Final Interplanetary Project EMA 550

by Nasser M. Abbasi

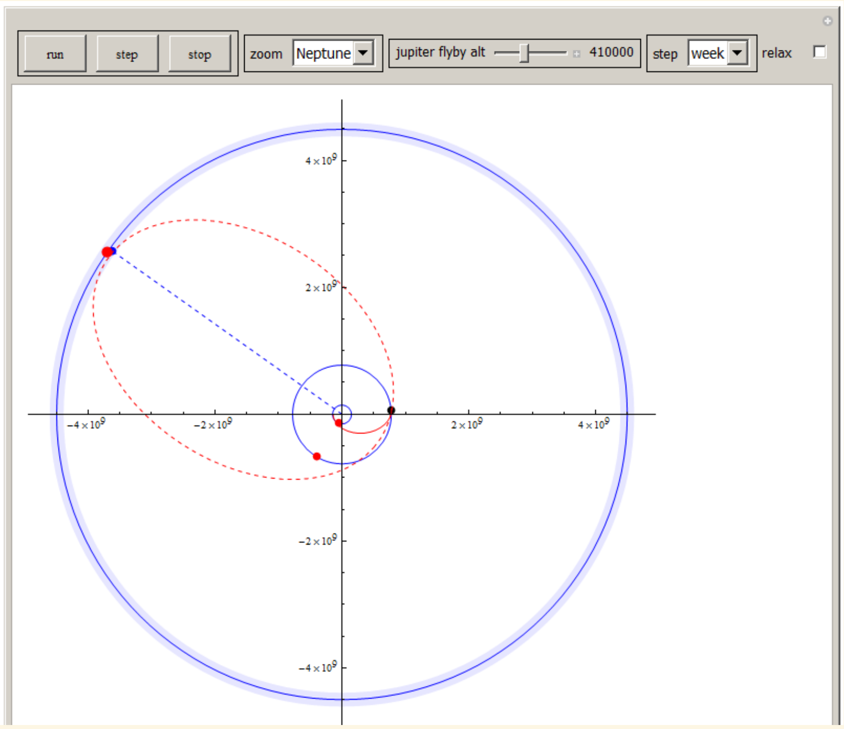
Introduction

The project was broken into 6 phases. This the high level summary of each phase.
1) The first phase was the waiting period to synchronize earth with Jupiter with the correct Hohmann angle. Once this was achieved, the probe is launched from LEO orbit.
2) The second phase is the escape from earth SOI using hyperbolic escape trajectory
3) The third phase is the travel over a Hohmann ellipse to reach Jupiter at the apogee location of the Hohmann transfer ellipse.
4) This stage the probe enters Jupiter SOI and performs a hyperbolic fly-by trajectory. The burnout distance used was based on trial and error experiments using the simulation written for this project in order to obtain a post fly-by ellipse that allowed the probe to reach Neptune orbit at the same time when Neptune was there.
5) This is the post-flyby stage, leaving Jupiter SOI and traveling on an ellipse to Neptune.
6) This is the final phase, the probe is now inside Neptune SOI. It enters a circular orbit around Neptune and remains there.

The final results will be shown here, followed by the step by step calculations done in each phase, then the simulation program will be described.

How was the final trajectory found?

One week of full time work was spend on writing the simulator, as this was the only method to find if a chosen input will lead to the probe meeting Neptune when it arrives to its orbit. The simulator takes as input the initial angular positions of Earth, Jupiter and Neptune in the ecliptic plane and using time step, advances the positions of the planets and the probe on its orbit. This is screen shot of the GUI of the simulator. It allows one to stop, run, and make one step at a time. The step size can be changed from one day to one week to one month.



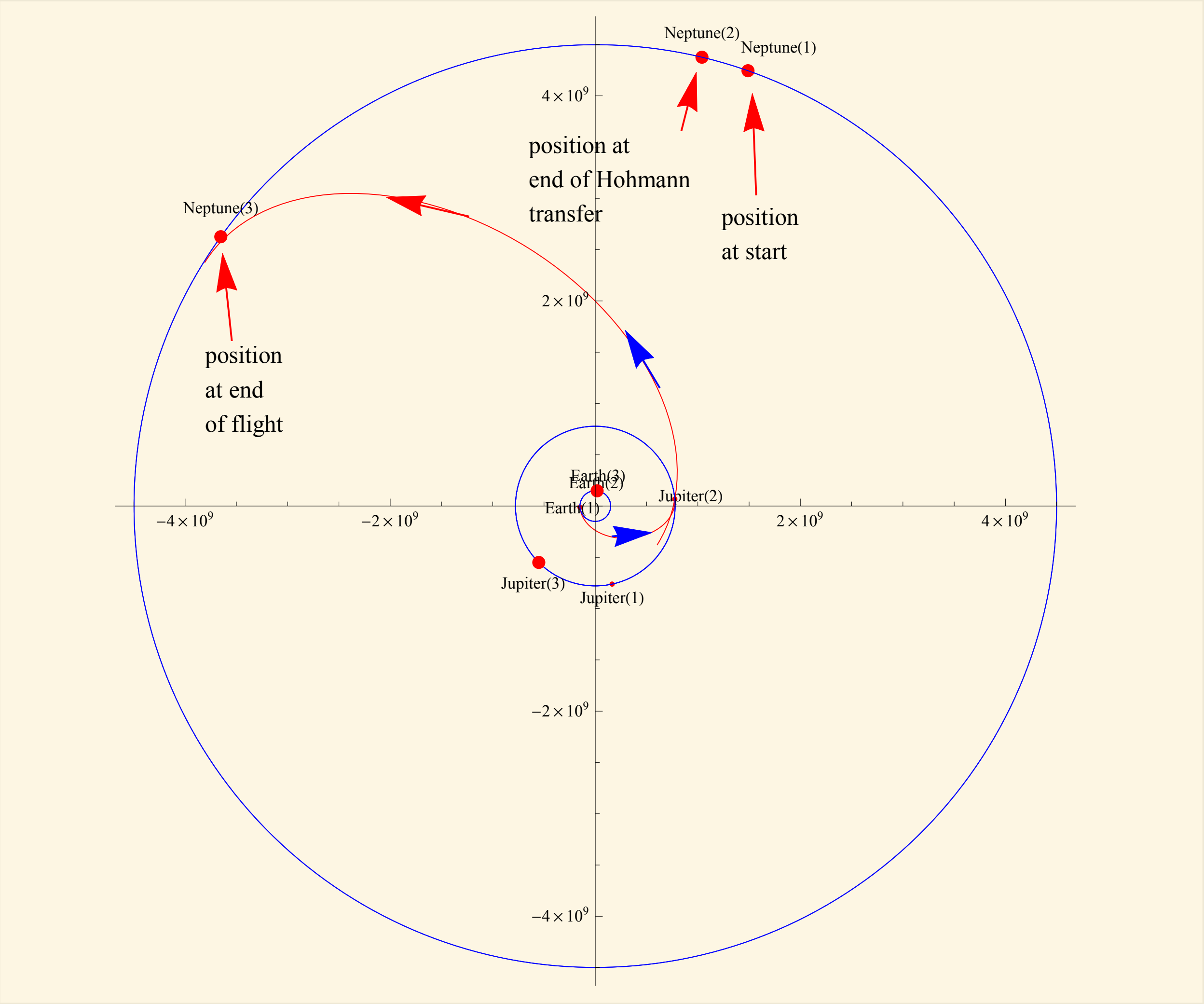
zoom Neptune

jupiter flyby alt 410000

step week

relax □

Once the simulator was completed, different starting positions for Earth, Jupiter and Neptune were tried. Each position used was obtained from the JPL Horizon web site. Different dates were selected. In addition, for each selected initial position, the altitude that the probe will be closest to Jupiter in its fly-by was modified using a slider in simulator. This resulted in different ellipse since the burn out distance r_{bo} is different. The closest altitude to Jupiter (r_{bo}) was modified from 200 000 km to 500 000 km above the surface of Jupiter. When none of the resulting trajectories found to be acceptable, if they did not lead to acceptable rendezvous with Neptune, another starting date was selected and the process was repeated. Acceptable rendezvous with Neptune is one which reaches Neptune within distance less that Neptune's SOI. This is the final trajectory selected



To speed the process of finding the final trajectory, the simulator used a varying time step. The simulation time step can be one day, one week, one month or even one year. However the accuracy of the resulting trajectories will become worst if the time step was made large. When a candidate trajectory was found using large time step (month for example) it was repeated again using one week time step, and then again using one day time step. Using the one day time step, the simulation will take about 15 minutes to complete. So this was a very time consuming part of the project to find the correct trajectory.

This table shows some of the dates and corresponding ecliptic longitude angles showing which initial position was selected

selected?	date	Earth	Jupiter	Neptune	Altitude above Jupiter (KM)
NO	3/21/2014	270	111.3	334.996	many
No	9/21/2014	356.76	126	336	many
No	10/01/2014	7.5	127	336	many
No	03/21/2016	180	166	339	many
No	03/21/2017	180	196	304	many
YES	3/21/2016	180	169	339	410000

Initial positions tried in simulation

■ The following table shows the time history for all the phases on the project

phase	date started	date completed
waiting for correct Hohmann angle between Earth/Jupiter	3/21/2016	12/26/2016
Start on Hohmann transfer, travel to Jupiter SOI	12/26/2016	9/20/2019
Enter and exist Jupiter SOI	9/20/2019	2/29/2020
travel on Ellipse from Jupiter to Neptune	2/29/2020	12/25/2054

Time schedule of complete trajectory

■ Show ΔV for fly-by and compare to Hohmann transfer

Trajectory	ΔV1 (km/s)	ΔV2 (km/s)	Total (km/s)
Fly–by	6.267	13.44	19.71
Direct Hohmann	8.22	14.91	23.133

Compare total ΔV using Fly–by and Direct Hohmann. Saving is over 3 km/sec

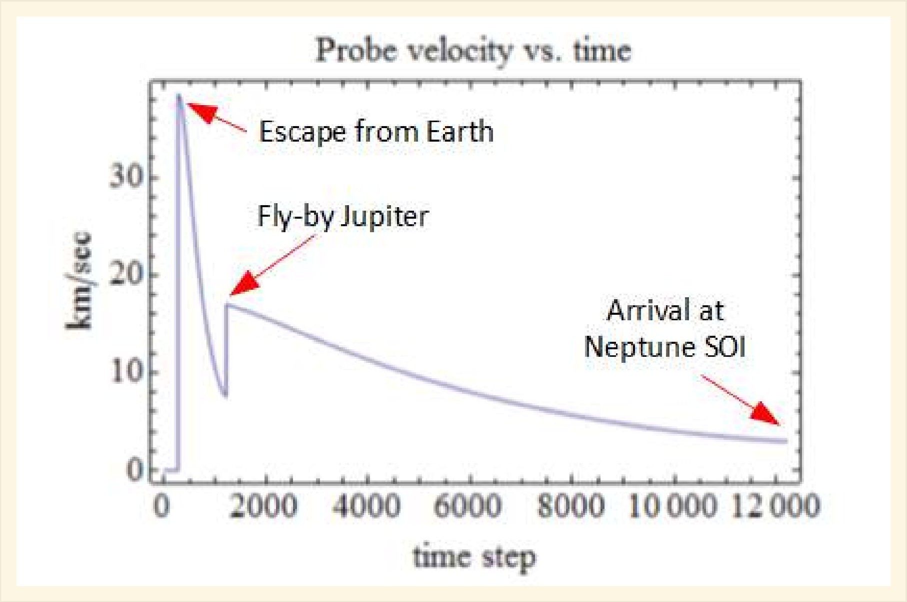
■ Show trajectory information for each phase (relevant data is shown)

Item	Earth escape Hyperbola	Hohmann transfer Earth/Jupiter	Fly–by Jupiter Hyperbola	Elliptical orbit Jupiter/Neptune
eccentricity e	2.291	0.6775	1.1199	0.726
semi–major a (km)	–	4.639×10^{8}	4.01×10^{6}	2.6×10^{9}
V_∞ (km/sec)	8.79	–	5.64	–
Departure speed V_D (km/s)	–	–	17.024	–
η (deg)	115.88	–	153.24	–
Turn angle θ (deg)	64.12	–	126.48	–
Flight path angle γ_d (deg)	–	–	15.45	–
True anomaly f (deg)	–	–	36.98	–

Orbits data found

■ Velocity profile of probe showing speed gain due to flyby

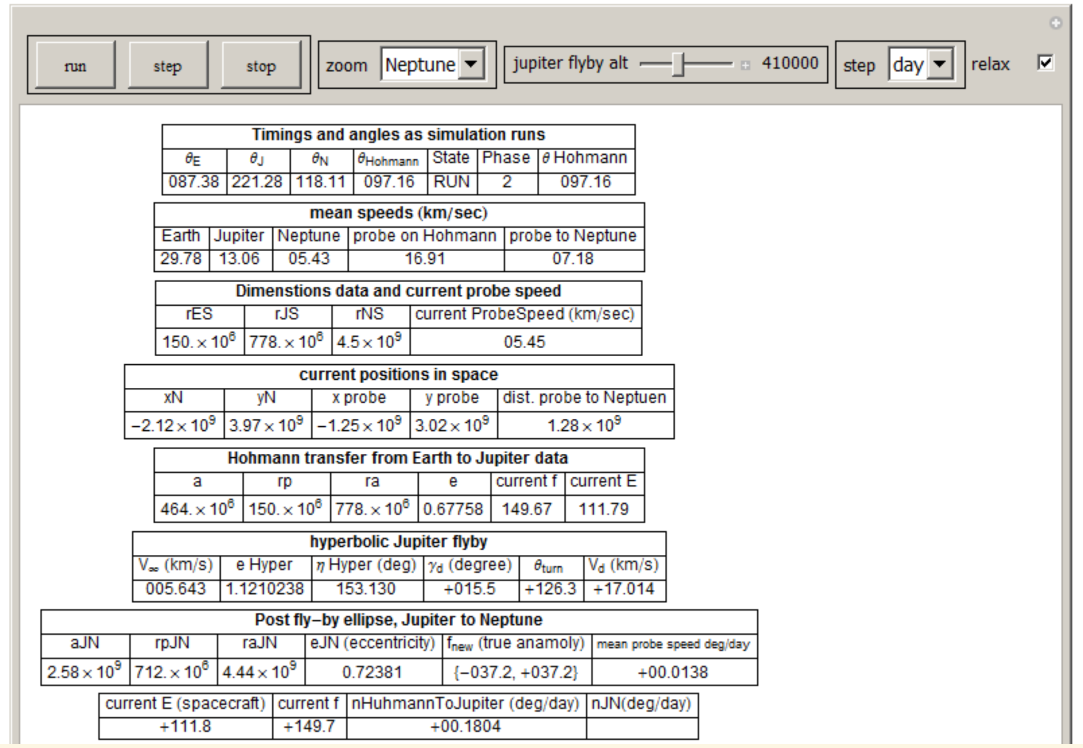
The simulator keeps track of current velocity of probe as it travels starting from Earth all the way to Neptune. It then plots the velocity vs. Time of the probe. This plot below was generated by the simulator and shows the speed gained during the fly-by phase. ΔV gained due to flyby was found to be 10.077 km/sec. This is free ΔV due to gravity assist.



The above shows that the fly-by Jupiter gave the probe almost 8 km/sec boost in speed relative to Sun.

Trajectory data gathering

The simulator contains an option to display all the information about the trajectory during its running. This display can be turned off if needed. This allows one to monitor each aspect of the orbit as it runs. Here is a screen show showing typical display during one simulation run



zoom Neptune

jupiter flyby alt 410000

step day

relax ☑

Timings and angles as simulation runs

θ_E	θ_J	θ_N	$\theta_{Hohmann}$	State	Phase	θ Hohmann
087.38	221.28	118.11	097.16	RUN	2	097.16

mean speeds (km/sec)

Earth	Jupiter	Neptune	probe on Hohmann	probe to Neptune
29.78	13.06	05.43	16.91	07.18

Dimenstions data and current probe speed

rES	rJS	rNS	current ProbeSpeed (km/sec)
$150. \times 10^6$	$778. \times 10^6$	4.5×10^9	05.45

current positions in space

xN	yN	x probe	y probe	dist. probe to Neptuen
-2.12×10^9	3.97×10^9	-1.25×10^9	3.02×10^9	1.28×10^9

Hohmann transfer from Earth to Jupiter data

a	rp	ra	e	current f	current E
$464. \times 10^6$	$150. \times 10^6$	$778. \times 10^6$	0.67758	149.67	111.79

hyperbolic Jupiter flyby

V_∞ (km/s)	e Hyper	η Hyper (deg)	γ_d (degree)	θ_{turn}	V_d (km/s)
005.643	1.1210238	153.130	+015.5	+126.3	+17.014

Post fly–by ellipse, Jupiter to Neptune

aJN	rpJN	raJN	eJN (eccentricity)	f_{new} (true anamoly)	mean probe speed deg/day
2.58×10^9	$712. \times 10^6$	4.44×10^9	0.72381	{−037.2, +037.2}	+00.0138

current E (spacecraft)	current f	nHuhmannToJupiter (deg/day)	nJN(deg/day)
+111.8	+149.7	+00.1804	

Step by step calculations

Constants used

In[7]:=

```
<< Calendar`
AU = 1.495978 * 10^8;
rE  = 6378; (*Earth radius*)
rJ = 71 492;(*Jupiter radius*)
rN = 24 764;(*Neptune radius*)
rES = 1 AU;(*Earth distance from sun*)
rJS = 5.203 AU;(*Jupiter distance from sun*)
rNS = 30.07 AU; (*Neptune distance from sun*)
(* SOI for each planet *)
eSOI = 9.24 * 10^5;
jSOI = 4.82 * 10^7;
nSOI = 8.67 * 10^7;
(*mu for each planet*)
μSun = 1.327 * 10^11;
μE = 3.986 * 10^5;
μJ = 126 686 534;
μN = 6 836 529;

(*speed of each planet, all relative to sun*)
```

$$sE = \sqrt{\frac{\mu Sun}{rES}};\text{(*km/sec*)}$$

$$sJ = \sqrt{\frac{\mu Sun}{rJS}};\text{(*km/sec*)}$$

$$sN = \sqrt{\frac{\mu Sun}{rNS}};\text{(*km/sec*)}$$

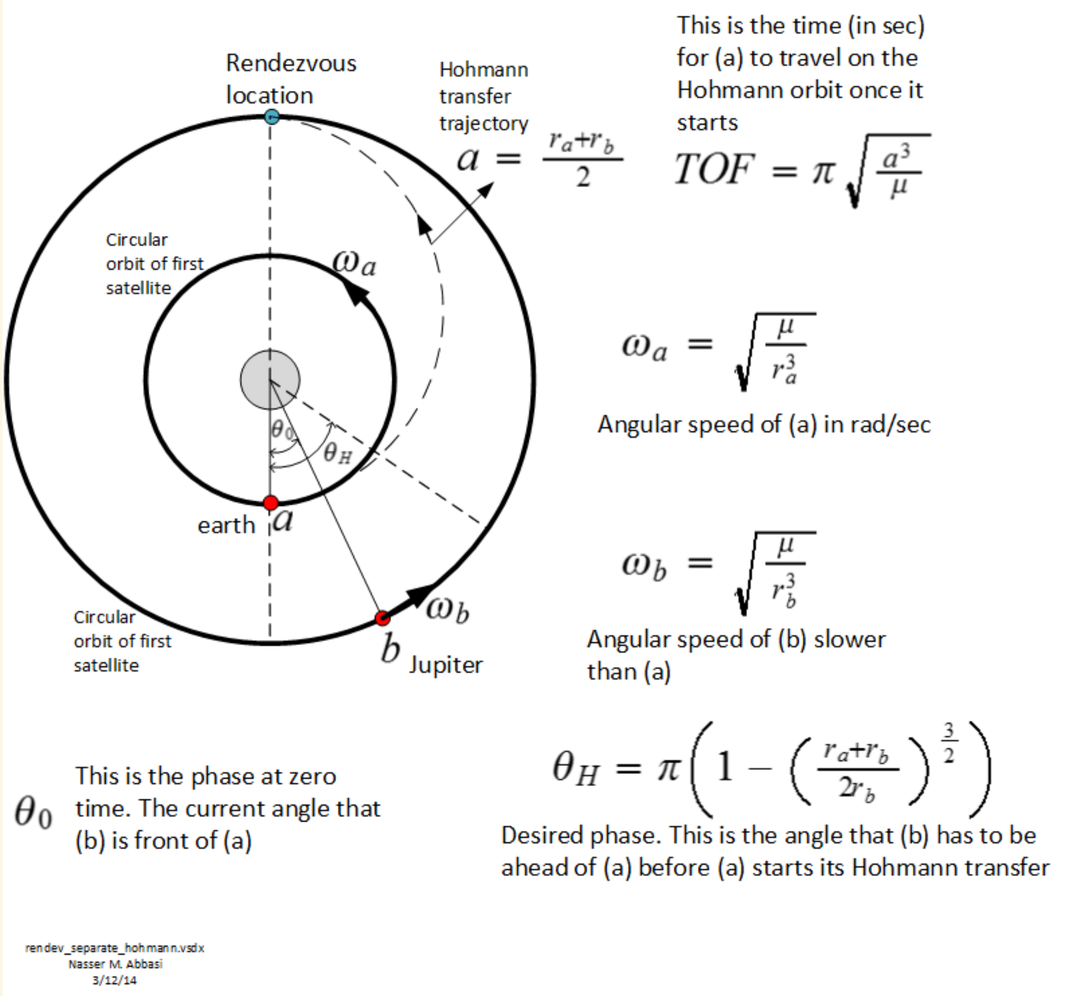
```
(*angular velocity of each planet*)
```

$$\omega E = \sqrt{\frac{\mu Sun}{rES^3}};\text{(*angular vecloity of earth*)}$$

$$\omega J = \sqrt{\frac{\mu Sun}{rJS^3}};\text{(*angular vecloity of earth*)}$$

$$\omega N = \sqrt{\frac{\mu Sun}{rNS^3}};\text{(*angular vecloity of earth*)}$$

Find the Hohmann angle needed rendezvous between Earth and Jupiter


Rendezvous location
Hohmann transfer trajectory
Circular orbit of first satellite
ω_a
θ_0
θ_H
earth
a
Circular orbit of first satellite
ω_b
b
Jupiter

$$a = \frac{r_a + r_b}{2}$$

This is the time (in sec) for (a) to travel on the Hohmann orbit once it starts

$$TOF = \pi\sqrt{\frac{a^3}{\mu}}$$

$$\omega_a = \sqrt{\frac{\mu}{r_a^3}}$$

Angular speed of (a) in rad/sec

$$\omega_b = \sqrt{\frac{\mu}{r_b^3}}$$

Angular speed of (b) slower than (a)

θ_0 This is the phase at zero time. The current angle that (b) is front of (a)

$$\theta_H = \pi\left(1 - \left(\frac{r_a + r_b}{2r_b}\right)^{\frac{3}{2}}\right)$$

Desired phase. This is the angle that (b) has to be ahead of (a) before (a) starts its Hohmann transfer

rendev_separate_hohmann.vsdx
Nasser M. Abbasi
3/12/14

In[28]:=

```
θEarthJupitor = Pi (1 - ((rES + rJS)/(2 rJS))^(3/2));
N@θEarthJupitor * 180 / Pi
```

Out[29]=

```
97.15821569
```

- Enter the initial positions. These have been found by simulation first. The simulation includes all these steps build into it. There are shown here in order to be able to show each step done outside of the simulation code.
  Note that 90 degrees were added to each position to make it compatible with standard coordinate system with positive x points to the right

```
θE0 = Mod[180 + 90, 360] Degree; (*Earth*)
θJ0 = (169 + 90) Degree; (*Jupiter*)
θN0 = Mod[(339 + 90), 360] Degree; (*Neptune*)
```

find wait time between Earth and Jupiter in order to find date when start Hohmann transfer.

- Find θ_0 the initial angle between earth and Jupiter at initial configuration

```
θ0 = θJ0 − θE0;
θ0 * 180. / Pi
```

```
−11.
```

- Adjust θ_0 if $\theta_H$is larger than θ_0 by adding 2π so not to get negative time

```
If [θ0 ≤ θEarthJupiter, θ0 = θ0 + 2 Pi];
θ0 * 180. / Pi
```

```
349.
```

- calculate wait time before starting Hohmann transfer. This is the time needed to sync with Jupiter

$$\text{waitTimeEarthJupiter0} = \frac{\theta 0 - \theta\text{EarthJupiter}}{\omega\text{E} - \omega\text{J}};$$

```
waitTimeEarthJupiter0 / (60 * 60 * 24) (*days*)
```

```
279.0431558
```

- Display the date the Hohmann transfer starts

```
currentDate = {2016, 3, 21};
currentDate = DaysPlus[currentDate, Ceiling[waitTimeEarthJupiter0 / (60 * 60 * 24)]]
```

```
{2016, 12, 26}
```

- Find *a* for the Hohmann transfer

$$\text{aEJ} = \frac{\text{rES} + \text{rJS}}{2}$$

$$4.639775767 \times 10^{8}$$

▫ find time of flight on the Hohmann transfer

$$\text{tof} = \pi \sqrt{\frac{\text{aEJ}^3}{\mu\text{Sun}}};$$

```
tof / (60 * 60 * 24 * 365) (*years*)
```

```
2.73308597
```

▫ Find total wait time which includes sync time and time of flight over Hohmann transfer

```
waitTimeEarthJupiter = waitTimeEarthJupiter0 + tof;
waitTimeEarthJupiter / ((60 * 60 * 24 * 365)) (*years*)
```

```
3.497587767
```

▫ display the date probe arrives to Jupiter SOI

```
currentDate = DaysPlus[currentDate, Round[tof / (60 * 60 * 24)]]
```

```
{2019, 9, 20}
```

▪ Make function to convert Gregorian date to Julian day (Not used at this time)

$$\text{toJD}[d\_, m\_, y\_] := 367\, y - \text{IntegerPart}\left[\frac{7\left(y + \text{IntegerPart}\left[\frac{m+9}{12}\right]\right)}{4}\right] + \text{IntegerPart}\left[\frac{275\, m}{9}\right] + d + 1\,721\,013.5;$$

```
toJD[20, 10, 2014]
```

$$2.4569505 \times 10^6$$

Hyperbolic escape from Earth

▫ Find eccentricity of Hohmann transfer ellipse

$$\text{eEJ} = \frac{\text{rJS} - \text{rES}}{\text{rES} + \text{rJS}}$$

```
0.6775753668
```

Find semi-minor axes for Hohmann ellipse (km)

$$bEJ = aEJ\sqrt{1 - eEJ^2}$$

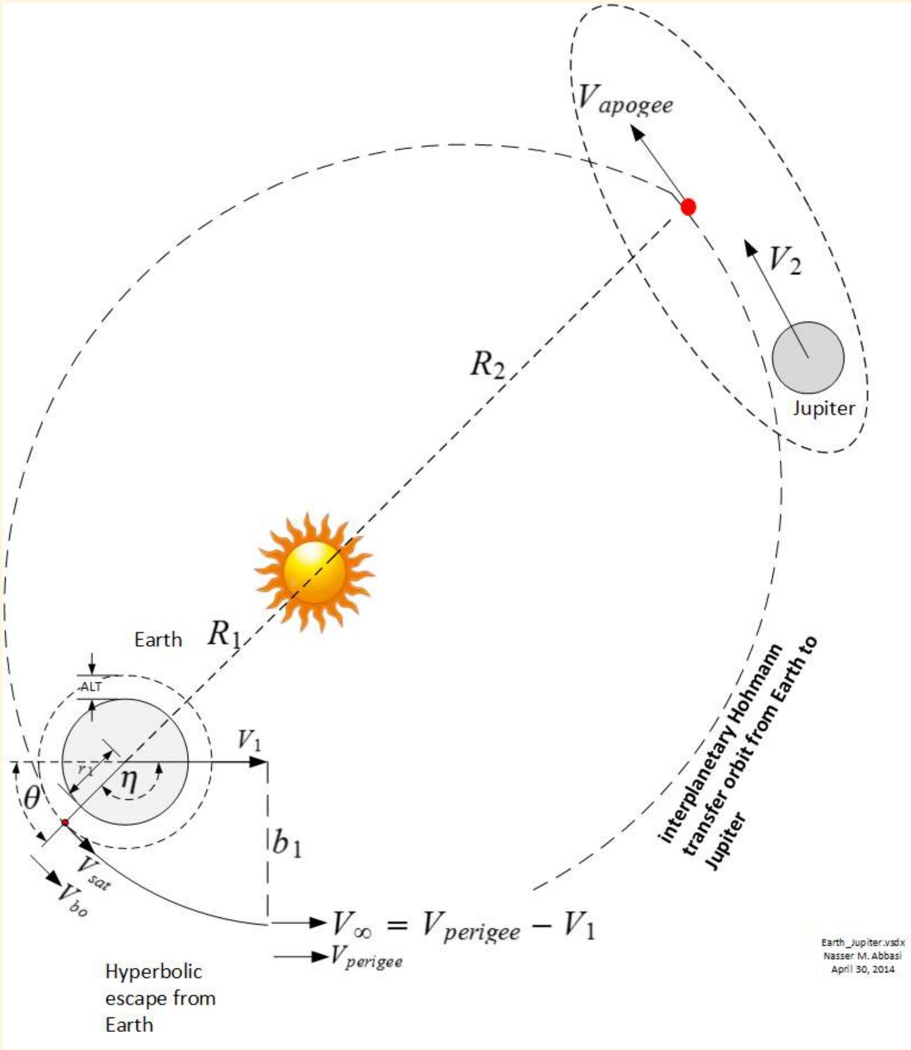
3.412338607×10^{8}

Find velocity at perigee V_p (KM/sec)

$$vp = \sqrt{\mu Sun\left(\frac{2}{rES} - \frac{1}{aEJ}\right)}$$

38.57570557

Show drawing (not to scale) of Earth escape hyperbolic trajectory



Find V_∞ to escape earth (km/sec)

$$\text{vInf} = \text{vp} - \sqrt{\frac{\mu\text{Sun}}{\text{rES}}}$$

```
8.792402687
```

Find r_{bo} the burn out radius (km)

```
rbo = rE + 300 (*300 KM is altitude*)
```

```
6678
```

Find V_{bo} the burn out speed using the energy equation (km/sec)

```
Clear[vbo];
```

$$\text{eq} = \frac{\text{vbo}^2}{2} - \frac{\mu\text{E}}{\text{rbo}} == \frac{\text{vInf}^2}{2} - \frac{\mu\text{E}}{\text{eSOI}};$$

```
vbo = First@Select[vbo /. NSolve[eq, vbo], # > 0 &]
```

```
13.99359259
```

Find ΔV_1 needed to escape earth

$$\text{delV1} = \text{Abs}\left[\text{vbo} - \sqrt{\frac{\mu\text{E}}{\text{rbo}}}\right]$$

```
6.267757388
```

Calculate the eccentricity of the hyperbolic escape from earth

$$\text{e} = \sqrt{1 + \frac{\text{vInf}^2\,\text{vbo}^2\,\text{rbo}^2}{\mu\text{E}^2}}$$

```
2.291080512
```

Calculate angle η where ΔV should be applied

$$\eta = \text{ArcCos}\left[\frac{-1}{\text{e}}\right];$$

$$\eta * 180/\text{Pi}$$

```
115.8792052
```

▫ Find the turn angle θ

```
θ = Pi − η;
θ * 180 / Pi
```

```
64.12079477
```

Hohmann transfer between Earth and Jupiter

▫ Find mean angular velocity on the Hohmann ellipses (rad/sec)

$$\text{nHuhmannToJupiter} = \sqrt{\frac{\mu\text{Sun}}{\text{aEJ}^3}}$$

$$3.644936553 \times 10^{-8}$$

▫ Find the angular positions that earth and Jupiter will have at the end of the Hohmann transfer. We calculated the time of flight from above. So using this time, and knowing the angular velocity of Earth the Jupiter, we can find the new angular positions in ecliptic plane.

First display time of flight to Jupiter in days (this is half the period of the Hohmann transfer ellipse)

```
(tof) / (60 * 60 * 24)
```

```
997.5763791
```

▫ Find the angle the earth will be at when probe starts Hohmann orbit

```
θE1 = θE0 + ωE * waitTimeEarthJupiter0;
Mod[θE1, 2 Pi] * 180 / Pi
```

```
185.0143788
```

▫ Find the angle Jupiter will be at when probe starts Hohmann orbit

```
θJ1 = θJ0 + ωJ * waitTimeEarthJupiter0;
Mod[θJ1, 2 Pi] * 180 / Pi
```

```
282.1725945
```

Find the angle the earth will be at when probe reach end of Hohmann to Jupiter

```
θE2 = θE0 + ωE * (waitTimeEarthJupiter0 + tof);
Mod[θE2, 2 Pi] * 180 / Pi
```

```
88.18792204
```

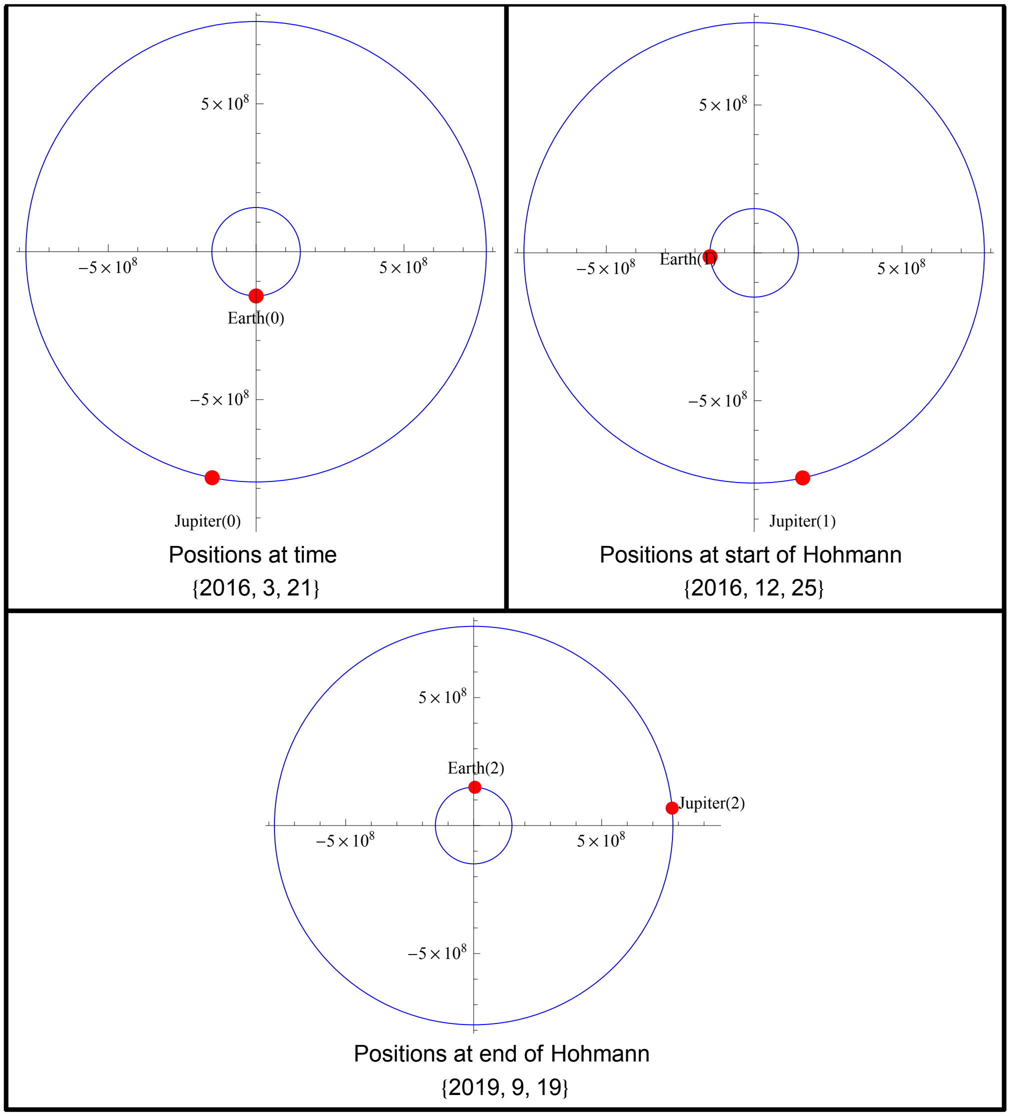
Find the angle Jupiter will be at with probe reach Jupiter

```
θJ2 = θJ0 + ωJ * (waitTimeEarthJupiter0 + tof);
Mod[θJ2, 2 Pi] * 180 / Pi
```

```
5.014378828
```

Draw diagram showing the initial Earth/Jupiter positions at t=0 and at start of Hohmann transfer and at end of Hohmann transfer

```
p
```

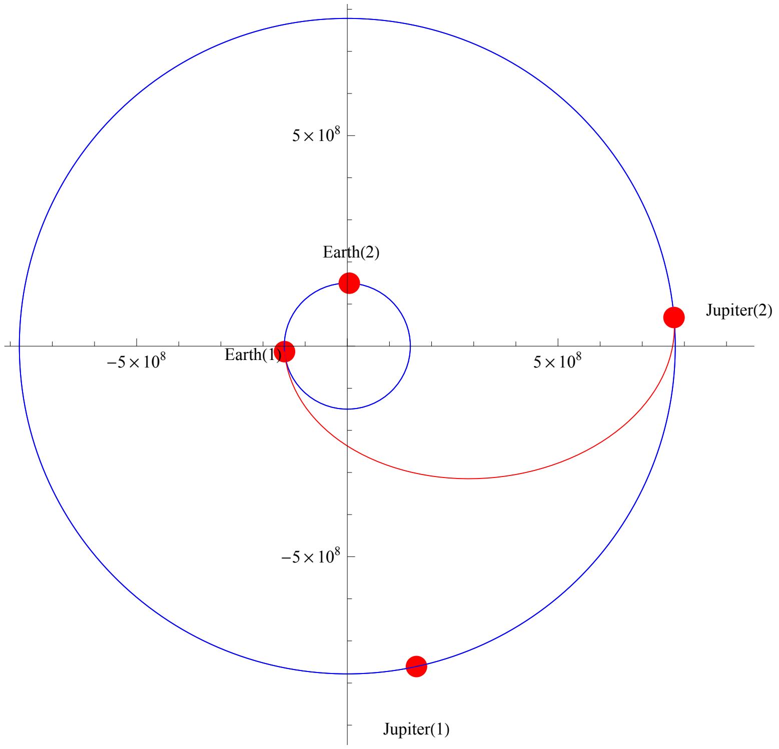


Positions at time {2016, 3, 21}

Positions at start of Hohmann {2016, 12, 25}

Positions at end of Hohmann {2019, 9, 19}

Draw diagram showing the Hohmann elliptic transfer orbit showing initial positions of planets and final positions all on one diagram



Configuration at end of Hohmann transfer
{2019, 9, 19}
with positions indicated by (1),(2) for start/end of transfer

Before making the fly-by Jupiter calculations, lets show the above diagram along with the position of Neptune as well. All to scale. Find the angle Neptune will be at when probe starts on Hohmann transfer from Earth to Jupiter

```
θN1 = θN0 + ωN * waitTimeEarthJupiter0;
Mod[θN1, 2 Pi] * 180 / Pi
```

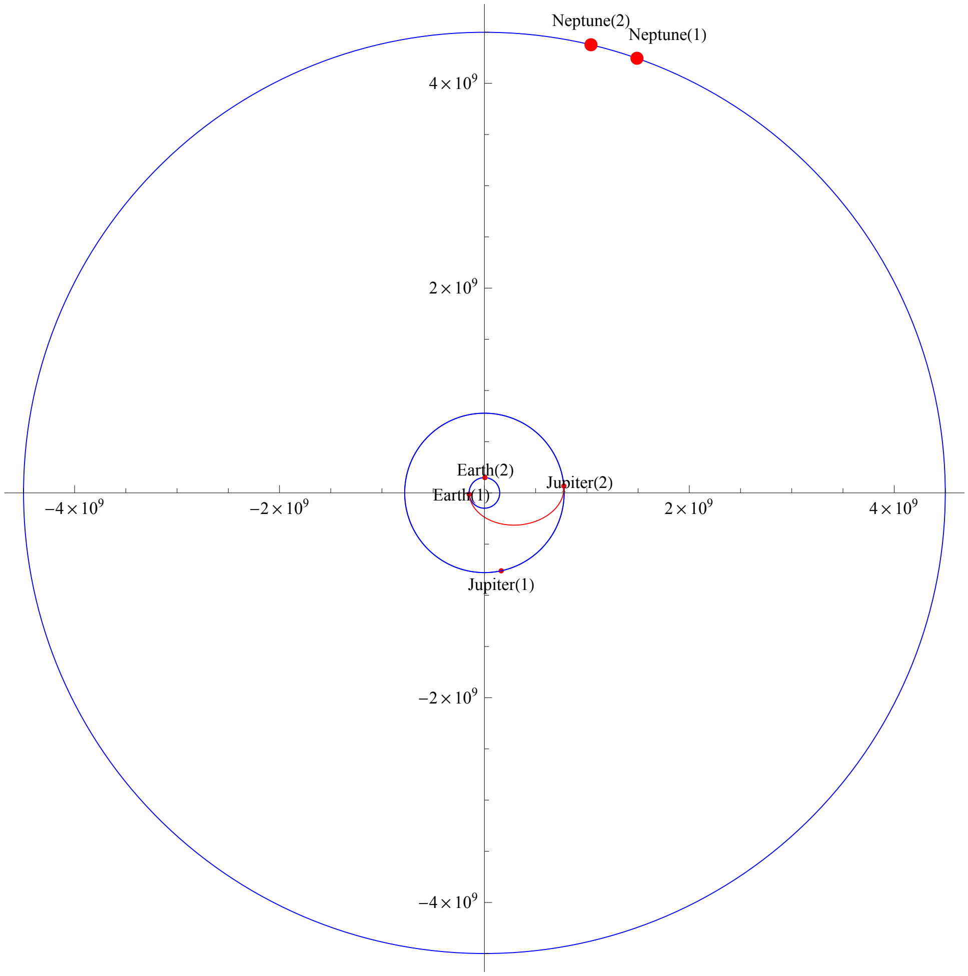
```
70.66784335
```

Find the angle Neptune will be at when probe ends the Hohmann transfer from Earth to Jupiter

```
θN2 = θN0 + ωN * (waitTimeEarthJupiter0 + tof);
Mod[θN2, 2 Pi] * 180 / Pi (*degree*)
```

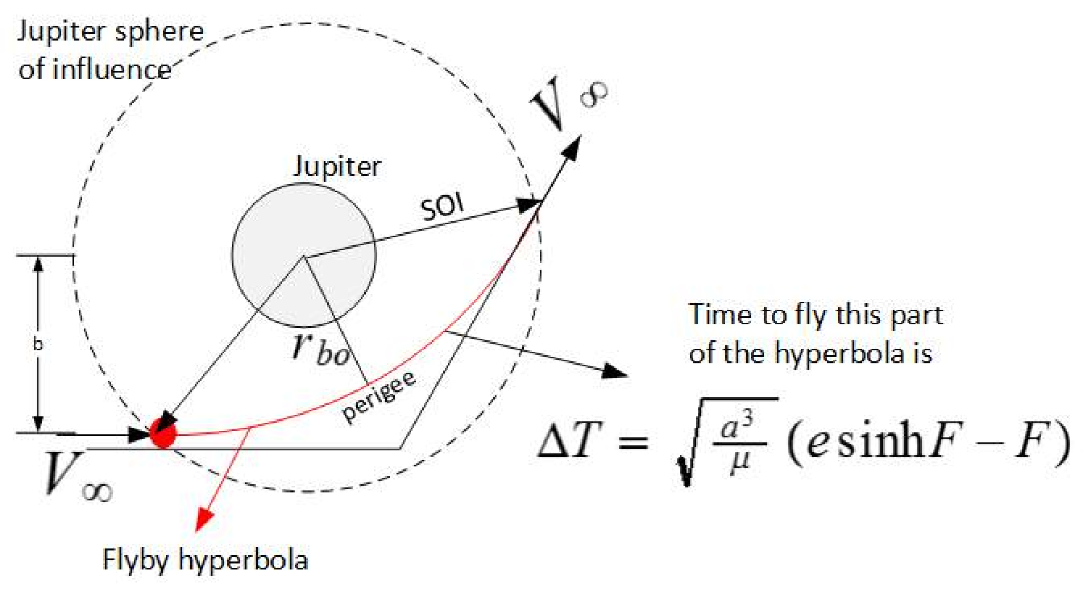
```
76.630366
```

◻ Draw diagram of the three planets at end of Hohmann transfer



Configuration at end of Hohmann transfer, all planets
{2019, 9, 19}
with positions indicated by (1),(2) for start/end of transfer

Fly-by Jupiter, Hyperbolic flyby



- Set the burn out radius r_{bo} . This was found by simulation below in order to obtain the rendezvous with Neptune (km)

```
alt = 410 000;
rbo = alt + rJ
```

```
481 492
```

- Find probe speed at entrance to Jupiter SOI

$$\text{va} = \sqrt{\mu\text{Sun}\left(\frac{2}{\text{rJS}} - \frac{1}{\text{aEJ}}\right)}\ (*\text{velocity of craft atJupiter entrance}*)$$

```
7.414127535
```

- Find V_∞ for Jupiter flyby (km/sec)

```
vInf = sJ - va
```

```
5.642948859
```

Find V_{bo} the burn out speed using the energy equation (km/sec)

Clear[vbo];

$$eq = \frac{vbo^2}{2} - \frac{\mu J}{rbo} == \frac{vInf^2}{2} - \frac{\mu J}{jSOI};$$

```
vbo = First@Select[vbo /. NSolve[eq, vbo], # > 0 &]
```

```
23.5119341
```

Calculate the eccentricity of the hyperbolic escape from Jupiter

$$e = \sqrt{1 + \frac{vInf^2\, vbo^2\, rbo^2}{\mu J^2}}$$

```
1.119944854
```

Find η angle

$$\eta = \text{ArcCos}\left[\frac{-1}{e}\right];$$

$$\eta * 180 / \text{Pi}$$

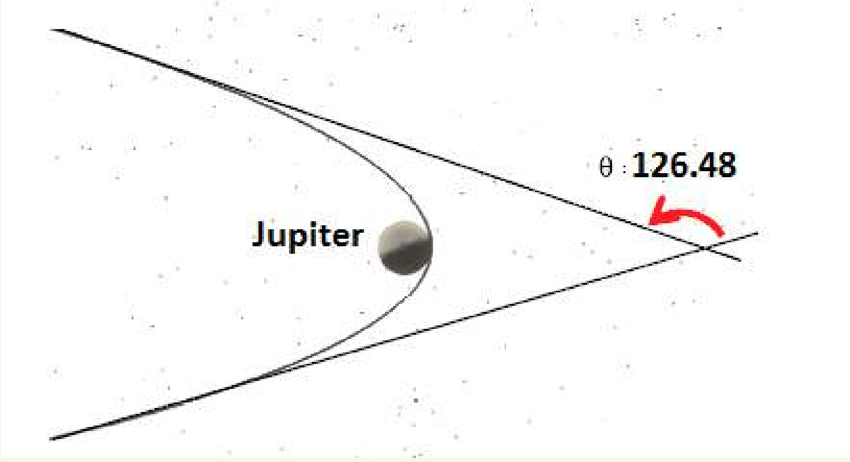
```
153.2400935
```

Find the turn angle

$$\theta = 2\,\eta - \text{Pi};$$

$$\theta * 180 / \text{Pi}$$

```
126.4801869
```



Find impact parameter *b* (km)

```
Clear[b]
eq = b vInf == rbo vbo;
b /. First@Solve[eq, b];
b = %
```

```
2.006186563×10^6
```

Find the departure velocity (km/sec)

```
vdJN = √(sJ^2 + vInf^2 − 2 sJ vInf Cos[θ])
```

```
17.02770468
```

Find semi-major axes (km) of the Hyperbolic fly-by trajectory. Since r_{bo} is r_p for the Hyperbolic, we can use $r_{bo} = a(e-1)$ to solve for *a*

```
Clear[aHyper]
eq = rbo == aHyper (e − 1);
aHyper = aHyper /. First@Solve[eq, aHyper]
```

```
4.014278105×10^6
```

Find the time probe is inside Jupiter SOI during fly-by. First, find the eccentric anomaly *F* of the hyperbolic trajectory when probe at SOI

```
Clear[F0];
eq = jSOI == aHyper (e Cosh[F0] − 1);
F0 = First@Select[(F0 /. NSolve[eq, F0, Reals]), # > 0 &]
```

```
3.143507611
```

Find the time inside Jupiter SOI. More than 4 months are spent inside Jupiter SOI. Yet, in the patched conic approximation, we assume the fly-by happens instantly and this time in simulation is not accounted for. But this is approximation.

```
tJ = 2 √(aHyper^3 / μJ) (e Sinh[F0] − F0);
tJ / (60 * 60 * 24) (*days*)
```

```
162.3555079
```

Find the date when probe leaves Jupiter SOI

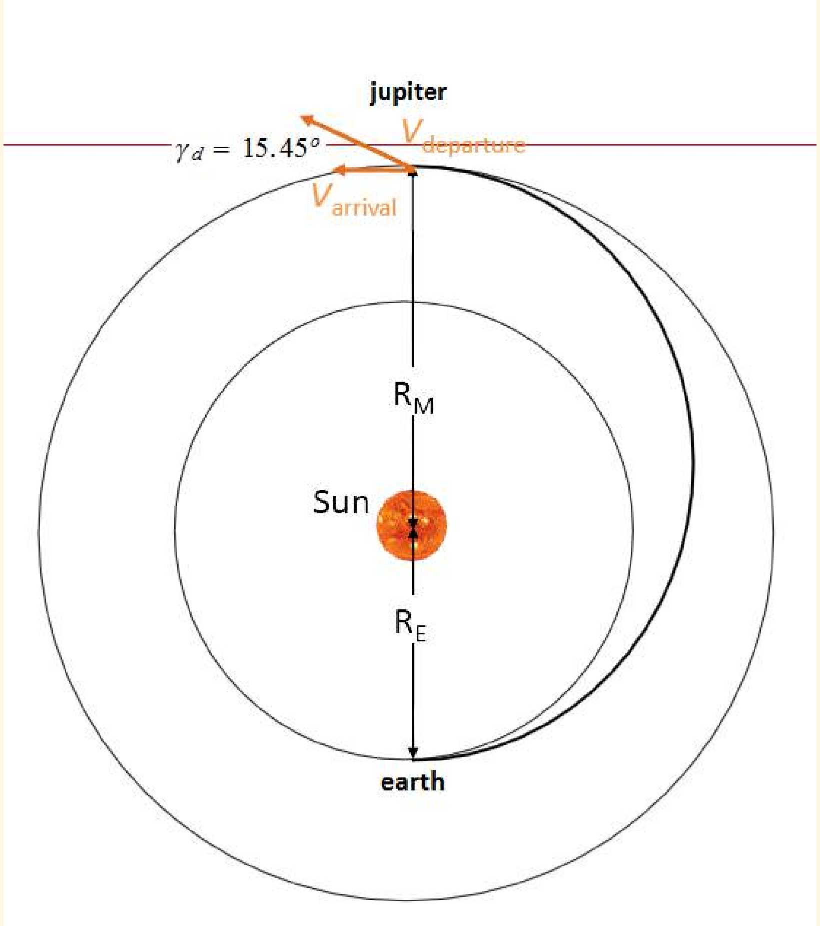
```
currentDate = DaysPlus[currentDate, Round[tJ / (60 * 60 * 24)]]
```

```
{2020, 2, 29}
```

Find the flight path angle at Jupiter for the new ellipse (post-fly ellipse) relative to sun

```
Clear[z];
sol = Quiet[Solve[Sin[z] == vInf Sin[θ] / vdJN, z]];
γ = z /. First@sol;
γ * 180 / Pi
```

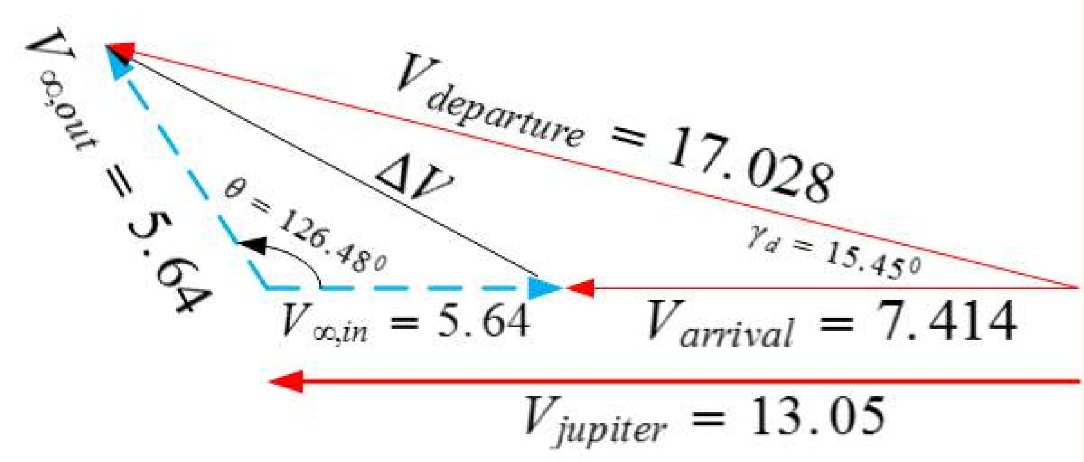
```
15.45400848
```



Departure from Jupiter. (Image edited from class handouts)

Velocity diagram of the fly-by Jupiter

A summary of the above calculations is now given in terms of velocity diagram



Velocity diagram for fly-by Jupiter. Red color means speed is relative to Sun. Blue color means relative to Jupiter. All values are in KM/SEC

▫ Find ΔV due to fly-by (km/sec)

```
delV = 2 vInf Sin[θ/2]
```

```
10.07719057
```

Post-fly by calculations of new Ellipse

▫ Find the semi-major axes *a* of the post-fly ellipses (KM)

```
Clear[z];
eq = vdJN == Sqrt[μSun (2/rJS - 1/z)];
aJN = z /. First@NSolve[eq, z]
```

2.600341362×10^9

□ Find the eccentricity of the post-fly ellipse, to transfer to Neptune

```
Clear[z];
```

$$eq = \text{Cos}[\gamma] == \sqrt{\frac{aJN^{2}\left(1-z^{2}\right)}{rJS\,(2\,aJN - rJS)}}\;;$$

```
sol = NSolve[eq, z];
sol = z /. sol;
e = First@Select[sol, # > 0 &]
```

```
0.7260062019
```

□ Find true anomaly at Jupiter for the new ellipse

```
Clear[z];
```

$$eq = rJS == \frac{aJN\left(1-e^{2}\right)}{1 + e\,\text{Cos}[z]};$$

```
(*sol=z/.First@FindRoot[eq,{z,θJ2}];*)
sol = z /. FindRoot[eq, {z, Pi/8}];
fJN = sol;
fJN*180/Pi
```

```
36.98646614
```

□ Since γ was positive (from above) then the true anomaly will be between zero and 180

□ Find r_p of the new ellipse (km)

```
rpJN = aJN (1 - e)
```

$$7.12477406\times10^{8}$$

□ Find r_a of the new ellipse (km)

```
raJN = aJN (1 + e)
```

$$4.488205318\times10^{9}$$

□ Find semi-minor axes of new ellipse (km)

$$bJN = aJN\sqrt{1-e^{2}}$$

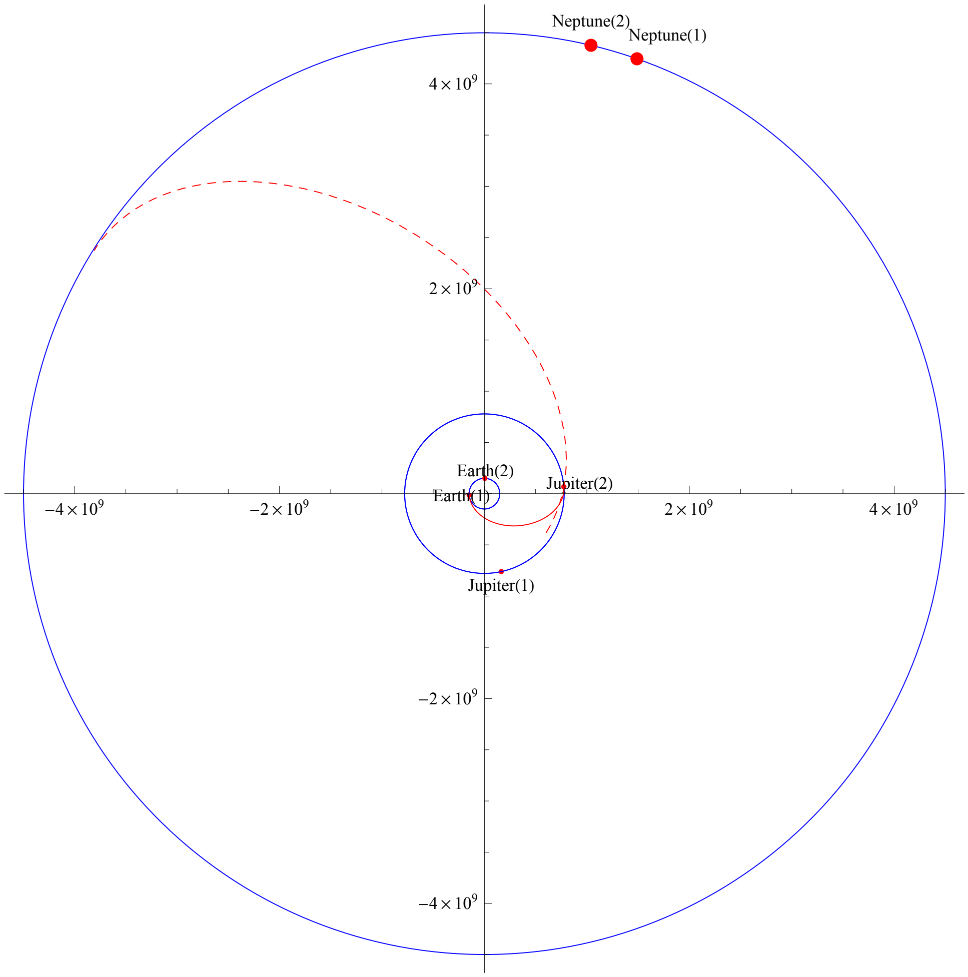
$$1.788223946\times10^{9}$$

- Find center of new ellipse

```
xc2 = −aJN∗e;
yc2 = 0;
```

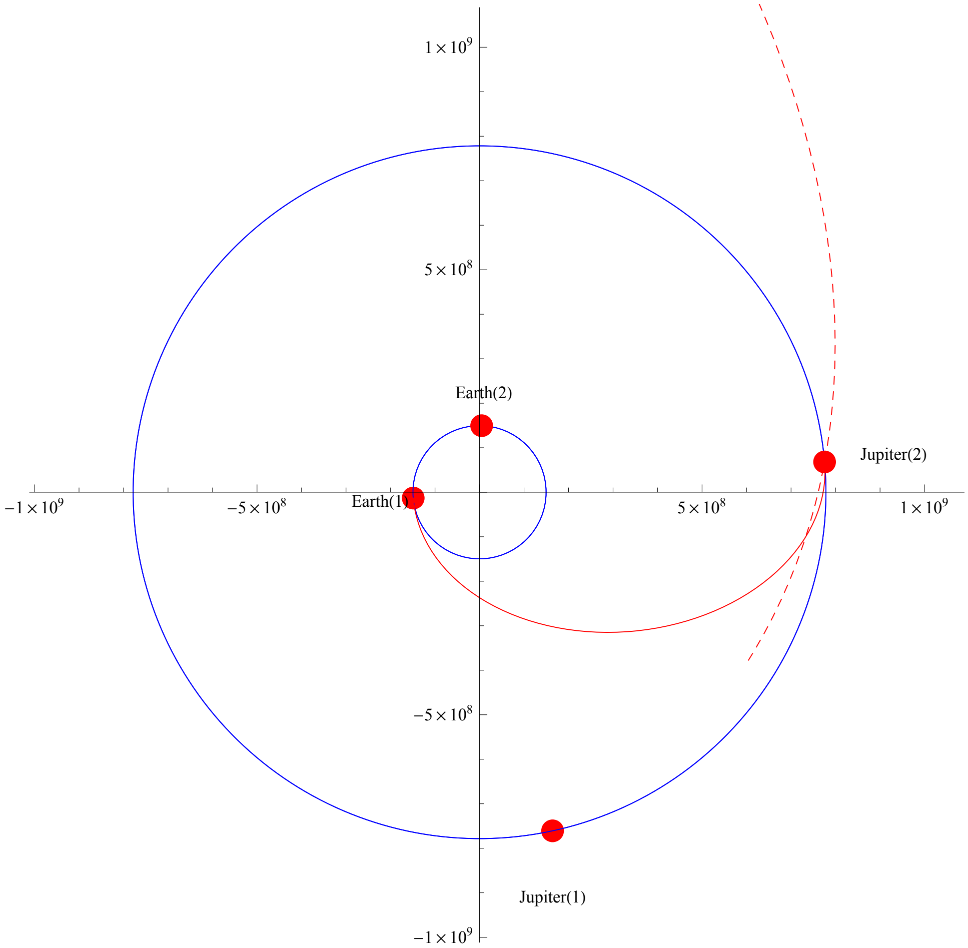
Transfer on new ellipse from Jupiter to Neptune, post-flyby

- Now that the new ellipse is found, it can be drawn to scale to show all trajectories found so far. This shows at the time when the Hohmann transfer was just completed with the new Ellipse draw showing the trajectory from Jupiter to Neptune, but the actual transfer has not started yet



Configuration at end of Hohmann transfer, all planets and final ellipse
{2019, 9, 19}
with positions indicated by (1),(2) for start/end of Hohmann transfer

▫ zoom into the above diagram showing the flyby Jupiter area



Showing the fly−by effect and the flight path angle
{2020, 2, 29}
with positions indicated by (1),(2) for start/end of Hohmann transfer

Show the positions of planets at end of trajectory when probe enters Neptune SOI. First find the time it takes to travel from Jupiter to Neptune on the new ellipse

Find E1, and E2 for new ellipse

```
Clear[E1];
eq = rJS == aJN (1 - e Cos[E1]);
E1 = First@Select[E1 /. Quiet[NSolve[eq, E1]], # > 0 &];
E1 * 180 / Pi
```

```
15.18098712
```

```
Clear[E2];
eq = rNS == aJN (1 - e Cos[E2]);
E2 = (E2 /. Quiet[FindRoot[eq, {E2, Pi / 5}]]);
E2 * 180 / Pi
```

```
179.9999997
```

$$\text{timeOfFlyOnNewEllipse} = \sqrt{\frac{\text{aJN}^{3}}{\mu\text{Sun}}}\,((\text{E2} - \text{E1}) - \text{e}\,\text{Sin}[\text{E2} - \text{E1}])$$

$$9.779114032\times 10^{8}$$

In days

```
tof2 = timeOfFlyOnNewEllipse / (60 * 60 * 24) + 1400
```

```
12718.41902
```

Find date it arrives to Neptune SOI

```
currentDate = DaysPlus[currentDate, Round[tof2]]
```

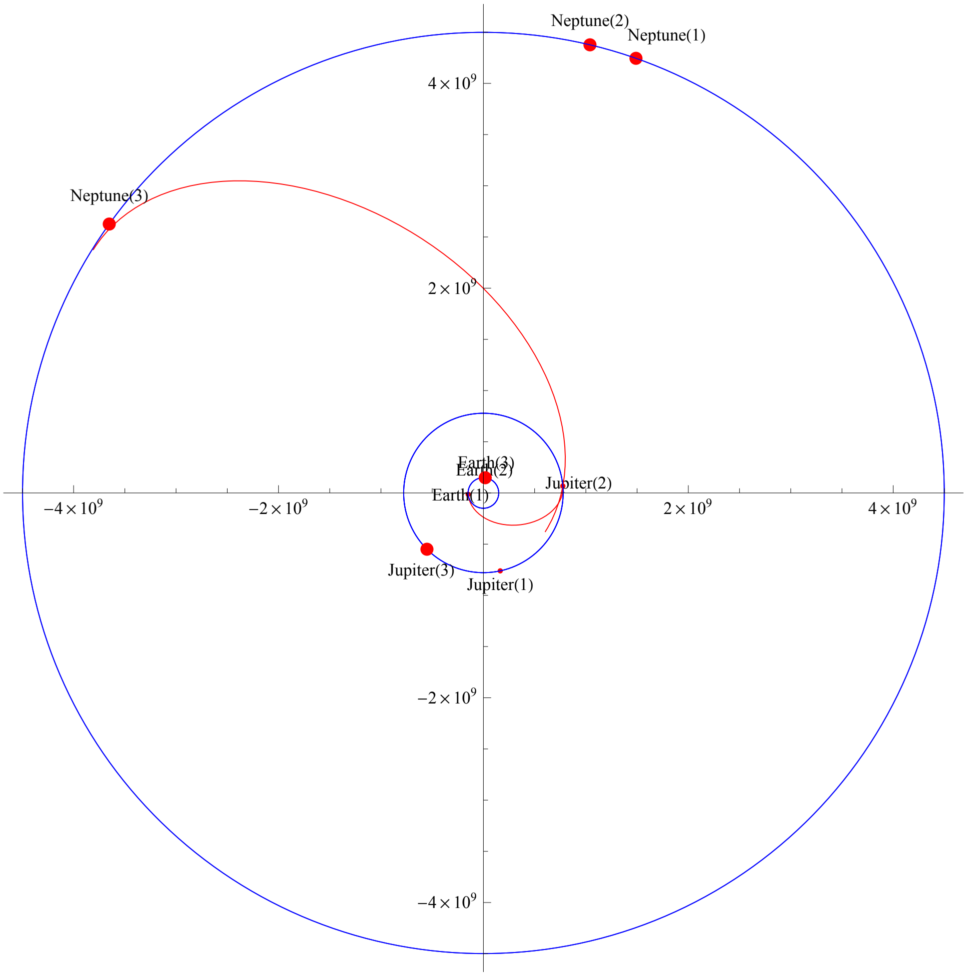
```
{2054, 12, 25}
```

Time on new ellipse in years

```
timeOfFlyOnNewEllipse / (60 * 60 * 24 * 365)
```

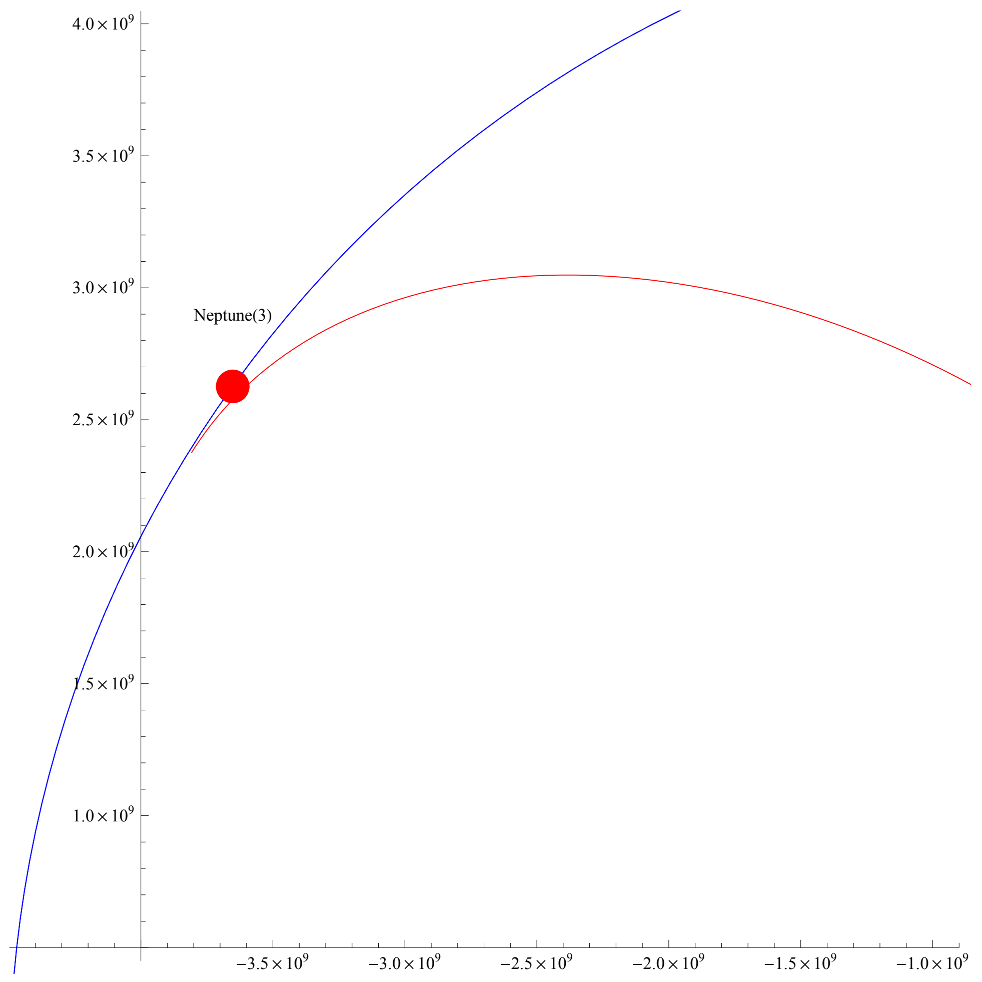
```
31.00936717
```

Find positions of all planets at the final day



Zoom in at the area where the probe enters Neptune SOI

```
Show[p1, p2, p3, p41, p7, p6, ImageSize → 500, PlotRange → {{−rNS, −0.2 rNS}, {0.1 rNS, 0.9 rNS}}]
```



Move probe into final circular orbit around Neptune, final ΔV applied

Now that probe is inside Neptune SOI, we use make a burn out to slow it down into a circular orbit around Neptune. First find the speed the probe is at when it enters Neptune SOI using the ellipse equation (km/sec). Simulation stops when probe is just inside Neptune SOI. Let the altitude above Neptune be 1000 KM as the final parking orbit. The probe arrive on tangential approach to Neptune, hence the speed at apogee is

$$v0 = \sqrt{\mu\text{Sun}\left(\frac{2}{\text{raJN}} - \frac{1}{\text{aJN}}\right)}$$

```
2.846226578
```

- **Find the required speed the probe in a circular orbit around Neptune (KM/sec) assuming 1000 km altitude above the surface**

$$v1=\sqrt{\mu N\left(\frac{1.}{rN+1000}\right)}$$

```
16.28962869
```

- **Find ΔV needed (km)**

$$delV2=v1-v0$$

```
13.44340211
```

Find total ΔV needed for the whole interplanetary trip and compare to if Hohmann transfer was used all the way from Earth to Neptune

$$delV=\mathrm{Abs}[delV1]+\mathrm{Abs}[delV2]$$

```
19.7111595
```

- **The above is ΔV using fly-by Jupiter. Now lets find ΔV assuming Hohmann transfer from Earth to Neptune. First find *a* for this new ellipse (km)**

$$aDirect=\frac{rES+rNS}{2}$$

$$2.324001823\times10^{9}$$

- **Find V_p needed (km/sec)**

$$vp=\sqrt{\mu Sun\left(\frac{2}{rES}-\frac{1}{aDirect}\right)}$$

```
41.43658381
```

Find needed V_∞ to escape Earth

$$\text{vInf} = \text{vp} - \sqrt{\frac{\mu\text{Sun}}{\text{rES}}}$$

```
11.65328093
```

Find r_{bo} the burn out radius

```
rbo = rE + 300 (*300 KM is altitude*)
```

```
6678
```

Find V_{bo} the burn out speed using the energy equation (km/sec)

```
Clear[vbo];
```

$$\text{eq} = \frac{\text{vbo}^2}{2} - \frac{\mu\text{E}}{\text{rbo}} == \frac{\text{vInf}^2}{2} - \frac{\mu\text{E}}{\text{eSOI}};$$

```
vbo = First@Select[vbo /. NSolve[eq, vbo], # > 0 &]
```

```
15.94720179
```

Find ΔV_1 needed to escape earth

$$\text{delV1Direct} = \text{Abs}\left[\text{vbo} - \sqrt{\frac{\mu\text{E}}{\text{rbo}}}\right]$$

```
8.22136659
```

Now find V_a at the apogee at Neptune end of the ellipse (km/sec)

$$\text{va} = \sqrt{\mu\text{Sun}\left(\frac{2}{\text{rNS}} - \frac{1}{\text{aDirect}}\right)}$$

```
1.378004117
```

Find the needed circular speed around Neptune (using SOI since that is what was used above)

$$\text{v3} = \text{N}@\sqrt{\frac{\mu\text{N}}{\text{rN} + 1000}}$$

```
16.28962869
```

Find ΔV2 needed at Neptune

```
delV2Direct = v3 - va
```

```
14.91162457
```

Find total delV

```
Abs[delV1Direct] + Abs[delV2Direct]
```

```
23.13299116
```

Therefore, when using flyby, total ΔV was 19.71 km/sec, and using direct Hohmann transfer, total ΔV is 23.13 The saving is about 3.4 km/sec.

Find the time to travel from Earth to Neptune if direct Hohmann transfer was made

The time in this case is half the period of the Hohmann transfer ellipse, which can be found as follows

$$\text{time} = \text{Pi}\sqrt{\frac{\text{aDirect}^3}{\mu\text{Sun}}};$$

```
time/(60*60*24*365) (*years*)
```

```
30.63814738
```

Appendix

Simulation program source code

```
In[93]:=
(*NotebookDelete[Cells[EvaluationNotebook[],GeneratedCell→True]];*)
Manipulate[
  tick;
  Module[
    {xE, yE, xJ, yJ, xN, yN, eq, sol, xcc, ycc, slope, eq1, eq2, debug = False, va, ra, z, delt, rbo, r, g0, now, x0, y0},

    If[(state == "RUN" || state == "STEP" || state == "INITIAL"),

      delt = Which[timeStep == "day", 60*60*24,
          timeStep == "week", 60*60*24*7,
          timeStep == "month", 60*60*24*30,
          timeStep == "year", 60*60*24*365
        ];

      xE = rES Cos[θE]; yE = rES Sin[θE];
      (*xJ=rJS Cos[θJ];yJ=rJS Sin[θJ];*)
      xJ = rJS Cos[θJx]; yJ = rJS Sin[θJx];
      xN = rNS Cos[θN]; yN = rNS Sin[θN];

      date = DaysPlus[date,
          Which[timeStep == "day", 1, timeStep == "week", 7, timeStep == "month", 30, timeStep == "year", 365]];
      now = Grid[{
            {"year", "month", "day"},
            {padIt2[date[[1]], 4], padIt2[date[[2]], 2], padIt2[date[[3]], 3]}
          }, Frame → All];

      If[showStats, g0 =
          Grid[{
              {Grid[{
                    {Style["Timings and angles as simulation runs", Bold], SpanFromLeft},
                    {"θE", "θJ", "θN", "θHohmann", "State", "Phase", "θ Hohmann"},
                    {padIt2[θE*180./Pi, {5, 2}],
                      (*padIt2[θJ*180./Pi,{5,2}],*)
                      padIt2[θJx*180./Pi, {5, 2}],
                      padIt2[θN*180./Pi, {5, 2}],
                      padIt2[θEarthJupitor*180./Pi, {5, 2}],
                      state,
                      padIt2[phase, 1],
                      padIt2[θEarthJupitor*180/Pi, {5, 2}]
                    }
                  }, Frame → All]
              },
              {Grid[{
                    {Style["mean speeds (km/sec)", Bold], SpanFromLeft},
                    {"Earth", "Jupiter", "Neptune", "probe to Neptune"},
                    {padIt2[sE, {4, 2}],
                      padIt2[sJ, {4, 2}],
                      padIt2[sN, {4, 2}],
                      padIt2[nJN * aJN, {4, 2}]
                    }
                  }, Frame → All]
              },
              {Grid[{
                    {Style["Dimenstions data and current probe speed", Bold], SpanFromLeft},
                    {"rES", "rJS", "rNS", "current ProbeSpeed (km/sec)"},
                    {EngineeringForm[rES, 3],
                      EngineeringForm[rJS, 3],
                      EngineeringForm[rNS, 3],
                      padIt2[currentProbeSpeed, {4, 2}]
                    }
                  }, Frame → All]
              },
              {Grid[{
                    {Style["current positions in space", Bold], SpanFromLeft},
                    {"xN", "yN", "x probe", "y probe", "dist. probe to Neptuen"},
                    {EngineeringForm[xN, 3],
                      EngineeringForm[yN, 3],
                      EngineeringForm[x, 3],
                      EngineeringForm[y, 3],
                      EngineeringForm[EuclideanDistance[{xN, yN, 0}, {x, y, 0}], 3]
                    }
                  }, Frame → All]
              },
              {Grid[{
                    {Style["Hohmann transfer from Earth to Jupiter data", Bold], SpanFromLeft},
                    {"a", "rp", "ra", "e", "current f", "current E"},
                    {EngineeringForm[aEJ, 3],
                      EngineeringForm[rES, 3],
                      EngineeringForm[rJS, 3],
                      padIt2[eEJ, {6, 5}],
                      padIt2[currentf * 180 / Pi, {5, 2}],
                      padIt2[currentE * 180 / Pi, {5, 2}]
                    }
                  }, Frame → All]
              },
              {Grid[{
                    {Style["hyperbolic Jupiter flyby", Bold], SpanFromLeft},
                    {"V∞ (km/s)", "e Hyper", "η Hyper (deg)", "γd (degree)", "θturn", "Vd (km/s)"},
                    {padIt2[vInfinityHyperJ, {6, 3}],
                      padIt2[eHyperJ, {8, 7}],
                      padIt2[ηHyperJ * 180 / Pi, {5, 3}],
                      padIt1[γJN * 180 / Pi, {4, 1}],
                      padIt1[θJNoriginal * 180 / Pi, {4, 1}],
                      padIt1[vdJN, {5, 3}]}
                  }, Frame → All]
              },
              {Grid[{
                    {Style["Post fly-by ellipse, Jupiter to Neptune", Bold], SpanFromLeft},
                    {"aJN", "rpJN", "raJN", "eJN (eccentricity)",
                      "f_new (true anamoly)", Style["mean probe speed deg/day", 9]},
                    {EngineeringForm[aJN, 3],
                      EngineeringForm[rpJN, 3],
                      EngineeringForm[raJN, 3],
                      padIt2[eJN, {6, 5}],
                      padIt1[fJNoriginal * 180 / Pi, {4, 1}],
                      padIt1[nJN * 180 / Pi * 60 * 60 * 24, {6, 4}]
                    }
                  }, Frame → All]
              },
              {Grid[{{"current E (spacecraft)",
                      "current f",
                      "nHuhmannToJupiter (deg/day)",
                      "nJN(deg/day)"},
                    {padIt1[currentE * 180 / Pi, {4, 1}],
                      padIt1[currentf * 180 / Pi, {4, 1}],
                      padIt1[nHuhmannToJupiter * 180 / Pi * 60 * 60 * 24, {6, 4}]}
                  }, Frame → All]
              }
            }]
        ];

        g = Grid[{{Graphics[{
                    (*{White,EdgeForm[Directive[Blue]],Disk[{0,0},rNS]},*)
                  {White, Opacity[0], EdgeForm[Directive[Blue]], Disk[{0, 0}, rNS]},
                  {White, Opacity[0], EdgeForm[Directive[Blue]], Disk[{0, 0}, rJS]},
                  {White, Opacity[0], EdgeForm[Directive[Blue]], Disk[{0, 0}, rES]},
                  {Blue, Opacity[.1], Thickness[0.022], EdgeForm[Gray], Circle[{0, 0}, rNS]},

                  Which[phase == 0,
                    {LightBlue, Opacity[.5], EdgeForm[Gray], Disk[{0, 0}, rJS, {θE, θE + θEarthJupiter }]}
                    ,
                    phase == 1,
                    {
                      Clear[currentE];
                      currentE = currentE /. First@

                          Quiet[NSolve[ tPhase1 == Sqrt[aEJ^3/μSun] (currentE - eEJ Sin[currentE]), currentE, Reals]];

                      currentE = Mod[currentE, 2 Pi];
                      currentR = aEJ * (1 - eEJ * Cos[currentE]);
                      x0 = aEJ Cos[currentE];
                      x0 = x0 - (aEJ - rES);
                      y0 = aEJ √(1 - eEJ^2) Sin[currentE];
                      r = RotationMatrix[-initialHohmann];
                      {x, y} = {x0, y0}.r;
                      currentProbeSpeed = √(μSun (2/currentR − 1/aEJ));

                      tPhase1 = tPhase1 + delt;
                      {
                        (*Rotate[{Blue,Disk[{x,y},size]},initialHohmann,{0,0}],*)
                        {Blue, Disk[{x, y}, size / 4]},
                        Rotate[{Red, Circle[{xc, yc}, {aEJ, bEJ}, {0, Pi}]}, initialHohmann, {0, 0}],
                        Rotate[{Red, Dashed, Line[{{−aEJ (1 + eEJ), 0}, {aEJ (1 − eEJ), 0}}]}, initialHohmann, {0, 0}]
                      }
                    },
                    phase == 2,
                    {
                      Clear[currentE];
                      currentE = currentE /. First@
                          Quiet[NSolve[tPhase2 == Sqrt[aJN^3/μSun] (currentE − eJN Sin[currentE]), currentE, Reals]];
                      currentE = Mod[currentE, 2 Pi];
                      currentR = aJN * (1 − eJN * Cos[currentE]);
                      x0 = aJN Cos[currentE];
                      x0 = x0 − (aJN − rJS);
                      y0 = aJN √(1 − eJN^2) Sin[currentE];
                      r = RotationMatrix[−(θJForPhase2)];
                      {x, y} = {x0, y0}.r;

                      currentProbeSpeed = √(μSun (2/currentR − 1/aJN));


                      If[EuclideanDistance[{xN, yN, 0}, {x, y, 0}] < nSOI,
                        state = "STOP"
                      ];

                      tPhase2 = tPhase2 + delt;

                      {Blue, Disk[{x, y}, size / 4]}, (*moving spacecraft*)
                      {Blue, Dashed, Line[{{0, 0}, {x, y}}]}, (*moving spacecraft*)

                      (*rendevouze location Jupiter and earth*)
                      (*{Black,Disk[{rJS Cos[θJForPhase2+fJN],rJS Sin[θJForPhase2+fJN]},size]},*)
                      {
                        (*original Hohmann Jupiter earth*)
                        Rotate[{Red, Circle[{xc, yc}, {aEJ, bEJ}, {0, Pi}]}, initialHohmann, {0, 0}],
                        (*new ellipse post flyby*)
                        Rotate[{Red, Dashed, Circle[{xc2, yc2}, {aJN, bJN}]}, θJForPhase2, {0, 0}]
                      }
                    }
                  ],
                  {Opacity[.4], Red, Disk[{xE, yE}, size]},
                    {Opacity[.4], Red, Disk[{xJ, yJ}, size]},
                    {Red, Disk[{xN, yN}, nSOI]}
                  }
                  , PlotRange → {{−maxX, maxX}, {−maxX, maxX}},
                  If[showStats, ImageSize → 400, ImageSize → 600], Axes → True
                ]
              }}]
    ];


    If[state == "RUN" || state == "STEP",
      t = t + delt;

      θE = Mod[θE + ωE ∗ delt, 2 Pi];
      θJx = Mod[θJx + ωJ ∗ delt, 2 Pi];
      θN = Mod[θN + ωN ∗ delt, 2 Pi];

      Which[phase == 0,
        If[Abs[(Mod[θE + θEarthJupitor, 2 Pi] − Mod[θJx, 2 Pi])] ≤ 5 Degree ,
          If[debug, Print["detected Hohmann lock in,Mod[θE+θEarthJupitor,2 Pi]=",
              Mod[θE + θEarthJupitor, 2 Pi], " Mod[θJx,2 Pi]=", Mod[θJx, 2 Pi]]];
          If[debug, Print["setting phase=1"]];
          phase = 1;
          aEJ = $\frac{rES + rJS}{2}$;
          eEJ = $\frac{rJS - rES}{rES + rJS}$;
          bEJ = aEJ $\sqrt{1 - eEJ^2}$ ;
          lockAngleWithJupiter = Mod[θE + Pi, 2 Pi];
          xf = rES Cos[θE];
          yf = rES Sin[θE];
          If[debug, Print["e Hohmann=", eEJ]];
          nHuhmannToJupiter = $\sqrt{\frac{\mu Sun}{aEJ^3}}$ ;
          initialHohmann = θE;
          xc = −aEJ eEJ;
          yc = 0;
          tPhase1 = 0
        ],
        phase == 1,
        If[Abs[lockAngleWithJupiter − θJx] ≤ Pi / 100,
          phase = 2;
          va = $\sqrt{\mu Sun\left(\frac{2}{rJS} - \frac{1}{aEJ}\right)}$ ; (∗velocity of craft atJupiter entrance∗)
```

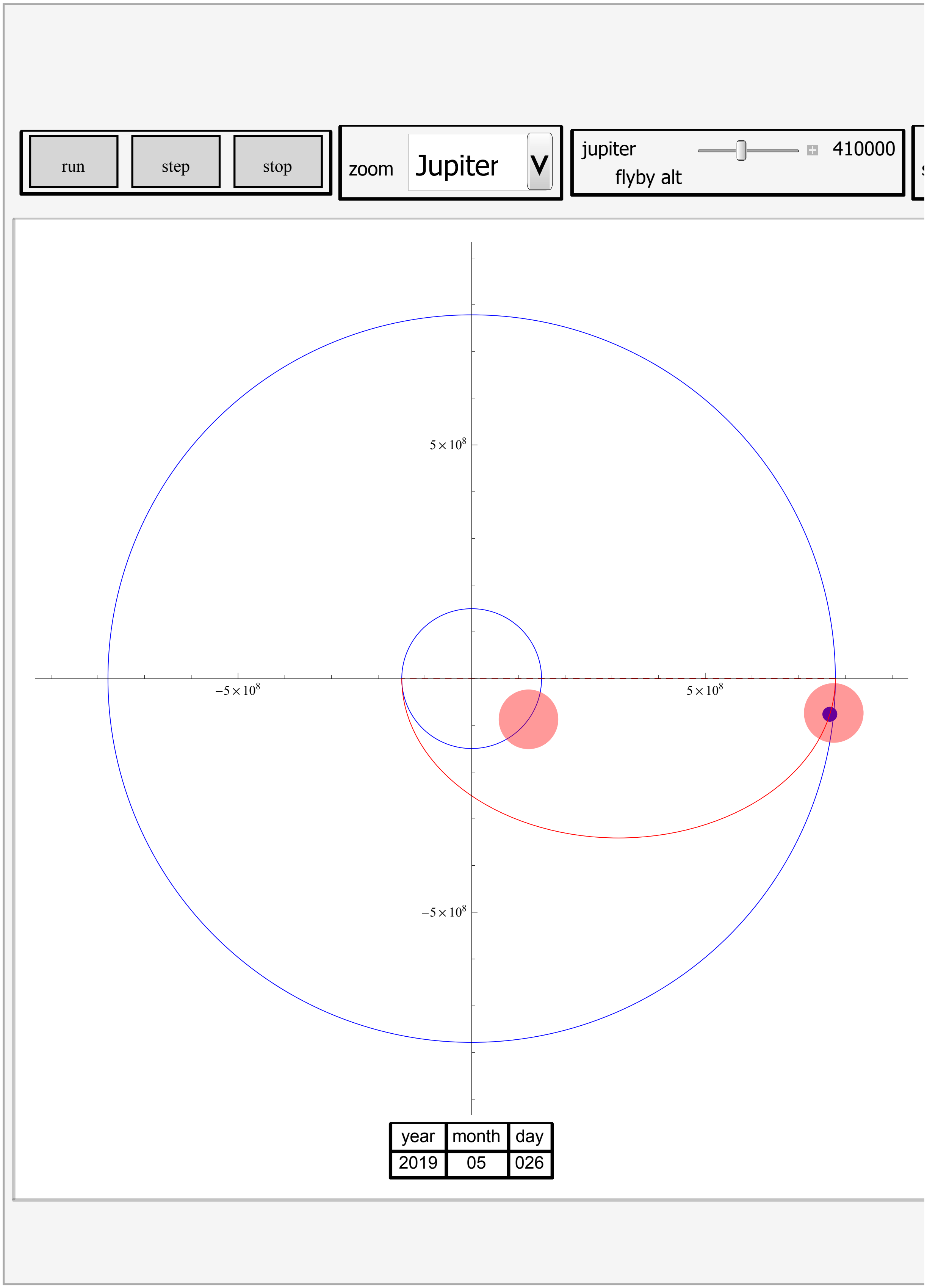
```
vInfinityHyperJ = sJ − va;
rbo = rJ + SOIrb0; (*use this KM*)
Clear[vbo];
eq = vbo^2/2 − μJ/rbo == vInfinityHyperJ^2/2 − μJ/jSOI;
vbo = First@Select[vbo /. NSolve[eq, vbo], # > 0 &];
eHyperJ = 1 + rbo vInfinityHyperJ^2/μJ;
ηHyperJ = ArcCos[−1/eHyperJ];
θJN = 2 ηHyperJ − Pi;
θJNoriginal = θJN;
vdJN = √((sJ^2 + vInfinityHyperJ^2 − 2 sJ vInfinityHyperJ Cos[θJN]));
Clear[z];
sol = Solve[Sin[z] == vInfinityHyperJ Sin[θJN]/vdJN, z];
γJNoriginal = z /. sol;
γJN = z /. First@sol;
Clear[z];
eq = vdJN == √(μSun (2/rJS − 1/z));
aJN = z /. First@NSolve[eq, z];
Clear[z];
eq = Cos[γJN] == √(aJN^2 (1 − z^2)/(rJS (2 aJN − rJS)));
sol = NSolve[eq, z];
sol = z /. sol;
eJN = First@Select[sol, # > 0 &];
Clear[z];
eq = rJS == aJN (1 − eJN^2)/(1 + eJN Cos[z]);
sol = z /. NSolve[eq, z];
fJNoriginal = sol;
fJN = If[γJN ≥ 0, First@Select[sol, # > 0 &], First@Select[sol, # < 0 &]];
rpJN = aJN (1 − eJN);
raJN = aJN (1 + eJN);
bJN = aJN √(1 − eJN^2);
θJForPhase2 = θJx − fJN;
xc2 = −aJN*eJN;
yc2 = 0;
```

```
        nJN = √(μSun/aJN^3) ;

        currentE = ArcCos[(1 - rJS/aJN)/eJN];

        currentf = 0;
        Clear[z];

        eq = Tan[fJN/2] == √((1 + eJN)/(1 - eJN)) Tan[z/2];

        z = z /. First@NSolve[eq, z];
        Clear[tPhase2];

        tPhase2 = tPhase2 /. First@Quiet[NSolve[tPhase2 == Sqrt[aJN^3/μSun] (z - eJN Sin[z]), tPhase2, Reals]];

        If[debug, Print["currentE for JN is =", currentE * 180/Pi]];
      ](*Entered SOI jupiter*)
    ]
  ];

  If[state == "RUN",
    vp[[vpIdx, 1]] = t;
    vp[[vpIdx, 2]] = currentProbeSpeed;
    vpIdx++;
    tick = Not[tick]
  ];

  If[showStats,
    Grid[{{g0}, {g}, {now}}],
    Grid[{{g}, {now}}]
  ]

],
Grid[{
    {Grid[{
          {Button[Text[Style["run", 12]], state = "RUN"; tick = Not[tick], ImageSize → {60, 35}],
            Button[Text[Style["step", 12]], state = "STEP"; tick = Not[tick], ImageSize → {60, 35}],
            Button[Text[Style["stop", 12]], state = "STOP"; tick = Not[tick], ImageSize → {60, 35}]
          }}, Frame → True],
      Grid[{
          {"zoom",
            PopupMenu[Dynamic[zoom, {zoom = #; Which[zoom == "Earth", maxX = 1.2 rES, zoom == "Jupiter",
                      maxX = 1.2 rJS, True, maxX = 1.2 rNS]; tick = Not[tick]} &], {"Earth", "Jupiter", "Neptune"},
              ImageSize → Tiny, ContinuousAction → False]
          }
        }, Frame → True],
      Grid[{{
            "jupiter flyby alt",
            Manipulator[Dynamic[SOIrb0, {SOIrb0 = #; tick = Not[tick]} &],
                    {1000, 10^6, 1000}, ImageSize → Tiny, ContinuousAction → True],
                  Dynamic[padIt2[SOIrb0, 6]]
                }}, Frame → True],
          Grid[{
                {"step",
                  PopupMenu[Dynamic[timeStep, {timeStep = #; tick = Not[tick]} &], {"day", "week", "month", "year"},
                    ImageSize → Tiny, ContinuousAction → False]
                }
              }, Frame → True],
          Grid[{
                {"relax", Spacer[2], Checkbox[Dynamic[showStats, {showStats = #; tick = Not[tick]} &]]}
              }]
        }
    }, Spacings → {0.4, .2}, Alignment → Center
],

{{showStats, False}, None},
(*hyper flyby Jupiter parameters*)
{{eHyperJ, 0}, None},
{{ηHyperJ, 0}, None},
{{vInfinityHyperJ, 0}, None},

{{x, 0}, None},
{{y, 0}, None},

{{maxX, 1.1 rNS}, None},
{{zoom, "Neptune"}, None},
{{SOIrb0, 410 000}, None},
(*{{SOIrb0,395000},None},*)
(*{{SOIrb0,390000},None},*)
{{timeStep, "week"}, None},
{{size, 10 000 rE}, None},
{{tick, False}, None},
{{state, "INITIAL"}, None},
{{phase, 0}, None},
(*set  03/21/2014 θE=180+90, θJ=111.30+90, θN=334.9963+90 *)
(*{{θJ,(111.30 +90)Degree},None},
{{θN,Mod[(334.9963 +90),360]Degree},None},
{{θE,270 Degree},None},*)

(*set 09/21/2014 very close *)
(*{{θJx,(126.2818+90)Degree},None},
{{θN,Mod[(336.1014+90),360]Degree},None},
{{θE,Mod[356.7575+90,360] Degree},None},*)


(*set 10/21/2014 very very close=========> *)
(*{{θJx,(128.6882+90)Degree},None},
{{θN,Mod[(336.2816+90),360]Degree},None},
{{θE,Mod[26.3119+90,360] Degree},None},*)

(*set 10/01/2014 ok, with 340,000 *)
```

```
(*{{θJx,(127.1652+90)Degree},None},
{{θN,Mod[(336.167+90),360]Degree},None},
{{θE,Mod[7.5386+90,360] Degree},None},*)

(*set 03/21/2016 *)
{{θJx, (169  + 90) Degree}, None},
{{θN, Mod[(339 + 90), 360] Degree}, None},
{{θE, 270 Degree}, None},

(*set 03/15/2016 *)
(*{{θJx,(168.5740+90)Degree},None},
{{θN,Mod[(339.3579+90),360]Degree},None},
{{θE,(174.6131+90) Degree},None},*)


(*set 03/30/2016 *)
(*{{θJx,(169.7170+90)Degree},None},
{{θN,Mod[(339.4480+90),360]Degree},None},
{{θE,(189.4931+90) Degree},None},*)

(*set 04/15/2016 *)
(*{{θJx,(170.9351+90)Degree},None},
{{θN,Mod[(339.5442+90),360]Degree},None},
{{θE,(205.2323+90) Degree},None},*)

(*set 05/15/2016 *)
(*{{θJx,(173.2164+90)Degree},None},
{{θN,Mod[(339.7246+90),360]Degree},None},
{{θE,(234.3716+90) Degree},None},*)

(*set 06/15/2016 *)
(*{{θJx,(175.5701+90)Degree},None},
{{θN,Mod[(339.9109+90),360]Degree},None},
{{θE,(264.1023+90) Degree},None},*)

(*set 01/01/2016 *)
(*{{θJx,(162.9198+90)Degree},None},
{{θN,Mod[(338.9131+90),360]Degree},None},
{{θE,(99.7590+90) Degree},None},*)

(*set 03/21/2000 θE=179.5877, θJ=43.4305+90, θN=304.3955+90*)
(*{{θJ,(43.4305+90)Degree},None},
{{θN,Mod[(304.3955+90),360]Degree},None},
{{θE,270 Degree},None},*)


(*set 03/21/2017 *)
(*{{θJ,(196.5839 +90)Degree},None},
{{θN,Mod[(341.5831 +90),360]Degree},None},
{{θE,270 Degree},None},*)

(*set 03/21/2020  OK *)
```

```
 (*{{θJ,Mod[(282.3034+90),360]Degree},None},
 {{θN,Mod[(348.1929 +90),360]Degree},None},
 {{θE,270 Degree},None},*)

 (*{{θJx,Pi/4},None},
 {{θJxx,45},None},*)

 {{t, 0}, None},
 {{tPhase1, 0}, None},
 {{tPhase2, 0}, None},
 {{date, {2016, 03, 21}}, None},
 {{a, 0}, None},
 {{e, 0}, None},
 {{nHuhmannToJupiter, 0}, None},
 {{nJN, 0}, None},
 {{currentE, 0}, None},
 {{currentf, 0}, None},
 {{initialHohmann, 0}, None},
 {{xf, 0}, None},
 {{yf, 0}, None},
 {{xc, 0}, None},
 {{yc, 0}, None},
 {{xc2, 0}, None},
 {{yc2, 0}, None},
 {{currentR, 0}, None},
 {{lockAngleWithJupiter, 0}, None},
 {{lockAngleWithNeputon, 0}, None},
 {{aJN, 0}, None},
 {{raJN, 0}, None},
 {{rpJN, 0}, None},
 {{bJN, 0}, None},
 {{eJN, 0}, None},
 {{fJN, 0}, None},
 {{fJNoriginal, 0}, None},

 {{aEJ, 0}, None},
 {{bEJ, 0}, None},
 {{eEJ, 0}, None},
 {{fEJ, 0}, None},
 {{θJForPhase2, 0}, None},

 {{γJN, 0}, None},
 {{γJNoriginal, 0}, None},
 {{θJN, 0}, None},
 {{θJNoriginal, 0}, None},
 {{vdJN, 0}, None},
 {{currentProbeSpeed, 0}, None},
 {{g, 0}, None},
 {{g0, 0}, None},
{{vp, Table[{0, 0}, {50*365}]}, None},
 {{vpIdx, 1}, None},
 TrackedSymbols :→ {tick},
ControlPlacement → Top,
Initialization :→
  (
    integerStrictPositive = (IntegerQ[#] && # > 0 &);
    integerPositive = (IntegerQ[#] && # ≥ 0 &);
    numericStrictPositive = (Element[#, Reals] && # > 0 &);
    numericPositive = (Element[#, Reals] && # ≥ 0 &);
    numericStrictNegative = (Element[#, Reals] && # < 0 &);
    numericNegative = (Element[#, Reals] && # ≤ 0 &);
    bool = (Element[#, Booleans] &);
    numeric = (Element[#, Reals] &);
    integer = (Element[#, Integers] &);
    (*---------------------------------------------*)
    padIt1[v_?numeric, f_List] := AccountingForm[v,
        f, NumberSigns → {"-", "+"}, NumberPadding → {"0", "0"}, SignPadding → True];
    (*---------------------------------------------*)
    padIt1[v_?numeric, f_Integer] := AccountingForm[Chop[v],
        f, NumberSigns → {"-", "+"}, NumberPadding → {"0", "0"}, SignPadding → True];
    (*---------------------------------------------*)
    padIt2[v_?numeric, f_List] := AccountingForm[v,
        f, NumberSigns → {"", ""}, NumberPadding → {"0", "0"}, SignPadding → True];
    (*---------------------------------------------*)
    padIt2[v_?numeric, f_Integer] := AccountingForm[Chop[v],
        f, NumberSigns → {"", ""}, NumberPadding → {"0", "0"}, SignPadding → True]
  )
]
```

Out[93]=



zoom Jupiter

jupiter flyby alt 410000

year	month	day
2019	05	026