INTERPLANETARY RIDESHARING: EXPLORING POTENTIAL CUBESAT TRAJECTORIES

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Aerospace Engineering

By

Liam Smith

June 2015

COMMITTEE MEMBERSHIP

TITLE:	Interplanetary Ridesharing: Exploring Potential CubeSat Trajectories
AUTHOR:	Liam Smith
DATE SUBMITTED:	June 2015
COMMITTEE CHAIR:	Kira Abercomby, Ph.D. Aerospace Engineering Assistant Professor
COMMITTEE MEMBER:	Jordi Puig-Suari, Ph.D. Aerospace Engineering Professor
COMMITTEE MEMBER:	Ricardo Tubio-Pardavila, Ph.D. Aerospace Engineering Lecturer
COMMITTEE MEMBER:	Robert Lock, Mars Program Formulation Office NASA Jet Propulsion Laboratory

ABSTRACT

Interplanetary Ridesharing: Exploring Potential CubeSat Trajectories

Liam Smith

Ever since the revolutionary CubeSat form factor took hold in the Aerospace industry, there has been a desire to send them further and further into space. This thesis introduces an optimization approach to deployment that explores new possibilities of interplanetary CubeSats. In this approach there are three categories of objective functions that are defined by the type of trajectory of a "primary" spacecraft, which carries the CubeSat deployer. These categories are flyby, orbiter, and lander. For each category the objective function starts with four design variables. These are the ΔV of the deployer broken up into three component directions and the true anomaly at the time of deployment. The method then calculates the mission specific objective to be minimized and uses Matlab®'s built in gradient-based optimizer, *fmincon*. The results show that in the flyby category, the CubeSat has a significantly different turning angle than the primary. The CubeSat can even flyby on the opposite side of the planet. In the orbiter case it is shown that the method works by testing it with two objective functions, the difference in inclination and the difference in eccentricity between the primary and the CubeSat. It is shown that the inclination can be changed by 0.1314° and the eccentricity can be changed by 0.0033. These values, although low in magnitude, are an order of magnitude greater than non-optimal deployment scenarios. Still, another optimization method is introduced to find out how much extra ΔV the CubeSat would need to reach a desired change. This shows that with just an extra 75 m/s of ΔV, the CubeSat can change its orbit by 5°. This could come from either a propulsion system or a modified deployer.

The final category, lander, used the flight path angle when entering the atmosphere as an objective. The method shows that flight path angle can be changed by 2.6°. Overall, these examples have proven that the method can find optimal solutions to CubeSat deployment scenarios at other planets.

Keywords: CubeSat, Interplanetary, Cal Poly, P-POD, Orbital Trajectories, Optimization, Orbital Dynamics, Mars, JPL, PolySat

ACKNOWLEDGMENTS

I acknowledge everybody. Most importantly though are my friends, family, and girlfriend who didn't kill me when things got a little crazy and I got a little weird.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

Page

NOMENCLATURE & COMMON ACRONYMS

Acronym	
Cal Poly	California Polytechnic State University
COE	Classical Orbital Elements
DCM	Direction Cosine Matrix
ECI	Earth Centered Inertial
LEO	Low Earth Orbit
PDSD	Poly-Picosatellite Deep Space Deployer
P-POD	Poly-Picosatellite Orbital Deployer
U	A standard CubeSat unit. 10x10x10cm

Variable	
e	Eccentricity
h	Specific Angular Momentum
I	Mass Moment of Inertia
i	Inclination
k	Spring Constant
m	Mass
r_p	Radius of Periapsis, Closest Approach
v	Velocity
X	Design Variable Vector, Inertial Coordinate Direction
x	Design Variable, Spring Displacement, Rotating Coordinate Direction
Y	Inertial Coordinate Direction
y	Rotating Coordinate Direction
Z	Inertial Coordinate Direction
z	Rotating Coordinate Direction
γ	Flight Path Angle, Entry Angle
Δ	Change (with another variable), Aiming Distance
δ	Turning Angle (hyperbolic trajectories)
θ	True Anomaly
μ	Gravitational Parameter
ω	Argument of Periapsis, Rotational Velocity
Ω	Right Ascension of Ascending Node (RAAN)

Subscript	
pp	Pusher Plate
ms	Main Spring
sp	Spring Plunger
sp_present	Preset displacement on Spring Plungers
3U	3U CubeSat
6U	6U CubeSat

1. Introduction

Introduced in 1999 by co-inventors Dr. Jordi Puig-Suari of California Polytechnic State University and Dr. Bob Twiggs of Stanford University, the CubeSat form factor has grown into an entire industry of its own. A CubeSat is a class of picosatellites designed as a secondary payload that allows for low cost missions with relatively short development times. The standard 10x10x10 cm^3, or 1U, CubeSat was developed with the goal of allowing a team of students to design, build, test, and operate an entire satellite from the very beginning stages to the death of the satellite, all within their academic careers. Since then, hundreds of universities, small companies, industry leading corporations, and individuals have embraced the idea of the CubeSat standard and expanded its capabilities.

1.1 Background

Ever since the revolutionary CubeSat form factor took hold in the Aerospace industry, there has been a desire to send the satellites further and further into space. But CubeSats have very well known roadblocks when it comes to exploring further than Low Earth Orbit. They lack power for communication, radiation shielding to protect their electronics outside of Earth's magnetic field, and significant propulsion for orbital maneuvers. While there are many teams trying to solve these exciting problems, Cal Poly has started to design an Interplanetary CubeSat Deployer, tentatively named the Poly-Picosatellite Deep Space Deployer or PDSD, which will open up a world of new possibilities for where CubeSats can go. The interplanetary deployer will be mounted on a primary interplanetary spacecraft, preserving the ridesharing mentality, and can be designed to solve the communication and radiation problems for most interplanetary CubeSats.

However, as of today even the most advanced CubeSat propulsion systems can only reach about a 100 m/s change in velocity. This means the CubeSat is still stuck on the trajectory of the primary. This thesis aims to change that.

1.2 Thesis Objective

Previous California Polytechnic State, San Luis Obispo (Cal Poly) students have conducted analysis to characterize a CubeSat's deployment velocity when exiting the current version of Cal Poly's P-POD (Poly-Picosatellite Orbital Deployer). This thesis aims to build off of that previous analysis, modify the equations of motion to fit current designs of the PDSD, and investigate where CubeSats can potentially go when deployed on interplanetary missions. To keep the scope of this thesis reasonable, three scenarios are considered and all of these scenarios have Mars as the target planet. This can be modified in future work with any celestial body in mind. The first scenario is the possibility of deploying a CubeSat during a planetary flyby. The CubeSat then has three options, fly by the planet as well, attempt to be captured into an orbit around the planet, or land/impact the planet. In the second scenario, there is already a primary spacecraft in a stable orbit around a planet. The CubeSat has the same three options: leave the planet on a hyperbolic trajectory, deploy into a sufficiently different orbit, or deploy and land on the planet. Realistically, the CubeSat will not be able to leave the planet or land on its surface in this case. The final scenario is where the primary is on a trajectory to land on the planets surface. Again, the CubeSat has three potential trajectories: flyby, orbit, or land. At first it seems that some of these cases are unrealistic, as a canister type deployer has very little hope of deploying a CubeSat into an escape trajectory from an orbiting

satellite, but others are extremely possible and potentially valuable. So to summarize, the overall goal is to deploy a CubeSat into a sufficiently different orbit than the primary.

Since there exists an infinite number of deployment sequences that will achieve an infinite amount of different CubeSat trajectories, the problem needs to be bounded. By analyzing these three scenarios with the bounds characterized by a P-POD type CubeSat deployer, this thesis will create a tool to find the most optimal trajectories and thus characterize the potential CubeSat missions that could arise from further development of the PDSD. Overall, this thesis can be used as a tool for interplanetary mission planners when exploring the possibilities of using CubeSats on their interplanetary missions.

2. Poly-Picosatellite Deep Space Deployer (PDSD)

The Poly-Picosatellite Deep Space Deployer is the product of a three month long design collaboration between Cal Poly CubeSat and NASA JPL. The goal of the project was to create a CubeSat deployer that would be used on interplanetary missions.

2.1 Similarities and Differences to the P-POD

The design of the PDSD was expected to be a 6U version of the P-POD with added features and capabilities that make it suitable for interplanetary missions. Because of this the core concept of the design is very similar to the P-POD. Figure 1 shows the two CubeSat deployers side by side for comparison. It is not immediately obvious from the figure but the deployment mechanics of the deployers are nearly identical. The deployers use a spring and rail type deployment method that is suitable for any CubeSat that follows the CubeSat standard. The PDSD has the capability of opening one door at a time or both doors simultaneously. This means that it can deploy any combination of CubeSats that add up to 3U from each side or one 6U CubeSat. Interplanetary CubeSats typically require more power, more redundancy, and more radiation shielding than the typical LEO (Low Earth Orbit) CubeSat, therefore only the 6U case is explored in this thesis. Also note that the PDSD is twice as wide as the P-POD, so it uses two main springs instead of one to deploy the CubeSat, and the pusher plate is twice as wide as its P-POD counterpart. The PDSD still has four rails and uses four spring plungers as a kick force to aid in deployment like the P-POD. Both deployers use hard-anodized aluminum rails with a low coefficient of friction. There are many other design characteristics of both deployers that are not relevant to the scope of this thesis.

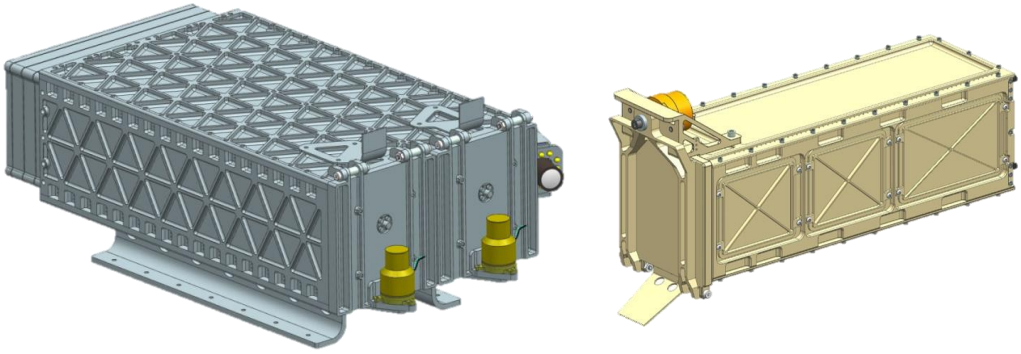

Figure 1. Side by side comparison of the PDSD (left) and the Mk. IV P-POD (right).[1]

3.CubeSat Deployment

The first step to understanding the complete deployment environment of the PDSD is to understand the P-POD, the system that is already in use for Low Earth Orbit (LEO). Once the dynamics of the P-POD are fully understood, it is a small step to scale the model to what is essentially a 6U version with added capabilities.

3.1 P-POD Deployment Dynamics

The deployment mechanism of the P-POD is a fairly simple spring mass system. Figure 2 is a simplified exploded view of an earlier revision of the P-POD.

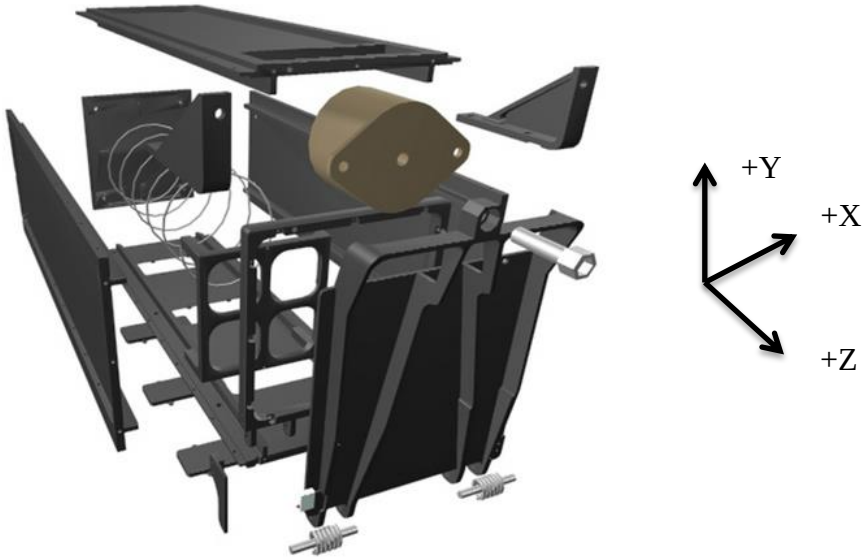

Figure 2. An exploded view of the P-POD Mk.II with coordinate system.[1]

The main spring is attached to the pusher plate, which is the primary method of deploying the CubeSat. There are also four spring plungers that add an initial force to the legs of the pusher plate to aid in deployment. For preliminary modeling, the velocity of the CubeSat when leaving the P-POD is calculated from the transfer of the potential energy of these

five springs into the kinetic energy of the CubeSat.[10] This energy equation is shown in Eq. 1.

$$\frac{1}{2}k_{ms}x_{ms}^2 + \frac{1}{2}k_{sp}x_{sp}^2 = \frac{1}{2}(m_{3U} + m_{pp})v_{3U}^2 + \frac{1}{2}k_{sp}x_{sp\_preset}^2 \quad (1)$$

In Eq. 1, *k* is a spring constant in newtons per meter (N/m), *x* is the displacement of the spring in meters (m), *m* is the mass in kilograms (kg), and *v* is the velocity in meters per second (m/s). The meaning of the subscripts on each variable can be found in Table 1. All of the forces are assumed to be acting directly on the center of mass of the CubeSat and in the +Z direction of the P-POD.

Table 1. Common subscript meanings

Subscript	Meaning
ms	Main Spring
sp	Spring Plunger
6U	6U CubeSat
pp	Pusher Plate
sp_present	Preset displacement on Spring Plungers

Rearranging Eq. 1 will solve for the velocity of the 3U CubeSat when leaving the P-POD as shown in Eq. 2. [10] All of the variables and subscripts are the same as in Eq. 1.

$$v_{3U} = \sqrt{\frac{k_{ms}{x_{ms}}^2 + k_{sp}{x_{sp}}^2 - k_{sp}{x_{sp\_preset}}^2}{(m_{3U} + m_{pp})}} \quad (2)$$

This method of determining the CubeSat's velocity is considered an energy method since it converts potential energy to kinetic energy. Using Eq. 2 and nominal values from the CubeSat Design Specification[1], the expected value for the velocity of a 3U CubeSat

when leaving the P-POD is 1.993 m/s. The deployment is then simulated using an ordinary differential equation solver in Matlab® with the equations of motion (EOMs) of the CubeSat. These EOMs are broken up into three categories. The first is when the pusher plate is still in contact with the spring plungers.

$$a = \frac{k_{ms}x_{ms}+k_{sp}(x_{sp}-x_{sp_{preset}})}{(m_{3U}+m_{pp})} \tag{3}$$

The next phase is when the spring plungers are no longer acting on the pusher plate and only the main spring is applying force.

$$a = \frac{k_{ms}x_{ms}}{(m_{3U}+m_{pp})} \tag{4}$$

The final phase of the motion is when main spring reaches its maximum displacement and the CubeSat leaves the P-POD.

$$a = 0 \tag{5}$$

Note that the displacement of each spring is the difference between the starting position and the position of the pusher plate in that iteration. The simulated results are shown in Fig. 3 with a steady-state velocity of 1.999 m/s. This agrees with the energy method from before and with the senior project titled: "CubeSat Orbit Prediction and Simulation from Deployment Conditions"[10] on which this analysis was based.

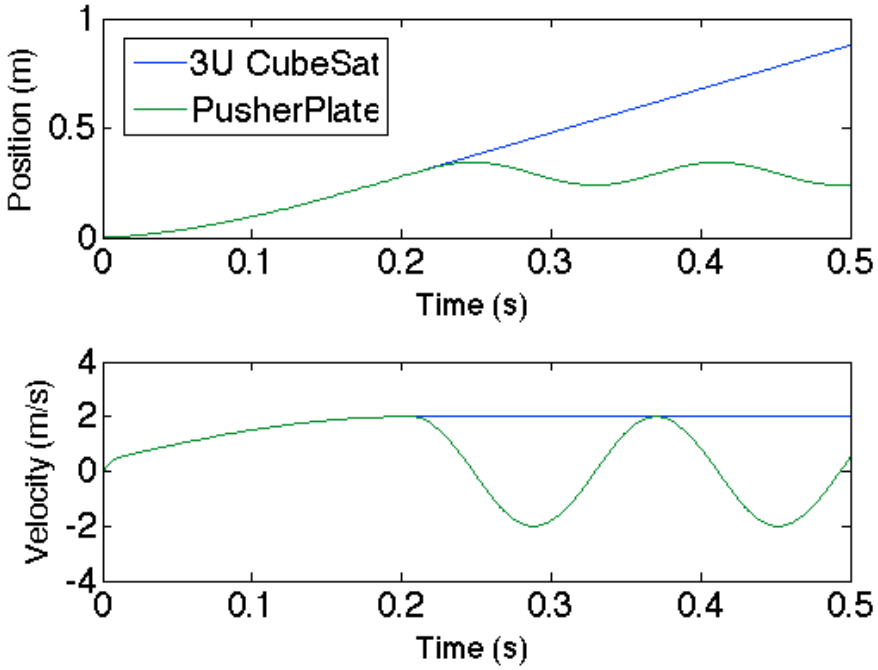

Figure 3. Simulated deployment of a 3U CubeSat.

3.2 PDSD Deployment Dynamics

As described above in section 2.1, the PDSD has a near identical deployment method as the P-POD. Because of this, the same methodology for deployment analysis is applied to the PDSD to find the ΔV that it can provide to a CubeSat. The following equations are modified from Eq. 1 and Eq. 2 to fit the 6U deployer.

$$\frac{1}{2}k_{ms1}x_{ms1}^2 + \frac{1}{2}k_{ms2}x_{ms2}^2 + \frac{1}{2}k_{sp}x_{sp}^2 = \frac{1}{2}(m_{6U} + m_{pp})v_{6U}^2 + \frac{1}{2}k_{sp}x_{sp\_preset}^2 \quad (6)$$

$$v_{6U} = \sqrt{\frac{2*k_{ms}{x_{ms}}^2 + k_{sp}{x_{sp}}^2 - k_{sp}{x_{sp\_preset}}^2}{(m_{6U}+m_{pp})}} \quad (7)$$

Since the two main springs in the PDSD are identical within tolerance, the two terms can be combined into one. Also, the 6U version of the pusher plate and the 6U CubeSat nominal mass are twice the mass of their 3U counterparts. Because of this the velocity of

the CubeSat leaving the PDSD is expected to be very similar to the velocity when leaving the P-POD. The difference in velocities would come from the fact that there are still only four spring plungers in the PDSD when everything else is doubled. Using Eq. 7 the expected velocity is 2.005 m/s. This is again compared to a simulation to verify the results. The plots in Fig. 4 are from the same simulation as the P-POD, but modified to fit the PDSD. They agree with the energy method with a steady-state velocity 2.016 m/s.

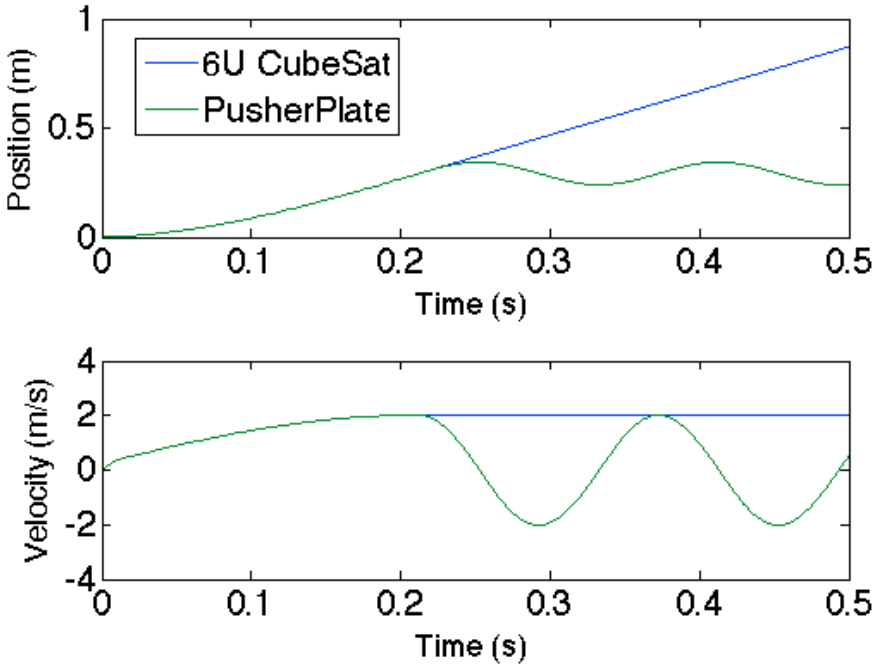

Figure 4. Simulated deployment of a 6U CubeSat

As expected, the 6U deployer has nearly identical deployment capabilities as the P-POD.

3.3 Deployment Uncertainties

As noted, the above method ignores any friction with the rails. This is a fair assumption for more reasons than just a low coefficient of friction provided by the anodization. The first concern to discuss is friction being a loss of kinetic energy. This would mean less

ΔV from the P-POD overall. This can be accounted for by adjusting the velocity of the CubeSat by 2%. It will be shown later on that 2% does not affect the optimal deployment at any noticeable level. The percentage is chosen because it is the accepted amount of energy that is lost to friction in the study of interior ballistics.[8] Interior ballistics is the dynamics of propelling a projectile from the time of ignition to the time that the projectile leaves its container (specifically a gun barrel) making the assumption that the P-POD and PDSD are similar to a gun barrel is an almost conservative guess. This is because a gun barrel applies friction on all sides of a projectile where a CubeSat deployer only applies friction along the rails.

The second reason ignoring friction could be considered a poor assumption is because there can be uneven friction on each rail. This can impose a velocity vector on the CubeSat that is not fully in the +Z direction of the P-POD. According to the CubeSat Design Specification[1], this deployment cone will not exceed 10°. Ten degrees essentially means another 2% of the total energy is not used in the deployment direction since the cosine of ten degrees is 0.98. This makes the total energy loss due to friction 4% and thus not a poor assumption.

The uneven friction can also induce a rotation on the CubeSat, converting some of the original potential energy into rotational kinetic energy. This loss of translational kinetic energy must also be accounted for. To do this, a simulation of the worst possible case is considered. This is a case in which the spring plungers on one side of the P-POD are at the maximum deflection while the spring plungers on the other side are not compressed at all. To analyze the rotation this would impart on the CubeSat, consider each corner of the rails moving at a separate velocity as shown below.

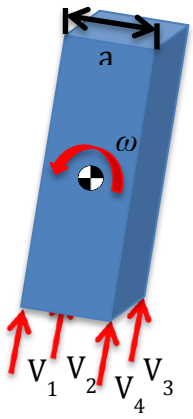

Figure 5. CubeSat right after deployment. All corners moving at different velocities.

Each corner can be considered to have one fourth of the velocity created by the main spring plus the velocity created by the associated spring plunger. This is shown in Eq. 8.

$$v_i = \sqrt{\frac{\frac{1}{4}k_{ms}{x_{ms}}^2 + k_{sp}{x_i}^2 - k_{sp}{x_{sp\_preset}}^2}{\left(\frac{1}{4}m_{3U} + \frac{1}{4}m_{pp}\right)}} \; ; (i = 1,2,3,4) \tag{8}$$

It can also be said that the velocity of the corner is as follows, where $\vec{V}_{CG}$ is the velocity of the CubeSat at its center of gravity, $\vec{\omega}$ is the angular rotation vector and $\vec{r}$ is the position vector in the body frame of the CubeSat.

$$\vec{v}_i = \vec{v}_{CG} + (\vec{\omega} \times \vec{r}) \tag{9}$$

Equation 10 is the scalar form of Eq. 9.

$$v_i = v_{CG} + \omega_{x,y,z} * R_{x,y,z} \tag{10}$$

Now the equation can be rearranged to solve for ω.

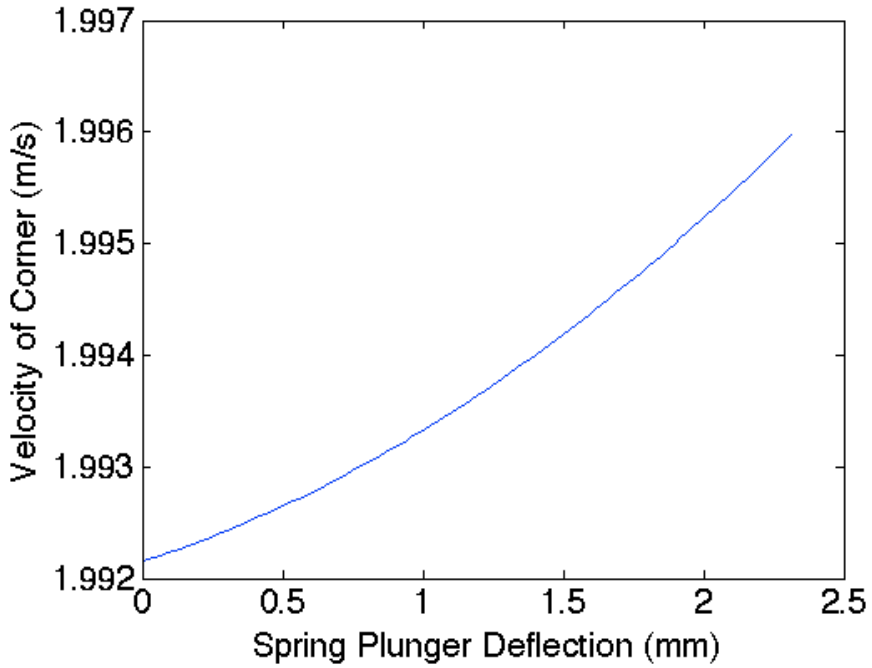

Figure 6. Velocity of the CubeSat at each corner based on varying spring plunger displacement

Using Eq. 10 and the range of possible deflections, Fig. 6 shows the range of possible velocities that each corner can have. This means that in the worst case, there can be a corner moving as much as 1.996 m/s. The overall velocity of a 3U CubeSat's center of gravity (CG) after deployment from the P-POD is 1.993 m/s, taken from the energy method used before. The distance of the corner is 0.05 m in the x- and y-directions (nominal distance from the CG to the CubeSat face as defined by the CubeSat Standard[1] and 0 in the z-direction (parallel to the velocities). From this, the rotation of the CubeSat comes out to be 3.44°/s around the x-axis, 3.44°/s around the y-axis, and 0 around the z-axis. This is a total rotation rate of 4.86°/s. Remember that this is a back-of-the-envelope approximation to solving for the rotation and should not be considered high fidelity. The rotational kinetic energy is then found to be 10^{-4} N using the following equation.

$$KE_{rotational} = \frac{1}{2} I_x {\omega_x}^2 + \frac{1}{2} I_y {\omega_y}^2 + \frac{1}{2} I_z {\omega_z}^2 \quad (11)$$

The CubeSat standard gives a nominal value of 0.029384 kg-m^2 for the moment of inertia, *I*, in both the x-direction and y-direction. The nominal value for the z-direction is 0.005 kg-m^2, but that is not used since the rotational velocity in that axis is 0.[1] This rotational kinetic energy is negligible compared to the overall deployment energy.

Even though it was just shown that these uncertainties could be ignored, the worst possible error will be applied to the system to see how it affects the optimal result as a bound to the problem. If the error results in a significant change then the problem will be revisited.

3.4 Reference Frames

Reference frames are the largest contributing factor to high stress in an aerospace engineering student. This is because a poorly defined reference frame makes any dynamics analysis completely useless in application. To prevent any of this confusion, the two reference frames that are used in this analysis will be completely defined now.

3.4.1 Earth Centered Inertial (ECI)

The Earth Centered Inertial frame is centered at the center of the earth. This is an inertial frame, meaning that it does not rotate as the earth rotates or as it orbits the sun. Figure 7 shows the reference frame.

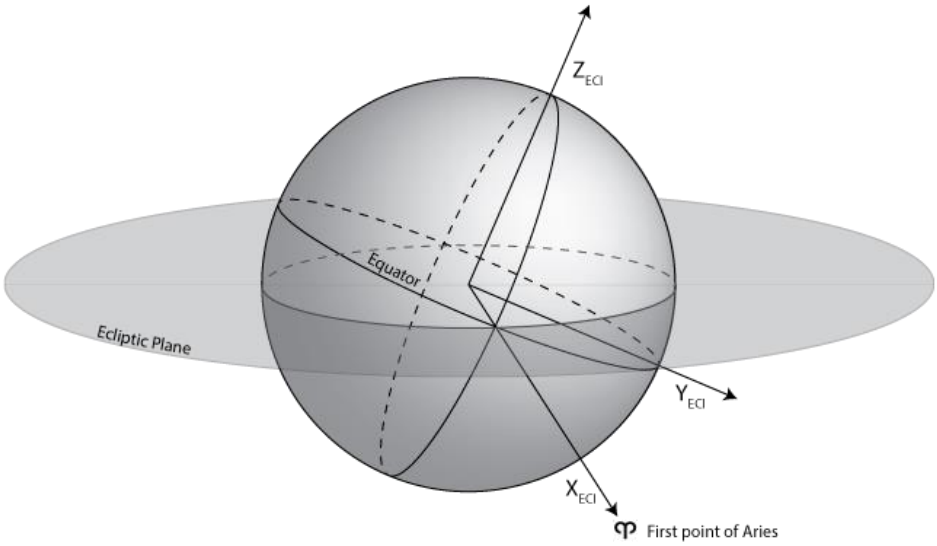

Figure 7. Earth Centered Inertial (ECI) frame.[7]

As seen above, the x-y plane is aligned with the equatorial plane of Earth. The x-direction, X_{ECI}, points in the direction of the first point of Aries. The z-direction, Z_{ECI}, is the vector from the center of Earth, through the North Pole (axis of rotation), and is perpendicular to the equatorial plane. The y-direction, Y_{ECI}, is the vector that completes the set to make a right handed Cartesian coordinate system ($Y_{ECI} = Z_{ECI} \times X_{ECI}$).

The ECI frame, however, is not used in this analysis. The inertial frame that is used is a Sun Centered Inertial frame and a Mars Centered Inertial frame. These frames are taken to be the same as ECI, and are used interchangeably, but translated to be at the center of the sun or Mars. There are other inertial frames that are centered at these bodies however they are not used to stay consistent in inertial space. This is useful in patched conics, which uses both planet centered and sun centered, heliocentric, orbits. There is an actual Mars centered inertial frame, however that is not used in order to stay consistent in the inertial frames.

3.4.2 Orbital Frame

The orbital frame is a rotating frame with respect to inertial space that has a few accepted definitions. The definition that is used here is the one given by Kane[5] and is depicted in Fig. 8.

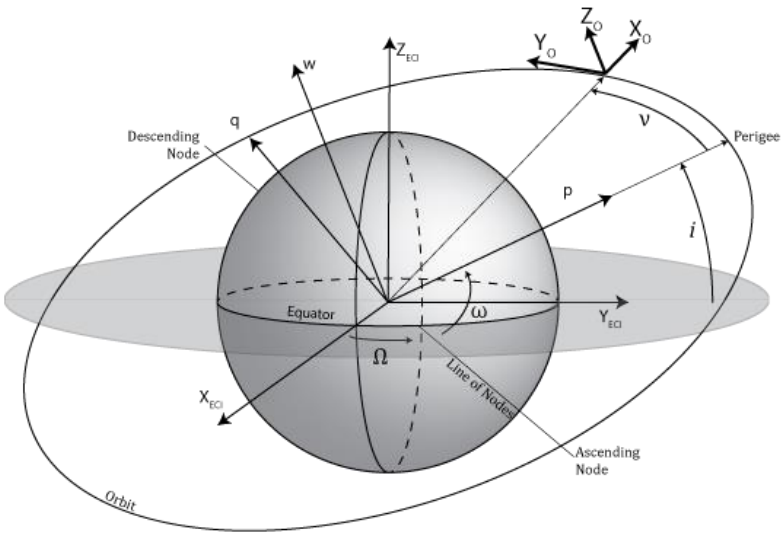

Figure 8. The orbital reference frame [x_O y_O z_O].[7]

In the orbital frame, the center is located at the center of the orbiting spacecraft and the x-y plane lines in the plane of the orbit. The x-direction, x_O, is always pointing in the direction of zenith. This means it points directly away from the center of the body it is orbiting. The z-direction, z_O, is pointing in the direction of the angular momentum vector of the orbit. The y-direction, Y_O, completes the right-handed Cartesian coordinate system ($y_O = z_O \times x_O$). In the case of circular orbits, the y-direction points along the velocity vector of the spacecraft.

3.4.3 Direction Cosine Matrix (DCM)

A direction cosine matrix, or DCM, converts a vector from one reference frame to another using direction cosines. The main DCM that is used is converting from ECI to the orbital frame, so that will be used to further explain the concept. The frame rotation is described in the equation below.

$$\vec{r}_{Orbital} = C_{ECI}^{Orbital}\, \vec{R}_{ECI} \tag{12}$$

In Eq. 12, the 3x1 (column) position vector in the ECI frame is rotated by the 3x3 direction cosine matrix to the 3x1 (column) position vector in the orbital frame. The components of the DCM are the projection of each component of the vector in the orbital frame onto the components of the vector in the ECI frame. A more detailed explanation can be found in Kane[7].

The following method shows how to create the DCM from ECI to the orbital reference frame. The basis of this DCM comes from the position and velocity vectors in a Mars or Sun centered inertial system to create the unit vectors in the orbital frame. To start, the following unit vectors are calculated.[7]

$$\vec{a}_1 = \frac{\vec{R}}{\|\vec{R}\|} \tag{13}$$

This is the direction of zenith.

$$\vec{a}_3 = \frac{\vec{R} \times \vec{V}}{\|\vec{R} \times \vec{V}\|} \tag{14}$$

This direction is parallel to the angular momentum vector and perpendicular to the orbital plane.

$$\vec{a}_2 = \vec{a}_3 \times \vec{a}_1 \tag{15}$$

This direction completes the right-hanged orthogonal set. Together, the three row vectors create the DCM as shown below.

$$C_{ECI}^{Orbital} = \begin{bmatrix} \vec{a}_1 \\ \vec{a}_2 \\ \vec{a}_3 \end{bmatrix} \tag{16}$$

Also note that to convert a vector the other direction, from the orbital plane to ECI, one can just use the transpose of the above DCM. This only works because the reference frames are orthogonal.

$$C_{Orbital}^{ECI} = \left[C_{ECI}^{Orbital}\right]^T \tag{17}$$

The aforementioned frames and DCMs are used periodically throughout the method to define the position and orientation of both the primary spacecraft and the CubeSat.

4. Deployment Optimization Design

To understand what missions are possible when deploying from the PDSD or P-POD, optimization is used to get the most out of the available ΔV. This is because CubeSat deployment is basically a ΔV maneuver that is constrained by the capabilities of the deployers described in the previous sections. Before jumping straight into high-level genetic algorithms and global optimizers that would turn this thesis into a dissertation, Matlab® has a built in constrained optimization function called *fmincon*. This function, although limited to a few optimization methods, provides enough capability to encompass the scope of this thesis. As a reminder, the overall design space is split into three categories based on the type of trajectory of the primary interplanetary spacecraft. Those cases are: flyby, orbiter, and lander. Each case will be demonstrated.

4.1 Objective Function

The objective function, or cost function, outputs the objective that the optimizer will minimize. Standard form for optimization has everything being minimized. To maximize, just minimize a negative objective function. The objective function for each case will be discussed in the corresponding section for that case. The general form is as follows:

$$obj = f(\vec{X}, paramater1, parameter2, \dots.) \tag{18}$$

where $\vec{X}$ is the vector of design variables. Any other needed parameter can be passed through as well.

4.2 Design Variables

A design variable is a variable that the optimizer manipulates to find the optimal solution. There are four design variables that are consistent for all of the cases. These are the three

orthogonal components of the deployment vector and the true anomaly of the primary at the time of deployment. The variables encompass when and how to deploy the CubeSats and are used to calculate the value of the objective function. All design variables are contained in the vector $\vec{X}$. If the design variables are bounded, the bounds can be inputted in separate vectors with corresponding indices. An example is provided below in Eq. 19.

$$\vec{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \overrightarrow{LB} = \begin{bmatrix} lb_{x1} \\ lb_{x2} \\ lb_{x3} \end{bmatrix}, \overrightarrow{UB} = \begin{bmatrix} ub_{x1} \\ ub_{x2} \\ ub_{x3} \end{bmatrix} \tag{19}$$

4.3 Constraints

The constraints are what control the nature of the problem. Constraints can come from the physics of the problem or from the design of the mission. Assuming a non-powered flyby, an example of a physical constraint is that the velocity of the spacecraft relative to the planet must be equal at the beginning and end of a non-powered flyby. This comes from the conservation of energy in reference to the planet. An example of a constraint based on mission design is that the deployment direction of the CubeSat is bounded by the attitude control system of the primary spacecraft. These constraints are neatly wrapped up in a user created Matlab® function that requires the design variables (and any other mission specific variables) as input. The output is formatted as two vectors. The first is a vector of inequality constraints that follow standard optimization form. That is $c(\vec{X}) \leq 0$. The second vector is a vector of equality constraints that also follow standard form; $c_{eq}(\vec{X}) = 0$. Note that the constraints may not be satisfied until the very end, so any intermediate solution may not be valid.

4.4 Limitations

All of the algorithms used in *fmincon* are gradient-based optimization algorithms. This means that both the objective function and the constraints must be continuous and have a continuous first derivative. Because of this, the solver may not work for every objective function. Imagine a case that had an objective function that was a combination of several orbital elements, all individually nonlinear themselves, with nonlinear weights. There exist combinations where this objective function would not have a continuous derivative. Also, gradient-based optimizers make the assumption that the objective function in unimodal, meaning that there is only one minimum. Again, it is very likely that there are certain combinations of orbital elements that will result in multimodal objective functions. This won't break the solver, but the solution that it finds will only be a local minimum and not necessarily the global minimum. The best way to account for this is to understand the nature of the objective function itself and choose an initial guess close to the global minimum. Since that is not always feasible, it is suggested to try multiple initial guesses and look for symmetry in the nature of the problem. The best solution would be to eventually develop and implement an optimization method based on genetic algorithms. Reference 3 is an example of such a genetic algorithm that treats the objective function as a sort of fitness level. The "children" with the highest fitness levels have a higher probability of "mating" and passing on its successful genes or traits that led to the high fitness score. Using this on the study could improve the optimization process. Since the scope of this thesis is focused on the potential CubeSat trajectories, more focus is spent on the analysis to create the objective functions and classify the different approaches than the actual optimization.

5. Primary Spacecraft: Flyby Trajectory

A spacecraft is on a flyby trajectory when it uses a planet or other celestial body to perform a gravity assist maneuver. During a flyby, the spacecraft has to maintain a very precise trajectory so that it slingshots around the planet towards the proper destination. This means the altitude of the flyby, the speed of the flyby, and the location above the planet are all determined by turning angle required to head towards the final destination. But flying by a planet is a great opportunity for science and should not be so constrained. Deploying CubeSats can lead to drastically different opportunities to perform science that are not so constrained. Spacecraft operators also know all too well that there is a terrifying period of time where the spacecraft actually goes behind the planet and loses all contact with Earth. Deploying a CubeSat as relay around the planet to provide telemetry or relay commands could be extremely beneficial. To find cases like these the deployment optimization technique is applied to flyby trajectories.

5.1 Hyperbolic Trajectories

This section is dedicated to defining the commonly used variables in hyperbolic trajectories. Figure 8 depicts the trajectory and all important variables and more.

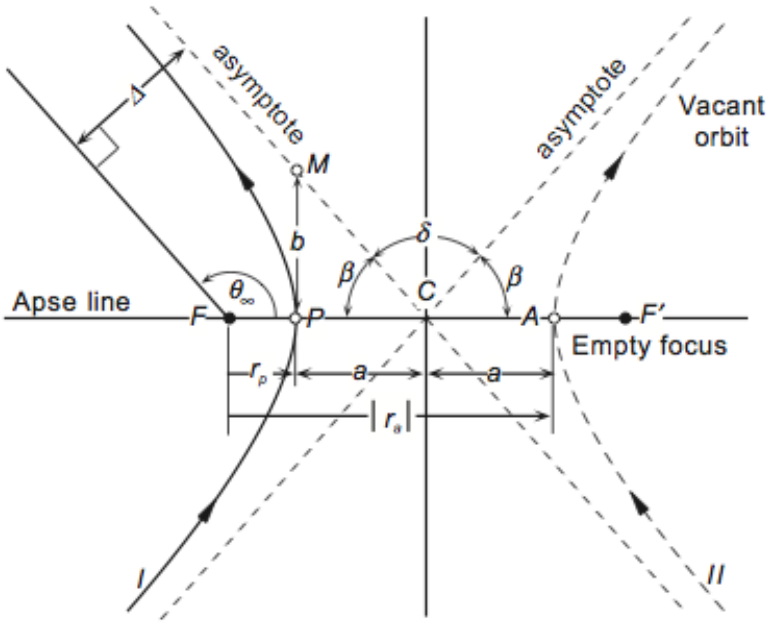

Figure 9. Hyperbolic trajectory.[2]

The hyperbolic excess speed, or V_∞, is defined as the speed of the spacecraft on a hyperbolic trajectory as it approaches an infinite distance away from the planet. The sphere of influence of the planet can be taken as an infinite distance since that is where the gravitational effect from the planet becomes so small relative to the sun that it can be neglected. The turning angle, δ, is the angle in which the V_∞ vector is turned due to the gravitational effects of the planet. The closest approach distance, r_p, is exactly as it sound and sometimes referred to as the radius of periapsis. The aiming radius, Δ, is the distance between the asymptote of the hyperbola and the planet. The last major metric is the true anomaly of the asymptote, or θ_∞. This is the true anomaly at which the orbit equation breaks down because the eccentricity of a hyperbola is greater than one and the object in an infinite distance away. It is accepted that this occurs at the sphere of influence of the planet.

5.2 Objective Function and Design Variables

The design variables for the flyby case are the four main deployment variables: the three deployment directions and the true anomaly at the time of deployment. This means the design variable vector is as follows.

$$\vec{X} = \begin{bmatrix} \Delta V_x \\ \Delta V_y \\ \Delta V_z \\ \theta \end{bmatrix} \tag{20}$$

In this case, the objective function also needs the dates at which the primary spacecraft leaves the starting planet (Earth) and arrives at the flyby planet (Mars). This means the objective function is as follows.

$$[\text{obj}] = \text{FlyByCubeSatObj}(X, JD_{\text{Earth}}, JD_{\text{Mars}}) \tag{21}$$

The objective function uses the dates in a patched conics approach to the interplanetary transfer. In this approach, the planets' locations are found using ephemeris data, and then along with the time of flight are used to solve the Lambert's problem to find the entire trajectory.

$$[v_{\text{Spacecraft1}}, v_{\text{Spacecraft2}}] = \text{UVlambert}(r_{\text{Earth}}, r_{\text{Mars}}, \Delta t, \mu_{\text{sun}}) \tag{22}$$

The V_∞ upon arrival at Mars is the velocity of Mars minus the spacecraft velocity found in the Lambert's problem. This V_∞ is the V_∞ of the primary and is what will be used as a reference of how much the nature of the flyby changes in each iteration. Furthermore, these dates are associated with θ_1 and θ_2, which constrain the deployment true anomaly to be within the heliocentric transfer orbit.

$$\overrightarrow{LB} = \begin{bmatrix} -\infty \\ -\infty \\ -\infty \\ \theta_1 \end{bmatrix}, \overrightarrow{UB} = \begin{bmatrix} \infty \\ \infty \\ \infty \\ \theta_2 \end{bmatrix} \tag{23}$$

Once the full trajectory of the primary is known, the CubeSat state is found by adding the ΔV vector, which is the first three design variables, to the state vector of the primary at the true anomaly of deployment. The state vector of the primary can be found by converting the Classical Orbital Elements, including the true anomaly at deployment, into cartesial coordinates as shown in Curtis.[2]

$$State_{CubeSat} = State_{Primary} + [0 \quad 0 \quad 0 \quad X_1 \quad X_2 \quad X_3] \tag{24}$$

This state is then propagated forward for the remaining transfer time using an ordinary differential equation solver in Matlab®. This time is found using the equation below.

$$t_{remaing} = (\theta_{UpperBound} - X_4)\frac{T}{360°} \tag{25}$$

In Eq. 25, *T* is the period of the entire heliocentric elliptical orbit that is used in patched conics. Once the CubeSat reaches the planet, the difference between the velocity of the CubeSat and the planet is the V_∞ of the CubeSat. The difference between this V_∞ and the V_∞ of the primary is the objective that needs to be maximized. A constraint is added on the deployment time to ensure that the CubeSat maintains a certain distance to the planet during the flyby. Put into standard form all of the constraints on the system are shown in Eqs. 26 and 27.

$$c = \frac{\|State_{CubeSat}(1{:}3) - R_{Mars}\|}{D_{maximum}} - 1 \tag{26}$$

$$c_{eq} = \frac{\|X(1{:}3)\|}{0.002} - 1 \tag{27}$$

5.3 CubeSat Flyby

The main concern with the flyby method is ensuring that the CubeSat still flies by the planet. In early tests, the optimizer quickly converged to deploying as close as possible to the Earth departure date of the primary. This is because the small perturbation in the velocity propagated forward to become a huge difference in position as the spacecraft travelled the long distance to Mars. The distance got so large that Mars would not gravitationally affect the CubeSat. To fix this, a constraint on this distance is applied. Naturally, the optimizer butts up against this constraint. This is because the earlier the CubeSat is deployed, the larger the difference in V_∞ vector. This means that the maximum relative distance between Mars and the CubeSat defines when the CubeSat should be deployed. Figure 10 shows the change in V_∞ due to deployment vs. the final distance between the CubeSat and Mars.

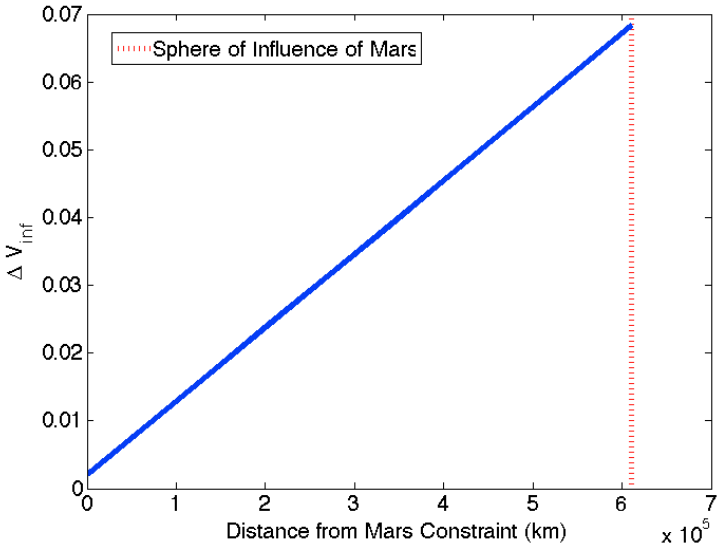

Figure 10. Distance constraint effect on hyperbolic excess speed.

The cases where the constraint on the relative distance between Mars and the CubeSat (D_{Mars} from Eq. 26) is larger allow for a larger change in V_∞. Since this is patched conics approach, the V_∞ is the only metric that is transferred over to the hyperbolic part of the flyby. Even so, it may be a good idea to stay closer to Mars so that the flyby can take some type of interesting scientific data. This means that to further demonstrate the method, the time to deploy is chosen to be the one that results in a change in V_∞ of 30 m/s. This means several things for the nature of the flyby. First, the radius of closest approach that allows for the smallest capture burn changes by 194 km. This radius is given by Eq. 28.[6]

$$r_p = \frac{2\mu}{V_\infty^2} \tag{28}$$

In the above equation, μ is the gravitational parameter of Mars and r_p is the radius of closest approach that allows for the smallest capture burn. The time of the deployment for this case is 7.5 hours before rendezvousing with Mars.

To try and truly understand how this affects the flyby, this change in V_∞ is modeled as an instantaneous burn at θ_∞ on a hyperbolic trajectory. To do this, first a hyperbolic trajectory is defined.

Table 2. COEs of the Primary Spacecraft in a hyperbolic trajectory around Mars.

Classical Orbital Elements	Orbit
Specific Angular Momentum (*h*)	30000 km^3/s^2
Inclination (*i*)	0°
Argument of Periapsis (ω)	0°
Right Ascension of Ascending Node (RAAN, Ω)	0°
Eccentricity (*e*)	2.9

Closest Approach (r_p)	4000 km
Aiming Distance (Δ)	7419 km
Turning Angle (δ)	40.3°

The COEs of this orbit are chosen such that the altitude of closest approach is about 4000 km. The trajectory is shown below in Fig. 11.

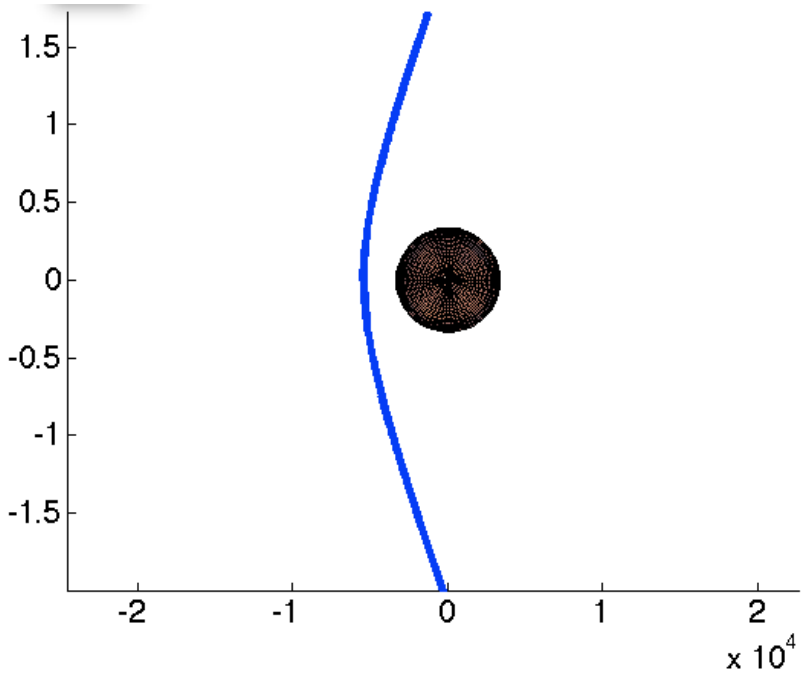

Figure 11. Mars flyby visualization

The burn of 30 m/s is applied to this orbit at θ_∞ in the opposite direction of the velocity vector. As Fig. 12 shows, the results are substantial. The CubeSat actually flies by the other side of the planet than the primary. This means that with some further research and some extra constraints, the CubeSat may be able to achieve specific metrics in the flyby. For example, if a constraint is added on the turn angle, then the deployment time will have to adjust so that the ΔV_∞ makes the turn angle satisfy the constraint. One of these flyby metrics can also be changed to be the objective function itself. More on this can be

found in the future work section. Table 3 then shows the metrics of the CubeSat flyby from this test. As mentioned, this is a drastic change in the trajectory. In heliocentric terms, the primary gains energy through the flyby where the CubeSat loses energy with respect to the orbit before the flyby.

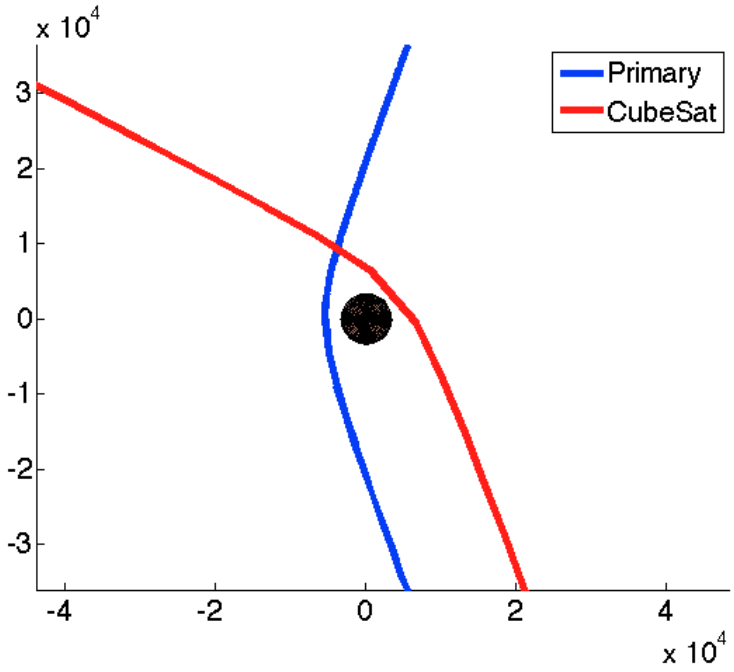

Figure 12. CubeSat flyby. The change in V_∞ is modeled as an instantaneous burn at θ_∞

Table 3. COEs of the CubeSat in a hyperbolic trajectory around Mars.

Classical Orbital Elements	Orbit
Specific Angular Momentum (*h*)	29121 km^3/s^2
Inclination (*i*)	0°
Argument of Periapsis (ω)	0°
Right Ascension of Ascending Node (RAAN, Ω)	0°
Eccentricity (*e*)	2.77
Closest Approach (r_p)	5250 km

Aiming Distance (Δ)	7600 km
Turning Angle (δ)	42.3°

5.4 Attempt at Capturing the CubeSat

It is fairly intuitive that there is not enough energy in a canister type deployer to insert a CubeSat into a capture orbit when the primary is on a flyby trajectory. To show just how much ΔV would be required, the minimum ΔV for capture burn at closest approach into a circular orbit is calculated as shown by Kaplan[6].

$$\Delta V_{min} = \frac{\left\|\vec{V}_\infty\right\|}{\sqrt{2}} \tag{29}$$

The lowest ΔV for capture would correspond to the lowest V_∞. To find the lowest V_∞, a new objective function is created. This objective function uses the Julian date of departure from Earth and the Julian date of the arrival at Mars as the design variables. The solution to the Lambert's problem again provides the heliocentric velocity, and then V_∞ is the difference between that velocity and the velocity of Mars. This time, this alone is the objective to be minimized. The dates of arrival and departure are constrained by the synodic period of Earth and Mars, which is the length of time it takes for Mars to complete a full revolution of its orbit relative to Earth and realign. This value is 780 days. The resulting trajectory results in a minimum V_∞ at Mars of 2.36 km/s with a time of flight of 224 days. This then corresponds to a minimum ΔV for capture of 1.67 km/s into a 15000 km circular orbit. This is magnitudes more ΔV than available from the CubeSat deployers and is just not possible, except in very specific situations. Even if the deployer could generate enough ΔV, Newton's 3rd law means that the same force that propelled the CubeSat would be applied on the Primary. This would greatly alter the course of the

primary on its flyby. If used correctly it could be beneficial, but that is research for another day. Too add to this, there are more opportunities that just the possibilities coming out of the deployer. As shown by the Mars Reconnaissance Orbiter (MRO), using deep space maneuvers can slightly lower the necessary burn for capture. MRO actually lowered the capture burn to around 0.9 km/s[11]. This is still too high for the P-POD alone, however utilizing other opportunities, such as aerocapture, could be worth researching

5.5 Thoughts on Landing the CubeSat

This situation is very similar to the upcoming lander case. Because of this, no extra cases where the primary is on a flyby trajectory and the CubeSat is on a landing trajectory are run.

6. Primary Spacecraft: Orbiter

In this case the primary spacecraft is orbiting a planet when the CubeSat is deployed. The primary spacecraft would have to be in extreme orbits to even consider deploying a CubeSat into either a trajectory to the planets surface or a trajectory to leave the planet. For this reason only deploying into a different elliptical orbit is considered. CubeSats already do this regularly around Earth but the deployers are typically attached to the launch vehicle itself and not the primary. An exception would be when CubeSats are deployed from the International Space Station (ISS), in which this analysis is extremely beneficial in understanding how the CubeSat orbit differs from the ISS orbit. The following details how the orbiter case is approached along with a few examples to prove the method.

6.1 Objective Function and Design Variables

The design variables for the orbiter case are again the four main deployment variables: the three deployment directions and the true anomaly of the deployment. Again, the design variables are as follows.

$$\vec{X} = \begin{bmatrix} \Delta V_x \\ \Delta V_y \\ \Delta V_z \\ \theta \end{bmatrix} \tag{30}$$

The deployment directions are constrained so that the norm of the deployment vector is the ΔV of the deployer. The deployment magnitudes are bounded by the capabilities of the deployer and the true anomaly is bounded to be within 0° and 360° to avoid the repeated solutions that would occur every orbit. This is shown in Eqs. 31 and 32.

$$\overrightarrow{LB} = \begin{bmatrix} -0.002 \\ -0.002 \\ -0.002 \\ 0 \end{bmatrix}, \overrightarrow{UB} = \begin{bmatrix} 0.002 \\ 0.002 \\ 0.002 \\ 360 \end{bmatrix} \quad (31)$$

$$c_{eq} = \frac{\|X(1{:}3)\|}{0.002} - 1 \quad (32)$$

With the classical orbital elements (COEs) of the primary spacecraft known, and the true anomaly to specify the place in the orbit, the primary orbit is fully defined. Using the definitions of each COE, geometry, and the orbit equation, the COEs are converted to a state vector as shown in Curtis.[2]

$$[\text{state}] = \text{COEs2RV}(\text{COEs},\mu) \quad (33)$$

The state vector consists of the position and velocity of the spacecraft relative to the body that it is orbiting. The state vector of the CubeSat is then the state vector of the primary plus the remaining three design variables, just as before.

$$State_{CubeSat} = State_{Primary} + [0 \quad 0 \quad 0 \quad X_1 \quad X_2 \quad X_3] \quad (34)$$

Converting this state vector back to the COEs is the final step before calculating the objective function. This process is detailed in Curtis.[2]

$$[\text{COEs}] = \text{rv2coes}(\text{state},\mu) \quad (35)$$

The objective function for this case is the difference between one or more of the classical orbital elements of the primary and the CubeSat. How the objective function is specifically defined is mission specific.

6.2 Orbiting CubeSat

To demonstrate the effectiveness of this method three subcases are tested. The first two will have the primary spacecraft in the same orbit with different objective functions. The next test will use the same objective function but a different orbit for the primary

spacecraft. All three tests are repeated but with different initial guesses for the design variables. The following are the two orbits that are defined as the primary spacecraft orbiting Mars (see Table 4 and Fig. 13).

Table 4. COEs of the Primary Spacecraft orbiting Mars.

Orbit 1: Classical Orbital Elements	Orbit 1	Orbit 2
Specific Angular Momentum (*h*)	35000 km^3/s^2	30000 km^3/s^2
Inclination (*i*)	70°	5°
Argument of Periapsis (ω)	20°	2°
Right Ascension of Ascending Node (RAAN, Ω)	50°	210°
Eccentricity (*e*)	0.3	0.05

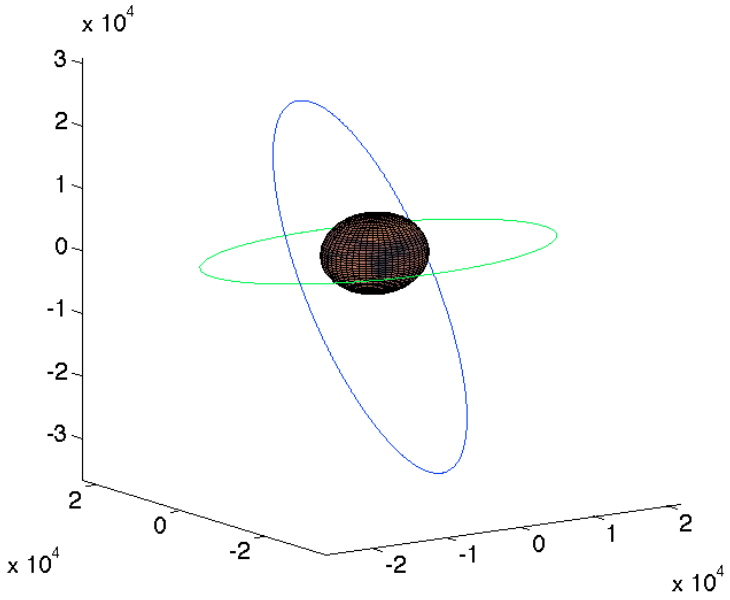

Figure 13. Visualization of Orbit 1 (blue) and Orbit 2 (green) around Mars.

For the first test of this imaginary mission it is deemed important to have a CubeSat in a different inclination. This will demonstrate the method's ability to maximize the change in inclination. For this reason the objective function is set to the difference in the

inclination between the primary and the CubeSat. The objective is negated to maximize this difference.

$$\Delta i = |COEsP.inc - COEsC.inc| \tag{36}$$

$$obj = -\Delta i \tag{37}$$

The second mission objective is to change the eccentricity of the CubeSat's orbit. This will show that the method works on other objective functions. The objective function is again negated to maximize the difference.

$$\Delta e = |COEsP.ecc - COEsC.ecc| \tag{38}$$

$$obj = -\Delta e \tag{39}$$

These are simple cases since the objective function is only a function of one orbital element. The following two initial guesses are used in each case. Note that ΔV components are in the inertial reference frame. These guesses are sufficiently different from each other so that, if they exist, any local minima will be found. If local minima are not found with these two guesses, it is suggested to add more guesses until the user is sufficiently satisfied that there are no local minima. It is not easy to rule out all local minima so the best approach is to try and understand the nature of the objective function.

$$\vec{X}_0 = \begin{bmatrix} \Delta V_x \\ \Delta V_y \\ \Delta V_z \\ \theta \end{bmatrix} = \begin{bmatrix} 0 \ km/s \\ 0 \ km/s \\ 0.002 \ km/s \\ 200° \end{bmatrix} or \begin{bmatrix} 0.002 \ km/s \\ 0 \ km/s \\ 0 \ km/s \\ 15° \end{bmatrix} \tag{40}$$

6.2.1 Test Case 1

This first test case uses the first primary orbit, first objective function, and both initial guesses. The results of these tests are reported in Table 5. The first three design variables are reported as both unit vectors in the mars centered inertial reference frame and unit

vectors in the orbital frame of the primary. The orbital frame is in parentheses. These solutions would be used as the directions to point the deployer. The fourth design variable is the true anomaly of the primary at deployment; meaning when in the orbit the CubeSats should be deployed. The final row in the table is the inclination change achieved when deploying under these conditions. This is the maximum solution that this method found.

Table 5. Test case 1 results.

Optimal Results	Guess 1	Guess 2
V_x	0.7191 (0)	-0.7193 (0)
V_y	-0.6037 (-0.002)	0.6041 (-0.001)
V_z	0.3442 (0.999)	-0.3431 (-0.999)
$\boldsymbol{\theta}$	165°	0°
$\boldsymbol{\Delta i}$	0.1314°	0.0677

It is clear that this case demonstrates one of the limitations discussed earlier. There are two local minima that exist in this design space. Using the suggestions described before, the first thing to notice is the symmetry. The deployment directions are opposite in the two local minima. Since the objective function is a plane change, the optimizer finds two minima that would result in moving the plane in the same direction. To understand the symmetry and investigate this further, multiple initial conditions are tested and plotted. The initial guess for the first three design variables are held constant and the fourth is varied between 0 and 360. The results show three distinct minima.

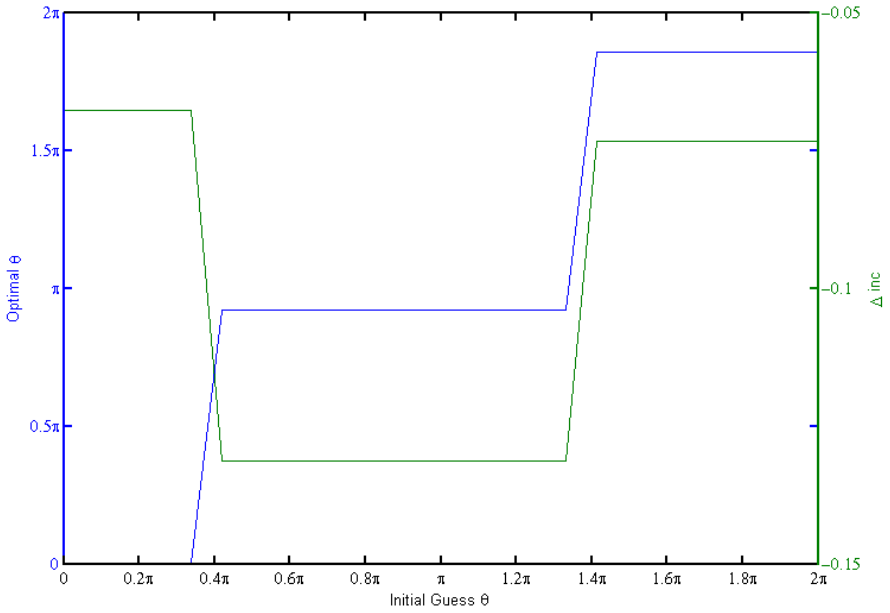

Figure 14. Vary the initial guess to look for local minima in Test Case 1

Figure 14 verifies the symmetry that the first two initial guesses showed as visualized by the fact that each initial guess converges to one of three distinct "bins" which represent three local minima. In reality, two of these local minima are the same such that the two real local optimal θ for deployment exist at 0 (with the same one at 360) degrees and what should be 180 degrees. This makes sense from an orbits point of view since those two points (periapsis and apoapsis) are when the spacecraft is moving the fastest and slowest respectively. It then makes sense that of the two minima, the one at apoapsis would be the global minimum. This is because when the spacecraft is moving slower, the ΔV from the deployer is a larger overall percentage of the velocity. This test case also shows the shortcomings of the gradient-based optimizers because the solutions are not exactly 0, 180, and 360 degrees. This needs to be explored in further research and is possibly due to propagating the state vector as opposed to the orbital elements.

Since the global minimum was expected to be at 180 degrees (and not 165) the value of the objective function is evaluated at that point. The Δi when deploying at 180 degrees is 0.1257 degrees. Since this is not as high as the objective at 165 degrees the optimizer moves away from it, verifying that the optimizer is working. To explore this further, a map of the objective function and constraints is created to better understand how the objective function changes with respect to each design variable. Since there are four design variables, there are not enough physical dimensions to vary all of them in one plot. This means that two are held constant and the other two are varied to create the contours shown in Fig. 15. The contour maps with the deployment true anomaly show the local minima associated with this objective function are located in distinct loops. An initial guess inside one of these loops would lead to that local minimum, just like the bins in Fig. 14. Again, the nature of the objective function looks correct, however it seems shifted or biased slightly towards 165 instead of 180. The plots in Fig. 15 that only vary the deployment directions show how there would not be local minima with respect to these variables. The objective function is linear and simply finds the minimum that satisfies the constraint in each case.

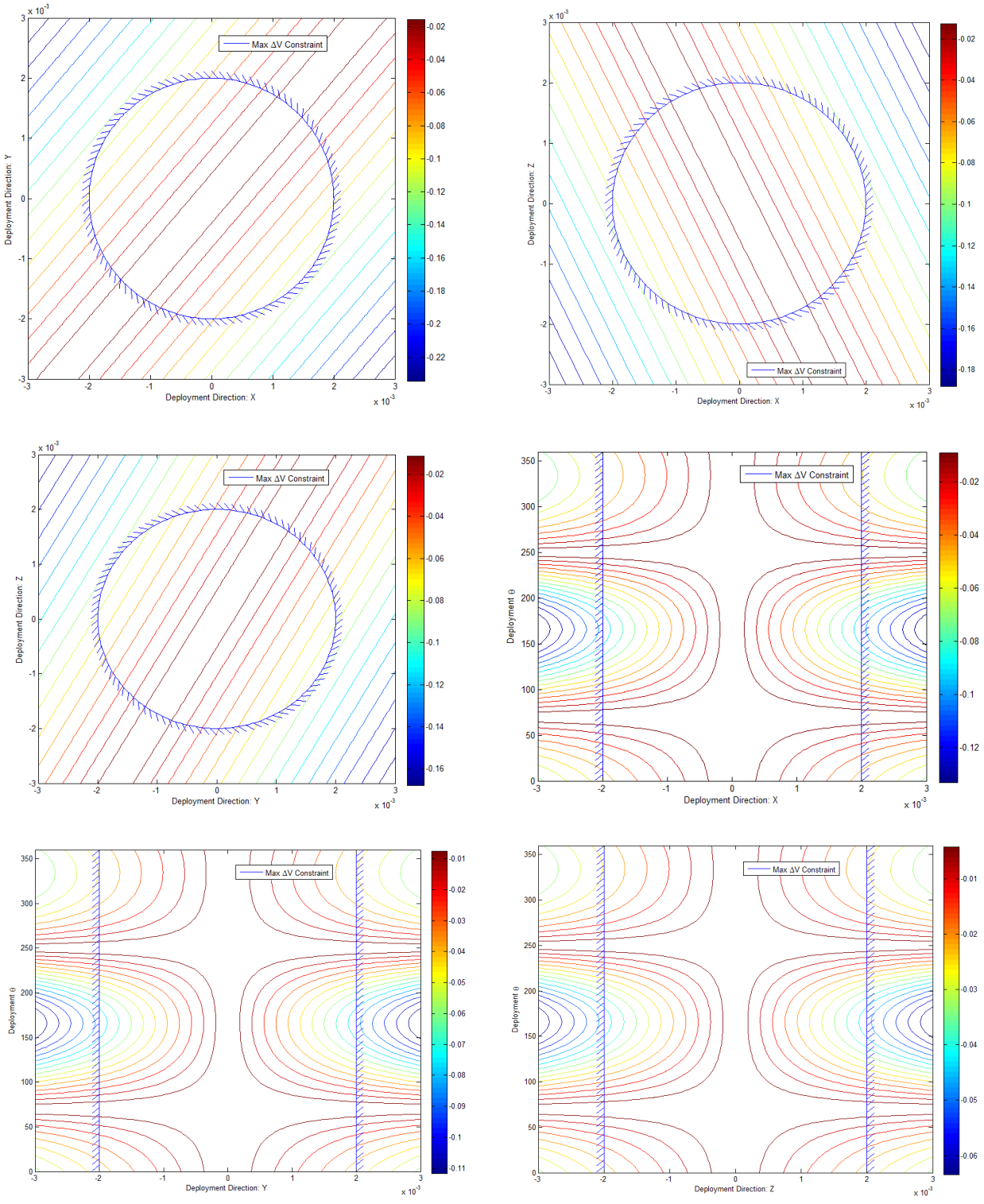

Figure 15. Six contour maps of the objective function. Each sector represents a combination of two of the four design variables

Overall, the magnitude of the change in orbit is very small. This may be discouraging, since it seems that the CubeSat is still stuck on the same trajectory of the primary. To prove that this tool is important, the optimal solution is compared to the solution in the first iteration. The value of the objective function on the first iteration is 0.04°. This is an order of magnitude less than the optimal solution. An order of magnitude is a large difference, especially considering that this orbit is not very elliptical. If the primary orbit had a higher eccentricity, say 0.7 or more, then the difference between the optimal solution and any other could be much more significant. Also, if the deployer is modified for more ΔV then the significance of that order of magnitude is increased.

6.2.2 Test Case 2

Test Case 2 is the same as Test Case 1 but with the other objective function, Δ*e*. The results are presented in the table below.

Table 6. Test case 2 results. An additional guess is made at X_0 = [0 0 0.002 300].

Optimal Results	Guess 1	Guess 2	Guess 3
V_x	0.4460 (0.000)	0.4460 (0.000)	0.4460 (0.015)
V_y	0.0554 (0.999)	0.0554 (0.999)	0.0554 (-0.999)
V_z	-0.8830 (0.000)	-0.8830 (0.000)	-0.8830 (0.000)
$\boldsymbol{\theta}$	180°	180°	360°
$\boldsymbol{\Delta e}$	-0.0033 (1%)	-0.0033 (1%)	-0.0033 (1%)

With the first two guesses it appears that this objective function does not have the same problem with local minima as the other. However, because local minima are highly likely when using gradient-based optimizers and the modality of the objective function is not known, an additional guess is added with a true anomaly at 360° as suggested. Sure enough, this extra guess found a local minimum at the other side of the orbit. Also, the deployment direction is the same in inertial space instead of opposite like in Test Case 1, even though the values at the minimum are practically equal. Again, trying to understand the solution calls for understanding the nature of the objective function. To change the eccentricity of the orbit, it makes sense to change the velocity only in the plane of the orbit. Near periapsis, the spacecraft needs to slow down to increase its eccentricity, meaning apply the ΔV in the opposite direction of its motion. At apoapsis, the spacecraft needs to speed up to circularize, meaning apply the ΔV in the same direction of its motion. These two maneuvers are in the same direction in inertial space because the spacecraft is moving in opposite directions at periapsis and apoapsis.

Once again, the resulting change in orbit due to deployment is small. This is because the fact that there is just so little ΔV in these canister deployers is unavoidable. Regardless, to show that deploying at the optimal solution can be worth it, the optimal solution achieves a change in eccentricity that is three times larger than the initial guesses. Once again, if the ΔV from the deployer were increased, the optimal solution would be able to achieve a lot more.

6.2.3 Test Case 3

The final test case will go back to the first objective function but will change the orbit of the primary spacecraft to orbit two. Once again, the results are presented in Table 7.

Table 7. Test case 3 results.

Optimal Results	Guess 1	Guess 2
V_x	-0.0428 (0)	0.0428 (0)
V_y	0.0742 (-0.001)	-0.0742 (0.001)
V_z	0.9963 (-0.999)	-0.9963 (0.999)
$\boldsymbol{\theta}$	178°	0°
$\boldsymbol{\Delta i}$	0.0845°	0.0764°

Once again, the method finds two optimal solutions. And again they are on opposite sides of the orbit, and deploying in opposite directions. This verifies that both the strengths and weaknesses of the method are not dependent on either the orbit of the primary, the objective function, or the initial guess.

6.3 Deployment Uncertainties

As mentioned previously, a 4% reduction of the deployer's ΔV capability is applied to the system to show that it is indeed negligible. Test Case 1 is used to compare the nominal ΔV to the reduced ΔV. The COEs of the CubeSat when optimally deployed at the nominal levels and the reduced levels, as well as the difference, are shown in Table 8.

Table 8. Comparing COEs for nominal and reduced ΔV capabilities.

Classical Orbital Elements	Nominal Case (100%)	Reduced Case (96%)	Difference
Specific Angular Momentum (*h*)	35081 km^3/s^2	35080 km^3/s^2	-1 km^3/s^2
Inclination (*i*)	69.9°	69.9°	5e-4°
Argument of Periapsis (ω)	20°	20°	1e-3°

RAAN (*Ω*)	49.9°	49.9°	2e-4°
Eccentricity (*e*)	0.29	0.29	5e-5

These deltas are smaller than the errors that would arise both computationally as well as those that exist due to perturbations in the orbits model.

To further demonstrate that the deployment uncertainties are negligible, the gradient of the objective function can be calculated using finite differencing to show that small changes do not drastically change the optimal location.

6.4 Landing Trajectories and Hyperbolic Trajectories

As mentioned previously, it is highly unlikely that there is enough energy in a CubeSat deployer to allow the CubeSat to either land or leave the planet. For reference, the typical escape velocity to leave Earth is 11 km/s. As for landing, the CubeSat has to come to a complete stop so all or most of the energy needs to be removed from the system. That is a lot of energy when the spacecraft is traveling at least 3 km/s and is thousands of kilometers above the surface. It would take extreme orbits for this to happen so these cases are not demonstrated but the reader is encouraged to explore these possibilities.

6.5 CubeSat with more ΔV

It was expected that the change in orbit due to deployment would be small in the orbiter case. It is unavoidable that the ΔV from the current canister type deployers is just not enough energy to drastically change the orbit. This raises the question on just how much ΔV is required to change the orbit. To figure this out, the inputs and output of the orbiter case are flipped. Now the input is the desired change in each orbital element and the

output is the required ΔV. Test Case 1 is used to demonstrate this approach. Recall from Test Case 1, the orbit of the "mother" spacecraft (instead of primary) is as follows.

Table 9. COEs of the Mother Spacecraft orbiting Mars.

Orbit 1: Classical Orbital Elements	Orbit 1
Specific Angular Momentum (*h*)	35000 km^3/s^2
Inclination (*i*)	70°
Argument of Periapsis (ω)	20°
Right Ascension of Ascending Node (RAAN, Ω)	50°
Eccentricity (*e*)	0.3

This time, the mission designer knows that putting a "daughter" CubeSat as a relay in a similar orbit but with a 5° difference in inclination would increase the time the spacecraft could be in contact with the rovers on the surface. Since the deployers cannot get the CubeSats there on their own, the mission designer needs to know how much extra ΔV they need. Whether the CubeSat needs its own propulsion or if the primary provides it is another matter. The results of this case are in Table 10.

Table 10. A modified Test Case 1 for extra ΔV results.

Optimal Results	Guess 1	Guess 2
V_x	0.7335 (0)	-0.1101 (-0.013)
V_y	-0.6095 (0.044)	0.0870 (0.046)
V_z	0.3007 (0.999)	-0.0450 (-0.999)
$\boldsymbol{\theta}$	166°	0°

$\boldsymbol{\Delta V}$	0.076 km/s	0.147 km/s
$\boldsymbol{\Delta i}$	5°	5°

As always, there are the local minima, but in this case one is clearly a global minimum. This means the user can request a 5° inclination change and the tool will report that the CubeSat needs to deploy in the optimal direction and true anomaly as before, but also with a minimum ΔV of 76 m/s. This extra 74 m/s would need to either come from the primary, CubeSat propulsion, or from design changes to the deployer itself. Either way, it can be used to understand what is required to achieve certain missions during the mission design phase. This leads to more of the mother/daughter class CubeSat missions as opposed to the primary/secondary classes that are currently used.

7. Primary Spacecraft: Lander

When the primary spacecraft is a lander, the dynamics of the problem completely change. The atmosphere of the planet, shape of the CubeSat, weather, and many other variables become huge factors. There are lifting entry trajectories, ballistic trajectories, heating concerns, impact velocities, and planetary protection to worry about. Since multiple entire theses can begin at the Karman line (altitude where the atmosphere begins), this thesis will end at the Karman line. Thus, the entry trajectory is modeled as a hyperbolic trajectory with a perigee radius equal to the radius of the planet. The optimization approach is then applied to some test cases to get an idea of where CubeSats can go.

7.1 Objective Function and Design Variables

The design variables for the lander case are once again, the three deployment directions and the true anomaly of deployment.

$$\vec{X} = \begin{bmatrix} \Delta V_x \\ \Delta V_y \\ \Delta V_z \\ \theta \end{bmatrix} \tag{41}$$

The difference with this case is that the true anomaly has to be bounded between 0 and θ_∞ to ensure that orbit equation does not break due to the hyperbolic trajectory.

$$\overrightarrow{LB} = \begin{bmatrix} -0.002 \\ -0.002 \\ -0.002 \\ 0 \end{bmatrix}, \overrightarrow{UB} = \begin{bmatrix} 0.002 \\ 0.002 \\ 0.002 \\ \theta_\infty \end{bmatrix} \tag{42}$$

The objective function then looks very similar to the orbiter case before diverting towards an entry trajectory metric. The method takes in the trajectory of the primary and uses the deployment true anomaly to get the state vector at that time. The CubeSat state vector is then found by adding the ΔV vector. This state vector is used to find the COEs of the

CubeSat trajectory. This is where the landing case differs from the orbiting case. Using the orbit equation, the true anomaly is found for when the CubeSat crosses the Karman line. This is accepted to be 80 km above the Martian surface based on the density profile provided by JPLs Mars Pathfinder mission.

$$\theta = \cos^{-1}\left(\frac{h^2}{(80+Radius_{Mars})\mu e} - \frac{1}{e}\right) \tag{43}$$

The CubeSat state is then found at this true anomaly, which is used to calculate the flight path angle of the CubeSat using Eqs. 44, 45 and 46.

$$v_{\perp} = \frac{h}{(80+Radius_{Mars})} \tag{44}$$

$$v_{raial} = \frac{\mu e \sin\theta}{h} \tag{45}$$

$$\gamma = \tan^{-1}\frac{v_{raial}}{v_{\perp}} \tag{46}$$

The flight path angle is also found for the primary. The difference between these two angles is the objective for this example. From here a re-entry aerodynamics analyst could use this change in flight path angle to calculate how the landing point, max deceleration, and max heating metrics changed and use those as objective functions instead. For more information reference the methods described by Hankey.[4]

7.2 CubeSat Lander

Two test cases are used to demonstrate the effectiveness of this approach on landing trajectories. These trajectories lie in the same plane and the primary lands in the same location, however the eccentricities of the trajectories are very different. This accounts for a shallow lifting type entry method as well as more direct ballistic approaches. Table 11 lists the COEs for these trajectories and Fig. 16 visualizes them.

Table 11. COEs of landing trajectories to Mars.

Classical Orbital Elements	Orbit 1	Orbit 2
Specific Angular Momentum (*h*)	19052 km^3/s^2	29515 km^3/s^2
Inclination (*i*)	0°	0°
Argument of Periapsis (ω)	0°	0°
Right Ascension of Ascending Node (RAAN, Ω)	0°	0°
Eccentricity (*e*)	1.5	5
Closest Approach (r_p)	3390 km	3390 km
True Anomaly of the asymptote (θ_∞)	131°	101°
Flight Path Angle at Karman Line (γ)	9.5°	11.3°

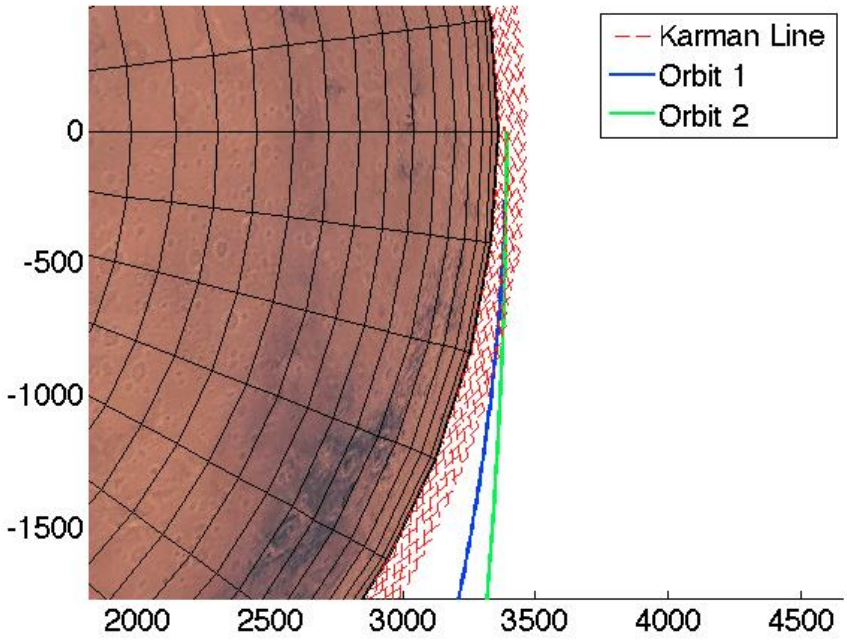

Figure 16. Landing trajectory test cases

Due to the abrupt end of the trajectory at the natural axis of symmetry of a hyperbola, local minima are not expected in the lander case. The results are in Table 12. A

visualization of these results is then presented in Fig. 17 for Orbit 1 followed by Fig. 18 for Orbit 2.

Table 12. Landing trajectories to Mars results. Inertial frame (orbit frame)

Optimal Results	Orbit 1	Orbit 2
V_x	0.0016 (0.0197)	0.0016 (0.0197)
V_y	0.0012 (-0.9998)	0.0012 (-0.9998)
V_z	0.000 (0.0000)	0.000 (0.0000)
$\boldsymbol{\theta}$	126.8°	96.5°
$\boldsymbol{\Delta\gamma}$	2.6°	0.8°

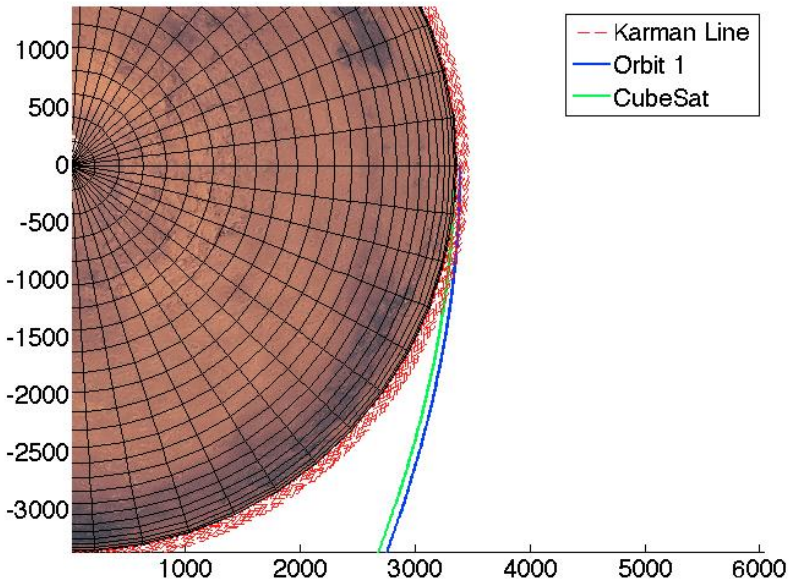

Figure 17. Orbit 1 lander results

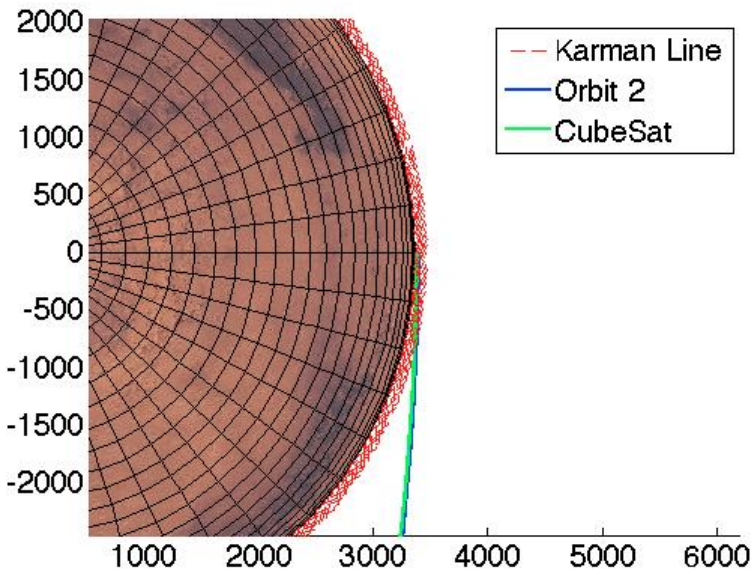

Figure 18. Orbit 2 lander results

Orbit 1 clearly has a much more drastic change in the flight path angle than Orbit 2. This is not that surprising since Orbit 1 has less energy, meaning the ΔV applied is a larger portion of the total energy in the system. Still, 2.6° drastically changes the entry dynamics and can significantly change the landing location on the planets surface. Another interesting phenomenon is that for both orbits the optimal deployment conditions are exactly the same: deploy in the opposite direction of motion and as close to θ_∞ as possible. This means that deployment before encountering the planet, similar to the flyby case, could mean more ΔV to change the flight path angle. The mission designer would just have to ensure that the CubeSat still lands on the planet through additional constraints.

7.3 CubeSat Orbiter

For this study, when the primary is landing on the planet it is modeled as a hyperbolic trajectory with the closest approach being the radius of the planet. This means that for the CubeSat to orbit the planet, all of the hyperbolic excess speed needs to be lost. This is a minimum of 2 km/s which would require a highly specific orbit. For this reason, this situation is not considered in this thesis. However, as previously mentioned, there is a great opportunity for exploring aerocapture CubeSats in this case.

7.4 CubeSat Flyby

As mentioned previously, the flyby case and the lander case for the primary are very similar. When the goal is a CubeSat flyby, use the approach assuming that the primary is also a flyby that has a closest approach that is very low. There would have to be a constraint to miss the atmosphere as well. This would allow for maximum ΔV since the flyby case looks to change flyby metrics instead of landing metrics.

8. Conclusions

8.1 Thesis Objective Review

Recall that the main objective of this thesis is to shift the mindset behind some of the current CubeSat limitations. CubeSats have followed in the footsteps of larger class satellites since the beginning at there is no reason for it to stop with interplanetary missions. This thesis aimed to introduce a method that can be used to best utilize the low thrust capabilities from the deployers.

8.2 Summary

The methods introduced for each type of interplanetary trajectory still have plenty of room for development. But the chosen examples show that there is a lot to gain from optimizing CubeSat deployment. The approach starts with the trajectory of the primary and four design variables to calculate a mission specific objective function. To demonstrate the strong and weak points of the method, multiple cases were tested. In the flyby case, with an objective function to maximize the change in V_∞, the small ΔV from deployment grows into a larger ΔV_∞ such that the CubeSat actually heads back towards Earth while the primary continues its trajectory into the outer solar system. In the orbiter case, multiple subcases with different objective functions, primary spacecraft, and initial guesses showcased both the strength and weaknesses of the optimizer. These subcases demonstrated how optimization creates a change in inclination that is an order of magnitude greater than the non-optimal case. Even more so, optimization shows that CubeSats and primaries can benefit greatly by evolving into a support-type mission instead of just a secondary mission. A small boost from the primary or newly developed

propulsion system can drastically move the orbit. And even without help, the optimum deployment configuration allows for up to an order of magnitude greater change than the suboptimal configurations Then in the landing case, there are so many design changes that can be made knowing that the flight path angle can change by over 2°. CubeSats can aim for the entry corridor that allows for the lowest heating to increase the mission duration, or aim to land on the top of a mountain and act as a relay for a rover. Entire theses can come from exploring each one of these cases further and developing very mission specific objective functions. These test cases also show the need for improvement. For some objective functions the solution is very dependent on the initial guess. Overall, the method is very promising. With further development the fundamental idea that simply starts with deployment design variables can evolve into a highly capable tool the opens doors for new missions.

8.3 Future Work

In general, all methods can benefit from a global optimization algorithm. The gradient-based methods are great to get an idea of what is happening but with the highly nonlinear nature of orbital mechanics it is just not feasible to always understand where local minima may occur.

8.3.1 Future Work: Flyby

After competing this method, the biggest obstacle for the flyby was using the patched conics approach. Patched conics is a great approach for rough calculations and ΔV numbers for interplanetary missions, however the assumptions that are made create discontinuities at the exact locations in which the optimal solution would likely be found.

To fix this problem, true trajectories of the primary need to be used as inputs. This is not too difficult but it requires defining more parameters of the mission. These parameters could be the turning angle or closest approach to the planet. This is far too specific for this thesis and these missions can be explored as theses on its own. To do this, fully define the trajectory of the primary, apply the deployer's ΔV to the state vector of the primary, and change the objective function to one of the hyperbolic parameters. Another fascinating objective function would be the change in the B-plane of the flyby. This could create a completely different ground track without completely changing the flyby. This way the CubeSat would have a small window after the flyby to send its data to the primary instead of having to solve that communications problem on its own.

8.3.2 Future Work: Orbiter

Mission designers most likely do not know which orbital elements they would want as the objective function for this case. In simple missions it might be as easy as maximizing inclination change, but in complex missions where the real goal is complex, like the maximum access time over a certain location on the ground, the objective function is not as clear. The accepted approach to choosing the objective function is to create a pareto frontier to help the designer properly weigh each objective. It is easy to imagine how complex these objective functions may become. Because of this, the orbiter case is the one that would benefit the most from a genetic algorithm or other global optimization approach. Other improvements can come from wrapping a Monte Carlo approach around the whole system as a way of choosing initial guesses.

8.3.3 Future Work: Lander

The future work on the lander method is probably the most exciting and complex. Stopping the objective function at the Karman line is simply a result of the vast number of directions that the objective function can take from that point. Deployables can change the lift-to-drag ratios of the CubeSat and then optimize for range. CubeSats can attempt skip-entry methods and set the number of skips as the objective function. Minimizing g-loads is an obvious path forward. And this thesis does not even consider deploying CubeSats during the primary's Entry-Descent-and-Landing (EDL) phase. The JPL's Curiosity rover actually needed to drop mass during its EDL phase. Those could have been CubeSats!

As a final note, all the code developed in this thesis is available through the Aerospace Engineering Department at Cal Poly. Specifically, Dr. Kira Abercromby.

References

[1] "Cal Poly CubeSat Program." N.p., n.d. Web. 10 Apr. 2015.

[2] Curtis, H. D. *Orbital Mechanics for Engineering Students*. 2nd edition. Elsevier Ltd., 2010.

[3] Gad, A. H. "Space Trajectories Optimization Using Variable-Chromosome-Length Genetic Algorithms". Mechanical Engineering Dissertation, Michigan Technological University, 2011.

[4] Hankey, W. L. *Re-Entry Aerodynamics* American Institute of Aeronautics and Astronautics Inc.,Washington, 1988.

[5] Kane, T. R. *Spacecraft Dynamics*. Internet-First University Press, Ithaca, NY, 2005.

[6] Kaplan, M. H. *Modern Spacecraft Dynamics & Control.* John Wiley & Sons Inc., 1976.

[7] Mehrparvar, A. "Attitude Estimation for a Gravity Gradient Momentum Biased Nanosatellite". Aerospace Engineering Master's Thesis, California Polytechnic State University, San Luis Obispo, CA, 2013.

[8] U.S. Army, Material Command. "AMCP 706-150: Interior Ballistics of Guns" Feb. 1965.

[9] Vallado, D. *Fundamentals of Astrodynamics and Applications*. 4th edition. Microcosm Press, Hawthorne, CA, 2013.

[10] Wolfenbarger, N. "CubeSat Orbit Prediction and Simulation from Deployment Conditions". Aerospace Engineering Senior Project, California Polytechnic State University, San Luis Obispo, CA, 2013

[11]Lock, R. Committee Member. NASA JPL.