2020

Team No. 480
Problem A

Ion Thrusters to Saturn

Abstract

In this problem, we are required to design an orbit that consumes the least fuel during the trip to Saturn. To solve the problem, we set up a design model of the trajectory consisting of two types of motion: self-acceleration spiral motion and gravity assist motion. Then we derive the equations of motion based on the law of universal gravitation. Next with Heun's method, we estimate the solutions and simulations of the spacecraft's motion using MATLAB. The value of the speed of the spacecraft, the resumed fuel and the required time are obtained by Matlab as well. Alternatively, we also use Lambert's problem to optimize the calculation in gravity assist motion. Finally we come to the conclusion that the designed orbit successfully places the spacecraft into the required circular orbit of Saturn, with fuel consumption 1329.953 kg and the trip duration 2726 days. During the trip, the ion thruster is in different states at different stages. When it is escaping from the Earth and moving from the Earth to Mars, it accelerates consistently. When it moves from Mars to Jupiter, it accelerates within the circle with radius 0.7 times of Jupiter's centered at Sun, otherwise, it will keep off and rely on the gravitational force. When it moves from Jupiter to Saturn, it totally relies on gravitational force. After being captured by Saturn, it decelerates to go to the required orbit. During all the stages, the thrust force is always parallel to the velocity to exert the greatest effect so that the mechanical energy is utilized most effectively.

Contents

1 Introduction

Saturn, as a mysterious planet in the solar system, has always been attracting the attention of scientists around the world. In this article, we are going to design an orbit that consumes the least fuel during the trip to Saturn.

We will first give a concept diagram of the spacecrafts' whole trajectory, then divide it into four parts and analyse them respectively. We are going to mainly use the low thrust orbit design and gravity-assist orbit design.

2 Background research

2.1 Ion thrusters

Ion thruster, also known as ion engine, is one of the space electric propulsion technologies, characterized by small thrust, high specific impulse. It is widely used in space propulsion, such as spacecraft attitude control, position maintenance, orbit maneuver and interstellar flight.

The principle is that the gaseous working medium is ionized at first, and the ions are spewing out under the action of a strong electric field. Ion thruster has the characteristics of high specific impulse, high efficiency and low thrust. Compared with traditional chemical propulsion, ion thrusters require a smaller working mass and are the most suitable for long-distance voyages among the most practical propulsion technologies.

2.2 Sphere-of-influence of planets

A Sphere of Influence is how large a planets influence reaches into space. Except for the Sun, all planet's sphere-of-influence (SOI) end at a finite point.

In this article, we will use the sphere-of-influence of Earth and Jupiter:

Planet	SOI
Earth	924642 km
Saturn	54454400 km

3 Assumptions and Notations

3.1 Assumptions

- When the spacecraft is in the sphere-of-influence of one planet, the force exerted on the spacecraft by all the other planets is not taken into account.
- During the whole motion, the cosmic dust and collisions are not considered, i.e. the drag and collision are not taken into account.

- When the spacecraft passes through a planet for gravity assist, the time to pass the sphere-of-influence can be neglect.
- When the spacecraft fly past by the planets for gravity assist, the place of the planets are in the exact place when the velocity vector of spacecraft forms the maximum angle with the radius vector of the planets.
- The motion of all the planets and spacecrafts are in the 2D dimension.

3.2 Notations

Notation	Definition
v_∞	the speed when the celestial body is flying out the sphere-of-influence of a planet
v^+	the velocity after the spacecraft obtains gravity assist from the target planet
v^-	the velocity before the spacecraft obtains gravity assist from the target planet
C_3	the square of v_∞
δ	the turn angle during the gravity assist
r_p	the radius of the near center of the hyperbolic orbit
a	the semi-major axis of a hyperbolic orbit
μ_B	the gravitational constant of the planet
E	eccentric anomaly associated with radius r
E_0	eccentric anomaly at r_0,

4 Model

4.1 Problem Overview

In this article, we are going to design the orbit of the spacecraft to launch it from Earth parking orbit to Jupiter with the minimum fuel consumption. To realize this, we need to consider two points:

- The whole running time of ion thrusters should be as short as possible.
- Ion thrusters should run at certain position in orbits to allow the rocket to reach its target orbit precisely.

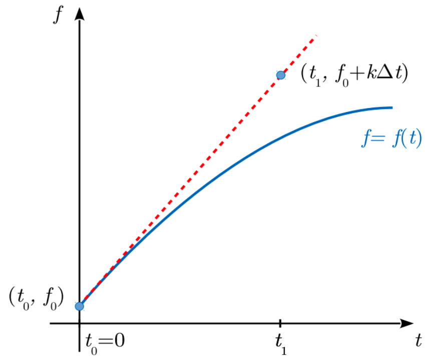
4.2 Theorems and Laws

4.2.1 Law of universal gravitation

The universal gravitation is the mutual attraction between any two objects caused by mass. The properties of the universal gravitation can be described as the force's line of action is the line between the center of mass of the two objects. Its magnitude is proportional to the mass of the two objects, and inversely proportional to the square of the distance between the two objects.It can be described in equation below:

$$\vec{F} = \frac{Gm_1m_2}{r^3} \cdot \vec{r} \tag{1}$$

Newton proved that the gravitational force on a particle outside a sphere whose density is a function of radius R is the same as if the mass of the whole sphere were concentrated at the centre of the sphere. Therefore, the equation above is valid during the calculation.



H]

Figure 1: Illustration of Euler's method.

4.2.2 Euler's method

Euler's method is a numerical method helping find the unique solution of the second-order ordinary differential equation (ODE) in our project:

$$\frac{\mathrm{d}^2 r}{\mathrm{d}t^2} = \frac{F(v, r, t)}{m} \tag{2}$$

with the initial conditions1 for the velocity v(0) = v0 and the position r(0) = r0.

Although the Newton's equation of motion (1) is a second-order ODE, it can be rewritten as a pair of coupled first-order ODEs by recalling the definitions of the velocity and the acceleration:

$$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \frac{\mathbf{F}(\mathbf{v}(t), \mathbf{r}(t), t)}{m}, \quad \frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} = \mathbf{v}(t) \tag{3}$$

Therefore the problem of solving a second-order ODE can be effectively reduced to the problem of solving two first-order ODEs of the form:

$$\frac{\mathrm{d}f}{\mathrm{d}t} = G(f(t), t) \tag{4}$$

Then, let us start with the Taylor expansion of f at the instant of time $t + \Delta t$ and neglect all terms of the order higher than one in Δt. Then:

$$f(t + \Delta t) = f(t) + G(f(t), t)\Delta t$$

Given the initial condition f(0) = f_0, we can find the (approximate) values of the function f in the required time interval, that is to numerically solve the first-order ODE (3) with that initial condition and the presentation is as follows:

4.2.3 Heun's method

Based on Euler's method, We may refine this method by first evaluating the slope at the left end of the interval, which is

$$k_1 = G(f_0, t_0),$$

and then at the right end. For the slope on the right end, we predict the (unknown) value of f at t1, that is the slope there is evaluated as

$$k_2 = G(f_1 + k_1 \Delta t, t_1).$$

Then, the average value of the slope at both ends is used to evaluate a corrected prediction f_1^* of the value of f at $t = t_1$ (see Fig. 2, where the first step of the method is shown). From then on, Similarly to the steps in Euler's method, the found approximate values of the function f can be found.

Implementing the described idea, the set of equations used to numerically solve this differential equation can be written in the following form

$$\begin{aligned} f_0^* &= f_0 \\ k_1 &= G\left(f_i^*, t_i\right) \\ k_2 &= G\left(f_i^* + k_1 \Delta t, t_i + \Delta t\right) \\ f_{i+1}^* &= f_i^* + \frac{1}{2}\left(k_1 + k_2\right) \Delta t \end{aligned}$$

It allows us to find the approximate value of f on the required interval and the presentation of Heun's method is as follows:

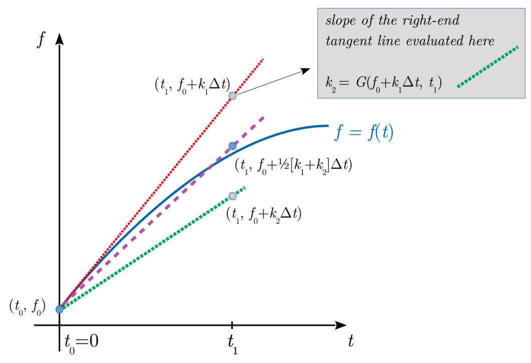

slope of the right-end tangent line evaluated here
$k_2 = G(f_0+k_1\Delta t,\ t_1)$
f
$(t_1,\ f_0+k_1\Delta t)$
$f = f(t)$
$(t_1,\ f_0+½[k_1+k_2]\Delta t)$
$(t_1,\ f_0+k_2\Delta t)$
$(t_0,\ f_0)$
$t_0=0$
t_1
t

Figure 2: Illustration of Heun's method.

4.2.4 Gravity assist

Planetary gravity assist orbit change is a method described as: the spacecraft that launches from a star flies close to another star (planet) use the gravitation of this star to change orbit and fly to another target star. With the aid of planetary gravity, the velocity increment can be obtained by utilizing the gravity of a planet to change orbit with the aid of the gravity of the planet in the middle, so as to save some fuel.

The illustration of the gravity assist are as Figure 3 and Figure 4 from two perspective[1].

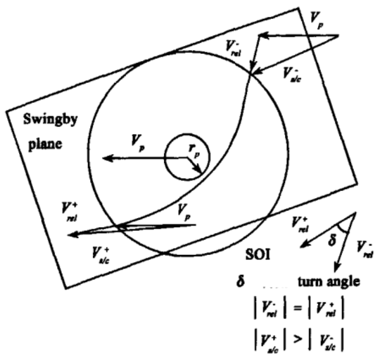


Figure 3: a. Planetary perspective [1].

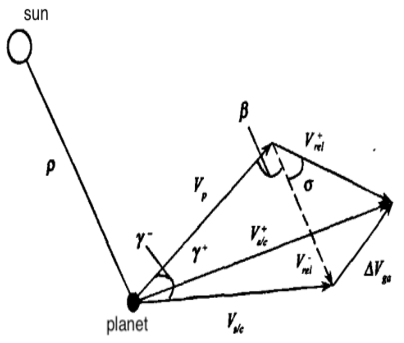


Figure 4: b. Sun perspective [1].

Let the velocity change of the detector relative to the central gravity before and after the gravity assist of the planet is ΔV, i.e.

$$\Delta V = V^+ - V^- = 2\ v_\infty \sin\frac{\delta}{2}$$

From the properties of hyperbolic orbits, the turn angle can be presented as

$$\sin\frac{\delta}{2} = \frac{1}{1 + (r_p/a)}. \tag{5}$$

Then, according to the energy equation for a hyperbolic orbital, we can get:

$$\sin\frac{\delta}{2} = \frac{1}{1 + (r_p V_\infty^2/\mu_B)} \tag{6}$$

4.2.5 Specific impulse

Specific impulse is defined as the impulse generated by the amount of unit propellant:

$$I_{sp} = \frac{dI}{dG} = \frac{F}{\dot{m} g_0},$$

where g_0 is standard gravitational acceleration and its magnitude is 9.80665 m/s. [2]

4.2.6 Conservation of Energy

The energy is conserved during the whole process of motion. The total amount of energy is the gravitational potential energy and the kinetic energy the spacecraft hold first. Hence the equation for conservation of energy could be written as

$$E_0 = E_p + E_k.$$

4.3 Basic model

4.3.1 Application of Heun's Method

In this article, we introduce the **gravity assist model**. The motion of the satellite is analysed with Heun's method.
Firstly, we build a Cartesian coordinate system with the sun as original point. The position of the satellite at moment t, is assumed as (x,y), then according to Newton's formula for universal gravitation, the acceleration of the satellite can be obtained as:

$$\begin{aligned} \ddot{x} = \dot{v}_x = -\frac{GMx}{r^3} \\ \ddot{y} = \dot{v}_y = -\frac{GMy}{r^3} \end{aligned} \tag{7}$$

where $r = \sqrt{x^2 + y^2}$ represents the distance between the satellite and central object. Since we have introduced the definition of Sphere-of influence for a planet, we do not have to take the effect of them when the satellite removes far enough from these planets.

We assume A_x is a function with respect to x, v_x and t, which represents the x component of acceleration. For infinitesimal moving process, the motion of satellite should satisfy the equations below:

$$\begin{aligned} \mathrm{a}_1 &= A(v_{x_i}, x, t) \\ \mathrm{a}_2 &= A(v_{x_i} + a_1\Delta t, x, t + \Delta t) \\ \mathrm{v}_{x_{i+1}} &= v_{x_i} + \frac{1}{2}(a_1 + a_2)\Delta t \\ \mathrm{x}_{i+1} &= x_i + \frac{1}{2}(v_{x_i} + v_{x_{i+1}})\Delta t \end{aligned}$$

By iterating the equation, the position of the satellite at any instant can be calculated.

4.3.2 The Whole model

The overall concept diagram is shown in Figure 5. To make it easier to analyse, we divide it into four process which includes, Spiral ascent stage (escaping from the Earth), receiving gravitational assist from Mars, receiving gravitational assist from Jupiter and captured by Saturn.

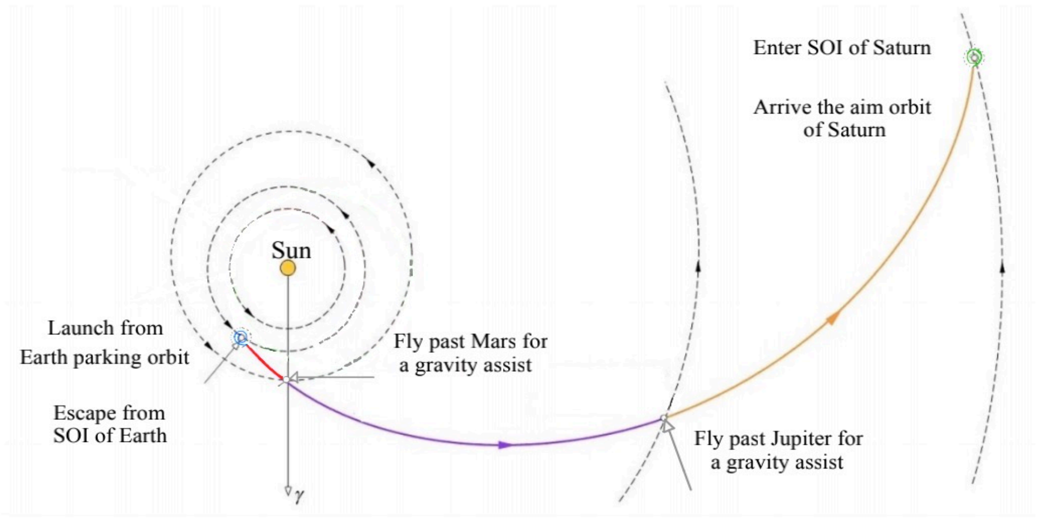

Enter SOI of Saturn
Arrive the aim orbit of Saturn
Sun
Launch from Earth parking orbit
Fly past Mars for a gravity assist
Escape from SOI of Earth
Fly past Jupiter for a gravity assist
γ

Figure 5: Concept diagram of the trajectory for gravity assist model

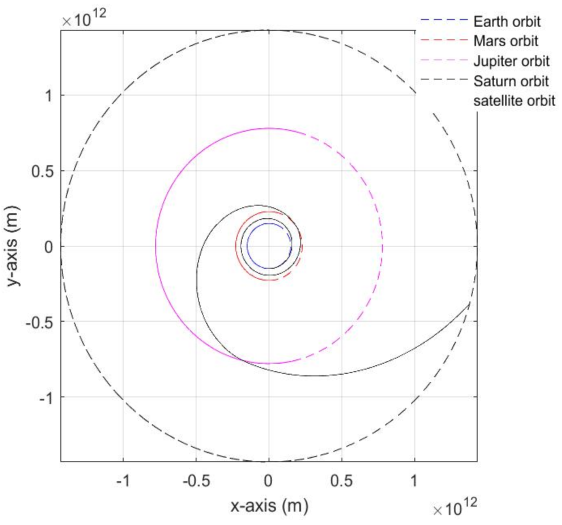


Figure 6: Flight plan for gravity assist model

Using Matlab, we could divide the concept diagram of the trajectory for gravity assist model, as shown above in Figure 6. In order to analyze the motion of the whole trajectory, we divided the trajectory into five parts, as shown below in figures in section 4.3.1 to section 4.3.5.

a. Spiral ascent stage

The ion thrusters are turned on when the spacecraft launches form the Earth parking orbit. As shown in Figure 7, the spacecraft keep doing spiral ascent motion until it reaches the boundary of the sphere-of-influence of Earth [2].

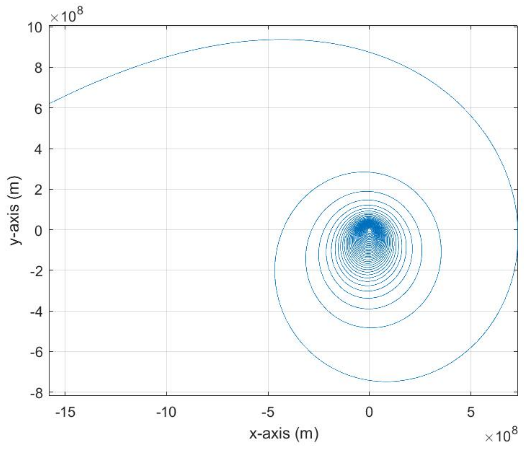


Figure 7: The Spiral ascent stage

b. Gravity assist from Mars

After the spacecraft escape from the sphere-of-influence of Earth, it continues the hyperbola under the action of the heliocentric gravity until it reaches close to Mars to have gravity assistance [3]. During this process, **the ion thrusters are turned on after the spacecraft travels half of the trajectory in this part**.

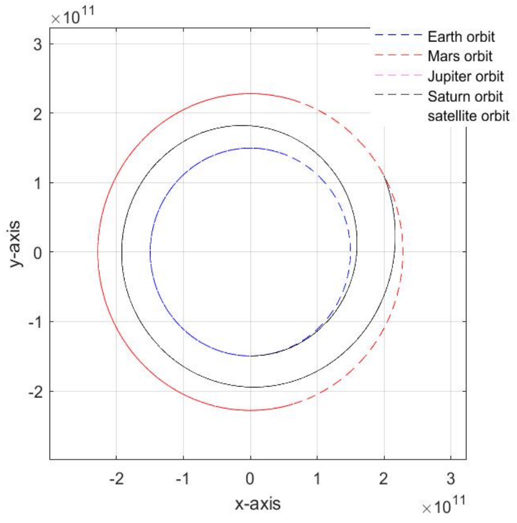


Figure 8: Gravity assist from Mars

c. Gravity assist from Jupiter

After the spacecraft passes by Mars, it continues the hyperbola(but not the same one as in section 4.3.3) under the action of the heliocentric gravity until it reaches close to Jupiter to have another gravity assist. During this process, **the ion thrusters are kept off**.

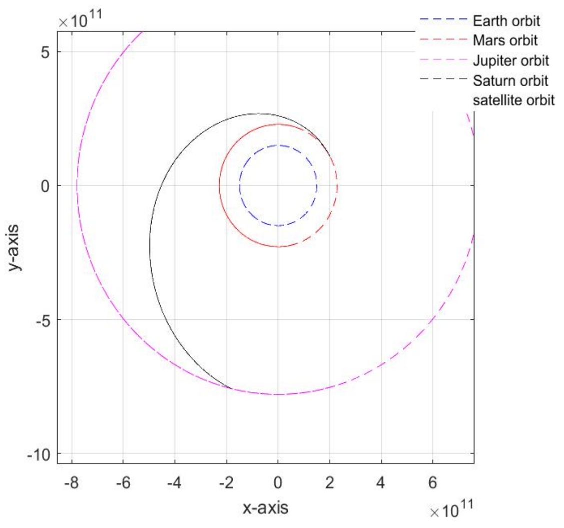


Figure 9: Gravity assist from Jupiter

d. Motion from Jupiter to Saturn

After the spacecraft passes by Jupiter, it continues the hyperbola(but not the same one as in section 4.3.4) under the action of the heliocentric gravity until it enters the sphere-of-influence of Saturn. During this process, **the ion thrusters are kept off**.

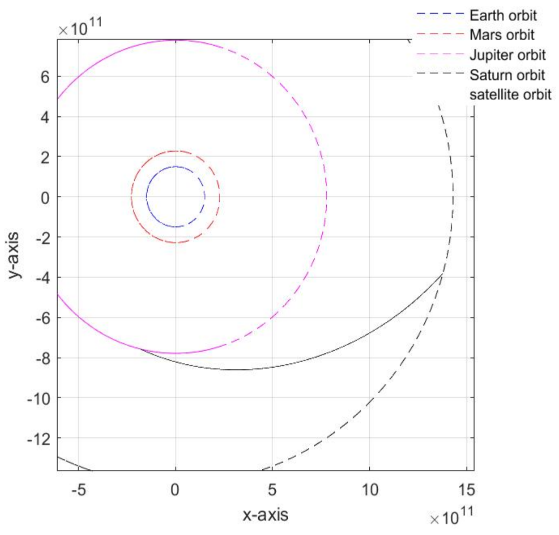


Figure 10: Gravity assist from Jupiter

e. Spiral descend stage

After the spacecraft enters the sphere-of-influence of Saturn, **the ion thruster are turned on**. It will decelerate the spacecraft to do spiral descend motion until it park in the acquired orbit.

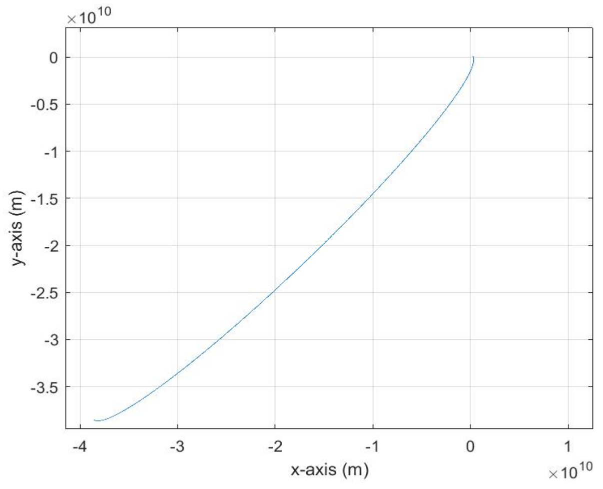


Figure 11: Spiral descend stage

5 Results

5.1 The gravity assist model

According to our calculation, we obtain the detailed values of speed, residual fuel, distance from the sun and time in each part of the model. The model is in Figure 6 and the data is recorded in Table 1. Based on the calculated values, we find the total length of the trajectory L and the duration T.

Event	Date	Residual Fuel (kg)	Distance From the Sun (m)	Speed (m/s)	
Start spiral ascending	2030.11.7	5000.000	1.50×10^{11}		
Escape From the Earth	2031.7.9	4784.768	1.50×10^{11}	30485	
Arrive in Mars	2033.5.1	4201.471	2.28×10^{11}	before	24362
				after	26821
Arrive in Jupiter	2036.8.3	3682.427	7.79×10^{11}	before	10422
				after	15609
Captured by Saturn	2041.1.6	3682.427	1.43×10^{12}	9410	
Enter circular orbit	2041.1.20	3670.047	1.43×10^{12}		

Table 1: Data of each stages with the gravity assist

5.2 Detailed Model Analysis

5.2.1 Spiral ascent stage

1. Initial condition

Regarding to the problem, the spacecraft is in a circular orbit around the Earth with period of 90 minutes. According to the orbital period formula:

$$T = 2\pi\sqrt{\frac{r^3}{GM}}$$

$$90 \times 60 = 2\pi\sqrt{\frac{r^3}{6.67 \times 10^{-11} \times 5.965 \times 10^{24}}}$$

r = 6.648×10^6 m.
The relationship between the period and the orbital radius is shown in Fig.16. It shows a trend of increasing and oscillating. As the radius increases, the period increases. As period approaches 250 days, the radius increases greatly as the spacecraft is ready to escape.
According to the orbital velocity formula:

$$v = \sqrt{\frac{GM}{r}} = \sqrt{\frac{6.67 \times 10^{-11} \times 5.965 \times 10^{24}}{6.648 \times 10^6}} = 7735.840m/s$$

The speed of the spacecraft is v=7735.840 m/s. The relationship between the period and the speed is shown in Fig.13. It shows a trend of decreasing and oscillating. As the period increases, the speed decreases. As period approaches 250 days, the speed approaches to a constant value.

2. Ascending stage
Traditional rocket typically utilizes fuel propulsion including hydrogen and nitrogen tetroxide. These fuels are able to produce huge driving force and provide the rocket a huge momentum with a short time. However, for ion thruster, the force it provides is rather small and the shape of the trajectory can not change intensely.

The ascending orbit, to some extent, is similar to a spiral line according to Figure 7, if we put the trajectory of this process into a polar coordinate system, the radius tends to with the increment of angle. Since the force that is perpendicular to the direction of velocity has no contribution to the increment of kinetic energy. To utilize the ion fuel effectively, this direction of driving force in this model is always set to be tangent the velocity.

Figure 12 and Figure 13 are generalized with MATLAB based on the statistics data and the corresponding code is attached in Appendix. Figure 12 shows the relationship between the earth-satellite distance and time. it can be noticed that the distance is not strictly monotone increasing with respect to time. Besides, from Figure 13, we can find even the ion thruster is keeping providing driving force, the magnitude of velocity still has a downward trend. This is because, when the satellite leaves away from the Earth, work is being done against the attractive forces between them which will consume part of kinetic energy.

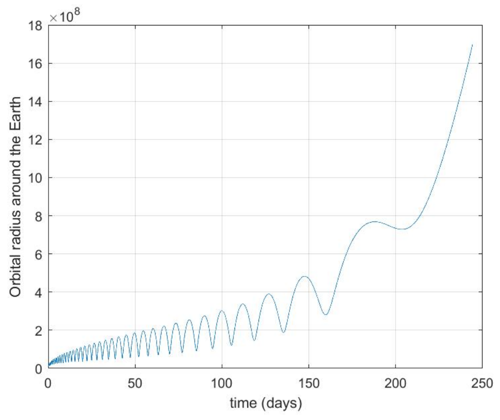


Figure 12: Relationship between orbital radius and period

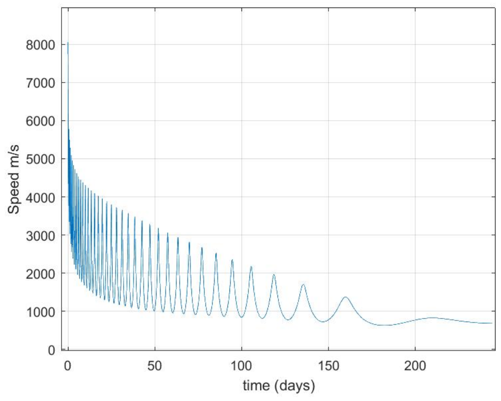


Figure 13: Relationship between speed and period

3. Final condition

The critical condition judging whether satellite would escape from the earth is to analyse the total mechanical energy of this object. As we have known, for a celestial body which is moving freely around the central object , if the mechanical energy of this body is smaller than zero, the trajectory would be an ellipse while the shape of trajectory for an object with negative mechanical energy would be a hyperbola. Therefore, zero mechanical energy would be a turning point for the shape of orbit and the shape in this situation is actually a parabola, which means the distance between the central object and the body could be infinite.

In this system, as the satellite keeps accelerating, the mechanical energy for unit mass of the satellite, which can be represented as $e = \frac{1}{2}v^2 - \frac{GM}{r}$, will increase. When e increases to a non-negative number, the satellite will absolutely leave the restriction of the earth and enter the deep space. This process would require 244 days and cost 215.232 kg ion fuel. The critical position is $1.69 \times 10^9 m$ away from the earth which is much smaller than the earth-sun distance (1.50×10^{11} m). Therefore, we can approximately regard the distance between the sun and satellite is unchanged during this stage.

5.2.2 Gravity assist from Mars

According to the document[3],in forward flyby, the relative velocity of the spacecraft to the celestial body at the point of departure should be the same as which at the point of entry. $\vec{V_i}$,$\vec{V_f}$ is the initial, final velocity of the spacecraft relative to the Sun. $\vec{V_{iB}^{\infty}}$, $\vec{V_{fB}^{\infty}}$ is the initial and final relative velocity of the spacecraft to the celestial body. $\vec{V_B}$ is the relative velocity of the celestial body to the Sun. This property can be described in the graphs and equations below.

$$\vec{V_i} = \vec{V_{iB}^{\infty}} + \vec{V_B}$$
$$\vec{V_f} = \vec{V_{fB}^{\infty}} + \vec{V_B}$$

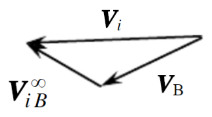


Figure 14: Before gravity assist

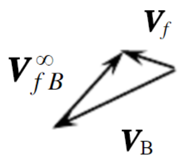


Figure 15: After gravity assist

1. Initial condition
The position of the Mars is:

$$x = 1.996 \times 10^{11}, y = 1.102 \times 10^{11}$$

The velocity of the Mars is: $V_B = 24000m/s$
Regarding Mars, the velocity vector is perpendicular to the position vector. By calculating sin and cos, we can get the x and y component of the velocity of Mars as:

$$v_{xB} = -11600.296m/s, v_{yB} = 21010.310m/s$$

The angle between the planet's velocity and the positive x-axis $\phi_1 = 118.904^o$. For the initial velocity of the spacecraft, we have data:

$$vx = -9203.395m/s, vy = 22557.199m/s, v = 24362.465m/s$$

Then we calculate the components of the relative velocity through equation $\vec{V_i} = \vec{V_{iB}^{\infty}} + \vec{V_B}$:

$$vxr = 2396.901m/s, vyr = 1546.889m/s$$

The angle between the initial relative velocity and the positive x-axis $\phi_2 = 32.837^o$

2. Traveling stage
In order to let the final velocity (Sun) be the biggest, the final relative velocity need to be in the same direction with the planet's velocity. According to the initial condition, we have the angle $\phi_1 = 118.904$, the angle $\phi_2 = 32.837$. The rotation angle is

$$\theta = \phi_1 - \phi_2 = 118.904 - 32.837 = 86.067\ ^{\circ}.$$

3. Final condition
Through the equation:

$$vx = |vr| \times cos(\phi_2) + vpx$$

$$vy = |vr| \times sin(\phi_2) + vpy$$

We get the final velocity (Sun) of the spacecraft as:

$$vx = -12568.352m/s, vy = 23693.753m/s, v = 26820.839m/s$$

5.2.3 Gravity assist from Jupiter

The principle is the same as above.

1. Initial condition
The position of the Jupiter is:

$$x = 1.379 \times 10^{12}, y = 3.803 \times 10^{11}$$

The velocity of the Jupiter is: $V_B = 13070m/s$
Using the same method as mentioned above, we can get the x and y component of the velocity of Jupiter as:

$$v_{xB} = 12712.063m/s, v_{yB} = -3037.821m/s$$

The angle between the planet's velocity and the positive x-axis $\phi_1 = -13.440^o$. For the initial velocity of the spacecraft, we have data:

$$vx = 9367.124m/s, vy = -4569.770m/s, v = 10422.370m/s$$

Then we calculate the components of the relative velocity:

$$vxr = -3344.939m/s, vyr = -1531.949m/s$$

The angle between the initial relative velocity and the positive x-axis $\phi_2 = -155.393^o$

2. Traveling stage
In order to let the final velocity (Sun) be the biggest, the final relative velocity need to be in the same direction with the planet's velocity. According to the initial condition, we have the angle $\phi_1 = -13.440$, the angle $\phi_2 = -155.393$. The rotation angle is

$$\theta = \phi_1 - \phi_2 = -13.440 + 155.393 = 141.953\ ^\circ.$$

3. Final condition
Through the equation:

$$vx = |vr| \times cos(\phi_2) + vpx$$

$$vy = |vr| \times sin(\phi_2) + vpy$$

We get the final velocity (Sun) of the spacecraft as:

$$vx = 14244.012m/s, vy = -6382.760m/s, v = 15608.700m/s$$

5.2.4 Spiral descend stage

1. Initial condition

At the moment that satellite enters the sphere-influence of Saturn, the absolute velocity (with respect to the sun) can be obtained from the former process, which can be expressed as $\vec{v} = 6.38 \times 10^3\hat{\imath} + 6.9 \times 10^3 m/s$ in the Cartesian coordinate system. Based on the average revolution speed and the position vector of the Saturn, the absolute velocity can be obtained as $\vec{v_s} = 2573\hat{\imath} + 9342\hat{\jmath}$. Therefore, the relative velocity of the satellite with repect to the Saturn is $\vec{v_r} = \vec{v} - \vec{v_s} = 3818\hat{\imath} - 2442\hat{\jmath}$.

For the terminate state, the movement is circular motion, the magnitude of velocity and radius of orbit can be obtained based on Newton's Law of universal gravitation. Finally, we can find $r_{final} = 2.71 \times 10^8 m$, and $v_{final} = 11824 m/s$. Notice that the exact position of Saturn when satellite enters the SOI is unknown and it is necessary for us to assume the position. In this model, we choose the satellite enters the SOI at $(-R_{sat}cos(45^o)$,$-R_{sat}sin(45^o))$. During this process, Ion thruster is used to provide drag force so that the satellite will be trapped by the gravity of Saturn. By applying the Heun's method, the simulated orbit is as Figure shows.

2. Final condition

When the distance between satellite and Saturn reaches the required value, the corresponding speed deduced with MATLAB is about 1.1000m/s and therefore the orbit is approximately circular.

5.3 Control Group

In the table below, we calculate the data under the condition of no gravity assist with the same initial condition. Let this group be the control group and compare it with the Table.1, Besides, the satellite's trajectory is shown in Figure 16.

By comparing the data in Table 1 and Table 2, we can find the scheme with gravity assist can save more than 1000 kg ion fuel while non-gravity assist scheme can save more time. In conclusion, the gravity assist can greatly reduce the consumption of the fuel, which verify the superiority of our plan over the plan of going to Saturn directly.

Event	Date	Residual Fuel(kg)	Distance From the Sun(m)	Speed(m/s)
Start spiral ascending	2030/11/7	5000.000	1.50×10^{11}	
Escape from the Earth	2031/7/9	4784.768	1.50×10^{11}	30485
Arrive in Mars	2033.5.1	4201.471	2.28×10^{11}	24362
Arrive in Jupiter	2037.2.17	2977.916	7.79×10^{11}	17246
Captured by Saturn	2038.9.18	2469.004	1.43×10^{12}	20328
Enter circular orbit	2038.10.2	2456.624	1.43×10^{12}	

Table 2: Data of each stage with no gravity assist[1]

1

[1]Since the Central objects for the first and last state are the earth and Saturn respectively, there is no meaning to study the speed with respect to the sun.

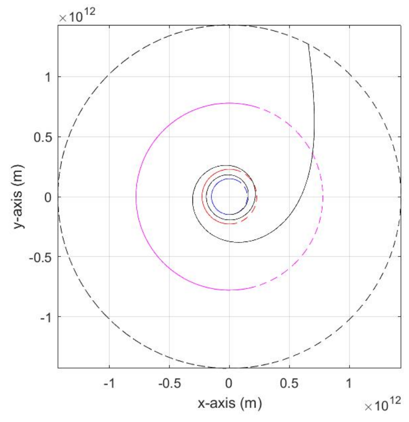


Figure 16: Flight orbit for control group

6 Discussion

6.1 Conclusion

During the whole process, we draw a concept model for the ideal spacecraft and then divide the trajectory into four parts. For the self-accelerate spiral motion part, we simplify this complex model by using the assumption that When the spacecraft is in the sphere-of-influence of one planet, the force exerted on the spacecraft by all the other planets is not taken into account. For the gravity assist motion part, we simplify this complex model by using the assumption that When the spacecraft fly past by the planets for gravity assist, the place of the planets are in the exact place when the velocity vector of spacecraft forms the maximum angle with the radius vector of the planets.

Using Heun's Method, we could solve the complex second order ODEs of motion, then Matlab is applied to find the approximate solution of the equation. Calculate the four models respectively, we finally obtain the total duration of the flight equals 2726 [d], consuming the fuel 1329.953 kg. Before we calculate, we first find the initial condition for the spacecraft on Earth parking orbit and the final condition for the spacecraft on the required circular orbit of Saturn.

Further more, we find that the ability for vary planets to provide gravity assist is different. In order to decrease the energy consumption, we can revise the model to pass by the planets with higher ability to provide gravity assist several times to gain more energy from the planet. Then, the consumption fuel of the whole process can be further reduced.

6.2 Limitations and Possible Improvement

6.2.1 Celestial revolution consideration

In this article, we consider that when the spacecraft pass by the planets, the place of the planets are in the exact place when the velocity vector of spacecraft forms the maximum angle with the radius vector of the planets. However, actually it is not the case. In fact, it is always the case that the spacecraft should wait for a particular moment at which if we launch the spacecraft, it can match the planet perfectly, i.e. When the spacecraft passes through the orbit of revolution of the planet, it is exactly passing by the planets from a close position (usually within 500 km from the surface of the planets)[4].

To make our model much better, we could consider the precise position or trajectory of a celestial body as it moves with time. To realize this, we can use the ephemeris files. Here we can use **JPL Planetary and Lunar Ephemeris DE421** [5].

The DE421 ephemeris file can be added to the Matlab files used before. After revising the existing files, we can build a new concept diagram and it could also be used to test our model.

6.2.2 Optimization of the model in gravity assist process

In the above model, we use Heun's method to analysis the gravity assist process. In fact, we can use *Lambert's problem* to find the final condition for section 5.2.2 and section 5.2.3 [6].

Lambert's problem is also known as classic astrodynamic problem is also known as the orbital twopoint boundary value problem. The time to traverse a trajectory depends only upon the length of the semi-major axis a of the transfer trajectory, the sum of the distances of the initial and final positions relative to a central body, and the length c of the chord joining these two positions. This relationship can be stated as follows:

$$tof = \mathrm{tof}\left(r_i + r_f, c, a\right)$$

From the following form of Kepler's equation, we can write:

$$t - t_0 = \sqrt{\frac{a^3}{\mu}(E - e\sin E)}$$

$$t = \sqrt{\frac{a^3}{\mu}}\left[E - E_0 - e\left(\sin E - \sin E_0\right)\right]$$

If we let $E = \alpha$ and $E_0 = \beta$ and with the eliptics relationships given by

$$r = a(1 - e\cos E)$$

$$x = a(\cos E - e)$$

$$y = a\sin E\sqrt{1 - e^2}$$

After some manipulations, we have

$$\cos\alpha = \left(1 - \frac{r + r_0}{2a}\right) - \frac{c}{2a} = 1 - \frac{r + r_0 + c}{2a} = 1 - \frac{s}{a}$$

$$\sin\beta = \left(1 - \frac{r + r_0}{2a}\right) + \frac{c}{2a} = 1 - \frac{r + r_0 - c}{2a} = 1 - \frac{s - c}{a}$$

With the use of half angle formulas given by

$$\sin\frac{\alpha}{2} = \sqrt{\frac{s}{2a}}, \qquad \sin\frac{\beta}{2} = \sqrt{\frac{s - c}{2a}}$$

the time-of-flight can be calculated as

$$t = \sqrt{\frac{a^3}{\mu}}\left[(\alpha - \beta) - (\sin\alpha - \sin\beta)\right]$$

The heliocentric required at launch and arrival are simply the differences between the velocity on the transfer trajectory determined by the solution of Lambert's problem and the heliocentric velocities of the two planets. So after some manipulations, v_∞ at launch or arrival can be obtained, as well as the energy $C3$ at launch or arrival. The optimized calculation figure is as follows:

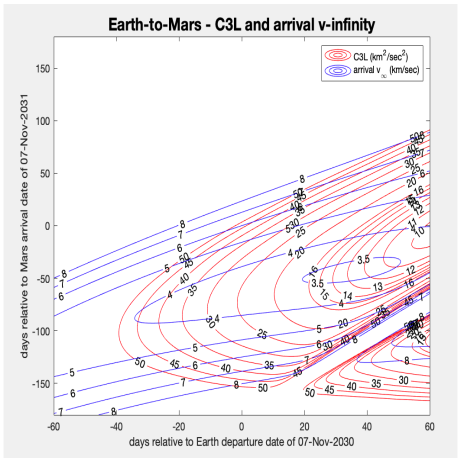


Figure 17: a. C3 and v_∞.

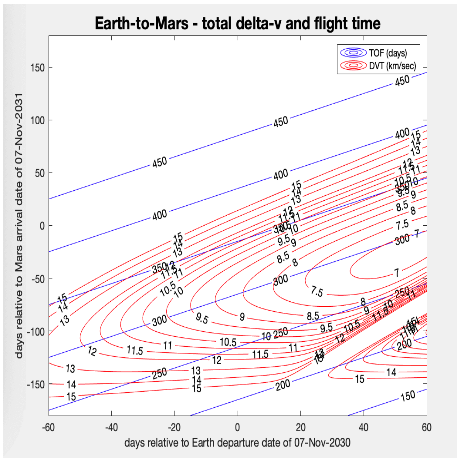


Figure 18: b. Δv and flight time.

6.3 Advantages

Firstly, the simulated orbit and relative deduction are based on the Heun's method. This method is much preciser than Euler's method.

Second, our model utilizes the gravity assist of the planets to the maximum extent. When considering the velocity change after passing through the planet, we choose the most effective deflection angle with which the satellite will obtain the maximum kinetic energy from planet's gravitational force instead of burning fuel.

Moreover, our model is based on many reasonable assumptions. For instance, the effect from other planets will not be taken into consideration if the satellite is in the SOI area. This assumption can not only simplify the complexity and calculation but also propel the process of analysis.

Above all, we tried our best to construct an analysable model of the trajectory of the spacecraft and analyse its motion in the whole process. Comparing with the control group in section 5.3, our group have overwhelming advantages. While the consumption fuel is greatly reduced compared with the control group, the duration is not much longer than it. To sum up, our model not only perform well in reducing the fuel consumption, but also can be regarded as excellent in the overall performance.

References

[1] Zhang xuhui, and Liu zhusheng. *Unpowered Swing—by Flight Orbit Design for Mars Exploration.* Journal of aerospace. 1729-1746 (2008).

[2] Yin dawei, Wen yuanlan wen, Liu Feng, and Liao ying. *Flight Orbit Design of Rover to the Mars.* China Academic Journal Electronic Publishing House. 40-43, (2009).

[3] Broucke, R. *The celestial mechanics of gravity assist.* Astrodynamics Conference. 1988.

[4] Petropoulos, Anastassios E., and James M. Longuski. *Shape-based algorithm for the automated design of low-thrust, gravity assist trajectories.* Journal of Spacecraft and Rockets 41.5 (2004): 787-796.

[5] William M. Folkner, James G. Williams, and Dale H. Boggs. *The Planetary and Lunar Ephemeris DE 421.* IPN Progress Report, 42-178 (2009).

[6] Izzo, Dario. *Revisiting Lambert's problem.* Celestial Mechanics and Dynamical Astronomy 121.1 (2015): 1-15.

A MATLAB Codes

```

    %acceleration
function [ax,ay]=acceleration(vx,vy,x,y,m0)
   G=6.67*10^(-11);            %gravitational constant
   M=5.965*10^(24);            %mass of earth
   F=0.4;                      %constant force
   A=1.02*10^(-5);             %rate of mass loss
   r=sqrt(x^2+y^2);
   v=sqrt(vx^2+vy^2);
   ax=-G*M/(r^3)*x+F*vx/(v*m0);
   ay=-G*M/(r^3)*y+F*vy/(v*m0);
end
```

```
    function [ax,ay]=acceleration2(vx,vy,x,y,m0,bool)
   G=6.67*10^(-11);            %gravitational constant
   M=1.9891*10^30;           %mass of earth
   F=0.4;
   r=sqrt(x^2+y^2);
   v=sqrt(vx^2+vy^2);
   if(bool==0)     %F as driving force
   ax=-G*M/(r^3)*x+F*vx/(v*m0);
   ay=-G*M/(r^3)*y+F*vy/(v*m0);
   elseif(bool==1)        %No F
        ax=-G*M/(r^3)*x;
        ay=-G*M/(r^3)*y;
   elseif(bool==-1)          %F as drag force
   ax=-G*M/(r^3)*x-F*vx/(v*m0);
   ay=-G*M/(r^3)*y-F*vy/(v*m0);
   end
end
```

```
    function e=energy(vx,vy,x,y)
     G=6.67*10^(-11);
     M=5.965*10^(24);
     r=sqrt(x^2+y^2);
     e=1/2*(vx^2+vy^2)-G*M/r;
end
```

```
    %Mechanical energy w.r.t the sun
function e=energy2(vx,vy,x,y)
     G=6.67*10^(-11);
     M=1.9891*10^30;
     r=sqrt(x^2+y^2);
     e=1/2*(vx^2+vy^2)-G*M/r;
end
```

```
%The process that satellite got captured by the Saturn and enters the
%circular orbit.
clear;
G=6.67*10^(-11); %gravitational
vx=3818;
vy=-2442;
M=5.68*10^26;
x(1)=-5.45*10^10/sqrt(2); %initial radius
y(1)=-5.45*10^10/sqrt(2);
v(1)=vy;
r(1)=x(1);
F=0.4;
A=1.02*10^(-5); %rate of mass loss
t(1)=0;
dt=100;             %time interval
index=1;
m0=3682.427;
while(1)
     disp(r(index));
     vx0=vx;
     vy0=vy;
    [ax_tem1,ay_tem1]=acceleration2(vx,vy,x(index),y(index),m0,-1);
    vx_tem= vx+ax_tem1*dt;
    vy_tem= vy+ay_tem1*dt;
    [ax_tem2,ay_tem2]=acceleration2(vx_tem,vy_tem,x(index),y(index),m0-A*dt,-1);
    vx=vx+1/2*(ax_tem1+ax_tem2)*dt;
    vy=vy+1/2*(ay_tem1+ay_tem2)*dt;
    index=index+1;
    x(index)=x(index-1)+1/2*(vx+vx0)*dt;
    y(index)=y(index-1)+1/2*(vy+vy0)*dt;
    t(index)=t(index-1)+dt;
    v(index)=sqrt(vx^2+vy^2);
    r(index)=sqrt(x(index)^2+y(index)^2);
    m0=m0-A*dt;
     if(r(index)<=2.71*10^8)
          break;
     end
end
t_day=t/(60*60*24);
figure(1);
plot(x,y);
xlabel("x-axis (m)");
ylabel("y-axis (m)");
axis equal;
grid on;

fprintf("capture:t= %f days\n",t_day(index));
fprintf("remaining mass when  entering orbit %f kg\n",m0);
fprintf("distance from the Saturn %f m\n",r(index));
fprintf("relative speed %f m/s \n\n\n",sqrt(vx^2+vy^2));
```

```
   %The model without gravity assist
clear;
G=6.67*10^(-11); %gravitational
M=1.989*10^30;     %mass of sun
[vx,vy,m0,time,radius,x0,y0]=escape();
v_relative=sqrt(vx^2+vy^2);
A=1.02*10^(-5);
v=29800;
vx=v+v_relative;
vy=0;
index=1;
x(index)=0;
y(index)=-149597870700;
r(index)=y(index);
v(index)=vx;
t(index)=time;
dt=100;
t_a=0;
check=1;
while(1)
   % disp(r(index));
        r_sat=1.43*10^12;
         r_Mars=2.28*10^11;
        r_jup=7.79*10^11;
  bool=0;
   [ax_tem1,ay_tem1]=acceleration2(vx,vy,x(index),y(index),m0,bool);
   vx_tem=vx+(ax_tem1)*dt;
   vy_tem=vy+(ay_tem1)*dt;
   index=index+1;
   x(index)=x(index-1)+1/2*(vx+vx_tem)*dt;
   y(index)=y(index-1)+1/2*(vy+vy_tem)*dt;
   t(index)=t(index-1)+dt;
   r(index)= sqrt((x(index))^2+(y(index))^2);
   vx=vx_tem;
   vy=vy_tem;
   m0=m0-A*dt;
   if (check==1 && r(index)>= r_Mars )
       index0=index;
       m1=m0;
       v1x=vx;
       v1y=vy;
       check=check+1;
   end
    if (check==2 && r(index)>=  r_jup)
        index1=index;
        m2=m0;
        v2x=vx;
       v2y=vy;
       check=check+1;
   end
```

```

    if(r(index)>=r_sat)
        break;
    end
end

t_day=t/(60*60*24);

fprintf("Arrive Mars:t=%f days\n",t_day(index0));
fprintf("Speed when arrive the Mars %f m/s\n",sqrt(v1x^2+v1y^2));
fprintf("remaining mass %f kg\n\n\n",m1);

fprintf("Arrive Jupiter t=%f days\n",t_day(index1));
fprintf("Speed when arrive the Jupiter %f m/s\n",sqrt(v2x^2+v2y^2));
fprintf("remaining mass %f kg\n\n\n",m2);

fprintf("Arrive Saturn:t=%f days\n",t_day(index));
fprintf("Speed when arrive the Saturn %f m/s\n",sqrt(vx^2+vy^2));
fprintf("remaining mass %f kg\n\n\n",m0);
disp(vx);
disp(vy);


figure(3)
plot(t_day,r);
xlabel("Time (day)");
ylabel("Distance from the Sun");

figure(4)
draw();
hold on;
plot(x,y,'k');
axis equal;
grid on;
xlabel("x-axis (m)");
ylabel("y-axis (m)");
axis equal;
```

```
    %Draw the trajectory of planets
function draw()
    r_earth=149597870700;
    r_Mars=2.28*10^11;
    r_jup=7.79*10^11;
    r_sat=1.43*10^12;
    xt = @(t) r_earth*cos(t);
    yt = @(t) r_earth*sin(t);
    fplot(xt,yt,'--b');

    hold on;
    xt = @(t) r_Mars*cos(t);
    yt = @(t) r_Mars*sin(t);
    fplot(xt,yt,'--r');

    hold on;
    xt = @(t) r_jup*cos(t);
    yt = @(t) r_jup*sin(t);
    fplot(xt,yt,'--m');

    hold on;
    xt = @(t) r_sat*cos(t);
    yt = @(t) r_sat*sin(t);
    fplot(xt,yt,'--k');

    hold on;
end
```

```
    %process for escaping the earth
function [v_x,v_y,m_remain,time,radius,X,Y]=escape()
clear;
G=6.67*10^(-11); %gravitational
vy=7735;
vx=0;
M=5.965*10^(24);
m0=5000;          %initial mass
x(1)=6.65*10^6; %initial radius
y(1)=0;
v(1)=vy;
r(1)=x(1);
F=0.4;
A=1.02*10^(-5); %rate of mass loss
t(1)=0;
dt=100;              %time interval
index=1;
while(1)
     vx0=vx;
     vy0=vy;
    [ax_tem1,ay_tem1]=acceleration(vx,vy,x(index),y(index),m0);
    vx_tem= vx+ax_tem1*dt;
    vy_tem= vy+ay_tem1*dt;
    [ax_tem2,ay_tem2]=acceleration(vx_tem,vy_tem,x(index),y(index),m0-A*dt);
    vx=vx+1/2*(ax_tem1+ax_tem2)*dt;
    vy=vy+1/2*(ay_tem1+ay_tem2)*dt;
    index=index+1;
    x(index)=x(index-1)+1/2*(vx+vx0)*dt;
    y(index)=y(index-1)+1/2*(vy+vy0)*dt;
    t(index)=t(index-1)+dt;
    v(index)=sqrt(vx^2+vy^2);
    r(index)=sqrt(x(index)^2+y(index)^2);
    m0=m0-A*dt;
    e=energy(vx,vy,x(index),y(index));

    if (e>=0)
```

```
        break;
    end

end


t_day=t/(60*60*24);

figure(1);
plot(x,y);
xlabel("x-axis (m)");
ylabel("y-axis (m)");
axis equal;
grid on;

figure(2)
plot(t_day,r);
grid on;
xlabel("time (days)");
ylabel("Orbital radius around the Earth");

figure(3)
plot(t_day,v);
grid on;
xlabel("time (days)");
ylabel("Speed m/s");

fprintf("escape the earth:t= %f days\n",t_day(index));
fprintf("remaining mass when escaping the earth %f kg\n",m0);
fprintf("distance from the earth %f m\n",r(index));
fprintf("relative speed when escape %f m/s \n\n\n",sqrt(vx^2+vy^2));
v_x=vx;
v_y=vy;
m_remain=m0;
radius=r(index);
X=x(index);
Y=y(index);
time=t(index);
end
```

```
    %earth to Mars to Jupiter to Saturn
clear;
G=6.67*10^(-11); %gravitational
M=1.989*10^30;    %mass of sun
[vx,vy,m0,time,radius,x0,y0]=escape();
v_relative=sqrt(vx^2+vy^2);
A=1.02*10^(-5);
v=29800;
vx=v+v_relative;
vy=0;
index=1;
```

```
x(index)=0;
y(index)=-149597870700;
r(index)=y(index);
v(index)=vx;
t(index)=time;
dt=100;
t_a=0;
%fly to Mars
while(1)
      r_Mars=2.28*10^11;
    if (r(index)>0.5*r_Mars )
        bool=0;
    else
        bool=1;
    end
    [ax_tem1,ay_tem1]=acceleration2(vx,vy,x(index),y(index),m0,bool);
    vx_tem=vx+(ax_tem1)*dt;
    vy_tem=vy+(ay_tem1)*dt;
    index=index+1;
    x(index)=x(index-1)+1/2*(vx+vx_tem)*dt;
    y(index)=y(index-1)+1/2*(vy+vy_tem)*dt;
    v(index)=sqrt(vx^2+vy^2);
    t(index)=t(index-1)+dt;
    r(index)= sqrt((x(index))^2+(y(index))^2);
    vx=vx_tem;
    vy=vy_tem;
    m0=m0-A*dt;

    if(~bool)
        t_a=t_a+dt;
    end
    if(r(index)>=r_Mars)
         index0=index;
         fprintf("x=%f, y=%f\n",x(index),y(index));
        break;

    end
end
t_day=t/(60*60*24);
fprintf("Arrive Mars:t=%f days\n",t_day(index));
fprintf("Speed when arrive the Mars %f m/s\n",sqrt(vx^2+vy^2));
fprintf("remaining mass %f kg\n\n\n",m0);

%gravity assist from Mars
v_rotate=24000;
sinv=y(index)/(sqrt(x(index)^2+y(index)^2));
cosv=x(index)/(sqrt(x(index)^2+y(index)^2));
vpx=-v_rotate*sinv;
vpy=v_rotate*cosv;
vxr=vx-vpx;
```

```
vyr=vy-vpy;
vr=sqrt(vxr^2+vyr^2);
fprintf("vx%f vy%f  v%f\n",vx,vy,sqrt(vx^2+vy^2));
fprintf("vpx%f vpy%f  vp%f\n",vpx,vpy,sqrt(vpx^2+vpy^2));
fprintf("vxr%f vyr%f\n",vxr,vyr);
anglep=cart2pol(vpx,vpy);
fprintf("anglep %f\n",anglep*180/pi);
angle=cart2pol(vxr,vyr);
fprintf("angle %f\n",angle*180/pi);
angle=cart2pol(vxr,vyr)+77*pi/180;
vx=abs(vr)*cos(angle)+vpx;
vy=abs(vr)*sin(angle)+vpy;
fprintf("vx%f vy%f v%f \n",vx,vy,sqrt(vx^2+vy^2));


%fly to Jupiter
 r_jup=7.79*10^11;
while(1)
    if(r(index)<0.7*r_jup)
   [ax_tem1,ay_tem1]=acceleration2(vx,vy,x(index),y(index),m0,0);
   m0=m0-A*dt;
    else
    [ax_tem1,ay_tem1]=acceleration2(vx,vy,x(index),y(index),m0,1);
    end

   vx_tem=vx+(ax_tem1)*dt;
   vy_tem=vy+(ay_tem1)*dt;
   index=index+1;
   x(index)=x(index-1)+1/2*(vx+vx_tem)*dt;
   y(index)=y(index-1)+1/2*(vy+vy_tem)*dt;
   t(index)=t(index-1)+dt;
    v(index)=sqrt(vx^2+vy^2);
   r(index)= sqrt((x(index))^2+(y(index))^2);
   vx=vx_tem;
   vy=vy_tem;

   if(r(index)>r_jup)
       index1=index;
       break;
   end

end
t_day=t/(60*60*24);
fprintf("Arrive Jupiter:t=%f days\n",t_day(index));
fprintf("Speed when arrive the Jupiter %f m/s\n",sqrt(vx^2+vy^2));
fprintf("remaining mass %f kg\n\n\n",m0);
fprintf("r %f m\n",r(index));


%gravity assist from Jupiter
```

```
v_rotate=13070;
sinv=y(index)/(sqrt(x(index)^2+y(index)^2));
cosv=x(index)/(sqrt(x(index)^2+y(index)^2));
vpx=-v_rotate*sinv;
vpy=v_rotate*cosv;
vxr=vx-vpx;
vyr=vy-vpy;
vr=sqrt(vxr^2+vyr^2);
fprintf("vx%f  vy%f   v%f\n",vx,vy,sqrt(vx^2+vy^2));
fprintf("vpx%f  vpy%f   vp%f\n",vpx,vpy,sqrt(vpx^2+vpy^2));
fprintf("vxr%f  vyr%f\n",vxr,vyr);
anglep=cart2pol(vpx,vpy);
fprintf("anglep %f\n",anglep*180/pi);
angle=cart2pol(vxr,vyr);
fprintf("angle %f\n",angle*180/pi);
 r_sat=1.43*10^12;
i=90;
angle=cart2pol(vxr,vyr)+i*pi/180;
vx=abs(vr)*cos(angle)+vpx;
vy=abs(vr)*sin(angle)+vpy;
fprintf("vx%f  vy%f  v%f  \n",vx,vy,sqrt(vx^2+vy^2));

%fly to Saturn
while(1)
   [ax_tem1,ay_tem1]=acceleration2(vx,vy,x(index),y(index),m0,1);
   vx_tem=vx+(ax_tem1)*dt;
   vy_tem=vy+(ay_tem1)*dt;
   index=index+1;
    v(index)=sqrt(vx^2+vy^2);
   x(index)=x(index-1)+1/2*(vx+vx_tem)*dt;
   y(index)=y(index-1)+1/2*(vy+vy_tem)*dt;
   t(index)=t(index-1)+dt;
   r(index)= sqrt((x(index))^2+(y(index))^2);
   vx=vx_tem;
   vy=vy_tem;

   if(r(index)>r_sat)
       index2=index;
       break;
   end

end
t_day=t/(60*60*24);
fprintf("Arrive Saturn:t=%f days\n",t_day(index));
fprintf("Speed when arrive the Saturn %f m/s\n",sqrt(vx^2+vy^2));
fprintf("remaining mass %f kg\n\n\n",m0);
fprintf("r %f m\n",r(index));

t_day=t/(60*60*24);
figure(5)
```

```
plot(t_day,r);
xlabel("Time (day)");
ylabel("Distance from the Sun");


figure(6)
draw();
hold on;
plot(x,y,'k');
legend("Earth orbit","Mars orbit","Jupiter orbit","Saturn orbit","satellite orbit
axis equal;
grid on;
xlabel("x-axis (m)");
ylabel("y-axis (m)");
axis equal;

figure(7)
period(x,y,1,index0);
axis equal;

figure(8)
period(x,y,index0,index1);
axis equal;

figure(9)
period(x,y,index1,index2);
axis equal;
```

```
   %function that draw the orbit for seperate stages
function []=period(x,y,index1,index2)
X=zeros(1);
Y=zeros(1);
   for scan=1:index2-index1+1
       X(scan)=x(scan+index1-1);
       Y(scan)=y(scan+index1-1);

   end
       draw();
       hold on;
       plot(X,Y,'-k');
       legend("Earth orbit","Mars orbit","Jupiter orbit","Saturn orbit","satellit
       grid on;
       xlabel("x-axis");
       ylabel("y-axis");
   end
```