Interplanetary Gravity Assisted Trajectory Optimizer (IGATO)

Jason Bryan

California Polytechnic State University, San Luis Obispo
Aerospace Engineering
Senior Project 2009-2010
Advisor: Dr. Kira Abercromby

1. Introduction

Interplanetary space travel is an extremely complicated endeavor that is severely limited by our current technological advancements. The amount of energy required to transport a spacecraft from one planet to the next, or even further, is extraordinary and in some cases is even impossible given our current propulsive capabilities. Due to these complications, the search for other means of exchanging energy became imperative to future space exploration missions. One particularly powerful method that was discovered, and the most commonly used one, is referred to as planetary gravity assist.

Planetary gravity assists, or planetary flybys for short, involve exchanging energy or momentum with a planet as a spacecraft flies past it. As long as an approaching spacecraft does not crash into the planet or get captured into a parking orbit, it will continue past the planet on a hyperbolic trajectory (Curtis 375). The idea is that, during this hyperbolic trajectory, there will occur a momentum exchange between the spacecraft and the planet. Since the mass of the planet is so massive, the small subtraction or addition of momentum results in a negligible change its net velocity. However, since the mass of the spacecraft is extremely light compared to the planet, the velocity imparted on the spacecraft from the planet can amount to sizeable quantities. Whether the spacecraft gains momentum or loses momentum entirely depends on the direction that it flies past the planet. If the spacecraft crosses in front of the planet's direction of motion, the flyby is referred to as a leading-side flyby and the spacecraft will lose heliocentric velocity. On the contrary, if the spacecraft crosses behind the planet's direction of motion, it's called a trailing-side flyby and the spacecraft gains heliocentric velocity (Curtis 376). In addition, the magnitude of the momentum exchange can be varied depending on the direction of the approach vector of the spacecraft. Thus, simply by flying past a planet, a spacecraft can choose to either gain or lose scalable quantities of velocity for practically free.

The attractiveness of planetary flybys to space exploration is obvious given our current space propulsion limitations. They offer a controllable, powerful, and repeatable method of changing the orbital energy of a spacecraft which allows for a much broader range of solar exploration. However, the unfortunate downside of planetary flybys is their inherent dependence on the proper alignment of planets. Precise timing and control is required in order to launch a spacecraft into orbit so that it flies past a desired planet in the right direction at exactly the right time. These complexities escalate tremendously when several planetary flybys are desired to be linked together, as they commonly are. In order to plan out multiple gravity assisted trajectories, complex and robust computer simulations are required to filter through the continuum of possibilities and select trajectories that optimally satisfy the mission requirements. This paper discusses one such computer simulation which seeks to minimize the propulsive delta v requirements of a spacecraft for a trajectory between two specified planets utilizing a specified number of planetary flybys along the way. It is important to note that this optimizer is not finished and this paper serves only as a mere precursor to an ensuing Master's Thesis. For this reason, this paper provides only a basic overview of the inter-workings of the optimizer and provides no conclusive evidence of its validity.

2. Methodology

2.1 Introduction

There are currently multiple approaches to gravity assisted orbit optimization. The approach outlined in this report is referred to as a patched Lamberts problem and utilizes a brute force tactic. The underlying principle of this strategy involves breaking down a trajectory into separate segments or *legs* and analyzing each leg one at time with respect to the leg prior to it. Each leg is defined as an orbital trajectory between planet A and planet B, where planets A and B can be either the same planet or different ones. The legs are analyzed with respect to constraint criteria which are established prior to running the simulation. If the criteria are satisfied, the next leg is analyzed and so forth until a complete and valid trajectory is found.

2.2 System Variables and Initial Conditions

The initial inputs to the system consist of many factors. Firstly, a planet itinerary must be chosen. This itinerary details the initial departure planet, the order of planetary flybys, and finally the desired arrival planet. The total number of legs in the trajectory is defined by *m* and therefore the total number of flyby planets is always m-1.

$$Planet\ Itinerary\ = \begin{Bmatrix} Departure\ Planet \\ Flyby\ Planet\ 1 \\ \vdots \\ Flyby\ Planet\ i \\ \vdots \\ Flyby\ Planet\ m-1 \\ Arrival\ Planet \end{Bmatrix} \qquad 2.2.1$$

Next, a range of possible departure Julian dates must be selected. These dates represent all the possible launch dates from the initial planet. Also, each leg utilizing the Lamberts Problem Algorithm must establish its own range of time of flights, or TOFs. The number of intervals within these ranges is defined by the global fidelity number, *N*.

$$Departure\ Julian\ Dates = \ [JD_1 \quad JD_2 \quad \dots \quad JD_i \quad \dots \quad JD_N\ ] \qquad 2.2.2$$

$$TOF = \ [TOF_1 \quad TOF_2 \quad \dots \quad TOF_i \quad \dots \quad TOF_N] \qquad 2.2.3$$

Legs involving a same-planet flyby do not require inputs other than N since the apse line rotation angle, η, will always be varied from zero to 360 degrees.

$$\begin{aligned} Apse\ Line\ Rotation\ Angles &= [\eta_1 \quad \eta_2 \quad \dots \quad \eta_i \quad \dots \quad \eta_N] \\ &= [\eta_1 \quad \eta_2 \quad \dots \quad \eta_i \quad \dots \quad 360°] \end{aligned} \qquad 2.2.4$$

Using the TOF and η vectors, the TOF/ETA matrix is assembled. In this matrix, each row is defined as a different leg within the trajectory. For legs that utilize the Lamberts Problem Algorithm, the row that corresponds to that leg in the TOF matrix is replaced by the TOF vector, otherwise the row is replaced by the η vector. For example, given the scenario of a mission involving a departure from Earth, two Venus flybys, and a arrival at Mercury, the TOF/ETA matrix would be assembled as follows

$$TOF/ETA\ Matrix\ (example) = \begin{bmatrix} TOF_{11} & TOF_{12} & \dots & TOF_{1i} & \dots & TOF_{1N} \\ \eta_{21} & \eta_{22} & \dots & \eta_{2i} & \dots & \eta_{2N} \\ TOF_{31} & TOF_{31} & \dots & TOF_{3i} & \dots & TOF_{3N} \end{bmatrix} \qquad 2.2.5$$

Here, the first row corresponds to the TOFs between Earth and Venus, the second row corresponds to the apse line rotation angles for the Venus-Venus flyby, and the third row corresponds to the TOFs between Venus and Mercury. Lastly, each planet specifies a minimum altitude that is acceptable during a flyby as shown below. Note the first and last entries are left not applicable because a flyby is not performed on the departure and arrival planets.

$$Minimum\ Altitude = \begin{bmatrix} n/a \\ Z_{Flyby\ Planet\ 1} \\ \vdots \\ Z_{Flyby\ Planet\ m-1} \\ n/a \end{bmatrix} \qquad 2.2.6$$

The next set of inputs deal with the acceptance criteria and constraints that connect each leg. For the departure leg, since there are no legs preceding it, the only constraint is that the departure v-infinity vector magnitude remain below a specified threshold. For the subsequent legs involving flybys, there are two pivotal criteria that must be satisfied. Firstly, the difference in approach v-infinity and departure v-infinity for that flyby planet must be below a specified tolerance. Secondly, the turn angle required by the spacecraft during the flyby must be greater than the minimum turn angle derived from the planet's lowest acceptable flyby altitude. Finally, for the arrival leg, the same set of criteria exists as the flyby legs with a couple additions. Similar to the departure leg, the arrival v-infinity vector magnitude and the total delta-v for the entire mission must be below their respective tolerances. Table 2.2.1 summarizes these acceptance criteria.

Table 2.2.1. Acceptance Criteria

	Departure Leg	**Flyby Legs**	**Arrival Leg**
Criteria	Launch V_{inf}^{+}	ΔV_{inf} δ_{Max}	ΔV_{inf} δ_{Max} Arrival V_{inf}^{-} Total Mission ΔV

2.3 Program Logic and Flow

The fundamental logic in this program utilizes a brute force method. A trajectory is constructed one leg at a time under the directive of the respective departure, flyby, and arrival protocols as shown below in figure 2.3.1.

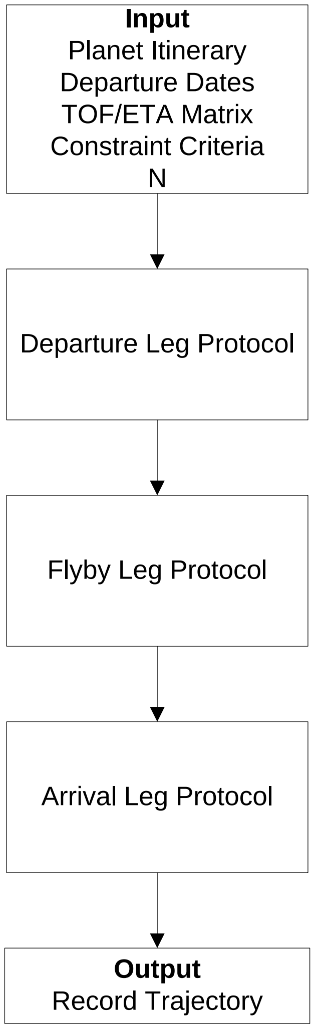

Figure 2.3.1. General Logic Diagram Overview

To begin, the first departure date from the vector outlined in equation 2.2.2 is chosen along with the first entry of the TOF/ETA matrix, similar to that of equation 2.2.5. With these inputs, the first trajectory is constructed using either the Lamberts Problem or Apse Line Rotation Algorithms depending on whether the second planet is different or the same as the first respectively. After the trajectory is defined, the constraint criteria are checked. In this case, if the departure v-infinity is less than the tolerance, the optimizer moves on to the flyby protocol. However, if the current trajectory fails to meet the criteria, the next sequential TOF or η is chosen from the first row of the TOF/ETA matrix and a new trajectory is formulated. This process continues until a valid departure trajectory is found or until all entries in the first row of the TOF/ETA matrix have been exhausted. In this latter case, in which the Nth TOF or η is still invalid, the optimizer backtracks to the initial simulation variable, the departure date, and selects the next sequential entry. It then selects the first TOF or η again from the first row of the TOF/ETA matrix and the process begins again. If no departure trajectory is found for all departure dates then the optimizer returns zero results. Figure 2.3.2 summarizes the logic of the departure leg protocol.

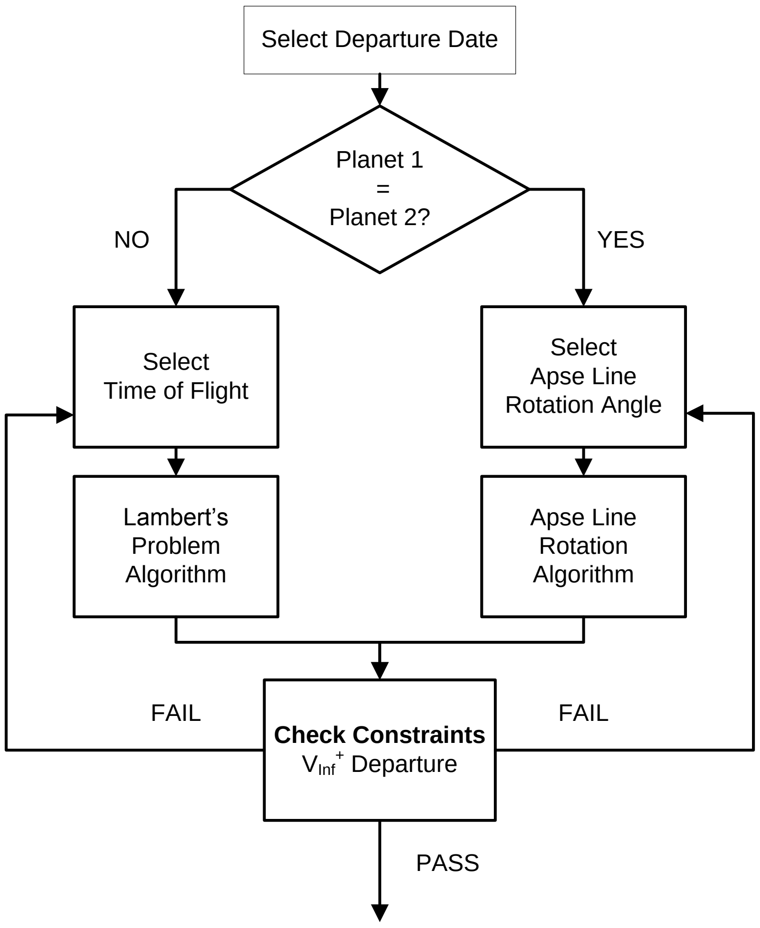

Figure 2.3.2. Departure Leg Protocol Block Diagram

Given that a valid departure trajectory is found, the optimizer then moves on to the flyby leg protocol. The flyby leg protocol is very similar to that of the departure leg, however, this protocol has slightly different constraint criteria and runs a variable number of times depending on the number of flybys selected. To begin, just as with the departure leg protocol, a TOF or η is chosen depending on whether the next planet is different or the same. The appropriate algorithm is then run to generate a trajectory connecting the two planets. At this point a new series of constraints must be checked.

The underlying principle of the optimizer is linking together different trajectories with the planetary flyby serving as the joint between them. By definition, the magnitude of the v-infinity vector of a spacecraft approaching a planet will be the same as the v-infinity vector leaving it after the flyby. Therefore, in order to perfectly link together two separate trajectories, the approaching v-infinity vector of the first trajectory must have the same magnitude as the departing v-infinity vector of the second trajectory. In reality however, it is nearly impossible to find two trajectories that perfectly satisfy this condition and a small difference in delta v will exist between the arriving and departing v-infinity vectors. In order to overcome this delta v, the spacecraft is required to make up the difference using its own propulsive capabilities during the flyby. The maximum delta v the spacecraft is allowed to administer during a given flyby is defined by the ΔV_{inf} tolerance as mentioned in Table 2.2.1. This is one of the two constraint criteria that must be satisfied in order to accept a trajectory within the flyby protocol. The other constraint is the maximum turn angle experienced by the spacecraft during the flyby which depends on the minimum flyby altitude of each planet as shown previously in equation 2.2.6. If these criteria are met, the optimizer moves on to the next flyby leg until the final flyby leg is reached in which it proceeds to the final arrival leg protocol. If however, the criteria is not met the optimizer backtracks and advances the soonest available variable. For example, if the constraints are not met during the first flyby leg, the next subsequent TOF or η will be chosen from the second row of the TOF/ETA matrix. If all TOFs or η's are exhausted in the second row, the optimizer backtracks to the departure leg protocol and selects the next TOF or η from the first row. If, instead, the optimizer exhausts a row of the TOF/ETA matrix during subsequent flyby legs, it then backtracks to the previous flyby leg and advances the TOF or η for that leg. In this way, the brute force methodology of the optimizer becomes clear as the optimizer continuously marches forward with one trajectory until it exhausts all of its current options and is forced to backtrack to and advance a previous variable. Figure 2.3.3 summarizes the flyby leg protocol.

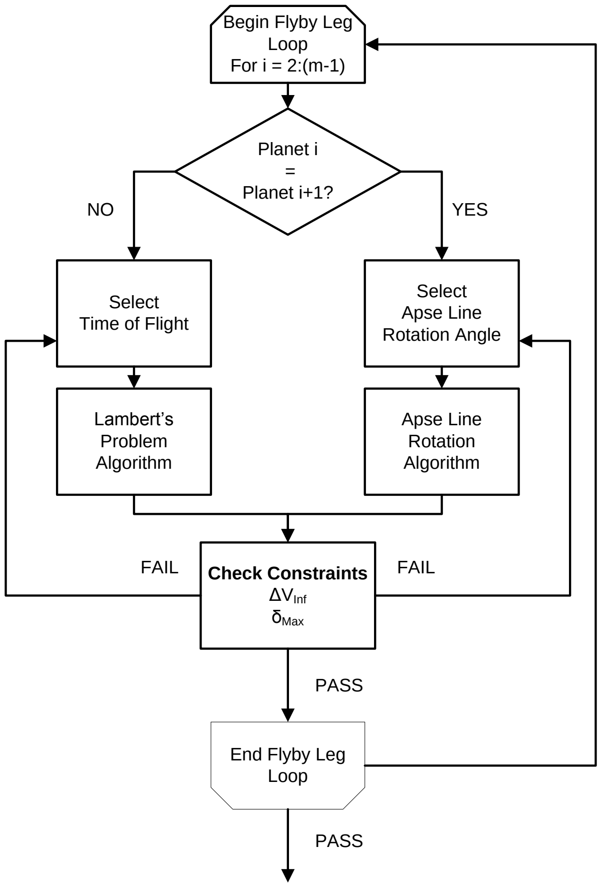
Begin Flyby Leg Loop For i = 2:(m-1)
Planet i = Planet i+1?
NO
YES
Select Time of Flight
Select Apse Line Rotation Angle
Lambert's Problem Algorithm
Apse Line Rotation Algorithm
FAIL
Check Constraints ΔV_{Inf} δ_{Max}
FAIL
PASS
End Flyby Leg Loop
PASS

Figure 2.3.3. Flyby Leg Protocol Block Diagram

Once the optimizer obtains a trajectory that satisfies the departure leg and flyby leg protocols, it advances to the final arrival leg protocol. This protocol is very similar to the two prior, however once again it has slightly different constraint criteria. During the arrival leg protocol, a trajectory from the last flyby planet to the final arrival planet is formulated. For this trajectory, two sets of constraint criteria must be satisfied. Firstly, the departing v-infinity vector must satisfy the delta-v tolerance for the flyby and the max turn angle must not be exceeded as mentioned previously. Secondly, the arrival v-infinity vector approaching the arrival planet and the total delta v of the entire mission must be below their respective specified tolerances as outlined in table 2.2.1. The total delta v is found by summing the initial departure v-infinity magnitude, to the required delta v's during each flyby, to the final arrival v-infinity magnitude.

$$Mission\ \Delta V = {V_{inf}}^{+}\Big|_{Depart} + \sum_{i=1}^{m-1} \Delta V_{inf,i} + {V_{inf}}^{-}\Big|_{Arrive} \qquad 2.3.1$$

If all of these conditions are met, a valid trajectory has been realized and the optimizer records the necessary data needed to reconstruct the trajectory in a separate text file. With a trajectory found, the optimizer advances its current local variable, if possible, or backtracks to and advances the variable proceeding it in order to look for more solutions. In this manner, the optimizer systematically pursues every possibility feasible given the initial conditions. Figure 2.3.3 summarizes the arrival leg protocol.

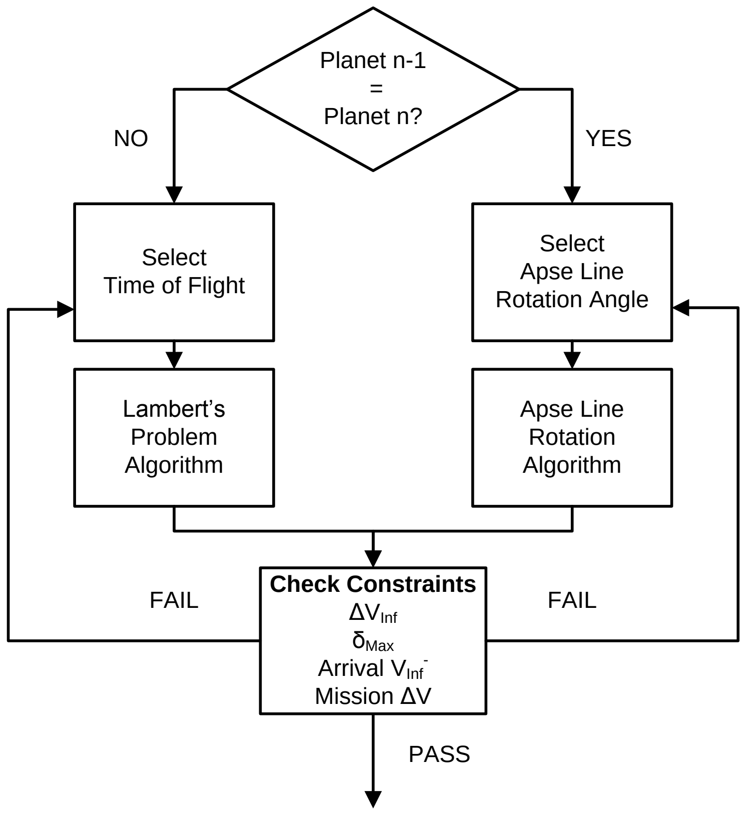

Figure 2.3.4. Arrival Leg Protocol Block Diagram

3. Conclusion

This paper outlined the method behind a powerful and effective interplanetary gravity assisted trajectory optimizer. This method utilizes a robust brute force tactic that constructs highly complicated multi-flyby trajectories by analyzing one leg at a time and constantly adhering to specified constraint criteria. The primary tools for the trajectory formulation consist of the widely utilized Lamberts Problem Algorithm and a homebuilt Apse Line Rotation Angle Algorithm. The meat of the optimizer resides in the protocol directives for each of the three main phases of the trajectory: departure, flyby(s), and arrival, which serve to navigate flow of the optimizer either forwards or backwards as it searches for a valid and optimum trajectory. In the future, this optimizer will be expanded to include the possibility of performing deep space maneuvers, it will feature a robust graphical user interface, and will provide sufficient evidence validating its accuracy.

4. References

Curtis, Howard. Orbital Mechanics for Engineering Students. Elsevier Ltd. Burlington, MA. 2007.

Vallado, David. Fundamentals of Astrodynamics. Microcosm Press and Springer. Hawthorne, CA. 2007.