Development of detumbling prototype for a 1U CubeSat

1. Introduction

Add intro here

2. Cubesat simulation

Before developing flight ready CubeSat, it is necessary to simulate the CubeSat in a virtual environment. Initially we had to develop an open source simulation script in Matlab for determining the global position, angular dynamics, detumbling effects etc which is catered to meet our requirements and specifications. The script was created based on "Space flight Mechanics" written by Dr. Carlos Jos´e Montalvo, University of South Alabama as well as his Matlab scripts from his official repository. The entire code is available on github at `https://github.com/NEONGASHMEN/arduinodemo1U`

2.0.1 Assumptions and reference values for the CubeSat

Orbital inclination = 98°
Mass of CubSat = $2kg$
Mass moment of inertia, $M_{3\times3} = \begin{bmatrix} 0.006 & 0 & 0 \\ 0 & 0.006 & 0 \\ 0 & 0 & 0.006 \end{bmatrix}$

Maximum magnetic moment from Magnetorquer = $0.2Am^2$
Resolution of the Magnetorquer = 256 (8bit)

The magnetic field model chosen was IGRF. General gravitational model is formulated where the gravitational perturbations from celestial bodies other than earth is neglected.Atmospheric drag as well as solar pressure is neglected.

2.0.2 The control algorithm for the simulation

The first step in our workflow was to define an arbitary earth centered, non - rotating reference frame. CubeSat parameters shall be defined with respect to this reference frame. On this reference frame the Z-axis points towards the north pole and the X-axis points towards the equator through the Prime Meridian. The conversion between the

aforementioned reference frame to the real world latitude and longitudes the equations given in 3.1 and 3.2 can be used.

The next step is to derive initial conditions for the CubeSat, referenced as initial state. Initial state is an array constituting of 13 elements.

$$S_i = [\ X_i \quad Y_i \quad Z_i \quad V_{xi} \quad V_{yi} \quad V_{zi} \quad q_{0i} \quad q_{1i} \quad q_{2i} \quad q_{3i} \quad w_{xi} \quad w_{yi} \quad w_{zi}\ ]$$

X_i, Y_i, Z_i are position coordinates given by (500,0,0) for 500km altitude at prime meridian

V_{xi}, V_{yi}, V_{zi} are velocity components with respect to body frame given by 3.3

$q_{0i}, q_{1i}, q_{2i}, q_{3i}$ are initial quarternions of CubeSat with respect to its body frame, initially taken as zero.

w_{xi}, w_{yi}, w_{zi} are initial angular velocities of CubeSat, initially taken as 10deg $/s$.

Runge Kutta 4 method of iteration 3.5 is used to find out consecutive states during the satellte's orbit. A partial differential eqution which yields the state parameters are fed to the RK-4 algorithm and iterated for consecutive timesteps to obtain the state parameters at consecutive timesteps. The PDE is of the form;

$$\frac{d[S]}{dt} = f([S], t) \quad \& \quad [S]_{t=0} = [S]_0$$

and $f()$ is given by,

$$\left(\frac{dX}{dt}, \frac{dY}{dt}, \frac{dZ}{dt}\right)_{t_i} = V_{t_{i-1}}$$

$$\left(\frac{d^2X}{dt^2}, \frac{d^2Y}{dt^2}, \frac{d^2Z}{dt^2}\right)_{t_i} = -\left(\frac{GM}{r^2}\right)\hat{r}$$

$$\left(\frac{dQ_0}{dt}, \frac{dQ_1}{dt}, \frac{dQ_2}{dt}, \frac{dQ_3}{dt}\right)_{t_i} = \omega_{t_{i-1}}$$

(Obtained from Euler angles to quarternions conversion 3.6)

$$\left(\frac{d^2\omega}{dt^2}\right) = I^{-1}(T_m)$$

(From rotational inertial equation 3.4)

Here the torque from magnetorquers T_m is computed from B-dot control algorithm.The algorithm takes in the current angular velocity of the CubeSat and the magnetic field around the satellite to determine the required magnetic moment that is to be produced by the Magnetorquer, inorder to achieve efficient detumbling. The B-dot algorithm states that the magnetic moment that is to be produced should be

mututally perpendcular to the angular velocity vector ($\vec{\omega}$) and the Earth's magnetic field ($\vec{B}$). To emulate real world noissy sensors, a random noise generation is also hard coded in the script. This gives magnetic field (magnetometer sensor) with an error of few microteslas as well as angular velocity (gyro-error) with an error of few milli radians. This gave us the opportunity to implement a filtering algorithm which minimises the error. A modified Kalman filter is employed 3.7 - combining the current sensor data and previous sensor value, thereby smoothening any steep variations in the measured value. This corrected data is fed to the B-dot algorithm.

$$\vec{\mu} = k.\left(\vec{B} \times \vec{\omega}\right)$$

Here the control variable k can be adjusted as per the mission's requirements to fine tune the rate of detumbling. In our case k is adjusted to generate magnetic moments between -0.2 to $0.2Am^2$. In the script, the IGRF model is used to obtain the magnetic field at current location. The torque produced, in the magnetometer is given by,

$$\vec{T_m} = \vec{\mu} \times \vec{B}$$

Each consecutive states are stored in each iteration of timestep. For post processing, the numerical values are exported as excel and for visualisation, plotted in a graph.

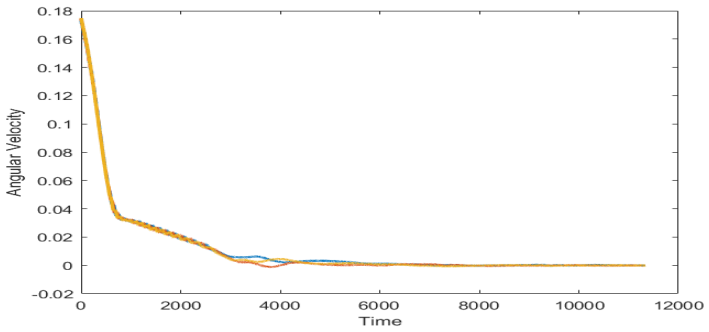
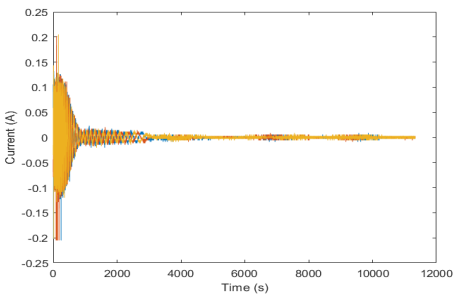
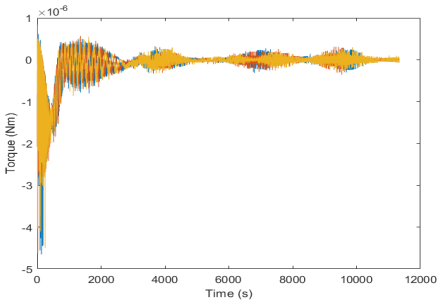


Figure 2.1: Decrease in Cubesat's angular velocity



(a) Current in magnetorquer coils



(b) Torque generated

From these results, the duration for detumbling from $\vec{\omega} = 10\hat{i} + 10\hat{j} + 10\hat{k}$ (in degrees) to $\vec{\omega} = 0.01\hat{i} + 0.013\hat{j} + 0.01\hat{k}$ was found to be 106.7667 mins (1.132 orbits) 2.1. Plot between Angular velocity of the CubeSat, Current in each magnetorquers and Torque produced from each torquers V/S time in seconds is given above 2.2a2.2b.

3. Equations

$$\left.\begin{array}{l}\rho = \sqrt{x^2 + y^2 + z^2} \\ \theta_E = \cos^{-1}\left(\frac{z}{\rho}\right) \\ \psi_E = \tan^{-1}\left(\frac{y}{x}\right)\end{array}\right\} \text{Cartesian to spherical coordinate conversion} \tag{3.1}$$

$$\left.\begin{array}{l}\lambda_{LAT} = 90 - \theta_E \frac{180}{\pi} \\ \lambda_{LON} = \psi_E \frac{180}{\pi} \\ h = \rho - R_E\end{array}\right\} \text{Latitude, Longitude and height} \tag{3.2}$$

$$v = \sqrt{\mu\left(\frac{2}{r} - \frac{1}{a}\right)} \quad \text{Vis-viva equation, NB: for the simulation } a \approx r \tag{3.3}$$

$$\dot{\vec{\omega}} = I^{-1}\left(T_{Propulsion} + T_{Magnetorquer} + T_{Reactionwheel} - \dot{\vec{\omega}} \times \vec{H} - \dot{I}\vec{\omega}\right) \tag{3.4}$$

$$\left.\begin{array}{l}\text{If, } \frac{dy}{dx} = f(t_n, y_n) \text{ and } y(t_n) = y_n \\ k_1 = f(t_n, y_n) \\ k_2 = f\left(t_n + \frac{h}{2}, y_n + h\frac{k_1}{2}\right) \\ k_3 = f\left(t_n + \frac{h}{2}, y_n + h\frac{k_2}{2}\right) \\ k_4 = f(t_n + h, y_n + hk_3) \\ k = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ y(t_n + h) = y_n + hk\end{array}\right\} \text{RK4} \tag{3.5}$$

$$\left.\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) - \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) - \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) \end{bmatrix}\right\} \text{Euler to Quarternion conversion} \tag{3.6}$$

$$X_{filtered} = S.X_{current} + (1-S)\,X_{previous} \quad , 0 < S < 1, \text{S being the sensor reliabilty} \tag{3.7}$$