National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

EE-379 Control Systems

Final Project Report

Self-Balancing Robot

Group Members:

1.	Faris Imran	135083
2.	Haris Mahmood	189641
3.	M. Ali Haider	179947
4.	Abdullah Ashfaq	129399

2nd May, 2019

Table of Contents

Idea

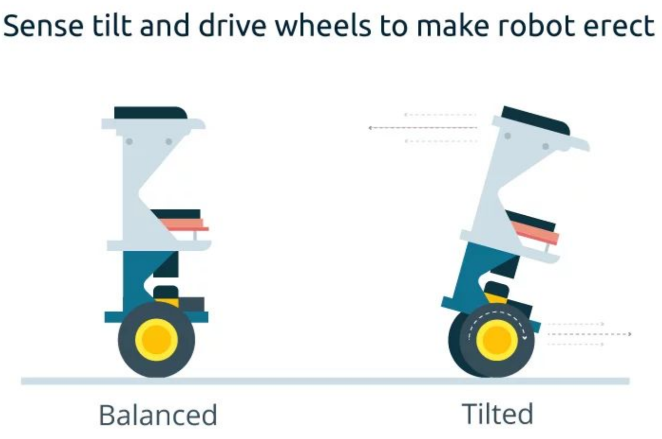
We will design a self-balancing robot. The primary goal of the project is to prevent the robot from falling over. If the time permits, we may extend the project by adding the capability to steer remotely.

Motivation

In the past decade, mobile robots have stepped out of the military and industrial settings, and entered civilian and personal spaces such as hospitals, schools and ordinary homes. While many of these robots for civil applications are mechanically stable, such as Aibo the Sony robotic dog, or four-wheel vacuum cleaners, one that ordinary on-lookers would find awe-inspiring is the Segway personal transport, a mechanically unstable, two-wheel self-balancing vehicle that has seen deployment for law-enforcement, tourism etc. The popularity of self-balancing robots has ignited active research on the control design for such platforms. It is fast becoming an important topic in both education and product development and therein lies our motivation for choosing self-balancing robots as our control systems project.

Background

The self-balancing robot is similar to an inverted pendulum. Unlike a normal pendulum which keeps on swinging once given a nudge, this inverted pendulum cannot stay balanced on its own. It will simply fall over.



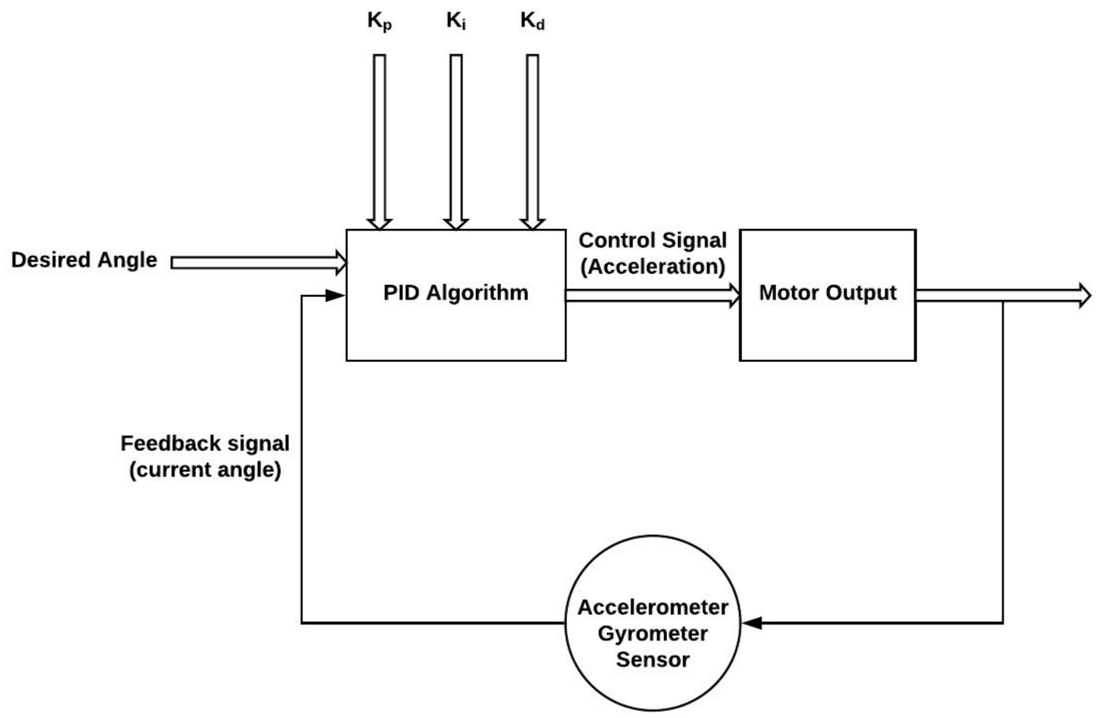
To prevent the robot from falling, the wheels have to move in a way which counters the applied force. In this regard, knowing the tilt angle is imperative. There are a wide array of sensors that can be used, such as inclinometers, light sensors, accelerometer or gyroscopes. However, each of these sensors have their shortcomings; the inclinometer takes a long time to converge to the angle it is currently at, light sensors are highly susceptible to background noise (ambient light and the reflective index of the surface it is operating in), gyroscopes have a bias and accelerometers are relatively noisy. We will use accelerometer and gyroscope for this project.

Methodology

To fulfill the purpose, the following method will be used:

- Derive dynamical equations based on the theory of the inverted pendulum
- Form transfer functions for the angle deviation, ψ, and position, x
- Find a controller that can control these two conditions
- Design a circuit
- Interface motors and sensors with the microcontroller
- Designing a chassis and assembling

Block Design


K_p
K_i
K_d
Desired Angle
PID Algorithm
Control Signal (Acceleration)
Motor Output
Feedback signal (current angle)
Accelerometer Gyrometer Sensor

The basic idea behind a PID controller or algorithm is to read a sensor, then compute the desired actuator output by calculating proportional, integral, and derivative responses and summing those three components to compute the output.

Mathematical Modeling

The system of interest is shown in Figure 1, where *F* is the force in Newtons, *m* is the mass of the pendulum rod in kilograms, *M* is the mass of the moving cart in kilograms, F_V is the force applied to the cart in Newtons, F_f is the force due to friction in Newtons,

g is the acceleration due to gravity in m·s^{-2}, and θ is the angle of the inverted pendulum measured from the vertical y-axis in radians.

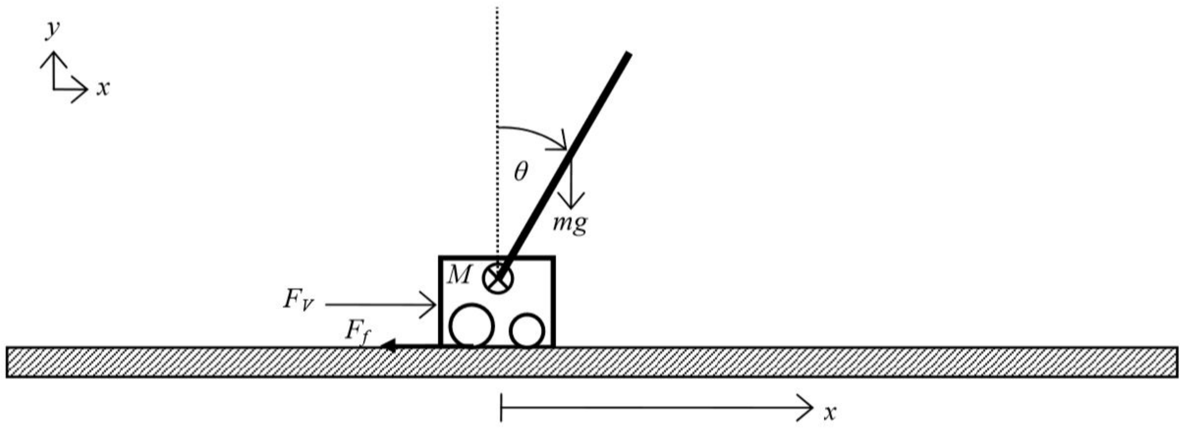


Figure 1: Schematic of the Inverted Pendulum System

Consider the free body diagrams shown in Figure 2. Furthermore, assume that the coordinates of the centroid (centre of gravity) of the pendulum, (x_G, y_G), are given by

$$x_G = x + l\sin\theta$$

$$y_G = l\cos\theta \qquad (1)$$

where l is the distance along the pendulum to the centre of gravity and x is the x-coordinate of the cart's position.

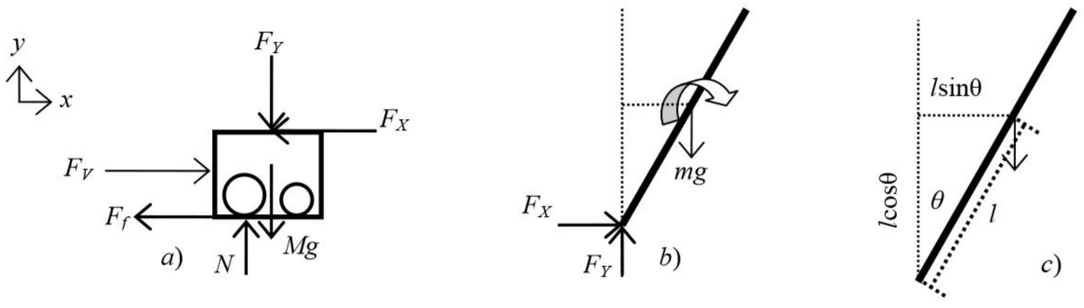


Figure 2: Free body Diagrams of (a) the Cart and (b) the Pendulum. (c) Determining the Required Distances.

For the horizontal motion of the cart, Newton's second law of motion

$$\sum F = M\frac{d^2x}{dt^2} \tag{2}$$

can be written as

$$M\frac{d^2x}{dt^2} = F_V - F_X - F_f \tag{3}$$

since, as can be verified from Figure 2a, there are only three forces acting in the x-direction. Assume that the friction force can be written as

$$F_f = \gamma_2\frac{dx}{dt} \tag{4}$$

Substituting Equation (4) into Equation (3) gives

$$M\frac{d^2x}{dt^2} = F_V - F_X - \gamma_2\frac{dx}{dt} \tag{5}$$

Similarly, from Equation (2), the horizontal motion of the pendulum can be written as

$$F_X = m\frac{d^2x_G}{dt^2} \tag{6}$$

The derivative on the right in Equation (6) can be simplified by determining the derivative of x_G using Equation (1). The first derivative can be found as follows

$$\begin{aligned}\frac{dx_G}{dt} &= \frac{d\left(x + l\sin\theta\right)}{dt}\\ &= \frac{dx}{dt} + l\frac{d\left(\sin\theta\right)}{dt}\\ &= \frac{dx}{dt} + l\cos\theta\frac{d\theta}{dt}\end{aligned} \tag{7}$$

where, since θ is a function of time, the chain rule was applied to the second line in order to obtain the final form of the first derivative. The second order derivative can be found by differentiating Equation (7), that is

$$\begin{aligned}\frac{d^2x_G}{dt^2} &= \frac{d}{dt}\left(\frac{dx}{dt}+l\cos\theta\frac{d\theta}{dt}\right)\\ &= \frac{d^2x}{dt}+l\frac{d}{dt}\left(\cos\theta\frac{d\theta}{dt}\right)\\ &= \frac{d^2x}{dt}+l\left(\frac{d\cos\theta}{dt}\frac{d\theta}{dt}+\cos\theta\frac{d^2\theta}{dt^2}\right)\\ &= \frac{d^2x}{dt}+l\left(-\sin\theta\left(\frac{d\theta}{dt}\right)^2+\cos\theta\frac{d^2\theta}{dt^2}\right)\\ &= \frac{d^2x}{dt}-l\sin\theta\left(\frac{d\theta}{dt}\right)^2+l\cos\theta\frac{d^2\theta}{dt^2}\end{aligned} \tag{8}$$

Combining Equation (8) with Equation (6) gives

$$F_X = m\left(\frac{d^2x}{dt^2}-l\sin\theta\left(\frac{d\theta}{dt}\right)^2+l\cos\theta\frac{d^2\theta}{dt^2}\right) \tag{9}$$

Using Equation (9), Equation (5) can be simplified to give

$$\begin{aligned}M\frac{d^2x}{dt^2} &= F_V - m\left(\frac{d^2x}{dt^2}-l\sin\theta\left(\frac{d\theta}{dt}\right)^2+l\cos\theta\frac{d^2\theta}{dt^2}\right)-\gamma_2\frac{dx}{dt}\\ M\frac{d^2x}{dt^2} &= F_V - m\frac{d^2x}{dt^2}+ml\sin\theta\left(\frac{d\theta}{dt}\right)^2-ml\cos\theta\frac{d^2\theta}{dt^2}-\gamma_2\frac{dx}{dt}\end{aligned} \tag{10}$$

The final form for the horizontal motion of the cart can be given as

$$\boxed{(M+m)\frac{d^2x}{dt^2}+\gamma_2\frac{dx}{dt} = F_V + ml\sin\theta\left(\frac{d\theta}{dt}\right)^2-ml\cos\theta\frac{d^2\theta}{dt^2}} \tag{11}$$

For the vertical motion of the pendulum, Equation (2) can be written as

$$F_Y - mg = m\frac{d^2y_G}{dt^2} \tag{12}$$

Similarly to the horizontal case, the derivative on the right in Equation (12) can be written as follow

$$\frac{dy_G}{dt} = \frac{d(l\cos\theta)}{dt} = -l\sin\theta\frac{d\theta}{dt} \tag{13}$$

$$\begin{aligned} \frac{d^2y_G}{dt^2} &= \frac{d}{dt}\left(-l\sin\theta\frac{d\theta}{dt}\right) \\ &= -l\left(\frac{d\sin\theta}{dt}\frac{d\theta}{dt} + \sin\theta\frac{d^2\theta}{dt^2}\right) \\ &= -l\left(\cos\theta\left(\frac{d\theta}{dt}\right)^2 + \sin\theta\frac{d^2\theta}{dt^2}\right) \\ &= -l\cos\theta\left(\frac{d\theta}{dt}\right)^2 - l\sin\theta\frac{d^2\theta}{dt^2} \end{aligned} \tag{14}$$

Using Equation (14), Equation (12) can be rewritten to give

$$F_Y - mg = m\left(-l\cos\theta\left(\frac{d\theta}{dt}\right)^2 - l\sin\theta\frac{d^2\theta}{dt^2}\right) \tag{15}$$

Thus, the vertical reaction force, F_Y, can be written as

$$F_Y = mg + m\left(-l\cos\theta\left(\frac{d\theta}{dt}\right)^2 - l\sin\theta\frac{d^2\theta}{dt^2}\right) \tag{16}$$

For any object, the relationship between the moment applied on an object and its angular acceleration is given by the following relationship

$$\sum\bar{M} = I\frac{d^2\theta}{dt^2} \tag{17}$$

where M^{-} is the moment due to a given force and defined as

$$\bar{M} = \vec{F}\times\vec{r} \tag{18}$$

where F is the force vector, r is the position vector of the object with respect to the point about which the moments are being summed, and I is the angular momentum of the object. For the pendulum, summing the moment around its centre of gravity, Equation F G G (17) can be written as

$$F_Y l\sin\theta - F_X l\cos\theta = I\frac{d^2\theta}{dt^2} \tag{19}$$

Substituting Equation (16) for F_Y and Equation (9) for F_X into Equation (19) gives

$$\begin{aligned}&\left(mg + m\left(-l\cos\theta\left(\frac{d\theta}{dt}\right)^2 - l\sin\theta\frac{d^2\theta}{dt^2}\right)\right)l\sin\theta \\ &-\left(m\left(\frac{d^2x}{dt^2} - l\sin\theta\left(\frac{d\theta}{dt}\right)^2 + l\cos\theta\frac{d^2\theta}{dt^2}\right)\right)l\cos\theta = I\frac{d^2\theta}{dt^2}\end{aligned} \tag{20}$$

Simplifying Equation (20) gives

$$\begin{aligned}&mgl\sin\theta - ml^2\sin\theta\cos\theta\left(\frac{d\theta}{dt}\right)^2 - ml^2\sin^2\theta\frac{d^2\theta}{dt^2} \\ &-ml\cos\theta\frac{d^2x}{dt} + ml^2\cos\theta\sin\theta\left(\frac{d\theta}{dt}\right)^2 - ml^2\cos^2\theta\frac{d^2\theta}{dt^2} = I\frac{d^2\theta}{dt^2}\end{aligned} \tag{21}$$

$$mgl\sin\theta - ml^2\left(\sin^2\theta + \cos^2\theta\right)\frac{d^2\theta}{dt^2} - ml\cos\theta\frac{d^2x}{dt} = I\frac{d^2\theta}{dt^2} \tag{22}$$

Since

$$\cos^2\theta + \sin^2\theta = 1 \tag{23}$$

Equation (22) can be simplified to give

$$\begin{aligned}&mgl\sin\theta - ml^2\frac{d^2\theta}{dt^2} - ml\cos\theta\frac{d^2x}{dt^2} = I\frac{d^2\theta}{dt^2} \\ &mgl\sin\theta - ml\cos\theta\frac{d^2x}{dt^2} = \left(I + ml^2\right)\frac{d^2\theta}{dt^2}\end{aligned} \tag{24}$$

Thus, the final equation for the angular position is given as

$$\boxed{\left(I + ml^2\right)\frac{d^2\theta}{dt^2} = mgl\sin\theta - ml\cos\theta\frac{d^2x}{dt^2}} \tag{25}$$

Therefore the equations of motion for the inverted pendulum on a moving cart can be written as

$$\boxed{\begin{cases}(M+m)\dfrac{d^2x}{dt^2} + \gamma_2\dfrac{dx}{dt} = F_V + ml\sin\theta\left(\dfrac{d\theta}{dt}\right)^2 - ml\cos\theta\dfrac{d^2\theta}{dt^2} \\ \left(I + ml^2\right)\dfrac{d^2\theta}{dt^2} = mgl\sin\theta - ml\cos\theta\dfrac{d^2x}{dt^2}\end{cases}} \tag{26}$$

For the system in the laboratory, the relationship for the force due to the voltage can be written as

$$F_V = \gamma_1 V \tag{27}$$

where γ_1 is conversion factor and V is the applied voltage in volts.

Linearised Model of the System

The model of the system given by Equation (26) is nonlinear and must be linearised in order to obtain a reasonable model for control purposes. Linearisation will be performed about the point x = 0 m and θ = 0 radians. Furthermore, it will be assumed that since θ is small

$$\begin{aligned} &\sin\theta \approx \theta \\ &\cos\theta \approx 1 \\ &\left(\frac{d\theta}{dt}\right)^2 \approx 0 \end{aligned} \tag{28}$$

Under these assumptions, Equation (26) can be rewritten as

$$\begin{cases} (M+m)\dfrac{d^2x}{dt^2} + \gamma_2\dfrac{dx}{dt} = F_V - ml\dfrac{d^2\theta}{dt^2} \\ (I+ml^2)\dfrac{d^2\theta}{dt^2} = mgl\theta - ml\dfrac{d^2x}{dt^2} \end{cases} \tag{29}$$

If it is assumed that the centre of the mass of the pendulum is equal to its centre of gravity, then I = 0, and Equation (29) reduces to

$$\begin{cases} (M+m)\dfrac{d^2x}{dt^2} + \gamma_2\dfrac{dx}{dt} = F_V - ml\dfrac{d^2\theta}{dt^2} \\ l\dfrac{d^2\theta}{dt^2} = g\theta - \dfrac{d^2x}{dt^2} \end{cases} \tag{30}$$

Substituting Equation (27) into Equation (30), gives

$$\begin{cases} (M+m)\dfrac{d^2x}{dt^2} + \gamma_2\dfrac{dx}{dt} = \gamma_1 V - ml\dfrac{d^2\theta}{dt^2} \\ l\dfrac{d^2\theta}{dt^2} = g\theta - \dfrac{d^2x}{dt^2} \end{cases} \tag{31}$$

Therefore, the linearised equations of motion for the pendulum and moving cart can be written as

$$\boxed{\begin{cases}(M+m)\dfrac{d^2x}{dt^2}+ml\dfrac{d^2\theta}{dt^2}+\gamma_2\dfrac{dx}{dt}=\gamma_1 V\\ l\dfrac{d^2\theta}{dt}+\dfrac{d^2x}{dt}=g\theta\end{cases}} \tag{32}$$

In order to uncouple the equations, that is force each equation to contain only derivatives that are functions of either θ or x, substitute Equation (32).2 into Equation (32).1. This gives

$$\begin{aligned}(M+m)\frac{d^2x}{dt^2}+ml\left(\frac{g\theta}{l}-\frac{1}{l}\frac{d^2x}{dt}\right)+\gamma_2\frac{dx}{dt}&=\gamma_1 V\\ (M+m)\frac{d^2x}{dt^2}+mg\theta-m\frac{d^2x}{dt}+\gamma_2\frac{dx}{dt}&=\gamma_1 V\\ M\frac{d^2x}{dt^2}+mg\theta+\gamma_2\frac{dx}{dt}&=\gamma_1 V\\ M\frac{d^2x}{dt^2}+\gamma_2\frac{dx}{dt}&=\gamma_1 V-mg\theta\\ \frac{d^2x}{dt^2}+\frac{\gamma_2}{M}\frac{dx}{dt}&=\frac{\gamma_1}{M}V-\frac{mg}{M}\theta\\ \frac{d^2x}{dt^2}&=-\frac{mg}{M}\theta-\frac{\gamma_2}{M}\frac{dx}{dt}+\frac{\gamma_1}{M}V\end{aligned} \tag{33}$$

Substituting Equation (33) into Equation (32).2 gives

$$\begin{aligned}l\frac{d^2\theta}{dt}+\left(-\frac{mg}{M}\theta-\frac{\gamma_2}{M}\frac{dx}{dt}+\frac{\gamma_1}{M}V\right)&=g\theta\\ l\frac{d^2\theta}{dt}&=g\theta+\frac{mg}{M}\theta+\frac{\gamma_2}{M}\frac{dx}{dt}-\frac{\gamma_1}{M}V\\ \frac{d^2\theta}{dt}&=\frac{M+m}{Ml}g\theta+\frac{\gamma_2}{Ml}\frac{dx}{dt}-\frac{\gamma_1}{Ml}V\end{aligned} \tag{34}$$

Thus, the uncoupled form of the equations of motion for this system are given as

$$\boxed{\begin{cases} \dfrac{d^2x}{dt^2} = -\dfrac{mg}{M}\theta - \dfrac{\gamma_2}{M}\dfrac{dx}{dt} + \dfrac{\gamma_1}{M}V \\ \dfrac{d^2\theta}{dt} = \dfrac{M+m}{Ml}g\theta + \dfrac{\gamma_2}{Ml}\dfrac{dx}{dt} - \dfrac{\gamma_1}{Ml}V \end{cases}} \tag{35}$$

The state-space form of a linear equation is given by

$$\frac{d\vec{x}}{dt} = A\vec{x} + B\vec{u} \tag{36}$$

where x is the state vector, u is the input vector, A is the state matrix, and B is the input matrix. If it is assumed that

$$\vec{x} = \begin{bmatrix} x \\ \theta \\ \dfrac{dx}{dt} \\ \dfrac{d\theta}{dt} \end{bmatrix}, \vec{u} = V \tag{37}$$

Equation (35) can be rewritten into state-space form by defining the matrices to be

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\dfrac{mg}{M} & -\dfrac{\gamma_2}{M} & 0 \\ 0 & \dfrac{M+m}{Ml}g & \dfrac{\gamma_2}{Ml} & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ \dfrac{\gamma_1}{M} \\ -\dfrac{\gamma_1}{Ml} \end{bmatrix} \tag{38}$$

Thus, the state-space can be given as

$$\begin{bmatrix} \dfrac{dx}{dt} \\ \dfrac{d\theta}{dt} \\ \dfrac{d^2x}{dt^2} \\ \dfrac{d^2\theta}{dt^2} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\dfrac{mg}{M} & -\dfrac{\gamma_2}{M} & 0 \\ 0 & \dfrac{M+m}{Ml}g & \dfrac{\gamma_2}{Ml} & 0 \end{bmatrix} \begin{bmatrix} x \\ \theta \\ \dfrac{dx}{dt} \\ \dfrac{d\theta}{dt} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dfrac{\gamma_1}{M} \\ -\dfrac{\gamma_1}{Ml} \end{bmatrix} V \tag{39}$$

System Modelling in MATLAB

We can implement any of the aforementioned forms for the representation of our system in MATLAB. For our case, we are using the State-Space representation and we have shown that one form can be converted into another with just a single line of command in MATLAB. The following code demonstrates the use of the state matrix (**A**), input matrix (**B**), output matrix (**C**), feed-forward matrix (**D**) to create the continuous-time state-space model of our system.

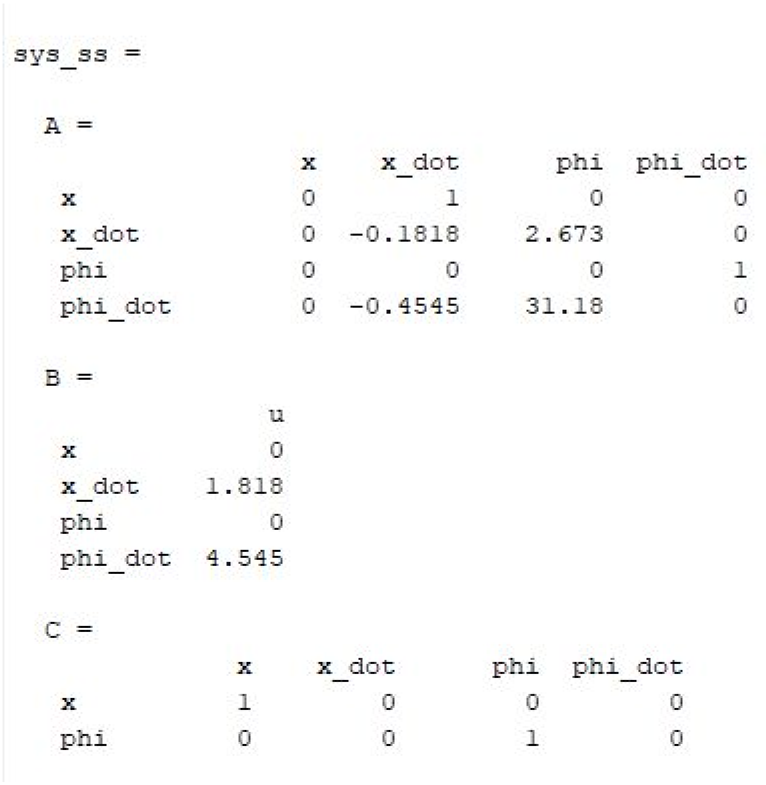
One thing to note here is that our system cannot be modelled as a SISO system since the pendulum angle and the cart position are both outputs of our system, hence this is a case of **Single Input-Multiple Outputs (SIMO)** system. Hence, this system results in two transfer functions in the frequency domain analysis, namely

1. Between the pendulum angle & the input force and,
2. Between the cart position & the input force, as shown below.

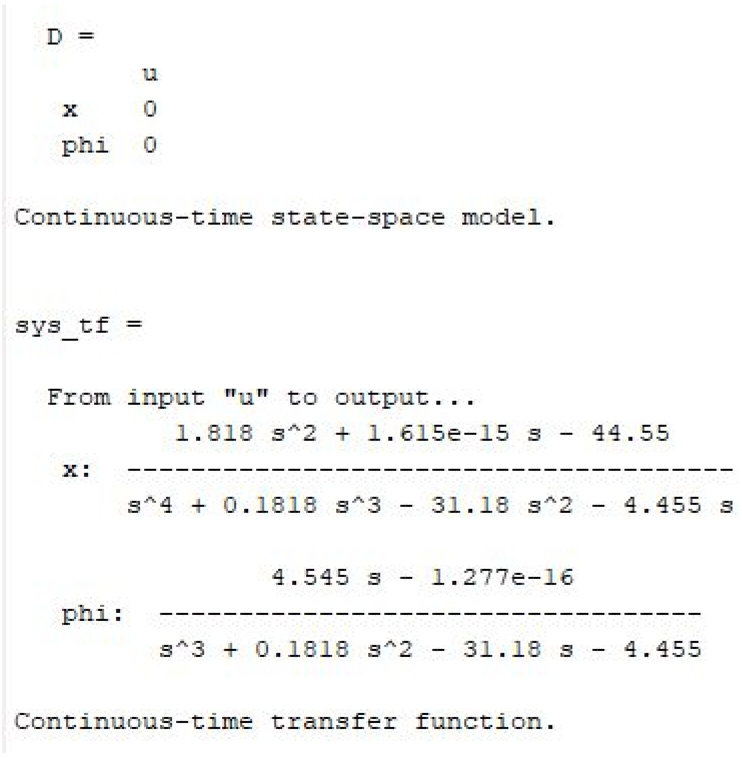
$$P_{pend}(s) = \frac{\Phi(s)}{F(s)} = \frac{\frac{ml}{q}s}{s^3 + \frac{b(I+ml^2)}{q}s^2 - \frac{(M+m)mgl}{q}s - \frac{bmgl}{q}}$$

$$P_{cart}(s) = \frac{X(s)}{F(s)} = \frac{\frac{(I+ml^2)s^2 - gml}{q}}{s^4 + \frac{b(I+ml^2)}{q}s^3 - \frac{(M+m)mgl}{q}s^2 - \frac{bmgl}{q}s}$$

where, $q = (M+m)l + ml^2 - (ml)^2$



```
sys_ss =

  A =
                     x     x_dot       phi   phi_dot
   x                 0         1         0         0
   x_dot             0   -0.1818     2.673         0
   phi               0         0         0         1
   phi_dot           0   -0.4545     31.18         0

  B =
                 u
   x             0
   x_dot     1.818
   phi           0
   phi_dot   4.545

  C =
                 x     x_dot       phi   phi_dot
   x             1         0         0         0
   phi           0         0         1         0
```



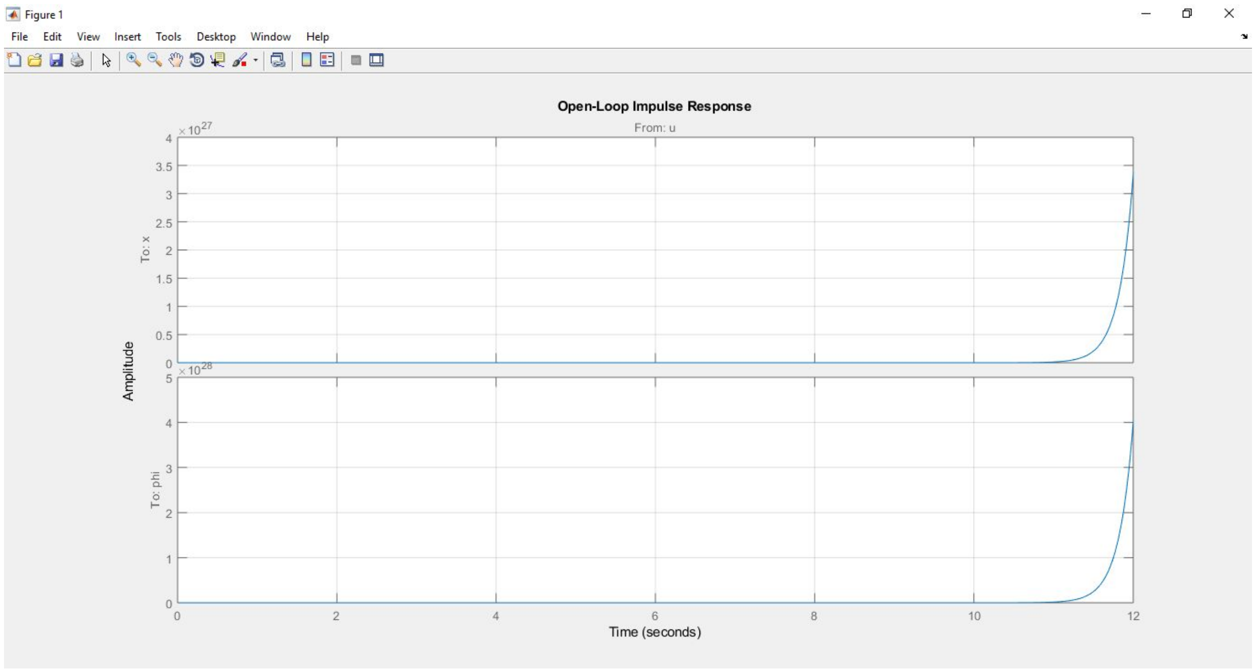
```
  D =
         u
   x     0
   phi   0

Continuous-time state-space model.


sys_tf =

  From input "u" to output...
            1.818 s^2 + 1.615e-15 s - 44.55
   x:  --------------------------------------
       s^4 + 0.1818 s^3 - 31.18 s^2 - 4.455 s

                 4.545 s - 1.277e-16
   phi:  ----------------------------------
         s^3 + 0.1818 s^2 - 31.18 s - 4.455

Continuous-time transfer function.
```

System Analysis in MATLAB

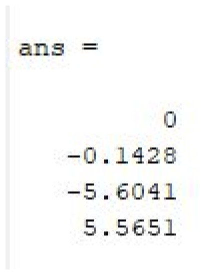
We begin by analysing our system in MATLAB for a single impulse input or disturbance and analysing the variation in our system outputs i.e.

1. Pendulum Angle (**Φ**)
2. Cart Position (**x**)

The open-loop impulse responses are given below:

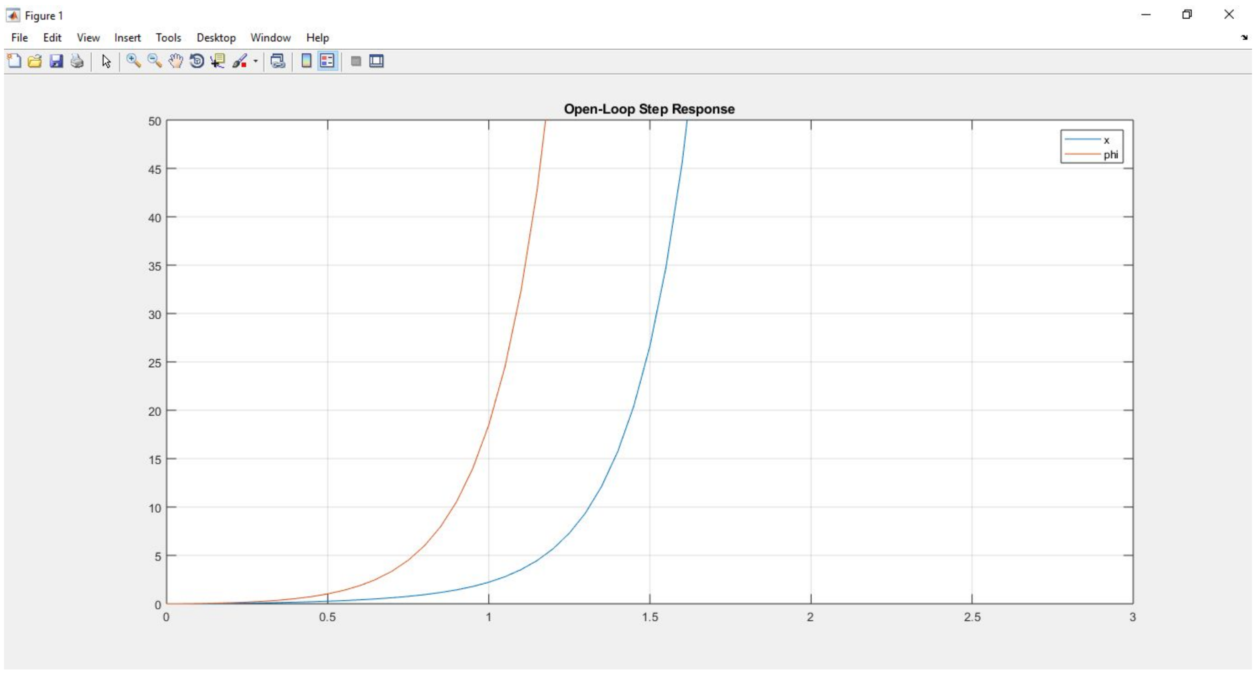


The observation of these responses shows us that the system is unstable no matter which output we wish to observe. The poles of the system can also give us sufficient information on the stability of our system. Since our system follows the SIMO type, we can either use both of our transfer functions to find separate poles of the two transfer functions. Another alternative can be to find the Eigenvalues of our state-space model which is analogous to finding the poles of the system.



```
ans =

         0
   -0.1428
   -5.6041
    5.5651
```

As shown above, one of the Eigenvalues lies in the right half place, making our system unstable, therefore the response to a step input will also grow unbounded. The open-loop step response of the system is shown below.



We can also find some characteristics to judge the responses of the system e.g. the root loci, the maximum values, settling times etc.

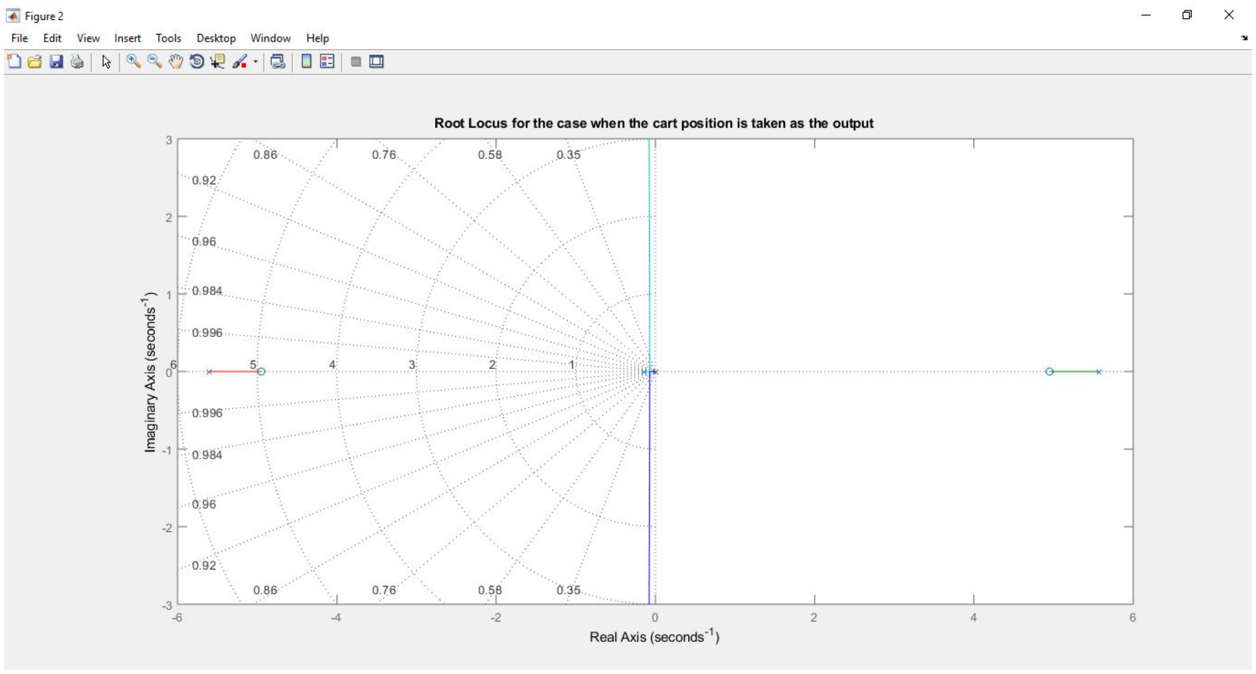


Figure 2
File Edit View Insert Tools Desktop Window Help
Root Locus for the case when the cart position is taken as the output
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
0.86
0.76
0.58
0.35
0.92
0.96
0.984
0.996
6
5
4
3
2
1

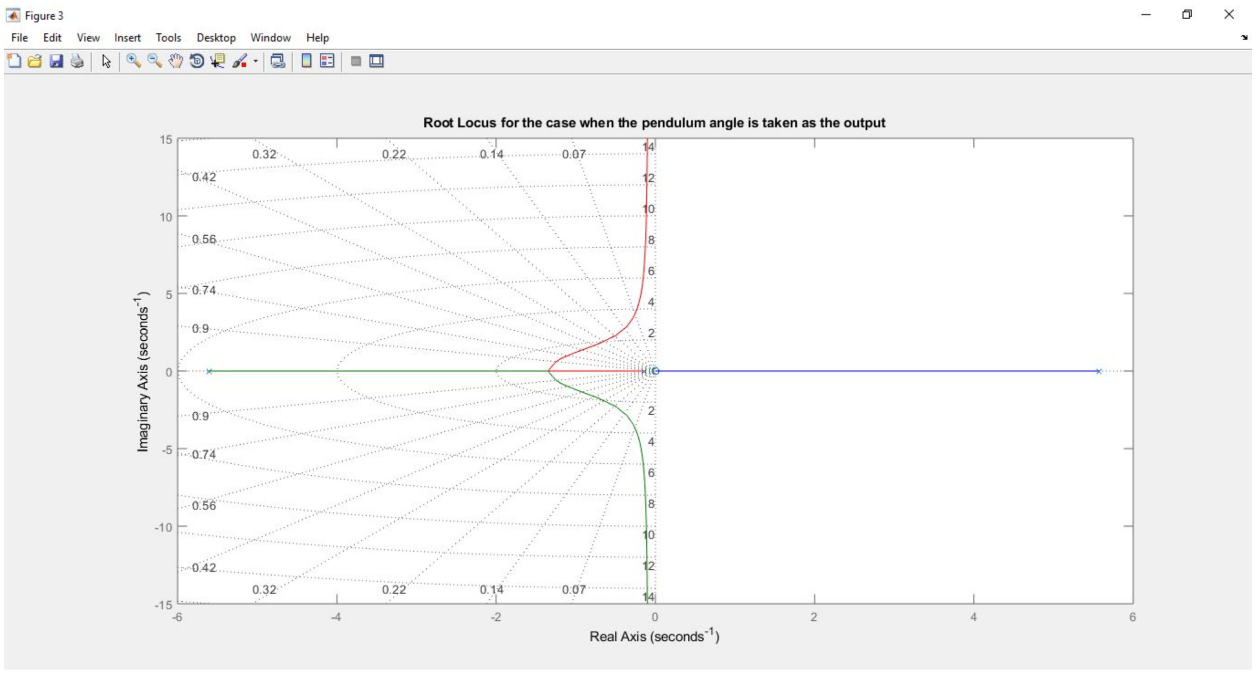
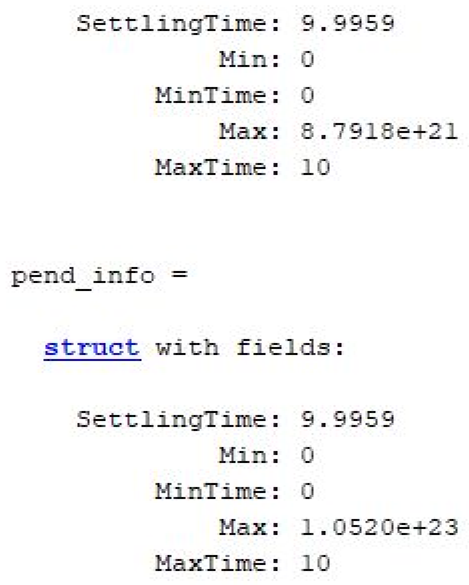


Figure 3
File Edit View Insert Tools Desktop Window Help
Root Locus for the case when the pendulum angle is taken as the output
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
0.32
0.22
0.14
0.07
0.42
0.56
0.74
0.9



```
  SettlingTime: 9.9959
           Min: 0
       MinTime: 0
           Max: 8.7918e+21
       MaxTime: 10


pend_info =

  struct with fields:

  SettlingTime: 9.9959
           Min: 0
       MinTime: 0
           Max: 1.0520e+23
       MaxTime: 10
```

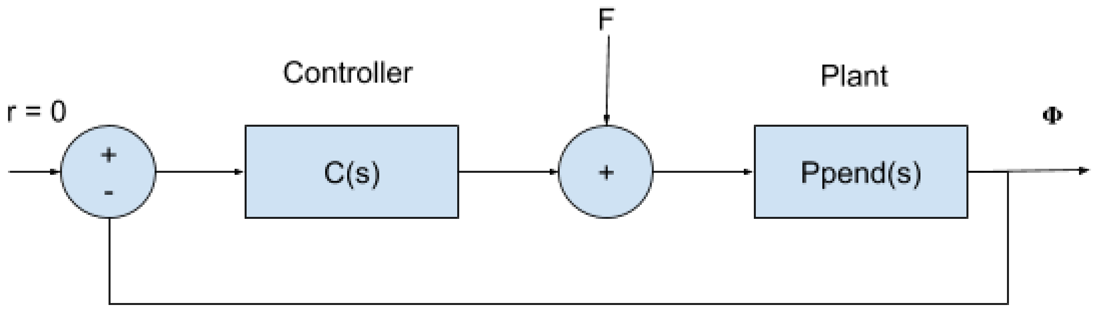
It is apparent from the analysis above that some sort of control will need to be designed to improve the response of the system.

PID Controller Design

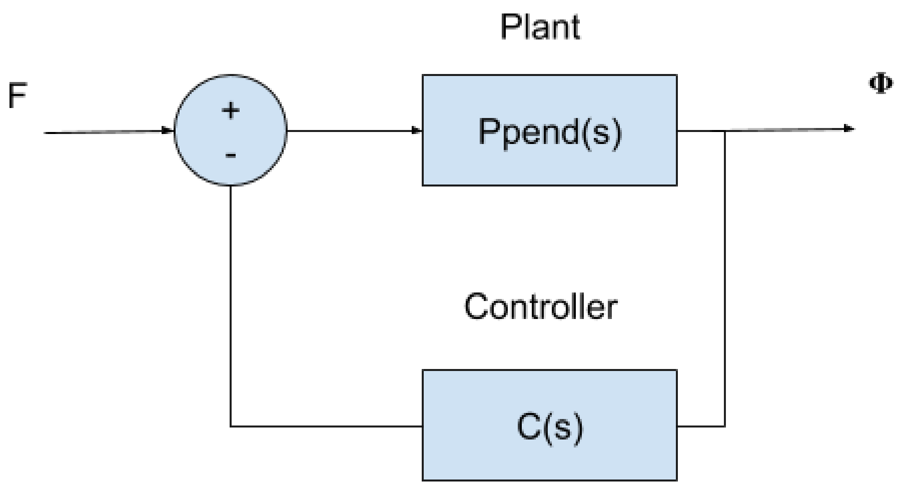
In order to design a PID controller for our system, we will be assuming our system to be a **Single Input-Single Output (SISO)** with the pendulum angle in consideration. The corresponding transfer function is given by:

$$P_{pend}(s) = \frac{\Phi(s)}{F(s)} = \frac{\frac{ml}{q}s}{s^3 + \frac{b(I+ml^2)}{q}s^2 - \frac{(M+m)mgl}{q}s - \frac{bmgl}{q}}$$

The requirement for our controller would be to maintain our pendulum vertically upwards after an initial disturbance force acts on the system. This disturbance can be considered as an impulse in nature, with the reference pendulum angle to be 0 deg/rad. The block diagram of this subsystem in consideration, is given below:



This block diagram can be represented in the following way for ease of analysis, showing the impulsive input disturbance force ***F*** as the input and the pendulum angle ***Φ*** as the output of the system:



The resulting transfer function becomes:

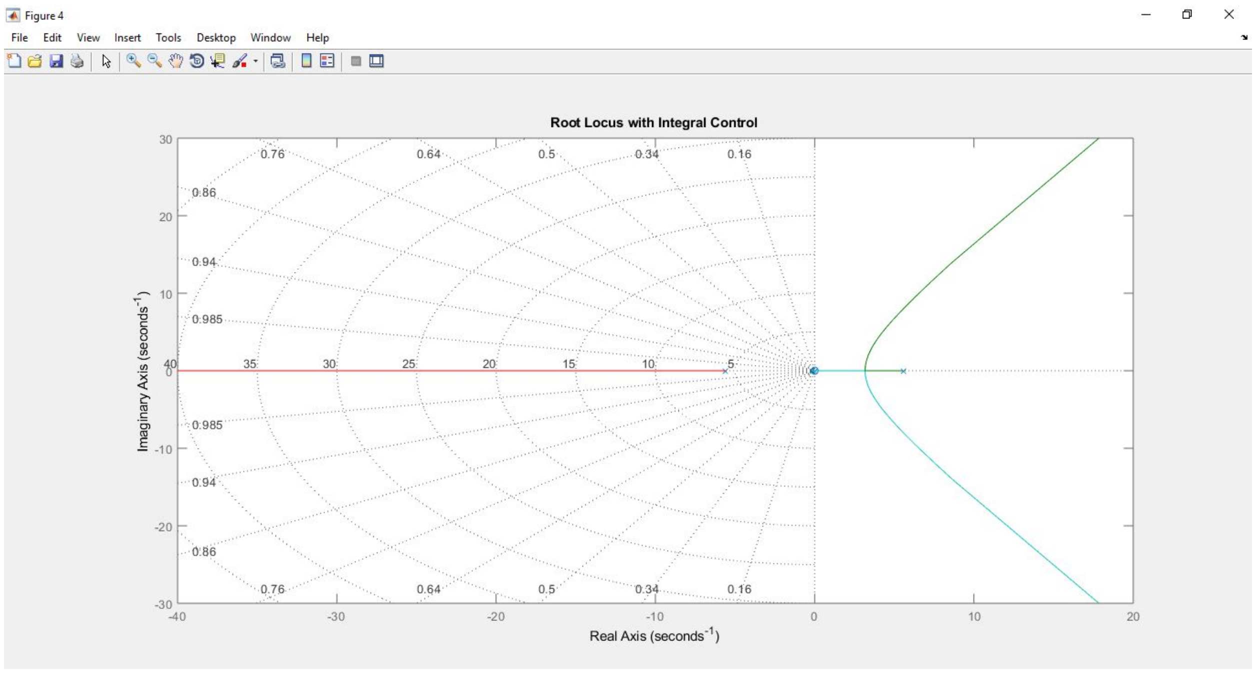
$$T(s) = \frac{\Phi(s)}{F(s)} = \frac{Ppend(s)}{1+Ppend(s)C(s)}$$

The controller will attempt to maintain the pendulum vertically upward when the cart is subjected to an impulse disturbance, under which, the design criteria that we have set is:

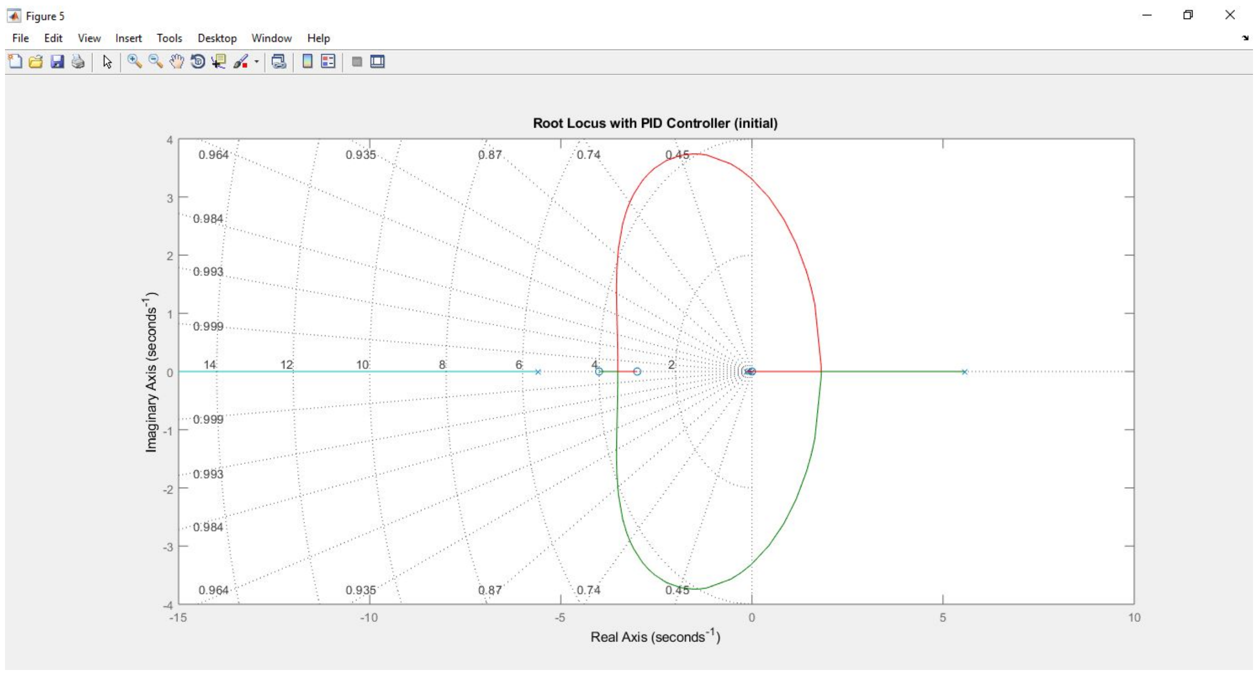
1. Settling time of less than 5 seconds.
2. Pendulum should not move more than 0.05 radians away from the vertical.

PID Controller Design for Pendulum Angle

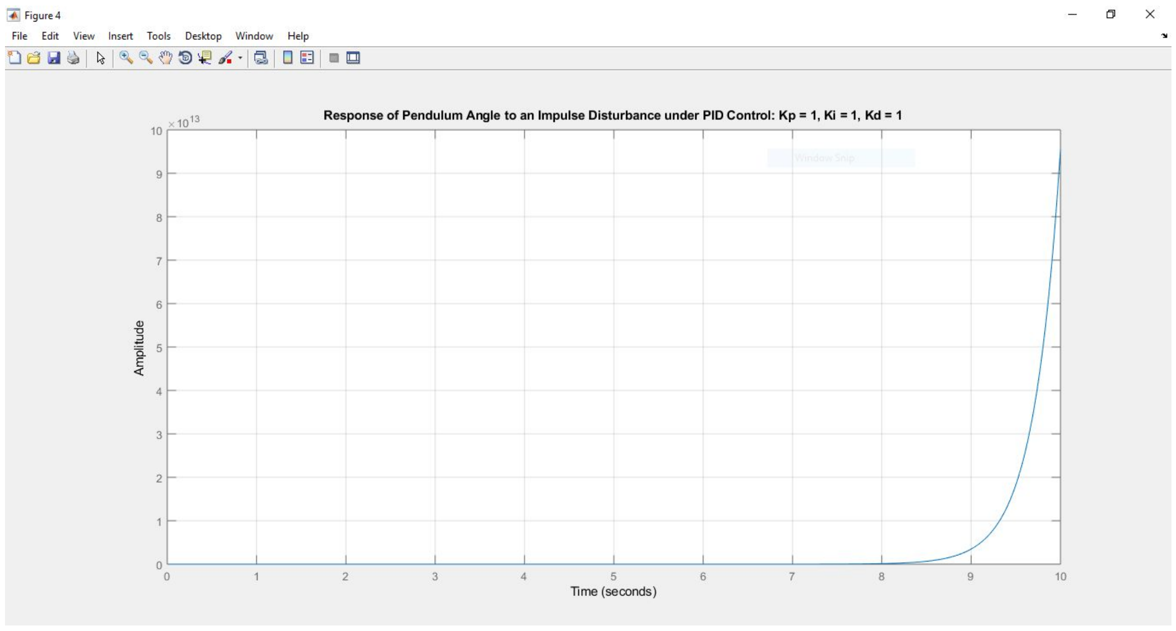
We begin by employing a PID controller for the pendulum angle with an initial set of values for K_p, K_i, K_D and then updating them by visually analysing the response of the system. Looking at the root locus for the pendulum angle shown above, it can be seen that one dominant branch of the root locus lies in the right half plane. This makes the system unstable. Hence, to make the system less stable, we being by adding an integrator to our system to cancel the zero at the origin and creating two dominant poles in the right half plane as shown below:



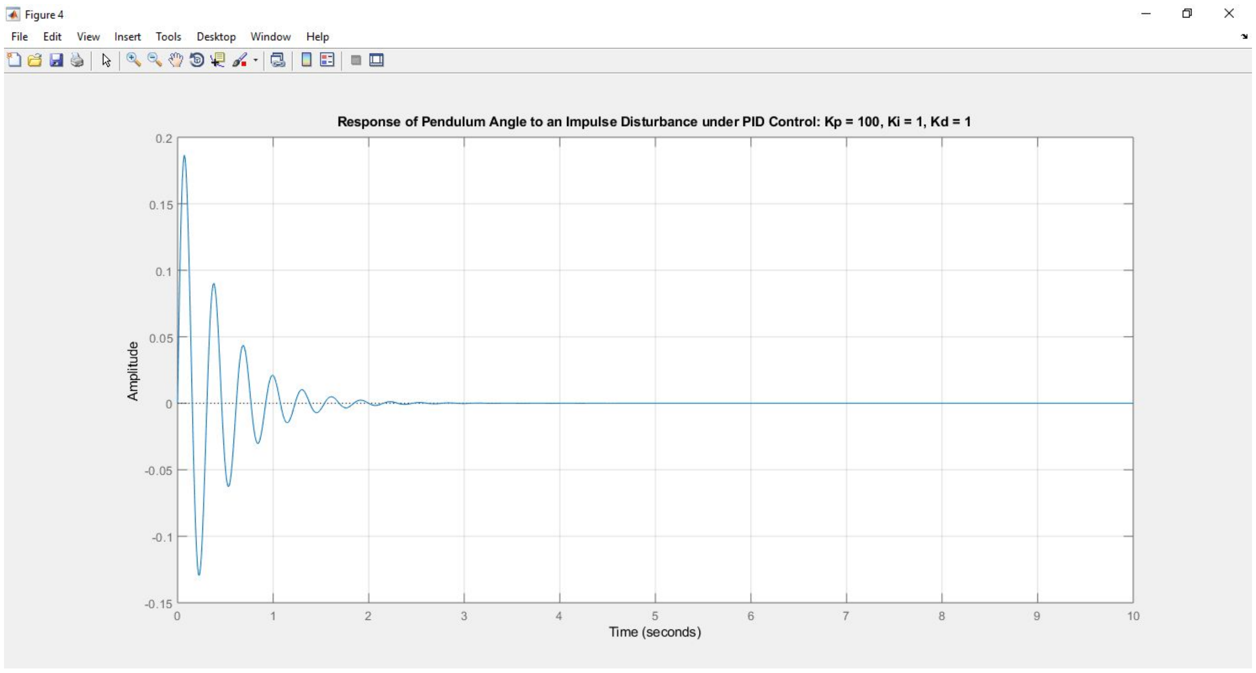
Now to pull the asymptotes lying in the right half plane into the left half plane, we can introduce two zeros into the left half plane to make sure that these dominant closed loop poles terminate at the finite zeros rather than moving to infinity.



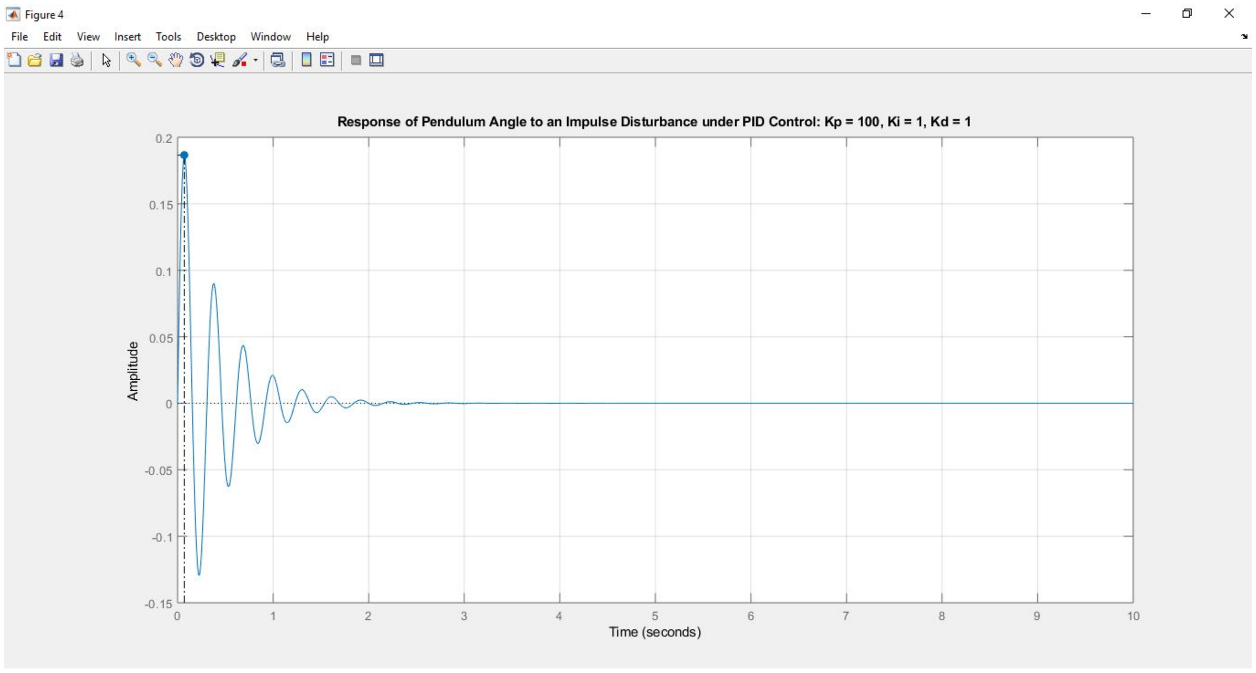
As seen, the gain K can now be set in order for the open loop dominant poles to enter the left half plane. After some trial and error, we achieved the following results for the closed loop impulse response of the pendulum angle:



We observe that the system is still unstable for these values of the K parameters of the PID controller. We begin by changing the values of the proportional gain K_p and noting what effect does it have on the response of our system, keeping in mind our design requirements as stated before.

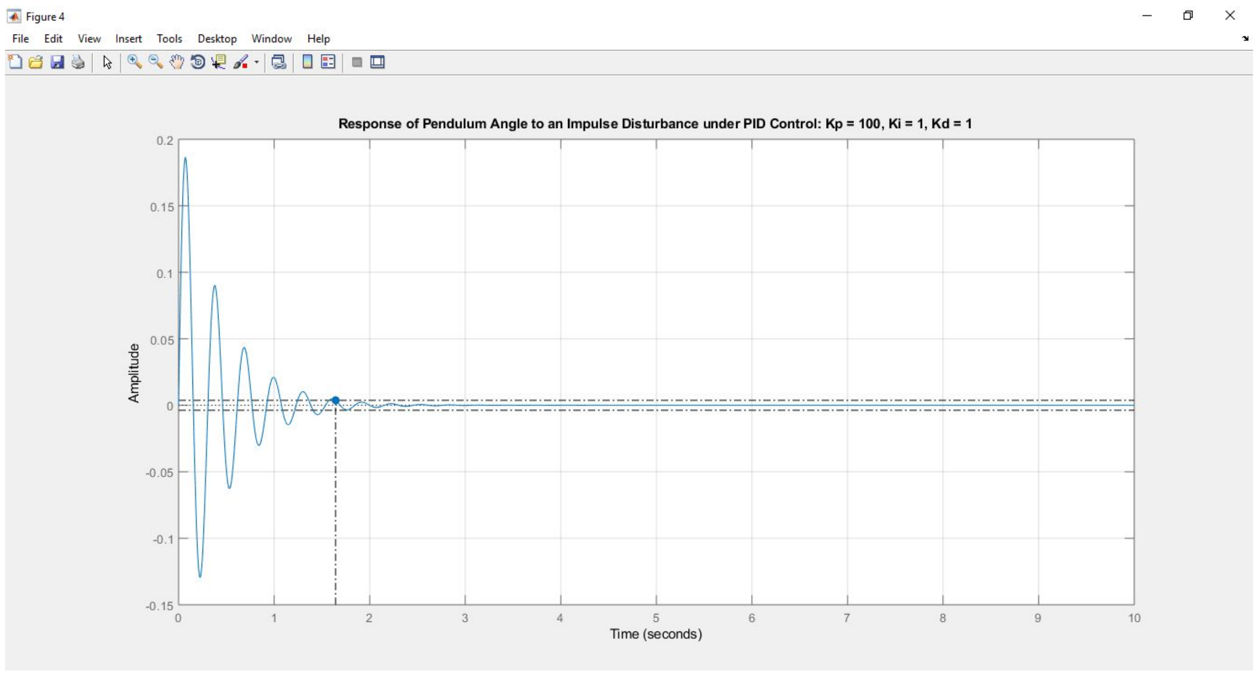


The peak amplitude and time:

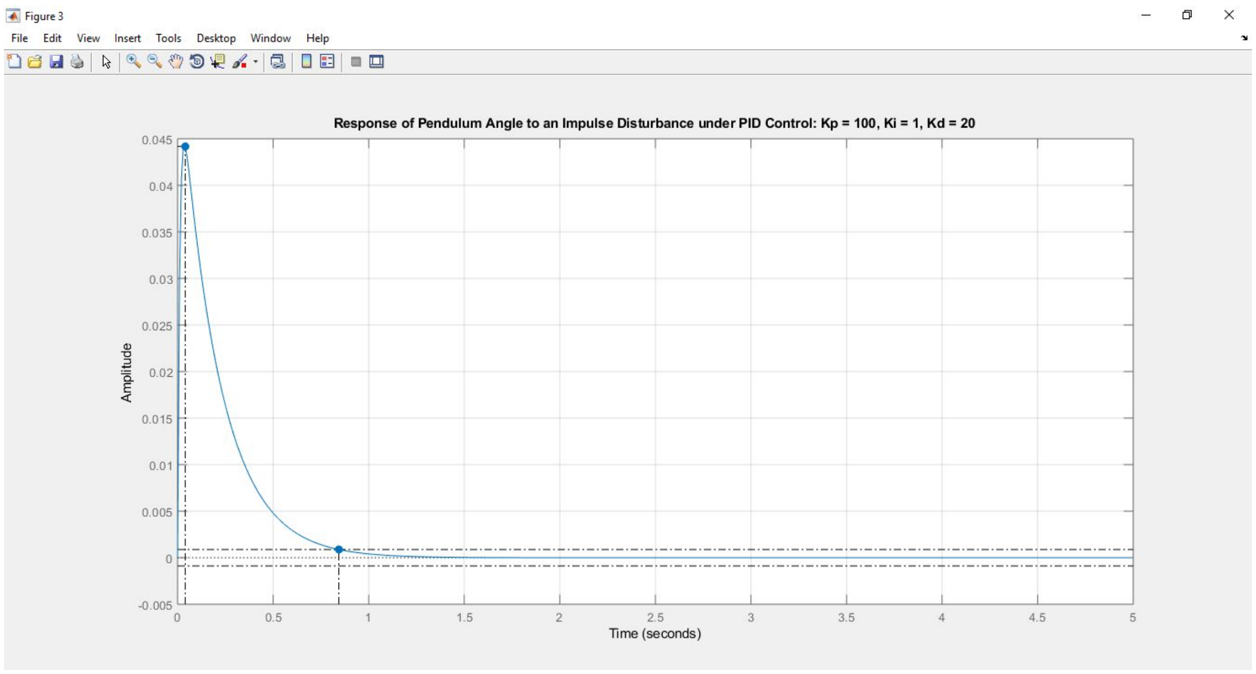


The peak response, however, is larger than the requirement of 0.05 radians. This additional overshoot often can be reduced by increasing the amount of derivative control, which will be seen below.

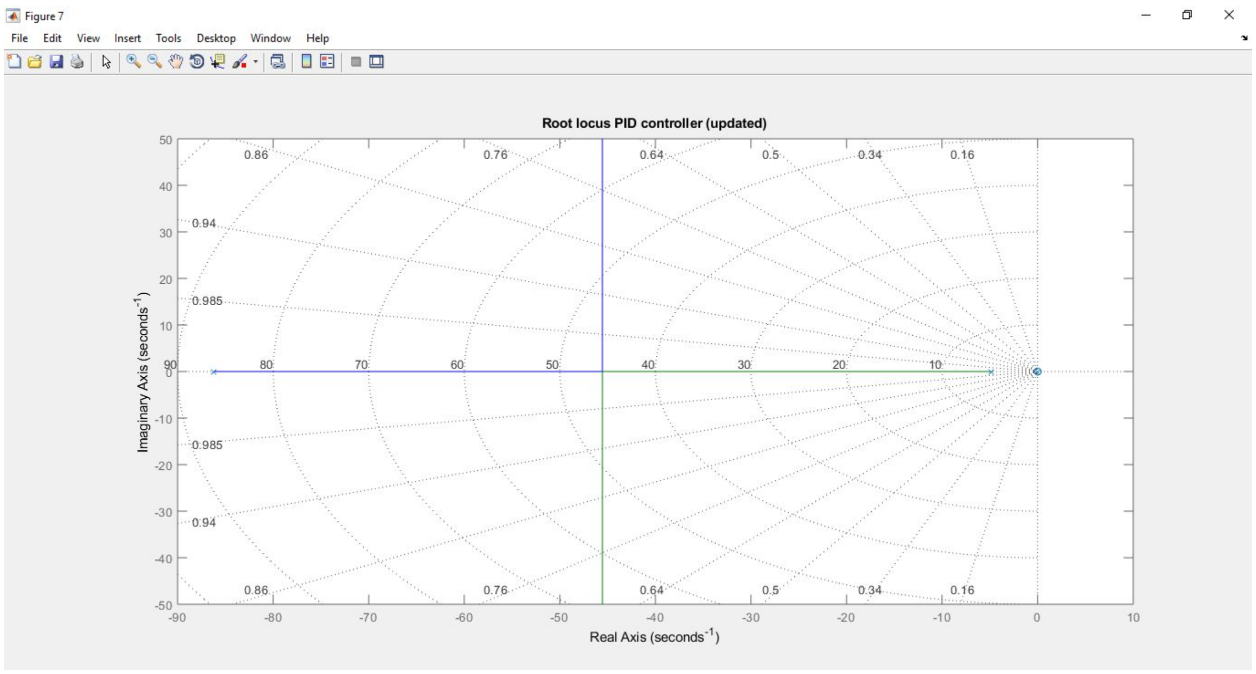
The settling time:



The settling time of the response is determined to be 1.64 seconds, which is less than the design requirement of 5 seconds. After some trial and error to reduce the overshoot by employing a derivative control, we arrive at an optimum controller design meeting all our requirements, with the impulse response shown below:



The root locus for the resulting system with the PID controlled now comes out to be:

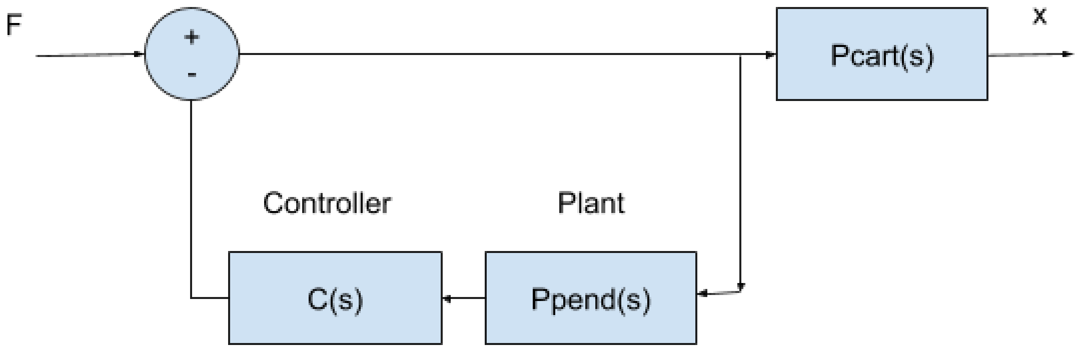


Since both of our design requirements have been met i.e. the pendulum does not deviate more than 0.05 radians from the vertical and the settling time of the pendulum angle is also less than 5 seconds, thus our PID controller for the pendulum angle has been designed with the following specifications:

1. **K_p = 100**
2. **K_i = 1**
3. **K_d = 20**

PID Controller Design for Cart Position

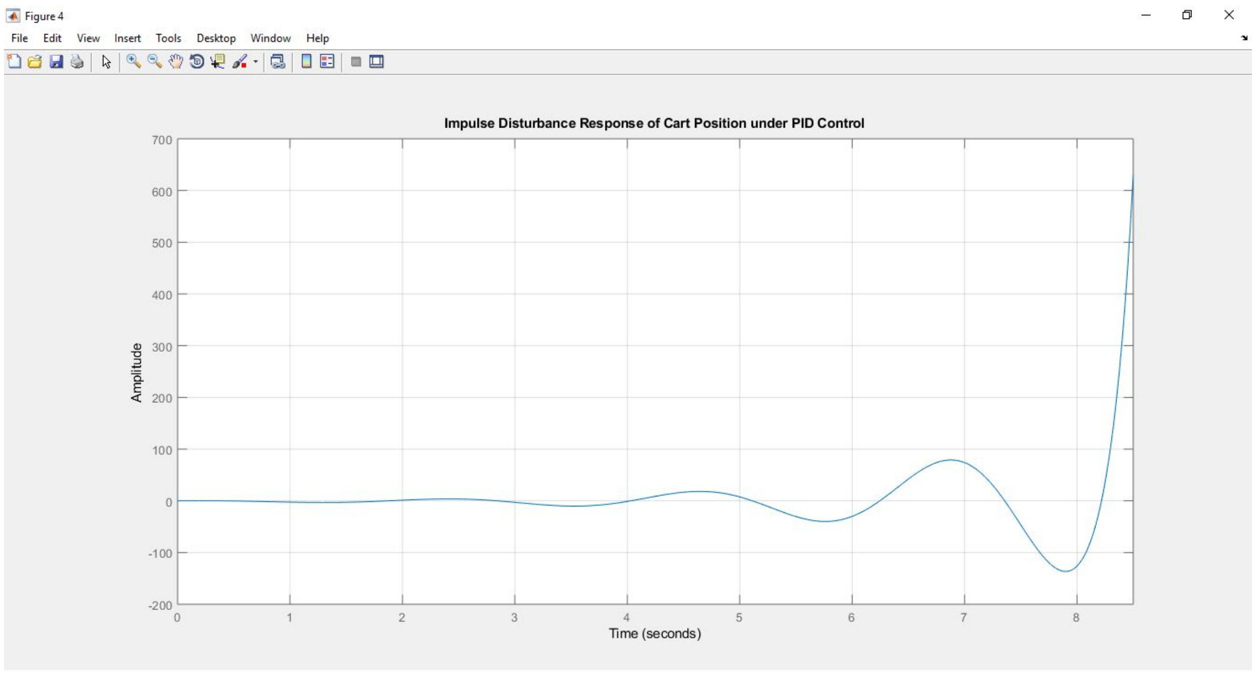
Modifying the aforementioned block diagram to show the input impulsive disturbance force ***F*** as the input to the system and incorporating the block for the cart position into the system as well to show the cart position ***x*** as the output of the system:



The resulting transfer function becomes:

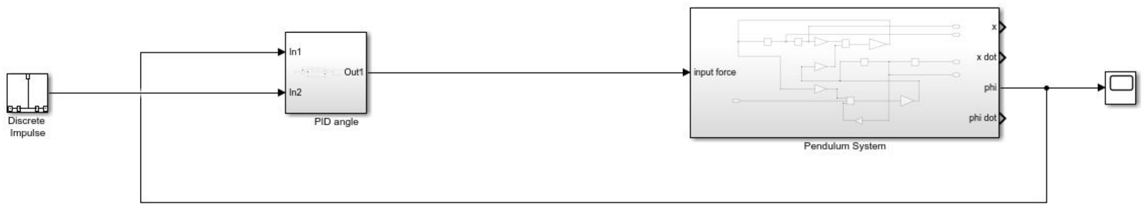
$$T(s) = \frac{X(s)}{F(s)} = \frac{Pcart(s)}{1+Ppend(s)C(s)}$$

As it is seen from the response below, even though the PID controller individually stabilizes the response of the pendulum angle, the cart position becomes unstable under an impulsive disturbance force, and cannot be controlled via a PID controller alone. Some other measures need to be taken to e.g. a state space controller.



System Modelling & Analysis in Simulink

Our self balancing robot realised in MATLAB, modelled as an inverted pendulum is shown below, with the plant (**Pendulum System**) and the PID controller (**PID angle**) shown as subsystems:

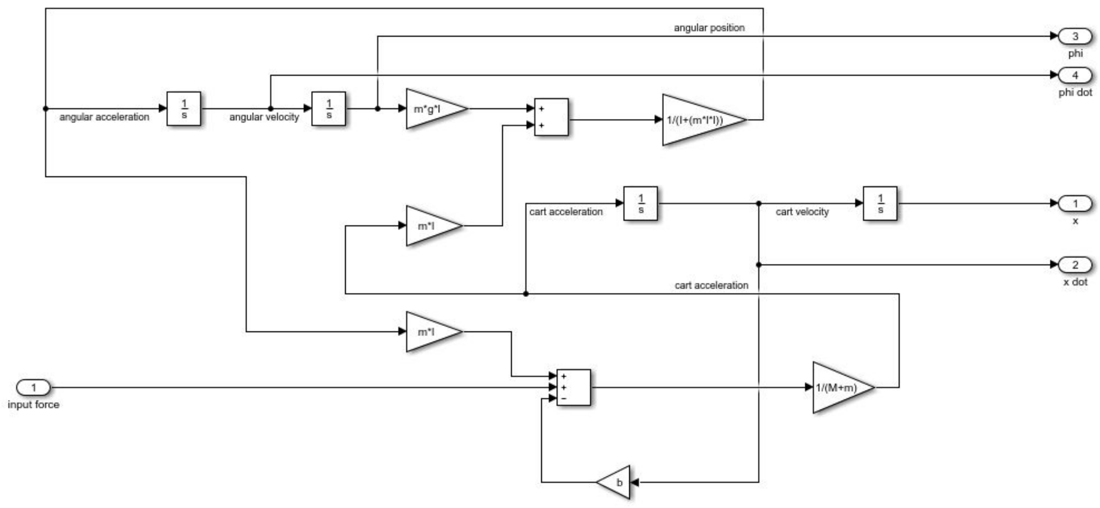


The Pendulum System block shown was created using our two primary linearized differential equations:

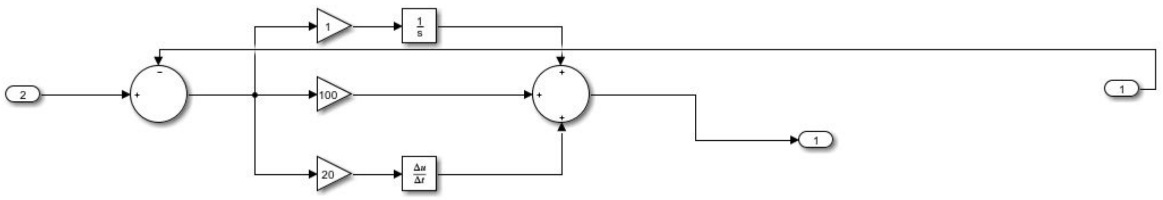
$$(I + ml^2)\ddot{\phi} - mgl\phi = ml\ddot{x}$$

$$(M + m)\ddot{x} + b\dot{x} - ml\ddot{\phi} = u$$

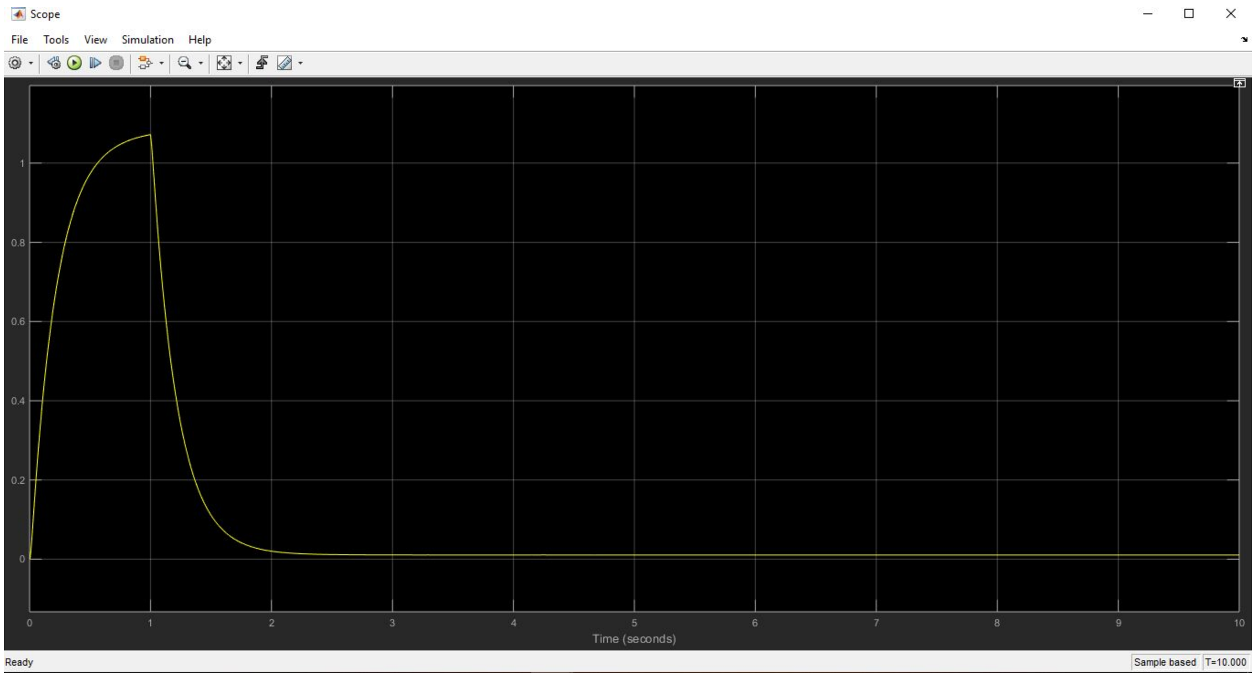
The Pendulum System block is expanded and shown below:



The PID controller block is expanded and shown below. The values for K_p, K_i, K_D have been adjusted to match with our MATLAB simulation before:



The impulse response of the pendulum angle under the PID control is shown below:



Both the MATLAB and Simulink simulation files are included with the project zip folder.

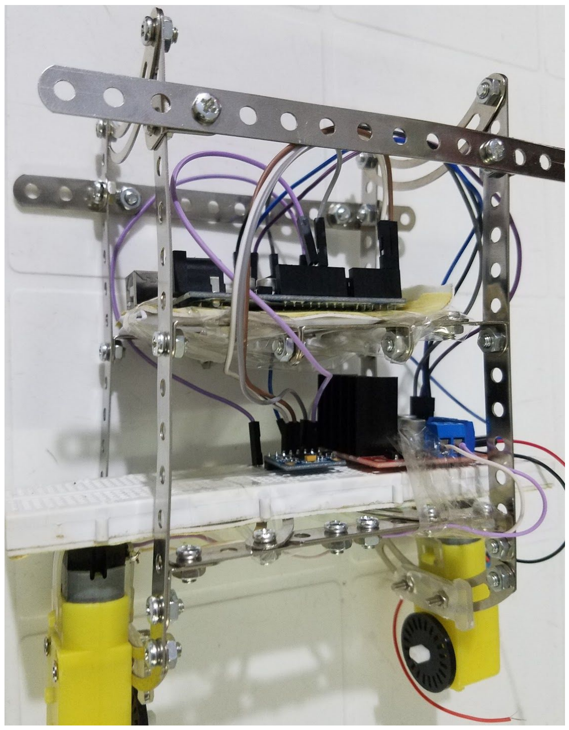
Assembly

The body of the robot should be such that it can be compared to the mathematical model of an inverted pendulum on a cart. In order to do so, the body of the robot must be symmetric. Center of gravity of the body should be between the tyres on the axis of the gyroscope.

To ensure symmetry and professional looking design, we decided to use mechano-sets for making the body. We could not find a set for a robot but we improvised and converted a ferris wheel into one.

The motors drain the battery a lot. Also motors need to be provided sufficient current and voltage to drive at the required speed to keep the robot balanced. Therefore, we decided to provide power to the robot externally. In the future version of this robot, we may try to overcome this shortcoming.

Ferris Wheel
HAND MINTED A ROMANTIC CLASSIC
IMPROVE THE CHILD'S ABILITY THINKING CREATIVITY.
TOOLS
WAITING FOR YOUR CREATION.
NO.592-17
Improve Intelligence
Fine Workmanship
Modeling Variety
Fun Filled



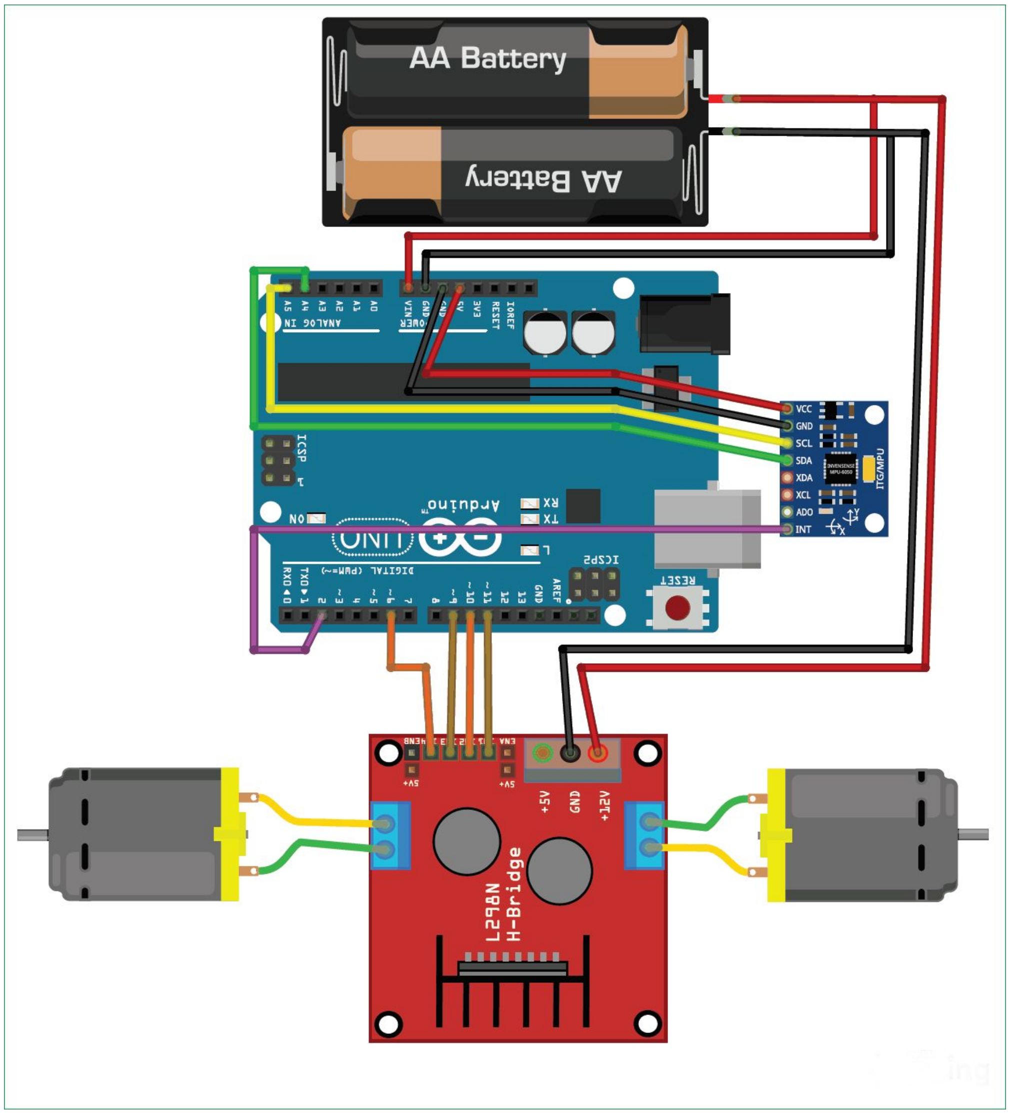
Components Used

The following equipment were required to assemble our self-balancing robot:

- Arduino UNO Microcontroller (x1)
- DC Motor (x2)
- L298N Motor Driver Module (x1)
- Accelerometer and Gyroscope MPU6050 Module (x1)
- Wheels (x2)
- Meccano Set (x1)
- Battery (x1)
- Breadboard (x1)
- Jumper wires

- Power Supply

Circuit Diagram



The above circuit diagram consists of an Arduino UNO microcontroller which is powered by a battery and is connected to two components, namely; Accelerometer and gyroscope module MPU650 and motor driver module L298N. The motor driver module is further connected to two DC motors having a wheel fixed on each motor. The accelerometer and gyroscope module is used as a feedback corresponding with the changes in the angular rotation or the angle by which the robot tilts. This feedback is fed to the Arduino which sends a corresponding signal to the motor driver to rotate the DC motor by the same angle in the opposite direction in order to maintain the balance of the robot. This circuit is mounted on a chassis of a robot constructed from the parts of meccano set.

Hardware Problems

We faced several problems. These are discussed below.

Metallic body: Our code used to get stuck at initialization on MPU. After hours of debugging code and wired connections, we found that the body of the robot was conducting contrary to our belief. The pins of the MPU were shorted and it did not work.

MPU Calibration: Each gyroscope has to be calibrated before being used. Without proper calibration, desired results can not be achieved. If the input is not correct, tuning PID controller will not have any effect on the stability of the robot.

Loose Connections: DC motors came with flimsy wires soldered to its terminals. During our testing, we realized that the working of motors was not reliable. We therefore soldered copper wires to DC motor terminals ourselves.

Arduino Code Explanation

```
#include "I2Cdev.h"
#include <PID_v1.h> //From
https://github.com/br3ttb/Arduino-PID-Library/blob/master/PID_v1.h
#include "MPU6050_6Axis_MotionApps20.h"
//https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050
```

This code is based on libraries for PID control and MPU6050 interfacing. These two do all the heavy lifting.

```
// orientation/motion vars
Quaternion q;           // [w, x, y, z]         quaternion container
VectorFloat gravity;    // [x, y, z]            gravity vector
float ypr[3];           // [yaw, pitch, roll]   yaw/pitch/roll container and gravity vector
```

These variables hold the data from the gyroscope sensor.

```
 // supply your own gyro offsets here, scaled for min sensitivity
   mpu.setXGyroOffset(0);
   mpu.setYGyroOffset(0);
   mpu.setZGyroOffset(0);
   mpu.setZAccelOffset(0);
```

MPU6050 needs to be calibrated before use. The values for offsets obtained are plugged here. These vary for each sensor.

```
//Initialise the Motor output pins
   pinMode (6, OUTPUT);
   pinMode (9, OUTPUT);
   pinMode (10, OUTPUT);
   pinMode (11, OUTPUT);
```

Pins 6 and 9 control motor A with the other two pins control motor B. One pin makes the motor rotate clockwise while the other pin makes it rotate counterclockwise.

```
        mpu.dmpGetQuaternion(&q, fifoBuffer); //get value for q
        mpu.dmpGetGravity(&gravity, &q); //get value for gravity
        mpu.dmpGetYawPitchRoll(ypr, &q, &gravity); //get value for ypr
        input = ypr[1] * 180/M_PI + 180;
```

Data is received from MPU when it calls interrupt. We only use the pitch (ypr[0,1,2] = yaw,pitch,roll). Value is converted from radians to degree and added to 180 because we are taking upward as 180 and downward as 0 degree.

```
PID pid(&input, &output, &setpoint, Kp, Ki, Kd, DIRECT);
pid.Compute();
```

This library works by calculating the calculating the error signal, its derivative and integrals to find the output to drive error to zeros.

$$u(t) = \mathrm{MV}(t) = K_\mathrm{p} e(t) + K_\mathrm{i} \int_0^t e(\tau)\, d\tau + K_\mathrm{d} \frac{de(t)}{dt},$$

where

K_p is the proportional gain, a tuning parameter,
K_i is the integral gain, a tuning parameter,
K_d is the derivative gain, a tuning parameter,
$e(t) = \mathrm{SP} - \mathrm{PV}(t)$ is the error (SP is the setpoint, and PV(t) is the process variable),
t is the time or instantaneous time (the present),

Compute is a function of PID library and uses the values of Kp, Ki, Kd, input and setpoint to find the correct voltage level which will drive the motor sufficiently in the falling direction. The value of output is then fed to the motor driver.

```
if (output>0) //Falling towards front
        Forward(); //Rotate the wheels forward
else if (output<0) //Falling towards back
        Reverse(); //Rotate the wheels backward
        }
else //If Bot not falling
        Stop(); //Hold the wheels still
```

This is where the robot is balanced. Forward and reverse function just make the tyres rotate. When the robot is straight, gyroscope gives zero pitch and motors stop.