ME351: Lab Report

Experiment 3:Stick Balance by Reaction Wheel

Group 6 | Week 8

Harshil Safi | 20110073

Hrishikesh CP | 20110078

Kamal Vaishnav | 20110089

Kareena Beniwal | 20110095

Under the Guidance of
Prof. Jayaprakash KR

Experiment Video Link: https://youtube.com/shorts/Sz0DGn2AAX0?feature=share

1. Objective

The objective of this experiment is to design a control system for balancing a stick with one reaction wheel using various electronic devices and components. The main aim is to control the rotation of the reaction wheel with the help of a DC motor and motor controller, and to keep the stick balanced using an IMU and a rotary encoder. The experiment will involve the use of an Arduino Nano microcontroller, a battery and battery charger, a voltage converter, and a printed circuit board. The ultimate goal is to successfully balance the stick using the designed control system.
Overall, the devices and components are used together to create a closed-loop control system that can balance the stick using the reaction wheel. The IMU and rotary encoder provide feedback to the Arduino Nano, which adjusts the speed and direction of the DC motor using the motor controller. The voltage converter and battery provide the necessary power to the system.

2. Experimental design and Fabrication details

Working and Use of Devices:

DC motor - The DC motor is used to rotate the reaction wheel. The single shaft BO motor series with a speed of 300 RPM is selected for this experiment.

Motor controller - The motor driver TB6612FNG module is used to control the speed and direction of the DC motor. It can drive two motors simultaneously and has a high current rating.

IMU - The MPU 6050 is an inertial measurement unit used to measure the orientation and motion of the stick. It consists of a gyroscope and an accelerometer.

Battery - The Orange 360mAh 3S 30C/60C (11.1V) Lithium Polymer battery pack is used to power the DC motor and the other electronic components.

Voltage converter - The Mini 360 Step-down buck converter power module is used to convert the battery voltage to the required voltage level for the electronic components.

Bread Board- The bread board is used to mount and connect the electronic components.

Arduino Nano - The Arduino Nano microcontroller is used to program and control the entire system. It receives the input from the IMU and the rotary encoder, and sends output signals to the motor controller to control the rotation of the DC motor and the reaction wheel.

CAD models of various parts fabricated to set the setup:

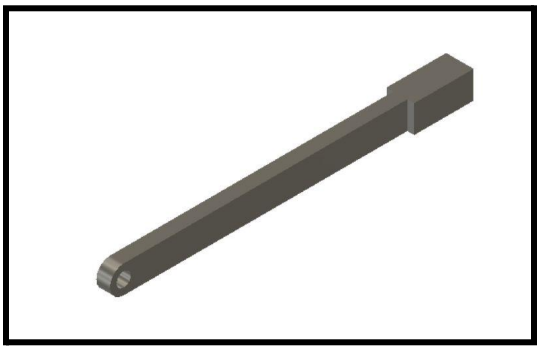
Fig: The stick

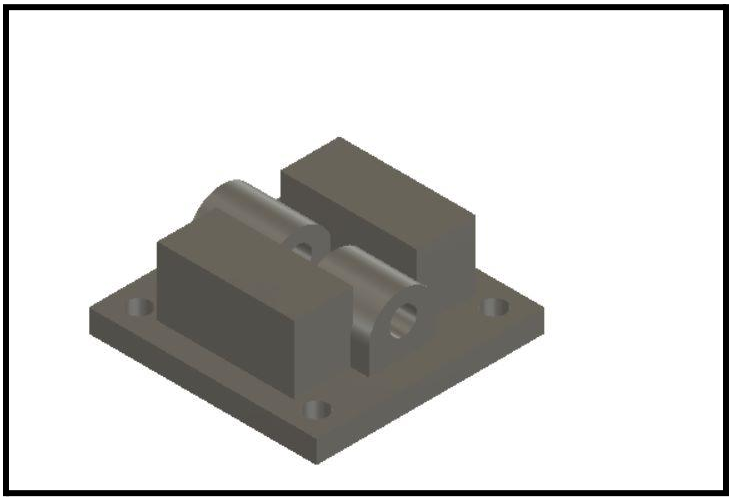
Fig: The base support


Fig: The Assembly

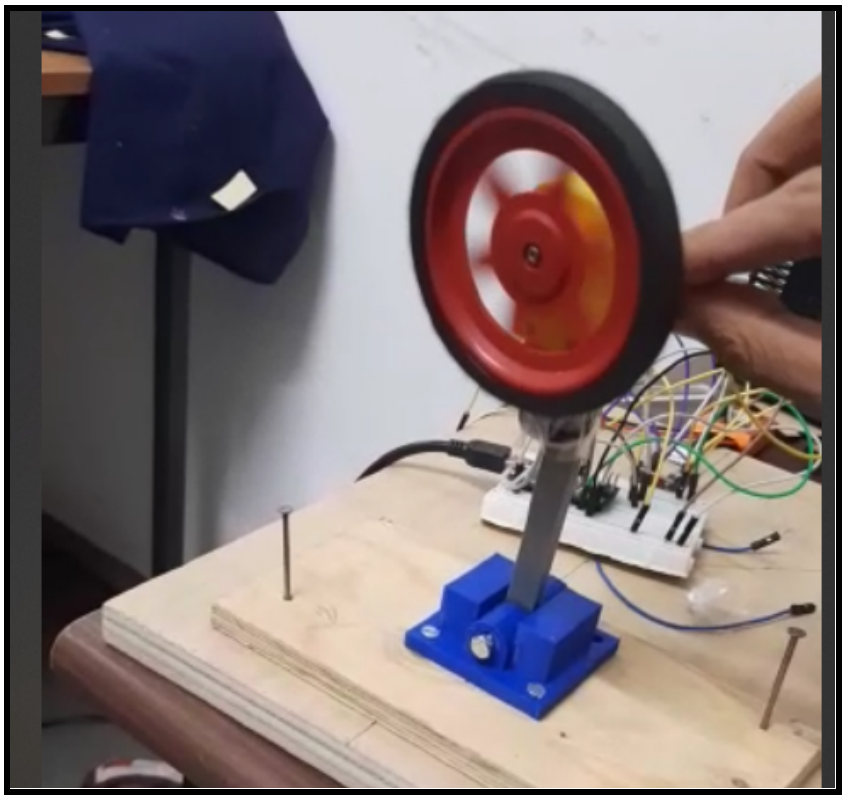

Fig: The set up

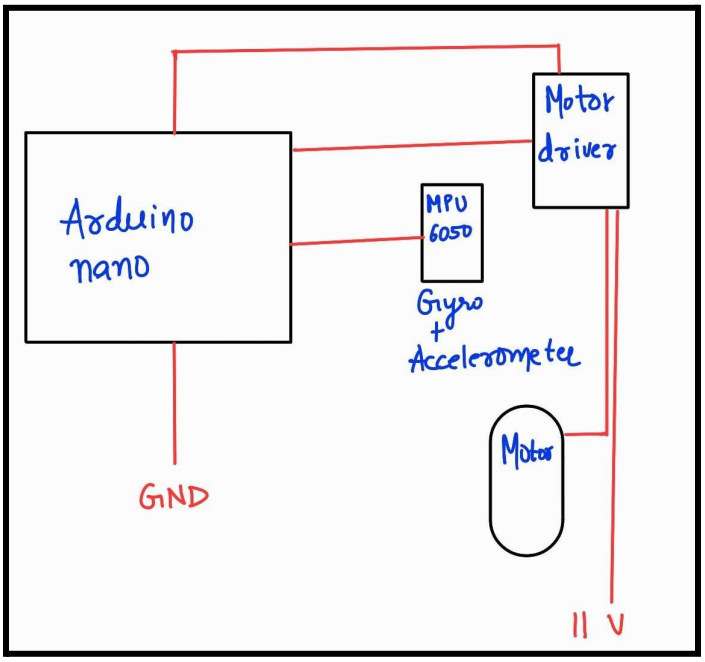

Fig: Electrical Connections

PID Control Algorithm:

A PID (proportional-integral-derivative) controller is a widely used control algorithm in engineering, which is commonly used to regulate and stabilize systems that have feedback.

The PID controller works by continuously adjusting the input signal to a system based on feedback information, in order to maintain a desired output value. The controller receives two inputs: the desired output value and the current value of the system output, which is measured by a sensor. The controller then calculates an error signal, which is the difference between the desired and current output values. *The controller uses this error signal to adjust the input signal to the system.*

The three main components of a PID controller are the proportional, integral, and derivative terms. The proportional term adjusts the input signal in proportion to the error signal. The integral term integrates the error signal over time, which can help to correct any steady-state errors that may be present. The derivative term calculates the rate of change of the error signal, which can help to reduce overshoot and oscillations in the system.

The output of the PID controller is the sum of the three terms, which can be expressed as follows:

$$u(t) = K_p\, e(t) + K_i \int e(t)dt + K_d\, de(t)/dt$$

where u(t) is the controller output at time t, e(t) is the error signal at time t, and K_p, K_i, and K_d are the proportional, integral, and derivative gains, respectively.

The proportional gain determines the magnitude of the controller's response to the error signal. A high proportional gain can result in a fast response but can also lead to overshoot and instability. The integral gain is used to correct any steady-state errors and can be adjusted to improve the system's accuracy. The derivative gain can help to reduce overshoot and oscillations by anticipating the system's response to changes in the error signal.

The gains of the PID controller can be tuned to optimize the system's performance, typically using trial and error or advanced optimization techniques such as model predictive control or particle swarm optimization.

In summary, the PID controller is a widely used control algorithm that uses feedback information to adjust the input signal to a system in order to maintain a desired output value.

The proportional, integral, and derivative terms are the main components of the controller, and their gains can be tuned to optimize the system's performance.

3. Mathematical/Theoretical Analysis

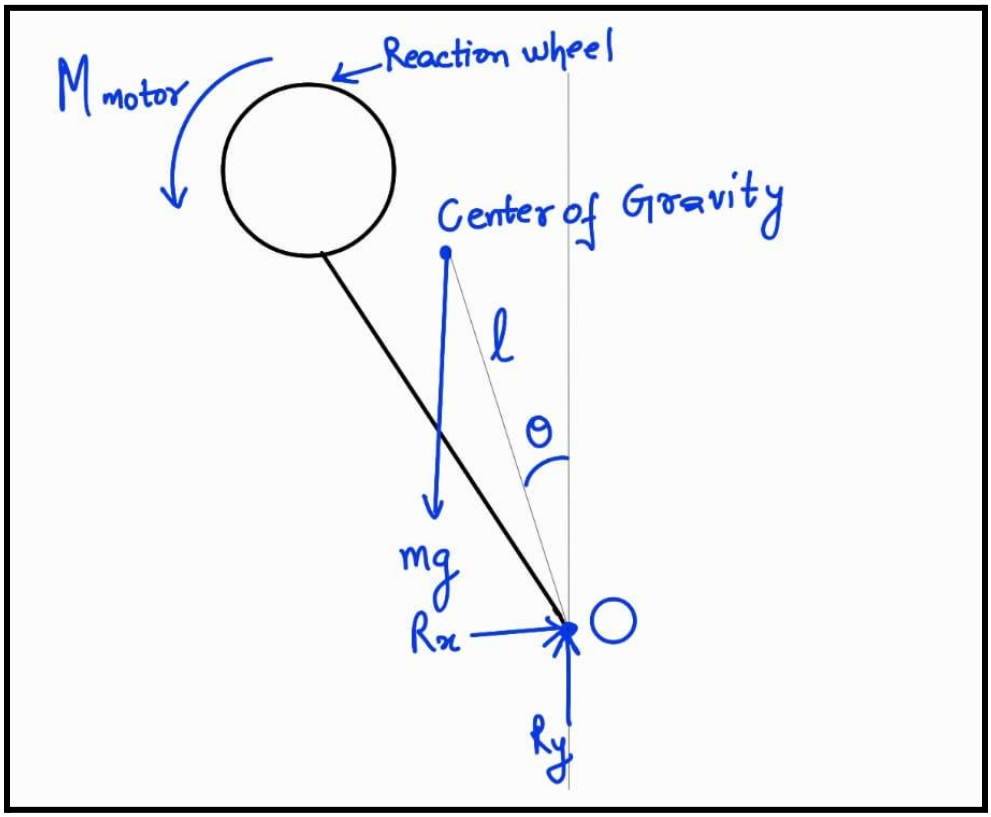

Fig: FBD of the system

z-direction is out of the plane of the paper.
According to above figure and Newton's second law, the rotational dynamics of the stick around O can be described as:

$$I\ddot{\theta} = mglsin(\theta) - M_{motor}$$

where I is the moment of inertia in the z-direction of the stick at O, θ is the angular deviation from the equilibrium point when thestick is upright, m is the mass of the system, l is the distance to the center of gravity and M_{motor} is the torque applied by the motor on the reaction wheel. M_{motor} can be thought of as the reaction torque between the stick and the reaction wheel. Therefore the rotational dynamics of the reaction wheel can be described as:

$$I_r\dot{\omega} = M_{motor}$$

where I_r is the combined moment of inertia of the reaction wheel and the rotor of the motor and ω is the angular velocity of the reaction wheel and the motor.

Motor dynamics:

The relation between the angular velocity of a motor, the voltage applied on the motor and the motor torque can be stated as:

$$M_{motor} = \frac{k_2\phi}{R_A}U - \frac{k_2\phi^2\omega}{R_A}$$

where $k_2\varphi$ is the torque constant, R_A is the terminal resistance and U is the voltage applied on the motor.

State Space Model:

To simplify the analysis of the system and the design of the controller, the system dynamics can be written as a set of first order linear differential equations. Since θ can be assumed to be small and sin(θ) ≈ θ for small θ, therefore:

$$I\ddot{\theta} = mgl\theta - M_{motor}$$

The state variables x_1, x_2 and x_3 are introduced as$x_1 = \theta$, $x_2 = \dot{\theta}$ and $x_3 = \omega$

$$\dot{x}(t) = Ax(t) + Bu(t))$$
$$y(t) = Cx(t) + Du(t)$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ \frac{mgR}{I} & 0 & \frac{k_2\phi^2}{R_A I} \\ 0 & 0 & -\frac{k_2\phi^2}{R_A I_r} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{k_2\phi}{R_A I} \\ \frac{k_2\phi}{R_A I_r} \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

which is the state space model of the system. Here, u(t) is the input variable, or the voltage in this case, and y(t) is the output variable, or the state that is to be controlled, which in this case is θ.

Control design:

The system itself is not asymptotically stable, which can be verified by computing

eigenvalues of matrix A. If the real part of the eigenvalues are strictly negative the system is asymptotically stable. To change the dynamics of the system and make it asymptotically stable, active feedback can be used by letting the voltage U depend on the state variables.
Here, a state space feedback controller with integral action was chosen as the controller. Integral action makes the system more robust to modeling errors and disturbance. A new state variable x_4 is introduced to denote the integrator state which is the integral of the output error. Let r denote the reference or the wanted output, then the following holds:

$$\dot{x}_4 = r - y = r - Cx$$

The system becomes a closed-loop system as the states are fed back to form the input as:

$$u(t) = -Lx = -\begin{bmatrix} L_1 & L_2 & L_3 & L_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

where the vector L contains the feedback gain. All states can be measured. The resulting system now becomes:

$$\begin{bmatrix} \dot{x} \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ x_4 \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r$$

And with the feedback :

$$\begin{bmatrix} \dot{x} \\ \dot{x}_4 \end{bmatrix} = \left(\begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} - \begin{bmatrix} B \\ 0 \end{bmatrix} L \right) \begin{bmatrix} x \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r$$

By choosing L, the poles of the system can be placed as desired. Since the system is asymptotically stable, it will converge to a static state that satisfies $r - Cx = 0$. In other words $x_1 = r$. In this case $r = 0$ since the stick shall be balancing around the equilibrium point where $\theta = 0$.

4. Results and discussions

The results of the experiment show that it is possible to balance a stick in the vertical position using a reaction wheel and a control system. The system components used in the experiment were a DC motor, motor controller, IMU, battery, rotary encoder, voltage converter, printed circuit board, and an Arduino Nano. The control algorithm used was a PID controller, and the stick used had a length of 1m.

The experiment showed that the reaction wheel was able to provide sufficient torque to counteract disturbances and maintain the stick in the vertical position. The control algorithm was able to adjust the motor torque based on the measured angle and maintain the stick's stability.

The results of the experiment demonstrate the feasibility of using a reaction wheel and a control system to balance a stick in the vertical position. The experiment highlights the importance of using a high-quality IMU to measure the stick's angle accurately and a control algorithm such as a PID controller to adjust the motor torque based on the measured angle.

The experiment also suggests several potential improvements that could be made to the system, such as using a faster motor, a motor with higher torque, or a more accurate sensor. Optimizing the control algorithm or implementing safety features could also improve the system's performance and reliability.

Overall, the experiment provides valuable insights into the design and implementation of a system for balancing a stick using a reaction wheel and a control algorithm. These insights could be useful for developing similar systems for applications such as robotics or spacecraft stabilization.

Several parameters such as the mass, the height of the center of mass and the moment of inertia of the reaction wheel are of importance to get the system stable. As expected it is important to implement a good controller and especially a correct set point so that the system does not try to stabilize towards a wrong angle. While methods for finding a good set point can be developed, the system is still very ***sensitive to changes in the center of mass*** and may still fail to balance after some time.

5. Source of discrepancy/mismatch

There are a few potential sources of discrepancy or mismatch in the proposed experiment:

Motor speed: The chosen DC motor has a speed of 300 RPM, which may not be fast enough to provide the necessary control for balancing the stick. Depending on the weight and length of the stick, a faster motor may be needed to make quick adjustments.

Motor torque: The motor's torque should be sufficient to move the reaction wheel with enough force to counteract the stick's motion. If the torque is too low, the system may not be able to maintain balance.

Sensor accuracy: The MPU 6050 is an inexpensive IMU that may have some inaccuracies in its measurements. This could affect the performance of the control system and lead to instability.

Control algorithm: The control algorithm used to balance the stick will play a significant role in the success of the experiment. A well-tuned and robust control algorithm is necessary to maintain stability in the face of disturbances.

Power supply: The chosen battery may not provide enough power to run the system for an extended period, depending on the motor's power consumption and the length of the experiment. A larger battery may be needed to ensure reliable operation.

Mechanical design: The mechanical design of the system, including the mounting of the motor and reaction wheel, may also affect its performance. Any vibrations or imbalances in the system could interfere with the control algorithm's ability to maintain stability.

It is important to carefully consider these potential sources of discrepancy and optimize the system accordingly to ensure the experiment's success.

6. Scope for improvement

There are several potential improvements that could be made to the proposed experiment to enhance its performance and reliability:

Use a higher speed motor: A faster motor would allow the reaction wheel to spin more quickly and provide more rapid adjustments to balance the stick.

Use a motor with higher torque: A motor with higher torque would be able to move the reaction wheel with greater force, which could be especially important for balancing longer or heavier sticks.

Use a more accurate sensor: A higher-quality IMU with better accuracy and precision could provide more reliable measurements, which could improve the control algorithm's ability to maintain stability.

Optimize the control algorithm: The control algorithm could be further refined and optimized to improve its performance and reliability, potentially by implementing more sophisticated control techniques such as model predictive control or adaptive control.

Use a larger battery: A larger capacity battery could provide more power to run the system for longer periods of time, which could be especially important for conducting extended experiments or demonstrations.

Optimize the mechanical design: The mechanical design of the system could be optimized to minimize vibrations and ensure that the motor and reaction wheel are securely mounted and balanced.

Implement safety features: It may be necessary to implement safety features such as limit switches or emergency stop buttons to ensure that the system can be quickly and safely shut down in the event of any issues or malfunctions.

By implementing these improvements, the experiment could be made more reliable, accurate, and effective, potentially leading to more successful outcomes and greater scientific insights.

7. Acknowledgements

We would like to express our sincerest gratitude to Mr. Jayprakash KR, Assistant Professor of Mechanical Engineering at IIT Gandhinagar, for his invaluable guidance and support throughout the completion of our experiment on ‘Stick Balance'. We also extend our appreciation to our TAs NagaVishnu, Vaibhav Tandel, Harsh Gupta, and Ishan for their dedicated support and motivation.

8. References

https://www.youtube.com/watch?v=4gS2i5fecFE
https://www.thingiverse.com/remrc/designs

9. Appendix

Arduino Code:

```
#include <Wire.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>

Adafruit_MPU6050 mpu;

// Define motor driver pins
const int PWMA = 3;
const int AIN1 = 5;
const int AIN2 = 6;
const int STBY = 9;

// Define variables for PID controller
float angle = 0.0;
float elapsedTime, time, timePrev;
float angle_previous_error, angle_error;
float kp = 100;
float ki = 0.1;
float kd = 0;
float desired_angle = 0;
float PID_p, PID_i, PID_d, PID_error;
int period = 50;
float duration;

// Define acceleration mapping values
const float MAX_ACCEL = 8.0;  // maximum acceleration value in m/s^2
const int MAX_SPEED = 255;   // maximum motor speed
```

```
const int MIN_SPEED = -255;    // minimum motor speed


void setup() {
  Serial.begin(9600);
  while (!Serial);

  if (!mpu.begin()) {
    Serial.println("Failed to find MPU6050 sensor");
    while (1);
  }

  mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
  mpu.setGyroRange(MPU6050_RANGE_500_DEG);
  mpu.setFilterBandwidth(MPU6050_BAND_5_HZ);

  // Set motor driver pins as outputs
  pinMode(PWMA, OUTPUT);
  pinMode(AIN1, OUTPUT);
  pinMode(AIN2, OUTPUT);
  pinMode(STBY, OUTPUT);

  // Set STBY pin HIGH to enable the motor driver
  digitalWrite(STBY, HIGH);
  time = millis();
}

void loop() {


  if (millis() > time+period)
  {
  time = millis();
  sensors_event_t accel_event, gyro_event, temp_event;
  mpu.getEvent(&accel_event, &gyro_event, &temp_event);
```

```
  float ax = accel_event.acceleration.x;
  float ay = accel_event.acceleration.y;
  float az = accel_event.acceleration.z;
  float angle =180*(asin(ay/9.81))/3.14;   // calculate angle in degrees

// Calculate error and perform PID calculations
    angle_error = desired_angle - angle;
    PID_p = kp * angle_error;
    PID_i = PID_i + (ki * angle_error * period / 1000);
    PID_d = kd * ((angle_error - angle_previous_error) / period);


  PID_error = PID_p + PID_i + PID_d;
  // Map PID error to motor speed range
    int motor_speed;
    if (angle_error <= 0) {
      digitalWrite(AIN1, LOW);
      digitalWrite(AIN2, HIGH);
    } else {
      digitalWrite(AIN1, HIGH);
      digitalWrite(AIN2, LOW);
    }
    // Set motor speed
    if(abs(motor_speed)<0)
    motor_speed=0;
    if(abs(motor_speed)>255)
    motor_speed=255;


    analogWrite(PWMA, abs(motor_speed));
    Serial.println(angle);
    angle_previous_error = angle_error;
   delay(50);
}
}
```