دانشگاه کردستان
University of Kurdistan
زانکۆی کوردستان

University of Kurdistan

Department of Mechanical Engineering

Automatic Control

Simulation of the Satellite Carrier's Relative Angle Controller to the Ground Using Thrust Vector Control

and

Fabrication a Dynamic Model Using a Rotational Inverted Pendulum Model

Supervisor:

Dr. Sirvan Farhadi

Presenter:

Yasin Salehi

1 Table of Contents

2 Introduction 3

3 Satellite Carrier Model 3

4 Derivation of Equations 4

4.1 Pendulum 4

4.2 Rod 5

4.3 DC Motor 5

4.4 Motor and Rod Assembly 5

4.5 System Characteristic Equations (Nonlinear Equations) 6

4.6 System Characteristic Equations (Linear Equations) 8

4.7 State-Space Equations (Linear Equations) 8

5 Fabrication of the dynamic model 9

5.1 Specifications of the Fabricated model 9

5.2 Components used 9

5.3 State-space equations for the real model (linear equations) 9

5.4 Fabrication Challenges 9

5.5 Wiring diagram of the constructed model 10

6 Simulation 11

6.1 Root Locus Analysis 11

6.2 Proportional Controller 11

6.3 PID Controller Design 14

7 Conclusion 15

8 References 15

9 Appendix 16

2 Introduction

The importance of automatic control in industries and everyday life is undeniable, and it can be said that automatic control systems are among the most important parts of today's advanced industries. In this project, we will pose a question and use tools such as dynamics and control science to find a satisfactory answer. Although this question has been repeatedly addressed in the field of control science and has received satisfactory answers, the nature of science and scientific methods is reproducibility. We aim to reproduce an answer to this issue in both theoretical and practical states. The question is:

"How can the relative angle of a satellite carrier to the ground be controlled?"

The need to pose such a question arises when you want to actively control the satellite carrier's angle relative to the ground to eliminate the effects of unmeasurable and unpredictable external factors such as wind or gravity. This can be done in two ways: one is using side fins, which will consume a lot of fuel since satellite carriers usually have to move at angles other than 90 degrees to the ground. Weight is always a critical factor for the satellite carrier's range. The other method is thrust vector control, which is the method examined in this project. In this project, we first define the satellite carrier model and derive the equations describing the satellite carrier's motion, then we use control science to control the main variable, i.e., the satellite carrier's angle relative to the ground.

3 Satellite Carrier Model

In this project, we consider the model as a pendulum representing the satellite carrier. This model provides a reasonably accurate approximation due to similar geometric features (one dimension being significantly larger than the other two) and a nearly balanced mass distribution along the pendulum length. For the forces exerted on the satellite carrier due to thrust vector changes, we consider a cart or carrier. Here, for simplicity in providing rotational force with common motors in the market compared to the linear motion of the cart requiring a rail and force transmission process, we consider the cart as a carrying rod and connect the pendulum to the carrying rod via an encoder (an encoder measures the angle error of the pendulum relative to the ground or the feedback sensor). The carrying rod plays the role of the cart and provides the force exerted on the satellite carrier due to thrust angle changes via a DC motor. Obviously, in this project, we consider the problem in 2D, but the results can be easily generalized to the real 3D state.

4 Derivation of Equations

4.1 Pendulum

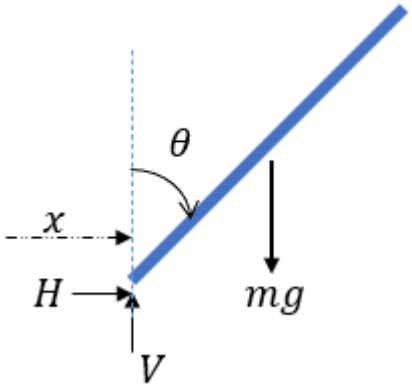

Figure 1

$Pendulum\ length = 2L$

$x_g = x + Lsin\theta$ (1)
$y_g = Lcos\theta$ (2)
$J\ddot{\theta} = \sum T$ (3)
$ma_x = H$ (4)
$ma_y = V - mg$ (5)

$$ma_x = H \longrightarrow m\frac{d^2x_g}{dt^2} = H \xrightarrow{(1)} m\ddot{x} + m(Lsin\theta)'' = H \qquad (6)$$

$$ma_y = V - mg \longrightarrow m\frac{d^2y_g}{dt^2} = V - mg \xrightarrow{(2)} 0 = V - mg \qquad (7)$$

Since we write the rod rotation equation around its center of mass and there is complete symmetry around the center of mass, $J = 0$. Therefore, equation (3) with the help of equation (7) is summarized as equation (8):

$$J = 0 \xrightarrow{(3)} 0 = \sum T \longrightarrow 0 = HLcos\theta - VLsin\theta \xrightarrow{(7)} H = mg\frac{sin\theta}{cos\theta} \qquad (8)$$

Now, we write the torques around the center of mass, resulting in the simplification of equation (3) as equation (9):

$$\ddot{x} + L(sin\theta)'' = g\frac{sin\theta}{cos\theta} \qquad (9)$$

4.2 Rod

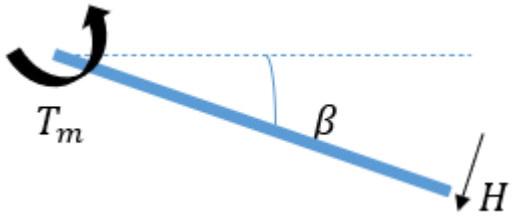

Figure 2

$Rod\ \ length = L'$

$$J\ddot{\beta} + b_0\dot{\beta} = \sum T \qquad (10)$$

In equation (10), the friction coefficient b_0 is considered for mechanical friction.

$$x = \beta L' \quad (11)$$

4.3 DC Motor

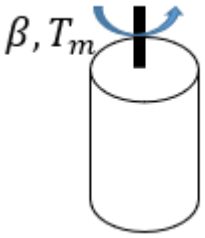

Figure 3

$$L_a\frac{di_a}{dt} + R_a i_a + K_b\dot{\beta} = e_a \quad (12)$$

In cylindrical DC motors (cross-section significantly smaller than length), L_a can be approximated as zero, making the equation a second-order function and simplifying the solution. Also, in these DC motors, the torque and back EMF constants are equal.

$$L_a = 0$$
$$T_m = K_T i_a \qquad (13)$$
$$K_b = K_T = K$$
$$R_a i_a + K\dot{\beta} = e_a \qquad (14)$$

4.4 Motor and Rod Assembly

$$(10), (11) \longrightarrow J\ddot{\beta} + b_0\dot{\beta} = T_m - HL' \xrightarrow{(13)} \frac{J}{K}\ddot{\beta} + \frac{b_0}{K}\dot{\beta} + \frac{HL'}{K} = i_a \quad (15)$$

$$(14),(15) \longrightarrow R_a(J\ddot{\beta} + b_0\dot{\beta} + HL') + K^2\dot{\beta} = Ke_a \longrightarrow R_aJ\ddot{\beta} + (R_ab_0 + K^2)\dot{\beta} + R_aL'H = Ke_a$$

$$\longrightarrow \quad R_aJ\ddot{\beta} + (R_ab_0 + K^2)\dot{\beta} + R_aL'(m\ddot{x} + mL(sin\theta)^{''}) = Ke_a \quad \longrightarrow$$

$$R_aJ\ddot{\beta} + (R_ab_0 + K^2)\dot{\beta} + R_aL'^2m\ddot{\beta} + R_amLL'(sin\theta)^{''} = Ke_a \quad \longrightarrow$$

$$(R_aJ + R_aL'^2m)\ddot{\beta} + (R_ab_0 + K^2)\dot{\beta} + R_amLL'(sin\theta)^{''} = Ke_a \quad (16)$$

4.5 System Characteristic Equations (Nonlinear Equations)

$$L'\ddot{\beta} + L(sin\theta)^{''} = g\frac{sin\theta}{cos\theta} \qquad (9)$$

$$(R_aJ + R_aL'^2m)\ddot{\beta} + (R_ab_0 + K^2)\dot{\beta} + R_amLL'(sin\theta)^{''} = Ke_a \quad (16)$$

$$(sin\theta)^{''} = \ddot{\theta}cos\theta - \dot{\theta}sin\theta \quad (17)$$

$$(9) \xrightarrow{(17)} L'\ddot{\beta} + L\ddot{\theta}cos\theta - L\dot{\theta}sin\theta = g\frac{sin\theta}{cos\theta} \qquad (18)$$

$$(16) \xrightarrow{(17)} (R_aJ + R_aL'^2m)\ddot{\beta} + (R_ab_0 + K^2)\dot{\beta} + R_amLL'\ddot{\theta}cos\theta - R_amLL'\dot{\theta}sin\theta = Ke_a \qquad (19)$$

$$(18) \longrightarrow \ddot{\theta} = \frac{gsin\theta}{Lcos^2\theta} + \frac{sin\theta}{cos\theta}\dot{\theta} - \frac{L'}{Lcos\theta}\ddot{\beta} \qquad (20)$$

$$(19) \longrightarrow \ddot{\beta} = \frac{K}{R_aJ+R_aL'^2m}e_a - \frac{(R_ab_0+K^2)}{R_aJ+R_aL'^2m}\dot{\beta} - \frac{R_amLL'}{R_aJ+R_aL'^2m}\ddot{\theta}cos\theta + \frac{R_amLL'}{R_aJ+R_aL'^2m}\dot{\theta}sin\theta \qquad (21)$$

$(20),(21) \longrightarrow$

$$\ddot{\theta} = \frac{gsin\theta}{Lcos^2\theta} + \frac{sin\theta}{cos\theta}\dot{\theta} - \frac{L'}{Lcos\theta}\left(\frac{K}{R_aJ + R_aL'^2m}e_a - \frac{(R_ab_0 + K^2)}{R_aJ + R_aL'^2m}\dot{\beta} - \frac{R_amLL'}{R_aJ + R_aL'^2m}\ddot{\theta}cos\theta + \frac{R_amLL'}{R_aJ + R_aL'^2m}\dot{\theta}sin\theta\right)$$

(21), (20) ⟶

$$\ddot{\beta} = \frac{K}{R_aJ + R_aL'^2m}e_a - \frac{(R_ab_0 + K^2)}{R_aJ + R_aL'^2m}\dot{\beta} - \frac{R_amLL'}{R_aJ + R_aL'^2m}\left(\frac{gsin\theta}{Lcos\theta} + \dot{\theta}sin\theta - \frac{L'}{L}\ddot{\beta}\right) + \frac{R_amLL'}{R_aJ + R_aL'^2m}\dot{\theta}sin\theta$$

Simplification ----------

$$\ddot{\theta} = \frac{gsin\theta}{Lcos^2\theta} + \frac{sin\theta}{cos\theta}\dot{\theta} - \frac{KL'}{L(R_aJ + R_aL'^2m)}\frac{e_a}{cos\theta} + \frac{L'(R_ab_0 + K^2)}{L(R_aJ + R_aL'^2m)}\frac{\dot{\beta}}{cos\theta} + \frac{R_amL'^2}{R_aJ + R_aL'^2m}\ddot{\theta} - \frac{R_amL'^2}{R_aJ + R_aL'^2m}\frac{\dot{\theta}sin\theta}{cos\theta}$$

$$\ddot{\beta} = \frac{K}{R_aJ + R_aL'^2m}e_a - \frac{(R_ab_0 + K^2)}{R_aJ + R_aL'^2m}\dot{\beta} - \frac{gR_amL'}{R_aJ + R_aL'^2m}\frac{sin\theta}{cos\theta} - \frac{R_amLL'}{R_aJ + R_aL'^2m}\dot{\theta}sin\theta + \frac{R_amL'^2}{R_aJ + R_aL'^2m}\ddot{\beta} + \frac{R_amLL'}{R_aJ + R_aL'^2m}\dot{\theta}sin\theta$$

Simplification ----------

$$\ddot{\theta} = \frac{R_aJ + R_aL'^2m}{R_aJ}\frac{gsin\theta}{Lcos^2\theta} - \frac{\dot{\theta}sin\theta}{cos\theta} - \frac{KL'}{R_aJL}\frac{e_a}{cos\theta} + \frac{L'(R_ab_0 + K^2)}{R_aJL}\frac{\dot{\beta}}{cos\theta}$$

$$\ddot{\beta} = \frac{K}{R_aJ}e_a - \frac{(R_ab_0 + K^2)}{R_aJ}\dot{\beta} - \frac{gmL'}{J}\frac{sin\theta}{cos\theta} - \frac{mLL'}{J}\dot{\theta}sin\theta + \frac{mLL'}{J}\dot{\theta}sin\theta$$

Simplified characteristic equations (nonlinear equations):

$$\ddot{\theta} = \frac{J + L'^2m}{J}\frac{gsin\theta}{Lcos^2\theta} - \frac{\dot{\theta}sin\theta}{cos\theta} - \frac{KL'}{R_aJL}\frac{e_a}{cos\theta} + \frac{L'(R_ab_0 + K^2)}{R_aJL}\frac{\dot{\beta}}{cos\theta} \quad (22)$$

$$\ddot{\beta} = \frac{K}{R_aJ}e_a - \frac{(R_ab_0 + K^2)}{R_aJ}\dot{\beta} - \frac{gmL'}{J}\frac{sin\theta}{cos\theta} \quad (23)$$

We use equations (22) and (23) to plot the pendulum and demonstrate its motion in MATLAB.

4.6 System Characteristic Equations (Linear Equations)

Linearization of equations (22) and (23):

Assumption: $sin\theta = \theta \ \ , cos\theta = 1 \ \ , \theta\dot{\theta} = 0$

$$\ddot{\theta} = \frac{g(J + L'^2 m)}{LJ}\theta + \frac{L'(R_a b_0 + K^2)}{LR_a J}\dot{\beta} - \frac{KL'}{LR_a J}e_a \qquad (24)$$

$$\ddot{\beta} = -\frac{gmL'}{J}\theta - \frac{(R_a b_0 + \dot{K}^2)}{R_a J}\dot{\beta} + \frac{K}{R_a J}e_a \qquad (25)$$

$$\ddot{\theta} = A1\,\theta + A2\,\dot{\beta} + B1\,e_a \qquad (26)$$

$$\ddot{\beta} = A3\,\theta + A4\,\dot{\beta} + B2\,e_a \qquad (27)$$

$$A1 = \frac{g(J + L'^2 m)}{LJ} \qquad A1 = \frac{g(J + L'^2 m)}{LJ} \qquad A3 = -\frac{gmL'}{J}$$

$$A4 = -\frac{(R_a b_0 + K^2)}{R_a J} \qquad B1 = -\frac{KL'}{LR_a J} \qquad B2 = \frac{K}{R_a J}$$

4.7 State-Space Equations (Linear Equations)

To design the controller, we first convert the equations to their state-space form and obtain the system in state-space representation. Therefore, from equations (26) and (27), we have:

$$\dot{x} = Ax + Be_a \qquad (28)$$

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ A1 & 0 & 0 & A2 \\ 0 & 0 & 0 & 1 \\ A3 & 0 & 0 & A4 \end{bmatrix} x + \begin{bmatrix} 0 \\ B1 \\ 0 \\ B2 \end{bmatrix} e_a \qquad (29)$$

$$\dot{x} = \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \end{bmatrix} = \begin{bmatrix} x_2 \\ \ddot{\theta} \\ x_4 \\ \ddot{\beta} \end{bmatrix} \qquad x = \begin{bmatrix} \theta \\ \dot{\theta} \\ \beta \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ A1 & 0 & 0 & A2 \\ 0 & 0 & 0 & 1 \\ A3 & 0 & 0 & A4 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ B1 \\ 0 \\ B2 \end{bmatrix}$$

5 Fabrication of the dynamic model

5.1 Specifications of the Fabricated model

$L' = 0.28\ m$ *#Rod length*

$L = 0.11\ m$ *#half Pendulum length*

$g = 9.81$ *#Gravity constant*

$K = \frac{0.588}{7.8}\ \frac{N.m}{A}$ *#DC motor torque constant*

$Ra = 3\ \Omega$ *# DC motor resistance*

$b0 = 0.1$ *#Mechanical resistance*

$m = 0.039\ Kg$ *# Pendulum weight*

5.2 Components used

The following components were used in the construction of this project:

1) 24V DC servo motor, model Yaskawa UGFMED-B5LGR51
2) Arduino Mega 2560
3) Autonics 2500 pulse encoder
4) DC motor driver 43A 27V, model BTS7960
5) 12V power supply (Arduino power supply)
6) 24V 10A power supply
7) Breadboard
8) Mounting plates and connecting rods

5.3 State-space equations for the real model (linear equations)

From equations (26) and (27) and obtaining their coefficients, we have:

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 102.75 & 0 & 0 & 12.9 \\ 0 & 0 & 0 & 1 \\ -5.33 & 0 & 0 & -5.07 \end{bmatrix} x + \begin{bmatrix} 0 \\ 3.18 \\ 0 \\ -1.25 \end{bmatrix} e_a$$

$A1 = 102.75$ $A2 = 12.9$ $A3 = -5.33$ $A4 = -5.07$ $B1 = 3.18$ $B2 = -1.25$

5.4 Fabrication Challenges

During the construction of the project, we encountered several issues which I'll detail here.

1) Encoder Signal: The signal B from the encoder, which represents the theta feedback (angle of the pendulum), was a square wave signal that reached a maximum approximate voltage of 1.7 volts. However, Arduino considers a minimum of 3 volts for high signals (or logic level '1'). Due to this voltage difference, Arduino couldn't accurately detect the angle or direction of rotation. This issue was resolved by adding a 1.5V AAA battery to compensate for the voltage difference.

2) Arduino Uno Limitations: Initially, we used the Arduino Uno controller, which, due to having only one pair of signal inputs for the encoder, couldn't provide a solution to our problem. This limitation was overcome by replacing it with the Arduino Mega controller.
3) Length of Lever Arm: The significant length of the lever arm caused increased sensitivity in the motor operation, leading to motor inertia issues. By minimizing the length of this lever arm, we were able to mitigate this problem.
4) Angular Refresh Rate: One fundamental issue we faced was the refresh rate of the pendulum angle. By reducing the refresh time to 1 millisecond, we initially achieved active control. However, these changes had an irreversible negative impact on the motor driver. With a refresh rate of 1 millisecond, the right-side H-bridge of the motor driver completely failed. This issue was resolved by replacing the motor driver and increasing the refresh rate to 10 milliseconds.

5.5 Wiring diagram of the constructed model

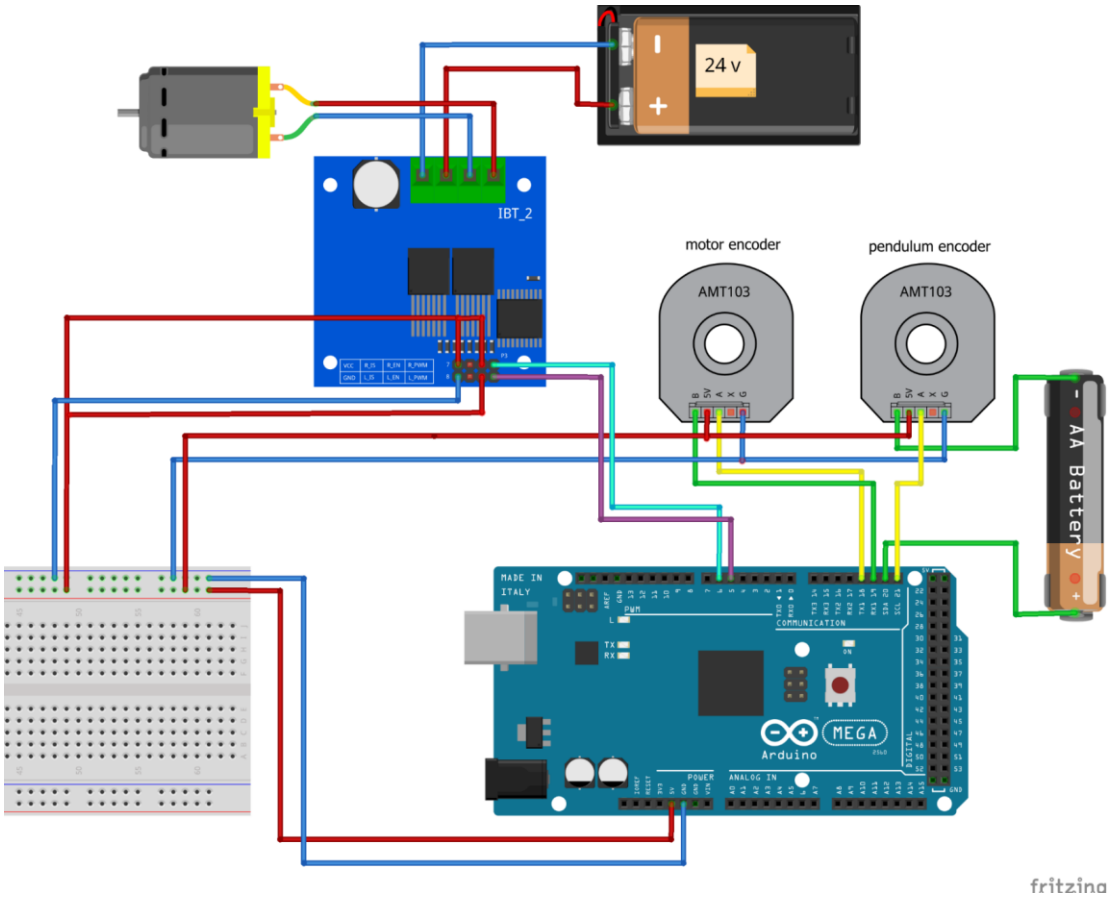

Figure 4

6 Simulation

6.1 Root Locus Analysis

The Root Locus Analysis of the closed-loop poles of the system can be observed in Figure 5.

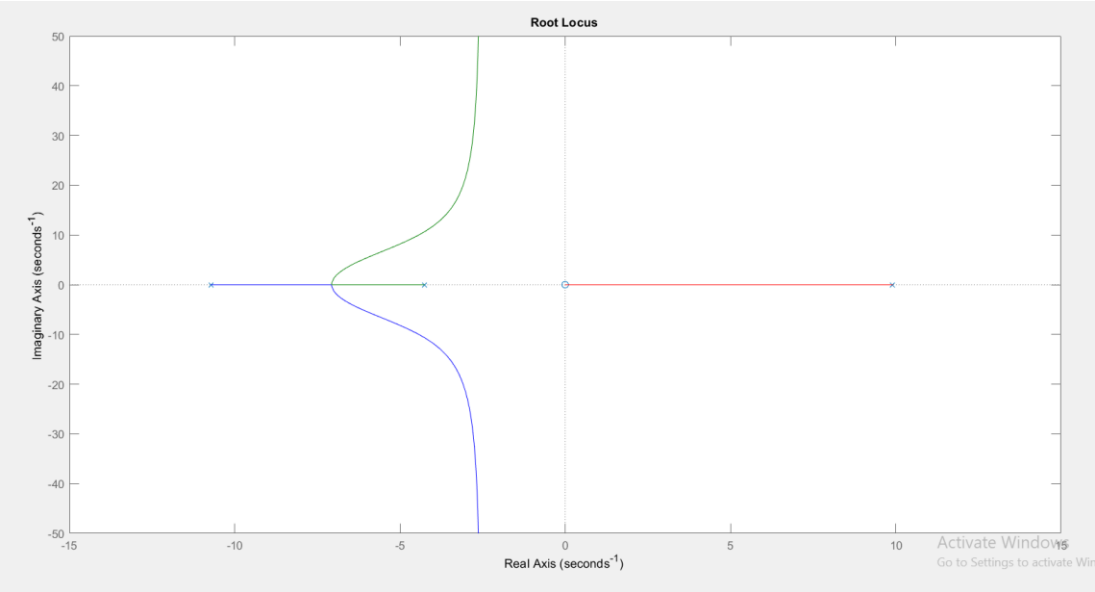

Figure 5*

* Figure 5* is obtained from the MATLAB code in the attached file root_locus.m.

The Fabricated Model has 4 poles at [0, 9.91, -10.72, -4.26] and two zeros at [0, 0].

6.2 Proportional Controller

We will use proportional control using state-space, where the controller coefficients are obtained from the LQR method with cost functions and weights (R) and (Q). With this method, we obtain 4 coefficients for the controller, each corresponding to its respective state variable $\left((\theta, \dot{\theta}, \beta, \dot{\beta})\right)$. In state feedback, we feedback the variables and compare them with their desired values, then subtract them. Afterwards, we multiply this difference by the control coefficients. This process generates the necessary control force signal to control the system or neutralize disturbances.

To visualize the position and response of the system in MATLAB, we use two functions: `nonlinear_sys.m` and `print_sys.m`, which encompass the characteristic functions of the nonlinear system. The files are attached. You can also refer to the MATLAB file `system.m` to see how the pendulum moves and execute it.

The folloing figures demonstrate this design.

- Root Locus Analysis of the closed-loop poles of the system with state proportional control are depicted Figure 6.

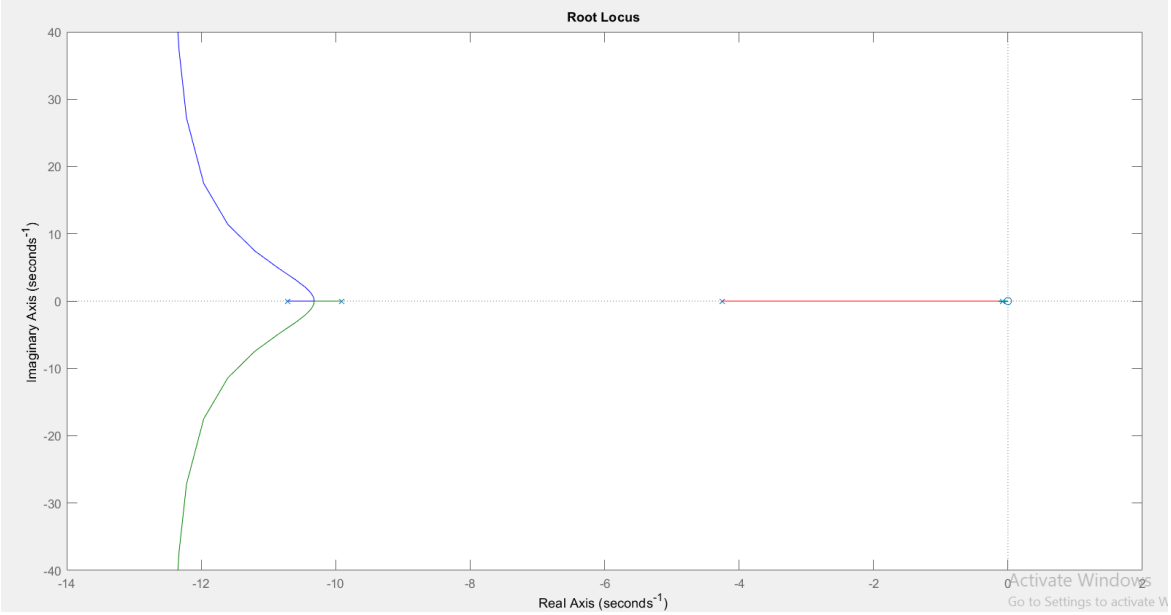

Figure 6

- The values of the state variables during control with the initial state $\theta = \pi + 0.5\ (rad)$ can be observed in Figure 7.

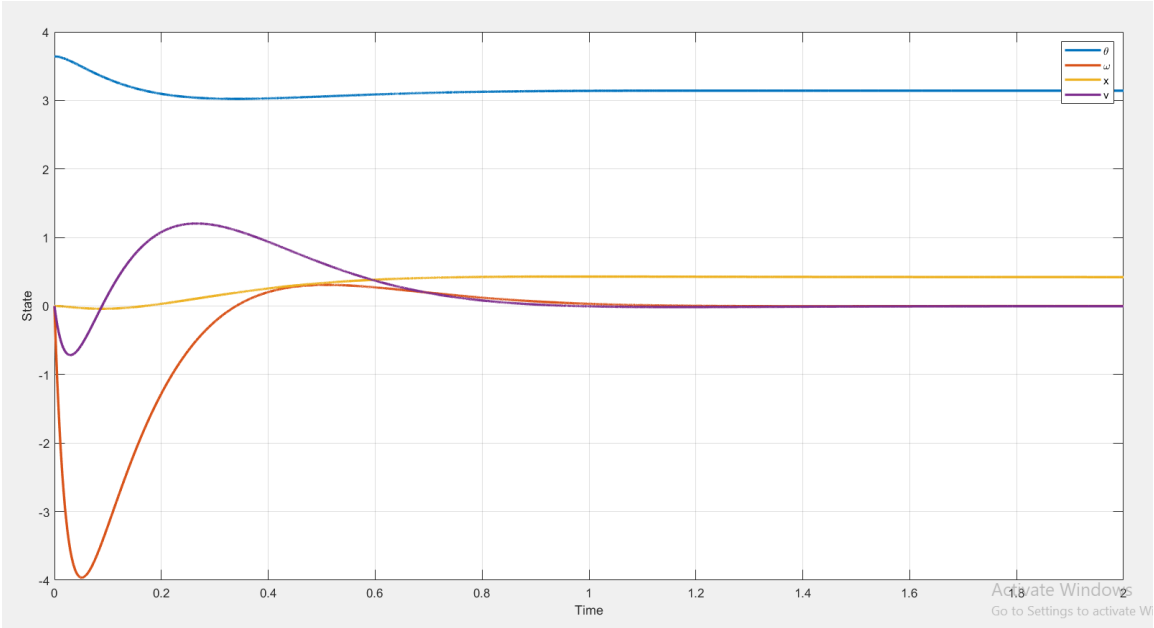

Figure 7

➢ The system's response to a unit step input can be observed in Figure 7.

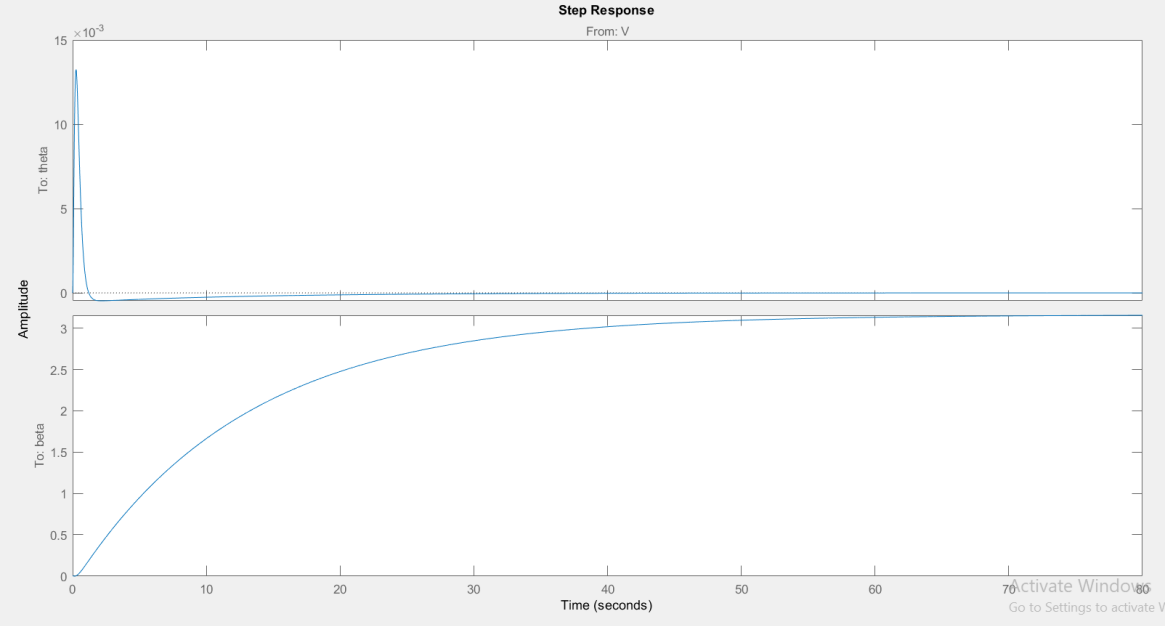

Figure 8

➢ The system's response to a unit impulse can be observed inFigure 9.

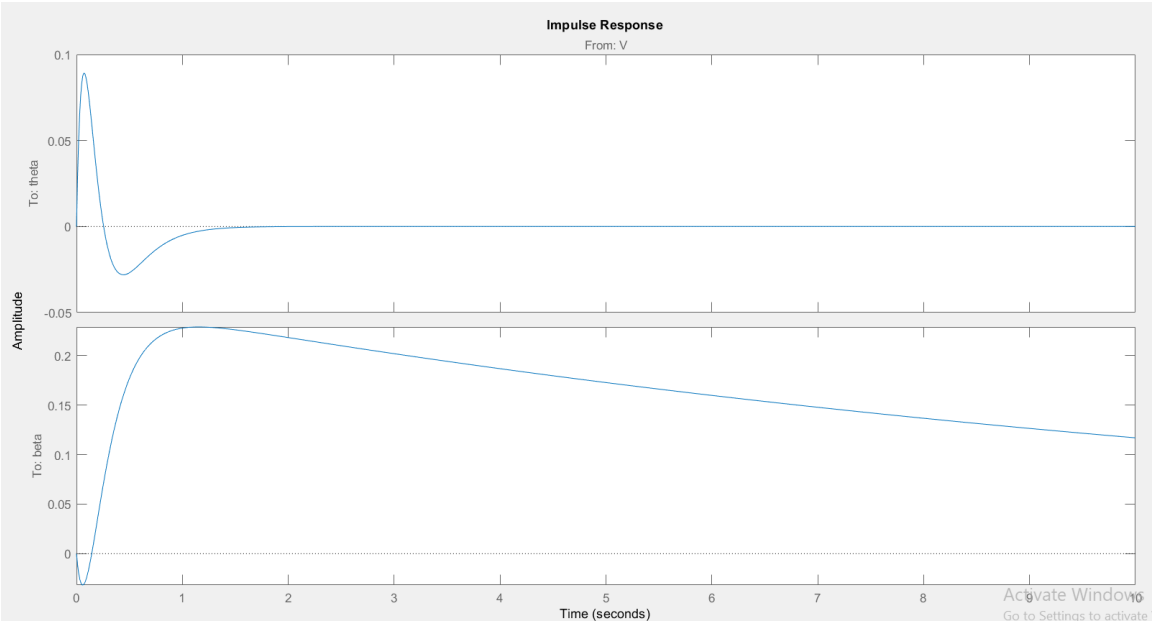

Figure 9

6.3 PID Controller Design

In this method, we use the **pidtune** command in MATLAB to obtain the PID controller coefficients. This command provides us with three optimized coefficients Kp, Ki, and Kd for the controller.

- The root locus plot illustrating the geometric locations of the closed-loop poles with the PID controller can be observed in Figure 10.

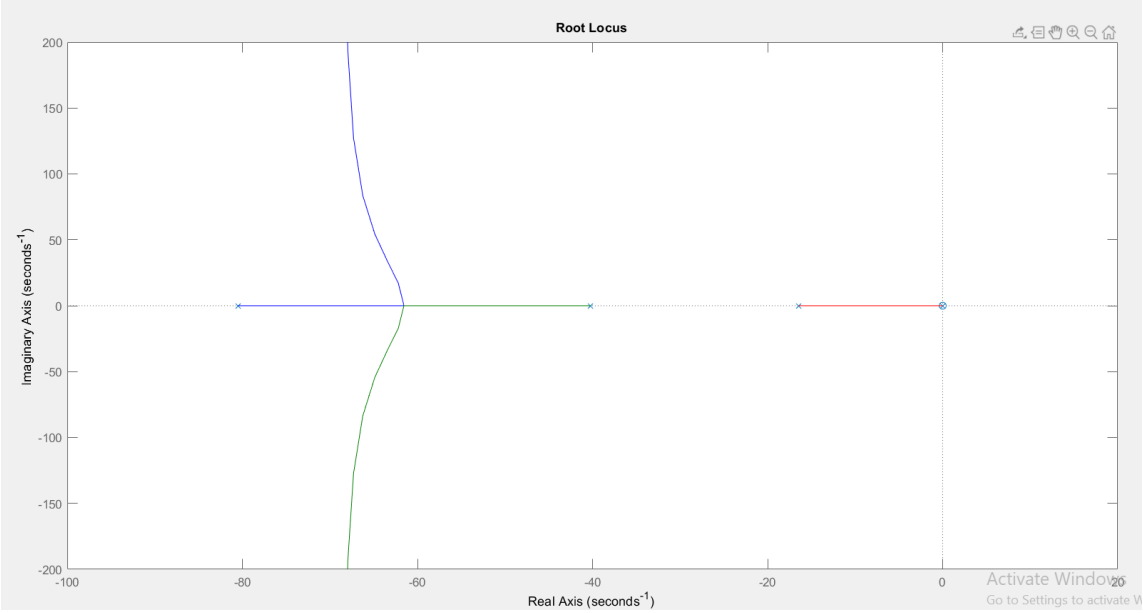

Figure 10

- The system's response to a unit step input can be observed in Figure 11.

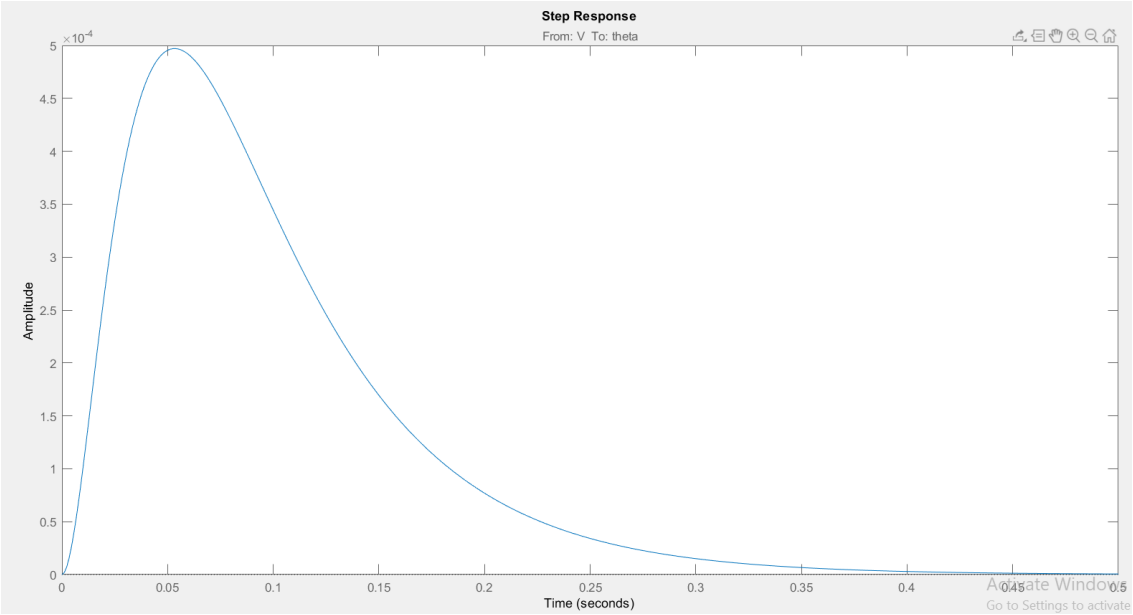

Figure 11

- The system's response to a unit impulse can be observed in Figure 12.

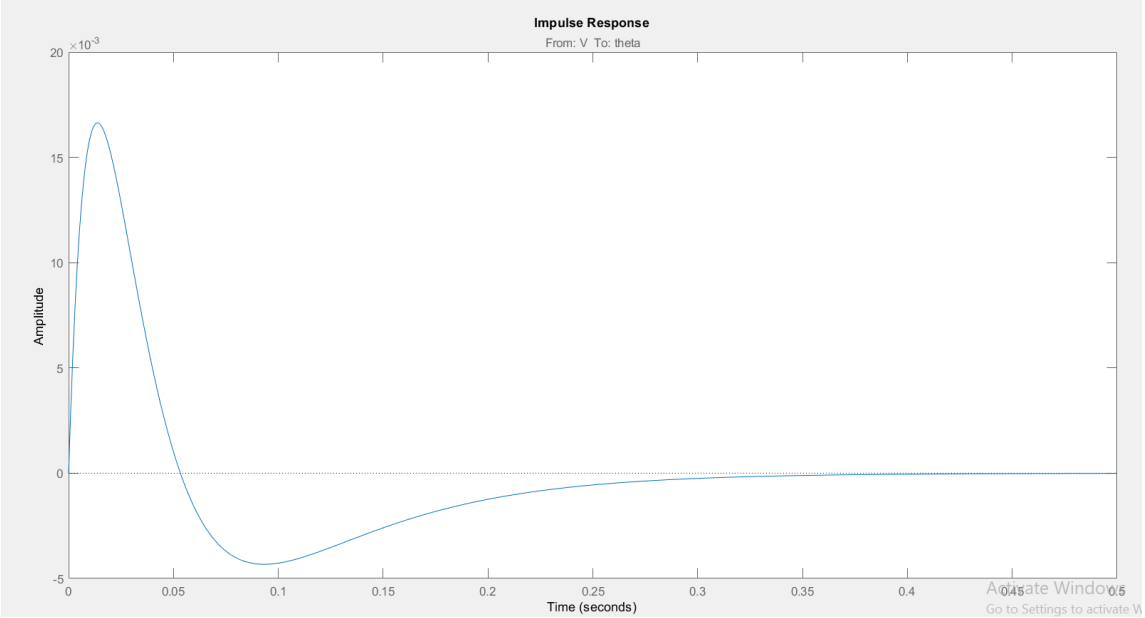

Figure 12

7 Conclusion

Initially, we modeled the satellite carrier into a more accessible model, like an inverted pendulum. Then, using the dynamics equations of the pendulum, we simulated the applied force through angle adjustment of the thrusters using a rod and a DC motor model. We extracted the equations for both through dynamics and electrical machinery. Now, we have access to the controller's tuning function, namely the motor voltage, and the output, namely the pendulum angle. We then proceed to design the controller. The controller we use is PID, and we obtained its three coefficients using the `pidtune` command in MATLAB. Now, it's time to apply this controller to our constructed dynamic model. We apply the three controller coefficients using the PID Arduino library to the Mega Arduino controller, and we observe that the controller effectively maintains the carrier upright. In this project, we have achieved our desired outcome, which is reproducing the response given to the carrier angle control.

8 References

[1]Steven L. Brunton, J. Nathan Kutz - Data-Driven Science and Engineering_ Machine Learning, Dynamical Systems, and Control-Cambridge University Press (2019)

[2] Dynamics / J. L. Meriam, L. G. Kraige, Virginia Polytechnic Institute and State University, J. N. Bolton, Bluefi eld State College.—Eighth edition.

[3] Modern Operating Systems By Andrew S. Tanenbaum(5th Edition) by Katsuhiko Ogata, Andrew S. TanenbaumHardcover, 912 Pages, Published 2009 by Prentice Hall

[4] Advanced Engineering Mathematics, 10th Edition by ERWIN KREYSZIG, HERBERT KREYSZIG,New York, New York,EDWARD J. NORMINTON,Associate Professor of Mathematics, Carleton UniversityProfessor of Mathematics,Ohio State UniversityColumbus , JOHN WILEY & SONS, INC.

[5] James K. Roberge. The mechanical seal. Bachelor's thesis, Massachusetts Institute of Technology, May 1960.

[6] Differential Equations", Dr. Masoud Nikoukar, Azadeh Publications, AmirKabir University of Technology.

9 Appendix

1) \Arduino code\inverted_pendulum.ino
2) \Matlab code\noneliniar_sys.m
3) \Matlab code\print_sys.m
4) \Matlab code\root_locus.m
5) \Matlab code\sys2.m
6) \Matlab code\system.m
7) \Matlab code\system_pid.m