θ

**r**(t)

**W1**

**W2**

θ

**r**(t0)

**W2**

**W1**

**r**(t)

Using the images above, **r**(t) in terms of the basis {**w1, w**2**, w2**} can be expressed using vector decomposition as:

1. The inertial velocity can be found by taking the derivative in I of **r**(t).

Computing the derivative, using the product and chain rule, results in:

1. Since the orbit is circular, the semi-major axis and radius are equal, speed of the orbit can be determined using the vis-viva equation:

Substituting a in for r and simplifying:

We can also find speed in a different fashion. We can take the derivative of the position vector and apply the transport theorem. Starting with the position vector:

Where is part of an orthonormal basis in which is always pointing towards the spacecraft, is along , and is equal to . This basis rotates with the spacecraft with an angular velocity of .

Taking the derivative in I of **r:**

Since isn’t expressed in I, the transport theorem must be applied.

Taking the magnitude of this velocity vector to find speed:

Equating the two velocity equations:

Solving for :

a and are constants in all orbits which means is also constant.

Since has the units of radians per second, can be found by multiplying by time elapsed:

d)

function [times, positions, velocities] = propagateOnCircle(pos,vel,t0,tf,mu,N)

%This function calculates position and veolocity N times evenly spaced betwen t0 and tf

%

%Function call: [times, postions, velocities] = propagateOnCircle(pos,vel,t0,tf,mu,N);

%

%Input: pos, a column vector with the initial ECI postion

%Input: vel, a column vector with the initial ECI inertial velocity

%Input: t0, the initial time. Time from apoapsis to initial postion

%Input: tf, the final time

%Input: mu, gravitational parameter

%Input: N, number of time intervals

%

%Output: times, column vector of times, starting at t0 and ending at tf. Length of N with even spacing

%Output: postions, matrix of size Nx3 containing ECI positions stored row-wise at every time in the times vector

%Output: velocities, matrix of size Nx3 containing ECI velocities stored row-wise at every time in the times vector

%

times=linspace(t0,tf,N);

times=times';

oe = rv2oe\_Hackbardt\_Chris(pos,vel,mu);

nui=oe(6);

a=oe(1);

thetaDot=sqrt(mu/a^3);

positions=zeros(N,3);

velocities=zeros(N,3);

positions(1,:)=pos';

velocities(1,:)=vel';

for i=2:N

theta=thetaDot\*(times(i)-times(i-1));

nuOfT=nui+theta;

oe(6)=nuOfT;

[rPCI,vPCI] = oe2rv\_Hackbardt\_Chris(oe,mu);

positions(i,:)=rPCI';

velocities(i,:)=vPCI';

nui=nuOfT;

end

end

e)

Time X Position Y Position Z Position X Velocity Y Velocity Z Velocity

\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

1992 -5614 -2446.4 2600.5 2.1276 -7.1342 -2.1184

2083.4 -5388.3 -3083.2 2392.6 2.8086 -6.792 -2.4274

2174.7 -5101.7 -3685.2 2157.8 3.4578 -6.3731 -2.709

2266.1 -4757.6 -4245.6 1898.5 4.0679 -5.8823 -2.9601

2357.5 -4359.8 -4758 1617.9 4.6322 -5.3251 -3.1777

2448.8 -3912.8 -5216.8 1319 5.1442 -4.7077 -3.3595

2540.2 -3421.6 -5616.6 1005.2 5.5981 -4.0373 -3.5033

2631.5 -2891.8 -5953.1 680.04 5.9888 -3.3212 -3.6076

2722.9 -2329.3 -6222.4 347.22 6.312 -2.5677 -3.6712

2814.3 -1740.6 -6421.4 10.479 6.5638 -1.7852 -3.6933

2905.6 -1132.2 -6548 -326.38 6.7416 -0.98257 -3.6738

2997 -511.03 -6600.6 -659.55 6.8433 -0.16883 -3.6127

3088.4 115.91 -6578.8 -985.28 6.8678 0.64682 -3.5109

3179.7 741.54 -6482.6 -1299.9 6.8147 1.4552 -3.3695

3271.1 1358.8 -6313.4 -1599.8 6.6847 2.2471 -3.1901

3362.5 1960.7 -6072.8 -1881.7 6.4793 3.0137 -2.9746

3453.8 2540.5 -5763.7 -2142.3 6.2007 3.7462 -2.7256

3545.2 3091.6 -5389.6 -2378.8 5.8522 4.4365 -2.4457

3636.5 3607.8 -4954.6 -2588.4 5.4375 5.0766 -2.1383

3727.9 4083.3 -4463.7 -2768.8 4.9615 5.6595 -1.8068

3819.3 4512.7 -3922.4 -2917.9 4.4295 6.1785 -1.4548

3910.6 4891.2 -3336.8 -3034.1 3.8475 6.6278 -1.0865

4002 5214.5 -2713.6 -3116.1 3.2221 7.0022 -0.70585

4093.4 5478.9 -2059.7 -3162.9 2.5603 7.2976 -0.31726

4184.7 5681.4 -1382.6 -3174 1.8696 7.5107 0.074919

4276.1 5819.9 -689.9 -3149.2 1.1578 7.6389 0.46625

4367.5 5892.6 10.6 -3088.9 0.43297 7.681 0.85232

4458.8 5898.8 710.98 -2993.8 -0.29678 7.6363 1.2288

4550.2 5838.5 1403.3 -2864.8 -1.0232 7.5055 1.5913

4641.5 5712.2 2079.9 -2703.5 -1.738 7.2899 1.936

4732.9 5521.5 2732.9 -2511.7 -2.4333 6.9921 2.2587

4824.3 5268.4 3355.1 -2291.6 -3.101 6.6153 2.556

4915.6 4955.9 3939.4 -2045.6 -3.7338 6.1638 2.8244

5007 4587.4 4479.3 -1776.4 -4.3244 5.6428 3.0609

5098.4 4167.2 4968.6 -1487.3 -4.8662 5.0581 3.2629

5189.7 3699.9 5401.8 -1181.3 -5.3531 4.4163 3.4281

5281.1 3190.9 5774 -862.06 -5.7796 3.7246 3.5546

5372.5 2645.8 6081.1 -533.05 -6.1408 2.991 3.6409

5463.8 2070.9 6319.5 -198.03 -6.4327 2.2235 3.6861

5555.2 1472.6 6486.6 139.24 -6.652 1.4309 3.6898

5646.5 857.68 6580.5 474.93 -6.7962 0.62224 3.6518

5737.9 233.08 6600.1 805.26 -6.8637 -0.19348 3.5725

5829.3 -394.15 6545.2 1126.5 -6.8538 -1.007 3.453

5920.6 -1016.9 6416.5 1435 -6.7665 -1.8092 3.2945

6012 -1628.2 6215.3 1727.4 -6.6027 -2.591 3.0987

f-g)

Chart

Description automatically generated