Grey Wolf Optimizer Based Super Twisted Sliding Mode Controller (STSMC) for Self-balancing Control of a Two-wheel Electric Scooter

Daniel Gosaye Tesfaye

A Thesis Submitted to

The Department of Electrical Power and Control Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Electrical Power and Control Engineering (Control Engineering)

Office of Graduate Studies

Adama Science and Technology University

June, 2021.

Adama, Ethiopia

Grey Wolf Optimizer Based Super Twisted Sliding Mode Controller (STSMC) for Self-balancing Control of a Two-wheel Electric Scooter

Daniel Gosaye Tesfaye

Advisor:

Dr. Tefera Terefe (Assistant professor)

A Thesis Submitted to

The Department of Electrical Power and Control Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of Master's in Electrical Power and Control Engineering (Control Engineering)

Office of Graduate Studies

Adama Science and Technology University

June, 2021.

Adama, Ethiopia

APPROVAL PAGE

I, the advisors of the thesis entitled "Grey Wolf Optimizer Based Super Twisted Sliding Mode Controller (STSMC) for Self-balancing Control of a Two-wheel Electric Scooter" and developed by Daniel Gosaye, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

DR. Tefera Terefe		
Advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by Daniel Gosaye Tesfaye have read and evaluated her thesis entitled "Grey Wolf Optimizer Based Super Twisted Sliding Mode Controller (STSMC) for Self-balancing Control of a Two-wheel Electric Scooter" and examined the candidate during open defense. This is, therefore, to certify that the thesis has been accepted in partial fulfillment of the requirement of the Degree of Masters in control engineering.

Chairperson	Signature	Date
Internal Examiner	Signature	Date
Dr. Beteley Teka		July 29 2021
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

Department Head	Signature	Date
School Dean	Signature	Date
Office of Postgraduate Studies, Dean	Signature	Date

DECLARATION

I hereby declare that this Master Thesis entitled "Grey Wolf Optimizer Based Super Twisted Sliding Mode Controller (STSMC) for Self-balancing Control of a Two-wheel Electric Scooter" is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged though citation.

Daniel Gosaye	_____	_____
Name of student	Signature	Date

RECOMMENDATION

I, the advisor of this thesis, hereby certify that I have read the revised version of the thesis entitled "Grey Wolf Optimizer Based Super Twisted Sliding Mode Controller (STSMC) for Self-balancing Control of a Two-wheel Electric Scooter" prepared under my guidance by Daniel Gosaye submitted in partial fulfillment of the requirements for the degree of Master of Science in Electrical Power and Control Engineering, postgraduate in control engineering.

Therefore, I recommend the submission of revised version of the thesis to the department following the applicable procedures.

DR. Tefera Terefe	_____	_____
Name of Advisor	Signature	Date

ACKNOWLEDGMENT

First and foremost, I want to express my gratitude to the Almighty God for providing me with the strength to continue through this difficult thesis journey. My sincerest gratitude goes to my adviser, Dr. Tefera Terefe, for his inspiration, invaluable advice, invaluable support, and constant encouragement during my thesis work. I also appreciate his insightful remarks and ideas, which have allowed me to strengthen my understanding. Last but not least, I'd want to express my gratitude and blessings to all of my family members, friends, and others that assisted me in any way during the completion of my thesis.

.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

DC	Direct Current
GA	Genetic Algorithm
GWO	Grey Wolf Optimization
HOSMC	Higher Order Sliding Mode Control
HTV	Human Transportation Vehicle
IAE	Integral Absolute Error
IMU	Inertial Measurement Unit
ISE	Integral Square Error
ITAE	Integral Time Absolute Error
MIMO	Multi- input Multi-output
PD	Proportional Derivative
PE	Perturbation Estimation
PID	Proportional Integral Derivative
PSO	Particle Swarm Optimization
RBFNN	Radial Basis Function Neural Network
SFC	State Feedback Controller
SISO	Single-input Single-output
SMC	Sliding Mode Control
SRWNN	Self-Recurrent Wavelet Neural Network
STSMC	Super Twisted Sliding Mode Controller
TWHT	Two Wheeled Human Transporter
TWSBV	Two-Wheeled Self-Balancing Vehicle
VSC	Variable Structure Control

LIST OF SYMBOLS

a	Acceleration
D	Distance between the contact patches of the wheel
g	Acceleration due to gravity
H_L, H_R	Reaction forces impact on the left and right wheels
H_{TL}, H_{TR}	Friction force between the ground and left and right wheels
i_a	Armature current
I_R	Moment inertia of the motor
J_B	Inertial moment of the chassis with respect to the z-axis
J_{TL}, J_{TR}	Inertial moment of the rotating masses with respect to the z-axis
K_e	DC motor back emf constant
K_f	Friction coefficient
K_m	DC motor torque constant
L	Distance between the z-axis and the gravity center of the scooter
L_a	Armature inductance
M_B	Mass of body
M_W	Mass of wheels
R	Radius of the wheel
R_a	Armature resistance
T_L, T_R	Input torque of the left and right wheels
V_a	Armature voltage
V_e	DC motor back emf
τ_a	Load torque
τ_m	Electromagnetic torque
θ	Pitch angle
δ	Yaw angle
ω_m	Motor speed
θ_{WL}, θ_{WR}	Pitch angle of the left and right wheels

ABSTRACT

Transportation is a rapidly expanding industry nowadays. Self-balancing personal transporter scooters were launched in response to the fast growth in demand for personal transporter vehicles. The two-wheel self-balancing scooter is based on an inverted pendulum system and this system is an unstable and nonlinear system. An inertial measurement unit (IMU) is a combination of an accelerometer and a gyroscope measurement used to estimate and obtain the tilt angle of the scooter. The super twisted sliding mode controller (STSMC) is used to balance the scooter by correcting the difference between the desired set point and the actual tilt angle and adjusting the direct current (DC) motor speed correspondingly. When the scooter is tilted forward, the motor moves forward to catch up, to balance the scooter. A proportional integral derivative (PID) controller is used to control the direction of the scooter. That means turning left or right. The STSMC parameters and PID parameters are tuned using Grey Wolf Optimization (GWO), Particle swarm optimization (PSO) and genetic algorithm (GA) for compression, and the controller's dynamic performance evaluation is done using MATLAB/Simulink. Accordingly, the GWO based STSMC for balancing control of scooters has a settling time of 0.2438sec, a rise time of 0.1542sec, and a 0% steady-state error for pitch angle, and the GWO based PID for direction control has a settling time of 0.7641sec, a rise time of 0.1893sec, and 0% steady-state error for yaw angle. On the other hand, the GWO based STSMC-PID control for balancing and direction control has better performance than the PSO and GA based STSMC-PID controllers, and it has a settling time of 0.2sec and 0% steady-state error. The scooter is designed to carry a human load of up to 90 kg. The simulation result for human load varies from 50kg to 90kg and the result shows that for different mass variations, the scooter balancing, and direction control are done in a perfect manner. Finally, simulation results of the proposed STSMC (for balancing) and PID (for turning right and turning left) controller for two-wheel self-balancing scooters based on GWO outperform those resulting from PSO based and GA based STSMC-PID.

Keywords: – *Scooter, Genetic algorithm, Grey wolf optimization, Particle swarm optimization, and STSMC- PID controller.*

CHAPTER ONE

1. INTRODUCTION

1.1 Background

Today, transportation is undoubtedly a fast-growing industry. The Segway Company launched self-balancing personal transporter scooters in response to the significant growth in demand for personal transporter vehicles. The self-balancing personal transporter, which is also a great representation of the personal mobility concept, is now widely used in many industries and institutions, such as the tourism industry, police departments, factories, and so on, with the goal of increasing the efficiency of human transportation and lowering costs. The benefits that this personal transporter vehicle offers, such as enhanced accessibility and zero fuel use, may be regarded as the ultimate solution to the approaching global difficulties caused by rising traffic and environmental pollution occurring all over the world. This two-wheel device represents a better version of the personal transporter types of vehicles that are being used nowadays and simply reaches into the hands of most of society. Multiple accelerometer and gyroscope sensors (a few additional) are used in self-balancing personal transporter models (mainly Segway models) to get acceleration readings and angular rate along different axes. The computational power required by the control unit and the cost are both drawbacks of using several sensors in Segway models.

The self-balancing scooter is responsible for continuously getting feedback on the platform's tilt (angle of inclination), adjusting the error with respect to the reference angle, and keeping the entire platform upright. Furthermore, the control unit of the self-balancing scooter has the capability of responding to any unexpected external force in order to return to a stable position, which improves the passenger's overall safety.

Due to the natural instability of dynamic systems, two-wheeled self-balancing devices are one of the most recent research topics in robotics [1]. The operation of this two-wheel device is based on the inverted pendulum concept. This concept is covered in all control engineering textbooks and is a popular topic among control researchers [2].

Like trying to balance a broom (stick) on the tip of your finger, the two-wheeled balancing scooter is a classic engineering issue based on the inverted pendulum. The term "balance" refers to the scooter's state of equilibrium, which is equivalent to standing upright. However, the system is out of balance, and it continues to fall away from the vertical axis. Therefore, a gyro chip is required to provide the scooter's angle position, which is then fed to the controller for the scooter to balance in the upper right position.

Balance a scooter, accurate information about the present tilt angle from a measurement unit is required. In addition, to adjust for tilt angle, a controller must be implemented. STSMC can control the scooter pitch angle for balancing (move forward and backward) and PID control the yaw angle of scooter which is used for direction control that means to turn right or left. The mathematical model is performed depending on newton 2nd law of motion, and the actuator DC motor is also modeled, and then system is simulated into MATLAB/Simulink.

In recent years, the concept of a mobile scooter has attracted the interest of control system researchers all around the world [3] [4]. Figure 1.1 shows both front and side views of a two-wheeled self-balancing scooter.

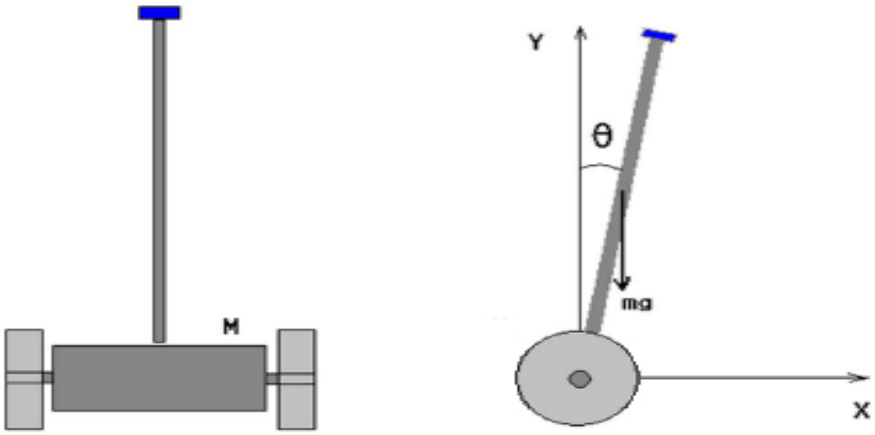


Figure 1.1:- Front and side views of two wheeled scooter [3].

Once the system is balanced (at zero-tilt angle), it can be ordered to move backward, forward and turn left or right.

This two-wheel scooter is made up of three primary components:

a. **Wheels:** used for the system to move forwards or backwards in order to balance the body and turn the system right or left.

b. **Chassis:** holds the system's motors, circuitry, and any other parts.
c. **Pendulum:** The components of the system that are causing the instabilities need to be stabilized.

1.2 Statement of the Problem

Nowadays, the two-wheel self-balancing scooter is the other means of transportation used in different places. When compared to a regular four-wheel vehicle, two-wheeled scooters have a number of advantages, including lower costs, smaller size, easier operation, and better parking. However, the instability that comes with it makes design and modeling much more difficult. This instability problem is solved by applying a robust controller. The Sliding mode controller (SMC) is one of the robust controllers, but this controller has a drawback known as chattering. This problem of traditional sliding mode control is solved by applying a higher order sliding mode (HOSM) controller and one of the HOSM controllers is super twisted sliding mode control (STSMC). The STSMC controller avoids the chattering problem while maintaining the other sliding mode properties.

This personal transporter vehicle provides benefits such as increased accessibility and zero fuel consumption, which might be considered the ultimate solution to the upcoming global issues caused by increased traffic and environmental pollution occurring all over the world. In our country, there are several tourist attractions, so to visit the place you need some device that transports people from one place to another. For example, Entoto park covers a large area and four-wheel cars do not enter the park, so to visit the place, people travel by foot, but when we developed this two-wheel electric scooter, it solved this problem as well, as it is simple to drive and suitable for all people, from young to old.

1.3 Objectives of the Study

1.3.1 General Objective

General The general objective of this thesis is to develop a two-wheel self-balancing electric scooter based on the inverted pendulum concept by using STSCM and PID controllers in MATLAB/Simulink.

.

1.3.2 Specific Objectives

The specific objectives of this research thesis are:

- To perform the simulation of a self-balancing scooter based on a mathematical model.
- To design an STSCM and PID controller for a two-wheel self-balancing scooter.
- To tune controllers (STSMC and PID) parameters using GWO, PSO, and GA optimization algorithms.
- To evaluate the performance of the developed self-balancing scooter in MATLAB/Simulink using a standard approach.
- To make relevant conclusions and recommendations based on the system result.

1.4 Scope of the Study

The thesis focuses on the mathematical modeling and simulation design of the two-wheel self-balancing scooter. Determine the specification of the scooter's actual parameters, such as mass of parson, mass of wheel, radius of the wheel, and distance between the contacts patches of the wheel, and so on. Depending on the mathematical mode of the scooter design, STSMC for balancing control and a PID controller for direction control in MATLAB/Simulink. STSMC-PID controller parameter tune using GWO, PSO and GA for comparison.

1.5 Limitations of the Study

The thesis is limited to the simulation of the system using MATLAB/Simulink by showing how the different components of the system interact with each other. This is because the implementation of a real system is difficult to procure the components. Along with it, it takes the additional time and cost incurred in making the actual prototype implementation.

1.6 Motivation of the Study

The advantages that this personal transporter vehicle provides, such as increased mobility and zero fuel use, can be viewed as the ultimate solution to the upcoming global issues caused by increased traffic and environmental degradation occurring all over the world. This transport device is considered for desired features such as no noise pollution, consistent speed, ease of use, low energy usage, and nonpolluting. As a result, scooters have a wide range of uses for transportation in densely populated and polluted cities, as well as in hospitals (because they are noiseless and environmentally friendly), shopping malls, airports, parks, and especially for

entertainment. As compared to a four-wheeled car, a two-wheeled scooter has several advantages, including lower costs, smaller size, simpler operation, better parking, and so on.

1.7 Significance of the Study

A scooter is an electric vehicle that is driven by human movement and allows for safe and balanced movement. It is used not only in the field of tourism, but also in offices, industrial buildings, airports, and other places. The nature of this means of transportation makes it ideal for moving around cities and doing tourism in a different way**,** without getting tired of long walks. Its safety and zero emissions, as it is an electric vehicle.

1.8 Thesis Organization

This thesis is organized into five chapters.

Chapter 1: - Presents the introduction of the research work, statement of the problems, objectives, significance, scope, and limitations of the research.

Chapter 2:- This chapter includes a literature review on two-wheel self-balancing electric scooters and related work.

Chapter 3:- Describes the methods followed in this thesis and the mathematical model of the scooter and DC motor are modeled. **T**his chapter also presents the control design (STSCM and PID) and optimization algorithms (GWO, PSO, and GA).

Chapter 4:- Presents the proposed results and discussions.

Chapter 5:- This chapter includes conclusions and recommendations.

CHAPTER TWO

2. LITERATURE REVIEW

2.1 Introduction

This section gives an overview of earlier efforts to build self-balancing devices. In the preliminary design of a self-balancing scooter, the knowledge gathered from reading various papers is quite useful.

The introduction of one popular commercial product (Segway) [5] has created a lot of interest in the wheeled inverted pendulum model. A dual-wheel robot features two wheels and an upright body structure that contains all of the circuits. When it is going to fall forward or backward, it adjusts its position to avoid becoming unstable [6].

JOE, a two-wheeled vehicle prototype constructed by Felix Grasser [7], a researcher at the Swiss Federal Institute of Technology, as shown in Figure 2.1. The basic goal of this robot is to keep its driver balanced on two coaxial wheels. Each of the coaxial wheels is powered by a DC motor. The vehicle is controlled by supplying torque to the appropriate wheels, and its design allows for stationary U-turns. The system is stabilized using a linear state space controller that uses sensory input from a gyroscope and motor encoders.

Figure 2.1:- JOE [7].

SEGWAY PT was designed by Dean L. Kamen [5] as a human transporter application based on a two-wheel robot. Figure 2.2 shows the model in detail. The device is operated by moving body

weight and balances on two parallel wheels. The user can navigate in small steps or curbs with this technology, allowing for easy navigation on a variety of terrains. To stay upright, this invention depends on five gyroscopes and a variety of additional tilt sensors. The additional sensors are included as a safety precaution, as only three gyroscopes are required for the entire system.

Figure 2.2:- Segway PT [5].

2.2 Recent Related Work on Self-balancing Device

There are many researchers' previous efforts to construct two-wheel self-balancing devices. Some of them are.

It is particularly difficult to operate a scooter based on an inverted pendulum model [8] [9] since it has highly nonlinear characteristics and uncertain state parameters. Previously, some studies focused on signal processing units, actuators and sensors, modeling, and control schemes in mobile robots. On top of that, the pole placement approach was used on a scooter. Author in [10] proposed a self-balancing human transportation device to test feedback controls such PID and pole-placement control.

For a self-balancing controller, linear quadratic regulation (LQR) was presented by LMM Thwin et al. [11]. This approach, on the other hand, works best when the scooter's parameters are constant and the scooter is linear at small tilt angles.

Tsai et al. [12] presented an adaptive control using radial basis function neural networks (RBFNNs) for a two-wheeled self-balancing scooter. The proposed two adaptive controllers

employing RBFNNs were optimized using a Backpropagation technique to achieve self-balancing and yaw control. However, neural network control takes a long time to train, requires a lot of memory, and can sometimes fall into a local optimum.

Gia Minh Thao et al. [13] Proposed a PD Sliding Mode Controller for Two-Wheeled Self-Balancing Robot. The proposed controller has two objectives: pitch angle regulation and tracking of the desired position. There are two loops in the proposed controller. The first loop is a pitch angle regulator using sliding mode. A PD position tracking controller is being used in the second loop. This author design a robust controller based on sliding mode control; however, sliding mode control produces significant chattering due to switching around the sliding surface.

Nguyen Ngoc Son et al. [14] this paper introduces an Adaptive Backstepping Self-balancing Control of a Two-wheel Electric Scooter Lyapunov stability-satisfying feedback control is paired with adaptive backstepping control. Using a recursive structure to identify the regulated function and estimate uncertain parameters, feedback control rule that efficiently manages Scooter's self-balancing controller. Adaptive backstepping control used in this simulation are robustness and good performance. However, Backstepping control requires accurate model parameters.

Pranav Sevekari et.al. [15] Introduces Robust Control for Stable and Safe Performance of a Two Wheeled Human Transporter. By proposing a higher order sliding mode control (HOSMC) for the two-wheeled human transporter (TWHT) system's stable and safe performance. For control development, a reduced order model is being considered. Higher-order sliding mode differentiators are used to estimate unmeasurable states. The proposed controller are evaluated in simulation and compared to a linear state feedback controller (SFC) to demonstrate their efficacy. The robustness of HOSMC is test by varying the weight of the rider.

Despite the promised robustness, real-time implementation of sliding mode control (SMC) is limited by a significant drawback known as chattering, which is a high frequency bang-bang type of control action. In order to overcome this drawback, or this problem will be settled by using a higher order sliding mode (HOSM) controller. One of the HOSM controllers is the super twisted sliding mode controller (STSMC) and this controller is implemented to remove the limitations of the SMC controller (like chattering) and simulated in MATLAB/Simulink and the results are presented.

The main contribution of this thesis is the design of a super twisting sliding mode controller and a PID controller for successful self-balancing and direction (turn right or left) control of a two-wheel self-balancing electric scooter. STSMC parameters and PID parameters are tuned using GWO, PSO and GA. As far as we know, the super twisted sliding mode controller tuned using GWO, PSO and GA has never been applied to the self-balancing control of an electric scooter before.

CHAPTER THREE

3. SYSTEM MODELING AND CONTROLLER DESIGN

3.1 Introduction

In this chapter, the mathematical model of the body, wheel, and actuator are modeled, and a super twisted sliding mode controller (STSMC) is applied for balancing the scooter and a PID controller is used for direction control of the scooter. The parameters of these two controllers are tuned using the grey wolf optimization algorithm (GWO), particle swarm optimization algorithm (PSO) and genetic algorithm (GA) based on objective function (J).

3.2 Materials

In this research work, software such as MATLAB/2017a, Microsoft Office 2013, and Math Type 6.0 equation editor are used. MATLAB is a computing software which consists of technical toolboxes and Simulink. Microsoft office is used for editing the thesis documentation. Math Type 6.0 is used for writing mathematical formulas and equations in Microsoft office word and PowerPoint presentation tools.

3.3 Methods

The methodology of this thesis mainly focuses on software algorithms. The thesis will be developed using a flowchart that lists all of the tasks to be performed.

Figure 3.1 below shows the flow chart of the thesis, which starts with a literature review, to collect information on topics related to the two-wheeled balancing scooter. After analyzing all of the resources, the next step is to complete the mathematical modeling for the scooter, which is the foundation of this thesis. Depending on the mathematical model and the characteristics of the two-wheel self-balancing scooter, the effective controller for this system is selected and the parameters of the controller are tuned using different optimization algorithms (GWO, PSO, and GA). To evaluate the system's performance, MATLAB/Simulink will be used as simulation tools.

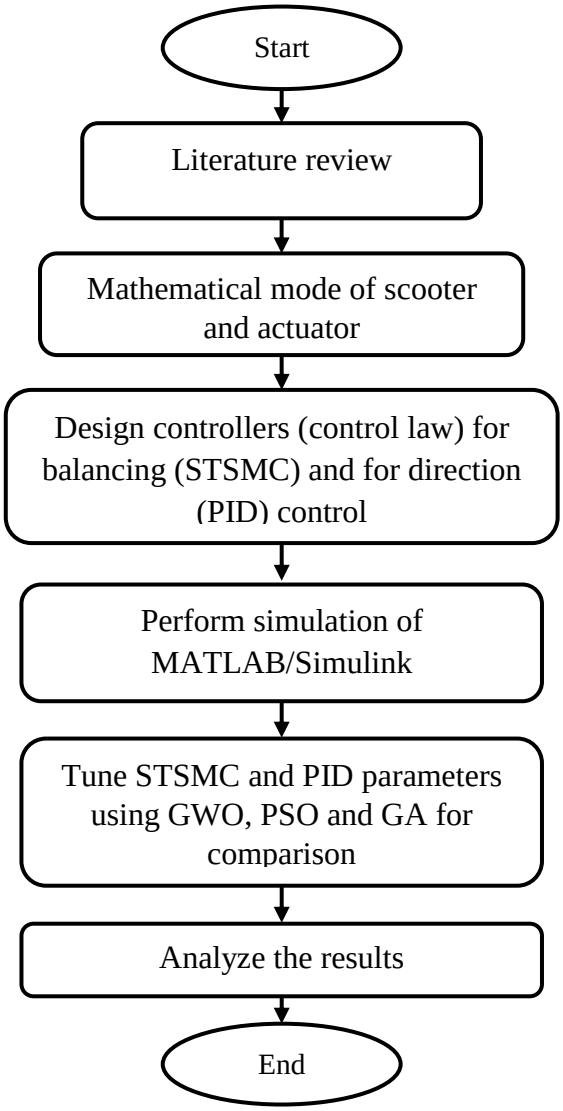


Figure 3.1:- Work flow.

3.4 Working Principle of Two-wheel Self-balancing Scooter

A self-balancing personal transporter's working principle is to continuously get feedback on the platform's tilt (angle of inclination), then correct for the error in relation to the reference angle, and keep the entire platform upright.

The key advantage of the proposed scooter is its ability to self-balance. This feature helps the scooter to always stay in equilibrium. The driver steers the scooter forward by adjusting their weight forward on the platform, and backward by shifting their weight backward on the

platform. In addition, the driver must direct the handlebar to the right or left to turn. The scooter is made up of two parallel coaxial wheels that are driven by two DC electric motors.

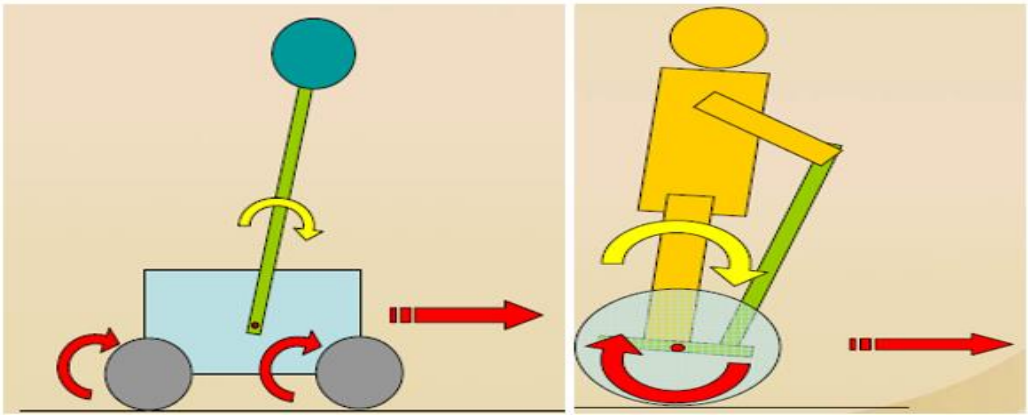


Figure 3.2:- Working principle of scooter.

3.5 The Proposed Block Diagram for Two-wheel Self-balancing Scooter

As shown in Figure 3.3 below, the proposed system block diagram of the two-wheel self-balancing scooter is a MIMO system. The system, by nature, is a non-linear and unstable system. So, to stabilize the system, it needs a controller.

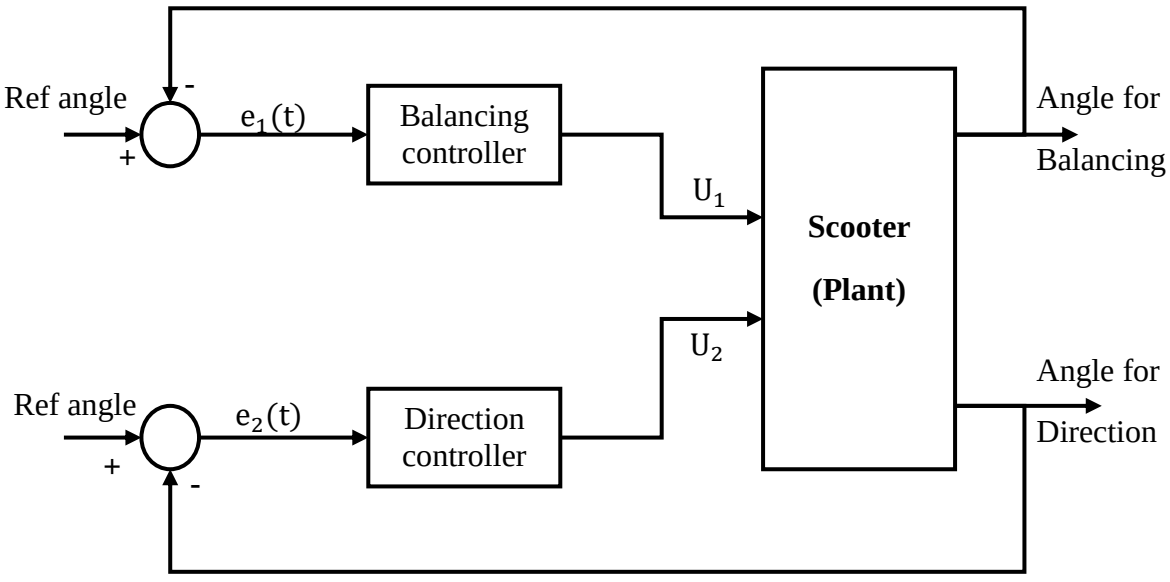


Figure 3.3:- Proposed Block Diagram for two-wheel self-balancing scooter.

This system has two inputs. Each of them is an angle. The first angle is used for balancing the scooter and the second one is used for direction control (turn right or left). This angle is

compared with the reference angle and the error signal is fed to the corresponding controller for balance (stabilize) and direction control of the two-wheel self-balancing scooter.

3.6 Mathematical Model of Scooter

An electric scooter is an under-actuated mechanical system driven by an electrical actuator. Hence, scooter modeling requires both mechanical subsystem modeling and electrical subsystem modeling.

The mathematical model of a scooter is separated into 3 parts.

a) Wheel modeling
b) Body modeling and
c) DC motor (actuator) modeling

To determine the mathematical model of the electric scooter, the Newton 2nd law method is applied [14].

3.6.1 Wheel Modeling

For the Scooter's left wheel (the same as the right wheel).

Figure 3.4:- Force Analysis of Right and left-Wheel [14].

Along horizontal axis X, apply Newton's 2nd Law of Motion

$$\sum F_x = ma \tag{3.1}$$

$$M_W \ddot{x}_{WL} = H_{TL} - H_L \tag{3.2}$$

Along vertical axis Y, apply Newton's 2nd Law of Motion

$$\sum F_Y = ma \tag{3.3}$$

$$M_W y_{WL} = V_{TL} - V_L - M_W g \tag{3.4}$$

The sum of the moments around the wheel's center equals to:

$$\sum M_O = I\alpha \tag{3.5}$$

$$J_{WL}\ddot{\theta}_{WL} = T_L - H_{TL}R \tag{3.6}$$

The relationship between linear motion component and angular motion component given by

$$x_{WL} = \theta_{WL}R \tag{3.7}$$

Moment of inertia of left wheel

$$J_{WL} = \frac{1}{2}M_{WL}R^2 \tag{3.8}$$

From (3.2) sum the left and right wheel equation

$$M_W(\ddot{x}_{WL} + \ddot{x}_{wR}) = (H_{TL} + H_{TR}) - (H_L + H_R) \tag{3.9}$$

From (3.6)

$$\begin{cases} J_{WL}\ddot{\theta}_{WL} = T_L - H_{TL}R & \text{for the left wheel and} \\ J_{WR}\theta_{WR} = T_R - H_{TR}R & \text{for the right wheel.} \end{cases} \tag{3.10}$$

Solve (3.10) as follow

$$\begin{cases} H_{TL} = \dfrac{T_L - J_{WL}\ddot{\theta}_{WL}}{R} \\ H_{TR} = \dfrac{T_R - J_{WR}\ddot{\theta}_{WR}}{R} \end{cases} \tag{3.11}$$

3.6.2 Modeling of Scooter Body

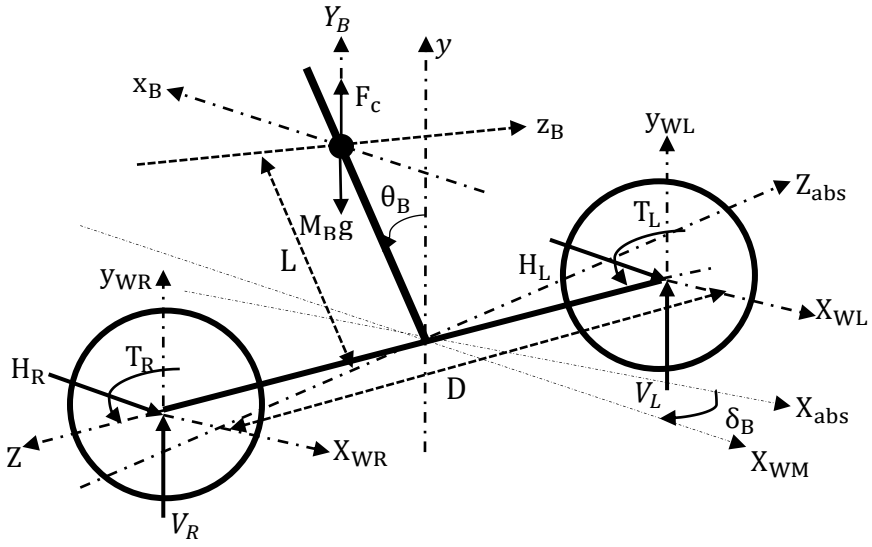


Figure 3.5:- Coordinate system of body of scooter [14].

Along horizontal axis X, apply Newton's 2nd Law of Motion

$$\sum F_x = ma \tag{3.12}$$

$$M_B\ddot{x}_B = H_L + H_R \tag{3.13}$$

Along vertical axis Y, apply Newton's 2nd Law of Motion

$$\sum F_y = ma \tag{3.14}$$

$$M_B\ddot{y}_B = V_L + V_R - M_Bg + \frac{T_L + T_R}{L}\sin(\theta_B) \tag{3.15}$$

The sum of the moments around the wheel's center equals to:

$$\sum M_O = I\alpha \tag{3.16}$$

$$J_\theta\ddot{\theta}_B = (V_L + V_R)L\sin(\theta_B) - (H_L + H_R)L\cos(\theta_B) - (T_L + T_R) \tag{3.17}$$

$$x_B = L\sin(\theta_B) + \frac{x_{WL} + x_{WR}}{2} \tag{3.18}$$

$$y_B = -L(1 - \cos(\theta_B)) \tag{3.19}$$

Moment of inertia of chassis

$$J_B = \frac{1}{3} M_B L^2 \tag{3.20}$$

$$\theta = \theta_B = \theta_W = \theta_{WL} = \theta_{WR} \tag{3.21}$$

$$x_{WM} = \frac{x_{WL} + x_{WR}}{2} \tag{3.22}$$

And for the rotation

$$\delta = \frac{x_{WL} - x_{WR}}{D} \tag{3.23}$$

For the chassis rotation:

$$J_\delta \ddot{\delta} = \frac{D}{2}(H_L - H_R) \tag{3.24}$$

After Solve (3.15) for $V_L + V_R$ and (3.13), (3.21) into (3.17) than

$$J_B \ddot{\theta} = M_B L(\ddot{y}_B \sin(\theta) - \ddot{x}_B \cos(\theta)) + M_B g L \sin(\theta) - (T_L + T_R)(1 + \sin^2(\theta)) \tag{3.25}$$

Substitute (3.22) into (3.18) than differentiate 2 time.

$$\ddot{x}_B = L\ddot{\theta}\cos(\theta) - L\dot{\theta}^2 \sin(\theta) + \ddot{x}_{WM} \tag{3.26}$$

And the same way as x_B differentiate y_B of (3.19) than

$$\ddot{y}_B = -L\ddot{\theta}\sin(\theta) - L\dot{\theta}^2 \cos(\theta) \tag{3.27}$$

Multiply (3.26) by $\cos(\theta)$and multiply (3.27) by $\sin(\theta)$ and subtract both equation than we get

$$\ddot{y}_B \sin(\theta) - \ddot{x}_B \cos(\theta) = -L\ddot{\theta} - \ddot{x}_{WM} \cos(\theta) \tag{3.28}$$

Substitute (3.28) and (3.20) into (3.25)

$$\frac{4}{3} M_B L^2 \ddot{\theta} + M_B L \cos(\theta) \ddot{x}_{WM} = M_B g L \sin(\theta) - (T_L + T_R)(1 + \sin^2(\theta)) \tag{3.29}$$

Substitute (3.11) , (3.13) and (3.21) into (3.9).

$$M_W(\ddot{x}_{WL}+\ddot{x}_{WR})=\left(\frac{T_L+T_R-(J_{WL}\ddot{\theta}+J_{WR}\ddot{\theta})}{R}\right)-M_B\ddot{x}_B \tag{3.30}$$

Whereas

$$\begin{aligned} J_{WL}+J_{WR}&=2J_W \\ \ddot{X}_{WL}+\ddot{X}_{WR}&=2\ddot{X}_{WM} \end{aligned} \quad \text{then} \tag{3.31}$$

$$2M_W\ddot{x}_{WM}=-M_B\ddot{x}_B+\frac{(T_L+T_R)}{R}-\frac{2J_W\ddot{\theta}}{R} \tag{3.32}$$

Solve (3.8) for the left and right wheel than sum two equation

$$J_{WL}+J_{WR}=R^2(M_{WL}+M_{WR}) \tag{3.33}$$

$$2J_W=R^2M_W \tag{3.34}$$

Substitute (3.26) and (3.34) into (3.32) than we get

$$(M_BL\cos(\theta)+RM_w)\ddot{\theta}+(2M_W+M_B)\ddot{x}_{WM}=M_BL\dot{\theta}^2\sin(\theta)+\frac{(T_L+T_R)}{R} \tag{3.35}$$

Solve (3.29) for $\ddot{x}_{WM}$ and substitute into (3.35) than

$$\begin{aligned} &\left((2M_W+M_B)-\frac{0.75(M_BL\cos(\theta)+M_WR)\cos(\theta)}{L}\right)\ddot{\theta}=\frac{0.75g(2M_W+M_B)\sin(\theta)}{L}- \\ &0.75M_B\sin(\theta)\cos(\theta)\dot{\theta}^2-\left(\frac{0.75(2M_W+M_B)(1+\sin^2(\theta))}{M_BL^2}+\frac{0.75\cos(\theta)}{RL}\right)(T_L+T_R) \end{aligned} \tag{3.36}$$

Solve (3.29) for $\ddot{\theta}$ and substitute into (3.35) than

$$\begin{aligned} &\left(-0.75\frac{\cos(\theta)(M_BL\cos(\theta)+M_WR)}{L}+(2M_W+M_B)\right)\ddot{x}_{WM}=-0.75(M_BL\cos(\theta)+M_WR) \\ &\frac{g\sin(\theta)}{L}+M_BL\sin(\theta)\dot{\theta}^2-\left(0.75\frac{(M_BL\cos(\theta)+M_WR)(1+\sin^2(\theta))}{M_BL^2}+\frac{1}{R}\right)(T_L+T_R) \end{aligned} \tag{3.37}$$

Solve H_{TL} from (3.2)

$$H_{TL}=M_W\ddot{x}_{WL}+H_L \tag{3.38}$$

Differentiate (3.7)

$$\ddot{\theta}_{WL}=\frac{\ddot{x}_{WL}}{R} \tag{3.39}$$

Substitute (3.38) into (3.6) than

$$\begin{cases} H_L = \dfrac{T_L}{R} - \ddot{x}_{WL}\left(M_W + \dfrac{J_W}{R^2}\right) & \text{for the left wheel} \\ H_R = \dfrac{T_R}{R} - \ddot{x}_{WR}\left(M_W + \dfrac{J_W}{R^2}\right) & \text{for the right wheel} \end{cases} \tag{3.40}$$

Subtract two equation of (3.40) and then substitute (3.23) than

$$H_L - H_R = \left(\frac{T_L - T_R}{R}\right) - D\ddot{\delta}\left(M_W + \frac{J_W}{R^2}\right) \tag{3.41}$$

Substitute (3.41) into (3.24) then

$$\left(J_\delta + \frac{1}{2}D^2\left(M_W + \frac{J_W}{R^2}\right)\right)\ddot{\delta} = \frac{1}{2}D\frac{T_L - T_R}{R} \tag{3.42}$$

We have

$$J_W = \frac{1}{3}M_B R^2 = J_\delta = \frac{1}{3}M_B\left(\frac{D}{2}\right)^2 = \frac{1}{12}M_B D^2 \tag{3.43}$$

Substitute (3.43) into (3.42) than we get

$$\ddot{\delta} = \frac{6}{(M_B + 9M_W)DR}(T_L - T_R) \tag{3.44}$$

Let

$$\begin{aligned} U_1 &= T_L + T_R \quad \text{and} \\ U_2 &= T_L - T_R \end{aligned} \tag{3.45}$$

The basic system of equations are (3.36), (3.37) and (3.44).

Let $x_1 = \theta$, $x_2 = \dot{\theta}$, $x_3 = x$, $x_4 = x$, $x_5 = \delta$ and $x_6 = \dot{\delta}$. State equations of the e-scooter is rewritten as

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f_1(x_1) + f_2(x_1, x_2) + g_1(x_1)(T_L + T_R) \\ x_3 = x_4 \\ \dot{x}_4 = f_3(x_1) + f_4(x_1, x_2) + g_2(x_1)(T_L + T_R) \\ \dot{x}_5 = x_6 \\ \dot{x}_6 = g_3(T_L - T_R) \end{cases} \tag{3.46}$$

Whereas

$$f_1(x) = \frac{\left(\frac{-0.75\,g\sin x_1}{L}\right)}{\left(\frac{0.75(M_W R + M_B L(\cos x_1))(\cos(x_1))}{(2M_W + M_B)L} - 1\right)}$$

$$f_2(x_1, x_2) = \frac{\left(\frac{0.75\,M_B L(\sin x_1)(\cos x_1)}{(2M_W + M_B)L}(x_2)^2\right)}{\left(\frac{0.75(M_W R + M_B L(\cos x_1))(\cos x_1)}{(2M_W + M_B)L} - 1\right)}$$

$$g_1(x_1) = \frac{\left(\frac{0.75(1 + (\sin x_1)^2)}{M_B L^2} + \frac{0.75(\cos x_1)}{(2M_W + M_B)RL}\right)}{\left(\frac{0.75(M_W R + M_B L(\cos(x_1)))(\cos x_1)}{(2M_W + M_B)L} - 1\right)}$$

$$f_3(x_1) = \frac{\left(\frac{-0.75\,g(M_W R + M_B L(\cos x_1))\sin(x_1)}{L}\right)}{\left(2M_W + M_B - \frac{0.75(M_W R + M_B L(\cos x_1))(\cos x_1)}{L}\right)}$$

$$f_4(x_1, x_2) = \frac{M_B(\sin x_1)(x_2)^2}{\left(2M_W + M_B - \frac{0.75(M_W R + M_B L(\cos x_1))(\cos x_1)}{L}\right)}$$

$$g_2(x_1) = \frac{\left(\frac{0.75(M_W R + M_B L(\cos x_1))(1 + (\sin x_1)^2)}{M_B L^2} + \frac{1}{R}\right)}{\left(2M_W + M_B - \frac{0.75(M_W R + M_B L(\cos x_1))(\cos x_1)}{L}\right)}$$

$$g_3 = \frac{6}{(M_B + 9M_W)DR}$$

Table 3.1 Parameters of scooter with specified value and unit [5] [14].

Symbol	Value (unit)	Description
θ	Degree	Pitch angle
δ	Degree	Yaw angle
M_W	6kg	Mass of the wheel
M_B	90kg	Mass of the body
R	0.2m	Radius of the wheel
L	1m	Distance between the z-axis and the gravity center of the scooter
D	0.6m	Distance between the contact patches of the wheel
g	9.8m/s^2	Acceleration due to Gravity

3.6.3 DC Motor Modeling

In control systems, the DC motor is a common actuator. It provides direct rotary motion and can also give translational motion when used in combination with wheels or drums and cables. The rotor's and armature's electric circuit free-body diagram are represented as shown in Figure 3.6.

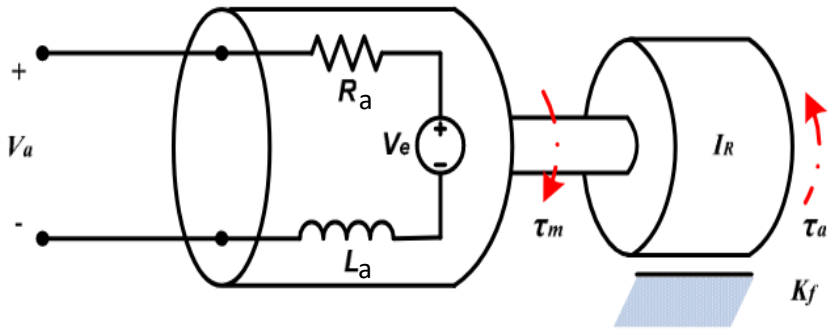


Figure 3.6:- Equivalent circuit of a DC motor [16].

The dynamic of DC motor is governed by the following equation. Using Kirchhoff's voltage law of DC motor is given by

$$v_a = R_a i_a + L_a \frac{d}{dt} i_a + v_e \tag{3.47}$$

Electric motor torque is proportional to armature current and magnetic field strength. By a constant factor K_m, the motor torque is proportional to only the armature current as shown below.

$$\tau_m = k_m i_a \tag{3.48}$$

By a constant factor K_e, the back emf is proportional to the shaft's angular velocity.

$$v_e = k_e \omega_m \tag{3.49}$$

The back emf and motor torque constant have the same numerical value, therefore we will use K to represent both.

By apply Newton's law to the motor system and we get

$$\tau_m = I_R \frac{d\omega}{dt} + k_f \omega_m + \tau_a \tag{3.50}$$

Substitute (3.49) into (3.47) and solve i_a of (3.48) and substitute again into (3.47) then

$$V_a = \frac{R_a}{K}\tau_m + \frac{L_a}{K}\frac{d\tau_m}{dt} + K\omega_m \tag{3.51}$$

Now, taking the Laplace transform of (3.50) and (3.51).

$$V_a = \frac{R_a}{K}\tau_m(s) + \frac{L_a}{K}s\tau_m(s) + K\omega_m(s) \tag{3.52}$$

$$\tau_m(s) = (I_R s + k_f)\omega_m(s) + \tau_a \tag{3.53}$$

From the above motor dynamics equation, the torque driving the right and left wheel can be expressed as.

$$\begin{cases} U_1 = \dfrac{K(I_R s + k_f)}{R_a(I_R s + k_f) + L_a s(I_R s + k_f) + K^2}(V_{aL} + V_{aR}) \\ U_2 = \dfrac{K(I_R s + k_f)}{R_a(I_R s + k_f) + L_a s(I_R s + k_f) + K^2}(V_{aL} - V_{aR}) \end{cases} \tag{3.54}$$

Table 3.2:- DC motor parameters value and unit [5] [17].

Parameters	Description	Value (unit)
R_a	Armature Resistance	$2.5\,\Omega$
L_a	Armature Inductance	0.0005H
v_a	Armature Voltage	V
v_e	DC motor Back Emf	V
k_e	DC motor back emf constant	0.58Vs/rad
k_m	DC motor torque constant	0.58 Nm/A
I_R	Moment inertia of the motor	$0.02 kg.m^2/s^2$

3.7 System Decoupling

The two-wheel scooter is a high coupled nonlinear system, according to the state-space representation. The decoupling the whole system into two separate sub-systems. The following decoupling transformation is used to convert V_θ and V_δ to the voltages of the left (V_{aL}) and right (V_{aR}) wheels motors, similar to the work in [18] [19] [20].

Assume

$$\begin{cases} V_\theta = V_{aL} + V_{aR} \\ V_\delta = V_{aL} - V_{aR} \end{cases} \tag{3.55}$$

So

$$\begin{cases} V_{aL} = \frac{1}{2}(V_\theta + V_\delta) \\ V_{aR} = \frac{1}{2}(V_\theta - V_\delta) \end{cases} \tag{3.56}$$

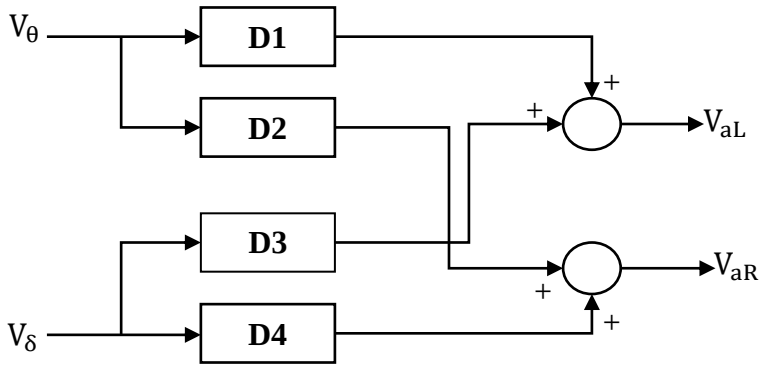


Figure 3.7:- Decoupling unit.

Where as

$$D1 = D2 = D3 = 0.5 \text{ and}$$
$$D4 = -0.5$$

3.8 Sliding Mode Control

Nonlinear sliding-mode control (SMC) has remarkable accuracy, robustness, and ease of tuning and implementation compared to other control approaches [21].

SMC systems are built to drive system states onto a specific state space surface known as a sliding surface. Sliding mode control maintains the states in the region of the sliding surface once it has been reached. As a result, the sliding mode controller is a two-part controller. The first phase entails designing a sliding surface. The second task is to choose a control law.

SMC has two major benefits. The first is that the dynamic behavior of the system can be changed by choosing a specific sliding function. Second, some uncertainties have no effect on the closed loop response. This principle applies to bounded model parameter uncertainties, disturbances, and non-linearity.

In reality, SMC provides for the control of nonlinear processes that are exposed to external disturbances and large model uncertainties.

Consider the nonlinear SISO system

$$\begin{aligned} \dot{x} &= f(x,t) + g(x,t)u \\ y &= k(x,t) \end{aligned} \tag{3.57}$$

In this case, y and u represent the output and input variables, respectively.

The control's goal is to make the output variable y follow a specified reference y_{des}, which means that the output error variable $e = y - y_{des}$ must tend to zero after a reasonable time transient.

SMC synthesis is separated into two stages, as stated earlier.

a) sliding surface design
b) control input design

3.8.1 Sliding Surface Design

The tracking error and a number of its variants are used to analyze the sliding surface as follow.

$$s = s\left(e, \dot{e}, \ldots, e^{(k)}\right) \tag{3.58}$$

The most common type of sliding surface are as following.

$$s = \dot{e} + c_0 e \tag{3.59}$$

$$s = \ddot{e} + c_1 \dot{e} + c_0 e \tag{3.60}$$

$$s = e^{(k)} + \sum_{i=0}^{k-1} c_i e^{(i)} \tag{3.61}$$

The number of derivatives to include ($k = r - 1$, where r is the input output relative degree). The equation $s = 0$ defines a surface in the error space known as a "sliding surface" from a geometrical point view. The controlled system's trajectories are driven onto a sliding surface, where the system's behavior meets the design specifications [22]. The sliding surface has the following typical form, which is dependent on only one scalar parameter, c.

$$s = \left(\frac{d}{dt} + c\right)^k e \tag{3.62}$$

$$\begin{aligned} k &= 1 \quad s = \dot{e} + ce \\ k &= 2 \quad s = \ddot{e} + 2c\dot{e} + c^2 e \end{aligned} \tag{3.63}$$

Whereas c is positive constant.

3.8.2 Control Input Design

The next stage is to identify a control action that directs the variable s to zero in a finite length of time, or a control that directs the system's paths onto the sliding manifold. There are different SMC:

- Traditional (or 1st order) SMC and
- High-order SMC.

3.9 1st order Sliding Mode Control

There are a wide variety of system types for which the sliding mode controller has been built, including MIMO systems, nonlinear and stochastic models, as well as models based on discrete time and infinite dimensions.

By utilizing a high-speed switching control rule, variable structure control (VSC) achieves two goals. To begin, it steers the state trajectory of the nonlinear plant onto a specific and user-selected sliding or switching surface in state space. This surface is known as the switching surface because a control path has one gain if the plant's state trajectory is above the surface and another gain if the trajectory is below the surface. Second, the plant's state trajectory on this surface is retained for all subsequent times. During the process, the structure of the control system varies from one to the next, gaining the name variable structure control.

Sliding mode control is a nonlinear control method in which several control mechanisms are built to ensure that trajectories always move towards a switching state. As a result, the final trajectory will not be totally contained within a single control structure [23]. The fundamental benefit of sliding mode control is its robustness, as well as its low sensitivity to changes in plant parameters and disturbances, which eliminates the need for exact modeling. Decoupling the system's overall motion into smaller, independent components is another benefit of sliding mode control. This simplifies feedback design. Slider mode control has shown to be useful in a wide range of situations because of its features.

Across the manifold s=0, the control is discontinuous.

$$u = -K\,\mathrm{sgn}(s) \tag{3.64}$$

Whereas

$$u = \begin{cases} -K & s > 0 \\ +K & s < 0 \end{cases} \qquad (3.65)$$

K is a positive constant.

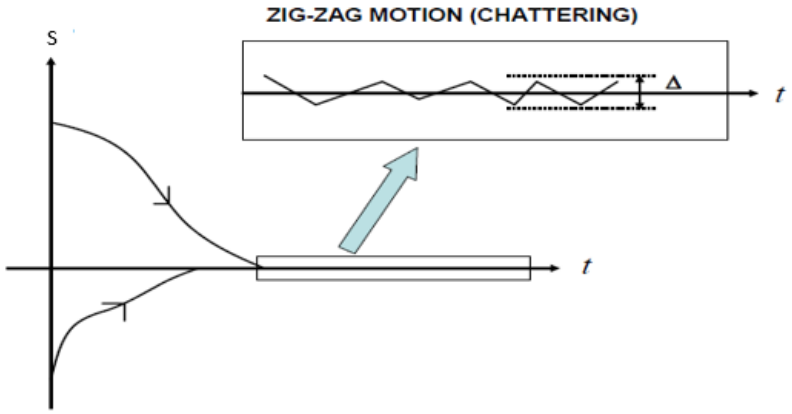


Figure 3.8:- Starting from various initial conditions, a typical evolution of the s variable [24].

To address the above issue (also known as the "chattering phenomenon"), smoothed implementations of sliding mode control approaches have been proposed, in which the discontinuous "sign" term is replaced by a continuous smooth approximation function [25] [26]. Two of them are as follows.

$$\text{SAT} \qquad u = -K\,\text{sat}(s,\varepsilon) \equiv -K\frac{s}{|s|+\varepsilon} \qquad \text{were } \varepsilon > 0 \;\; \varepsilon \approx 0$$

$$\text{TANH} \qquad u = -K\tanh(s/\varepsilon) \qquad (3.66)$$

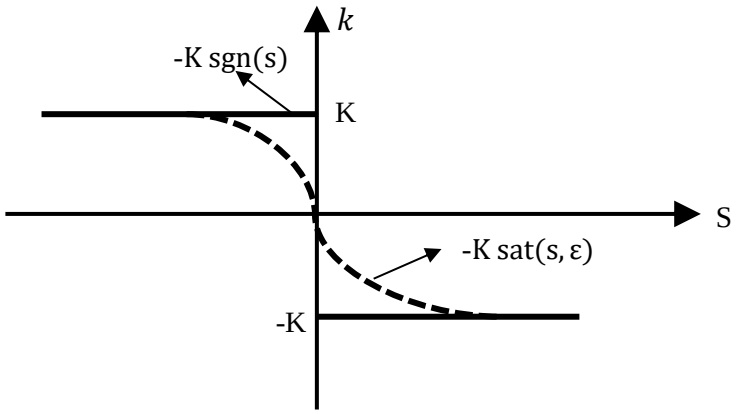


Figure 3.9:- Smooth approximations function (saturation function).

Unfortunately, this technique works only under such situations, such as when there are no hard uncertainties and the control action to reduce them can be set to zero in sliding mode.

3.10 2nd Order Sliding Mode Control

Some problems can be mitigated by using the smooth approximations mentioned above, but this comes at the loss of robustness [27]. 2nd order SMC algorithms are an effective alternative that completely solves the chattering problem while maintaining robustness. One of popular 2nd order SMC algorithm is the so-called Super Twisting Algorithm [28]**.**

3.10.1 Super Twisting Sliding Mode Controller

For systems of relative degree-1, STSMC is a viable alternative to traditional 2nd order SMC for avoiding chattering without affecting tracking output [29].

The high order sliding mode (HOSM) technique is another option for eliminating chattering. The fundamental disadvantage of this control approach is that the stability proofs are relied on geometrical methods, because presenting the Lyapunov function is a tough operation. The proposed STSMC is based on quadratic like Lyapunov functions, allowing for the establishment of an explicit relationship between the controller design parameters [30].

As a rule, the sliding mode control signal consists of two elements. Dynamical systems and sliding surfaces are the focus of one type of control. It is also important to maintain the system dynamics on the sliding surface by using a switching control.

As a result, the sliding surface is defined as the following:

$$s = ce + \dot{e} \tag{3.67}$$

Where c is a sliding constant and e and $\dot{e}$ are the error and error derivative respectively. The SMC is given by:

$$u = u_{eq} + u_c \tag{3.68}$$

Where u_{eq} is the equivalent control without considering system uncertainty and external disturbance. Its purpose is to keep the sliding surface variables under control. The equivalent control is derived by considering that the derivative of the surface is null $\dot{s} = 0$. u_c is the discrete control, which ensures convergence such that $s\dot{s} < 0$.

In traditional sliding mode control, switching control is usually adopted as:

$$u_c = -K\,\mathrm{sgn}(s) \tag{3.69}$$

K is a positive constant, whereas sign(s) is a symbolic function defined by:

$$\mathrm{sgn}(s) = \begin{cases} -1\,, & s < 0 \\ 0\,, & s = 0 \\ +1\,, & s > 0 \end{cases} \tag{3.70}$$

However, this switching control has a weakness that causes significant chattering in high-frequency switches, lowering the performance of traditional sliding mode control. High order sliding mode control not only provides the same robustness benefits as standard sliding mode control, but it can also remove or reduce chattering while keeping high precision. STSMC is a higher order sliding mode control that avoids chattering while maintaining the other sliding mode properties. It is divided into two parts: a discontinuous sliding variable function and a continuous derivative function. Expressed as

$$\begin{aligned} u_c &= -C_1\sqrt{|s|}\,\mathrm{sgn}(s) + v \\ \dot{v} &= -C_2\,\mathrm{sgn}(s) \end{aligned} \tag{3.71}$$

Where are C_1 and C_2 are positive constant

The super-twisting algorithm is a nonlinear variation of the conventional PI controller [24]. Figure 3.10 show a better explanation of this comparison.

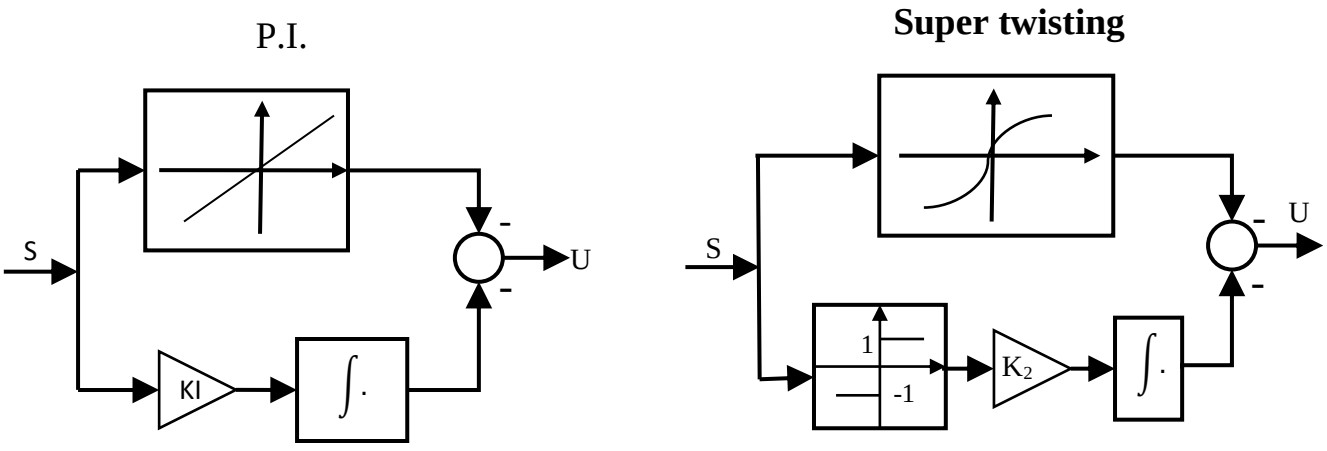


Figure 3.10:- PI (left) and Super-Twisting (right) controller block diagrams.

3.11 Design Super Twisted Sliding Mode Controller (STSMC) for Balancing Scooter

Firstly, we design the sliding mode function (sliding surface) as

$$s = ce + \dot{e} \tag{3.72}$$

Where s is sliding surface, e is error and c must satisfy the Hurwitz condition.

Define tracking error for the pitch angle as

$$\begin{cases} e = x_1 - x_{1ref} \\ \dot{e} = \dot{x}_1 - \dot{x}_{1ref} \\ \ddot{e} = \ddot{x}_1 - \ddot{x}_{1ref} = \dot{x}_2 - \ddot{x}_{1ref} \end{cases} \quad \text{and} \tag{3.73}$$

Where $x_{1ref} = \theta_{ref}$, the reference value of pitch angle.

Therefore, we have

$$\dot{s} = c\dot{e} + \ddot{e} = c(\dot{x}_1 - \dot{x}_{1ref}) + (\dot{x}_2 - \ddot{x}_{1ref}) \tag{3.74}$$

But

$$\dot{x}_2 = f_1(x_1) + f_2(x_1, x_2) + g_1(x_1)U_1 \tag{3.75}$$

$$\dot{s} = c(\dot{x}_1 - \dot{x}_{1ref}) + f_1(x_1) + f_2(x_1, x_2) + g_1(x_1)U_1 - \ddot{x}_{1ref} \tag{3.76}$$

Take this into account as a Lyapunov candidate function

$$V = \frac{1}{2}s^2 \tag{3.77}$$

$$\dot{V} = s\dot{s} \tag{3.78}$$

$$\dot{s}s = s(c(\dot{x}_1 - \dot{x}_{1ref}) + (f_1(x_1) + f_2(x_1, x_2) + g_1(x_1)U_1 - \ddot{x}_{1ref})) \tag{3.79}$$

In order to make the balancing scooter remains on the surface the Lyapunov stability requirement must be fulfilled i.e., to satisfy the condition $\dot{s}s < 0$. Conventional sliding mode control is design as.

$$U_1 = \frac{c(\dot{x}_{1ref} - \dot{x}_1) - f_1(x_1) - f_2(x_1, x_2) + \ddot{x}_{1ref}}{g_1(x_1)} - \eta\,\mathrm{sgn}(s) \tag{3.80}$$

Where c and η are constant

From (3.80) designed controller has two parts as (3.68)

So

$$u_c = \eta \, \mathrm{sgn}(s) \text{ and} \tag{3.81}$$

$$u_{eq} = \frac{c(\dot{x}_{1ref} - \dot{x}_1) - f_1(x_1) - f_2(x_1, x_2) + x_{1ref}}{g_1(x_1)} \tag{3.82}$$

The Super-twisting sliding mode controller [31] [32] [33] is given by:

$$\begin{cases} u_c = -C_1 |s|^{\frac{1}{2}} \mathrm{sgn}(s) + v \\ \dot{v} = -C_2 \, \mathrm{sgn}(s) \end{cases} \tag{3.83}$$

Where C_1 and C_2 are constant

From (3.83) we have

$$u_c = -C_1 |s|^{\frac{1}{2}} \mathrm{sgn}(s) - C_2 \int_0^t \mathrm{sgn}(s) d\tau \tag{3.84}$$

Therefore from (3.82) and (3.84) the proposed super twisted sliding mode controller is given by

$$U_1 = u_{eq} + u_c \tag{3.85}$$

$$U_1 = \frac{c(\dot{x}_{1ref} - \dot{x}_1) - f_1(x_1) - f_2(x_1, x_2) + \ddot{x}_{1ref}}{g_1(x_1)} - C_1 |s|^{\frac{1}{2}} \mathrm{sgn}(s) - C_2 \int_0^t \mathrm{sgn}(s) d\tau \tag{3.86}$$

This designed controller for a two-wheel self-balancing electric scooter is evaluated using MATLAB/Simulink. The parameters of STSMC C1, C2 and c are tuned using GWO, PSO and GA. And STSMC are used to control the balancing of scooters.

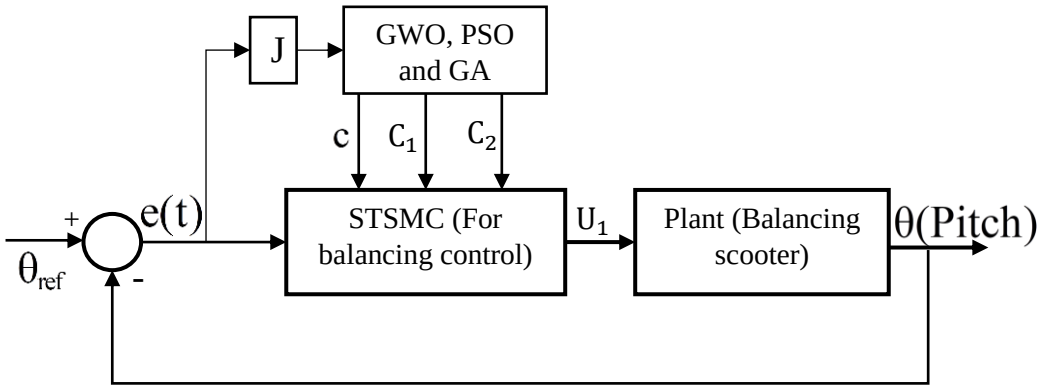


Figure 3.11:- Proposed block diagram for balancing scooter.

3.12 PID Controller

For industrial applications, the PID controller is the most widely used controller. PID controller design schemes are easy and robust in nature. The final PID algorithm is of the form, where "u" represents the controller output.

$$U = K_P e(t) + K_i \int e(t)dt + K_d \frac{d}{dt} e(t) \tag{3.87}$$

Where as K_P , K_i and K_d are proportional, integral and derivative gain respectively.

A proportional controller (K_p) reduces the rise time and lowers the steady-state error, but never completely eliminates stead-state error. While an integral control (K_i) will eliminate the steady-state error, it can increase the transient response. The effect of a derivative control (K_d) is to improve the system's stability, reduce overshoot, and improve the transient response.

Table 3.3:- PID controller parameters effects.

Close loop response	Rising time	Overshoot	Settling time	Steady-state error
K_P	↓	↑	slight change	↓
K_I	↓	↑	↑	Eliminate
K_D	slight change	↓	↓	slight change

3.13 Design PID Controller for Direction Control of Scooter

This PID controller is used for direction control of scooter that means to turn left and right.

$$\begin{cases} U_2 = K_p e(t) + K_i \int e(t)dt + K_d \dfrac{d}{dt} e(t) \\ e(t) = \delta_{ref} - \delta_{ac} \end{cases} \tag{3.88}$$

Whereas δ_{ref} is the reference yaw angle, δ_{ac} is the actual measured value of yaw angle and K_P, K_i and K_d are gains of PID controller.

The parameters of PID controller are tuned using GWO, PSO and GA to get optimal response. And this controller is used to control direction (turn left and right) of scooter.

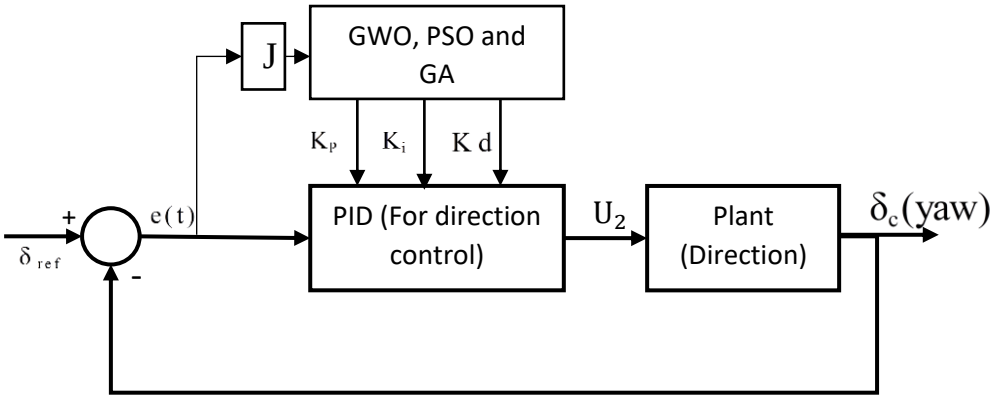


Figure 3.12:- Block diagram of direction control for scooter.

3.14 Objective Function

The most important step in using any optimization technique is to select the objective functions that will be used to assess each population's fitness (particle). These are some of the objective functions.

$$IAE = \int_0^{\tau} |e(t)| dt \tag{3.89}$$

$$ISE = \int_0^{\tau} e(t)^2 dt \tag{3.90}$$

$$ITAE = \int_0^{\tau} t|e(t)| dt \tag{3.91}$$

This two-wheel self-balancing scooter is a MIMO system, so the cost function which minimizes the error is the sum of two cost functions [34]. So, it is rewritten in the form of

$$J = ITAE = \int_0^{\tau} t(|e_1(t)| + |e_2(t)|)dt \tag{3.92}$$

Whereas $e_1(t) = \theta_{ref} - \theta_{ac}$, $e_2(t) = \delta_{ref} - \delta_{ac}$ and θ is pitch angle and δ is yaw angle

The STSMC and PID controllers are used to minimize the error signals, or they can define more carefully, in terms of error criterion, the value of the above-mentioned performance indices should be minimized.

3.15 Introduction to Optimization Algorithms

In this section, GWO, POS and GA are briefly introduced. The STSMC-PID controllers are tuned using the described algorithms, which are used for balancing and direction (turn left or right) control of the scooter.

3.15.1 Grey Wolf Optimization (GWO)

In 2014, Mirjalili et al. introduced the GWO [35] algorithm, a search and tuning approach that simulates the hunting mechanism of grey wolves. The search optimization method is based on grey wolf hunting actions and sorting solutions by group hierarchy, many studies [36] [37] [38] have been published that examine the GWO algorithm in detail. Social hierarchy and hunting behavior of grey wolf packs in the wild are simulated by this method. Grey wolf's lives in a highly organized pack. Alpha, beta, delta and omega wolves are the four various ranks of wolves in a pack.

As show in Figure 3.13, the simulations are Alpha (α) which is as leader, Beta (β), Delta (δ) and Omega (ω) as least weak, hierarchical respectively. The alpha is the leader of the pack and is responsible for making decisions about sleeping, hunting and waking times, among other things. Alpha wolves are followed by the rest of the pack. As a result, it is the hierarchy's highest-ranking grey wolf. In the hierarchy of the grey wolf, Beta is the second-highest ranking member. The betas are wolves who help the alpha with decision-making and other group activities. If one of the alpha wolves dies or becomes too old, the beta wolf is the most likely contender to become the alpha wolf. While controlling the lower-level wolves, the beta wolf must respect the alpha. The beta approves the alpha's directives and provides input to the alpha throughout the pack.

Delta wolves are the third rung of the wolf hierarchy and dominate omega wolves (lowest ranking grey wolf). Scouts, sentinels, elders, hunters, and caretakers are all examples of wolves

in this group. Scouts are in charge of monitoring the territory's borders and warning the pack of any dangers. Sentinels guard and safeguard the pack. Elders are wolves who have previously held the positions of alpha or beta. Hunters help the alphas and betas by hunting animals and supplying food for the pack.

Omega is the grey wolf with the lowest rank. The omega serves as a scapegoat (victim-who is blamed for the mistakes or faults of other). They are the last of the wolves who can eat.

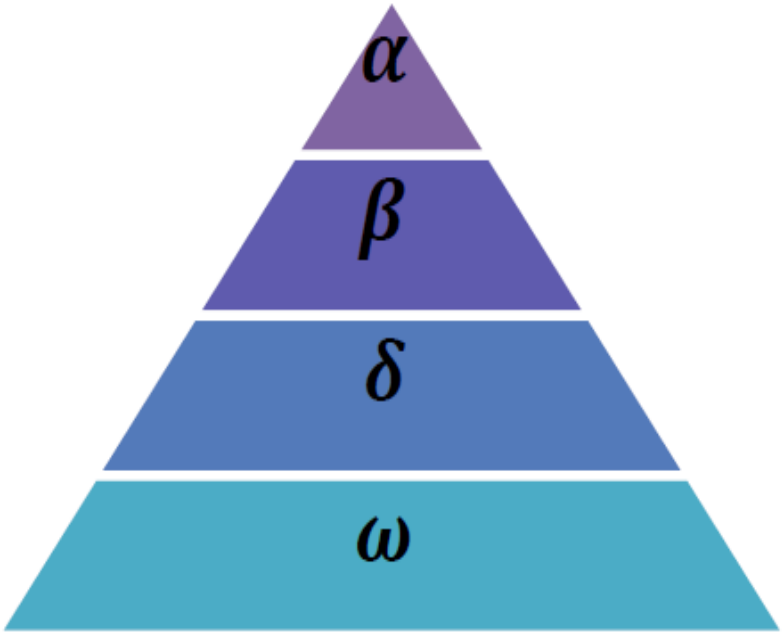


Figure 3.13:- Grey wolf hierarchy [35].

The main step of grey wolfs hunting are as shown in Figure 3.14.

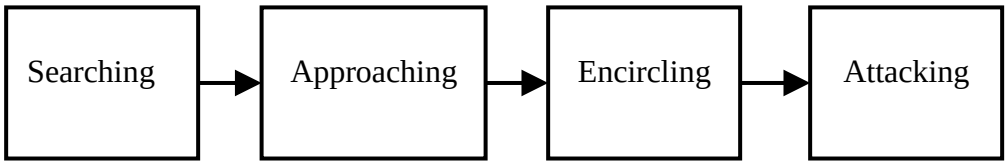


Figure 3.14:- Step of grey wolf hunting.

Mathematical model of algorithm

This section includes mathematical models of the social hierarchy, as well as models of tracking, surrounding, and attacking prey.

a. social hierarchy

When creating GWO, alpha (α) wolf is the fittest solution is used to mathematically model the social hierarchy of grey wolves. The second and third best solutions are beta (β) and delta (δ) wolves, respectively. The remaining candidate solutions are omega (ω) wolf. The GWO algorithm's hunting (optimization) is led by α, β and δ. These three wolves are followed by the omega wolves.

b. Encircling prey

During the hunt, grey wolves encircle their target. The following formulae can be used to model encircling behavior mathematically:

$$\vec{D} = \left|\vec{C}.\vec{X}_p(t) - \vec{X}(t)\right| \tag{3.93}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A}.\vec{D} \tag{3.94}$$

Where t is iteration, X is position of a grey wolf and $\vec{X}_P$ is position of the prey.

In the following equation, the vectors $\vec{A}$ and $\vec{C}$ are determined

$$\vec{A} = 2\vec{a}.\vec{r}_1 - \vec{a} \tag{3.95}$$

$$\vec{C} = 2.\vec{r}_2 \tag{3.96}$$

The value of $\vec{a}$ is linearly decrease from 2 to 0. $\vec{r}_1$ and $\vec{r}_2$, are random vectors in between [0, 1]. By altering the values of $\vec{A}$ and $\vec{C}$, other places surrounding the best search agents can be reached with respect to their current position.

c. Hunting

The first three best solutions obtained thus far are saved to mathematically model the hunting actions of grey wolves, and the other search agents (including the omegas) are obligated to change their positions in accordance with the status of the best search agents. [35] The following formulas are suggested.

$$\vec{D}_\alpha = \left|\vec{C}_1.\vec{X}_\alpha - \vec{X}\right|, \quad \vec{D}_\beta = \left|\vec{C}_2.\vec{X}_\beta - \vec{X}\right|, \quad \vec{D}_\delta = \left|\vec{C}_3.\vec{X}_\delta - \vec{X}\right| \tag{3.97}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1.(\vec{D}_\alpha), \quad \vec{X}_2 = \vec{X}_\beta - \vec{A}_2.(\vec{D}_\beta), \quad \vec{X}_3 = \vec{X}_\delta - \vec{A}_3.(\vec{D}_\delta) \tag{3.98}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{3.99}$$

The search method begins with a random population of grey wolves generated by GWO (candidate solutions). Iteratively, the $\alpha, \beta, \text{and } \delta$ e wolves determine the probable position of the prey. Each candidate solution's distance from the prey is updated.

d. Attacking prey

When the prey stops moving, grey wolves finish the hunt by attacking it, as previously stated. We reduce the value of $\vec{a}$ to mathematical model approaching the prey. The $\vec{a}$ value also decreases the fluctuation range of $\vec{A}$. When the number of iterations $\vec{A}$ of drops from 2 to 0, it is said to be at random in the interval $[-a, a]$. Candidate solutions tend to diverge from the prey when $|A| \geq 1$ and converge towards the prey when $|A| < 1$. Finally, $|A| < 1$ forces the wolves to assault the prey.

e. Search for prey

Grey wolves primarily search based on the $\alpha, \beta, \text{and } \delta$ positions. They split apart to look for prey and then converge to attack it.$\vec{A}$ is employed with random numbers higher than or less than 1 to drive the search agent to diverge from the prey in order to mathematically describe divergence. An optimization algorithm that favors exploration and avoids local optima, GWO is able to act in an uncontrolled way throughout the entire process of optimization. In contrast to $\vec{A}$, $\vec{C}$ z is not linearly lowered. To emphasize exploration/exploitation not only during initial iterations, but also during final iterations, GWO requires that C always provide random values at all times.

Local optima stagnation is a common problem, especially in the last iterations. This component can be of great help in these situations. Figure 3.15 shows the pseudo code for the GWO algorithm.

Initialize population

Initialize a, A, and C

Calculate fitness value. %depending on objective function

Xα=1st fitness

Xβ=2nd best

Xδ=3rd best

***While** (j < final iterations)*

***for** each search agent*

Update the position current search agent's by (3.99)

end

Update C, a, and A

Calculate fitness value. % depending on objective fun. (ITAE)

Update Xα, Xβ, and Xδ

j=j+1

end

***Return** Xα as optimal value of controller (STSMC and PID)*

Figure 3.15:- The GWO algorithm's pseudocode.

3.15.2 Particle Swarm Optimization (PSO)

Based on the social behavior of birds or fish, PSO is a well-known algorithm for optimizing a system. It is improved by Eberhart and Kennedy in 1995. In recent years, In a wide range of applications, the PSO algorithm has been used, including mathematics difficulties, theoretical optimization problems, and highly specialized engineering domains [39] [40] [41].

Algorithm for locating positions or places in search space that are near to the global minimum or maximum solution (s). Because there are a large number of parameters that must be optimized, there is a large amount of search space. For example, if the search space has n dimensions, the variable number of the objective function will be n as well. Algorithm PSO has a lot of parameters. The fitness value at that location is calculated using each particle's current position. Position $x^i \in R^n$, velocity, i.e. rate of position change (v^i), and previous best positions $(p^i)$are the three parameters for each particle. Furthermore, the global best location refers to the position of the particle with the best fitness value.

In the search space, each particle is represented by a set of coordinates denoted by xi. The fitness function is used to evaluate the present positions of all particles during the search process. Following that, the fitness value of each particle is compared to the current position, and the best position is saved in the previous best positions (p^i). To put it another way, the previous best positions also save positions of particles with better values. Update the velocity of the particle and particle position (v^i, x^i) was determined by equation below.

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (p_i^t - x_i^t) + c_2 r_2 (g_i^t - x_i^t) \quad (3.100)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad i = 1, 2, ..., n \quad (3.101)$$

Where as

p^i: - Best position of particle

g^i: - Global best

Whereas t is the number of iteration, c_1 and c_2 are the positive constants, ω is the weight inertia and r_1 and r_2 are two random numbers which are changing in the range [0, 1]. PSO algorithm pseudocode is shown in Figure 3.16 , below.

Initialize parameter

Initialize randomly position (xi)

Initialize randomly velocity (Vi)

Set pbest and gbest in particle

While *(t < final iterations)*

Calculate fitness value of new particle xi using objective function (ITAE)

If *xi is best than pbest*

Set xi as pbest

end

If *xi is best than gbest*

Set x_i *as gbest*

end

for *i: No. Population*

Update (x^i) *and (Vi) of particle by using equation* (3.100) *and* (3.101)

end

t=t+1

end

Return *gbest as optimal value of controller (STSMC and PID)*

Figure 3.16:- Pseudo code of the PSO algorithm.

3.15.3 Genetic Algorithm (GA)

This type of algorithm uses mutation, inheritance, crossover, and selection, methods inspired by evolutionary biology in order to find solutions to problems. GA is a type of searching algorithm. It looks for an optimal solution to a problem in a solution space. The method of searching is a key feature of the GA [42] [43]. The algorithm generates a "population" of possible solutions to the problem and allows them to "evolve" over time to find better and better ones.

The population is a set of candidate solutions that the algorithm considers. Over the generations of the algorithm, new members are "born" into the population, while others "die" out. The term "individual" refers to a single solution in a population. An individual's fitness is a measurement of how "good" the solution that the individual represents is. The higher the fitness, the better the solution, of course, it is based on the problem to be solved.

In the natural world, the survival of the fittest is similar to the selection process. Individuals are chosen for "cross-over" (breeding) based on their fitness values. The more fit an individual is, the more likely the individual will be able to reproduce. The cross-over happens when the two solutions are mixed to create two new individuals. Everyone has a small chance of mutating during each generation, which will modify the individual in some way.

The algorithm ends when the population has attained a suitable fitness level or a maximum number of generations has been produced. If the method has finished due to the maximum number of generations, a good solution may or may not have been found [44]. Genetic optimization algorithm is used in matlab either in code or GA optimization toolbox app.

The pseudo code of the GA is presented in Figure 3.17.

```
Initialize parameter
Create initial population
While (t < Max number of iterations)
        Calculate the value of fitness for individual in the population using (ITAE)
    For i: max iteration
        Select individuals for the new population –selection
        Perform cross-over operation
        Perform mutation of individual
        Evaluate individuals
        Replace old population with the new one
    end
 t=t+1
end
```

Figure 3.17:- Pseudo code of the GA.

3.16 Overall System Structure

The two-wheel self-balancing scooter mathematical model is designed based on Newton's 2nd law of motion. This designed model is a nonlinear and unstable system. In order to stabilize this system, it needs an appropriate controller design like STSMC and a PID controller. As shown in Figure 3.18 below, the system is a MIMO (two input, two output) system. The 1st input is the pitch angle, which is used for balancing the scooter. This angle is compared with the reference angle and fed to the controller (STSMC) to balance the vehicle. The other input angle is the yaw angle used for direction control (turn left or turn right) of the scooter. This output angle is compared with the reference angle and fed to the controller (PID). These two controllers are tuned using optimization algorithms (GWO, PSO, and GA) in order to produce a satisfactory result. The controller output passes through a decoupling unit and is applied to the DC motor (actuator) to produce appropriate torque that derives the scooter's balance (move forward or backward) or direction control (turn left or right).

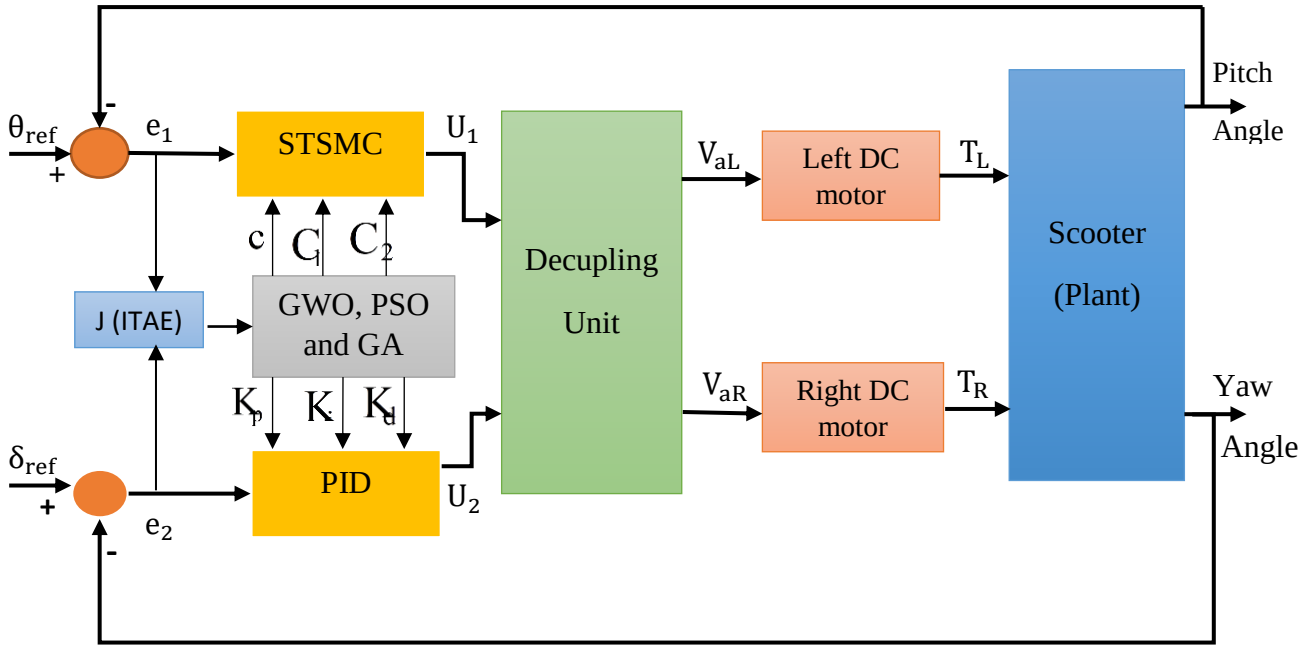


Figure 3.18:- Overall system structure.

The following are the primary characteristics of the proposed Scooter controller:

a. Scooter balancing controller is used to keep the scooter balanced at a certain pitch angle $\theta = 0$ rad. In this paper, STSMC control is applied to design a balancing controller (move

forward or backward) for the Scooter. Way of moving backward and forward shown in Figure 3.19.

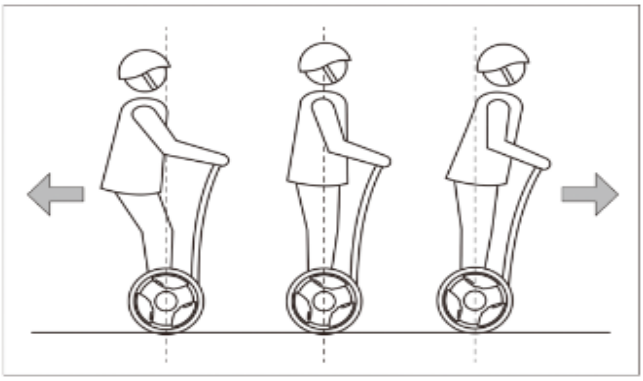


Figure 3.19:-Riding Forward and Backward [5].

b. A left and right-turning controller (direction controller) is used to control the Scooter in the left and right directions. A PID controller is used to design a left and right-turning controller for the Scooter in this research. Way of turning left and right of scooter shown in Figure 3.20.

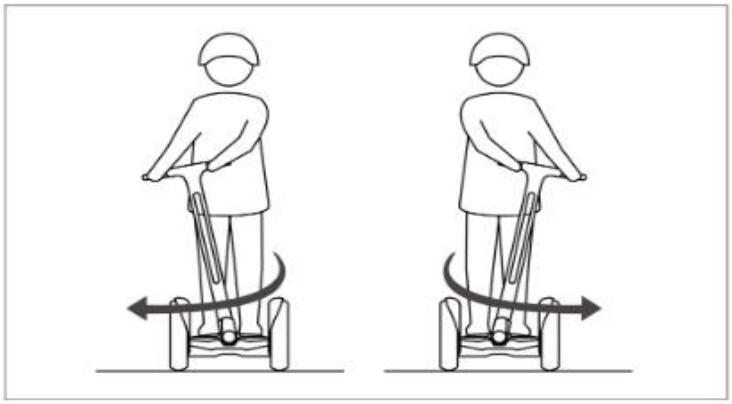


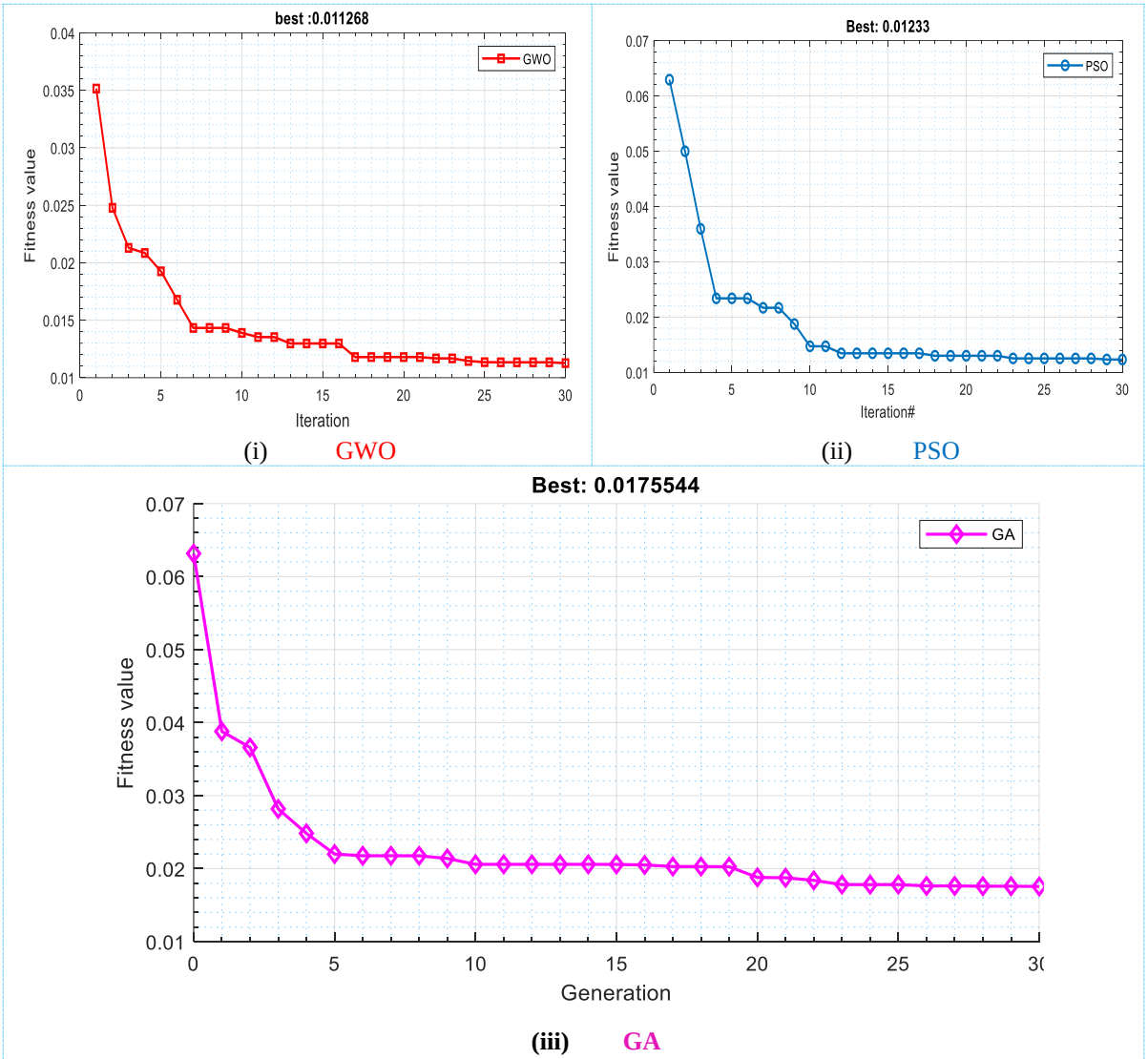
Figure 3.20:- Turning Left and Right way [5].

To balance (stabilizing) and direction control for the scooter, the parameters of the STSMC and PID controller are tuned by using grey wolf optimization (GWO), particle swarm optimization (PSO) and genetic algorithm (GA). Tuning the parameters of controllers using these three methods helps to compare which tuning algorithm is best for this specified application. This algorithm is compared in terms of fitness value, computation time, and so on. The Figure 3.21 below shows the convergent curve of this three-optimization algorithm.

Table 3.4: - Boundary of tuned parameters of controllers.

Boundary	STSMC			PID		
	C	C1	C2	Kp	Ki	Kd
Lower bound (L_b)	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
Upper bound (U_b)	15	15	15	600	600	600

The GWO, PSO and GA are employed for adjusting of the control parameters with respect to the proposed cost function (J). The tuning ranges of the parameters are set as denoted in Table 3.4 and in Table 3.5, the parameters of the proposed algorithms are given.



(i) GWO (ii) PSO

(iii) GA

Figure 3.21:- Convergent curve of GWO, PSO and GA.

Table 3.5:- Parameters of proposed algorithm.

Algorithm	Parameters of Algorithms		
GWO	Number of Search Agent		20
	Maximum Iteration		30
PSO	Number of Particle (No. P)		20
	Maximum Iteration		30
	Inertia weight	ω_{init}	0.2
		ω_{final}	0.9
	Personal best value (c1)		2
	Neighborhood best value (c2)		2
GA	Population size		20
	Generation		30

Figure 3.22 shows the convergence performances of the GWO, PSO, and GA. This graph clearly shows that GWO is better in both the start and the end. The PSO and GA converge at a slower rate than the GWO. As a result, the GWO outperforms the PSO and GA in terms of response by reducing the cost function more.

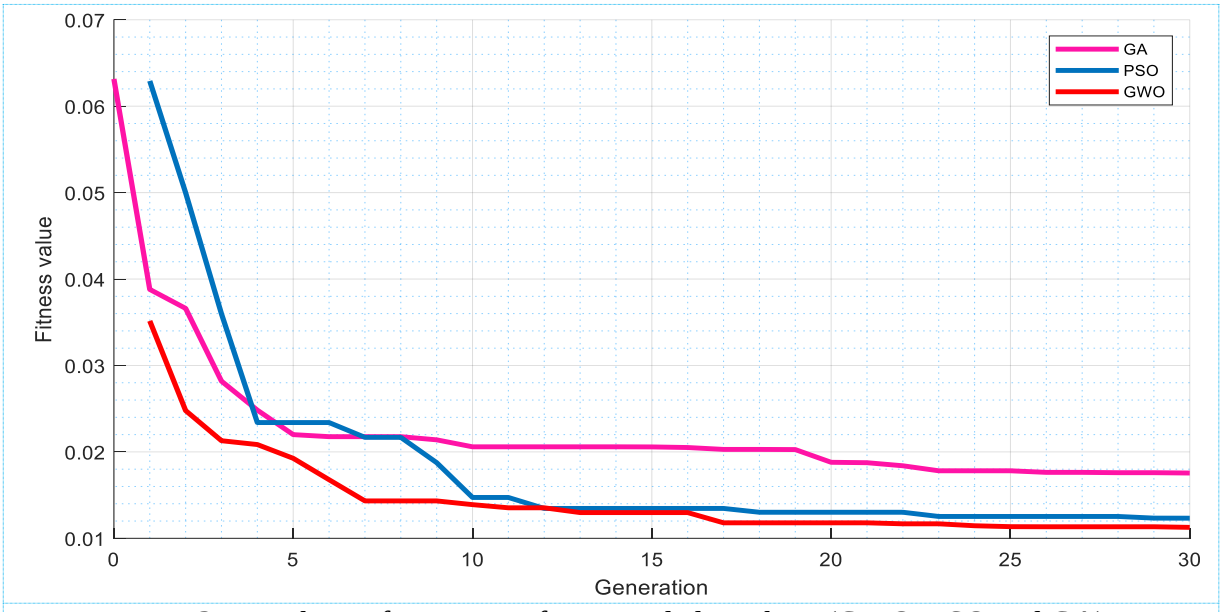


Figure 3.22:- The performances of proposed algorithms (GWO, PSO and GA).

The proposed algorithms are applied to the STSMC and PID controllers. Firstly, GWO, PSO and GA are exported to MATLAB by coding. The proposed algorithms are tuned for 30 iterations to get the minimum value of the cost function (J). Table 3.6 shows the optimal values of STSMC and PID parameters after tuning, corresponding to the minimum cost function value provided by the optimal GWO, PSO, and GA. Moreover Table 3.6, are also shown the minimum cost function value (Jmin) and the elapsed time of optimizations.

Table 3.6:- Optimal value of STSMC and PID controller.

Controller parameter		GWO	PSO	GA
STSMC	C	1.19631	0.8918425	6.21169
	C1	1.058693	4.526697	14.98673
	C2	10.46597	7.404644	14.9664
PID	Kp	598.3769	452.4186	535.2570
	Ki	587.7062	435.4414	332.4875
	Kd	61.80542	62.78395	94.3456
Best value (Jmin)		0.011268	0.01233	0.01755
Elapsed time		786.248511 sec	885.102937 sec	1149.346 sec

In terms of elapsed time (time taken to execute the code) and best value, the GWO is better than PSO and GA. This elapsed time (competition time) depends on the window, processor of the PC, MATLAB version, step time, and so on. So, in this work window is window 10, the possessor is 2.5, the MATLAB version is R2017a and for step size of 5sec.

CHAPTER FOUR

4. RESULTS AND DISCUSSIONS

4.1 Introduction

The simulation was performed based on the parameters of the scooter in Table 3.1 and Table 3.2 STSMC is used for balancing the scooter and the PID controller is used for direction control (turn left and turn right). The parameters of these two controllers are tuned using GWO, PSO and GA in order to produce good performance of the system. MATLAB/Simulink of two-wheel electric scooters is designed as shown in Figure 4.1.

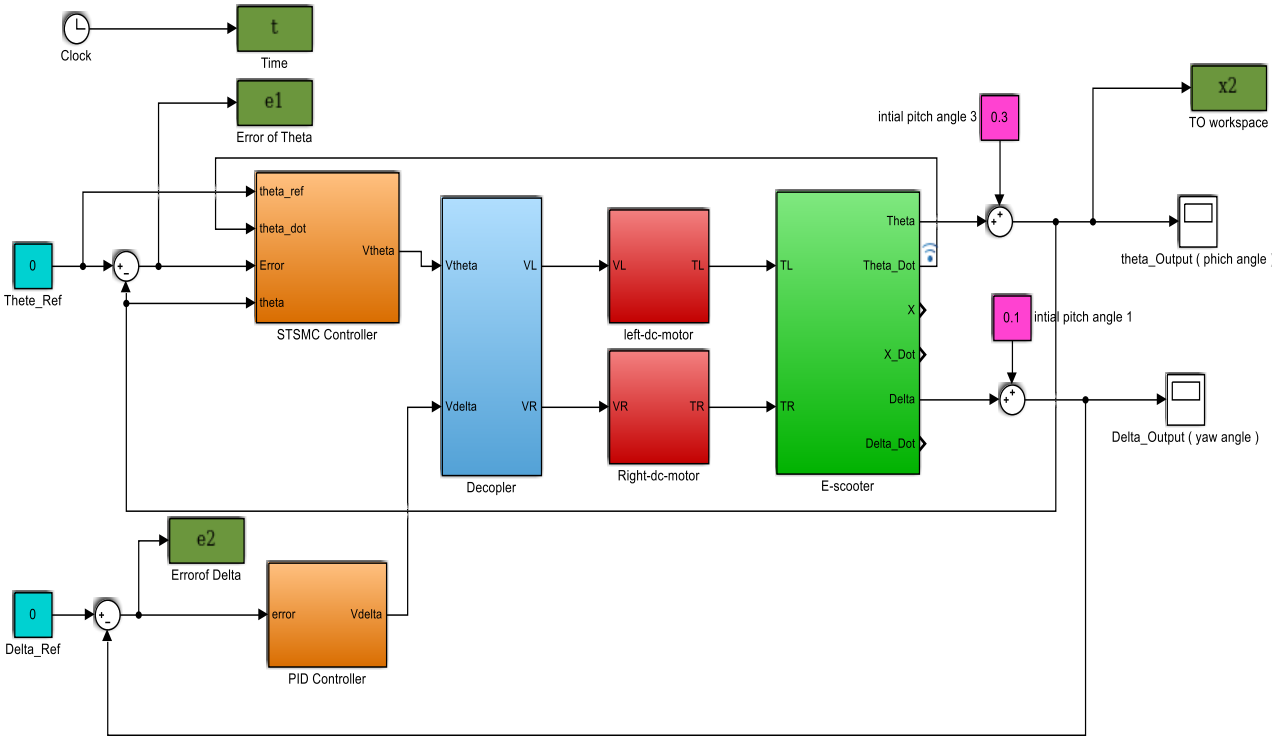


Figure 4.1:- MATLAB/Simulink model of two-wheel self-balancing electric scooter

Some discerption of the MATLAB/Simulink model in Figure 4.1.

a) The orange color shows the designed controllers used in this work (i.e., STSMC and PID).
b) The red color shows the actuator of the system (for the right and left wheel).
c) The light blue color shows the decoupling unit.

d) The Green color shows the plant (scooter).
e) The Cyan color shows the two-reference angle for balancing and direction control (pitch and yaw angle).
f) The Magenta color shows the initial pitch and yaw angle.

A self-balancing scooter (or scooter) is a highly nonlinear system and is unstable. Without any controller, the response of the plant (scooter) is not stable, as shown in Figure 4.2. The response of each output path to the tilt angle for balancing and direction control.

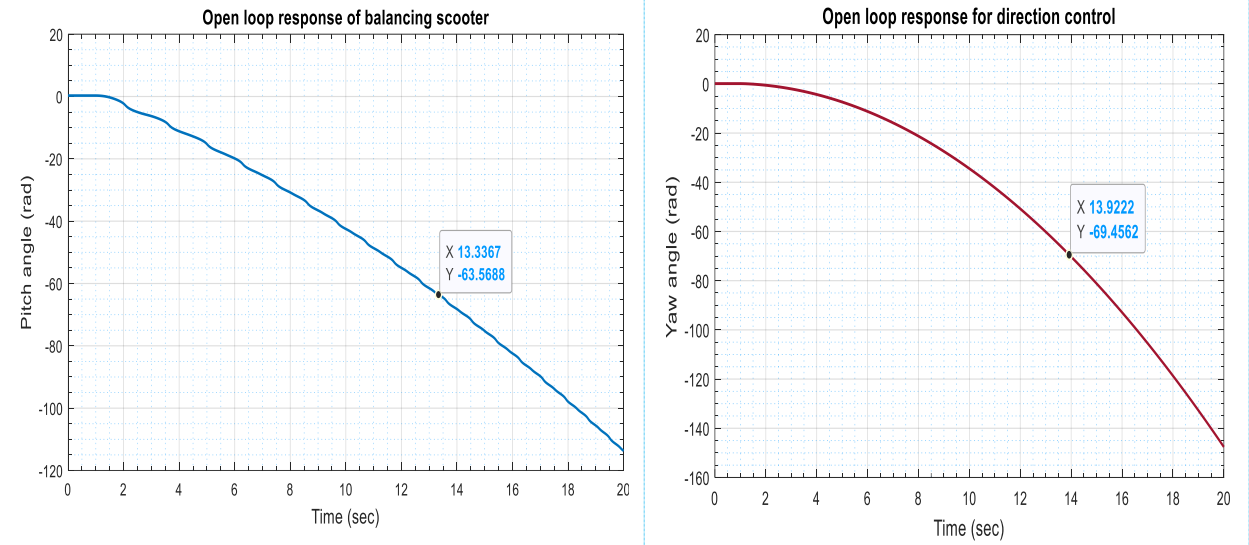


Figure 4.2:- Open loop response of scooter for balancing and direction.

The system is unstable (very rapid increase in output due to small input signal), thus it needs a robust controller (like STSMC). The scooter controller design is created by combining the self-balancing controller with the left and right-turning controllers.

4.2 Simulation Results

The parameters of the controller (STSMC and PID) are tuned (optimized) using three optimization algorithms (GWO, PSO, and GA). The comparison of GWO, PSO, and GA algorithms for STSMC and PID controllers is shown in Figure 4.3 and Figure 4.4. With an initial pitch angle of for balancing and a yaw angle for direction control (turn left and right). The result shows that GWO has better response than PSO and GA optimization algorithms in terms of time domain specification, as shown in

Table 4.1 below. For both balancing and direction control responses, there is 0% steady state error. This result shows that a GWO based STSMC-PID is good at balancing and direction control of two-wheel electric scooters.

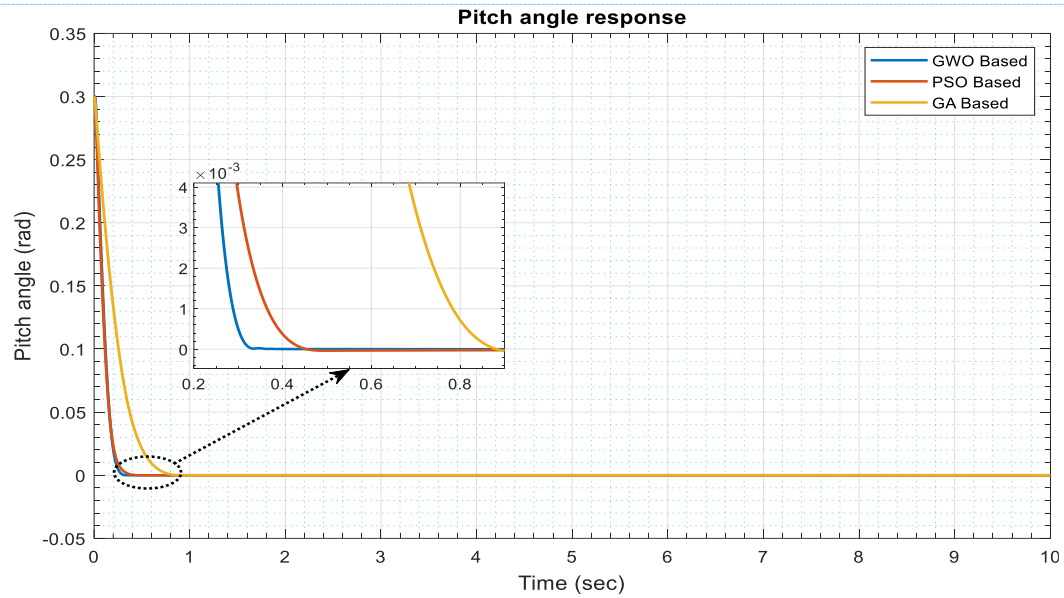


Figure 4.3:- Pitch angle response with initial θ = 0.3 rad for GWO, PSO and GA optimization algorithm.

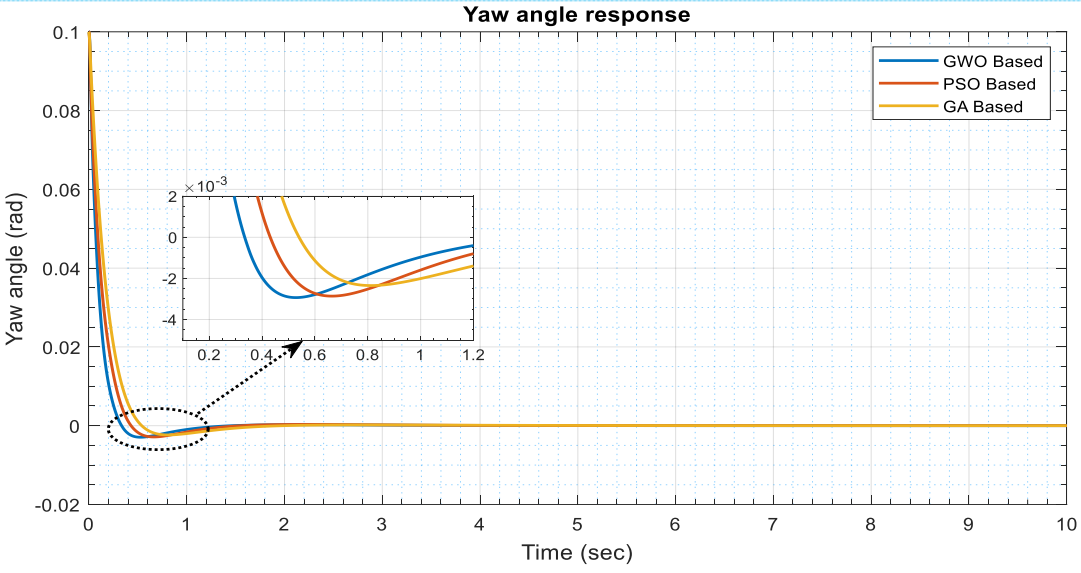
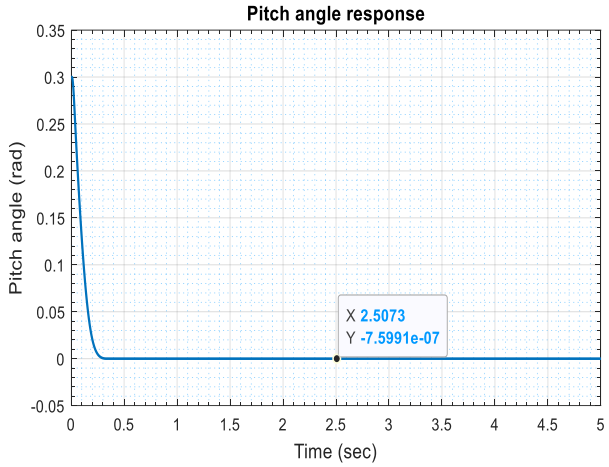


Figure 4.4:- Yaw angle response with initial δ = 0.1 rad for GWO, PSO and GA optimization algorithm.

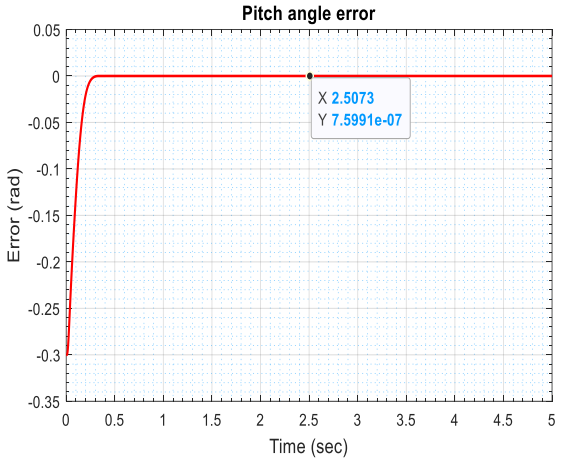
Table 4.1:- GWO, PSO and GA algorithms comparison of system performance.

Algorithm	Pitch angle at θ = 0.3 rad		Yaw angle at δ = 0.1 rad	
	Rise Time(sec)	Settling Time(sec)	Rise Time(sec)	Settling Time(sec)
GWO Based	0.1542	0.2438	0.1893	0.7641
PSO Based	0.1614	0.2750	0.2607	0.9133
GA Based	0.4146	0.6490	0.3198	1.0030

Figure 4.5 shows the response of the scooter with an initial pitch angle (θ = 0.3 rad), and Figure 4.6 shows the response of the scooter with an initial yaw angle (δ = 0.1 rad). The pitch and yaw angle of the scooter converge to zero (reference value) in two conditions. For both pitch angle and yaw angle responses, the corresponding errors show that as there is no steady state error (0%). These results demonstrate that the scooter can keep balance well and direction control is done in a perfect manner.

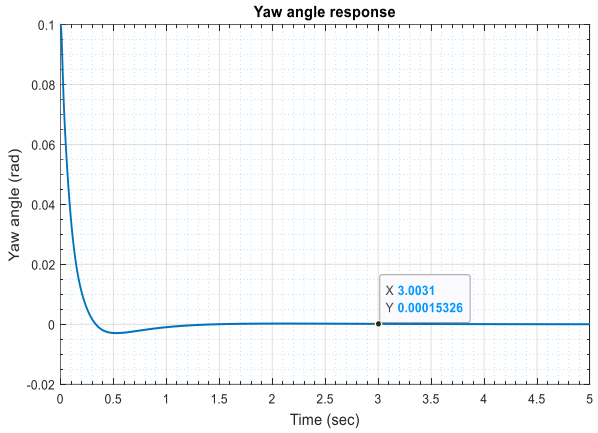


(i) Pitch angle response

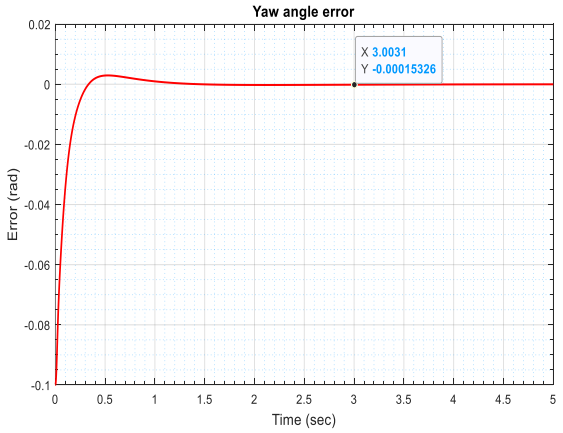


(ii) Error

Figure 4.5:- Pitch angle response with initial θ = 0.3 rad and the corresponding error with reference θ = 0 rad.



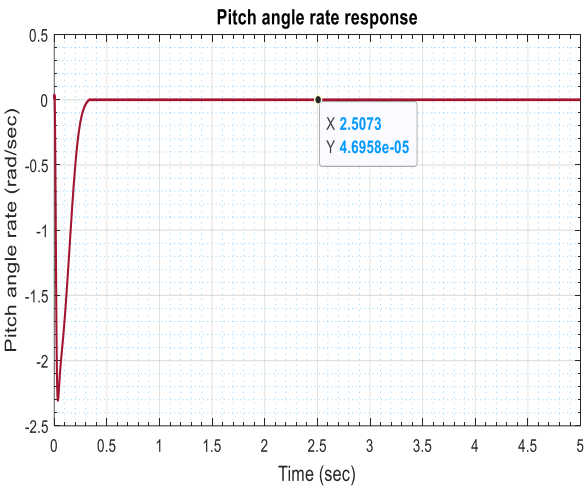
(i) yaw angle response



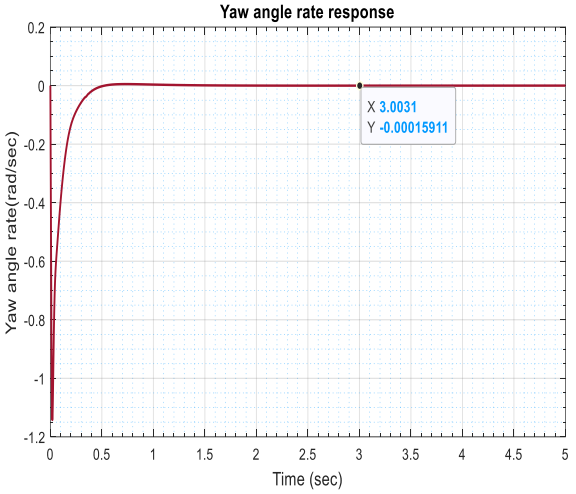
(ii) error

Figure 4.6:- Yaw angle response with initial $\delta = 0.1$ rad and the corresponding error with reference $\delta = 0$ rad.

The equivalent pitch angle rate at $\theta = 0.3$ rad and yaw angle rate at $\delta = 0.1$ rad show in the Figure 4.7. Angle rate of the scooter converge to zero for the two conditions.



(i) pitch angle rate response

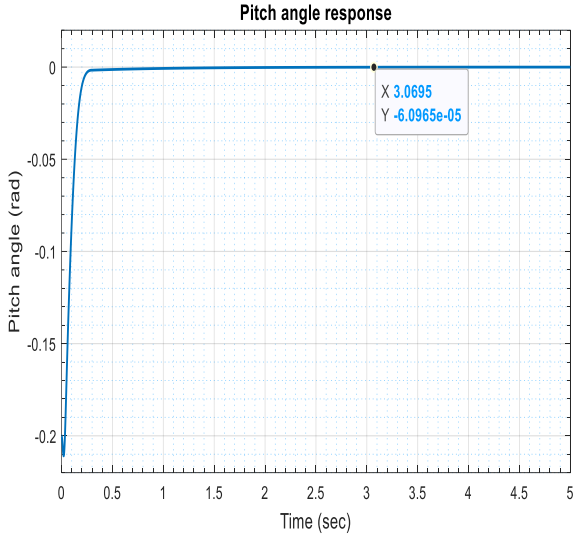


(ii) Yaw angle rate response

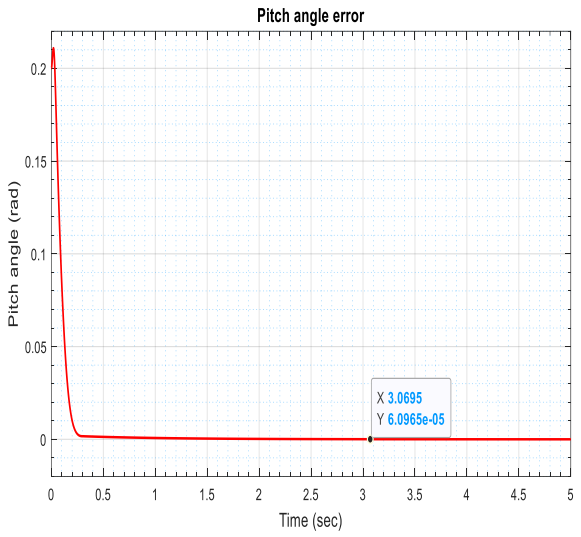
Figure 4.7:- Pitch angle rate response for initial $\theta = 0.3$ rad and yaw angle rate response for initial $\delta = 0.1$ rad.

Figure 4.8 shows the response of the scooter to another initial pitch angle $\theta = -0.2$ rad, and Figure 4.9 shows the response of the scooter to an initial yaw angle at $\delta = -0.15$ rad. The pitch angle and yaw angle of the scooter converge to zero in two conditions. These results also

demonstrate that the scooter can keep balance well and the direction change of the scooter is done in a perfect manner for other variations of initial pitch and yaw angle.

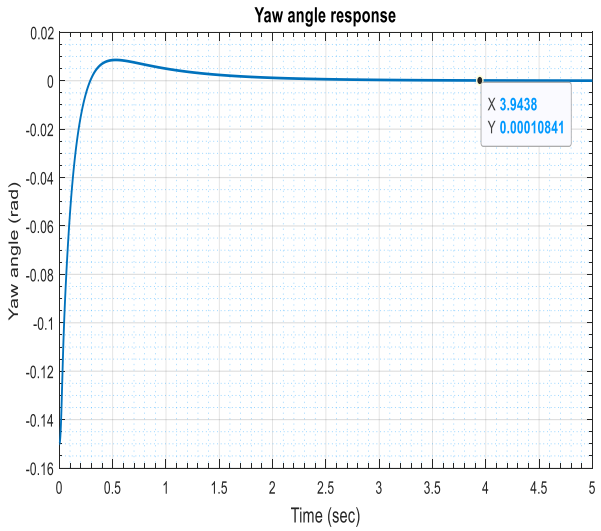


(i) Pitch angle response

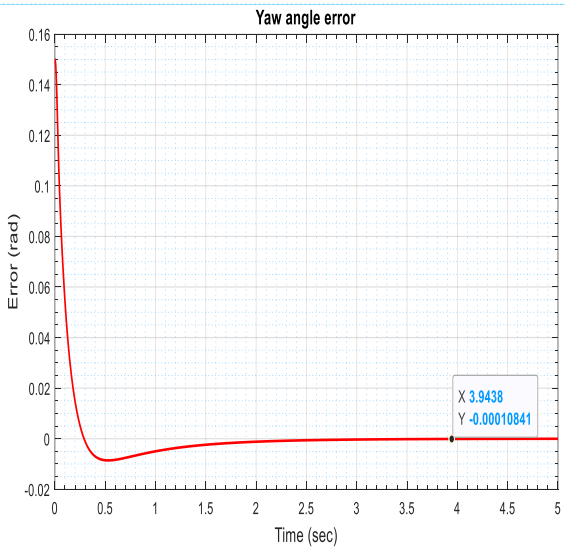


(ii) Error

Figure 4.8:- Pitch angle response with initial θ= -0.2 rad and the corresponding error with reference θ = 0 rad.



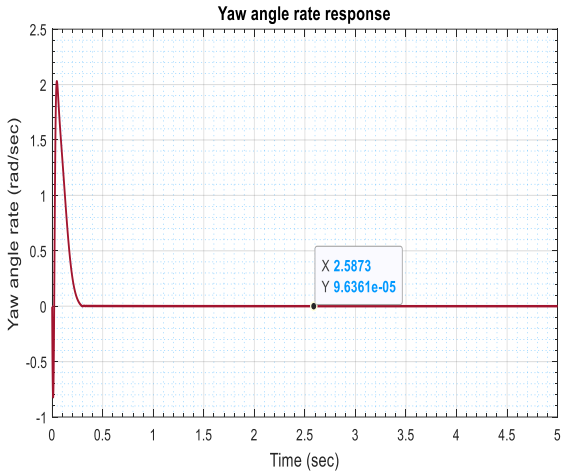
(i) Yaw angle response



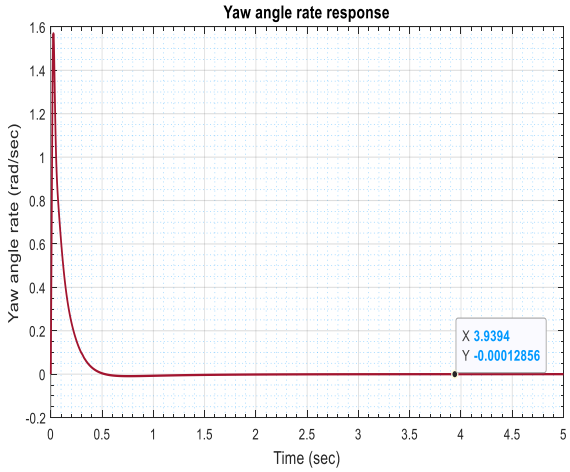
(ii) Error

Figure 4.9:- Yaw angle response with initial angle of δ = -0.15 rad and the corresponding error with reference δ = 0 rad.

The equivalent pitch angle rate at θ= -0.2 rad and yaw angle rate at δ = -0.15 rad show in the Figure 4.10 below angle rate of the scooter converge to zero in two conditions.



(i) pitch angle rate response



(ii) Yaw angle rate response

Figure 4.10:- Pitch angle rate response for initial θ = -0.2 rad and yaw angle rate response for initial δ = -0.15 rad.

In this work, a scooter can carry a human load of up to 90 kg. Figure 4.11 shows the response to human load varying from 50kg to 90kg. The mass of 90kg has a rising time of 0.1542sec and a settling time of 0.2438sec for initial pitch angle θ = 0.3 rad and has a smaller rising time and settling time than 75kg, 65kg, and 50kg loads.

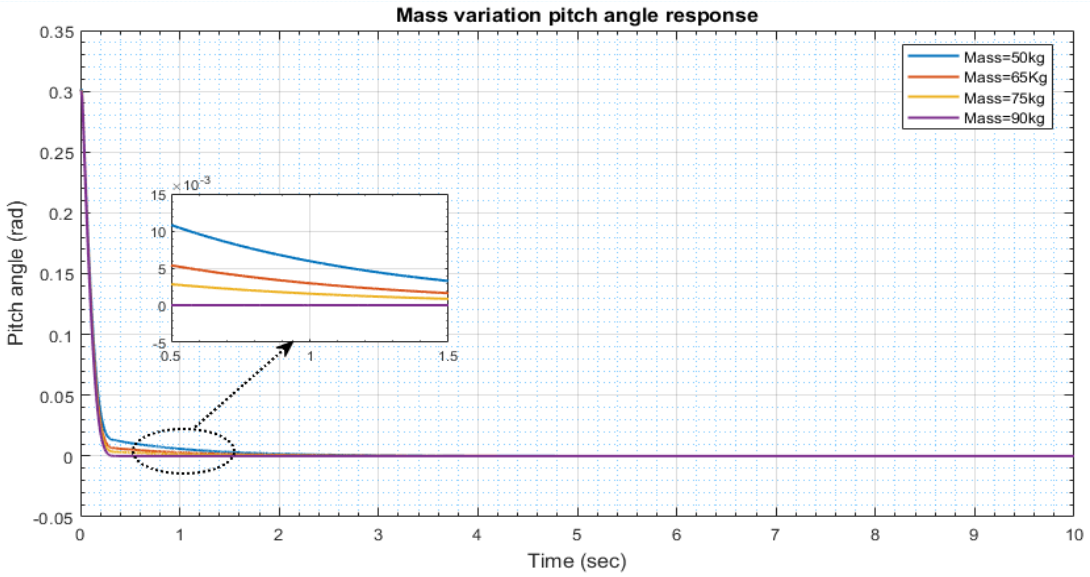


Figure 4.11:- Pitch angle response with initial θ = 0.3 rad for mass variation from 50kg to 90kg.

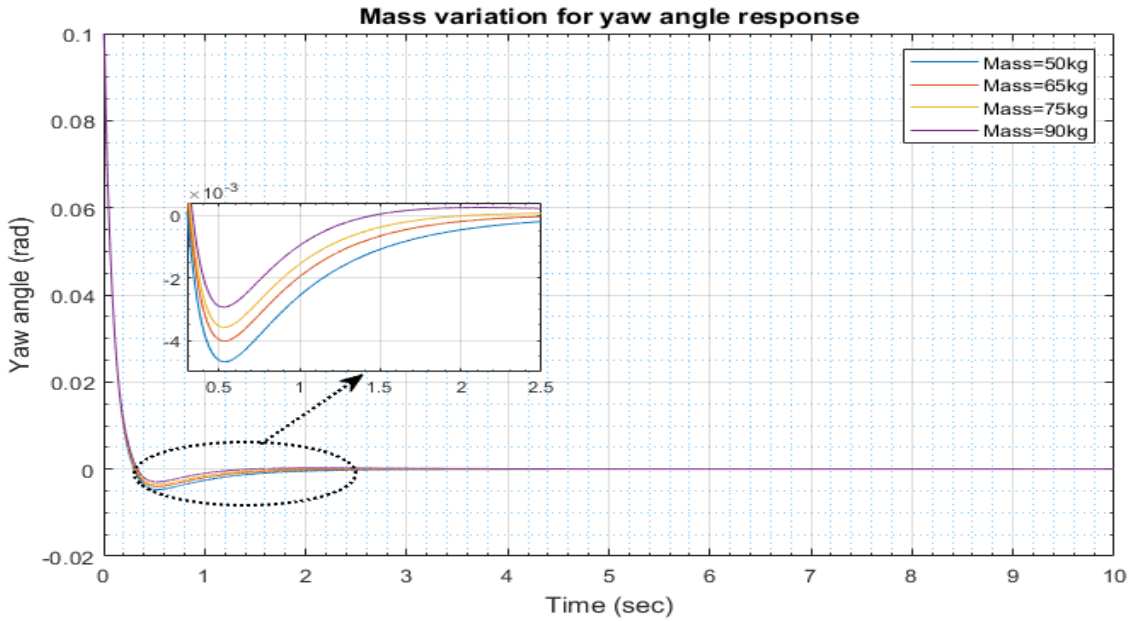


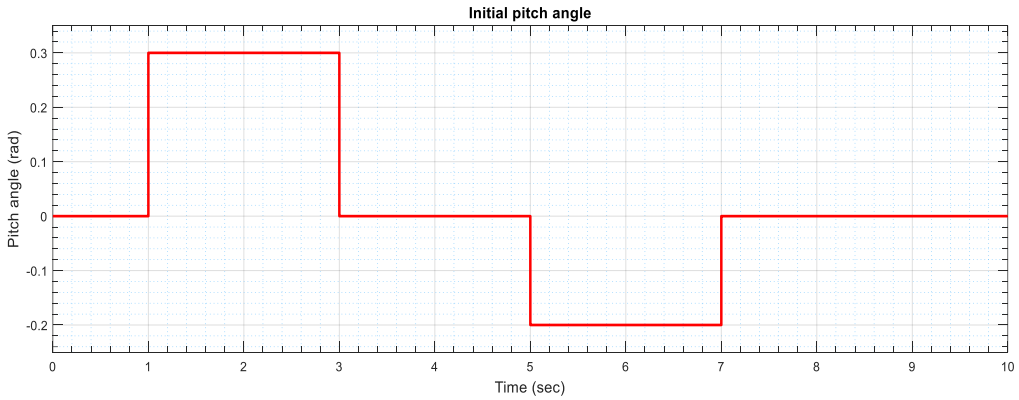
Figure 4.12:- Yaw angle response with initial δ = 0.1 rad for mass variation from 50kg to 90kg.

Figure 4.12 shows the response to a human load varying from 50kg to 90kg. The mass of 90kg has a rising time of 0.1893sec and a settling time of 0.7641sec for an initial yaw angle of δ = 0.1 rad and has a smaller rising time and settling time than 75kg, 65kg, and 50kg loads. The results show that for different mass variations, the scooter balance and direction control have good dynamic performance for initial pitch angle θ = 0.3 rad and for yaw angle δ = 0.1 rad. One of the benefits of using sliding mode control is that it has low sensitivity to plant parameter variations, like the mass of humans.

Table 4.2:- For human load variation, comparison of system performance.

Human load	Pitch angle at $\theta = 0.3rad$		Yaw angle at $\delta = 0.1rad$	
	Rise Time(sec)	Settling Time(sec)	Rise Time(sec)	Settling Time(sec)
50Kg mass	0.1838	0.9865	0.1839	1.1445
65kg mass	0.1674	0.4044	0.1857	0.9891
75kg mass	0.1608	0.2707	0.1870	0.8944
90kg mass	0.1542	0.2438	0.1893	0.7641

4.3 Simulation Result for Pitch and Yaw Angle Variation



(a) Initial Pitch angle

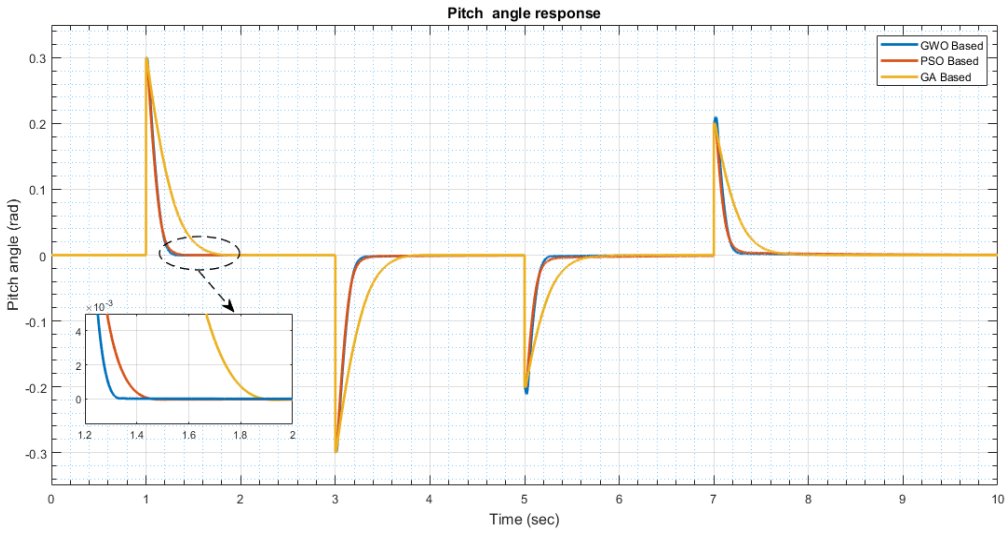
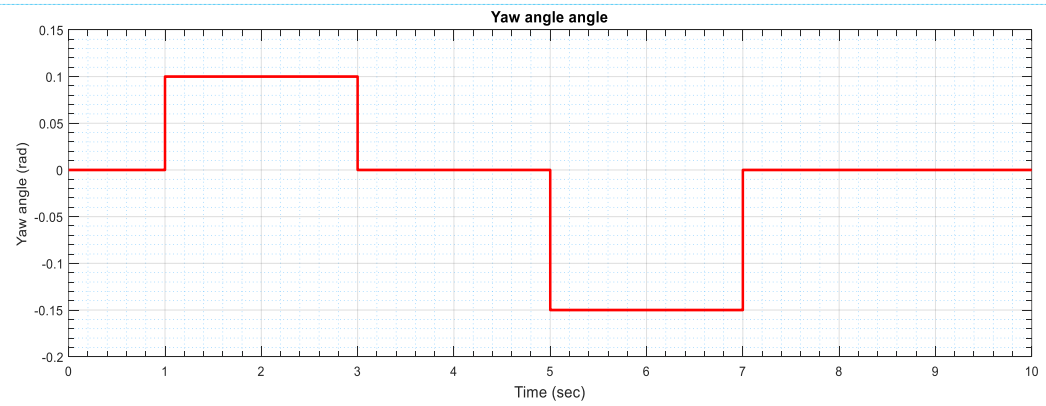


Figure 4.13:- Pitch angle response for initial pitch angle (a)

Figure 4.13 show Pitch angle response for the initial pitch angle $\theta = 0.3(u(t-1) - u(t-3)) \; -0.2(u(t-5) - u(t-7))$ as show in Figure 4.13(a) and the results demonstrate that the scooter can keep balance (stabilize) well for different pitch angle change.



(b) Intial Yaw angle

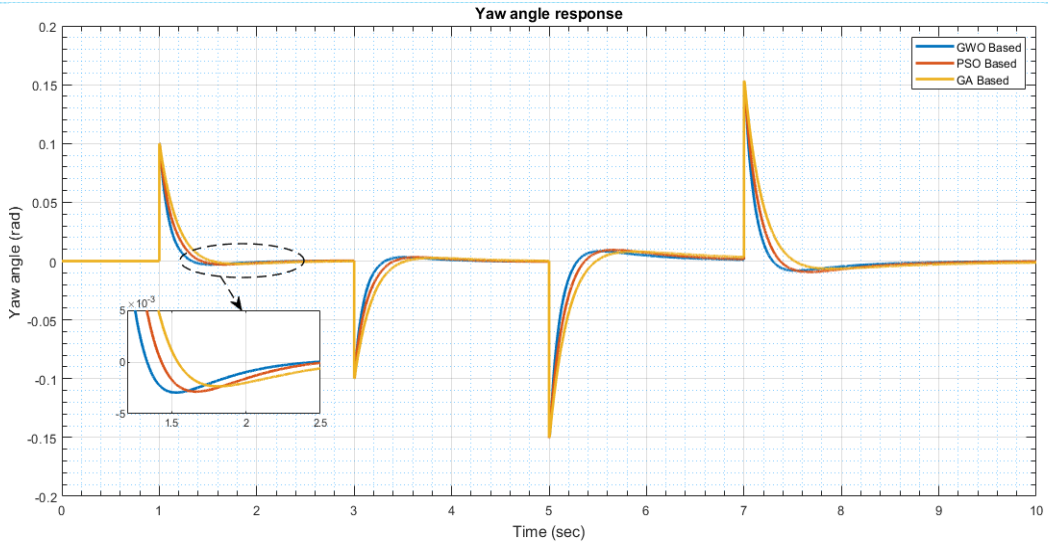
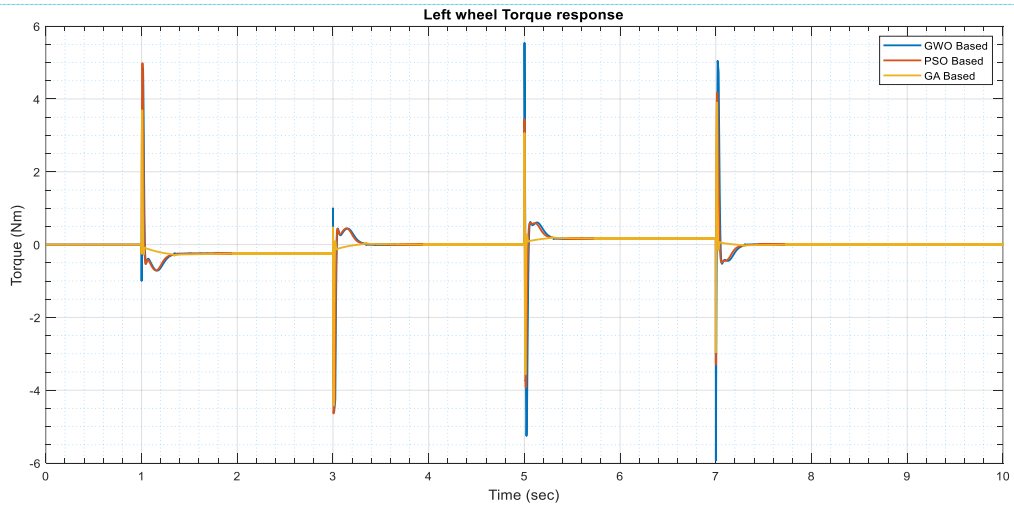
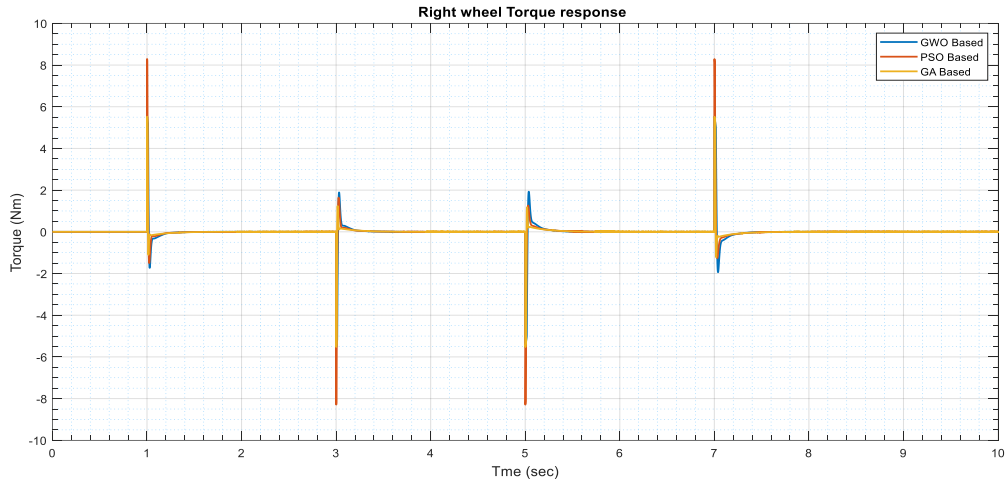


Figure 4.14:- Yaw angle response for initial yaw angle (b).

In the same way Figure 4.14 show the yaw angle response, for initial yaw angle of $\delta = 0.1(u(t-1) - u(t-3)) - 0.15(u(t-5) - u(t-7))$ in the Figure 4.14 (b) and the result shows that scooter direction control is done in the perfect manner for different yaw angle change.



(c) Left wheel torque



(d) Right wheel torque

Figure 4.15:- Left wheel (c) and Right wheel (d) Torque response for intial Pitch angle (a) and Yaw angle (b).

Figure 4.15 show the left wheel and right wheel torque produced to balancing and direction control of scooter for a given initial pitch angles of Figure 4.13 (a) and yaw angles of Figure 4.14 (b).

In general, two wheel self-balancing scooter efficiently controlled through initial pitch and initial yaw angle, and all simulation result shows Scooter can operate in stably and with good performance.

5. CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

A self-balancing scooter is a highly nonlinear and unstable system. Without any controller, the response of the plant (scooter) is not stable (very rapid increase in output due to small input signal) as shown in the result of open loop response.

In this paper, a super twisted sliding model controller (STSMC) method has been proposed for self-balancing control of a scooter, and PID control has been proposed for direction control (turning left and right). Scooter parameters like mass of body, mass of wheel, radius of wheel and DC motor parameters are identified for mathematical design of the scooter in MATLAB/Simulink.

The controller used in this work (STSMC and PID) parameters are tuned using grey wolf optimization (GWO), particle swarm optimization (PSO) and genetic algorithm (GA). These optimization algorithms are compared in terms of competition (elapsed) time, fitness value, and this result shows that GWO has a small elapsed time and good fitness value when compared with PSO and GA.

The proposed STSMC and PID control tuned using GWO are good at balancing and direction control of the scooter when compared with PSO and GA as shown in the simulation results. In this work, the scooter can carry a human load of up to 90 kg. The simulation results show that the scooter balance and direction control is done in a perfect manner for initial pitch angle $\theta = 0.3 \text{ rad}$ and for yaw angle $\theta = 0.1 \text{ rad}$. Finally, simulation results demonstrate that a two-wheel self-balancing electric scooter based on the STSMC-PID controller using GWO has good performance.

5.2 Recommendations

Nowadays, transportation is a fast-growing industry. The self-balancing scooter is one of the transportation used in many areas. This work is limited to simulation using MATLAB/Simulink. I recommend the hardware implementation of this two-wheel self-balancing scooter based on STSMC and PID using real-time operating systems like digital signal processing (DSP). In

implementation, an inertial measurement unit (IMU), which is a combination of an accelerometer and gyroscope measurement, is used to estimate and obtain the tilt angle of the scooter. Angle from IMU sensor needs to be added fitter like a Kalman filter or complementary filter.

On the other hand, on a two-wheel self-balancing scooter, by adding some future, it is possible to change it to a wheelchair that is used by people for whom waking is difficult or impossible due to illness, injury, or disability.

Acknowledgements: This research thesis was funded by Adama Science and Technology University under the grant number of **ASTU/SM-R/165/20**, Adama, Ethiopia.

REFERENCES

[1] S. S.Kim, "Robust transition control of underactuated two-wheeled self-balancing vehicle with semi-online dynamic trajectory planning," *Mechatronics,* vol. 68, 2020.

[2] Saqib Irfan, Adeel Mehmood, Muhammad Tayyab Razzaq and Jamshed Iqbal, "Advanced sliding mode control techniques for Inverted Pendulum: Modelling and simulation," *Engineering Science and Technology, an International Journal,* vol. 21, no. 4, pp. 753-759, 2018.

[3] Mahmoud Khaled, Ali Mohammed and M.Shaaban Ibraheem, "Balancing a Two Wheeled Robot," July 2009.

[4] G.Rigatos, K.Busawon, J.Pomares and M.Abbaszadeh, "Nonlinear optimal control for the wheeled inverted pendulum system," *Robotica,* vol. 38, no. 1, pp. 29-47, 2019.

[5] D. Kamen, "Segway," [Online]. Available: www.segway.com. [Accessed 2021].

[6] U. Adeel, K. S. Alimgeer, O. Inam, A. Hameed, M. Qureshi, and M. Ashraf, "Autonomous Dual Wheel Self Balancing Robot Based on Microcontroller," vol. 3, no. 1, p. 843–848, 2013.

[7] F. Grasser, A. D. Arrigo, S. Colombi, A. C. Rufer and S. Member, "JOE: A Mobile, Inverted Pendulum," p. 107–114, February, 2002.

[8] O. Boubaker, "The Inverted Pendulum Benchmark in Nonlinear Control Theory: A Survey," *International Journal of Advanced Robotic Systems,* vol. 10, no. 5, May 2013.

[9] Vijayanand Kurdekar and Samarth Borkar, "Inverted Pendulum Control: A Brief Overview," *International Journal of Modern Engineering Research,* vol. 3, no. 5, pp. 2924-2927, Sep - Oct. 2013.

[10] S. Lin and C. Tsai, "Development of a Self-Balancing Human Transportation Vehicle for the Teaching of Feedback Control," *IEEE Transactions on Education,* vol. 52, no. 1, pp. 157-168, Feb. 2009.

[11] Lwin Mg Mg Thwin and Ye Chan, "Application of LQR control for two-wheel self-balancing robot," *journal of Myanmar Academy of Arts and Science. Research Conference.,* vol. XVIII, no. 2c, 2020.

[12] C.-C. Tsai, H.-C. Huang, and S.-C. Lin, "Adaptive Neural Network Control of a Self-balancing Two-Wheeled Scooter," *IEEE Transactions on Industrial Electronics,* pp. 1420-1428, April 2010.

[13] Gia Minh Thao, N, Nghia Duong and Phan Quang , "A PD Sliding Mode Controller for Two-Wheeled Self-Balancing Robot," *research gate,* September 2011.

[14] Ngoc-Son Nguyen and Ho Pham Huy Anh, "Adaptive Backstepping Self-Balancing Control of Two-Wheel Electric Scooter," *International Journal of Advanced Robotic Systems,* vol. 11, no. 10, October 2014.

[15] Pranav Sevekari, Bhagyashri Tamhane and Shailaja Kurode , "Robust Control for Stable and Safe Performance of a Two Wheeled Human Transporter," *IFAC-Papers On Line,* vol. 53, no. 1, pp. 616-621, 2020.

[16] Askar Azizi, Hamid Nourisola Amin Sadeghi-Emamgholi and Fahime Naderisafa , "Adaptive PSO-LS-wavelet H∞ control for two-wheeled self-balancing scooter," *International Journal of Control, Automation and Systems,* vol. 15, p. 2126–2137, 06 September 2017.

[17] W. Zhou, 2008. [Online]. Available: https://hdl.handle.net/10289/2423. [Accessed 12 feb 2021].

[18] Lin, SC., Tsai, CC. & Huang. HC, "Adaptive Robust Self-Balancing and Steering of a Two-Wheeled Human Transportation Vehicle," *J Intell Robot Syst 62,* p. 103–123, 2011.

[19] Zhang Zheng and Meng Teng, "Modeling and Decoupling Control for Two-Wheeled Self-Balancing Robot," *28th Chinese Control and Decision Conference (CCDC)*, 2016.

[20] Long chen, Hai Wang,Yunzhi Huang, Zhaowu Ping, Ming Yu, Xuefeng Zheng,Mao Ye andYounhao Hu, "Robust hierarchical sliding mode control of a two-wheeled self-balancing vehicle using perturbation estimation," *mechanical system and signal processing*, 2020.

[21] W. Alam, A. Mehmood, K. Ali, U. Javaid, S. Alharbi and J. Iqbal, "Nonlinear control of a flexible joint robotic manipulator with experimental validation," *Strojniški vestnik-Journal of Mechanical Engineering*, vol. 64(2018), pp. 47-55, 2018.

[22] Zhao jie and Ren Sijing, "Sliding Mode Control of Inverted Pendulum Based on State Observer," *Sixth International Conference on Information Science and Technology (ICIST)*, pp. 322-326, May 6-8,2016 .

[23] Esmaeili, N., Alfi, A and Khosravi, H, "Balancing and Trajectory Tracking of Two-Wheeled Mobile Robot Using Backstepping Sliding Mode Control: Design and Experiments," *J Intell Robot Syst*, p. 601–613, 2017.

[24] R.Decarol and S.Zak, "A Quick Introduction to Sliding Mode Control and Its Applications 1," *semantic scholar*, 2008.

[25] Vadim Utkin, Alex Poznyak, Yury Orlov and Andrey Polyakov, "Conventional and high order sliding mode control," *Journal of the Franklin Institute*, vol. 357, no. 15, pp. 10244-10261, 2020.

[26] V. Utkin, "Discussion Aspects of High-Order Sliding Mode Control," *IEEE Transactions on Automatic Control*, vol. 61, no. 3, pp. 829-833, March 2016.

[27] Y. Rizal, R. Mantala, S. Rachman and N. Nurmahaludin, "Balance Control of Reaction Wheel Pendulum Based on Second-order Sliding Mode Control," *2018 International Conference on Applied Science and Technology (iCAST)*, pp. 51-56, 2018.

[28] Jun Liu, Liang Gao, Junjie Zhang and Feng Yan, "Super-twisting algorithm second-order sliding mode control for collision avoidance system based on active front steering and direct yaw moment control," *Journal of Automobile Engineering* , vol. 235, no. 1, pp. 43-54, 2021.

[29] Binbin Yan, Pei Dai, Ruifan Liu, Muzeng Xing and Shuangxi Liu, "Adaptive super-twisting sliding mode control of variable sweep morphing aircraft," *Aerospace Science and Technology*, vol. 92, pp. 198-210, 2019.

[30] Feng Z and Fei J, "Design and analysis of adaptive Super-Twisting sliding mode control for a microgyroscope," January 3, 2018.

[31] S. Ouchen, M. Benbouzid, F. Blaabjerg, A. Betka and H. Steinhart, "Direct Power Control of Shunt Active Power Filter using Space Vector Modulation based on Super Twisting Sliding Mode Control," *IEEE Journal of Emerging and Selected Topics in Power Electronics,* 2020.

[32] Amjad J Humaidi and Alaq F Hasan, "Particle swarm optimization–based adaptive super-twisting sliding mode control design for 2-degree-of-freedom helicopter," vol. 53, no. 9-10, pp. 1403-1419, June 02, 2019.

[33] Akun Zhao, Panfeng Huang and Fan Zhang, "Dynamic Modeling and Super-Twisting Sliding Mode Control for Tethered Space Robot," *Acta Astronautica (2017),* 2017.

[34] Al-Mayyahi, A., Wang, W.and Birch, P., "Design of Fractional-Order Controller for Trajectory Tracking Control of a Non-holonomic Autonomous Ground Vehicle," *J Control Autom Electr Syst 27,* p. 29–42, 2016.

[35] Mirjalili, S., Mirjalili, S. M. and Lewis, A., "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46- 61, 2014.

[36] Rabii Fessi, Hegazy Rezk and Soufiene Bouallègue, "Grey Wolf Optimization Based Tuning of Terminal Sliding Mode Controllers for a Quadrotor," *Computers, Materials & Continua,* vol. 68, no. 2, pp. 2265-2282, 2021.

[37] Sen, M. A. and Kalyoncu, M., "Optimal Tuning of PID Controller Using Grey Wolf Optimizer Algorithm for Quadruped Robot," *Balkan Journal of Electrical and Computer Engineering,* vol. 6, no. 1, pp. 29-35, 2018.

[38] Singh, N. and Singh, S. B., "Hybrid algorithm of particle swarm optimization and Grey Wolf optimizer for improving convergence performance," *Journal of Applied Mathematics*, pp. 1-15, 2017.

[39] Zarringhalami, M., Hakimi, S. M. and Javadi, M. , "Optimal Regulation of STATCOM Controllers and PSS Parameters Using Hybrid Particle Swarm Optimization," *14th International Conference on Harmonics and Quality of Power, Bergamo,* pp. 1-7, 2010.

[40] Panda, S. and Padhy N. P., "Comparison of particle swarm optimization and Genetic Algorithm for FACTS-based controller design," *International journal of Applied Soft Computing,* vol. 8, no. 4, pp. 1418-1427, 2008.

[41] Ekinci, S., "Application and comparative performance analysis of PSO and ABC algorithms for optimal design of multi-machine power system stabilizers," *Gazi University Journal of Science,* vol. 29, no. 2, pp. 323-334, 2016.

[42] Chen Zhou, Xinhui Liu, Wei Chen and Feixiang Xuand Bingwei Cao, "Optimal Sliding Mode Control for an ActiveSuspension System Based on a Genetic Algorithm," 4 December 2018.

[43] D. C. Meena and A. Devanshu, "Genetic algorithm tuned PID controller for process control," *2017 International Conference on Inventive Systems and Control (ICISC),* pp. 1-6, 2017.

[44] Farahmandrad, M., Ganjefar, S., Talebi, H.A. and Bayati.M, "Design of higher-order sliding mode controller based on genetic algorithm for a cooperative robotic system," *Int. J. Dynam. Control 8,* p. 269–277, 2020.

APPENDIXES

Appendixes 1:- Simulink block

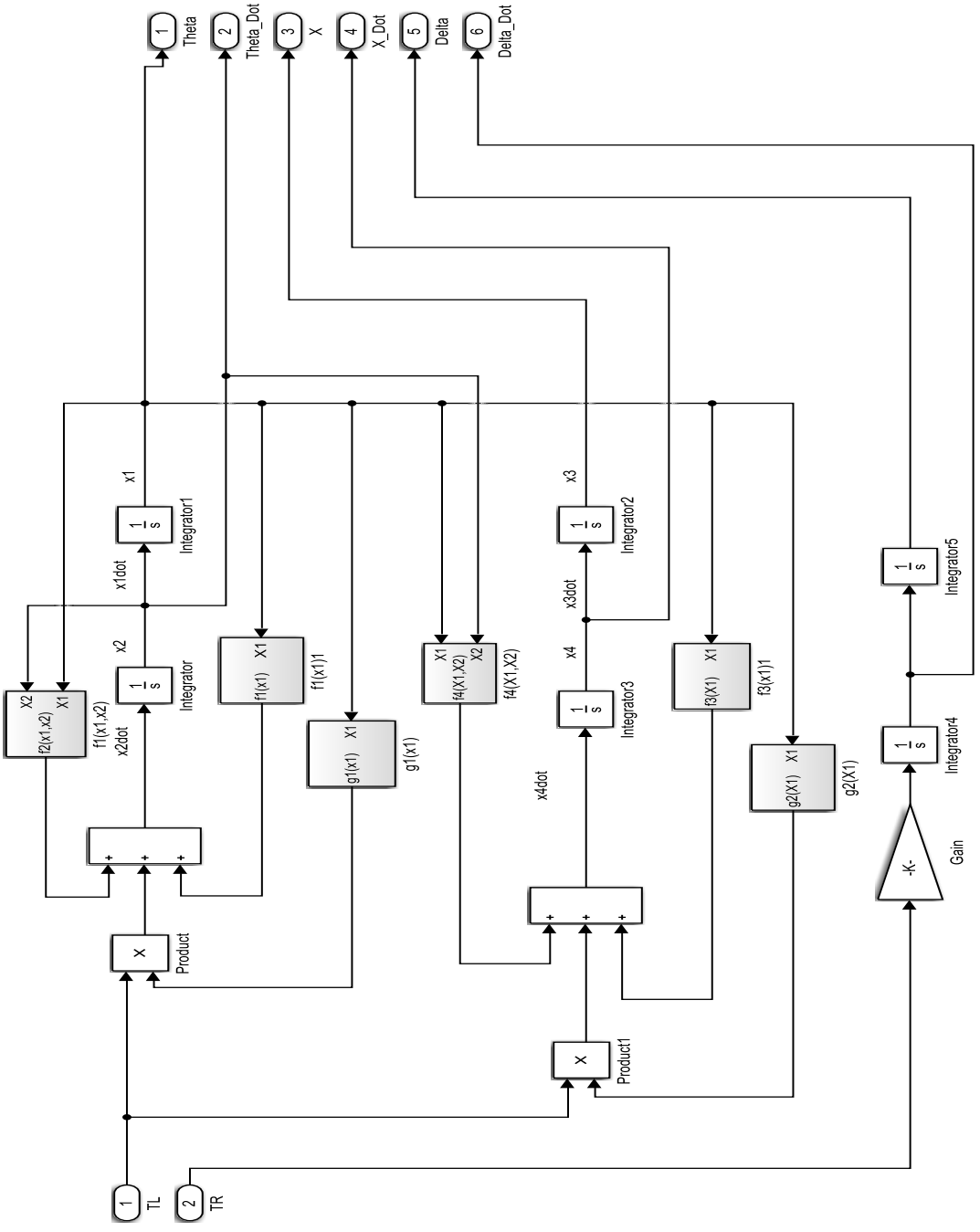


Figure 1- Simulink block for eScooter (plant)

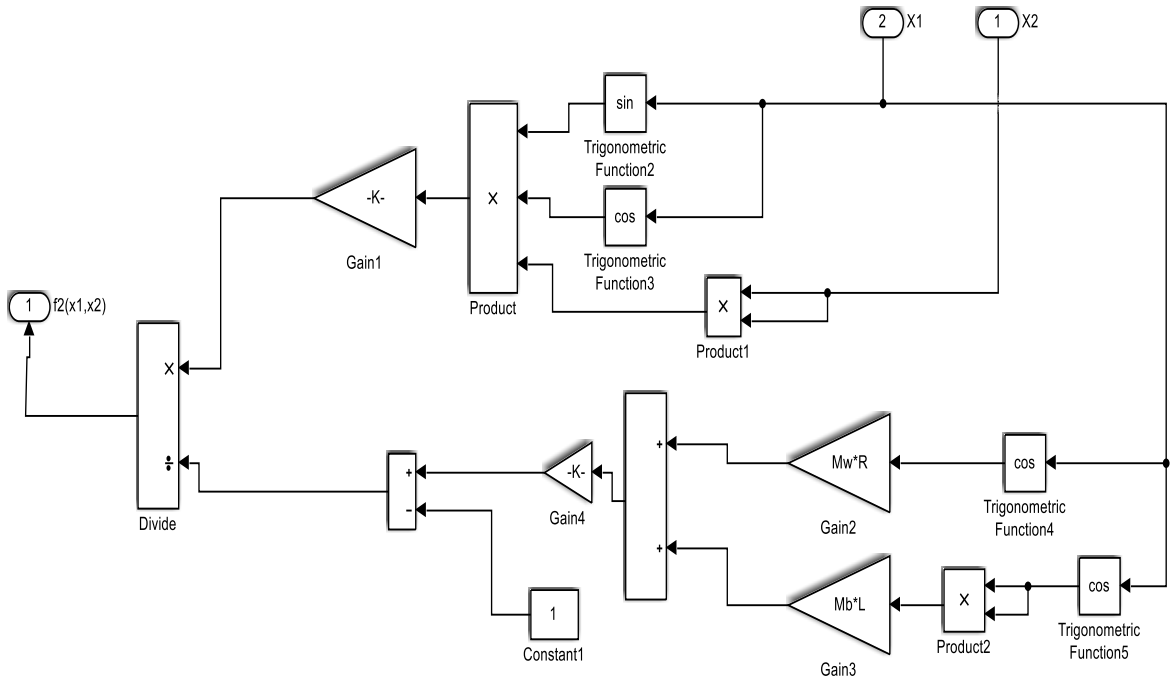


Figure 2- Simulink model of f2(x1, x2)

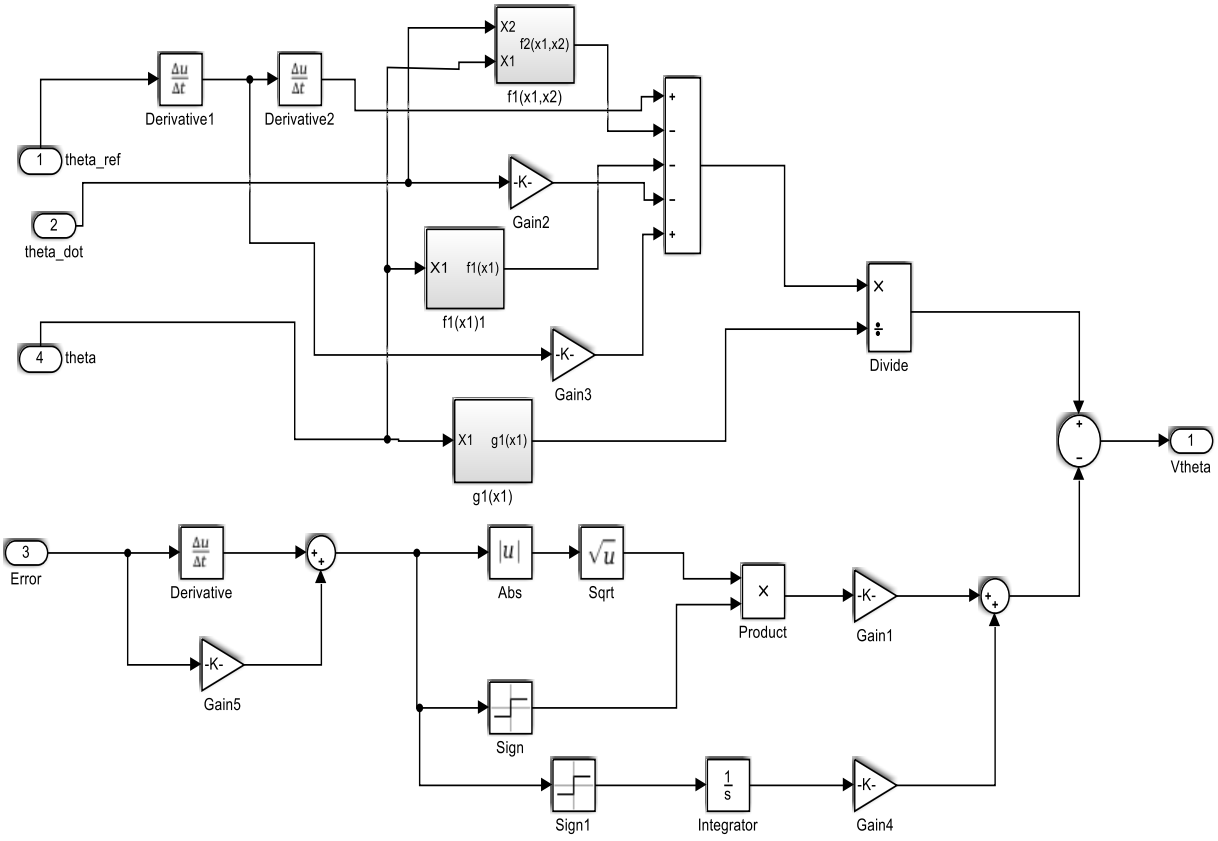


Figure 3- designed super twisted sliding mode controller

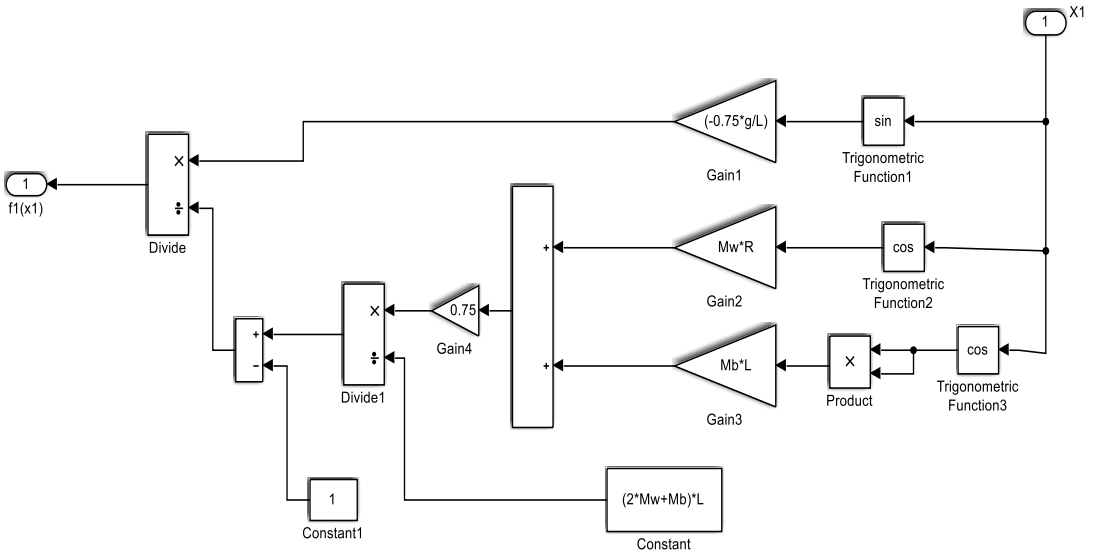


Figure 4- Simulink model of f1(x1)

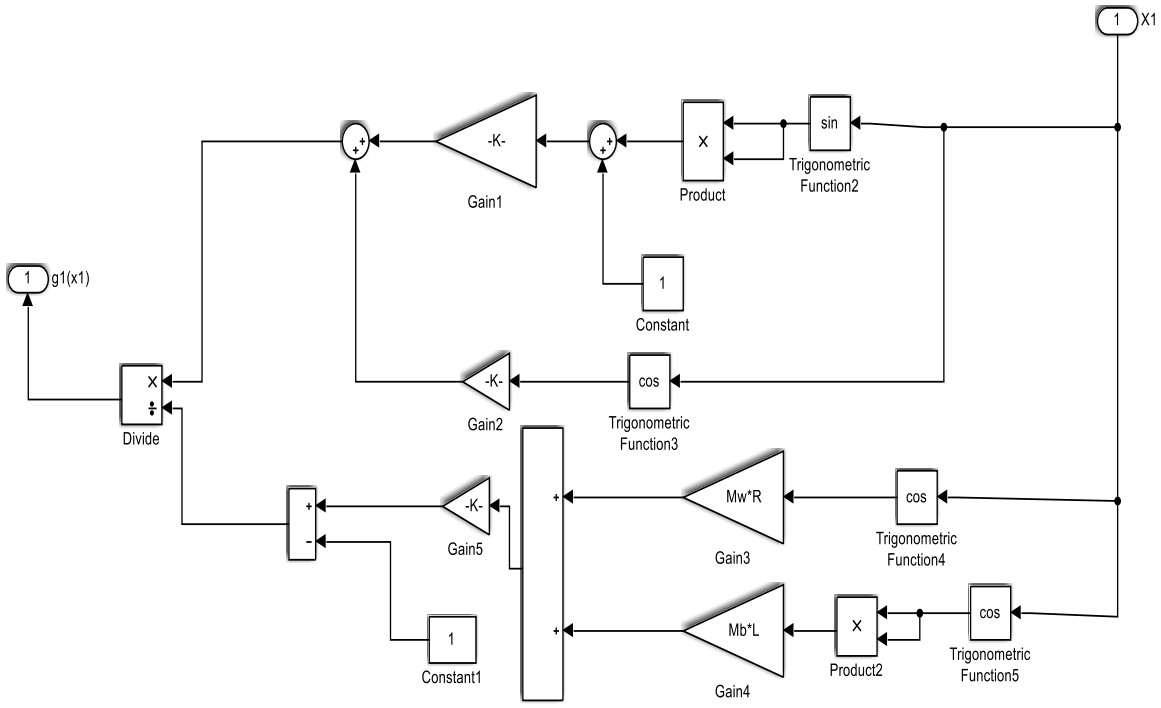


Figure 5- Simulink model of g1(x1)

Appendixes 2: - Some optimization algorithm (GA code)

```
function Tune_comp_sco
tic % start timer
clear
close all
options= optimoptions('ga','PlotFcn',@gaplotbestf);
options.PopulationSize= 20;
options.MaxGenerations= 30;   % Increase value if needed
nvar= 6; % Number of parameters
LB= [0.0001 0.0001 0.0001 0.0001 0.0001 0.0001]; % Lower bound of tuning parameters
UB= [15 15 15 600 600 600]; % Upper bound of tuning parameters

[x,fval]= ga(@EvalObj,nvar,[],[],[],[],LB,UB,[],options)

% Updates Kp, Ki and Kd in simulink model workspace for verification
simulink_model= 'comp_sco';
load_system(simulink_model);
gains= get_param(simulink_model,'modelworkspace');
gains.assignin('c', x(1));
gains.assignin('C1', x(2));
gains.assignin('C2', x(3));
gains.assignin('Kp', x(4));
gains.assignin('Ki', x(5));
gains.assignin('Kd', x(6));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% EvalObj measures a performance index from the simulink model for tuning
% obj= EvalObj(x) returns the objective function value obj calculated using
% the gain vector, x= [Kp Ki].

function obj= EvalObj(x)
% Updates the values of C1, C2, c, Kp, Ki and Kd in Simulink model workspace
simulink_model= 'comp_sco';
load_system(simulink_model);
gains= get_param(simulink_model,'modelworkspace');
gains.assignin('c', x(1));
gains.assignin('C1', x(2));
gains.assignin('C2', x(3));
gains.assignin('Kp', x(4));
gains.assignin('Ki', x(5));
gains.assignin('Kd', x(6));
% Simulates the simulink model to get the outputs
simOut= sim(simulink_model,'SaveOutput','on');

% Retrieves time t and error e
t= simOut.find('t');
```

```
e1= simOut.find('e1');
e2= simOut.find('e2');
% Calculate a performance index using the trapezoidal integration rule
n= length(t);
ee1= abs(e1);
ee2= abs(e2);
obj= 0;
for i= 2:n
   stepsize= t(i)-t(i-1);
      obj= obj+0.5*t(i)*(ee1(i-1)+ee1(i)+ee2(i-1)+ee2(i))*stepsize; % ITAE
end
end
toc % end timer
end
```