Analysis of Motor Misalignment, Jet-Damping and Mass Variation on a Spinning Thrusting CubeSat

Dhruv Jain[*]
Purdue University, West Lafayette, Indiana, 47906

With the advent of CubeSats [1], it is essential to use better dynamical models to incorporate the loss of propellant, especially to understand the effect of spinning-up maneuvers. This paper discusses the numerical simulation setup and results for a spinning-up axis-symmetric CubeSat and its comparison with analytical solutions [4]. In the analysis it is assumed that the torques are small but due to the jet-damping and use of propellant during the maneuver, the Principal Moments of Inertia and torques are modelled as a function of the time varying mass and center of mass of a satellite. In addition, a more realistic scenario is considered where the thruster is misaligned and there is a motor offset.

I. Nomenclature

M_x	=	Torque along body x-axis
f_z	=	Force along body z-axis
α	=	Motor misalignment angle
r	=	Radius of cylindrical propellant tank
h	=	Distance from Nozzle to the CubeSat center of mass
m_b	=	Mass of CubeSat without propellant
m_{t0}	=	Initial mass of propellant
$\dot{m}_t$	=	Propellant mass change rate

II. Introduction

In the 21st Century through the advances in electronics and the need for lower temporal and higher spatial Earth Observation and weather data, there has been widespread acceptance and increased usage of CubeSats [1]. This is also due to their lower cost of development and faster production compared to the first and second generation of satellites in the 20th Century. CubeSats offer unique capabilities that were not possible through bulkier satellites such as the formation of LEO satellite constellation for Earth Observation and Communication. However, their smaller sizes have posed unique challenges and challenged the assumptions

[*]Graduate Student, School of Aeronautics and Astronautics, West Lafayette, Indiana, USA-47906

used to calculate the analytical solutions [2].

CubeSats usually have a micro-propellant systems on board and during a spin-up maneuver, the usage of propellant causes jet-damping and change in mass of the satellite. This in turn changes the center of mass, which leads to variation in Principal Moments of Inertia (PMOI). Previously, in most of the analysis the change in mass of satellite due to propellant usage was ignored [3]. However, it is critical to include this component for CubeSats as the mass of the CubeSat without propellant (m_b) and mass of propellant (m_t) are comparable, thus the change in total mass due to change in m_t cannot be ignored. In addition, a more realistically scenario is considered where the thruster is misaligned.

Note: (x,y,z) correspond to Body Frame axes and (X, Y, Z) correspond to Inertial Frame axes.

For analysis in this paper, the following assumptions are made:

1) Axisymmetric satellite, $I_x = I_y$
2) At T=0, the body fixed angles and quasi-velocities except for w_z are all 0
3) Thruster is misaligned by angle α
4) Thruster leads to torque in only along the body x-axis (M_x)
5) Mass of propellant decreases linearly, thus PMOI and total mass decrease linearly
6) Propellant is assumed to be a solid propellant, to avoid any sloshing effects
7) Force from the thruster is constant
8) External torques are 0

In order to better understand the dynamics of a CubeSat during a spin-up maneuver, an analytical solution is presented using the assumptions laid out and numerical integration is used to find the numerically exact solution. Thereafter, the two methods are compared and insights gained from the two methods are presented.

III. Analysis

The differential equations that define the attitude of a satellite can be described by the Euler's Equations of motion [5],

$$ {}^{e}\dot{\bar{H}}^{c} = \bar{M}^{c} \tag{1}$$

The typical approach taken by Longuski et al. [2,3,8], equation (1) works well when the change of mass of the satellite due to propellant usage is negligible. However, as discussed, there is a need to incorporate the time derivative terms of PMOI and the effect of change in center of mass. This is well Incorporated in the formulation by Martin [7] in equation (3.13) and by Liang et al. [4] in equation (1). Thus, the Euler's Equations of Motion for the motion under consideration:

$$\dot{\omega}_x(t) = M_x(t)/I_x(t) - [(I_z(t) - I_y(t))/I_x(t)]\omega_y(t)\omega_x(t) - [\dot{I}_x(t) - \dot{m}(h(t)^2 + d^2)]\omega_x(t)/I_x(t) \tag{2}$$

$$\dot{\omega}_y(t) = M_y(t)/I_y(t) - [(I_x(t) - I_z(t))/I_y(t)]\omega_z(t)\omega_x(t) - [\dot{I}_y(t) - \dot{m}h(t)^2]\omega_y(t)/I_y(t) \tag{3}$$

$$\dot{\omega}_z(t) = M_z(t)/I_z(t) - [(I_y(t) - I_x(t))/I_z(t)]\omega_x(t)\omega_y(t) - [\dot{I}_z(t) - \dot{m}d^2]\omega_z(t)/I_z(t) \tag{4}$$

where, h is the distance from satellite's center of mass to nozzle throat. We know that this is a function of time as the center of mass changes as the propellant is used up during a maneuver. Thus, unlike the assumption made by Martin [7], the formulation is the same as the one considered in [4] and similar to the one presented in equation (18) in [6] by Ha et al. . d is the motor offset and is constant. $\dot{m}$ is the mass change due to use of propellant and assumed constant.

As M_y and M_z are considered to be 0, so the first term in equation (3) and (4) are dropped.

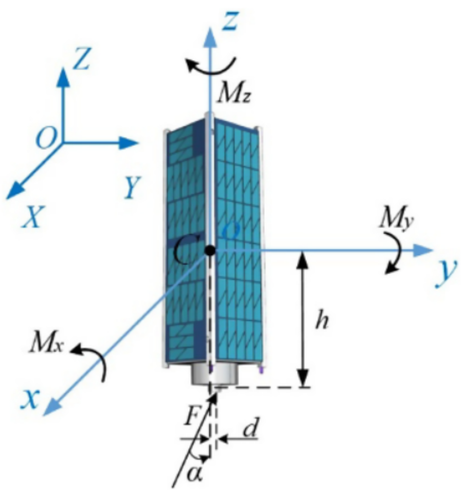


Fig. 1 CubeSat Model (From Liang et al., Ref. 4, p. 3622)

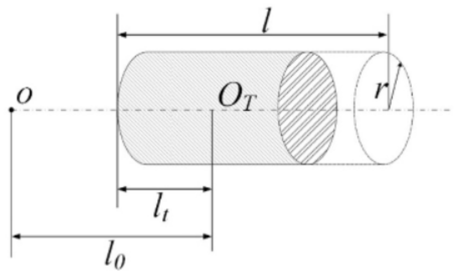


Fig. 2 Motor Parameters (From Liang et al., Ref. 4, p. 3624)

Consider the CubeSat model in Fig. 1 from Liang et al. [4] and the Motor parameters shown in Fig. 2 from Liang et al. [4]. These will come handy in visualizing and better understanding the setup. In order to more intuitively incorporate the various parameter changes that are being considered; The CubeSat without any propellant is modelled without propellant, propellant is modelled separately and the mutual center of mass's effect and individual terms are added to reproduce the parameter changes on the entire setup.

Let PMOI of the CubeSat without the propellant in the body-frame be represented by I_{bx}, I_{by} and I_{bz}. It is important to note that these values remain constant during maneuver.

In Fig. 2 created by Liang et al. [4], O_T is the propellant center of mass at the initial time. We assume it to be aligned with the principal axes of CubeSat. l_0 is the distance between O_T and the origin of the body-fixed reference frame.The initial length of propellant is l and we know that it will change as propellant is used, and r is the radius of the propellant tank, which is constant. Lastly, l_t is the distance between O_T and tip of propellant in the tank. We can see that this is also dependent on the amount of propellant left.

The change in properties of the propellant, i.e. l_t and m_t can be described as:

$$l_t = l_{t0} + \dot{l}_t t \tag{5}$$

$$m_t = m_{t0} + \dot{m} t \tag{6}$$

Using this information and setup, h can be defined as:

$$h = \frac{(l_0 + l)m_b + (l - l_t)m_t}{m_b + m_t} \tag{7}$$

Using the above information, the PMOI of the propellant about its center of mass can be expressed as ([4], equation (23-24)):

$$I_{tx} = I_{ty} = m_t \left(\frac{r^2}{4} + \frac{l_t^2}{3} \right) \tag{8}$$

$$I_{tz} = m_t \frac{r^2}{2} \tag{9}$$

As in the setup we are considering the effect on PMOI due to decrease in m_t, so the time derivative of the I_t's are given as:

$$\dot{I}_{tx} = \dot{I}_{ty} = \dot{m}_t \left(\frac{r^2}{4} + \frac{l_t^2}{3} \right) + \frac{2}{3} m_t l_t \dot{l}_t \tag{10}$$

$$\dot{I}_{tz} = \dot{m} m_t \frac{r^2}{2} \tag{11}$$

An important step is to use parallel axis theorem to transform the I_t's from the propellant center of mass to the center of the CubeSat. Using parallel axis theorem and adding the PMOI of the CubeSat without propellant, the PMOI of the CubeSat is given as:

$$I_x = I_{bx} + I_{tx} + m_t (l_t + l_0)^2 \tag{12}$$

$$I_y = I_{by} + I_{ty} + m_t (l_t + l_0)^2 \tag{13}$$

$$I_z = I_{bz} + I_{tz} \tag{14}$$

The corresponding time derivatives of the PMOI's are:

$$\dot{I}_x = \dot{I}_{tx} + \dot{m}_t (l_t + l_0)^2 + 2 m_t (l_t + l_0) \dot{l}_t \tag{15}$$

$$\dot{I}_y = \dot{I}_{ty} + \dot{m}_t (l_t + l_0)^2 + 2 m_t (l_t + l_0) \dot{l}_t \tag{16}$$

$$\dot{I}_z = \dot{I}_{tz} \tag{17}$$

Since, the thruster is assumed to be misaligned by angle α and due to the motor position offset, the axial torque due to the motor is given by:

$$M_x(t) = f_z (h(t) sin\alpha + d cos\alpha) \tag{18}$$

It is crucial to realize that despite f_z being considered to be constant, M_x is not constant as h is a function of time.

The attitude and the relevant can be realized by including the kinematic equations along with dynamic equations from the Euler's Equations of Motion. There are 12 forms of body Euler angles [3], arbitrarily the 3-1-2 rotation is chosen. The corresponding kinematic equations are:

$$\dot{\phi}_x = \omega_x cos\phi_y + \omega_z sin\phi_y \tag{19}$$

$$\dot{\phi}_y = \omega_y - (\omega_z cos\phi_y + \omega_x sin\phi_y) tan\phi_x \tag{20}$$

$$\dot{\phi}_z = (\omega_z cos\phi_y - \omega_x sin\phi_y) sec\phi_x \tag{21}$$

This completes the formulation for the setup, which can be numerically integrated to understand the dynamics of the spinning thrusting CubeSat.

However, it is imperative to find, if possible, analytical solutions by making further minor assumptions to understand the general behavior of the system. These solutions can then be used to provide good initial guesses and to predict downstream behavior without going through the non-trivial process of numerically integrating the PDE's.

The analytical solutions presented by Liang et al. [4] provide better analytical solutions than those presented in [3,6,7,8] as Liang et al. works with the similar assumptions as presented in this paper. Authors of [3,6,7,8] consider more set of assumptions and the analytical solutions presented in those papers can be extracted from the ones presented in [4] by making the required assumptions. Liang et al. [4] calculated and presented an even more general analytical solution than for the assumptions made.

These are the additional assumptions made to find the analytical solution:

1) $\dot{I}_i$'s are negligible => $\dot{I}_i$ terms are dropped from (2-4)
2) d^2, (motor offset) is much smaller than h^2 => d^2 terms are dropped from (2-4) Thus, the setup reduces to EOM defined in [2] with an additional jet-damping term = $-\dot{m}h^2$ in (2-3).
3) ϕ_x, ϕ_y and $\phi_y \omega_x$ are considered small => equations (19-21) reduce to:

$$\dot{\phi}_x = \omega_x + \omega_{z0}\phi_y \tag{22}$$

$$\dot{\phi}_y = \omega_y - \omega_{z0}\phi_x \tag{23}$$

$$\dot{\phi}_z = \omega_{z0} \tag{24}$$

In addition, let I_x = I_y = I and k = (I_z-I)/I.

From [4], for only M_x and $\omega_z 0$ to be nonzero:

$$\omega_x = \frac{aM}{a^2 + I^2k^2\omega_{z0}^2} + \frac{exp(-at/I)}{a^2 + I^2k^2\omega_{z0}^2}[-aM_x cos(k\omega_{z0}t) + IkM_x\omega_{z0} sin(k\omega_{z0}t)] \tag{25}$$

$$\omega_y = \frac{IkM_x\omega_{z0}}{a^2 + I^2k^2\omega_{z0}^2} + \frac{exp(-at/I)}{a^2 + I^2k^2\omega_{z0}^2}[-IkM_x\omega_{z0}cos(k\omega_{z0}t) - aM_xsin(k\omega_{z0}t)] \tag{26}$$

$$\omega_z = \omega_{z0} \tag{27}$$

$$\phi_x = [A_{0x}+A_1cos(\omega_{z0}t)+A_2sin(\omega_{z0}t)+I\omega_{z0}(A_3cos(k\omega_{z0}t)+A_4sin(k\omega_{z0}t)(cosh(at/I)-sinh(at/I))]/C \tag{28}$$

$$\phi_y = [A_{0y}-A_1sin(\omega_{z0}t)+A_2cos(\omega_{z0}t)+I\omega_{z0}(A_3sin(k\omega_{z0}t)-A_4cos(k\omega_{z0}t)(cosh(at/I)-sinh(at/I))]/C \tag{29}$$

$$\phi_z = \omega_{z0}t \tag{30}$$

where A_{0x}, A_{0y}, A_1, A_2, A_3, A_4 and C are defined and explained in [4] on page 3623.

As both the numerical and analytical setup is developed and explained, we now compare the two and get insight about the setup.

IV. Numerical Simulation

MATLAB is used to numerically integrate the 6 equations of motion. An inbuilt integrator, ode45, which is an explicit RK 45 solver is used with restol = abstol = 1e-12.

It is important to note that l_t and m_t are defined as a peicewise function, where after time, t, when m_{t0} = -$\dot{m}_t t$, the $\dot{m}_t t$ term is dropped and similarly $\dot{l}_t t$ is dropped.

Parameters used for simulation, many of the parameter values are from [4]:

Symbol	Description	Value	Unit
l	Initial length of propellant	0.045	m
l_0	Distance between O_T and 0	0.15	m
l_{t0}	Distance between O_T and tip of propellant	0.0225	m
$\dot{l}_{t0}$	Rate change of l_{t0} with time	-0.0056	m/s
m_b	Mass of CubeSat without propellant	3	kg
m_{t0}	Initial mass of propellant	0.1	kg
$\dot{m}_{t0}$	Rate change of mass of propellant with time	-0.025	kg
d	motor offset	0.001	m
α	Motor misalignment	0.25	deg
r	Radius of propellant tank	0.01	m
I_{bx}	Moment of inertia about body x-axis	0.035	kgm^2
I_{by}	Moment of inertia about body y-axis	0.035	kgm^2
I_{bz}	Moment of inertia about body z-axis	0.007	kgm^2
f_z	Body-fixed force about z-axis	30	N

Table 1 List of parameters

Initial Conditions:

Symbol	Value	Unit
ω_x	0	rad/s
ω_y	0	rad/s
ω_z	25	rad/s
ϕ_x	0	rad
ϕ_y	0	rad
ϕ_z	0	rad

Table 2 Initial Conditions

It is important to keep in mind that for Analytical solution, $I_i(t) = I_i(0)$ and $M_x(t) = M_x(0)$.

A. Check to see if Numerical Setup and Analytical Solution are correct

In order to check if the numerical setup and analytical solution are correct. A simpler case is first considered than the one described in the Analysis section. We know that making more simplifying assumptions than initially defined like $\dot{l}_{t0}$=0 and $\dot{l}_{t0}$=0, will enable us to simplify the problem to only thruster misalignment. It is easy to speculate that the numerical solutions and analytical solutions should agree and the difference of the 6 states, namely ω_x, ω_y, ω_z, ϕ_x, ϕ_y and ϕ_z should agree to high accuracy.

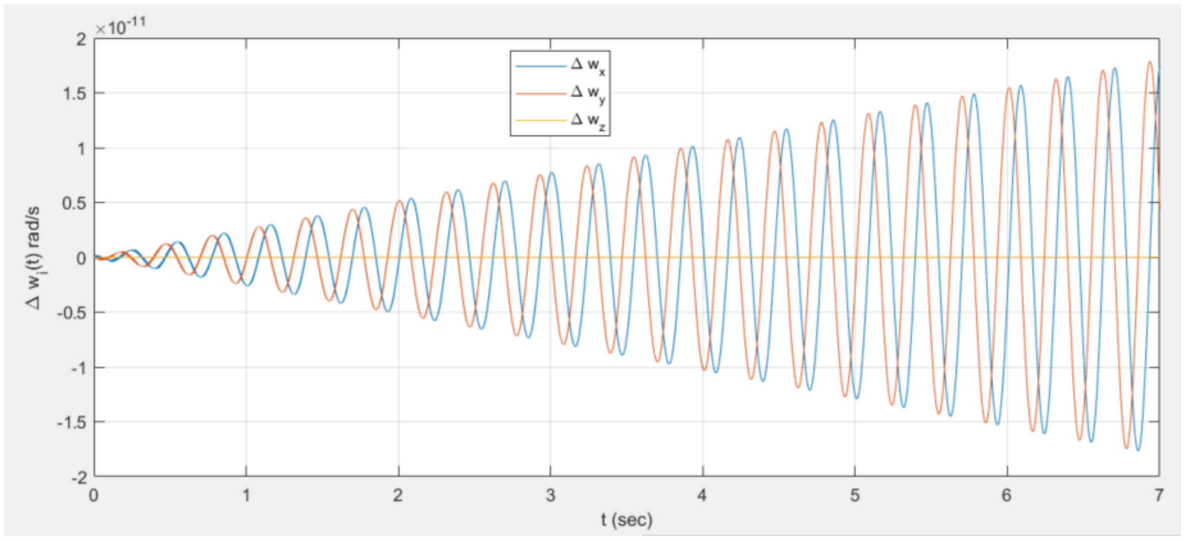


Fig. 3 $\Delta\omega_i$(t) simplified setup

Note: Axis equal in Fig. 3-7 was intentionally not used, to see the small variations if any exists.

Fig. 3 shows the difference in numerical and analytical solution for the three quasi-velocities over 7 seconds for the simplified setup considered in this subsection. It can be observed after realizing that the y-scale is ≈ 1e-11. Thus, the solution from the two method is effectively the same. The small variation exists

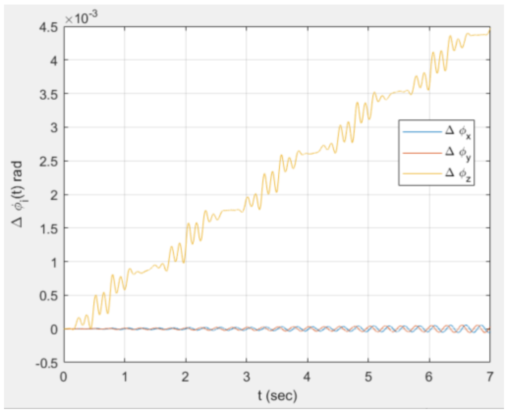


Fig. 4 $\Delta\phi_i$**(t) simplified setup**

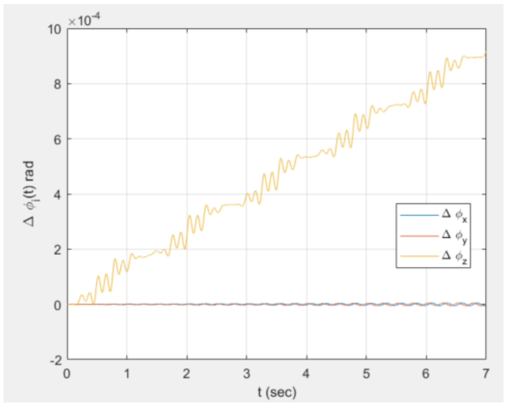


Fig. 5 $\Delta\phi_i$**(t) simplified setup + No motor Offset**

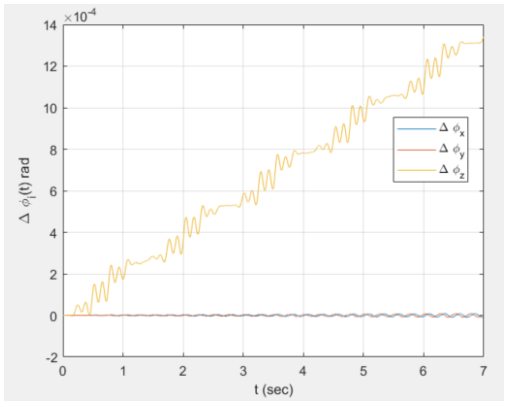


Fig. 6 $\Delta\phi_i$**(t) simplified setup + No Thruster Misalignment**

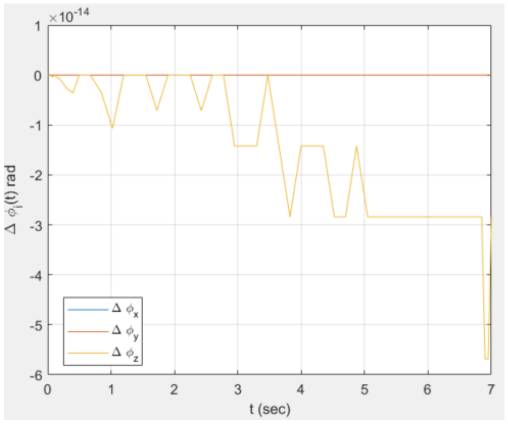


Fig. 7 $\Delta\phi_i$**(t) simplified setup + No Motor Offset + No Thruster Misalignment**

due to numerical integration error accumulation. It is important to remember that even in this simplified setup, the numerical setup considers the thruster misalignment and motor offset, whereas the the analytical does not.

Fig. 4 shows the difference in numerical and analytical solution for the three body angles over 7 seconds for the same setup as for which Fig. 3 data was computed. It is interesting to note that the difference in ϕ_z solution from the two methods is significant and even the difference for the other two angles is visible close to the end of simulation.

The three body angles were further investigated to pinpoint the source of difference in solution from the two methods. It is again important to remark that the difference in quasi-velocities from the two methods is close to 0, so it is surprising to see a significant difference in the body angles. We know the two assumptions that differentiate the setup of the two methods is that numerical setup considers the thruster misalignment and motor offset.

The effect of the two terms were isolated to find which assumption is leading to the surprising result. First, the simplified setup in this section was used but with the additional assumption that motor offset is 0. Using this, the body angles difference from the two methods was recomputed and plotted in Fig. 5. We see that the difference magnitude has dropped by one order than from the model used to compute the values in Fig. 4. Thereafter, body angles difference from the two methods was recomputed but this time using the simplified setup and the assumption that there is no thruster misalignment.The results of this setup are shown in Fig. 6. It is again surprising to see that the results are similar to Fig. 5 and the difference is significant.

At last, both the assumptions were included to the simplified setup, effectively making the same set of assumptions for numerical setup as that were made for the analytical solutions. It can be observed from Fig. 7 that the difference in the solution from the two methods is $\approx$ 1e-14 ≈ 0, which we expected.

Also, the quasi-velocities difference in solutions for the 4 setups considered in Fig. 4 - Fig. 7 were roughly the same as presented in Fig. 3.

B. Using setup defined in Analysis: Thruster Misalignment, Jet-Damping, Mass Variation

After checking the solutions from the two methods for the various setups, we revert back to the most realistic setup that we have considered and explained in detail in the Analysis section, i.e. a spinning thrusting CubeSat with thruster misalignment, jet-damping, motor offset and mass variation. Using the code written and attached in appendix, the 6 states for this setup were numerically simulated for 7 secs and the corresponding analytical solutions were computed using equations (25)-(30).

Fig. 8 and Fig. 9 are plots of the time histories of the three quasi-velocities for Numerical and Analytical solutions. We can notice in Fig. 8 that ω_x has an amplitude of 0.07 and varies between $\pm$ 0.07 rad/s. In addition, ω_y also has an amplitude of 0.07 but it varies between 0 and -0.14. In Fig. 9, we see that the analytical ω_z stays constant at 25rad/s but the numerical value linearly increases. From Fig. 8, Fig. 9 and Fig. 12, we can see that despite the difference in assumptions of the two methods, analytical solutions well approximate the numerical solutions for the first second, but thereafter due to the secular terms and other variations it is no longer a good approximate and progressively gets worse.

After looking at the ϕ_i values plotted in Fig. 10, 11 and 13, using the two method for the same setup as the Fig. 8,9 and 12, we can see a similar trend where the analytical solutions provide a decent approximation for the first second and thereafter it increasingly worsens. Nonetheless, the analytical solution sheds some light on vaguely how the evolution of the body angles and the quasi-velocities will pan out.

Furthermore, the numerical solutions were used to compute the body-fixed reference frame Angular Momentum vector. The transverse components were then plotted as shown in Fig. 14. It shows an interesting behavior as the center of the cone seems to lie on the x-axis but off the y-axis. In addition, the angular momentum vector seems to decrease in magnitude as at the end of 7 seconds the angular moment vector is closer to the center than it was at the beginning.

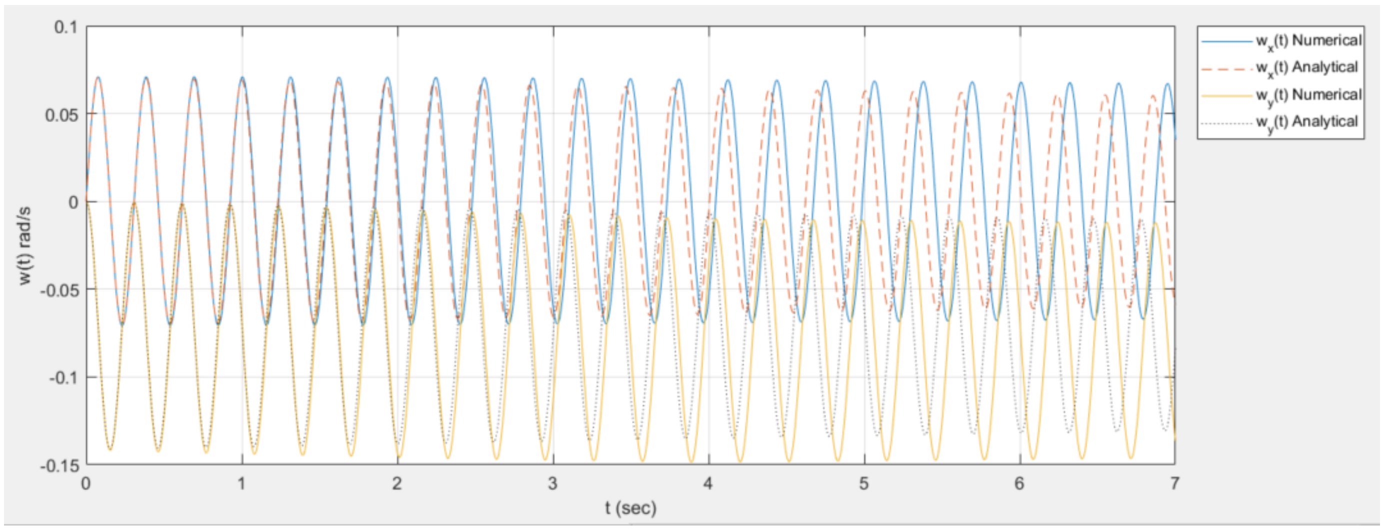


Fig. 8 ω_x**(t) and** ω_y**(t), Numerical and Analytical**

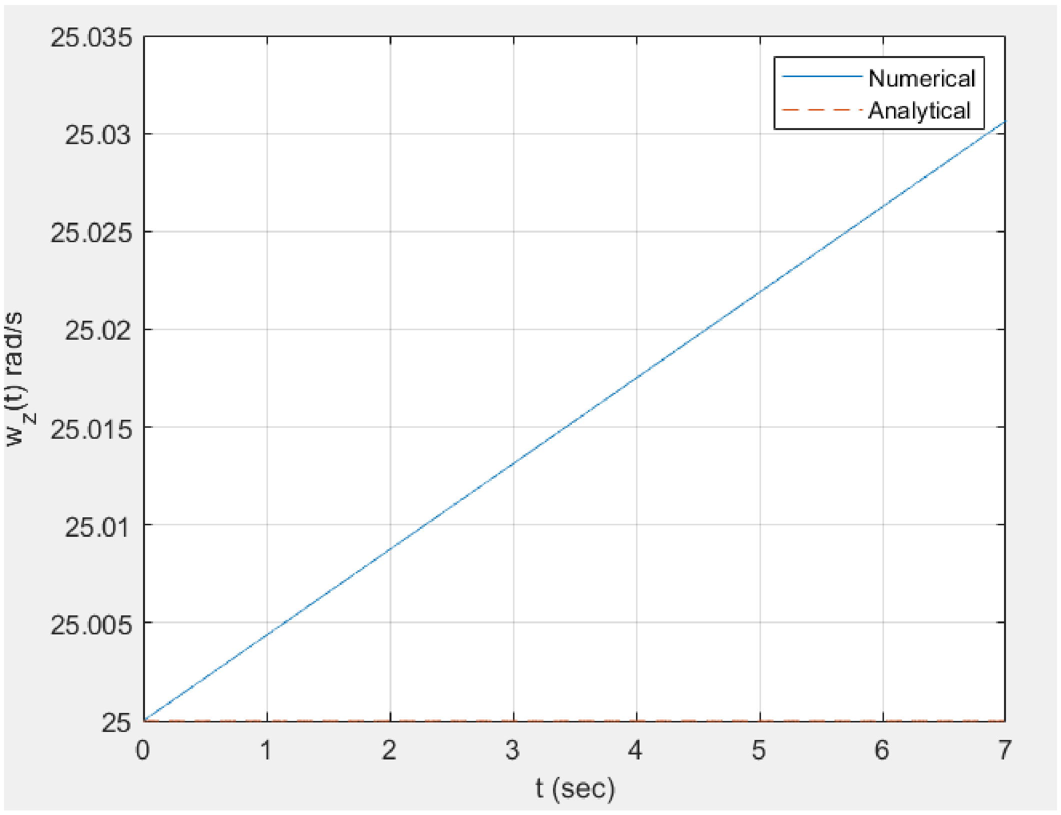


Fig. 9 ω_z**(t), Numerical and Analytical**

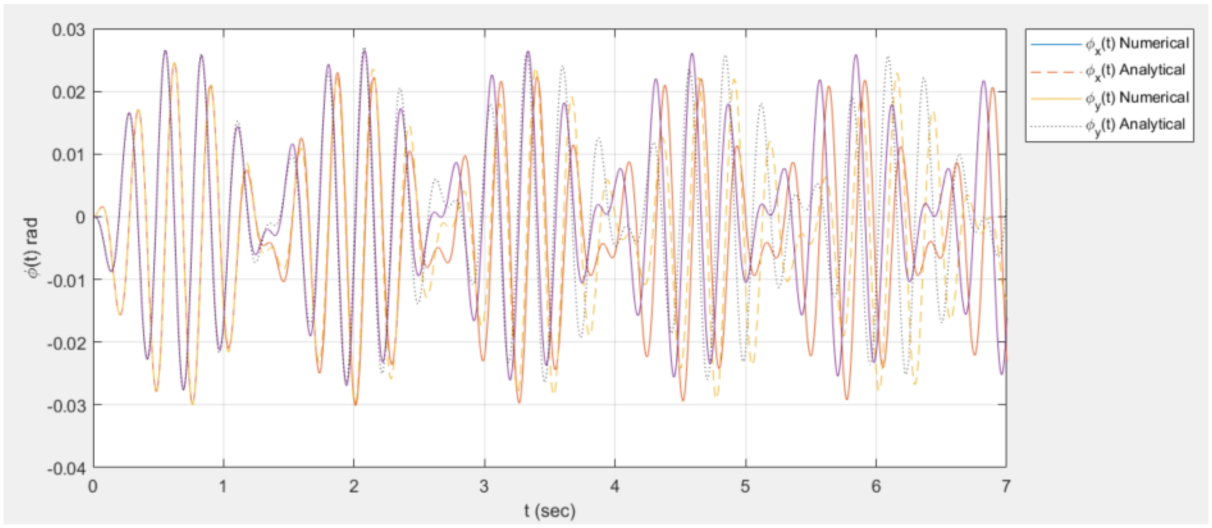


Fig. 10 ϕ_x**(t) and** ϕ_y**(t), Numerical and Analytical**

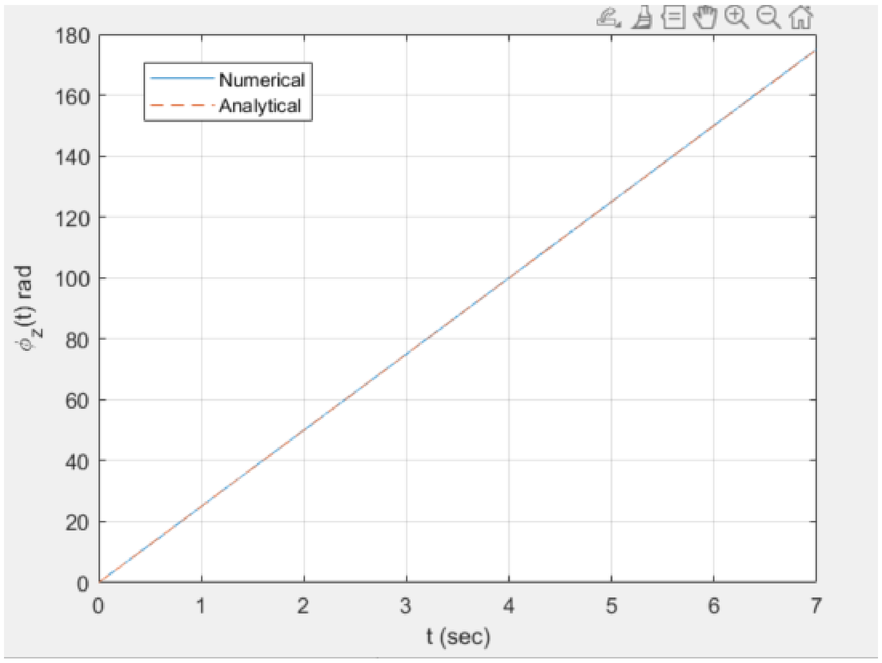


Fig. 11 ϕ_z**(t), Numerical and Analytical**

V. Conclusions and Future Work

It is critical to note that all the conclusions and analysis is only for the chosen parameter values and the analysis may be vary if any of the key parameters is changed.

1) The simplified setup, i.e. assuming no Jet-damping and mass variation in the setup explained in the Analysis section, as well as the other combination of simplified setup with other assumptions as

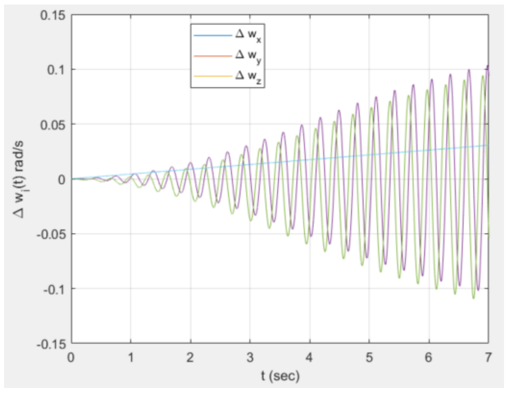


Fig. 12 $\Delta\omega_i$**(t) Numerical-Analytical**

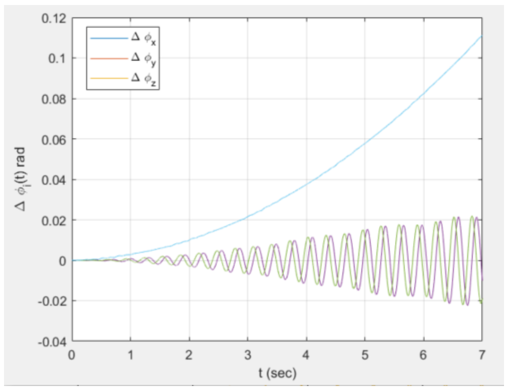


Fig. 13 $\Delta\phi_i$**(t) Numerical-Analytical**

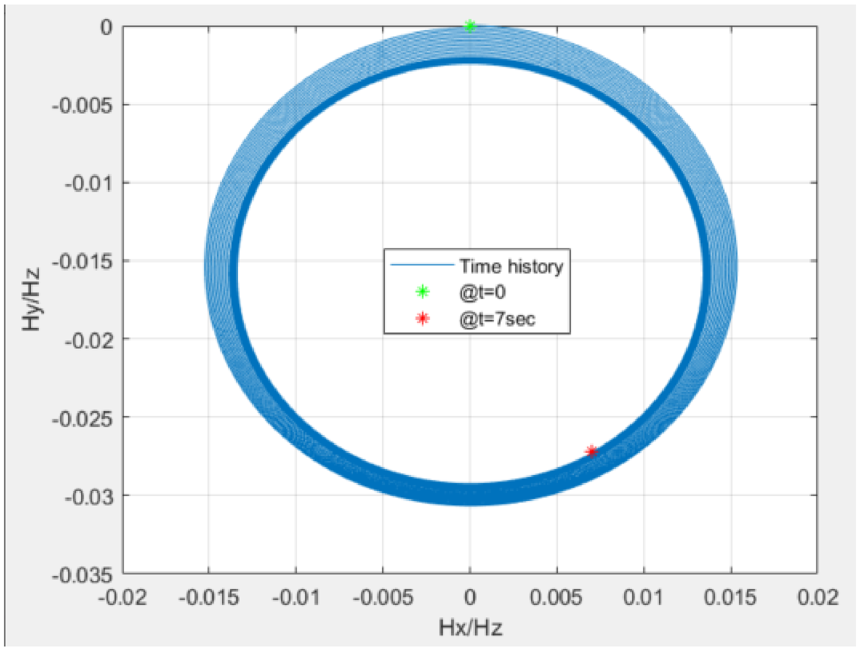


Fig. 14 $H_x(t)/H_z(t)$ **vs** $H_y(t)/H_z(t)$**, Numerical**

discussed for Fig. 5-7, lead to nearly 0 difference in the ω_i's between the numerical and analytical solution for all the setups.

2) For the above setups but for ϕ_i's, it was noticed that all setups except for the one where there is no Jet-damping, mass variation, thruster misalignment and motor offset, the difference the the three body angles is 0.
3) It also sheds light on the fact that as the setups gets increasingly realistic the numerical ϕ values deviate much more than the analytical values, compared to the ω's. Thus, ϕ's are more sensitive to the assumptions than ω states.
4) In order to find a concrete answer behind exactly which particular assumption of the analytical solution is the poorest assumption requires further analysis. However, I speculate that the small angle

approximation is one of the primary approximations that leads to poor analytical solution.

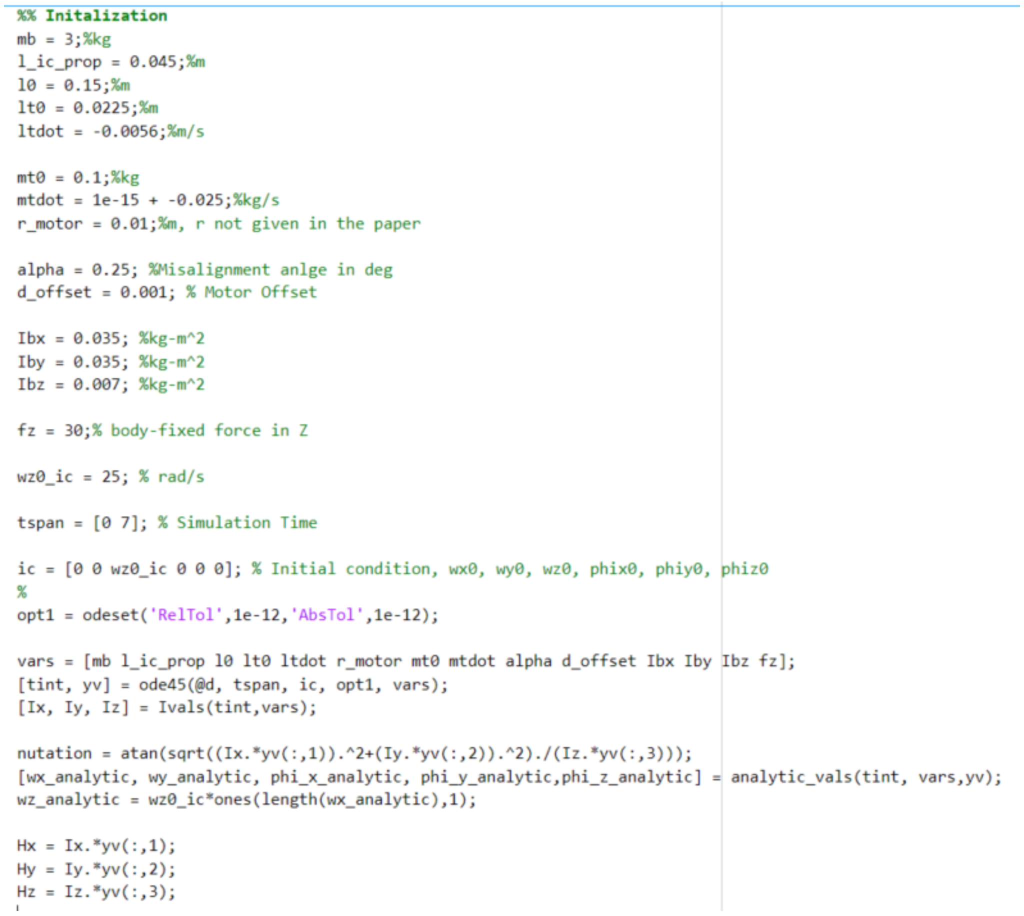
5) We can conclude from the analysis of the most realistic case, i.e. with Jet-damping, motor offset, mass variation and thrust misalignment, despite analytical solution having multiple assumptions it provides a good approximation for the first second and thereafter the analytical solution is increasingly poorer compared to the numerical solution.
6) Despite the various assumptions that were made to calculate the analytical solution, it is interesting to see that the analytical solution can give a good idea about the general shape of the evolution of each of the transverse related shapes.
7) Furthermore, the angular momentum vector is centered about a point on the x-axis that is of the y-axis but its magnitude decreases with time. Thus, this setup may be useful for angular momentum dumping.

References

1) Shiroma, W.A., Martin, L.K., Akagi, J.M. et al. CubeSats: A bright future for nanosatellites. cent.eur.j.eng 1, 9–15 (2011). https://doi.org/10.2478/s13531-011-0007
2) Ayoubi, M. A., and Longuski, J. M. (January 11, 2008). "Analytical Solutions for Translational Motion of Spinning-Up Rigid Bodies Subject to Constant Body-Fixed Forces and Moments." ASME. J. Appl. Mech. January 2008; 75(1): 011004. https://doi.org/10.1115/1.2755110
3) Longuski, James, Kia, T. and Breckenridge, W.. (1990). Annihilation of angular momentum bias during thrusting and spinning-up maneuvers. 37.
4) Zhenhua Liang, Wenhe Liao and Xiang Zhang, Velocity pointing error analysis for symmetric spinning thrusting Cubesat, Advances in Space Research, Volume 63, Issue 11, 2019, Pages 3621-3631, ISSN 0273-1177, https://doi.org/10.1016/j.asr.2019.02.006.
5) D. T. Greenwood. Principles of Dynamics, pages 304–306, 390–391. Prentice Hall, Upper Saddle River, NJ, second edition, 1988.
6) Jozef C. van der Ha, Frank L. Janssens, Jet-Damping and Misalignment Effects During Solid-Rocket-Motor Burn, 2005, Journal of Guidance, Control, and Dynamics, 412-420, 28, 3, https://arc.aiaa.org/doi/abs/10.2514/1.38
7) Martin, Kaela Mae, "Maneuver analysis for spinning thrusting spacecraft and spinning tethered space-craft" (2015). Open Access Dissertations. 512. https://docs.lib.purdue.edu/open_access_dissertations/512
8) J. M. Longuski and T. Kia. A parametric study of the behavior of the angular momentum vector during spin rate changes of rigid body spacecraft. Journal of Guidance, Control, and Dynamics, 7(3):295–300, 19874

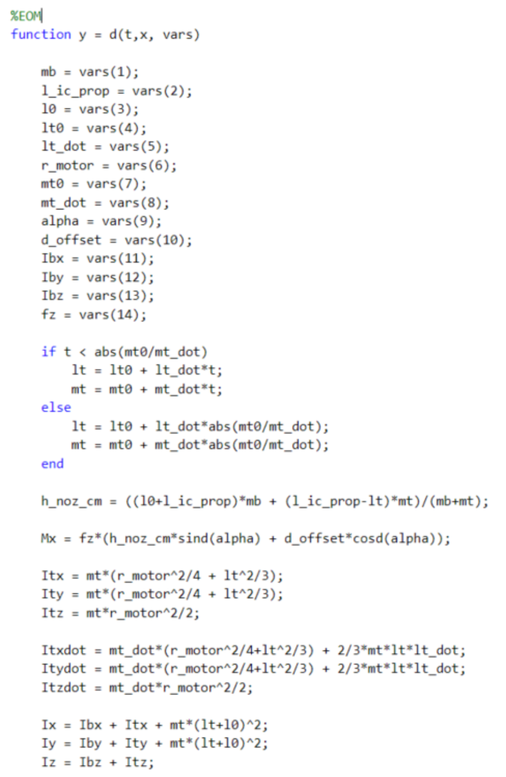
Appendix

Code Snippets:



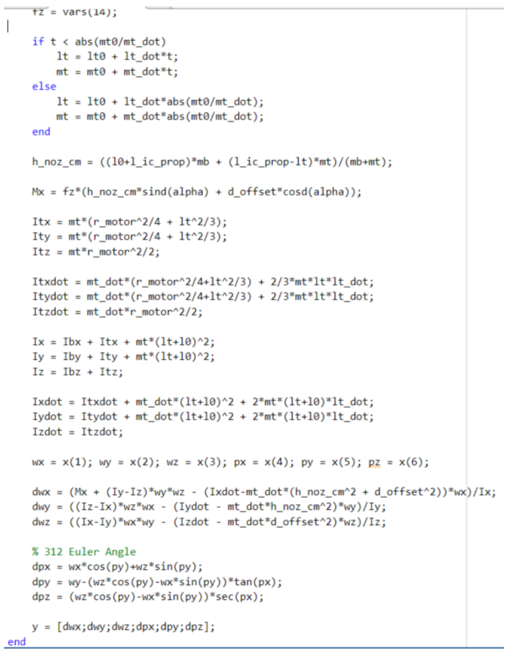
```
%% Initalization
mb = 3;%kg
l_ic_prop = 0.045;%m
l0 = 0.15;%m
lt0 = 0.0225;%m
ltdot = -0.0056;%m/s

mt0 = 0.1;%kg
mtdot = 1e-15 + -0.025;%kg/s
r_motor = 0.01;%m, r not given in the paper

alpha = 0.25; %Misalignment anlge in deg
d_offset = 0.001; % Motor Offset

Ibx = 0.035; %kg-m^2
Iby = 0.035; %kg-m^2
Ibz = 0.007; %kg-m^2

fz = 30;% body-fixed force in Z

wz0_ic = 25; % rad/s

tspan = [0 7]; % Simulation Time

ic = [0 0 wz0_ic 0 0 0]; % Initial condition, wx0, wy0, wz0, phix0, phiy0, phiz0
%
opt1 = odeset('RelTol',1e-12,'AbsTol',1e-12);

vars = [mb l_ic_prop l0 lt0 ltdot r_motor mt0 mtdot alpha d_offset Ibx Iby Ibz fz];
[tint, yv] = ode45(@d, tspan, ic, opt1, vars);
[Ix, Iy, Iz] = Ivals(tint,vars);

nutation = atan(sqrt((Ix.*yv(:,1)).^2+(Iy.*yv(:,2)).^2)./(Iz.*yv(:,3)));
[wx_analytic, wy_analytic, phi_x_analytic, phi_y_analytic,phi_z_analytic] = analytic_vals(tint, vars,yv);
wz_analytic = wz0_ic*ones(length(wx_analytic),1);

Hx = Ix.*yv(:,1);
Hy = Iy.*yv(:,2);
Hz = Iz.*yv(:,3);
```

Fig. 15 Initialization and Function Calls



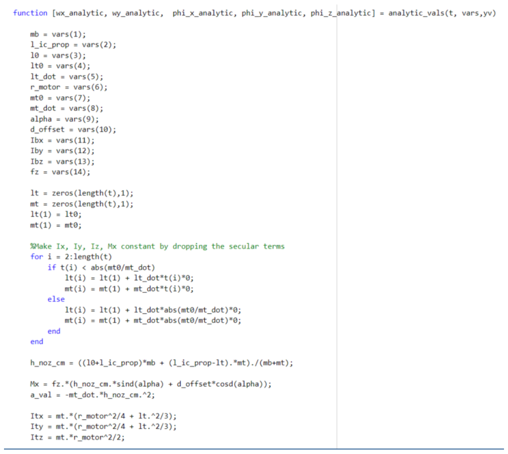
```
%EOM
function y = d(t,x, vars)

    mb = vars(1);
    l_ic_prop = vars(2);
    l0 = vars(3);
    lt0 = vars(4);
    lt_dot = vars(5);
    r_motor = vars(6);
    mt0 = vars(7);
    mt_dot = vars(8);
    alpha = vars(9);
    d_offset = vars(10);
    Ibx = vars(11);
    Iby = vars(12);
    Ibz = vars(13);
    fz = vars(14);

    if t < abs(mt0/mt_dot)
        lt = lt0 + lt_dot*t;
        mt = mt0 + mt_dot*t;
    else
        lt = lt0 + lt_dot*abs(mt0/mt_dot);
        mt = mt0 + mt_dot*abs(mt0/mt_dot);
    end

    h_noz_cm = ((l0+l_ic_prop)*mb + (l_ic_prop-lt)*mt)/(mb+mt);

    Mx = fz*(h_noz_cm*sind(alpha) + d_offset*cosd(alpha));

    Itx = mt*(r_motor^2/4 + lt^2/3);
    Ity = mt*(r_motor^2/4 + lt^2/3);
    Itz = mt*r_motor^2/2;

    Itxdot = mt_dot*(r_motor^2/4+lt^2/3) + 2/3*mt*lt*lt_dot;
    Itydot = mt_dot*(r_motor^2/4+lt^2/3) + 2/3*mt*lt*lt_dot;
    Itzdot = mt_dot*r_motor^2/2;

    Ix = Ibx + Itx + mt*(lt+l0)^2;
    Iy = Iby + Ity + mt*(lt+l0)^2;
    Iz = Ibz + Itz;
```

Fig. 16 Numerical Integration EOM setup 1



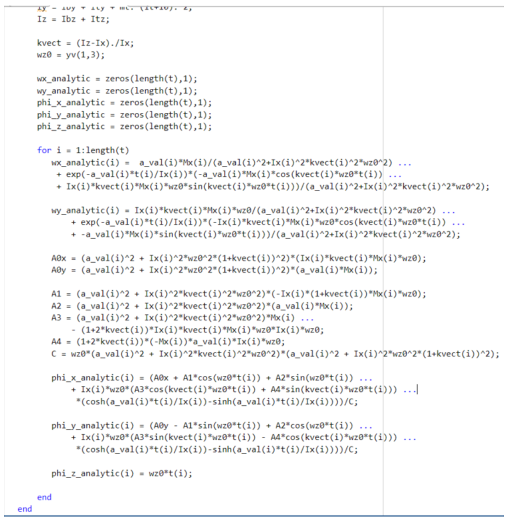
```
    fz = vars(14);

    if t < abs(mt0/mt_dot)
        lt = lt0 + lt_dot*t;
        mt = mt0 + mt_dot*t;
    else
        lt = lt0 + lt_dot*abs(mt0/mt_dot);
        mt = mt0 + mt_dot*abs(mt0/mt_dot);
    end

    h_noz_cm = ((l0+l_ic_prop)*mb + (l_ic_prop-lt)*mt)/(mb+mt);

    Mx = fz*(h_noz_cm*sind(alpha) + d_offset*cosd(alpha));

    Itx = mt*(r_motor^2/4 + lt^2/3);
    Ity = mt*(r_motor^2/4 + lt^2/3);
    Itz = mt*r_motor^2/2;

    Itxdot = mt_dot*(r_motor^2/4+lt^2/3) + 2/3*mt*lt*lt_dot;
    Itydot = mt_dot*(r_motor^2/4+lt^2/3) + 2/3*mt*lt*lt_dot;
    Itzdot = mt_dot*r_motor^2/2;

    Ix = Ibx + Itx + mt*(lt+l0)^2;
    Iy = Iby + Ity + mt*(lt+l0)^2;
    Iz = Ibz + Itz;

    Ixdot = Itxdot + mt_dot*(lt+l0)^2 + 2*mt*(lt+l0)*lt_dot;
    Iydot = Itydot + mt_dot*(lt+l0)^2 + 2*mt*(lt+l0)*lt_dot;
    Izdot = Itzdot;

    wx = x(1); wy = x(2); wz = x(3); px = x(4); py = x(5); pz = x(6);

    dwx = (Mx + (Iy-Iz)*wy*wz - (Ixdot-mt_dot*(h_noz_cm^2 + d_offset^2))*wx)/Ix;
    dwy = ((Iz-Ix)*wz*wx - (Iydot - mt_dot*h_noz_cm^2)*wy)/Iy;
    dwz = ((Ix-Iy)*wx*wy - (Izdot - mt_dot*d_offset^2)*wz)/Iz;

    % 312 Euler Angle
    dpx = wx*cos(py)+wz*sin(py);
    dpy = wy-(wz*cos(py)-wx*sin(py))*tan(px);
    dpz = (wz*cos(py)-wx*sin(py))*sec(px);

    y = [dwx;dwy;dwz;dpx;dpy;dpz];
end
```

Fig. 17 Numerical Integration EOM setup 2



```
function [wx_analytic, wy_analytic,  phi_x_analytic, phi_y_analytic, phi_z_analytic] = analytic_vals(t, vars,yv)

    mb = vars(1);
    l_ic_prop = vars(2);
    l0 = vars(3);
    lt0 = vars(4);
    lt_dot = vars(5);
    r_motor = vars(6);
    mt0 = vars(7);
    mt_dot = vars(8);
    alpha = vars(9);
    d_offset = vars(10);
    Ibx = vars(11);
    Iby = vars(12);
    Ibz = vars(13);
    fz = vars(14);

    lt = zeros(length(t),1);
    mt = zeros(length(t),1);
    lt(1) = lt0;
    mt(1) = mt0;

    %Make Ix, Iy, Iz, Mx constant by dropping the secular terms
    for i = 2:length(t)
        if t(i) < abs(mt0/mt_dot)
            lt(i) = lt(1) + lt_dot*t(i)*0;
            mt(i) = mt(1) + mt_dot*t(i)*0;
        else
            lt(i) = lt(1) + lt_dot*abs(mt0/mt_dot)*0;
            mt(i) = mt(1) + mt_dot*abs(mt0/mt_dot)*0;
        end
    end

    h_noz_cm = ((l0+l_ic_prop)*mb + (l_ic_prop-lt).*mt)./(mb+mt);

    Mx = fz.*(h_noz_cm.*sind(alpha) + d_offset*cosd(alpha));
    a_val = -mt_dot.*h_noz_cm.^2;

    Itx = mt.*(r_motor^2/4 + lt.^2/3);
    Ity = mt.*(r_motor^2/4 + lt.^2/3);
    Itz = mt.*r_motor^2/2;
```

Fig. 18 Analytical Solution Computation setup 1



```
    Iz = Ibz + Itz;

    kvect = (Iz-Ix)./Ix;
    wz0 = yv(1,3);

    wx_analytic = zeros(length(t),1);
    wy_analytic = zeros(length(t),1);
    phi_x_analytic = zeros(length(t),1);
    phi_y_analytic = zeros(length(t),1);
    phi_z_analytic = zeros(length(t),1);

    for i = 1:length(t)
       wx_analytic(i) =  a_val(i)*Mx(i)/(a_val(i)^2+Ix(i)^2*kvect(i)^2*wz0^2) ...
        + exp(-a_val(i)*t(i)/Ix(i))*(-a_val(i)*Mx(i)*cos(kvect(i)*wz0*t(i)) ...
        + Ix(i)*kvect(i)*Mx(i)*wz0*sin(kvect(i)*wz0*t(i)))/(a_val(i)^2+Ix(i)^2*kvect(i)^2*wz0^2);

       wy_analytic(i) = Ix(i)*kvect(i)*Mx(i)*wz0/(a_val(i)^2+Ix(i)^2*kvect(i)^2*wz0^2) ...
           + exp(-a_val(i)*t(i)/Ix(i))*(-Ix(i)*kvect(i)*Mx(i)*wz0*cos(kvect(i)*wz0*t(i)) ...
           + -a_val(i)*Mx(i)*sin(kvect(i)*wz0*t(i)))/(a_val(i)^2+Ix(i)^2*kvect(i)^2*wz0^2);

       A0x = (a_val(i)^2 + Ix(i)^2*wz0^2*(1+kvect(i))^2)*(Ix(i)*kvect(i)*Mx(i)*wz0);
       A0y = (a_val(i)^2 + Ix(i)^2*wz0^2*(1+kvect(i))^2)*(a_val(i)*Mx(i));

       A1 = (a_val(i)^2 + Ix(i)^2*kvect(i)^2*wz0^2)*(-Ix(i)*(1+kvect(i))*Mx(i)*wz0);
       A2 = (a_val(i)^2 + Ix(i)^2*kvect(i)^2*wz0^2)*(a_val(i)*Mx(i));
       A3 = (a_val(i)^2 + Ix(i)^2*kvect(i)^2*wz0^2)*Mx(i) ...
           - (1+2*kvect(i))*Ix(i)*kvect(i)*Mx(i)*wz0*Ix(i)*wz0;
       A4 = (1+2*kvect(i))*(-Mx(i))*a_val(i)*Ix(i)*wz0;
       C = wz0*(a_val(i)^2 + Ix(i)^2*kvect(i)^2*wz0^2)*(a_val(i)^2 + Ix(i)^2*wz0^2*(1+kvect(i))^2);

       phi_x_analytic(i) = (A0x + A1*cos(wz0*t(i)) + A2*sin(wz0*t(i)) ...
           + Ix(i)*wz0*(A3*cos(kvect(i)*wz0*t(i)) + A4*sin(kvect(i)*wz0*t(i))) ...
            *(cosh(a_val(i)*t(i)/Ix(i))-sinh(a_val(i)*t(i)/Ix(i))))/C;

       phi_y_analytic(i) = (A0y - A1*sin(wz0*t(i)) + A2*cos(wz0*t(i)) ...
           + Ix(i)*wz0*(A3*sin(kvect(i)*wz0*t(i)) - A4*cos(kvect(i)*wz0*t(i))) ...
            *(cosh(a_val(i)*t(i)/Ix(i))-sinh(a_val(i)*t(i)/Ix(i))))/C;

       phi_z_analytic(i) = wz0*t(i);

    end
end
```

Fig. 19 Analytical Solution Computation setup 2