الجمهورية الجزائرية الديمقراطية الشعبية
People's Democratic Republic of Algeria
وزارة التعليم العالي والبحث العلمي
Ministry of Higher Education and Scientific Research
جامعة العربي التبسي – تبسة
Larbi Tébessi University -Tébessa
Faculty of Sciences and Technology
Departement of electrical engineering

Thesis

Presented to obtain the academic master's degree in

Automatic

Option: Automatic and Systems

Candidates:
Mr. AMRANE Naceur
Mr. FETNI Ibrahim

Supervisor : Dr.OUNNAS Djamel

Design and Implementation of Adaptive Neural PID for a Nonlinear System

Publicly defended on: 11/06/2022 in front of the jury composed of

Mr. THELAIDJIA Tawfik	MCA	President
Mr. OUNNAS Djamel	MCA	Supervisor
Mr. LEMITA Abdallah	MCB	Examiner

Promotion: 2021/2022

Aknowlgment

We would like to thank all the people who contributed to this work.

We extend a special thanks to ***Dr. Djamel Ounnes****,*

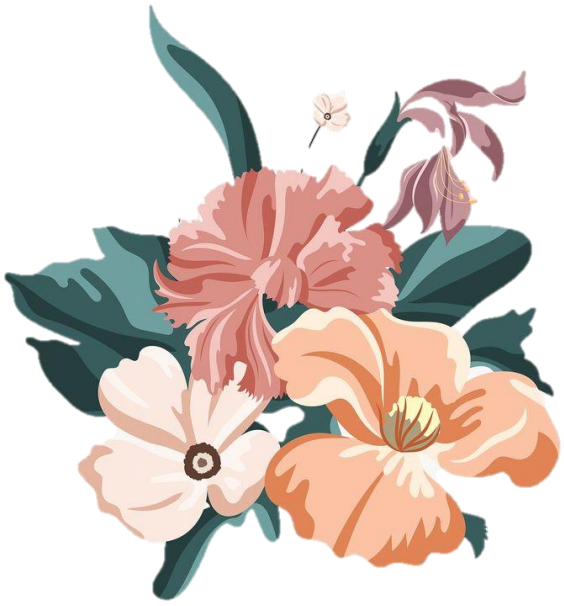
We would like to thank him for his great contribution to the completion of this work and wish him a successful academic life

We would like to thank all of our teachers who passed by us in our course of study specially:

"Djari Abdlhamid*,* ***Thelaidjia Toufik, Lemita Abdallah****,* ***Aziz Bougadoum, Yousfi Laatra, Djedi Abdlghani, Guiza dhaouadi, Bougharen Abla, Metatla Samir, Amieur Toufik, Loudjani Abdlhak****"*

To the doctoral student ***Hama Soltani*** *for his contribution to this work*

To all employees of the University of Arabi Tebessi.



Dedication

I am very pleased to dedicate with simple work:

To my parents my father ***Amrane Sebti***

& my mother ***Jouini Houriya***

, To my brothers ***Khalil*** *and* ***Ayoub****,*

To my sister ***Ahlem*** *and her son* ***Mohammed Assil*** *and my little sister* ***Aridj****,*

& to all the family ***Amrane*** *specially my grandfather* ***Taher***

My god have mercy on him, & my grandmother ***Hada***

My god have mercy on him my god have mercy on her, ***Naceur***

my god have mercy on him, ***Hamza****,* ***Yacine****,* ***Ahmed****,* ***Sifi****,* ***Youcef****,* ***Rachid****,*

Nourddine*,* ***Chrait****, and* ***Hmouda***

my god have mercy on him ***,Abdlbaki****,* ***Kadour****,* ***Lamin****,* ***Cherif****,* ***Abdlkarim****,* ***Samir****,*

Saddek*,* ***Belgacem****,* ***Djalel****,* ***Djamel****,* ***Tarek****,*

To my uncles ***Ali****,* ***Nourddine****,* ***Madjid****,*

To all my friends specially ***Seddik Amrane****,* ***Sadi Kais****,* ***Djabali Salem****,* ***Dhia Gherdi****,*

Kafi Redhwan*,* ***Houcine Salah****,* ***Aimen Mohammed*** *,* ***Ghoul Mounir****,* ***Salah A****,*

Azzedine L*,* ***Aymen L****,* ***Omar A****,* ***Belgacem L****,* ***Houssem S****,* ***Hichem L****,* ***Saber S****,* ***Ali***

L*,* ***Ramzi A****,* ***Soufian L****,* ***Saif C****,* ***Bilel L****,*

To all my neighbors, to all my teachers, to everyone who knows me, to my colleague

in the memory ***Fetni Ibrhim****, to all the Automatic and system class.*

Naceur Amrane

Dedication

To my dead father Fetni Saad My god have mercy on him, to my mother Khazene Mehaniya, to my brother, Kamel, Ismail, Boubaker ,To all the family Fetni, to all my friend specially

“Naceur, khoubaib, Med taher, tarek, Raouf, oussama, ilyes, Houssam”

Mohamed , mounir, kqis, madjid.

Fetni ibrahim

List of figures

Chapter 1

Chapter 2

Chapter 3

Chapter 4

List of tables

Abstract

It is usually difficult to solve the control problem of any complex nonlinear system . In our work we present an effective control method based on an adaptive neural PID controller, and apply it for a widely and to check typical academic nonlinear system such as inverted pendulum. There are a huge amount of controllers witch have been applied for this system and we compare one of them such as a classical PID controller and see the additional advantage of the adaptive neural PID, then design and implement it with a typical example for the inverted pendulum such as Two Wheels Self-balancing Robot.

Keywords

Nonlinear system. Adaptive Neural PID. Inverted Pendulum. Neural Network. DC Motor. PID controller. Ziegler Nichols. Arduino. Two Wheels Self-Balancing Robot.

الملخص

عادة يكون من الصعب حل مشكلة التحكم في أي نظام غير خطي معقد. في عملنا، نقدم طريقة تحكم فعالة تعتمد على وحدة تحكم PID العصبية التكيفية، ونطبقها على نطاق واسع ومتنوع لنظام أكاديمي غير خطي نموذجي مثل النواس المعكوس، وهناك عدد كبير من المتحكمات المطبقة بالفعل على هذا النظام ونقارن احد هذه المتحكمات و قد اخترنا متحكم PID مع ما تحصلنا عليه مع نتائج المتحكم PID العصبي التكيفي، ثم تصميمها وتنفيذها بمثال نموذجي للنواس المقلوب مثل الروبوت ذاتي التوازن بعجلتين باستخدام أردوينو

Résumé

Il est généralement difficile de résoudre le problème de contrôle d'un système non linéaire complexe. Dans notre travail, nous présentons une méthode de contrôle efficace basée sur un contrôleur PID neuronal adaptatif, et l'appliquons à un système non linéaire académique typique largement et varié tel qu'un pendule inversé, il existe une énorme quantité de contrôleurs déjà appliqués pour ce système et nous en comparons un d'entre eux comme un contrôleur PID classique et voyez l'avantage supplémentaire du PID neuronal adaptatif, puis concevez-le et mettez-le en œuvre avec un exemple typique pour le pendule inversé tel que le robot auto-équilibré à deux roues.

CONTENTS

CONTENTS

CONTENTS

Chapter IV: Real Implementation for the TWSBR

CONTENTS

General introduction

General introduction

In the industrial field, the controlled system has usually great nonlinearity, including spacecraft system, vehicle system, robot system, power system, chemical reaction system, etc. It is hard to get a precise control performance even by the intelligent control methods, including adaptive control, fuzzy control, and neural network control and decoupling control. So many mixed control methods have been presented, such as PID neural network. Due to the characteristics of self-learning, self-organizing and self-adaptation, PID neural network would automatically identify the parameters of controlled system and adjust them according to system changes.

Firstly, we will begin with the control of our nonlinear system by using classical PID controlling, witch is widely used in control engineering but it has some disadvantage while it is applied for a complex nonlinear system because the parameters of the controller $K_p \; and \; K_i \; and \; K_d$ are usually fixed and with the poor capability of dealing with uncertainty, parameter variations and external disturbance, the PID controller is not the best controller to apply for a complex nonlinear system.

In recent years, there has been extensive interest in self-tuning these three controller gains. For examples, the PID self-tuning methods based on the relay feedback technique were presented for a class of systems. An adaptive PID control tuning was proposed to cope with the control problem for a class of uncertain chaotic systems with external disturbance. A genetic algorithm was used to find the optimum tuning parameters of the PID controller by taking integral absolute error as fitting function. Sliding mode control (SMC) is one of the popular strategies to deal with uncertain control systems. Neural networks (NN) have been used for modeling and control of complex physical systems because of their ability to handle complex input-output mapping without detailed analytical models of the systems [22].

The neural network controllers have emerged as a tool for difficult control problems of unknown nonlinear systems. Neural networks (NN) have been used for modeling and control of complex physical systems because of their ability to handle complex input-output mapping without detailed analytical models of the systems [22].

In our work, we choose to deal with an inverted pendulum as a complex nonlinear system, because it is one of the most popular systems used for illustrating various control techniques. The goal of controlling of the pendulum is to balance the pendulum in the upright position when it initially starts with some nonzero angle off the vertical position. This system is a typical and academic

nonlinear control problem, so many techniques have been already applied for this system, like, model-based control, fuzzy control, neural network (NN) control, genetic algorithms (GAs)-based control, and so on, so we will make our Adaptive neural PID controller to this system and we will apply it with a "Tow Wheels Self-Balancing Robot" and see the results.

In the first chapter, we will talk about the modeling of the inverted pendulum and the modeling of the actuator, we will begin by using stepper motor and we will mention the problems that we will face using actuator, then we will use the DC motor and see the suitable actuator for our system then we will combine them to get our TWSBR modeling.

In the second chapter, we will control the system "TWSBR" with a classical PID and make a simulation in MATLAB Simulink using Ziegler Nichols method for tuning our controller parameter such as K_p , $and\ K_i$, $and\ K_d$, and mention the problem that we will face with choosing this parameter and disadvantage of the PID controller.

In the third chapter we will control our system "TWSBR" with a Neural Adaptive PID controller, we will firstly look at a brief background for the neural network and then we will use single neuron adaptive PID controller, we will compare the results that we get with the previous results that we get from the classical PID controller and discuss why it is better to use Neural Adaptive PID controller instead of the classical PID controller, and see the difference between the two controller.

In the last chapter, we will take about the design and the implementation of our TWSBR using Arduino Nano, and talk about all the components that we will use to build our robot, and then we will mention the results that we will get and how the system behaves

CHAPTER 01:

System modeling

1. Introduction

A self-balancing robot is a robot, balances itself on two wheels, by constantly correcting its position .In the '80s, a Japanese Professor, Kazuo Yama Fuji, built the first model to simulate the behavior of an inverted pendulum. Since then, many different prototypes have been built by many researchers. The widespread accessibility of economical electronic components has made it a fascinating project for makers and students [1]. The TWSBR system is a dynamically unstable system and has a behavior equivalent to an inverted pendulum on a cart with wheels, therefore, its modeled using as a reference to the inverted pendulum system. The TWSBR system can be considered as a mechanical platform composed of two coupled subsystems: the main body (pendulum) and the assembled rotation system (pendulum cart).

2. Analytical system model

The physical problem of the balancing robot is well described by the widely analyzed inverted pendulum. It is commonly modelled as a rigid rod fastened by a frictionless joint to a rigid cart moving in one direction. The simplification that the wheel base can be seen as a cart sliding on a frictionless surface was made. This model definition is inspired by Math Works tutorial about inverted pendulum. See Figure 1 for the simplification steps in this project [2].

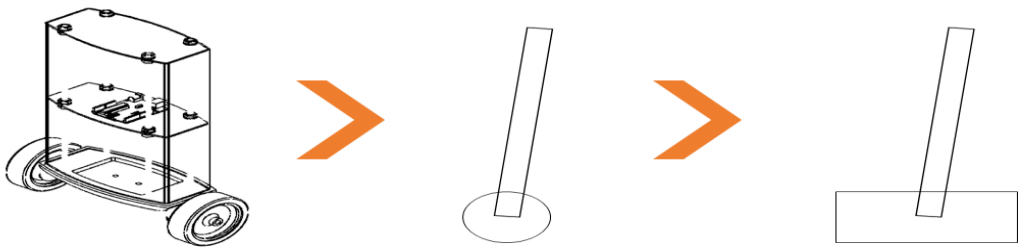


Figure I. 1. Simplification steps of the inverted pendulum

This is a nonlinear multi-variable and highly unstable system. Inverted pendulum-cart system are shown below, in Fig. 2. As a simplifying hypothesis, the cart friction with the surface and the pendulum friction with air will be neglected. In this model there is an assumption that the gravity

force acts on a point of mass in all instances and are externally influenced by a force that acts on x axis, moving the pendulum, system acting like a cart [4].

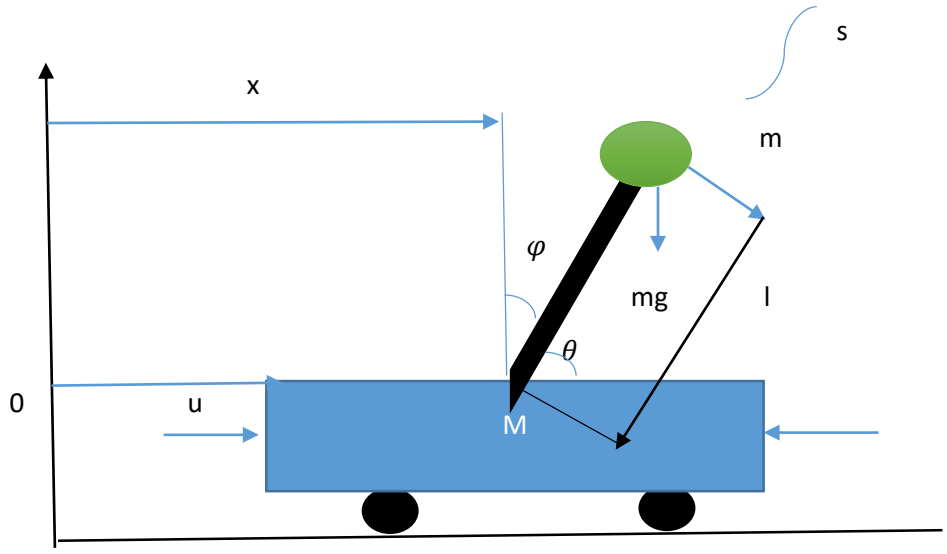


Figure I. 2. Equivalent system for the robot.

2.1. Transfer function

We will begin by the dynamic equations that we can derived from the forces in Fig.1.

$$u = (M + m)\ddot{x} + f\dot{x} + ml\ddot{\theta}cos\,\theta - ml\dot{\theta}^2 \sin\theta \qquad (I.1)$$

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

Where g is the gravitational constant, m is the pendulum mass, M is the cart mass, θ is the tilt angle, f is a friction parameter, l – bar length and I is the pendulum moment of inertia [3].

The non-linear equations in the linear form [4]:

$$(I + ml^2)\ddot{\varphi} - mgl\varphi = ml\ddot{x} \qquad (I.3)$$

$$(M + m)\ddot{x} + b\dot{x} - ml\ddot{\varphi} = u \qquad (I.4)$$

Rewriting the equations as transfer functions renders: [3]

$$\varphi(s) = \frac{\frac{ml}{q}s}{s^3 + \frac{f(I+ml^2)}{q}s^2 - \frac{(M+m)mgl}{q}s - \frac{fmgl}{q}}\,u(s) \qquad (I.5)$$

$$X(s) = \frac{\frac{f(I+ml^2)s^2 - gml}{q}}{s^4 + \frac{f(I+ml^2)}{q}s^3 - \frac{(M+m)mgl}{q}s^2 - \frac{fmgl}{q}s}\, u(s) \qquad (I.6)$$

where: $q = (M+m)(I+ml^2) - (ml)^2$

Note

In this work, we will just work with the angle, when it comes to the position, we will just make a simulation without implement it.

2.2. State space model

This arrangement is viable, since an inverted pendulum is intuitively unstable. The differential equations are linearized . According to [2], the system can then also be described in State Space form with the states being$\dot{x}, \ddot{x}, \dot{\varphi}, \ddot{\varphi}$. So from the linear equations (3) and (4) we can obtained the state space model [2].

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\varphi} \\ \ddot{\varphi} \end{bmatrix} = A \begin{bmatrix} x \\ \dot{x} \\ \varphi \\ \dot{\varphi} \end{bmatrix} + B\,u \qquad (I.7)$$

$$y = Cu \qquad (I.8)$$

The matrices A, B, C, D [3]

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+ml^2)f}{I(M+m)+Mml^2} & \frac{m^2gl^2}{I(M+m)+Mml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-mlf}{I(M+m)+Mml^2} & \frac{mgl(M+m)}{I(M+m)+Mml^2} & 0 \end{bmatrix} \qquad (I.9)$$

$$B = \begin{bmatrix} 0 \\ \frac{I+ml^2}{I(M+m)+Mml^2} \\ 0 \\ \frac{ml}{I(M+m)+Mml^2} \end{bmatrix} \qquad (I.10)$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad (I.11)$$

$$D = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad (I.12)$$

2.3. System parameters

We will take the parameters by measurement as following:

Parameters	Values
M	0.3 [kg]
m	0.2 [kg]
I	0.0015 [kgm^2]
g	9.81 [m/s^2]
f	0.1
l	0.15 [m]

Table.I.1: the TWBR parameters

With these parameters inserted to the MATLAB model, the State Space system
Looks as:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\varphi} \\ \ddot{\varphi} \end{bmatrix} = \begin{bmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -0.2857 & 4.2043 & 0 \\ 0 & 0 & 0 & 1.0000 \\ 0 & -1.4286 & 70.0714 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \varphi \\ \dot{\varphi} \end{bmatrix} + \begin{bmatrix} 0 \\ 2.1444 \\ 0 \\ 14.2857 \end{bmatrix} u \qquad (I.13)$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \varphi \\ \dot{\varphi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \qquad (I.14)$$

2.4. Open loop response

Using all the equations and the state space model, a MATLAB simulation yielded a result, which we were aware about that the system is unstable without any control

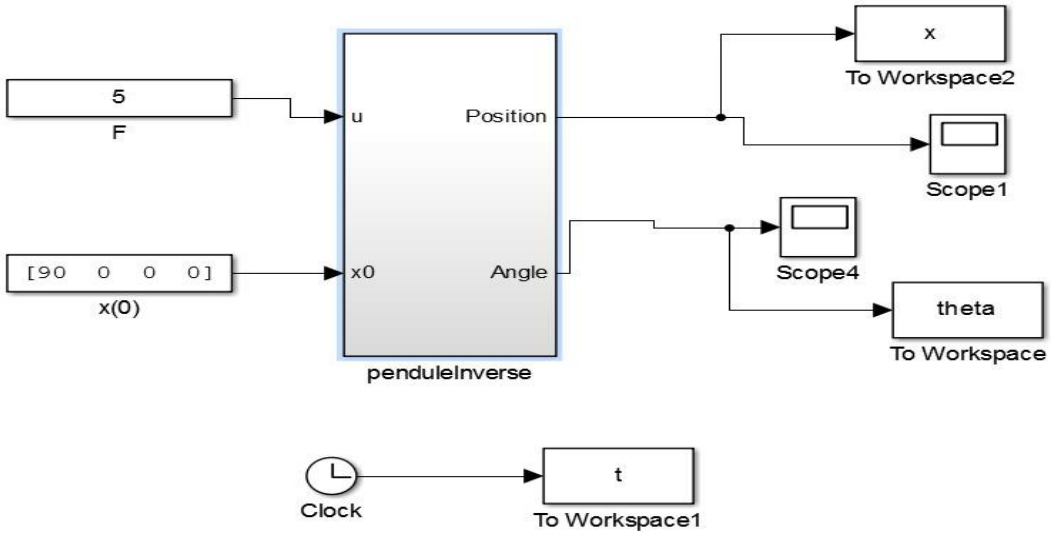


Figure I. 3. Simulink model of Open loop response

The result

1- Angle

- With initials values $x0 = [0,0,0,0]$

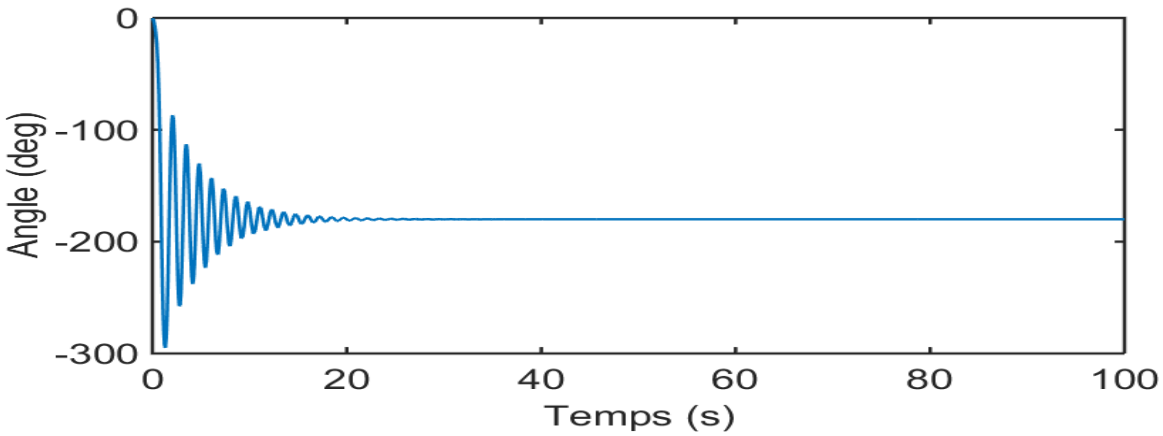


Figure I. 4. Angle open loop response with $x_0 = [0, 0, 0, 0]$

In the open loop response, with initial values $x_0 = [0, 0, 0, 0]$ we notice that the system goes to the stable point we consider it 180°, the angle reached this point after 30 seconds after the oscillation between 0 and 300°

- With initials values $x_0 = [0, 0, 180, 0\ ]$:

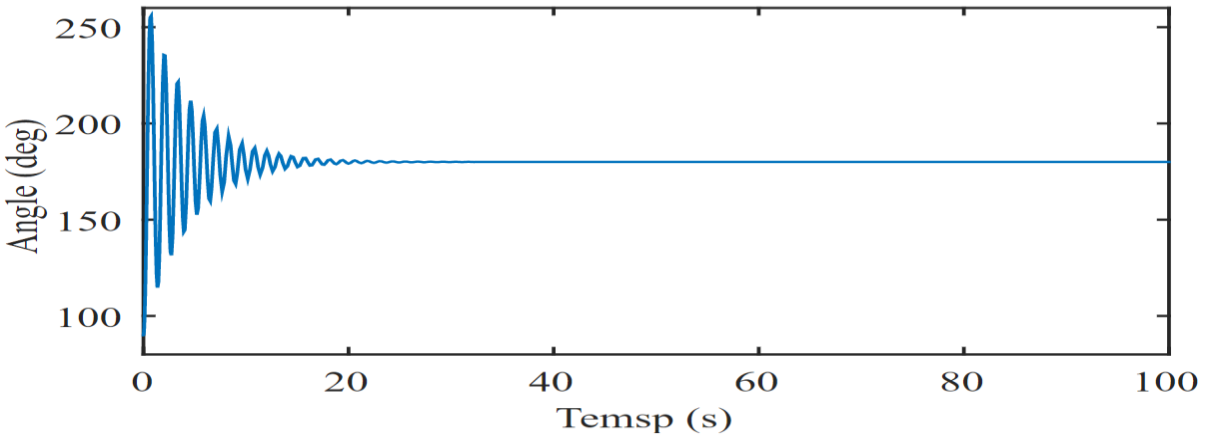


Figure I. 5. Angle with open loop control with 180 as initial values

Interpretation of result: This time we test the open loop response with an initial angle value equal to 180°, and we noticed that the system change its direction and continue the oscillation between 250° and 0°, and reached the stable point 180° and stabilize after 30 seconds

2- Position

- With initials values $x_0 = [0, 0, 0, 0]$

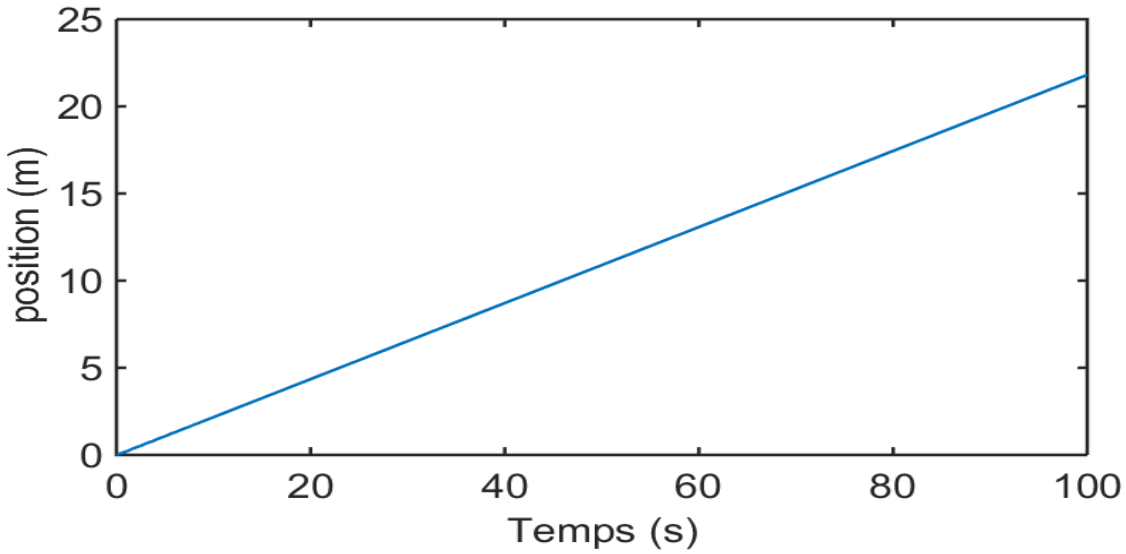


Figure I. 6. Position open loop response with $x0 = [0, 0, 0, 0]$

- With initials values

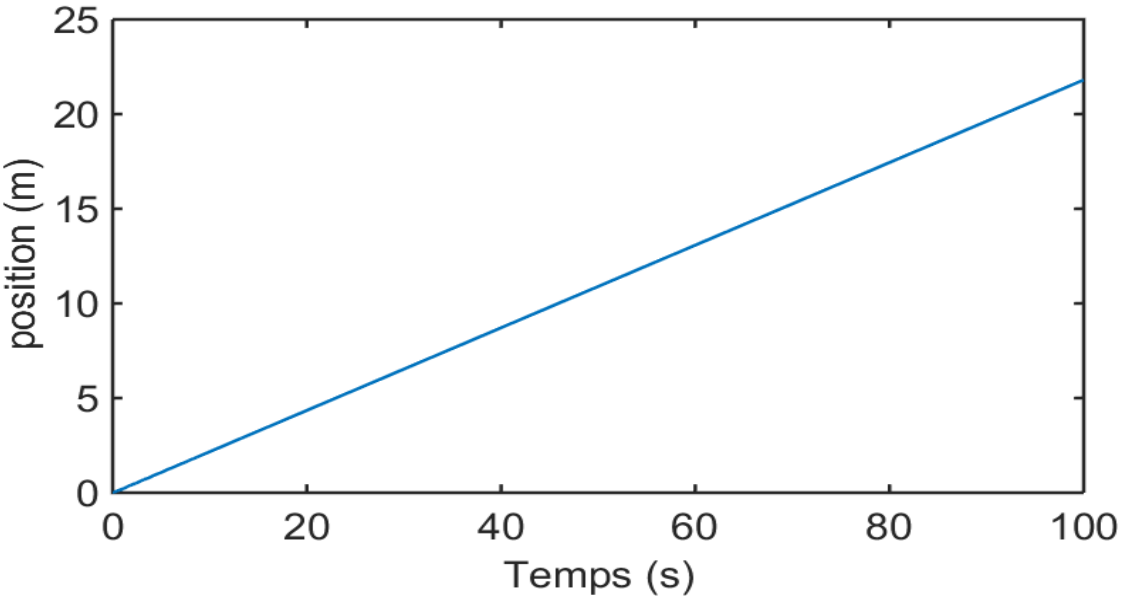


Figure I. 7. Position open loop response with $x0 = [0, -45, 0, 0]$

Modelling based on the inverted pendulum shows that the system is unstable without a controller, therefore, we need to make a controller for this system and we will begin with PID controller and then Neural PID controller and see how the system will act.

3. Actuators

3.1. Linear Stepper Motor Model

3.1.1. Mathematical Modelling

In this section, the mathematical model of a hybrid stepper motor is derive. Fig. I.8 Shows the schematic drawing of a simple two-phase hybrid stepper motor with three teeth [5]

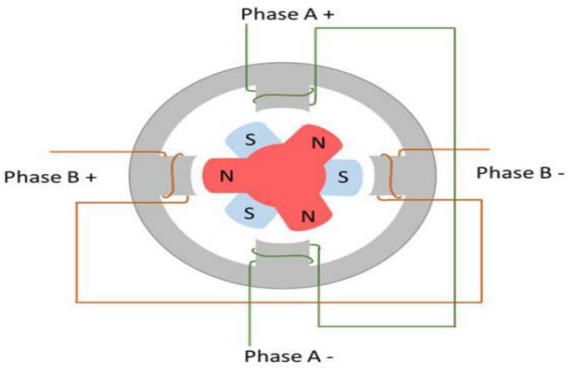


Figure I. 8. Schematic view of a stepper motor with three teeth

The mathematical model of a stepper model could be divided into two sub-models, electrical model and mechanical model. Starting with the electrical model, each phase of the stepper motor phases could be modelled as an RL circuit plus a back electromotive force (emf)

In Figure I.8. According to the differential equations of phase a and phase b are given by (I.15) and (I.16), respectively [5].

$$L_a \frac{di_a(t)}{dt} = -R_a i_a(t) - e_a(t) + u_a(t) \qquad (I.15)$$

$$L_b \frac{di_b(t)}{dt} = -R_b i_b(t) - e_b(t) + u_b(t) \qquad (I.16)$$

Where $e_a(t) = k_m \omega_m \sin(p\theta_m)$ and $e_b(t) = k_m \omega_m \cos(p\theta_m)$ [5]

In equation (I.15) and (I.16), the phase resistances and inductances are assumed equal, $R_a = R_b = R$ [Ω] and $L_a = L_b = L$ [H]. Moreover, and are the terminal voltages [V], $e_a(t)$ and $e_b(t)$ are the back emf [V], k_m is the motor constant, p is the number or motor poles pairs (teeth), ω_m is the rotor (mechanical) angular speed [rad/s], θ_m and is the rotor (mechanical) angular position [rad] [5].

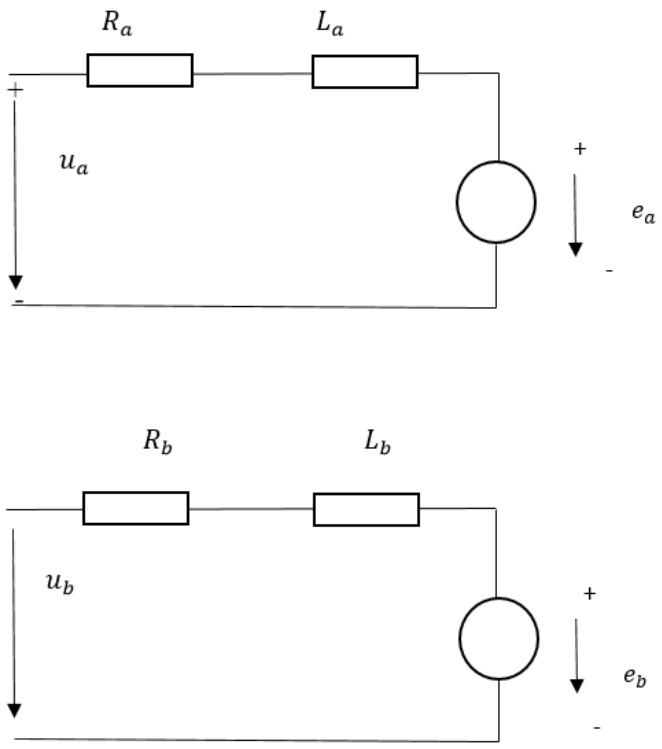


Figure I. 9. Equivalent circuits of hybrid stepper motor.

The shaft of the hybrid stepper motor, which represents the mechanical part of the system is modelling as a rigid body subjected to different torques as shown in Fig. 7. The differential equation of the mechanical sub-system is given by (I.17) [5].

$$J_m \frac{d\omega_m}{dt} = \tau_{em} - B\omega_m - \tau_{dm} - \tau_l \qquad (I.17)$$

Where: $\tau_{em} = K_m(-i_a \sin(p\theta_m) + i_b \cos(p\theta_m))$ (I.18)

and $\tau_{dm} = T_{dm} \sin(2p\theta_m + \alpha)$ (I.19)

In the mechanical model, J_m [kg. m^2] is the motor moment of inertia, B [kg/s.m] is the motor viscous friction coefficient, τ_{em} [$N.m$] is the electromagnetic torque, T_{dm} [$N.m$] is the detent torque applied, α is the phase shift related to τ_{dm} , and τ_l [$N.m$] is the applied load torque [5].

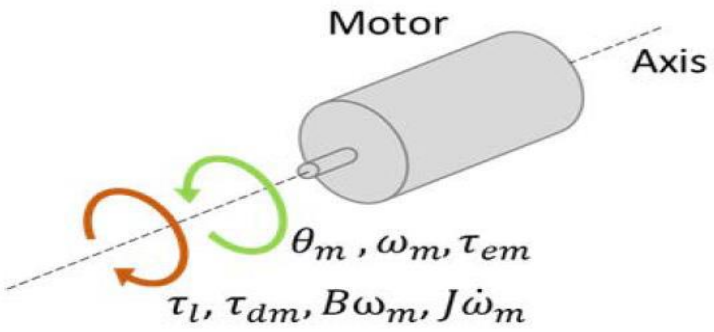


Figure I. 10. Mechanical sub-system of the hybrid stepper motor

3.2. DC Motor

3.2.1. Principle of DC Motor

The principle of the DC motor is to convert direct current (DC) electrical energy into mechanical energy. It consists of a stator (inductor) an armature (rotor) with windings of insulated wire which are energized by a commutator though brushes [13].

A stator magnetic field B is created by permanent magnet or excitation coil supplied from the DC source. The rotor winding (armature conductors), which can rotate, is placed in this stator magnetic field B. This armature is energized by a direct current, which creates a rotating magnetic field. This rotating field across the armature reacts with the stator magnetic field to create a force F on the rotor winding which causes it to rotate. This force F acts on both conductors and we have $\vec{F} = I \cdot \vec{l} \wedge \vec{B}$. With the Fleming's left hand rule, we can determine the direction of the force, which acts on the both conductors. Therefore, there are two forces created. Each of these acts on a conductor in the opposite direction and creates a torque [13].

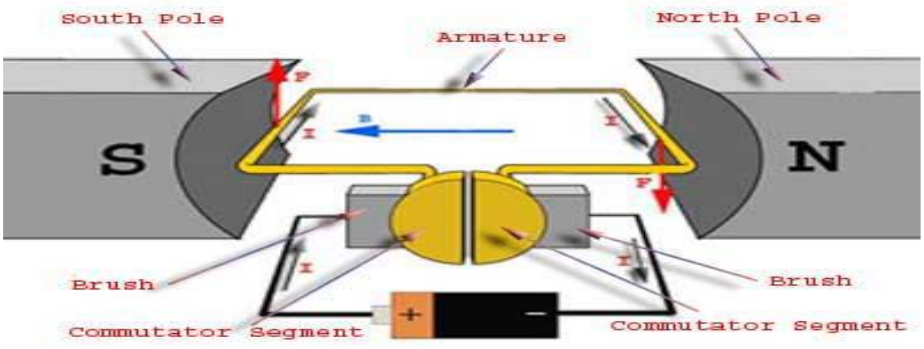


Figure I. 11. *Schema of the principle of DC Motor*

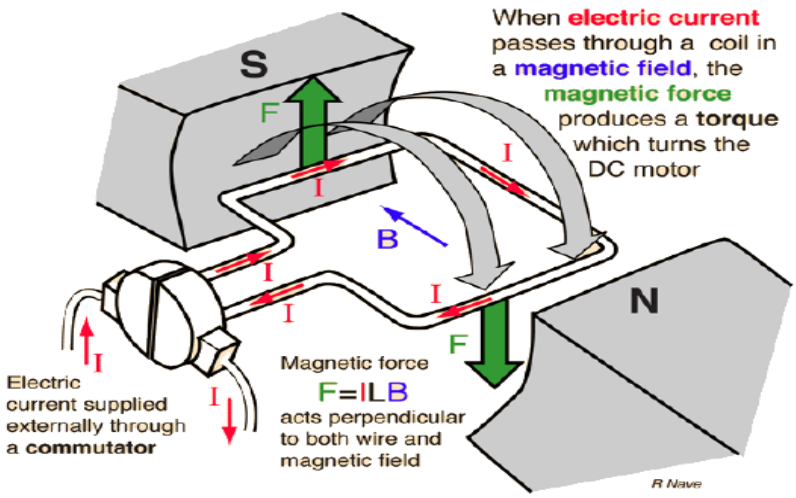


Figure I. 12. Schema of the operation of DC Motor

The brushes allow the rotor winding to continue its rotation when it reaches the perpendicular position. In fact, the brushes allow the current direction in the both conductors to commutate when this position is reached. The direction of the current is inverted, the direction of the force is too (Fleming's left hand rule) and the rotation can continue [13].

3.2.2. Components of the DC Motor

3.2.2.1. Inductor / Stator

In the DC Motor, the stator is the stationary part. It consists of permanent magnets or winding of excitation coil to create the magnetic field thanks to the electric current, which passes through this winding. When the stator is composed of permanent magnets, there are not power losses by Joule heating but the magnetic excitation field is constant. This solution is more expensive depending on the size of the motor [13].

3.2.2.2. Armature / Rotor

The rotor of the DC Motor is composed of iron laminated sheets package. Into the package slots, rotor winding is immersed. This is very difficult to product and it is expensive. Winding ends are connected to the commutator, which is created by the copper lamellas. By means of carbonic brushes, rotor winding is connecting to the fix part of machine. This technology allows to create a rotating magnetic field which interacts with the magnetic field created by the stator and generates the rotation [13].

3.2.3. Types of DC Motors

3.2.3.1. Separately excited DC Motor

A shunt field windings are supplied from a separate constant DC power source (like Battery) for producing the magnetic flux, are represented by resistor R_f. The resistor R_{fc} represents an external variable resistor (sometimes lumped together with the field coil resistance) used to control the amount of current in the field circuit [14].

The armature windings are represented by back $emf\ E_b$ and a resistorR_a. Are supplied from a DC power source ($\boldsymbol{V_t}$) [14].

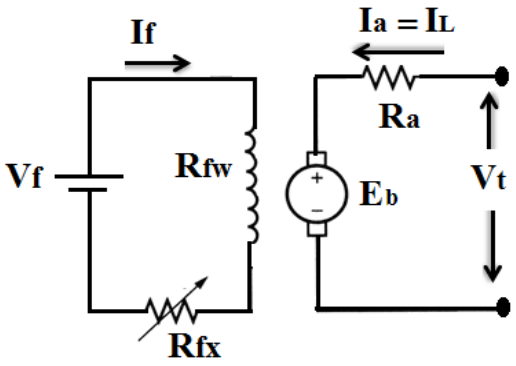


Figure I. 13. Separately Excited DC Motor

$$E_b = K_a\ \varphi\ w_m \quad (I.20)$$

$$V_t = E_b + I_a R_a \quad (I.21)$$

$$V_f = I_f * R_f \quad (I.22)$$

$$T_a = K_a\ \varphi\ I_a \quad (I.23)$$

R_{fw}: Resistance of field winding.

R_{fc}: Resistance of control rheostat used in field circuit.

$R_f = R_{fw} + R_{fx}$: Total field resistance

R_a: Resistance of armature circuit.

3.2.3.2. Self-Excited DC Motor

Field windings gets its power from the armature terminals of the motor [14].

3.2.3.2.1. Shunt DC motor

A shunt winding gets its power from the armature terminals of the motor. Shunt field winding connected across (in parallel with) the armature terminals [14].

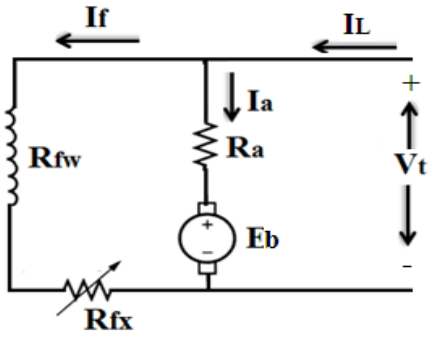


Figure I. 14. Shunt DC motor

$$V_f = V_t = I_f * R_f \quad (I.24)$$

$$V_t = E_b + I_a R_a \quad (I.25)$$

$$E_b = K_a \varphi w_m \quad (I.26)$$

$$I_L = I_a - I_f 0 \quad (I.27)$$

$$T_a = K_a \varphi I_a \quad (I.28)$$

3.2.3.2.2. Series DC motor

The series field winding connected in series with the armature windings [14].

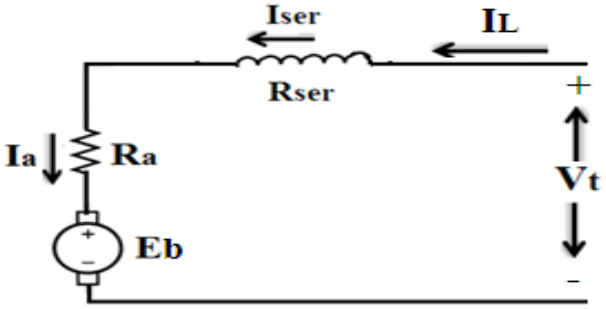


Figure I. 15. Series DC motor

$$I_L = I_{ser} = I_a \quad (I.29)$$

$$V_t = E_b + I_a(R_a + R_{ser}) \quad (I.30)$$

$$T_a = K_a \varphi I_a \quad (I.31)$$

$$\varphi = C I_a \quad (I.32)$$

$$T_a = K_a C I_a^2 \quad (I.33)$$

3.2.3.2.3. Compound DC motor

Both **shunt** and **series** field windings are connected with the armature windings in short-shunt or long-shunt [14].

3.2.3.2.3.1. Short shunt compound DC Motor

When the shunt field winding is connected directly across the armature terminals, it is called a[14]

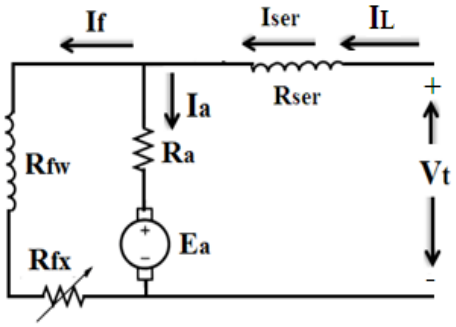


Figure I. 16. Short-shunt compound motor.

$$I_L = I_{ser} \quad (I.34)$$

$$I_a = I_L - I_f \quad (I.35)$$

$$I_f = \frac{V_t + I_{ser} * R_{ser}}{R_f} \quad (I.36)$$

$$V_t = E_b + (I_a * R_a) + (I_{ser} * R_{ser}) \quad (I.37)$$

3.2.3.2.3.2. Long shunt compound DC motor

When the shunt field winding is connected across the load, it is called a **long-shunt compound motor** [14].

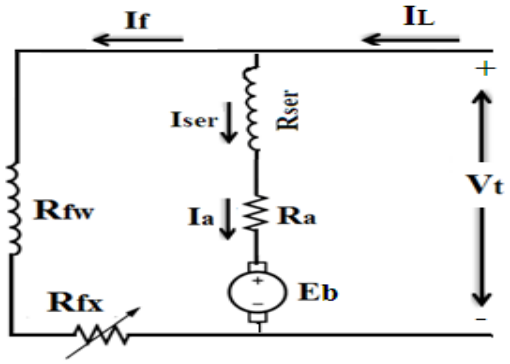


Figure I. 17. Long shunt compound DC motor

$$I_f = \frac{V_t}{R_f} \quad (I.38)$$

$$I_a = I_{ser} = I_L - I_f \quad (I.39)$$

$$V_t = E_b + (I_a * R_a) + (I_{ser} * R_{ser}) \quad (I.40)$$

$$V_t = E_b + I_a(R_a + R_{ser}) \quad (I.41)$$

3.2.4. DC Motor modelling

In an armature-controlled DC motor, the excitation for the field winding is kept constant and the torque is varied by varying the supply voltage connected to the armature. In some cases, a permanent magnet is used instead of field winding to produce the magnetic flux, which is again independent of the armature current. Such motors are called Permanent magnet DC motors.

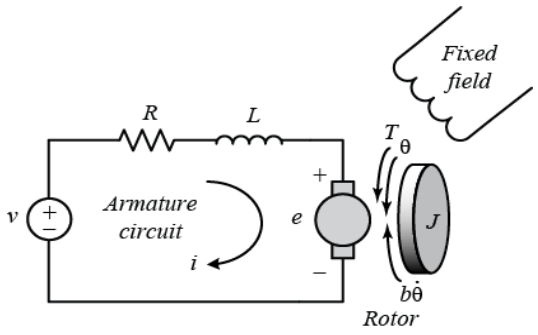


Figure I. 18. Armature-Controlled DC Motor.

The mathematical equations for this electrical system are represented below in detail:

By Kirchhoff voltage law [15],

$$R_a i_a(t) + L_a \frac{di_a(t)}{dt} = v_a(t) - e_b(t) \quad (I.42)$$

Where $e_b(t)$ is the back EMF produced in the DC motor which is proportional to derivative of angular position of the shaft nothing but speed or angular velocity [15].

$$e_b(t) = K_e\omega(t) \tag{I.43}$$

Torque of DC motor can be expressed in terms of product of field flux and armature current. Since in armature-controlled DC Motor field flux is constant, Torque varies only with respect to Armature current. Such torque expression can be given as [15].

$$T(t) = K_t i_a(t) \tag{I.44}$$

The mathematical expression presiding over this DC Motor is given by [15]

$$J\frac{d\omega(t)}{dt} + B\omega(t) = T(t) - T_L(t) \tag{I.45}$$

where:
- R_a : is the armature resistance (Ω)
- L_a : is armature inductance (H)
- J: is the equivalent moment of inertia of the motor ($kg.m^2/s^2$)
- B: is the viscous friction coefficient of the motor ($N.m.s/rad$)
- K_t: *is the torque constant* (Nm/A)
- K_e: *is the back emf constant* ($V/rad/s$)
- $i_a(t)$: *is the armature current* (A)
- $v_a(t)$: *is the armature voltage* (V)
- $\omega(t)$: is the angular speed (rad/s)
- $T_L(t)$: is the load torque across the motor shaft (Nm)

3.2.4.1. Transfer function of armature-controlled DC motor

From the equations (I.42), (I.43), (I.44) and (I.45) we obtained [16]:

$$\begin{cases} L_a\frac{di_a(t)}{dt} + R_a i_a(t) + K_e\omega(t) = v_a(t) \\ J\frac{d\omega(t)}{dt} + B\omega(t) - K_t i_a(t) = -T_L(t) \end{cases} \tag{I.46}$$

Assume that all the initial conditions are zero. Taking the Laplace transform of equations (I.46)

Results in:

$$\begin{cases} L_a sI_a(s) + R_a I_a(s) + K_e\Omega(s) = V_a(s) \\ Js\Omega(s) + B\Omega(s) - K_t I_a(s) = -T_L(s) \end{cases} \tag{I.47}$$

The transfer function relating the armature voltage $V_a(s)$ and angular velocity $\Omega(s)$, with $T_L(s) = 0$:

$$\frac{\Omega(s)}{V_a(s)} = \frac{K_t}{L_aJs^2+(L_aB+R_aJ)s+R_aB+K_tK_e} \quad (I.48)$$

4. Conclusion

In this chapter, we talked about our system modeling for each of the shape and the actuators, there are many choices with the actuators, and we worked with both of stepper motor and DC motor, but we faced a problem with the stepper motor because it was extremely heavy for our system. Therefore, we need to use another actuator, and we choose direct current motor because it is light and simple to use and we expect to have good results.

PID Control of TWSBR

1. Introduction

It is well known that the classical proportional-integral derivative (PID) controller still plays a dominating role in engineering control systems by far. For example, it was reported that over 95% of process control loops are designed based on PID controller. A more recent survey published in 2017 shows that the impact rating of PID control is still much higher than the widely studied advanced control techniques, and that "we still have nothing that compares with PID"[17]. Designing a PID controller for a complex nonlinear system is appropriate to know How true is this saying, and the effectiveness of this controller and its shortcomings, when it comes to the tuning the PID controller can be tuned by simple rule-based tuning methods like Ziegler-Nichols, unlike many other model-based tuning of the PID parameter.

2. Continuous and Digital Control

2.1. Feedback control system

The feedback is an electric signal, which is transferred from the output to the controller, so the controller would be able to calculate how the output is different from the required value. The controller would know the past state of the system output with the help of this feedback signal. The controller would calculate the error and it would vary the system input to get the required output. The closed-loop system is also called a feedback system because it fed back the output into the input so that the errors can be cleared and the required output quality can be maintained. The feedback control is to remove the measured disturbances. The feedback controls are automated and it should do the sensing, calculating, and manipulating to be performed by the equipment, and that each element must communicate with the other elements in the control system [6].

2.1.1. The characteristics of feedback

- Good accuracy
- It has a slight tendency towards oscillation (instability)
- The non-linear effects are reduced
- The effects of external disturbances or noise can be reduced
- The effects of external disturbance and noise are reduced
- Product quality can be increased
- The response to the variation in the input is high[6]

2.1.2. Feedback control process

In a control system, a feedback loop is an important tool; the feedback loop will consider the system output and this will help the system to alter its operation in order to get the required output. In order to do this task, a system should have controllers, compensators, and feedback structures to the system. The feedback control system is composed of sensors, controllers, process systems, etc. The major components of a feedback control system are input, the process that is being controlled, output, sensing elements, controllers, and actuating devices. The feedback control would only need minimal knowledge of the process that is to be controlled [6].

The feedback control can be considered as the easy way to automate the control of a process. The sensors in the control system would measure the actual value of the controlled variable. Therefore, after measuring the value it will be transmitted to the feedback controller. The controller would compare the measured value to the desired value and the difference between these values is considered as error. Therefore, if there were an error then the controller would send a proportional output to the control value. The controller would send a signal to the plant and sets the process variable, according to the set point. So in case if there is a disturbance that affects the process variable of the plant then the sensor in the feedback section would detect this and sends a signal to the controller [6].

2.1.3. Objectives of feedback control

The major objective of feedback control is to ensure that the output value of the system must be similar to the required value. This is done by tracking the reference trajectories or by maintaining close to their set points. The feedback signal would resist the disturbance signal, which affects the output, and it can improve the performance of the system. The feedback control can also stabilize the unstable plant [6].

2.1.4. Types of feedback system

2.1.4.1. Negative feedback

By definition, negative feedback is when a change (increase/decrease) in some variable results in an opposite change (decrease/increase) in a second variable. This is demonstrated in Figure II.1 where a loop represents a variation toward a plus that triggers a correction toward the minus, and vice versa. Negative feedback leads to a tight control situation whereby the corrective

action taken by the controller forces the controlled variable toward the set point, thus leading the system to oscillate around equilibrium [7].

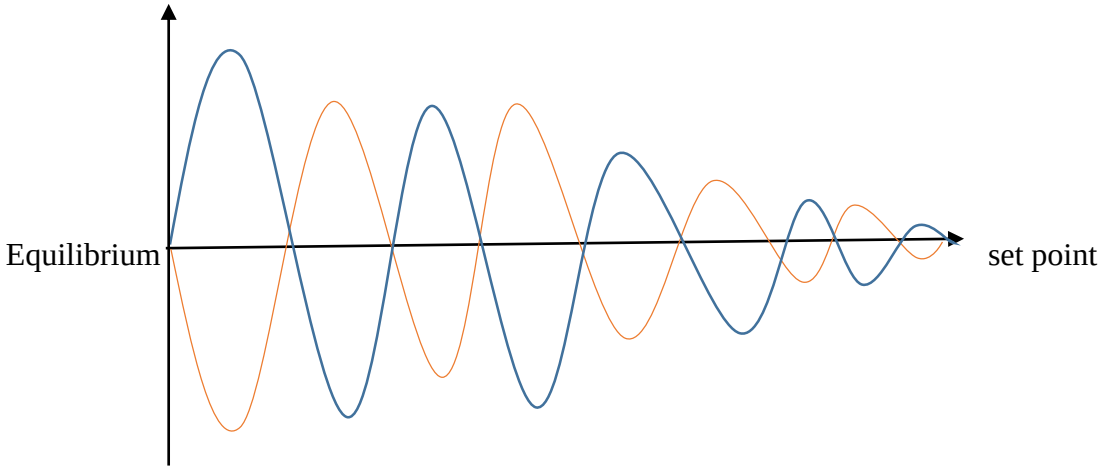


Figure II. 1. Negative Feedback: Maintenance of equilibrium and convergence

2.1.4.2. Positive Feedback

As opposed to negative feedback, positive feedback is when a change (increase/decrease) in some variable results in a subsequently similar change (increase/decrease) in a second variable. In some cases, positive feedback leads to an undesirable behavior whereby the system diverges away from equilibrium. This can cause the system to either run away toward infinity, risking an expansion or even an explosion, or run away toward zero, which leads to a total blocking of activities (Figure II.2) [7].

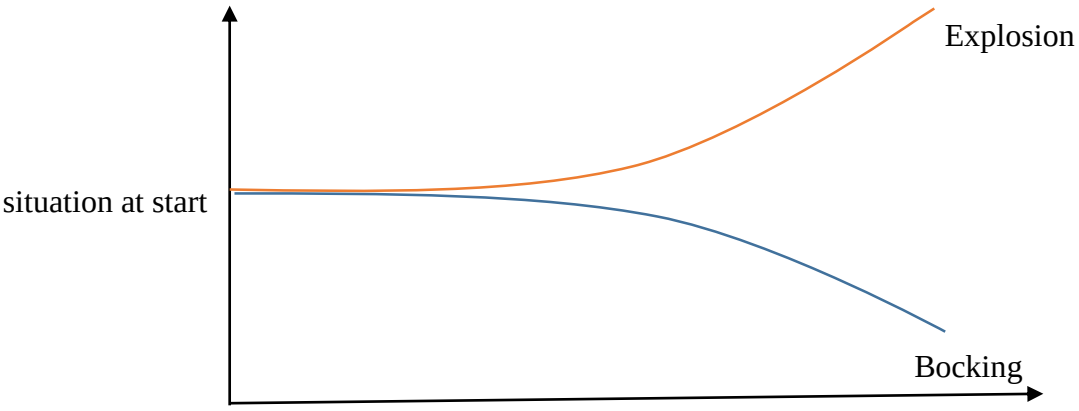


Figure II. 2. Positive Feedback: Exponential Growth and divergent behavior, no intermediate situation

2.1.5. Advantages and disadvantages of a feedback control

The unique architecture of the feedback control provides for many advantages and disadvantages. It is important to look at the exact application the control will be used for before determining which type of control will be the best choice (see Cascade Control System sand Feed Forward Control) [7].

Advantages: The advantages of feedback control lie in the fact that the feedback control obtains data at the process output. Because of this, the control takes into account unforeseen disturbances such as frictional and pressure losses. Feedback control architecture ensures the desired performance by altering the inputs immediately once deviations are observed regardless of what caused the disturbance. An additional advantage of feedback control is that by analyzing the output of a system, unstable processes may be stabilized. Feedback controls do not require detailed knowledge of the system and, in particular, do not require a mathematical model of the process. Feedback controls can be easily duplicated from one system to another. A feedback control system consists of five basic components: (1) input, (2) process being controlled, (3) output, (4) sensing elements, and (5) controller and actuating devices. A final advantage of feedback control stems from the ability to track the process output and, thus, track the system's overall performance [7].

Disadvantages: Time lag in a system causes the main disadvantage of feedback control. With feedback control, a process deviation occurring near the beginning of the process will not be recognized until the process output. The feedback control will then have to adjust the process inputs in order to correct this deviation. This results in the possibility of substantial deviation throughout the entire process. The system could possibly miss process output disturbance and the error could continue without adjustment. Generally, feedback controllers only take input from one sensor. This may be inefficient if there is a more direct way to control a system using multiple sensors. Operator intervention is generally required when a feedback controller proves unable to maintain stable closed-loop control. Because the control responds to the perturbation after its occurrence, perfect control of the system is theoretically impossible. Finally, feedback control does not take predictive control action towards the effects of known disturbances [7].

3. Digital control

In most modern engineering systems, it is necessary to control the evolution with time of one or more of the system variables. Controllers are required to ensure satisfactory transient and steady-state behavior for these engineering systems. To guarantee satisfactory performance in the presence of disturbances and model uncertainty, most controllers in use today employ some form of negative feedback. A sensor is needed to measure the controlled variable and compare its behavior to a reference signal. Control action is based on an error signal defined as the difference between the reference and the actual values. The controller that manipulates the error signal to determine the desired control action has classically been an analog system, which includes electrical, fluid, pneumatic, or mechanical components. These systems all have analog inputs and outputs (i.e., their input and output signals are defined over a continuous time interval and have values that are defined over a continuous range of amplitudes). In the past few decades, digital controllers whose inputs and outputs are defined at discrete time instances have often replaced analog controllers. The digital controllers are in the form of digital circuits, digital computers, or microprocessors. Intuitively, one would think that controllers that continuously monitor the output of a system would be superior to those that base their control on sampled values of the output. It would seem that control variables (controller outputs), that change continuously would achieve better control than those that change periodically. This is in fact true! Had all other factors been identical for digital and analog control, analog control would be superior to digital control. What, then, is the reason behind the change from analog to digital that has occurred over the past few decades [18].

Digital control systems employ a computer as a fundamental component in the controller. The computer typically receives a measurement of the controlled variable, also often receives the reference input, and produces its output using an algorithm. This output is usually converted to an analog signal using a D/A converter, then amplified by a power amplifier to drive the plant [18].

3.1. Structure of a digital control system

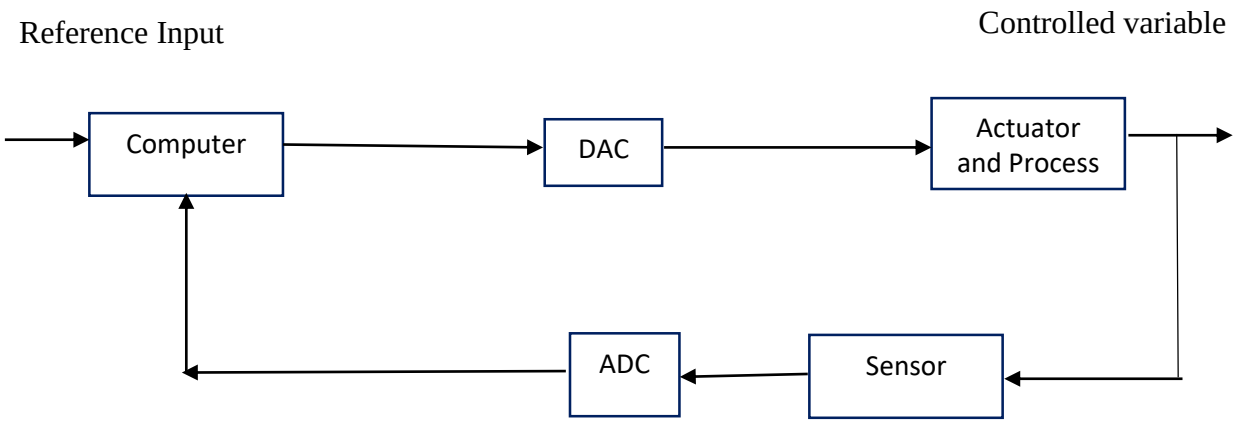


Figure II. 3. Configuration of a digital control system

To control a physical system or process using a digital controller, the controller must receive measurements from the system, process them, and then send control signals to the actuator that effects the control action. In almost all applications, both the plant and the actuator are analog systems. This is a situation where the controller and the controlled do not "speak the same language," and some form of translation is required. The translation from controller language (digital) to physical process language (analog) is performed by a digital-to-analog converter or DAC. The translation from process language to digital controller language is performed by an analog-to-digital converter, or ADC. A sensor is needed to monitor the controlled variable for feedback control. The combination of the elements discussed here in a control loop is shown in Figure 10. Variations on this control configuration are possible. For example, the system could have several reference inputs and controlled variables, each with a loop similar to that of Figure 10. The system could also include an inner loop with digital or analog control [18].

4. PID controller

4.1. Brief history

There is an apocryphal story you might have heard. A brilliant graduate student was working at a prestigious institution under a famous professor of control theory. This ingenious student managed to solve several of the deepest longstanding problems of control theory, developing a nonlinear, adaptive control algorithm that was guaranteed to converge globally, under extremely

general conditions of noise and modeling uncertainty, to a controller that rep- resented the best possible trade-offs among stability, robustness, and performance, both transient and steady state. All that remained to be done was the computer implementation. Unfortunately, the computational burden was immense, and years passed before a sufficiently powerful computer could be harnessed to perform the massive computations. Finally, the algorithm was implemented, and a group of distinguished researchers, all experts in the most advanced methods and theories of control, waited expectantly for the ultimate controller. When the computations were finished, the answer appeared PID [12].

Brief history of PID Controller PIDs combine proportional-integral-derivative control action. In 1788, James Watt included a flyball governor, the first mechanical feedback device with only a proportional function, into his steam engine. The flyball governor controlled the speed by applying more steam to the engine when the speed dropped lower than a set point, and vice versa. In 1933, the Taylor Instrumental Company introduced the first pneumatic controller with a fully tunable proportional controller. However, a proportional controller is not sufficient to control speed thoroughly, as it amplifies error by multiplying it by some constant (K_p). The error generated is eventually small, but not zero. In other words, it generates a steady state error each time the controller responds to the load. Around 1930s, control engineers discovered that steady state error could be eliminated by resetting the set point to some artificial higher or lower value, as long as the error nonzero. This resetting operation integrates the error, and the result is added to the proportional term; today this is known as Proportional-Integral controller. In 1934-1935, Foxboro introduced the first PI controller. However, PI controllers can over-correct errors and cause closed-loop instability. This happens when the controller reacts too fast and too aggressively; it creates a new set of errors, even opposite to the real error. This is known as “hunting” problem. In 1920s, there were suggestions of including the rate of change of error in conjunction with PI controller. In 1940, Taylor Instrument Companies successfully produced the first PID pneumatic controller; the derivative action was called “pre-act”. With an extra derivative action, problems such as overshoot and hunting are reduced. However, issues like finding the appropriate parameter of PID controllers were yet to be solved. In 1942, Taylor Instrument Company's Ziegler and Nichols introduced Ziegler-Nichols tuning rules. Their well-known paper “Optimum settings for automatic controllers”, presented two procedures for establishing the appropriate parameters for PID controllers. However, the PID controller was not popular at that time, as it was not a simple

concept; the parameters the manufacturers required to be tuned did not make much sense to the users. In the mid 1950's, automatic controllers were widely adopted in industries. A report from the Department of Scientific and Industrial Research of United Kingdom state, "Modern controlling units may be operated mechanically, hydraulically, pneumatically or electrically. The pneumatic type is technically the most advanced and many reliable designs are available. It is thought that more than 90 percent of the existing units are pneumatic." The report indicated the need to implement controllers in electrical and electronic form. In 1951, The Swartwout Company introduced their first electronic PID controller, based on vacuum tube technology. Around 1957, the manufacturers started to realize the possibility of implementing the controllers in transistors. In 1959, the first solid-state electronic controller was introduced by Bailey Meter Co. The advantage of using electronic instrument to implement PID controller was explored more deeply years later. They are not only capable of including the functions available in pneumatic instruments, but even more complicated mathematical operations can be carried out as well. Electronic PID controllers became more common and more acceptable since then [8].

4.2. PID Controller

A PID controller is a simple three terms equation. Two terms are functions of the error between the measured angle of the robot and the desired angle, which is 90 degrees, the third term is just a constant. Each term is multiplied in a parameter [9].

These three parameters are called proportional, integral, and derivative parameters. They are denoted, K_p, K_i and K_d respectfully. Below is the equation for a PID controller output [9]:

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{e(t)}{dt} \qquad (II.1)$$

The proportional term applies appropriate proportional changes for error (which is the difference between the set point and process variable) to the control output. In fact, many control applications work quite well with only proportional control [10].

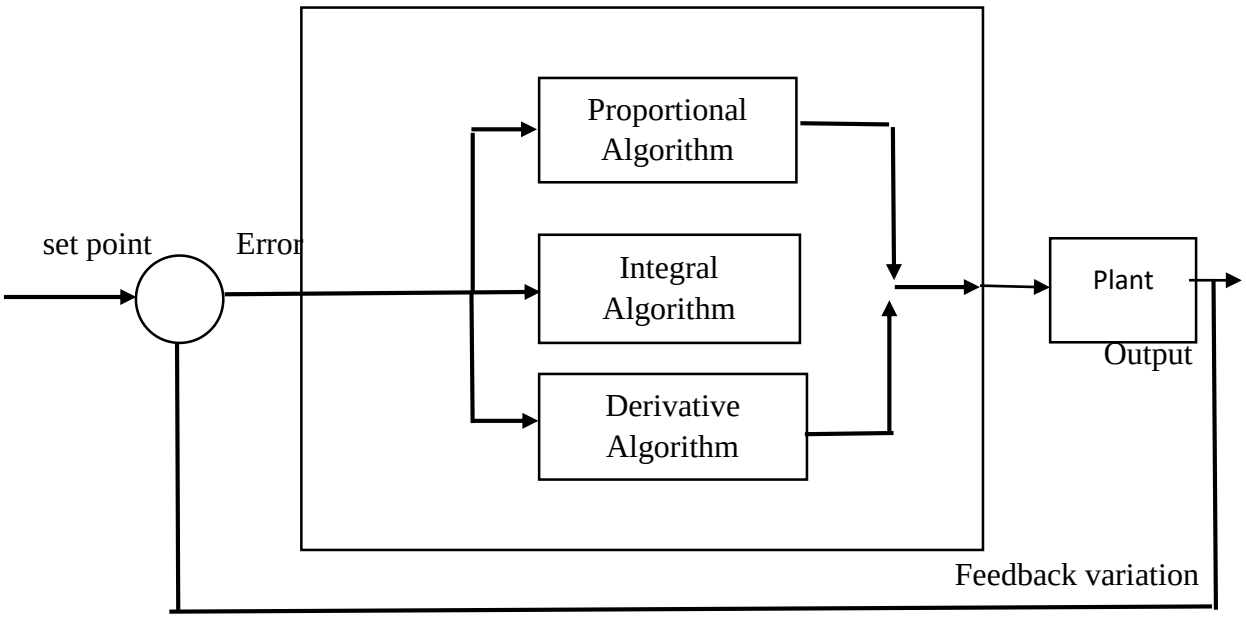


Figure II. 4. Diagram Bloc of PID Controller

4.3. Type of Classical *PID* Controller

There are three most commonly used classical *PID* controller, namely parallel, ideal or ISA and series or interacting *PID* controller:

4.3.1. Parallel PID

Proportional, integral and derivative actions are working separately with each other and combine effect of these three actions are act in the system.

- The parallel *PID* controller provides a control effort u(t) given by:

$$u(t) = k_p e(t) + k_i \int e(t)dt + k_d \frac{de(t)}{dt} \qquad (II.2)$$

- The corresponding controller transfer function is defined as the ratio of the controller _output U(s) and error E(s) as:

$$C(s) = \frac{U(s)}{E(s)} = k_p + \frac{k_i}{s} + k_d s \qquad (II.3)$$

- The parallel structure of a classical *PID* controller.

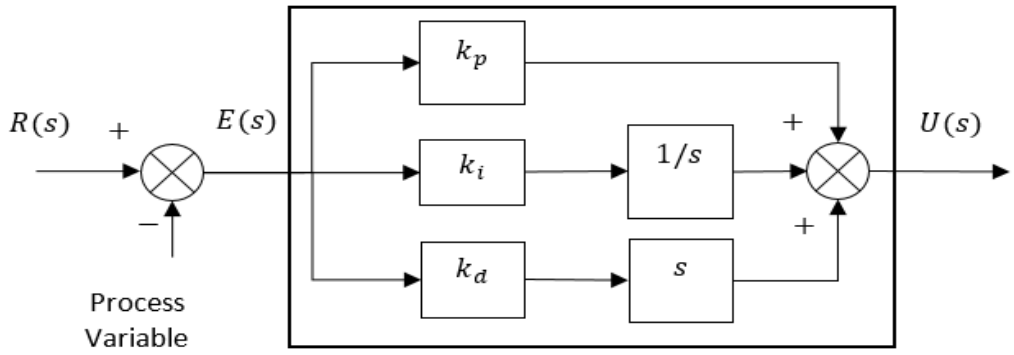


Figure II. 5. Parallel structure of a classical PID controller

4.3.2. Ideal PID

The gain constant K_p is distributed to all term. So, changes in K_p affects all other terms.

- The ideal PID controller provides a control effort u(t) given by:

$$u(t) = k_p\left(e(t) + T_i\int e(t)dt + T_d\frac{de(t)}{dt}\right) \qquad (II.4)$$

- The corresponding controller transfer function is defined as the ratio of the controller output U(s) and error E(s) as:

$$C(s) = \frac{U(s)}{E(s)} = k_p\left(1 + \frac{T_i}{s} + T_d s\right) \qquad (II.5)$$

- The ideal structure of a classical PID controller.

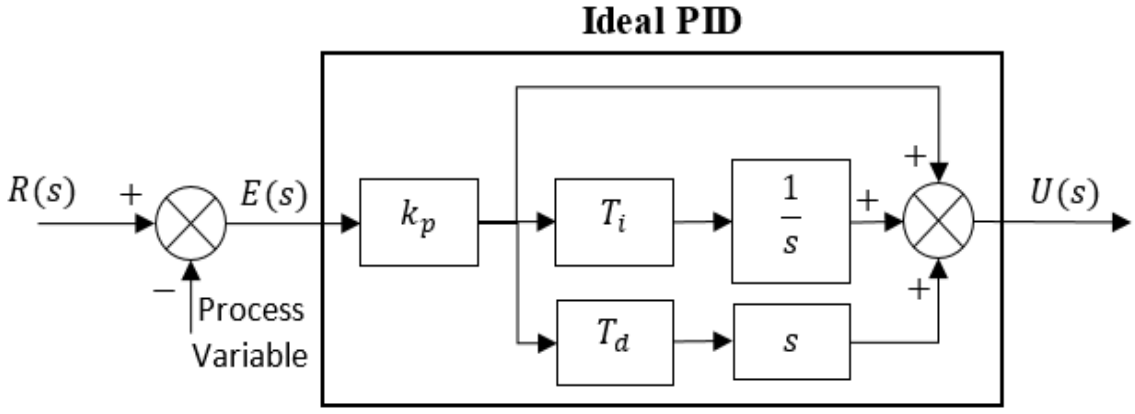


Figure II. 6. Ideal structure of a classical PID controller

4.3.3. Series PID

The gain constant kp is distributed to all terms same as idealPID, but in this form integral and derivative constant have an effect on proportional action.

- The series PID controller provides a control effort u(t) given by:

$$u(t) = k_p\left(e(t) + T_i\int e(t)dt\right)\left(e(t) + T_d\frac{de(t)}{dt}\right) \qquad (II.6)$$

- The corresponding controller transfer function is defined as the ratio of the controller output U(s) and error E(s) as:

$$C(s) = \frac{U(s)}{E(s)} = k_p\left(1 + \frac{T_i}{s}\right)(1 + T_d s) \qquad (II.7)$$

- The series structure of a classical PID controller.

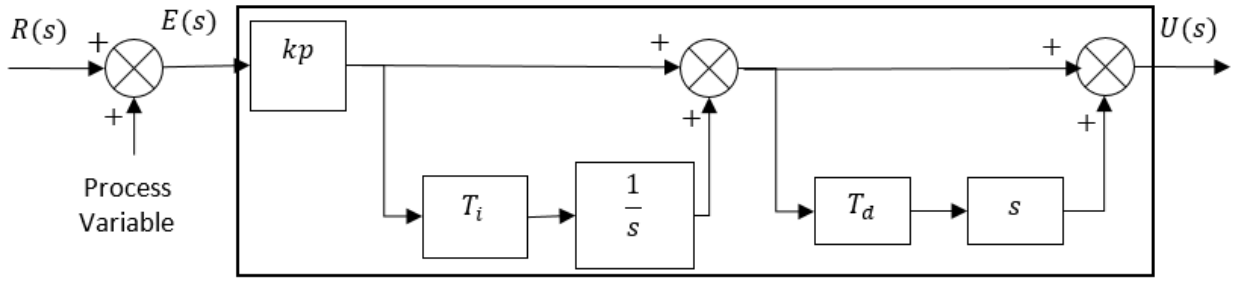


Figure II. 7. Series structure of a classical PID controller

4.4. PID Tuning

PID tuning is the process of finding the values of proportional, integral, and derivative gains of a PID controller to achieve desired performance and meet design requirements [20], all general methods for control design can be applied to PID control. A number of special methods that are tailor-made for PID control have also been developed, these methods are often called tuning methods. Irrespective of the method used, it is essential to always consider the key elements of control, load disturbances, sensor noise, process uncertainty and reference signals. The most well known tuning methods are those developed by Ziegler and Nichols. They have had a major influence on the practice of PID control for more than half a century. The methods are based on characterization of process dynamics by a few parameters and simple equations for the controller

parameters. It is surprising that the methods are so widely referenced because they give moderately good tuning only in restricted situations. Plausible explanations may be the simplicity of the methods and the fact that they can be used for nonlinear system [21].

There are three methods to tune the PID controller parameter such as:

1- Manual tuning
2- Heuristic tuning
3- Auto tuning

The most popular method using to tune the PID parameter is Ziegler-Nichols tuning method

4.4.1. Ziegler–Nichols Rules for Tuning PID Controllers

Ziegler and Nichols proposed rules for determining values of the proportional gain K_p, integral time T_i and derivative time T_d based on the transient response characteristics of a given plant. Such determination of the parameters of PID controllers or tuning of PID controllers can be made by engineers on-site by experiments on the plant. (Numerous tuning rules for PID controllers have been proposed since the Ziegler–Nichols proposal. They are available in the literature and from the manufacturers of such controllers.) [19].

There are two methods called Ziegler–Nichols tuning rules: the first method and the second method. We shall give a brief presentation of these two methods [19].

4.4.1.1. First Method

In the first method, we obtain experimentally the response of the plant to a unit-step input, as shown in Figure II.8. If the plant involves neither integrator(s) nor dominant complex-conjugate poles, then such a unit-step response curve may look S-shaped, as shown in Figure II.8. This method applies if the response to a step input exhibits an S-shaped curve. Such step-response curves may be generated experimentally or from a dynamic simulation of the plant. The S-shaped curve may be characterized by two constants, delay time L and time constant T. The delay time and time constant are determined by drawing a tangent line at the inflection point of the S-shaped curve and determining the intersections of the tangent line with the time axis and line c(t)=K, as shown in Figure II.8 [19].

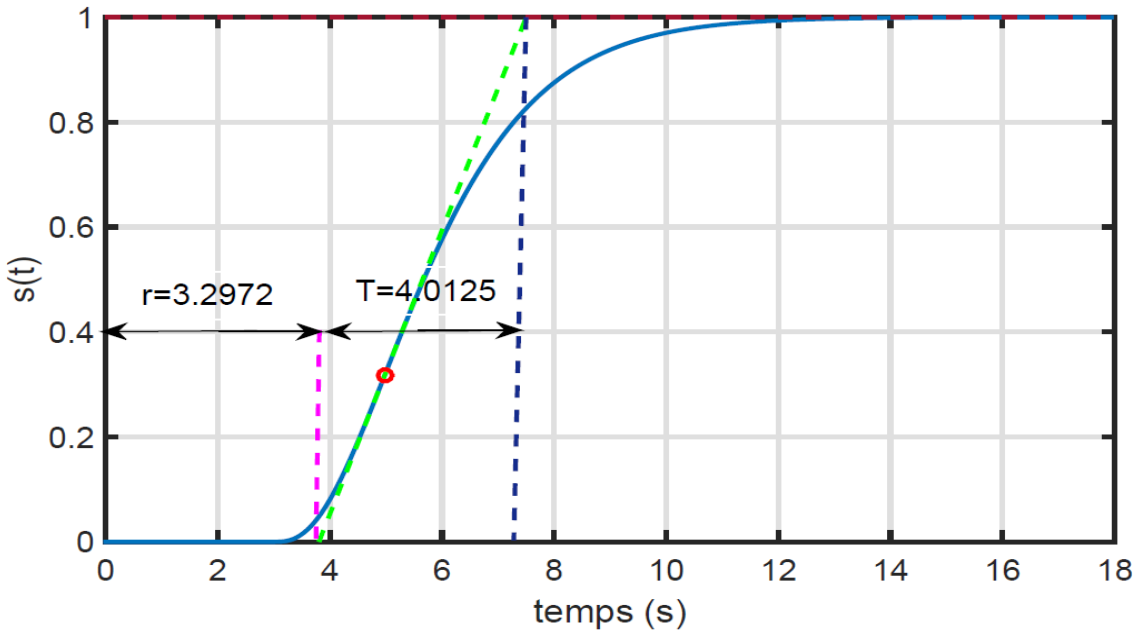


Figure II. 8. Unit step response of a plant and its s-shaped curve example

Table 1. Z-N first method table

Type of controller	KP	Ti	Td
P	T/L	∞	0
PI	0.9 T/L	L/0.3	0
PID	1.2 T/L	2L	0.5L

Ziegler and Nichols suggested to set the values of K_p , T_i and T_d according to the formula shown in Table 1. Notice that the PID controller tuned by the first method of Ziegler–Nichols rules gives[19].

$$\begin{aligned} G_c(s) &= K_p\left(1+\frac{1}{T_i s}+T_d s\right) \\ &= 1.2\frac{T}{L}\left(1+\frac{1}{2Ls}+0.5Ls\right) \\ &= 0.6T\frac{\left(s+\frac{1}{L}\right)^2}{s} \end{aligned} \qquad (II.8)$$

Thus, the PID controller has a pole at the origin and double zeros at s = –1/L [19].

4.4.1.2. Second Method: In the second method, we first set $T_i = \infty$ and $T_d = 0$ using the proportional control action only, increase K_p from 0 to a critical value K_{cr} at which the output first exhibits sustained oscillations. (If the output does not exhibit sustained oscillations for whatever value K_p may take, then this method does not apply.) Thus, the critical gain K_{cr} and the corresponding period P_{cr} are experimentally [19].

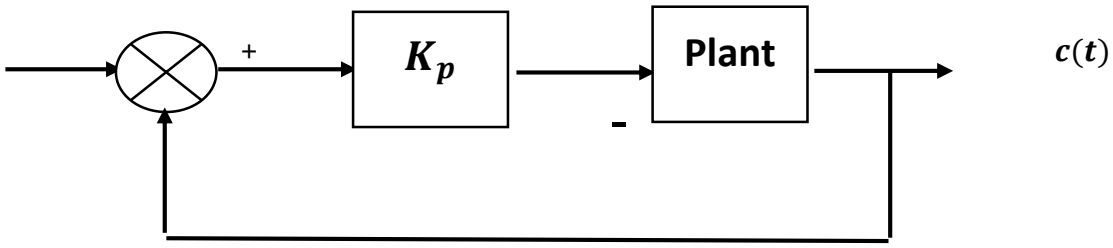


Figure II. 9. Closed-loop system with a proportional controller

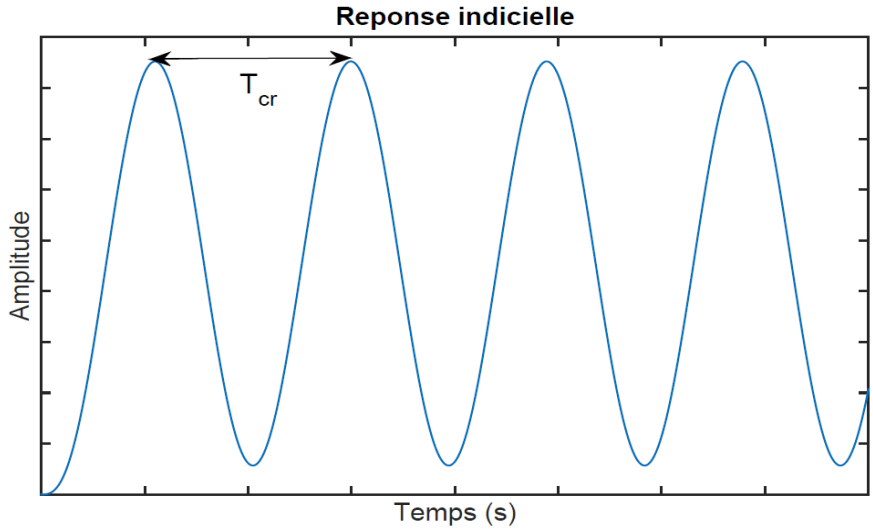


Figure II. 10. Sustained oscillation with period P_{cr} (P_{cr} is measured in sec.)

Determined (see Figure II.10). Ziegler and Nichols suggested that we set the values of the parameters K_p , T_i , and T_d according to the formula shown in Table 1 [19].

Table 2. Ziegler–Nichols Tuning Rule Based on Critical Gain K_{cr} and Critical Period P_{cr}(Second Method)

Type of controller	K_p	Ti	Td
P	$0.5\ K_{cr}$	∞	0
PI	$0.45\ K_{cr}$	$1/1.2\ P_{cr}$	0
PID	$0.6\ K_{cr}$	$0.5\ P_{cr}$	$0.125\ P_{cr}$

$$G_c(s)\& = K_p\left(1 + \frac{1}{T_i s} + T_d s\right) \quad (II.9)$$

$$= 0.6K_{\mathrm{cr}}\left(1 + \frac{1}{0.5P_{\mathrm{cr}}s} + 0.125P_{\mathrm{cr}}s\right) \quad (II.10)$$

$$= 0.075K_{\mathrm{cr}}P_{\mathrm{cr}}\frac{\left(s+\frac{4}{P_{\mathrm{cr}}}\right)^2}{s} \quad (II.11)$$

4.4.1.3. Control of TWSBR using two Ziegler-Nichols method

The PID control method is used to improve the system response to self-balancing robots. The response system obtained without using the PID control system was still far from the desired response and even the robot did not reach the specified set point. The first thing to do is utilized PID control method to calculate the values of K_p , K_i ,and K_d as parameters of the PID controller. The determination of PID parameters is based on the reaction of the closed-loop system in the transfer function of self-balancing robots. A K_p variable is added first to the system, thus the closed-loop transfer function is determined by adding the proportional gain (K_p), as follows [11].

$$\frac{\varphi(s)}{u(s)} = \frac{14{,}286\mathrm{s}}{s^3 + 0.286s^2 - 70\mathrm{s} - 0.0294} \quad (II.12)$$

$$\frac{\varphi(s)}{u(s)} = \frac{K_p\frac{14{,}286\mathrm{s}}{s^3 + 0.286s^2 - 70\mathrm{s} - 0.0294}}{1 + K_p\left(\frac{14{,}286\mathrm{s}}{s^3 + 0.286s^2 - 70\mathrm{s} - 0.0294}\right)} \quad (II.13)$$

$$\frac{\varphi(s)}{u(s)} = \frac{14{,}286\ K_p}{14{,}286\ K_p + s^3 + 0.286s^2 - 70s - 0.0294} \quad (II.14)$$

The stability of a system can be seen from the location of the pole system in the field s, if the poles of the system are located to the left of the field s, then the system is stable. To find out the location of the poles in a system, hence we used the stability of Routh. The first column in Routh stability shows the polar location of a system, if the variable in the column is, positive then it can be ascertained that a system has a pole on the left side of the field s. To get K_p value to meet the Routh stability criteria, it can be determined by referring to the transfer function that has been obtained in equation (42).

$$s^3 \rightarrow 1 \qquad -70 \quad (II.15)$$

$$s^2 \rightarrow 0.286 \qquad -0.0294 + 14.286 \quad (II.16)$$

$$s^1 \rightarrow \qquad \frac{-20.02 + (0.0294 - 14.286K_p)}{0.286} \quad (II.17)$$

$$s^0 \rightarrow -0.0294 + 14.286K_p \quad (II.18)$$

Then, s^1 can be analyzed as follows

$$\frac{-20.02 + (0.0294 - 14.286K_p)}{0.286} \geq 0 \quad (II.19)$$

$$-70 + 0.103 - 49.95 \geq 0 \quad (II.20)$$

$$-69.897 - 49.95K_p \geq 0 \quad (II.21)$$

$$K_p \geq -1.399 \quad (II.22)$$

Whereas, for s^0, as follows:

$$-0.0294 + 14.286\ K_p \geq 0 \quad (II.23)$$

$$K_p \geq 0.00205 \quad (II.24)$$

Therefore, we acquired the range of K_p for TWBR as follows:

$$0 \leq K_p \leq 0{,}00205 \quad (II.25)$$

K_{cr} and P_{cr} were yielded by substituting $j\omega$ in the variable s for denominator of equation (21). Denominator function over the closed-loop system commonly referred to as the equation of system characteristics. System characteristic equations with K_p parameter on self-balancing robots is as follows [11].

$$s^3 + 0.286s^2 - 70s - 0.0294 + 14{,}286\,K_p = 0 \quad (II.26)$$

$$-j\omega^3 - 0.286j\omega^2 + 70j\omega - 0.0294 + 14{,}286\,K_p = 0 \quad (II.27)$$

$$(-0.286\omega^2 - 0.0294 + 14{,}286\,K_p) + j\,(-\omega^3 + 70\omega) = 0 \quad (II.28)$$

From equation (II.28), we obtained two parts, namely the real and imaginary parts. The imaginary part will be used to get the value of ω, while the real part is used to get the K_p value [11].

$$(-\omega^3 + 70\omega) = 0 \quad (II.29)$$

$$\omega = 8.366. \quad (II.30)$$

According to equation (II.30), we yielded P_{cr}

$$P_{cr} = \frac{2\pi}{\omega} = \frac{2\pi}{8.366} = 0.751\ second \quad (II.31)$$

The K_{cr} value was obtained by using the real part by substituting the value ω.[11]

$$-0.286\omega^2 - 0.0294 + 14{,}286\,K_p = 0 \quad (II.32)$$

$$K_p = 1.3999 \quad (II.33)$$

$K_p = K_{cr} = 1.3999$. With K_{cr} and P_{cr} obtained values, thus, the value of the Ziegler-Nichols type two PID parameters for TWBMR is presented in Table 3. After the parameters of the K_P, K_I, and K_D values are obtained, a simulation can be carried out by adding the PID controller to the transfer function of the self-balancing robot, as shown in Figure II.11 [11].

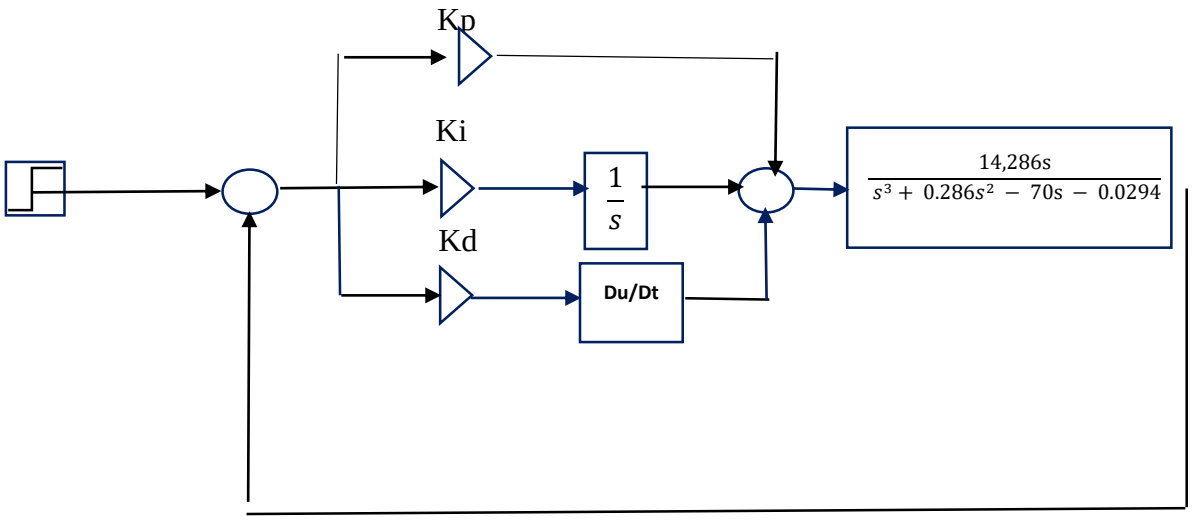


Figure II. 11. TWSBR system diagram with PID controller

Table 3. Parameters of the PID Ziegler-Nichols for TWSBR

Parameter	PID value
K_p	4.0602
$T_i = K_i$	4.4100
$T_d = K_d$	1.1025

Note

These parameters will not give as the desired results, so after testing and multiply these parameters with 2 and with 10 and with 25 we find that the best results will be with the parameters multiplied by 25

4.4.1.3.1 Simulation and Discussion

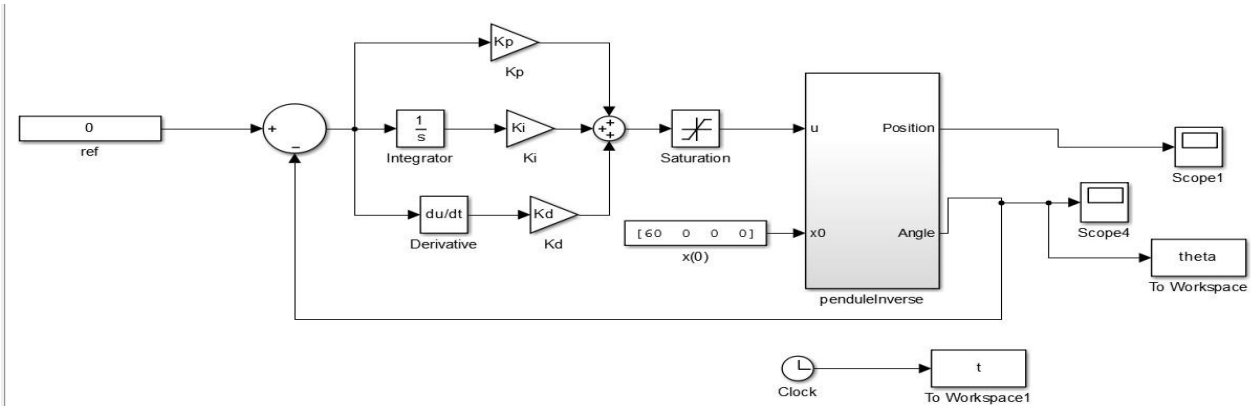


Figure II. 12. Simulink model of classical PID control.

Results:

- With initial values $x_0 = [0, 0, -45, 0]$

1- Angle:

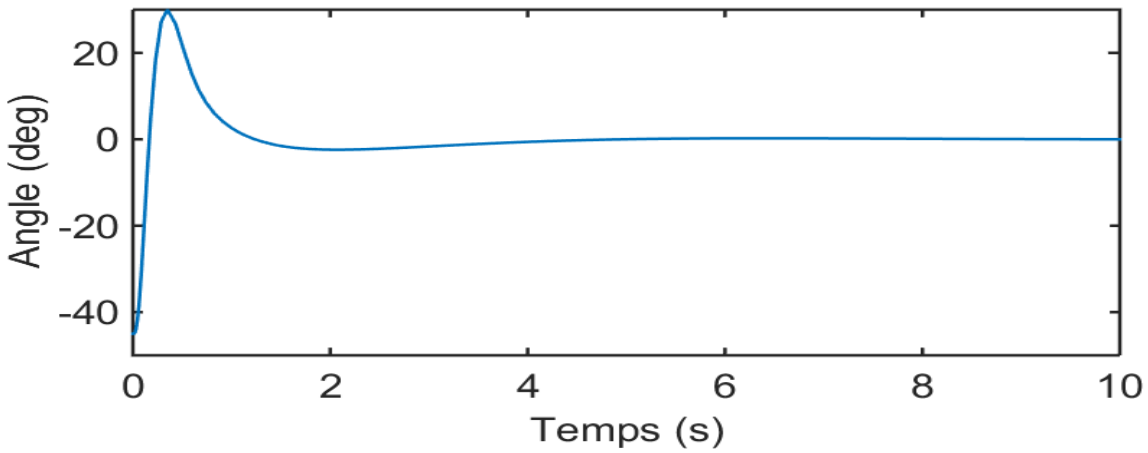


Figure II. 13. Response of angle with initial values $x_0 = [0, 0, -45, 0]$

This is the expected result, multiplied by 25 for theK_p, K_i, and K_d values, as shown in Figure 15. We noticed that the system reached its instable equilibrium point that we consider it 0° after a small rise time and a small overshoot, with the initial value -45 we can say that the TWSBR simulation is suitable for implementation with this PID controller parameters. The rise time is quite fast but still not perfect because of the controller itself not because the parameters of the controller and the value of steady-state error and settling time are quite small but still not perfect too.

2- Control input:

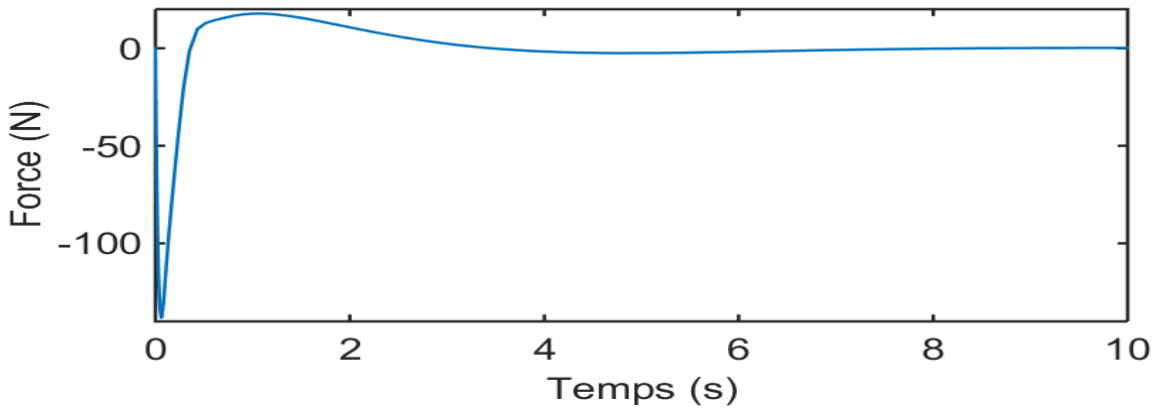


Figure II. 14. Power response with initial values $x_0 = [0, 0, -45, 0]$

- With initial values $x_0 = [0, 0, 60, 0]$:

1- Angle

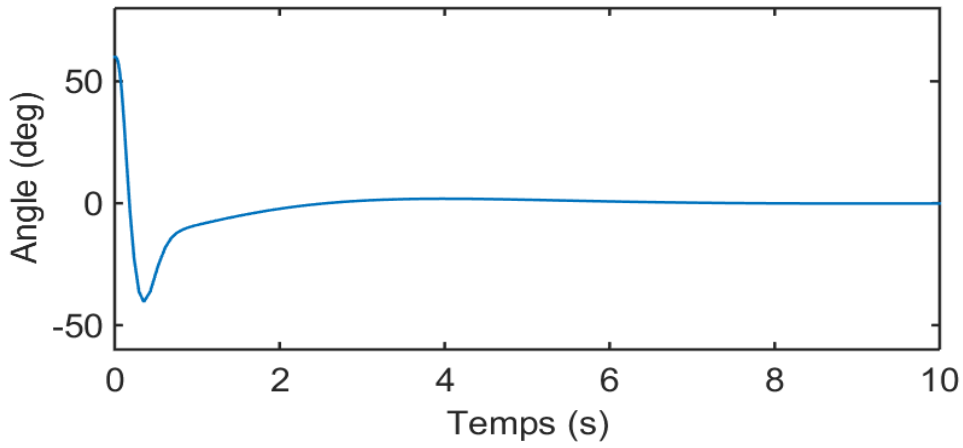


Figure II. 15. Response of angle with initial values $x_0 = [0, 0, 60, 0]$

This result with 60° of initial value shows that the system, the system reached its instable equilibrium point with almost 2 seconds, the overshoot value is pretty higher than the overshoot value that we found it in the previous section with initial values -45°, as well as the rise time. All the performances of the system are acceptable.

2- Control input

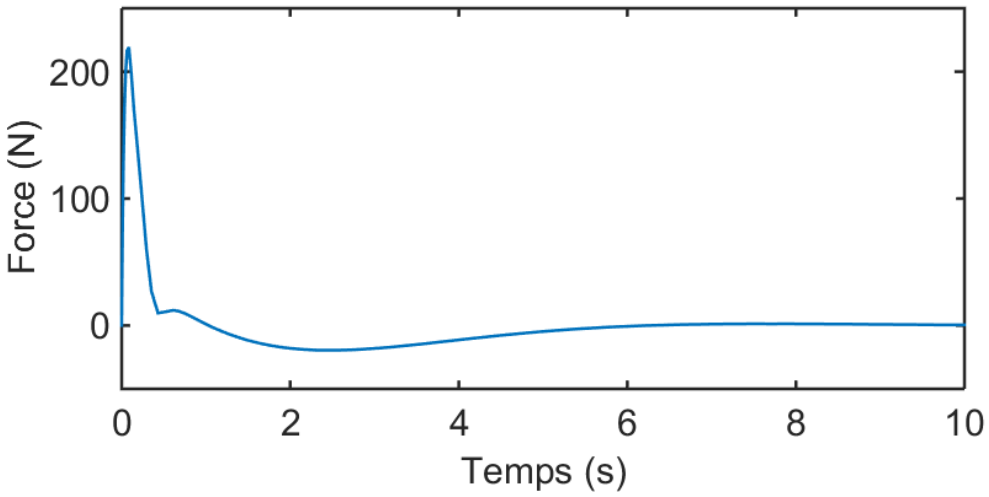


Figure II. 16. Power response with initial values $x_0 = [0, 0, 60, 0]$

5. Conclusion

Classical PID controller is a good controller but not a perfect one, however the algorithm of the tuning to get the parameters of the controller $K_p \; and \; K_i \; and \; K_d$ or even a manually determine of them, the results will not be as good as we want specially the system performances such as the overshoot, and the rise time and the delay time, the settling time and the peak time values , in this chapter we made a PID controller for our TWSBR and we find good results " minimum overshot and minimum delay time and minimum rise time" , The robot balances itself on two wheels without falling and with a slight jitter. However, we need better performances.

Therefore, we need to design another controller and compare the results and take the best controller.

CHAPTER 03

Adaptive Neural PID control for TWSBR

1. Introduction

An artificial neural network is a classifier modeled on the functioning of the human brain, which is very different from the way computer code is usually written.

A human brain contains a huge amount of nerve cells, the neurons. Each of these cells is connected to many other similar cells, creating a very complex network of signal transmission. Each cell collects input from all the other nerve cells to which it is connected, and if it reaches a certain threshold, it sends a signal to all the cells to which it is connected.

In control, engineering neural network is widely used algorithm and applying on many complex nonlinear system, and by mixing the neural network and the PID controller we get a high accuracy controller called Neural Network PID or NNPID.

2. Neural Network

2.1. History of Neural Networks

The study of the human brain is thousands of years old. With the advent of modern electronics, it was only natural to try to harness this thinking process. The first step toward artificial neural networks came in 1943 when Warren McCulloch, a neurophysiologist, and a young mathematician, Walter Pitts, wrote a paper on how neurons might work. They modeled a simple neural network with electrical circuits [8].

Reinforcing this concept of neurons and how they work was a book written by Donald Hebb. The Organization of Behavior was written in 1949.It pointed out that neural pathways are strengthened each time that they are used [8].

As computers advanced into their infancy of the 1950s, it became possible to begin to model the rudiments of these theories concerning human thought. Nathanial Rochester from the IBM research laboratories led the first effort to simulate a neural network. That first attempt failed. However, later attempts were successful. It was during this time that traditional computing began to flower and, as it did, the emphasis in computing left the neural research in the background [8].

Yet, throughout this time, advocates of "thinking machines" continued to argue their cases. In 1956, the Dartmouth Summer Research Project on Artificial Intelligence provided a boost to both artificial intelligence and neural networks. One of the outcomes of this process was to

stimulate research in both the intelligent side, AI, as it is known throughout the industry, and in the much lower level neural processing part of the brain [8].

In the years following the Dartmouth Project, John von Neumann suggested imitating simple neuron functions by using telegraph relays or vacuum tubes. In addition, Frank Rosenblatt, a neuro-biologist of Cornell, began work on the Perceptron. He was intrigued with the operation of the eye of a fly. Much of the processing which tells a fly to flee is done in its eye. The Perceptron, which resulted from this research, was built in hardware and is the oldest neural network still in use today. A single-layer perceptron was found to be useful in classifying a continuous-valued set of inputs into one of two classes. The perceptron computes a weighted sum of the inputs, subtracts a threshold, and passes one of two possible values out as the result. Unfortunately, the perceptron is limited and was proven as such during the "disillusioned years" in Marvin Minsky and Seymour Papert's 1969 book Perceptrons [8].

In 1959, Bernard Widrow and Marcian Hoff of Stanford developed models they called ADALINE and MADALINE. These models were named for their use of Multiple ADAptive LINear Elements. MADALINE was the first neural network to be applied to a real world problem. It is an adaptive filter, which eliminates echoes on phone lines. This neural network is still in commercial use [8].

Unfortunately, these earlier successes caused people to exaggerate the potential of neural networks, particularly in light of the limitation in the electronics then available. This excessive hype, which flowed out of the academic and technical worlds, infected the general literature of the time. Disappointment set in, as promises were unfilled. In addition, a fear set in as writers began to ponder what effect "thinking machines" would have on man. Asimov's series on robots revealed the effects on man's morals and values when machines where capable of doing all of mankind's work. Other writers created more sinister computers, such as HAL from the movie 2001[8].

These fears, combined with unfulfilled, outrageous claims, caused respected voices to critique the neural network research. The result was to halt much of the funding. This period of stunted growth lasted through 1981[8].

In 1982, several events caused a renewed interest. John Hopfield of Caltech presented a paper to the national Academy of Sciences. Hopfield's approach was not to simply model brains but to create useful devices. With clarity and mathematical analysis, he showed how such networks

could work and what they could do. Yet, Hopfield's biggest asset was his charisma. He was articulate, likeable, and a champion of a dormant technology [8].

At the same time, another event occurred. A conference was held in Kyoto, Japan. This conference was the US-Japan Joint Conference on Cooperative/Competitive Neural Networks. Japan subsequently announced their Fifth Generation effort. US periodicals picked up that story, generating a worry that the US could be left behind. Soon funding was flowing once again [8].

By 1985, the American Institute of Physics began what has become an annual meeting - Neural Networks for Computing. By 1987, the Institute of Electrical and Electronic Engineer's (IEEE) first International Conference on Neural Networks drew more than 1,800 attendees [8].

By 1989 at the Neural Networks for Defense meeting Bernard Widrow told his audience that they were engaged in World War IV, "World War III never happened," where the battlefields are world trade and manufacturing. The 1990 US Department of Defense Small Business Innovation Research Program named 16 topics, which specifically targeted neural networks with an additional 13 mentioning the possible use of neural networks [8].

Today, neural networks discussions are occurring everywhere. Their promise seems very bright, as nature itself is the proof that this kind of thing works. Yet, its future, indeed the very key to the whole technology, lies in hardware development. Currently most neural network development is simply proving that the principal works. This research is developing neural networks that, due to processing limitations, take weeks to learn. To take these prototypes out of the lab and put them into use requires specialized chips. Companies are working on three types of neuro chips - digital, analog, and optical. Some companies are working on creating a "silicon compiler" to generate a neural network Application Specific Integrated Circuit (ASIC). These ASICs and neuron-like digital chips appear to be the wave of the near future. Ultimately, optical chips look very promising. Yet, it may be years before optical chips see the light of day in commercial applications [8].

2.2. Neuron and the neural network

Figure III.1 shows a diagram with the general structure of an artificial neuron.

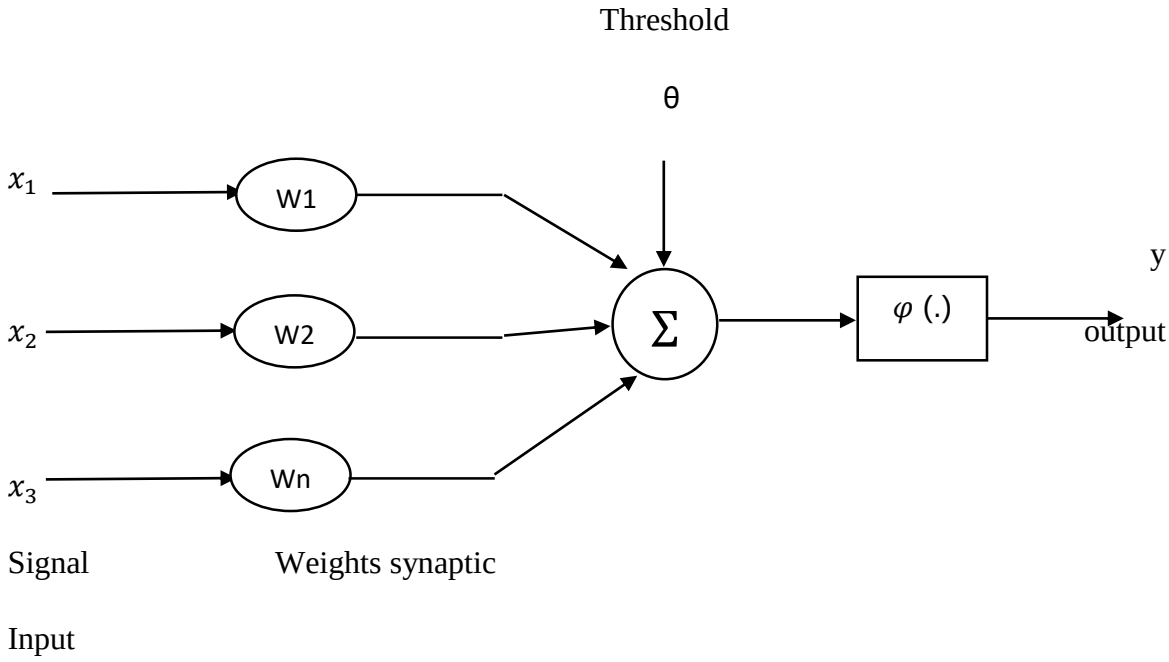


Figure III. 1 General structure of an artificial neuron

An artificial neuron is considered as an elementary element of information processing. It receives inputs and produces a result at the output [23].

$$u = \sum_{j}^{n} w_j x_j + \theta = W'X + \theta \qquad (III.1)$$

$$y = \varphi(u) \qquad (III.2)$$

$x_1 , x_2 \ldots . x_n$; are the external inputs. y is the output. $w_1 , w_2 \ldots . w_n$ Are the weights associated with each connection. x Is the input vector, $w'$is the weight vector, θ is called the bias.

The function φ is called the activation function, it is a nonlinear function [23].

Different activation functions can be used, among which we can mention: sign function, sigmoid, hyperbolic tangent, Gaussian .and the choice of a type of function depends on the application.

Artificial neural networks are combinations of elementary functions called formal neurons, or simply neurons associated in layers and operating in parallel. Each elementary processor computes a unique output based on the information it receives. Any hierarchical structure of networks is obviously a network [23].

2.3. Properties of neural networks

Artificial neural networks have a fundamental property which justify the growing interest in them and that they are capable of intervening in very diverse domains, and which distinguishes them from classical data processing techniques.

- Neural networks are universal approximates: This property can be stated as follows: Any sufficiently regular bounded function can be approximated uniformly, with good accuracy, in a finite domain of the space of its variables, by a neural network that has a finite number of hidden neurons, all having the same activation function and a linear output neuron [24]-[26].

-Parsimony: When modeling a process from its data, we always try to obtain the most

Always try to obtain the most satisfactory results possible with a minimum number of parameters. We say that we are looking for the most parsimonious approximation. To obtain a nonlinear model of a given accuracy, an RN needs fewer adjustable parameters than conventional regression methods (e.g. polynomial regression). However, the number of data needed to fit the model is directly related to the number of its parameters [25]-[27].

2.4. Activation functions

The activation function is a function that defines the internal state of the neuron according to its input. It can be a linear, non-linear, continuous or discontinuous function [29].

2.4.1. The binary activation function

The binary function is a linear function whose output is limited to two values.

It is equal to "1" for positive sign values and "0" for negative sign values [29]

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{si } \boldsymbol{x} > \boldsymbol{0} \\ 0 & \text{si } \boldsymbol{x} < \boldsymbol{0} \end{cases} \qquad (III.3)$$

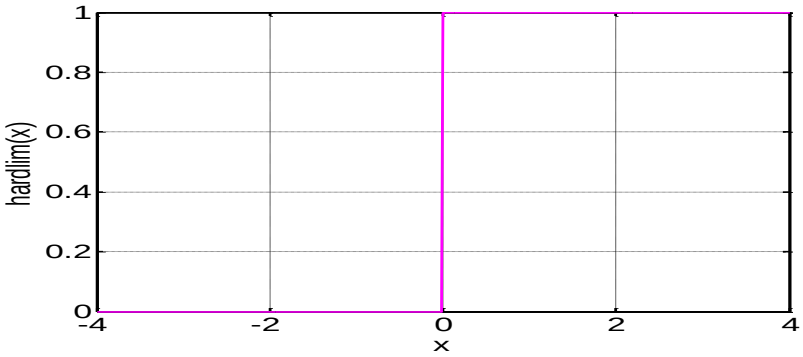


Figure III. 2. Binary activation function

2.4.2. The sign activation function

The sign function is a linear function whose output value is equal to "1" for positive sign values and "-1" for negative sign values [29].

$$f(x) = \begin{cases} 1 & \text{si} \quad x > 0 \\ -1 & \text{si} \quad x < 0 \end{cases} \qquad (III.4)$$

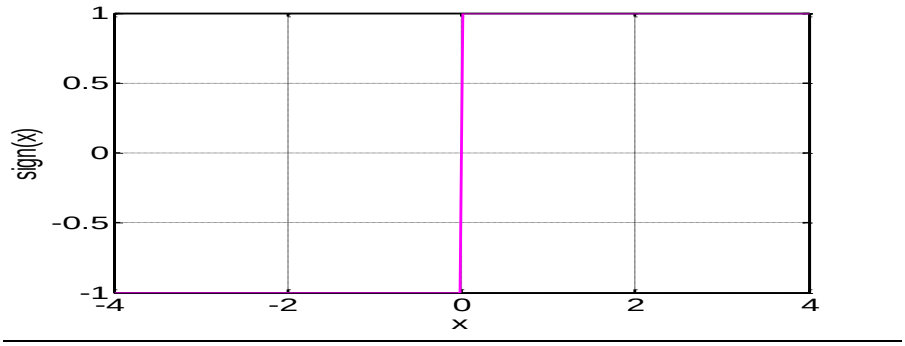


Figure III. 3. Sign activation function

2.4.3. The linear activation function

The linear activation function is widely used in the output layer of neural networks. Its mathematical expression is given as follows [29]**:**

$$f(x) = x \qquad (III.5)$$

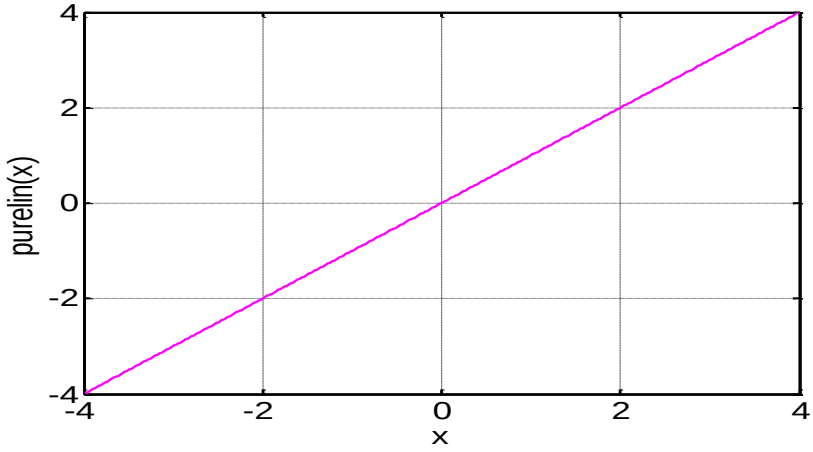


Figure III. 4. Linear activation function

2.4.4. The linear activation function with saturation

The mathematical expression of the linear activation function with saturation is given as follows [29]:

$$f(x) = \begin{cases} 1 & \text{si } x \geq a \\ \frac{1}{a} & \text{si } -a < x < a \\ -1 & \text{si } x \leq 0 \end{cases} \qquad (III.6)$$

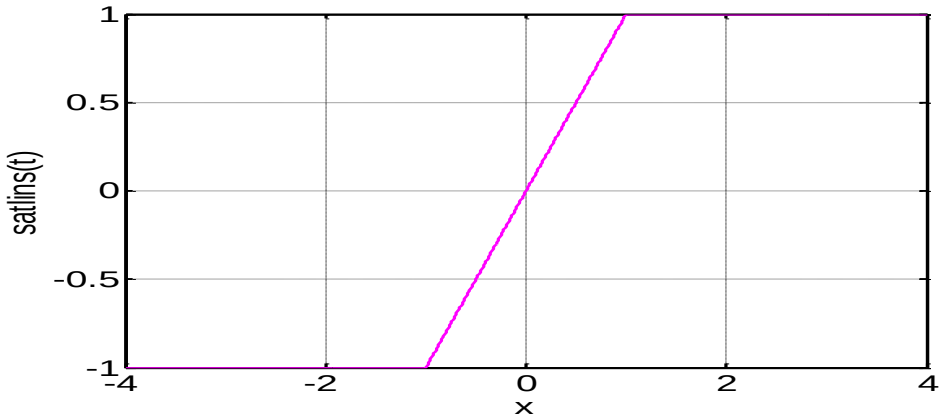


Figure III. 5. Linear activation function with saturation

2.4.5. The sigmoid activation function

The sigmoid function is a non-linear function whose value varies between "1" and "0". Its mathematical model is written as [29]:

$$f(x) = \frac{1}{1 + e^{-x}} \qquad (III.7)$$

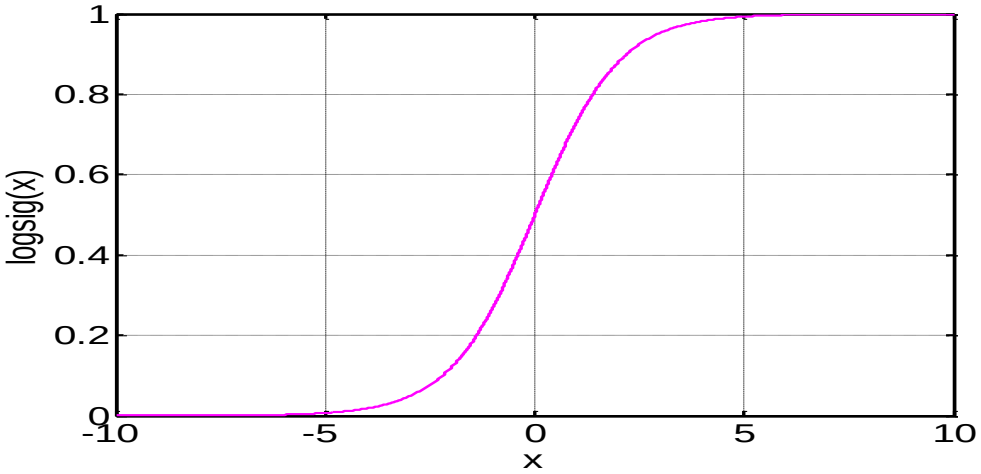


Figure III. 6. Sigmoid activation function

2.4.6. The hyperbolic tangent activation function

The hyperbolic tangent function is also a non-linear function whose value varies between "+1" and "-1". Its mathematical expression is given by [29]:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \qquad (III.8)$$

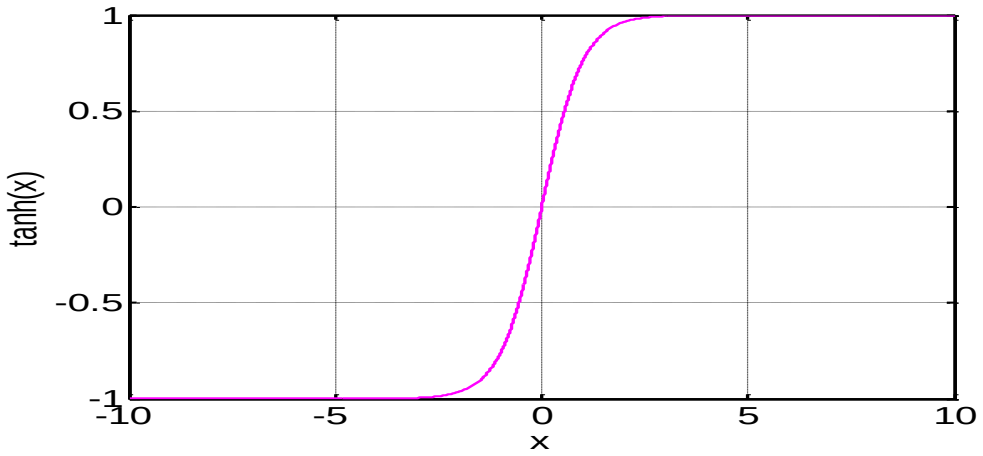


Figure III. 7. Hyperbolic tangent activation function

2.5. Learning

The objective of learning is to provide a method for the network to adjust these parameters when presented with new to be processed. Some rescaux differentiate between a training phase and an operating phase (the case of the Perceptron), or not (the case of the rdscau ART). We usually distinguish three learning paradigms: supervised, unsupervised and hybrid [30].

2.5.1 Supervised learning

In this case, the network is given the data to be processed and the expected response. The network performs an evaluation of the data, and then compares the value obtained with the desired value; it will then modify its internal parameters to minimize the error observed. Reinforcement learning (also called Reward and Penalty learning, i.e. ARP) is a variant of the supervised approach, in this framework the network is provided with a critique that qualifies the calculated response [30].

2.5.2. Unsupervised learning

In this paradigm, no information (besides the data to be learned) is provided to the system. The system has to discover the underlying structure of the data in order to organize them into clusters [30].

2.3.5. Hybrid learning: This approach combines numerical methods (neural networks, genetic algorithms) and symbolic methods. Some authors use the term hybrid learning to refer to a

supervised, unsupervised coupling; in this case, it is a network that puts in parallel or in series, a network trained in supervised mode and another in unsupervised mode [30].

3. Neural Adaptive PID

The classical PID control is one of the most utilized techniques in the standard process control due to its simple structure and high reliability. This PID controller, however, could not easily adapt to the problem of time-variation and the non-linearity of the industry processes, because the parameters has been already fixed with a derived constant value [32].

With the application of intelligent control and neural networks have also become hot spot of control field. Neurons network is based on result of the brain physiological research, it simulates some mechanism and mechanism of brain, and it is the network of topological structure, which is established with artificial, it processes information by responding on continuous or staccato input state, the essence of neural network is nonlinear system, multilayer neural network has the ability of any function, it brought the unified model to the description of the nonlinear system, neuron network has very strong comprehensive ability, it can well solve the input information redundancy, it also can properly coordinate the conflicting input information, can deal with the system information which is difficult to describe by the model or rules. The single neuron PID control is mainly studied, the experimental results show that the method has stronger robustness than common PID control method [31].

Artificial neural network (ANN) has been reported to be used as a control system for a time-dependent nonlinear system. The ANN-based controller system has been usually applied for state feedback controller design, nonlinear system control, nonlinear dynamical system identification, and control synthesis. The ANN is a massive parallel-distributed processor made up of a simple processing neuron for memorizing the knowledge and making it available after training the networks weights through a determined learning algorithm. A simple but powerful neural network is a multi-layer perceptron (MLP) with one hidden layer, trained by using a back-propagation learning mechanism for updating the artificial neural networks parameters. However, the learning mechanism of the ANN needs higher computational cost, which hampering the used of this technique as a real-time control system. In order to improve the performances of ANN-based control system, especially in reducing the networks learning time, a single neuron adaptive PID control, which is the hybrid of neuron learning mechanism with a classical PID control technique,

has been proposed. Beside the structure of a single neuron, adaptive PID controller is very simple, the controller can learn and be adjusted on line during the working process, and hence it can be utilized in a complex environment variation of a real plant. Various neural networks weight update algorithm are proposed for a single neuron adaptive PID (SNA-PID) controller system, such as Hebb learning rule, quadratic object function, and auto gain regulation. However, even though the SNA-PID controller system has a self-adaptation and robustness capability, the system performance parameters, such as rise time, settling time, and overshoot, is still open for improvement. As the system performance parameters are depend on the defined quadratic parameter function to be minimized, in this paper, a new quadratic performance function to be minimized is proposed. This new quadratic performance function is accomplished by using an additional error function that calculated from the difference between the input control signal and the actual control signal as an inverse form from the actual output of the plant [32].

3.1. The model of single neuron

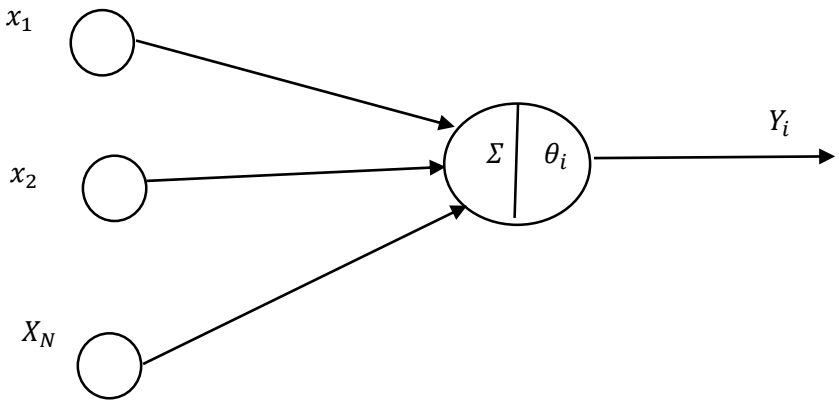


Figure III. 8. Model of single neuron

As the basic unit of the neural network, the structure of single neural is simple and easy to calculate, after the human brain neurons is simple abstracted, the artificial neural called as McCulloch-Pitts model is get, which is shown as Fig 34. , among them, is information received by neurons, is the connection strength, which is called as right. The role of the input signal is combined by some calculation, which is called as net input and is signed. According to the different operation way, there are many methods of expressing net input, the most simple is linear weighted summation [31].

The role caused the state change of neurons, assnd the output $y_i$of the neuron i is the function g (•) of current state, so, the mathematical expressions for the model are Eq.III.9 and Eq.III.10.[31]

$$net_i = \sum_{j=1}^{N} \omega_{ij} x_j - \theta_i \qquad (III.9)$$

$$y_i = g(\text{ net }_i) \qquad (III.10)$$

Including, is the threshold of neuron *i*.

3.1.1. Single neuron adaptive PID controller

3.1.1.1. The normal incremental digital PID controller

The algorithms of normal incremental digital PID can be expressed as follows, [33]

$$u(k) = u(k-1) + k_p e(k) + k_i \Delta e(k) + k_d \Delta^2 e(k)$$

Where is sampling number, k=0,1,2… $u(k)$, is the output of PID controller, k_p is scale parameter, k_i is integration parameter, k_d is differential parameter, $e(k)$ is the error between the expected output and the actual output, , $\Delta e(k)$,$\Delta^2 e(k)$ are the first difference, the second difference of *e*(*k*) .[33]

However, the PID controller with fixed parameters cannot meet the requirements of high-performance control when the operation condition changes. In this section, a combination of a single neuron and an incremental PID controller, called the SNA-PID controller, is presented to overcome the limitation of parameter tuning in our TWSBR.

3.1.2. Structure of single neuron adaptive PID controller

The single neutron adaptive PID can realize the self-study and adaptive function by adjusting the weigh coefficients and the tune role of the weigh coefficient is a select study regular, which is easy for field debugging. The method can improve the dynamic character of nonlinear time-varied object, and guarantees the control system works in the best status, so it is better than PID controller. The sketch graph of the single neutron adaptive PID controller is shown as Figure III.9 [34].

Where is the desired value or set point, $y_r(k)$ is the desired value after filtered, α (0<α<1) is a coefficient and it can adjust robust performance of the system. The more α is large, the more the system response is slow and robustness is good; the more α is small, the more the system response is fast and robustness is bad, usually α=0.2.

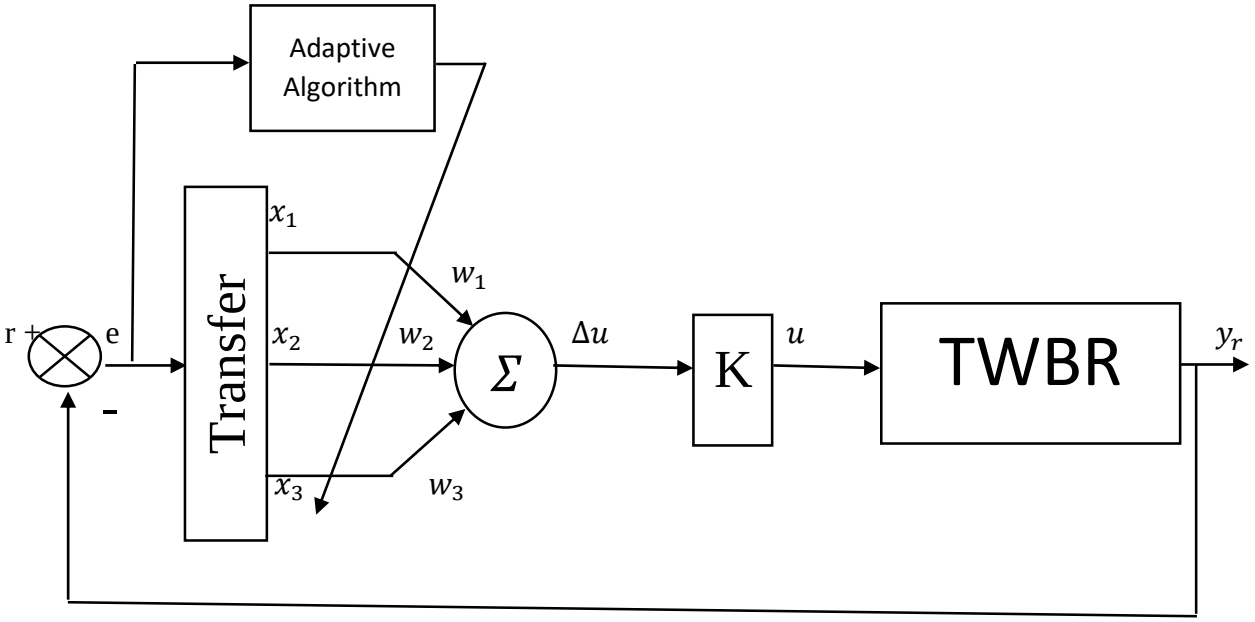


Figure III. 9. Sketch graph of the single neutron adaptive PID controller

Where is the desired value or set point, $y_r(k)$ is the desired value after filtered, α (0<α<1) is a coefficient and it can adjust robust performance of the system. The more α is large, the more the system response is slow and robustness is good; the more α is small, the more the system response is fast and robustness is bad, usually α=0.2.

The input of converter is the errors, and the output is the state variable x_1, x_2, x_3 they written as following [34]:

$$x_1(k) = e(k) - e(k-1) \qquad (III.11)$$

$$x_2(k) = e(k) \qquad (III.12)$$

$$x_3(k) = e(k) - e(k-1) + e(k-2) \qquad (III.13)$$

With the error at time k is e (k) = r (k) - y (k), the error at time k-1 is e (k -1), and for k-2 is e (k -2), respectively. The outputs of the state convertor are then inputted to the single neuron acting like a PID, with the weights matrix of the neuron is defined as W = $(w_1, w_2, w_3)^T$. The output of the neuron [32]

The control signal $u(K)$ at Kth sampling time is then obtained through: [35]

$$u(k) = u(k-1) + \Delta u(k) \qquad (III.14)$$

With:

$$\Delta u(k) = K[w_1(k)x_1(k) + w_2(k)x_2(k) + w_3(k)x_3(k)] \quad (III.15)$$

K is the proportional coefficient of the single neuron and K > 0. w_1 , w_2 and w_3 are respectively the weight values ofx_1, x_2 and x_3.[35]

The SNA-PID controller aims at minimizing the error between the reference value and the Measurement of the closed-loop system. Considering the well-known mean square error (MSE), The cost function $J(K)$ at sampling time k+1 is defined as: [35]

$$J(k+1) = \frac{1}{2}(r(k+1) - y(k+1))^2 = \frac{1}{2}(e(k+1))^2 \quad (III.16)$$

If the method of gradient descent is adopted here to minimize the cost function $J(K)$, then the gradient descent can be formulated as: [35]

$$w_j(k) = w_j(k-1) - \eta_j \nabla w_j J(k) \quad (III.17)$$

Where the $\nabla w_j J(k)$ is the gradient vector of $J(k)$, It implies the direction of weight updating is along the negative gradient direction. η_j Is the learning rate of the hidden layer. Thus, applying the chain rule, the rule of updating weight is written as: [35]

$$\begin{aligned} \Delta w_j(k) &= -\eta_j \nabla w_j J(k) = -\eta_j \frac{\partial J(k)}{\partial w_j(k)} = -\eta_j \frac{\partial J(k)}{\partial y(k)} \frac{\partial y(k)}{\partial u(k)} \frac{\partial u(k)}{\partial w_j(k)} \\ &= \eta_j K e(k) x_j(k) \frac{\partial y(k)}{\partial u(k)} \end{aligned} \quad (III.18)$$

For simplicity, Eq (III.18) can be rewritten as: [35]

$$\Delta w_j(k) = \eta_j \, \mathrm{Ke}(k) \, x_1(k)\beta(k) = \eta'_j e(k) x_j(k) \beta(k) \quad (III.19)$$

Where:

$$\frac{\partial y(k)}{\partial u(k)} = \beta(k) \quad (III.20)$$

In our work, a model-free solution is proposed by replacing $\beta(k)$with input (k) .

At the low speed, $\frac{\partial y(k)}{\partial u(k)}$ the is relatively small. With the growth of speed, $\frac{\partial y(k)}{\partial u(k)}$ gradually increases. Such variation trend completely coincides with the change of $u(k)$ [35].

$$w_j(k+1) = w_j(k) + \eta'_j e(k) x_j(k) \beta(k) \qquad (III.21)$$

In practice, in order to ensure the convergence and robustness of SNA-PID, the weights are required to be normalized before calculation of the control variable. Therefore, Eq (III.21) can be rewritten as [35] :

$$\Delta u(k) = K \sum_{i=1}^{3} w'_i(k) x_i(k) \qquad (III.22)$$

And

$$w'_i(k) = w_i(k) / \sum_{i=1}^{3} |w_i(k)| \text{(III. 23)}$$

With

$$\begin{aligned} w_1(k+1) &= w_1(k) + \eta'_1 e(k) u(k) x_1(k) \\ w_2(k+1) &= w_2(k) + \eta'_2 e(k) u(k) x_2(k) \\ w_3(k+1) &= w_3(k) + \eta'_3 e(k) u(k) x_3(k) \end{aligned} \qquad (III.24)$$

$\eta'_1, \eta'_2, \eta'_3$ Is respectively proportion, integral, differential learning rate; the different learning rate is adopted to adjust respective weighting coefficients.

3.2 Simulation

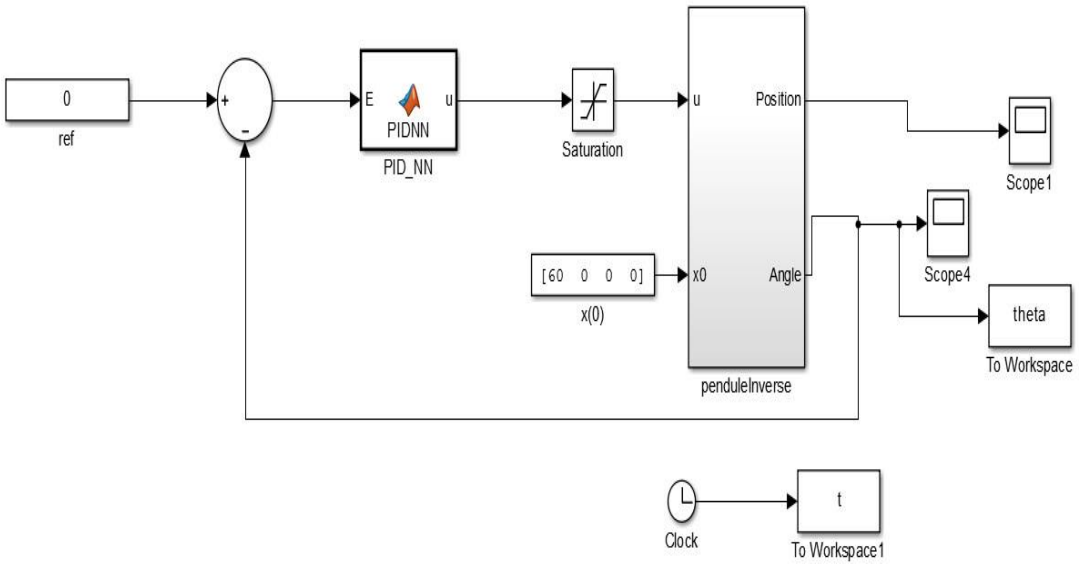


Figure III. 10. Simulink model of adaptive neural PID controller

Results:

1- Angle:

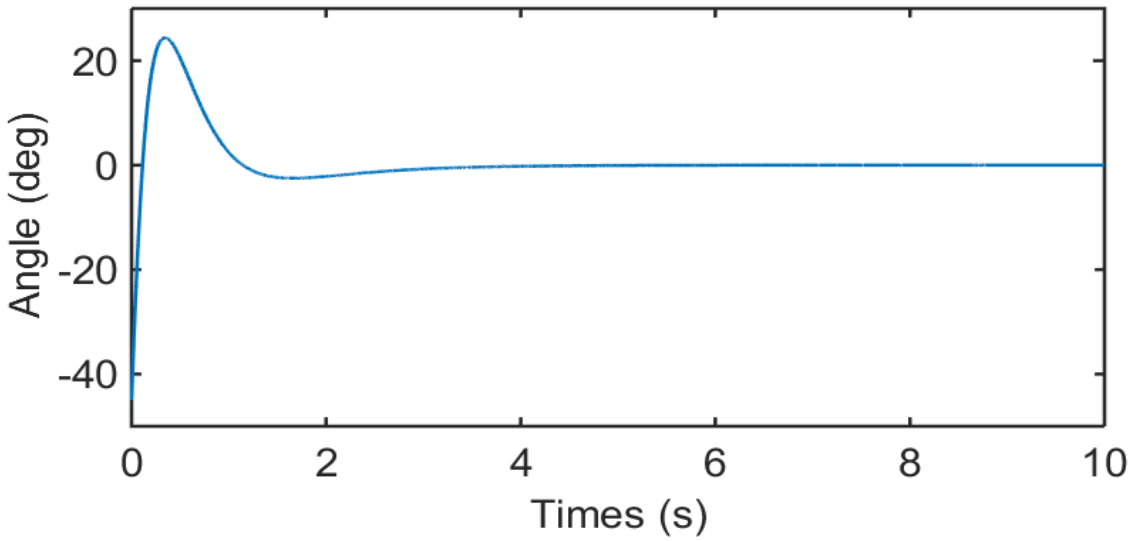


Figure III. 11. Response of angle with initial values $x0 = [0, 0, -45, 0]$

The system make a perfect overshoot, as well as rise time and steady-state and the settling time it is very suitable, after 2.5 seconds the system reach the instable equilibrium point that we consider it previously to 0° (upright position).

2- Control input :

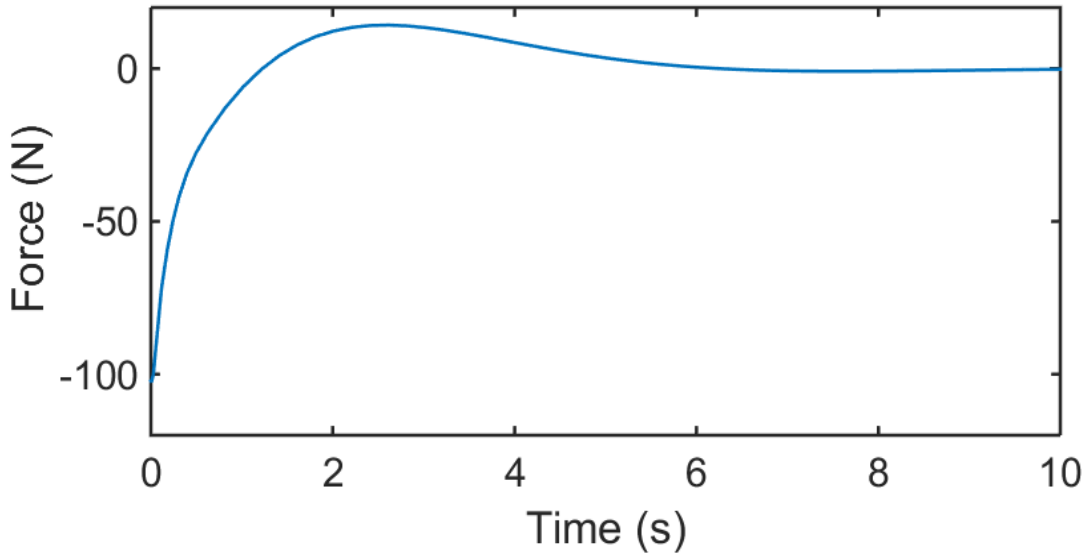


Figure III. 12. Response of input control (force)

4. Comparison

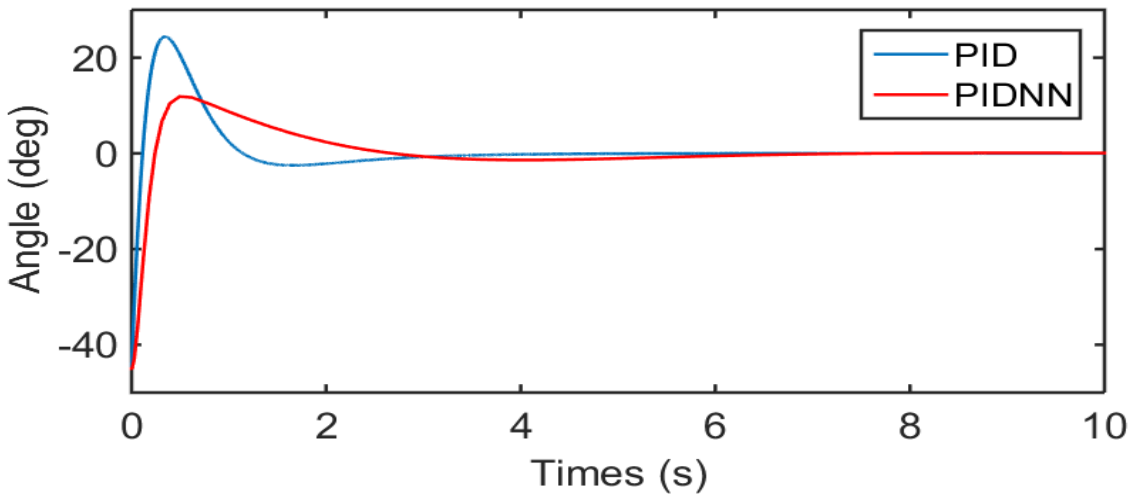


Figure III. 13. Comparison between PID and NNPID

After comparing the classical PID controller and the Adaptive Neural PID controller, we notice that the adaptive neural PID has batter performances (small overshoot, fast rise time, small steady-state error, small settling time) than the classical PID.

5. Conclusion

When examining the system, whether by simulation or by implementation, we noticed that the adaptive neural PID controller is accurate with the performances of the system “delay time and rise time and overshoot” and it gave us great results when we applied it to our non-linear system, both in theory and in practical, for this reason we can canclude that the adaptive neural PID controller is much better than the classical PID controller to deal with complex nonlinear systems.

Chapter 4: Real implementation for the Two Wheels Self-balancing Robot

1. Introduction

Two –wheeled or self-balancing robot is an unstable dynamic system unlike other four-wheeled stale robots that are in equilibrium state. By unstable, here, we mean that the robot is free to fall ahead or backward direction without any application of force. Self-balancing means the robot balancing itself in an equilibrium state, 90 degrees upright position. This project works on the inverted pendulum concept. We are making use of Arduino Uno to build the self-balancing robot. We are using the inertial measurement unit MPU6050 for measuring the current tilt angle [39]. An Adaptive Neural PID controller will be able to control the pendulum angle. Software and hardware components have been used in making the proposed robot. Only one software is used which is Arduino IDE, and different hardware is used such as Arduino Nano as the brain, DC Motor to provide the motion, and its driver L293D, MPU-6050 to get the orientations.

2. Software

2.1. Arduino IDE

Arduino IDE is simple to use software yet the best one to code, compile and upload the code in Arduino microcontroller boards. The Arduino team develops it. One can code on other platforms too, such as visual code studio, but Arduino IDE is official and accepted worldwide. It is the first one to receive all the updates of the libraries and the boards. Many libraries are available which make it convenient to code in the Arduino IDE. A default window of the Arduino IDE is shown in Figure IV.1 [36].

Figure IV. 1. A default Arduino IDE window

3. Hardware

3.1 Arduino Nano

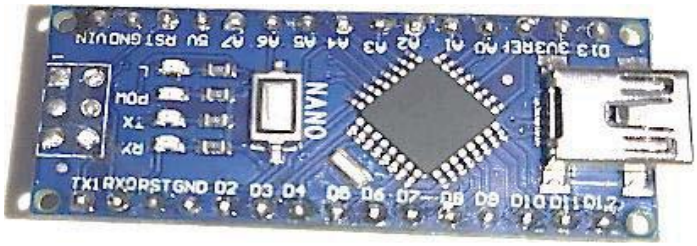


Figure IV. 2. Arduino Nano

The Arduino Nano is used in this paper as the brain of the robot. This microcontroller is the smallest member of the Arduino family. It is based on the ATmega328microcontroller, which is capable for several projects. It is a 30 pin breadboard-friendly board with 22 I/O pins. It can work as same as Arduino UNO in almost all projects, but due to its small size, it is the perfect choice to work with. The main difference between UNO and Nano is very minute. Nano has a different architecture than UNO and has 8 analog pins, whereas UNO has only 6 analog pins. It works with a Mini-B USB cable instead of a power jack. An Arduino Nano is shown in Figure IV.2. Pin description of Arduino Nano is provided in [36].

Table 4. Pin description for Arduino Nano

Type	Pins	Description
Power	Vin, 3.3 V, 5 V, GND	Vin is used for giving supply to the Arduino board, which can vary from 6 to 12 V. 5 V and 3.3 V give output voltages as Stated. GND refers to the ground pin.
Reset	RST	This is reset pin in Arduino which resets The Arduino.
Analog	A0-A7	These are analog input pins. Arduino has an inbuilt ADC (analog to digital

		converter) which helps to measure the Input voltage from 0 to 5V. Arduino doesn't have a DAC (digital to analog converter) but it can do PWM (pulse width modulation) which are locate at in digital pins and can be used to give some of the functions of an analog output as Well. A4(SDA), A5(SCL): These pins are used for TWI (Two-Wire Interface) Communication. (SCL: serial clock line, SDA: serial data)
Digital Pins	D0-D13	These are digital input-output pins These pins give logic high (5V) or logic low (0V). They can also be used as input logic High and logic low. 3,5,6,9,11: These are some 8-bit PWM Output present on the board. 13: Pin 13 is for inbuilt LED
Serial	Rx, Tx	Rx-Receiver Tx-Transmitter These are used for serial communication.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication. SS, MOSI, MISO, and SCK stand for Slave Select, Master out Slave in, Master in Slave out,

		In addition, Serial clock.
AREF	AREF	To provide reference input voltage.

3.2 MPU-6050

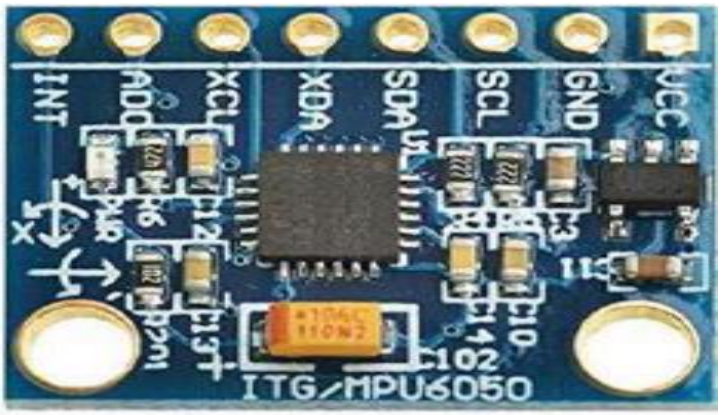


Figure IV. 3. MPU-6050

The MPU-6050 is a motion-tracking device. Here, this sensor gets the alignment of the bot, which is further processed to take the control actions. The MPU-6050, as shown in Fig. , constitutes a Micro-Electro-Mechanical Systems (MEMS) 3-axis accelerometer and a MEMS 3-axis gyroscope on a single chip. It also has a temperature sensor inbuilt on it. MPU-6050 delivers accurate results and can provide hundreds of measurements per second. It captures the x, y, and z orientation at the same time. It uses the *I2C*-bus to interact with the Arduino. Pin diagram and pin description of MPU-6050 is given in Figure IV.3 and Table 5, respectively [36].

Table 5. Pin description for MPU-6050

Type	Pins	Description
Power	VCC, GND	VCC is 5 V input. GND is for grounding.
Communication	SDA (Serial Data), SCL (Serial Clock)	SDA transfers data and SCL provides clock pulse For communication.
Auxiliary Communication	XDA (Auxiliary Serial Data), XCL (Auxiliary Serial Clock)	Can be used to interface other modules as Arduino have limited SDA and XCL ports.
Other	AD0, INT	AD0 (Slave address at 0th bit, i.e. this is 0th number

		bit of a 7-bit slave address of the module) is used when more than 1 MPU 6050 modules are used to communicate in Synchronous. INT is interrupt pin to indicate that data is Available for MCU to read.

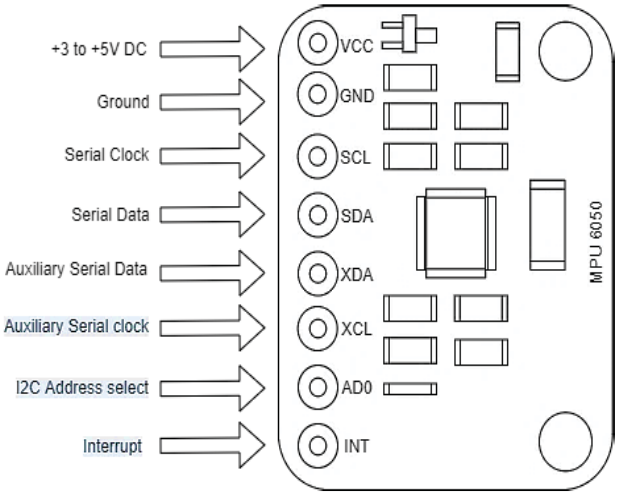


Figure IV. 4. Pin arrangement of MPU-6050

3.3. Motor driver IC L293D

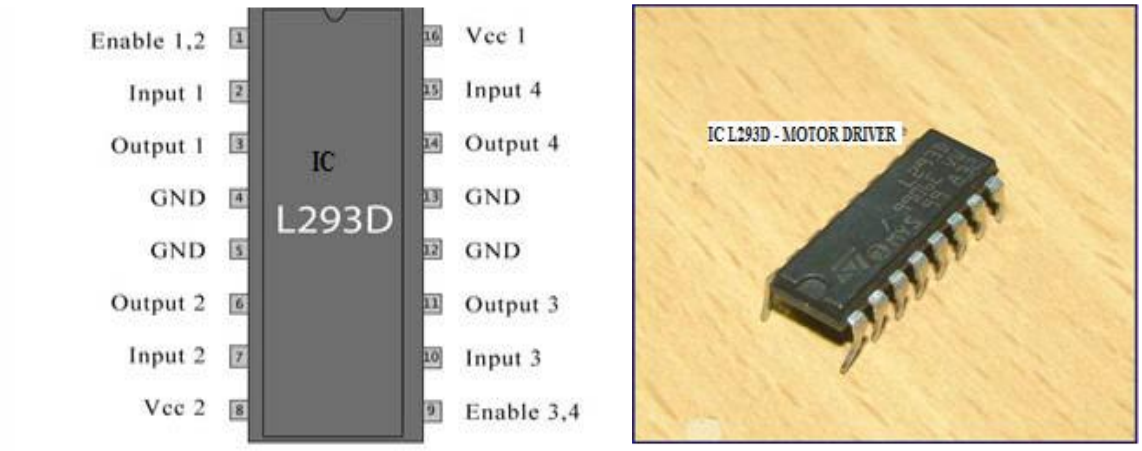


Figure IV. 5. Pin diagram along with description

A motor driver Integrated circuit (L293D) chip is designed to control and regulate motors .It is a dual H-bridge motor driver Integrated Circuit. They are generally used in mechanics and robotics. It acts as an interface between motor and Arduino microprocessor in the circuit. L293D, L293NE are most commonly used motor driver Integrated circuits from L293 series. L293D is designed to control up to maximum of two direct current motors simultaneously when they are integrated with Arduino Nano. It helps to regulate the flow of current before it finally reaches the motor. It becomes a necessity and need to use IC L293D due to different requirement of current and voltages by microprocessors (low) and 5V DC motor (high) as it acts as a moderator and balances the flow of current. It protects the circuit from overload current and provides protection against overload temperature. Current should not be directly supplied to the motor because it can damage the motor or even the microcontroller. It has an output capability and provides bidirectional current of 600 mA per channel. The maximum or peak current, which can flow through per channel as output, is 1.2 Amp. It has Enable facility and internal clamp diodes. Input voltage is up to 1.5V-36V, which is also high noise immunity (logical "0"). Various and un-similar PWM signals are received because a motor driver IC interfaces with the microcontroller. A motor driver IC is also responsible for achieving required outputs for the speed variation of the DC motor [37].

3.3.1. Controlling of dc motor with L293D

In order to have a complete control over DC motor, we have to control its speed and rotation direction. This can be achieved by combining these two techniques, and in our work we just need to control the rotation but we will talk about how to control the speed as well [38].

- PWM – For controlling speed
- H-Bridge – For controlling rotation direction

3.3.1.1 PWM – For controlling speed

The speed of a DC motor can be controlled by varying its input voltage. A common technique for doing this is to use PWM (Pulse Width Modulation) [38].

PWM is a technique where average value of the input voltage is adjusted by sending a series of ON-OFF pulses [38].

The average voltage is proportional to the width of the pulses known as **Duty Cycle** [38]**.**

The higher the duty cycle, the greater the average voltage being applied to the dc motor (High Speed) and the lower the duty cycle, the less the average voltage being applied to the dc motor (Low Speed) [38]. Below image illustrates PWM technique with various duty cycles and average voltages [38].

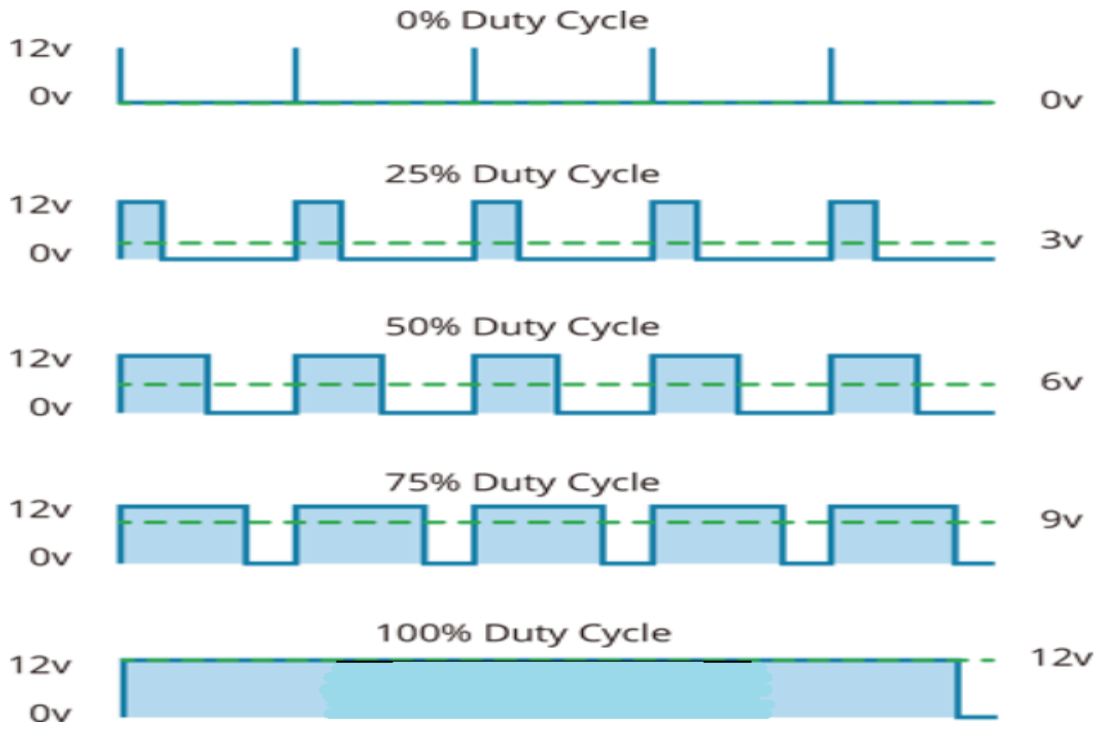


Figure IV. 6. Pulse Width Modulation (PWM) Technique

3.3.1.2. H-Bridge – For controlling rotation direction

It is composed with two H-bridge that are basic circuits. It is a simple circuit for regulating a rated motor with low current. L293D comprises of 16 pins which includes 4 (ground, input and output) pins and 2 (Enable and Voltage) pins Because of it the DC motors can operate in both reverse and forward motion. To rotate it in forward and reverse directions logic function ‘01’ and ‘00’are used respectively. 1 and 9 are two enable pins for two motors respectively and they should be of high value to start operating. The drivers are enabled in pairs bipolar stepping motors, loads in high positive power supply applications, relays and solenoids which are fabricated to run various inductive loads. TTL compatible inputs along with totem-pole circuit (present with pseudo-Darlington source and sink) which are also enabled in pairs. It is suitable for various motor

applications or solenoid with reversible drive. L293D IC first receives signals send by the microcontroller and then emanates the response signal to the motor. Input signal received from the input helps to switch the outputs accordingly [37].

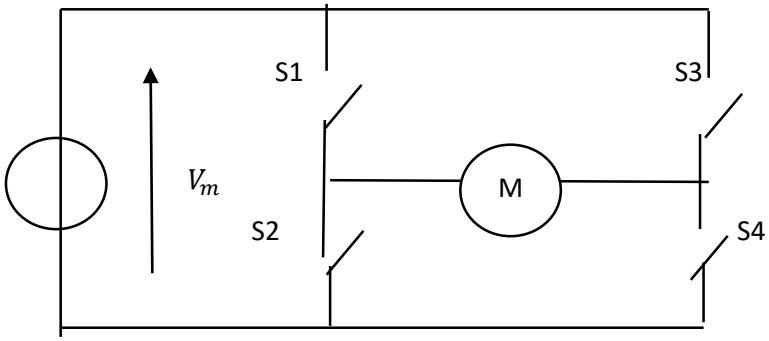


Figure IV. 7. Basic H-bridge circuit diagram

3.3.2. Control Pins

For each of the L293D's channels, there are two types of control pins, which allow us to control speed and spinning direction of the DC motors at the same time viz. Direction control pins & Speed control pins [38].

3.3.2.1 Direction control pins

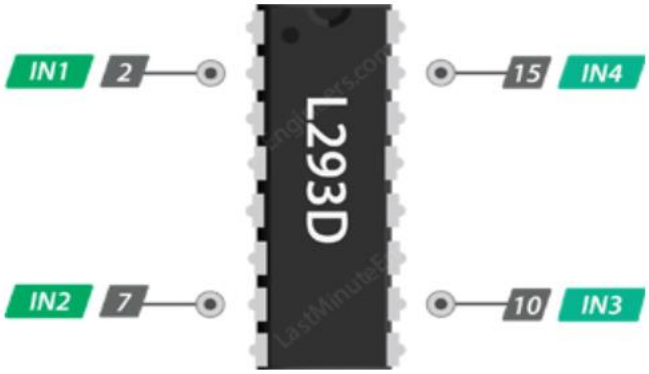


Figure IV. 8. Direction control pins

Using the direction control pins, we can control whether the motor spins forward or backward. These pins actually control the switches of the H-Bridge circuit inside L293D IC [38].

The IC has two direction control pins for each channel. The IN1, IN2 pins control the spinning direction of the motor A while IN3, IN4 control motor B [38].

The spinning direction of a motor can be controlled by applying either a logic HIGH (5 Volts) or logic LOW (Ground) to these pins. The below chart illustrates how this is done [38].

Table 6. The spinning direction

IN1	IN2	Spinning Direction
Low(0)	Low(0)	Motor OFF
High(1)	Low(0)	Forward
Low(0)	High(1)	Backward
High(1)	High(1)	Motor OFF

3.4. Direct Current motors

Figure IV. 9. DC motors

A DC (Direct Current) motor is based on a fact that similar poles (magnetic poles) repel and dissimilar poles attract each other. An electromagnetic field is generated in a coil of wire when a current passes through it and is focused at the center of the coil. When the current changes its direction or intensity or switching action on and off) in the coil, the magnetic field can be changed be reversed by 180 degrees or can simply generate switching magnetic field. In a Dc motor, a stator has a stationary magnets and armature has windings wring around the insulated stacked around iron pole commonly known as stack teeth with ends finishing at the commutator. Armature consists of bearings, which are mounted at the middle of the motor and connections of commutator. The winding is winded around armature and is known as armature winding which uses conductor (single or parallel) wires, which are wrapped around stack teeth. EMF's (Electromagnetic fields)

strength depends on several factors such as current in the coil, size of the coil and the material ringed around the coil. Direction of EMF depends upon number of turns and sequence of turns in a coil. By removing and injecting the coil inside and A greater control over the DC motor can be established by designing the Dc motor in such a way so that the magnetic fields generated by the stator fields using electromagnets. Forced air can be used for cooling the DC motors at high power usage level. A DC motor has linear Torque –Speed relationship. Here, load and speed are inversely proportional to each other. It provides protection against overload current, locked rotor, RFI/EMI caused by PWM control, and also protects against instantaneous reversing and dynamic braking [37].

Figure IV. 10. DC motor contacted with the wheel

3.5. Bluetooth Module

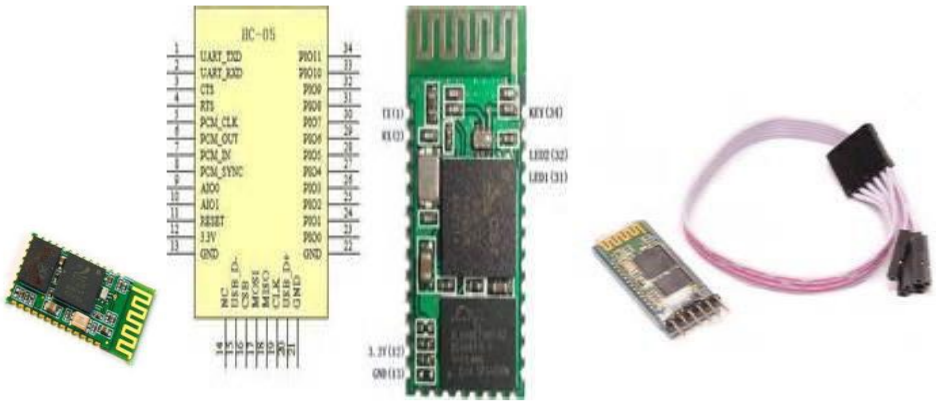


Figure IV. 11. Images of Bluetooth modulo HC-05

The various features of Bluetooth module are hardware and software features. The hardware features such as Low Power 1.8V Operation in the range of 1.8 to 3.6V Input/ output, typical sensitivity (-80dBm), PIO control , RF transmission power , programmable UART interface baud rate, edge connector and antenna integrated gives it edge above others. Various Software features are CTS and RTS system to control data streaming. It has a default baud (Data bits:8) rate and supports it (9600,19200,38400,57600,115200,230400,460800) with data control and no parity. It can generate PIO0 and PIO1 pulses which disconnect low and high signals and they can be connected to blue and red light emitting diodes separately. Red light starts to blink when master and slave are paired together and blue light blinks 1 time after every 2-second interval. It has auto-pairing with pin code "0000", auto-pairing device used for connection, auto-connect to connect to the last device on power and auto-reconnect to reconnect after 30 minutes when disconnected during beyond the range of connections used [37] .

3.6. Connecting wires and soldering

For our project of this two wheels self-balancing robot we use a PCB circuit.

- First, we will begin with the circuit on "EasyEDA".

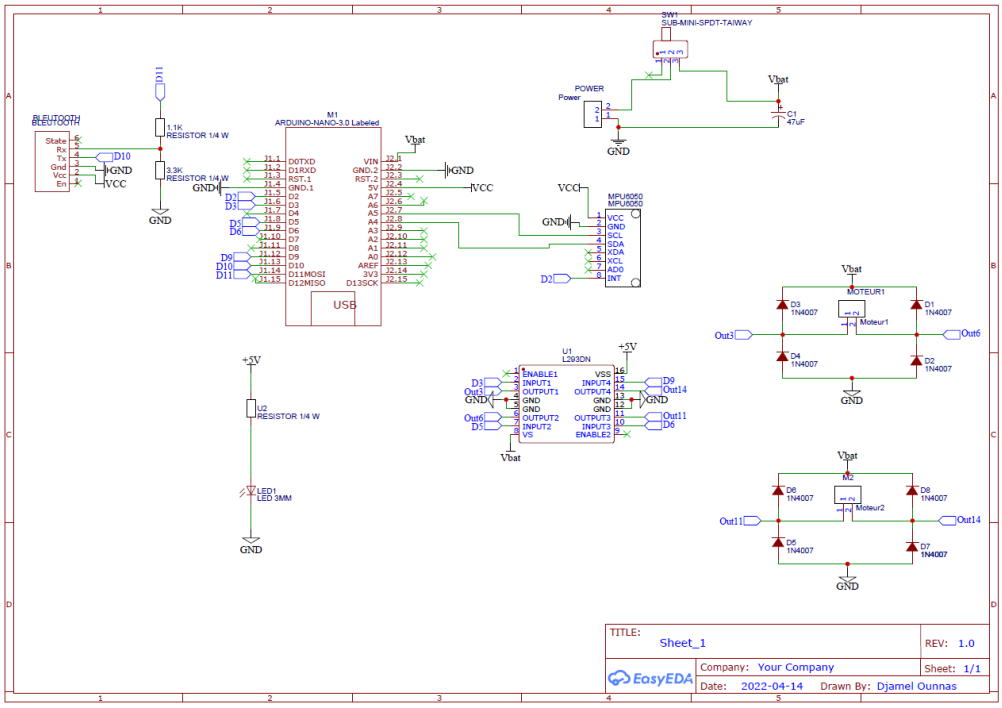


Figure IV. 12. Schematic on "EasyEDA"

- the PCB circuit it will appear as following**:**

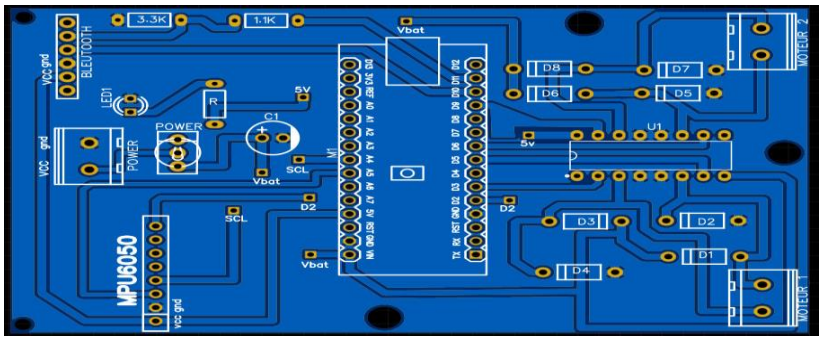


Figure IV. 13. PCB circuit on "EasyEDA"

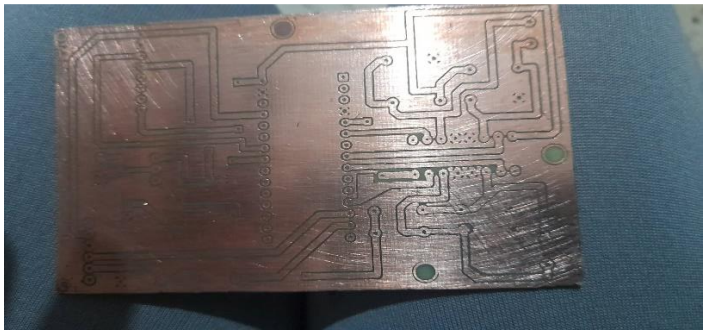


Figure IV. 14. Printed PCB

- After we print our circuit it looks as following:

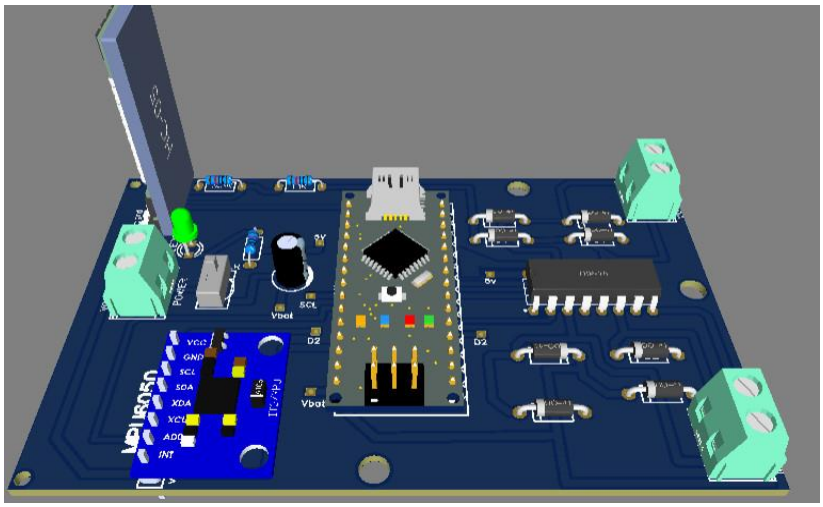


Figure IV. 15. 3D PCB circuit

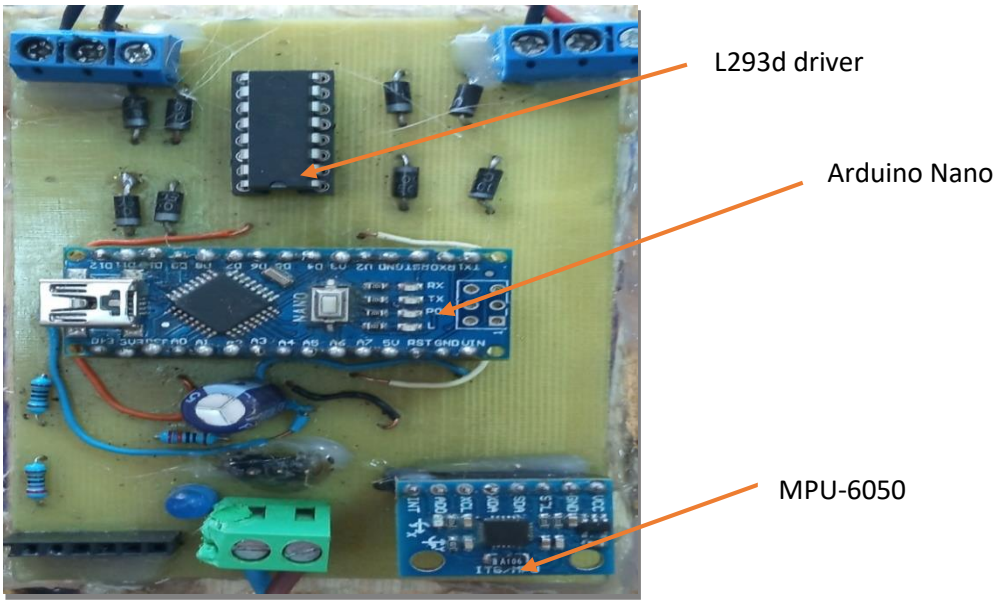


Figure IV. 16. Our electronic circuit

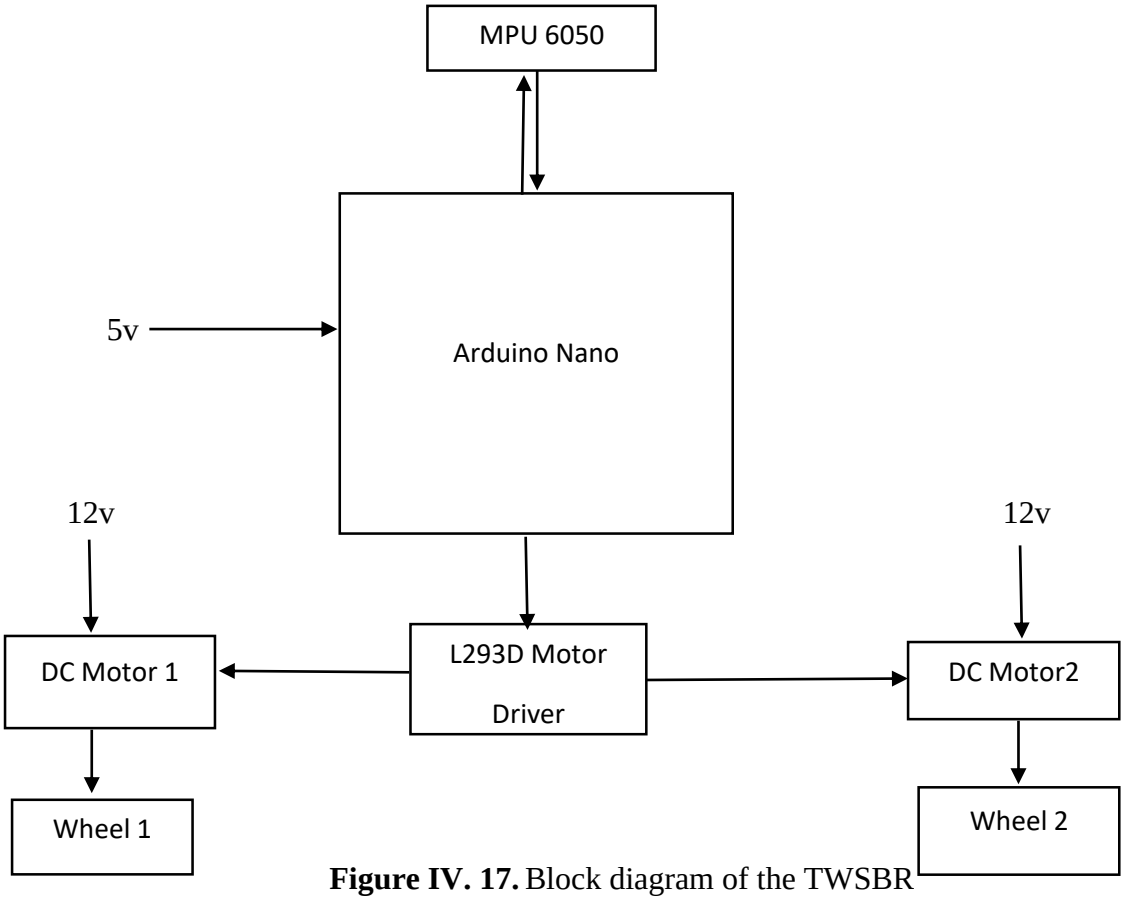


Figure IV. 17. Block diagram of the TWSBR

4. Block Diagram and Working

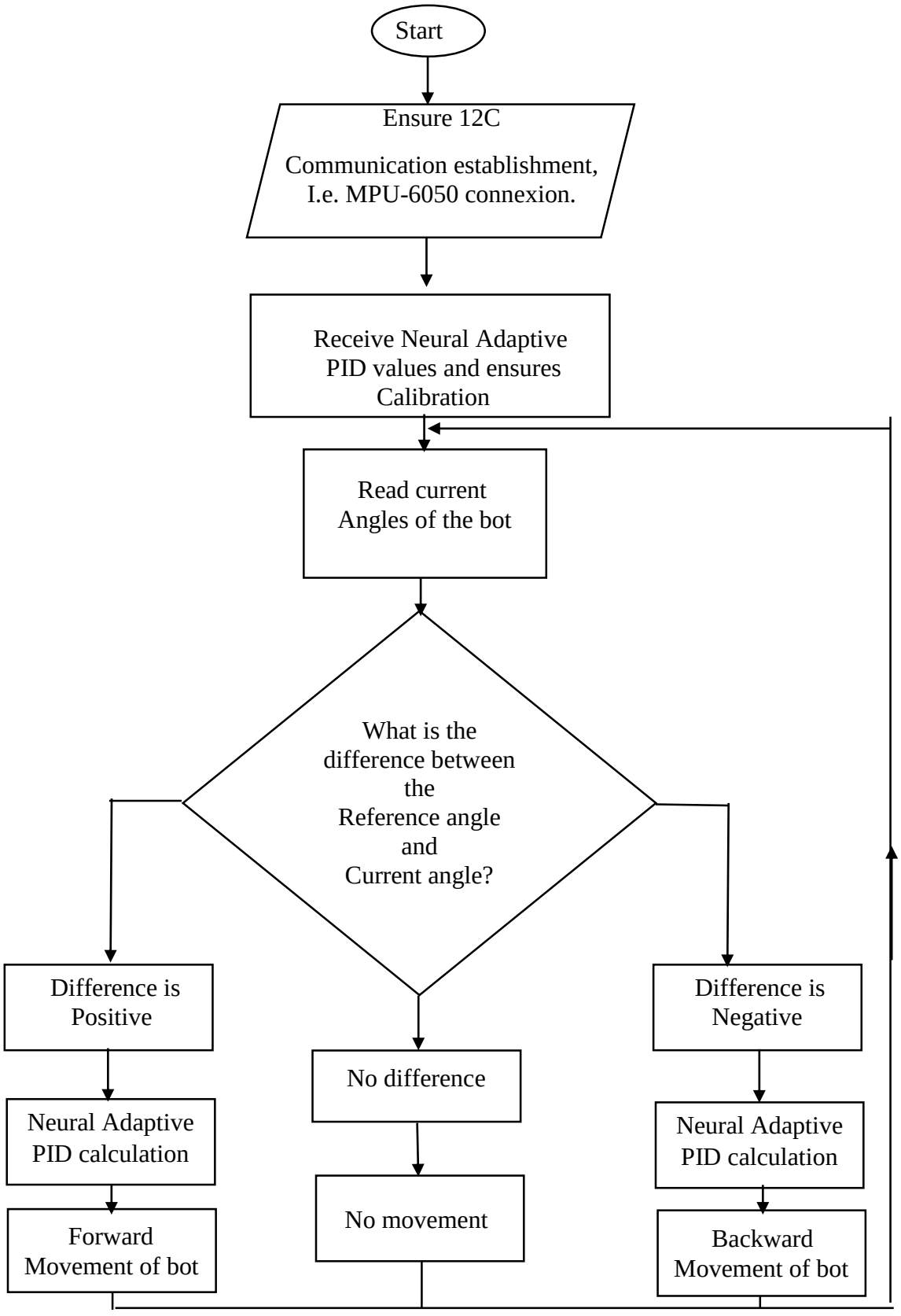


Figure IV. 18. *Neural Adaptive PID Algorithm*

To keep the robot adjusted, the engines must neutralize the robot falling. This activity requires criticism and revising components. The input component is the MPU6050 (gyroscope and accelerometer), which gives both speeding up and pivot in each of the three tomahawks. The Arduino utilizes this to know the redressing component is the engine and wheel mix. [39]
This is done by the implementation of the Neural Adaptive PID controller.

We need to check if the robot is inclining towards the front or towards the back utilizing the MPU6050 and after that if it's inclining towards the front we need to turn the wheels forward way and on the off chance that it is inclining towards the back we need to pivot the wheels in the invert bearing [39].

To know the present position of the robot we utilize the MPU6050, which is a 6-pivot accelerometer and gyrator sensor consolidated. So as to get a solid estimation of position from the sensor we have to utilize the estimation of both accelerometer and whirligig, in light of the fact that the qualities from accelerometer has commotion issues and the qualities from spinner will in general float with time. Therefore, we need to join both and get the estimation of yaw pitch and move of our robot, of which we will utilize, just the estimation of yaw [39].

For the robot to balance properly, it should have a good center of gravity. That can be achieved by properly placing all the components on the hardware of the robot on a suitable place [39].

5. TWSBR

Impalement the system with the components that we have mentioned earlier, such as printed PCB and two DC Motors connected with two wheels, L293D driver and Arduino nano programming by Arduino IDE.

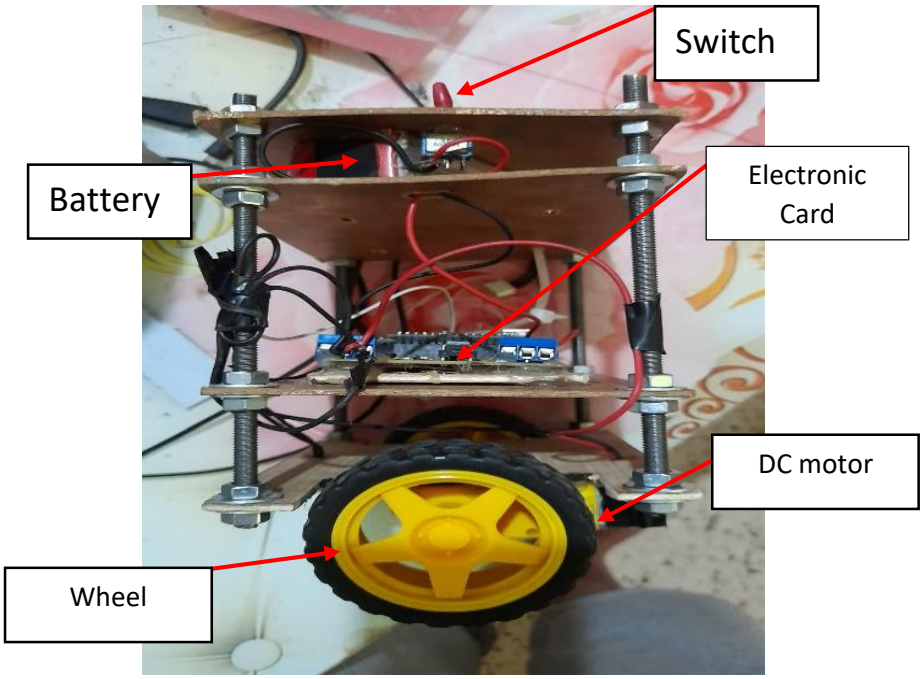


Figure IV. 19. Our TWSBR

6. Conclusion

After some testing and calibrating, the two wheels self-balancing robot balanced itself independently. It required appropriate values for the Adaptive Neural PID controller fed to the Arduino and accurate measurement from MPU-6050 sensor, the implementation helped us to compare the results obtained in the second chapter with the results obtained in the third chapter.

General conclusion

General conclusion

The adaptive neural PID controller gives impressive results in maintaining the balance of the inverted pendulum that we have formed in a TWSBR, in contrast to the other controllers, which we discussed in our work an example for widely used controller “PID controller”, which gave us acceptable results when applied in angle control of the instable equilibrium point. The results of the adaptive neural PID controller were more applicable, and after we implement the project with the controller we found a similar results to the simulation where the robot’s balance as good as we want and adaptive.

In the first chapter we have discussed the modeling of our inverted pendulum as well as the modelling of the actuators that we used in the project we have noticed some disadvantage with the stepper motor because of its weight, so we decided to work with the DC motor that we found it more suitable for this system.

In the second chapter we have discussed the control of our nonlinear system by a classical PID controller, the results showed that the PID controller give a good balancing and get the instable equilibrium point with good performances but the results can be better with another advanced controller.

In the third chapter, we have discussed the control of our nonlinear system by an Adaptive PID controller, we have gave a brief entry about the neural networks and we applied a single neuron for the controller as Single neuron adaptive PID controller. The results were perfect and the system get its instable equilibrium point with perfect performances, after the simulation the system was ready to be implement and see the practical results.

In the last chapter, we have implemented our nonlinear system with Arduino Nano, with printed PCB. The results were as we found it with the simulation as good as we want. The robot were balancing itself and get its instable equilibrium point that we consider it 0°, even with initial values (push or pull) the robot returned to the 0°.

REFERENCES

[1] Shekhawat, A. S., & Rohilla, Y. (2020). Design and control of two-wheeled self-balancing robot using Arduino. *2020 International Conference on Smart Electronics and Communication (ICOSEC)*. IEEE page 1025

[2] *Two-wheeled self-balancing robot - diva portal*. (n.d.). Retrieved March 5, 2022, from https://www.diva-portal.org/smash/get/diva2:916184/FULLTEXT01.pdf Page 5

[3] Paulescu, F.-C., Szeidert, I., & Filip, I. (n.d.). *Two-wheeled self-balancing robot*. IEEE Xplore. Retrieved March 5, 2022, page 000034 from https://ieeexplore.ieee.org/document/9465568

[4] Jamil, O., Jamil, M., Ayaz, Y., & Ahmad, K. (2014). Modeling, control of a two-wheeled self-balancing robot. *2014 International Conference on Robotics and Emerging Allied Technologies in Engineering (ICREATE)*. IEEE .page 194

[5] Karadeniz, A. M., Alkayyali, M., & Szemes, P. T. (n.d.). *Modelling and Simulation of Stepper Motor For Position Control Using LabVIEW- researchgate*. Retrieved March 5, 2022, from https://www.researchgate.net/publication/324261648

[6] Ashlin. (2021, May 31). *What is a feedback control system and what are its types?* Instrumentation and Control Engineering. Retrieved March 25, 2022, from https://automationforum.co/what-is-a-feedback-control-system-and-what-are-its-types/

[7] Libretexts. (2021, December 4). *11.1: Feedback Control*. Engineering LibreTexts. Retrieved March 25, 2022, from https://eng.libretexts.org/Bookshelves/Industrial_and_Systems_Engineering/Book%3A_Chemical_Process_Dynamics_and_Controls_(Woolf)/11%3A_Control_Architectures/11.01%3A_Feedback_control_What_is_it%3F_When_useful%3F_When_not%3F_Common_usage.

[8] *Master languages with lingualeo*. lingualeo.com. (n.d.). Retrieved March 25, 2022, from https://lingualeo.com/en/jungle/brief-history-of-pid-controller-229634

[9] Ali, A., Mohamedy, A., Salimz, A., El-Aminx, E. and Ahmed, O., 2021. Design and Implementation of Two-Wheeled Self-Balancing Robot Using PID Controller. *2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE),*.

[10] Administrator. (2017, December 24). *PID controller-working and tuning methods*. Electronics Hub. Retrieved March 26, 2022, from https://www.electronicshub.org/pid-controller-working-and-tuning-methods/#How_a_PID_Controller_Works

[11] Mudeng, V., Hassanah, B., Priyanto, Y. T., & Saputra, O. (2020). Design and simulation of two-wheeled balancing mobile robot with PID Controller. International Journal of Sustainable Transportation Technology, 3(1), 1219.

[12] Orlando .A .O (2010) PID Control Servo/regulation performance and robustness issues, UniversitatAuto`noma de Barcelona , spain.

[13] Kenzo. S , (2017). Modernization of a traction system for metro vehicles, Diploma thesis , Czech Technical University, Prague.

[14] Ahmed M. T. Ibraheem Al-Naib, (May 2019), DC Motors, Northern Technical University,Iraq, researchgate, https://www.researchgate.net/publication/332835517

[15] S. thangaprakash, & murugappan, M. (2022). Advances in electrical and Computer Technologies: Select Proceedings. SPRINGER VERLAG, SINGAPOR.

[16] Guest. (n.d.). Transfer function of DC Motor. Pdfcoffee.com. Retrieved June 3, 2022, from https://pdfcoffee.com/transfer-function-of-dc-motor-pdf-free.html

[17] Zhang, J. & Guo, L. (2019). Theory and design of PID controller for Nonlinear Uncertain Systems. IEEE Control Systems Letters, 3(3), 643–648. https://doi.org/10.1109/lcsys.2019.2915306

[18] Fadali, M. S., & Visioli, A. (2013). Introduction to digital control. Digital Control Engineering, 1–8. https://doi.org/10.1016/b978-0-12-394391-0.00001-0

[19] Ogata, K. (2010). Modern Control Engineering. Prentice Hall.

[20] PID tuning. PID Tuning - MATLAB & Simulink. (n.d.). Retrieved June 3, 2022, from https://www.mathworks.com/discovery/pid-tuning.html

[21] Karl J, Å. (2002). Control System Design. Lecture notes for ME 155A.

[22] Fallahi, M., & Azadi, S. (2009). Adaptive control of an inverted pendulum using adaptive PID neural network. 2009 International Conference on Signal Processing Systems. https://doi.org/10.1109/icsps.2009.110

[23] Merzouka. N. (2009). Etude des performances des réseaux de neurones dynamiques à représenter des systèmes réels : une approche dans l'espace d'état. [Master, Setif-1 University]

[24] J.GHOUILI . (2005) «Commande sans capteur d'une machine asynchrone avec estimation de la vitesse par les réseaux de neurones» Doctoral dissertation , Quebec University

[25] G.DREYFUS, J-M.MARTINEZ,M. SAMUELIDES, M.B.GORDON, F.BADRAN, S.THIRIA ET L.HERAULT (2002) «Réseaux de neurone, Méthodologies et Application » Paris

[26] B.GOSSELIN (1996) «Application de réseaux de neurones artificiels a la connaissance automatique de caractères manuscrits» Doctoral dissertation , Mons University.

[27] L.BAGHLI (1999) « Contribution à la commande de la machine asynchrone, l'utilisation de la logique floue, des réseaux de neurones et des algorithmes génétiques » Thèse de Doctorat, University of Nancy 1

[28] O. NERRAND, P. ROUSSEL-RAGOT, L. PERSONNAZ & G. DREYFUS « Neural Networks and Non-linear Adaptive Filtering: Unifying Concepts and New Algorithms.» Neural Computation, Vol. 5, pp 165-199, 1993

[29] Lemita.A. (2021). Méthodologie d'optimisation de la commande du procédé de traitement des eaux usées par boues activées basée sur les algorithmes évolutionnaires. [Doctoral dissertation, Ferhat Abbas Setif-1 University]

[30] Senouci. M , Baghdadi. H. A. RÉSEAUX DE NEURONES THÉORIE ET PRATIQUE, (Publication No 9961.0.0902.9)

[31] Zhang, Y. H., Zhao, D. A., & Zhang, J. S. (2012). Research on single neuron adaptive PID control. Applied Mechanics and Materials, 150, 174–177. https://doi.org/10.4028/www.scientific.net/amm.150.174

[32] Kusumoputro, B., & Rif'an, M. (2016). An improved single neuron adaptive PID controller system based on additional error of an inversed-control signal. Advanced Science Letters, 22(10). https://doi.org/10.1166/asl.2016.7002

[33] Wang, M., Cheng, G., & Kong, X. (2011). A single neuron self-adaptive PID controller of Brushless DC Motor. 2011 Third International Conference on Measuring Technology and Mechatronics Automation. https://doi.org/10.1109/icmtma.2011.70

[34] Lin, H., Changlin, M., & Feng, L. (2007). Study of adaptive PID controller based on single neuron and genetic optimization. 2007 8th International Conference on Electronic Measurement and Instruments. https://doi.org/10.1109/icemi.2007.4350432

[35] Tang, W., Wang, L., Gu, J., & Gu, Y. (2020). Single Neural Adaptive PID control for small UAV Micro-Turbojet engine. Sensors, 20(2), 345. https://doi.org/10.3390/s20020345

[36] Shekhawat, A. S., & Rohilla, Y. (2020). Design and control of two-wheeled self-balancing robot using Arduino. 2020 International Conference on Smart Electronics and Communication (ICOSEC). https://doi.org/10.1109/icosec49089.2020.9215421

[37] Khanna, A., & Ranjan, P. (2015). Solar-powered Android-based speed control of DC motor via secure Bluetooth. 2015 Fifth International Conference on Communication Systems and Network Technologies. https://doi.org/10.1109/csnt.2015.100

[38] Last Minute Engineers. (2022, May 15). Control DC motors with L293D Motor Driver IC & Arduino. Last Minute Engineers. Retrieved June 3, 2022, from https://lastminuteengineers.com/l293d-dc-motor-arduino-tutorial/

[39] Tripathi, M., Bajariya, F., Vishwakarma, S., & Shaikh, Y. (2019). *Self Balancing Robot Using Arduino Uno*.