8-23-2011

Attitude Dynamics and Passive Control of a Thrusting, Spinning Spacecraft with Fuel Sloshing

Farhad A. Goodarzi
Santa Clara University

Santa Clara University

Department of Mechanical Engineering

Date: August 23, 2011

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Farhad A. Goodarzi

ENTITLED

Attitude Dynamics and Passive Control of a Thrusting, Spinning Spacecraft with Fuel Sloshing

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

Thesis Advisor
Dr. Mohammad A. Ayoubi

Chairman of Department
Dr. Drazen Fabris

Attitude Dynamics and Passive Control of a Thrusting, Spinning Spacecraft with Fuel Sloshing

By

Farhad A. Goodarzi

Dissertation

Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Mechanical Engineering
in the School of Engineering at
Santa Clara University, 2011

Santa Clara, California

In the hallway of the university Wittgenstein asked a colleague: "I've always wondered why for so long people thought that the sun revolved around the earth."

"Why?" said his surprised interlocutor, "well, I suppose it just looks that way."

"Hmm", retorted Wittgenstein, "and what would it look like if the earth revolved around the sun?"

This puzzled the interlocutor.

Acknowledgments

I wish to express my gratitude to my supervisor, Dr. Mohammad Ayoubi who was abundantly helpful and offered invaluable assistance, support and guidance. Deepest gratitude are also due to Dr. Arun Banerjee who helped me on different part of this project.

I wish to express my love and gratitude to my beloved Mother; for her understanding, supporting and endless love, through the duration of my studies.

Farhad A. Goodarzi

Attitude Dynamics and Passive Control of a Thrusting, Spinning Spacecraft with Fuel Sloshing

Farhad A. Goodarzi

Department of Mechanical Engineering
Santa Clara University
Santa Clara, California
2011

ABSTRACT

We use Kane's method to present the equations of motion of a spinning spacecraft with three momentum wheels, a nutation damper, and a spherical pendulum. The spherical pendulum is adopted as a simple mechanical equivalent of fuel sloshing in partially-filled tanks. The proposed model is an extension of the existing model in the literature. We verify and validate our model for two cases: flat spin and a simple reorientation maneuver. Numerical simulations are in agreement with existing results in the literature.

Errors in proper orientation of a desired velocity vector of a thrusting, spinning spacecraft are caused by thrust misalignment of various kind. We use a trapezoidal thrust scheme to reduce the error in a spacecraft with fuel sloshing. Equations of motion of the system are derived using Kane's method accounting time-vary moment of inertia, torque and mass with planar and spherical pendulums models of fuel slosh that are the most accurate mechanical equivalent models available in the literature. Comparison between the Star 48B thrust scheme and proposed trapezoidal thrust scheme are proposed with optimizing the trapezoidal thrust scheme in order to minimize the average velocity pointing error. Simulation results for an existing simple thrusting rigid body model confirm the accuracy of our model.

Nomenclature

$AVPE$	= Average velocity pointing error, rad
a	= linear acceleration, m/s^2
b_x, b_y, b_z	= body fixed frame basis vectors
c	= damping coefficient, $N.s/m$
C_S	= spherical pendulum torque coefficient, $N.s/m$
c	= cosine
d	= center of mass(COM) offset, m
F_{Max}	= maximum thrust force, N
F_P	= pendulum force, N
F_r	= Generalized active forces, N
F_r^*	= Generalized inertia forces, N
F_{Th}	= thrust force, N
$h(t)$	= distance from throat of nozzle to COM, m
I_x, I_y, I_z	= principal moments of inertia, $kg.m^2$
k	= spring stiffness, N/m
k_P	= pendulum spring stiffness, N/m
l_P	= planar pendulum length, m
l_S	= spherical pendulum length, m
m_B	= body mass, kg
m_P	= planar pendulum mass, kg
m_S	= spherical pendulum mass, kg

m_Q = nutation damper point mass, kg

$\dot{m}$ = mass flow rate, kg/s

n_x, n_y, n_z = inertial frame basis vectors

p_x, p_y, p_z = pendulum frame basis vectors

q = nutation damper displacement, m

R = 3×3 transformation matrix

s = sine

T_{fP} = spherical pendulum torque, $N.m$

t = time, s

t_b = burn time, s

t_r = ramp up time, s

t_{rf} = ramp down time, s

$u_1, u_2, ..., u_{13}$ = generalized speeds

v = linear velocity, m/s

z_0 = spherical pendulum attachment distance to center of mass, m

z_Q = nutation damper point mass vertical distance to center of mass, m

α = thrust misalignment angle, rad

γ_1, γ_2 = velocity pointing error angles, rad

$\mathcal{I}$ = Total impulse, $N.s$

μ = mass fraction

$\rho(t)$ = velocity pointing error, rad

ψ_1, ψ_2 = relative orientation angles of spherical pendulum, rad

ψ, θ, ϕ = Euler angles, rad

Ω = spin rate, rpm

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The problem of coning motion of a spin-stabilized spacecraft was observed first in the 1980s during orbit transfer in Perigee Assist Module, Delta Class (PAM-D) solid boost rockets. Several hypotheses for this motion were proposed but based on flight data, only two of them could survive. The first one attributes the coning to instability in the combustion chamber, which is the so-called Jet Gain theory[5,6] and the second one is called Slag Pool theory, which states that the source of coning motion is the liquified slag accumulated in the aft portion of a solid rocket fuel.[2,7,8] An equivalent mechanical system—a spherical pendulum model—has been adopted to model the fuel motion in partially-filled tanks which are often located at the forward end of the spacecraft.[1,4,9,18,20,21]

1.1 Literature Review

In the last two decades, several researchers have studied the stability problem of the spacecraft's coning motion. Mingori and Yam[19] found a linear stability criterion for an axisymmetric spacecraft, using a plannar pendulum as a mechanical equivalent of fuel sloshing. Cochran and Kang[3] studied the attitude motion of a spinning asymmetric rigid body with a spherical pendulum on it. Yam et al.[22] investigated the stability of the previous model when a dissipative energy element is added. Or[21] examined the stability of a thrusting, spinning spacecraft. Kang and

Cochran[15] studied nonlinear resonance motion of an axisymmetric rigid body with a spherical pendulum as a model of a slag pool in steady state spin. In 2008, Kang and Lee[16] investigated the attitude motion of a rigid spacecraft with a momentum wheel along the spin axis and a spherical pendulum. They carried out some numerical simulations for two cases: 1) no viscocity in the model and 2) linear viscous dissipation.

The problem of coning motion of thrusting, spinning spacecraft was first reported in the1960s in some class of upper stage solid rocket boosters. Since then, many researchers have studied this problem and proposed different theories and models to justify this behavior. A partial list of some excellent papers in modeling and stability analysis are Refs.[1,3,4,8,10,16,18–22]

1.2 Thesis Overview

This study is an extension of the existing models in literature. We consider a spinning upper stage spacecraft with partially-filled fuel tanks with dissipation in moving mass. The model is assumed to have a constant mass with three momentum wheels, a spherical pendulum, and a dissipative moving mass or a nutation damper. We use Kane's method to derive the nonlinear equations of motion for such a spacecraft. The model is verified and compared with the numerical results reported by Kang and Lee.[16]

We test the efficacy of the trapezoidal thrust scheme on the velocity pointing error of a thrusting, spinning gyrostat with fuel sloshing and thrust misalignment.[17] We adopted two existing equivalent mechanical models, a spherical and a planar pendulum, to model the fuel sloshing in partially-filled tanks.[1,4,10,18,20,21] In addition, we introduce "optimal" trapezoidal thrust profiles by minimizing the average velocity

pointing error and satisfying the burn time, maximum thrust, and specific impulse constraints. In this project, we use the flight test data of star 48B rocket[10,12,13]— a Perigee Assisted Module (PAM-D)— to simulate the motions and to optimize and test the optimal trapezoidal thrust profiles. The numerical results confirm the effectiveness of this type of passive control method.

Chapter 2

Modeling of Spacecraft

2.1 Introduction

We considered the spacecraft as a rigid body with three momentum wheels, a point mass, spring and nutation damper attached to the rigid body. There are different ways to drive the equation of motion of a system such as, Newton-Euler and Kane's method.

We choose Kane's method to drive the equation of motion for our system in order to get the most simplified and accurate model.

In this chapter, We briefly discuses the Kane's method and then show how we set up the spacecraft model. Equation of motion for planar and spherical pendulums are presented in the following.

2.2 Kane's Method

Kane's method can be viewed as an automated version of the Motion Law, where no analyst decisions are required to produce a minimal set of dynamics equations in which unwanted reaction forces and torques do not appear. The key concept underlying Kane's method is that of partial velocities, which are described in more detail below. Partial Velocities:

Generalized coordinates q_r - time-varying translations and rotations selected to

define the position of all points and the orientation of all rigid bodies.

$$q_r \qquad ; \qquad (r = 1, ..., n) \tag{2.1}$$

where n is the number of degrees of freedom. Generalized speeds u_r - time-varying linear functions of the $\dot{q}_r$ ' s selected so as to simplify expressions for velocities of points and angular velocities of rigid bodies.

$$u_r \triangleq \sum Y_{rs}\dot{q}_s + Z_r \qquad ; \qquad (r = 1, ..., n) \tag{2.2}$$

where Y_{rs} and Z_r are functions of $q_1, ..., q_n$ and the time t. Note that while $u_r \triangleq \dot{q}_r$ is the simplest and most obvious definition, a more advantageous, though more complex, definition may also exist (see, for example, Mitiguy and Kane, 1996). For any definition, above equation must yield unique solutions for $\dot{q}_1, ..., \dot{q}_n$ as a function of $u_1, ..., u_n$.

Example: Consider the constrained velocity ${}^N\underline{v}^P$ of a fictitious particle P in a Newtonian reference frame N, where P is part of a larger system requiring three generalized coordinates:

$${}^N\underline{v}^P = (\dot{q}_1 cosq_3 + \dot{q}_2 sinq_3)\underline{n}_1 + (-\dot{q}_1 sinq_3 + \dot{q}_2 cosq_3)\underline{n}_2 - q_1\underline{n}_3 \tag{2.3}$$

where n_1 , n_2 , and n_3 form a right-handed set of mutually perpendicular unit vectors fixed in N. If we define

$$u_1 \triangleq \dot{q}_1 cosq_3 + \dot{q}_2 sinq_3 \qquad ; \qquad u_2 \triangleq -\dot{q}_1 sinq_3 + \dot{q}_2 cosq_3 \qquad ; \qquad u_3 \triangleq \dot{q}_3 \tag{2.4}$$

Then we can rewrite ${}^N\underline{v}^P$ as

$$ {}^N\underline{v}^P = u_1\underline{n}_1 + u_2\underline{n}_2 - q_1\underline{n}_3 \tag{2.5} $$

Partial angular velocities ω_r and partial velocities v_r - time-varying linear functions of the u_r ' s determined by inspection and which greatly facilitate the formulation of equations of motion.

$$ \underline{\omega} = \sum \underline{\omega}_r u_r + \underline{\omega}_t \tag{2.6} $$

$$ \underline{v} = \sum \underline{v}_r u_r + \underline{v}_t \tag{2.7} $$

where ω is the angular velocity of a rigid body, v is the velocity of a point, and ω_r , v_r , ω_t , and v_t are functions of $q_1, ..., q_n$ and t. In principle, partial angular velocities need only be formed for those rigid bodies subjected to applied torques or possessing inertia, while partial velocities need only be formed for those points subjected to applied forces or possessing mass. Example: By rewriting the above expression for ${}^N\underline{v}^P$ as

$$ {}^N\underline{v}^P = u_1\underline{n}_1 + u_2\underline{n}_2 + u_3(0) - q_1\underline{n}_3 \tag{2.8} $$

the three partial velocities associated with particle P are found to be

$$ {}^N\underline{v}_1^P = \underline{n}_1 \qquad ; \qquad {}^N\underline{v}_2^P = \underline{n}_2 \qquad ; \qquad {}^N\underline{v}_3^P = 0 \tag{2.9} $$

while ${}^N\underline{v}_t^P = -q_1\underline{n}_3$. Generalized active forces F_r - quantities formed by taking dot (i.e., scalar) products of partial velocities and active (i.e., applied) forces and dot products of partial angular velocities and active torques. For each point P_i

subjected to an applied force,

$$(F_r)_{P_i} = \underline{v}_r^{P_i}.R_{P_i} \qquad ; \qquad (r = 1, ..., n) \tag{2.10}$$

where $v_r^{P_i}$ is the r^{th} partial velocity of P_i and R_{P_i} is the resultant of all contact and distance forces acting on P_i . Similarly, for each rigid body B_j subjected to an applied torque,

$$(F_r)_{B_j} = \underline{\omega}_r^{B_j}.T_{B_j} \qquad ; \qquad (r = 1, ..., n) \tag{2.11}$$

where $\omega_r^{B_j}$ is the r^{th} partial angular velocity of B_j and T_{B_j} is the resultant of all couples acting on B_j .The r^{th} generalized active force F_r can then be determined by summing the results over all points P_i and all rigid bodies B_j :

$$F_r = \sum (F_r)_{P_i} + \sum (F_r)_{B_j} \qquad ; \qquad (r = 1, ..., n) \tag{2.12}$$

where κ is the number of points subjected to applied forces and λ is the number of rigid bodies subjected to applied torques.

Generalized inertia forces F_r^* - quantities formed by taking dot products of partial velocities and inertia forces and dot products of partial angular velocities and inertia torques. For each point P_i possessing mass,

$$(F_r^*)_{P_i} = \underline{v}_r^{P_i}.R_{P_i}^* \qquad ; \qquad (r = 1, ..., n) \tag{2.13}$$

where Pi v_r is the r^{th} partial velocity of P_i and $R_{P_i}^*$ is the inertia force for P_i , defined as

$$R_{P_i}^* \triangleq -m_{P_i}\underline{a}^{P_i} \tag{2.14}$$

where m_{P_i} is the mass of P_i and a^{P_i} is the acceleration of P_i . Similarly, for each rigid body B_j possessing inertia,

$$(F_r^*)_{B_j} = \underline{\omega}_r^{B_j}.T_{B_j}^* \qquad ; \qquad (r = 1, ..., n) \tag{2.15}$$

where $\omega_r^{B_j}$ is the r^{th} partial angular velocity of B_j and B_j^* is the inertia torque for B_j , defined as

$$T_{B_j}^* \triangleq -\underline{\alpha}^{B_j}.\underline{\underline{I}}^{B_j/B_j^*} - \underline{\omega}^{B_j} \times \underline{\underline{I}}^{B_j/B_j^*}.\underline{\omega}^{B_j} \tag{2.16}$$

where I^{B_j/B_j^*} is the inertia dyadic of B_j about its mass center B_j^* , ω^{B_j} is the angular velocity of B_j , and α^{B_j} is the angular acceleration of B_j . Note that a dyadic is an expression of the form

$$\underline{\underline{I}} = \sum \sum \underline{\underline{I}}_{jk} n_j n_k \tag{2.17}$$

The r_{th} generalized inertia force F_r^* can then be determined by summing the results over all points P_i and all rigid bodies B_j:

$$F_r^* = \sum (F_r^*)_{P_i} + \sum (F_r^*)_{B_j} \tag{2.18}$$

where μ is the number of points possessing mass and is the number of rigid bodies possessing inertia. Equations of motion $F_r + F_r^* = 0$- once all generalized active forces and generalized inertia forces are known, the equations of motion can be formulated by simply adding the results:

$$F_r + F_r^* = 0 \qquad ; \qquad (r = 1, ..., n) \tag{2.19}$$

One arrives at above equation by following a very systematic process which does not require high level mathematics, calculation of unwanted interaction forces, or use of virtual work principles.

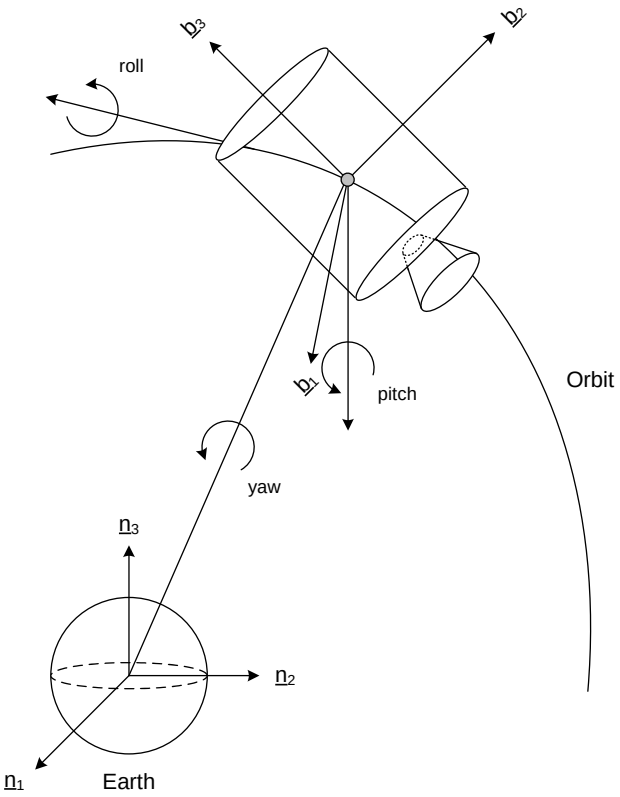


Fig. 2.1: Gyrostat model in the orbit.

2.3 Derivation of Equation of Motion

Consider a spin-stabilized spacecraft with three momentum wheels, w_1, w_2, w_3, aligned with the gyrostat (spacecraft+momentum wheels) principal axes, $\underline{b}_1$, $\underline{b}_2$, and $\underline{b}_3$ and a nutation damper along the body axis $\underline{b}_1$ as shown in Fig. 2.3. We denote the center-of-mass of the spacecraft and momentum wheels by G^* and the central principal moment-of-inertia of gyrostat by I^{G^*}. We adopt the mechanical equivalent of fuel sloshing, a spherical pendulum of mass m_P, to model the fuel sloshing in the gyrostat. The nutation damper is represented by a mass, m_Q, supported by a spring of stiffness, k, and damping coefficient, c.

2.3.1 Generalized speeds

We define the generalized speeds, u_i ; $(i = 1, \ldots, 12)$ as follows:

$$u_i = {}^N \underline{V}^{G^*} . \underline{b}_i \qquad ; \qquad (i = 1, 2, 3) \tag{2.20}$$

$$u_{3+i} = {}^N \underline{\omega}^{G^*} . \underline{b}_i \qquad ; \qquad (i = 1, 2, 3) \tag{2.21}$$

$$u_{6+i} = {}^{G^*} \underline{\omega}^{w_i} . \underline{b}_i \qquad ; \qquad (i = 1, 2, 3) \tag{2.22}$$

$$u_{9+i} = \dot{\psi}_i \qquad ; \qquad (i = 1, 2) \tag{2.23}$$

$$u_{12} = \dot{q} \tag{2.24}$$

where u_1, u_2, and u_3 are the components of the velocity of the gyrostat center of mass, G^*, u_4, u_5, and u_6 are the components of the angular velocity of the gyrostat. The angular speed of momentum wheels w_1, w_2, and w_3 are u_7, u_8, and u_9, respectively. The generalized speeds $u_{10} = \dot{\psi}_1$ and $u_{11} = \dot{\psi}_2$ are used to specify the orientation of the massless rod of the pendulum in the gyrostat as shown in Fig. **??**. In Eq. (2.24), q is the displacement of the mass particle, m_Q.

2.3.2 Velocities and Accelerations

The velocity of point G^*, angular velocity of gyrostat, angular velocity of the momentum wheels with respect to inertial frame N can be written as:

$${}^N \underline{V}^{G^*} = u_1 \, \underline{b}_1 + u_2 \, \underline{b}_2 + u_3 \, \underline{b}_3 \tag{2.25}$$

$${}^N \underline{\omega}^{G} = u_4 \, \underline{b}_1 + u_5 \, \underline{b}_2 + u_6 \, \underline{b}_3 \tag{2.26}$$

$${}^N \underline{\omega}^{w_1} = (u_4 + u_7) \, \underline{b}_1 + u_5 \, \underline{b}_2 + u_6 \, \underline{b}_3 \tag{2.27}$$

$$^{N}\underline{\omega}^{w_2} = u_4\ \underline{b}_1 + (u_5 + u_8)\ \underline{b}_2 + u_6\ \underline{b}_3 \tag{2.28}$$

$$^{N}\underline{\omega}^{w_3} = u_4\ \underline{b}_1 + u_5\ \underline{b}_2 + (u_6 + u_9)\ \underline{b}_3 \tag{2.29}$$

Similarly, if we denote the displacement and velocity of the nutation damper mass point, m_Q, with respect to the damper fixed-frame ($\underline{b}_1$, $\underline{b}_2$, $\underline{b}_3$), q, and $\dot{q}$, respectively, the velocity of the point Q will be

$$^{N}\underline{V}^{Q} =^{N} \underline{V}^{G^*} +^{N} \underline{\omega}^{G} \times (z_Q\underline{b}_3 + q\underline{b}_1) + \dot{q}\underline{b}_1 \tag{2.30}$$

and we get

$$^{N}\underline{V}^{Q} = (u_1 + u_5 z_Q + u_{12})\underline{b}_1 + (u_2 - u_4 z_Q + q u_6)\underline{b}_2 + (u_3 - q u_5)\underline{b}_3 \tag{2.31}$$

The acceleration of point G^* with respect to the inertial frame, N, can be determined as

$$^{N}\underline{a}^{G^*} = \frac{^{G}d}{dt}\ ^{N}\underline{V}^{G^*} +^{N} \underline{\omega}^{G} \times^{N} \underline{V}^{G^*} \tag{2.32}$$

or

$$^{N}\underline{a}^{G^*} = (\dot{u}_1 + Z_1)\underline{b}_1 + (\dot{u}_2 + Z_2)\underline{b}_2 + (\dot{u}_3 + Z_3)\underline{b}_3 \tag{2.33}$$

Here Z_1, Z_2, and Z_3 are defined as

$$Z_1 \triangleq u_5 u_3 - u_6 u_2 \tag{2.34}$$

$$Z_2 \triangleq u_6 u_1 - u_4 u_3 \tag{2.35}$$

$$Z_3 \triangleq u_4 u_2 - u_5 u_1 \tag{2.36}$$

The acceleration of the nutation damper, point Q, can be written from:

$$^{N}\underline{a}^{Q} = \frac{^{G}d}{dt}\ ^{N}\underline{V}^{Q} +^{N}\ \underline{\omega}^{G} \times^{N}\ \underline{V}^{Q} \tag{2.37}$$

as

$$^{N}\underline{a}^{Q} = (\dot{u}_1 + \dot{u}_{12} + z_Q\dot{u}_5 + Z_7)\underline{b}_1 + (\dot{u}_2 - z_Q\dot{u}_4 + q\dot{u}_6 + Z_8)\underline{b}_2 + (\dot{u}_3 - q\dot{u}_5 + Z_9)\underline{b}_3 \tag{2.38}$$

where Z_7, Z_8, and Z_9 are defined as:

$$Z_7 \triangleq Z_1 - qu_5^2 + z_Q u_4 u_6 - qu_6^2 \tag{2.39}$$

$$Z_8 \triangleq Z_2 + qu_4 u_5 + 2u_{12}u_6 + z_Q u_5 u_6 \tag{2.40}$$

$$Z_9 \triangleq Z_3 - z_Q u_4^2 - 2u_{12}u_5 - z_Q u_5^2 + qu_4 u_6 \tag{2.41}$$

2.3.3 Spherical Pendulum

The spherical pendulum has a fixed length of l_P and is attached below point G^* on the $\underline{b}_3$ axis as shown in Figs. 2.2, 2.3. It can move freely inside the gyrostat. The angular velocity of pendulum with respect to inertial frame N is

$$^{N}\underline{\omega}^{P} =^{N}\ \underline{\omega}^{G} + \dot{\psi}_1\ \underline{b}_3 - \dot{\psi}_2\ \underline{p}_2 \tag{2.42}$$

We can show that the transformation matrix from body-fixed frame ($\underline{b}_1$, $\underline{b}_2$, $\underline{b}_3$) into the pendulum coordinate system ($\underline{p}_1$, $\underline{p}_2$, $\underline{p}_3$) is

$$R(\psi_1, \pi/2 - \psi_2) = R_2(\pi/2 - \psi_2)R_3(\psi_1) \tag{2.43}$$

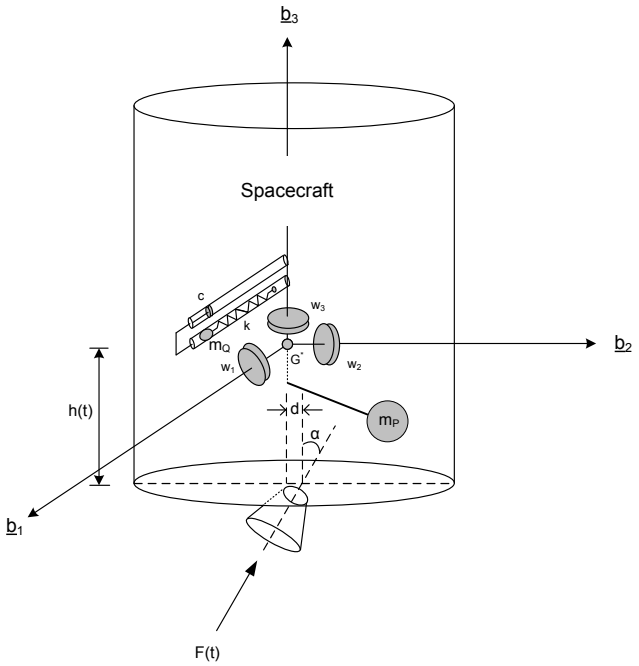


Fig. 2.2: A gyrostat model with spherical pendulum.

$$R(\psi_1, \pi/2 - \psi_2) = \begin{bmatrix} c\psi_1 s\psi_2 & s\psi_1 s\psi_2 & -c\psi_2 \\ -s\psi_1 & c\psi_1 & 0 \\ c\psi_2 c\psi_2 & c\psi_2 s\psi_1 & s\psi_2 \end{bmatrix} \tag{2.44}$$

Using the transformation matrix Eq. (2.44) and Eq. (2.26), Eq. (2.42) can be written in the body-fixed frame as:

$$^N\underline{\omega}^P = (u_4 + u_{11} s\psi_1)\ \underline{b}_1 + (u_5 - u_{11} c\psi_1)\ \underline{b}_2 + (u_{10} + u_6)\ \underline{b}_3 \tag{2.45}$$

The velocity of the pendulum mass m_P can be written as:

$$^N\underline{V}^P = ^N\ \underline{V}^{G^*} + ^N\ \underline{\omega}^P \times \underline{r}^{G^*P} \tag{2.46}$$

where

$$\underline{r}^{G^*P} = l_P(c\psi_1 s\psi_2 \underline{b}_1 + s\psi_1 s\psi_2 \underline{b}_2 - c\psi_2 \underline{b}_3) + z_o \underline{b}_3 \tag{2.47}$$

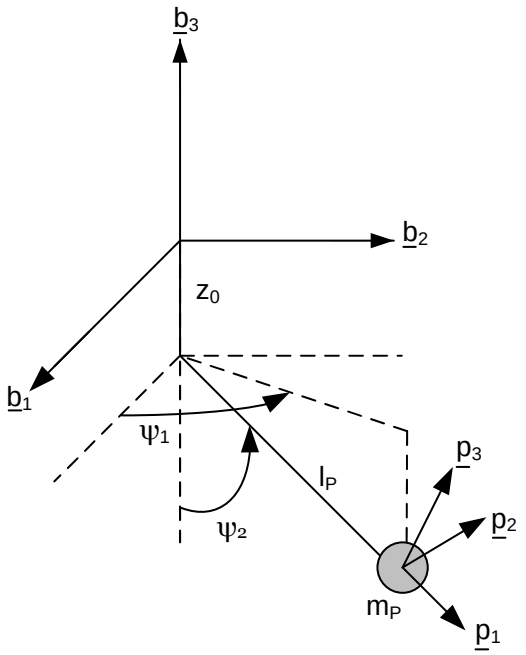


Fig. 2.3: Angles ψ_1 and ψ_2 represent the relative orientation of spherical pendulum inside the spacecraft.

After simplification, we get

$$
\begin{aligned}
{}^N\underline{V}^P = &\{u_1 + l_P[-(u_5 - u_{11}c\psi_1)c\psi_2 - (u_6 + u_{10})s\psi_1 s\psi_2] + z_o(u_5 - c\psi_1 u_{11}\} \ \underline{b}_1 \\
&+ \{u_2 + l_P[(u_6 + u_{10})c\psi_1)s\psi_2 + (u_4 + u_{11}s\psi_1)c\psi_2] - z_o(u_4 + s\psi_1 u_{11}\} \ \underline{b}_2 \\
&+ \{u_3 + l_P[(u_4 + u_{11}s\psi_1)s\psi_1 s\psi_2 - (u_5 - u_{11})c\psi_1)c\psi_1 s\psi_2]\} \ \underline{b}_3
\end{aligned}
\tag{2.48}
$$

Similarly, for the absolute acceleration of the pendulum mass P, we obtain

$$
{}^N\underline{a}^P = \frac{{}^G d}{dt} \, {}^N\underline{V}^P + {}^N\underline{\omega}^G \times {}^N\underline{V}^P \tag{2.49}
$$

or

$$
\begin{aligned}
{}^N\underline{a}^P = \{&\dot{u}_1 - l_P s\psi_1 s\psi_2 \dot{u}_{10} + l_P c\psi_1 c\psi_2 \dot{u}_{11} - l_P c\psi_2 \dot{u}_5 - l_P s\psi_1 s\psi_2 \dot{u}_6 + z_o(\dot{u}_5 - c\psi_1 \dot{u}_{11}) \\
&+ Z_4\}\underline{b}_1 + \{\dot{u}_2 + l_P c\psi_1 s\psi_2 \dot{u}_{10} + l_P c\psi_2 s\psi_1 \dot{u}_{11} - l_P c\psi_2 \dot{u}_4 - l_P c\psi_1 s\psi_2 \dot{u}_6 \\
&- z_o(\dot{u}_4 - s\psi_1 \dot{u}_{11}) + Z_5\}\underline{b}_2 + \{\dot{u}_3 + l_P s\psi_2 \dot{u}_{11} + l_P s\psi_2 s\psi_1 \dot{u}_4 - l_P s\psi_2 c\psi_1 \dot{u}_5 \\
&+ z_o(c\psi_1 u_{11} u_5 - s\psi_1 u_{11} u_4 - u_4^2 - u_5^2) + Z_6\}\underline{b}_3
\end{aligned}
\tag{2.50}
$$

where Z_4, Z_5, and Z_6 are defined as follows:

$$
\begin{aligned}
Z_4 \triangleq\ & l_P c\psi_1 s\psi_2 u_{10}^2 - l_P c\psi_1 s\psi_2 u_{11}^2 + Z_1 + l_P s\psi_1 s\psi_2 u_4 u_5 - l_P c\psi_1 s\psi_2 u_5^2 \\
&- l_P c\psi_2 u_4 u_6 - l_P c\psi_1 s\psi_2 u_6^2 + 2l_P u_{11} s\psi_2 u_5 - 2l_P u_{11} c\psi_2 s\psi_1 u_6 - 2l_P u_{10} c\psi_2 s\psi_1 u_{11} \\
&- 2l_P u_{10} c\psi_1 s\psi_2 u_6 + z_o(u_4 u_6 + s\psi_1 u_{10} u_{11} + s\psi_1 u_{11} u_6)
\end{aligned}
\tag{2.51}
$$

$$
\begin{aligned}
Z_5 \triangleq\ & -l_P s\psi_1 s\psi_2 u_{10}^2 - l_P s\psi_1 s\psi_2 u_{11}^2 + Z_2 + l_P c\psi_1 s\psi_2 u_4 u_5 - l_P s\psi_1 s\psi_2 u_4^2 \\
&- l_P c\psi_2 u_5 u_6 - l_P s\psi_1 s\psi_2 u_6^2 - 2l_P u_{11} s\psi_2 u_4 + 2l_P u_{11} c\psi_1 c\psi_2 u_6 + 2l_P u_{10} c\psi_1 c\psi_2 u_{11} \\
&- 2l_P u_{10} s\psi_1 s\psi_2 u_6 + z_o(u_5 u_6 - c\psi_1 u_{10} u_{11} - c\psi_1 u_{11} u_6)
\end{aligned}
\tag{2.52}
$$

$$
\begin{aligned}
Z_6 \triangleq\ & l_P c\psi_2 u_{11}^2 + Z_3 + l_P c\psi_2 u_4^2 + l_P c\psi_2 u_5^2 - (z_o - 2l_P c\psi_2) u_{11}(s\psi_1 u_4 - c\psi_1 u_5) \\
&+ 2l_P s\psi_2 c\psi_1 u_4 u_{10} + 2l_P s\psi_2 s\psi_1 u_{10} u_5 + l_P c\psi_1 s\psi_2 u_4 u_6 + l_P s\psi_1 s\psi_2 u_5 u_6 - z_o(u_4^2 + u_5^2)
\end{aligned}
\tag{2.53}
$$

2.3.4 Generalized Active and Inertia Forces

If F_r^* and F_r represent the generalized inertia and active forces, respectively, then Kane's Eqs. can be written as:[14]

$$-F_r^* = F_r \qquad ; \qquad (r = 1, ..., 12) \tag{2.54}$$

Negative of the generalized inertia force is given by

$$-F_r^* = (\underline{\underline{I}}^{G^*}.{}^N\underline{\alpha}^G + {}^N\underline{\omega}^G \times \underline{\underline{I}}^{G^*}.{}^N\underline{\omega}^G).{}^N\underline{\omega}_r^G + \sum_{i=1}^{3}(\underline{\underline{I}}^{w_i^*}.{}^N\underline{\alpha}^{w_i} + {}^N\underline{\omega}^{w_i} \times \underline{\underline{I}}^{w_i^*}.{}^N\underline{\omega}^{w_i}).{}^N\underline{\omega}_r^{w_i}$$
$$+ m_G\ {}^N\underline{a}^{G^*}.{}^N\underline{V}_r^{G^*} + m_P\ {}^N\underline{a}^P.{}^N\underline{V}_r^P + m_Q\ {}^N\underline{a}^Q.{}^N\underline{V}_r^Q \qquad , \qquad (r = 1, ..., 12) \tag{2.55}$$

where

$$\underline{\underline{I}}^{G^*} = \begin{bmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{bmatrix} \tag{2.56}$$

$$\underline{\underline{I}}^{w_1^*} = \begin{bmatrix} I_w & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.57}$$

$$\underline{\underline{I}}^{w_2^*} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & I_w & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.58}$$

and

$$\underline{\underline{I}}^{w_3^*} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & I_w \end{bmatrix} \tag{2.59}$$

The partial velocities and partial angular velocities appearing in Eq. (2.55) are calculated and summarized in Table 2.1. The Generalized Active Force, F_r is written

Table 2.1: Partial Velocity and Partial Angular Velocity

r	${}^N\underline{V}_r^{G^*}$	${}^N\underline{\omega}_r^G$	${}^N\underline{\omega}_r^{w_1}$	${}^N\underline{\omega}_r^{w_2}$	${}^N\underline{\omega}_r^{w_3}$	${}^N\underline{V}_r^P$	${}^N\underline{V}^Q$	${}^{\widehat{Q}}\underline{V}^Q$
1	$\underline{b}_1$	0	0	0	0	$\underline{b}_1$	$\underline{b}_1$	0
2	$\underline{b}_2$	0	0	0	0	$\underline{b}_2$	$\underline{b}_2$	0
3	$\underline{b}_3$	0	0	0	0	$\underline{b}_3$	$\underline{b}_3$	0
4	0	$\underline{b}_1$	$\underline{b}_1$	$\underline{b}_1$	$\underline{b}_1$	$-z_o\underline{b}_2 + l_P[c\psi_2\underline{b}_2 + s\psi_1 s\psi_2\underline{b}_3]$	$-z_Q\underline{b}_2$	0
5	0	$\underline{b}_2$	$\underline{b}_2$	$\underline{b}_2$	$\underline{b}_2$	$z_o\underline{b}_1 + l_P[-c\psi_2\underline{b}_1 - c\psi_1 s\psi_2\underline{b}_3]$	$z_Q\underline{b}_1 - q\underline{b}_3$	0
6	0	$\underline{b}_3$	$\underline{b}_3$	$\underline{b}_3$	$\underline{b}_3$	$l_P[-s\psi_2 s\psi_2\underline{b}_1 + c\psi_1 s\psi_2\underline{b}_2]$	$q\underline{b}_2$	0
7	0	0	$\underline{b}_1$	0	0	0	0	0
8	0	0	0	$\underline{b}_2$	0	0	0	0
9	0	0	0	0	$\underline{b}_3$	0	0	0
10	0	0	0	0	0	$l_P[-s\psi_1 s\psi_2\underline{b}_1 + c\psi_1 s\psi_2\underline{b}_2]$	0	0
11	0	0	0	0	0	$-z_o c\psi_1\underline{b}_1 - z_o\underline{b}_2 +$ $l_P[c\psi_1 c\psi_2\underline{b}_1 + s\psi_1 c\psi_2\underline{b}_2 + s\psi_2\underline{b}_3]$	0	0
12	0	0	0	0	0	0	$\underline{b}_1$	$\underline{b}_1$

as:[14]

$$\begin{aligned} \underline{F}_r = \underline{F}_{gG}.{}^N\underline{V}_r^{G^*} + \sum_{i=1}^{3} \underline{T}_i.{}^G\underline{\omega}_r^{w_i} + \underline{F}_{gQ}.{}^N\underline{V}_r^Q + \underline{F}_{gP}.{}^N\underline{V}_r^P - \sum_{i=1}^{3} \underline{T}_i.{}^N\underline{\omega}_r^G \\ + \underline{F}_{Th}.{}^N\underline{V}_r^F - (kq + c\dot{q}).{}^{\widehat{Q}}\underline{V}_r^Q + \underline{M}_{gG}.{}^N\underline{\omega}_r^G + \underline{T}_{fP}.{}^N\underline{\omega}_r^G \qquad ; \qquad (r = 1, \ldots, 12) \end{aligned} \tag{2.60}$$

where $\underline{F}_{gG}$ is the gravitational force on Gyrostat in the body-fixed frame, $\underline{T}_i$ is the torque on Momentum wheel i, $\underline{F}_{gQ}$ is the gravitational force on point mass Q, $\underline{F}_{Th}$ is the thrust force on Gyrostat applied at point F, and $\underline{M}_{gG}$ is the gravitational

torque on Gyrostat and $\underline{T}_{fP}$ is the linear viscous torque model on pendulum P.

$$\underline{T}_{fP} = (-C_2 l_P^2 u_{11} s\psi_1, C_2 l_P^2 u_{11} c\psi_1, -C_1 l_P^2 u_{10} s^2\psi_2) \tag{2.61}$$

The partial velocity ${}^N\underline{V}_r^F$ of the thrust application point F are calculated and shown in Table 2.2.

Table 2.2: Partial Velocity ${}^N\underline{v}_r^F$

r	${}^N\underline{v}_r^F$
1	$\underline{b}_1$
2	$\underline{b}_2$
3	$\underline{b}_3$
4	$-l_{F_3}\underline{b}_2 + l_{F_2}\underline{b}_3$
5	$l_{F_3}\underline{b}_1 - l_{F_1}\underline{b}_3$
6	$-l_{F_2}\underline{b}_1 + l_{F_1}\underline{b}_2$
7	0
8	0
9	0
10	0
11	0
12	0

The generalized active force can be written from Eq. (2.60) in the following vector

form

$$\{F\} = \begin{bmatrix} F_{gG_1} + F_{gP_1} + F_{gQ_1} + F_1 \\ F_{gG_2} + F_{gP_2} + F_{gQ_2} + F_2 \\ F_{gG_3} + F_{gP_3} + F_{gQ_3} + F_3 \\ F_4 \\ F_5 \\ F_6 \\ T_1 \\ T_2 \\ T_3 \\ l_P(F_{gP_2}c\psi_1 - F_{gP_1}s\psi_1)s\psi_2 \\ -(z_0 - l_P c\psi_2)(F_{gP_1}c\psi_1 + F_{gP_2}s\psi_1) + F_{gP_3}l_P s\psi_2 \\ -kq(t) - c\dot{q}(t) + F_{gQ_1} \end{bmatrix} \tag{2.62}$$

Where

$$F_4 = M_{gG_1} + F_{gP_2}l_P c\psi_2 - l_{F_3}F_2 + l_{F_2}F_3 + F_{gP_3}l_P s\psi_1 s\psi_2 - F_{gP_2}z_0 - F_{gQ_2}z_Q - C_2 l_P^2 s\psi_1 u_{11} \tag{2.63}$$

$$F_5 = M_{gG_2} - F_{gP_1}(l_P c\psi_2 - z_0) + l_{F_3}F_1 - l_{F_1}F_3 - F_{gP_3}l_P c\psi_1 s\psi_2 + F_{gQ_1}z_Q + C_2 l_P^2 c\psi_1 u_{11} - F_{gQ_3}q(t) \tag{2.64}$$

$$F_6 = M_{gG_3} + F_{gP_2}l_P c\psi_1 s\psi_2 - l_{F_2}F_1 + l_{F_1}F_2 - F_{gP_1}l_P s\psi_1 s\psi_2 - C_1 l_P^2 s^2\psi_2 u_{10} + F_{gQ_2}q(t) \tag{2.65}$$

2.4 Euler angles and Equation of Motion

We can show that generalized inertia forces, $-F_r^*$, can be written in matrix form as:

$$\{-F^*\} = [M]\{\dot{U}\} + \{C\} \tag{2.66}$$

where

$$\{\dot{U}\} = [\dot{u}_1, \dot{u}_2, ..., \dot{u}_{12}]^T \tag{2.67}$$

and

$$\{C\} = [c_1, c_2, ..., c_{12}]^T \tag{2.68}$$

The elements of vector C are

$$c_1 = m_B Z_1 + m_P Z_4 + m_Q Z_7 \tag{2.69}$$

$$c_2 = m_B Z_2 + m_P Z_5 + m_Q Z_8 \tag{2.70}$$

$$c_3 = m_B Z_3 + m_P Z_6 + m_Q Z_9 \tag{2.71}$$

$$c_4 = -I_w u_6 u_8 + (I_3 - I_2) u_6 u_5 + I_w u_5 u_9 + m_P(-z_o + l_P c\psi_2) Z_5 + m_P l_P s\psi_1 s\psi_2 Z_6 - m_Q z_Q Z_8 \tag{2.72}$$

$$\begin{aligned} c_5 = {} & I_w u_6 u_7 + u_4 u_6 (I_1 - I_3) - I_w u_4 u_9 - m_P(l_P c\psi_2 + z_o) Z_4 - m_P l_P c\psi_1 s\psi_2 Z_6 \\ & + m_Q z_Q Z_7 - m_Q q(t) Z_9 \end{aligned} \tag{2.73}$$

$$c_6 = -I_w u_5 u_7 + u_4 u_5 (I_2 - I_1) + I_w u_8 u_4 - m_P l_P s\psi_1 s\psi_2 Z_4 + m_P l_P s\psi_2 c\psi_1 Z_5 + m_Q q(t) Z_8 \tag{2.74}$$

$$c_7 = c_8 = c_9 = 0 \tag{2.75}$$

$$c_{10} = -m_P l_P s\psi_2 s\psi_1 Z_4 + m_P l_P s\psi_2 c\psi_1 Z_5 \tag{2.76}$$

$$c_{11} = m_P l_P c\psi_2 c\psi_1 Z_4 + m_P l_P c\psi_2 s\psi_1 Z_5 + m_P l_P s\psi_2 c\psi_2 Z_6 \tag{2.77}$$

$$c_{12} = m_Q Z_7 \tag{2.78}$$

We notice that the matrix $[M]$ is a symmetric matrix with the lower half written explicitly as

$$[M] = \begin{bmatrix}
m_{1,1} & & & & & & & & & & & \\
0 & m_{2,2} & & & & & & & & & & \\
0 & 0 & m_{3,3} & & & & & & & & & \\
0 & -m_{5,1} & m_{4,3} & m_{4,4} & & & & & & & & \\
m_{5,1} & 0 & m_{5,3} & m_{5,4} & m_{5,5} & & & & & & & \\
m_{6,1} & -m_{10,1} & 0 & m_{6,4} & m_{6,5} & m_{6,6} & & & & & & \\
0 & 0 & 0 & I_w & 0 & 0 & I_w & & & & & \\
0 & 0 & 0 & 0 & I_w & 0 & 0 & I_w & & & & \\
0 & 0 & 0 & 0 & 0 & I_w & 0 & 0 & I_w & & & \\
m_{10,1} & m_{10,2} & 0 & m_{10,4} & m_{10,5} & m_{10,6} & 0 & 0 & 0 & m_{1010} & & \\
m_{11,1} & m_{11,2} & m_{11,3} & m_{11,4} & m_{11,5} & 0 & 0 & 0 & 0 & 0 & m_{1111} & \\
m_Q & 0 & 0 & 0 & m_Q z_Q & 0 & 0 & 0 & 0 & 0 & 0 & m_Q
\end{bmatrix} \tag{2.79}$$

where the nonzero elements of matrix $[M]$ are:

$$m_{1,1} = m_{2,2} = m_{3,3} = m_B + m_P + m_Q \tag{2.80}$$

$$m_{5,1} = m_P z_o - m_P l_P c\psi_2 + m_Q z_Q \tag{2.81}$$

$$m_{6,1} = -m_P l_P s\psi_1 s\psi_2 \tag{2.82}$$

$$m_{10,1} = -m_P l_P s\psi_1 s\psi_2 \tag{2.83}$$

$$m_{11,1} = m_P l_P c\psi_1 c\psi_2 - m_P z_0 c\psi_1 \tag{2.84}$$

$$m_{11,2} = m_P l_P c\psi_2 s\psi_1 - m_P z_o s\psi_1 \tag{2.85}$$

$$m_{4,3} = m_P l_P s\psi_1 s\psi_2 \tag{2.86}$$

$$m_{11,3} = m_P l_P s\psi_2 \tag{2.87}$$

$$m_{4,4} = I_1 + I_w + m_Q z_Q^2 + m_P l_P^2 c\psi_2^2 + m_P l_P^2 s\psi_1^2 s\psi_2^2 + m_P z_0(z_0 - 2l_P c\psi_2) \tag{2.88}$$

$$m_{5,4} = -m_P l_P^2 c\psi_1 s\psi_1 s\psi_2^2 \tag{2.89}$$

$$m_{6,4} = m_P l_P^2 c\psi_1 c\psi_2 s\psi_2 - l_P m_P z_0 c\psi_1 s\psi_2 - m_Q z_Q q \tag{2.90}$$

$$m_{10,4} = m_P l_P^2 c\psi_1 c\psi_2 s\psi_2 - l_P m_p z_0 c\psi_1 s\psi_2 \tag{2.91}$$

$$m_{11,4} = m_P l_P^2 s\psi_1 + m_P z_0(z_0 - 2l_P c\psi_2) s\psi_1 \tag{2.92}$$

$$m_{5,3} = -l_P m_P c\psi_1 s\psi_2 - m_Q q \tag{2.93}$$

$$m_{10,2} = l_P m_P c\psi_1 s\psi_2 \tag{2.94}$$

$$m_{5,5} = I_2 + I_w + m_P l_P^2 c\psi_2^2 + m_Q z_Q^2 + m_P l_P^2 c\psi_1^2 s\psi_2^2 + m_p z_0(z_0 - 2l_p c\psi_2) + m_Q q^2 \tag{2.95}$$

$$m_{6,5} = m_{10,5} = m_P l_P^2 c\psi_2 s\psi_1 s\psi_2 - l_P m_P z_0 s\psi_1 s\psi_2 \tag{2.96}$$

$$m_{11,5} = -m_P l_P^2 c\psi_1 - m_P z_0 c\psi_1(z_0 - 2l_P c\psi_2) \tag{2.97}$$

$$m_{6,6} = I_3 + I_w + m_Q q^2 + m_P l_P^2 s\psi_2^2 \tag{2.98}$$

$$m_{10,6} = m_P l_P^2 s\psi_2^2 \tag{2.99}$$

$$m_{10,10} = m_P l_P^2 s\psi_2^2 \tag{2.100}$$

$$m_{11,11} = m_P l_P^2 + m_P(z_0^2 - 2z_0 l_P c\psi_2) \tag{2.101}$$

Thus, the equations of motion for the gyrostat with spherical pendulum and a mass-spring-damper system has been obtained as:

$$[M]\{\dot{U}\} + \{C\} = \{F\} \tag{2.102}$$

or

$$\{\dot{U}\} = [M]^{-1}(\{F\} - \{C\}) \tag{2.103}$$

Equation (2.103) consist of twelve highly nonlinear ordinary differential equations. We use Euler angles yaw, pitch, roll (ψ, θ, ϕ) to describe the orientation of the gyrostat in inertial frame, N. The kinematic equations are:[11]

$$\dot{\psi} = \frac{1}{c\theta}(u_5 s\varphi + u_6 c\varphi) \tag{2.104}$$

$$\dot{\theta} = u_5 c\varphi - u_6 s\varphi \tag{2.105}$$

$$\dot{\varphi} = \frac{1}{c\theta}(u_5 s\theta s\varphi + u_6 s\theta c\varphi) + u_4 \tag{2.106}$$

where c and s denote the cosine and sine functions, respectively. The set of Eq. (2.103) along with the kinematic equations, Eq. (2.104)–Eq. (2.106), constitute a mathematical model of the system.

2.5 Model of Gyrostat with Planar Pendulum

Another slosh model in the literature proposed for describing the coning motion of spinning spacecraft during thrusting maneuver is a planar pendulum.[19] We define new generalized speeds for a flexible, planar pendulum as follows, where ψ_1 is the

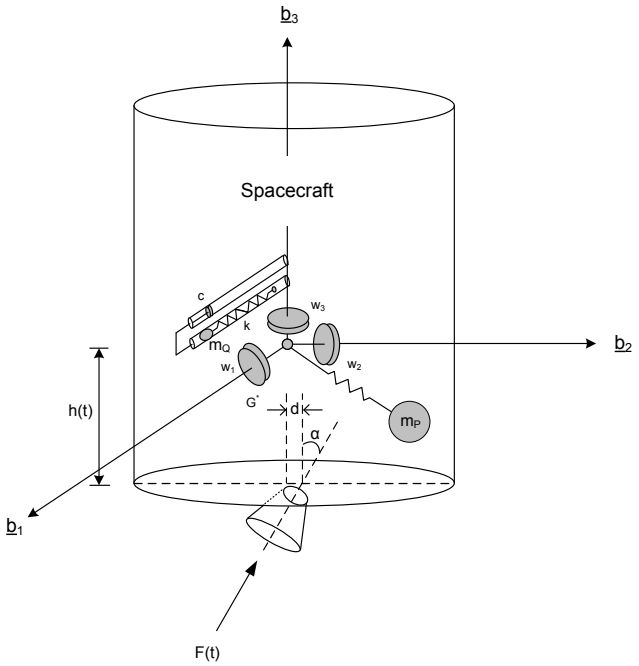


Fig. 2.4: A gyrostat model with planar pendulum.

angle of the pendulum form $\underline{b}_1$, and l_P is the length of this flexible pendulum:

$$u_{10} = \dot{\psi}_1 \tag{2.107}$$

$$u_{11} = \dot{l}_P \tag{2.108}$$

The pendulum force on the gyrostat can be written in the body-fixed frame as:

$$\underline{F}_P = -k_P l_P cos\psi_1 \underline{b}_1 - k_P l_P sin\psi_1 \underline{b}_y \tag{2.109}$$

The equation of motion for the gyrostat with a planar pendulum can be obtained as:

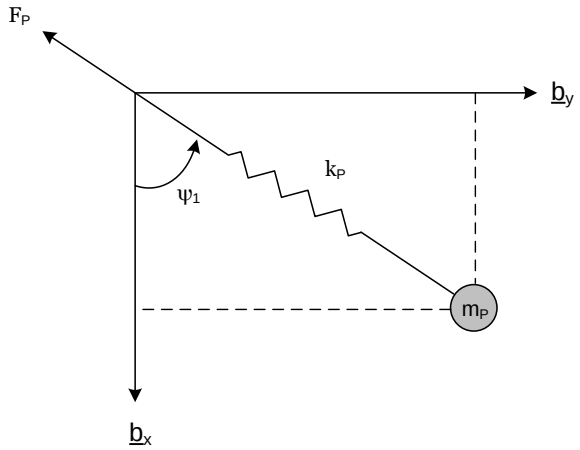


Fig. 2.5: planar pendulum.

Negative of the generalized inertia force is given by:[14]

$$-F_r^* = (\underline{\underline{I}}^{G^*}.{}^N\underline{\alpha}^G + {}^N\underline{\omega}^G \times \underline{\underline{I}}^{G^*}.{}^N\underline{\omega}^G).{}^N\underline{\omega}_r^G + \sum_{i=1}^{3}(\underline{\underline{I}}^{w_i^*}.{}^N\underline{\alpha}^{w_i} + {}^N\underline{\omega}^{w_i} \times \underline{\underline{I}}^{w_i^*}.{}^N\underline{\omega}^{w_i}).{}^N\underline{\omega}_r^{w_i}$$
$$+ m_G\ {}^N\underline{a}^{G^*}.{}^N\underline{V}_r^{G^*} + m_P\ {}^N\underline{a}^p.{}^N\underline{V}_r^p \qquad ; \qquad (r = 1, ..., 11) \tag{2.110}$$

The Generalized Active Force in this case, F_r is written as:[14]

$$\underline{F}_r = \underline{F}_{gG}.{}^N\underline{V}_r^{G^*} + \sum_{i=1}^{3}\underline{T}_i.{}^G\underline{\omega}_r^{w_i} + \underline{F}_{gP}.{}^N\underline{V}_r^P - \sum_{i=1}^{3}\underline{T}_i.{}^N\underline{\omega}_r^G$$
$$+ \underline{F}_{Th}.{}^N\underline{V}_r^F + \underline{M}_{gG}.{}^N\underline{\omega}_r^G + \underline{F}_P.{}^N\underline{v}_r^P \qquad ; \qquad (r = 1, \ldots, 11) \tag{2.111}$$

where the spring stiffness, k_P, is given by:[22]

$$k_P = \gamma^2 m_P(1-\mu)\Omega^2 \tag{2.112}$$

and

$$\mu = \frac{m_P.m_B(t)}{m_P + m_B(t)} \tag{2.113}$$

In Eq. (2.113), $m_B(t)$ is time-varying gyrostat mass, m_P is pendulum mass, Ω is the gyrostat spin rate, and γ is the stiffness constant.

Putting Eq. (2.110), Eq. (2.111) in Eq. (2.54) leads to a set of equations in the form of Eq. (2.103), with a new set of matrices, M, F, C, U.

Chapter 3

Velocity Pointing Error

Errors in proper orientation of a desired velocity vector of a thrusting, spinning spacecraft are caused by thrust misalignment of various kind. Equations of motion of the system were derived in the previous chapter using Kane's method accounting time-vary moment of inertia, torque and mass with planar and spherical pendulums models of fuel slosh that are the most accurate mechanical equivalent models available in the literature.

We described the velocities transformation matrix from body-fixed frame to inertial frame in the following. The next part is the equations and relations used to present a new optimized trapezoidal thrust scheme in order to reduce the average velocity pointing error in the proposed spacecraft model.

3.1 Velocities Transformation

Velocities in body fixed frame, v_x, v_y, v_z can be obtained by solving the equations of motion and can be written in inertial space using transformation matrix between orbital frame and body fixed frame as following. The absolute velocity vector of the gyrostat can be transformed from body-fixed frame to inertial frame by

$$v_x \underline{n}_x + v_y \underline{n}_y + v_z \underline{n}_z = {}^N R^B_{321} . {}^N \underline{V}^{G^*} \tag{3.1}$$

where the the transformation matrix for a 3-2-1 Euler angle sequence from body

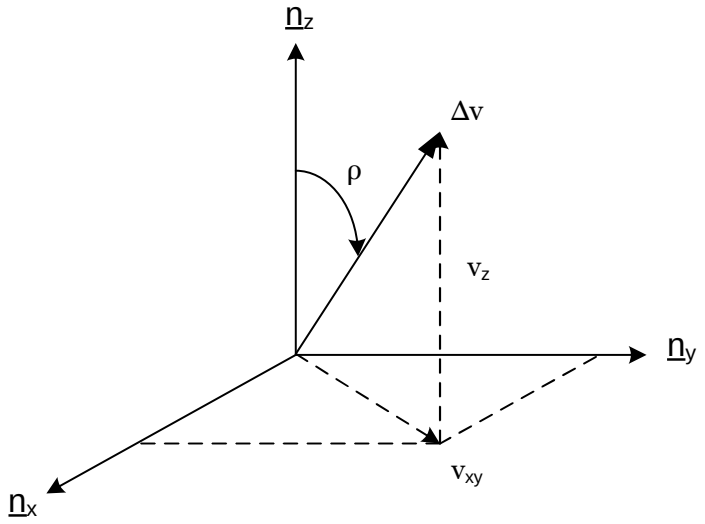


Fig. 3.1: Velocity pointing error.

to inertial frames is given by

$$ {}^{N}R^{B}_{321} = \begin{bmatrix} c\theta c\psi & c\psi s\theta s\phi - c\phi s\psi & c\phi c\psi s\theta + s\phi s\psi \\ c\theta s\psi & c\phi c\psi + s\theta s\phi s\psi & -c\psi s\phi + c\phi s\theta s\psi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{3.2} $$

It has been shown that we can define the velocity pointing error angles, γ_x, γ_y using the transverse velocities, ΔV_x, ΔV_y:[13]

$$ tan\gamma_x = \Delta V_x / \Delta V_z \tag{3.3} $$

$$ tan\gamma_y = \Delta V_y / \Delta V_z \tag{3.4} $$

We can obtain the following velocity pointing error equation as shown in Fig. 3.1:[13]

$$ \rho(t) = tan^{-1}\left(\frac{v_{xy}}{v_z}\right) = tan^{-1}\left(\frac{\sqrt{v_x^2 + v_y^2}}{v_z}\right) \tag{3.5} $$

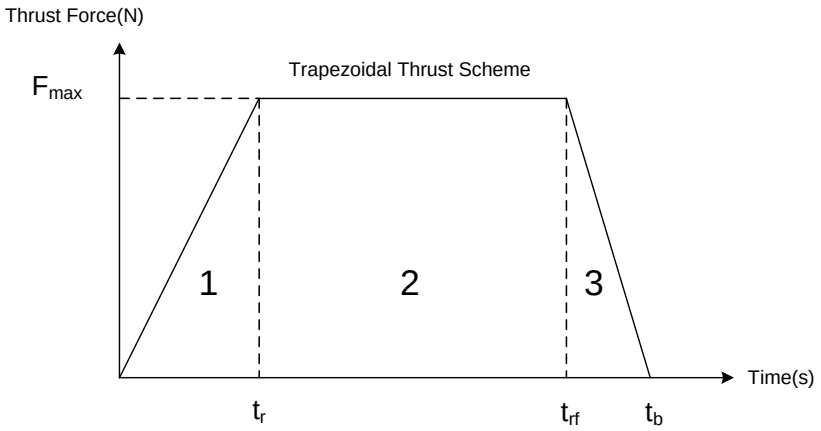


Fig. 3.2: Trapezoidal thrust scheme.

we can obtain the following equation for average velocity pointing error:

$$AVPE = \frac{1}{b-a}\int_a^b \rho(t)dt = \frac{1}{85.3}\int_0^{85.3} \rho(t)dt \tag{3.6}$$

3.2 Optimizing the Trapezoidal Thrust Scheme

We can use different thrust profiles in order to model the thrusting spacecraft. Star 48B thrust force used to confirm our model with real flight data which describes in numerical simulation part and shows the accuracy of our job. We tried to reduce the average velocity pointing error using another thrust scheme which Longuski[13] presented as shown in Fig. 3.2 and considered the ramp up and ramp down times to be equal to each other. It is vital to consider the maximum thrust, the burn time and total impulse remain constant between the two comparisons.[13]

We presented a optimized trapezoidal thrust scheme and considered a variable ramp up and ramp down times to get the least average velocity pointing error in respect of the ramp up time. Now consider the general trapezoidal trust profile shown in Fig. 3.2. In the step, we optimize the trapezoidal thrust scheme by finding the optimal

ramp up time, t_r^*, which minimizes AVPE subject to the following constraints:

$$Total\ Impulse = \mathcal{I} = \int_0^{t_b} F(t)dt \tag{3.7}$$

We calculate the total impulse for both trapezoidal thrust scheme presented by Longuski[13] and the optimized trapezoidal thrust scheme and put these two equal to each other in order to the total impulse remain constant because the PAM designed to tolerate a certain maximum chamber pressure.

$$\mathcal{I} = \frac{t_r \times F_{max}}{2} + (t_{rf} - t_r) \times F_{max} + \frac{t_r \times F_{max}}{2} \tag{3.8}$$

Simplifying the above equation we would have:

$$t_{rf} = t_r + 64.73 \tag{3.9}$$

and as shown in Fig. 3.2:

$$t_{rf} \leq t_b \tag{3.10}$$

So

$$t_r \leq t_b - 64.73 \tag{3.11}$$

Chapter 4

Numerical Simulation

We use MATHEMATICA to carry out and integrate the equations of motion and the kinematic equations. First, we considered the spacecraft with three momentum wheels, point mass, spring, nutation damper with spherical pendulum and constant mass and moment of inertia to examine the attitude motion of the spacecraft and the nutation damper effect on the body angular velocities and pitch angle. The results have discussed in the attitude motion simulation part.

Then, the simulation run out for a thrusting, spinning spacecraft with three momentum wheels, planar and spherical pendulums with time-vary mass and moment of inertia considering various kind of thrust misalignments in order to find the velocity pointing error of each case. Trapezoidal thrust scheme presented in this part, shows we can reduce the average velocity pointing error using this thrust profile instead of the Star 48B thrust profile.

Finally, we tried to optimize the trapezoidal thrust scheme in order to get the least average velocity pointing error for the spacecraft. Numerical simulation results presented in this section show the accuracy of our job.

4.1 Attitude Motion of the Spacecraft

The spacecraft properties and initial conditions are as follows:[14,16]

Table 4.1: Partial list of parameters

m_B	5274.4 kg
m_P	0.1 m_B
m_Q	0.01 m_B
l_P	0.15 m
z_0	0
z_Q	1 m
C_1,C_2	22.2
c	105 N.s/m
k	52 N/m

The inertia dyadic of gyrostat and momentum wheels are, respectively,

$$\underline{\underline{I}}^{G^*} = \begin{bmatrix} 1402.4 & 0 & 0 \\ 0 & 1292.5 & -20 \\ 0 & -20 & 1375.7 \end{bmatrix} \; kg.m^2 \tag{4.1}$$

and

$$\underline{\underline{I}}^{w_i^*} = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.17 \end{bmatrix} \; kg.m^2 \quad , \quad (r = 1, 2, 3) \tag{4.2}$$

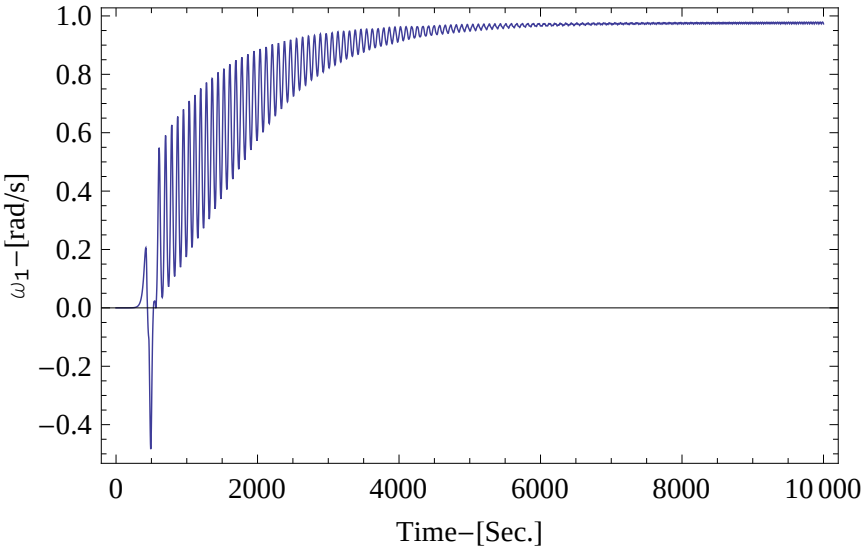
The initial conditions are assumed to be $\theta(0) = \varphi(0) = \psi(0) = \dot{\psi}_1(0) = \dot{\psi}_2(0) = 0^\circ$, and $\psi_1(0) = \psi_2(0) = \pi/180^\circ$, $q(0) = \dot{q}(0) = 0.01m$. In the following, we study the attitude motion of the spacecraft in two cases: flat spin and reorientation maneuvers.

4.1.1 Case1-Flat Spin Maneuver:

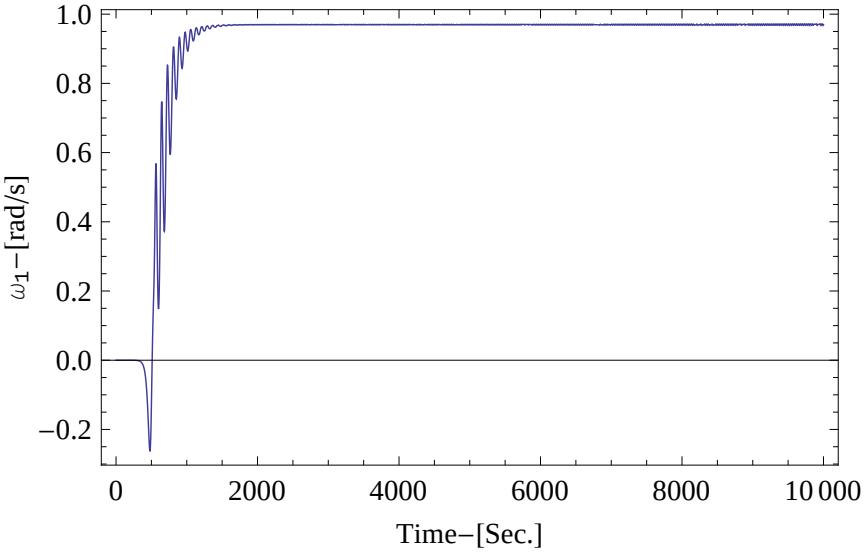
We assume the spacecraft has an initial spin rate of 20 rpm around its minimum moment of inertia, $\underline{b}_2$ axis, and zero about other axes. i.e. $\omega_0 = (0, 20, 0)\ rpm$. As

we expect, due to internal energy dissipation, the spacecraft reorient and spin about its maximum moment of inertia $\underline{b}_1$ axis. The time history of angular velocities and the nutation angle of gyrostat with and without nutation damper are shown in Figs. 4.1, 4.2, 4.3, 4.4, respectively.

It can seen that the nutation damper decreases the response time and reduces the oscillations.

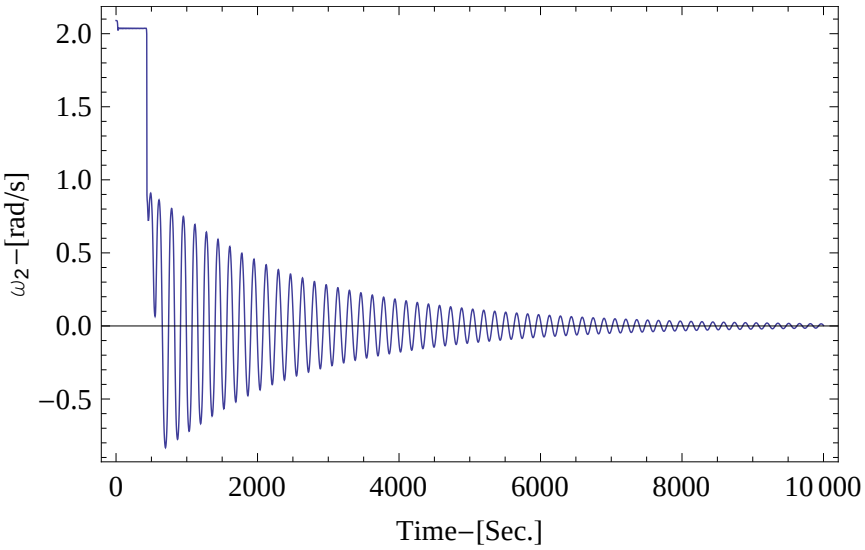


(a) Without nutation damper

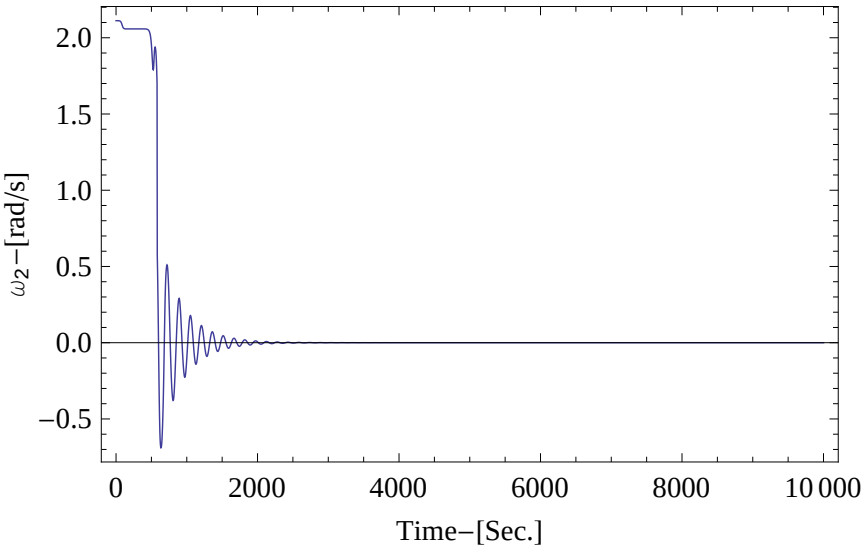


(b) With nutation damper

Fig. 4.1: Body angular velocity, $\omega_1(t)$, for case 1.

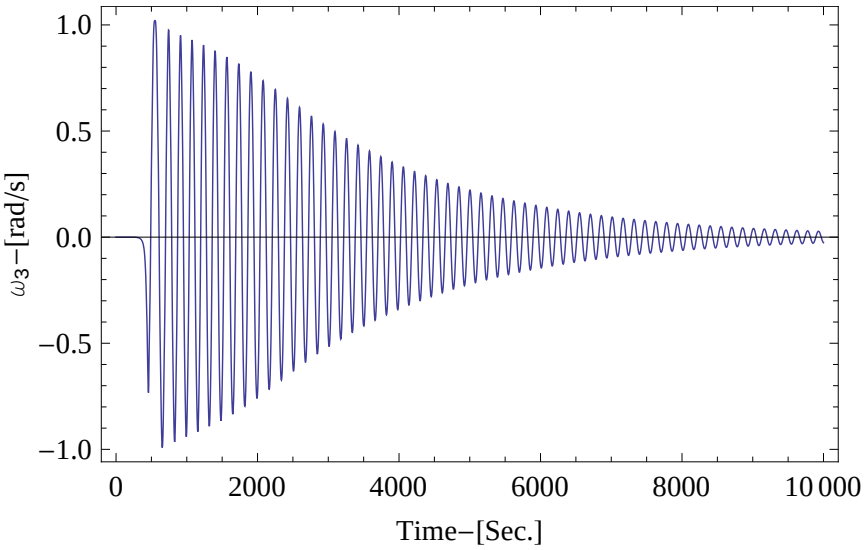


(a) Without nutation damper

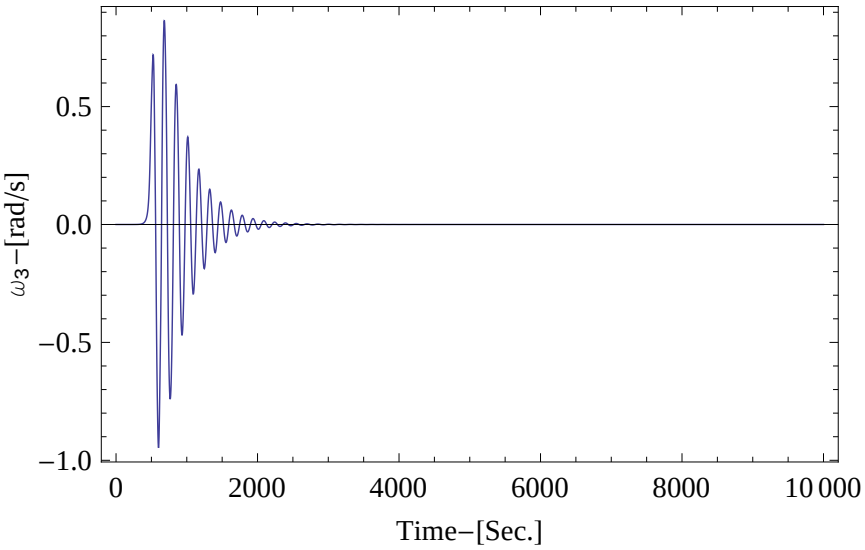


(b) With nutation damper

Fig. 4.2: Body angular velocity, $\omega_2(t)$, for case 1.



(a) Without nutation damper



(b) With nutation damper

Fig. 4.3: Body angular velocity, $\omega_3(t)$, for case 1.

4.1.2 Case 2-Reorientation Maneuver:

In this case, the spacecraft has an initial spin rate of 5 rpm around its maximum moment of inertia, $\underline{b}_1$ axis, and zero around other axes. i.e. $\omega_0 = (5, 0, 0)\ rpm$. Moreover, we assume the third momentum wheel, along the body $\underline{b}_3$ axis, spin up according to the following piecewise function:[16]

$$^{G^*}\omega^{w_3} = \begin{cases} 0 & 0 \leq t < 1,000 \\ 0.9375(t - 1000) & 1,000 \leq t < 7,400 \\ 6,000 & 7,400 \leq t \leq 10,000 \end{cases} \tag{4.3}$$

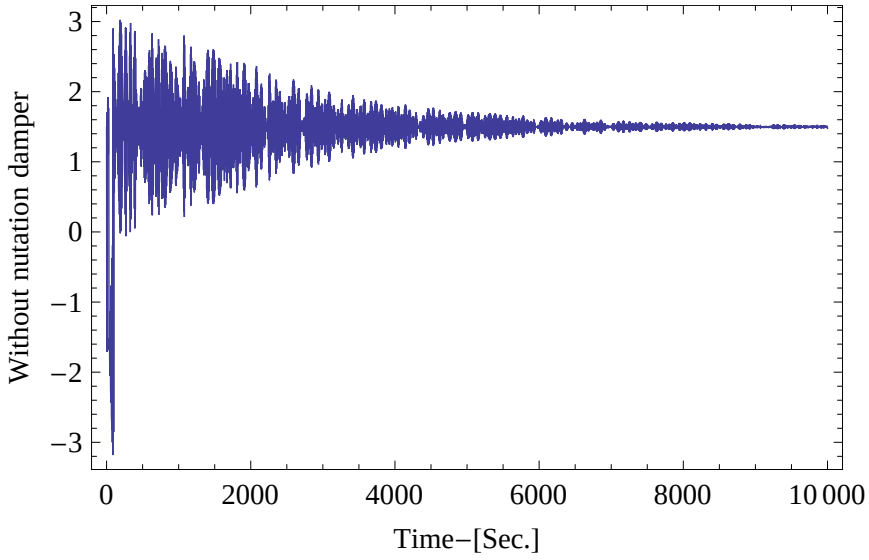
The angular velocity components of the gyrostat are shown in Fig. 4.5(a). It can be seen that the spacecraft originally spins around the maximum moment of inertia and then gradually it reorients and spins around the $\underline{b}_3$ axis.

Fig. 4.5(b) shows the pitch angle of the gyrostat. As we predict, the pitch angle starts at zero and goes to 1.57 rad (90 degrees). We also test our model for the case reported by Kang.[16] The comparison shows that the results are in agreement.

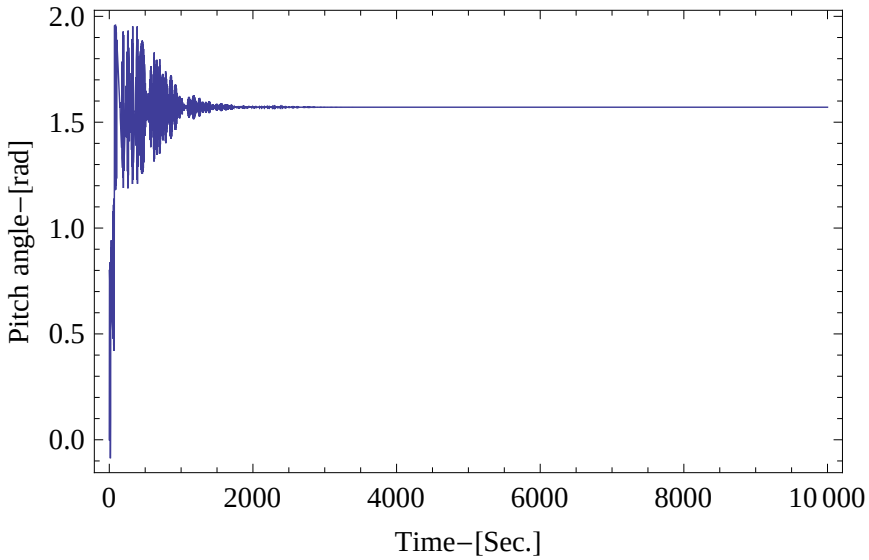
4.2 Annihilation of Velocity Pointing Error

In this section, we plotted the velocity pointing error of the spacecraft with planar and spherical and two different thrust forces. Compression between the trapezoidal thrust scheme and STAR 48B thrust profile shows that using the trapezoidal thrust scheme; we are able to reduce the average velocity pointing error in the proposed spacecraft.

In the next step, we tried to optimize the trapezoidal thrust scheme in order to

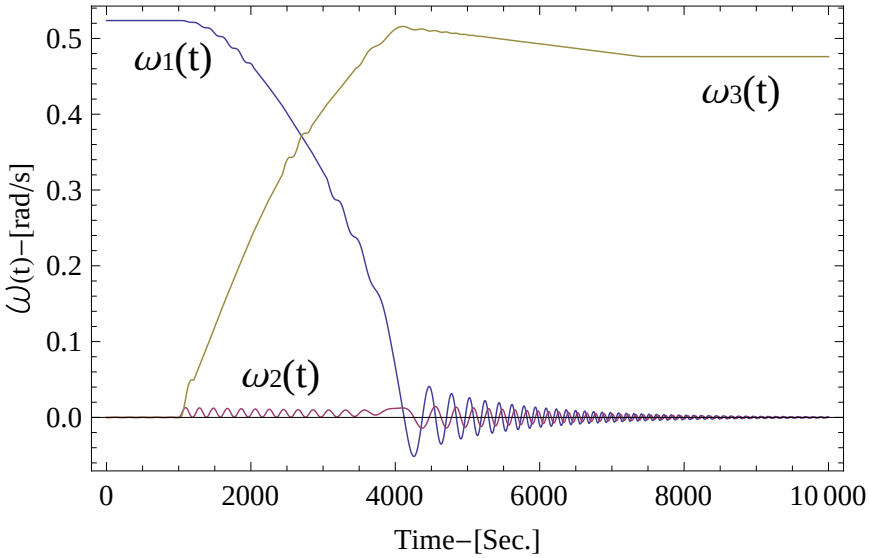


(a) Without nutation damper

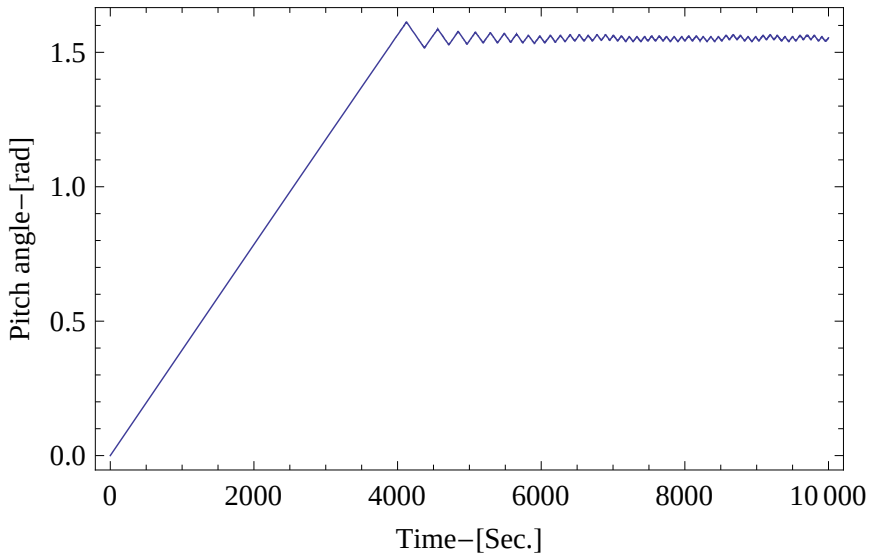


(b) With nutation damper

Fig. 4.4: Gyrostat pitch angle for case 1.



(a) Angular velocities



(b) Pitch angle

Fig. 4.5: Body angular velocities and pitch angle for case 2.

Table 4.2: Spacecraft and PAM-D Data For for Numerical Simulation

Property	Notation	Quantity
Initial PMOI about x axis	I_{x_0}	858 $Kg.m^2$
Final PMOI about x axis	I_{x_f}	222 $Kg.m^2$
Initial PMOI about y axis	I_{y_0}	858 $Kg.m^2$
Final PMOI about y axis	I_{y_f}	222 $Kg.m^2$
Initial PMOI about z axis	I_{z_0}	401 $Kg.m^2$
Final PMOI about z axis	I_{z_f}	102 $Kg.m^2$
Spherical Pendulum torque constant	C_S	2.4 $N.s/m$
Planar Pendulum stiffness constant	γ	0.5
Initial spacecraft mass	m_B	2500 Kg
Mass rate flow	$\dot{m}$	-24 Kg/s
Pendulum mass	m_P	250 Kg
Spherical pendulum length	l_P	0.3 m
Maximum COM offset	d	0.02 m
Thrust misalignment angle	α	0.25 deg
Initial distance from throat to COM	h_0	0.8 m
Final distance from throat to COM	h_f	1.55 m
Initial spin rate	Ω_0	70 rpm

achieve the least average velocity pointing error for both planar and spherical pendulums.

The mass properties and geometrical data[10,13] of the Ulysses spacecraft with STAR 48B engine are summarized in Table 4.2.

4.2.1 Average Velocity Pointing Error

We examined our model for the special case which is studied by Javorsek and Longuski[13] with the given thrust profiles which are shown in Figs. 4.6(a) and 4.6(b). The results are in agreement with the results reported in Ref.[13] Then, we proposed and test the hypothesis which is: "The trapezoidal thrust scheme given in Fig. 4.6(b) is effective for a thrusting, spinning spacecraft with internal moving part." Numerical simulation proves the hypothesis to be correct. Velocity pointing error for a simple rigid body, rigid body with a planar pendulum, and rigid body with spherical pendulum with STAR 48B thrust profile are presented in the Figs. 4.7(a), 4.7(b), 4.7(c) respectively.

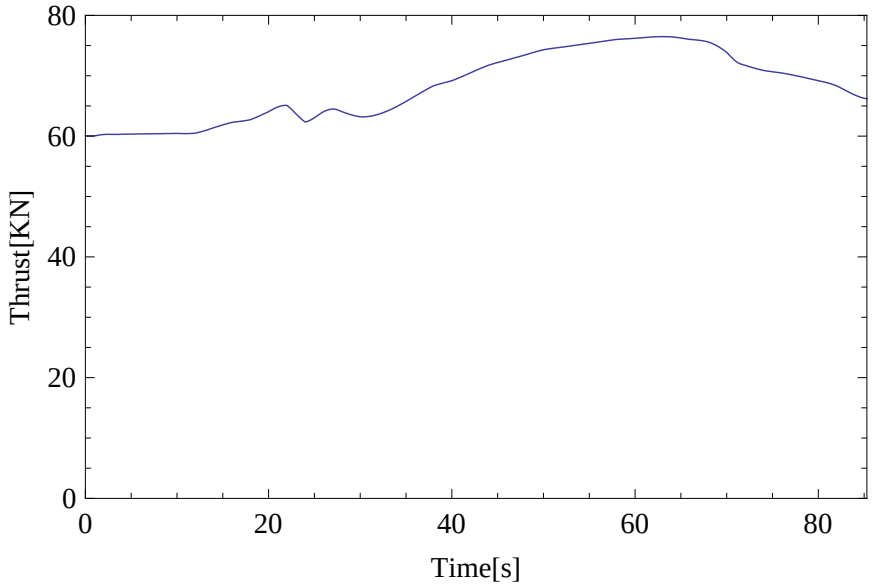
Velocity pointing error for a simple rigid body, rigid body with a planar pendulum, and rigid body with spherical pendulum with trapezoidal thrust scheme are presented in the Figs. 4.8(a), 4.8(b), 4.8(c) respectively. The average velocity pointing error which is defined for the thrust duration as

$$AVPE = \frac{1}{t_b}\int_0^{t_b} \rho(t)dt = \frac{1}{85.3}\int_0^{85.3} \rho(t)dt \tag{4.4}$$

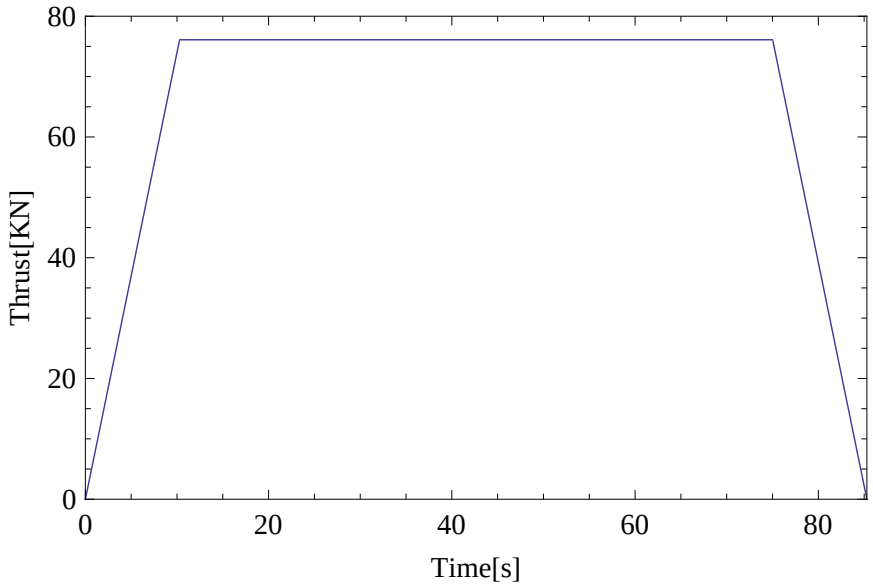
are listed in Table 4.3 for real(STAR 48B) and Trapezoidal thrust profiles with planar and spherical pendulum models. The results show that the AVPE for gyrostat with planar and spherical pendulum models could reduce by about 98%.

4.2.2 Optimization of Trapezoidal Thrust Scheme

Now consider the general trapezoidal trust profile shown in Fig. 4.6(b). In this step, we optimize the trapezoidal thrust scheme by finding the optimal rise time,

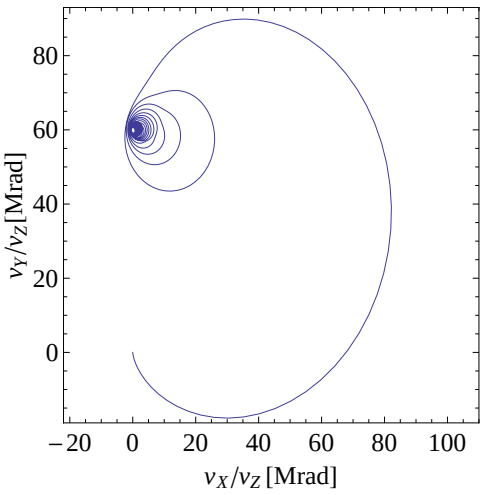


(a) Star 48B thrust profile.

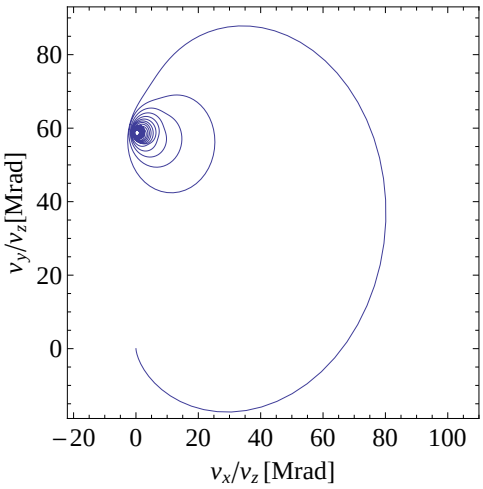


(b) Trapezoidal

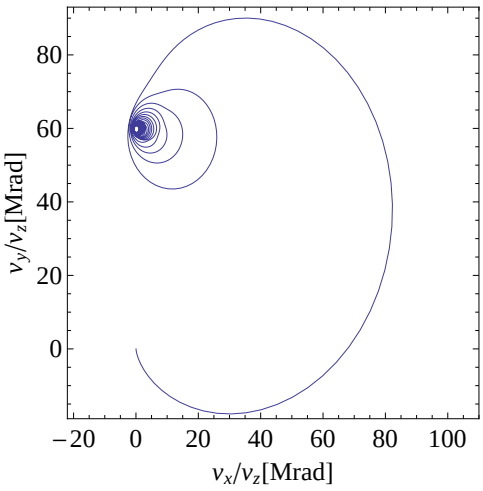
Fig. 4.6: Thrust profile



(a) Rigid body



(b) Rigid body with a planar pendulum



(c) Rigid body with a spherical pendulum

Fig. 4.7: Velocity pointing error for Star 48B thrust.

Table 4.3: Results summery

Thrust profile	cases	Average Velocity pointing error
STAR 48B	Rigid body	60.17 $Mrad$
	Rigid body with planner pendulum	60.85 $Mrad$
	Rigid body with spherical pendulum	61.32 $Mrad$
Trapezoidal	Rigid body	0.811 $Mrad$
	Rigid body with planner pendulum	1.008 $Mrad$
	Rigid body with spherical pendulum	1.442 $Mrad$

t_r^*, which minimizes AVPE subject to the following constraints:

$$t_{rf} = t_r + 64.73 \tag{4.5}$$

and as shown in Fig. 4.6(b):

$$t_{rf} \leq t_b \tag{4.6}$$

So

$$t_r \leq t_b - 64.73 \tag{4.7}$$

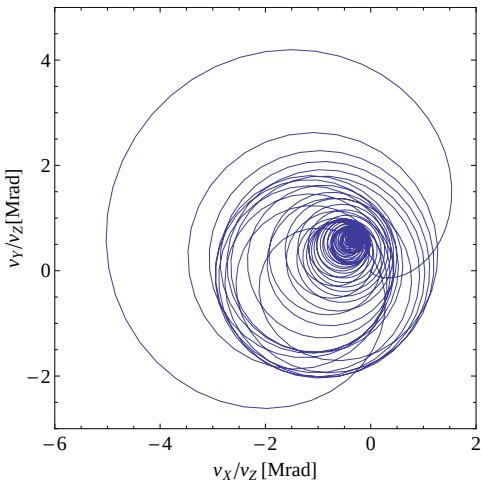
Here the burn time, the maximum thrust force, and the total impulse for STAR 48B engine are $85.3s$, 76,100 N, and $5.71 \times 10^6 N.s$, respectively. We plotted the AVPE versus ramp up time for planar and spherical pendulums as shown in Figs. 4.9(a), 4.9(b) respectively and noticed that the function is a multi-modal function, i.e., it has more than one minimum point. The optimal ramp up times and the associated velocity pointing errors for gyrostat with planar and spherical pendulum models are listed in Table 4.4. It can be seen easily from Table 4.4 that the global optimal point for planar and spherical pendulums are 17.76 and 18.65 seconds, respectively. We confirmed the 60 $mrad$ for the average velocity pointing error for the STAR 48B thrust profile, but using the trapezoidal thrust scheme, We are able to get same error with lower spin rate. In fact, with trapezoidal thrust scheme we would just

Table 4.4: The Optimal Ramp up Time and Associated AVPE.

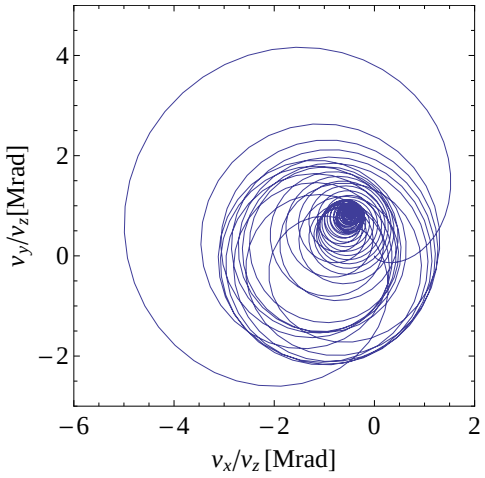
Pendulum	$t_r^*[s]$	AVPE $[Mrad]$
Planar	17.76	0.5251
	16.91	0.5321
	14.38	0.5496
	10.15	0.5971
Spherical	18.65	0.2918
	17.99	0.3162
	14.91	0.3411
	10.03	0.4267

need to spin up the spacecraft at 25 $mrad$ instead of the 70$mrad$ used for the STAR 48B thrust profile to get the same velocity pointing error.

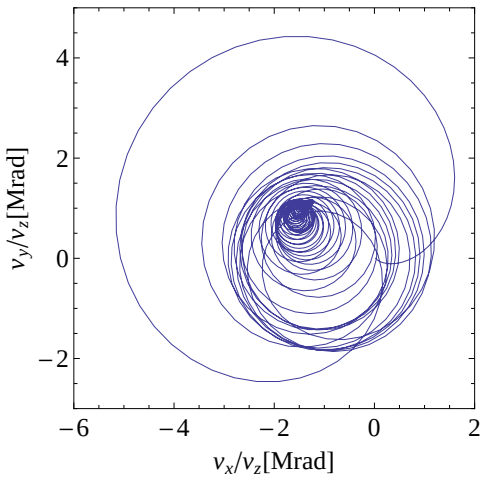
Finally, we plotted the average velocity pointing error in respect of the spin rate using the optimized trapezoidal thrust scheme for both planar and spherical pendulums as shown in Figs. 4.10(a), 4.10(b) respectively. We considered the ramp up times for planar and spherical pendulums to be 17.76 s and 18.65 s respectively as summarized before in Table 4.4.



(a) Rigid body

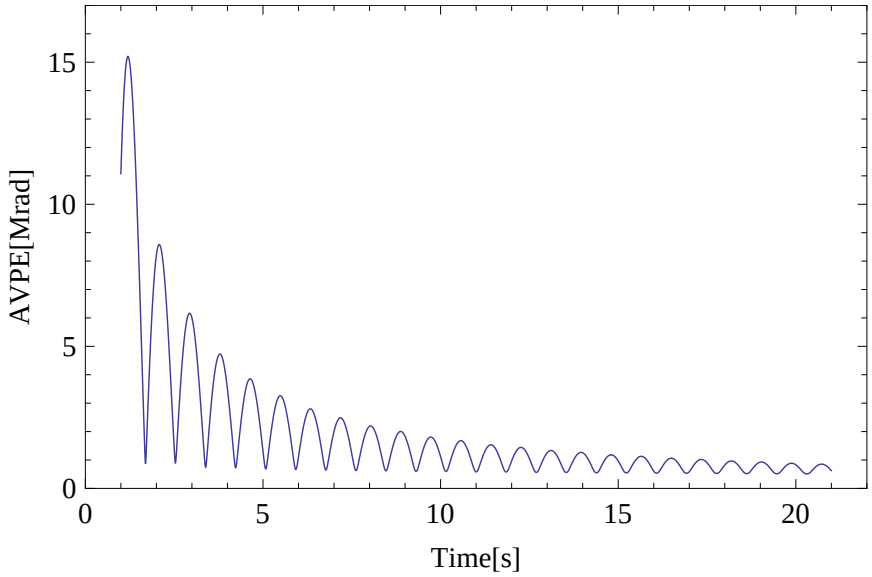


(b) Rigid body with a planar pebdulum

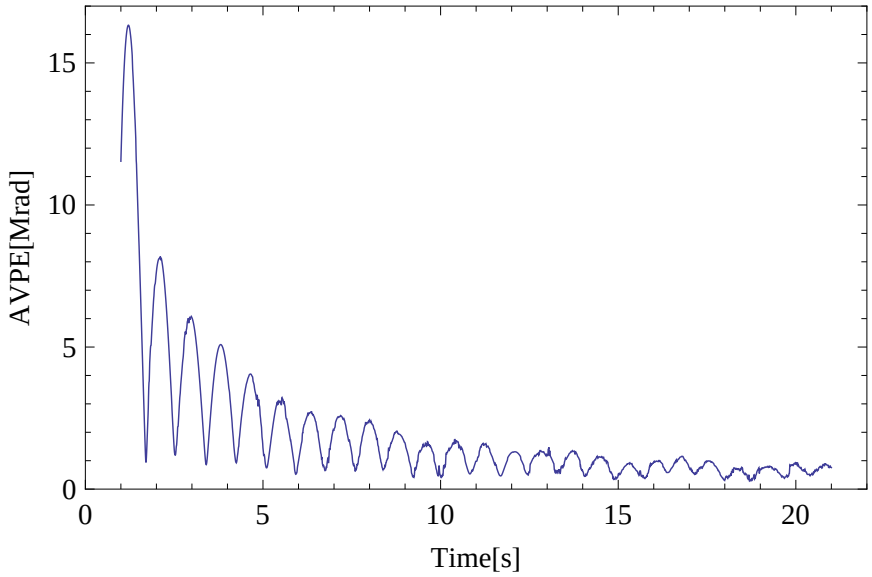


(c) Rigid body with a spherical pendulum

Fig. 4.8: Velocity pointing error for trapezoidal Thrust scheme.

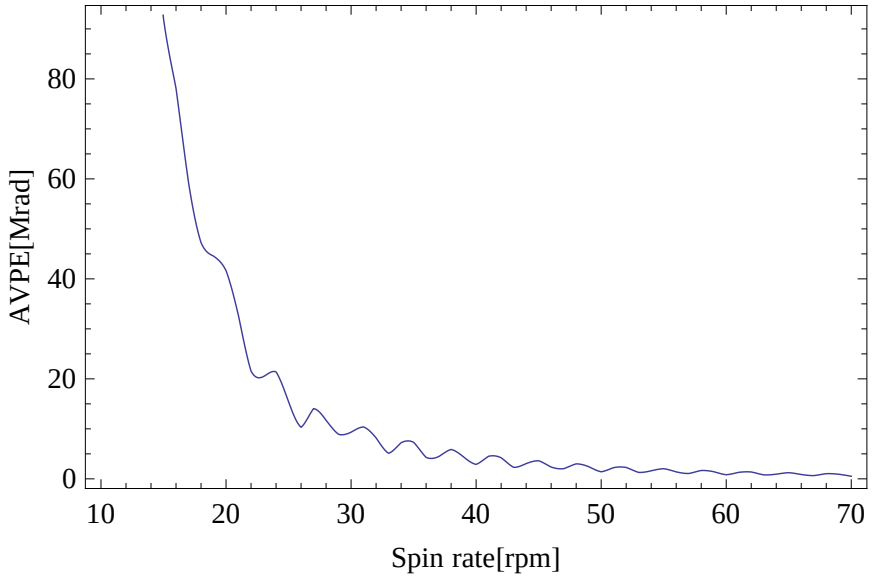


(a) Rigid body with planner pendulum

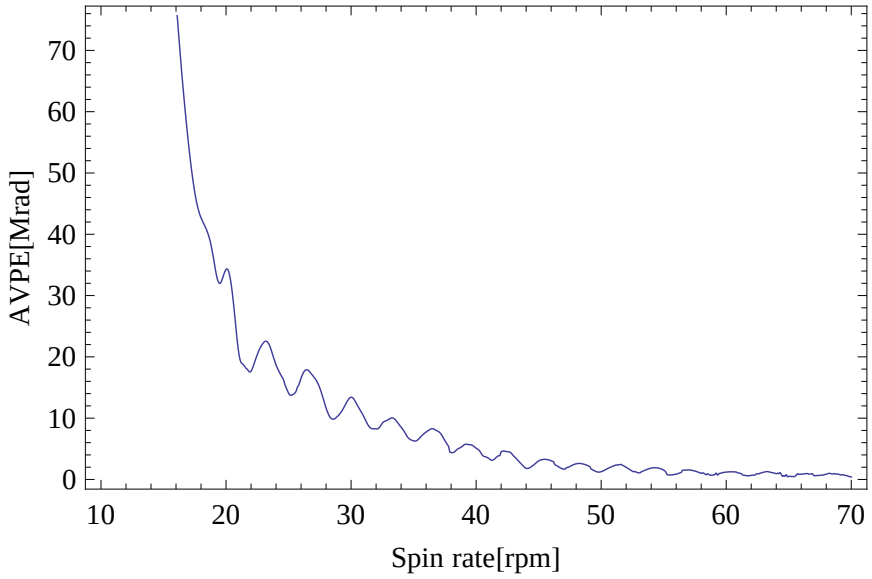


(b) Rigid body with spherical pendulum

Fig. 4.9: AVPE vs. ramp up time.



(a) planar pendulum



(b) spherical pendulum

Fig. 4.10: Average velocity pointing error vs. spin rate.

Chapter 5

Conclusion

We present a mathematical model of a gyrostat with a spherical pendulum as a mechanical equivalent of fuel sloshing in a partially-filled tank, and a nutation damper. We verify and validate our model for two standard maneuvers. This simplified model may be useful for stability analysis and nutation damper design. Also, it may provide insight into the nature of motion of fuel inside a rigid body.

The proposed model shall be improved by considering multiple tanks instead of one tank and by coupling the translational motion with attitude motion of a gyrostat in circular and elliptical orbits. We use a detailed mathematical model for a gyrostat with three momentum wheels and use for fuel slosh a spherical pendulum model and a new flexible planar pendulum to study the attitude motion of a thrusting, spinning mass-variable gyrostat with fuel sloshing or slag. We hypothesized that the existing trapezoidal thrust scheme for reducing the velocity pointing error of a rigid body works for a gyrostat with a moving internal parts such as a pendulum.

Numerical simulation shows that this hypothesis is correct. We also fine-tune the trapezoidal thrust scheme to minimize the average velocity pointing error subject to some engine characteristics constraints. The proposed trapezoidal scheme can be used as a passive control technique and allows for a lower spin rate.

The suggested future works to continue this project could be determining the optimal thrust profile in thrusting, spinning rigid spacecraft and rigid spacecraft with fuel sloshing with constant moment of inertia and also variable moment of inertia. Furthermore, design an active control of velocity pointing error for the spacecraft can be a good idea to continue this project.

Bibliography

[1] N. H. Abramson. The dynamic behavior of liquids in moving containers with applications to space vehicle technology. Technical report, NASA SP-106, 1966.

[2] H. Cheung and N. S. Cohen. Performance of solid propellants containing metal additives. *AIAA Journal*, 3(2):250–257, 1965.

[3] J. E. Cochran and J. Y. Kang. Nonlinear stability analysis of the attitude motion of a spin-stabilized upper stage. *Advances in the Astronautical Sciences*, 75(1):345–364, 1991.

[4] F. O. Eke. Dynamics of variable mass systems with applications to the star 4b solid rocket motor. *Journal of Guidance, Control and Dynamics*, 54(11):671–684, 1983.

[5] G. A. Flandro. Interaction of internal waves in a spinning solid propellant rocket motor with spacecraft motion. Technical report, AFRPL Report, October 1982.

[6] G. A. Flandro, W. K. VanMoorhem, and R. Shorthill. Fluid mechanics of spinning rockets. Technical Report TR-86-072, AFRPL Report, October 1987.

[7] R. A. Fredrick, J. A. Nichols, and J. Rogerson. Slag accumulation measurments in a strategic solid rocket motor. In *ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, number 96-2783, June 1996.

[8] V. E. Haloulakas. Slag mass accumulation in spinning solid rockets motors. *Journal of Propulsion*, 7(1):14–21, 1991.

[9] D. E. Hill and J. R. Baumgarten. Dynamic simulation of spin-stabilized spacecraft with sloshing fluid stores. *Journal of Guidance, Control and Dynamics*, 11(6):597–599, 1988.

[10] Daniel Eugene Hill. Dynamics and control of spin-stabilized spacecraft with sloshing fluid stores. Technical report, Iowa State University, Iowa, 1985.

[11] P. C. Hughes. *Spacecraft Attitude Dynamics*. Dover Publications Inc., 2004.

[12] R. W. Humble, G. N. Henry, and W. J. Larson. *Space Propulsion Analysis and Design, 1st ed.* Primis Custom Publishing, New York, 1995.

[13] Daniel Javorsek and James M. Longuski. Velocity pointing errors associated with spinning thrusting spacecraft. *Journal of Spacecraft and Rockets*, 37(3):359–365, 2000.

[14] Thomas R. Kane and David A. Levinson. *Dynamics Theory and Applications*. Mc-Graw Hill, Inc., 1985.

[15] Ja Young Kang and J. E. Cochran. Resonant motion of a spin-stabilized thrusting spacecraft. *Journal of Guidance, Control and Dynamics*, 27(3):356–364, 2004.

[16] Ja Young Kang and Sangchul Lee. Attitude acquisition of a satellite with a partially filled liquid tank. *Journal of Guidance, Control and Dynamics*, 31(3):790–793, 2008.

[17] R. N. Knauber. Thrust misalignments of fixed-nozzle solid rocket motors. *Journal of Spacecraft and Rockets*, 33(6):794–799, 1996.

[18] R. X. Meyer. Coining instability of spacecraft during periods of thrust. *Journal of Spacecraft and Rockets*, 33(6):781–788, 1996.

[19] D. L. Mingori and Y. Yam. Nutation instability of a spinning spacecraft with internal mass motion and axial thrust. In *AIAA/AAS Astrodynamics Conference*, number 86-2271, August 1986.

[20] A. C. Or. Rotor-pendulum model for the perigee assist module nutation anomaly. *Journal of Guidance, Control and Dynamics*, 15(2):297–303, 1992.

[21] A. C. Or and A. D. Challoner. Stability of spinning spacecraft containing shallow pool of liquid under thrust. *Journal of Guidance,Control and Dynamics*, 17(5):1019–1027, 1994.

[22] Y. Yam, D. L. Mingori, and D. M. Halsmer. Stability of a spinning axisymmetric rocket with dissipative internal mass motion. *Journal of Guidance, Control, and Dynamics*, 20(2):306–312, 1996.

Appendix A

MATHEMATICA Codes

A.1 Flat Spin Maneuver

Flat Spin Maneuver

Figures presented in the flat spin maneuver part are generated using the codes in the following. These figures are shown in page 33, 34, 35 and 37.

- **Constants**

```
mp = mP = 500; (*Kg*)mb = mB = 5274.4; (*Kg*)mq = mQ = 52.744; (*Kg*)
cc = 105(*Kg.m/s^2*); kk = 52(*N/m*); yQ = 1; (*m*)I1 = 1402.4;
I2 = 1292.5; I3 = 1375.7; l = .15; (*m*)u7[t_] = u8[t_] = u9[t_] = 0;
```

- **Velocities and Accelerations**

```
NωB[t_] = {u4[t], u5[t], u6[t]};
Nωw1[t_] = {u4[t] + u7[t], u5[t], u6[t]};
Nωw2[t_] = {u4[t], u5[t] + u8[t], u6[t]};
Nωw3[t_] = {u4[t], u5[t], u6[t] + u9[t]};
NωP[t_] = NωB[t] + {0, 0, D[ψ1[t], t]} + D[ψ2[t], t] * {Sin[ψ1[t]], -Cos[ψ1[t]], 0};
rBP[t_] = l * {Cos[ψ1[t]] Sin[ψ2[t]], Sin[ψ1[t]] Sin[ψ2[t]], -Cos[ψ2[t]]};
NVB[t_] = {u1[t], u2[t], u3[t]};
NVP[t_] = NVB[t] + Cross[NωP[t], rBP[t]];
rBQ[t_] = {q[t], 0,  yQ};
NVQ[t_] = NVB[t] + Cross[NωB[t], rBQ[t]] + {D[ q[t], t], 0, 0};
```

Simplifying the Equations

```
z1[t_] = u5[t] u3[t] - u6[t] u2[t];
z2[t_] = u6[t] u1[t] - u4[t] u3[t];
z3[t_] = u4[t] u2[t] - u5[t] u1[t];
z4[t_] = - l Cos[ψ1[t]] Sin[ψ2[t]] ((D[ψ1[t], t])^2) -
   l Cos[ψ1[t]] Sin[ψ2[t]] ((D[ψ2[t], t])^2) + l Sin[ψ1[t]] Sin[ψ2[t]] u4[t] u5[t] -
   l Cos[ψ1[t]] Sin[ψ2[t]] ((u5[t])^2) - l Cos[ψ2[t]] u4[t] u6[t] -
   l Cos[ψ1[t]] Sin[ψ2[t]] ((u6[t])^2) + 2 l D[ψ2[t], t] Sin[ψ2[t]] u5[t] -
   2 l D[ψ2[t], t] Cos[ψ2[t]] Sin[ψ1[t]] u6[t] - 2 l D[ψ1[t], t] Cos[ψ2[t]] Sin[ψ1[t]] D[ψ2[t], t] -
   2 l D[ψ1[t], t] Cos[ψ1[t]] Sin[ψ2[t]] u6[t] + z1[t];
z5[t_] = - l Sin[ψ1[t]] Sin[ψ2[t]] ((D[ψ1[t], t])^2) - l Sin[ψ1[t]] Sin[ψ2[t]] ((D[ψ2[t], t])^2) +
   l Cos[ψ1[t]] Sin[ψ2[t]] u4[t] u5[t] - l Sin[ψ1[t]] Sin[ψ2[t]] ((u4[t])^2) -
   l Cos[ψ2[t]] u5[t] u6[t] - l Sin[ψ1[t]] Sin[ψ2[t]] ((u6[t])^2) -
   2 l D[ψ2[t], t] Sin[ψ2[t]] u4[t] + 2 l D[ψ2[t], t] Cos[ψ2[t]] Cos[ψ1[t]] u6[t] +
   2 l D[ψ1[t], t] Cos[ψ2[t]] Cos[ψ1[t]] D[ψ2[t], t] -
   2 l D[ψ1[t], t] Sin[ψ1[t]] Sin[ψ2[t]] u6[t] + z2[t];
z6[t_] = l Cos[ψ2[t]] ((D[ψ2[t], t])^2) + l Cos[ψ2[t]] ((u4[t])^2) + l Cos[ψ2[t]] ((u5[t])^2) +
   2 l u4[t] Cos[ψ2[t]] Sin[ψ1[t]] D[ψ2[t], t] - 2 l u5[t] Cos[ψ2[t]] Cos[ψ1[t]] D[ψ2[t], t] +
   2 l D[ψ1[t], t] Sin[ψ2[t]] Cos[ψ1[t]] u4[t] + 2 l D[ψ1[t], t] Sin[ψ2[t]] Sin[ψ1[t]] u5[t] +
   l u4[t] Sin[ψ2[t]] Cos[ψ1[t]] u6[t] + l u5[t] Sin[ψ2[t]] Sin[ψ1[t]] u6[t] + z3[t];
NaQ[t_] = {u3[t] u5[t] - q[t] u5[t]^2 - u2[t] u6[t] + yQ u4[t] u6[t] - q[t] u6[t]^2 +
    u1'[t] + D[q'[t], t] + yQ u5'[t], -u3[t] u4[t] + q[t] u4[t] u5[t] + u1[t] u6[t] +
    q'[t] u6[t] + yQ u5[t] u6[t] + u6[t] q'[t] + u2'[t] - yQ u4'[t] + q[t] u6'[t],
   u2[t] u4[t] - yQ u4[t]^2 - u1[t] u5[t] - q'[t] u5[t] - yQ u5[t]^2 +
    q[t] u4[t] u6[t] - u5[t] q'[t] + u3'[t] - q[t] u5'[t]}; (*m/s^2*)
NaB[t_] = {D[u1[t], t] + z1[t], D[u2[t], t] + z2[t], D[u3[t], t] + z3[t]}; (*m/s^2*)
NaP[t_] =
  {z4[t] - l Sin[ψ1[t]] Sin[ψ2[t]] D[D[ψ1[t], t], t] + l Cos[ψ1[t]] Cos[ψ2[t]] D[D[ψ2[t], t], t] -
    l Cos[ψ2[t]] u5'[t] - l Sin[ψ1[t]] Sin[ψ2[t]] u6'[t] + D[u1[t], t],
   z5[t] + l Cos[ψ1[t]] Sin[ψ2[t]] D[D[ψ1[t], t], t] + l Cos[ψ2[t]] Sin[ψ1[t]] D[D[ψ2[t], t], t] +
    l Cos[ψ2[t]] u4'[t] + l Cos[ψ1[t]] Sin[ψ2[t]] u6'[t] + D[u2[t], t],
   D[u3[t], t] + z6[t] + l Sin[ψ1[t]] Sin[ψ2[t]] u4'[t] - l Cos[ψ1[t]] Sin[ψ2[t]] u5'[t] +
    l Sin[ψ2[t]] D[D[ψ2[t], t], t]}; (*m/s^2*)
NαB[t_] = D[NωB[t], t]; (*rad/s^2*)
Nαw1[t_] = D[Nωw1[t], t] + Cross[NωB[t], Nωw1[t]]; (*rad/s^2*)
Nαw2[t_] = D[Nωw2[t], t] + Cross[NωB[t], Nωw2[t]]; (*rad/s^2*)
Nαw3[t_] = D[Nωw3[t], t] + Cross[NωB[t], Nωw3[t]]; (*rad/s^2*)
```

Moment of Inertia Matrices

$$IB = \begin{pmatrix} I1 & 0 & 0 \\ 0 & I2 & 0 \\ 0 & 0 & I3 \end{pmatrix};\quad Iw1 = \begin{pmatrix} .17 & 0 & 0 \\ 0 & .1 & 0 \\ 0 & 0 & .1 \end{pmatrix};\quad Iw2 = \begin{pmatrix} .1 & 0 & 0 \\ 0 & .17 & 0 \\ 0 & 0 & .1 \end{pmatrix};\quad Iw3 = \begin{pmatrix} .1 & 0 & 0 \\ 0 & .1 & 0 \\ 0 & 0 & .17 \end{pmatrix};$$

- **Generalized Inertia Forces**

```
GIForce1[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u1[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u1[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u1[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u1[t]] +
    mB * NaB[t].D[NVB[t], u1[t]] + mp * NaP[t].D[NVP[t], u1[t]] + mQ * NaQ[t].D[NVQ[t], u1[t]];
GIForce2[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u2[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u2[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u2[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u2[t]] +
    mB * NaB[t].D[NVB[t], u2[t]] + mp * NaP[t].D[NVP[t], u2[t]] + mQ * NaQ[t].D[NVQ[t], u2[t]];
GIForce3[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u3[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u3[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u3[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u3[t]] +
    mB * NaB[t].D[NVB[t], u3[t]] + mp * NaP[t].D[NVP[t], u3[t]] + mQ * NaQ[t].D[NVQ[t], u3[t]];
GIForce4[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u4[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u4[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u4[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u4[t]] +
    mB * NaB[t].D[NVB[t], u4[t]] + mp * NaP[t].D[NVP[t], u4[t]] + mQ * NaQ[t].D[NVQ[t], u4[t]];
GIForce5[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u5[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u5[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u5[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u5[t]] +
    mB * NaB[t].D[NVB[t], u5[t]] + mp * NaP[t].D[NVP[t], u5[t]] + mQ * NaQ[t].D[NVQ[t], u5[t]];
GIForce6[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u6[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u6[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u6[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u6[t]] +
    mB * NaB[t].D[NVB[t], u6[t]] + mp * NaP[t].D[NVP[t], u6[t]] + mQ * NaQ[t].D[NVQ[t], u6[t]];
GIForce10[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], D[ψ1[t], t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], D[ψ1[t], t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], D[ψ1[t], t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], D[ψ1[t], t]] +
    mB * NaB[t].D[NVB[t], D[ψ1[t], t]] + mp * NaP[t].D[NVP[t], D[ψ1[t], t]] +
    mQ * NaQ[t].D[NVQ[t], D[ψ1[t], t]];
GIForce11[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], D[ψ2[t], t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], D[ψ2[t], t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], D[ψ2[t], t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], D[ψ2[t], t]] +
    mB * NaB[t].D[NVB[t], D[ψ2[t], t]] + mp * NaP[t].D[NVP[t], D[ψ2[t], t]] +
    mQ * NaQ[t].D[NVQ[t], D[ψ2[t], t]];
GIForce12[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], D[q[t], t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], D[q[t], t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], D[q[t], t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], D[q[t], t]] +
    mB * NaB[t].D[NVB[t], D[q[t], t]] +
    mp * NaP[t].D[NVP[t], D[q[t], t]] + mQ * NaQ[t].D[NVQ[t], D[q[t], t]];
```

Solving the Equation of Motion(Without Nutation Damper)

```
sol = NDSolve[{
    GIForce1[t] == 0, GIForce2[t] == 0, GIForce3[t] == 0, GIForce4[t] == -.5 Sin[ψ1[t]] ψ2'[t],
    GIForce5[t] == .5 Cos[ψ1[t]] ψ2'[t], GIForce6[t] == -.5 ((Sin[ψ2[t]])^2) ψ1'[t],
    GIForce10[t] == 0, GIForce11[t] == 0, GIForce12[t] == -kk * q[t] - cc * D[q[t], t],
    D[ψ[t], t] == (u5[t] Sin[φ[t]] + u6[t] Cos[φ[t]]) / (Cos[Θ[t]]),
    D[Θ[t], t] == u5[t] Cos[φ[t]] - u6[t] Sin[φ[t]],
    D[φ[t], t] == ((u5[t] Sin[φ[t]] Sin[Θ[t]] + u6[t] Cos[φ[t]] Sin[Θ[t]]) / (Cos[Θ[t]])) + u4[t],
    u1[0] == 0, u2[0] == 0, u3[0] == 0
    , u4[0] == 0, ψ1'[0] == 0, ψ2'[0] == 0, u5[0] == 10 * 2 * π / 60, u6[0] == 0, Θ[0] == 0,
    ψ[0] == 0, φ[0] == 0, q'[0] == 0, q[0] == .01, ψ1[0] == π / 180, ψ2[0] == π / 180},
   {u1, u2, u3, u4, u5, u6, ψ1', ψ2', φ, Θ, ψ, q'}, {t, 0, 10000}, Method →
    {StiffnessSwitching, Method → {ExplicitRungeKutta, Automatic}}, MaxSteps → 1000000]
```

- **Plotting the Angular Velocities Vs. Time(Without Nutation Damper)**

```
p7 = Plot[Evaluate[{Θ[t]} /. sol], {t, 0, 10000}, PlotRange → All, Frame → True,
  FrameLabel → {"Time-[Sec.]", "Without nutation damper"}, LabelStyle → Directive[10, Black]]
```

```
p1 = Plot[Evaluate[{u4[t]} /. sol], {t, 0, 10000}, PlotRange → All, Frame → True,
  FrameLabel → {"Time-[Sec.]", "Without nutation damper"}, LabelStyle → Directive[10, Black]]
```

```
p2 = Plot[Evaluate[{u5[t]} /. sol], {t, 0, 10000}, PlotRange → All, Frame → True,
  FrameLabel → {"Time-[Sec.]", "Without nutation damper"}, LabelStyle → Directive[10, Black]]
```

```
p3 = Plot[Evaluate[{u6[t]} /. sol], {t, 0, 10000}, PlotRange → All, Frame → True,
  FrameLabel → {"Time-[Sec.]", "Without nutation damper"}, LabelStyle → Directive[10, Black]]
```

- **Solving the Equation of Motion(Without Nutation Damper)**

```
sol2 = NDSolve[{
    GIForce1[t] == 0, GIForce2[t] == 0, GIForce3[t] == 0, GIForce4[t] == -.5 Sin[ψ1[t]] ψ2'[t],
    GIForce5[t] == .5 Cos[ψ1[t]] ψ2'[t], GIForce6[t] == -.5 ((Sin[ψ2[t]])^2) ψ1'[t],
    GIForce10[t] == 0, GIForce11[t] == 0, GIForce12[t] == -kk * q[t] - cc * D[q[t], t],
    D[ψ[t], t] == (u5[t] Sin[φ[t]] + u6[t] Cos[φ[t]]) / (Cos[Θ[t]]),
    D[Θ[t], t] == u5[t] Cos[φ[t]] - u6[t] Sin[φ[t]],
    D[φ[t], t] == ((u5[t] Sin[φ[t]] Sin[Θ[t]] + u6[t] Cos[φ[t]] Sin[Θ[t]]) / (Cos[Θ[t]])) + u4[t],
    u1[0] == 0, u2[0] == 0, u3[0] == 0
    , u4[0] == 0, ψ1'[0] == 0, ψ2'[0] == 0, u5[0] == 10 * 2 * π / 60, u6[0] == 0, Θ[0] == 0,
    ψ[0] == 0, φ[0] == 0, q'[0] == 0, q[0] == .01, ψ1[0] == π / 180, ψ2[0] == π / 180},
   {u1, u2, u3, u4, u5, u6, ψ1', ψ2', φ, Θ, ψ, q'}, {t, 0, 10000}, Method →
    {StiffnessSwitching, Method → {ExplicitRungeKutta, Automatic}}, MaxSteps → 1000000]
```

- **Plotting the Angular Velocities Vs. Time(With Nutation Damper)**

```
p8 = Plot[Evaluate[{Θ[t]} /. sol2], {t, 0, 10000}, PlotRange → All, Frame → True,
  FrameLabel → {"Time-[Sec.]", "With nutation damper"}, LabelStyle → Directive[10, Black]]
```

```
p4 = Plot[Evaluate[{u4[t]} /. sol2], {t, 0, 10000}, PlotRange → All, Frame → True,
  FrameLabel → {"Time-[Sec.]", "With nutation damper"}, LabelStyle → Directive[10, Black]]
```

```
p5 = Plot[Evaluate[{u5[t]} /. sol2], {t, 0, 10 000}, PlotRange → All, Frame → True,
  FrameLabel → {"Time-[Sec.]", "With nutation damper"}, LabelStyle → Directive[10, Black]]

p6 = Plot[Evaluate[{u6[t]} /. sol2], {t, 0, 10 000}, PlotRange → All, Frame → True,
  FrameLabel → {"Time-[Sec.]", "With nutation damper"}, LabelStyle → Directive[10, Black]]
```

A.2 Reorientation Maneuver

Reorientation Maneuver

Figures presented in the reorientation maneuver part are generated using the codes in the following.
These figures are shown in page 38.

- **Constant**

```
mb = mB = 5274.4; (*Kg*)mq = mQ = 52.744; (*Kg*)cc = 105(*N.m/s^2*); kk = 52; (*N/m*)
yQ = 1; (*m*)I1 = 1402.4; I2 = 1292.5; I3 = 1375.7; l = .15; (*m*)u7[t_] = u8[t_] = 0;
u9[t_] = Which[ 0 <= t <= 1000, 0,
     1000 < t <= 7400, 0.9375 (t - 1000),
     7400 < t ≤ 25 000, 6000] * (2 * π / 60);(*rad/s*)
```

- **Velocities and Accelerations**

```
NωB[t_] = {u4[t], u5[t], u6[t]};
Nωw1[t_] = {u4[t] + u7[t], u5[t], u6[t]};
Nωw2[t_] = {u4[t], u5[t] + u8[t], u6[t]};
Nωw3[t_] = {u4[t], u5[t], u6[t] + u9[t]};
NVB[t_] = {u1[t], u2[t], u3[t]};
rBQ[t_] = {q[t], 0,  yQ};
NVQ[t_] = NVB[t] + Cross[NωB[t], rBQ[t]] + {D[ q[t], t], 0, 0};
z1[t_] = u5[t] u3[t] - u6[t] u2[t];
z2[t_] = u6[t] u1[t] - u4[t] u3[t];
z3[t_] = u4[t] u2[t] - u5[t] u1[t];
NaQ[t_] =
  {u3[t] u5[t] - q[t] u5[t]^2 - u2[t] u6[t] + yQ u4[t] u6[t] - q[t] u6[t]^2 + u1'[t] + D[q'[t], t] +
    yQ u5'[t], -u3[t] u4[t] + q[t] u4[t] u5[t] + u1[t] u6[t] + q'[t] u6[t] + yQ u5[t] u6[t] +
    u6[t] q'[t] + u2'[t] - yQ u4'[t] + q[t] u6'[t], u2[t] u4[t] - yQ u4[t]^2 - u1[t] u5[t] -
    q'[t] u5[t] - yQ u5[t]^2 + q[t] u4[t] u6[t] - u5[t] q'[t] + u3'[t] - q[t] u5'[t]};
NaB[t_] = {D[u1[t], t] + z1[t], D[u2[t], t] + z2[t], D[u3[t], t] + z3[t]};
NαB[t_] = D[NωB[t], t];
Nαw1[t_] = D[Nωw1[t], t] + Cross[NωB[t], Nωw1[t]];
Nαw2[t_] = D[Nωw2[t], t] + Cross[NωB[t], Nωw2[t]];
Nαw3[t_] = D[Nωw3[t], t] + Cross[NωB[t], Nωw3[t]];
IB = ( I1 | 0 | 0 )
     ( 0  | I2 | 0 )
     ( 0  | 0 | I3 ); Iw1 = ( .17 | 0 | 0 )
                            ( 0  | .1 | 0 )
                            ( 0  | 0 | .1 ); Iw2 = ( .1 | 0 | 0 )
                                                   ( 0  | .17 | 0 )
                                                   ( 0  | 0 | .1 ); Iw3 = ( .1 | 0 | 0 )
                                                                          ( 0  | .1 | 0 )
                                                                          ( 0  | 0 | .17 );
```

Generalized Inertia Forces

```
GIForce1[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u1[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u1[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u1[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u1[t]] +
    mB * NaB[t].D[NVB[t], u1[t]] + mQ * NaQ[t].D[NVQ[t], u1[t]];
GIForce2[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u2[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u2[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u2[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u2[t]] +
    mB * NaB[t].D[NVB[t], u2[t]] + mQ * NaQ[t].D[NVQ[t], u2[t]];
GIForce3[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u3[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u3[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u3[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u3[t]] +
    mB * NaB[t].D[NVB[t], u3[t]] + mQ * NaQ[t].D[NVQ[t], u3[t]];
GIForce4[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u4[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u4[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u4[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u4[t]] +
    mB * NaB[t].D[NVB[t], u4[t]] + mQ * NaQ[t].D[NVQ[t], u4[t]];
GIForce5[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u5[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u5[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u5[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u5[t]] +
    mB * NaB[t].D[NVB[t], u5[t]] + mQ * NaQ[t].D[NVQ[t], u5[t]];
GIForce6[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], u6[t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], u6[t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], u6[t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], u6[t]] +
    mB * NaB[t].D[NVB[t], u6[t]] + mQ * NaQ[t].D[NVQ[t], u6[t]];
GIForce12[t_] := (IB.NαB[t] + Cross[NωB[t], IB.NωB[t]]).D[NωB[t], D[q[t], t]] +
    (Iw1.Nαw1[t] + Cross[Nωw1[t], Iw1.Nωw1[t]]).D[Nωw1[t], D[q[t], t]] +
    (Iw2.Nαw2[t] + Cross[Nωw2[t], Iw2.Nωw2[t]]).D[Nωw2[t], D[q[t], t]] +
    (Iw3.Nαw3[t] + Cross[Nωw3[t], Iw3.Nωw3[t]]).D[Nωw3[t], D[q[t], t]] +
    mB * NaB[t].D[NVB[t], D[q[t], t]] + mQ * NaQ[t].D[NVQ[t], D[q[t], t]];
```

Solving the Equation of Motion

```
sol = NDSolve[{
    GIForce1[t] == 0, GIForce2[t] == 0, GIForce3[t] == 0, GIForce4[t] == 0,
    GIForce5[t] == 0, GIForce6[t] == 0, GIForce12[t] == -kk * q[t] - cc * D[q[t], t],
    D[ψ[t], t] == (u5[t] Sin[φ[t]] + u6[t] Cos[φ[t]]) / (Cos[ϴ[t]]),
    D[ϴ[t], t] == u5[t] Cos[φ[t]] - u6[t] Sin[φ[t]],
    D[φ[t], t] == ((u5[t] Sin[φ[t]] Sin[ϴ[t]] + u6[t] Cos[φ[t]] Sin[ϴ[t]]) / (Cos[ϴ[t]])) + u4[t],
    u1[0] == 0, u2[0] == 0, u3[0] == 0
    , u4[0] == 5 * 2 * π / 60, u5[0] == 0, u6[0] == 0, ϴ[0] == 0, ψ[0] == 0, φ[0] == 0,
    q'[0] == 0, q[0] == .01}, {u1, u2, u3, u4, u5, u6, φ, ϴ, ψ, q'}, {t, 0, 10000}, Method →
    {StiffnessSwitching, Method → {ExplicitRungeKutta, Automatic}}, MaxSteps → 1000000]
```

Plotting the Angular Velocities Vs. Time

```
Plot[Evaluate[{u4[t], u5[t], u6[t]} /. sol], {t, 0, 10 000}, PlotRange → All, Frame → True,
 FrameLabel → {"Time-[Sec.]", "ω(t)-[rad/s]"}, LabelStyle → Directive[13, Black, Bold]]
```

- ### Plotting the Nutation Angle Vs. Time

```
Plot[Evaluate[{φ[t]} /. sol], {t, 0, 10 000}, PlotRange → All, Frame → True,
 FrameLabel → {"Time-[Sec.]", "Θ(t)-[rad]"}, LabelStyle → Directive[13, Black, Bold]]
```

A.3 AVPE with STAR 48B Thrust Profile

Spherical Pendulum with Star 48B Thrust Profile

Figures presented in the Average velocity pointing error part are generated using the codes in the following.
These figures are shown in page 42.

- **Constants**

```
mp = 2.268; (*Kg*)
l = 1.83; (*m*)
dd[t_] = -(.02 / 85.3) t + .02; (*m*)
mB[t_] = -24 t + 2500; (*Kg*)
h[t_] = ((1.55 - .8) / 85.3) t + .8; (*m*)
```

- **Velocities and accelerations**

```
NVB[t_] = {u1[t], u2[t], u3[t]};
NωB[t_] = {u4[t], u5[t], u6[t]};
NαB[t_] = D[NωB[t], t];
NaB[t_] = D[NVB[t], t] + Cross[NωB[t], NVB[t]];
NωP[t_] = NωB[t] + {0, 0, D[ψ1[t], t]} + D[ψ2[t], t] * {Sin[ψ1[t]], -Cos[ψ1[t]], 0};
rBP[t_] = l * {Cos[ψ1[t]] Sin[ψ2[t]], Sin[ψ1[t]] Sin[ψ2[t]], -Cos[ψ2[t]]};
NVP[t_] = NVB[t] + Cross[NωP[t], rBP[t]];
NaP[t_] = D[NVP[t], t] + Cross[NωB[t], NVP[t]];
I1[t_] = -((858 - 222) / 85.3) t + 858;
I2[t_] = -((858 - 222) / 85.3) t + 858; I3[t_] = -((401 - 102) / 85.3) t + 401;
IB[t_] = ( I1[t]  0     0
           0      I2[t]  0
           0      0      I3[t] );
```

Generalized Inertia Forces

```
GIForce1[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u1[t]]] +
   mB[t] * NaB[t].D[NVB[t], u1[t]] + mp * NaP[t].D[NVP[t], u1[t]];
GIForce2[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u2[t]]] +
   mB[t] * NaB[t].D[NVB[t], u2[t]] + mp * NaP[t].D[NVP[t], u2[t]];
GIForce3[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u3[t]]] +
   mB[t] * NaB[t].D[NVB[t], u3[t]] + mp * NaP[t].D[NVP[t], u3[t]];
GIForce4[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u4[t]]] +
   mB[t] * NaB[t].D[NVB[t], u4[t]] + mp * NaP[t].D[NVP[t], u4[t]];
GIForce5[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u5[t]]] +
   mB[t] * NaB[t].D[NVB[t], u5[t]] + mp * NaP[t].D[NVP[t], u5[t]];
GIForce6[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u6[t]]] +
   mB[t] * NaB[t].D[NVB[t], u6[t]] + mp * NaP[t].D[NVP[t], u6[t]];
GIForce10[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).
     D[NωB[t], D[ψ1[t], t]]] +
   mB[t] * NaB[t].D[NVB[t], D[ψ1[t], t]] + mp * NaP[t].D[NVP[t], D[ψ1[t], t]];
GIForce11[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).
     D[NωB[t], D[ψ2[t], t]]] +
   mB[t] * NaB[t].D[NVB[t], D[ψ2[t], t]] + mp * NaP[t].D[NVP[t], D[ψ2[t], t]];
```

Star 48B Thrust Force

```
pts = {{0, 13500 * 4.44822}, {.49, 13500 * 4.44822}, {.76, 13500 * 4.44822}, {.8, 13500 * 4.44822},
   {1, 13500 * 4.44822}, {1.2, 13510 * 4.44822}, {1.4, 13520 * 4.44822}, {1.51, 13530 * 4.44822},
   {1.7, 13540 * 4.44822}, {2, 13550 * 4.44822}, {4, 13560 * 4.44822}, {6, 13570 * 4.44822},
   {8, 13580 * 4.44822}, {10, 13590 * 4.44822}, {12, 13600 * 4.44822}, {14, 13800 * 4.44822},
   {16, 14000 * 4.44822}, {18, 14100 * 4.44822}, {20, 14400 * 4.44822}, {22, 14632 * 4.44822},
   {24, 14016 * 4.44822}, {26, 14400 * 4.44822}, {27.2, 14496 * 4.44822}, {28, 14400 * 4.44822},
   {30, 14208 * 4.44822}, {32, 14304 * 4.44822}, {34, 14592 * 4.44822}, {36, 14976 * 4.44822},
   {38, 15360 * 4.44822}, {40, 15552 * 4.44822}, {42, 15840 * 4.44822}, {44, 16128 * 4.44822},
   {46, 16320 * 4.44822}, {48, 16512 * 4.44822}, {50, 16704 * 4.44822}, {52, 16800 * 4.44822},
   {54, 16896 * 4.44822}, {56, 16992 * 4.44822}, {58, 17088 * 4.44822}, {60, 17126 * 4.44822},
   {62, 17184 * 4.44822}, {64, 17184 * 4.44822}, {66, 17088 * 4.44822}, {68, 16992 * 4.44822},
   {70, 16608 * 4.44822}, {71.2, 16224 * 4.44822}, {72, 16128 * 4.44822},
   {74, 15936 * 4.44822}, {76, 15840 * 4.44822}, {78, 15706 * 4.44822}, {80, 15552 * 4.44822},
   {82, 15360 * 4.44822}, {83.1, 15168 * 4.44822}, {85.3, 14880 * 4.44822}};
α = (.25 (π / 180) / (85.3)) t;
Tf = Interpolation[pts];
TfP[t_] = {-2.4 Sin[ψ1[t]] ψ2'[t], 2.4 Cos[ψ1[t]] ψ2'[t], 2.4 ψ1'[t]};
rBff[t_] = {0, dd[t], -h[t]};
Fthrust[t_] = {0, Tf[t] Sin[α], Tf[t] Cos[α]};
```

- **Generalized Active Forces**

```
GAForce1[t_] :=
  Simplify[TfP[t].D[NωB[t], u1[t]] + Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u1[t]]];
GAForce2[t_] := Simplify[TfP[t].D[NωB[t], u2[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u2[t]]];
GAForce3[t_] := Simplify[TfP[t].D[NωB[t], u3[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u3[t]]];
GAForce4[t_] := Simplify[TfP[t].D[NωB[t], u4[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u4[t]]];
GAForce5[t_] := Simplify[TfP[t].D[NωB[t], u5[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u5[t]]];
GAForce6[t_] := Simplify[TfP[t].D[NωB[t], u5[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u6[t]]];
GAForce10[t_] := Simplify[TfP[t].D[NωB[t], D[ψ1[t], t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), D[ψ1[t], t]]];
GAForce11[t_] := Simplify[TfP[t].D[NωB[t], D[ψ2[t], t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), D[ψ2[t], t]]];
```

- **Solving the Equation of Motion**

```
sol = NDSolve[{
   GIForce1[t] == GAForce1[t], GIForce2[t] == GAForce2[t],
   GIForce3[t] == GAForce3[t], GIForce4[t] == GAForce4[t], GIForce5[t] == GAForce5[t],
   GIForce6[t] == GAForce6[t], GIForce10[t] == GAForce10[t], GIForce11[t] == GAForce11[t],
   D[ψ[t], t] == (u5[t] Sin[φ[t]] + u6[t] Cos[φ[t]]) / (Cos[θ[t]]),
   D[θ[t], t] == u5[t] Cos[φ[t]] - u6[t] Sin[φ[t]],
   D[φ[t], t] == ((u5[t] Sin[φ[t]] Sin[θ[t]] + u6[t] Cos[φ[t]] Sin[θ[t]]) / (Cos[θ[t]])) + u4[t]
   , u1[0] == 0, u2[0] == 0, u3[0] == 0, u4[0] == 0, u5[0] == 0, u6[0] == 70 * 2 * π / 60,
   φ[0] == 0, ψ[0] == 0, θ[0] == 0, ψ1[0] == π / 180, ψ2[0] == π / 180, ψ1'[0] == 0, ψ2'[0] == 0},
  {u1, u2, u3, u4, u5, u6, ψ, θ, φ, ψ1, ψ2}, {t, 0, 85.3}, MaxSteps → 1 000 000]
```

- **Velocities Trasfrmation**

```
φ11[t_] = φ[t] /. sol[[1]];
θ11[t_] = θ[t] /. sol[[1]];
ψ11[t_] = ψ[t] /. sol[[1]];
RΦ1[t_] = RotationMatrix[-φ11[t], {1, 0, 0}];
Rθ2[t_] = RotationMatrix[-θ11[t], {0, 1, 0}]; Rψ3[t_] = RotationMatrix[-ψ11[t], {0, 0, 1}];
RψθΦ = Transpose[RΦ1[t].Rθ2[t].Rψ3[t]];
Velocity[t_] = RψθΦ.{u1[t] /. sol[[1]], u2[t] /. sol[[1]], u3[t] /. sol[[1]]};
```

- **Plotting the Velocity pointing Error**

```
ParametricPlot[
 Evaluate[1000 {Velocity[t][[1]] / Velocity[t][[3]], Velocity[t][[2]] / Velocity[t][[3]]}],
 {t, 0, 85.3}, Frame → True, AspectRatio → 1, FrameLabel → {"vx/vz[Mrad]", "vy/vz[Mrad]"},
 TextStyle → {FontFamily → "Times", FontSize → 16},
 PlotRange → {{-22, 110}, {-19, 93}}, Axes → {False, False}]
```

A.4 AVPE with Trapezoidal Thrust Scheme

Planar Pendulum with Trapezoidal Thrust Scheme

Figures presented in the Average velocity pointing error part are generated using the codes in the following.
These figures are shown in page 45.

- **Constant**

```
trf = 10.2857; (*s*)
tr = 10.2857; (*s*)
tb = 85.3; (*s*)
Ω = 70 * 2 * π / 60; (*rad/s*)
mp = 2.268; (*Kg*)
dd[t_] = -(.02 / tb) t + .02; (*m*)
mB[t_] = -24 t + 2500; (*Kg*)
kk = 0.25 ((mp * mB[t]) / (mp + mB[t])) (Ω^2); (*N/m*)
h[t_] = ((1.55 - .8) / tb) t + .8; (*m*)
```

- **Velocities and Accelerations**

```
NωB[t_] = {u4[t], u5[t], u6[t]};
NωP[t_] = NωB[t] + {0, 0, D[ψ1[t], t]};
NVB[t_] = {u1[t], u2[t], u3[t]};
NVP[t_] = NVB[t] + Cross[NωP[t], {l[t] Cos[ψ1[t]], l[t] Sin[ψ1[t]], 0}] +
   D[l[t], t] * {Cos[ψ1[t]], Sin[ψ1[t]], 0};
NαB[t_] = D[NωB[t], t];
NaB[t_] = D[NVB[t], t] + Cross[NωB[t], NVB[t]];
NaP[t_] = D[NVP[t], t] + Cross[NωB[t], NVP[t]];
I1[t_] = -((858 - 222) / 85.3) t + 858;
I2[t_] = -((858 - 222) / 85.3) t + 858; I3[t_] = -((401 - 102) / tb) t + 401;
IB[t_] = ( I1[t]   0      0
           0       I2[t]  0
           0       0      I3[t] );
```

Generalized Inertia Forces

```
GIForce1[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u1[t]]] +
   mB[t] * NaB[t].D[NVB[t], u1[t]] + mp * NaP[t].D[NVP[t], u1[t]];
GIForce2[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u2[t]]] +
   mB[t] * NaB[t].D[NVB[t], u2[t]] + mp * NaP[t].D[NVP[t], u2[t]];
GIForce3[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u3[t]]] +
   mB[t] * NaB[t].D[NVB[t], u3[t]] + mp * NaP[t].D[NVP[t], u3[t]];
GIForce4[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u4[t]]] +
   mB[t] * NaB[t].D[NVB[t], u4[t]] + mp * NaP[t].D[NVP[t], u4[t]];
GIForce5[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u5[t]]] +
   mB[t] * NaB[t].D[NVB[t], u5[t]] + mp * NaP[t].D[NVP[t], u5[t]];
GIForce6[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u6[t]]] +
   mB[t] * NaB[t].D[NVB[t], u6[t]] + mp * NaP[t].D[NVP[t], u6[t]];
GIForce10[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).
     D[NωB[t], D[ψ1[t], t]]] +
   mB[t] * NaB[t].D[NVB[t], D[ψ1[t], t]] + mp * NaP[t].D[NVP[t], D[ψ1[t], t]];
GIForce13[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).
     D[NωB[t], D[l[t], t]]] +
   mB[t] * NaB[t].D[NVB[t], D[l[t], t]] + mp * NaP[t].D[NVP[t], D[l[t], t]];
```

Thrust Force

```
α = ((.25 (π / 180)) / tb) t;
Tf[t_] = Piecewise[{{(76100 / tr) t, 0 ≤ t ≤ tr},
    {76100, tr < t ≤ trf}, {-(76100 / tb - trf) (t - trf) + 76100, trf < t ≤ tb}}];
rBff[t_] = {0, dd[t], -h[t]};
FSD[t_] = {-kk * l[t] Cos[ψ1[t]], -kk * l[t] Sin[ψ1[t]], 0};
Fthrust[t_] = {0, Tf[t] Sin[α], Tf[t] Cos[α]};
```

Generalized Active Forces

```
GAForce1[t_] :=
  Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u1[t]] + FSD[t].D[NVP[t], u1[t]]];
GAForce2[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u2[t]] +
    FSD[t].D[NVP[t], u2[t]]];
GAForce3[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u3[t]] +
    FSD[t].D[NVP[t], u3[t]]];
GAForce4[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u4[t]] +
    FSD[t].D[NVP[t], u4[t]]];
GAForce5[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u5[t]] +
    FSD[t].D[NVP[t], u5[t]]];
GAForce6[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u6[t]] +
    FSD[t].D[NVP[t], u6[t]]];
GAForce10[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), D[ψ1[t], t]] +
    FSD[t].D[NVP[t], D[ψ1[t], t]]];
GAForce13[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), D[l[t], t]] +
    FSD[t].D[NVP[t], D[l[t], t]]];
```

Solving the Equation of Motion

```
sol = NDSolve[{
     GIForce1[t] == GAForce1[t], GIForce2[t] == GAForce2[t],
     GIForce3[t] == GAForce3[t], GIForce4[t] == GAForce4[t], GIForce5[t] == GAForce5[t],
     GIForce6[t] == GAForce6[t], GIForce10[t] == GAForce10[t], GIForce13[t] == GAForce13[t],
     D[ψ[t], t] == (u5[t] Sin[φ[t]] + u6[t] Cos[φ[t]]) / (Cos[Θ[t]]),
     D[Θ[t], t] == u5[t] Cos[φ[t]] - u6[t] Sin[φ[t]],
     D[φ[t], t] ==
      ((u5[t] Sin[φ[t]] Sin[Θ[t]] + u6[t] Cos[φ[t]] Sin[Θ[t]]) / (Cos[Θ[t]])) + u4[t]
     , u1[0] == 0, u2[0] == 0, u3[0] == 0, u4[0] == 0, u5[0] == 0, u6[0] == Ω, φ[0] == 0,
     ψ[0] == 0, Θ[0] == 0, ψ1[0] == π / 180, l[0] == .01, ψ1'[0] == 0, l'[0] == 0},
    {u1, u2, u3, u4, u5, u6, ψ, Θ, φ, ψ1, l}, {t, 0, 85.3}, MaxSteps → 1 000 000];
```

- ### Velocities Transformation

```
φ11[t_] = φ[t] /. sol[[1]];
Θ11[t_] = Θ[t] /. sol[[1]];
ψ11[t_] = ψ[t] /. sol[[1]];
RΦ1[t_] = RotationMatrix[-φ11[t], {1, 0, 0}];
RΘ2[t_] = RotationMatrix[-Θ11[t], {0, 1, 0}]; Rψ3[t_] = RotationMatrix[-ψ11[t], {0, 0, 1}];
RψΘΦ = Transpose[RΦ1[t].RΘ2[t].Rψ3[t]];
Velocity[t_] = RψΘΦ.{u1[t] /. sol[[1]], u2[t] /. sol[[1]], u3[t] /. sol[[1]]};
```

- ### Plotting the Velocity Pointing Error

```
ParametricPlot[
 Evaluate[1000 {Velocity[t][[1]] / Velocity[t][[3]], Velocity[t][[2]] / Velocity[t][[3]]}],
 {t, 0, 85.3}, Frame → True, AspectRatio → 1, FrameLabel → {"vx/vz [Mrad]", "vy/vz[Mrad]"},
 TextStyle → {FontFamily → "Times", FontSize → 16},
 PlotRange → {{-6, 2}, {-3, 5}}, Axes → {False, False}]
```

A.5 AVPE Vs. Ramp Up Time

AVPE Vs. Ramp Up Time for Spherical Pendulum

Figures presented in the Optimizing the Trapezoidal thrust scheme part are generated using the codes in the following.
These figures are shown in page 46.

- **Constants**

```
f[tr_] := Module[{a = tr},
  Off[NIntegrate::ncvb];
  Off[NIntegrate::slwcon];
  trf = a + 64.766; (*s*)
  tb = 85.3;  (*s*)
  Ω = 70 * 2 * π / 60;  (*rad/s*)
  mp = 2.268;  (*Kg*)
  l = 1.83; (*m*)
```

- **Thrust Misalignments**

```
dd[t_] = -(.02 / tb) t + .02; (*m*)
mB[t_] = -24 t + 2500; (*Kg*)
h[t_] = ((1.55 - .8) / tb) t + .8; (*m*)
```

- **Velocities ana Accelerations**

```
NVB[t_] = {u1[t], u2[t], u3[t]};
NωB[t_] = {u4[t], u5[t], u6[t]};
NαB[t_] = D[NωB[t], t];
NaB[t_] = D[NVB[t], t] + Cross[NωB[t], NVB[t]];
NωP[t_] = NωB[t] + {0, 0, D[ψ1[t], t]} + D[ψ2[t], t] * {Sin[ψ1[t]], -Cos[ψ1[t]], 0};
rBP[t_] = l * {Cos[ψ1[t]] Sin[ψ2[t]], Sin[ψ1[t]] Sin[ψ2[t]], -Cos[ψ2[t]]};
NVP[t_] = NVB[t] + Cross[NωP[t], rBP[t]];
NaP[t_] = D[NVP[t], t] + Cross[NωB[t], NVP[t]];
I1[t_] = 858; I2[t_] = 858; I3[t_] = -((401 - 102) / tb) t + 401;
IB[t_] = ⎛ I1[t] |   0   |   0   ⎞
         ⎜ ------+-------+------- ⎟
         ⎜   0   | I2[t] |   0   ⎟
         ⎜ ------+-------+------- ⎟
         ⎝   0   |   0   | I3[t] ⎠ ;
```

Generalized Inertia Forces

```
GIForce1[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u1[t]]] +
   mB[t] * NaB[t].D[NVB[t], u1[t]] + mp * NaP[t].D[NVP[t], u1[t]];
GIForce2[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u2[t]]] +
   mB[t] * NaB[t].D[NVB[t], u2[t]] + mp * NaP[t].D[NVP[t], u2[t]];
GIForce3[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u3[t]]] +
   mB[t] * NaB[t].D[NVB[t], u3[t]] + mp * NaP[t].D[NVP[t], u3[t]];
GIForce4[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u4[t]]] +
   mB[t] * NaB[t].D[NVB[t], u4[t]] + mp * NaP[t].D[NVP[t], u4[t]];
GIForce5[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u5[t]]] +
   mB[t] * NaB[t].D[NVB[t], u5[t]] + mp * NaP[t].D[NVP[t], u5[t]];
GIForce6[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u6[t]]] +
   mB[t] * NaB[t].D[NVB[t], u6[t]] + mp * NaP[t].D[NVP[t], u6[t]];
GIForce10[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).
     D[NωB[t], D[ψ1[t], t]]] +
   mB[t] * NaB[t].D[NVB[t], D[ψ1[t], t]] + mp * NaP[t].D[NVP[t], D[ψ1[t], t]];
GIForce11[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).
     D[NωB[t], D[ψ2[t], t]]] +
   mB[t] * NaB[t].D[NVB[t], D[ψ2[t], t]] + mp * NaP[t].D[NVP[t], D[ψ2[t], t]];
α = ((.25 (π / 180)) / tb) t;
```

Thrust Force

```
Tf[t_] = Piecewise[{{(76 100 / a) t, 0 ≤ t ≤ a},
    {76 100, a < t ≤ trf}, {-(76 100 / tb - trf) (t - trf) + 76 100, trf < t ≤ tb}}];
TfP[t_] = {-2.4 Sin[ψ1[t]] ψ2'[t], 2.4 Cos[ψ1[t]] ψ2'[t], -2.4 ψ1'[t]};
rBff[t_] = {0, dd[t], -h[t]};
Fthrust[t_] = {0, Tf[t] Sin[α], Tf[t] Cos[α]};
```

Generalized Active Forces

```
GAForce1[t_] :=
  Simplify[TfP[t].D[NωB[t], u1[t]] + Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u1[t]]];
GAForce2[t_] := Simplify[TfP[t].D[NωB[t], u2[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u2[t]]];
GAForce3[t_] := Simplify[TfP[t].D[NωB[t], u3[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u3[t]]];
GAForce4[t_] := Simplify[TfP[t].D[NωB[t], u4[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u4[t]]];
GAForce5[t_] := Simplify[TfP[t].D[NωB[t], u5[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u5[t]]];
GAForce6[t_] := Simplify[TfP[t].D[NωB[t], u6[t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u6[t]]];
GAForce10[t_] := Simplify[TfP[t].D[NωB[t], D[ψ1[t], t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), D[ψ1[t], t]]];
GAForce11[t_] := Simplify[TfP[t].D[NωB[t], D[ψ2[t], t]] +
    Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), D[ψ2[t], t]]];
```

Solving the Equation of Motion

```
sol = NDSolve[{
     GIForce1[t] == GAForce1[t], GIForce2[t] == GAForce2[t],
     GIForce3[t] == GAForce3[t], GIForce4[t] == GAForce4[t], GIForce5[t] == GAForce5[t],
     GIForce6[t] == GAForce6[t], GIForce10[t] == GAForce10[t], GIForce11[t] == GAForce11[t],
     D[ψ[t], t] == (u5[t] Sin[φ[t]] + u6[t] Cos[φ[t]]) / (Cos[Θ[t]]),
     D[Θ[t], t] == u5[t] Cos[φ[t]] - u6[t] Sin[φ[t]],
     D[φ[t], t] ==
       ((u5[t] Sin[φ[t]] Sin[Θ[t]] + u6[t] Cos[φ[t]] Sin[Θ[t]]) / (Cos[Θ[t]])) + u4[t]
     , u1[0] == 0, u2[0] == 0, u3[0] == 0, u4[0] == 0, u5[0] == 0, u6[0] == Ω, φ[0] == 0,
     ψ[0] == 0, Θ[0] == 0, ψ1[0] == π / 180, ψ2[0] == π / 180, ψ1'[0] == 0, ψ2'[0] == 0},
    {u1, u2, u3, u4, u5, u6, ψ, Θ, φ, ψ1, ψ2}, {t, 0, 85.3}, MaxSteps → 1000000];
```

- **Velocities Transformation**

```
φ11[t_] = φ[t] /. sol[[1]];
Θ11[t_] = Θ[t] /. sol[[1]];
ψ11[t_] = ψ[t] /. sol[[1]];
RΦ1[t_] = RotationMatrix[-φ11[t], {1, 0, 0}];
RΘ2[t_] = RotationMatrix[-Θ11[t], {0, 1, 0}]; Rψ3[t_] = RotationMatrix[-ψ11[t], {0, 0, 1}];
RψΘΦ = Transpose[RΦ1[t].RΘ2[t].Rψ3[t]];
Velocity[t_] = RψΘΦ.{u1[t] /. sol[[1]], u2[t] /. sol[[1]], u3[t] /. sol[[1]]};
```

- **Velocity Pointing Error**

```
ρ[t_] = 1000 ArcTan[Evaluate[
       Sqrt[(((Velocity[t][[1]])^2) + ((Velocity[t][[2]])^2))] / (Velocity[t][[3]])]];
 AverageVelocityPointingError = N[(1 / 85.3) NIntegrate[ρ[t], {t, 0, 85.3}]];
 ; AverageVelocityPointingError]
```

- **Plotting the Average Velocity Pointing Error Vs. Ramp up Time**

```
nahaei = Interpolation[Table[{tr, f[tr]}, {tr, 1, 21, .001}]];

Plot[nahaei[t], {t, 1, 21}, FrameLabel → {"Time[s]", "AVPE[Mrad]"},
 TextStyle → {FontFamily → "Times", FontSize → 16},
 Axes → {True, True}, PlotRange → {{0, 22}, {0, 17}}, Frame → True]
```

A.6 AVPE Vs. Spin Rate

AVEP Vs. Spin Rate for Planar Pendulum

Figures presented in the Optimizing the Trapezoidal thrust scheme part are generated using the codes in the following.
These figures are shown in page 47.

- **Constans**

```
f[Ω_] := Module[{a = Ω},
  Off[NIntegrate::ncvb];
  Off[NIntegrate::slwcon];
  tr = 17.76;  (*s*)
  trf = tr + 64.766;  (*s*)
  tb = 85.3;  (*s*)
  mp = 2.268; (*Kg*)
```

- **Thrust misalignments**

```
dd[t_] = -(.02 / tb) t + .02; (*m*)
mB[t_] = -24 t + 2500; (*Kg*)
kk = 0.25 ((mp * mB[t]) / (mp + mB[t])) (Ω^2); (*N/m*)
h[t_] = ((1.55 - .8) / tb) t + .8; (*rad/s^2*)
```

- **Velocities and Accelerations**

```
NωB[t_] = {u4[t], u5[t], u6[t]};
NωP[t_] = NωB[t] + {0, 0, D[ψ1[t], t]};
NVB[t_] = {u1[t], u2[t], u3[t]};
NVP[t_] = NVB[t] + Cross[NωP[t], {l[t] Cos[ψ1[t]], l[t] Sin[ψ1[t]], 0}] +
   D[l[t], t] * {Cos[ψ1[t]], Sin[ψ1[t]], 0};
NαB[t_] = D[NωB[t], t];
NaB[t_] = D[NVB[t], t] + Cross[NωB[t], NVB[t]];
NaP[t_] = D[NVP[t], t] + Cross[NωB[t], NVP[t]];
I1[t_] = -((858 - 222) / 85.3) t + 858;
I2[t_] = -((858 - 222) / 85.3) t + 858; I3[t_] = -((401 - 102) / tb) t + 401;
IB[t_] = ( I1[t] |  0   |  0
           ------+------+------
            0   | I2[t] |  0
           ------+------+------
            0   |  0   | I3[t] );
```

Generalized Inertia Forces

```
GIForce1[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u1[t]]] +
   mB[t] * NaB[t].D[NVB[t], u1[t]] + mp * NaP[t].D[NVP[t], u1[t]];
GIForce2[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u2[t]]] +
   mB[t] * NaB[t].D[NVB[t], u2[t]] + mp * NaP[t].D[NVP[t], u2[t]];
GIForce3[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u3[t]]] +
   mB[t] * NaB[t].D[NVB[t], u3[t]] + mp * NaP[t].D[NVP[t], u3[t]];
GIForce4[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u4[t]]] +
   mB[t] * NaB[t].D[NVB[t], u4[t]] + mp * NaP[t].D[NVP[t], u4[t]];
GIForce5[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u5[t]]] +
   mB[t] * NaB[t].D[NVB[t], u5[t]] + mp * NaP[t].D[NVP[t], u5[t]];
GIForce6[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).D[NωB[t], u6[t]]] +
   mB[t] * NaB[t].D[NVB[t], u6[t]] + mp * NaP[t].D[NVP[t], u6[t]];
GIForce10[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).
     D[NωB[t], D[ψ1[t], t]]] +
   mB[t] * NaB[t].D[NVB[t], D[ψ1[t], t]] + mp * NaP[t].D[NVP[t], D[ψ1[t], t]];
GIForce13[t_] := FullSimplify[(IB[t].NαB[t] + Cross[NωB[t], IB[t].NωB[t]]).
     D[NωB[t], D[l[t], t]]] +
   mB[t] * NaB[t].D[NVB[t], D[l[t], t]] + mp * NaP[t].D[NVP[t], D[l[t], t]];
```

Thrust Force

```
α = ((.25 (π / 180)) / tb) t;
Tf[t_] = Piecewise[{{(76100 / tr) t, 0 ≤ t ≤ tr},
    {76100, tr < t ≤ trf}, {-(76100 / tb - trf) (t - trf) + 76100, trf < t ≤ tb}}];
rBff[t_] = {0, dd[t], -h[t]};
FSD[t_] = {-kk * l[t] Cos[ψ1[t]], -kk * l[t] Sin[ψ1[t]], 0};
Fthrust[t_] = {0, Tf[t] Sin[α], Tf[t] Cos[α]};
```

Generalized Active Forces

```
GAForce1[t_] :=
  Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u1[t]] + FSD[t].D[NVP[t], u1[t]]];
GAForce2[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u2[t]] +
    FSD[t].D[NVP[t], u2[t]]];
GAForce3[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u3[t]] +
    FSD[t].D[NVP[t], u3[t]]];
GAForce4[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u4[t]] +
    FSD[t].D[NVP[t], u4[t]]];
GAForce5[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u5[t]] +
    FSD[t].D[NVP[t], u5[t]]];
GAForce6[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), u6[t]] +
    FSD[t].D[NVP[t], u6[t]]];
GAForce10[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), D[ψ1[t], t]] +
    FSD[t].D[NVP[t], D[ψ1[t], t]]];
GAForce13[t_] := Simplify[Fthrust[t].D[(NVB[t] + Cross[NωB[t], rBff[t]]), D[l[t], t]] +
    FSD[t].D[NVP[t], D[l[t], t]]];
```

Solving the Equation of Motion

```
sol = NDSolve[{
     GIForce1[t] == GAForce1[t], GIForce2[t] == GAForce2[t],
     GIForce3[t] == GAForce3[t], GIForce4[t] == GAForce4[t], GIForce5[t] == GAForce5[t],
     GIForce6[t] == GAForce6[t], GIForce10[t] == GAForce10[t], GIForce13[t] == GAForce13[t],
     D[ψ[t], t] == (u5[t] Sin[φ[t]] + u6[t] Cos[φ[t]]) / (Cos[Θ[t]]),
     D[Θ[t], t] == u5[t] Cos[φ[t]] - u6[t] Sin[φ[t]],
     D[φ[t], t] ==
      ((u5[t] Sin[φ[t]] Sin[Θ[t]] + u6[t] Cos[φ[t]] Sin[Θ[t]]) / (Cos[Θ[t]])) + u4[t]
     , u1[0] == 0, u2[0] == 0, u3[0] == 0, u4[0] == 0, u5[0] == 0, u6[0] == a * 2 * π / 60,
     φ[0] == 0, ψ[0] == 0, Θ[0] == 0, ψ1[0] == π / 180, l[0] == .01, ψ1'[0] == 0, l'[0] == 0},
    {u1, u2, u3, u4, u5, u6, ψ, Θ, φ, ψ1, l}, {t, 0, 85.3}, MaxSteps → 1 000 000];
```

- **Velocities Transformation**

```
φ11[t_] = φ[t] /. sol[[1]];
Θ11[t_] = Θ[t] /. sol[[1]];
ψ11[t_] = ψ[t] /. sol[[1]];
RΦ1[t_] = RotationMatrix[-φ11[t], {1, 0, 0}];
RΘ2[t_] = RotationMatrix[-Θ11[t], {0, 1, 0}]; Rψ3[t_] = RotationMatrix[-ψ11[t], {0, 0, 1}];
RψΘΦ = Transpose[RΦ1[t].RΘ2[t].Rψ3[t]];
Velocity[t_] = RψΘΦ.{u1[t] /. sol[[1]], u2[t] /. sol[[1]], u3[t] /. sol[[1]]};
```

- **Velocity Pointing Error**

```
ρ[t_] = 1000 ArcTan[Evaluate[
       Sqrt[(((Velocity[t][[1]])^2) + ((Velocity[t][[2]])^2))] / (Velocity[t][[3]])]];
 AverageVelocityPointingError = N[(1 / 85.3) NIntegrate[ρ[t], {t, 0, 85.3}]];
 ; AverageVelocityPointingError]
```

- **Plotting the Average Velocity Pointing Error Vs. Spin Rate**

```
nahaei = Interpolation[Table[{Ω, f[Ω]}, {Ω, 0, 70, 1}]];

Plot[nahaei[Ω], {Ω, 10, 70}, FrameLabel → {"Spin rate[rpm]", "AVPE[Mrad]"},
 TextStyle → {FontFamily → "Times", FontSize → 16}, Axes → {True, True}, Frame → True]
```