Orbital Dynamics Simulation of a LEO CubeSat

Andrew Lindburg

February 21, 2024

Abstract

This document outlines the methodology and considerations for simulating the orbital dynamics of a CubeSat in Low Earth Orbit (LEO). It covers the initial orbital parameters, the effects of atmospheric drag, solar radiation pressure, gravity perturbations, and third-body effects on the satellite's trajectory over time.

1 Introduction

The simulation aims to provide insights into the satellite's behavior under various perturbative forces. Understanding these dynamics is crucial for mission planning, satellite design, and operational strategy development.

2 Orbital Parameters

The initial orbit of the satellite is defined by the following classical orbital elements:

- Semi-major axis, a, defines the size of the orbit. We will choose a value of 180 km for the satellite's altitude. This means our semi-major axis $a = 6550$ km.
- Eccentricity, e, indicates the shape of the orbit, with $e = 0$ representing a circular orbit. We choose $e = 0$ for this simulation.
- Inclination, i, the orbit's tilt with respect to the Earth's equatorial plane. The inclination of our satellite's orbit with respect to the Earth's equatorial plane is specified as:
$$i = 0^\circ \tag{1}$$
and shows the satellite's orbit runs along the equator.
- Right Ascension of the Ascending Node (RAAN), Ω, which determines the orientation of the orbit in the equatorial plane.
- Argument of Perigee, ω, specifies the orientation of the elliptical orbit in the orbital plane.

- True Anomaly, ν, indicates the satellite's position in its orbit at a specific time.

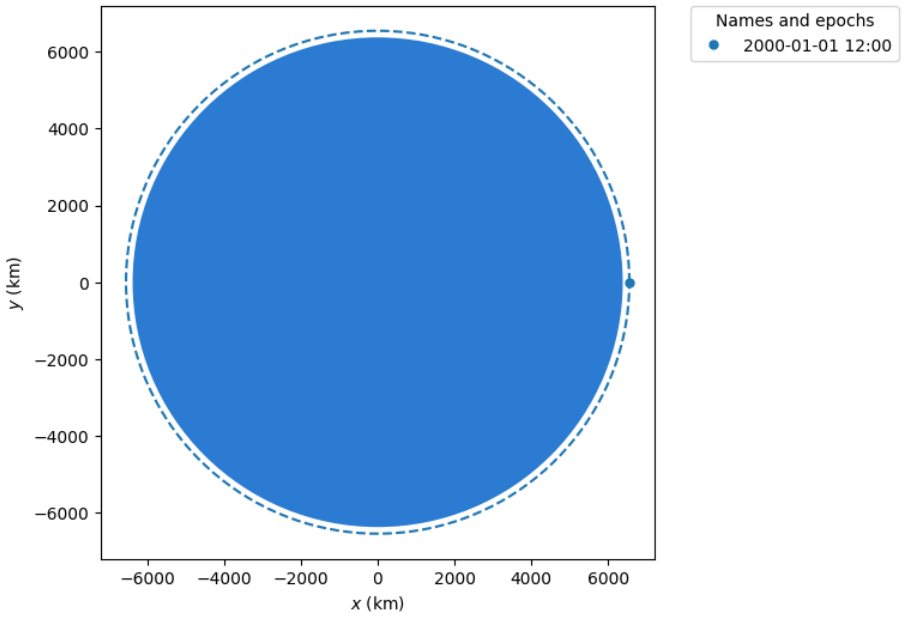
3 Simulation Without Perturbations Results

```python
from poliastro.bodies import Earth
from poliastro.twobody import Orbit
from poliastro.plotting import StaticOrbitPlotter
from astropy import units as u

# LEO orbit parameters
a = (6371 + 180) * u.km  # Earth radius + 400 km altitude
ecc = 0 * u.one  # circular orbit
inc = 0 * u.deg  # inclination, for example purposes
raan = 0 * u.deg  # arbitrary for this example
argp = 0 * u.deg  # not significant for circular orbits
nu = 0 * u.deg  # arbitrary for circular orbits

# Creating the orbit
leo_orbit = Orbit.from_classical(Earth, a, ecc, inc, raan, argp, nu
    )

# Plotting the orbit
plotter = StaticOrbitPlotter()
plotter.plot(leo_orbit)
```



As we can see in the above simulation, an unperturbed orbit will be perfectly circular in our case.

4 Atmospheric Drag Perturbation

4.1 Atmospheric Drag

Atmospheric drag reduces the satellite's velocity and causes orbital decay. It is modeled by the equation:

$$F_{\text{drag}} = -\frac{1}{2} C_d A \rho v^2 \tag{2}$$

where C_d is the drag coefficient, A is the cross-sectional area, ρ is the atmospheric density, and v is the satellite's velocity.

4.1.1 Cross-Sectional Area

The cross-sectional area, A, of a CubeSat is crucial for calculating atmospheric drag. For a standard 1U CubeSat with dimensions of 10 cm $\times$ 10 cm $\times$ 10 cm, the cross-sectional area facing the direction of motion is given by:

$$A = 10\,\text{cm} \times 10\,\text{cm} = 100\,\text{cm}^2 = 0.01\,\text{m}^2. \tag{3}$$

4.1.2 Atmospheric Drag Coefficient

The atmospheric drag coefficient, C_d, varies based on the satellite's shape and orientation. For CubeSats in Low Earth Orbit (LEO), C_d typically ranges from 2.0 to 2.8. A possible value is:

$$C_d = 2.6. \tag{4}$$

for a satellite at such a low altitude.

4.1.3 Atmospheric Density

Atmospheric density, ρ, in LEO is subject to significant variation with altitude and solar activity. It can range from approximately $1 \times 10^{-12}\,\text{kg/m}^3$ to $5 \times 10^{-9}\,\text{kg/m}^3$ or higher. For the purpose of preliminary calculations, a representative value at a specific LEO altitude might be:

$$\rho = 5.46 \times 10^{-10}\,\text{kg/m}^3. \tag{5}$$

4.1.4 Drag Force Calculation

The altitude of a satellite in Low Earth Orbit (LEO) typically ranges from 160 km to 2,000 km above the Earth's surface. The orbital velocity required for a satellite to maintain a stable orbit depends on its altitude. In our case, we chose a satellite that is 180 km above the Earth's surface.

The orbital velocity (v) of a satellite is given by the formula derived from Newton's law of universal gravitation and circular motion:

$$v = \sqrt{\frac{GM}{r}} \tag{6}$$

Substituting the given values:

$$F_d = \frac{1}{2} \times 2.6 \times 0.01 \times 5.46 \times 10^{-10} \times 7800^2 = 0.000432\,\text{N} \tag{7}$$

Therefore, the calculated drag force is 0.000432 Newtons.

4.2 Runge-Kutta Solution for Atmospheric Drag and Gravitational Forces

4.2.1 Solution to Differential Equations for Spacecraft Motion under Gravitational and Drag Forces

The motion of a spacecraft in orbit is governed by the fundamental laws of celestial mechanics, primarily Newton's second law of motion, which, when applied to orbital dynamics, leads to the differential equations governing the spacecraft's trajectory. The presence of non-conservative forces such as atmospheric drag complicates the solution of these equations, requiring numerical methods for accurate predictions.

The equation of motion for a spacecraft under the influence of Earth's gravity and atmospheric drag can be expressed as:

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{a}_{\text{drag}}, \tag{8}$$

where $\ddot{\mathbf{r}}$ is the spacecraft's acceleration vector, μ is the standard gravitational parameter of the Earth, $\mathbf{r}$ is the position vector of the spacecraft relative to the center of the Earth, and $\mathbf{a}_{\text{drag}}$ represents the acceleration due to atmospheric drag.

The drag acceleration can be modeled as:

$$\mathbf{a}_{\text{drag}} = -\frac{1}{2} C_D \frac{\rho}{m} A v_{\text{rel}}^2 \hat{\mathbf{v}}_{\text{rel}}, \tag{9}$$

where C_D is the drag coefficient, ρ is the atmospheric density, m is the mass of the spacecraft, A is the cross-sectional area orthogonal to the velocity vector, v_{rel} is the relative velocity of the spacecraft with respect to the atmosphere, and $\hat{\mathbf{v}}_{\text{rel}}$ is the unit vector in the direction of the relative velocity.

Given the nonlinear nature of these equations and the dependency of atmospheric density on altitude, which in turn varies with the spacecraft's position, an analytical solution is infeasible for most practical cases. Therefore, numerical integration methods, such as the Runge-Kutta family of solvers, are employed to approximate the spacecraft's trajectory over time. The Cowell method, implemented in various astrodynamics software packages, uses these numerical techniques to solve the coupled differential equations, accounting for both gravitational and drag forces to predict the orbit of a spacecraft.

Numerical integration proceeds by discretizing the time domain into small steps and iteratively solving the equations of motion to update the spacecraft's state vector, comprising its position and velocity. This approach enables the

modeling of complex orbital dynamics, including perturbations from atmospheric drag, with high precision over extended periods.

In summary, the accurate prediction of a spacecraft's orbit under the influence of gravitational and atmospheric drag forces requires the numerical solution of nonlinear differential equations. The use of advanced numerical methods and computational tools is essential for addressing the challenges posed by orbital dynamics in the presence of non-conservative forces.

4.2.2 Simulation

```python
from poliastro.bodies import Earth
from poliastro.twobody import Orbit
from poliastro.plotting import StaticOrbitPlotter
from poliastro.twobody.propagation import cowell
from astropy import units as u
import numpy as np

# Define constants for atmospheric drag calculation
Cd = 2.6  # Drag coefficient, typical for CubeSats
A = 0.01 * u.m**2  # Cross-sectional area, example for a 1U CubeSat
m = 5 * u.kg  # Mass of the CubeSat, example value

# Define the atmospheric density model (exponential model for simplicity)
def atmospheric_density(alt):
    rho0 = 1.225 * u.kg / u.m**3  # Sea level density
    H = 8500 * u.m  # Scale height
    return rho0 * np.exp(-alt / H)

# Define the drag acceleration function
def drag_acceleration(t0, state, k):
    v = state[3:]  # Extract the velocity vector
    alt = ((np.linalg.norm(state[:3]) * u.m) - Earth.R).to(u.km)
    rho = atmospheric_density(alt)  # Atmospheric density
    v_mag = np.linalg.norm(v) * (u.km / u.s)  # Velocity magnitude
    a_drag = -0.5 * Cd * A * rho * v_mag**2 / m * (v / v_mag)
    return a_drag.to(u.km / u.s**2).value

# LEO orbit parameters
a = (6371 + 180) * u.km  # Earth radius + 180 km altitude
ecc = 0 * u.one  # Circular orbit
inc = 0 * u.deg  # Inclination, equatorial orbit
raan = 0 * u.deg  # Right Ascension of the Ascending Node
argp = 0 * u.deg  # Argument of Perigee
nu = 0 * u.deg  # True Anomaly

# Creating the orbit
leo_orbit = Orbit.from_classical(Earth, a, ecc, inc, raan, argp, nu)

# Propagate the orbit considering atmospheric drag
times = np.linspace(0, 86400, num=1000) * u.s  # Propagate for one day
rr, vv = cowell(Earth.k, leo_orbit.r, leo_orbit.v, times, ad=drag_acceleration)
```

```
42
43 # Plotting the orbit
44 plotter = StaticOrbitPlotter()
45 plotter.plot(leo_orbit, label="Initial orbit")
```

4.3 Gravity Perturbations

The non-uniform Earth's gravitational field affects the satellite's orbit, especially in terms of precession and nodal regression.

4.4 Solar Radiation Pressure

4.4.1 Calculating Solar Radiation Pressure

The force exerted by SRP on a satellite is determined by the equation:

$$F_{SRP} = \frac{P_{SR} \cdot A \cdot \cos(\theta)}{c} \tag{10}$$

where:

- P_{SR} is the solar radiation pressure at Earth's distance from the Sun, approximately $4.56 \times 10^{-6}\, N/m^2$,
- A is the satellite's effective cross-sectional area exposed to sunlight,
- θ is the angle between the sunlight direction and the normal to the satellite's surface,
- c is the speed of light in vacuum.

4.4.2 Calculating the Force of Solar Radiation Pressure on the CubeSat

Given the solar radiation pressure $P_{SR} = 4.54 \times 10^{-6}\,\mathrm{N/m^2}$ and the CubeSat's effective cross-sectional area $A = 0.01\,\mathrm{m^2}$ exposed directly to sunlight ($\theta = 0°$), the force exerted by solar radiation pressure on the CubeSat can be calculated as:

$$F_{SRP} = P_{SR} \cdot A = 4.54 \times 10^{-6}\,\mathrm{N/m^2} \cdot 0.01\,\mathrm{m^2} = 4.54 \times 10^{-8}\,\mathrm{N}. \tag{11}$$

This force, while small, is non-negligible for precision orbital maneuvers and long-term orbit stability analysis, highlighting the importance of accounting for solar radiation pressure in CubeSat mission planning and design.

This force will act in a direction radially away from the sun. May need to use some vector analysis to model this properly in a simulation.

4.4.3 Significance and Implementation

The impact of SRP on a satellite's orbit is influenced by its area-to-mass ratio (A/m) and orientation relative to the Sun. Incorporating SRP effects into satellite simulation models requires consideration of the satellite's geometry, material properties, and operational orbit, including periods of Earth's shadow which reduce SRP. For a CubeSat, the area-to-mass ratio is high so SRP effects cannot be ignored.

4.5 Third-Body Effects

The gravitational influence from the Moon and Sun alters the satellite's trajectory, which can be significant over time.

5 Simulation With Drag Force and Gravity Perturbation Results

6 Simulation With Drag Force, Gravity Perturbation, and Solar Radiation Pressure Results

7 Conclusion