PRELIMINARY TRAJECTORY DESIGN FOR A SOLAR POLAR OBSERVATORY USING SEP AND MULTIPLE GRAVITY ASSISTS

Luca Corpaccioli[1a], Ron Noomen[1b], Stijn De Smet[2a], Jeffrey Parker[2b], and Jonathan Herman[2c]

[1a]*MSc student, Delft University of Technology. Kluyverweg 1 - Delft - The Netherlands. +31-6-2717-4897, l.corpaccioli@student.tudelft.nl*

[1b]*Assistant Professor, Delft University of Technology. Kluyverweg 1 - Delft - The Netherlands. +31-15-27-85377, R.Noomen@tudelft.nl*

[2a]*Graduate Research Assistant, University of Colorado. 429 UCB - Boulder - USA. +1-303-492-3105, Stijn.DeSmet@colorado.edu*

[2b]*Assistant Professor, University of Colorado. 429 UCB - Boulder - USA. +1-303-492-3105, parkerjs@colorado.edu*

[2c]*Graduate Research Assistant, University of Colorado. 429 UCB - Boulder - USA. +1-303-492-3105, jon.herman@colorado.edu*

Abstract: *Satellite solar observatories have always been of central importance to heliophysics; while there have been numerous such missions, the solar poles have been extremely under-observed. This paper proposes to use low-thrust as well as multiple gravity assists to reach the enormous energies required obtain high heliocentric inclinations. Two novel methods are used to provide initial guesses to a low-thrust trajectory optimizer. Results show that the complete trajectory would take between 3-4 years, although scientific observations can start already after 2 years. This assumes starting masses of 1000-2000 kg, and payload masses of 500-1300 kg. The best inclinations reached are 55-65 degrees to the ecliptic, while maintaining a perihelion and aphelion of 0.5 and 1 AU respectively. More research still needs to be performed to attempt to reach a global optimum. Further investigation is suggested to extend the mission for further objectives, such as lowering perihelion, or further cranking inclination.*

Keywords: *Solar Observatory, Multiple Gravity Assists, Low-thrust, Ecliptic Inclination, Solar Electric Propulsion (SEP), Trajectory Optimization, GASP*

1. Introduction

The motivation for this research stems from the great importance of heliophysics to modern society, having strong implications on many fields, including climatology, meteorology and satellite design. Over the last 20 years there have been over 15 missions whose primary objective was to observe the Sun, and its interaction with the Earth's environment. While they have all collected a wealth of information, in most cases these missions have operated in Earth orbit. Few exceptions include ESA's SOHO, at the Sun-Earth L1 point, NASA's STEREO, operating in a 1 AU circular orbit, as well as other missions under development, such as ESA's SoIO and NASA's Solar Probe Plus, both planning to fly more ambitious orbits. In the case of the former, reaching a perihelion of 0.28 degrees and an inclination of 34 degrees to the ecliptic, and of the latter, grazing the sun's surface at 0.034 AU.

A spacecraft which deserves special mention is NASA's Ulysses: launched in 1990, it is the only satellite which managed to directly observe the Sun's poles. This severe lack of data with respect

to the solar poles is in part explained by the extremely high energies needed to reach such orbits, requiring to drastically increase the inclination with respect to the Sun's equator. Ulysses could only achieve this by performing a gravity assist around Jupiter, allowing it to reach an inclination of more than 80 degrees; unfortunately this also implied its orbital period was extremely long, and therefore it only observed the poles three times (latest was in 2008) at which point the Radioisotope Thermoelectric Generator (RTG) power and hydrazine levels were too low for a fourth active pass.

After more than 20 years since Ulysses launched, the Sun's poles still remain an extremely attractive goal. Flying a mission which not only is capable of observing them, but with a significantly higher frequency, will provide invaluable data for heliophysics.

ESA's SoIO partially plans to achieve this, however it only has a high propulsion system on board [1]. Significantly better results can be achieved by using low-thrust, specifically solar electric propulsion (SEP). This paper will design and describe a trajectory to maximize inclination with respect to the ecliptic plane by utilizing low-thrust propulsion as well as gravity assists, expanding existing studies on the matter [2, 3].

Lastly, before diving in the methodology, it is important to illustrate some important equations regarding the use of a fly-by for inclination gain. The details of the derivation are not shown here, however using simple trigonometric relations equation 1 shows how the fly-by parameters relate to inclination in heliocentric coordinates.

$$i = \arctan\left(\frac{V_\infty \sin(\theta)}{V_p - V_\infty \cos(\theta)}\right) \tag{1}$$

Here, θ is the angle between the V_∞ vector and the planet's velocity vector, discounting any components in the radial direction of the planet's position vector. It should be added that here we assume circular planet orbits in the ecliptic plane. This isn't the case (as planets' orbits always have a small eccentricity and inclination), however it serves as a very good initial guess for an optimizer.

The equation shows that, all other things being equal, higher V_∞ will produce higher inclinations. Differentiating equation 1 with respect to θ and equating the expression to zero, the condition to maximize inclination is shown in equation 2.

$$\theta_{opt} = \arccos\left(\frac{V_\infty}{V_p}\right) \tag{2}$$

Furthermore, it should be stated that on a VNC frame centered along a planet's velocity axis (with the x axis along the planet's velocity, the y axis along the radius vector of the planet, and z-axis to complete the right hand rule) there should be no V_∞ component in y-direction.

2. Methodology

2.1. Low-Thrust Modeling tool: BOLTT

Low-thrust trajectories are constructed using the BOLTT tool [5, 6, 7] (Boulder Optimization of Low-Thrust Trajectories), which is modeled after the Sims-Flanigan method [8]. The method itself is not the focus of the paper; here, only a rough outline of the algorithm is presented. If interested, the reader can find relevant sources in the bibliography.

This approach models low-thrust by applying many small ΔVs at regular intervals along the trajectory. The method is illustrated in figure 1.

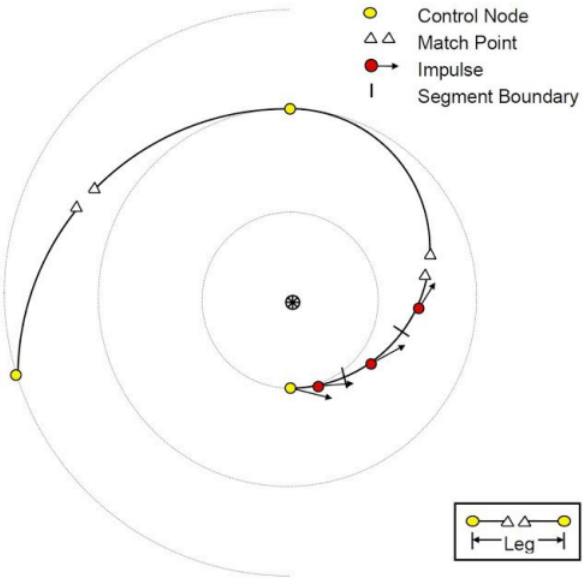

Figure 1. Sims-Flanigan model diagram [8].

A general outline is described as follows:

- The trajectory is defined by setting various control nodes. These represent the planets along the sequence (however this is not strictly necessary, and a control node can be defined anywhere). There needs to be at at least a departure and arrival node, plus any number of additional ones in between (referred to here as fly-by nodes).
- Each fly-by node is set to have a forward and a backward leg, the departure node is set to have only a forward leg, and the arrival has only a backward leg.
- Each leg in turn is subdivided into segments at equal time intervals. Each segment has an impulsive ΔV at its center. This is constrained by the current mass of the spacecraft, duration of the segment and the available thrust (which in turn is dependent on the available power).
- The trajectory is propagated along said legs using a chosen force model (in this case the classic two-body equations) and the ΔVs. The end of each leg is known as a match point.
- The trajectory is then constrained to be continuous. This implies that the spacecraft's velocity before and after fly-by planets respects non-powered gravity assist equations and that the match points are joined up sequentially.

- Further constraints can also be applied. These can include, for instance, maximum power available (dependent on heliocentric distance if using SEP), or limited departure V_∞, depending on a selected launcher.
- Lastly, the optimizer SNOPT [9] is used to optimize the trajectory with all previously set constraints.
- A modification is made to the classic Sims-Flanigan method. This is to deal with the fact that the spacecraft's mission does not end at the arrival planet, but rather it performs a last fly-by and enters a heliocentric orbit. Thus, three more variables are added to the state vector. These describe the V_∞ of the spacecraft as it exits the fly-by, and thus also determine the final heliocentric orbit reached.

This method is regarded as quite robust and fast, however it still remains a local optimizer. It needs good initial guesses for some elements of the state vector. As will be seen in the remaining sections of the paper, the focus will be on providing initial guesses for V_∞ at each planetary fly-by, as well as fly-by and departure dates.

2.2. Structure of the trajectory model

This section describes the overall logic behind the design of the trajectory. As was seen in section 1. the maximum achievable inclination depends on the V_∞ magnitude. Therefore, this mission is comprised of two phases.

- V_∞ increase phase. An EVEV trajectory is used to maximize V_∞ magnitude at Venus arrival.
- Inclination cranking phase. Multiple resonant Venus fly-bys rotate the arrival V_∞ vector to its target position to maximize inclination.

Some more details should be mentioned with regards to each phase. Firstly, it is important to justify the EVEV sequence. For this mission, Earth and Venus are the only planets considered, as Mercury and Mars are too small, and Jupiter is too far. Secondly one must consider that, at least in the ballistic case, V_∞ magnitude can only increase if the next planet in the sequence is different than the previous planet. Performing a fly-by at the same planet only has the effect or rotating the V_∞, and not to increase it. From this fact one can conclude that the planet before Venus must be Earth, eliminating an eventual EVEVV sequence. It is also assumed that a launcher can reach any departure V_∞, this eliminates the need for an EEVEV. Potential remaining options are EVVEV or EVEEV. After a brief investigation it was found that these do not add significant advantages to an EVEV trajectory, which is of course significantly faster. Of course more encounters can be added, such as EVVEEV, however these are not considered here. To provide an initial condition for this phase, a trajectory is formed using ballistic arcs connected at planet periapses by a ΔV. The method itself is explained in detail in subsection 2.3.. This method provides an initial guess for V_∞ at each planet in the sequence, as well as dates of departure, fly-bys, and arrival. These will then be fed into BOLTT to optimize a low-thrust trajectory, with arrival V_∞ magnitude as the cost function to be maximized.

In the second phase, the target V_∞ vector for best inclination must be reached, from whatever value it was at Venus arrival. Unfortunately, since V_∞ is quite high, one fly-by will not be sufficient to achieve the desired rotation. Multiple fly-bys of the same planet are necessary, and can only occur if resonant orbits are chosen (or quasi-resonant if the spacecraft has propulsive capabilities). Note that it would be extremely difficult to reach high inclination orbits by "bouncing" between Venus and Earth. This is because both planets' orbits are roughly in the ecliptic, thus arcs which connect them will also lie in this plane for most of the time. Unfortunately, the same initial guess tool for the first phase can no longer be used, since the latter uses a Lambert solver to compute arcs; this solver can not work in the case of resonant transfers. A new method is developed, described in detail in subsection 2.4., which uses parametric circles on a sphere to describe the rotations of the V_∞ vector. Again, this method models the spacecraft as having no propulsive capabilities. Like before, an initial guess for V_∞ vectors at each fly-by is obtained, together with the fly-by dates. These are fed into BOLTT once more, this time using inclination as the cost function to be maximized.

2.3. Modified gravity assist space pruning - GASP

An initial guess for the first phase is made using a modified GASP method. The original method [10] is freely available to those interested. However, it presents some disadvantages; this paper will describe a modification of such method to attempt to address such limitations. Perhaps the biggest disadvantage is that it necessitates computing large sections of the search space, and then identifying promising fly-by date areas from this. The new proposed method is able to identify these areas by following the contours only, thus evaluating the cost function significantly less times.

The method is divided in three parts: departure constraining, gravity-assist constraining, and backward constraining. It should be noted at this point that the original GASP uses a breaking maneuver constraint too, however in this case the objective is not to enter orbit around the final planet in the sequence.

The user is required to select some inputs, seen below:

- Planet sequence: in this mission the sequence in question is Earth-Venus-Earth-Venus (EVEV).
- Departure date range and times of flight range.
- Maximum V_∞ for earth departure.
- Minimum radius of fly-by for each planet (this is the planet's radius plus a certain tolerance to account for atmospheres and other features).
- Maximum ΔV at fly-by periapsis.

Departure constraining In the original GASP, a full pork-chop plot is constructed using the departure dates and times of flight to the first planet, with required V_∞. For those unfamiliar, a pork chop plot gives the required V_∞ magnitude (or sometimes C_3 energy) to go from one planet to another as a function of departure date and time of flight. A Lambert solver is used to generate the required Keplerian arcs, and using the starting planet's ephemeris, one can compute the required V_∞ magnitude at departure (the V_∞ at arrival is not computed for departure constraining). More

information can be found at [11]. Usually, one then selects only those combinations of departure date and time of flight which satisfy the V_∞ requirement (usually dictated by the launcher available). In this alteration of GASP, rather than computing the entire 2D search space, a smarter contour-following algorithm is employed which is capable of determining the exact edges of the feasible areas, thus giving certain "islands" inside which there are permissible transfers. This algorithm is outlined in figure 2. In this figure we see a sample island, as the dotted black irregular shape. Note that this is not known a priori, and the objective is to find such a shape.

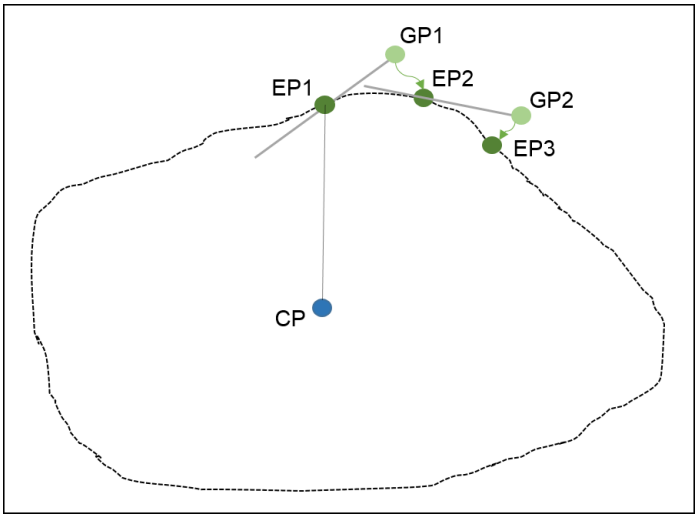

Figure 2. Island following algorithm for departure.

The first step is to find various local minima of the pork chop plot. The Lambert function that would otherwise be used to compute the V_∞ is used as a cost function, for an optimizer. Such a function has the form $V_\infty = F(t_{dep}, T)$, where t_{dep} is the departure date and T is the time of flight.

Next, an extremely coarse grid over the departure dates and times of flight combinations (in the order of 10-20 total points) is made, and a local optimizer is run on each of these, to find the minimum V_∞. It should be stated that the objective function is basically smooth everywhere, and thus a gradient-based optimizer will do. This returns a local minimum for each combination of the grid.

As can be expected, many are duplicates, and others are above the V_∞ threshold, these are eliminated. In this case, let us assume that only one minimum is left, labelled CP in figure 2.

Next, a numerical solver is used which, starting from CP, finds the point along a vertical line which corresponds to the V_∞ threshold, this is shown as the edge point $EP1$. At this point, the derivative with respect to both variables is determined numerically. This yields a vector of the steepest direction.

Perpendicular to this vector is the direction of zero derivative. A step is made along this path to arrive to the guess point, labeled $GP1$ in figure 2. Lastly, the numerical solver is run once again, starting from this guess point, to find a new point which corresponds to the V_∞ threshhold, in this case edge point $EP2$. This process can be repeated again: a new guess $GP2$ is found using the

derivative at $EP2$, and this is used to solve for a new edge point $EP3$. When the algorithm returns sufficiently close to where it started, the shape is considered closed, and the algorithm stops. If there are more than one island, the user selects the preferred one. In the end, rather than having a pork chop plot, one remains with a graph resembling figure 3.

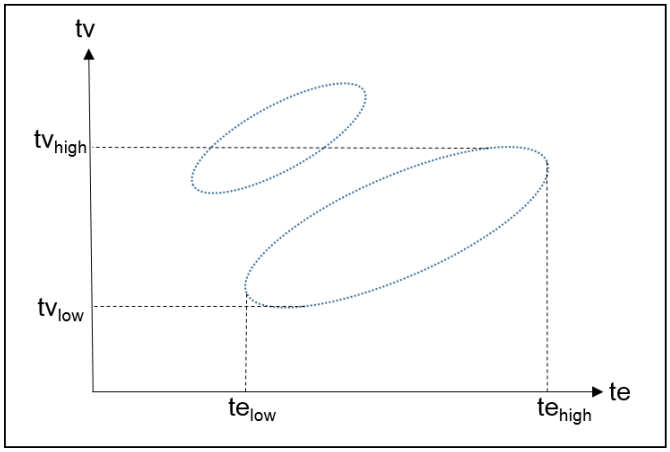

Figure 3. Departure constraining in modified GASP algorithm.

Here, te is the departure date from Earth, and tv is the arrival date at Venus. The axis are not to scale. Two things should be noted: firstly, the remainder of the pork chop plot is unknown, and secondly, the variable on the y-axis is now arrival date rather than time of flight. As can be seen, two islands have been identified in this example, and only the larger one has been selected. From this, the bounds on the Earth departure and Venus arrival have been marked. All other combinations of dates are unfeasible, and can thus be eliminated, this is the departure constraining.

Fly-by constraining Fly-by constraining is slightly more complex than the departure, however the basics are still the same: to find islands of feasible dates inside which the fly-by is attainable (a fly-by is considered attainable if its radius is above the defined threshold and the required ΔV is below the threshold). This time however, the variable being plotted is no longer V_∞, but rather it is a 1 or a 0, to signify whether the given combination produces a viable fly-by or not. Note that to compute the fly-by radius and ΔV, one needs an incoming and an outgoing Keplerian arc to be joined at the planet's periapse. This in turn requires three dates; in this case, the Earth departure, the Venus fly-by, and a new Earth arrival.

As before, it is possible to evaluate the entire search space, or use a smarter contour-following algorithm. First, let us examine how points in this new search space are computed; for this, consider figure 4.

The new plot would be the left portion of the figure, where on the x-axis we have the fly-by date at Venus, and on the y-axis we have the arrival date at Earth (which is in turn, another fly-by). Note that this is labeled te2, since this is the second Earth encounter. The right portion of the figure is the previous graph (in this case, it was the departure plot from Earth). In the new plot (on the left), the Venus fly-by dates are on the x-axis, whereas on the old plot, these dates are on the y-axis. First, it

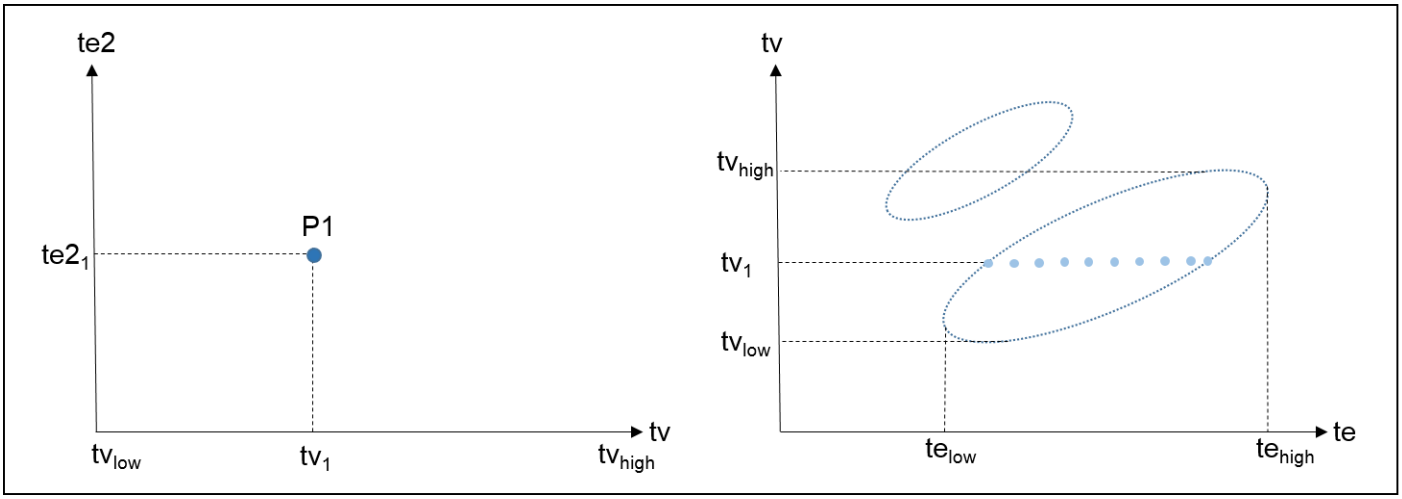

Figure 4. Evaluation of point on fly-by plot in modified GASP algorithm.

is important to state that on the new x-axis of the fly-by plot, the bounds are limited thanks to the departure constraining.

Now suppose one wishes to investigate point $P1$, which occurs at Venus fly-by date tv_1 and ends at Earth on date $te2_1$. In order to determine if this combination produces a feasible fly-by, one needs to also look at the previous island. Here, the Venus fly-by date tv_1 is also marked. At this location, a sample of Earth departure dates is taken, these are displayed as the lightly marked dots. A loop goes over all these Earth departure dates, and computes fly-by parameters using also tv_1 and $te2_1$. If both fly-by parameters are within their threshold, for any Earth departure date, $P1$ is given a 1, otherwise it becomes a 0. If the reader wishes to know how the two fly-by parameters are computed given two Keplerian arcs, [11] provides valuable information on the matter. It is a simple use of classic Keplerian mechanics and the patched conics model.

As before, a system must be employed to search for the island boundaries rather than having to compute the entire search space. Since this new function is no longer continuous, any gradient-based solver will fail. The solution is to employ a bisection method to find the border the feasibility. This in turn requires to have two sample points, one inside the island, and one outside. For a theoretical background on the bisection method, refer to [12]. This system is best exemplified using figure 5

The algorithm may appear complicated, however it is easily explained. Again, let us consider a sample island shown as the irregular dotted shape (this shape is unknown, and the objective is to find the outline). Rather than going around the entire shape like with the departure constraining, the edges are split into the top border and bottom. The two algorithms are identical, so only the top border version will be explained.

As before, the first step is to run a coarse grid search of the entire space, so as to find points which are inside the islands. Points which return 0s are eliminated, any other points are kept (since it is impossible to know ahead of time the general shapes of the island, one must assume each point with

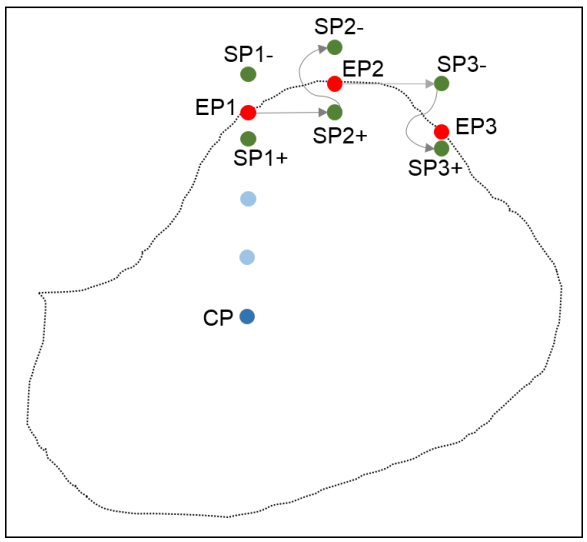

Figure 5. Island following algorithm for fly-by plot.

a 1 is its own island). In this case, let us assume that only one point has been found, labeled as the center point *CP* in figure 5.

Next, regular vertical steps are taken until the algorithm finds a 0. It is thus known that the true edge must lie between the last two sample points, labeled $SP1-$ and $SP1+$, where the $-$ implies it is outside the island, and vice versa for the other. A simple bisection algorithm is used between the two to find the edge point, labeled $EP1$. Next, a right horizontal step is taken from this edge point, and it is evaluated. This is shown by the arrow. As can be seen, this new sample point $SP2+$ returns a 1, which implies the other sample point is found above. Therefore vertical upward steps are taken (shown by the appropriate arrow) until the algorithm finds a point outside the island, shown as $SP2-$. Having a point inside and outside the island, the bisection method is employed once more to find the new edge point, labeled $EP2$ in the figure.

The next step is the same as the previous: a horizontal side step is taken from $EP2$, shown again by the arrow. This time however the found sample point is $SP3-$ whose value is 0; this implies the other sample point $SP3+$ must lie vertically downwards (shown by the arrow). The new edge point $EP3$ is therefore found as before.

Note that as the top edge is found, the bottom edge must also be determined. Once the algorithm can no longer find a solution, or when the top solution is lower than the bottom, the side right of the center point has been found. The left side is found using the same algorithm, taking horizontal left steps.

This simple algorithm, whilst having some disadvantages, proves to be rather robust. Once islands of feasible fly-bys are found, one ends up with a diagram resembling that on the left part of figure 6. At this point, the entire process repeats again, with the next fly-by, until one arrives at the last planet in the sequence; this is the fly-by constraining.

Backward Constraining Backward constraining is not as complex as the previous, and does not require to draw new islands. It is best illustrated with an example; for simplicity, let us suppose that the trajectory is just EVE, and therefore only the first two graphs are necessary: the departure island, and the Venus fly-by island. An example of these two is shown in figure 6, where the fly-by diagram is on the left, and the departure diagram is on the right.

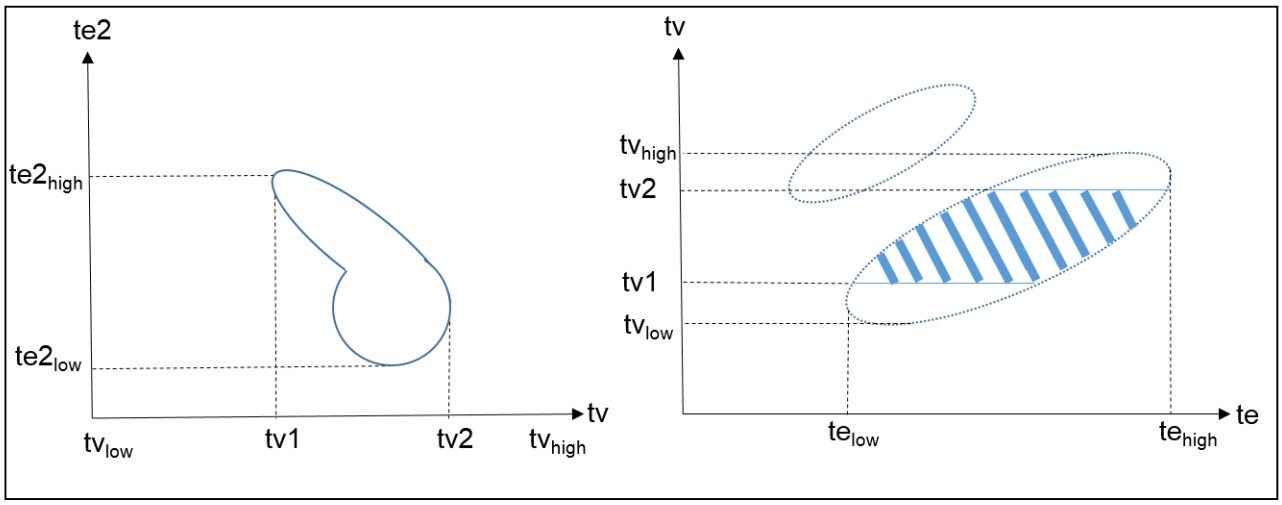

Figure 6. Backward constraining in modified GASP algorithm.

One can thus see that the island for the fly-by diagram is a bit odd. This usually the case: these kinds of islands tend to be rather irregular and unpredictable. The dates of arrival at Earth have been bounded by $te2_{high}$ and $te2_{low}$. At the same time, that island shows that the dates of the Venus fly-by have been further restricted to the range tv1 to tv2. This restriction must be propagated backwards (thus the name backward constraining). These bounds are placed on the right portion of the figure, where one can see that only the dashed area is left for available Venus dates. For the full trajectory, one must first apply fly-by constraints until the last planet in the sequence, and then progressively backward constrain through until the beginning.

2.4. V_∞ sphere

As explained in subsection 2.2. this is the method by which an initial guess of the inclination cranking portion of the trajectory is obtained. Again, these are generated assuming pure Keplerian motion, disregarding any propulsive capabilities. As will be seen later, the effect of low-thrust can be partially accounted for later.

The inputs for this algorithm is simply the V_∞ vector which arrives at the last Venus encounter. Then, using a series of resonant fly-bys, the objective is to rotate this to the optimal location for maximum inclination (explained in section 1.), in the least amount of time. As already explained, a Lambert solver tends to perform poorly around a resonant transfer, and in fact completely breaks down in case of a perfect resonance. Thus, this new method is developed.

First, a coordinate change is employed, from the classic Sun-centered J2000 frame used until now,

to one that is Venus centered, and aligned with the planet's velocity. Given a planet's state, the transformation matrix from this frame to the ecliptic J2000 is defined by equation 3.

$$T = [\hat{V} \quad \hat{N} \quad \hat{C}] \tag{3}$$

where the various parameters are defined in equation 4.

$$\hat{V} = \frac{\vec{V}_{pl}}{\left|\vec{V}_{pl}\right|} \quad \hat{N} = \frac{\vec{R}_{pl} \times \vec{V}_{pl}}{\left|\vec{R}_{pl} \times \vec{V}_{pl}\right|} \quad \hat{C} = \hat{V} \times \hat{N} \tag{4}$$

Now the x-axis is aligned with the planet's velocity. A further 90 degrees rotation is made along the x-axis, so as to roughly point the z-axis in the direction of the planet's "pole", and thus maintain convention. Figure 7 summarizes the current definitions.

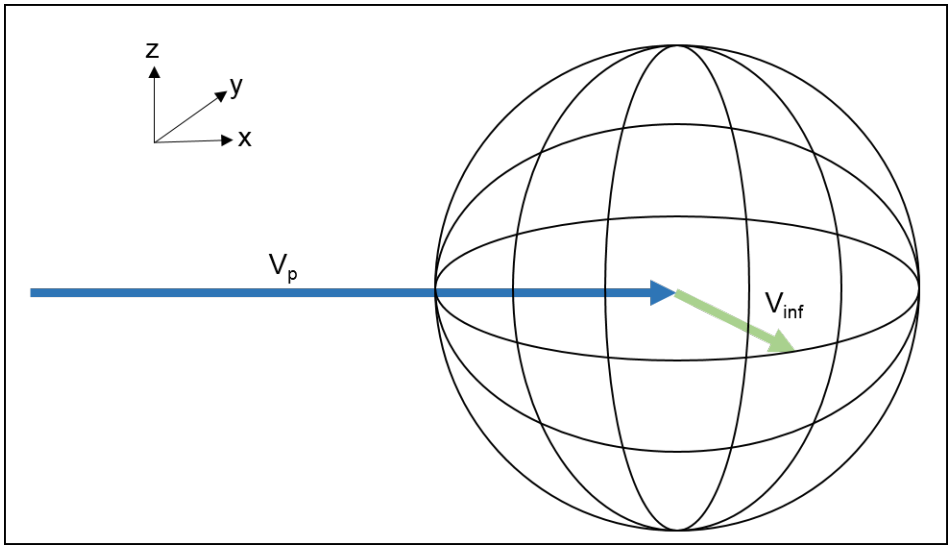

Figure 7. Definition of V_∞ sphere.

As can be seen, the axis system is now aligned with the planet's velocity, in a VNC frame. One also recognizes the familiar V_p vector: the planet's velocity, and the V_∞ vector at its tip; their summation will give the heliocentric velocity parameters of the spacecraft. More importantly, the concept of the V_∞ sphere has been introduced: a sphere whose radius is the magnitude of V_∞, which represents all the possible trajectories the spacecraft can take thanks to fly-bys. It should be noted that if the spacecraft could fly arbitrarily close to the planet's center, any point on the sphere could be achieved with one fly-by, otherwise, the rotation of the vector is limited by the familiar gravity-assist relation

given in equation 5.

$$\alpha = 2\arcsin\left(\frac{1}{1+\frac{r|V_\infty|^2}{\mu_{pl}}}\right); \quad r > R_{pl} + h_{min} \tag{5}$$

In order to reach any point on the sphere, one must perform multiple resonant fly-bys, to gradually rotate the V_∞ vector from its starting direction to the desired one. However, it is worth noting that this interpretation immediately gives valuable information on the kinds of orbits that can be achieved. If the V_∞ vector is pointed along the x-axis, it is equivalent to adding a maneuver along the velocity vector, which serves to increase or decrease semi-major axis. If V_∞ points along the y-axis, this is equivalent to burning along the radial direction, which serves to increase eccentricity. Finally, if the V_∞ is aligned along the z-axis, it is equivalent to burning out of plane, and thus serves to increase inclination.

The next step is representing the various resonances and fly-by locations on the sphere. These are circles lying on the sphere, which will be represented parametrically. In general, a circle in three dimensional space is represented by equation 6.

$$\vec{P} = R\cos(t)\,\vec{u} + R\sin(t)\,\vec{n}\times\vec{u} + \vec{c} \tag{6}$$

Here, P is the three-dimensional coordinates of the circle points, u is a unit vector from the center of the circle to any point on the circumference, R is the radius, n is a unit vector perpendicular to the plane of the circle, and c is a vector pointing to the center of the circle form the origin. From this, one then defines a circle which determines all the points that can be reached within one fly-by. Figure 8 shows how this can be derived.

This is a cut of the V_∞ sphere through which the V_∞ vector passes. Thus the parameters for a parametric circle can easily be defined as in equation 7

$$R = |V_\infty|\sin(\alpha) \quad \vec{c} = V_\infty\cos(\alpha) \quad \vec{n} = \frac{1}{|V_\infty|}V_\infty \quad \vec{u} = V_\infty \times \left(V_\infty + \begin{bmatrix}1\\1\\0\end{bmatrix}\right) \tag{7}$$

Note that there are multiple ways to define $\vec{u}$, the one presented here ensures there are never any singularities. Anything on the sphere on or inside this circle can be reached with one fly-by.

Next, let us take a closer look at the definition of resonant transfers. The only requirement for a resonant transfer is that the period spacecraft and planet is related by a ratio of two (usually small) integers. This in turn puts a requirement on the semi-major axis of the orbit, which then places a requirement on the x component of the V_∞ in figure 7. Since this is the only requirement, there

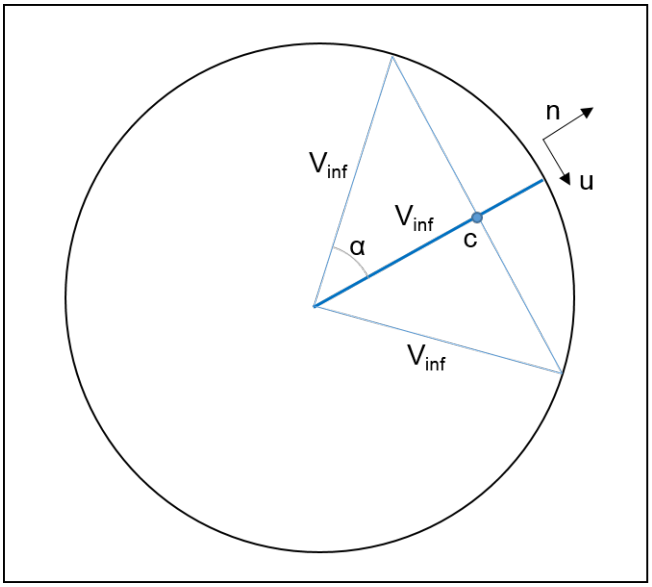

Figure 8. Derivation of parametric circle parameters for fly-by circle.

exists an infinite number of V_∞ vectors which satisfy this; these lie on a circle on the sphere with constant x. These circles can be defined with the help of figure 9.

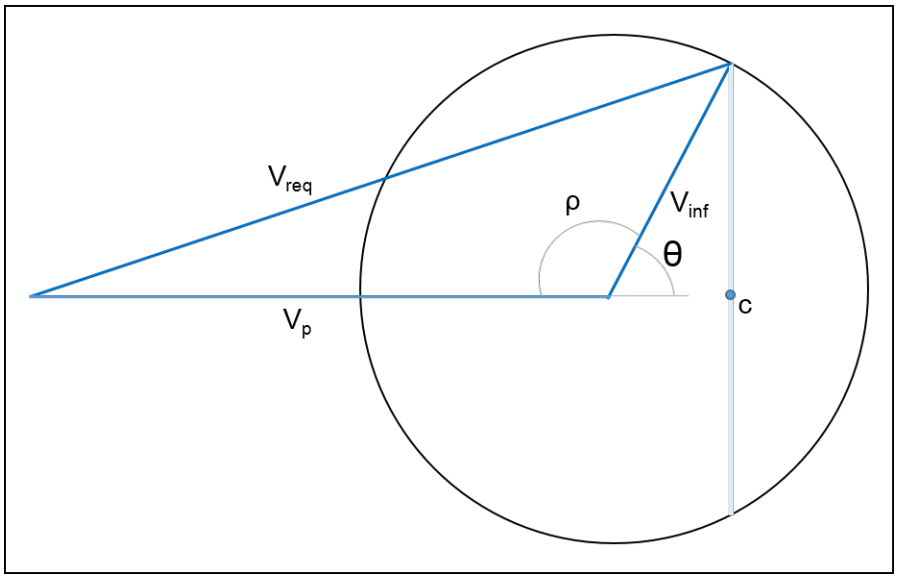

Figure 9. Derivation of parametric circle for resonance circles.

In this case, V_{req} is the required velocity magnitude to attain the desired resonance. This is simply calculated knowing standard Keplerian equations. Given a resonant period of the desired orbit, the semi-major axis is computed as $a = \left(\left(\frac{T}{2\pi}\right)^2 \mu_{sun}\right)^{\frac{1}{3}}$, and the required velocity is calculated as $V_{req} = \sqrt{\mu_{sun}\left(\frac{2}{r_{pl}} - \frac{1}{a}\right)}$. Simple trigonometric relations can be then used to compute all the parameters described in equation 6. One can see that $\rho = \arccos\left(\frac{V_{req}^2 - V_\infty^2 - V_p^2}{-2V_\infty V_p}\right)$, and $\theta = \pi - \theta$.

Thus one can see that equation 8 describes the circle parameters for the resonant circles:

$$R = |V_\infty| \sin(\theta) \quad \vec{c} = \begin{bmatrix} |V_\infty| \cos(\theta) \\ 0 \\ 0 \end{bmatrix} \quad \vec{n} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \vec{u} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{8}$$

Thus at this point, figure 10 describes how the circles relate to each other on the V_∞ sphere.

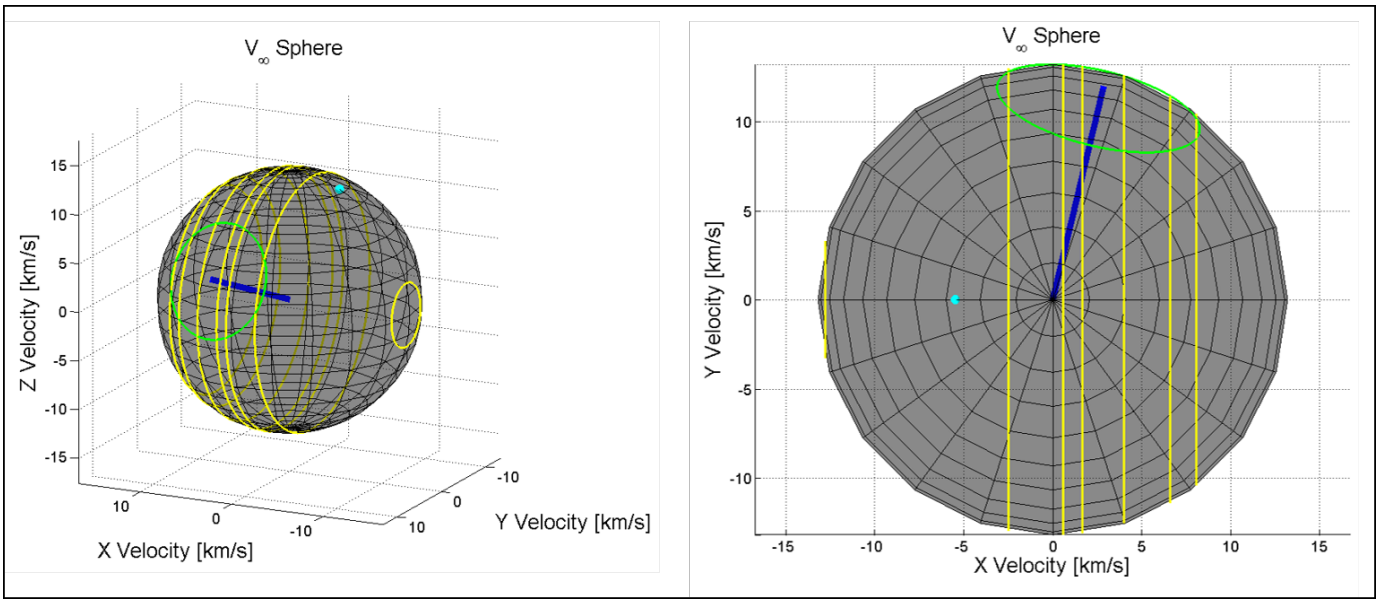

Figure 10. Complete diagram of V_∞ spheres. Left is 3D view, right is top down view.

The left part of figure is a 3D view, and the right part is a top-down view. Firstly, it should be noted that only the sphere is drawn, and the V_p vector from figure 7 is left out. The blue line stemming from the center of the sphere is the V_∞ vector, and the green circle is the fly-by circle. The yellow circles are the resonant transfers, such that any point there achieves a resonance. Referring to the right portion of the figure, the resonances represented are as follows, starting from the positive X axis and moving towards the negative: 4:1, 3:1, 2:1, 3:2, 4:3, 1:1, 1:2. Where, for instance, a 3:2 resonance implies the planet orbits 3 times while the spacecraft orbits twice. Lastly, the cyan dot is the optimal location for the V_∞ for highest inclination. In the case shown, note that almost all yellow circles are inside the green, which implies that within one fly-by the spacecraft can almost achieve any resonance.

The next step is to determine how each resonance connects to the others, given the current fly-by circle. This can be done by looking at the right portion of figure 10. As an be seen, the resonances appear as straight lines with constant x value. Thus one can determine the value on the fly-by circle that has the highest and lowest x value; if a resonance has an x value within these two, it implies it is reachable.

One starts by taking the fly-by circle equation 6, isolating the x component, differentiating with

respect to the parameter t and equating to zero. The result are the parameter t values which yield the maximum and minimum x, expressed by equation 9.

$$t_1 = \arctan\left(\frac{n_y u_z - n_z u_y}{u_x}\right) \qquad t_2 = t_1 + \pi \tag{9}$$

Here the subscripts define which component of the vector to use. The highest and lowest xs are easily computed by putting these ts back in the original expression. Given these expressions, a simple loop is run which places a fictitious V_∞ on each of the resonances to see how a fly-by can reach the others. From this, a table can be constructed which relates resonances to each other, as shown in figure 11.

A	**4:1**	**3:1**	**2:1**	**3:2**	**4:3**	**1:1**	**B**
4:1	3:1	2:1	3:2	4:3	1:1	4:3	4:3
3:1	2:1	3:2	4:3	1:1	3:2	3:2	1:1
	A	4:1	3:1	2:1	2:1	B	
		A	4:1	3:1	B		

Figure 11. Resonant grid, illustrating how resonances connect to each other with one fly-by.

Here, the value in bold is the resonance in question, and the values underneath are all the resonances which can be reached in a single fly-by. A is the starting V_∞ and B is the target vector. Let us follow an example of how the two can be connected

- The starting vector A can connect to a 4:1 and a 3:1 resonance. Let's say the spacecraft hops onto the 3:1 resonance. The spacecraft takes three full planet years to complete this.
- Once it has done so, one must look at the 3:1 column, which in turn can connect to a 2:1, 3:2, 4:1 or back to A.
- The spacecraft goes onto a 3:2 resonance, which takes another 3 years to complete. Again, we look at the 3:2 column to see which other resonances are available to it.
- The 1:1 is now selected. This takes a year to complete, after which the spacecraft can connect to the final vector B, as illustrated by the 1:1 column.

Thus the full trjaectory is a 3:1 $\rightarrow$ 3:2 $\rightarrow$ 1:1, which takes a total of 7 years. There many possibilities for transfers, thus, a simple discrete dynamic programming (DDP) [13] algorithm is implemented to determine the optimal path from A to B

The last step, once an optimal sequence has been found, is determine the values of the V_∞ vectors as they go from resonance to resonance. Referring back to figure 10, this previous step has determined which yellow circles should be used, however one must still compute where along them the V_∞ should be placed. A simple logic can be derived, by observing that the target vector will always

be along the x axis, with zero y component. Hence if the V_∞ has a y component, the program will choose the point along the resonant circle which minimizes it, keeping in mind this point must always be on or inside the green fly-by circle. Graphically, this translates to rotating the V_∞ vector such that it decreases in y and gains in z, until it reaches the y=0 longitude line on the sphere.

To do this, it is important to determine the intersects of the fly-by circle with the resonant circle in question. Again, this is easily computed by equating the two parametric equations. The derivation is further simplified by ignoring the z dimension (as shown in the right portion of figure 10). If this is done, the fly-by circle becomes a 2D ellipse, and the resonant circle becomes a straight line. The expressions which describe the two intersections are shown in equation 10. These parameter values, once plugged back in the 3D circle equations, give the coordinates of the intersections.

$$t_1 = 2\arctan\left(\frac{\sqrt{B^2-K^2+V^2}-B}{K-V}\right) \qquad t_2 = 2\arctan\left(\frac{-\sqrt{B^2-K^2+V^2}-B}{K-V}\right) \tag{10}$$

where $B = R\,(n_y u_z - n_z u_y)$, $V = u_x R$ and $K = c_x - c'_x$. All parameters with an apostrophe refer to the values of the resonant circle, whereas all others refer to those of the fly-by circle. Again, substituting these into the original equations gives the coordinates of the intersection points. Having these intersects, it is possible to distinguish the following scenarios

- The two intersects have y values of opposite sign. This implies the fly-by circle goes over the point y=0. Hence the V_∞ vector is placed on this point.
- The two intersects have y values of equal sign. This implies that y must be reduced as much as possible. This can be achieved by selecting one of the two intersects (whichever has a smaller y value).

Given this simple law, one can easily find all the exact points which lead to the target vector

Before concluding this section, an important exception should be mentioned. Since the DDP scheme determines reachable resonances by the x component of the resonant circles as well as the target vector, it is possible to have a situation where the fly-by circle's x values are within the target vector, but the latter still cannot be reached. As previously mentioned, the algorithm always attempts to minimize the y value, however if there aren't enough fly-bys, this cannot be brought down enough to reach the target. In these cases, the solution is to perform more fly-bys, each time rotating the V_∞ vector upwards, reducing y and increasing z. The fastest way of doing this is to lock on to the 1:1 resonance, since this is the fastest. Unfortunately this DDP cannot handle this, therefore if the algorithm runs into this problem, rather than attempting to reach the target vector, it first reaches the 1:1 resonance, and then uses it to gradually reduce the y component, as described in the bullet points above. An example of this is shown in figure 17. Here, the 1:1 resonance is immediately reached, however it still cannot connect to the required direction. Thus multiple 1:1 fly-bys are made (shown in pink) to gradually approach this.

3. Results

The result section is organized as follows: first the GASP analysis plots will be shown, after which the optimized low-thrust transfer results will be displayed, as computed by BOLTT. Next, the V_∞ sphere results are presented. A 3-dimensional view of the spheres plus the movement along the resonant circles is shown. The last subsection shows the optimized low-thrust trajectory, again created by BOLTT.

3.1. GASP approach

Here, the graphical results for the GASP analysis are shown. Before delving in the details, it should be mentioned that departure refers to date 9500 in MJD2000. This corresponds to the 4th of January of 2026 (dates here are written as DD/MM/YYYY).

Figure 12 shows the pork chop plot for the Earth to Venus departure, color coded with V_∞ magnitude (as described by the scale on the right, in km/s). This is the required excess velocity to depart from Earth and reach Venus. The red markers show the results of the island algorithm.

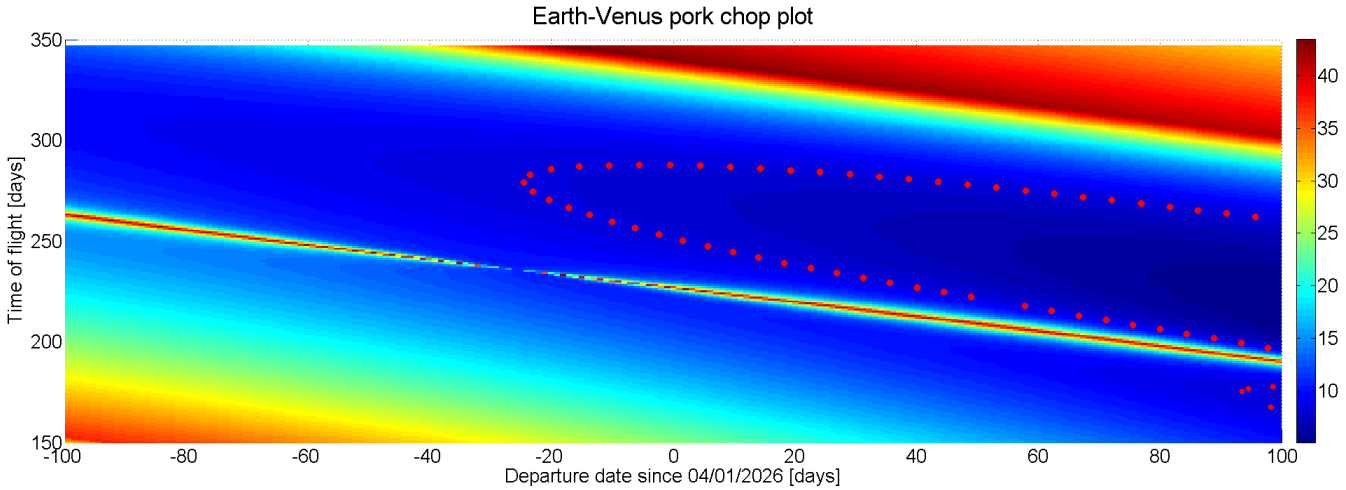

Figure 12. EV pork-chop plot, with feasible V_∞ island, plotted against V_∞ in km/s

The island algorithm does not have to stop at the pre-set boundaries of the pork chop plot, and can keep on exploring areas of the search space that were previously not included in the bounds. However in this case the islands were set to stay within these bounds. Here, the island boundary is set at 8.5 km/s. Table 1 shows the results for the speed of the algorithm.

Table 1. Resolution and computation time comparison for departure plot.

	Complete plot	Island algorithm
Time [s]	12.3	2.8
Resolution [days]	1×0.75	$<0.1\times0.1$

Resolution is defined slightly different for the methods. In the case of the complete plot, it simply describes the size of the grid, whereas for the island algorithm, it means the point is considered

converged once it is within 0.1 days in both directions to the ideal $V_\infty = 8.5$ km/s location. The larger island is chosen to proceed with the next fly-by diagram. In order to move to the fly-by constraining, some slight adjustments are made. The departure island is converted from departure date-time of flight, to departure date-arrival date. Then, the range of arrival dates serves as the new x-axis for the Venus fly-by diagram. Figure 13 gives this diagram from Venus back to Earth.

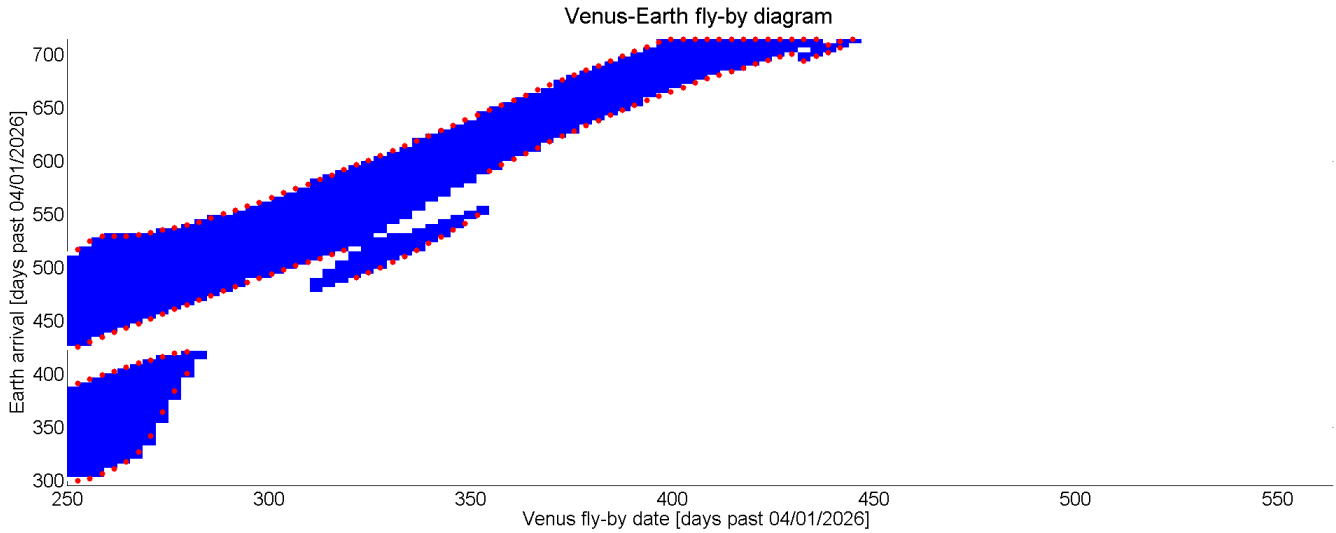

Figure 13. VE fly-by plot in blue, with identified fly-by islands in red.

The blue areas represent locations of viable transfers (defined here as a fly-by periapsis above an altitude of 300 km for both Earth and Venus, and having a ΔV impulse of less than 1.5 km/s). The red markers are the contours found by the island algorithm. Table 2 gives a summary of the time required to obtain this.

Table 2. Resolution and computation time comparison for VE fly-by.

	Complete plot	Island algorithm
Time [s]	40.1	22.2
Resolution [days]	3.18×4.2	0.5

Again, the resolution should be appropriately defined. In the case of the complete plot, it defines the size of the grid. In the case of the island algorithm, it is the maximum length the bisection line can have to reach convergence. It should be noted that in order to have the same resolution as the island algorithm, the complete plot would take 2166 seconds.

For this diagram, the smaller island is chosen, and a new fly-by plot is generated which brings the spacecraft to the last planet in the sequence. The new diagram is shown in figure 14. Note that the larger island would have also been an option, however, for the purposes of this paper, only one island is investigated.

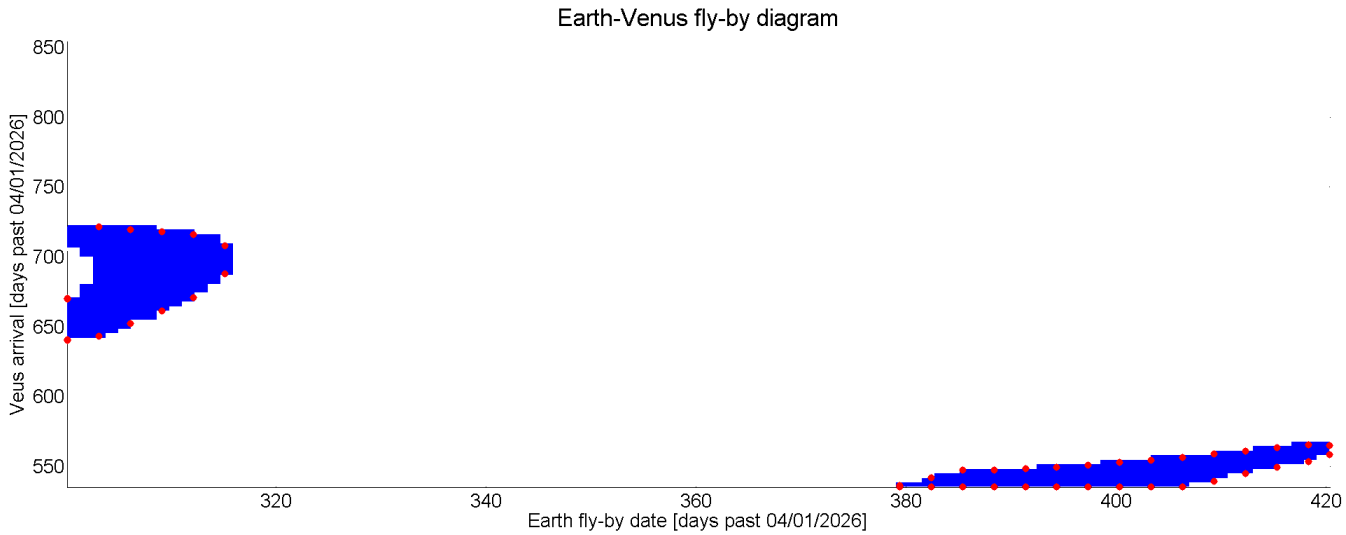

Figure 14. EV fly-by plot in blue, with identified fly-by islands in red.

Table 3 again gives the details of the computation time taken.

Table 3. Resolution and computation time comparison for EV fly-by.

	Complete plot	Island algorithm
Time [s]	38.441	30.150
Resolution [days]	1.21×3.23	0.5

This time, it appears the computation times are similar. This is both due to the nature of the islands, as well as the worse resolution; if the complete plot were computed to a resolution of 0.5 days the computation would take 604 seconds. From these, the leftmost island is chosen. Again, the other island could have been chosen, and would certainly constitute an interesting point of further study.

Having the complete transfer, the backward constraining is applied, and the final time boundaries are shown in table 4.

Table 4. Final boundaries on fly-by dates as computed by modified GASP.

Stage	Lower boundary (days since 04/01/2026)	Upper boundary (days since 04/01/2026)
Earth departure	-5	45
Venus fly-by	235	270
Earth fly-by	290	320
Venus arrival	630	740

Note that a small margin of about 5-10 days has been applied on each of these. In order to obtain an initial guess for the V_∞ vectors at each planet in the sequence, a random combination of dates is chosen within the bounds, and a Lambert solver is used to connect each planet. The subsequent V_∞ vectors are computed and used as the initial guess.

3.2. Low-thrust EVEV trajectories

At this point, an initial guess on the V_∞ and ranges on the dates of the planetary encounters are known. There are however other initial values which must be set in order to run the full low-thrust simulation: the starting mass, the dry mass, the power available and the utilized launcher. These are all treated as independent variables. The settings of these are shown in table 5.

Table 5. Selected values for initial conditions of EVEV simulation.

Variable	Selected value
Launch mass, m_0 [kg]	2000, 1500, 1000
Final mass, m_f [kg]	TBD
Power [W]	$3m_0$, $2m_0$, m_0
Launcher	Atlas 5-551, Delta IV M+(4,2), Falcon 9

Three things should be noted. Firstly, the final mass does not have pre-defined values, instead it is set during the runs, to ensure one obtains a proper distribution of results. Secondly, the SEP power depends on the initial mass, such that the Watts correspond to the mass in a linear relationship. Lastly, the launchers are selected not so much based on popularity or availability but rather on how well they span the launch curve diagram. Other values from table 5 have been chosen based on ESA's upcoming solar observatory, SoIO [1].

The final mass represents a bound below which the optimizer cannot go. This has to be implemented, due to the cost function used. The optimizer only seeks to maximize V_∞ magnitude, meaning it will always expel as much propellant as possible to reach this objective, reducing the mass to its lowest possible value. Since there is still another phase to the trajectory (the Venus resonance portion) the minimum final mass is treated as a variable for the first phase of the trajectory.

Assuming one has 3 values to test for each of the variables as seen in the table, theoretically one should expect 81 results (3^4), however many combinations are unfeasible, and are therefore discarded (for instance using the weakest launcher with the heaviest mass). A total of 57 combinations are left. An example of a feasible trajectory is shown in figure 15, for a spacecraft weighing 2000 kg, a final mass of 1500 kg (excluding an extra 120kg of power subsystem), a power level of 4000 W and launched with an Atlas 5. In this example, the spacecraft reached a final V_∞ of 19.452 km/s at Venus arrival. The view in figure 15 is in the classic heliocentric J2000 frame seen from "above".

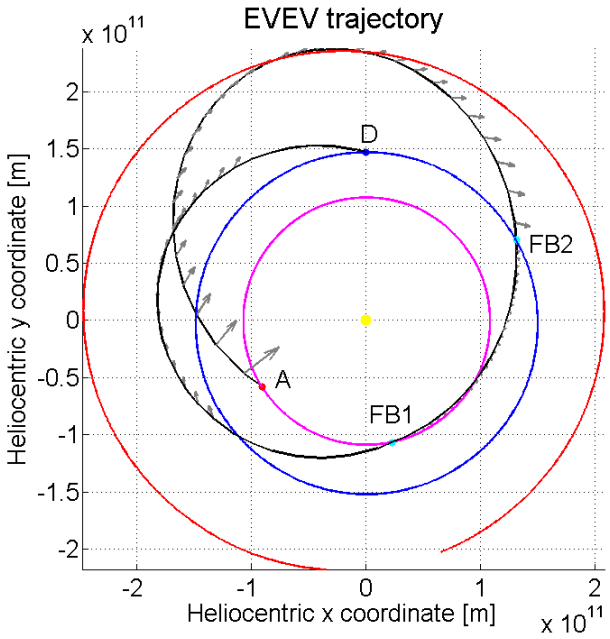

Figure 15. EVEV trajectory, m_0=2000 kg, P=4000 W, m_{pay}=1500 kg, launched with Atlas 5.

In the figure, the planets' orbits are represented as red, blue and pink for Mars, Earth and Venus respectively. The trajectory is shown in black, with the gray arrows representing the discretized low-thrust impulsive maneuvers. Lastly, the nodes are labelled as D, FB1, FB2, and A corresponding to departure, fly-by 1, fly-by 2, and arrival. As one might expect, this is the general shape of this phase of the trajectory, so other such diagrams will not be shown. More details regarding the trajectory are shown in figure 16.

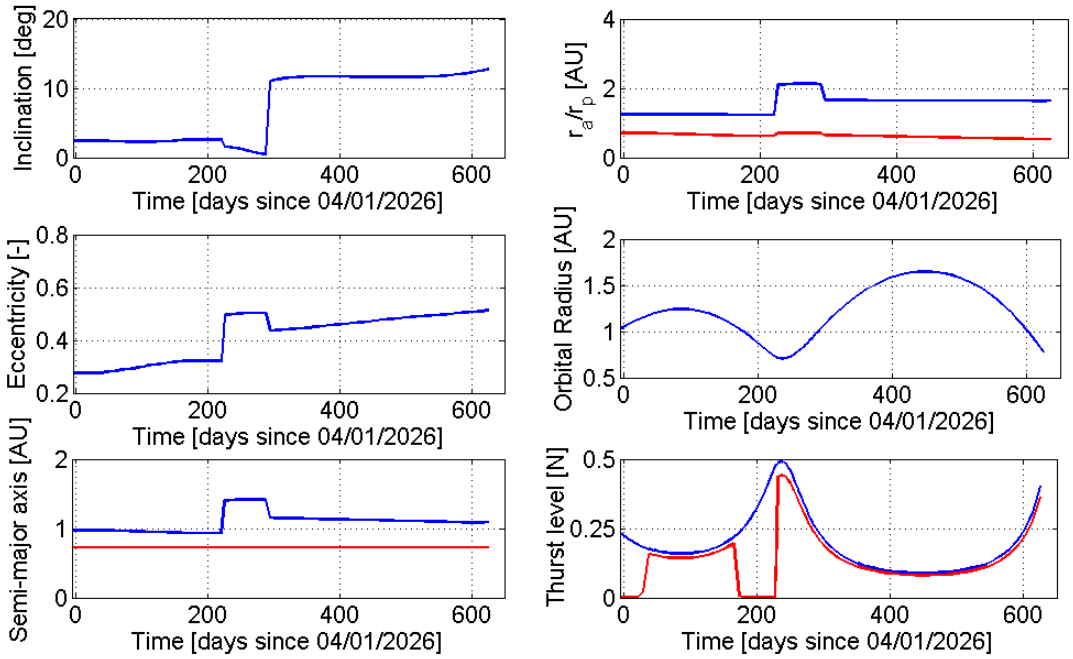

Figure 16. EVEV data, m_0=2000 kg, P=4000 W, m_{pay}=1500 kg, launched with Atlas 5.

A few of these plots require some additional explanation. The semi-major axis plot also contains a red line to signify the semi-major axis needed for a 1:1 resonance with Venus. The r_a/r_p plot is the apoapsis periapsis plot, shown in blue and red respectively. Lastly the thrust plot shows the theoretical maximum achievable thrust in blue, and the actual thrust level in red (note that a 90% duty cycle is implemented, such that the theoretical maximum can never actually be reached).

3.3. V_∞ sphere results

For the resonance portion of the trajectory, only the example shown above is used, rather than all other 57. Using the V_∞ sphere method, the optimal fly-by sequence is shown in figure 17.

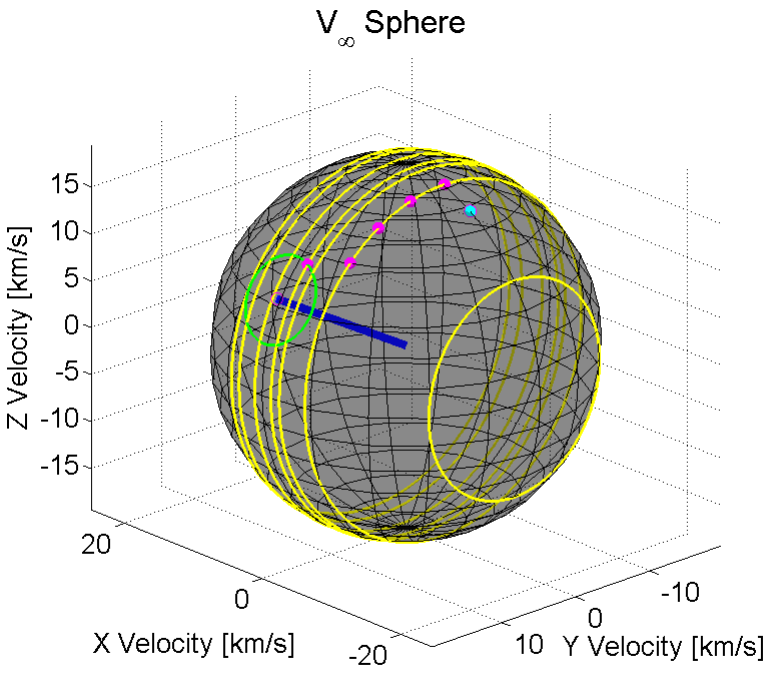

Figure 17. V_∞ sphere results for resonant Venus portion. Fly-by radius increase factor 1.

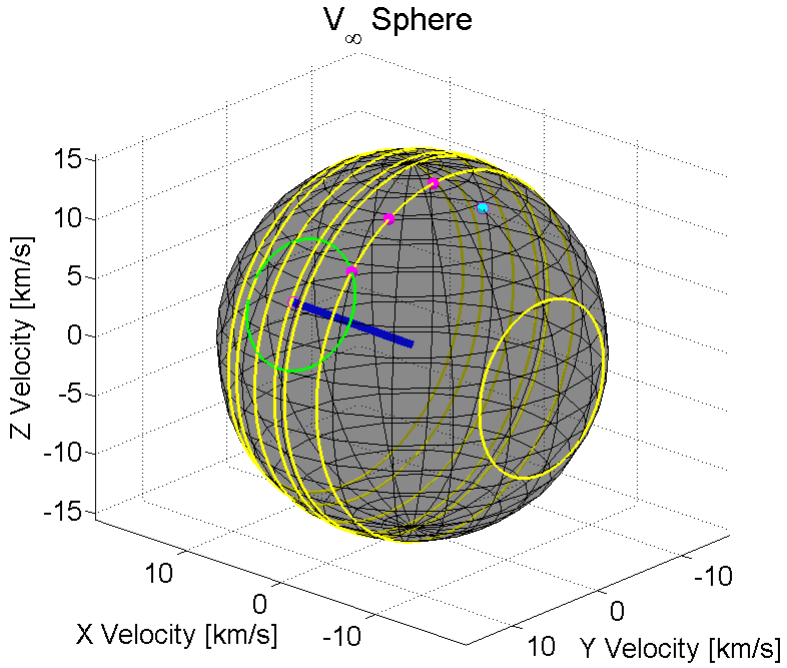

Figure 18. V_∞ sphere results for resonant Venus portion. Fly-by radius increase factor 1.25.

Again, all definitions from figure 7 apply. The pink marks represent the optimal V_∞ path through the various fly-bys. The sequence is a simple 3:2→1:1, after which the vector rotates upwards along the last resonance until it can reach the target vector. As mentioned above, low-thrust can

open up the design space; this is simulated by increasing the radius of the fly-by circle. Here, it is enlarged by a factor 1.25, and the diagram is shown in figure 18. The 1.25 factor was chosen after some experimental data. It was seen that the low-thrust system starts having difficulties constructing feasible trajectories after a radius increase factor of 1.5, thus 1.25 is chosen to remain within reasonable bounds.

It is immediately obvious that the target vector can be reached in significantly less fly-bys, furthermore, the spacecraft can now reach the 1:1 resonance immediately, without having to waste 3 Venus years approaching it in the 3:2 resonance. Using these two as initial conditions, BOLTT can now be implemented to simulate the resonant trajectory. For now, it is assumed that the payload is of 500 kg, table 6 gives a summary of results for higher payload masses.

3.4. Venus resonance trajectory

First, the input from figure 17 is used. Figure 19 shows this resonant portion of the trajectory, from a side view, to better display the repetition periods.

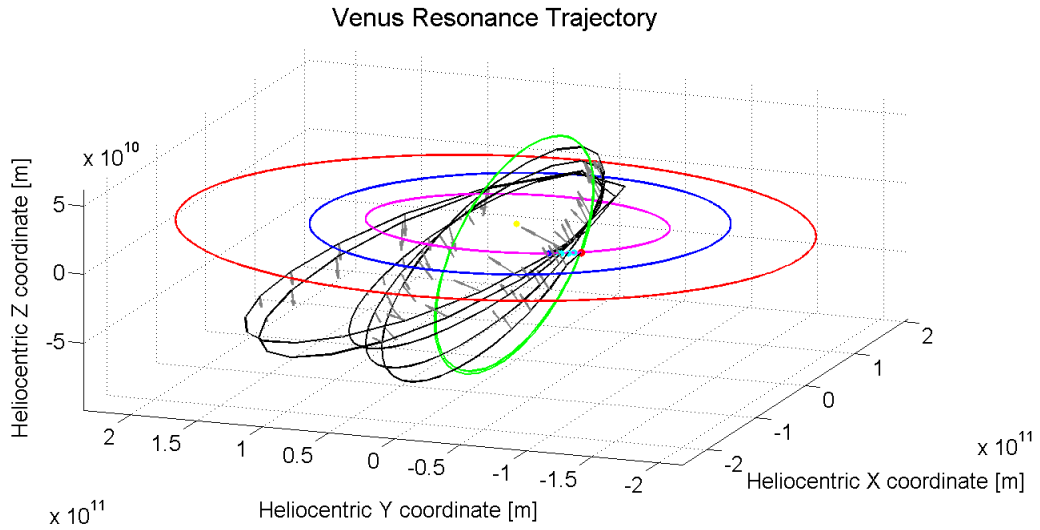

Figure 19. Venus resonance trajectory with combined low-thrust.

The color schemes are the same as in figure 15, with an added green portion. The latter is the trajectory after the last fly-by, which is to maximize inclination, without concerns of maintaining a resonance. The details for this orbit are given in figure 20.

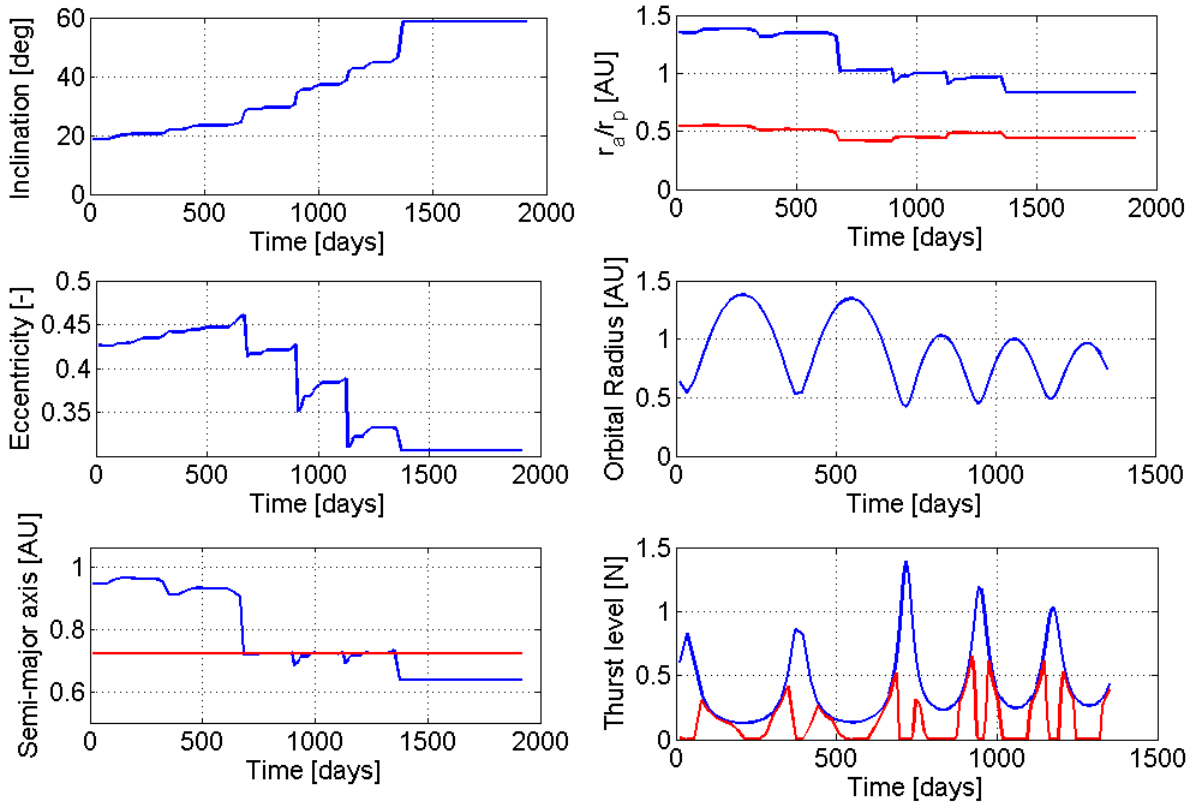

Figure 20. Details of Venus resonance trajectory. Radius increase factor 1.

The layout has already been shown before, the only difference is that at the end of some plots the trajectory is propagated after the last fly-by, to show the final orbit reached. This final orbit is ballistic (thus the orbital elements don not change), and occurs from 1351 days onward. Furthermore, the time on the x-axis is taken after arrival at Venus from the previous mission phase.

The last results for this chapter concern the V_∞ sphere where the fly-by circle has the 1.25 increase factor. The trajectory plot is not shown, however figure 21 gives the details for this new configuration.

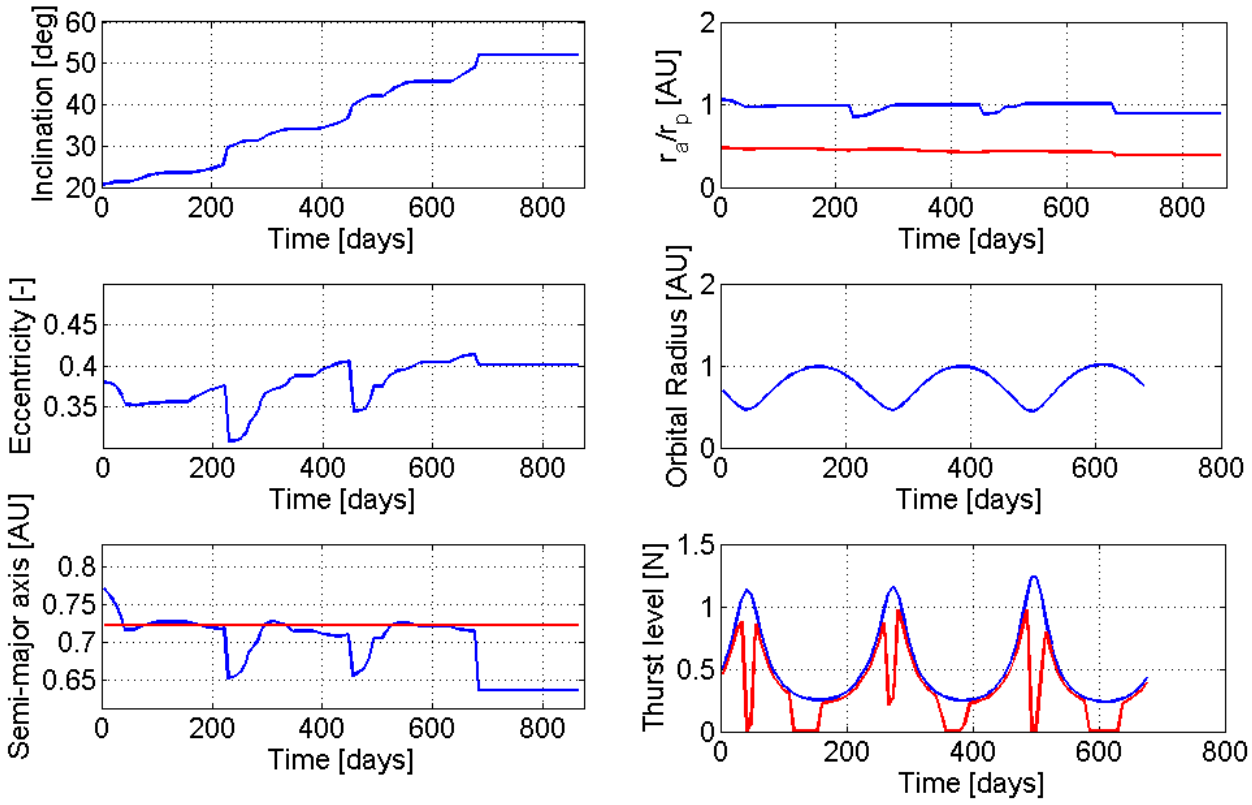

Figure 21. Details of Venus resonance trajectory. Radius increase factor 1.25.

From the graph it is clear even with the increase factor of 1.25 on the V_∞ circle radius, the low-thrust system is able to find a feasible trajectory, and reaches the maximum inclination in significantly less time. The final inclination however is lower than the previous. This can be explained by the fact that in the former case (increase factor 1) the complete trajectory is already feasible with no thrust, thus all propulsive capabilities can be devoted to increasing inclination. In the latter case (increase factor 1.25), the spacecraft needs to partially thrust so as to make the trajectory feasible. Keeping the same initial inputs table 6 gives a summary of the maximum achievable inclinations for different payload masses.

Table 6. Final inclination for varying payload masses. Radius increase factor 1.25.

Payload mass [kg]	Achieved inclination [deg]
1300	32.71
1100	37.21
900	41.84
700	46.49
500	51.82

4. Conclusion and notes for further research

At last, some important conclusions may be drawn about the final trajectory. With regards to the modified GASP method, it was shown that it surpasses the speed of the original, as well as providing good initial guesses for BOLTT. However it can be seen in figure 13 that this method is not able to replicate the exact island boundaries in case of complex shapes. Furthermore, it's possible that not all islands are found, due to the initial coarse grid search. For now, only one island out of each plot is examined, in the future it might be worth investigating more, as well as looking for alternative planetary combinations. Lastly, the island finding method can contain some inaccuracies, so it might be worth investigating more "intelligent" methods.

From the 57 BOLTT simulations it was shown that V_∞ magnitudes of 15-20 km/s can easily be reached both by Atlas 5 and Delta 4. The Falcon 9 has only a few feasible trajectories, and usually involve high powered systems with low masses.

The V_∞ sphere method was shown to produce time optimal resonant sequences to connect (almost) any V_∞ to the target vector for highest inclination. Not only does it provide good initial guesses to BOLTT, but the fly-by circle radius increase factor method also works. This latter has proved extremely useful, since it allows to skip slow resonances (like the 3:2), and move straight to the 1:1. On top of all this, this method (as well as the modified GASP), can be used extremely effectively for high thrust systems, as they both model the trajectory ballistically.

The BOLTT results for the resonant portion show that as the spacecraft performs fly-bys, it gradually exchanges semi-major axis and eccentricity for inclination. BOLTT almost always uses low thurst to burn out of plane, thus cranking inclination, rather than using it to somehow better line up the next fly-by. The relative contribution of low-thrust and fly-bys to the final inclination increase is quite varied. Usually with high low-thrust involvement (due for example to high propellant masses), the planetary fly-bys have a lower relative contribution. On average, each fly-by contributes somewhere in the range of 4-5 degrees (except the last, since it is not bounded by a resonance requirement). In the end, depending on desired payload mass, the final inclinations reached can be between 32 to 59 degrees.

Perhaps the most interesting area to expand on, is continuing the mission after having stayed in this inclined orbit for some time. It might be interesting to see what other orbits can be reached with left over propellant. Alternatively, one can also stay in a 1:1 Venus resonance, and use the planet to move into yet another orbit, perhaps increasing eccentricity, so as to lower the perihelion.

5. References

[1] Janin, G. "Trajectory design for the Solar Orbiter mission." Jornadas de Trabajo en Mecnica Celeste, , No. 25, pp. 177–218, 2004.

[2] Herman, J. and Noomen, R. "Preliminary Mission Design for a Far-Side Solar Observatory using Low-Thrust Propulsion." Charleston, SC, Jan 29 - Feb 2, 2012. 22nd AAS/AIAA Space Flight Mechanics Meeting - AAS 12-210.

[3] Kawakatsu, Y. and Kawaguchi, J. "Trajectory Options for Solar Polar Region Observation Mission." Journal of Aerospace Engineering, Sciences and Applications, Vol. 4, No. 3, pp. 93–103, jul 2012.

[4] Labunsky, A. Multiple gravity assist interplanetary trajectories. Gordon and Breach Science Publishers, Australia, 1998. ISBN 90-5699-090-X.

[5] De Smet, S., Parker, J., Herman, J., and Noomen, R. "Preliminary Design of a Crewed Mars Flyby Mission using Solar Electric Propulsion." Santa Fe, NM, January 26-30, 2014. 24th AAS/AIAA Space Flight Mechanics meeting - AAS 14-366.

[6] De Smet, S. Preliminary Design of a Crewed Mars Flyby Solar Electric Propulsion Mission. Master's thesis, Delft University of Technology, December 2014. Http://repository.tudelft.nl/view/ir/uuid

[7] De Smet, S., Parker, J., Herman, J., and Noomen, R. "Mission Design for a Crewed Earth-Venus-Mars-Flyby Mission using Solar Electric Propulsion." Breckenridge, CO, January 30 - February 4, 2015. 32nd annual AAS Guidance and Control Conference - AAS 15-091.

[8] Sims, J., Finlayson, P., Rinderle, E., Vavrina, M., and Kowalkowski, T. "Implementation of a Low-Thrust Trajectory Optimization Algorithm for Preliminary Design." Vol. 3, pp. 1872–1881. Keystone, CO, August 21-24, 2006. Collection of Technical Papers - AIAA/AAS Astrodynamics Specialist Conference.

[9] Gill, P. "Users Guide for SNOPT Version 7: Software for Large-Scale Nonlinear Programming." Tech. rep., University of California, San Diego, 2008.

[10] Izzo, D., Becerra, V. M., Myatt, D., Nasuto, S. J., and Bishop, J. M. "Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories." Journal of Global Optimization, Vol. 38, No. 2, pp. 283–296, 2007.

[11] Kemble, S. Interplanetary mission analysis and design. Springer in association with Praxis Pub, Berlin, 2006. ISBN 3-540-29913-0.

[12] Burden, R. Numerical analysis. Prindle, Weber, Schmidt, Boston, Mass, 1985. ISBN 0-87150-857-5.

[13] Bertsekas, D. Dynamic programming and optimal control. Athena Scientific, Belmont, Mass, 2012. ISBN 1-886529-44-2.

[14] Radice, G. "Deployment Considerations for Spacecraft Formation at Sun-Earth L2 Point." Nonlinear Dynamics and Systems Theory, Vol. 6, No. 4, pp. 401–411, 2006.