Interplanetary Flight Using the Planetary Gravity Assist Maneuver
A project by 1Lt Douglas J Hopper

Mech 532
Capt Chris Hall
Introduction

One problem with interplanetary travel is that it takes so long. The alternative to spending long periods in heliocentric transfer orbits to reach other planets is to increase speed. Speed can be increased by expending more fuel, but it can also be increased by use of planetary gravity assist maneuver (flyby). The purpose of this project was to investigate the use of the flyby maneuver to increase speed without cost in fuel. The motivation for saving fuel is that a small savings in fuel needed for injection to interplanetary orbit can have a larger effect on the weight of the vehicle. The simple (no flyby) transfer orbit which minimizes fuel used is called the Hohmann transfer, and is illustrated in figure 1 for a transfer between Earth and Mars.

Figure 1. Hohmann Transfer.
The specific impulse required for an outbound interplanetary voyage, and the planetary assist available to an interplanetary spacecraft were the subjects of investigation for this project. Several simplifying assumptions were made. The patched conic approximation was used to propagate the transfer orbits. This means that the spacecraft was assumed to be in a two body heliocentric orbit when it was outside the planetary sphere of influence (SOI), and once inside the SOI, it was in another two body orbit about the planet. This was the assumption which is furthest from the truth, but using it can give us a general idea of the magnitudes involved, and that was the primary purpose. The planetary orbits were assumed to be circular and coplanar. This also contains error, but again, the magnitude of the departure of reality from this assumption is small. All other forces besides the gravitational attraction between the two bodies in question were neglected including radiative and magnetic forces and relativistic effects.

The Effect of Increasing Specific Impulse

The Hohmann transfer shown in figure 1 takes about 258 days. In order to get onto the transfer orbit, the geocentric orbit must be hyperbolic. Figure 2 shows the geometry of the injection from a geocentric circular parking orbit to a hyperbolic one. An impulse, \( v \), applied at perigee produces the hyperbolic orbit. The impulse required depends on the hyperbolic excess speed, \( v \) desired. For the Hohmann transfer, this value can be calculated
from equations (1) and (2).

\[ V_i = \sqrt{2\mu_\oplus \left( \frac{1}{r_e} - \frac{1}{r_e + r_t} \right)} \]  

(1)

\[ \Delta V = \sqrt{(V_i - V_\oplus) + \frac{2\mu_\oplus}{r_e} - \sqrt{\frac{\mu_\oplus}{r_e}}} \]  

(2)

\( \Delta V \) impulse actually applied to spacecraft at injection  
\( r_e \) radius of Earth's orbit  
\( r_t \) radius of target planet's orbit  
\( r_o \) radius of geocentric parking orbit  
\( V_e \) orbital velocity of Earth about the sun  
\( V_i \) Heliocentric departure speed for Hohmann transfer  
\( \mu_\oplus \) gravitational parameter of Earth  
\( \mu_\odot \) gravitational parameter of the sun

For a Hohmann transfer from a 200 km parking orbit about Earth to Mars, \( \Delta V \) is about 3.62 km/sec. If more impulse is used, the reduction in time can be quite large, especially at first. To illustrate the magnitude of this effect, the transfer time to the planets Mars and Jupiter were plotted as functions of the impulse applied, \( \Delta V \) (see figures 2 and 3).
Figure 2. Flight Time vs. Impulse Applied (Mars)

Figure 3. Flight Time vs. Impulse Applied (Jupiter)
From figure 2, we can see that an increase in $\Delta v$ to 4 km/sec will reduce the flight time to about 150 days. The additional impulse required at Mars for capture is another consideration, and beyond the scope of this project. The matlab function used to generate figures 2 and 3 is in the appendix and is called "fly".

A similar trend can be seen for the flight to Jupiter, but Mars is between us and Jupiter, so we have opportunity to use the planetary flyby.

The Planetary Gravity Assist Maneuver

The flight to Jupiter on a Hohmann transfer ellipse would take about 1000 days (see figure 3). Using extra impulse at injection can cut that time in half for reasonable increases in applied impulse. If our planning is good, though, the planet Mars can help us on our journey. If our transfer orbit takes us past the planet Mars, its gravity will bend our trajectory somewhat. The effect of this bending was the main thrust of the project, and the results, which were calculated for a heliocentric orbit from Earth to Jupiter past Mars could readily be generalized to other interplanetary trajectories.

The planetary gravity assist maneuver (flyby) goes like this: The craft is on a heliocentric orbit until it crosses the boundary of the SOI for the flyby planet. After it crosses that
boundary, the effect of the sun is neglected and it goes into orbit about the flyby planet with a velocity, \( v_\infty \), equal to the vector difference in their initial heliocentric speeds. Since it came in from "infinity", it will always be on a hyperbolic orbit with respect to the planet, and its speed, \( v_\infty \) when it leaves (crosses the SOI outbound) will be the same as when it arrived with respect to the planet, but in a different direction. The angle through which it turns is called \( \delta \). The heliocentric orbit of the spacecraft has been altered during the flyby, since its heliocentric velocity, \( v_3 \) is now the vector sum of the planet's speed and the \( v_\infty \), which has been turned (see figure 4).

\[
V_3 = V_\infty + V_p
\]

\[
V_\infty = V_2 - V_p
\]

Figure 4. The Flyby
The program which simulates this effect simply finds the two dimensional vector $v_{\infty}$, rotates it through the angle $\delta$, finds the new heliocentric speed, $v_3$, and deduces its heliocentric orbit. It then calculates the time of flight from flyby to the target planet using Kepler's Equation. (See equations 4-6; figure 5; appendix "flyby1").

\[ v_3 = \arccos \left( \frac{P/{r_{\text{max}}}}{e} - 1 \right) \]  
\[ E = \frac{e + \cos \nu}{1 + e \cos \nu} \]  
\[ t_4 - t_3 = \frac{E_4 - e \sin E_4}{n} - \frac{E_3 - e \sin E_3}{n} \]

1. Initial position (Earth)
2. Mars encounter
3. Mars departure
4. Jupiter arrival
   True anomaly
5. $E$ Eccentric anomaly
6. $n$ Mean motion
Figure 5. Transfer to Jupiter with a Mars Flyby
To begin with, it was assumed that the planet Mars had the ability to turn the trajectory as much as desired, that is, that it could command a $\delta$ of up to 180 degrees. With that assumption, the Earth to Jupiter flight time with a Mars flyby was plotted as a function of $\delta$ in figure 6.

![Graph](image)

*Figure 6. Trip Time vs Turning Angle*

It can be seen that when $\delta$ equals zero, the time is simply the Hohmann transfer time, and that for $\delta = 0.1$ radian ($\sim 5$ degrees), the time drops sharply to 600 days. The first part of the trip, from Earth to Mars takes about 87 days, which is constant for a given injection speed.
It does not seem reasonable that the planet Mars or any other planet would really have unlimited ability to turn a spacecraft, since a turn of 180 degrees implies a parabolic orbit with respect to the planet, where $v_\infty$ is equal to zero. Indeed, there is a maximum turning angle which can be imposed by a given planet on a spacecraft, dictated by the planet's mass and radius. It is not desirable to hit the planet, or even its atmosphere while making the flyby, so the limiting case is where the radius of periapse for the planetary flyby orbit is the radius of the outer reaches of its atmosphere. Given this limitation, the impact parameter, $b$, can be calculated from equation (7).

$$
b = \frac{r_{flyby}}{v_\infty} \sqrt{\frac{v_\infty^2}{v_\infty^2 + \frac{2M_{flyby}}{r_{flyby}}}} \quad (7)
$$

The maximum turning angle that can be imposed by a planet depends, then only on the magnitude of $v_\infty$ with which the craft crosses the SOI. This can be calculated from equation (9), and maximum turning angle as a function of $v_\infty$ for a flyby of Mars is plotted in Figure 7.

$$
e = \sqrt{1 + 2E \left(\frac{r_{flyby} v_\infty \sin \frac{b}{r_\infty}}{M_{flyby}}\right)^2} \quad (8)
$$

$$
\delta = 2 \arcsin \left(\frac{1}{e}\right) \quad (9)
$$
A more massive planet can command a greater turning angle than Mars, whose mass is just one tenth that of Earth's. Figure 8 shows the maximum turning angles about Venus, Jupiter and Mars as functions of $v_\infty$. 

Figure 7. Maximum Turning Angle vs Velocity
If the spacecraft were launched from Earth with just enough speed to make the Hohmann transfer to Jupiter, its \( v_\infty \) is approximately 6.32 km/sec, so a turn of half a radian (\( \sim 30^\circ \)) is quite possible and a smaller turn angle can be obtained by making the flyby farther from the planet. Looking back to Figure 5, it can be seen that half a radian of turn can reduce the flight time to Jupiter from 1000 days to about 430 days without requiring any extra impulse from rockets or other thrusters. If the transfer orbit is faster than the Hohmann, then Mars will have less ability to turn the velocity, but it can still have beneficial effect. Even more promising is that the orbit to which the
spacecraft is initially injected can actually be slower than the Hohmann (less than enough impulse applied to make it to Jupiter), but with the flyby, the craft not only makes it to Jupiter, but the time is less than with the direct transfer orbit without the flyby. Figure 9 shows the trip time vs $\delta$ for $v_i = 1.295$ AU/TU (Hohmann), 1.25 AU/TU and 1.20 AU/TU. The curves with x's through them represent values of $\delta$ for which the spacecraft would not have made the trip. The model breaks down, and plots imaginary times.

![Trip time vs turning angle graph](image-url)
Conclusion

The planetary flyby is a fascinating maneuver that has been used successfully on many interplanetary probes including Voyager I and II, which made spectacular flybys of Jupiter, its moons and other planets. It is interesting that there exist ways in nature to "conserve" energy, and maximize the use of our feeble power with the use of our minds. There are many aspects of planetary flybys not considered here, nor was any new ground broken, perhaps, but it was interesting to get a taste of the subject.
function f = fly

% The idea is to plot the flight time versus injection speed for a 200
% km orbit from earth to mars and jupiter
alt = 200; % kilometers
mew = 1.00000000; %gravitational parameter
;
du = 6378.145;
tu = 13.44686457*60;
AU = 1.4959966e8; %units corrections
TU = 58.132821*24*3600;
r0 = alt + du; % correction for initial altitude

% this part will calculate the Hohmann injection speed needed for
% a trip to mars or jupiter
R1 = AU; % earth distance
R2 = 1.524*AU; % mars distance
%R2 = 5.203*AU; % jupiter distance

m2=mew*AU^3/TU^2;
vhoh=sqrt(2*m2*(1/R1-(1/(R1+R2))));
m=mew*du^3/tu^2;
vinf=vhoh - (AU/TU);
v0=sqrt((vinf^2)+(2*m/r0)) % needed to be injected onto hype
delv(1)=v0-sqrt(m/r0); %actually applied to attain Hohmann

v=delv(1);
R2=R2/AU;
for i=1:1:300,
    delv(i)=v;
    v0=v +sqrt(m/r0);
    vinf = sqrt(v0^2-(2*m/r0));
    vhoh=vinf+(AU/TU);
    vhoh=vhoh*AU/AU;
    V=[vhoh 0 0]';
    %looking down from sun's north pole,
    R=[0 -1 0]';
    % earth is at the bottom of the picture, ccw
    [e,p]=orbel2(R,V);
    % sends and receives canonical units
    nu =acos(((p/R2)-1)/e); % gives the value of true anomaly when
    % the transfer orbit reaches mars or jupiter
    % allowing us to calculate the time of flight
    a=p/(1-e^2);
    E=acos((e+cos(nu))/(1+e*cos(nu)));
    n=sqrt(mew/a^3);
    t = (E-e*sin(E))/n;
    time(i)= t*58.132821; % to get us a time in days, which is what we want

v=v+.01; % counting index, and also the velocity on the
end
plot (delv,time),title('Flight time vs. impulse applied'), xlabel('impulse (Km/sec)')
function f= flyby1

for i = 1:200,
if i==199,
    delta = 0.003;
elseif i==200,
    delta = 0;
else
    delta = pi/(i+3);
end
del(i)=delta;

% Here's the meat of the project
% Can we get good values for flyby trajectories:
% this part will calculate the Hohmann injection speed needed for
% a trip to mars or jupiter

mew = 1.0;
AU = 1.4959965e8;
TU = 58.132821; % constants

r1 = 1.0; % earth distance
R1 = [0 -r1 0]';
rm = 1.524; % earth vector
RM = [0 -rm 0]';

rj = 5.203; % mars distance
RJ = [0 -rj 0]';
vm = sqrt(mew/rm); % mars speed
vj = sqrt(mew/rj); % jupiter speed

vho=2*mew*sqrt(1/r1-(1/(r1+rj))));
/l=[1.25 0 0]';

% This part sends and receives from the other programs
[nul,e,p]=orbel3(R1,V1); % sends and receives canonical units

nu2 =acos(((p/rm)-1)/e); % gives the value of true anomaly when
% the transfer orbit reaches mars distance
% allowing us to calculate the time of flight
a=p/(1-e^2);
E=acos((e+cos(nu2))/(1+e*cos(nu2)));
n=sqrt(mew/a^3);

t12 = (E-e*sin(E))/n; % now we have the time up to the flyby
time= t12*TU; % to get us a time in days, which is what we want

[R2,V2]=rvee2(p,e,nu2); % gets our new position and velocity vector

% Here goes the flyby-- watch closely
VINF=[0 0 0]';
VP = [vm 0 0]';
VP = rot3(VP,-nu2);
VP = VP';
VINF = V2-VP;
VINF = rot3(VINF,-delta);
VINF = VINF';
/3 = VP + VINF; % that was it-- no time elapsed
R3 = R2; \quad \% (No distance either...)
{[nu3,e,p]= orbel3(R3,V3);
u4 =acos((p/rj)-1)/e); \quad \% true anomaly upon reaching jupiter
a=p/(1-e^2);}
n=sqrt(mew/a^3);
E3=acos((e+cos(nu3))/(1+e*cos(nu3)));
E4=acos((e+cos(nu4))/(1+e*cos(nu4)));
F3=acosh((e+cos(nu3))/(1+e*cos(nu3)));
F4=acosh((e+cos(nu4))/(1+e*cos(nu4)));

if e<= 1.0000, \quad \% This is in case that the flyby gives us
\quad t13 = (E3-e*sin(E3))/n; \quad \% a hyperbolic orbit as it often will
\quad t14 = (E4-e*sin(E4))/n;
else
\quad n=sqrt(mew/(-a)^3);
\quad t13 = (e*sinh(F3)-F3)/n;
\quad t14 = (e*sinh(F4)-F4)/n;
end

\quad t34 = t14 - t13;
\quad time2 = t34*TU;
\quad total(i) = time + time2;

end
plot(del,total),xlabel('turning angle about Mars (radians)'),ylabel('Total trip')
function f = turning

% We will find the velocities and flight path angles associated with
% various values of the turning angle, delta, for a planetary flyby

rinf=1.0e5;
mew = 3.986012e5; % gravitation for earth: to be scaled by mass of planet
m=.107;
j=318.0;
v=.817;
mew = mew*v; % gravitational parameter for given planet (changed for each)
rsup= 3380; % radius of Mars
rsup= 71370; % radius of Jupiter
rsup = 6178; % radius of Venus
i=1;
for vinf = 1:30,
    v(i)=vinf;
b=(rsup/vinf)*sqrt(vinf^2+(2*mew/rsup)); % impact parameter
    gamma=asin(b/rinf);
    E=vinf^2/2; % flight path angle
    h=rinf*vinf*sin(gamma);
    e=sqrt(1+2*E*(h/mew)^2);
    d(i)=2*asin(1/e);
    i=i+1;
end

axis([0,3.5,0,31]);
plot (d,v),xlabel('Maximum Turning Angle (radians)'),ylabel('Velocity at infinit

The following functions were called during the run of the
main program for the project. They are printed all together here
% to make it more convenient.

This function returns the magnitude of a vector to its fourth
element.

function[a]=mag(a)
a(4)=sqrt(a'*a);

This function rotates a vector about its third axis.

function[v2]=rot3(v1,a)
v2(1)=(v1(1)*cos(a))+(v1(2)*sin(a));
v2(2)=(-v1(1)*sin(a))+(v1(2)*cos(a));
v2(3)=v1(3);

This thing gives the r and v vectors when given the classical orbital
elements. The orbits are all in plane, so there are no angles.

function [R,V]=rv(p,e,nu)
mew = 1.000000;
r=p/(1+e*cos(nu));
R=[r*cos(nu)
r*sin(nu)
0];
V=sqrt(mew/p)*[-sin(nu)
e*cos(nu)
0];
R=R';
V=rot3(V,pi/2);
V=V';

This function will calculate the orbital elements from position and
velocity in canonical units. Again, no angles, so only nu, e and p
are needed to describe the orbit.

function [nu,e,p] = orbel3(R,V)
K=[0 0 1]';
r=mag(R);
v=mag(V);
mew = 1.000000;
H=cross(R,V);
p=(H'*H)/mew;
h=mag(H);
N=cross(K,H);
\n=mag(N);
E=(1/mew)*((V'*V)-(mew/r(4)))*R-(R'*V)*V);
e=mag(E);
e=e(4);
nu = acos((E'*R)/(e*r(4)));
if (R'*V)< 0,
    nu = 2*pi-nu;
end
AUTONOMOUS ONBOARD ORBIT DETERMINATION

Troy Irwin
Air Force Institute of Technology, Wright-Patterson AFB, Ohio

I. INTRODUCTION

The objective of this project was to develop and validate via simulation a method of autonomous onboard orbit determination for a spacecraft. Autonomous in this context means without reference to any other man-made object, such as another satellite or a ground station. In addition to the autonomy requirement, desired features of the method are that no new hardware be required on the satellite, and that the method be capable of determining the orbit without any a priori knowledge of the orbit.

The method developed here is a variation of the method developed by Ward [1] and enhanced by Hicks [2]. The orbit is determined with Gibb's method using three Earth sensor and star sensor observations. Ward and Hicks, on the other hand, used multiple observations as input to an estimator to determine the orbit, and their method required a priori knowledge of the orbit.

Computer simulations based on the space station Freedom orbit show the method works if the observed star is normal to the right ascension of the ascending node (RAAN). The method breaks down if the observed star is not normal to the RAAN, with interesting results nonetheless.

II. PROBLEM DEFINITION & APPROACH

The problem definition is given two angle measurements (made without reference to any outside man-made object or device) determine an inertial position vector. Then with three such vectors determine the Keplerian orbital elements.

The approach is based on Ward's presumption of a spacecraft which uses an Earth horizon sensor and star sensor for attitude determination and control. Ward proposed the sensors also be used to determine the cosine of the angle between the satellite's local vertical, or zenith, and the observed star. This project proposes the horizon sensor also be used to measure the angle of the disk of the Earth as seen from the satellite (figure 1). Inertial position vectors can be determined from the sensor measurements. Once three vectors have been found, Gibb's method is used to compute a velocity vector. Last, the position and velocity vectors are converted to Keplerian elements.

Assumptions made to simplify the mathematics are that the Earth is a perfect sphere and sensor measurements are noiseless. As in all "simple" orbital mechanics projects, two body motion is assumed. A further assumption which does not effect the mathematics but which simplified coding both the truth model and the onboard algorithm is that the star is continuously visible from the spacecraft, i.e., Earth eclipses of the star were not modeled.
Figure 1: Observation Geometry

Figure 2: Radial Distance Measurement
Determining the Inertial Position Vector

First, the satellite's radial distance from the center of the Earth must be determined. Using horizon sensor measurements of the subtended angle of the disk of the Earth as seen from the satellite, the distance \( r \) is

\[
 r = \frac{r_e \alpha}{\sin \alpha / 2}
\]

where \( r_e \) is the mean equatorial radius of the Earth and \( \alpha \) is the measured angle (figure 2). The satellite's position vector in a rotating spherical coordinate reference frame attached to the position vector, the U frame, is then

\[
 r = [r \ 0 \ 0].
\]

The next task is to rotate \( r \) into the geocentric equatorial frame (also called Earth centered inertial (ECI)) frame. Two rotations are required to accomplish this. First, \( r \) is rotated from the U frame into a "pseudo-perifocal" frame (PPF) (figure 3). In the PPF the x axis is in the orbital plane in the direction of the projection of the star's unit vector on the plane of the orbit; the z axis is perpendicular to the orbital plane; and the y axis is in the orbital plane and completes the right handed system. The angle between the U frame and the PPF is \( \chi \) ("chi," not "ex"), and is equal to \( \gamma \) ("gamma") provided the observed star is in the orbital plane. If the star is out of the orbital plane then \( \chi \) can be found from the empirically derived relation

\[
 \cos \chi = \frac{\cos \gamma}{\cos \delta}
\]

where \( \delta \) is the the angle between the star's unit vector and the orbital plane and will be explained further in a moment. Thus the rotation matrix from the U frame to the PPF is

\[
 C_{up} = \begin{bmatrix}
 \cos \chi & -\sin \chi & 0 \\
 \sin \chi & \cos \chi & 0 \\
 0 & 0 & 1
\end{bmatrix}
\]

Recall the output of the sensors is the cosine of \( \gamma \), not \( \gamma \) itself. However, there is no quadrant ambiguity. The sign of \( \cos \gamma \) puts \( \gamma \) either in quadrants I/IV or quadrants II/III. The difference between any two successive measurements of \( \cos \gamma \) can be used to determine \( \gamma \) uniquely, since \( \cos \gamma \) is always increasing when \( \gamma \) is in quadrants III/IV and always decreasing for quadrants I/II (figure 4).

Next, the PPF must be rotated into the "stellar" frame (SF). The SF is simply a rotation of the ECI cartesian frame through the right ascension and declination of the star. The PPF is rotated through \( -\delta \) ("delta") degrees, where as mentioned earlier \( \delta \) is the angle between the star's unit vector and
Figure 3: Pseudo-Perifocal Plane
**Figure 4**: Quadrant Resolution

**Figure 5**: Pseudo-Perifocal to Stellar Frame
the orbital plane (figure 5). Thus the rotation matrix from the PPF to the SF is:

\[
C_{sp} = \begin{bmatrix}
\cos d & 0 & \sin d \\
0 & 1 & 0 \\
-\sin d & 0 & \cos d
\end{bmatrix}
\] (5)

If \(d\) is not known a priori, it can be empirically determined from the \(\cos g\) measurement. When \(x=0\) or 180 degrees, the absolute value of \(g\) will be minimum, and from geometry will be equal to \(d\) (figure 6). Therefore, the satellite can determine \(d\) by checking the difference between successive measurements of \(\cos g\), as described above. When the difference changes from positive to negative or vice versa, signifying movement from quadrant IV to I (or II to III), then the current absolute value of \(\cos g\) can be used for \(\cos d\), and \(\sin d\) can be determined from trigonometric identities.

One limitation of determining \(d\) this way is there is ambiguity in whether the observed star is "above" or "below" the orbital plane (figure 7). For simplicity, the derivations here assume the star is above the plane, and care will be taken in the simulations to only choose stars above the plane.

The second rotation is one from the SF to the ECI frame through the declination (dec) and right ascension (ra) of the star. The rotation matrix for this common procedure is

\[
C_{is} = \begin{bmatrix}
\cos(ra) \cos(dec) & -\sin(ra) & -\cos(ra) \sin(dec) \\
\sin(ra) \cos(dec) & \cos(ra) & -\sin(ra) \sin(dec) \\
\sin(dec) & 0 & \cos(dec)
\end{bmatrix}
\] (6)

The right ascension and declination of the chosen star can be obtained from a reference such as The Astronomical Almanac and stored onboard the satellite. Computation of (6) will only have to be done once upon initialization of the onboard program.

Finally, (2) and (4) through (6) are combined to map the sensor observations into an ECI vector:

\[
R = [C_{is}][C_{sp}][C_{pu}]r
\] (7)

Calculate a Velocity Vector with Gibb's Method

Once three position vectors have been found, Gibb's method can be used to determine a velocity vector associated with any one of the position vectors. For this project, the velocity vector associated with the most recent position vector is solved for because that position vector tells the satellite "where I am," as opposed to the other two vectors which only tell the satellite "where I was." The following description is a summary of the Bate, Mueller, and White (BMW) version of Gibb's method [3].

Given three vectors determined from the horizon and star sensor measurements, designate the oldest as \(R_1\), the next as \(R_2\), and the most recent as \(R_3\), and
Figure 6: Measuring $\delta$

Figure 7: Above/Below Ambiguity
their magnitudes as \( r_1, r_2, \) and \( r_3 \), respectively. Then form three vectors \( D, N, \) and \( S \), where

\[
D = R_1 x R_2 + R_2 x R_3 + R_3 x R_1 \tag{8}
\]

\[
N = r_3 (R_1 x R_2) + r_1 (R_2 x R_3) + r_2 (R_3 x R_1) \tag{9}
\]

and \( S = (r_2 - r_3) R_1 + (r_3 - r_1) R_2 + (r_1 - r_2) R_3. \tag{10} \)

\( D \) and \( N \) are normal to the orbital plane in the direction of the angular momentum vector, and \( S \) lies in the orbital plane along the semi-latus rectum. Also, note \( S \) is a 0 vector for circular orbits.

Since the velocity vector at \( R_3 \) is of interest, form the vector

\[
B = D x R_3 \tag{11}
\]

and the scalar

\[
L = \left( \frac{\mu u}{DN} \right)^{\frac{1}{2}} \tag{12}
\]

where \( \mu u \) is the gravitational parameter of the Earth. Finally

\[
V_3 = \frac{LB + LS}{r_3}. \tag{13}
\]

BM\(^W\) advocate several checks to ensure the three vectors are coplanar and describe a two body orbit. These checks are not performed in this project because of a priori knowledge of the truth model and a desire for computational simplicity.

**Convert \( R \) and \( V \) to Keplerian Elements**

The last major step is to convert \( R_3 \) and \( V_3 \) to Keplerian elements. As a side benefit of using the Gibb's method to determine \( V_3 \), the eccentricity, semi-major axis, inclination, and right ascension of the ascending node (RAAN) can be easily calculated based on BM\(^W\). Only a few additional calculations are required to determine the argument of perigee and true anomaly. Schutz's method is used to eliminate quadrant ambiguities. [4]

First, the eccentricity (\( e \)) is

\[
e = S/D. \tag{14}
\]

Next, the semi-latus rectum (\( p \)) is

\[
p = N/D \tag{15}
\]

and the semi-major axis (\( a \)) is

\[
a = \frac{p}{1-e^2}. \tag{16}
\]
Recall N is in the direction of the angular momentum, and denoting it as 
\( N = N_x \mathbf{i} + N_y \mathbf{j} + N_z \mathbf{k} \), the inclination \( i \) and right ascension of the ascending node (RAAN) can be found via:

\[
\cos i = \frac{N_z}{N} \tag{17}
\]

and

\[
\tan \text{RAAN} = \frac{N_y}{N_x}. \tag{18}
\]

Before determining the argument of perigee and true anomaly, several intermediate quantities must be computed. Since \( S \) is in the orbital plane in the direction of the semi-latus rectum, \( Q = S \times N \) is a vector in the plane of the orbit in the direction of perigee. Denote \( Q = Q_x \mathbf{i} + Q_y \mathbf{j} + Q_z \mathbf{k} \) and rotate 
\( Q \) through the RAAN and inclination into a frame whose \( x \) axis is along the line of nodes, the \( y \) axis is 90 degrees ahead of the \( x \) axis in the orbital plane, and the \( z \) axis is in the direction of \( D \) and \( N \). \( Q \) in this new frame is

\[
Q = \begin{bmatrix}
Q_x \cos \text{RAAN} + Q_y \sin \text{RAAN} \\
-Q_y \sin \text{RAAN} \cos i + Q_y \cos \text{RAAN} \cos i + Q_z \sin i \\
Q_x \sin \text{RAAN} \sin i - Q_y \cos \text{RAAN} \sin i - Q_z \cos i
\end{bmatrix}. \tag{19}
\]

The argument of perigee \( \omega \) ("little omega") is now readily available from

\[
\tan \omega = \frac{-Q_y \sin \text{RAAN} \cos i + Q_y \cos \text{RAAN} \cos i + Q_z \sin i}{Q_x \cos \text{RAAN} + Q_y \sin \text{RAAN}}. \tag{20}
\]

Last but not least the true anomaly \( \nu \) ("nu") must be found. To begin, recall

\[
r = \frac{p}{1 + e \cos \nu} \tag{21}
\]

and rearrange terms to find

\[
\cos \nu = \frac{p - r}{r e}. \tag{22}
\]

Differentiate (21) with respect to time, and recognize

\[
\dot{v} = \frac{h}{r^2}, \tag{23}
\]

where the \( h \) is the magnitude of the angular momentum vector \( R_3 \times V_3 \). Combine (21) and (23) and rearrange terms to find

\[
\dot{i} = \frac{h e \sin \nu}{r(1 + e \cos \nu)}. \tag{24}
\]

However, \( \dot{i} \) is the magnitude of the projection of the velocity vector on the
position vector, which may be rewritten as

\[ \mathbf{r} = \mathbf{R3} \cdot \mathbf{V3} \quad \frac{1}{r} . \] (25)

Combining (24) and (25) and rearranging terms,

\[ \sin v = \frac{p \cdot (\mathbf{R3} \cdot \mathbf{V3})}{r \cdot h \cdot e} . \] (26)

Finally, (22) and (26) are combined to find \( v \):

\[ \tan v = \frac{p \cdot \mathbf{R3} \cdot \mathbf{V3}}{h \cdot (p-r)}. \] (27)

III. TRUTH MODEL

The truth model is a simple routine which takes the given Keplerian elements, integrates the mean motion, produces a position vector, and calculates simulated observations of \( a \) and \( \cos q \). The input elements are semi-major axis, eccentricity, inclination, RAAN, argument of perigee, and true anomaly at epoch.

First, the satellite's mean motion is computed from Kepler's third law, where

\[ n = \frac{1}{\sqrt{\mu}} = \frac{2}{(a^3)}. \] (28)

Next, the mean anomaly \( M \) is computed from

\[ M = nt \] (29)

where \( t \) is the time since the satellite passed perigee. \( M \) is used to solve for eccentric anomaly \( E \) in Kepler's equation

\[ M = E - e \sin E \] (30)

using a Newton-Raphson iteration technique. \( M \) is used as the first estimate of \( E \) in the iteration.

Once \( E \) has been solved for, the true anomaly is found from

\[ \cos v = \frac{e - \cos E}{e \cos E - 1} . \] (31)

If \( E \) is greater than 180 degrees, \( v \) is changed to \(-v\) plus 360 degrees to resolve the quadrant ambiguity.

The last value needed is the semi-latus rectum \( (p) \). Since \( a \) and \( e \) are known,

\[ p = a(1-e^2) . \]
With $e$, $p$, and $v$ known, the radial distance at $v$ is

$$r = \frac{p}{1 + e \cos v}$$

(32)

and a position vector in the perifocal frame is

$$R = \begin{bmatrix} r \cos v \\ r \sin v \\ 0 \end{bmatrix}.$$  

(33)

The final step is to rotate $R$ into the geocentric equatorial frame. This is done by

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} r \cos v \\ r \sin v \\ 0 \end{bmatrix} = \begin{bmatrix} Rx \\ Ry \\ Rz \end{bmatrix}.$$  

(34)

where

$$r_{11} = \cos RAAN \cos w - \sin RAAN \sin w \cos l,$$
$$r_{12} = -\cos RAAN \sin w - \sin RAAN \cos w \cos l,$$
$$r_{13} = \sin RAAN \sin l,$$
$$r_{21} = \sin RAAN \cos w + \cos RAAN \sin w \cos l,$$
$$r_{22} = -\sin RAAN \cos w + \cos RAAN \sin w \cos l,$$
$$r_{23} = -\cos RAAN \sin l,$$
$$r_{31} = \sin w \sin l,$$
$$r_{32} = \cos w \sin l,$$
$$r_{33} = \cos l.$$

To simulate the observation of $\cos g$, the observed star's right ascension and declination are used to generate a unit vector in the direction of the star. The unit vector is

$$s = sxI + syJ + szK$$

(35)

where

$$sx = \cos dec \cos ra,$$
$$sy = \cos dec \sin ra,$$
$$sz = \sin dec.$$

Using the definition of the dot product, the simulated $\cos g$ observation is

$$\cos g = \frac{Rx \ sx + Ry \ sy + Rz \ sz}{r \ l}.$$  

(36)

Finally, the simulated alpha measurement is calculated by rearranging (1) to

$$\alpha = 2 \arcsin \left( \frac{re}{r} \right).$$  

(37)

IV. DATA REDUCTION

The output of the onboard algorithm is the Keplerian orbital elements, and direct comparison without data reduction is sufficient to analyze the ultimate results. However, it is desirable to study the intermediate results of the onboard algorithm: the position vectors computed from the sensor
observations. To do this, the computed position vector is rotated into the U frame of the truth model position vector. This rotation matrix is

\[
C = \begin{bmatrix}
\cos(ra)\cos(dec) & \sin(ra)\cos(dec) & \sin(dec) \\
-sin(ra) & \cos(ra) & 0 \\
-cos(ra)\sin(dec) & -\sin(ra)\sin(dec) & \cos(dec)
\end{bmatrix}
\]  
(38)

where

\[ra = \arctan \frac{Ry}{Rx}\]  
(39)

and

\[dec = \arctan \frac{Rz}{(Rx^2 + Ry^2)^{1/2}}\]  
(40)

Combining (7), (32), and (38), an error vector for each observation can be defined as

\[E = [C]R - \begin{bmatrix} r \end{bmatrix} \]  
(41)

where \(R\) is the observed position vector and \(r\) is the magnitude of the truth model position vector. The components of \(E\) are the radial, in-track, and cross-track errors, respectively, between the position vector determined by the onboard algorithm and the truth model position vector at the same time. The magnitude of the error is

\[error = (Ex^2 + Ey^2 + Ez^2)^{1/2}\]  
(42)

V. SIMULATION RESULTS AND ANALYSIS

Simulations were run to characterize the performance of the onboard orbit determination algorithm. Each simulation had the following inputs in common:

- semi-major axis: 6785.58 km
- eccentricity: .00001
- argument of perigee: 0 deg
- true anomaly at epoch: 0 deg
- time between sensor observations: 200 sec
- run time: 5400 sec
- initial \(\cos d\): .5

The semi-major axis is the same as proposed for the space station Freedom, and was used by Hicks to validate his method. Hicks used 0 eccentricity, but this caused division by 0 due to (10), so \(e=0.00001\) was chosen. Argument of perigee and true anomaly at epoch were arbitrarily set to 0 for convenience. 200 seconds between sensor observations is the optimal interval Hicks found for his configuration and the Freedom orbit, so it was used here to provide a basis for comparison. Each simulation ran for 5400 seconds to observe the performance over one orbital period. The algorithm needs an initial value of \(\cos d\) to use until the real value is measured at \(X=0/180\), and \(\cos d=.5\) was arbitrarily chosen.
Case I: Star in Orbital Plane

In the first run, the additional baseline Freedom orbital parameters of i=28 degrees and RAAN=45 degrees were input to the truth model. Then "Testar" was chosen as the observed star. Testar is in the plane of the orbit, and its position on the celestial sphere is:

right ascension 135 deg  
declination 28 deg.

As shown in table I, the elements computed by the onboard algorithm converged to the truth model values as soon as the satellite passed through X=0, as expected. The radial, in-track, and cross-track errors are shown in figure 8. The spike at v=90 and 270 degrees (X=0/180) is because the quadrant resolution logic discussed earlier does not work for the first observation after the quadrant changes. Since cos g was not measured precisely at X=0 due to the 200 second interval between observations, it is possible for the sign of the difference in cos g to not change even though the quadrant has changed. If cos g could be measured precisely at X=0 then this spike would not be present.

Case II: Observed Star Out of Plane

The second simulation tried to validate the method for the general case of a star arbitrarily out of the orbital plane. The RAAN and inclination for Freedom were the same as case I. The observed star for this simulation was Deneb, whose position on the celestial sphere in July 1986 was

right ascension 49 deg 45 min  
declination 45 deg 14 min [5].

Results of the case II simulation are shown in table II. Although the algorithm converges as expected for semi-major axis and eccentricity, it does not converge to inclination and RAAN. Upon closer inspection, table II shows the RAAN is the same as Deneb's right ascension minus 90 degrees. Figure 9 shows the overall position error averages about 3,000 km.

To understand these unexpected results, refer again to figure 5. Figure 5 misleadingly implies the y axis of the PPF and the SF are the same, hence the 2 rotation from the PPF and the SF given as

\[
C_{sp} = \begin{bmatrix}
\cos d & 0 & \sin d \\
0 & 1 & 0 \\
-sin d & 0 & \cos d
\end{bmatrix}
\]

(5)

In fact, the y axis is not the same in the PPF and the SF except in the special case where the line of nodes is normal to the unit vector in the direction of the star. Figure 10 shows the general relationship between the PPF and the SF, where an additional rotation through the angle B ("beta") is required. Then (5) becomes

\[
C_{sp} = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos B & -\sin B \\
0 & \sin B & \cos B
\end{bmatrix} \begin{bmatrix}
\cos d & 0 & \sin d \\
0 & 1 & 0 \\
-sin d & 0 & \cos d
\end{bmatrix}
\]

(43)
**TABLE I**

**CASE I: STAR IN PLANE, NORMAL TO RAAN**

<table>
<thead>
<tr>
<th>$t$ (sec)</th>
<th>$a$ (km)</th>
<th>$e$</th>
<th>$i$ (deg)</th>
<th>RAAN (deg)</th>
<th>$w$ (deg)</th>
<th>$\nu$ (deg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0010</td>
<td>-0.0000</td>
<td>0.0033</td>
<td>0.1039</td>
<td>-0.1352</td>
<td>0.0899</td>
<td>-0.1798</td>
</tr>
<tr>
<td>0.2010</td>
<td>-0.0006</td>
<td>0.0043</td>
<td>0.1142</td>
<td>0.0611</td>
<td>0.0881</td>
<td>-0.1033</td>
</tr>
<tr>
<td>0.4010</td>
<td>6.7855</td>
<td>0.0000</td>
<td>0.0320</td>
<td>-0.1350</td>
<td>-0.0818</td>
<td>-0.0371</td>
</tr>
<tr>
<td>0.6010</td>
<td>6.7859</td>
<td>0.0000</td>
<td>0.0564</td>
<td>0.1764</td>
<td>0.1361</td>
<td>0.0444</td>
</tr>
<tr>
<td>0.8010</td>
<td>6.7855</td>
<td>0.0000</td>
<td>0.1059</td>
<td>0.1313</td>
<td>0.1692</td>
<td>-0.1528</td>
</tr>
<tr>
<td>1.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0900</td>
<td>0.1350</td>
<td>-0.0620</td>
<td>0.0648</td>
</tr>
<tr>
<td>1.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0900</td>
<td>0.1350</td>
<td>-0.0620</td>
<td>0.0777</td>
</tr>
<tr>
<td>1.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0900</td>
<td>0.1350</td>
<td>-0.0384</td>
<td>0.0657</td>
</tr>
<tr>
<td>1.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.1114</td>
<td>-0.0133</td>
<td>-0.0206</td>
<td>-0.1320</td>
</tr>
<tr>
<td>1.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0118</td>
<td>0.1047</td>
</tr>
<tr>
<td>2.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0001</td>
<td>0.1294</td>
</tr>
<tr>
<td>2.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>0.1424</td>
</tr>
<tr>
<td>2.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>0.1554</td>
</tr>
<tr>
<td>2.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>0.1683</td>
</tr>
<tr>
<td>2.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1787</td>
</tr>
<tr>
<td>3.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1658</td>
</tr>
<tr>
<td>3.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1528</td>
</tr>
<tr>
<td>3.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1399</td>
</tr>
<tr>
<td>3.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1269</td>
</tr>
<tr>
<td>3.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1140</td>
</tr>
<tr>
<td>4.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>-0.0001</td>
<td>-0.1009</td>
</tr>
<tr>
<td>4.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0503</td>
<td>-0.1420</td>
</tr>
<tr>
<td>4.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>-0.0681</td>
<td>-0.0071</td>
</tr>
<tr>
<td>4.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0492</td>
<td>-0.1114</td>
</tr>
<tr>
<td>4.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0001</td>
<td>-0.0494</td>
</tr>
<tr>
<td>5.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.0364</td>
</tr>
<tr>
<td>5.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.0234</td>
</tr>
<tr>
<td>5.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0280</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.0105</td>
</tr>
</tbody>
</table>

All values times 1.0E+03
Case I: Star In Plane

Figure 8
<table>
<thead>
<tr>
<th>$t$ (sec)</th>
<th>$a$ (km)</th>
<th>$e$</th>
<th>$i$ (deg)</th>
<th>RAAN (deg)</th>
<th>$\theta$ (deg)</th>
<th>$\psi$ (deg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0010</td>
<td>-0.0000</td>
<td>0.0011</td>
<td>0.0662</td>
<td>-0.1310</td>
<td>-0.0889</td>
<td>-0.1800</td>
</tr>
<tr>
<td>0.2010</td>
<td>-0.2132</td>
<td>0.0010</td>
<td>0.0890</td>
<td>0.0496</td>
<td>-0.1771</td>
<td>-0.1769</td>
</tr>
<tr>
<td>0.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0900</td>
<td>0.0495</td>
<td>0.0020</td>
<td>0.0062</td>
</tr>
<tr>
<td>0.6010</td>
<td>6.7855</td>
<td>0.0000</td>
<td>0.0969</td>
<td>-0.1185</td>
<td>-0.0078</td>
<td>-0.0939</td>
</tr>
<tr>
<td>0.8010</td>
<td>6.7855</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0183</td>
<td>0.0950</td>
</tr>
<tr>
<td>1.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0626</td>
<td>0.0637</td>
</tr>
<tr>
<td>1.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0405</td>
<td>0.0618</td>
<td>0.0774</td>
</tr>
<tr>
<td>1.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>0.0905</td>
</tr>
<tr>
<td>1.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0405</td>
<td>0.0616</td>
<td>0.1035</td>
</tr>
<tr>
<td>1.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>0.1165</td>
</tr>
<tr>
<td>2.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>0.1295</td>
</tr>
<tr>
<td>2.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>0.1424</td>
</tr>
<tr>
<td>2.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>0.1554</td>
</tr>
<tr>
<td>2.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0405</td>
<td>0.0616</td>
<td>0.1683</td>
</tr>
<tr>
<td>2.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0405</td>
<td>0.0616</td>
<td>-0.1787</td>
</tr>
<tr>
<td>3.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0405</td>
<td>0.0617</td>
<td>-0.1657</td>
</tr>
<tr>
<td>3.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0073</td>
<td>-0.0383</td>
<td>0.0868</td>
<td>-0.1504</td>
</tr>
<tr>
<td>3.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0092</td>
<td>-0.0412</td>
<td>0.1007</td>
<td>-0.1479</td>
</tr>
<tr>
<td>3.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0405</td>
<td>0.0479</td>
<td>-0.1134</td>
</tr>
<tr>
<td>3.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0618</td>
<td>-0.1143</td>
</tr>
<tr>
<td>4.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>-0.1011</td>
</tr>
<tr>
<td>4.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>-0.0882</td>
</tr>
<tr>
<td>4.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0405</td>
<td>0.0616</td>
<td>-0.0752</td>
</tr>
<tr>
<td>4.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0405</td>
<td>0.0616</td>
<td>-0.0623</td>
</tr>
<tr>
<td>4.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>-0.0493</td>
</tr>
<tr>
<td>5.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>-0.0364</td>
</tr>
<tr>
<td>5.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>-0.0234</td>
</tr>
<tr>
<td>5.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0082</td>
<td>-0.0406</td>
<td>0.0616</td>
<td>-0.0105</td>
</tr>
</tbody>
</table>

All values times $1.0E+03$
Case II: Star Out of Plane

Figure 9
\( \hat{p}_y \) in orbital plane normal to \( \hat{p}_x \)

\( \hat{s}_z \) in equatorial plane to \( \hat{s}_x \)

\( \hat{p}_z, \hat{s}_z \) not shown

**Figure 10**: General Relationship Between PPF and SF
The problem now is $B$ is unknown. While $B$ is apparently a constant function of inclination, RAAN, and the right ascension of the observed star, the function is not obvious. As an estimate however, this simulation was re-run using $d$ as an estimate for $B$. The results of this run are shown in Table III and Figure II. Using $d$ as an estimate of $B$ results in convergence of $i$, RAAN, $v$, and $v$ to within one degree and reduces average position error to about 100 km.

Unfortunately, the $B=d$ estimate is only applicable to this specific case. Consider a spacecraft in an equatorial orbit viewing an out of plane star. In that case $B$ is clearly $0$, not $d$.

Case III: Observed Star Out of Plane, Normal to RAAN

The third case attempts to validate the algorithm for a specific orbital configuration taking into account the previously discovered limitation that $B$ is unknown. Once again Testar is observed by the satellite, which is in an orbit with RAAN=45 degrees and $i$=20 degrees. Results of this simulation are shown in Table III and Figure 12. As in case I, the elements computed by the onboard algorithm converged to the truth model values as soon as the satellite passed through $x=0$.

Several other simulations were run with inclinations greater than the declination of Testar. Each of these runs had erroneous results for inclination and RAAN. This was expected due to the ambiguity in knowing if the star is above or below the orbital plane.

VI. CONCLUSIONS & RECOMMENDATIONS

The method provided unacceptable results if the star is arbitrarily out of the plane of the orbit. However, the results for simulations based on a priori knowledge were encouraging. Specifically, if the RAAN is known, a star can be selected 90 degrees ahead or behind the the RAAN. Inclination only need be known to the extent necessary to ensure the observed star is in or "above" the orbital plane. Measurements were taken from an Earth sensor and star sensor already on the spacecraft for attitude determination.

If further work is done in this area, the focus should be on determining $B$, either in closed form or numerically. The method can also be improved by eliminating the ambiguity in whether or not the observed star is above or below the orbital plane, and improving the quadrant resolution logic. Last, the method could be modified to determine some other elements, such as equinoctial, to eliminate the singularities associated with Keplerian elements.

If these improvements are made, especially determining $B$, this method has potential application as a robust source of an estimate of the orbit. This estimate can be used to initialize a higher fidelity method, such as Hicks. The ultimate benefit of autonomous orbit determination methods like this is reduced ground station support for the ever growing population of satellites.
Case Two: Star Out of Plane, Beta Estimated

Position error (km)

True anomaly (degrees)

FIGURE 11
## TABLE III

**CASE III: STAR OUT OF PLANE, NORMAL TO RAAN**

<table>
<thead>
<tr>
<th>$t$ (sec)</th>
<th>$a$ (km)</th>
<th>$e$</th>
<th>$i$ (deg)</th>
<th>RAAN (deg)</th>
<th>$\nu$ (deg)</th>
<th>$\nu'$ (deg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0010</td>
<td>-0.0000</td>
<td>0.0033</td>
<td>0.1017</td>
<td>-0.1352</td>
<td>0.0899</td>
<td>-0.1797</td>
</tr>
<tr>
<td>0.2010</td>
<td>-0.0006</td>
<td>0.0043</td>
<td>0.1141</td>
<td>0.0610</td>
<td>0.0881</td>
<td>-0.1032</td>
</tr>
<tr>
<td>0.4010</td>
<td>6.7855</td>
<td>0.0000</td>
<td>0.0320</td>
<td>-0.1350</td>
<td>-0.0819</td>
<td>-0.0381</td>
</tr>
<tr>
<td>0.6010</td>
<td>6.7859</td>
<td>0.0000</td>
<td>0.0544</td>
<td>0.1793</td>
<td>0.1332</td>
<td>0.0434</td>
</tr>
<tr>
<td>0.8010</td>
<td>6.7855</td>
<td>0.0000</td>
<td>0.1064</td>
<td>0.1312</td>
<td>0.1685</td>
<td>-0.1566</td>
</tr>
<tr>
<td>1.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0900</td>
<td>0.1350</td>
<td>-0.0596</td>
<td>0.0612</td>
</tr>
<tr>
<td>1.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0900</td>
<td>0.1350</td>
<td>-0.0505</td>
<td>0.0639</td>
</tr>
<tr>
<td>1.4010</td>
<td>6.7859</td>
<td>0.0000</td>
<td>0.0900</td>
<td>0.1350</td>
<td>-0.0027</td>
<td>0.0227</td>
</tr>
<tr>
<td>1.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.1066</td>
<td>-0.0233</td>
<td>-0.0169</td>
<td>-0.0993</td>
</tr>
<tr>
<td>1.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0118</td>
<td>0.1047</td>
</tr>
<tr>
<td>2.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0001</td>
<td>0.1294</td>
</tr>
<tr>
<td>2.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>0.1424</td>
</tr>
<tr>
<td>2.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>0.1554</td>
</tr>
<tr>
<td>2.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>0.1683</td>
</tr>
<tr>
<td>2.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1787</td>
</tr>
<tr>
<td>3.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1658</td>
</tr>
<tr>
<td>3.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1528</td>
</tr>
<tr>
<td>3.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1399</td>
</tr>
<tr>
<td>3.6010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1269</td>
</tr>
<tr>
<td>3.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1140</td>
</tr>
<tr>
<td>4.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.1009</td>
</tr>
<tr>
<td>4.2010</td>
<td>6.7854</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0503</td>
<td>-0.1420</td>
</tr>
<tr>
<td>4.4010</td>
<td>6.7858</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>-0.0581</td>
<td>-0.0071</td>
</tr>
<tr>
<td>4.6010</td>
<td>6.7855</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0492</td>
<td>-0.1114</td>
</tr>
<tr>
<td>4.8010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0001</td>
<td>-0.0494</td>
</tr>
<tr>
<td>5.0010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.0364</td>
</tr>
<tr>
<td>5.2010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.0234</td>
</tr>
<tr>
<td>5.4010</td>
<td>6.7856</td>
<td>0.0000</td>
<td>0.0200</td>
<td>0.0450</td>
<td>0.0000</td>
<td>-0.0105</td>
</tr>
</tbody>
</table>

*all values times 1.0E+03*
Case III: Star Out of Plane, Normal to RAAN

FIGURE 12
BIBLIOGRAPHY


A PRELIMINARY ANALYSIS OF ORBITAL RENDEZVOUS
WITH AN EARTH/MOON ELEVATOR

By: Capt Stewart J. Kowall

For: Capt Chris Hall

UNITED STATES AIR FORCE INSTITUTE OF TECHNOLOGY
A proposal has been made to construct an elevator from the moon to the earth. Such an elevator would be able to transport people and material between the earth/moon system to make use of the moons' vast mineral resources. Unfortunately, physicists have calculated that the base of the elevator on the surface of the earth would have to be the same size as the earth's diameter; given today's materials [REF 1]. Clearly, this is impractical. An alternative to the earth/moon elevator is an elevator that extends from the moon (which would have a base substantially smaller) to some distance above the earth. Given that such an elevator could be built, there is also the very difficult problem of rendezvousing with the free end of the elevator/tether. This paper will take a preliminary look at three aspects of this problem. First, since there is a difference in velocity between the tether and any satellite attempting to rendezvous with it, an analysis must be made to determine the amount of time in which the satellite and tether will be in close proximity to each other. Second, a simplified look at the equations of motion between the tether and satellite will be given. Finally, the phasing required for the rendezvous via a Hohmann transfer will be addressed.

To start the analysis, a few simplifying assumptions are made. The elevator is modeled as a rigid body which extends from the surface of the moon to some distance above the earth. The model proposes that a satellite departing from a space station in a 300 NM parking orbit around the earth initiates a Hohmann transfer in such a way as to rendezvous with the tether exactly at apoapsis. The transfer orbit is in the same equatorial plane.
as the moon. Finally, the model is based on a two body system using the mass of the earth as the gravitational force. The heart of the rendezvous problem is given be the fact that the tether and the satellite will be moving at two different velocities. The velocity of a satellite in a circular orbit is given by

$$V_s = \sqrt{\mu/R}$$

where $R$ is the radius from the center of the earth. Since the tether is a rigid body extending from the moon, any distance along the tether will have the same angular velocity as the moon. Consequently, the velocity of the end of the tether is given by

$$V_t = R\dot{\theta}$$

where $R =$ radius from the center of the earth
$\dot{\theta} =$ angular velocity of the moon

Table 1 gives a comparison of relative velocities. Clearly, if a satellite attempted a rendezvous by injecting itself into a circular orbit at the same altitude as the tether, it would quickly pass the tether. In order to rendezvous with the tether, the satellite and tether have to moving at approximately the same speed. However, if the satellites' speed is adjusted to match the tethers, there will be an immediate divergence of relative position. This is given by the fact that while the tether will be in a circular path around the earth, the satellite will be in an elliptic orbit around the earth. In order to determine the
<table>
<thead>
<tr>
<th>Radius (Km)</th>
<th>Velocity of Satellite in Circular Orbit (Km/sec)</th>
<th>Velocity of Tether (Km/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>48,050</td>
<td>2.889</td>
<td>.128</td>
</tr>
<tr>
<td>96,100</td>
<td>2.034</td>
<td>.256</td>
</tr>
<tr>
<td>144,150</td>
<td>1.663</td>
<td>.384</td>
</tr>
<tr>
<td>162,650</td>
<td>1.565</td>
<td>.433</td>
</tr>
<tr>
<td>165,400</td>
<td>1.552</td>
<td>.440</td>
</tr>
<tr>
<td>192,200</td>
<td>1.440</td>
<td>.512</td>
</tr>
<tr>
<td>288,300</td>
<td>1.176</td>
<td>.767</td>
</tr>
<tr>
<td>336,350</td>
<td>1.089</td>
<td>.859</td>
</tr>
</tbody>
</table>

Table 1. Relative Velocities of Tether and Satellite.

Note: Moon is assumed to be in a 388,400 Km circular orbit.
feasibility of a rendezvous, an analysis of the amount of time the tether and satellite are in close proximity to each other must be made. A MATLAB code was written to plot the relative position vectors of the tether and satellite. The code assumes that the satellite and tether rendezvous in the same orbital plane at apoapsis. Further, just before impact, a delta v is imparted to the satellite, tangent to its flight path, to match the velocity of the tether. Since the change in velocity is applied exactly at apoapsis, it has the effect of circularizing (+Δv) or "elliptizing" (−Δv) the orbit. But, it will not change the PQW axis orientation of the orbit. All information regarding the new orbit is known and the satellites subsequent position at different times are given by repeated solutions to Keplers' problem. See Appendix 1 for the MATLAB code used. Figures 2 and 3 show the relative position vectors and relative distances for the satellite and tether at the specified radii. A few comments about the plots. An iterative process was used to determine the amount of time it took for the satellite to drift 3 KM away from the tether after velocities were matched. The 3 KM distance was chosen arbitrarily as a limit. The idea is to determine how much time is available for the satellite and tether to be in close proximity to each other so that the satellites maneuvering thrusters could be used to effect a rendezvous. Figure 1 plots the time available vs radii for the 3 KM limit. As would be expected, very little time is available at closer distances to the earth due to the greater difference in velocities coupled with greater eccentricities of the new orbit. As the tether gets
farther from the earth, the time available increases and the satellites new orbit becomes more circular. This trend increases to the limit where the tether is the moon itself. The time available for rendezvous is infinite since both tether and satellite are in circular orbits. Figure 1 should be viewed cautiously at greater radii given that the problem is modeled on a 2 body system and the gravitational force of the moon is ignored.

Two interesting considerations concerning the length of the tether were also determined using the MATLAB code. First, through an iterative process, it was determined that at tether radii of 162,500 KM, any object falling off of the tether would be in an elliptic orbit about the earth that would have a periapsis radius of less than 50 NM from the earth’s surface. As a result, tether radii of less than 162,500 KM present a potential hazard should a catastrophic failure occur. Second, from an
Figure 2. Relative Position Vectors (___ = satellite, --- = tether)
Figure 3. Relative Distances
energy point of view, it was determined that at a tether radius of 165,400 KM the satellite would be at the tethers speed at apoapsis, negating the need for a delta v adjustment.

While the 3 KM limit was chosen to determine the amount of time available for a rendezvous, it does not solve the problem of how a rendezvous might be accomplished. The analysis thus far only gives an approximation of how much time the satellite and tether are in close proximity; given no other forces are acting on the satellite. In order to determine if a rendezvous may be accomplished, a study of the equations of motion between the satellite and tether must be made.
Figure 4. Vector Relationship Between Satellite and Tether

Once the satellite is in close proximity to the tether, it must maneuver relative to the tether in order to effect a rendezvous. If we take the earth to be an inertial frame and want to determine how a satellite moves relative to the tether we start with the position vector \( \mathbf{r} \),

\[
\mathbf{r}_i = (R_o + r) \hat{\mathbf{r}} + R_o \theta \hat{\mathbf{\theta}} + z \hat{\mathbf{z}}
\]

Taking the second derivative of this vector with respect to the
tether frame yields

\[ \ddot{\mathbf{r}} = \ddot{r}\hat{\mathbf{r}} + R_0 \ddot{\theta}\hat{\mathbf{\theta}} + \ddot{z}\hat{\mathbf{z}} \]

However, the acceleration of the position vector in an inertial frame relative to the tether is given by

\[ \ddot{\mathbf{r}} = \ddot{\mathbf{r}} + 2\mathbf{w}\times\dot{\mathbf{r}} + \mathbf{w}\times(\mathbf{w}\times\mathbf{r}) \]

which yields

\[ \ddot{\mathbf{r}} = \left[ \ddot{r} - 2R_0 \dot{\theta} \dot{w}_t - w_t^2(R_0 + r) \right]\hat{\mathbf{r}} + \left[ R_0 \ddot{\theta} + 2w_t \dot{r} - R_0 \theta w_t^2 \right]\hat{\mathbf{\theta}} + \ddot{z}\hat{\mathbf{z}} \]

where \( R_0 = \) tether radius
\( w_t = \) angular velocity of the tether

Since we are modeling a 2 body system, the only force acting on the satellite is given by

\[ a_g = -\frac{\mu \mathbf{r}}{r^3} = -\mu \left[ \frac{(R_0 + r)\hat{\mathbf{r}} + R_0 \theta \hat{\mathbf{\theta}} + z\hat{\mathbf{z}}}{[R_0^2 + 2R_0 \dot{r} + r^2 + R_0^2 \dot{\theta}^2 + z^2]^{3/2}} \right] \]

Using the binomial theorem to expand the denominator and neglecting higher order terms

\[ a_g = -\frac{\mu R_0 \hat{\mathbf{r}}}{R_0^3} - \frac{\mu}{R_0^3} \left[ -2r\hat{\mathbf{r}} + R_0 \theta \hat{\mathbf{\theta}} + z\hat{\mathbf{z}} \right] \]

NOTE: This analysis is presented for 2 objects in circular orbits in REF 2.
If an external force (thrust) is applied, the total acceleration is

$$\ddot{\mathbf{a}} = \ddot{\mathbf{a}}_o + \ddot{\mathbf{a}}_g$$

Finally, using Newton's second law

(1) \(\ddot{\mathbf{u}}_r: a_x + \ddot{r} + r(-w_t^2 - 2n^2) + \dot{\theta}(-2R_ow_t) + (-w_t^2R_o + n^2R_o) = 0\)

(2) \(\ddot{\mathbf{u}}_\theta: a_y + R_0\ddot{\theta} + \theta(-R_0w_t^2 + n^2R_o) + \dot{r}(2w_t) = 0\)

(3) \(\ddot{\mathbf{u}}_z: a_z + \ddot{z} + n^2z = 0\)

where \(n^2 = \mu/R_o^3\)

\(w_t\) = angular velocity of the tether

These equations are valid for small displacements in the radial and out of plane directions, however, they are correct for any magnitude change in the \(\mathbf{u}_\theta\) direction.

Equation (3) is a simple harmonic whose solution is

$$z(t) = z_o\cos(nt) + \dot{z}_o/n\sin(nt)$$

for initial conditions \(z_o, \dot{z}_o\) at \(t=0\). Equations (1) and (2) are coupled but can be solved numerically. Figure 5 shows some numerical solutions for different radii. The solutions assume no accelerations are applied to the satellite after its velocity is matched to the tethers. A comparison of these plots to the relative distance plots in Figure 3 show agreement. For the satellite to maintain a relative position with the tether, it must be continuously applying thrust to counteract the force of gravity from the earth. The magnitude of the thrust can be computed from the \(\dot{r}\) and \(\dot{\theta}\) solutions to equations (1) and (2). Therefore, while
it is possible to maintain relative position with the tether, it has to be done using a continuous burn. Consequently, orbital rendezvous of this nature will have to be relatively quick in order to conserve fuel resources.

Figure 5. Numerical solutions
The problem of phasing for the Hohmann transfer turned out to be very simple [REF 3]. In order for the satellite to be in the same location as the tether at apoapsis, the satellite and tether must be aligned such that both will be at the rendezvous point at apopasis of the transfer ellipse.

![Diagram of satellite and tether](image)

**Figure 6. Relative Positions for Hohmann Transfer**

This can be expressed mathematically as

$$P_{TO} = \frac{P_i - \alpha P_m}{\frac{2}{2\pi}}$$

Where $P_{TO}$ is the period of the transfer orbit and $P_m$ is the period of the tether (2380155.104 sec). Since the tether is a rigid body, its period will be the same as the period of the moon; regardless of the length of the tether. Expanding the
equation we find
\[
\frac{\pi((r_{po} + r_t)/2)^{3/2}}{\sqrt{\mu}} = \frac{\pi - \alpha(2360155.104 \text{ sec})}{2\pi}
\]

where \( r_{po} \) = radius of the parking orbit
\( r_t \) = radius of the tether

For the given model, we can fix the \( r_{po} = 6933.745 \) KM for a 300 NM orbit and solve for \( \alpha \). Figure 7 shows the plot of phase angle versus tether radii from 6933 KM to 384,400 KM (parking orbit to the moon).

\[
\alpha = \pi - \left[ \frac{2\pi}{2} \left( \frac{6933.745 + r_t}{\mu} \right)^{3/2} / 2360155.104 \right]
\]

Figure 7. Phase Angle Requirements

A problem arises when the tether and satellite are not aligned properly for the Hohmann transfer. As shown in Fig 7, the proper phase angle for a tether radius of 192,200 KM is \( 156^\circ \) (2.723 rad). But if the tether is not at that angle relative to the
quired until the two objects are at the required angle is called the Synodic Period and is given by

\[ P_s = \frac{2 \cdot \pi \cdot t}{w_s - w_t} \]

where \( w_s \) = angular velocity of the satellite in its 300 NM parking orbit.

\( w_t \) = Angular velocity of the tether

If the initial angular displacement is less than \( 2 \cdot \pi \), then the waiting period can be calculated by

\[ t_w = \frac{\text{delta alpha}}{w_s - w_t} \]

Since \( w_s \) and \( w_t \) are constant values, the waiting time is just a linear relationship and is plotted in Figure 8. With a maximum waiting time of 1.6 hours, phasing for rendezvous would not be a problem.

Figure 8. Waiting Time vs Phase Angle Error
APPENDIX ONE: MATLAB CODES
function [vt] = project1(radius)
rcs = 6378.145 + 555.6;
mu = 3.986012e5;
vc = sqrt(mu/rcs);
theta dot = 2*pi/(27.31661*24*3600);
vt(1) = radius*theta dot;
vcst = sqrt(mu/radius);
vt(6) = (rcs+radius)/2;
energy = -mu/(2*vt(6));
umin = sqrt(2*(energy+mu/radius));
ymax = sqrt(2*(energy+mu/rcs));
vt(2) = vt(1)-umin;
hol = (radius*umin);
vt(5) = sqrt(1+(2*energy*hol^2/mu^2));
pold = vt(6)*(1-vt(5)^2);
u = pi;
vt(8) = pi;
vr(4) = radius;
hnew = radius*vt(1);
vt(10) = hnew^2/mu;
energy new = (vt(1)*2/2)-(mu/vt(4));
vt(9) = sqrt(1+(2*energy new*hnew^2/mu^2));
vt(7) = vt(10)/(1-vt(9)^2);
newnu = acos((vt(10)/vt(4)-1)/vt(9));
vt(3) = pi;
rp=vt(7)*(1-vt(9))

%vt(1) = velocity of the tether
%vt(2) = delta v imparted to the satellite
%vt(3) = E eccentric anomaly immediately after delta v applied
%vt(4) = radius at delta v (i.e position at t=0)
%vt(5) = old eccentricity
%vt(6) = old a
%vt(7) = new a
%vt(8) = old eccentric anomaly
%vt(9) = new eccentricity
%vt(10) = pnew
function [post, poss] = position1(vt, deltat, radius, El)
    nu = acos(((vt(9) - cos(El)) / (vt(9) * cos(El) - 1)));
    nu = 2*pi - nu;
    r = vt(10) / (1 + vt(9) * cos(nu));
    poss(1) = r * cos(nu);
    poss(2) = r * sin(nu);
    thetadot = 2*pi / (27.31661 * 24 * 3600);
    thetat = deltat * thetadot;
    post(1) = -radius * cos(thetat);
    post(2) = -radius * sin(thetat);
MATLAB code written to solve the equations of motion for the satellite relative to the end of the elevator.

```matlab
F='def';
t0=0;
tf=1500;
r0=[.001; 0; 0; 0];
tol = 1.0e-10;
trace=1;
[t,r]=ode45(F,t0,tf,r0,tol,trace);
```
function dot=def(t,r)
K=240250; wt=.000002662; mu=3.986012e5; n=sqrt(mu/r0^3);
R=r(1); RD=r(2); TH=r(3); THD=r(4);
f1 = -R*(-wt*2-2*n^2) - THD*(-2*r0*wt) - (-wt^2*r0 + n^2*r0);
f2 = (-TH*(-r0*wt^2 + n^2*r0) - RD*2*wt)/r0;
    dot=[r(2); f1; r(4); f2];
REFERENCES

1. L.L. Van Zandt, "No Thanks, I'll Take the Elevator", E-Mail distribution from the Department of Physics, Purdue University, W. Lafayette, IN 47907


Satellite Maneuvering Options
for Short Notice Area-in-Space Avoidance

Prepared by
1Lt Anthony Nash
Mech 532
4 Dec 92
Introduction

Purpose:
The objective of this project was to analyze the effect of burn duration and direction on satellite maneuvering distances. The results were to be used to compare the effects of warning time, fuel usage, and thruster size on satellite survivability through evasive maneuvers. A comparison was to be made to the maneuvering distances achieved through instantaneous burns. The purpose of these short time-of-flight maneuvers is for satellite survivability. In many cases, the only defense a satellite has against and anti-satellite missile (ASAT), is to maneuver. Of particular interest, is the avoidance of an unguided, nuclear armed ICBM timed to intercept the satellite at some future time, in its highly predictable orbit. The satellite must maneuver a great distance in order to avoid the explosion, and more importantly, the radiation (this distance is designed into the satellite in the trade-off of maneuver vs shielding). Also, the satellite generally does not have long to maneuver, since thrusting will not begin until the ICBM launch is detected. This is the reason for the large number of variables (warning time, fuel usage, thruster size, satellite altitude, satellite mass, etc) in the project. All these variables effect not only the satellite survivability as analyzed in this project, but the ability of the satellite to perform its mission.

Approach:
There were several steps to this project. The first step was to calculate the maneuver distance for an impulsive burn. The second step was to write a Matlab program which would simulate the near continuous maneuvering and propagation of a satellite in its orbit. Finally, runs were made varying the burn time, the total time available, and the maneuvering thruster size.
Theory

Assumptions:

One of the primary assumptions made in this project was that the satellite was influenced by two body orbital mechanics with both the earth and the satellite acting as simple point masses. This means that the semi-major axis, the longitude of ascending node, and the argument of perigee were all constants. Although the variation of each of these parameters would change as the orbital elements are changed through maneuver, the time span in question was so small as to make these variations negligible.

Another assumption was the constant mass of the satellite. Although the satellite mass would decrease as the satellite maneuvers, the maneuver time was so short, and the motor efficiency so variable, I decided not to include this factor in the project. Motor efficiency is in actuality another variable which must be analyzed along with thrust, and time of burn.

A third assumption was that the satellite is three axis stabilized such that one axis always points in the direction of travel. This was a design choice, and effected the direction of thrust, given that all thrusters are fixed on the satellite's body. Changing the facing and stability of the satellite would change the rotation matrix used to convert from the body fixed frame to the IJK frame. An orientation for the satellite had to be chosen for the project. There is nothing special about this particular orientation.

One final assumption was that a continuous burn could be simulated by successive one second burns. This was an acceptable assumption since most satellites move less then one tenth of a degree in one second, making the spatial separation between burns small. Also, for time steps much smaller than this, round-off errors in the computer could have started to effect the results. The Matlab program is written such that the time step can be varied.

Mathematical Technique:

The first step in this project was to calculate the maneuver distance for an instantaneous burn. This was done by using the equation: \[ D = \Delta V \times t \] Where \( D \) is the total
distance moved, $\Delta V$ is the change in velocity, and $t$ is the time allowed for maneuver.

The second step was to calculate the nominal position of the satellite at the intercept point. The location of this point was known to be along the satellite's orbit, some $\Delta t$ from the start time of the maneuver. Given the satellites Classical Orbital Elements (COE's) before maneuver, propagating the satellite forward was performed by solving Kepler's equation: $\Delta M = n*\Delta t + M_0$, where $n = \sqrt{\mu/a^3}$, and $M_0 = E_0 - e*sinE_0$, and $E_0 = (e*cos(\nu))/(1 + e*\cos(\nu))$. Then, knowing the new $M$, I reversed the process (using a Newton iteration to solve for $E$) and found the new $\nu$. All the other COE's remained constant over this time since this was the case without maneuvering. I then calculated the new $R$ and $V$ vectors in the IJK frame. This $R$ vector is the position of the point to be avoided (or the center of the area to be avoided in the case of a nuclear ASAT).

Now that the point to be avoided is known, the location of the satellite after the maneuver must be calculated. Inputs to the program were satellite mass, thruster size and direction (in body fixed coordinates), and burn step time. The magnitude of the $\Delta V$ applied at each step time (one second for the analysis preformed) was: $\Delta V = \delta t*F/mass$, where $\delta t$ is the step time, and $F$ is the thruster force in Newtons. This was derived from Newton's equation: $F = ma$. Before this can be added to the velocity vector of the satellite, it must be rotated into the IJK frame. The rotation was done using two rotation matrices. The first rotated the $\Delta V$ vector from the body fixed frame into the PQW frame, and the second rotated it from there into the IJK frame. Knowing that body fixed frame was defined by: $dV_{sat}(1) = dV_{sat}(2) x dV_{sat}(3)$, $dV_{sat}(2) = V_{pqw}$ (velocity in the PQW frame), and $dV_{sat}(3) = W$ (perpendicular to the orbital plane). See figure #1. I was able to calculate the rotation angle between the body fixed frame and the PQW frame. It was found by taking the dot product, $\theta = \cos^{-1}[V_{pqw} \cdot Q/Q*V_{pqw}]$. The rotation matrix was then a three rotation through $C_{sp}$:

\[
C_{sp} = \begin{bmatrix}
\cos(\theta) & -\sin(\theta) & 0 \\
\sin(\theta) & \cos(\theta) & 0 \\
0 & 0 & 1
\end{bmatrix}
\]
Then, once I had the $\Delta V$ vector in the PQW frame, I used the rotation matrix found on pg 82 of BMW, to rotate $\Delta V$ into the IJK frame. Once in the IJK frame, the $\Delta V$ vector was added to the previous velocity vector of the satellite, to arrive at the new velocity vector of the satellite. A new velocity vector is calculated after each impulsive burn.

The next step is to propagate the satellite forward in its new orbit by the step size, $\delta t$. This was done by first finding the new COE's and then finding the change in the true anomaly using Kepler's equation as described above. Knowing the new COE's and the updated true anomaly, the new position and velocity vectors in the IJK frame can be computed. Another burn is then performed, the satellite velocity vector is updated, and the satellite is propagated forward $\delta t$ again. This process is repeated until the maximum allowed burn time is reached.

When the maximum burn time is reached, the satellite is propagated forward in its new orbit to the impact time. Then, the final position vector is calculated, as well as the distance from the impact point. The program is set up to increment the burn time from one second to the impact time (holding $\Delta V$ at each burn constant). This provides a curve of total $\Delta V$ versus distance from the impact point for a given thrust and time to impact. The total $\Delta V$ is the $\Delta V$ at each incremental burn, times the number of incremental burns. For $\delta V$ of one second: $\Delta V_{tot} = \Delta V \times t_{burn}$

Discussion

Results:

Once the program was written, the next step was to use the program to compare the small step size approximation of a continuous burn, to the instantaneous $\Delta V$ assumption, as well as to demonstrate the tradeoffs a satellite designer would face concerning altitude, thruster size, and satellite mass. To do this, I ran twelve test cases, divided into four scenarios, each with thrusting in all three directions. The four scenarios were: 1000 N thruster 30 seconds and 15 seconds before impact, and a 500 N thruster 30 seconds and 15 seconds before impact. The graphs of the results can be
seen in figures #2-5. For the sake of comparison, the following values were held constant for each test case:

mass of the satellite - \( m = 10,000 \) kg
initial classical orbital elements:
\[
\begin{align*}
a &= 7500 \text{ km} & \Omega &= 0 \text{ deg} \\
e &= 0 & \omega &= 0 \text{ deg} \\
i &= 0 \text{ deg} & \nu &= 0 \text{ deg}
\end{align*}
\]
step size - \( \delta t = 1 \) sec

Then, for each time to impact (30 sec and 15 sec), I graphed the distance verses the total \( \Delta V \) expended for a constant magnitude thrust in each of the three directions. This allowed me to compare on one graph, the relative maneuver efficiency for each direction of burn. I also recorded the maximum distance from the impact point, and the new classical orbital elements assuming burn time equalled impact time. This helped show how the direction of the thrust affected the orbit. The results of the twelve test cases are shown below with the thrust vector shown in body fixed coordinates.

### Case #1
figure #2; red
thrust = \([0; 1000; 0]\) N
impact time - 30 sec
\[
\begin{align*}
a &= 981,600 \text{ km} & \Omega &= 0 \text{ deg} \\
e &= 0.9924 & \omega &= 0.5241 \text{ deg} \\
i &= 0 \text{ deg} & \nu &= 0.4632 \text{ deg}
\end{align*}
\]
maximum distance - 89.48 km

### Case #2
figure #2; green
thrust = \([1000; 0; 0]\) N
impact time - 30 sec
\[
\begin{align*}
a &= 7542.6 \text{ km} & \Omega &= 0 \text{ deg} \\
e &= 0.4004 & \omega &= 248.10 \text{ deg} \\
i &= 0 \text{ deg} & \nu &= 113.97 \text{ deg}
\end{align*}
\]
maximum distance - 83.55 km

### Case #3
figure #2; blue
thrust = \([0; 0; 1000]\) N
impact time - 30 sec
\[
\begin{align*}
a &= 7542.6 \text{ km} & \Omega &= 0.8183 \text{ deg} \\
e &= 0.0056 & \omega &= 0.0115 \text{ deg} \\
i &= 23.54 \text{ deg} & \nu &= 0.8624 \text{ deg}
\end{align*}
\]
maximum distance - 46.18 km
Case #4
figure #3; red
thrust = [0; 500; 0] N
impact time - 30 sec
a = 13,732 km Ω = 0 deg
e = 0.4538 ω = 0.6649 deg i = 0 deg ν = 0.7501 deg
maximum distance - 33.49 km

Case #5
figure #3; green
thrust = [500; 0; 0] N
impact time - 30 sec
a = 7510.6 km Ω = 0 deg
e = 0.2045 ω = 259.46 deg i = 0 deg ν = 102.39 deg
maximum distance - 36.69 km

Case #6
figure #3; blue
thrust = [0; 0; 500] N
impact time - 30 sec
a = 7510.6 km Ω = 0.8102 deg
e = 0.0014 ω = 0.0029 deg i = 11.78 deg ν = 0.8630 deg
maximum distance - 23.21 km

Case #7
figure #4; red
thrust = [0; 1000; 0] N
impact time - 15 sec
a = 7713 km Ω = 0 deg
e = 0.4538 ω = 0.3187 deg i = 0 deg ν = 0.3839 deg
maximum distance - 17.37 km

Case #8
figure #4; green
thrust = [1000; 0; 0] N
impact time - 15 sec
a = 7521.2 km Ω = 0 deg
e = 0.2043 ω = 259.41 deg i = 0 deg ν = 101.52 deg
maximum distance - 18.84 km

Case #9
figure #4; blue
thrust = [0; 0; 1000] N
impact time - 15 sec
a = 7521.2 km Ω = 0.3910
e = 0.0028 ω = 0.0014 deg i = 11.78 deg ν = 0.4453 deg
maximum distance - 11.97 km
Case #10
figure #5; red
thrust = [0;500;0] N
impact time - 15 sec
a = 9570.5 km Ω = 0 deg
e = 0.2163 ω = 0.3542 deg
i = 0 deg ν = 0.4253 deg
maximum distance - 7.31 km

Case #11
figure #5; green
thrust = [500;0;0] N
impact time - 15 sec
a = 7505.3 km Ω = 0 deg
e = 0.1027 ω = 264.90 deg
i = 0 deg ν = 95.98 deg
maximum distance - 8.92 km

Case #12
figure #5; blue
thrust = [0;0;500] N
impact time - 15 sec
a = 7505.3 km Ω = 0.3901 deg
e = 0.0071 ω = 0.0003 deg
i = 5.8934 deg ν = 0.4455 deg
maximum distance - 5.99 km

In general, thrusting in the dVsat(1) direction only changed the shape of the orbit, by making it elliptical, but without changing the semi-major axis. Thrusting in the dVsat(2) direction changes the semi-major axis and eccentricity of the orbit a great deal (See figure #6). Finally, thrusting in the dVsat(3) direction only changes the inclination of the orbit.

As can be seen from the graphs, thrusting out-of-plane, maneuvers the satellite the least. In the four cases analyzed, the out-of-plane burn only maneuvered the satellite 50%-80% of the maximum maneuver distance. The benefit of this maneuver is that it effects the orbit the least. Since the maneuver is out of plane, the vast majority of the energy goes into changing the inclination of the orbit, which is the most energy expensive orbital maneuver. The most efficient maneuver was the in-plane thrust, perpendicular to the velocity vector (in the dVsat(1) direction). This had the advantage of changing the orbital shape (increasing eccentricity and regressing the argument of perigee) while actually moving the satellite away from the intercept point. This second factor primarily effects the maneuver distance for short intercept times. Thrusting in the direction of the velocity (dVsat(2) direction) greatly increases the semi-major axis of the orbit, increasing the velocity of the satellite at perigee, near which most of the intercepts occurred. Unfortunately, maneuvering in this direction also accelerate the satellite
toward the intercept point. The primary reason for the abnormality on figure #2 (where the red line crosses the green) is because a ΔV of 3 km/sec in the dVsat(2) direction is almost enough to create a parabolic orbit (e = 0.9945). So, at this point, the satellite is barely in an elliptical orbit, and is travelling at 10.3 km/sec!

For long intercept times (approaching one half the period of the original orbit) maneuvering in the dVsat(2) direction will be the most beneficial since the semi-major axis will be vastly larger, the period will be longer, and the distance between the apogees of the two orbits will be far apart, as well as occurring at different times. By thrusting in the dVsat(1) direction, the period of the orbit is unchanged, but the eccentricity is changed. One half a period later, the satellite will once again intercept the original orbit. This is the same with the out-of-plane thrusting. So, if the intercept period is one half the original orbital period, the only way to survive is to burn in the dVsat(2) direction.

The comparison to the instantaneous burn can really only be done for the thrusting in the direction of velocity. The distances achieved for the four scenarios are:

<table>
<thead>
<tr>
<th></th>
<th>Distance Achieved (km)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>instantaneous</td>
</tr>
<tr>
<td>1000 N, 30 sec</td>
<td>90.0</td>
</tr>
<tr>
<td>1000 N, 15 sec</td>
<td>22.5</td>
</tr>
<tr>
<td>500 N, 30 sec</td>
<td>45.0</td>
</tr>
<tr>
<td>500 N, 15 sec</td>
<td>12.0</td>
</tr>
</tbody>
</table>

In all cases for the same ΔV, the instantaneous maneuver had a greater maneuver distance than the iterative maneuver. For the short times-of-flight used in the test cases, it is close enough for use as a first cut.
Errors:

There are several areas for error in this project. The first is that I did not integrate the actual equations of motion for a thrusting satellite. Instead, I used iterative impulsive thrusts, one second apart. This is a reasonable approximation since one second is a small percent of the orbital period. By using this method, additional errors were introduced. These additional errors primarily involve round-off errors by the computer. Because one second is such a small percent of the orbital period, the new position and velocity vectors calculated every second, should be only slightly different from the position and velocity vectors from one second earlier. In fact, they may not be much more than the round-off errors encountered in calculating these numbers. After compounding these errors for each second of iteration, they may actually become noticeable. Fortunately, the test cases examined were limited to 30 seconds duration, of one second step sizes, which is short enough that the round-off errors are not significant. However, if a much smaller step size were used, or a much longer intercept time (approaching a period), the round-off errors could become a factor.

Conclusions

The first goal of this project was to examine the validity of an achievable maneuver distance estimated by instantaneous burns. Since this approximation assumes thrusting along the velocity direction, that is the set of results against which it was compared. It turns out that the instantaneous burn approximation is a fairly good first cut. In fact, it could be very useful to determine if maneuvering is even a viable option, particularly for a satellite, already in orbit, with a known ΔV capability. But, it is only a first cut, and does not contain enough detail to assist the satellite designer. That approximation does not take into account thrust requirements, fuel expenditures, the new orbital elements, or even the direction of the maneuver. The iterative method employed in this project allows the satellite designer to see how changes in altitude (which effects time to impact), and thrust capabilities, for example, effect the survivability of the satellite. Additionally, the iterative method shows
what the final orbit will look like after maneuver, telling
the designer whether the satellite must be maneuvered back
into its original orbit, or if it is in a close enough orbit
to perform its mission.

The second goal of this project was to analyze the
effects of maneuver direction on maneuver distance. Not
surprisingly, the direction of the maneuver greatly effects
the maneuver distance. The instantaneous \( \Delta V \) approximation
method assumed that maneuvering in the direction of motion
was the most efficient. The iterative method showed that in
fact, maneuvering in the direction of motion (for small \( \Delta V \)
and short times) does not provide the largest distance from
impact. Maneuvering in the dVsat(1) direction
(perpendicular to the motion) provides the largest distance
from impact. As was expected, out of plane maneuvers were
the least efficient, but also perturbed the orbit the least.

These are just general conclusions. The real decision
on the optimal direction of maneuver would depend upon the
actual satellite, its function, and even the type of
interceptor (speed and warhead size) going after it. The
program was written general enough that these parameters can
be entered and the results analyzed to determine what works
best for each particular scenario.
\( \mathbf{v}_{\text{sat}(2)} \times \mathbf{v}_{\text{sat}(3)} = \mathbf{v}_{\text{sat}(1)} \)

\( \mathbf{v}_{\text{sat}(3)} \) is perpendicular to the orbital plane.
\( \mathbf{v}_{\text{sat}(1)} \) and \( \mathbf{v}_{\text{sat}(2)} \) are in the orbital plane.
\( \mathbf{v}_{\text{sat}(2)} \) is in the V direction.

Satellite Body Frame

Figure #1
Figure 3

Thrust = [0, 0.5, 1.0 N - blue]
Thrust = [5.0, 10.0, 50.0 N - green]
Thrust = [50.0, 100.0, 500.0 N - red]

Delta V (km/sec)

5G0 N for 30 sec

70
60
50
40
30
20
10
0
Figure 4

Thrust = [0, 8, 1000] N - Blue
Thrust = [1000, 0, 0] N - Green
Thrust = [0, 1000, 0] N - Red

DETA \ A

1000 N FOR 15 SEC
```plaintext
This is the maneuver main program which propagates a satellite forward by a delta t, computes a new orbit given a delta V, and propagates the satellite forward along the new orbit by the same delta t. Finally, the final distance between the two satellites at t+delta t, is computed.

Variables
all angles are in degrees, distances in km and time in seconds
a - semi-major axis
ecc - eccentricity
inc - inclination
omega - longitude of ascending node
w - argument of perigee
nu - true anomaly
R,V - position and velocity vectors in geocentric equatorial (IJK) coordinates at time = t (before maneuvering or propagating)
Ro - position vector in geocentric equatorial (IJK) coordinates at time = impact (without burn)
Vo - velocity vector in geocentric equatorial (IJK) coordinates at time = impact (without burn)
Rn, Vn - new position and velocity vectors after delta V and at time = impt in geocentric equatorial (IJK) coordinates
dist - distance between original satellite position and the new satellite position after delta V at impact
dVsat - delta V of the satellite expressed in the body fixed, satellite centered (roll/pitch/yaw) frame
a0,ecc0,inc0,omega0,w0,nu0 - COE's of original orbit after delta t (no burn)
t - elapsed time of the burn
an,eccn,incn,omegan,wn,nun - COE's of orbit after burn (no delta t)
abt,eccbt,incbt,omegabt,wbt,nubt - COE's of orbit after burn and delta t thrustt - total time of thrust (sec)
mass - mass of satellite (kg)
thrust - force of thrust of the engines maneuvering the satellite (N)
imp - time until impact with satellite (sec)
totdV - total delta V imparted (km/sec)
coast - time between completion of burn, and impact (sec)

first, define all the inputs to the program
```

Page 1
a = 6600; 
 eccentric = 0; 
 inclination = 0; 
 argument of perigee = 0; 
 true anomaly = 0; 
 mean anomaly = 0; 
 initial mean time = 15; 
 thrust = [0;0;1000] \% 1 lbf = 4.448 N 
 mass = 10000; 

% define the step interval (sec) 
 delt = 1; 

% determine the amount of delt per step interval 
 dVs = thrust/mass*delt; 
 initialize totdV 
 for i = 1:impt, 
 totdV(i) = 0; 
 end 

% convert original orbit to R and V vectors 
 [Rstart,Vstart] = randv(a,ecc,inc,omega,w,nu); 

% propagate the original orbit forward by the impact time 
 [a0,ecc0,inc0,omega0,w0,nu0] = prop(a,ecc,inc,omega,w,nu,impt); 

% compute the position and velocity vectors of original orbit 
 at impact time 
 [Ro,Vo] = randv(a0,ecc0,inc0,omega0,w0,nu0); 

% iterate for increasing thrust times but constant impact times 
 for thrustt = 1:impt, 

% redefine initial conditions 
 Rburn = Rstart; Vburn = Vstart; eccbt = ecc; nutbt = nu; 
 abt = a; incbt = inc; omegabt = omega; wbt = w; 

% Now to perform the maneuver 
% burn once per time step for the duration of the thrust time 
 for t = 1:thrustt, 

% convert delta V into IJK coordinates and compute new V 
 [dV] = deltav(Rburn,Vburn,dVs); 
 Vburn = Vburn + dV;
propogate Rburn and Vburn forward by deltat

[Rburn, Vburn] = burn(Rburn, Vburn, deltat);
magdV = norm(dV);
totdV(thrustt) = totdV(thrustt) + magdV;
end

[abt, eccbt, incbt, omegabt, wbt, nubt, arglebt, lonebtl, lonprbt] = elorb(Rburn, Vburn);

now, propagate the satellite forward to the impact point
after all burns have been completed (coast phase)

coast = impt - thrustt;
[an, eccn, incn, omegan, wn, nun] = prop(abt, eccbt, incbt, omegabt, wbt, nubt, coast);

[Rn, Vn] = randv(an, eccn, incn, omegan, wn, nun);
al(thrustt) = an; eccl(thrustt) = eccn; incl(thrustt) = incn;
omegal(thrustt) = omegan; w1(thrustt) = wn; nul(thrustt) = nun;

compute the IJK vector from the non maneuvering satellite
to the maneuvering satellite after delta t
also compute the COE's after all burns and propogation are complete

rIJK = Ro - Rn;
dist(thrustt) = norm(rIJK);
end
RANDV.M

\% RANDV
\% This program takes the classical orbital elements,
\% and converts them into R and V vectors expressed in
\% the inertial Geocentric Equatorial (IJK) frame.
\% Written by: Tony Nash Oct 92
\% Variable definitions
\% a - semi major axis (km)
\% ecc - eccentricity
\% inc - inclination (deg)
\% omega - longitude of ascending node (deg)
\% w - argument of perigee (deg)
\% nu - true anomaly (deg)
\% R - position vector in IJK (km)
\% V - velocity vector in (km/sec)
\% h - angular momentum vector
\% n - line of nodes vector
\% C - 3x3 rotation matrix to convert PQW to IJK
\% function [R,V] = randv(a,ecc,inc,omega,w,nu)
\% The gravitational constant is (km^3/sec^2)
\% u = 3.986004418*10^5;
\% convert the angles from degrees to radians
\% rad = pi/180.0; \% converts deg to rad
\% inc = inc*rad;
\% omega = omega*rad;
\% w = w*rad;
\% nu = nu*rad;
\% convert to PQW coordinates
\% Rpqw - position vector in PQW coordinates (km)
\% Vpqw - velocity vector in PQW coordinates (km/sec)
\% p - semi latus rectum
\% p = a*(1.0 - ecc^2);
\% position vector in PQW
\% Rpqw = [p*cos(nu)/(1.0 + ecc*cos(nu))
\% p*sin(nu)/(1.0 + ecc*cos(nu))
\% 0];
\% velocity vector in PQW
\% Vpqw = [-sqrt(u/p)*sin(nu)
\% sqrt(u/p)*(ecc + cos(nu))
0];
%
% Now to convert Rpqw and Vpqw to RIJK and VIJK
%
% The rotation matrix to convert PQW to IJK is:
%
    C = [cos(omega)*cos(w)-sin(omega)*sin(w)*cos(inc) -cos(omega)*sin(w)-sin(omega)*cos(w)*sin(inc)
         sin(omega)*cos(w)*sin(inc) cos(omega)*cos(omega)*sin(inc) -sin(omega)*sin(w)+c
         cos(omega)*cos(w)*cos(inc) -cos(omega)*sin(omega)*sin(inc)
         sin(w)*sin(inc) cos(w)*sin(inc) cos(inc)];
%
% The position vector in IJK coordinates is:
%    R = C*Rpqw;
%
% The velocity vector in IJK coordinates is:
%    V = C*Vpqw;
%
% Finally, I need to return all angles to deg
%
    deg = 1.0/rad;
    inc = inc*deg;
    omega = omega*deg;
    w = w*deg;
    nu = nu*deg;
propagation (prop)

This subroutine propagates true anomaly forward in time. This routine assumes no external perturbations and that all the orbital elements remain constant.

Variables

nu - true anomaly (deg)
mean0 - mean anomaly at start time (deg)
mean - mean anomaly (deg)
E0 - eccentric anomaly (deg)
deltat - time through which nu must propagate (sec)
ecc - eccentricity of the orbit
a - semi-major axis of the orbit
N - mean motion of the satellite

function [a, ecc, inc, omega, w, nu] = prop(a, ecc, inc, omega, w, nu, deltat);
define the gravitational parameter
u = 3.986012*10^5;
convert nu from degrees to radians
rad = pi/180;
nu = nu*rad;
define the mean motion
N = sqrt(u/a^3);
convert nu to E0
E0 = acos((ecc*cos(nu))/(1+ecc*cos(nu)));
convert E0 to mean
mean = E0 - ecc*sin(E0);
if nu > pi,
    mean = 2*pi - mean;
end
propagate mean forward by time deltat
dmean = N*deltat;
while dmean > 2*pi,
    dmean = dmean - 2*pi;
end

now, convert back to true anomaly
I must solve Kepler's equation: mean = E0 - ecc*sin(E0) for E0
mn - trial values for mean
En,En1 - trial values for E0

dE0 = dmean;
while dE0-ecc*sin(dE0)-dmean > 10^(-9),
\[ dE_0 = dE_0 - \frac{(dE_0 - e \cdot \sin(dE_0) - d\text{mean})}{(1 - e \cdot \cos(dE_0))}; \]

end

while \( dE_0 > 2\pi \),
    \( dE_0 = dE_0 - 2\pi \);
end

\[ d\nu = \arccos\left(\frac{\cos(dE_0) - e}{1 - e \cdot \cos(dE_0)}\right); \]

if \( dE_0 > \pi \),
    \( d\nu = 2\pi - d\nu \);
end

\[ \text{nu} = \text{nu} + d\nu; \]

\% finally, converting all angles to degrees

\[ \text{deg} = \frac{180}{\pi}; \]

\[ \text{nu} = \text{nu} \cdot \text{deg}; \]

\[ \text{mean} = \text{mean} \cdot \text{deg}; \]

\[ E_0 = E_0 \cdot \text{deg}; \]
Vpqw = [-sqrt(u/p)*sin(nu)
         sqrt(u/p)*(ecc + cos(nu))
         0];

% determine the angle between the body centered frame and the
% PQW frame to compute rotation matrix
magVp = norm(Vpqw);
theta = acos(Vpqw(2)/magVp);

% compute the rotation matrix from Vsat to PQW
Csp = [cos(theta) -sin(theta) 0; sin(theta) cos(theta) 0; 0 0 1];

% The rotation matrix to convert PQW to IJK is:
C = [cos(omega)*cos(w)-sin(omega)*sin(w)*cos(inc) -cos(omega)*sin(w)-sin(omega)*cos(w)*cos(inc) sin(omega)*cos(w)+cos(omega)*sin(w)*cos(inc) -sin(omega)*sin(w)*cos(inc) -cos(omega)*sin(w)*sin(inc) sin(omega)*cos(w)*sin(inc) cos(omega)*sin(w)*cos(inc)];

% rotate dVsat to dVpqw
dVpqw = Csp*dVsat;

% rotate dVpqw to dV (dV is in IJK frame)
dV = C*dVpqw;
% burn
% This function predicts the position and velocity
% vectors in the IJK frame given the present postion
% and velocity vectors, and the change in time.
% It is assumed that t = 0 is epoch time, and therefore,
% t = dt, nu = dnu, E = dE and m = dm

function [Rn,Vn] = burn(R,V,dt)
% define all constants
  u = 3.986012*10^5; % km^3/sec^2
  magR = norm(R);
  magV = norm(V);
% now, compute the COE's of the orbit to obtain ecc, and a
% [a,ecc,inc,omega,w,nu,argle,lonep,lompr] = elorb(R,V);
  nu = nu*pi/180;
% mean motion, n
  n = sqrt(u/a^3);
% compute dnu, dm, and dE
% propagate mean forward by time deltat
  dm = n*dt;
  while dm > 2*pi,
    dm = dm - 2*pi;
  end
% now, convert back to true anomaly
% I must solve Kepler's equation: m = E - ecc*sinE for E
  dE = dm;
  while dE - ecc*sin(dE) - dm > 0.00001,
    dE = dE - (dE - ecc*sin(dE) - dm)/(1 - ecc*cos(dE));
  end
  while dE > 2*pi,
    dE = dE - 2*pi;
  end
  dnu = acos((cos(dE)-ecc)/(1-ecc*cos(dE)));
  if dE > pi,
\[ dnu = 2\pi - dnu; \]
end
\[ nul = nu + dnu; \]
\[ nul = nul * 180 / \pi; \]
\[ [Rn, Vn] = randv(a, ecc, inc, omega, w, nul); \]
ELORB

This procedure accepts the position and velocity vectors expressed in the Geocentric-Equatorial frame, and determines the classical orbital elements. All angles are in degrees, and the semi-major axis is in KM.

Variables

R - position vector of the satellite in IJK frame (km)
V - velocity vector of the satellite in IJK frame (km/sec)
a - semi-major axis (km)
ecc - eccentricity (magnitude of ecc)
inc - inclination (deg)
omega - longitude of ascending node (deg)
w - argument of perigee (deg)
nu - true anomaly (deg)
mean - mean anomaly (deg)
angle - argument of latitude (deg)
lonpr - longitude of periapsis (deg)
lonep - longitude at epoch (deg)
h - angular momentum vector
E - eccentricity vector
p - semi-latus rectum
n - nodal vector

function [a, ecc, inc, omega, w, nu, angle, lonpr, lonep] = elorb(R, V);

The gravitational constant is
u = 3.986012*10^5;

The magnitude of the position and velocity vectors
magR = norm(R);
magV = norm(V);

Next, determine h, n, and ecc

The angular momentum vector is
h = [R(2)*V(3)-R(3)*V(2); R(3)*V(1)-R(1)*V(3); R(1)*V(2)-R(2)*V(1)];
magh = norm(h);
phi = acos(magh/(magR*magV));

The nodal vector is
n = [-h(2);h(1);0];
magn = norm(n);

some useful constants
c1 = magV^2/u - 1/magR;
c2 = (R(1)*V(1) + R(2)*V(2) + R(3)*V(3))/u;
% The eccentricity vector and its magnitude
E = [c1*R(1) - c2*V(1); c1*R(2) - c2*V(2); c1*R(3) - c2*V(3)];
ec = norm(E);
%
Check for parabolic and hyperbolic orbits
%
if ecc > 0.9999,
a = -1;
%
determine semi-major axis if orbit is ellipse
else a = -u/(magV^2 - (2*u/magR));
end
% determine inclination of orbit
inc = acos(h(3)/magh);
%
% check for circular and/or equatorial orbits
%
if ecc < 0.000001,
w = 0;
%
circular and equatorial
%
if abs(h(3)/magh) > 0.99999,
omega = 0;
lonep = acos(R(1)/magR);
if R(2) < 0,
lonep = 2*pi - lonep;
end
nu = lonep;
end
%
circular and inclined
%
if abs(h(3)/magh) < 0.99999,
omega = acos(n(1)/magn);
if n(2) < 0,
omega = 2*pi - omega;
end
ndotr = n(1)*R(1) + n(2)*R(2) + n(3)*R(3);
argle = acos(ndotr/(magn*magR));
if R(3) < 0.0,
argle = 2*pi - argle;
end
nu = argle;
end
end

if ecc > 0.000001,
    if abs(h(3)/magh) > 0.99999,
        inc = 0;
        omega = 0;
        lonpr = acos(E(1)/ecc);
        if E(2) < 0,
            lonpr = 2*pi - lonpr;
        end
    w = lonpr;
    edotr = E(1)*R(1) + E(2)*R(2) + E(3)*R(3);
    nu = acos(edotr/((ecc*magR)));
    rdotv = R(1)*V(1) + R(2)*V(2) + R(3)*V(3);
    if rdotv < -0.001,
        nu = 2*pi - nu;
    end
end

if abs(h(3)/magh) < 0.99999,
    omega = acos(n(1)/magn);
    if n(2) < 0.0,
        omega = 2*pi - omega;
    end
    ndote = n(1)*E(1) + n(2)*E(2) + n(3)*E(3);
    w = acos(ndote/(magn*ecc));
    if E(3) < 0.0,
        w = 2*pi - w;
    end
    edotr = E(1)*R(1) + E(2)*R(2) + E(3)*R(3);
    nu = acos(edotr/((ecc*magR)));
    rdotv = R(1)*V(1) + R(2)*V(2) + R(3)*V(3);
    if rdotv < 0.0,
        nu = 2*pi - nu;
    end
    argle = w + nu;
    while argle > 2*pi,
        argle = argle - 2*pi;
    end
    lonpr = omega + w;
    while lonpr > 2*pi,
        lonpr = lonpr - 2*pi;
    end
    lonep = omega + w + nu;
while lonep > 2*pi,
    lonep = lonep - 2*pi;
end
end

% now, to convert all the angles from radians to degrees

deg = 180/pi;
inc = inc*deg;
omega = omega*deg;
w = w*deg;
nu = nu*deg;
mean = mean*deg;
E0 = E0*deg;
argle = argle*deg;
lonep = lonep*deg;
lonpr = lonpr*deg;
DELTA V

This procedure adds the delta V of an instantaneous burn to the existing velocity of the satellite. This procedure takes a delta V as applied in body centered roll/pitch/yaw coordinates and changes it into a delta V in inertial geocentric equatorial (IJK) coordinates.

Variables
all velocities are in km/sec, all angles in deg

dVsat(1) - delta V of the satellite along the local verticle - aircraft pitch \{dVsat(1) = dVsat(2) \times dVsat(3)\}
dVsat(2) - delta V of the satellite in the direction of motion - aircraft roll
dVsat(3) - delta V of the satellite perpendicular to the orbital plane - aircraft yaw
dVpqw - delta V of the satellite in the perifocal (PQW) frame
dV - delta V of the satellite in the IJK frame
Vpqw - velocity of the satellite (PQW) before delta V
Csp - coordinate transformation matrix from body centered to PQW
C - coordinate transformation matrix from PQW to IJK
phi - angle between body centered frame and PQW frame

function \([dV] = \text{deltav}(R,V,dVsat)\);
The gravitational constant is (km^3/sec^2)
u = 3.986012 \times 10^5;
magR = norm(R);
magV = norm(V);

determine the COE's
[a,ecc,inc,omega,w,nu,argle,lonep,lonpr] = elorb(R,V);
p = a*(1 - ecc^2);

convert all COE angles to radians
rad = pi/180;
inc = inc*rad;
omega = omega*rad;
w = w*rad;
u = nu*rad;

now, compute the velocity vector in PQW