AE 240 Assignment

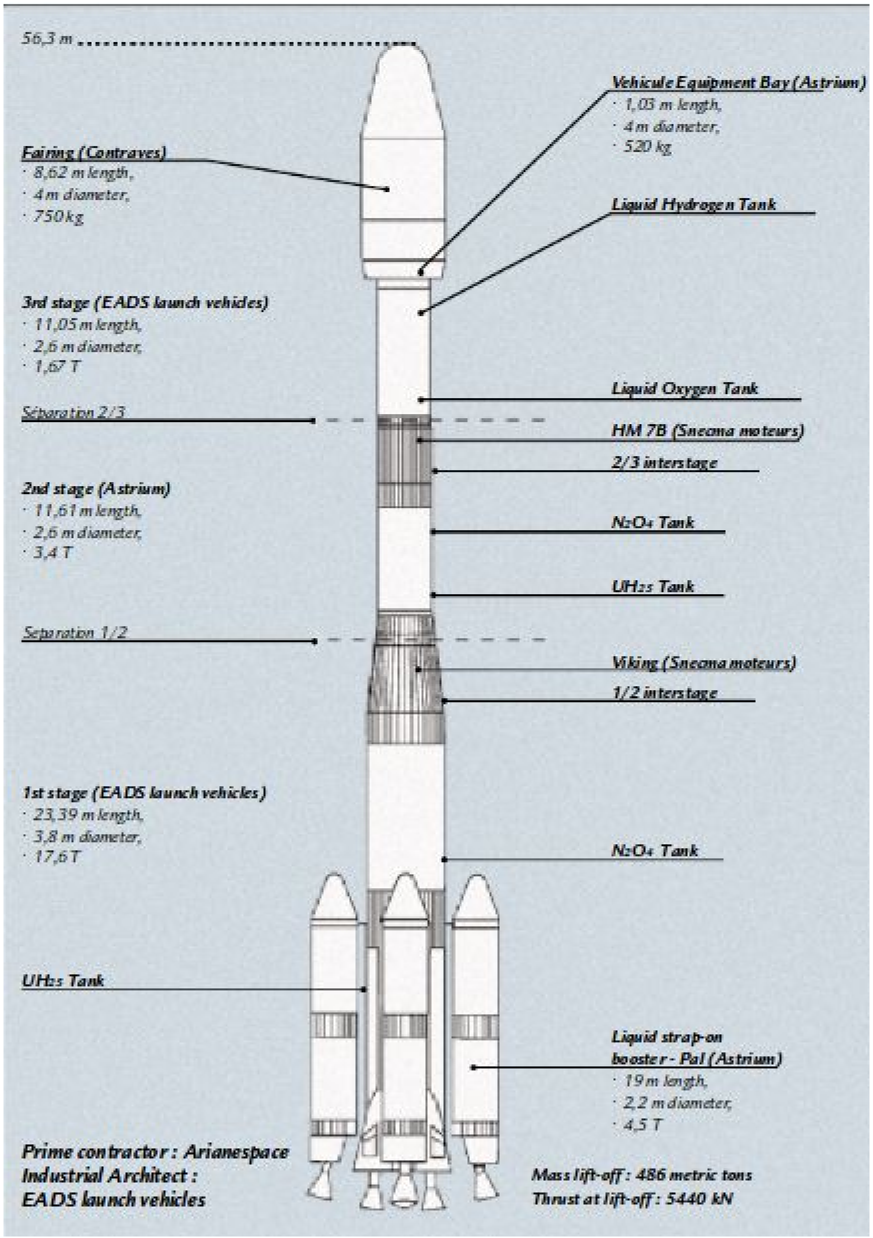
ARIANE 4, Intelsat 901

Achyutha Krishna Koneti, 140070053

17th April, 2018

Introduction

Intelsat 901 (IS-901) was the first of 9 new Intelsat satellites launched in June 2001 at 342°E, providing Ku-band spot beam coverage for Europe, as well as C-band coverage for the Atlantic Ocean region, and provides features such as selectable split uplink for SNG, tailored for increased communications demands such as DTH and Internet.



First, the satellite will be injected into a GTO from which it will be maneuvered into the geostationary orbit.

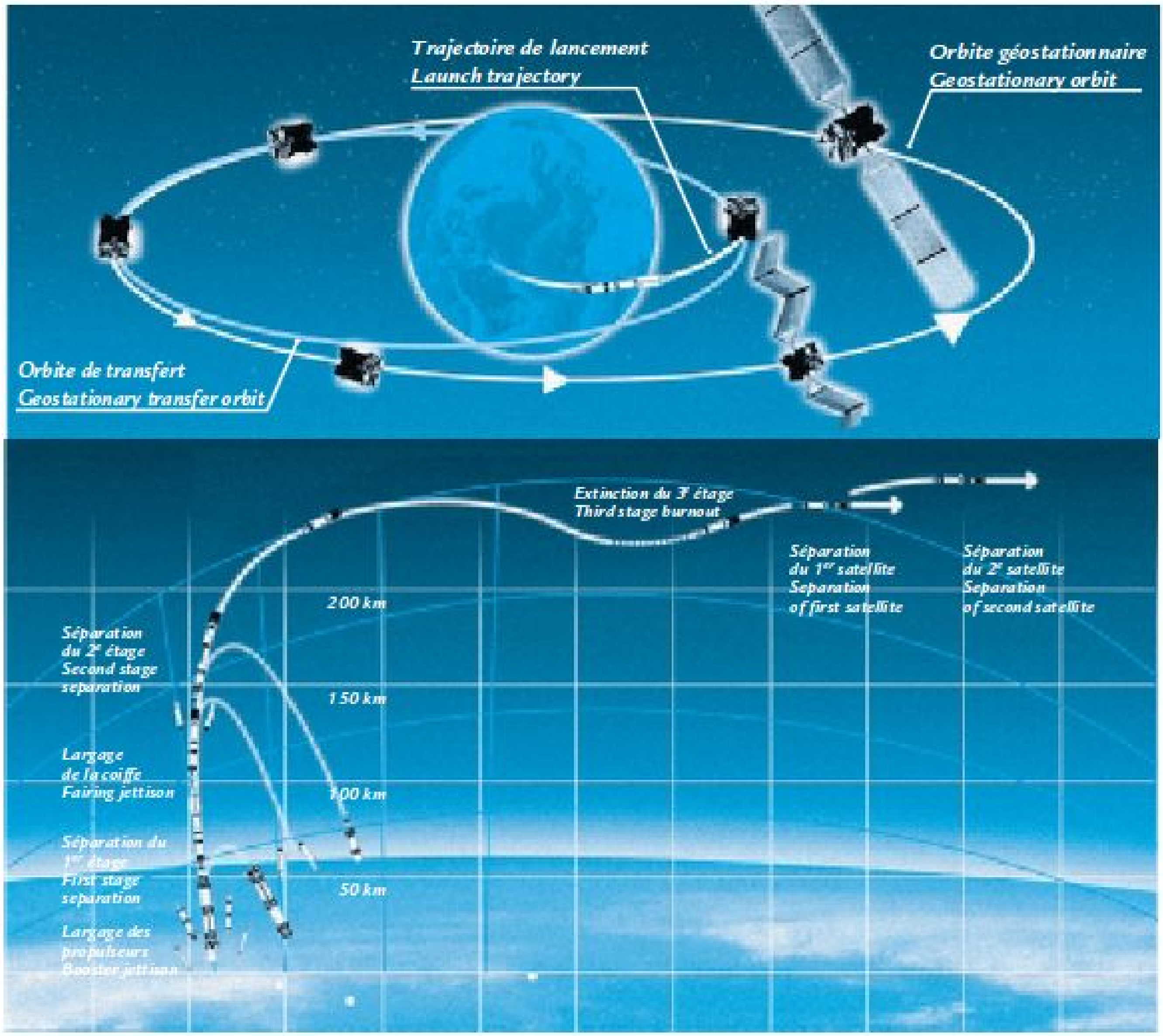
Injection Orbit:

Perigee	225 Km
Altitude apogee	35805 Km
Inclination	7 Degrees

Satellite Name: Intelsat 901 (IS-901)
Status: active
Position: 18° W (18° W)
Launch date: 9-Jun-2001
Launch mass (kg): 4723
Dry mass (kg): 1972
Power (kW): 8.6
Perigee (km): 35770
Apogee (km): 35805
Period (min): 1436.1
Orbit: GEO

Trajectory:

TYPICAL TRAJECTORY FOR STANDARD GEOSTATIONARY TRANSFER ORBIT AND GROUND STATION VISIBILITY



Various Events in the mission:

HO	*Ignition of first stage and liquid strap-on boosters engines*
+ 4,4 s	*Lift-off.*
+ 16 s	*End of vertical ascent phase of pitch motion (10 s duration).*
+ 2 mn 30 s	*Liquid strap-on booster jettison.*
+ 3 mn 31 s	*First stage separation.*
+ 3 mn 34 s	*Second stage ignition.*
+ 4 mn 24 s	*Fairing jettison.*
+ 5 mn 43 s	*Second stage separation.*
+ 5 mn 48 s	*Third stage ignition.*
+ 6 mn 30 s	*Launcher acquired by Natal station.*
+ 12 mn 30 s	*Launcher acquired by Ascension Island station.*
+ 17 mn 30 s	*Launcher acquired by Libreville station.*
+ 18 mn 49 s	*Third stage shutdown sequence.*
+ 20 mn 56 s	*INTELSAT 901 separation.*
+ 22 mn 13 s	*Start of the third stage avoidance maneuver.*
+ 24 mn 10 s	*End of ARIANESPACE Flight 141 mission.*

Data of Rocket:

==

gross_mass["boosters"] = 43772*4

gross_mass["stage1"] = 245900

gross_mass["stage2"] = 37130

gross_mass["stage3"] = 12310

gross_mass["GTO"] = 4723

==

inert_mass["boosters"] = 4493*4

inert_mass["stage1"] = 17900

inert_mass["stage2"] = 3625

inert_mass["stage3"] = 1570

==

ISP["boosters"] = 278

ISP["stage1"] = 278

ISP["stage2"] = 296

ISP["stage3"] = 446

==th

rust["liftoff"] = 5440000

thrust["boosters"] = 752003

thrust["stage1"] = 3034100

thrust["stage2"] = 720965

thrust["stage3"] = 62703

==bu

rntime["boosters"] = 142

burntime["stage1"] = 205

burntime["stage2"] = 129

burntime["stage3"] = 781

==

Calculations:

Stage1:

Assuming constant beta which is sum of beta_boosters and beta_stage1
Beta = 2218 and vertical motion

m(t) = Mo - beta*t

$$V_b(t) = g_0 I_{sp} \ln \frac{m_0}{(m_0 - m_p)} - \tilde{g}\left(\frac{m_p}{\beta}\right)$$

<u>End of vertical ascent:</u>

V = 69.88 m/s

M = 454241.0 Kg

H = 154.38 m

After this a pitch kick of 0.24 degrees has been given
Now the motion is proper gravity turn with uniform pitch rate of g *sin(Theta_o)/V = 0.00128

$$h(\theta) = \frac{\tilde{g}}{4q_0^2}(\cos 2\theta_0 - \cos 2\theta) + h_0$$

$$\ln \frac{m_0}{m} = \frac{2\tilde{g}}{q_0 g_0 I_{sp}}(\sin\theta - \sin\theta_0)$$

$$V(t) = \frac{\tilde{g}\sin\theta}{q_0}$$

$$x(\theta) = \frac{\tilde{g}}{2q_0^2}\left[(\theta-\theta_0) - \frac{(\sin 2\theta - \sin 2\theta_0)}{2}\right] + x(\theta_0)$$

When boosters are ejected

V = 2853.4237804073123 m/s

H = 9895 m X = 1169 m

M = ~ 160.7T theta = 10.86 deg

When stage 1 ends

V = 3984.7 m/s

H = 20.1 Km X= 3.4 Km

M ~= 89.2 T theta = 15.36 deg

Stage2:

With the above initial conditions turn with constant pitch rate will be pursued

At the end of stage 2

V = 5998 m/s

H = 50.6 Km X= 14.5 Km

M ~= 39.2 T theta = 25 deg

Stage3:

With above initial conditions turn with constant pitch rate , at the end of mission

V = 9848.5 m/s

H = 209.6 Km X= 260 Km

M ~= 4.8 T theta = 82.33 deg

Since this is close to the injection GTO, we will go ahead with orbit maneuver

Velocity at apogee of this orbit(209.6 Km perigee and 35805 Km apogee altitudes) = 1597.7 m/s

Velocity at apogee of almost geostationary orbit's apogee(35805, 35770) = 3073.33 m/s

Delta_V = 1475.6 m/s in direction of motion to increase the speed as shown in trajectory diagram

Code:

Link to the github for this project is [here]

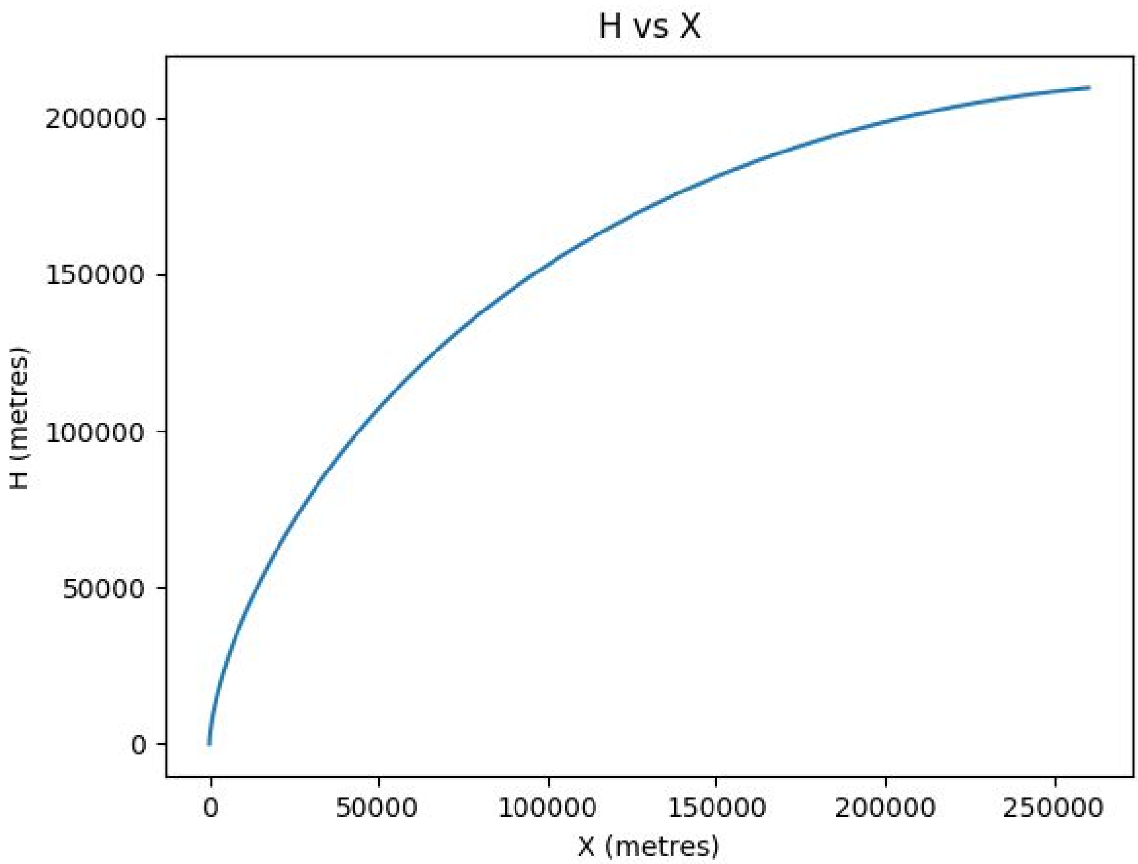
```
from consts import *
from data import *
import math
import matplotlib.pyplot as plt


def timperiod(perigee_alt, apogee_alt):
    a = (perigee_alt + apogee_alt)/2.0 + r_e
    return (4*(pi**2)*(a**3)/mu)**0.5

def velocity(hp, ha, perigee=False):
    h = (2* mu * (hp+r_e) * (ha+r_e)/ (hp + ha + 2*r_e))**0.5
    if(perigee):
        return h/(hp+r_e)
    else:
        return h/(ha+r_e)

def beta(_type):
    return (gross_mass[_type]-inert_mass[_type])/burntime[_type]

def g(height):
    return go/((1+(height/r_e))**2)

time = [i for i in xrange(20*60+56+1)]

event_list = sorted(events.keys())

V = [0.0]
M = [float(sum(gross_mass.values()))]
#M = [486000.0]
H = [0.0]
X = [0.0]

G = [go]

Theta_o = 0.24*pi/180.0
Theta = [Theta_o]

for t in time[1:events["7third stage end"]+1]:
    if t <= events["0end of vertical ascent"]:
        _beta = (beta("boosters") + beta("stage1"))
```

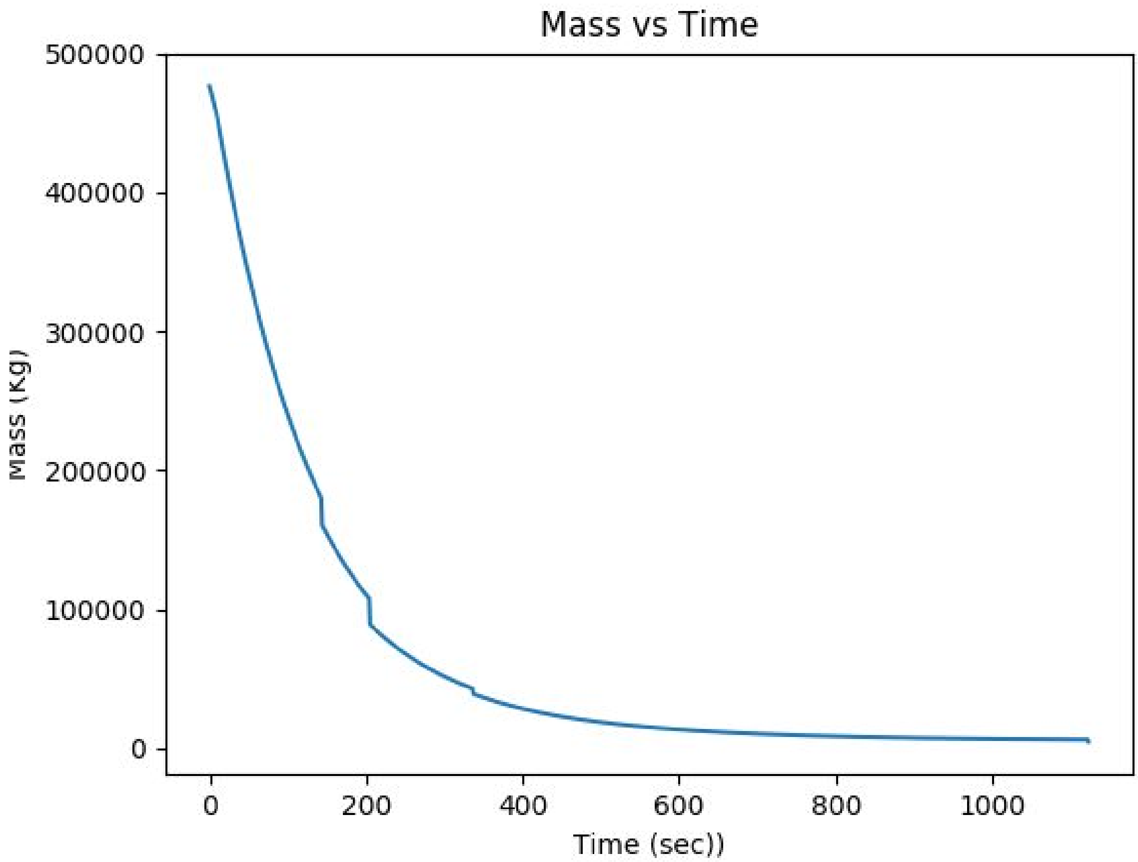
```
        M.append(M[-1]- _beta)
        _V = (M[-1]/M[0])
        V.append(-1*go*ISP["boosters"]*math.log(_V)-G[-1]*t)
        H.append( (M[0]*go*ISP["boosters"]/_beta)* (_V*math.log(_V)+(1-_V)) - 0.5*G[-1]*(t**2))
        G.append(g(H[-1]))
        X.append(0)
        if(t == events["0end of vertical ascent"]):
            qo = G[-1] * math.sin(Theta_o)/V[-1]
            Mo = M[-1]
            To = t
            Ho = H[-1]

    elif t <= events["1booster ejection"]:
        Theta.append(Theta[-1]+qo)
        H.append(G[-1]/(4*(qo**2))*(math.cos(2*Theta[0]) - math.cos(2*Theta[-1])) + Ho )
        X.append(G[-1]/(2*(qo**2))*( Theta[-1]-Theta[0]-
1/2.0*(math.sin(2*Theta[-1])-math.sin(2*Theta[0]))))
        M.append(Mo* (math.e**(-2*G[-1]/(qo*go*ISP["boosters"]) * (math.sin(Theta[-1]) -
math.sin(Theta[0]) ))) )
        V.append(G[-1]*math.sin(Theta[-1])/ qo)
        G.append(g(H[-1]))
        if(t == events["1booster ejection"]):
            M[-1] = M[-1] - inert_mass["boosters"]
            M1 = M[-1]
            T1 = t
            H1 = H[-1]
            X1 = X[-1]
            Theta_1 = Theta[-1]
        #pass

    elif t <= events["2first stage seperation"]:

        Theta.append(Theta[-1]+qo)
        H.append(G[-1]/(4*(qo**2))*(math.cos(2*Theta_1) - math.cos(2*Theta[-1])) + H1 )
        X.append(G[-1]/(2*(qo**2))*( Theta[-1]-Theta_1-
1/2.0*(math.sin(2*Theta[-1])-math.sin(2*Theta_1))) + X1 )
        M.append(M1* (math.e**(-2*G[-1]/(qo*go*ISP["stage1"]) * (math.sin(Theta[-1]) -
math.sin(Theta_1) ))) )
        V.append(G[-1]*math.sin(Theta[-1])/ qo)
        G.append(g(H[-1]))

        if(t == events["2first stage seperation"]):
            M[-1] = M[-1] - inert_mass["stage1"]
            #M[-1] = M[0] - gross_mass["boosters"] - gross_mass["stage1"]
            M3 = M[-1]
            T3 = t
            H3 = H[-1]
```

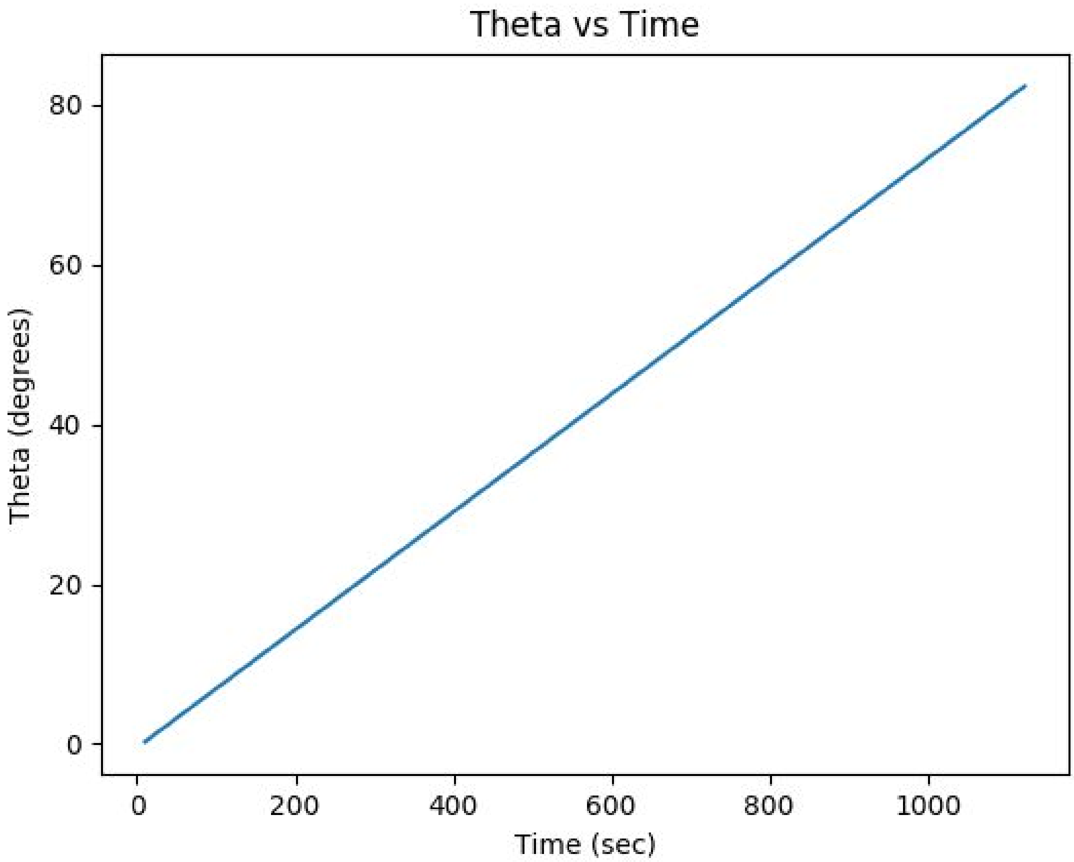
```
        X3 = X[-1]
        Theta_3 = Theta[-1]

    #elif t <= events['3second stage ignition']:
    #elif t <= events['4fairing ejection']:
    elif(t <= events['5second stage seperation']):
        Theta.append(Theta[-1]+qo)
        H.append(G[-1]/(4*(qo**2))*(math.cos(2*Theta_3) - math.cos(2*Theta[-1])) + H3 )
        X.append(G[-1]/(2*(qo**2))*( Theta[-1]-Theta_3-
1/2.0*(math.sin(2*Theta[-1])-math.sin(2*Theta_3))) + X3 )
        M.append(M3* (math.e**(-2*G[-1]/(qo*go*ISP["stage2"]) * (math.sin(Theta[-1]) -
math.sin(Theta_3) ))) )
        V.append(G[-1]*math.sin(Theta[-1])/ qo)
        G.append(g(H[-1]))

        if(t == events["5second stage seperation"]):
            #M[-1] = M[0] - gross_mass["boosters"] - gross_mass["stage1"] -
gross_mass["stage2"]
            M[-1] = M[-1] - inert_mass["stage2"]
            M4 = M[-1]
            T4 = t
            H4 = H[-1]
            X4 = X[-1]
            Theta_4 = Theta[-1]
    #elif t <= events['6third stage ignition']:
    #    pass
    elif t <= events['7third stage end']:
        Theta.append(Theta[-1]+qo)
        H.append(G[-1]/(4*(qo**2))*(math.cos(2*Theta_4) - math.cos(2*Theta[-1])) + H4 )
        X.append(G[-1]/(2*(qo**2))*( Theta[-1]-Theta_4-
1/2.0*(math.sin(2*Theta[-1])-math.sin(2*Theta_4))) + X4 )
        M.append(M4* (math.e**(-2*G[-1]/(qo*go*ISP["stage2"]) * (math.sin(Theta[-1]) -
math.sin(Theta_4) ))) )
        V.append(G[-1]*math.sin(Theta[-1])/ qo)
        G.append(g(H[-1]))

        if(t == events["7third stage end"]):
            M[-1] = M[-1] - inert_mass["stage3"]

clearCanvas()
plt.plot(X, H)
plt.savefig("H_X.png")
plt.close()

plt.plot(time[:len(M)], M)
plt.savefig("M.png")
plt.close()
```

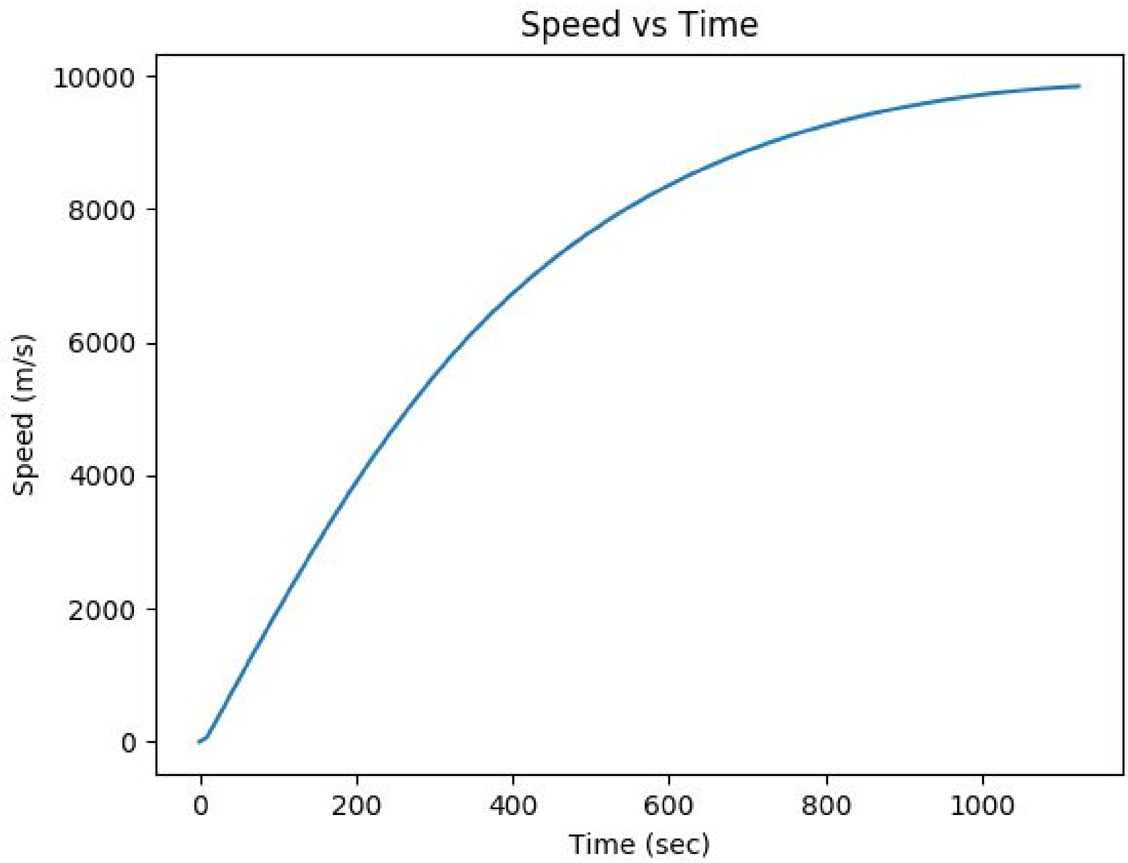
```
plt.plot(time[:len(V)], V)
plt.savefig("V.png")
plt.close()

plt.plot([i*180/pi for i in Theta])
plt.savefig("Theta.png")
plt.close()
```

Plots:


H vs X
H (metres)
X (metres)
0
50000
100000
150000
200000
250000
0
50000
100000
150000
200000


Mass vs Time
500000
400000
300000
200000
100000
0
Mass (kg)
0
200
400
600
800
1000
Time (sec))


Theta vs Time
Theta (degrees)
80
60
40
20
0
0
200
400
600
800
1000
Time (sec)


Speed vs Time
10000
8000
6000
4000
2000
0
Speed (m/s)
0
200
400
600
800
1000
Time (sec)

Sources:

https://www.satbeams.com/satellites?norad=26824

https://en.wikipedia.org/wiki/Intelsat_901

http://www.astronautix.com/a/ariane44l.htm