The University Physics Competition

Problem A
Sending an ion drive propelled spacecraft to Saturn

Team: 218

Contents

List of Figures

List of Tables

Notations

G	Gravitational Constant, $g = 6.67 \times 10^{-11}\text{Nm}^2\text{kg}^{-}2$
a	Semi-Major Axis
b	Semi-Minor Axis
v_i	Initial Velocity
v_f	Final Velocity
v_e	Exhaust Velocity
m_0	Initial Mass of the Spacecraft
$m_f(t)$	Mass of the Spacecraft at t
$\dot{v}$	$\frac{dv}{dt}$
∂_x v	$\frac{\partial v}{\partial x}$
I_{sp}	Specific Impulse
g_0	Standard Gravity, $g_0 = 9.80665\ \text{ms}^{-2}$
EOMs	Equations of Motion
μ	$G\ M$
SOI	Sphere of Influence
GTO	Geostationary Transfer Orbit

Table 1: Notations Used

Problem Statement - Ion Thrusters to Saturn

A spacecraft is in a circular orbit around the Earth, with an orbital period of 90 minutes. The spacecraft has a total mass of 5,000 kg, including fuel. The spacecraft is equipped with ion thrusters that can provide a constant thrust of 400 millinewtons, with a specific impulse of 4,000 seconds. Our goal is to place the spacecraft into a circular orbit around Saturn, with an orbital period of 40 hours. What is the minimum amount of fuel this will take? What will be the duration of the trip? How should the thrusters be controlled during the journey?

1 Abstract

This paper consists of the analysis of finding a path to place a spacecraft into a circular orbit around Saturn following a certain parameters. An attempt is made to find the best case scenario of the trajectory of the spacecraft. The spacecraft uses ion thrusters unlike regular chemical propulsion. These ion thrusters ionizes a neutral gas by extracting some electrons out of atoms, creating a cloud of positive ions and creates thrust by accelerating ions.
Our goal was to minimise the fuel required which would allow us to carry more payload on board, while also keeping the a considerate time of flight. Our approach was inspired by the Cassini's tour trajectory to reach Saturn and the spacecraft Deep Space 1.

2 Introduction

Multiple successful Saturn missions has been conducted to study the planet and its system. Previous missions include

All the missions were conducted using fuel propellant based spaceships. Our proposed mission will be using Ion thrusters, which is a form of electric propulsion. Konstantin Tsiolkovsky in 1911 first proposed the idea of ion propulsion. It was initially proposed for near-vacuum conditions at high altitude, but thrust was demonstrated with ionized air streams at atmospheric pressure. Later, in 1959, Harold R. Kaufman at the NASA built a working ion thruster.

Mission	Launch Date	Type
Pioneer 11	September 1979	Flyby
Voyager	November 1980	Flyby
Cassini	July 2004	Orbiter

Table 2: Previous Missions to Saturn

It just uses fundamental Newton's laws of dynamics. Since a rocket propelled spacecraft in free flight derives its only acceleration from discharge of propellant mass, its equation of motion follows directly from conservation of the total momentum of the spacecraft and its exhaust stream:

$$m\,\dot{v} \;=\; -\dot{m}\,v_e \tag{1}$$

where,

m is the mass of the spacecraft at any given time

$\dot{v}$ is the acceleration vector

v_e is the velocity vector of the exhaust jet relative to the spacecraft

$\dot{m}$ is the rate of change of the Spacecraft's mass due to Propellant-Mass Expulsion.

The product $\dot{m}v_e$ is called the thrust of the rocket, T. Δv characteristic velocity increment, is a generalized indicative of the energetic difficulty of the particular mission or maneuver, where Δv is

$$\Delta v \;=\; v_f - v_i \;=\; v_e\,\ln\left(\frac{m_0}{m_f}\right) \tag{2}$$

for missions of large Δv, the burden of thrust generation must shift from high rates of ejection of propellant mass to high relative exhaust velocities. conventional chemical rockets are fundamentally limited by their available combustion reaction energies and heat transfer tolerances to exhaust speeds of a few thousand meters per second, whereas many attractive space missions entail characteristic velocity increments at least an order of magnitude higher, so we resort to Electric Propulsion.

Electric Propulsion Electric Propulsion is classified into to 3 categories.

1. Electrothermal propulsion, wherein the propellant is heated by some electrical process, then expanded through a suitable nozzle.
2. Electrostatic propulsion, wherein the propellant is accelerated by direct application of electrostatic forces to ionized particles.
3. Electromagnetic propulsion, wherein the propellant is accelerated under the combined action of electric and magnetic fields.

Ion Thrusters is a type of Electrostatic propulsion. Ion thrusters have been used in multiple in-space missions. It was first demonstrated in Space Electric Rocket Test (SERT) test missions and was successful. Later, Multiple space missions have used ion thrusters.

Spacecraft	Launch year	Max Acceleration (m/s^2)
Deep Space 1[1]	1998	1.892×10^{-4}
Hayabusa[2]	2003	4.706×10^{-4}
Dawn	2007	7.473×10^{-5}

Ion Thruster Technology An ion thruster consists of three basic parts:-

1. Ion Sources - Cesium, Mercury, Argon, Krypton, and most commonly Xenon can be used but Cesium and mercury are not used as they erode the grids in the thrusters.
2. Accelerator Grids - Positive ions are extracted from the source and accelerated downstream by a system of grids configured to achieve the desired exhaust velocity with minimum beam impingement.
3. Neutralizers – The ejected ions are electrostatically neutralized by providing a flux of electrons.

2.1 Working

Propellant atoms are injected into the discharge chamber and are ionized by electron bombardment, forming a plasma and are accelerated by potential difference with the anode, he electrons can be accelerated by an oscillating electric field. The positively charged ions diffuse towards the chamber's extraction system (2 or 3 multi-aperture grids). After ions enter the plasma sheath at a grid hole, they are accelerated and pushed out. The negative voltage of the accelerator grid prevents electrons of the beam plasma outside the thruster from streaming back to the discharge plasma. Lower-energy electrons are emitted from a separate cathode, called the neutralizer, into the ion beam to ensure that equal amounts of positive and negative charge are ejected. Neutralizing is needed to prevent the spacecraft from gaining a net negative charge, which would attract ions back toward the spacecraft and cancel the thrust.

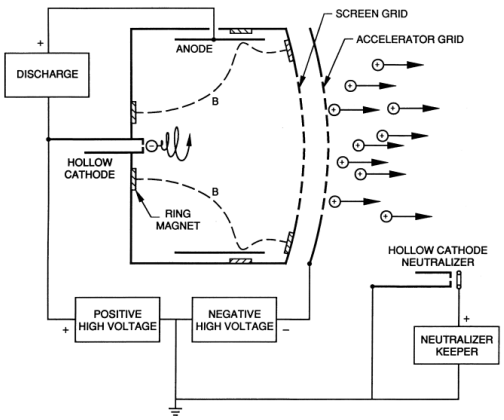


Figure 1: Ion Thruster

3 Initial Conditions

$$\begin{aligned}
\text{Initial Mass, } m_0 &= 5000\ kg \\
\text{Specific Impulse, } I_{sp} &= 4000\ s \\
\text{Time Period of the Initial Orbit, } P_i &= 90\ min \\
\text{Radius of the Initial Orbit, } R_i &\approx 281 + R_{earth}\ km \\
\text{Time Period of the Final Orbit, } P_f &= 40\ hr \\
\text{Radius of the Final Orbit, } R_i &\approx 271 \times 10^5 km \\
\text{Thrust, } T &= 400\ mN
\end{aligned}$$

4 Tsiolkovsky Rocket Equation

Known as classical Rocket Equation or Ideal Rocket Equation

$$\Delta v = v_f - v_i = v_e \ln\left(\frac{m_0}{m_f}\right) = I_{sp}\, g_0 \ln\left(\frac{m_0}{m_f}\right) \tag{3}$$

where,

v_i is the initial velocity

v_f is the final velocity

m_0 is the initial total mass

m_f is the final total mass

$v_e = I_{sp}\, g_0$ is the exhaust velocity

I_{sp} is the Specific Impulse

$g_0 = 9.80665\ \text{ms}^{-2}$ is the Standard Gravity

It is a mathematical equation that describes the motion of a body that can apply thrust to itself by expelling some of its mass at high velocity.

Acceleration Based Form:

$$a = -\frac{p\, v_e}{m(t)} \tag{4}$$

Mass Form:

$$m_f = m_0 \exp\left(-\frac{\Delta v}{v_e}\right) \tag{5}$$

Note for Low-Thrust Propulsions: In low-thrust propulsion since the burn duration is very long the accuracy of the results decreases, although, using the acceleration based form(Eq. (4)) to numerically integrate for small enough burn duration will be effective.

5 Low Thrust Maneuvers

5.1 Spiral Climb/Descent

Note: This Maneuver is helpful during the exit from Earth SOI.

Equations of Motion for a body in central field are,

$$\frac{d^2r}{dt^2} - r\frac{d\theta}{dt} + \frac{\mu}{r^2} = a_r \tag{6}$$

$$\frac{d^2\theta}{dt^2} + \frac{2}{r}\frac{dr}{dt}\frac{d\theta}{dt} = \frac{a_\theta}{r} \tag{7}$$

Here we consider the thrust is null in the radial direction and acceleration is only directed towards tangential direction.

If the acceleration of the spacecraft is small enough, then extrapolating from the symmetrical nature of the initial circular orbit it's pretty safe to assume the orbit would remain circular.

Using Eq. (6) and setting a_r and $\frac{d^2r}{dt^2}$ to Zero,

$$\therefore \quad \frac{d\theta}{dt} = \sqrt{\frac{\mu}{r^3}} \tag{8}$$

Differentiating again,

$$\frac{d^2\theta}{dt^2} = -\frac{3}{2}\sqrt{\frac{\mu}{r^5}}\frac{dr}{dt} \tag{9}$$

Substituting in Eq. (7) and integrating with consideration to the initial condtition r_0 and t_0,

$$\frac{1}{\sqrt{r_0}} - \frac{1}{\sqrt{r}} = \frac{a_\theta}{\sqrt{\mu}}\,(t - t_0) \tag{10}$$

By Manipulation,

$$r = \frac{r_0}{(1 - a\theta t/v_0)} \tag{11}$$

$$\text{where,} \quad v_0 = \sqrt{\frac{\mu}{r_0}} \text{ and } t = 0$$

From Eq. (11) it is clear that the trajectory will be Spiral, climbing or decaying will depend upon a_θ. This equation will be consistent for our use, since the average acceleration will be very less, of the order of 10^{-5} in this Earth-to-Saturn Scenario. However, the catch is the thrust vector should always remain normal to the position vector which is certainly not logical, hence, small eccentricity would be established over time, but for the initial phase it remains of interest.

5.2 Hohmann Transfer Orbit

Hohmann Transfer Orbit is a transfer elliptical orbit used to transfer between two circular orbits of different radii. It mostly uses the least amount of fuel in transferring from one orbit to another.

It requires a particular alignment to transfer which restricts to only some launch-windows.

Hohmann orbits requires high thrust engines to maintain the orbit, however, low-thrust engines can perform a very good approximation of the hohmann orbit by careful and gradual acceleration.

For low-thrust only maneuvers, then continuously firing the low-thrust, while having very high-efficiency engine would be a good approximation.

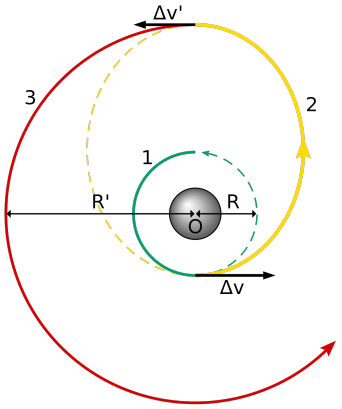


Figure 2: Hohmann Transfer Orbit

Using the **vis-viva equation**, (also referred to as Orbital Energy Invarience Law),

$$v = \sqrt{\mu\left(\frac{2}{r} - \frac{1}{a}\right)} \tag{12}$$

The delta-v required for Hohmann Transfer can be calculated easily,

$$\Delta v_1 = \sqrt{\frac{\mu}{r_1}}\left(\sqrt{\frac{2r_2}{r_1 + r_2}} - 1\right) \tag{13}$$

$$\Delta v_2 \;=\; \sqrt{\frac{\mu}{r_2}}\left(1-\sqrt{\frac{2r_1}{r_1+r_2}}\right) \tag{14}$$

The total delta-v is given by,

$$\Delta v \;=\; \Delta v_1 + \Delta v_2 \tag{15}$$

Using the Kepler's third Law, the total time of flight for the transfer is,

$$t_f \;=\; \pi\sqrt{\frac{(r_1+r_2)^2}{8\,\mu}} \tag{16}$$

5.3 Low-Thrust Acceleration Profiles

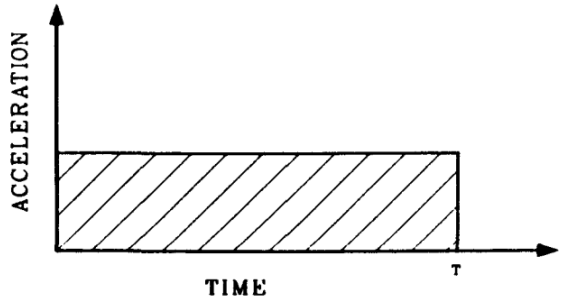


Figure 3: Optimum acceleration program for obtaining a predetermined velocity in a given time with the least propellant.[Keaton(2002)]

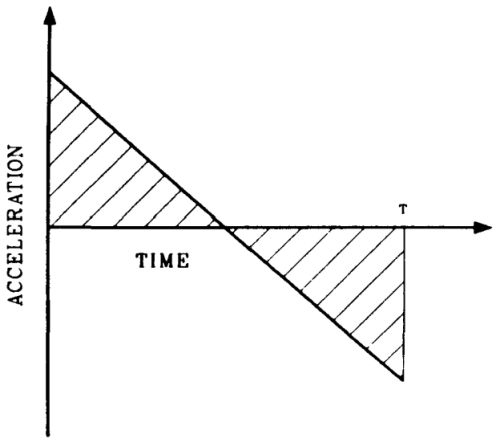


Figure 4: Optimum acceleration program for traveling from one point to another point in a given time with the least propellant.[Keaton(2002)]

5.4 Lambert's Problem

Lambert's problem, solved by 'Lagrange' gives us Lambert's Theorem which states that the transfer time of a body moving between two points on a conic trajectory is a function only of the sum of the distances of the two points from the origin of the force, the linear distance between the points, and the semi-major axis of the conic.

$$\ddot{r} \;=\; -\mu \cdot \frac{\hat{r}}{r^2} \tag{17}$$

Lambert's problem is the boundary value problem for the differential equation. Lambert's formula is applied for two different time t_1, t_2 and position vectors $\bar{r_1} = r_1\hat{r_1},\;\; \bar{r_2} = r_2\hat{r_2}$

where,

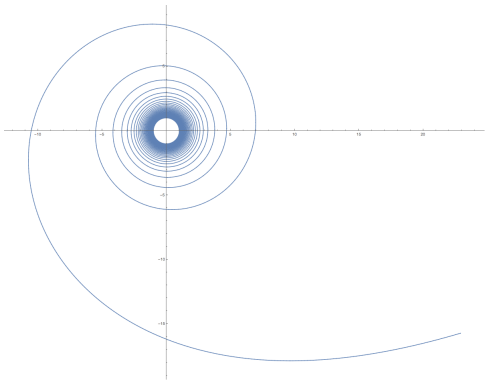
r_1 is the initial position vector of orbit

r_2 is the final position vector of orbit

6 Orbit Calculations

6.1 Maneuver I: TO GTO

It'll basically be the setup maneuver for Maneuver II, we will use the subsection 5.1, Spiral Climb Maneuver to get to the GTO, at around 35000 km from our Initial Orbit.



We will keep the integration in the burns as low as possible to reduce the involvement of eccentricity.

The Orbit can be represented by a climbing spiral such as the one in the figure,

6.1.1 Orbital Parameters

Burns Timings are after 2021-05-01 00:01

1. Burn I

Burn At	1965.71 s
m_0	5000 kg
m_f	4856.020 kg
Δv	1.146149 km/s
Burn Duration	0.45 yr
Low-Thrust Equiv. Parameters	
Burn Start Time	-7057825.45 s
Burn Stop Time	7061756.87 s

Table 3: Burn I

2. Burn II

Burn At	18518.78 s
m_0	4856.020 kg
m_f	4747.22 kg
Δv	0.888861 km/s
Burn Duration	0.34 yr
Low-Thrust Equiv. Parameters	
Burn Start Time	-5316221.30 s
Burn Stop Time	5353258.86 s

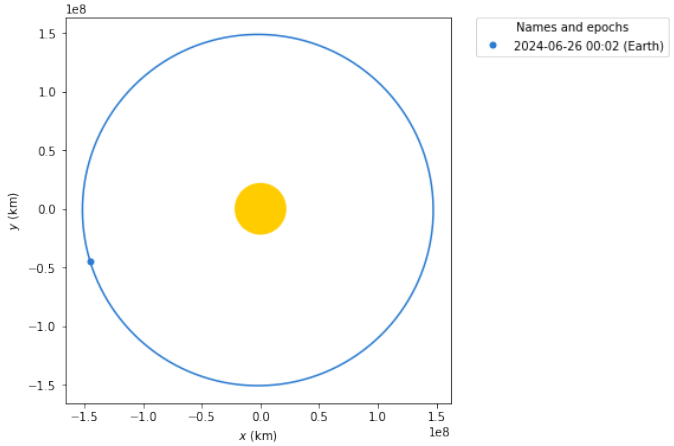
Table 4: Burn II

6.2 Maneuver II: Transfer Orbit

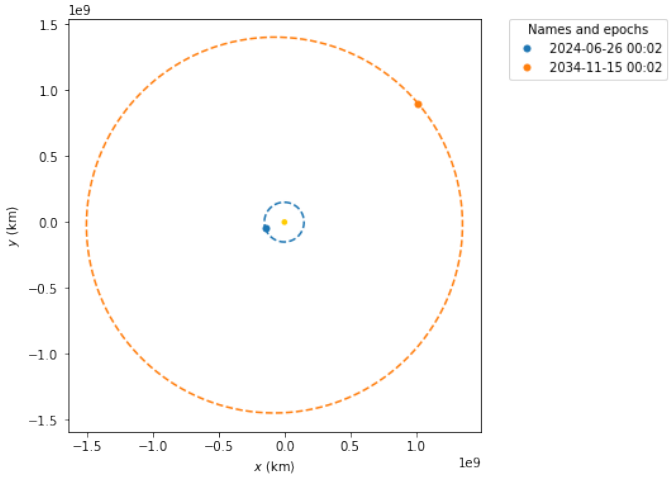
6.2.1 Orbital Parameters

Burns Timings are after 2024-06-01 00:02

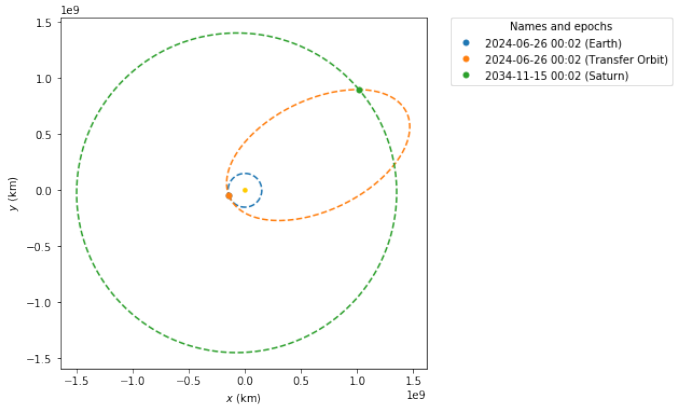
1. Burn III
2. Burn IV



(a) Earth Orbit at Launch Date



(b) Earth and Saturn Orbit at Launch and Arrival Date Resp.



(c) Transfer Orbit

Figure 5: Maneuver II

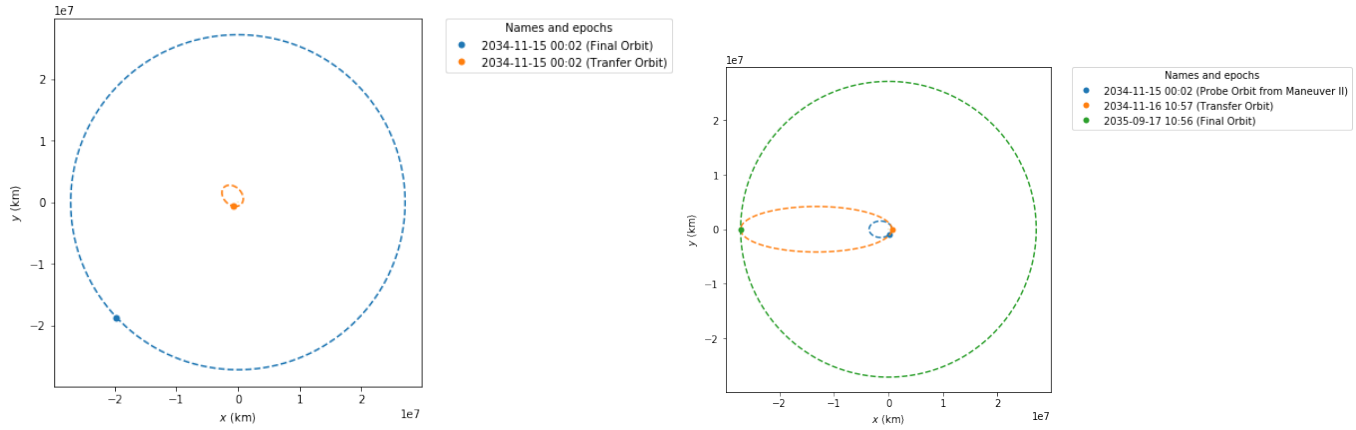
Burn At	0 s
m_0	4747.22 kg
m_f	3604.69 kg
Δv	10.80 km/s
Burn Duration	3.55 yr
Low-Thrust Equiv. Parameters	
Burn Start Time	-56021954.70 s
Burn Stop Time	56021954.70 s

Table 5: Burn III

Burn At	$3.278\,016 \times 10^8$ s
m_0	3604.69 kg
m_f	2966.00 kg
Δv	7.65 km/s
Burn Duration	1.98 yr
Low-Thrust Equiv. Parameters	
Burn Start Time	296 484 670.8 s
Burn Stop Time	359118529.2 s

Table 6: Burn IV

6.3 Maneuver III: Orbit Insertion



(a) Maneuver III Orbits

(b) Transfer Orbit

Figure 6: Maneuver III

6.3.1 Orbital Parameters

Burns Timings are after 2034-11-15 00:02

1. Burn V

Burn At	125737.09 s
m_0	2966.00 kg
m_f	2910.38 kg
Δv	0.7426 km/s
Burn Duration	0.18 yr
Low-Thrust Equiv. Parameters	
Burn Start Time	-2601636.61 s
Burn Stop Time	398490.79 s

Table 7: Burn V

2. Burn VI

Burn At	26351950.84 s
m_0	2910.38 kg
m_f	2842.30 kg
Δv	0.9286 km/s
Burn Duration	0.21 yr
Low-Thrust Equiv. Parameters	
Burn Start Time	23013544.50 s
Burn Stop Time	29690357.18 s

Table 8: Burn VI

7 Conclusion

The trajectory of the spacecraft was successfully calculated that reached Saturn's orbit in a considerate amount of time that used the least amount of fuel possible. The epoch for starting the burn from parking orbit is 2021-05-01 00:01.

After the Maneuver I, the date of starting the burn for Maneuver II is 2034-11-15 00:02, keep in mind the burn starting time of T - 2601636.61 s.

Overall, it took 2157.7 kg of fuel with its thrusters controlled by Burn Starting and Stopping time mentioned with each Maneuver.

Time of Flight is simply the difference between the initial epoch and the final epoch, which is nearly 14 yr.
2035-09-17 10:56 − 2021-05-01 00:33

This is a direct trajectory and therefore requires less time. If fuel consumption is further required to reduce, gravitational assist or sling-shot orbits can be implemented to further reduce the time and fuel, although optimization would be required in that case also, furthermore, we could take into consideration the effects of the gravitational fields of other astronomical objects during the mission, like Mars and Jupiter.

There are two major advantages of such low-thrust propulsion are: in a strong gravitational field, such as near the Earth, freighter missions with low-thrust propulsion requires much less propellant as conventional chemical engines do, though the time of flight is much longer, the total mission duration may be comparable. Second, in a weak gravitational field, such as occurs at one astronomical unit from the Sun, missions with NEP are actually faster than chemical missions with comparable propellant mass.

A successful evolutionary space program must have efficient transportation vehicles in the supporting infrastructure. For flexibility and the promotion of growth, hybrid rockets combining propulsion system concepts may be the best vehicles. Low-thrust propulsion systems are excellent alternatives to impulse-thrust propulsion systems for unmanned freighter missions in space. Specific examples of low thrust propulsion, such as nuclear-electric propulsion (NEP), often promise superior performance when compared with the best chemical engine that exists. But impulse-thrust rockets are best for whisking people from low earth orbit (LEO) through the Earth's radiation belts and beyond. People might use small chemical rockets to rendezvous with larger NEP rockets near the fringes of the Earth's potential well. A comprehensive mission that will include both types of rockets is the best case scenario for future missions.

8 Acknowledgement

We would like to thanks all the contributors and most importantly creator of poliastro, an open-sourced package for orbital dynamics.

References

[Keaton(2002)] P. W. Keaton. Low-Thrust Rocket Trajectories. (November), 2002.

[Kuninaka et al.(2009)Kuninaka, Nishiyama, Shimizu, Funaki, Koizumi, Hosoda, and Nakata] H. Kuninaka, K. Nishiyama, Y. Shimizu, I. Funaki, H. Koizumi, S. Hosoda, and D. Nakata. Hayabusa Asteroid Explorer Powered by Ion Engines on the way to Earth. *31st International Electric Propulsion Conference*, pages IEPC–2015–267, 2009. URL http://erps.spacegrant.org/uploads/images/images/iepc{_}articledownload{_}1988-2007/2009index/IEPC-2009-267.pdf.

[Martínez-Sánchez and Lozano(2015)] M. Martínez-Sánchez and P. Lozano. Session 6: Analytical Approximations for Low Thrust Maneuvers. *MIT OpenCourseWare 16.522 Space Propulsion*, (6), 2015. URL http://ocw.mit.edu.

[Pritchett et al.(2018)Pritchett, Zimovan, and Howell] R. E. Pritchett, E. M. Zimovan, and K. C. Howell. Impulsive and low-thrust transfer design between stable and nearly-stable periodic orbits in the restricted problem. *Space Flight Mechanics Meeting, 2018*, (210009), 2018. doi: 10.2514/6.2018-1690.

[Rayman et al.(2000)Rayman, Varghese, Lehman, and Livesay] M. D. Rayman, P. Varghese, D. H. Lehman, and L. L. Livesay. Acta Astronautica 47,. *Acta Astronautica*, 475:4–8, 2000.

9 Codes/ Algorithms

```
def new_get_cost(m_0, delta_V):

#-----------------------------------------------------
g_0 = 9.80665 * u.m / (u.second * u.second)
G = 6.67408e-11 * u.m**3 * u.kg**(-1) * u.s**(-2)
M_Earth = 5.972e24 * u.kg
R_Earth = (6371 * u.km).to(u.m)
#-----------------------------------------------------
#-----------------------------------------------------
# TP_orbit = 29499.67820209 * u.s # (90 * u.min).to(u.s)
# m_0 = 5000 * u.kg
m_0 = m_0.to(u.kg)
delta_V = delta_V.to(u.m/u.s)
I_sp = 4000 * u.second
T = 400e-3 * u.kg * u.m  / (u.second * u.second) # newtons
#-----------------------------------------------------
v_e = g_0 * I_sp
p = -1 * T / v_e
m_f = m_0 * np.exp(-1 * (delta_V) / v_e)
Time_Elapsed = (m_f - m_0)/p
a_0 = -1 * p * v_e/ m_0
a_f = -1 * p * v_e/ m_f
print(m_0, m_f)
print(a_0, a_f)
print(delta_V)
print(Time_Elapsed, Time_Elapsed.to(u.year))
new_get_cost(5000*u.kg, 1146.14916511*u.m/u.s)
new_get_cost(4856.020*u.kg, 888.86125395*u.m/u.s)
new_get_cost(4747.22*u.kg, 10.80*u.km/u.s)
new_get_cost(3604.69*u.kg, 7.65*u.km/u.s)
new_get_cost(2966.00*u.kg, 0.7426167361242552*u.km/u.s)
new_get_cost(2910.38*u.kg, 0.9285591733102699*u.km/u.s)d
```

```
        # coding: utf-8

# In[380]:


import matplotlib.pyplot as plt
from astropy import units as u
from poliastro.bodies import Earth, Mars, Sun, Saturn
from poliastro.twobody import Orbit
from poliastro.maneuver import Maneuver
from poliastro.plotting.static import StaticOrbitPlotter
from poliastro.plotting import OrbitPlotter3D
import numpy as np
from astropy import time
from poliastro.twobody import thrust
from poliastro.twobody.propagation import cowell
from poliastro.ephem import Ephem
from poliastro.util import norm, time_range


# In[381]:


date_launch = time.Time("2024-6-26 00:01", scale="utc").tdb # Starting Mission from parking orbit
date_arrival = time.Time("2034-11-15 00:01", scale="utc").tdb


# In[382]:


earth = Ephem.from_body(Earth, time_range(date_launch,end=date_arrival))
saturn = Ephem.from_body(Saturn, time_range(date_launch,end=date_arrival))


# In[383]:


op = StaticOrbitPlotter()
op.plot_body_orbit(Earth, date_launch, label="Earth")


# In[384]:


ss_earth = Orbit.from_ephem(Sun, earth, date_launch)
ss_saturn = Orbit.from_ephem(Sun, saturn, date_arrival)

print(ss_saturn.rv()[0].to(u.km))
print(ss_saturn.rv()[1].to(u.km/u.s))
```

```
# In[385]:


op = StaticOrbitPlotter()
op.plot(ss_earth, label="Earth")
op.plot(ss_saturn, label="Saturn")


# In[282]:


man_lambert = Maneuver.lambert(ss_earth, ss_saturn) # Maneuver II


# In[288]:


ss_trans, ss_target = ss_earth.apply_maneuver(man_lambert, intermediate=True)


# In[366]:


op = StaticOrbitPlotter()
op.plot(ss_earth, label="Earth")
op.plot(ss_trans, label="Transfer Orbit")
op.plot(ss_saturn, label="Saturn")


# In[365]:


plotter = OrbitPlotter3D() # StaticOrbitPlotter() #
plotter.set_attractor(Sun)

plotter.plot_body_orbit(Earth, date_launch, label="Earth at launch position")
plotter.plot_body_orbit(Saturn, date_arrival, label="Saturn at arrival position")#, color="C1")
plotter.plot_trajectory(ss_trans.sample(max_anomaly=350*u.deg), label="Tranfer Orbit")#, color="C2")


# In[364]:


#Impulse
man_lambert.impulses[1][0]


# In[362]:


#Impulse
print(man_lambert.impulses[0][1].to(u.km/u.s))
```

```
print(norm(man_lambert.impulses[0][1].to(u.km/u.s)))


# In[363]:


print(man_lambert.impulses[1][1].to(u.km/u.s))
print(norm(man_lambert.impulses[1][1].to(u.km/u.s)))


# In[386]:


# Maneuver 3


# In[387]:


ss_trans_in = ss_trans.propagate(date_arrival)


# In[388]:


ss_trans_in_sat = ss_trans_in
ss_trans_in_sa = ss_trans_in_sat.change_attractor(Saturn, force=True)


# In[389]:


ss_ = Orbit.from_ephem(Saturn, saturn, date_arrival)
ss_final_inc = ss_.circular(Saturn, alt=271e5*u.km).propagate(date_arrival)


# In[390]:


op = StaticOrbitPlotter()

op.plot(ss_final_inc, label="Final Orbit")
op.plot(ss_trans_in_sa, label="Tranfer Orbit")
# op.plot(ss_saturn_i, label="Probe around Saturn")


# In[391]:


# ss_trans_in_sa = ss_trans_in_sa.propagate(1*u.day)
# # man_inc_final = Maneuver.lambert(ss_final_inc, ss_trans_in_sa)
man_inc_final = Maneuver.hohmann(ss_trans_in_sa, 271e5*u.km )
```

```
ss_trans_final, ss_final = ss_trans_in_sa.apply_maneuver(man_inc_final, intermediate=True)


# In[392]:


op = StaticOrbitPlotter()

op.plot(ss_trans_in_sa, label="Probe Orbit from Maneuver II")
op.plot(ss_trans_final, label="Transfer Orbit")
op.plot(ss_final, label="Final Orbit")


# In[393]:


man_inc_final.impulses


# In[394]:


print(man_inc_final.impulses[1][1].to(u.km/u.s))
print(norm(man_inc_final.impulses[1][1].to(u.km/u.s)))


    import matplotlib.pyplot as plt
from astropy import units as u
from poliastro.bodies import Earth, Mars, Sun, Saturn
from poliastro.twobody import Orbit
# from poliastro.maneuver import Maneuver
from poliastro.plotting.static import StaticOrbitPlotter
from poliastro.ephem import Ephem
# import pandas as pd
import numpy as np
from astropy import time

"""
  Symbol meaning [1 au= 149597870.700 km, 1 day= 86400.0 s]:

    JDTDB    Julian Day Number, Barycentric Dynamical Time
      EC     Eccentricity, e
      QR     Periapsis distance, q (au)
      IN     Inclination w.r.t X-Y plane, i (degrees)
      OM     Longitude of Ascending Node, OMEGA, (degrees)
      W      Argument of Perifocus, w (degrees)
      Tp     Time of periapsis (Julian Day Number)
      N      Mean motion, n (degrees/day)
      MA     Mean anomaly, M (degrees)
      TA     True anomaly, nu (degrees)
      A      Semi-major axis, a (au)
      AD     Apoapsis distance (au)
```

```
      PR     Sidereal orbit period (day)
"""

def read_jpl_file(file_):

    df = pd.read_csv(file_ + ".jpl", delimiter="=", index_col=0, header=None)
    data = df.to_dict()[1]

    data['EC'] = float(data['EC' ].strip(" "))
    data['QR'] = float(data['QR' ].strip(" "))
    data['IN'] = float(data['IN' ].strip(" "))
    data['OM'] = float(data['OM' ].strip(" "))
    data['W '] = float(data['W ' ].strip(" "))
    data['Tp'] = float(data['Tp' ].strip(" "))
    data['N '] = float(data['N ' ].strip(" "))
    data['MA'] = float(data['MA' ].strip(" "))
    data['TA'] = float(data['TA' ].strip(" "))
    data['A '] = float(data['A ' ].strip(" "))
    data['AD'] = float(data['AD' ].strip(" "))
    data['PR'] = float(data['PR'].strip(" "))

    return data

def Planet(planet_name):

    data = read_jpl_file(planet_name)

    return Orbit.from_classical(
        attractor=Sun,
        a = data["A "] * u.AU,
        ecc = data["EC"] * u.one,
        inc = data["IN"] * u.deg,
        raan = data["OM"] * u.deg,
        argp = data["W "] * u.deg,
        nu = data["TA"] * u.deg
    )

def calc_distance(r1, r2):

    deltax = r2[0].to(u.m) - r1[0].to(u.m)
    deltay = r2[1].to(u.m) - r1[1].to(u.m)
    deltaz = r2[2].to(u.m) - r1[2].to(u.m)

    final_r = np.sqrt(deltax*deltax + deltay*deltay + deltaz*deltaz)

    return final_r.to(u.au)

# Earth = Planet("Earth")
# Mars = Planet("Mars")
# Jupiter = Planet("Jupiter")
# Saturn = Planet("Saturn")
```

```
UPS_epoch = time.Time("2034-05-01 00:00:00", scale="utc").tdb

# Earth = Earth.propagate(UPS_epoch)
# Mars = Mars.propagate(UPS_epoch)
# Jupiter = Jupiter.propagate(UPS_epoch)
# Saturn = Saturn.propagate(UPS_epoch)

# earth = Orbit.from_body_ephem(Earth, epoch=UPS_epoch)
# saturn = Orbit.from_body_ephem(Saturn, epoch=UPS_epoch)

e_ = Ephem.from_body(attractor=Sun, body=Earth, epochs=UPS_epoch)
s_ = Ephem.from_body(attractor=Sun, body=Saturn, epochs=UPS_epoch)

earth = Orbit.from_ephem(ephem=e_, attractor=Sun, epoch=UPS_epoch)
saturn = Orbit.from_ephem(ephem=s_, attractor=Sun, epoch=UPS_epoch)

integration = 2 * u.day
total_calc = 1*365 * u.day
dse = []
epochs = []

iterations = total_calc/integration

print("----------------------------------------\nInitial\n----------------------------------------")
print("Earth", earth.rv())
print("Saturn", saturn.rv())
d_ = calc_distance(earth.rv()[0], saturn.rv()[0])
minimum_d = d_.value

print("DSE", )
print("--------------------------------------------------------------------------------")

propagate_Earth = earth
propagate_Saturn = saturn

for i in range(0, int(iterations)):

    propagate_Earth = propagate_Earth.propagate(10*u.day)
    propagate_Saturn = propagate_Saturn.propagate(10*u.day)

    d_ = calc_distance(propagate_Earth.rv()[0], propagate_Saturn.rv()[0])
    dse.append(d_.value)

    epochs.append(propagate_Earth.epoch.strftime("%Y-%m-%d %H:%M"))

    if(minimum_d>d_.value):
        print("$ DSE -----> " + str(d_), "\n @ Epoch ---> " + propagate_Earth.epoch.strftime("%Y-%m-%d %H:


dse = np.array(dse)
np.save("DSE_", dse)
np.save("epochs", epochs)
```