Abstract

This document provides an overview of the components and design considerations for a self-balancing robot. The purpose of the robot is to maintain its equilibrium and balance in an upright position using a combination of sensors, actuators, and control algorithms. The key components selected for this project include the frame, Arduino UNO microcontroller, MPU 6050 sensor, L298N motor driver, battery, DC geared motors, switch, HC-05 Bluetooth module, wires, and wheels.

The frame serves as the body of the robot, providing structural support and housing for the various components. The Arduino UNO microcontroller acts as the brain of the robot, processing sensor data and generating control signals for the motors. The MPU 6050 sensor is used to measure the robot's orientation and acceleration, providing crucial feedback for maintaining balance.

The L298N motor driver enables the control of the DC geared motors, which are responsible for the robot's movement and balance adjustments. A suitable battery is chosen to power the robot and ensure sufficient runtime. A switch is incorporated to control the robot's power supply.

The HC-05 Bluetooth module facilitates wireless communication with external devices for remote control or data monitoring purposes. Wires are used for electrical connections between the components. Wheels are essential for the robot's mobility and stability.

The design process involves using SolidWorks software for the mechanical design of the robot. A flowchart is created to visualize the control algorithm, which employs PID (Proportional-Integral-Derivative) control. The PID control equation and transfer function are discussed, providing insight into the control algorithm's implementation.

Block diagrams are used to illustrate the system's overall architecture, showing the interconnections between the components. The document presents the robot's response before and after implementing the PID control, highlighting the improvement in balancing performance.

Real-world applications of self-balancing technology are also mentioned, including self-balancing unicycles, self-balancing bikes, Segway robots, hoverboards, and self-balancing wheelchairs.

Overall, this content serves as a comprehensive guide to understanding the components, design process, and control principles involved in building a self-balancing robot. It provides valuable insights for individuals interested in robotics and autonomous systems.

Content

Table of Contents

Abstract 1

Content 2

List of figures 3

self-balancing robot 4

Components and why we select it? 5

- **Frame (Body)** 5
- **Arduino UNO** 5
- **MPU 6050** 6
- **Motor driver We use L298N.** 6
- **Battery** 7
- **DC motor - Geared motors** 7
- **Switch** 7
- **The HC-05 Bluetooth module** 8
- **Wire** 8
- **Wheel** 8

SolidWorks Design 9

Flowchart 10

PID control 11

- **PID Control Equation** 12

Transfer Function 13

Block diagram 14

Response before 15

Transfer Function after the PID 18

Response after PID Control 19

Real application 22

- **Self-balancing unicycle** 22
- **Monocycle Is the World’s First self-Balancing Bike** 22
- **Segway Robot** 23
- **Hoverboard, also known as a self-balancing scooter.** 23
- **Self-balancing Wheelchairs.** 23

References 25

List of figures

Figure1. 1: Frame (Body) 5
Figure1. 2: Arduino UNO.......... 5
Figure1. 3: MPU 6050.......... 6
Figure1. 4: Motor driver We use L298N. 6
Figure1. 5: Battery.......... 7
Figure1. 6: DC motor 7
Figure1. 7: Switch 7
Figure1. 8: The HC-05 Bluetooth module.......... 8
Figure1. 9: SolidWorks Design 9
Figure1. 10 SolidWorks Design.......... 9
Figure1. 11Self-balancing unicycle.......... 22
Figure1. 12 Monocycle 22
Figure1. 13: Segway Robot.......... 23
Figure1. 14: Hoverboard 23
Figure1. 15: Self-balancing Wheelchairs 24

Self-balancing robot

A self-balancing robot is a fascinating and innovative creation that showcases the integration of mechanical components, electronics, and intelligent control algorithms. This self-balancing robot is designed with two DC motors that serve as its primary propulsion and stabilization mechanism. These motors are connected to wheels or legs, allowing the robot to move in a controlled manner while maintaining its balance.

To effectively control the movement and balance of the robot, a motor driver is employed. The motor driver acts as an interface between the Arduino Uno, the microcontroller at the heart of the system, and the DC motors. It provides the necessary power and signal conditioning to enable precise control over the motors' speed and direction, allowing the robot to navigate its environment and adjust its balance as needed.

The Arduino Uno, a popular microcontroller board, serves as the brain of the self-balancing robot. It receives data from various sensors and executes sophisticated control algorithms to maintain the robot's equilibrium. In this case, the control algorithm employed is the PID (Proportional-Integral-Derivative) control method. PID control enables the robot to continuously calculate the error between its desired and actual tilt angles, and then apply appropriate corrective actions to maintain balance. By analyzing sensor data and adjusting the motor speeds accordingly, the robot can swiftly react to changes in its orientation and maintain an upright position.

To accurately measure the tilt and acceleration of the robot, an MPU6050 sensor is integrated into the system. The MPU6050 is an Inertial Measurement Unit (IMU) sensor that combines a gyroscope and an accelerometer. It provides precise information about the robot's angular velocity and linear acceleration in real-time. By fusing data from both the gyroscope and accelerometer, the self-balancing robot can accurately determine its orientation and make precise adjustments to its motor speeds, ensuring optimal balance and stability.

In addition to its self-balancing capabilities, this robot also incorporates a Bluetooth module. The Bluetooth module establishes a wireless communication link between the robot and a controlling device such as a smartphone or a computer. This feature allows the robot to be operated remotely, like an RC car. By sending commands wirelessly, users can control the robot's movements and explore its surroundings without physically interacting with it. This remote-control functionality adds an extra layer of convenience and versatility to the self-balancing robot, expanding its range of applications and possibilities.

To provide a sturdy yet customizable structure for the robot, a 3D printed frame is utilized. 3D printing technology allows for the creation of intricate and precise designs, enabling the construction of a lightweight frame tailored specifically for the self-balancing robot. The 3D printed frame not only provides structural support but also allows for easy modification and customization to accommodate additional components or design enhancements.

Components and why we select it?

Frame (Body)

We designed the body of the robot on the Solidworks program with a 3D system, and after completing it, we printed the design on a CNC machine.

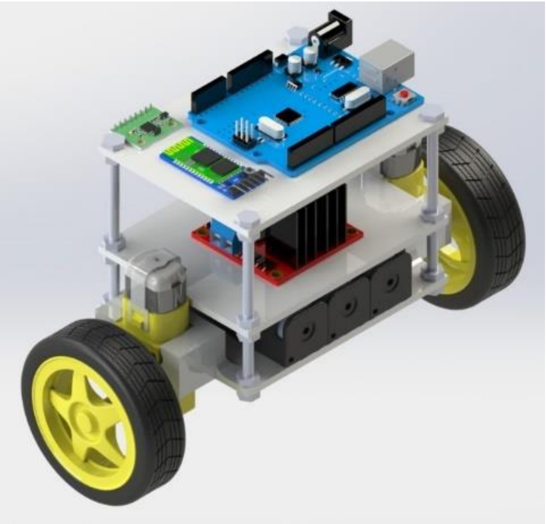


Figure1. 1: Frame (Body)

Arduino UNO

Uno is the popular Arduino due to its smaller size and compatibility with software and hardware in the system Uno has a power regulator on the board to control the power coming to the controller. This helps to maintain constant power in the board. Also, it protects the board from short circuits.

A programmable microcontroller is used so that the tasks could be modified easily in the system with ATMega 16U2.

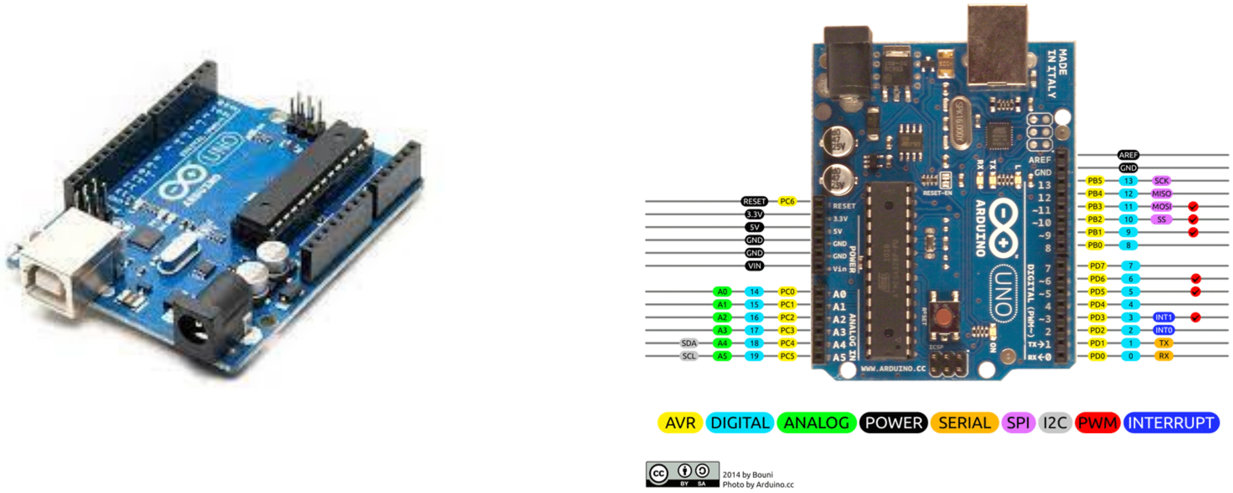


Figure1. 2: Arduino UNO

MPU 6050

MPU6050 sensor module is a complete 6-axis Motion Tracking Device. It combines 3-axis Gyroscope, 3-axis Accelerometer and Digital Motion Processor all in small package. Also, it has the additional feature of an on-chip Temperature sensor. It has I2C bus interface to communicate with the microcontrollers. It has an Auxiliary I2C bus to communicate with other sensor devices like 3-axis Magnetometer, Pressure sensor etc. If a 3-axis Magnetometer is connected to auxiliary I2C bus, then MPU6050 can provide complete 9-axis Motion Fusion output.

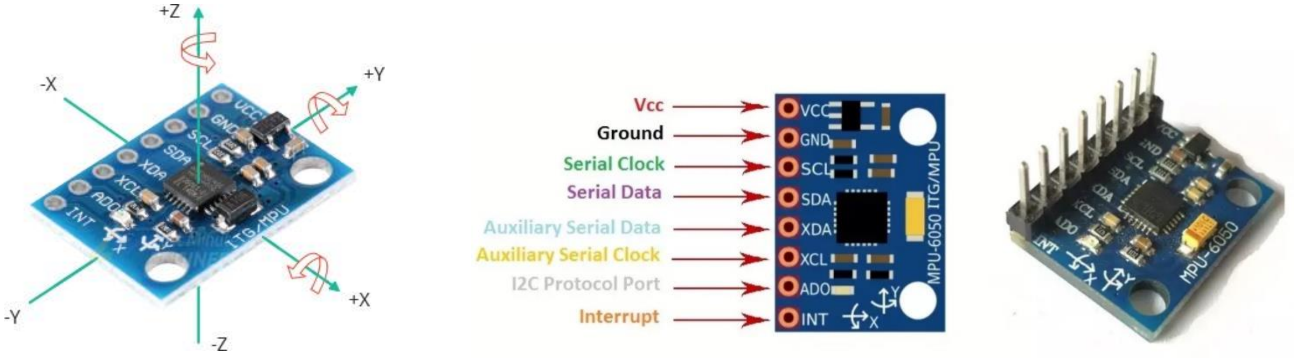


Figure1. 3: MPU 6050

Motor driver We use L298N.

L298N module is a high voltage, high current dual full-bridge motor driver module for controlling DC motor and stepper motor. It can control both the speed and rotation direction of two DC motors. This module consists of an L298 dual-channel H-Bridge motor driver IC. This module uses two techniques for the control speed and rotation direction of the DC motors. These are PWM – For controlling the speed and H-Bridge – For controlling rotation direction. These modules can control two DC motor or one stepper motor at the same time.

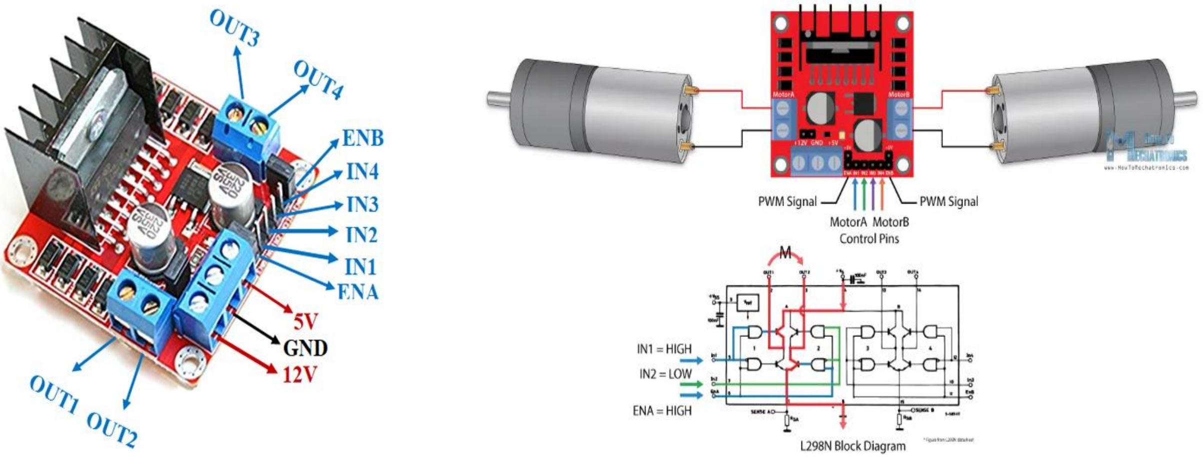


Figure1. 4: Motor driver We use L298N.

Battery

Compared with traditional battery technology, lithium-ion batteries charge faster, last longer, and have a higher power density for more battery life in a lighter package. When you know a little about how they work, they can work that much better for you.

Figure1. 5: Battery

DC motor - Geared motors

A gear motor is an all-in-one combination of a motor and gearbox. The addition of a gearbox to a motor reduces the speed while increasing the torque output. The most important parameters regarding gear motors are speed (rpm), torque (lb-in) and efficiency (%).

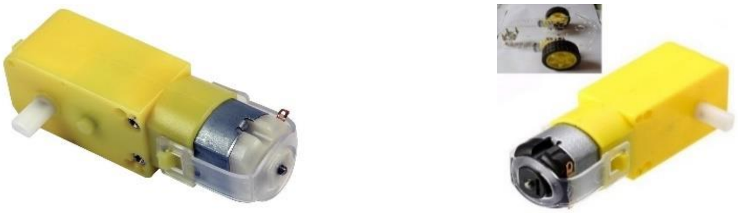


Figure1. 6: DC motor

Switch

work for AC 250V 6A, AC 125V 10A; Mini Switch on/off with 24 * 21 * 15mm(L*W*H), easy to setup.

Setup: 2 pins 2 position small boat rocker switch easy to install. the "A" terminal is connected to your equipment, the "B" terminal is connected to "L" (AC). NOTE: PLEASE POWER OFF before installation.

Widely Used: Widely used for various kinds of electrical products, instrument, car, boat, household appliances such as lights, water dispenser, treadmill, coffee pot, speaker, electric car, motorcycle, TV, massage machine etc.

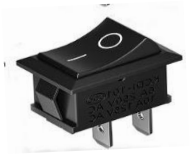


Figure1. 7: Switch

The HC-05 Bluetooth module

The Bluegiga WT11 module on the Arduino BT provides Bluetooth® communication with computers, phones, and other Bluetooth® devices. The WT11 communicates with the ATmega328P via serial (shared with the RX and TX pins on the board). It comes configured for 115200 baud communication. The module should be configurable and detectable by your operating system's Bluetooth® drivers, which should then provide a virtual com port for use by other applications. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board over this Bluetooth® connection. The board can also be reprogrammed using this same wireless connection.

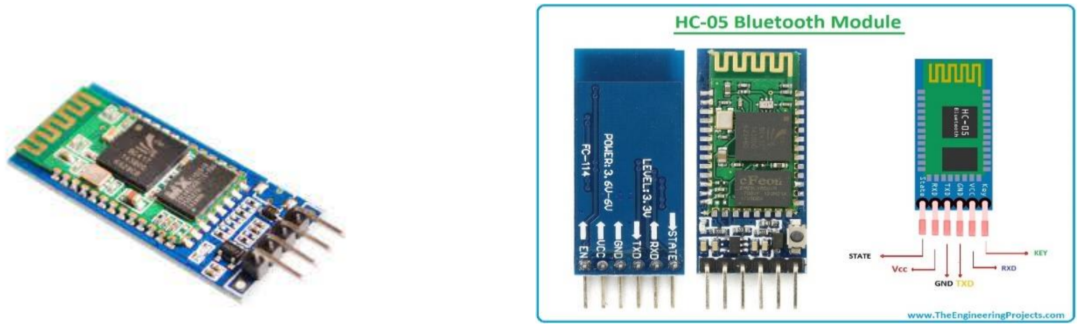


Figure1. 8: The HC-05 Bluetooth module

Wire

Wheel

SolidWorks Design

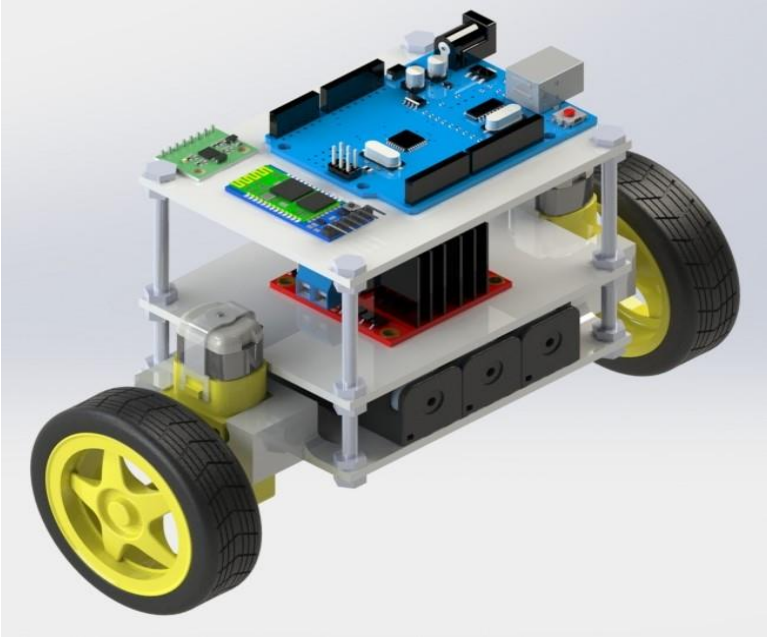


Figure1. 9: SolidWorks Design

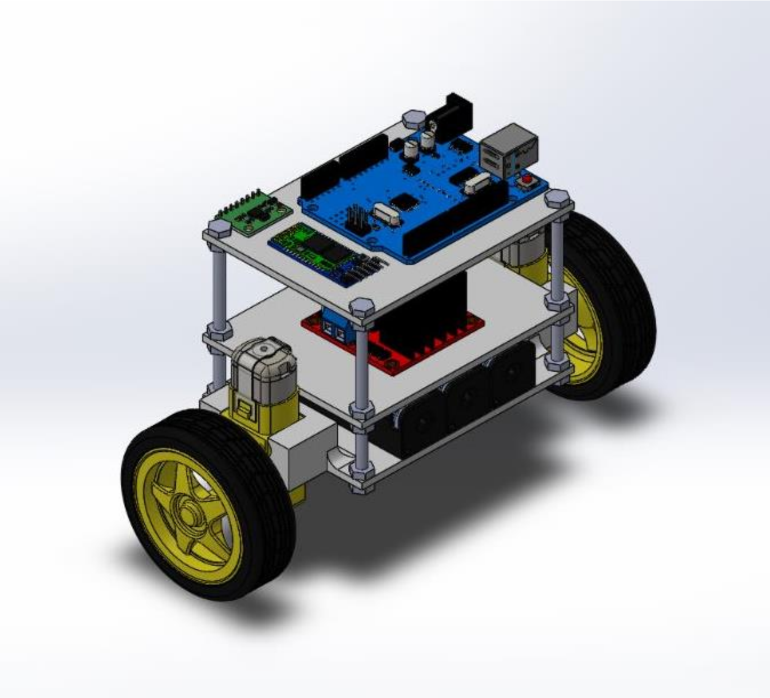
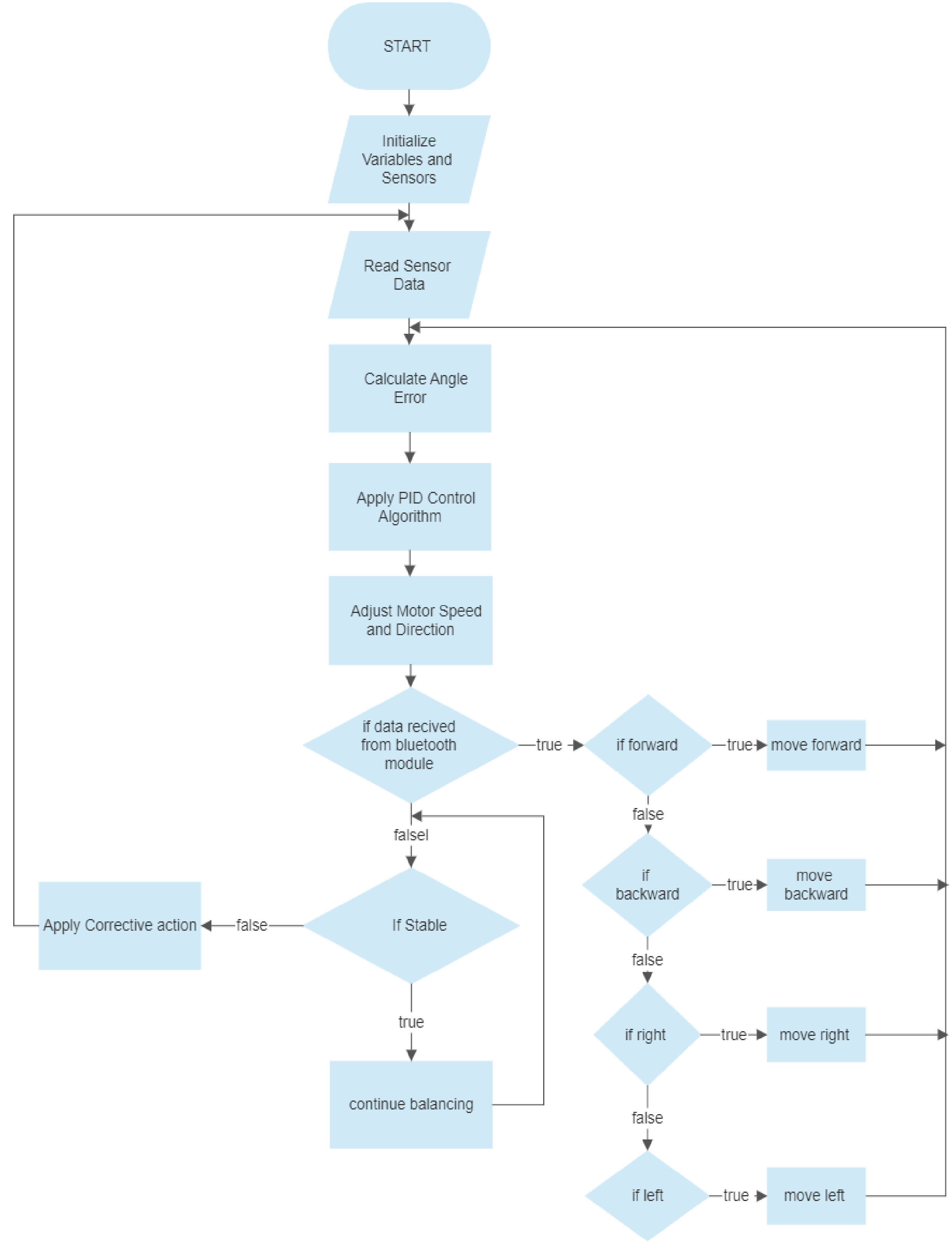


Figure1. 10 SolidWorks Design

Flowchart


START
Initialize Variables and Sensors
Read Sensor Data
Calculate Angle Error
Apply PID Control Algorithm
Adjust Motor Speed and Direction
if data recived from bluetooth module
true
if forward
true
move forward
false
if backward
true
move backward
false
if right
true
move right
false
if left
true
move left
falsel
If Stable
false
Apply Corrective action
true
continue balancing

PID control

short for Proportional-Integral-Derivative control, is a widely used feedback control algorithm in engineering and automation systems. It is designed to regulate and maintain a desired setpoint or target value for a specific process variable by continuously adjusting a control output.

In a PID controller, three components work together to achieve control:

Proportional (P), Integral (I), and Derivative (D).

The proportional component takes the current error, which is the difference between the desired setpoint and the actual value of the process variable and generates a control output proportional to the error. This component provides an immediate response to the error, contributing to the stability and speed of the control system.

The integral component considers the cumulative error over time and calculates the integral of the error. It adds a corrective term to the control output, which helps to eliminate steady-state errors that may arise due to factors like system biases or external disturbances.

The derivative component determines the rate of change of the error with respect to time. By considering the slope or gradient of the error, it anticipates future trends and provides a damping effect on the control output. This helps to counteract sudden changes or oscillations in the system and improves the overall stability and responsiveness.

The combination of the three components, proportional, integral, and derivative, allows the PID controller to effectively respond to different types of errors and adapt to varying system dynamics. By tuning the individual gains of each component, the controller can be optimized for specific applications, ensuring accurate and efficient control.

PID control is a powerful and widely adopted control algorithm that enables precise regulation of a process variable by continuously adjusting a control output based on the error, integral of the error, and derivative of the error. Its versatility and effectiveness make it a fundamental tool in various industries, including robotics, automation, temperature control, and many other control systems.

PID Control Equation

u(t) = Kp * e(t) + Ki * ∫[0, t] e(t) dt + Kd * de(t)/dt

Terms:

- Proportional term: Kp * e(t)
- Integral term: Ki * ∫[0, t] e(t) dt
- Derivative term: Kd * de(t)/dt

Where:

- u(t) is the control output at time t.
- Kp is the proportional gain.
- Ki is the integral gain.
- Kd is the derivative gain.
- e(t) is the error between the desired setpoint and the actual value at time t.
- ∫[0, t] represents the integral over the time interval [0, t].
- de(t)/dt is the rate of change of the error with respect to time.

The proportional term (Kp * e(t)) provides an immediate response to the current error.

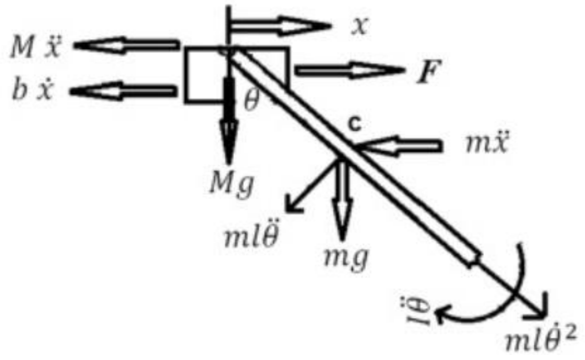
The integral term (Ki * ∫[0, t] e(t) dt) considers the accumulated error over time.

The derivative term (Kd * de(t)/dt) accounts for the rate of change of the error.

Advantage and limitations for each mode:

Response	**Rise Time**	**Over shot**	**Settling time**	**S-S Error**
K_p	Decrease	Increase	NT	Decrease
K_i	Decrease	Increase	Increase	Eliminate
K_d	NT	Decrease	Decrease	NT

Transfer Function



$$\sum F_x = 0$$

$$\boldsymbol{F - b\dot{x} - M\ddot{x} - m\ddot{x} - ml\ddot{\theta}\cos\theta + ml\dot{\theta}^2\sin\theta = 0} \qquad \textbf{(1)}$$

$$\sum F_y = 0$$

$$\boldsymbol{Mg + mg - ml\ddot{\theta}\sin\theta + ml\dot{\theta}^2\cos\theta = 0} \qquad \textbf{(2)}$$

$$\sum M_A = 0$$

$$\boldsymbol{mgl\sin\theta + I\ddot{\theta} + ml^2\ddot{\theta} + ml\ddot{x}\cos\theta = 0} \qquad \textbf{(3)}$$

Equations (1), (2), and (3) are combined, hence the force equation is obtained alone the horizontal and vertical directions,

$$\boldsymbol{(I + ml^2)\theta + mgl\sin\theta = -ml\ddot{x}\cos\theta} \qquad \textbf{(4)}$$

$$\boldsymbol{(M + m)\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F} \qquad \textbf{(5)}$$

Equations (4) and (5) are two linear equations of the transfer function,

where $q = \pi$. With assumption $\theta = \pi + \emptyset$,

$$\boldsymbol{\cos\theta = -1} \qquad \textbf{(6)}$$

$$\boldsymbol{\sin\theta = -\emptyset} \qquad \textbf{(7)}$$

$$\boldsymbol{\frac{d^2}{dt^2} = 0} \qquad \textbf{(8)}$$

After Processing the equations (6), (7) and (8) approach to non-linear, then we obtained two variations of motion equations. The U value represents the input,

$$\boldsymbol{(I + ml)^2\ddot{\emptyset} - mgl\emptyset = ml\ddot{x}} \qquad \textbf{(9)}$$

$$\boldsymbol{(M + m)\ddot{x} + b\dot{x} + ml\ddot{\emptyset} = U} \qquad \textbf{(10)}$$

By applying Laplace transform on equations (9) and (10),

$$\boldsymbol{(I+ml^2)\ddot{\emptyset}(s)s^2 - mgl\emptyset(s) = mlX(s)s^2} \qquad \textbf{(11)}$$

$$\boldsymbol{(M+m)X(s)s^2 + bX(s)s + ml\emptyset(s)s^2 = U(s)} \qquad \textbf{(12)}$$

We yielded the transfer function for TWBMR from the equations (11) and (12).

$$\boldsymbol{\frac{\emptyset(s)}{U(s)} = \frac{\frac{ml}{q}S}{S^3+\frac{b(I+ml^2)}{q}S^2-\frac{(M+m)mgl}{q}S-\frac{bmgl}{q}}} \qquad \textbf{(13)}$$

Where $\boldsymbol{q = [(M+m)(I+ml^2)-(ml)^2]}$

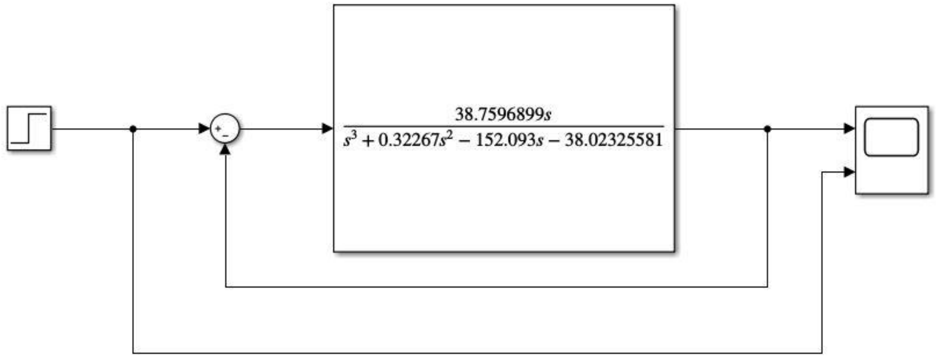
The parameters in table 1 are substituted in equation (13) by replacing the (q) value in equation (13), we obtained the transfer function of TWBMR.

Variables	**Description**	**Value**
$\boldsymbol{M}$	Mass of the cart	0.1 kg
$\boldsymbol{m}$	Mass of pendulum rod	0.3 kg
$\boldsymbol{b}$	Viscous friction coefficient	0.1 N/m/sec
$\boldsymbol{l}$	One half pendulum rod length	0.025 m
$\boldsymbol{I}$	Inertia moment pendulum rod	$4.36875 \times 10^{-4}\ kg.m^2$
$\boldsymbol{F}$	Applied Force to the cart	$kg.m/s^2$
$\boldsymbol{g}$	Gravity	$9.81\ m/s^2$
$\boldsymbol{\theta}$	Pendulum angle	rad

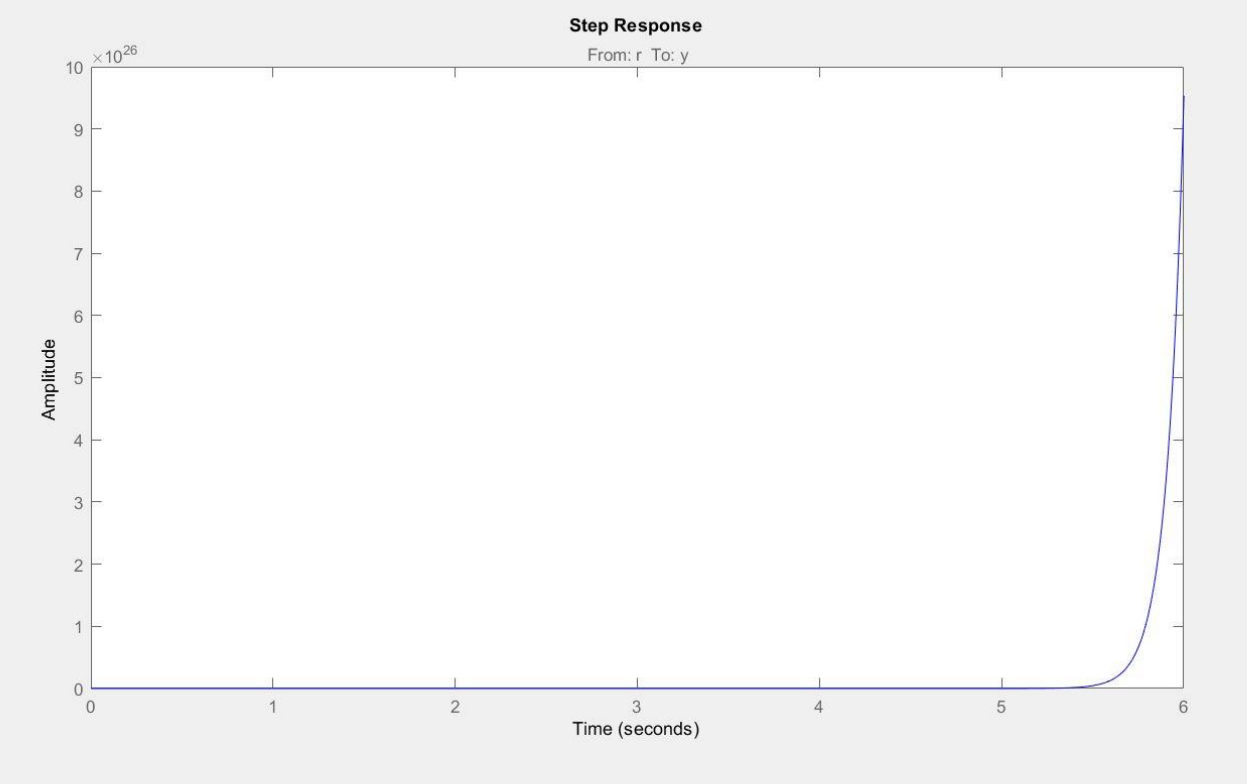
After Substitute:

$$\frac{38.7596899s}{s^3+0.32267s^2-152.093s-38.02325581}$$

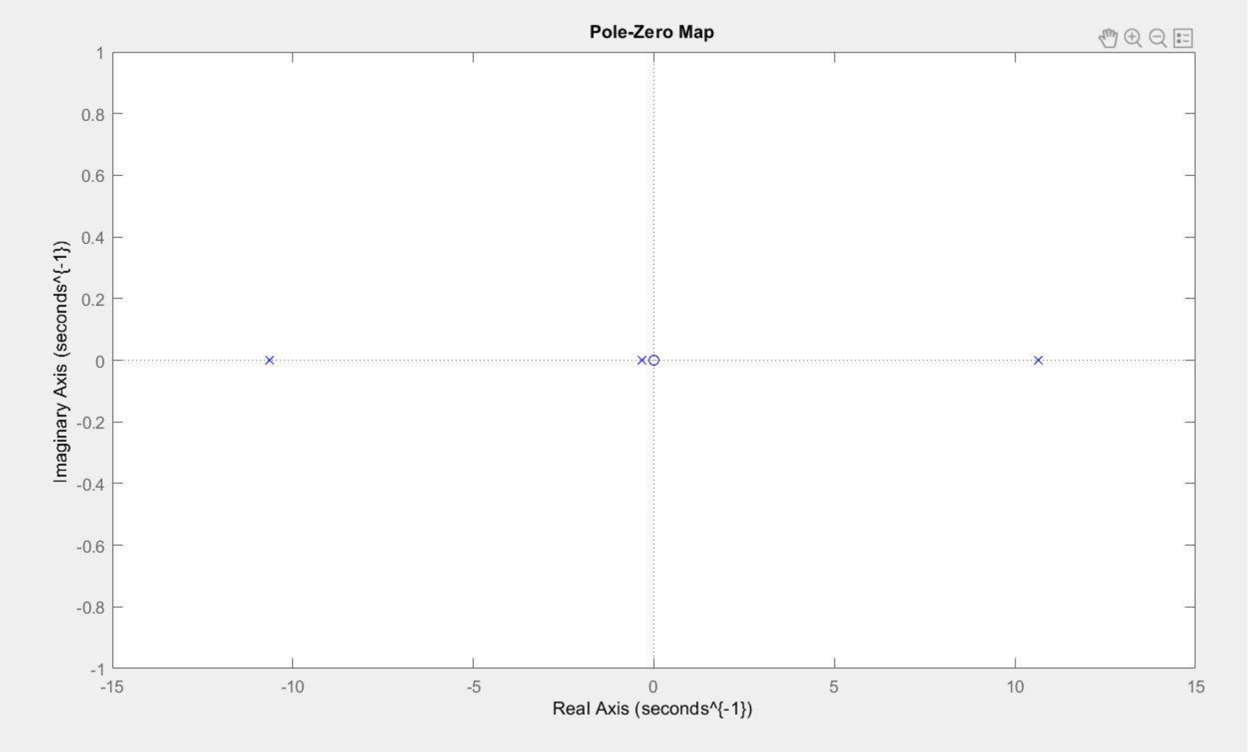
Block diagram



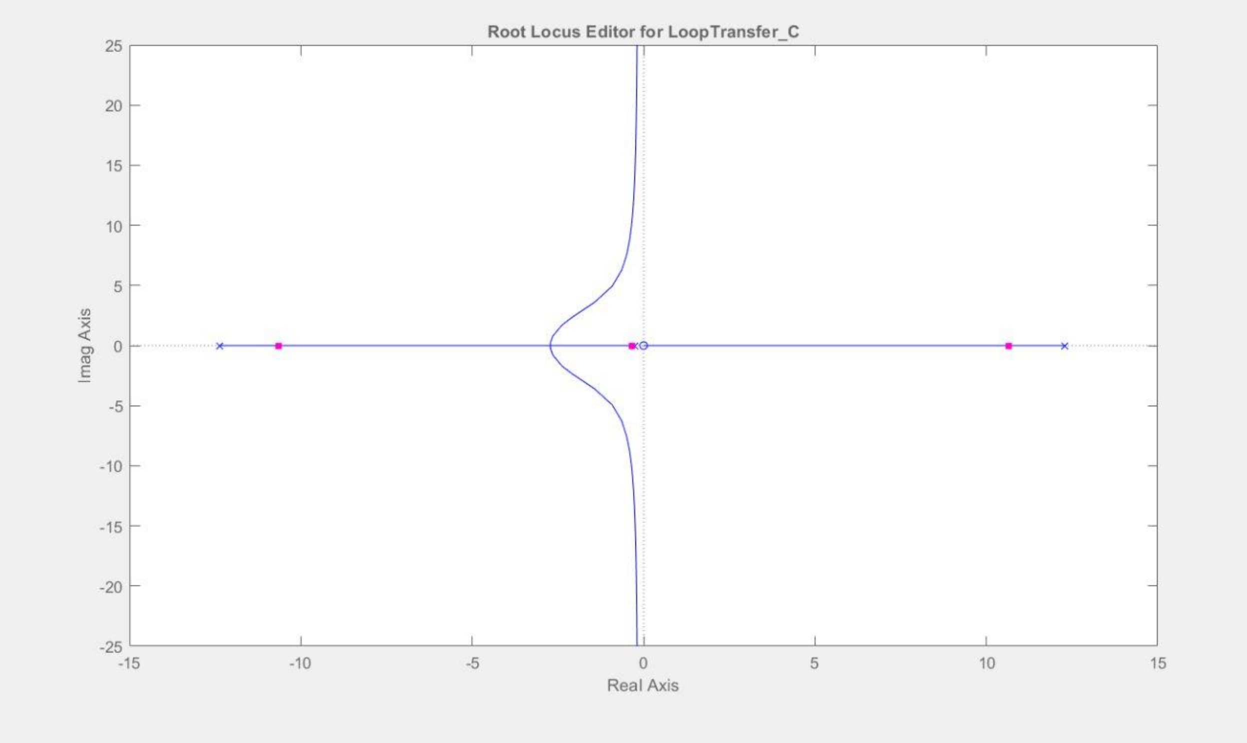
Response before



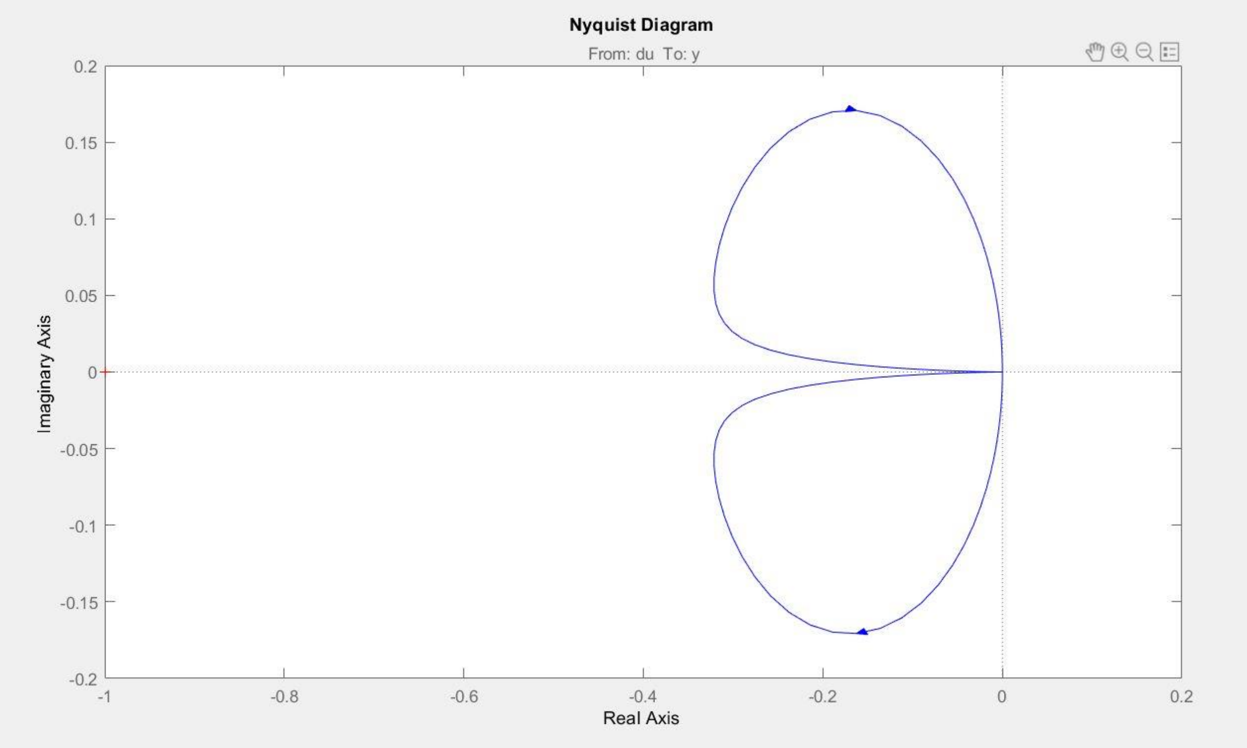
the plot shows that Response is unstable, as response increases to infinity with time.



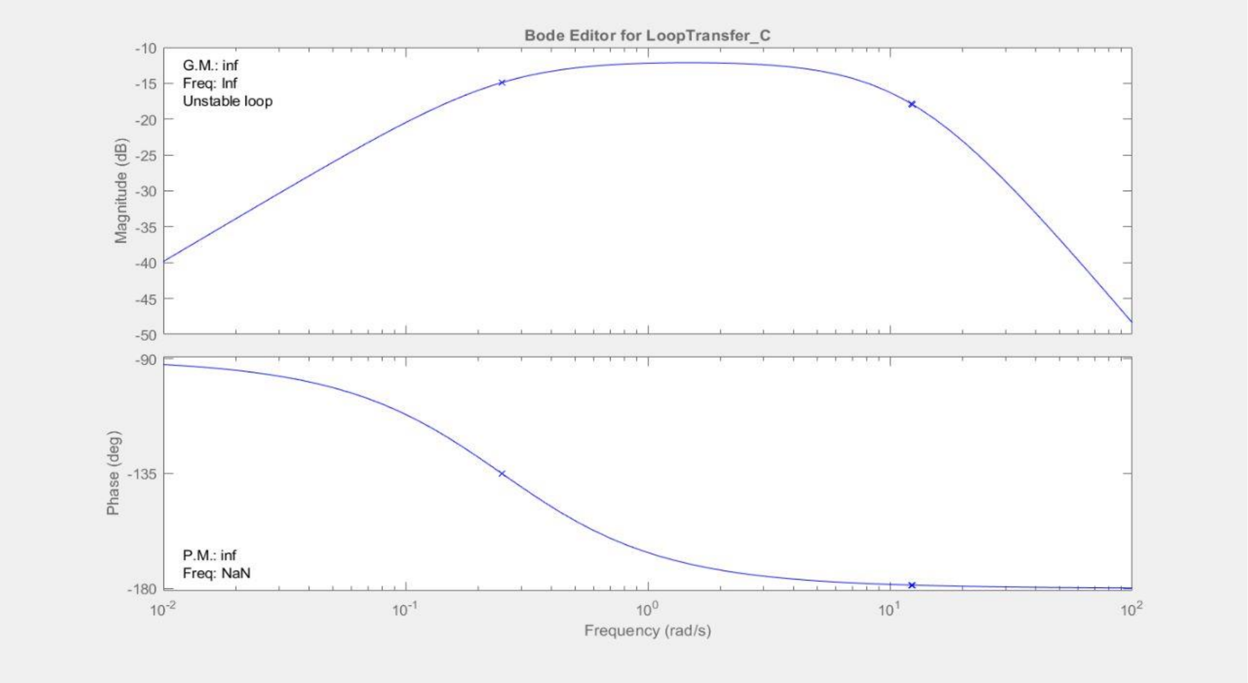
The system has 3 poles (two at left, one at right) (P1(-12.4), P2(-0.25),P3(12.3)),
1 zeros(marginal zero)



Take-off point (-2.72), while there is a pole on the right, the system is unstable.



No rotation about -1, but there is a pole on right, so the system is unstable. **(Z=N+P = 1)**

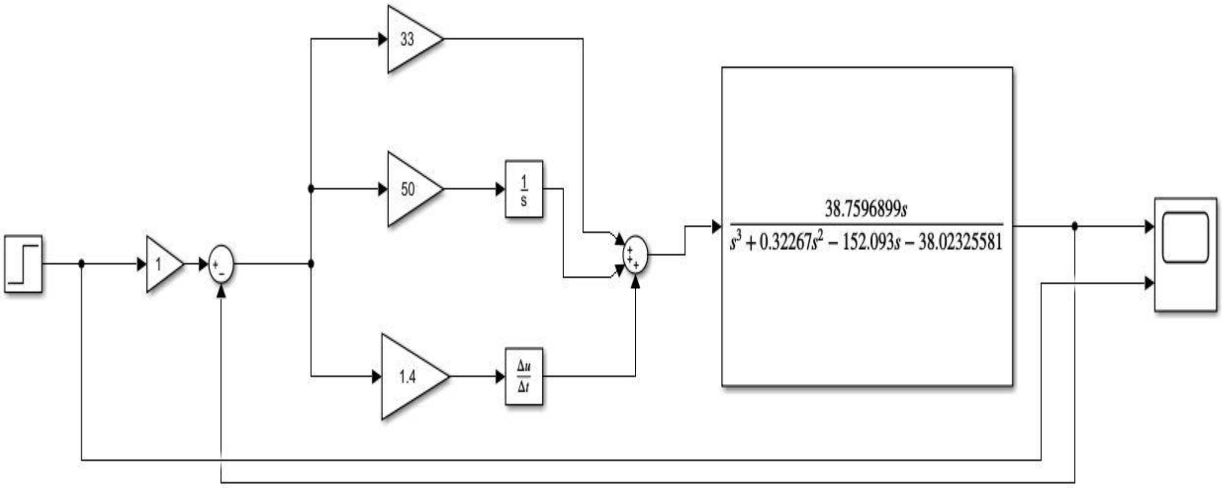

Bode Editor for LoopTransfer_C
G.M.: inf
Freq: Inf
Unstable loop
Magnitude (dB)
-10
-15
-20
-25
-30
-35
-40
-45
-50
Phase (deg)
-90
-135
-180
P.M.: inf
Freq: NaN
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
Frequency (rad/s)

//

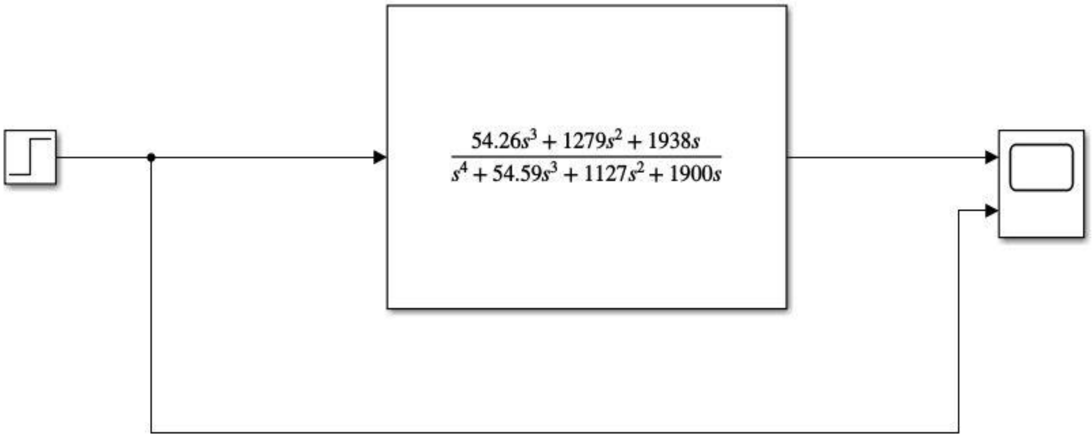
Transfer Function after the PID

K_p	**33**
K_d	**1.4**
K_i	**50**

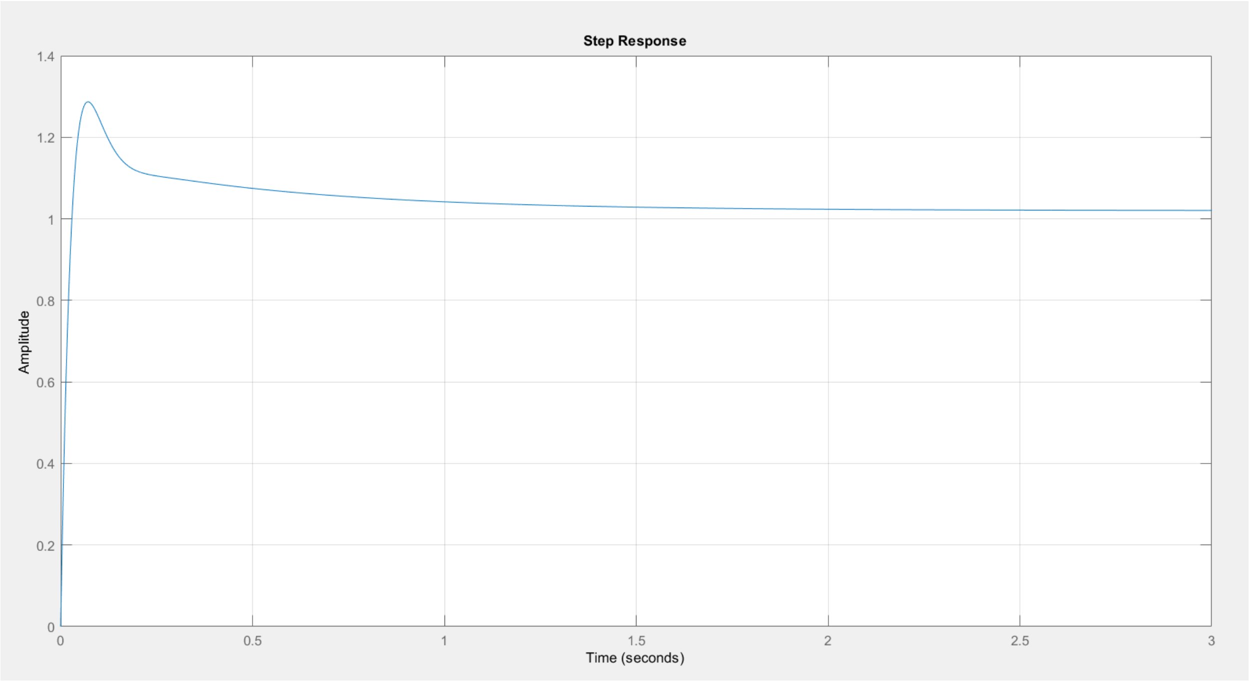
$$\frac{54.26s^3+1279s^2+1938s}{s^4+54.59s^3+1127s^2+1900s}$$



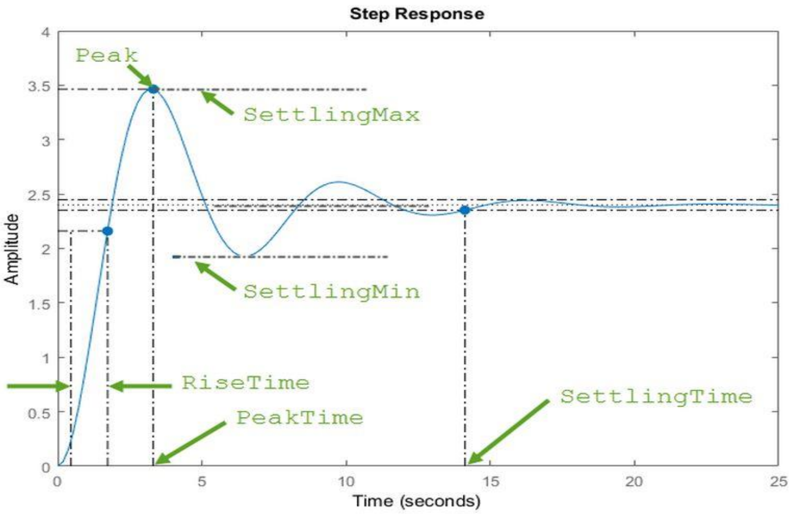
1
33
50
1.4
$\frac{1}{s}$
$\frac{\Delta u}{\Delta t}$
$\frac{38.7596899s}{s^3+0.32267s^2-152.093s-38.02325581}$

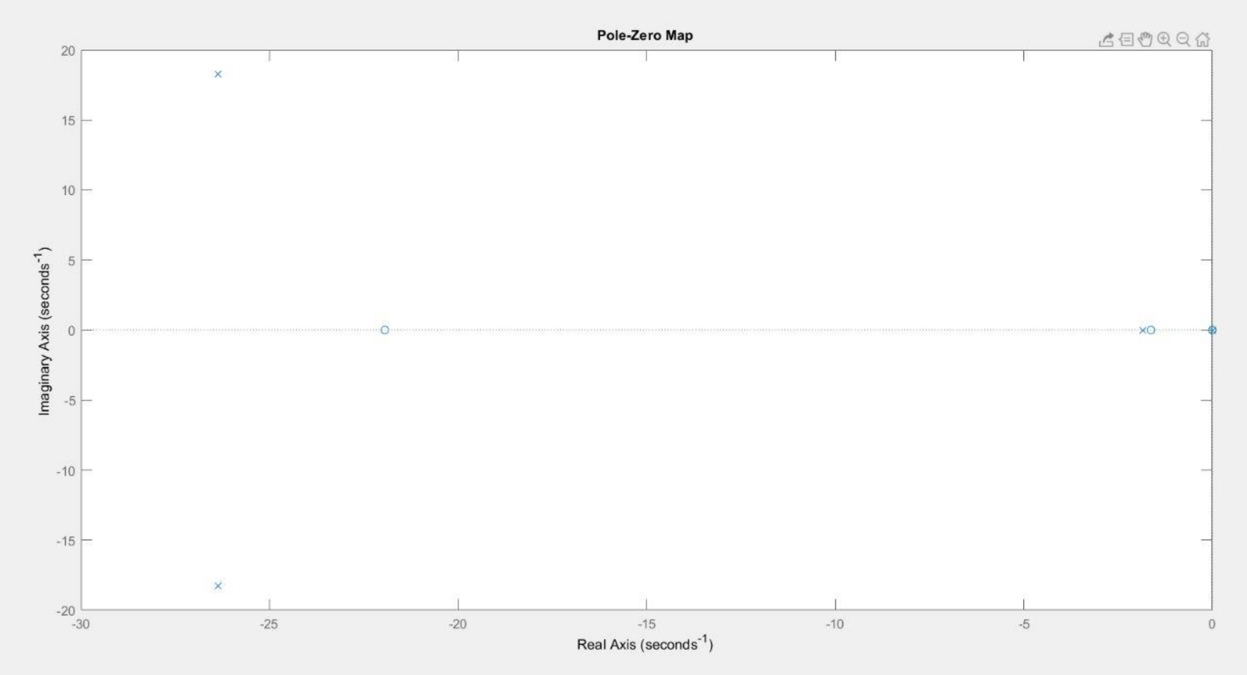


Response after PID Control

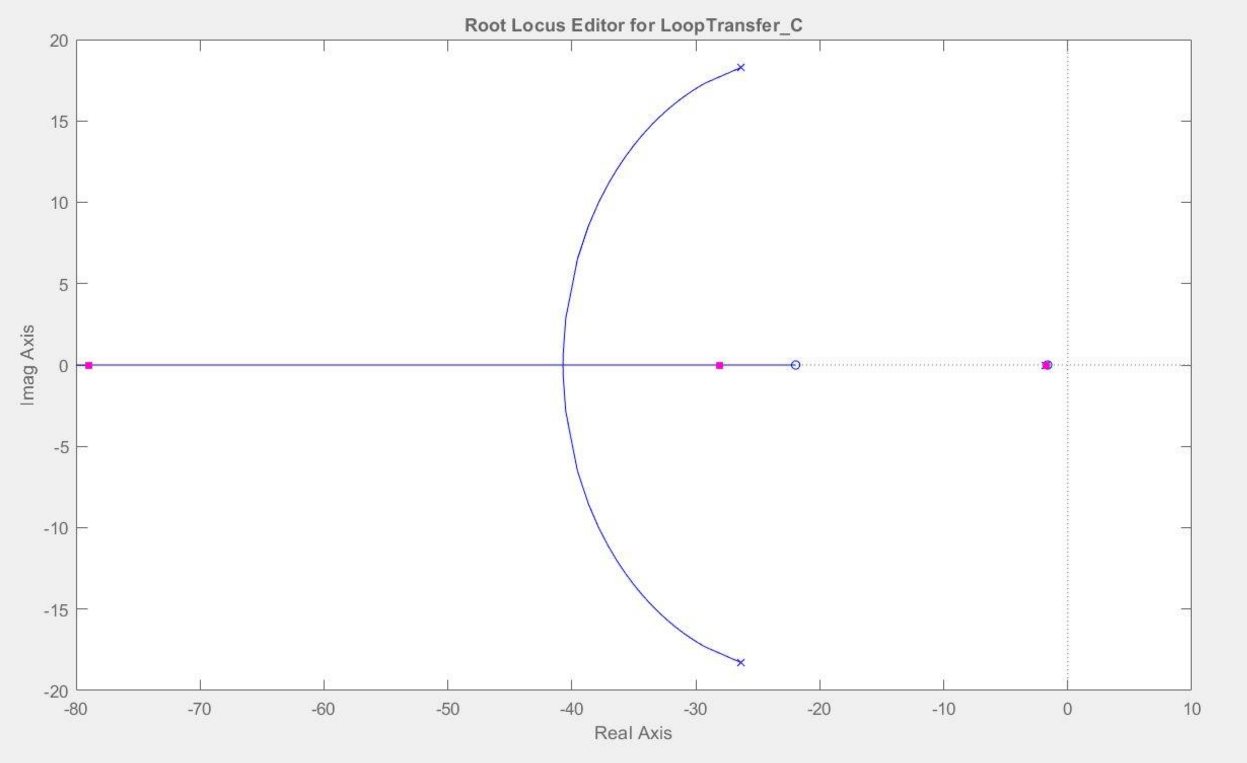


- RiseTime: 0.0233
- SettlingTime: 1.0341
- SettlingMin: 0.9393
- SettlingMax: 1.2870
- Overshoot: 26.1779
- Peak: 1.2870
- PeakTime: 0.0716

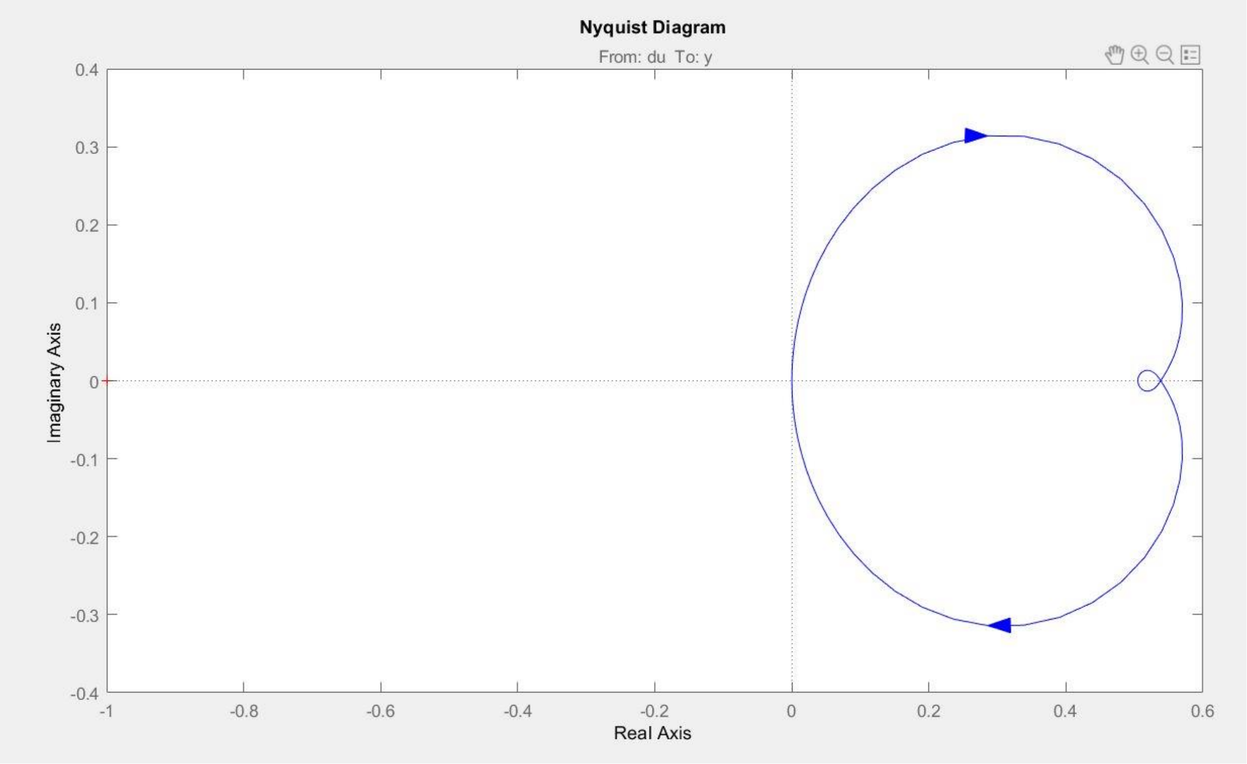




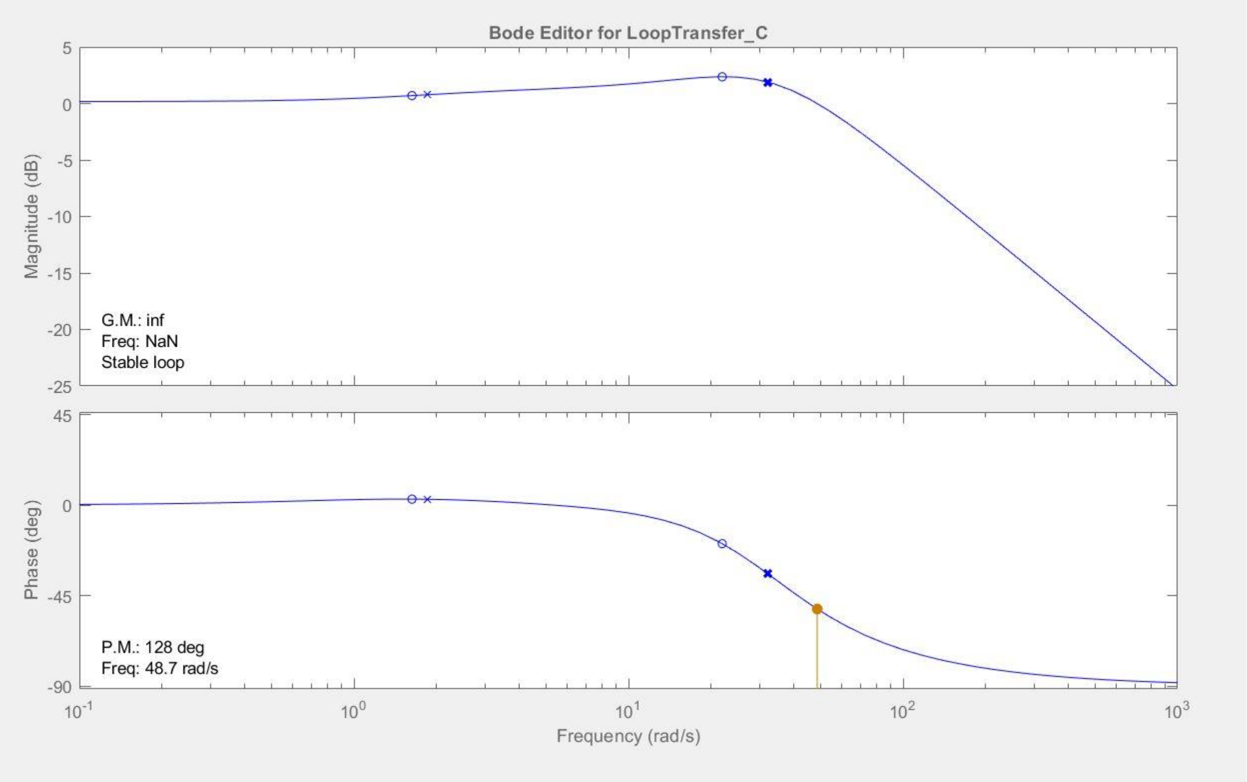
The system has 4 poles (all at left) (P1(-26.4-18.3j), P2(-26.4+18.3j), P3(-1.85),P4(0)), 3 zeros(Z1(-21.9),Z2(-1.63),Z3(0))



Take-off point (-40.7), there is not pole on the right, the system is stable forever.



No rotation about -1, there is no poles on right, so the system is stable. **(Z=N+P = 0)**



//

Real application

Self-balancing unicycle

a self-powered unicycle that balances itself in three dimensions. The conveyance strips away everything unnecessary (like pedals and handlebars) and then adds advanced technology (like gyroscopic acceleration and regenerative braking) for an ultra-fun ride.

Figure1. 11Self-balancing unicycle

Monocycle Is the World's First self-Balancing Bike

The Monocycle is a perfect city bike that can be used for short distance travels and can be parked in the least available of spaces. Moreover, the bicycle comes with a strong electric motor to support the riders, so they need not push too hard to speed up the bike.

The cycle works on the principle that the rider controls the speed of the cycle by his movement like leaning forward or backward for speeding up or stopping. Also, the gadgets used in this device are quite advanced like the accelerometers and gyroscopes which have a lot of balancing algorithms.

Figure1. 12 Monocycle

Segway Robot

can be used in such a way as to travel forward or backward, they can even be used as vehicles by humans. Robotic designing uses two sensors. An accelerometer, an Electro-mechanic device that measures the Inertial Acceleration Force, while Gyroscope measures the body Angular Velocity. Using an improved technology based on public transportation as a basic solution is one of the viewpoints. Among the various methods, Mptds can help, improving the transportation without the car and for short trips.

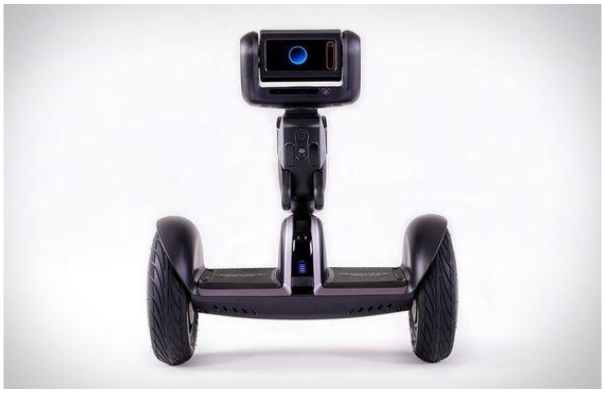


Figure1. 13: Segway Robot

Hoverboard, also known as a self-balancing scooter.

Is a personal transportation device that uses gyroscopic sensors, accelerometers, and microprocessors to maintain balance and propel the rider forward or backward. The rider controls the hoverboard by shifting their body weight, and the device adjusts the motor speed to maintain stability.

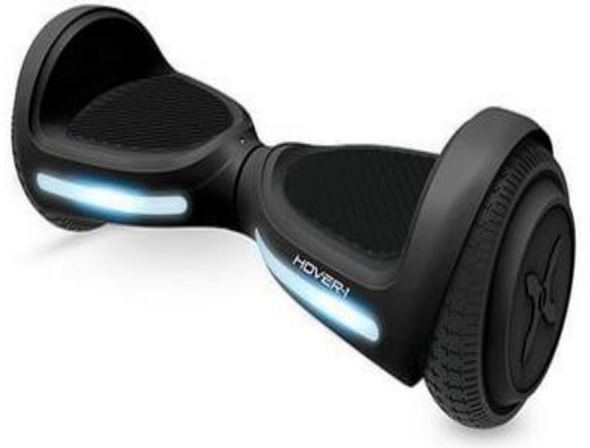


Figure1. 14: Hoverboard

Self-balancing Wheelchairs.

Self-balancing wheelchairs are innovative mobility devices that use advanced technology to provide enhanced maneuverability, stability, and independence for people with mobility impairments. They employ sensors, actuators, and control systems to maintain balance while navigating different terrains. These wheelchairs offer benefits such as improved maneuverability, stability on various surfaces, increased independence, and enhanced user safety. They aim to improve accessibility, freedom, and quality of life for individuals with mobility impairments.

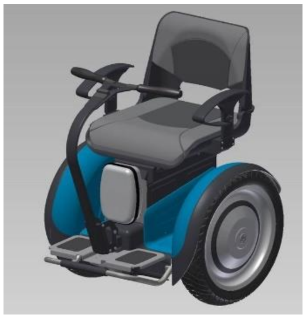


Figure1. 15: Self-balancing Wheelchairs

.

References

PID Control:

1. Astrom, K. J., & Hagglund, T. (1995). PID Controllers: Theory, Design, and Tuning. Instrument Society of America.
2. Åström, K. J., & Murray, R. M. (2008). Feedback Systems: An Introduction for Scientists and Engineers. Princeton University Press.
3. Ogata, K. (2010). Modern Control Engineering. Pearson Education.

Self-Balancing Robots:

1. Balch, T., & Hybinette, M. (1999). Balance control strategies for a two-wheeled dynamically stable robot. Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems.
2. Mehta, A., Kumar, S., & Varadarajan, S. (2015). PID controller-based balancing of an inverted pendulum. Procedia Computer Science, 58, 402-407.
3. Zhang, L., & Wang, Y. (2020). PID Control and Fuzzy Control Algorithms for Self-Balancing Robot. Proceedings of the 2020 International Conference on Smart Manufacturing and Factory Automation.

Websites:

Self-Balancing Robots:

1. https://www.researchgate.net/publication/341052911