**EE-361: Principles of Feedback Control – Fall 2021**

**Lab Project**

**Twin Rotor MIMO System**

**Hamza Irfan, Yabudullah Bakhtiar, Hasan Tariq, & Sadaf Shaikh**

1. **Introduction**
2. Problem Statement

We aim to model and implement the Twin-Rotor-MIMO Systems (TRMS). Although the original TRMS setup involves free rotation in both, the vertical plane (pitch) and the horizontal plane (yaw), the beam in our model will only be rotating in the vertical plane. We do plan, however, to eventually reach two degrees of freedom of motion for the beam. The plant will stabilize and reposition rotors, each situated at the far end of the beam, using a DC motor-based servomechanism.

1. Application

The Twin Rotor MIMO System (TRMS) resembles behavior of a helicopter. From a control perspective it is used to model higher order nonlinear systems with coupling between the rotors. It is used in aerodynamics for modeling and stability of a system. If either of the rotor faces external disturbance, the controller automatically adjusts the thrust of the rotors to balance the beam (system). The system is designed to be a simpler version of a helicopter or a bi-copter depending how the angle of attacks of the two rotors are adjusted.

1. Motivation

The motivation behind doing this project is to apply the theoretical knowledge in classes to real-life applications. We chose this topic specifically because of the complex modelling that this system offers to further learn about significance of modelling in a system. Furthermore, another reason to work on this project was to learn about controllers how they operate in real time and impact of a controller in feedback control system.

1. **Methods and Materials**
2. Block Diagram

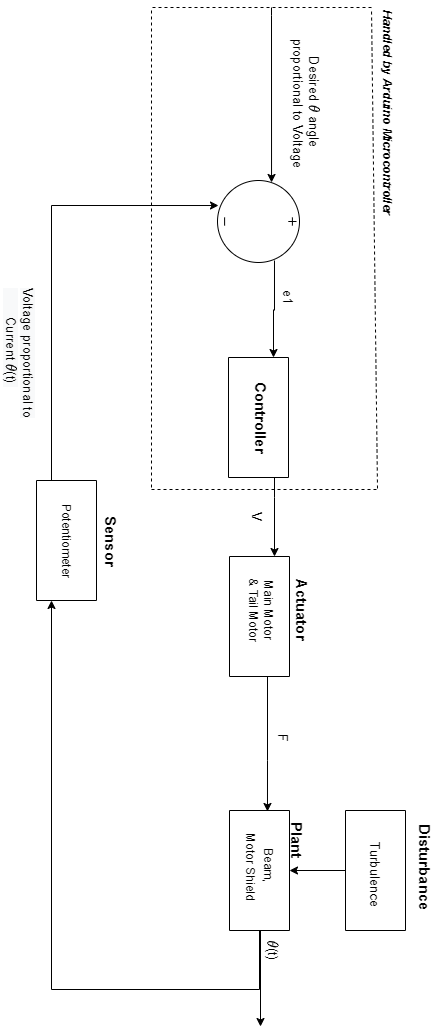


Figure 1: Block Diagram

The block diagram above represents our system. The aim of the project is to control the position of the beam and balance it at the assigned set point. The sensor of the system is a potentiometer that takes in angular displacement and converts it into output voltage which is then sent to the controller that checks it against the input signal and minimizes the error to 0. There are two motors in our system - tail motor and the main motor. These motors are the actuating mechanisms of our system. They are controlled by the controller and turned on whenever the beam is displaced from its set point position to bring it back to its desired position. The plant comprises of the beam fitted with motors on both ends of the beam which is held by a vertical support from the center. The possible source of disturbance in our system can be turbulence from wind or external force which causes movement of the beam.

In our project we built the mathematical models for the sensor, plant and motors using grey-box modelling, first principles, and grey-box modelling, respectively. We built a relationship between the input voltage provided to the motors and the thrust generated due to the propeller rotation to determine the order of the motor system. We used MPU-9250 to measure orientation and displacement of the beam. We used Kalman filter to sensor fuse the readings from the three-axis accelerometer and the three-axis gyroscope for accurate sensor readings.

We determined the motor resistance in the following way:

1. Approach
2. Place one terminal of the multi-meter on any one of the terminals of the motor and place the other terminal of the multi-meter on the other terminal of the motor.
3. Switch the power supply of the multi-meter.
4. Press the “Resistance (Ω)” parameter on the multi-meter.
5. Record the first 5 fluctuating readings.
6. Take average to obtain the value of Rm.

Single number of trials were made. Within that one trial, an average reading was chosen for the resistance of the armature coil of the motor.

1. Equipment & Experimental Setup

Equipment: Multi-meter, probes, breadboard, DC motor under experiment

Experimental setup

1. Connect both terminals of the motor with the breadboard, each connected on different rows.
2. Connect probes to the terminals of the motor.
3. Connect the other ends of the probes to the multimeter.
4. Data Collection

Following is the data collected to find the armature resistance of the armature coil:

Table 1: Armature Resistance Readings

|  |  |
| --- | --- |
| S # | Resistance (Ω) |
| 1 | 386.33 |
| 2 | 384.41 |
| 3 | 378.26 |
| 4 | 378.90 |
| 5 | 378.77 |
| 6 | 379.0 |

Rm = = = 380.945 Ω

1. Mathematical Model
2. Actuator

Our project consists of two armature controlled DC motors. Taking the motor as frame of reference, the transfer function of the main motor and the tail motor can be derived in the following way:

Applying KVL around the loop in the armature circuit,

( 1)

Assuming negligible motor inductance ( and knowing that,

( 2)

And,

( 3)

And,

( 4)

where, and are contants.

Now equating equation (3) and (4) to give,

( 5)

Substituting the expression for and in equation (1),

( 6)

Taking Laplace transform on both sides of equation (6),

( 7)

where,

Representing the DC motor model in terms of lumped parameters in the transfer function ,

( 8)

( 9)

and are the resistance and self-inductance of armature coil inside and, and are rotor moment of inertia, moment of inertia of hub (a cylindrical metal disc used to mount the propeller) and propeller moment of inertia respectively.

( 10)

1. Plant

The following nomenclature will be used to model the plant:

The mass of the main DC-motor with main rotor

The mass of the main shield

The mass of the main part of the beam

The mass of the tail motor with tail rotor

The mass of the tail shield

The mass of the tail part of the beam

The mass of the counterweight

The mass of the counterweight beam

Length of the main beam

Length of the tail beam

Gravitational acceleration

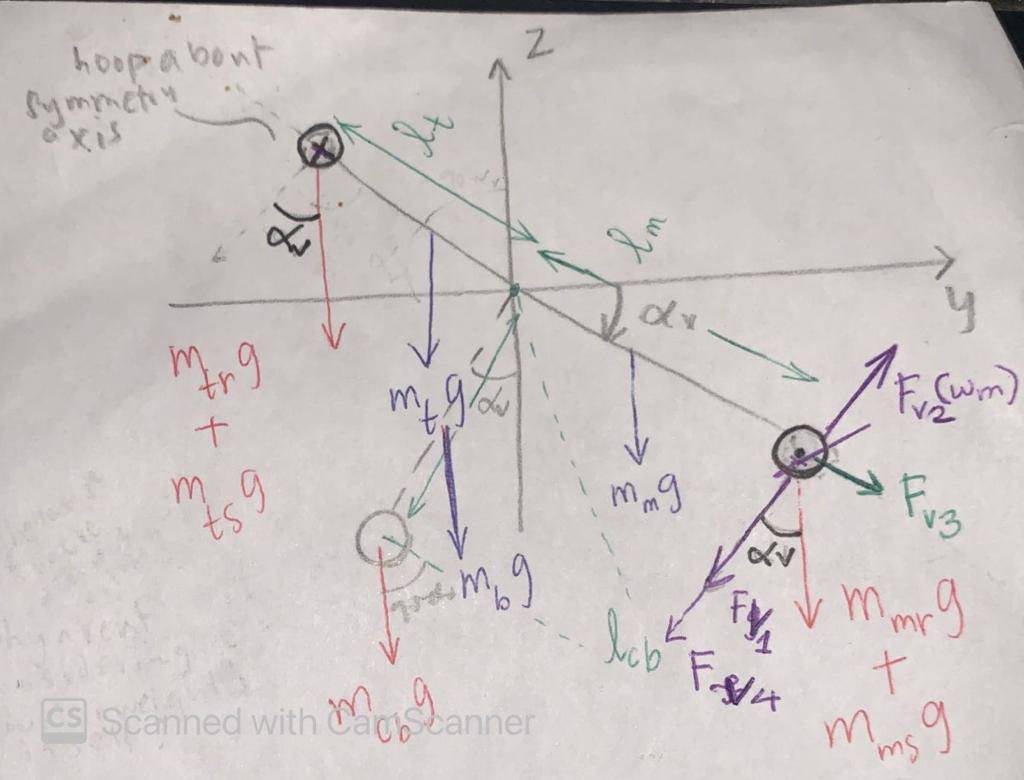


Figure 2: Schematic Front view of TRMS – Fv1(component of gravitational force), Fv2(Thrust), Fv3, Fv4

The non-linear model of the plant can be derived in the following way:

1. Moments in the Vertical Plane

According to Newton’s second law of motion,

( 11)

where, is the sum of components of moment of forces, is the sum of moments of inertia relative to the horizontal axis (x axis) of individual parts of the plant, and is the pitch angle.

Then,

( 12)

( 13)

where, is the moment of inertia of the components on the tail side, is the moment of inertia of the components on the main side, and is the moment of inertia of the components on the pendulum side.

And,

( 14)

( 15)

( 16)

The moment of force due to gravitational forces on the plant is given by,

( 17)

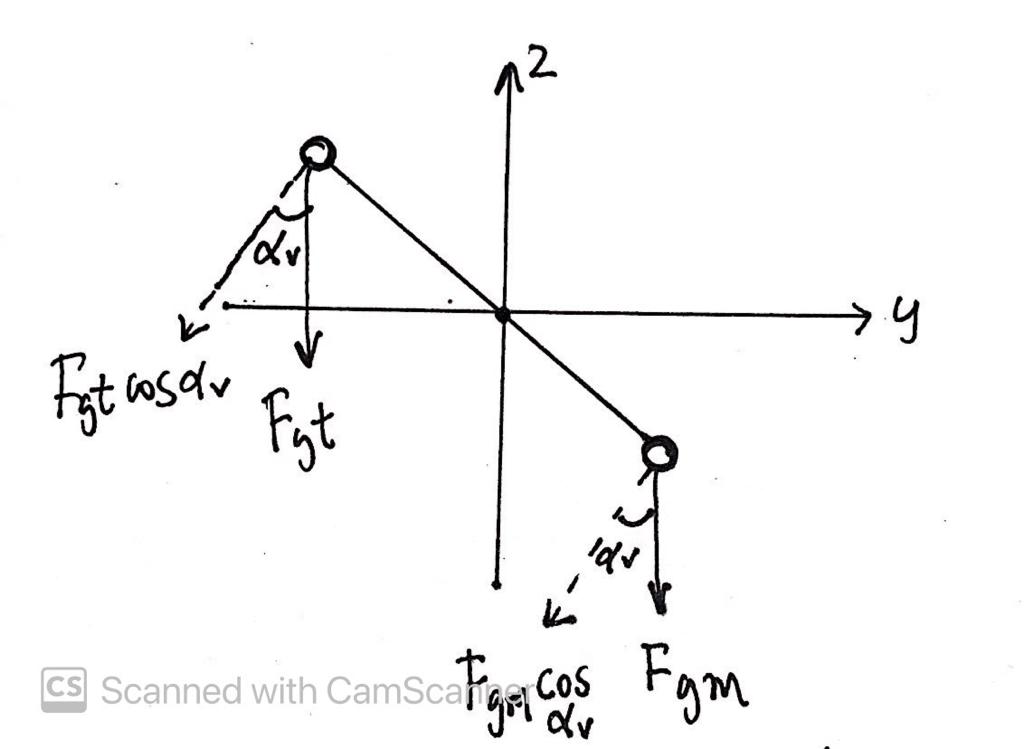


Figure 3: Direction of gravitational forces.

It can be noted that , , and are divided by the factor 2 because the gravitational forces act on center of masses and their center of masses exist on distances , , .

The moment of force developed by main motor’s thrust due to rotation of main propeller is given by,

( 18)

where, Fv2 is the propulsive force exerted by the main propeller and is a function of the angular velocity of the main motor, ωm.

Given that the beam is displaced at an angle αv in the vertical axis and rotates around the vertical axis with an angular velocity, Ωh, in the horizontal plane, then the centrifugal force due to the rotation in the horizontal plane for a mass, m, situated at a distance, x, from the pivot, is given by,

( 19)

And,

( 20)

The component of Fc acting the vertical axis is given by,

( 21)

Then, for any given center of mass, the centrifugal moment is given by substituting (17) in (16) and (16) in (18) to get,

( 22)

The total centrifugal moment corresponding to the motion of the beam around the vertical axis then becomes,

( 23)

where,

( 24)

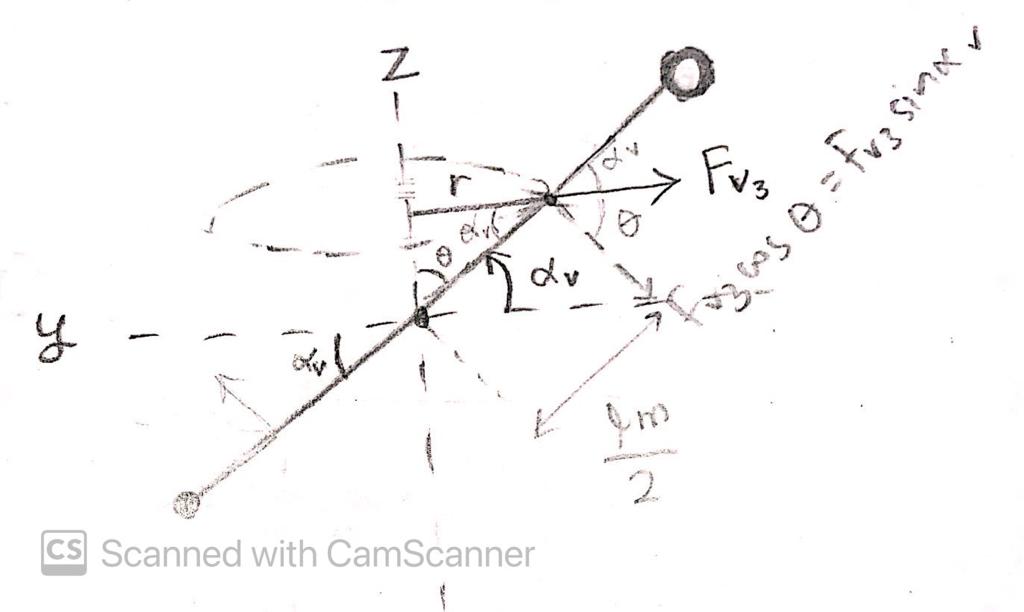


Figure 4: Centrifugal Moment due to rotation about horizontal plane

The moment due to frictional force experienced by main motor is given by,

( 25)

Where, v is the linear velocity of the main motor and its shield in the vertical axis and is the proportionality constant.

1. Moments in the Horizontal Plane

According to Newton’s second law of motion,

( 26)

where, is the sum of components of moment of forces in horizontal plane, is the sum of moments of inertia relative to the horizontal axis (x axis) of individual parts of the plant, and is the yaw angle.

Then,

( 27)

( 28)

where,

( 29)

( 30)

( 31)

The moment of propulsive force (Thrust) due to tail rotor and propeller can be expressed as,

( 32)

where, given that the beam is displaced at an angle αv in the vertical axis and rotates around the horizontal axis, the component of the thrust experienced by the tail motor in the horizontal plane depends on the pitch angle.

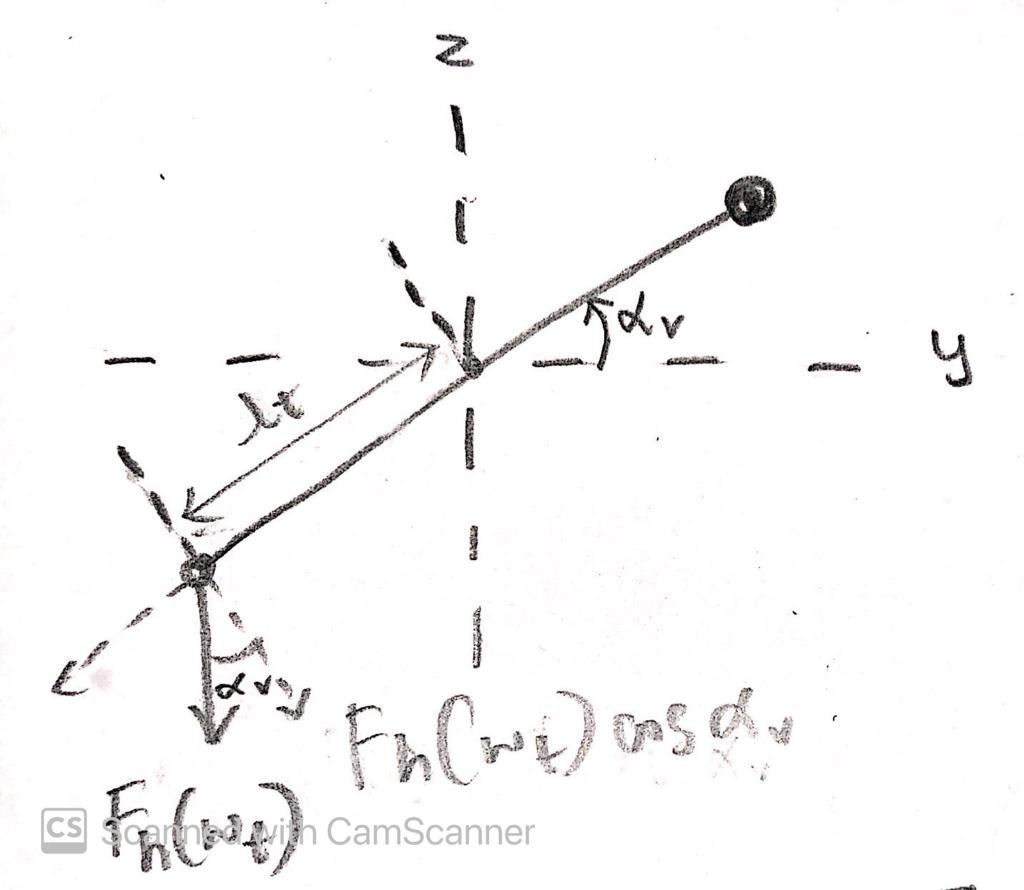


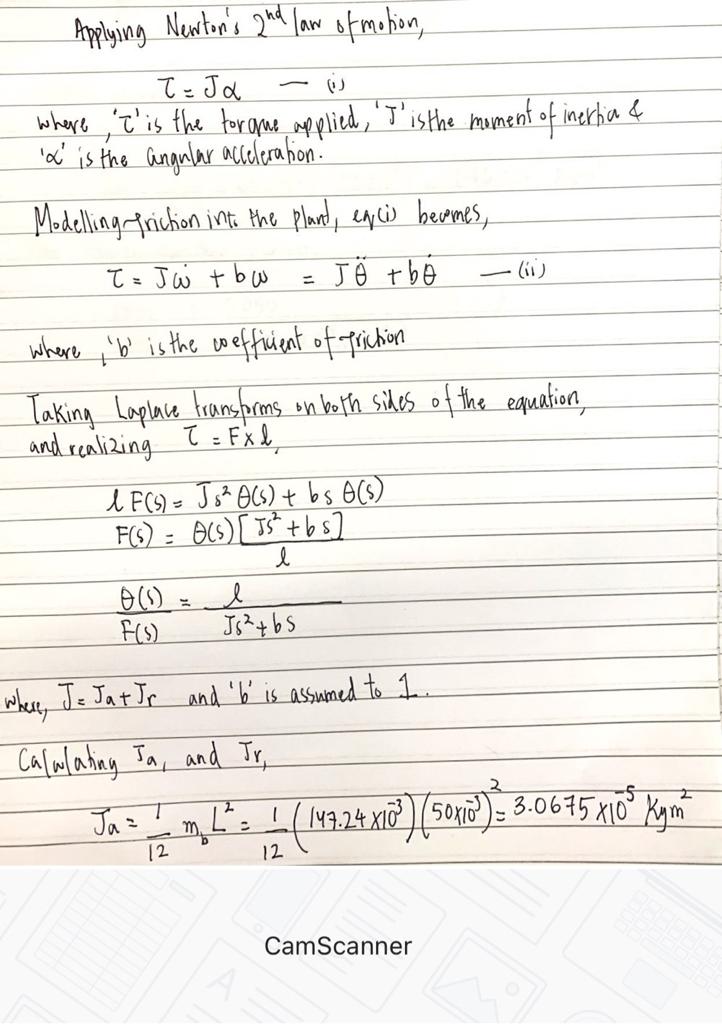
Figure 5: Moment of Force due to Thrust exerted by Tail Motor

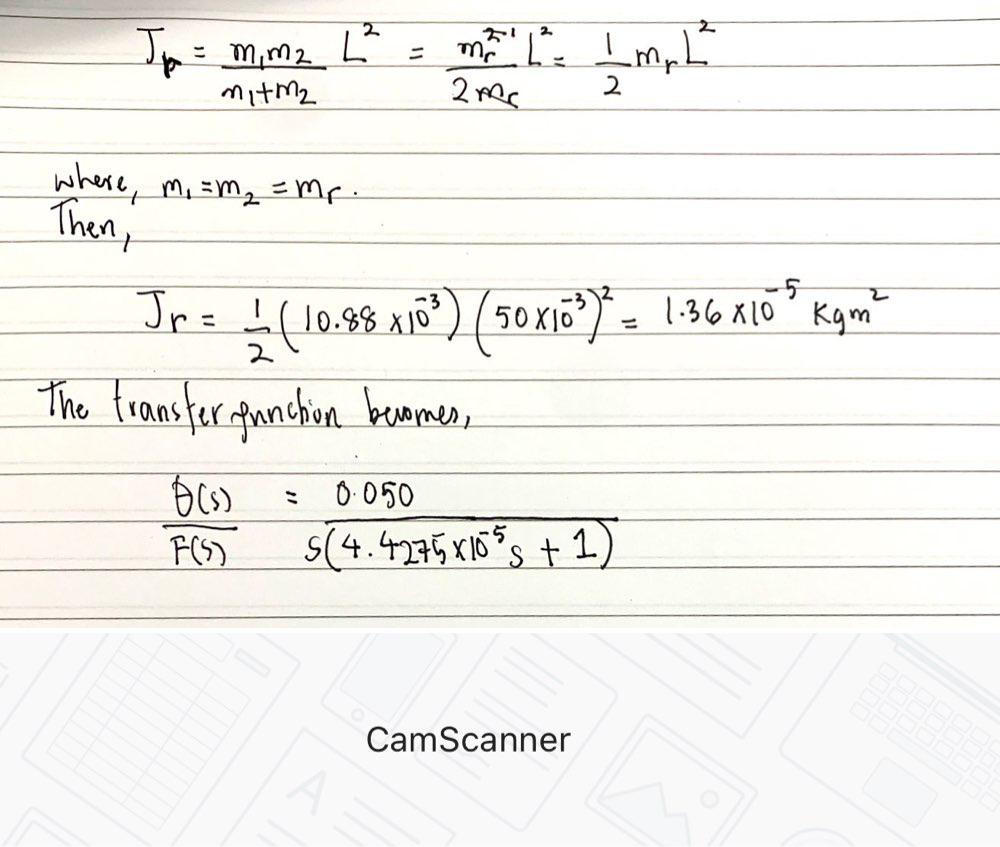
The moment of frictional force experienced by tail motor, shield, and propeller is given by,

( 33)

Where, v is the linear velocity of the tail motor and its shield in the horizontal axis and is the proportionality constant.

Alternatively,





Our final mathematical model, then, becomes,

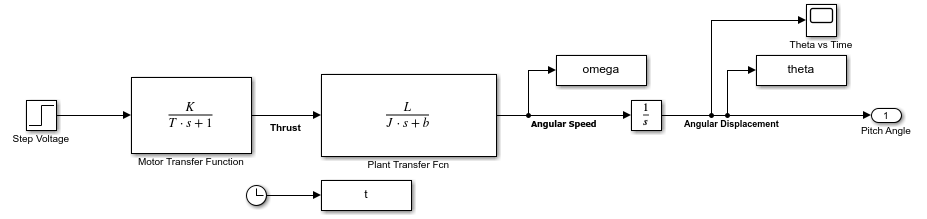
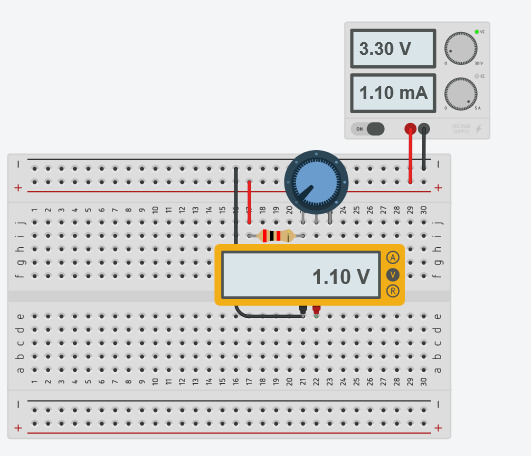


Figure 6: Final Mathematical Model

1. Sensors

Sensors play a key role in any control and feedback systems as they provide the system with information to make required adjustments to get desired output. Our desired output is the angular position of the beam, more specifically the pitch and the yaw angle. For sensing the angular position accurately, we have worked with potentiometers and Inertial Measurement Unit (IMU). For pitch control, we tested different type of potentiometers and settled with linear wire-wound single-turn 1k potentiometer. A linear potentiometer in contrast to audio potentiometers has a linear relationship with position instead of a logarithmic relation. We only needed a maximum of 180° freedom which is sufficiently available in the 280° single-turn pot. We use wire-wound instead of ceramic because it has a higher power rating which is good because we have disturbance from the motor and it has a relative high impedance which gives a smooth and stable output. Choosing the rating of the potentiometer is equally important and we went with the 1k version of the sensor since the ADC we use to read the sensor output recommends a low resistance sensor to be used. The ADC we use is the INTERNAL 10-bit ADC built-in to the Arduino UNO development board which has an AVR Atmega328p microcontroller. The Vref of the ADC is 1.1V and we have 3.3V available on the microcontroller so we have to use a voltage divider circuit to excite the potentiometer with 1.1V. Since we are using a 1K potentiometer, we need a 2K resistor to make the appropriate voltage divider as demonstrated in the circuit below.



Despite taking all the steps mentioned above, the readings still carried a lot of noise. This was catered by making a low pass filter using a 100nF capacitor as specified in the atmega328p datasheet in the ADC section. On the software we also use a moving average algorithm which smooths the readings based on the last 10 readings which almost diminishes any noise and the output obtained is linear and stable as shown in the figures below.

For Yaw control we repeat the same process for selection but use a multiturn potentiometer instead which has a total of 10 turns with a 10KΩ. We use twisted shielded pair of wires for both the potentiometers to minimize noise by signal attenuation.

Methodology to Obtain Measured Results:

1. Used an ADC to measure voltage outputs.
2. The potentiometer is attached to an angle scale and turned.
3. At intervals of 20° the value of ADC pin is noted.
4. Since an ADC has 10 pins, so it has a total of 2^10 = 1024 bits.
5. The value of bits is calculated at 20° intervals.
6. The reference voltage applied to the potentiometer is 3.3V.
7. The voltage at each pin is determined by dividing the reference voltage by value of total value of bits and then multiply by value of bits calculated at that point and applying moving average for smoothing.
8. Now we have a relation between the angle and voltage output.
9. A graph is plotted using excel and a mathematical equation is derived.
10. The equation is modeled using block diagrams in Simulink and the same angle input is provided to obtain theoretical results.
11. The experimental results from ADC and theoretical results from MATLAB are plotted, the difference in values is the error in values.

Sources of Error in Results:

The following are the possible reasons behind differences in measured and simulated results:

1. Internal resistance of potentiometers.
2. Accuracy of ADC bits.
3. Measurement of angles of potentiometer turning.
4. Recording the results too quickly without letting the readings to stabilize.
5. AC/DC coupling because of the back EMF generated by the DC motors.

Transfer Function:

The output voltage of the potentiometer will be proportional to the shaft position in case of the rotary motion.

Here is the proportional constant. For N turn rotary potentiometer the total displacement covered by the variable arm is radians. Thus, the proportional constant is given by

Where is the number of turns on the potentiometer and is the reference voltage applied.

Thus, the general transfer function for potentiometer is:

In case of Pitch Potentiometer, N = 1

In case of Yaw Potentiometer, N = 10

Measured Values and Derived Relationship between and

For Pitch Potentiometer:

1. Table of values
2. Graph of Theta vs ADC Expected and ADC Measured
3. Graph of Theta vs Voltage

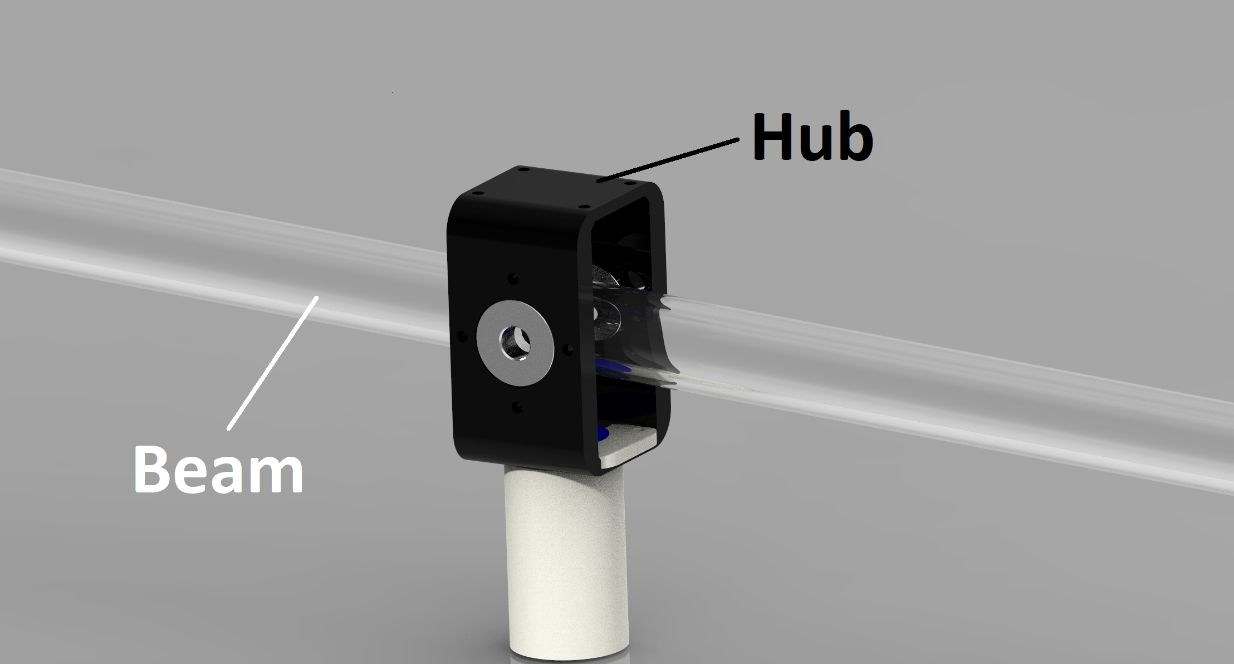
We also used MPU-9250 which is a 9-axis Inertial Measurement Unit. For Pitch we use the sensor reading of Y-axis Gyroscope angular velocity and X-axis, Z-axis accelerometer values with respect to gravity. The angular displacement, although precise, has drift because of the lack of reference and the angular position being the integral of the sensed reading. The accelerometer on the other hand gives a very accurate value but has noise being in reference to gravity and having the positional displacement also affecting the angular displacement obtained. We fuse both the values of accelerometer and gyroscope to obtain a precise and accurate reading of the pitch and yaw angle. First trying a simple complementary filter, we settled with a Kalmaan implementation for sensor fusion which gave us the sensor readings with an accuracy of ±0.5°.

1. Process of Plant Design and Fabrication

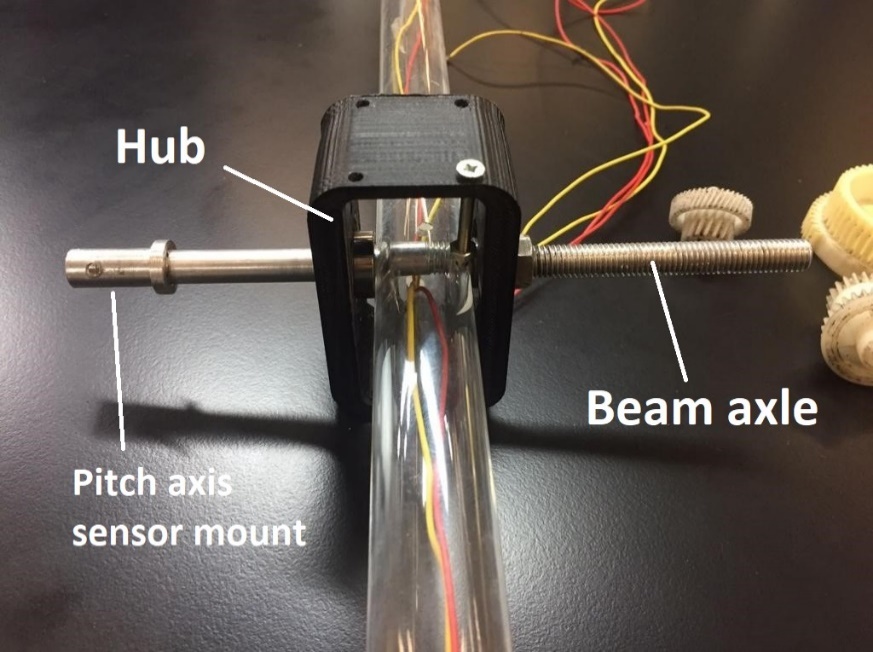
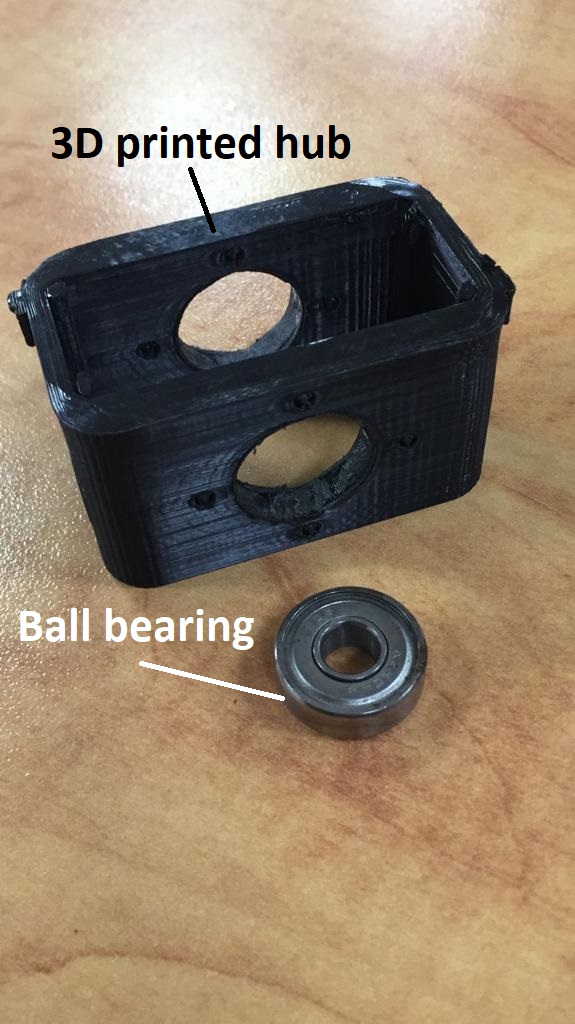
The plant was designed keeping the following constraints in mind:

1. Bearings should be used wherever applicable to ensure friction forces are minimum.
2. Yaw rotation should ideally be free from wire entanglement to allow greater number of turns for angle correction.
3. Actuator motor angles should be adjustable to convert plant from having 1-axis to 2-axis freedom of movement.

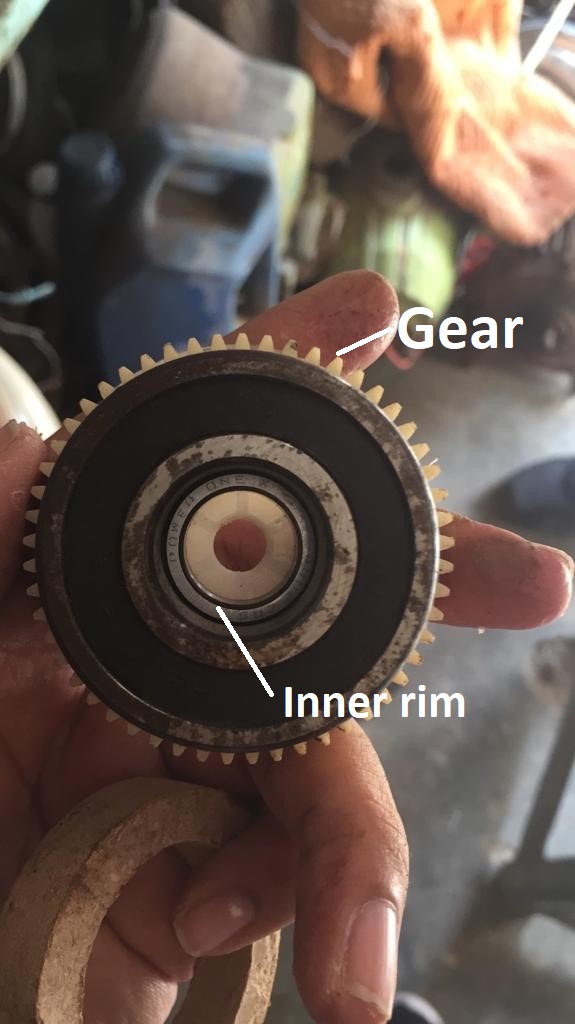
The plant was first designed on a CAD program named PTC Creo following the reference images collected from various online sources and research papers.



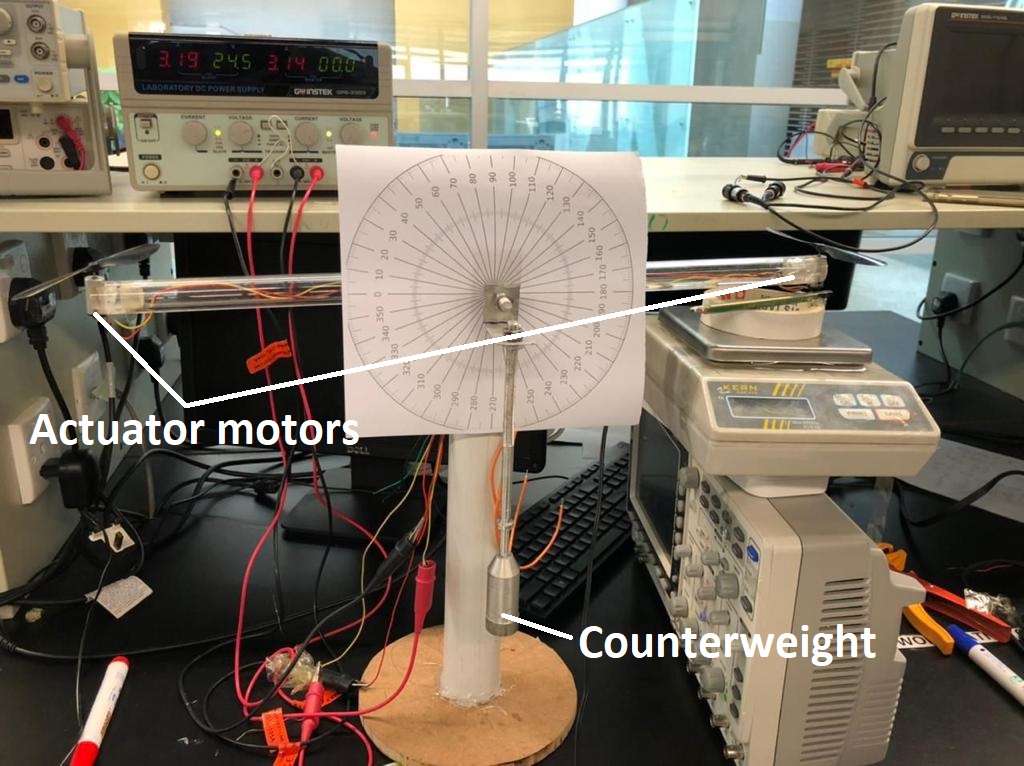
The design initially included several 3D printed parts but was later modified to only use a single 3D printed part called the hub. The hub connects the beam and the main pillar together allowing 2 axis of movement. The rest of the mechanism has been fabricated using different hardware techniques.



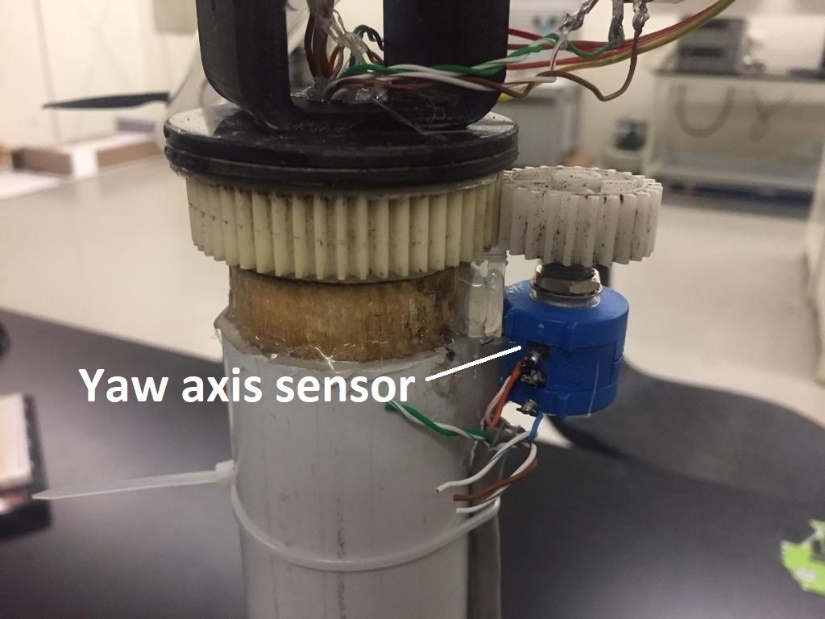
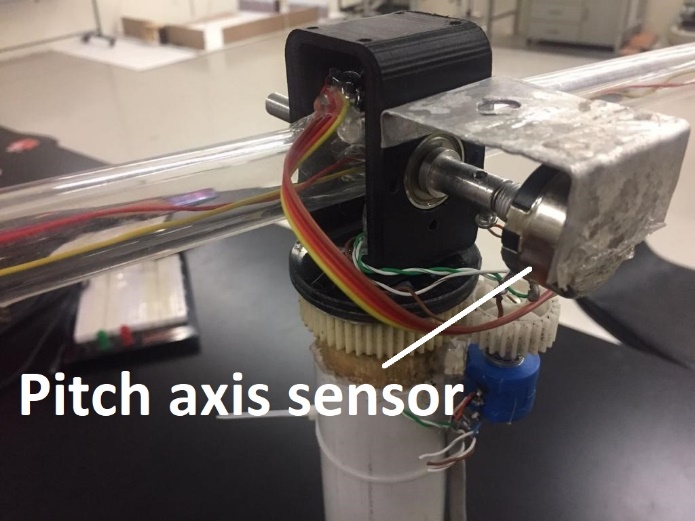
The beam is connected to the hub with an 8mm axle that attaches to the hub through 2 ball bearings to reduce rotational friction in the pitch axis. The motor wires travel through the acrylic tubing beam and exit through a center hole in the bottom of the hub.



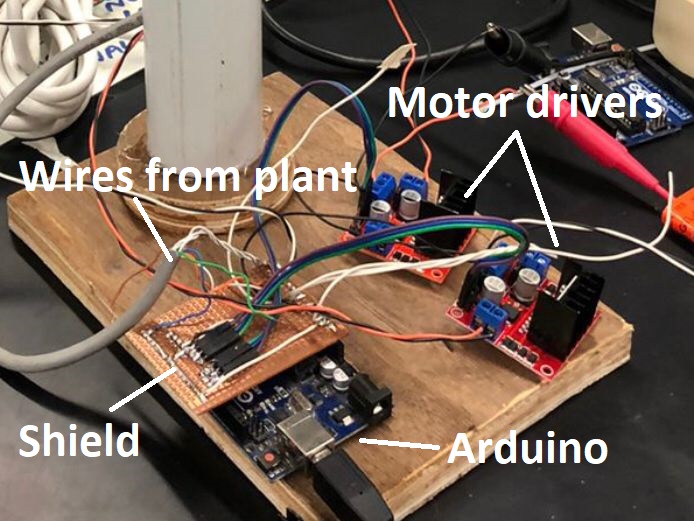
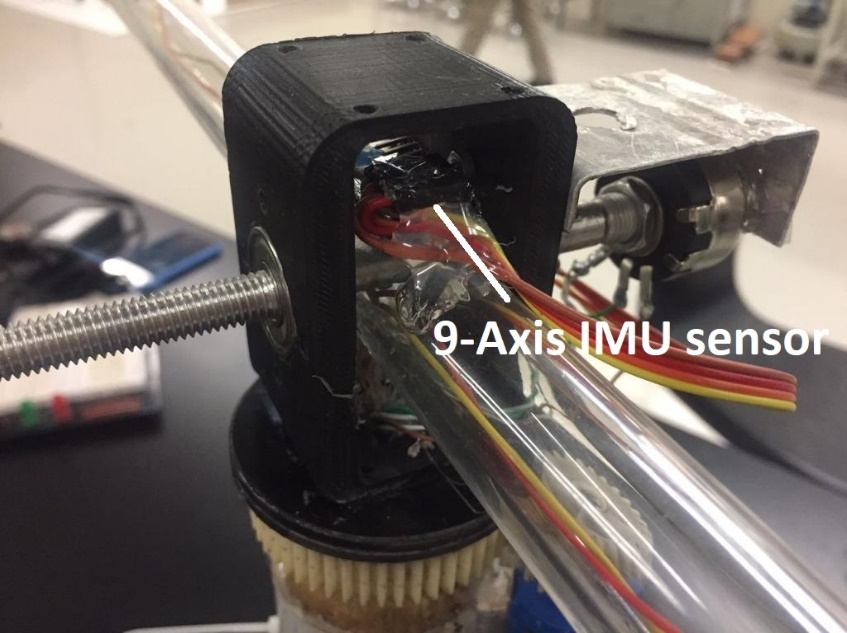
The hub is attached to the main pillar with a bigger ball bearing with an attached gear. The inner rim of the bearing is attached to the main pillar through a wooden mount. A smaller hole in the center of the wooden mount allows all of the sensor and motor wires from the hub to pass through to the main pillar. This keeps the wires from getting entangled due to rotation in yaw axis.



Two 3.7v DC motors with propellers are attached at the ends of the beam that act as actuators. The motors are bi-directional and can provide both upwards and downwards thrust. The counterweight is attached to the main axle with an L bracket which is used to balance the plant in the case when unbalanced actuators and beam lengths are used on either side. Since our plant is balanced on either side the counterweight was later removed.



The pitch rotation sensor (black potentiometer) is attached to the hub with an aluminum mount and is driven by the main beam axle that connects the beam to the hub to measure the angle of rotation of the beam in the pitch axis. The axle rotates with the beam via a screw acting as a pin which is inserted after attaching the beam. The yaw rotation sensor (blue potentiometer) is attached to the main pillar and is driven by the bigger ball bearing gear to measure angle of rotation of the hub and beam in the yaw axis.



A digital 9-Axis IMU sensor is also attached on the center of the beam which includes an accelerometer, magnetometer and gyroscope which is interfaced directly to an Arduino Uno. The base of the plant has all the electronics attached which include the motor drivers for driving the actuator motors and an Arduino Uno to implement the digital controller logic. A custom Arduino shield has been developed that has all the sensor wires directly soldered to it. This allows all of the sensor and actuator hardware to be completely disconnected if needed to change or debug the Arduino Uno microcontroller.

1. **Approach for Controller Design**

**Control Code Logic**

|  |
| --- |
| void loop() { |
|  | readMPU(); |
|  | Serial.print("Angle: ");Serial.print(kalAngleX);Serial.print("\t\t");Serial.print("PID: "); |
|  |  |
|  | //control logic |
|  | previous\_error = error; |
|  | error = kalAngleX - desired\_angle; //kalAngleX comes from MPU, desired angle is set to zero degrees |
|  | pid\_p = kp\*error; |
|  |  |
|  | if(-3 <error <3) |
|  | { |
|  | pid\_i = pid\_i+(ki\*error); |
|  | } |
|  |  |
|  | pid\_d = kd\*((error - previous\_error)/dt); |
|  |  |
|  | PID = pid\_p + pid\_i + pid\_d; |
|  | if (PID>35) PID = 35; |
|  | else if (PID<-35) PID = -35; |
|  | Serial.print(PID); |
|  | // PID = 30; |
|  | // Serial.print(timer);Serial.print(",");Serial.println(kalAngleX); |
|  | actuate(); |
|  | } |

**Code for Actuation**

|  |
| --- |
| void actuate() |
|  | { |
|  | motorSpeedA = map(abs(PID), 0, 35, 0, 255); |
|  | motorSpeedB = map(abs(PID), 0, 35, 0, 255); |
|  | Serial.print("\t\tmotor speed ");Serial.println(motorSpeedA); |
|  | if (PID < 0) //clockwise angular rotation |
|  | { |
|  | thrustAUp(); //left motor produces thrust upwards |
|  | thrustBDown();//right motor produces thrust downwards |
|  | } |
|  | else if (PID > 0) //anticlockwise angular rotation |
|  | { |
|  | thrustBUp();//right motor produces thrust upwards |
|  | thrustADown();//left motor produces thrust downwards |
|  | } |
|  | } |

The inclination angle of the beam is controlled by the PID controller. The error generated by the difference between the desired value of pitch angle and the angle from the IMU board is determined. The desired value of the inclination is set to 0 degrees and the actual angle of the beam is received from the IMU board. The total output of the PID controller is the sum of algebraic expressions consisting of proportional, integrative, and derivative constants. First, keeping kp=1 and ki=0, kd=0 gives us the error itself. This will cause the beam to oscillate around 0 degrees since the values of PID causes error to increase proportionally. By trial and testing, the values of kp, ki, and kd were set such that the beam swiftly restores its desired value of pitch angle. The values of kp, ki, and kd that gave a swift and a desired response were 1.5, 0.004, and 0.01, respectively.

1. **Simulation & Experimental Results**
2. Sensing Data

Table 2 : Analog to Digital Mapping

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.No. | Actual ADC | Angle | ADC | Expected |
| 1 | 4095 | 0 | 1 | 0 |
| 2 | 3736 | 360 | 360 | 409.6 |
| 3 | 3155 | 720 | 941 | 819.2 |
| 4 | 2707 | 1080 | 1389 | 1228.8 |
| 5 | 2301 | 1440 | 1795 | 1638.4 |
| 6 | 1895 | 1800 | 2201 | 2048 |
| 7 | 1485 | 2160 | 2611 | 2457.6 |
| 8 | 1079 | 2520 | 3017 | 2867.2 |
| 9 | 662 | 2880 | 3434 | 3276.8 |
| 10 | 291 | 3240 | 3805 | 3686.4 |
| 11 | 0 | 3600 | 4096 | 4096 |

Figure 7: Graph for Analog to Digital Mapping for Sensors

The graph (figure 7) shows that the expected and ADC voltage values are approximately linear. However, there is little discrepancy between the values. Otherwise both results exhibit a linear trend. It is safe to assume that there is a direct relationship between angular displacement and output voltage.

1. Actuation Data

Table 3: Actuation Data for Motors

|  |  |  |
| --- | --- | --- |
| Voltage (V) | Thrust(N) | Mass(g) |
| 1 | 0.000981 | 0.1 |
| 1.2 | 0.007848 | 0.8 |
| 1.4 | 0.020601 | 2.1 |
| 1.6 | 0.034335 | 3.5 |
| 2 | 0.059841 | 6.1 |
| 2.4 | 0.08829 | 9 |
| 2.8 | 0.109872 | 11.2 |
| 3.2 | 0.130473 | 13.3 |
| 3.4 | 0.148131 | 15.1 |
| 3.6 | 0.161865 | 16.5 |
| 3.7 | 0.162846 | 16.6 |
| 3.8 | 0.172656 | 17.6 |
| 4 | 0.17658 | 18 |
| 4.2 | 0.182466 | 18.6 |
| 4.3 | 0.193257 | 19.7 |

Figure 8: Graph of Input Voltage (V) and Thurst(N) Generated by Motors

The data in table 3 was collected by mounting a protractor around the pivot of the arm and placing a weighing machine directly below one of the motors. The arm was positioned to achieve its mean position. The other motor that did not have a weighing machine underneath it was given a constant input voltage. Any thrust generated by it would be translated into the reading displayed on the weighing machine on the other end. Then, multiplying the mass reading with the gravitational constant and converting it into kg would give the thrust exerted by the motor at that input voltage.

The graph in figure 8 verifies a linear relationship between the input voltage and the thrust generated. Consequently, the model for the motor is assumed as linear and first order.

1. Validation of the Model with the Actual System Response

**Matlab Code**

L=0.050; % length of beam - 50mm, mass of beam = 147.24 g, mass of motor = 10.88 g

Ja = 1/12\*(147\*10^-3)\*(50\*10^-3)^2; % Moment of inertia of Arm/Beam; J = 1/12 \*m\_b\*L^2

Jr = 1/2\*0.01088\*(50\*10^-3)^2; % Moment of inertia of motors acting as point masses serperated by distance L J=1/2 \*m\_m\*L^2

J = Ja + Jr;

b =1; % coefficient of friction

K = 80;

T = 0.6; % tau

sim("Final\_model");

t\_s = 0:0.01:2.75;

impulse = t\_s==1;

unitstep = t\_s>=1; %for a step input

figure;

plot(t\_s,unitstep, 'LineWidth',2); hold on;

plot(t,theta, 'LineWidth',2);hold on;

plot(t, omega, 'LineWidth',2); hold on;

sample\_t = (sample/10^6)-4.580472+1;

rad = -(Degrees-29).\*2\*pi/360;

plot(sample\_t,rad, 'LineWidth',2);

legend("Step Voltage(V)","Modelled {\theta}(rad)", "Modelled {\omega}(rad/s)", "Actual {\theta}(rad)");

xlabel("Time (s)");

ylabel("Amplitude");

title("100% Duty Cycle");

**Graph**

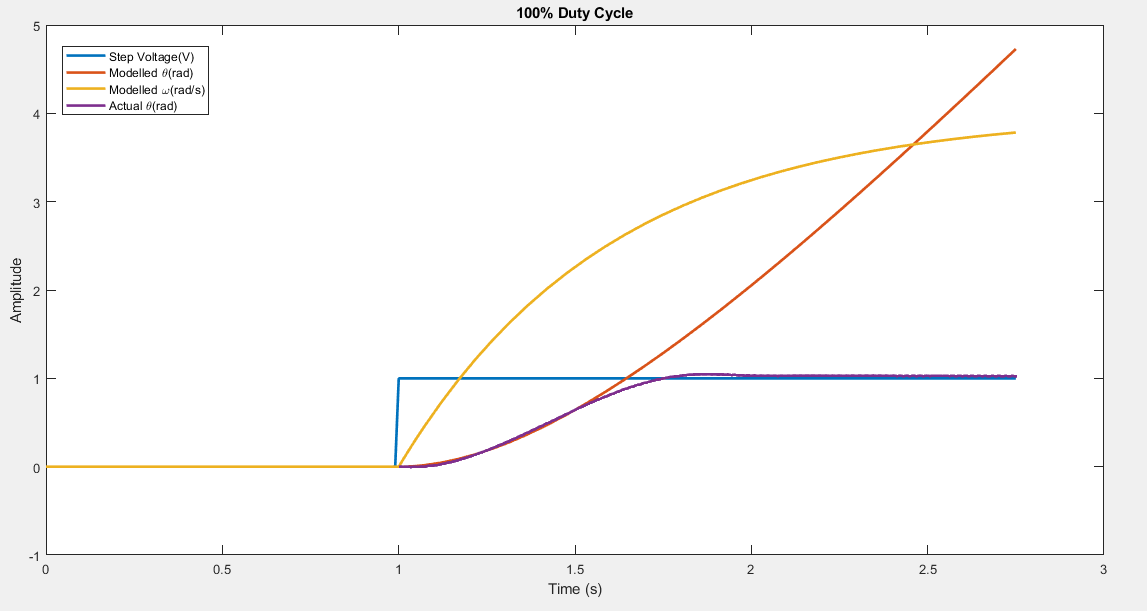


Figure 9: K1 = 80, τ1 = 0.6s

**Matlab Code**

L=0.050; % length of beam - 50mm, mass of beam = 147.24 g, mass of motor = 10.88 g

Ja = 1/12\*(147\*10^-3)\*(50\*10^-3)^2; % Moment of inertia of Arm/Beam; J = 1/12 \*m\_b\*L^2

Jr = 1/2\*0.01088\*(50\*10^-3)^2; % Moment of inertia of motors acting as point masses serperated by distance L J=1/2 \*m\_m\*L^2

J = Ja + Jr;

b =1; % coefficient of friction

K = 35;

T = 0.1; % tau

sim("Final\_model");

t\_s = 0:0.01:3;

impulse = t\_s==1;

unitstep = t\_s>=1; %for a step input

figure;

plot(t\_s,unitstep, 'LineWidth',2); hold on;

plot(t,theta, 'LineWidth',2);hold on;

plot(t, omega, 'LineWidth',2); hold on;

sample\_t = (sample85/10^6)-12.915544+(1-0.21969);

rad = -(Degrees85-30).\*2\*pi/360;

plot(sample\_t,rad, 'LineWidth',2);

legend("Step Voltage(V)","Modelled {\theta}(rad)", "Modelled {\omega}(rad/s)", "Actual {\theta}(rad)");

xlabel("Time (s)");

ylabel("Amplitude");

title("85% Duty Cycle");

**Graph**

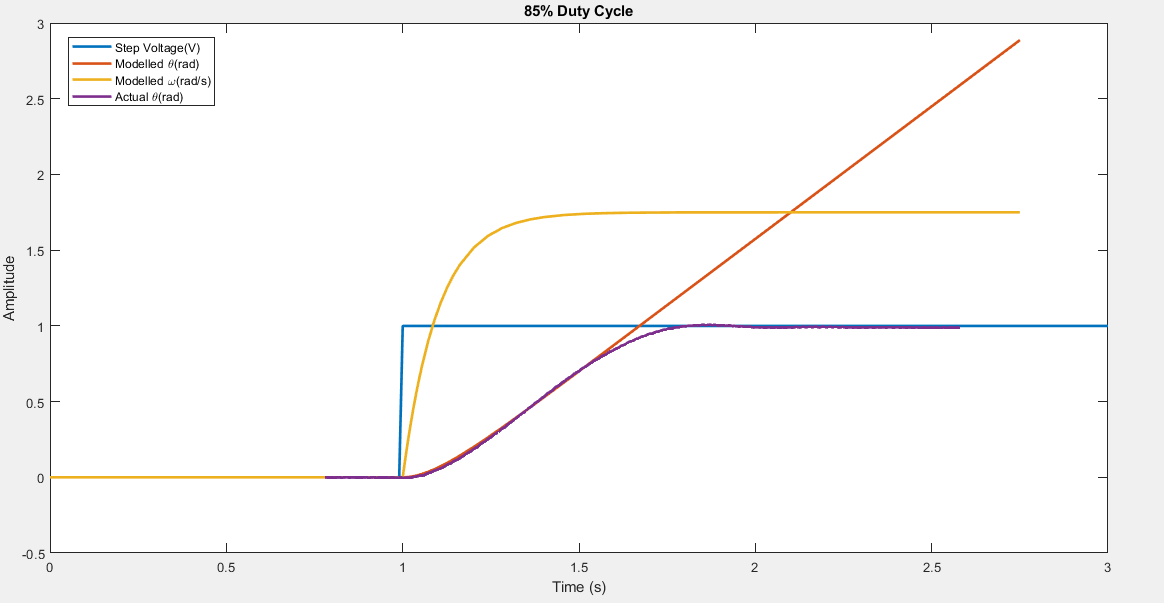


Figure 10: K2 = 35, τ2 = 0.1s

**Matlab Code**

L=0.050; % length of beam - 50mm, mass of beam = 147.24 g, mass of motor = 10.88 g

Ja = 1/12\*(147\*10^-3)\*(50\*10^-3)^2; % Moment of inertia of Arm/Beam; J = 1/12 \*m\_b\*L^2

Jr = 1/2\*0.01088\*(50\*10^-3)^2; % Moment of inertia of motors acting as point masses serperated by distance L J=1/2 \*m\_m\*L^2

J = Ja + Jr;

b =1; % coefficient of friction

K = 27;

T = 0.09; % tau

sim("Final\_model");

t\_s = 0:0.01:3;

impulse = t\_s==1;

unitstep = t\_s>=1; %for a step input

figure;

plot(t\_s,unitstep, 'LineWidth',2); hold on;

plot(t,theta, 'LineWidth',2);hold on;

plot(t, omega, 'LineWidth',2); hold on;

sample\_t = (sample50/10^6)-22.380416+0.5;

rad = -(Degrees50-30).\*2\*pi/360;

plot(sample\_t,rad, 'LineWidth',2);

legend("Step Voltage(V)","Modelled {\theta}(rad)", "Modelled {\omega}(rad/s)", "Actual {\theta}(rad)");

xlabel("Time (s)");

ylabel("Amplitude");

title("50% Duty Cycle");

**Graph**

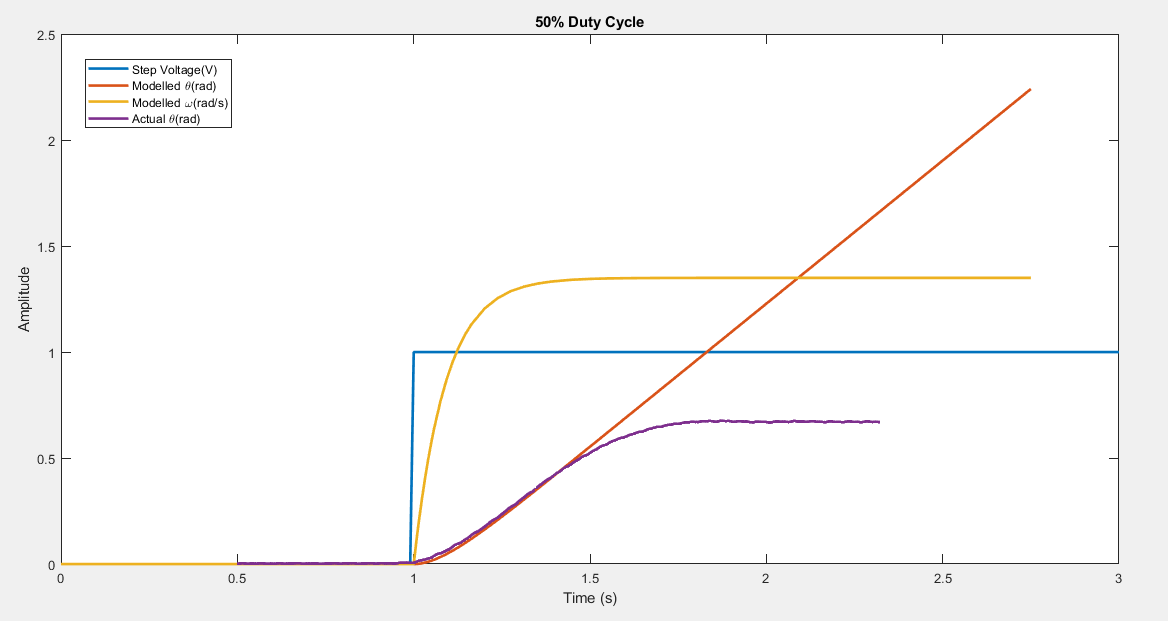


Figure 11: K3 = 27, τ3 = 0.09s

Then, (10) becomes,

( 34)

Since we are using grey-box modelling to model the motor, we find K and τ in equation (10) by providing 100 %, 85 % and 50 % duty cycle of step input to our actual system and tune the values of K and τ in each iteration until it replicates the actual response. The final K and τ were achieved by taking average of the values.

The actual responses in figure 9, 10, and 11 were obtained by building a code on Arduino IDE. We used PID Controller to control the angular displacement of the beam. If our desired angle is set to 30 degrees, the controller is designed such that the beam restores its position at 30 degrees with the initial angular displacement set to zero as shown above in the graphs. Since the theoretical angular displacement is not controlled, the response due to the step voltages causes the theoretical response to exponentially grow.

1. **Conclusions**

Although the aim of our project was to control both the yaw and the pitch angle of a Twin Rotor MIMO System, due to time constraints and the presence of a phenomenon called “cross-coupling” between the pitch angle and the yaw angle, we were only able to control the pitch angle. It was observed that increasing Kp proportionally increased error in the system. Adding Kd did not significantly affect the response with small angle disturbances. With Ki, the response became swift and faster. Regardless, the concepts regarding the PID controller and the proportional, integrative and derivative constants made better sense with the completion of the project.