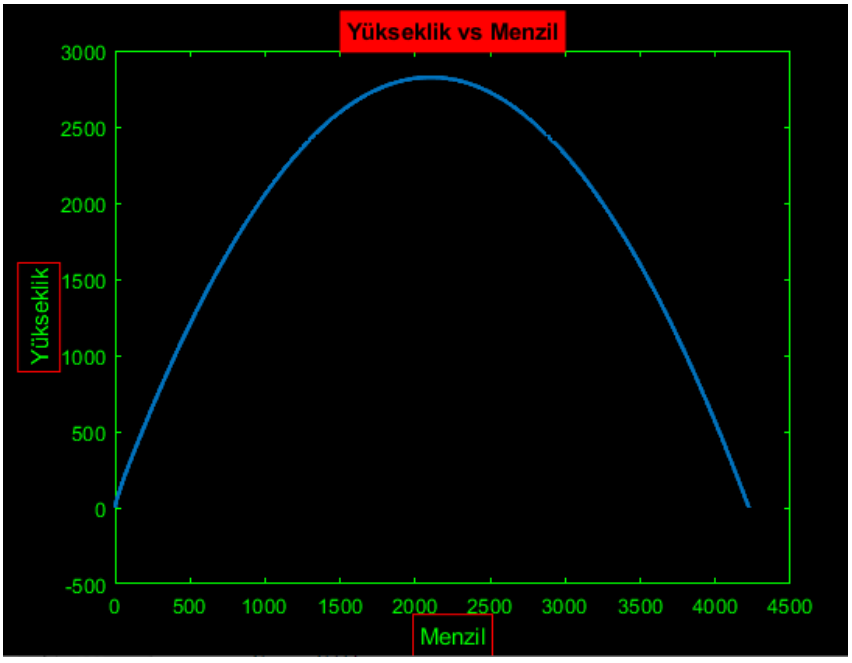
Tablo 2. Doğrulama Çalışması Diğer Verileri **Değer**

Başlangıç Kütlesi [kg]	25
Atış Noktası Rakımı [m]	980
Başlangıç Yakıt Kütlesi [kg]	4.659
Özgül İtki (Isp) [s]	209.5
İtki Profili Dosyası	“veri_itki_F_2022.xlsx”
Aerodinamik Veri Seti Dosyası	“veri_aero_Cd_2022.xlsx”
Roket Çapı [m]	0.14

	Değer
Pozisyon [m]	[0, 0, 0]
Hız (bileşke) [m/s]	2
Uçuş Yolu Açısı [derece]	85
Başlangıç Kütlesi [kg]	Roket kütlesi
Atış Noktası Rakımı [m]	980

Benzetim çalışması sonucu bulduğumuz çıktılar aşağıdaki gibidir.

	Değer
Maksimum Mach Sayısı [-]	1.046685
Tepe Noktası Pozisyonu [m]	[2102.408, 0 , 2825.2308]
Tepe Noktası Hızı (bileşke) [m/s]	88.16135
Tepe Noktası Mach Sayısı [-]	0.15779
Tepe Noktası Zamanı [s]	19.00567



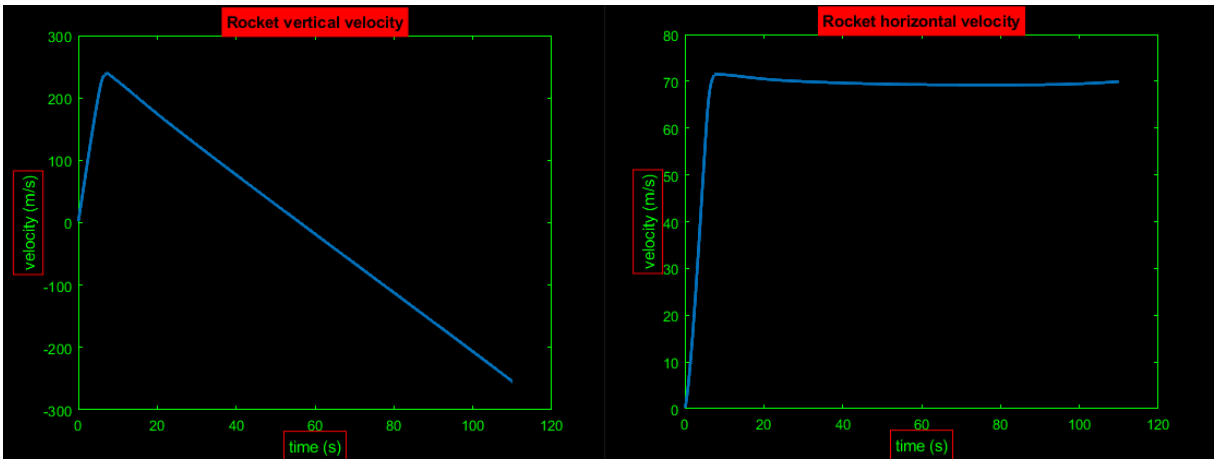
Benzetim Sonuçları

Euler-Cromer yöntemi kullanılarak yükseklik, açı, hız, menzil değerleri bulunmuştur. Kullanılan formüller aşağıdadır.

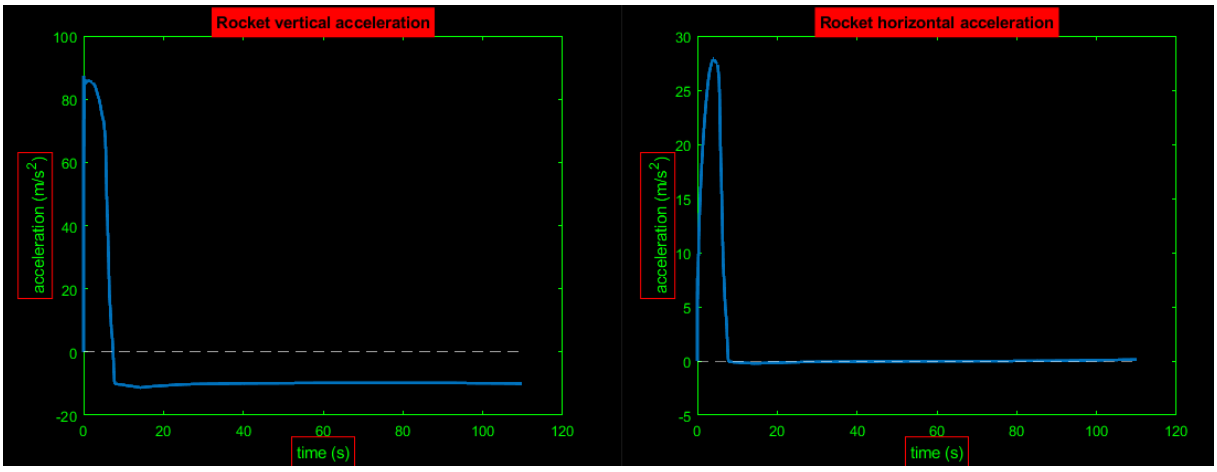
$$\vec{F}_{net} = \vec{F}_{thrust} - \vec{F}_{drag} - \vec{F}_G$$

$$m\frac{d^2z}{dt^2} = F_T(t) - F_D(z, \dot{z}^2) - F_G(z, t)$$

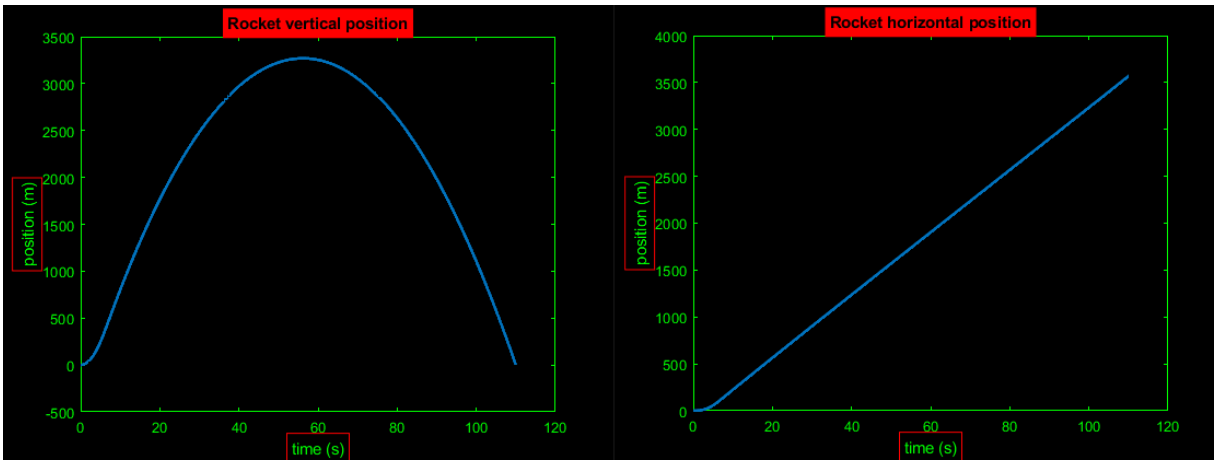
Öncelikle roketin dikey ve yatay hızları bulunmuştur. Dikey ve yatay hızın zamana göre grafiği aşağıdadır.



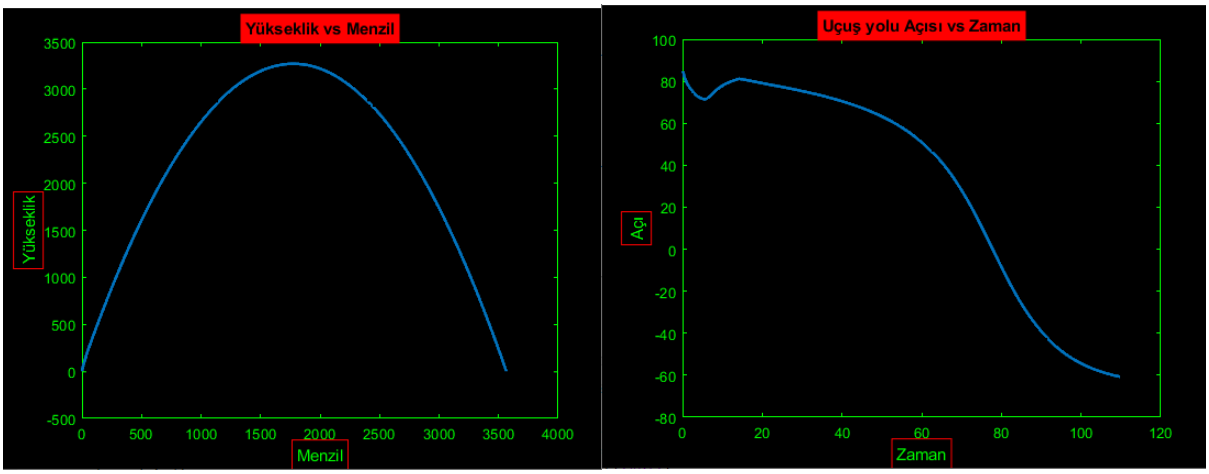
Ardından zamana bağlı dikey ve yatay ivme değerleri hesaplanmıştır. Grafikler aşağıdaki gibidir.



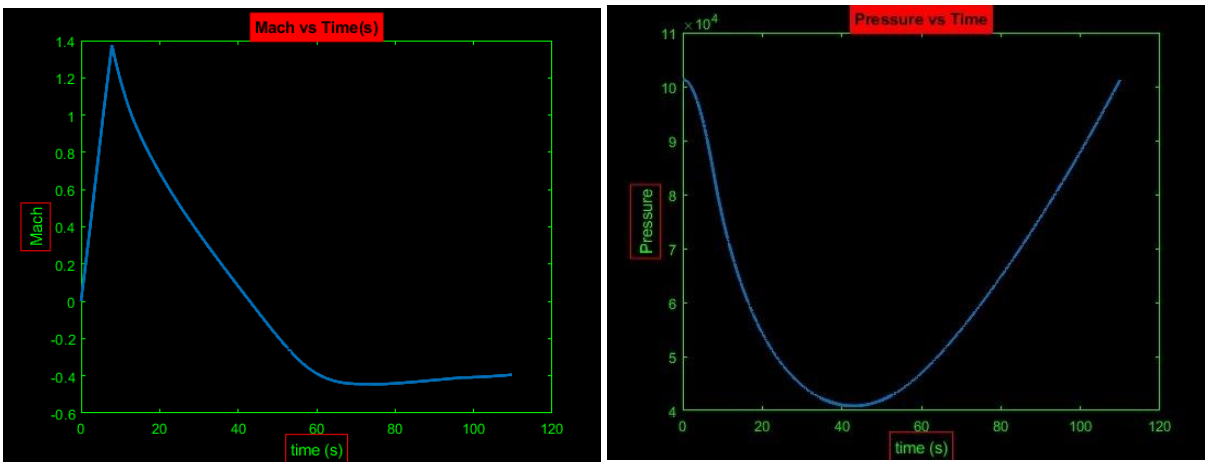
İrtifa ve menzil değerlerinin zamana göre değişimini gösteren grafikler aşağıdadır.



Ek olarak, yükseklik menzil grafiği ve uçuş yolu açısı grafiği aşağıdadır.



Ardından mach zaman grafiği ve basınç zaman grafiği aşağıdadır.




```
Command Window
>>   fprintf('Maximum dinamik basınç(s):  %d \n ',max(PP));
Maximum dinamik basınç(s):  101325
```

	OpenRocket Değeri (a)	Benzetim Değeri (b)	Yüzdece Fark (b-a)/a*100
Maksimum Mach Sayısı [-]	0.8	1.37	1.046685
Tepe Noktası Pozisyonu [m]	[1181 , 0, 3059]	[1776, 0 , 3269]	50.8
Tepe Noktası Hızı (bileşke) [m/s]	26	69	165
Tepe Noktası Mach Sayısı [-]	0.78	0.334	57.17
Tepe Noktası Zamanı [s]	25.1	31.04	23.66

REFERANSLAR

[1] https://pages.vassar.edu/magnes/2019/05/12/computational-simulation-of-rocket-trajectories/

[2] http://www.braeunig.us/space/atmmodel.htm

[3] http://www.rocketmime.com/rockets/rckt_sim.html#Drag