Addis Ababa University
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering

Super Twisted Sliding Mode Controller for Self-balancing Control of a Two-Wheeled Electric Scooter Based on Grey Wolf Optimizer

A Thesis Submitted to School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Control Engineering

By

YALEW BALKEW KEBEDE

Advisor

LEBSEWORK NEGASH (Ph.D)

April 7, 2026

Addis Ababa, Ethiopia

Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Super Twisted Sliding Mode Controller for Self-balancing Control of a Two-Wheeled Electric Scooter Based on Grey Wolf Optimizer

By: YALEW BALKEW KEBEDE

APPROVED BY BOARD OF EXAMINERS

Name:	Date:	Signature:
(Dean, School of Graduate Committee)		
(Advisor)		
(Internal Examiner)		
(External Examiner)		

Declaration

I, the undersigned, declare that this MSc thesis entitled "Super Twisted Sliding Mode Controller for Self-balancing Control of a Two-Wheeled Electric Scooter Based on Grey Wolf Optimizer" is the result of my own independent research work and has not been presented for fulfillment of any degree in this or any other University and all sources and materials used for the thesis are properly acknowledged.

Name of the student	Date	Signature

This thesis has been submitted for examination with my approval as a University advisor.

Name of the Advisor	Date	Signature

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to Almighty God for all the everlasting blessings He has bestowed upon me throughout my life and for giving me the ability to conduct research effectively.

I would like to express my sincere gratitude to everyone who provided me with great support during the completion of my master's thesis. My deepest appreciation goes to my advisor, Dr. Lebsework Negash, who supported me with clear illustrations and helped me gain a solid understanding of the subject, both in practice and in writing.

It is an honor to recognize my organization, Ethiopian Electric Utility, for their co-operation and permission throughout the duration of my master's program.

Lastly, I am privileged to express my deepest gratitude to my beloved family and friends for their personal support. I thank them for always being my main source of motivation to succeed.

Yalew Balkew

Abstract

The rapid expansion of the transportation industry has intensified the demand for efficient, reliable, and intelligent personal mobility solutions. Self-balancing two-wheeled scooters have emerged as an innovative response to this need, offering compact, energy-efficient, and environmentally friendly transportation. However, ensuring stability and robustness in such systems remains a major challenge due to their nonlinear, time-varying, and inherently unstable dynamics, especially under external disturbances and load variations.To address this gap, this study proposes a robust control strategy for a self-balancing two-wheeled personal transporter modeled as an inverted pendulum system. The main objective is to design and evaluate a Super Twisting Sliding Mode Controller (STSMC) capable of maintaining balance and directional stability under varying load and disturbance conditions. The STSMC was chosen because it effectively eliminates the chattering phenomenon associated with conventional Sliding Mode Control (SMC) while retaining its strong robustness and finite-time convergence properties—making it particularly suitable for highly nonlinear and uncertain systems such as self-balancing scooters.To enhance performance, the STSMC parameters are optimized using two metaheuristic algorithms Grey Wolf Optimization (GWO) and Particle Swarm Optimization (PSO) and their results are compared. The dynamic model is developed and simulated in MATLAB/Simulink, and the controller's performance is evaluated using time-domain metrics such as rise time, settling time, and steady-state error.Simulation results demonstrate that the GWO based STSMC achieves superior performance, with a settling time of 0.215 s, a rise time of 0.1345 s, and zero steady-state error for pitch angle control. Similarly, for yaw control, it achieves a settling time of 0.2719 s and a rise time of 0.2157 s. The controller maintains stable operation for varying rider masses up to 100 kg, outperforming the PSO based STSMC in both balancing and directional control.The GWO-based STSMC also exhibits faster convergence speed, lower steady-state error, and enhanced robustness against disturbances compared to the PSO-based controller.These findings highlight the effectiveness of the proposed GWO-optimized STSMC as a robust and efficient control approach for modern self-balancing transporters. The developed controller improves system stability, adaptability, and disturbance rejection, contributing to the advancement of intelligent personal mobility systems for real-world applications.

Keywords:STSMC controller, Scooter, Grey wolf optimization, and Particle swarm optimization.

Contents

List of Figures

List of Tables

List of Symbols

D	Distance between the two wheels contact
dx, dy	Disturbance due to uneven terrain affecting the horizontal and vertical motion
$d\delta$	Disturbance affecting the yaw angle dynamics due to uneven terrain
G	Gravitational acceleration
H_L, H_R	Reaction force between the wheels and the ground
F_{fL}, F_{fR}	Friction force between the wheels and the ground
i_a	Current flowing through Armature Coil
I_R	Motor Inertia
J_B	Inertia of the body chassis
J_{TL}, J_{TR}	Rotating moment of inertia
K_e	DC motor back EMF coefficient
K_m	Torque constant of DC motor
L	Distance between the center of gravity and the z-axis
L_a	Inductance of armature
M_B	Mass of the body
M_W	Mass of the wheels
τ_L, τ_R	Torques of the left and right
V_a	Voltage supplied to the Armature
V_e	Counter EMF of the motor
θ	Balance angle
δ	Direction angle
$\Delta H_L, \Delta H_R$	Horizontal Ground Reaction Forces H_L and H_R due to uneven ground

Abbreviations

ANFIS	Adaptive Neuro-Fuzzy Inference System
BLDC	Brushless Direct Current
CF	Cascaded-Forward
DNN	Deep Neural Network
DC	Direct Current
FF	Feed-Forward
ITAE	Integral of Time-weighted Absolute Error
FLC	Fuzzy Logic Controller
GWO	Grey Wolf Optimization
HOSMC	Higher Order Sliding Mode Control
IMU	Inertial Measurement Unit
LQR	Linear Quadratic Regulator
MATLAB	Matrix Laboratory
MIMO	Multiple Input Multiple Output
MS	Mini-Segway
MP	Minimum Phase
MSE	Mean Squared Error
NMP	Non-Minimum Phase
NN	Neural Network
PID	Proportional Integral Derivative
PSO	Particle Swarm Optimization
PT	Personal Transporter
SBTWES	Smart Battery Two-Wheel Electric Scooter
SFC	State Feedback Controller
SHIs	Special Sample and Hold Inputs
TWBMR	Two-Wheeled Mobile Robot
TWHT	Two-Wheeled Human Transporter
TWIP	Two-Wheel Inverted Pendulum
STSMC	Super-Twisting Sliding Mode Controller

Chapter 1

Introduction

1.1 Background

Without a doubt, the transportation sector is expanding quickly these days. In response to the notable increase in demand for personal transporter vehicles, The Segway Company introduced self-balancing personal transporter scooters.

In order to increase the effectiveness of human transportation and reduce costs, the self-balancing personal transporter, which is also a great example of the personal mobility concept, is now widely used in many industries and institutions, including the tourism sector, police departments, factories, and so forth. This personal transporter vehicle offers benefits such as increased accessibility and zero fuel use, making it a potential solution to global challenges such as traffic congestion and pollution. This two-wheel contraption is a superior alternative to modern personal transporters, making it accessible to the majority of society. Self-balancing personal transporters, such as Segways, use multiple accelerometer and gyroscope sensors to measure acceleration and angular rate over many axes. Using more sensors in Segway models has drawbacks, including increased computational power and costs.

The self-balancing scooter monitors the platform's tilt, adjusts it based on the reference angle, and maintains its upright position. The self-balancing scooter's control unit can adjust to external forces and return to a stable position, ensuring passenger safety. Two-wheeled self-balancing devices have become a popular research topic in robotics due to its inherent instability in dynamic systems. This two-wheel contraption operates using the inverted pendulum concept. This concept is widely discussed in control engineering textbooks and among control researchers.

Balancing a two-wheeled scooter, similar to balancing a broom on a fingertip, is a common engineering challenge based on the inverted pendulum principle. The phrase "balance" refers to a scooter's condition of equilibrium, similar to standing upright. However, the system is unbalanced and continues to deviate from the vertical. A gyro chip measures the scooter's angle and sends it to the controller for balancing at the upper right position. To balance a scooter, use a measurement unit to accurately determine the current tilt angle.

In order to compensate for tilt angle, a controller must be created. STSMC controls scooter pitch angle for balancing and yaw angle for direction control (right/left turns).The mathematical model is based on Newton's second law of motion. The actuator DC motor is also modeled and the system is then simulated in MATLAB/Simulink. Control system researchers worldwide have shown a growing interest in mobility scooters. The

scooter's performance is mostly determined by its mechanical model, signal processing methodology, and control mechanism.

The two-wheeled self-balancing robot's nonlinear MIMO underactuated system makes it difficult to maintain equilibrium while climbing or descending a hill, even when there are no detectable disturbances. Standing electric scooters are typically preferred by the younger generation, but they are gradually gaining appeal as elderly and the mobility handicapped understand they provide an alternative mode of transportation, are low-cost to operate and maintain, and are far more than simply a novelty.

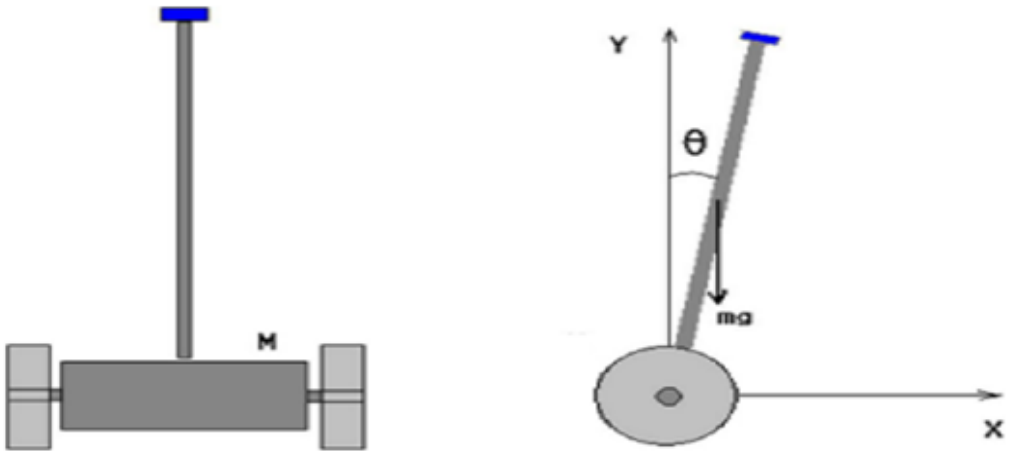


Figure 1.1: depicts the front and side perspectives of a two-wheeled self-balancing scooter.

The system can be Developed to travel forward, backward, or turn left or right once it is balanced (at zero tilt angle). There are three main parts to this two-wheel scooter:
a. Wheels: utilized to shift the system forward or backward, balancing the body and allowing for left or right system rotation.
b. chassis: It contains the circuitry, motors, and other components of the system.
c. Pendulum: It is necessary to stabilize the system's components that are creating the instabilities.

1.2 Statement of the problem

Problem Statement

The need for environmentally friendly and user-adaptable personal mobility devices has increased due to the growing need for small, efficient, and sustainable transportation options. Electric motor-powered self-balancing two-wheeled transports have become a viable substitute for traditional automobiles due to their low energy consumption, minimal emissions, and user-friendliness. Such transporters can be useful and adaptable vehicles for both daily movement and leisure use in countries like ours that have abundant tourist attractions. By encouraging inclusivity and accessibility, this technology offers substantial social benefits in addition to environmental ones.
for flexibility and intuitiveness in control. Moreover, the effectiveness of the device is further strengthened by its capability to handle rough terrain conditions, making it

suitable for use in off-road terrains, parks, and urban streets. However, despite these advantages, there are fundamental problems associated with the dynamical system of the self-balancing two-wheeled device due to its fundamentally unstable and nonlinear characteristics, behaving akin to an inverted pendulum. A control approach that allows continuous adjustment of torque values in response to any perturbation, as well as the effects of rider input and varying terrain, must be developed in order to provide stability and steering capability of the system. In highly nonlinear systems such as these, conventional linear controllers often prove to be inadequate in offering adequate robustness. The STSMC is chosen because it can manage model uncertainties, external disturbances, and parameter fluctuations without the scattering effect that is typically connected to traditional Sliding Mode Control (SMC). It is an effective way to stabilize and control self-balancing personal transporters in real-world operating situations because of its excellent robustness and finite-time convergence.

1.3 objectives of the study

1.3.1 General Objectives

Using STSM controllers in MATLAB/Simulink, the overall goal of this thesis is to control a two-wheel self-balancing electric scooter based on the inverted pendulum concept.

1.3.2 Specific objectives

- ▷ Develop a comprehensive and accurate mathematical model of the Self-Balancing Two-Wheeled Electric Scooter (Segway) to represent its nonlinear dynamic behavior.
- ▷ Simulate the developed mathematical model in MATLAB/Simulink to analyze the system's dynamic response under various operating conditions and disturbance scenarios.
- ▷ Design a Super-Twisting Sliding Mode Controller (STSMC) to enhance system robustness and minimize chattering while ensuring fast convergence and high tracking accuracy.
- ▷ Optimize the STSMC controller parameters using nature-inspired optimization algorithms, specifically Grey Wolf Optimization (GWO) and Particle Swarm Optimization (PSO), and conduct a comparative analysis of their performance in terms of convergence speed, control accuracy, and ITAE cost minimization.
- ▷ Draw comprehensive conclusions and provide recommendations based on simulation results, highlighting the effectiveness of the proposed control strategies and optimization techniques.

1.4 Scope of the study

The scope of this research work is to derive a mathematical model of the Self-Balancing Two-Wheel Electric Scooters dynamics considering external disturbances and parametric

uncertainties and design an advanced control strategy (STSMC) for trajectory control of a Segway using MATLAB/SIMULINK model.

1.5 Limitation of the study

This thesis is limited to the simulation of the system using MATLAB/Simulink, demonstrating how the system's many components interact with one another. This is due to the difficulty of obtaining the components needed to create an actual system. It also includes the additional time and cost associated with creating the actual prototype implementation.

1.6 Motivation of the study

The benefits of this personal transporter vehicle, such as enhanced mobility and zero fuel consumption, might be considered as the ultimate solution to the looming global difficulties created by growing traffic and environmental deterioration all over the world. This mode of transportation is being explored since it has no noise pollution, a steady speed, is simple to use, uses little energy, and is nonpolluting. Scooters are widely used for transportation in densely populated and dirty cities, hospitals (because to their quiet and environmentally favorable nature), retail malls, airports, parks, and entertainment. In comparison to a four-wheeled car, a two-wheeled scooter has various advantages, including reduced expenses, smaller size, simpler operation, better parking, and more.

1.7 Significance of the study

An electric scooter is a human-powered vehicle that facilitates balanced and safe mobility. It is utilized in workplaces, commercial buildings, airports, and other locations in addition to the tourism industry. Due to its design, this mode of transportation is perfect for exploring cities and doing unique trips without growing weary of long hikes. Because it is an electric car, it is safe and emits no pollution.

1.8 Thesis organization

The overall thesis is organized in to five distinct chapters including this introductory chapter. Further details of the later chapters are herein presented
Chapter two: A review of the literature on two-wheel self-balancing electric scooters and related works is included in this chapter.
Chapter three: explains the techniques used in this thesis and how the scooter's and DC motor's mathematical models are created. The control design (STSMC) and optimization techniques (GWO and PSO) are also presented in this chapter.
Chapter four: outlines the suggested findings, discussions and comments.
Chapter five: There are recommendations and conclusions in this chapter.

Chapter 2

Literature Review

2.1 Introduction

This section provides a summary of previous attempts to construct self-balancing devices. Reading through numerous publications might provide valuable information for the initial design of a self-balancing scooter.

The Segway is a two-wheeled, self-balancing electric scooter invented by Dean Kamen and brought to market in 2001 [2]. It is a personal transport device, making it the first of its kind. The Segway PT (Personal Transporter) utilizes gyroscopic sensors and balance sensors to enable the rider to control the scooter by shifting their body weight. It is powered by lithium-ion batteries and uses brushless DC motors in each wheel. The Segway PT is designed for short-distance travel in urban and suburban areas, offering a unique and intuitive way to move around. Despite initial hype, Segway faced challenges in finding a widespread market due to its limited practicality in certain settings. However, it remains a significant innovation in the field of personal transportation.

Figure 2.1 displays JOE, a two-wheeled vehicle prototype built by Felix Grasser [3], a researcher at the Swiss Federal Institute of Technology. This robot's primary objective is to maintain its driver's balance on its two coaxial wheels. Each DC motor powers one of the coaxial wheels. The vehicle can make stationary U-turns because of its design, which is accomplished by applying torque to the proper wheels. A linear state space controller that employs motor encoders and gyroscope sensory input is used to stabilize the system.

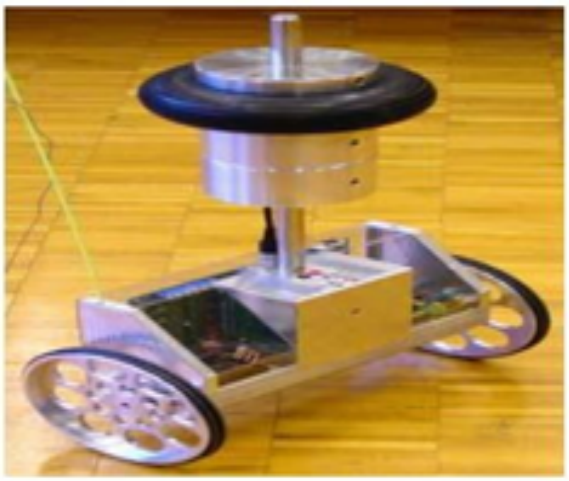


Figure 2.1: JOE inverted pendulum.

Dean L. Kamen [2] created SEGWAY PT, an application for human transporters based on a two-wheeled robot. The model is shown in detail in Figure 2.2. Body weight is moved and balanced on two parallel wheels to operate the device. With this technology, the user can navigate over a variety of terrains easily by taking short steps or stepping on curbs. This design uses multiple tilt sensors in addition to five gyroscopes to keep upright. Since only three gyroscopes are needed for the overall system, the extra sensors are included as a safety measure.

Figure 2.2: Segway PT.

2.2 Current Relevant Research on Self-Balancing Mechanism

Recent studies have focused on improving the performance, perception, and autonomy of Segways, particularly in challenging environments. A key area of active research involves enhancing control algorithms to better handle disturbances and the effects of uneven terrain. However, many existing methods for stabilizing and controlling Segways, including PID control [4], [5], [6] , optimal control [7], [8], intelligent control [9], [10], sliding mode control [11], [12], [13], and feedback linearization [14],exhibit limitations in robustness and adaptability, especially under dynamic and uncertain conditions. These methods usually encounter difficulties in achieving stability and accuracy on irregular terrain because of disturbances from terrain, which include bumps and slopes, as well as variation in parameters. While there has been considerable progress made with Segway control systems, very little has been documented regarding how irregular terrain affects Segway dynamics. This is mainly due to the fact that most research has only been conducted on flat terrain. This lack of attention to terrain impact leaves a significant gap in the development of reliable and robust Segway control systems for real-world, rough terrain

conditions. Early studies on controlling Segways often used the PID control algorithm. In [5] , the authors developed a Segway with two DC motors, a driver board, a microcontroller, and sensors, using PID to control motor speed. In [4], a self-balancing Segway was developed and simulated using MATLAB to test its stability under road disturbances, employing PID and linear quadratic regulator (LQR) control. More advanced control methods have been introduced. In [7], a nonlinear optimal control design was proposed for a two-wheeled inverted pendulum (TWIP) platform. In [15], a model-free LQR control was developed using deep neural networks and reinforcement learning.

The primary objective of this SBTWES is to maintain the balance of its rider on two wheels aligned with each other. Both wheels receive power from a DC motor. According to a study conducted by Ahmed and Saleh in 2020 [10], the use of a Neural Network (NN) controller and a traditional PID controller for Segway showed results in terms of controlling the position and handle bear angle. The study concludes that the Neural Network (NN) controller offers better performance and accuracy in controlling the Segway compared to the PID controller. However, a notable limitation of this paper is that it does not address how the NN controller performs under varying environmental conditions or dynamic loads.

the Paper [16] examines the creation and management of the segway that employes LQR adjustable gain. It discusses the difficulties of managing the segway when the driver's weight is unpredictable and offers the use of an altered LQR to solve this problem. The study presents experimental and simulated findings that shows how successful the control method is. However, it does not explore the long-term adaptability of the altered LQR controller to continuous or abrupt ground disturbance and weight changes over time.

An adaptive backstepping control strategy for a self-balancing two-wheel electric scooter is presented in the paper[10]. The adaptive backstepping self-balancing control system for a two-wheel electric scooter is described in this publication. It combines adaptive backstepping control with feedback control that complies with Lyapunov stability. A feedback control rule is constructed to efficiently regulate the scooter's self-balancing feature by utilizing the regulated function, and uncertain parameter estimation is performed using a recursive structure. In this simulation, the adaptive backstepping control demonstrates good performance and resilience. It is important to note, however, that backstepping control requires accurate model parameters, and the proposed solution does not address its limitations, such as how uneven terrain or external disturbances might affect the scooter's stability.

The paper [17] Presents a Robust Control System to Ensure Safe and Consistent Functioning of a Two-Wheeled Human Transporter. In order to improve the two-wheeled human transporter's (TWHT) stability and safety, a higher order sliding mode control (HOSMC) is proposed. Higher-order sliding mode differentiators are used to estimate states that cannot be directly monitored in the control development process, which takes into account a reduced order model. Proposed controllers are then tested by means of simulations and compared to a linear state feedback controller (SFC). These simulations use different weights for the rider to evaluate the robustness of HOSMC. Yet, it is evident that the study does not provide an investigation into how well the controller operates under real operating conditions, which may include rough terrain, noise in the sensors, or any external disturbance. Additionally, although STSMC is highly robust to uncertainties and disturbances, it may have several problems such as chattering, computational complexity, and parameter sensitivity.

The paper [10] addresses the replacement of the conventional PID controller for indus-

trial application of the Segway, which is a personal transportation vehicle, by the use of neural network controller. In the research, a simple inverted pendulum control system is used to assess the functioning of the Segway in terms of its capability of regulating its cart position and pendulum angle for improvement of its stability and responsiveness. Comparison of the outcomes obtained from both the PID and neural network controllers is made. From the overall comparison, it can be said that neural network controller outperforms PID. But the method might have limitations in replicating real-life scenarios.
The "modeling, simulation, and control of a dynamically unstable two-wheeled mobile robot (TWBMR)" are investigated in work of [18]. The system's tracking capability and system stability improve by designing smart controllers that can manage nonlinearities. Feed-forward (FF) and Cascade Forward (CF) networks, whose weights were properly adjusted based on their hidden layer configurations, were used to model the robot in order to obtain minimal MSE. In addition, ANFIS technique was adopted to optimize control parameters. Simulation tests on various input references revealed that the intelligent controller surpassed the fuzzy logic controller (FLC) for accurate tracking.
Recently, nonlinear control techniques for Segways have attracted more attention. In [19], a nonlinear sliding mode controller (SMC) was developed for Segway trajectory tracking, dealing with disturbances. In [20], adaptive sliding mode controllers addressed actuator faults and unknown parameters. Sliding mode super-twisting algorithms were used. While these methods have shown good results, there is still room for improvement in handling complex environments and disturbances for Segways.

This paper [21] presents a novel approach to discretizing nonminimum phase (NMP) systems using special sample and hold inputs (SHIs), transforming them into minimum phase (MP) systems. The study demonstrates that a dual-loop control tuning method enhances the performance of a mini-Segway (MS) by optimizing the closed-loop system. Experimental results indicate a significant reduction in oscillation magnitude—over 65% compared to baseline controllers—highlighting the effectiveness of proposed method.

Although Segway systems are characterized by their complicated dynamic behavior and strong nonlinearity, they were initially developed to be used in cities. It is hard to guarantee that the vehicle will operate effectively under conditions of uneven terrain. There exist several advanced control methods suggested to overcome this issue, but they involve substituting the existing controllers with more sophisticated ones, and there are only a few cases where the method has been thoroughly examined or optimized for use in difficult conditions. Besides, most of the techniques developed so far address either the problem of balance or the issue of controlling the direction of the movement but not both simultaneously. The present study aims to fill this gap by applying Super Twisting Sliding Mode Control (STSMC) technology to solve the problem of balancing and controlling the direction of the movement in difficult terrains.

Chapter 3

System Modelling

3.1 Introduction

This chapter presents the methodology, which includes the materials and software tools used, as well as the methods followed to achieve the settled goal. The development of a mathematical model that includes a Segway's body, wheels [22], and actuator is discussed. A STSMC is developed to handle balancing and steering. The STSMC parameters are fine-tuned using GWO and PSO algorithms, with the parameters of both controllers adjusted in relation to an objective function (J).A Super Twisted Sliding Mode Controller (STSMC) are used to balance the Segway. Additionally, the direction of the Segway is managed.

3.2 Followed Methods

Generally, figure 3.1 lists the methods used in this study to fulfill the necessary goals. It shows how this thesis was completed, beginning with a review of relevant literature to identify gaps in knowledge and potential remedies. The mathematical modeling for the thesis is established by extensive research using many sources. Following the acquisition of the overall dynamic modeling, controller design is completed and the controller's parameters are adjusted by the application of metaheuristic algorithms (GWO and PSO). The results from these tuned parameters are then used for STSMC for pitch angle control and yaw angle control. Next, MATLAB/Simulink is used to implement the simulation.

3.3 The two-wheel self-balancing scooter's operating principle

The first of its kind, the Segway is a self-balancing, portable personal mobility device. The Segway's cutting-edge technology allows it to move you over most terrains quickly and silently while emitting no harmful emissions into the atmosphere. It's enjoyable and simple to use. The self-balancing feature of the suggested scooter is its main benefit. This characteristic aids in the scooter's constant stability.

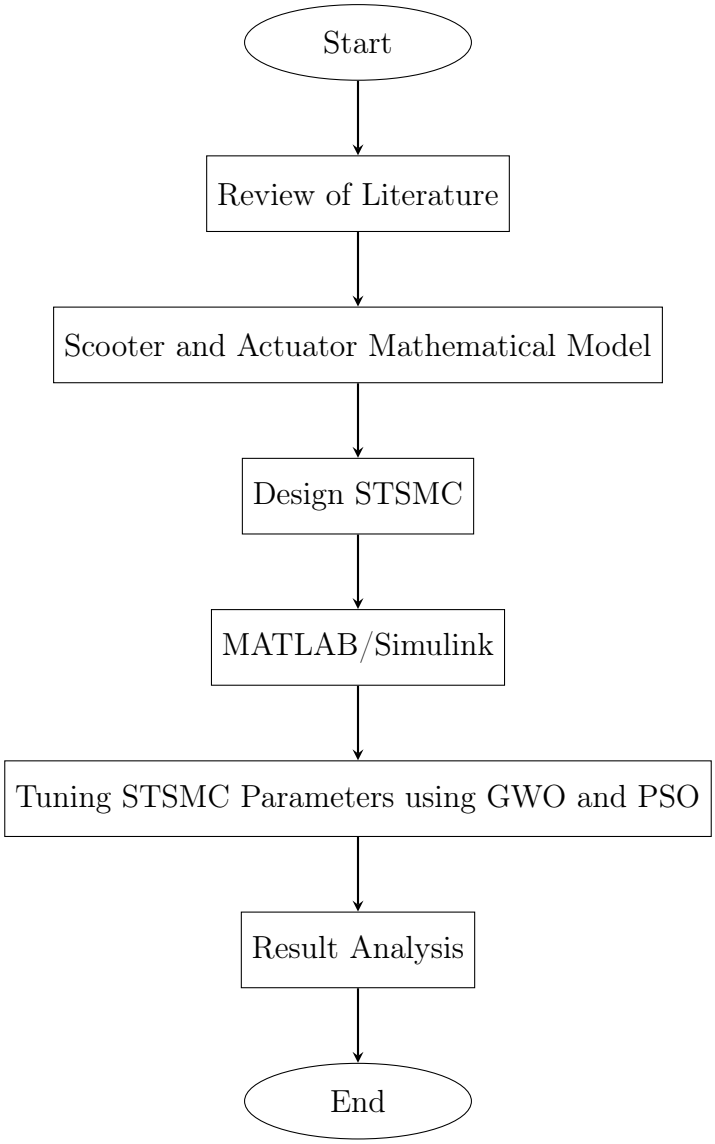


Figure 3.1: Flowchart of the Followed Methods

The Segway's dynamics are exactly like those of the inverted pendulum, a well-known control problem. For propulsion or balance, the servos can be used to rotate the wheels forwards or backwards. The rider only needs to lean forward or backward in the desired direction to accelerate or decelerate. "Gliding" on a Segway is easy and quick to pick up.

3.4 System Modelling

As illustrated in Figure 3.2 block diagram depicts the system for a segway, which operates as a Multiple Input Multiple Output (MIMO) system. Due to its inherent characteristics, this system is both nonlinear and unstable. To rectify this instability, a controller is necessary. The system features two inputs, both representing angles. The first one governs the scooter's balance, while the second angle controls its direction (either turning left or right). These angles are compared with a reference angle, and any deviation forms an error angle. This error angle is then directed to the respective controller, which works to stabilize the scooter's balance and control its direction.

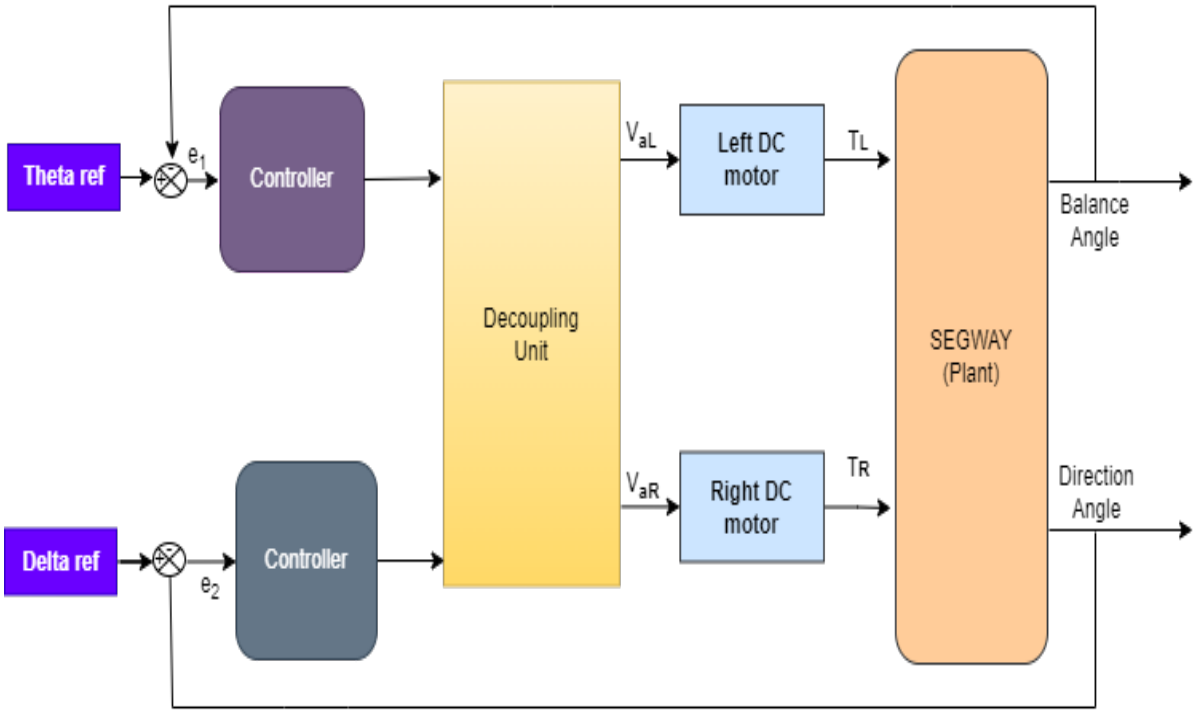


Figure 3.2: Block Diagram for Segway.

3.5 Mathematical Model of Scooter

To effectively create a control system guiding the Segway to a desired position and integrating various behaviors with its inputs, we first need a mathematical model of the Segway. The Segway's movements involve rotation around its vertical axis and forward/backward motion along the axis between its two wheels. The Segway is an electrical actuator-driven mechanical system that is underactuated, therefore modeling it involves taking into account both its electrical and mechanical subsystems. There are three parts to the mathematical model: a) Wheel Modeling: This part focuses on how the wheels rotate around the Segway's vertical axis and move along the central axis between them. b) Body Modeling: Here, we examine the dynamics of the Segway's body, including its balance and response to external forces. c) DC Motor (Actuator) Modeling: This component deals with modeling the electric actuator that drives the Segway.

Assumptions taken in developing this non-linear TWMR model:

- Left and right wheels are analogous to each other, and their frictional forces are considered.
- The actuators' dynamics are considered.
- The robot platform is assumed to be rigid and not distorted during movements.
- The effect of terrain on motor torque is not considered, while the impact on the chassis and wheels is taken into account.

To derive the mathematical model,by applying Newton's second law method and design each part of the model separately.

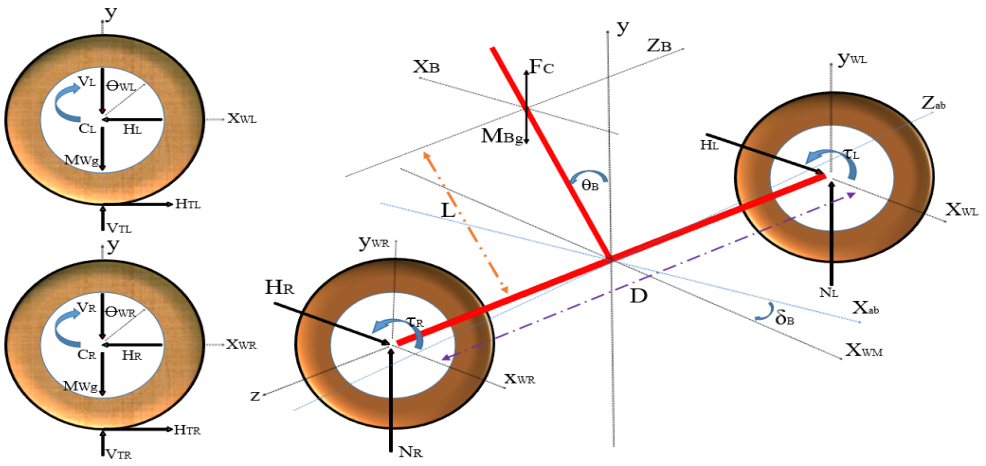


Figure 3.3: Segway free body diagram

3.5.1 Dynamics of Wheel

According to Newton's second law, the forces acting on both left and right wheels are analyzed. The dynamic force analysis around the X-axis for the left wheel is detailed, and it is noted that the right wheel exhibits similar characteristics.
The forces operating on the wheels are examined by using Newton's second rule of motion along the horizontal (X-axis):

$$\sum F_{\text{net},x} = ma \tag{3.1}$$

$$M_W \ddot{X}_{WL} = F_{FL} - H_L \tag{3.2}$$

Where F_{FL} is the friction force between the wheels and the ground, and H_L is the reaction force between the wheels and the ground.

The forces operating on the wheels are examined by using Newton's second rule of motion along the vertical (Y-axis):

$$\sum F_{\text{net},y} = m\, a_y \tag{3.3}$$

$$M_W \ddot{y}_{WL} = N_{TL} - V_L - M_W g \tag{3.4}$$

The total of the moments around the center of the wheel is equal to:

$$\sum M_0 = I\alpha \tag{3.5}$$

$$I_W \ddot{\theta}_{WL} = \tau_L - F_{FL} R \tag{3.6}$$

The correlation between the angular and linear motion components is provided by:

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

Left wheel moment of inertia:

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

Add the equation for the left and right wheels:

$$M_W(\ddot{X}_{WL} + \ddot{X}_{WR}) = (F_{FL} + F_{FR})_{H_L+H_R} \tag{3.9}$$

$$\begin{cases} I_{WL}\ddot{\theta}_{WL} = \tau_L - F_{FL}R & \text{for the left wheel} \\ I_{WR}\ddot{\theta}_{WR} = \tau_R - F_{FR}R & \text{for the right wheel} \end{cases} \tag{3.10}$$

Solve Eq. (3.10) as follows:

$$\begin{cases} F_{TL} = \frac{\tau_L - I_{WL}\ddot{\theta}_{WL}}{R} & \text{for the left wheel} \\ F_{TR} = \frac{\tau_R - I_{WR}\ddot{\theta}_{WR}}{R} & \text{for the right wheel} \end{cases} \tag{3.11}$$

3.5.2 Dynamics of Body

The scooter may be shown rotating around its tilt angle (θ) and angular velocity (ω) in the above picture. It can also rotate along its vertical axis (δ) and corresponding $\dot{\delta}$. In addition, the location x and its matching speed $\dot{x}$ are related to the wheels' linear motion. The forces operating on the wheels are examined by using Newton's second rule of motion along the horizontal (X-axis):

$$\sum F_{\text{net},x} = m\, a_x \tag{3.12}$$

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

Where, H_L and H_R are the reaction forces between the wheels and the ground.

The forces operating on the wheels are examined by using Newton's second rule of motion along the vertical (Y-axis):

$$\sum F_{\text{net},y} = m\, a_y \tag{3.14}$$

$$M_B\ddot{Y}_B = V_L + V_R - M_B g + \frac{\tau_L}{L} + \frac{\tau_R}{L}\sin(\theta_B) \tag{3.15}$$

The total of the moments around the center of the wheel is equal to:

$$\sum M_0 = I\alpha \tag{3.16}$$

$$I_B\ddot{\theta}_B = (V_L + V_R)L\sin(\theta_B) - (H_L + H_R)L\cos(\theta_B) - (\tau_L + \tau_R) \tag{3.17}$$

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

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

Moment of inertia of chassis:

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

Defining tilt angles as:

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

The average X position of the wheel:

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

And for the rotation:

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

For the chassis rotation:

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

After solving Eq. (3.15) for $V_L + V_R$ and inserting Eq. (3.13) and Eq. (3.21) into Eq. (3.17):

(3.25)

Substitute Eq. (3.22) into Eq. (3.18) and differentiate it twice:

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

Similarly, differentiate y_B of Eq. (3.19), then:

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

Multiply Eq. (3.26) by $\cos(\theta)$ and Eq. (3.27) by $\sin(\theta)$, then subtract the resulting equations:

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

Substitute Eq. (3.28) and Eq. (3.20) into Eq. (3.25):

$$\frac{4}{3}M_B L^2\ddot{\theta} + M_B L\cos(\theta)\ddot{X}_{WM} = M_B gL\sin(\theta) - (\tau_L + \tau_R)(1 + \sin^2(\theta)) \tag{3.29}$$

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

$$M_W(\ddot{X}_{WL} + \ddot{X}_{WR}) = \frac{\tau_L + \tau_R - (I_{WL}\ddot{\theta} + I_{WR}\ddot{\theta})}{R} - M_B\ddot{X}_B \tag{3.30}$$

Where:

$$\begin{aligned} I_{WL} + I_{WR} &= 2I_W \\ \ddot{X}_{WL} + \ddot{X}_{WR} &= 2\ddot{X}_{WM} \end{aligned} \tag{3.31}$$

$$2M_W\ddot{X}_{WM} = -M_B\ddot{X}_B + \frac{\tau_L + \tau_R}{R} - \frac{2I_W\ddot{\theta}}{R} \tag{3.32}$$

After solving for the left and right wheels, add the two equations from Eq. (3.8).

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

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

Substitute Eq. (3.26) and Eq. (3.34) into Eq. (3.32) yields:

$$(M_B L\cos(\theta) + RM_W)\ddot{\theta} + (2M_W + M_B)\ddot{X}_{WM} = M_B L\dot{\theta}^2\sin(\theta) + \frac{\tau_L + \tau_R}{R} \tag{3.35}$$

Solve Eq. (3.29) for $\ddot{X}_{WM}$ and substitute into Eq. (3.35):

$$\begin{aligned} &\left[(2M_W + M_B) - \frac{0.75(M_B L\cos(\theta) + M_W R)\cos(\theta)}{L}\right]\ddot{\theta} \\ &= \frac{0.75g(2M_W + M_B)\sin(\theta)}{L} - 0.75M_B\sin(\theta)\cos(\theta)\dot{\theta}^2 \\ &\quad - \left[\frac{0.75(2M_W + M_B)(1 + \sin^2(\theta))}{M_B L^2} + \frac{0.75\cos(\theta)}{RL}\right](\tau_L + \tau_R) \end{aligned} \tag{3.36}$$

Solve Eq. (3.29) for $\ddot{\theta}$ and substitute into Eq. (3.35):

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

Solve F_{fL} from Eq. (3.2):

$$F_{fL}=M_W\ddot{X}_{WL}+H_L\tag{3.38}$$

Differentiate Eq. (3.7):

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

Substitute Eq. (3.38) into Eq. (3.6):

$$\begin{aligned}H_L&=\frac{\tau_L}{R}-\ddot{X}_{WL}\left(M_W+\frac{I_W}{R^2}\right)\quad\text{(for the left wheel)}\\H_R&=\frac{\tau_R}{R}-\ddot{X}_{WR}\left(M_W+\frac{I_W}{R^2}\right)\quad\text{(for the right wheel)}\end{aligned}\tag{3.40}$$

Subtract one equation from the other in Eq. (3.40) and substitute into Eq. (3.23):

$$H_L-H_R=\frac{\tau_L+\tau_R}{R}-D\ddot{\delta}\left(M_W+\frac{I_W}{R^2}\right)\tag{3.41}$$

Substitute Eq. (3.41) into Eq. (3.24):

$$\left(I_\delta+\frac{1}{2}D^2\left(M_W+\frac{I_W}{R^2}\right)\right)\ddot{\delta}=\frac{1}{2}D\frac{\tau_L+\tau_R}{R}\tag{3.42}$$

Then:

$$I_W=\frac{1}{3}M_BR^2=I_\delta=\frac{1}{3}M_B\left(\frac{D}{2}\right)^2=\frac{1}{12}M_BD^2\tag{3.43}$$

Substitute Eq. (3.43) into Eq. (3.42):

$$\ddot{\delta}=\frac{6}{(M_B+9M_W)DR}(\tau_L+\tau_R)\tag{3.44}$$

Let:

$$U_1=\tau_L+\tau_R,\quad U_2=\tau_L-\tau_R\tag{3.45}$$

Eq. (3.36), (3.37), and (3.44) are the fundamental system equations.
Let $x_1=\theta$, $x_2=\dot{\theta}$, $x_3=x$, $x_4=\dot{x}$, $x_5=\delta$, and $x_6=\dot{\delta}$.

Where, x_1 is the balance angle, x_2 is the rate of change of the balance angle, x_3 is the displacement, x_4 is the speed of the Segway, x_5 is the direction angle, and x_6 is the rate of change of the direction angle.

The state equations of the Segway are rewritten as:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f_1(x_1) + f_2(x_1, x_2) + g_1(x_1)(\tau_L + \tau_R) \\ \dot{x}_3 = x_4, \\ \dot{x}_4 = f_3(x_1) + f_4(x_1, x_2) + g_2(x_1)(\tau_L + \tau_R) \\ \dot{x}_5 = x_6 \\ \dot{x}_6 = g_3(\tau_L - \tau_R) \end{cases} \tag{3.46}$$

Where:

$$\begin{aligned} f_1(x_1) &= \frac{-0.75g\sin(x_1)/L}{0.75(M_W R + M_B L\cos(x_1))\cos(x_1)/((2M_W + M_B)L) - 1}, \\ f_2(x_1, x_2) &= \frac{0.75M_B L\sin(x_1)\cos(x_1)x_2^2}{0.75((M_W R + M_B L\cos(x_1))\cos(x_1)/((2M_W + M_B)L) - 1)}, \\ g_1(x_1) &= \frac{0.75(1+\sin^2(x_1))/M_B L^2 + 0.75\cos(x_1)/((2M_W + M_B)RL)}{0.75((M_W R + M_B L\cos(x_1))\cos(x_1)/((2M_W + M_B)L) - 1}, \\ f_3(x_1) &= \frac{-0.75g(M_W R + M_B L\cos(x_1))\sin(x_1)/L}{2M_W + M_B - 0.75(M_W R + M_B L\cos(x_1))\cos(x_1)/L}, \\ f_4(x_1, x_2) &= \frac{M_B\sin(x_1)x_2^2/L}{2M_W + M_B - 0.75(M_W R + M_B L\cos(x_1))\cos(x_1)/L}, \\ g_2(x_1) &= \frac{0.75(M_W R + M_B L\cos(x_1))(1+\sin^2(x_1))/M_B L^2 + 1/R}{2M_W + M_B - 0.75(M_W R + M_B L\cos(x_1))\cos(x_1)/L}, \\ g_3 &= \frac{6}{(M_B + 9M_W)DR}. \end{aligned}$$

Table 3.1: Parameter of Segway values and unit [15]

Symbol	Value	Description
θ	Radians	Balance angle
δ	Radians	Direction angle
M_w	7 kg	Wheel mass
M_r	90 kg	Rider's mass
R	0.2 m	Wheel radius
L	1 m	Distance b/n the Segway's center of gravity and Z-axis
D	0.6 m	Distance b/n contact of the wheels
g	9.81 m/s^2	Acceleration due to gravity

3.6 Model with Uneven Terrain

3.6.1 Modeling Terrain-Induced Variations in Forces on the Body

Consider disturbance terms in the vertical force balance equation (Y-axis), updated as:

$$M_B\,\ddot{y}_B = V_L + V_R - M_B g + \frac{\tau_L + \tau_R}{L}\sin(\theta_B) + dy(t) \tag{3.47}$$

where $dy(t)$ represents the disturbance due to uneven terrain affecting the vertical motion of the body.

The horizontal force balance equation (X-axis) is also updated with the disturbance term as:

$$M_B\,\ddot{X}_B = H_L + H_R + dx(t) \tag{3.48}$$

where $dx(t)$ represents the disturbance due to uneven terrain affecting the horizontal motion of the body.

3.6.2 Modeling Terrain-Induced Variations in Forces on the Wheels

The vertical ground reaction forces V_L and V_R are updated to include the terrain effects:

$$\begin{aligned} V_L &\to V_L + \Delta V_L(t) \\ V_R &\to V_R + \Delta V_R(t) \end{aligned} \tag{3.49}$$

where $\Delta V_L(t)$ and $\Delta V_R(t)$ represent variations due to uneven terrain affecting the wheels.

Similarly, the horizontal ground reaction forces H_L and H_R are updated as:

$$\begin{aligned} H_L &\to H_L + \Delta H_L(t) \\ H_R &\to H_R + \Delta H_R(t) \end{aligned} \tag{3.50}$$

where $\Delta H_L(t)$ and $\Delta H_R(t)$ are variations caused by the terrain.

3.6.3 Modeling Terrain-Induced Variations in Chassis Rotation

For the chassis rotation, the equation is updated as:

$$I_\delta \ddot{\delta}_{WL} = \frac{D}{2}\big[(H_L + \Delta H_L(t)) - (H_R + \Delta H_R(t))\big] + d\delta(t) \tag{3.51}$$

where $d\delta(t)$ represents the disturbance affecting the yaw angle dynamics due to uneven terrain.

3.6.4 Overall System Model with Uneven Terrain

Finally, arranging and substituting these equations, the overall system model with uneven terrain becomes:

$$\begin{aligned}
\dot{X}_1 &= X_2 \\
\dot{X}_2 &= f_1(X_1) + f_2(X_1, X_2) + g_1(X_1)(\tau_L + \tau_R) + dy(t) \\
\dot{X}_3 &= X_4 \\
\dot{X}_4 &= f_3(X_1) + f_4(X_1, X_2) + g_2(X_1)(\tau_L + \tau_R) + dx(t) + \frac{\Delta H_L(t) + \Delta H_R(t)}{M_W} \\
\dot{X}_5 &= X_6 \\
\dot{X}_6 &= g_3\big[(\tau_L - \tau_R) + (\Delta H_L(t) - \Delta H_R(t))\big] + d\delta(t)
\end{aligned} \tag{3.52}$$

3.6.5 DC Motor and its Modelling

The motor tends to be one of the more reliable parts of the electric scooter and usually isn't the cause of repair issues. If the motor is not rotating, more likely it is a controller or battery issue.

Given their growing appeal, it's critical to comprehend the electric motor technology that drives electric scooters. This time there are two types of motors used in the electric scooters: Brushless direct current (BLDC) electric motors and brushed DC motors. A three-phase permanent magnet motor that runs on DC voltage is known as a brushless DC motor. This type of motor's strong torque and high efficiency make it a popular choice for electric vehicles.

The Brushless Direct Current (BLDC) Motors operate without the use of mechanical brushes due to the incorporation of the electronic control system switching the current in the motor winding. It may also be referred to as the synchronous DC motor or electronic commutation motors. Since the use of brushes is not needed in BLDC motors, they have no need for any maintenance, friction, and less heating generation. As a result, they make the best choice for electric scooters. When fixed firmly on the scooter, the BLDC motors effectively transmit energy and reduce the weight of the scooter. A unique form of the brushless DC motor is the hub motor. It replaces the need for conventional chain/belt drives and provides a neat outlook. Hub motors are being increasingly utilized in electric scooters because of their effectiveness and low-maintenance characteristics. To conclude, hub motors and BLDC motors are commonly used due to their high efficiency and effectiveness. Each of them comes with unique features, and the selection of the type of motor is determined by various factors, including available budget and the design of the scooter. Depending on the particular needs and design factors, regular DC motors have a number of advantages over brushless DC (BLDC) motors in scooters:

- ▷ Simplicity: Ordinary DC motors have a simpler design compared to BLDC motors. They typically have fewer components, which can make them easier to manufacture, maintain, and repair.

- ▷ Cost: Generally speaking, BLDC motors are more expensive than regular DC motors. Lower production costs may arise from DC motors' easier design and construction.

- ▷ Control: DC motors can be controlled using simple methods such as varying the voltage or current supplied to the motor. This can make them easier to control and integrate into scooter systems that do not require advanced control algorithms.

- ▷ Starting Torque: When compared to BLDC motors, DC motors usually have a larger starting torque. This can be useful in situations where high torque is needed at low speeds, like when accelerating a scooter.

- ▷ Regenerative Braking: DC motors can be used for regenerative braking systems, where the motor acts as a generator to convert kinetic energy back into electrical energy during braking. While BLDC motors can also be used for regenerative braking, DC motors may offer simpler implementation in some cases.

However, it is essential to consider the limitations of normal DC motors since they lack power density and have low efficiency and require high maintenance when compared with BLDC motors. The choice between normal DC motors and BLDC motors for electric scooters will depend on a number of aspects such as requirements and costs. The other type of motor that can be used in electric scooters is an axial flux motor. Unlike radial flux motors, the design of axial flux motors allows the magnetic flux to flow through the motor parallel to the axis of rotation rather than perpendicularly to it. Axial flux motors have certain advantages compared to radial flux motors such as better cooling, power density, and even compactness. Axial flux motors may be used in electric scooters despite the fact that there are certain difficulties in designing them. In terms of the application of the axial flux motor in an electric scooter, it should be considered that cooling, mounting, and drive are important factors. Compared to axial motors, DC motors are preferable in electric cars because:

- ▷ Efficiency: DC motors can transfer more electrical energy into mechanical energy because they are often more efficient than axial motors. The scooter may perform better overall and have a longer battery life as a result.

- ▷ Control: DC motors have superior control over torque and speed, enabling smoother acceleration and deceleration. This might make the vehicle easier to handle and enhance the overall riding experience.

- ▷ Size and weight: Compared to axial motors with comparable power output, DC motors are often smaller and lighter. This can make the scooter easier to maneuver and transport by lowering its total weight.

- ▷ Cost:DC motors are frequently more affordable than axial motors, which makes them a more appealing choice for enterprises trying to reduce production costs.

In general, DC motors are a reliable choice for many electric scooters because of their efficiency, controllability, compactness, low weight, and cost-effectiveness. DC motors are the most favored actuators in electric drives because of their high starting torque and wide range of speed control.

For the given application, the DC motor is regarded as the primary actuator in this study. The below figure shows the schematic diagram of a DC motor's mechanical and electric parts. The dynamics of the DC motor can be expressed as the following equations by applying Kirchhoff's Voltage Law:

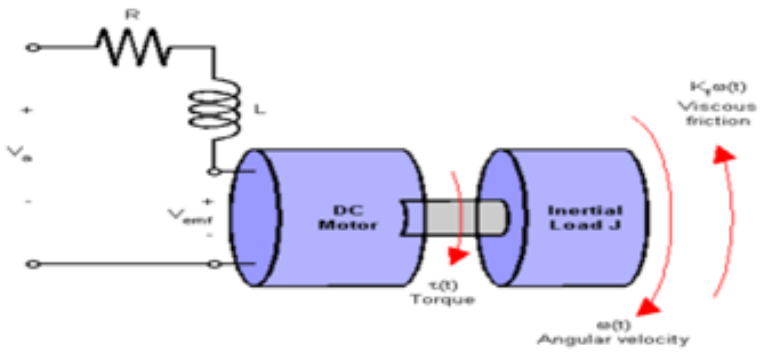


Figure 3.4: Schematic diagram of a DC motor's mechanical and electric parts.

$$V_a = R_a i_a + L_a \frac{d}{dt} i_a + V_e \tag{3.53}$$

Armature current and magnetic field strength are directly correlated with electric motor torque, as shown below. The armature current determines the motor torque alone, with a constant factor K_m:

$$\tau_m = K_m i_a \tag{3.54}$$

K_e is a constant factor that determines how the back emf is related to the shaft's angular motion:

$$V_e = K_e \omega_m \tag{3.55}$$

By applying Newton's law to the motor system:

$$\tau_m = I_R \frac{d}{dt} \omega_m + K_f \omega_m + \tau_a \tag{3.56}$$

Substitute Eq. (3.55) into Eq. (3.53) and solve for i_a in Eq. (3.54), then substitute again into Eq. (3.53):

$$V_a = \frac{R_a}{K_m} \tau_m + \frac{L_a}{K_m} \frac{d}{dt} \tau_m + K \omega_m \tag{3.57}$$

Then taking the Laplace transform of Eq. (3.56) and Eq. (3.57):

$$V_a(s) = \frac{R_a}{K} \tau_m(s) + \frac{L_a}{K} s\tau_m(s) + K\omega_m(s) \tag{3.58}$$

$$\tau_m = I_R s\omega_m(s) + K_f \omega_m(s) + \tau_a \tag{3.59}$$

Finally, rearranging and substituting Eq. (3.59) into Eq. (3.58):

$$V_a(s) = \frac{R_a}{K} \tau_m(s) + \frac{L_a}{K} s\tau_m(s) + \frac{K}{I_R s + K_f} \tau_m(s) \tag{3.60}$$

This can be the voltage for the left and right motor as:

$$V_a R(s) = \frac{R_a}{K} \tau_m R(s) + \frac{L_a}{K} s\tau_m R(s) + \frac{K}{I_R s + K_f} \tau_m R(s) \tag{3.62}$$

Assuming:

$$\begin{aligned} U_1 &= \tau_m L(s) + \tau_m R(s) \\ U_2 &= \tau_m L(s) - \tau_m R(s) \end{aligned} \tag{3.63}$$

Then relating the motor dynamics with the Segway:

$$\begin{aligned} U_1 &= \frac{K_t(J_s S + K_e)}{R_a(J_s S + K_e) + L_a S(J_s S + K_e) + K_t^2}(V_a L + V_a R) \\ U_2 &= \frac{K_t(J_s S + K_e)}{R_a(J_s S + K_e) + L_a S(J_s S + K_e) + K_t^2}(V_a L - V_a R) \end{aligned} \tag{3.64}$$

Table 3.2: DC Motor parameter [2]

DC Motor Parameter	**Description**	**Unit**
R_a	Armature Resistance	2.5ohms
L_a	Armature Inductance	0.00055H
V_a	Voltage on Armature	15V rated
V_e	Back EMF	V
K_n	Torque constant	0.58Nm/A
K_e	Motor back EMF constant	0.58Nm/A
J_r	Moment of inertia of the motor	0.02Kg·m^2

3.7 System Decoupling

System decoupling in control engineering refers to the process of designing control systems such that the influence or coupling between different variables or components of the system is minimized or eliminated. This is achieved by implementing control strategies or mechanisms that ensure changes or disturbances in one part of the system do not significantly affect other parts, thus enhancing system stability, performance, and robustness [23][24].

In this work, decoupling between yaw and pitch angles is achieved by designing independent sliding surfaces for each angle within the supertwisted sliding mode control framework, ensuring that control actions targeting pitch do not adversely affect yaw stability and vice versa.

The state-space representation of the Segway suggests that it is a heavily coupled nonlinear system. The system's separation into two independent subsystems can be achieved by translating U_L and U_R into the voltages of the wheel motors on the left (V_{WL}) and right (V_{WR}), using the decoupling transformation below.

$$\begin{cases} V_\theta = V_L + V_R \\ V_\delta = V_L - V_R \end{cases}$$

This shows that:

$$\begin{cases} V_L = \frac{1}{2}(V_\theta + V_\delta) \\ V_R = \frac{1}{2}(V_\theta - V_\delta) \end{cases} \tag{3.65}$$

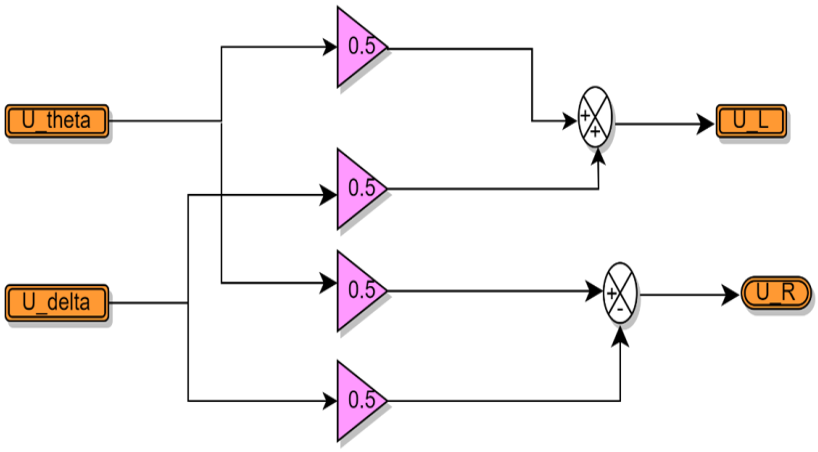


Figure 3.5: Decoupling circuit for the System

Chapter 4

Controller Design

4.1 Overview of Sliding Mode Control

One kind of control approach intended to provide system robustness against uncertainties and disturbances is sliding mode control. The fundamental notion of sliding mode control is to drive the trajectory of the system's states to converge at a predetermined sliding surface, where the dynamics of the system are more straightforward and controllable[25][26]. The sliding surface and the control law are the two main elements of sliding mode control. A hyperplane in the state space called the sliding surface is created such that the system's state trajectory stays on it when it reaches it. The purpose of the control law is to direct the system's state trajectory in the direction of the sliding surface and maintain it there. Typically, a control law consists of two parts: a continuous control signal that stabilizes the system on the sliding surface once it has reached it, and a discontinuous control signal that alternates between various modes of operation to drive the system towards the sliding surface.

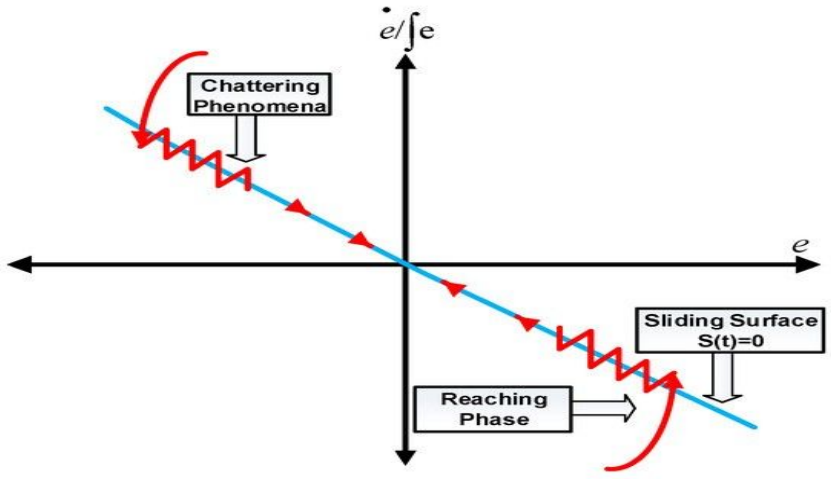


Figure 4.1: Graphical interpretation of the sliding mode controller

There are two steps in the sliding mode controller design:

1. Creating a sliding surface with the intention of eliciting a desired system response from the plant that is constrained to it. This suggests that, in order to meet the switching surface equation, the state variables of the plant dynamics are confined.

2. Driving the plant's state trajectory towards the sliding surface by constructing switched feedback gains. The generalized Lyapunov stability theory serves as the foundation for these constructions.

The sliding mode controller is based on the reaching law and includes both the reaching and sliding phases. The reaching phase drives the system to the stable manifold, while the sliding phase ensures the system slides to equilibrium.

Among the many benefits of sliding mode control are its simplicity of implementation, insensitivity to modeling errors, and resilience to uncertainties and disturbances. However, it can also have negative aspects including high control effort and chattering, which is the fast switching between control modes.

Overall, sliding mode control is a powerful and effective control strategy that can be used in a wide range of applications to achieve robust and reliable performance. SMC offers two main advantages. The first is that selecting a certain sliding function can alter the system's dynamic behavior. Secondly, the closed-loop response is still unaffected by certain uncertainty. This principle holds true for non-linearity, disturbances, and bounded model parameter uncertainties.

In practice, SMC allows for the control of nonlinear processes that are subject to significant model uncertainty and outside disturbances.

Consider the nonlinear single input- single output system

$$\dot{X} = f(x,t) + g(x,t)u, \quad Y = K(x,t) \tag{4.1}$$

By applying the general equation proposed by Slotine and Li [**?**], we can derive the sliding surface as follows:

$$s = \left(\frac{d}{dt} + \lambda\right)^{n-1} e \tag{4.2}$$

where λ is the sliding gain (a positive real value) and n is the dimension of the sliding surface.

4.2 Super Twisting Sliding Mode Control

The High-Order Sliding Mode (HOSM) technique provides an effective approach to reducing chattering. Typically, a sliding mode control signal consists of two components: one that addresses the dynamics of the system and sliding surfaces, and another that ensures the system remains on the sliding surface through switching control[27][28].

The sliding surface is defined as:

$$s = \lambda e + \dot{e} \tag{4.3}$$

The sliding mode control (SMC) is expressed as:

$$u = u_{eq} + u_s \tag{4.4}$$

where u_{eq} represents the equivalent control law, which is designed to maintain control over the sliding surface variables when system uncertainty and external disturbances are not considered.

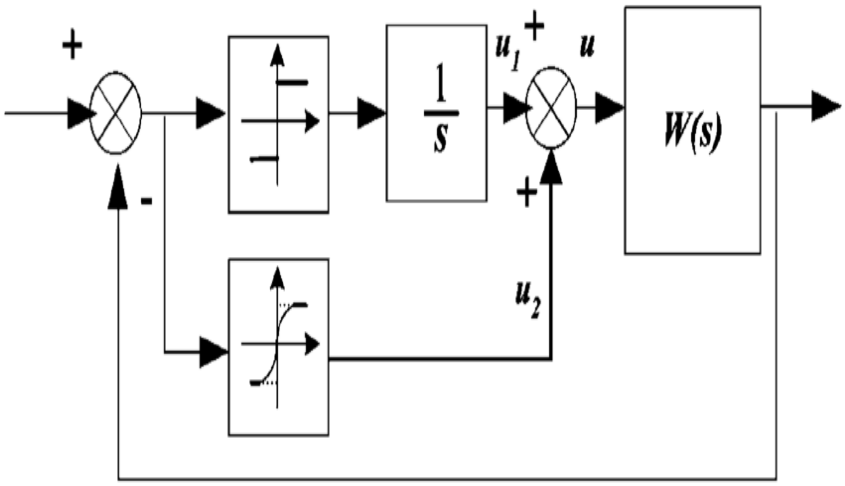


Figure 4.2: Super-Twisting controller block diagram[1]

4.2.1 Design of Super Twisting Sliding Mode Control Law

The STSMC is designed for two purposes: balancing the Segway and controlling its direction[27][29].

Balancing Control

Using the sliding surface defined in (4.1):

$$\begin{cases} s = \lambda e + \dot{e} \\ \dot{s} = \lambda \dot{e} + \ddot{e} \end{cases} \tag{4.5}$$

where s is the sliding surface and e is the error.

For the tilt angle:

$$\begin{cases} \dot{e}_\theta = \dot{x}_1 - \dot{x}_{1\text{ref}} = x_2 - \dot{x}_{1\text{ref}} \\ \ddot{e}_\theta = \ddot{x}_1 - \ddot{x}_{1\text{ref}} = \dot{x}_2 - \ddot{x}_{1\text{ref}} \end{cases} \tag{4.6}$$

This leads to:

$$\begin{aligned} S_\theta &= i(x_1 - x_{1\text{ref}}) + (\dot{x}_1 - \dot{x}_{1\text{ref}}) \\ \dot{S}_\theta &= i(\dot{x}_1 - \dot{x}_{1\text{ref}}) + (\dot{x}_2 - \ddot{x}_{1\text{ref}}) \end{aligned} \tag{4.7}$$

Using system dynamics:

$$\dot{x}_2 = f_1(x_1) + f_2(x_1, x_2) + g_1(x_1)U_1 + d_y(t) \tag{4.8}$$

Substituting into the sliding dynamics:

$$\dot{S}_\theta = i(\dot{x}_1 - \dot{x}_{1\text{ref}}) + f_1(x_1) + f_2(x_1, x_2) + g_1(x_1)U_1 + d_y(t) - \ddot{x}_{1\text{ref}} \tag{4.9}$$

To ensure stability using the Lyapunov function:

$$\begin{aligned} V_\theta &= \frac{1}{2} S_\theta^2 \\ \dot{V}_\theta &= S_\theta \dot{S}_\theta \end{aligned} \tag{4.10}$$

Substituting (4.9) into (4.10) and solving:

$$\dot{V}_\theta = S_\theta[i(\dot{x}_1 - \dot{x}_{1\text{ref}}) + f_1(x_1) + f_2(x_1, x_2) + g_1(x_1)U_1 + d_y(t) - \ddot{x}_{1\text{ref}}] \tag{4.11}$$

The equivalent control term is:

$$U_{1eq} = \frac{-i(\dot{x}_1 - \dot{x}_{1\text{ref}}) - f_1(x_1) - f_2(x_1, x_2) - d_y(t) + \ddot{x}_{1\text{ref}}}{g_1(x_1)} \tag{4.12}$$

Direction Control

Similarly, for direction control:

$$\dot{S}_\delta = j(x_6 - \dot{x}_{5\text{ref}}) + g_3(x_1)[U_2 + \Delta H_L(t) - \Delta H_R(t)] + d_\delta(t) - \ddot{x}_{5\text{ref}} \tag{4.13}$$

The equivalent control term is:

$$U_{2eq} = \frac{-j(x_6 - \dot{x}_{5\text{ref}}) - g_3(x_1)(\Delta H_L(t) - \Delta H_R(t)) - d_\delta(t) + \ddot{x}_{5\text{ref}}}{g_3(x_1)} \tag{4.14}$$

Switching Control Term

The STSMC switching term is given by:

$$\begin{aligned} U_s &= -\lambda_1 |S|^{1/2} \text{sgn}(S) + v \\ \dot{v} &= -\lambda_2 \text{sgn}(S) \end{aligned} \tag{4.15}$$

4.2.2 Overall Control Input

Combining the equivalent and switching terms, the overall control inputs are:

$$\begin{cases} U_1 = \dfrac{-i(\dot{x}_1 - \dot{x}_{1\text{ref}}) - f_1(x_1) - f_2(x_1, x_2) - d_y(t) + \ddot{x}_{1\text{ref}}}{g_1(x_1)} - i_1 |S_\theta|^{1/2} \text{sgn}(S_\theta) - i_2 \displaystyle\int \text{sgn}(S_\theta) \\ U_2 = \dfrac{-j(x_6 - \dot{x}_{5\text{ref}}) - g_3(x_1)(\Delta H_L(t) - \Delta H_R(t)) - d_\delta(t) + \ddot{x}_{5\text{ref}}}{g_3(x_1)} - j_1 |S_\delta|^{1/2} \text{sgn}(S_\delta) - j_2 \displaystyle\int \text{sgn}(S_\delta) \end{cases} \tag{4.16}$$

The developed controller is tested using MATLAB 2023a/Simulink. The parameters of the STSMC $(i, i_1, i_2, j, j_1, j_2)$ are optimized using PSO and GWO to manage the balance and direction of the Segway.

4.3 Objective Function

An objective function is a mathematical formula used in optimization problems to describe the goal that needs to be achieved. The crucial first step in applying any optimization method is choosing the objective functions that will evaluate the fitness of each population member (particle). Here are some examples of objective functions[30]:

$$\text{IAE} = \int_0^t |e(\tau)| \, d\tau \tag{4.17}$$

$$\text{ITAE} = \int_0^t \tau |e(\tau)| \, d\tau \tag{4.18}$$

$$\text{ISE} = \int_0^t e(\tau)^2 \, d\tau \tag{4.19}$$

Hence, the proposed system has two state variables to be controlled. There will be 2 error signals as balance angle and direction angle error. Therefore, the above equation can be rewritten as:

$$\text{ITAE} = \int_0^t \tau \big(|e_1(\tau)| + |e_2(\tau)|\big)\, d\tau \tag{4.20}$$

Where e_1 and e_2 are the error signals of balancing angle and direction angle difference, respectively.

4.4 Introduction to Optimization Algorithms

Optimization algorithms are mathematical techniques used to find the best solution to a problem from a set of possible solutions. These algorithms are widely used in various fields such as engineering, economics, finance, and machine learning. The goal of optimization algorithms is to minimize or maximize an objective function while satisfying certain constraints. Some common optimization algorithms include gradient descent, genetic algorithms, simulated annealing, grey wolf optimization, and particle swarm optimization. These algorithms play a crucial role in solving complex problems efficiently and effectively. This section provides a brief overview of GWO and PSO. The methods mentioned above are utilized to fine-tune the STSMC controllers, which control the scooter's balancing and direction (turning left or right).

4.4.1 Grey Wolf Optimization (GWO)

Grey Wolf Optimization (GWO) is a nature-inspired optimization algorithm that is based on the social hierarchy and hunting behavior of grey wolves. The algorithm was proposed by Mirjalili et al. in 2014 and has gained popularity due to its simplicity and efficiency in solving optimization problems [31][32][33].

In GWO, a population of candidate solutions, known as wolves, is iteratively updated to search for the optimal solution. The algorithm mimics the hunting behavior of grey wolves, where the alpha wolf represents the best solution found so far, the beta wolf represents the second-best solution, the delta wolf represents the third-best solution, and the omega wolf represents the worst solution. During each iteration, the position of each wolf is updated based on the positions of the alpha, beta, and delta wolves. This update is guided by three main search operators: encircling prey, attacking prey, and following the leader. These operators help balance exploration (searching for new solutions) and exploitation (refining existing solutions) to efficiently converge towards the optimal solution.

Overall, Grey Wolf Optimization is a powerful optimization algorithm that can be applied to a wide range of optimization problems, including engineering design, machine learning, and data mining.

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

The STSMC are used to reduce error signals, or they can more precisely specify the value of the aforementioned performance indices that should be reduced in terms of error criteria.

The simulations are Alpha (α) as the leader, Beta (β), Delta (δ), and Omega (ω) as the least weak, hierarchical, as shown in the figure below.

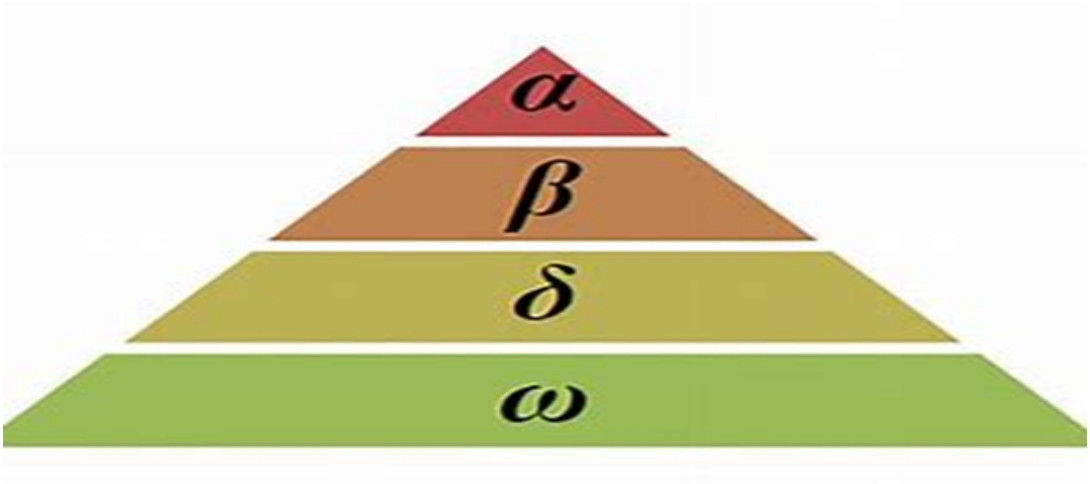


Figure 4.3: the hierarchy of grey wolfs

The alpha is the pack's leader and is in charge of choosing when to hunt, sleep, and wake up, among other things. The pack as a whole follows the alpha wolves. It is therefore the highest-ranking grey wolf in the hierarchy. Beta is ranked second among the grey wolves in their hierarchy.

The alpha receives assistance from the betas, who are wolves, in decision-making and other group tasks. The beta wolf is the most likely candidate to succeed the alpha wolf in the event that one of the alpha wolves passes away or grows too old.

The beta wolf has to show respect for the alpha while managing the lower-level wolves. The beta gives the alpha input from throughout the pack and ratifies the alpha's instructions.

Omega wolves, the lowest ranked grey wolves, are subjugated by delta wolves, who occupy the third rung of the wolf hierarchy. Wolves in this group include hunters, sentinels, elders, scouts, and caregivers. Scouts are responsible for keeping an eye on the boundaries of the area and alerting the pack to any threats. Sentinels protect and watch over the pack. Wolves who have had alpha or beta roles in the past are known as elders. Hunters provide nourishment for the pack and aid the alphas and betas by taking down prey. The lowest ranking grey wolf is called Omega. The omega acts as a scapegoat, a victim held accountable for the transgressions or failings of others. These are the last remaining eating wolves.

The main stages involved in hunting grey wolves are illustrated in the figure below.

Algorithm Mathematical Model

Mathematical models of the social order and of pursuing, encircling, and attacking prey are presented in this section.

Social hierarchy

When developing GWO, the social order of grey wolves is mathematically modeled using the alpha (α) wolf, which is the fittest solution. The beta (β) and delta (δ) wolves rank second and third, respectively, as the best solutions. Omega (ω) wolf is the remaining possible solution. α, β, and δ lead the GWO algorithm's hunting (optimization) process. The omega wolves come after these three wolves.

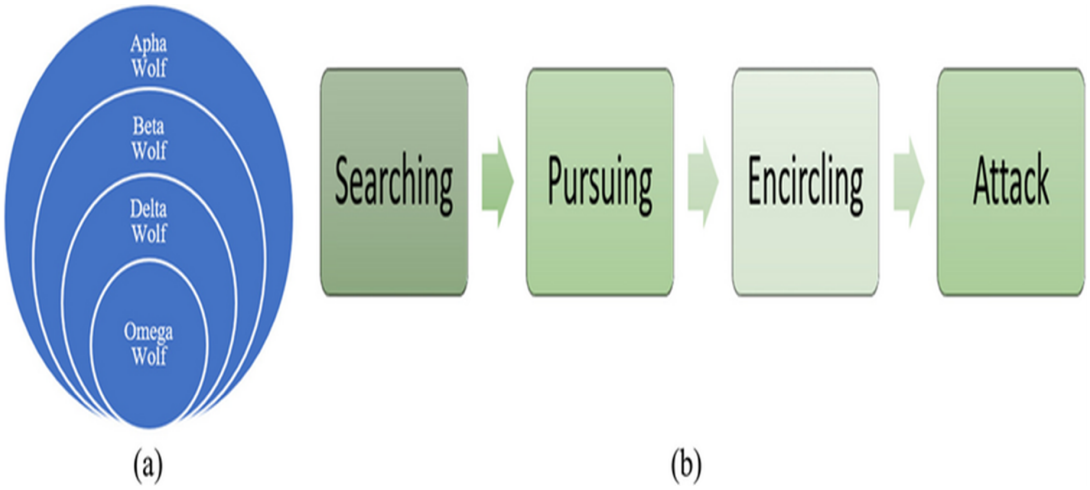


Figure 4.4: Cycle of hunting for gray wolves

Encircling prey

When a predator encircles its prey, it usually indicates that it is moving in a semi-circular or circular pattern to either capture the prey or block its escape options. Pack-hunting animals like wolves and lions frequently exhibit this behavior, in which many individuals cooperate to encircle and isolate their target prior to launching a concerted attack.
By restricting the prey's capacity to escape and raising the likelihood of a successful hunt, encircling prey gives predators a tactical advantage. It also helps them to coordinate their movements and communicate with each other to execute a synchronized attack. Overall, encircling prey is a common predatory behavior that demonstrates the intelligence and cooperation of certain animal species in their pursuit of food.

Grey wolves circle their prey during the hunt. Mathematically, encircling behavior can be modeled using the calculations below:

$$\mathbf{D} = |\mathbf{C} \cdot \mathbf{X}_P(t) - \mathbf{X}(t)| \tag{4.21}$$

$$\mathbf{X}(t+1) = \mathbf{X}_P(t) - \mathbf{A} \cdot \mathbf{D} \tag{4.22}$$

Where t is the iteration, $\mathbf{X}$ represents the grey wolf's location, and $\mathbf{X}_P$ represents the prey's position.

By changing the values of $\mathbf{A}$ and $\mathbf{C}$, alternative locations around the best search agents can be reached with regard to their present position. The value of $\mathbf{a}$ decreases linearly from 2 to 0. The random vectors $\mathbf{r_1}$ and $\mathbf{r_2}$ are in the range of [0, 1].

Hunting

The remaining search agents, including the omegas, are required to adjust their locations in line with the status of the top search agents. The first three best solutions found so far are stored to mathematically simulate the hunting behaviors of grey wolves. The recommended formulas are as follows:

$$\mathbf{D}_\alpha = |\mathbf{C_1} \cdot \mathbf{X}_\alpha - \mathbf{X}| \tag{4.23}$$

$$\mathbf{D}_\beta = |\mathbf{C_2} \cdot \mathbf{X}_\beta - \mathbf{X}| \tag{4.24}$$

$$\mathbf{D}_\delta = |\mathbf{C_3} \cdot (\mathbf{X}_\delta - \mathbf{X})| \tag{4.25}$$

$$\mathbf{X_1} = \mathbf{X}_\alpha - \mathbf{A_1} \cdot \mathbf{D}_\alpha \tag{4.26}$$

$$\mathbf{X_2} = \mathbf{X}_\beta - \mathbf{A_2} \cdot \mathbf{D}_\beta \tag{4.27}$$

$$\mathbf{X_3} = \mathbf{X}_\delta - \mathbf{A_3} \cdot \mathbf{D}_\delta \tag{4.28}$$

$$\mathbf{X}(t+1) = \frac{\mathbf{X_1} + \mathbf{X_2} + \mathbf{X_3}}{3} \tag{4.29}$$

GWO (candidate solutions) generates a random population of grey wolves at the start of the search process. The α, β, and δ wolves determine the likely position of the prey iteratively. The distance of each potential solution to the prey is updated.

Search for prey

The main search sites used by grey wolves are α, β, and δ. They dispersed to hunt, then came together to assault their victim. To mathematically depict divergence, $\mathbf{A}$ is used with random integers greater than or less than 1 to cause the search agent to diverge from the prey. GWO is an optimization technique that might behave in an unpredictable manner during the whole optimization process. It promotes exploration and stays away from local optima. GWO requires that $\mathbf{C}$ always provide random values at all times in order to emphasize exploration/exploitation not only during initial iterations but also during final iterations. This is in contrast to $\mathbf{A}$, which is lowered linearly. Stalling at the local optima is a frequent issue, particularly in the last iterations. This element can be quite beneficial in certain circumstances.

The GWO algorithm's pseudo code is displayed in the figure below.

Algorithm 1 Grey Wolf Optimizer (GWO)

Initialize the grey wolf population, X_i $(i = 1, 2, \ldots, n)$
Initialize a, A, and C
Calculate the fitness of each search agent
$X_\alpha \leftarrow$ the best search agent
$X_\beta \leftarrow$ the second best search agent
$X_\delta \leftarrow$ the third best search agent
$t \leftarrow 0$
while $t <$ Max number of iterations **do**
  for each search agent **do**
    Update the position of the current search agent
  end for
  Update a, A, and C
  Calculate the fitness of all search agents
  Update the fitness of $X_\alpha, X_\beta, X_\delta$ using the following rule:
  if current alpha's fitness better than previous fitness **then**
    fitness alpha $\leftarrow$ current fitness
    fitness beta $\leftarrow$ previous alpha's fitness
    fitness delta $\leftarrow$ previous beta's fitness
  end if
  if current beta's fitness better than previous fitness **then**
    fitness beta $\leftarrow$ current fitness
    fitness delta $\leftarrow$ previous beta's fitness
  end if
  if current delta's fitness better than previous fitness **then**
    fitness delta $\leftarrow$ current fitness
  end if
  Update X_α, X_β, and X_δ
  $t \leftarrow t + 1$
end while
return X_α

4.4.2 Particle Swarm Optimization

The Particle Swarm Optimization (PSO) algorithm is a computational method used to solve optimization problems by simulating the social behavior of birds or fish. It was introduced by Eberhart and Kennedy in 1995. In recent years, the PSO algorithm has been applied to a wide variety of fields [34][35].

Finding locations in search space that are close to the global minimum or maximum solution(s) is accomplished by an algorithm. The PSO algorithm contains numerous parameters. Each particle's current position is used to calculate the fitness value at that location. The three characteristics for each particle are position ($x_i \in \mathbb{R}^n$), velocity (i.e., rate of position change; v_i), and previous best positions (p_i). Moreover, the particle with the highest fitness value is said to be in the global best location.

Every particle is represented by a set of coordinates in the search space called x_i. During the search phase, the fitness function is utilized to assess each particle's current

position. The best location is then saved in the previous best positions (p_i), and each particle's fitness value is compared to its current position after that.

The following equation was used to update the particle's velocity and compute its position:

$$V_i(t+1) = \omega V_i^t + c_1 r_1 (p_i^t - x_i^t) + c_2 r_2 (g_i^t - x_i^t) \tag{4.30}$$

$$x_i^{t+1} = x_i^t + V_i^{t+1} \tag{4.31}$$

Where, V_i^t is the individual velocity at iteration t, ω is the weight of inertia parameter, c_1 and c_2 are the acceleration coefficients, r_1 and r_2 are random numbers within [0,1], x_i^t is the individual position i at iteration t, and p_i, g_i are the best individual position and global best position at iteration t.

Algorithm 2 Particle Swarm Optimization (PSO)

Initialize particles' positions x_i and velocities v_i.
Evaluate fitness for each particle.
Set personal best $p_i = x_i$ and global best g.
for $t = 1$ to $MaxIter$ **do**
  for each particle i **do**
    Update velocity:

$$v_i = \omega v_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (g - x_i)$$

    Update position:

$$x_i = x_i + v_i$$

    Evaluate fitness at new position.
    if fitness is better than p_i **then**
      Update p_i.
    end if
    if $f(p_i)$ is better than $f(g)$ **then**
      Update global best g.
    end if
  end for
end for
Return global best g.

To modify the control parameters in relation to the suggested cost function (J), the GWO and PSO are used. Table 4.1 shows the set tuning ranges for the parameters, and Table 4.2 lists the parameters of the suggested methods.

4.5 Overall system structure

Newton's second law of motion provides the foundation for the mathematical model of the two-wheel self-balancing scooter. This model presents an inherently unstable, nonlinear system that necessitates a robust control strategy for stabilization. To control the stability and performance of the system, the design makes use of a Sliding Mode Controller, namely the Super-Twisting Sliding Mode Controller (STSMC).

The graphic below shows how the system functions as a Multi-Input Multi-Output (MIMO) system. The pitch angle and the yaw angle are the main inputs. In order to keep the scooter balanced and upright, the pitch angle is essential. In order to achieve balance, the STSMC continuously monitors this angle and compares it to a reference angle.

On the same note, the control of direction that helps the scooter move either left or right is achieved using the yaw angle. For precise control of the direction of the scooter, the STSMC receives this signal and compares it to the reference yaw angle. Advanced optimization algorithms such as Grey Wolf Optimizer (GWO) and Particle Swarm Optimization (PSO) are used for optimal tuning of the STSMC parameters to enhance performance. The decoupling unit will take the optimal controller outputs and send the appropriate control signal to the left and right DC motors.

This integrated strategy, which is based on the STSMC and backed by optimization algorithms, guarantees that the scooter will continue to be responsive and stable in a variety of situations.

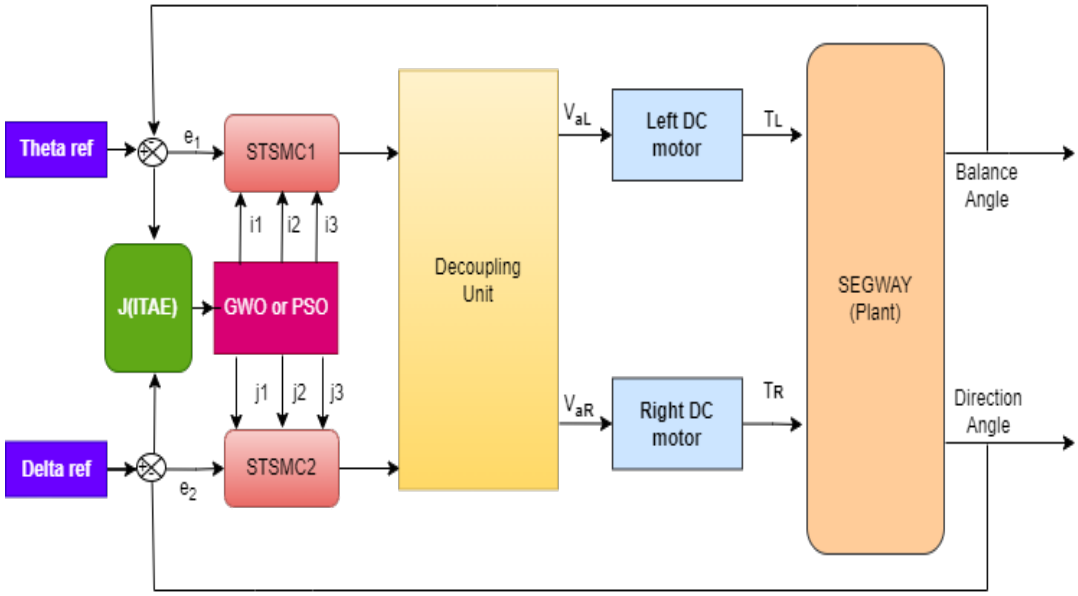


Figure 4.5: overall configuration of the system

Table 4.1: Parameters of GWO and PSO Algorithms

Algorithm	Parameters of Algorithms	Values
GWO	Number of Search Agents	20
	Maximum Iteration	30
PSO	Number of Particles (No. P)	20
	Maximum Iteration	30
	Inertia Weight (ω_{init})	0.2
	Inertia Weight (ω_{final})	0.9
	Personal Best Value (c_1)	2
	Neighborhood Best Value (c_2)	2

Table 4.2: Boundary Values for STSMC1 and STSMC2

Boundary	**STSMC1**			**STSMC2**		
	i1	**i2**	**i3**	**j1**	**j2**	**j3**
Lower bound (LB)	0.05	0.05	0.05	0.05	0.05	0.05
Upper bound (UB)	15	15	15	15	15	15

The figure below illustrates the convergence characteristics of Grey Wolf Optimizer (GWO) and Particle Swarm Optimization (PSO). As depicted, GWO demonstrates superior performance both at the initial and final stages of convergence. In contrast, PSO exhibits slower convergence in terms of both balance and direction control. Consequently, GWO outperforms PSO by achieving a lower Integral of Time-weighted Absolute Error (ITAE).

The GWO is superior to PSO in response to best fitness value (minimum ITAE) while PSO has better elapsed time to reach optimal value.

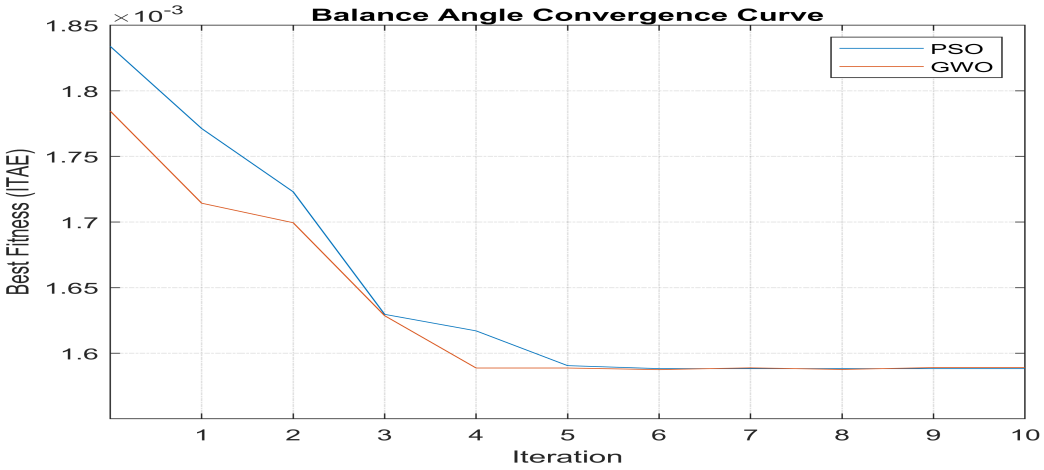


Figure 4.6: Convergence curve of GWO and PSO for balance angle.

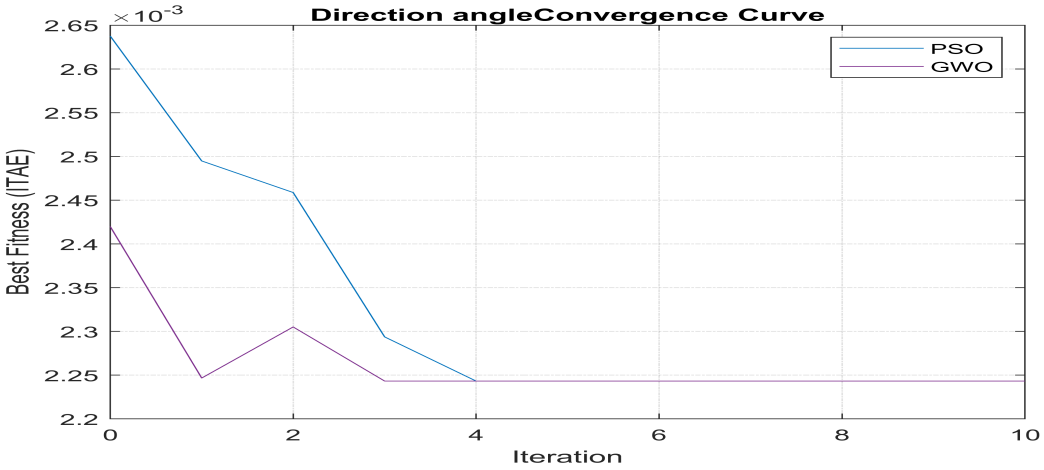


Figure 4.7: Convergence curve of GWO and PSO for direction angle.

Table 4.3: Tuned gain values

Controller Gain Coefficients	**GWO**	**PSO**
STSMC1		
i_1	12.5592	12.5362
i_2	14.9924	15
i_3	15	15
STSMC2		
j_1	12.0556	13.9488
j_2	15	14.9503
j_3	14.6605	5.5833
Optimal value (ITAE)	**(1.58+2.2) e-3**	**(1.6+2.4) e-3**
Elapsed time	2272.28	1950.15.

Chapter 5

Results and Discussion

5.1 Introduction

In this chapter, the simulation of the Segway is conducted using MATLAB/Simulink software to test and verify the performance of the designed STSMC controller for balance angle and for direction control.
The simulation was conducted using the parameters of the segway detailed in Tables 3.1 and 3.2. STSMC controller is implemented for balancing the scooter, and also employed for managing steering of wheels. The parameters for these controllers are optimized using GWO and PSO to achieve optimal system performance. The MATLAB/Simulink model for the two-wheel electric scooter is illustrated in Figure 5.1 model with STSMC.

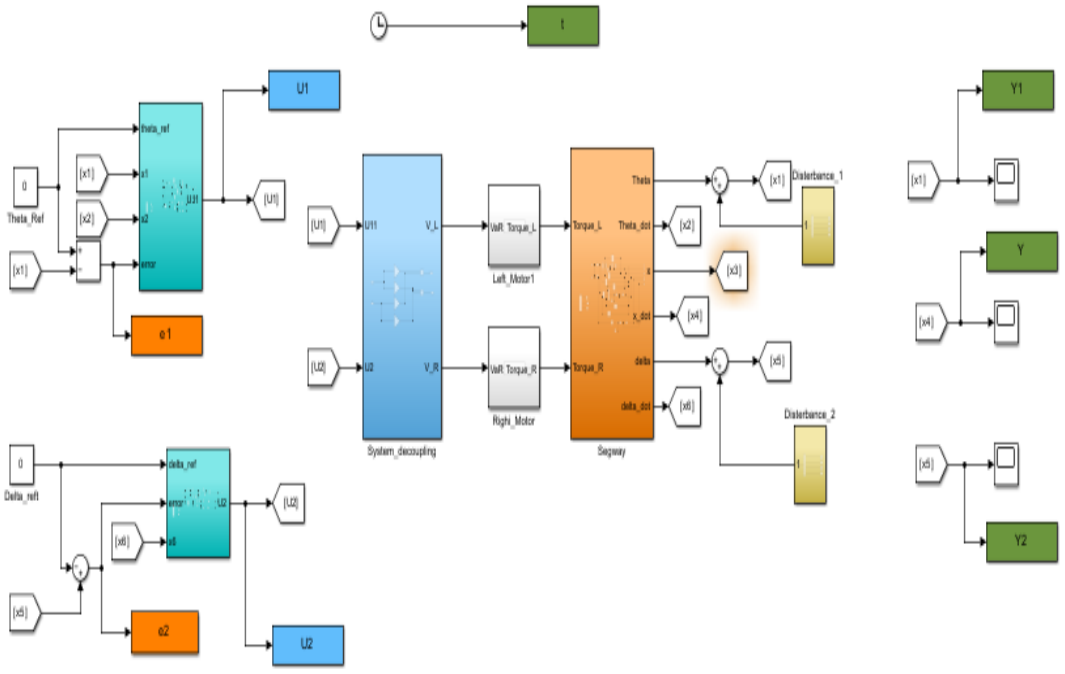


Figure 5.1: Segway model with STSMC.

5.2 Simulation Results

The parameters of the designed STSMC controller are optimized using two algorithms: GWO and PSO. Figures 5.2 and 5.3 present a comparison of these two optimization techniques for the STSMC controller. Based on the rider's input command for direction control and pitch angle for balance, the results demonstrate that GWO performs better than PSO with respect to time domain criteria, as shown in Table 5.1 below. The results for direction control and balance both show a 0 % steady-state inaccuracy, indicating that the GWO-optimized STSMC controller is effective for balancing and directional control of the Segway.

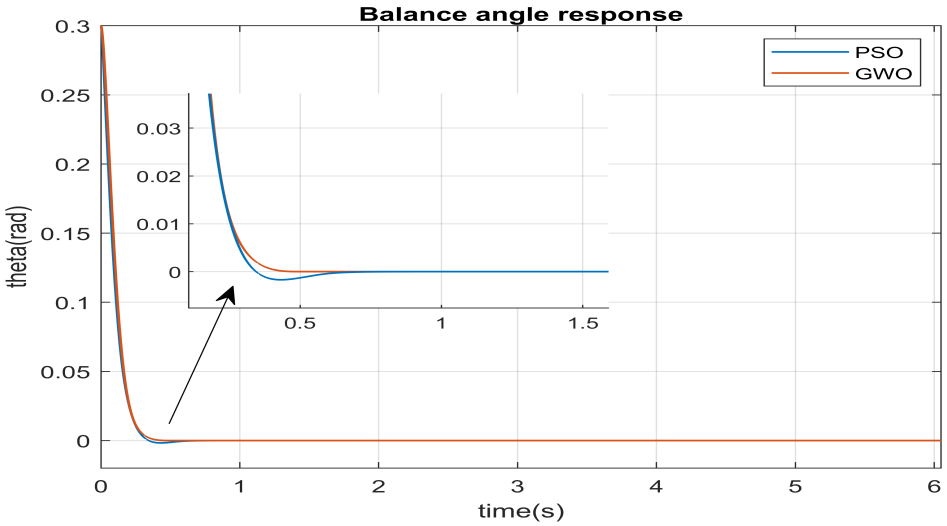


Figure 5.2: Balance angle response to rider's tilt 0.3 rad for GWO and PSO.

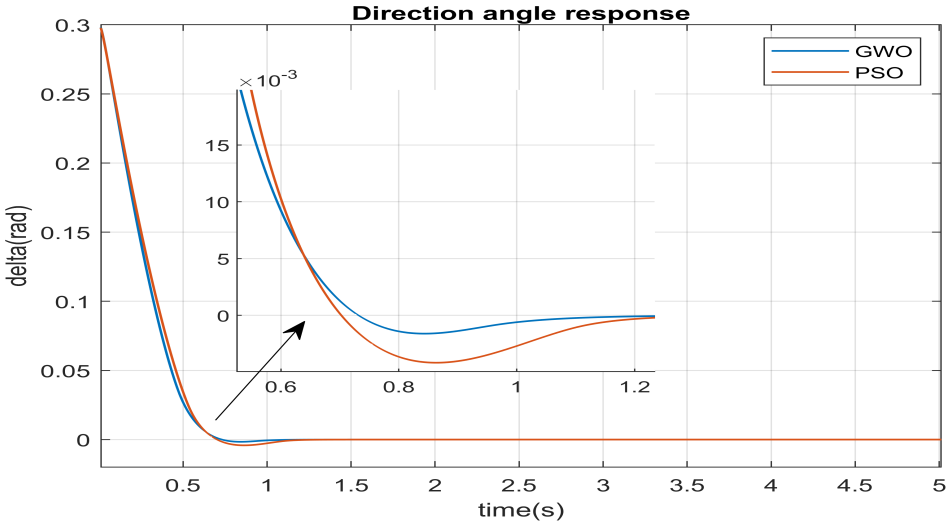


Figure 5.3: Direction angle response to rider's tilt 0.3 rad for GWO and PSO.

Table 5.1: Performance of the GWO and PSO based STSMC.

Controller	PSO		GWO	
	Balance Angle	Direction Angle	Balance Angle	Direction Angle
Rise Time (s)	0.1802	0.2794	0.1345	0.2157
Settling Time (s)	0.2538	0.3495	0.2150	0.2719
Overshoot (%)	0.16	0.05	0.12	0.049
Steady State Error	0	0	0	0

Figure 5.4 shows the response of the scooter to another initial pitch angle $\theta = -0.2$ rad, and Figure 5.5 shows the response of the scooter to an initial yaw angle at $\delta = -0.2$ rad. The pitch angle and yaw angle of the scooter converge to zero in both conditions. These results also demonstrate that the scooter can maintain balance well, and the direction change of the scooter is done perfectly for other variations of initial pitch and yaw angles.

The desired pitch and yaw angles in steady state are 0°. Values such as 0.3 rad (17.2°) or -0.2 rad ($-11.5°$) represent disturbances or initial conditions. These non-zero initial angles are used to demonstrate the system's robustness, but the final target for both angles remains 0°.

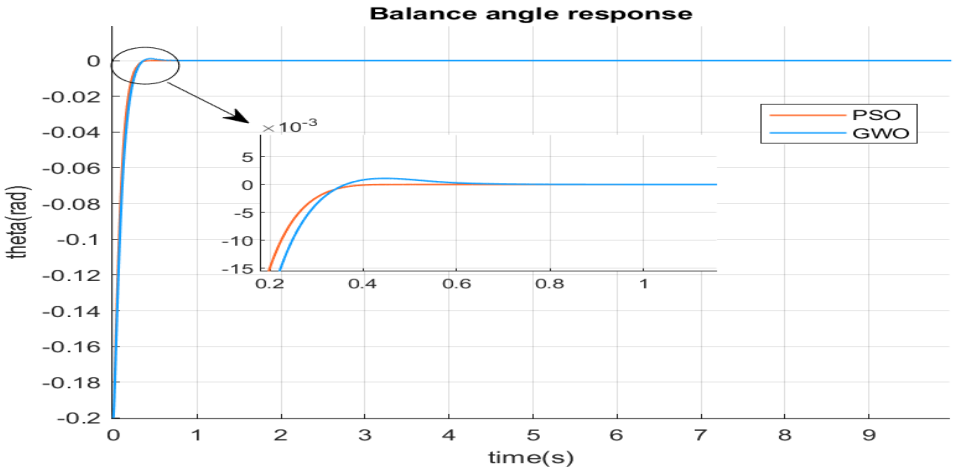


Figure 5.4: Balance angle response to rider's tilt -0.2 rad.

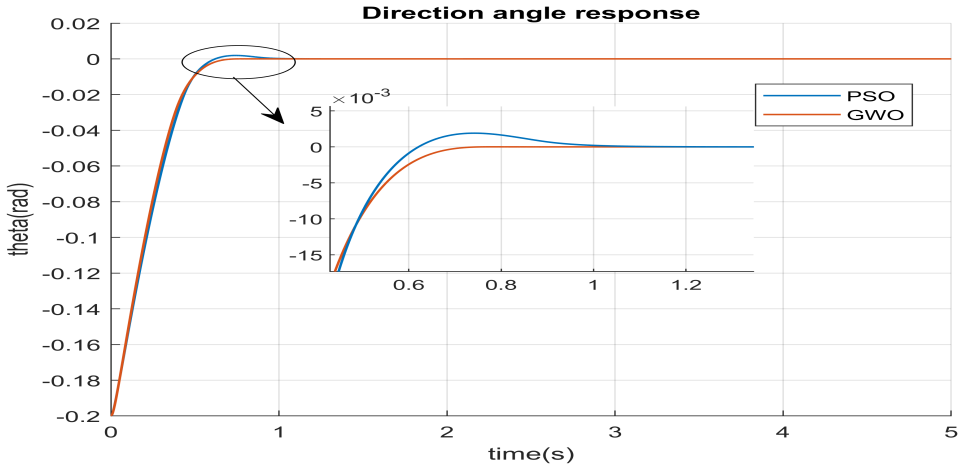
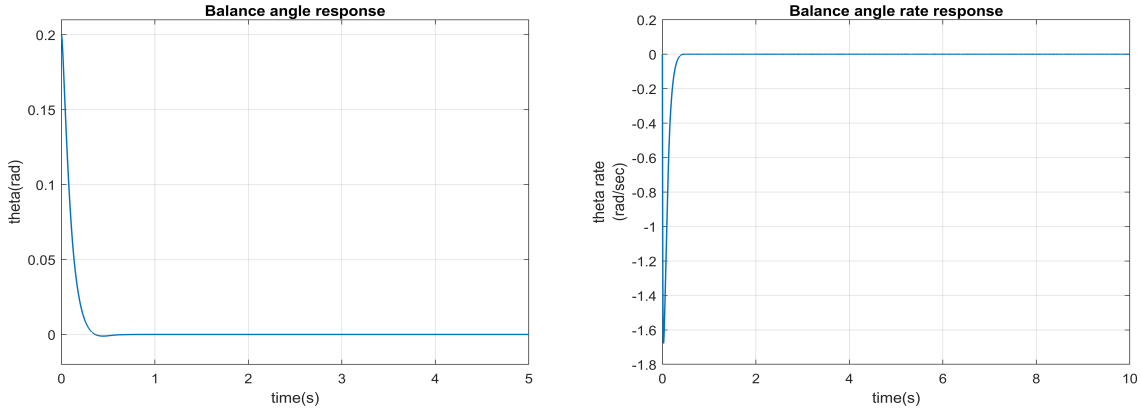


Figure 5.5: Direction angle response to rider's tilt -0.2 rad.

Figure 5.6 combines the balance angle response to an initial rider's tilt of $\theta = 0.2$ rad (Figure 5.6a) with the corresponding balance angle rate response to $\theta = 0.2$ rad (Figure 5.6b). Similarly, Figure 5.7 combines the yaw angle response to an initial condition of $\delta = 0.1$ rad (Figure 5.7a) with the corresponding yaw angle rate response to $\delta = 0.1$ rad (Figure 5.7b).

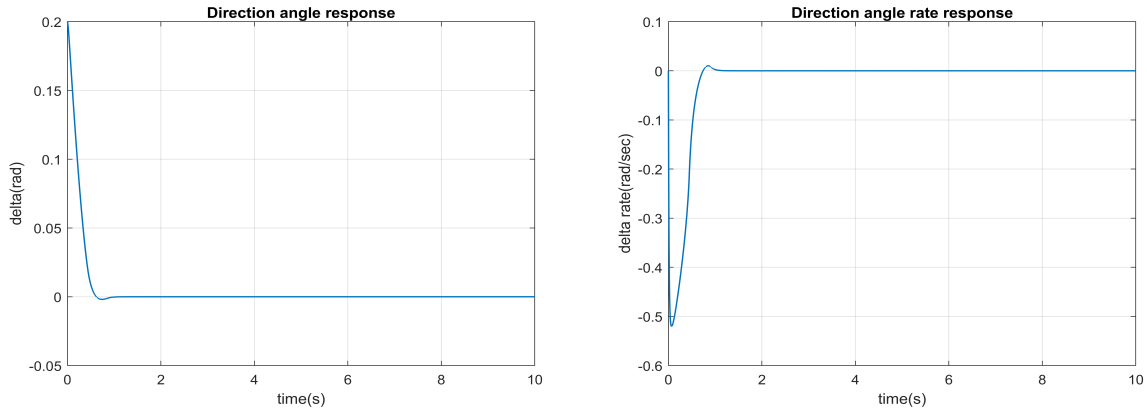
In all cases, the angles and angular rates converge to zero, demonstrating the system's stability under various initial conditions. These results highlight the effectiveness of the control system in maintaining balance and ensuring smooth directional adjustments.



(a) Balance angle response to rider's tilt $\theta = 0.2$ rad.

(b) Balance angle rate response to rider's tilt $\theta = 0.2$ rad.

Figure 5.6: Balance angle and balance angle rate responses to rider's tilt.



(a) Direction angle response to rider's tilt $\delta = 0.1$ rad.

(b) Direction angle rate response to rider's tilt $\delta = 0.1$ rad.

Figure 5.7: Direction angle and direction angle rate responses to rider's tilt.

In this work, the Segway is designed to carry a human load of up to 100 kg. Figure 5.8 illustrates the response to a varying human load between 60 kg and 100 kg for an initial pitch angle of $\theta = 0.3$ rad. Similarly, Figure 5.9 shows the response for a varying human load from 60 kg to 100 kg, but for an initial yaw angle of $\delta = 0.2$ rad. The results demonstrate that, despite the mass variations, the Segway maintains good dynamic performance in both balance and direction control. For the initial pitch angle $\theta = 0.3$

rad, the system adjusts effectively to stabilize the Segway, while for the initial yaw angle $\delta = 0.2$ rad, the direction is kept within desired limits. One of the key advantages of employing Robust nonlinear control techniques, such as STSMC , is their low sensitivity to changes in system parameters, including human mass. This robustness ensures consistent and reliable performance even under varying load conditions, making the system highly adaptive to real-world scenarios where the rider's weight may fluctuate.

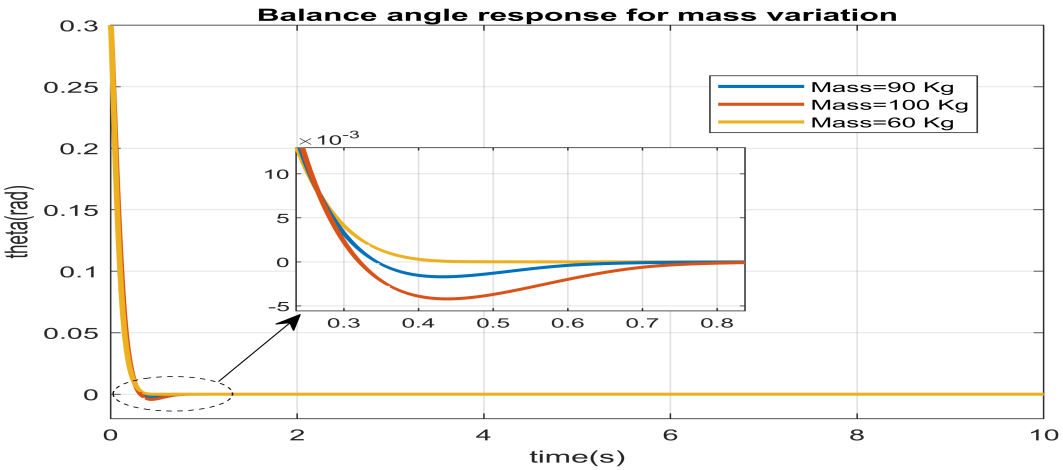


Figure 5.8: Balance angle response with initial $\theta = 0.3$ rad for mass variation from 60 kg to 100 kg.

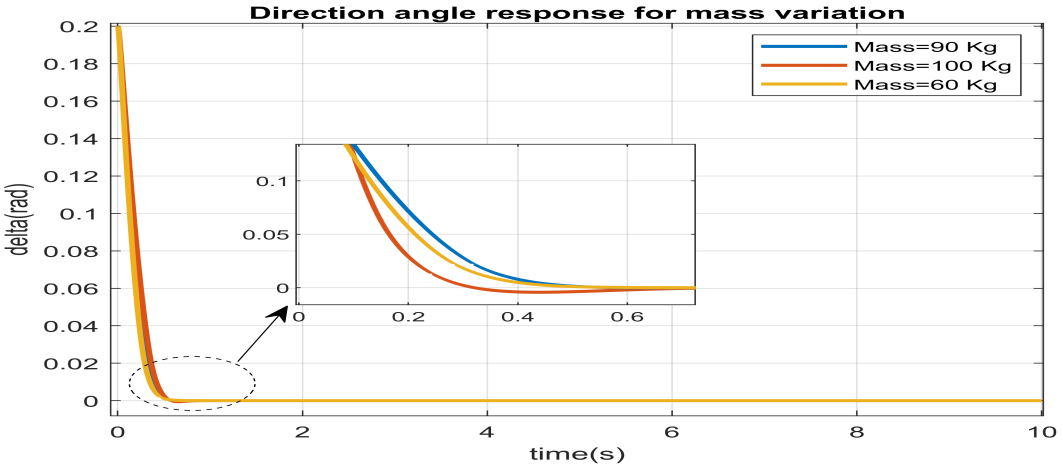


Figure 5.9: Direction angle response with initial $\delta = 0.2$ rad for mass variation from 60 kg to 100 kg.

Figures 5.10 and 5.11 illustrate the system's response to a rider-induced tilt angle of $\theta = 0.2$ rad and a directional input angle of $\delta = 0.2$ rad, accounting for the effects of uneven terrain. The analysis considers ground disturbances occurring between $t = 2$ and $t = 4$ seconds, with the predefined values of dx, dy, dHL, dHR, and $d\delta$ specified below. The results indicate that the system exhibits low sensitivity to minor terrain variations, maintaining stable performance.

$$\begin{cases} dy = 0.3 \cdot \sin(10t) \\ dx = 0.3 \cdot \cos(10t) \\ dHL = 0.04 \cdot \cos(6t) \\ dHR = 0.04 \cdot \cos(6t) \\ d\delta = 4.5 \cdot \exp(-3.5t) \end{cases}$$

These adjustments reflect the impact of small terrain variations, making the system more representative of real-world disturbances. The disturbances are considered in the time range $t = 2$ to $t = 4$ seconds, where the response remains stable and exhibits low sensitivity to these variations.

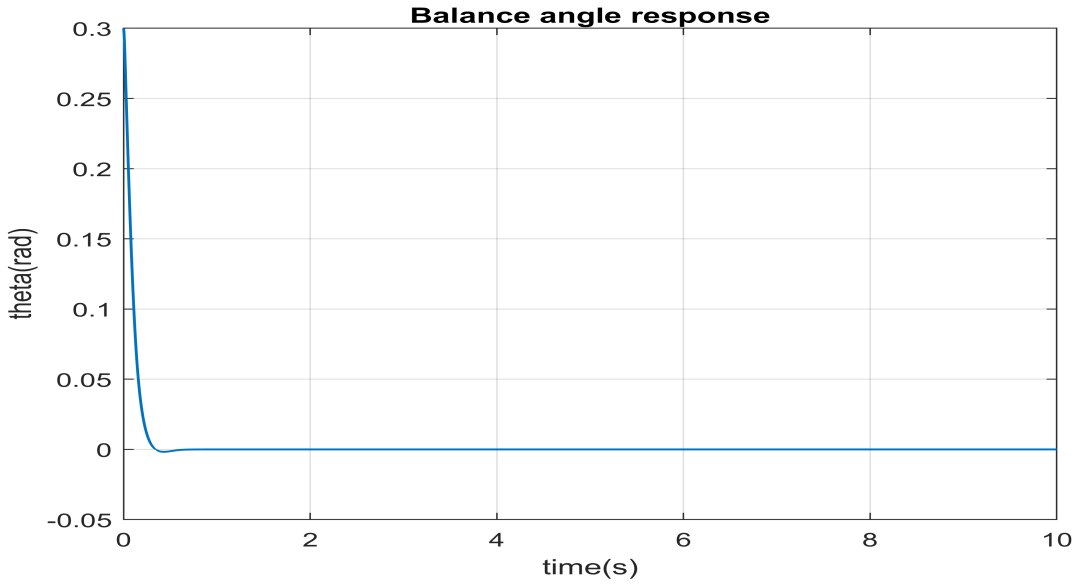


Figure 5.10: Balance angle response with initial $\theta = 0.3$ rad for terrain disturbance.

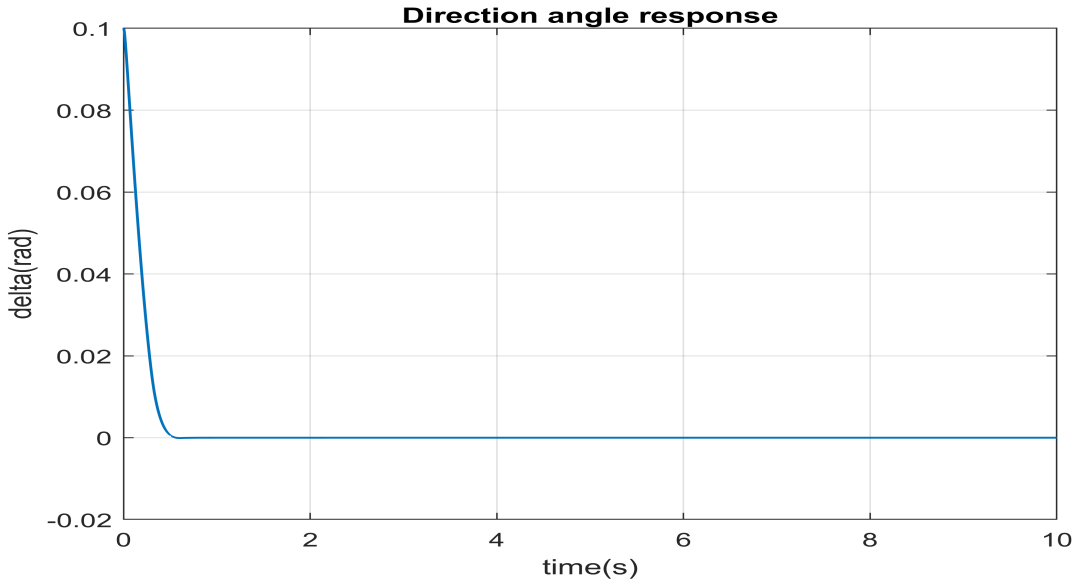


Figure 5.11: Direction angle response with initial $\delta = 0.1$ rad for terrain disturbance.

5.3 Simulation Result for Balance and Direction Angle Variation

Figure 5.12 shows the balance angle response for the rider's tilt order pitch angle $\theta = 0.3[U(t) - U(t-2) - U(t-4) + U(t-6)]$, demonstrating that the Segway can effectively maintain balance despite varying pitch angle changes. The system adjusts accordingly to these changes in input and helps maintain the stability of the angle of balance irrespective of any tilt of the rider. This shows the efficiency of the control scheme used, as it helps adjust the response of the Segway to ensure that its balancing action is not compromised at all times. Being able to manage several angles of tilt while maintaining balance is extremely important for the efficient functioning of the Segway.

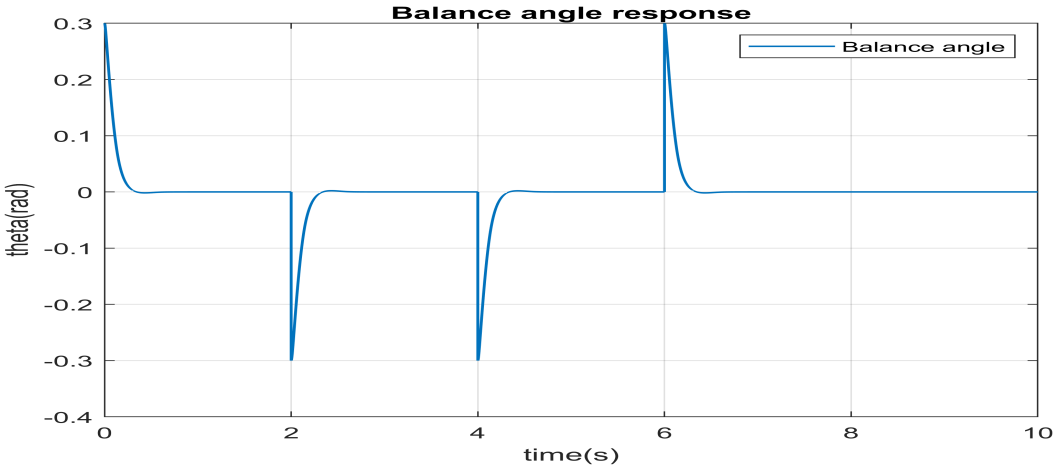


Figure 5.12: response of balance angle for the rider's tilt order.

Figure 5.13 shows the direction angle response for the rider's order angle $\delta = 0.2[U(t-1) - U(t-3) - U(t-5) + U(t-7)]$, demonstrating the system's superior performance in handling yaw angle changes over time. The results indicate that the proposed control system is effective in adjusting the Segway's direction accurately, ensuring it stays aligned with the desired trajectory. This is particularly important for maintaining smooth and reliable steering, especially during dynamic or rapid changes in the rider's input. The system's ability to handle these directional shifts effectively ensures that the Segway remains stable and on course, making it highly robust for real-world navigation tasks.

Figures 5.14 and 5.15 illustrate the system's response when the predefined test signals from Figures 5.12 and 5.13 are applied on uneven terrain. The results demonstrate that the proposed controllers effectively compensate for disturbances, maintaining both balance and the desired directional angle.

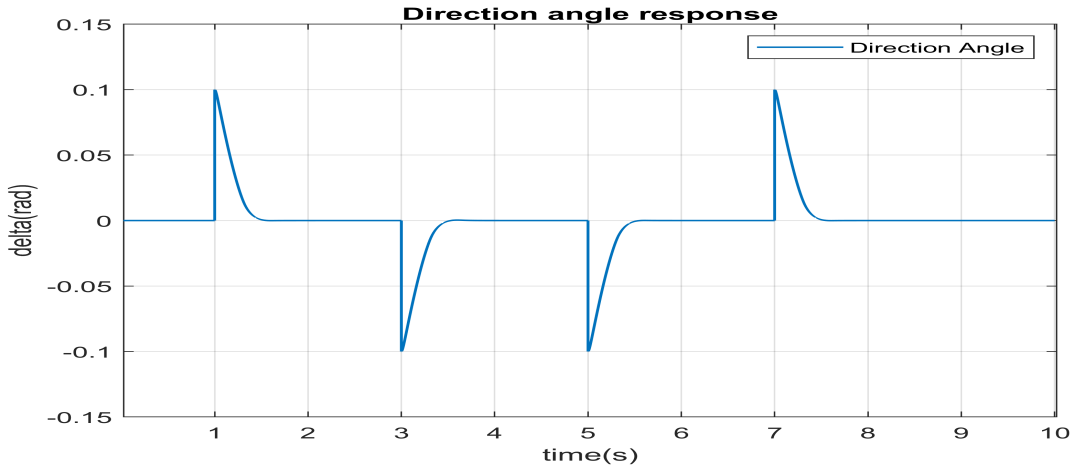


Figure 5.13: response of direction angle for the rider's order.

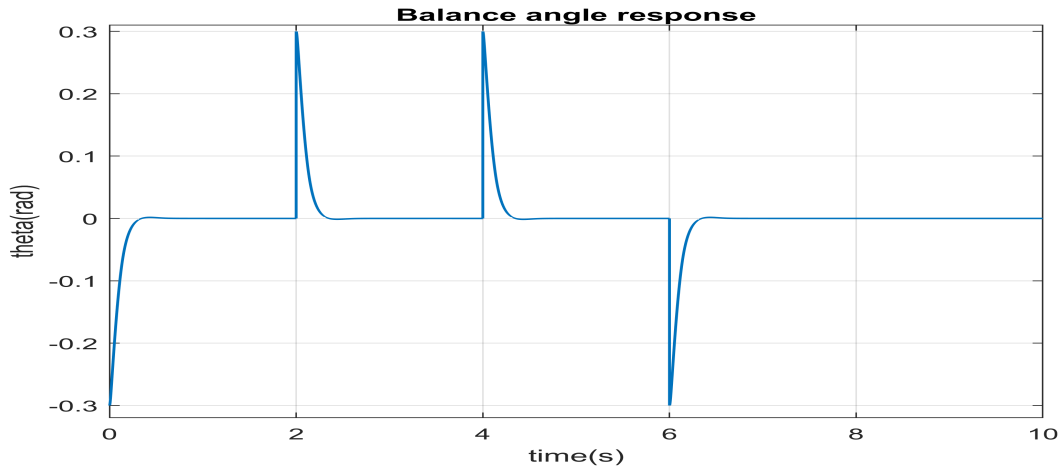


Figure 5.14: Response of balance angle for the rider's tilt order.

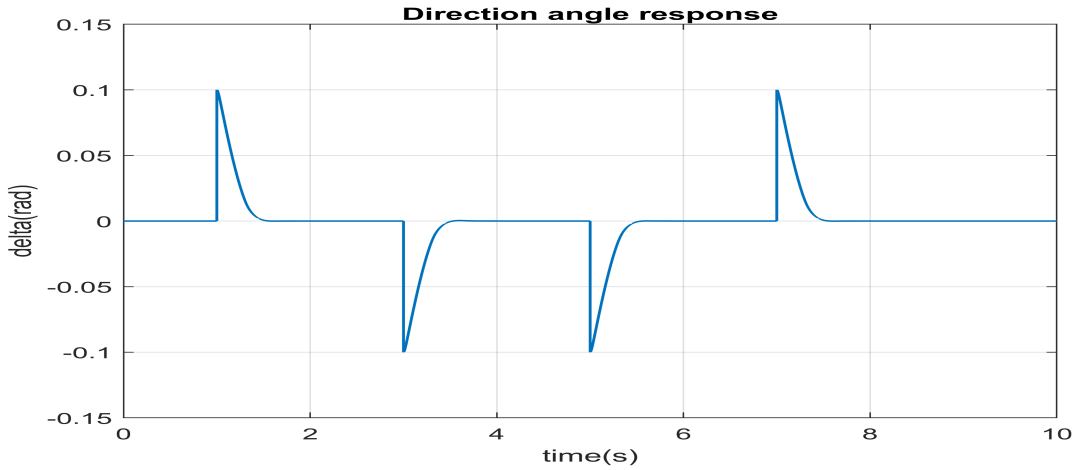


Figure 5.15: Response of direction angle for the rider's order.

The speed response of the Segway under terrain disturbances is illustrated in the figure below. As observed, the speed exhibits variations between 2 and 4 seconds due to ground

disturbances. This behavior demonstrates that the controller effectively compensates for the disturbance by adjusting the speed, ensuring the Segway remains upright and stable.

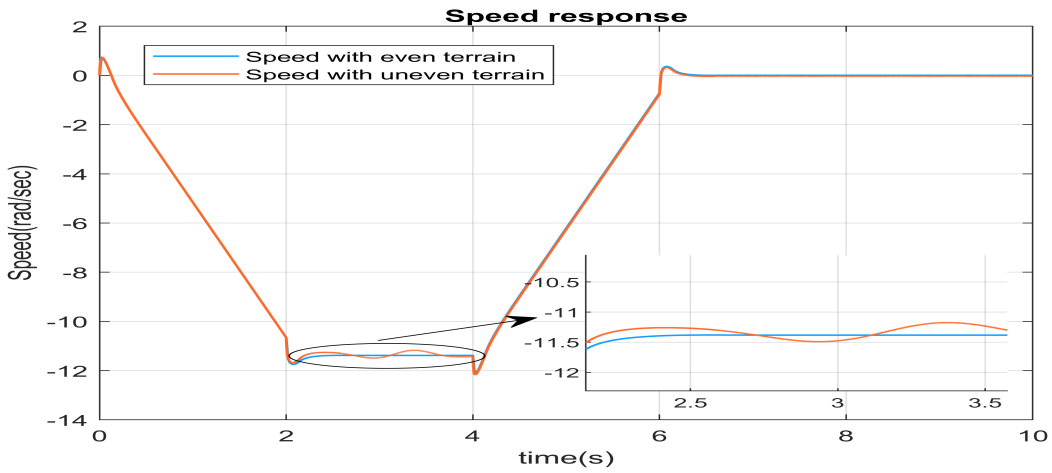


Figure 5.16: Speed response of Segway with uneven terrain.

The control current signals generated by the STSMC controllers, responsible for regulating the balance and direction angles, are depicted in Figures 5.17 and 5.18, respectively.

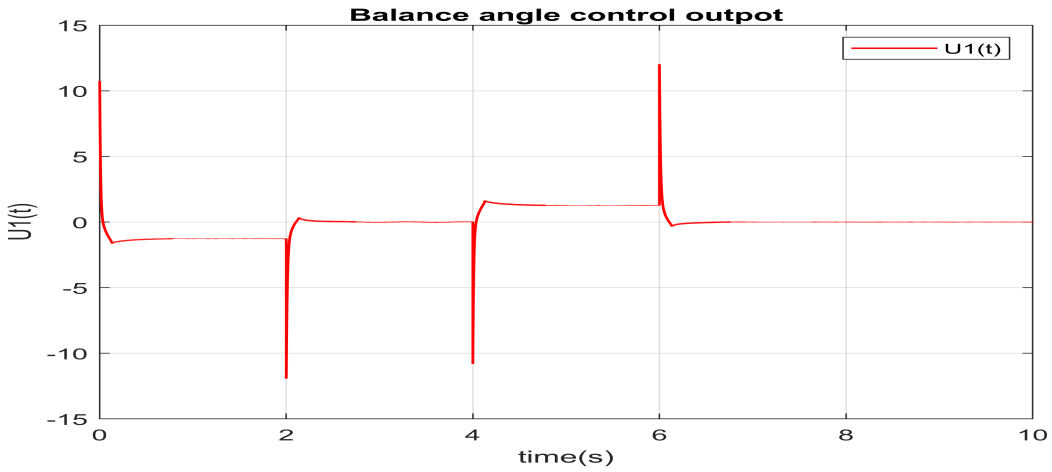


Figure 5.17: Balance angle controller output U_1.

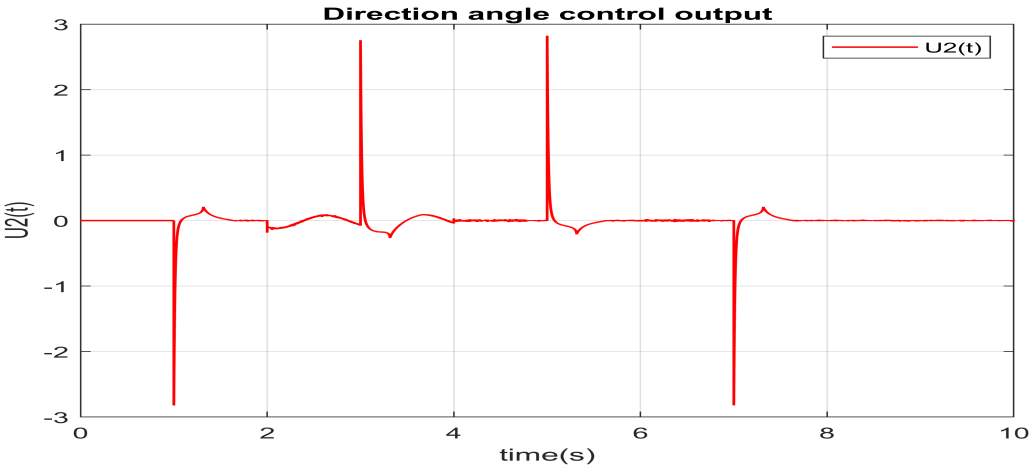


Figure 5.18: Direction angle controller output U_2.

Figure 5.19 presents the pitch angle response for an initial pitch angle of $\theta = 0.3\,[U(t) - U(t-2) - U(t-4) + U(t-6)]$.The results illustrate that the Segway effectively maintains balance and stabilizes under varying pitch angle conditions and mass variation.

Similarly, Figure 5.20 depicts the yaw angle response for an initial yaw angle of $\delta = 0.2\,[U(t-1) - U(t-3) - U(t-5) + U(t-7)]$. The results indicate that the scooter achieves precise directional control under different yaw angle variations and mass variation.

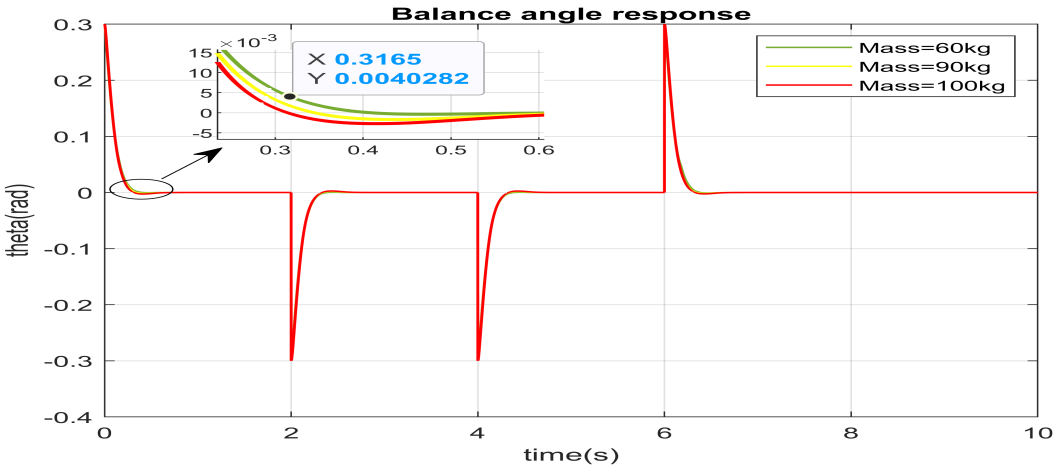


Figure 5.19: Balance angle response for different initial angle and mass.

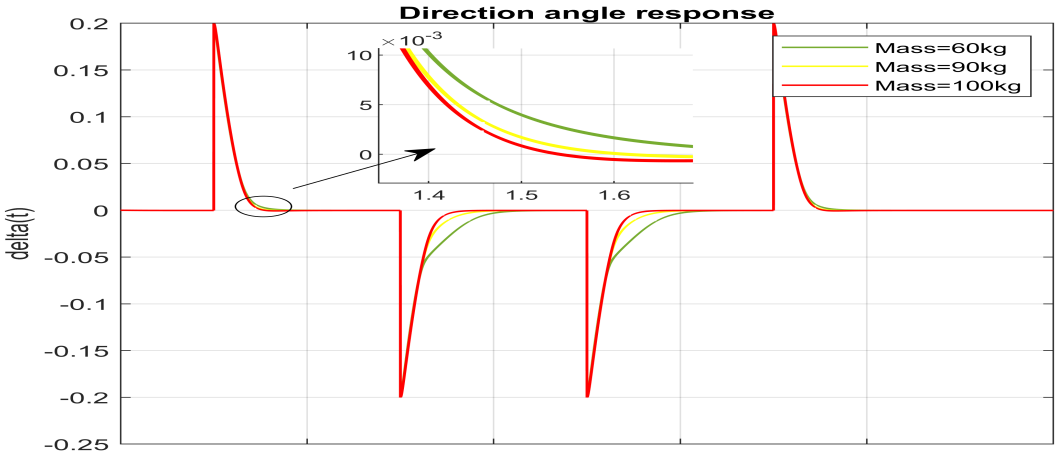


Figure 5.20: Direction angle response for different initial angle and mass.

Finally, to enhance the realism of the test signals by eliminating abrupt transitions and replacing them with gradual changes using a ramp function, the controller's performance was evaluated. The results demonstrate excellent performance in both balance and directional angle control.Figure 5.21 and 5.22 shows the test signals used for balance and direction order.

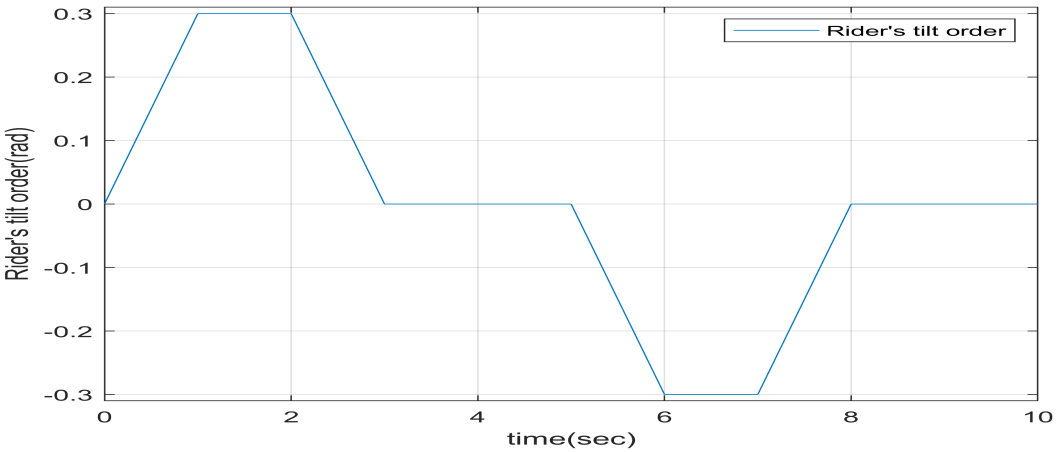


Figure 5.21: Balance angle order.

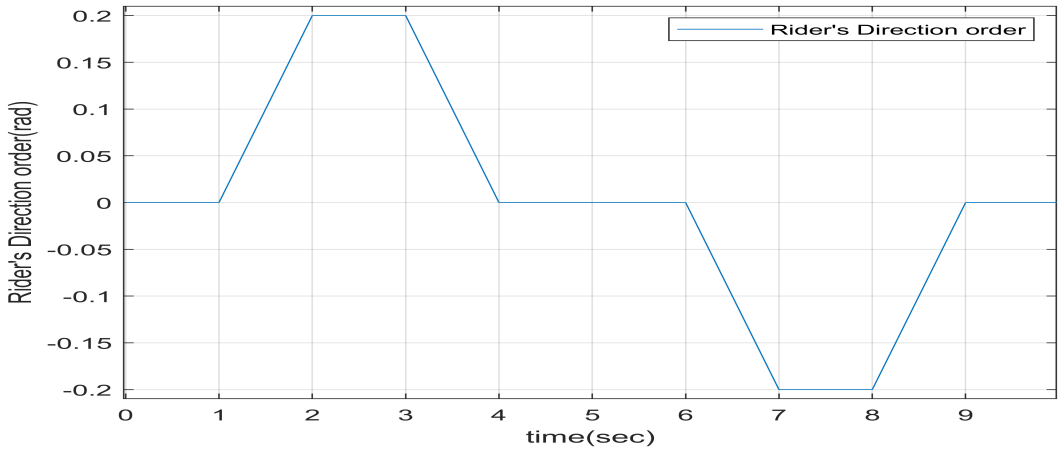


Figure 5.22: Direction angle order.

The rider's commands are generated by first creating ramp signals, which are then combined to produce a pulse signal. Figure 5.23 illustrates the balance angle response for the test signal, which mimics real-world input signals as shown in Figure 5.21. The results indicate that the Segway effectively maintains balance across various pitch angle variations, demonstrating robust stability under different conditions.

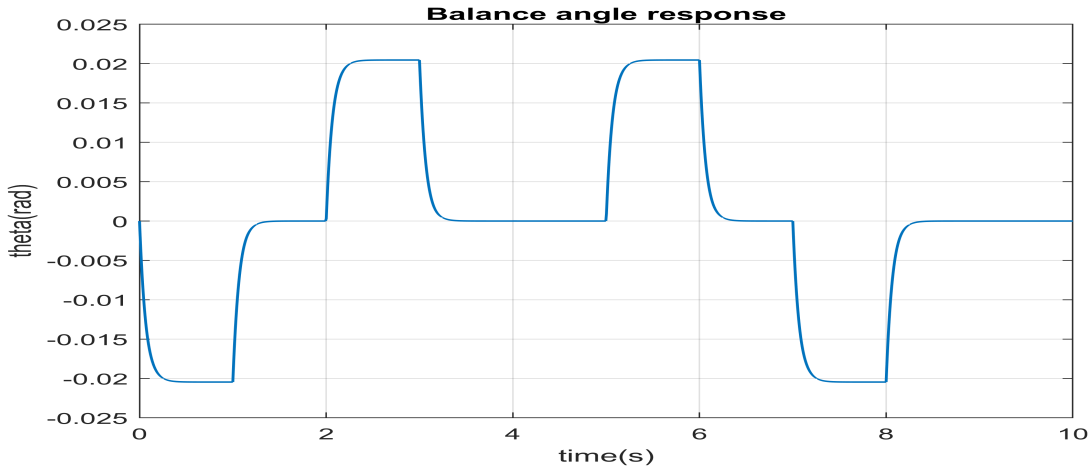


Figure 5.23: Balance angle response for rider order

Figure 5.24 presents the direction angle response corresponding to the test signal, designed to closely simulate real-world inputs as depicted in Figure 5.22. The results highlight the Segway's ability to maintain stable balance across a range of pitch angle variations, effectively demonstrating its capability to adapt to dynamic changes in the rider's inputs.

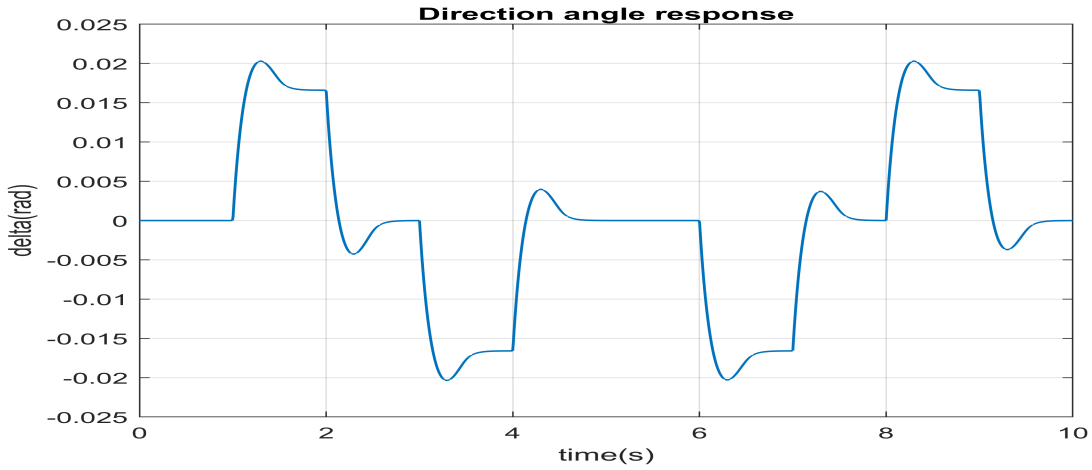


Figure 5.24: Direction angle response for rider order.

The control current signals produced by the Super Twisting Sliding Mode Controllers (STSMC), which are responsible for regulating both the balance and directional angles, are presented in Figures 5.25 and 5.26, respectively. These figures illustrate the signals corresponding to the balance and direction control mechanisms, highlighting the effectiveness of the controllers in achieving the desired performance in each aspect.

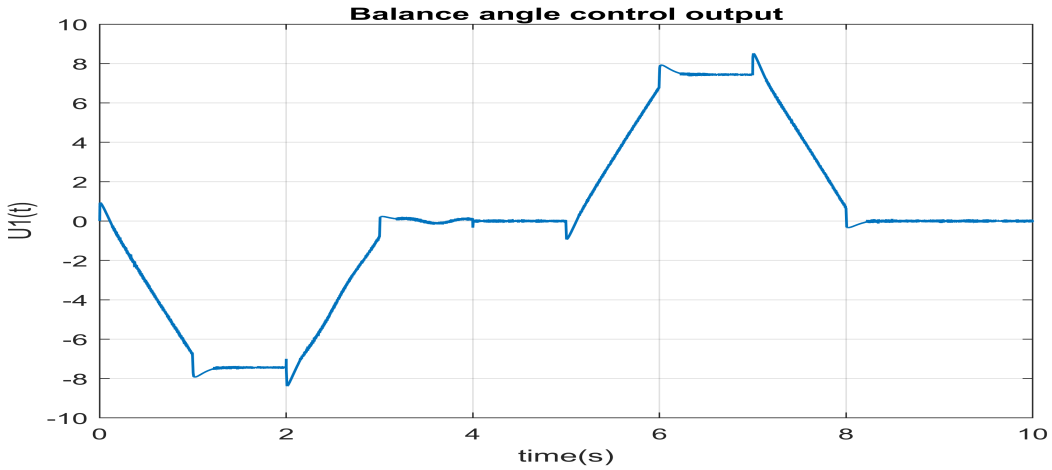


Figure 5.25: Balance angle controller output U_1.

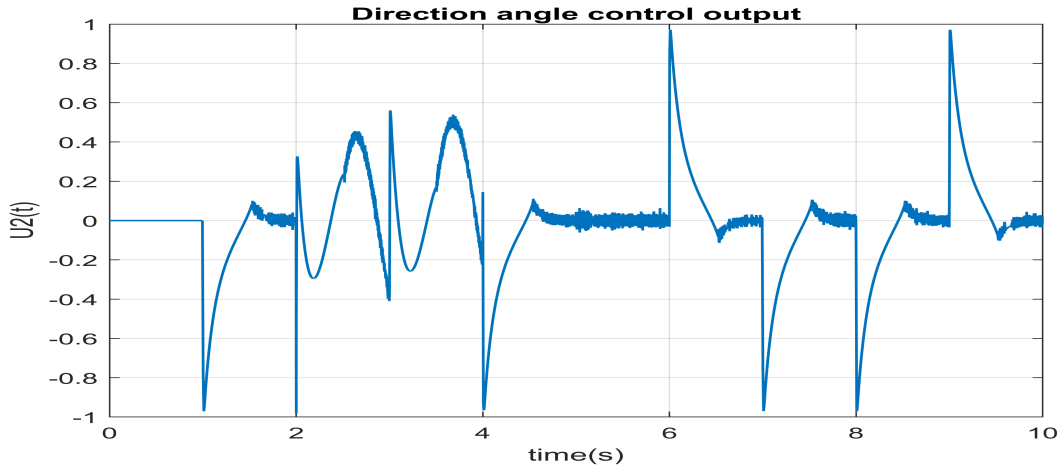


Figure 5.26: Direction angle controller output U_2.

Finally, let us assume that an external force is applied to the rider, affecting both its tilt and direction. Figure 5.27 illustrates the Segway's balance response when subjected to an external force disturbance applied between 4 and 5 seconds, with an impact rate of 0.1 radians in tilt angle. The results demonstrate the Segway's ability to maintain stability, keeping the rider upright.

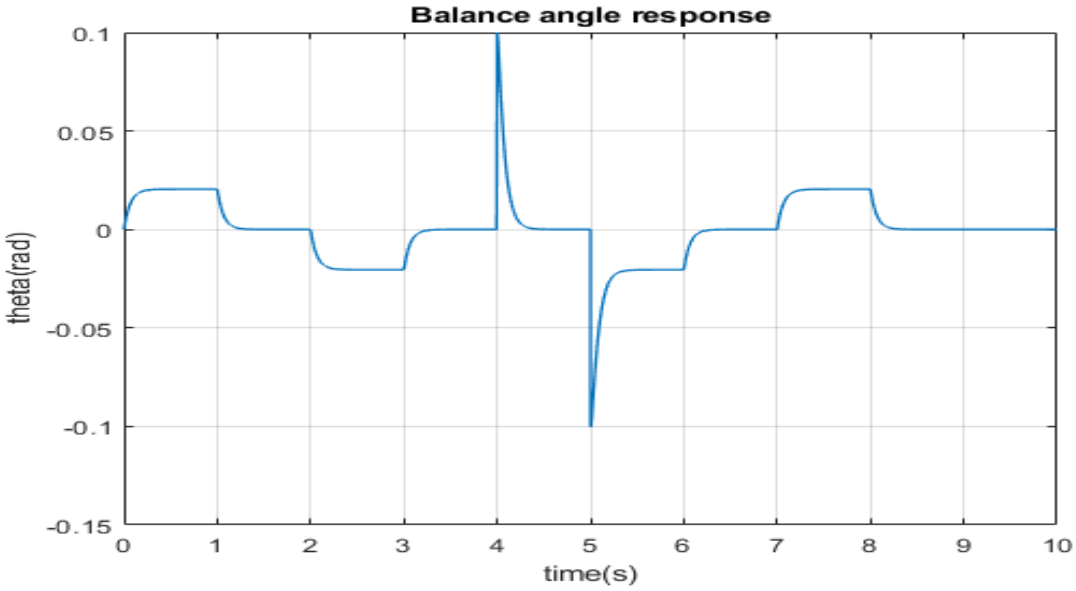


Figure 5.27: Segway balance response under an external force disturbance.

Similarly, Figure 5.28 presents the direction angle response under the same disturbance, influencing the rider's twisting motion. The results indicate that the Segway successfully maintains the required wheel steering direction, ensuring stability and proper maneuverability.

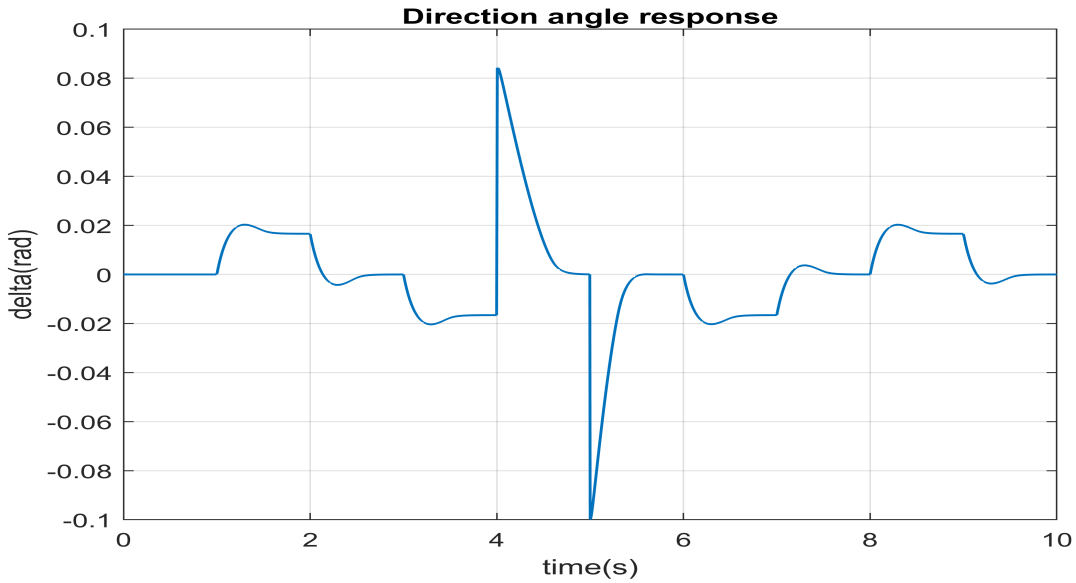


Figure 5.28: Segway direction response under an external force disturbance.

In summary, the Segway system is effectively controlled through various tilt and directional commands. The simulation results confirm that the two-wheeled electric scooter operates with stability and demonstrates excellent performance.

Chapter 6

Conclusion and Future Work

6.1 Conclusions

Since the self-balancing scooter, which is considered to be an inverted pendulum, is inherently nonlinear and highly unstable, any slight disturbance of the inputs in an open-loop design can easily result in instability. With the help of knowledge on inverted pendulum dynamics, it was proposed to use the Super Twisted Sliding Mode Controller (STSMC) as an efficient means to stabilize the inverted pendulum as well as control its direction and stability. The crucial parameters like mass of the body, mass and radius of the wheels, and characteristics of the DC motors were calculated and modeled using MATLAB/Simulink to precisely represent the dynamics of the inverted pendulum. In this regard, the Grey Wolf Optimizer (GWO), a powerful metaheuristic optimization approach, was employed for optimal tuning of the controller. From the results, it has been observed that GWO shows superior optimization capabilities by achieving optimal fitness value through reduced computation time. Simulation results reveal that, irrespective of the conditions, the inverted pendulum using the optimized STSMC by the Grey Wolf Optimizer is capable of balancing and maintaining direction with accuracy. The system effectively balances at different angles and remains stable while carrying a load of maximum 100 kg.

6.2 Recommendations

Self-balancing scooters have attracted considerable interest due to the simplicity and flexibility that personal mobility devices have exhibited. Through simulations carried out using the MATLAB/Simulink software, this paper proves the proficiency with which the Super Twisted Sliding Mode Controller (STSMC), which was optimized using Grey Wolf Optimizer (GWO) algorithm, controls not only balance but also steering of a two-wheeled self-balancing scooter. The main area of future research should concentrate on implementing the proposed controllers in the experimental setup, thus making it possible to learn about the reliability of the algorithms used. Despite the high resistance level that has been demonstrated by the STSMC, uncertainties and changes that occur when it is in use in the real world call for improvements aimed at enhancing adaptability. Adaptive controllers, as opposed to static controllers, seem promising as tools for coping with uncertainty and parameter changes because of their dynamic character. Another potential area of investigation could include expanding the capabilities of the scooter to increase their functionality. For instance, incorporating load transportation features

may enhance utility of the device considerably. The scooter would be an invaluable tool for small business operations and urban logistics if it were equipped with modular attachments for carrying tools, commodities, or small equipment.

Bibliography

[1] I. Boiko and L. Fridman, "Analysis of chattering in continuous sliding-mode controllers," *IEEE transactions on automatic control*, vol. 50, no. 9, pp. 1442–1446, 2005.

[2] Wikipedia, "Segway - wikipedia," https://en.wikipedia.org/wiki/Segway, accessed: Feb. 15, 2024.

[3] F. Grasser, A. D'arrigo, S. Colombi, and A. C. Rufer, "Joe: a mobile, inverted pendulum," *IEEE Transactions on industrial electronics*, vol. 49, no. 1, pp. 107–114, 2002.

[4] H. H. Mohsin, A. A. Aldair, and W. A. Al-Hussaibi, "Robust control design for two-wheel self-balanced mobile robot," *Iraqi Journal for Electrical and Electronic Engineering*, vol. 19, no. 1, pp. 38–46, Jun 2023.

[5] J. Meng, A. Liu, Y. Yang, Z. Wu, and Q. Xu, "Two-wheeled robot platform based on pid control," in *2018 5th International Conference on Information Science and Control Engineering (ICISCE)*. IEEE, 2018, pp. 1011–1014.

[6] V. Mudeng, B. Hassanah, Y. T. K. Priyanto, and O. Saputra, "Design and simulation of two-wheeled balancing mobile robot with pid controller," *International Journal of Sustainable Transportation Technology*, vol. 3, no. 1, pp. 12–19, 2020.

[7] S. Kim and S. Kwon, "Nonlinear optimal control design for underactuated two-wheeled inverted pendulum mobile platform," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 6, pp. 2803–2808, 2017.

[8] L. Guo, S. A. A. Rizvi, and Z. Lin, "Optimal control of a two-wheeled self-balancing robot by reinforcement learning," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 6, pp. 1885–1904, 2021.

[9] P. Bozek, Y. L. Karavaev, A. A. Ardentov, and K. S. Yefremov, "Neural network control of a wheeled mobile robot based on optimal trajectories," *International Journal of Advanced Robotic Systems*, vol. 17, no. 2, p. 1729881420916077, 2020.

[10] A. A. Ahmed and A. S. Alshandoli, "On replacing a pid controller with neural network controller for segway," in *2020 International Conference on Electrical Engineering (ICEE)*. IEEE, 2020, pp. 1–4.

[11] Y. Yang, X. Yan, K. Sirlantzis, and G. Howells, "Application of sliding mode trajectory tracking control design for two-wheeled mobile robots," in *2019 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. IEEE, 2019, pp. 109–114.

[12] J.-X. Xu, Z.-Q. Guo, and T. Lee, "Design and implementation of integral sliding-mode control on an underactuated two-wheeled mobile robot," *Industrial Electronics, IEEE Transactions on*, vol. 61, pp. 3671–3681, 07 2014.

[13] I. F. Bouguenna, A. Azaiz, A. Tahour, and A. Larbaoui, "Electronic differential and neuro-fuzzy sliding mode control with extended state observer for an electric vehicle system," in *E3S Web of Conferences*, vol. 61. EDP Sciences, 2018, p. 00007.

[14] B. Porter, "Design of tracking systems for a class of multivariable linear systems with slow and fast modes," *IFAC Proceedings Volumes*, vol. 10, no. 6, pp. 113–117, 1977.

[15] U. Adeel, K. Alimgeer, O. Inam, A. Hameed, M. Qureshi, and M. Ashraf, "Autonomous dual wheel self balancing robot based on microcontroller," *Journal of Basic and Applied Scientific Research*, vol. 3, no. 1, pp. 843–848, 2013.

[16] A. K. Chari *et al.*, "Stability analysis of lqr-anfis control schemes on 2-degree-of-freedom inverted pendulum systems," *arXiv preprint arXiv:2312.06346*, 2023.

[17] P. Sevekari, B. Tamhane, and S. Kurode, "Robust control for stable and safe performance of a two wheeled human transporter," *IFAC-PapersOnLine*, vol. 53, no. 1, pp. 616–621, 2020.

[18] I. Chawla, V. Chopra, and A. Singla, "Performance comparison of pid and anfis controller for stabilization of x and xy inverted pendulums," in *Intelligent Systems Design and Applications: 18th International Conference on Intelligent Systems Design and Applications (ISDA 2018) held in Vellore, India, December 6-8, 2018, Volume 1*. Springer, 2020, pp. 182–192.

[19] J. Mu, X.-G. Yan, S. K. Spurgeon, and Z. Mao, "Nonlinear sliding mode control of a two-wheeled mobile robot system," *International Journal of Modelling, Identification and Control*, vol. 27, no. 2, pp. 75–83, 2017.

[20] M. Ayyıldız and U. Tilki, "Adaptive sliding mode based fault tolerant control of wheeled mobile robots," *Automatika*, vol. 64, no. 3, pp. 467–483, 2023.

[21] Y. Wang and G. Zhu, "Performance improvement of an nmp mini segway using sample and hold inputs," *Applied Sciences*, vol. 13, no. 2, p. 1070, 2023.

[22] N. N. Son and H. P. H. Anh, "Adaptive backstepping self-balancing control of a two-wheel electric scooter," *International Journal of Advanced Robotic Systems*, vol. 11, no. 10, p. 165, 2014.

[23] S.-C. Lin, C.-C. Tsai, and H.-C. Huang, "Adaptive robust self-balancing and steering of a two-wheeled human transportation vehicle," *Journal of Intelligent & Robotic Systems*, vol. 62, pp. 103–123, 2011.

[24] Z. Zheng and M. Teng, "Modeling and decoupling control for two-wheeled self-balancing robot," in *2016 Chinese Control and Decision Conference (CCDC)*. IEEE, 2016, pp. 5263–5267.

[25] 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, no. 1, pp. 47–55, 2018.

[26] Z. Jie and R. Sijing, "Sliding mode control of inverted pendulum based on state observer," in *2016 Sixth International Conference on Information Science and Technology (ICIST)*. IEEE, 2016, pp. 322–326.

[27] B. Yan, P. Dai, R. Liu, M. Xing, and S. Liu, "Adaptive super-twisting sliding mode control of variable sweep morphing aircraft," *Aerospace Science and Technology*, vol. 92, pp. 198–210, 2019.

[28] J. Liu, L. Gao, J. Zhang, and F. Yan, "Super-twisting algorithm second-order sliding mode control for collision avoidance system based on active front steering and direct yaw moment control," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 235, no. 1, pp. 43–54, 2021.

[29] Z. Feng and J. Fei, "Design and analysis of adaptive super-twisting sliding mode control for a microgyroscope," *PloS one*, vol. 13, no. 1, p. e0189457, 2018.

[30] Y. Wang, J.-P. Li, X. Xue, and B.-C. Wang, "Utilizing the correlation between constraints and objective function for constrained evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 29–43, 2019.

[31] S. Mirjalili, S. Mirjalili, and A. Lewis, "Grey wolf optimizer advances in engineering software. 69 46–61," 2014.

[32] R. Fessi, H. Rezk, and S. Bouallègue, "Grey wolf optimization based tuning of terminal sliding mode controllers for a quadrotor," *Comput. Mater. Contin*, vol. 68, pp. 2256–2282, 2021.

[33] N. Singh and S. Singh, "Hybrid algorithm of particle swarm optimization and grey wolf optimizer for improving convergence performance," *Journal of Applied Mathematics*, vol. 2017, no. 1, p. 2030489, 2017.

[34] P. Dziwiński, Ł. Bartczuk, and J. Paszkowski, "A new auto adaptive fuzzy hybrid particle swarm optimization and genetic algorithm," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 10, no. 2, pp. 95–111, 2020.

[35] Ü. Önen, A. Çakan, and İ. İlhan, "Particle swarm optimization based lqr control of an inverted pendulum," *Engineering and Technology Journal*, vol. 2, no. 5, pp. 168–174, 2017.

Appendixes

Appendix 1: Simulink block

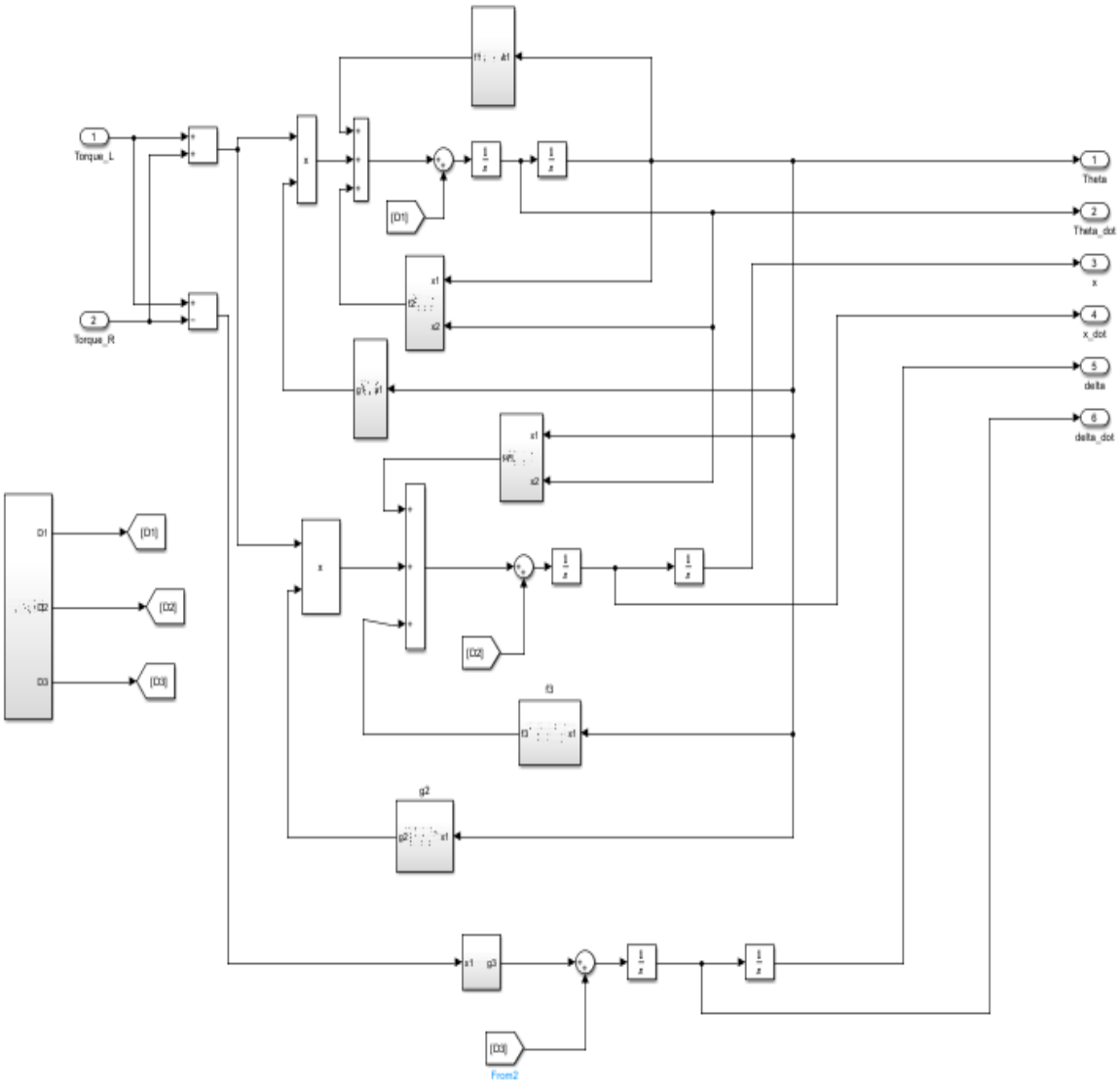


Figure 6.1: Simulink model of segway with ground Disterbance.

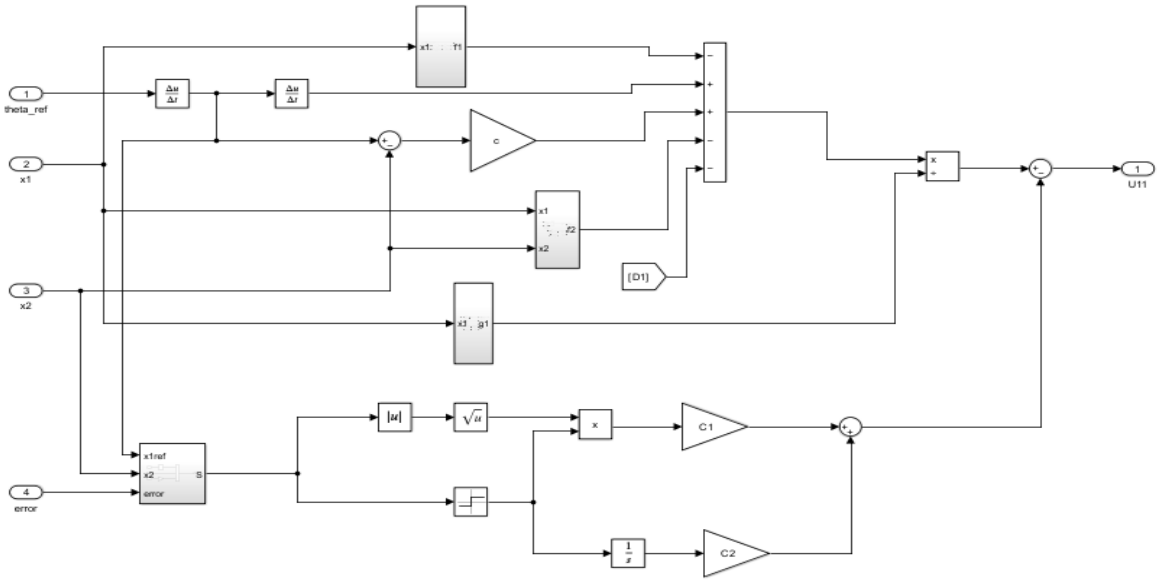


Figure 6.2: STSMC for Balance angle control.

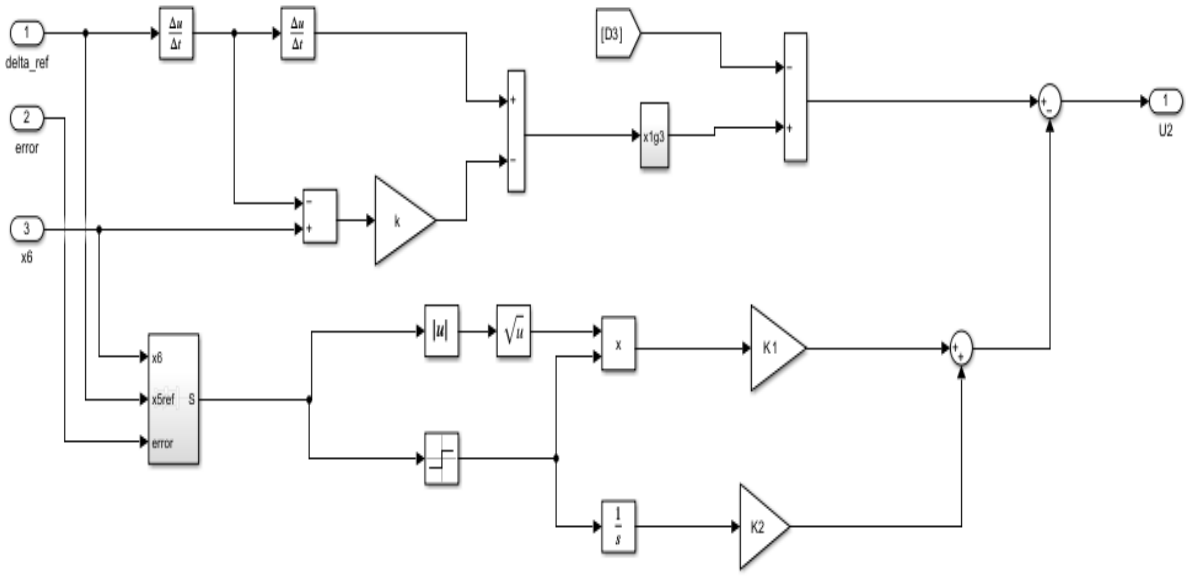


Figure 6.3: STSMC for Dirction angle control.

Appendix B:Some matlab codes

MATLAB Function: Tune_simulink_model_with_GWO

```
function Tune_simulink_model_with_GWO
% Parameters
numWolves = 25;                % Number of wolves
maxIterations = 20;            % Maximum number of
   iterations
dim = 3;                       % Problem dimensions (c, C1,
    C2)
lb = [0.05, 0.05, 0.05];       % Lower bounds
ub = [15, 15, 15];             % Upper bounds

% Initialize wolves
wolves = rand(numWolves, dim) .* (ub - lb) + lb;  % Random
    initialization within bounds
fitness = zeros(numWolves, 1);                    % Fitness
    values of wolves
bestFitnessHistory = zeros(maxIterations, 1);     % Track
   best fitness per iteration

% Evaluate fitness of initial wolves
for i = 1:numWolves
fitness(i) = EvalObj(wolves(i, :));          % Evaluate
   objective function
end

% Identify Alpha, Beta, Delta wolves
[~, idx] = sort(fitness);
AlphaPos = wolves(idx(1), :); Alpha = fitness(idx(1));
BetaPos = wolves(idx(2), :); Beta = fitness(idx(2));
DeltaPos = wolves(idx(3), :); Delta = fitness(idx(3));

% GWO iterations
for iter = 1:maxIterations
a = 2 - iter * (2 / maxIterations);          % Linearly
   decreasing a

for i = 1:numWolves
% Update each wolf's position
for d = 1:dim
% Influence of Alpha
r1 = rand(); r2 = rand();
A1 = 2 * a * r1 - a;
C1 = 2 * r2;
D_alpha = abs(C1 * AlphaPos(d) - wolves(i, d));
X1 = AlphaPos(d) - A1 * D_alpha;

% Influence of Beta
r1 = rand(); r2 = rand();
```

```
        A2 = 2 * a * r1 - a;
        C2 = 2 * r2;
        D_beta = abs(C2 * BetaPos(d) - wolves(i, d));
        X2 = BetaPos(d) - A2 * D_beta;

        % Influence of Delta
        r1 = rand(); r2 = rand();
        A3 = 2 * a * r1 - a;
        C3 = 2 * r2;
        D_delta = abs(C3 * DeltaPos(d) - wolves(i, d));
        X3 = DeltaPos(d) - A3 * D_delta;

        % Update wolf position
        wolves(i, d) = (X1 + X2 + X3) / 3;
        end

        % Boundary check
        wolves(i, :) = max(min(wolves(i, :), ub), lb);

        % Evaluate fitness
        fitness(i) = EvalObj(wolves(i, :));
        end

        % Update Alpha, Beta, Delta
        [~, idx] = sort(fitness);
        AlphaPos = wolves(idx(1), :); Alpha = fitness(idx(1));
        BetaPos = wolves(idx(2), :); Beta = fitness(idx(2));
        DeltaPos = wolves(idx(3), :); Delta = fitness(idx(3));

        % Store best fitness for plotting
        bestFitnessHistory(iter) = Alpha;

        % Display iteration results
        fprintf('Iteration %d: Best Fitness (ITAE) = %.6f\n', iter
            , Alpha);
        end

        % Plot the convergence curve
        figure;
        plot(1:maxIterations, bestFitnessHistory, 'LineWidth', 2);
        xlabel('Iteration');
        ylabel('Best Fitness (ITAE)');
        title('GWO Convergence Curve');
        grid on;

        % Display best solution
        disp('Best Parameters:');
        disp(AlphaPos);
        disp('Best Fitness Value (ITAE):');
        disp(Alpha);
        end
```

```
91
92          function obj = EvalObj(params)
93          % Updates the values of c, C1, C2 in Simulink model
               workspace
94          simulink_model = 'simulink_model_of_segway';
95          load_system(simulink_model);
96          gains = get_param(simulink_model, 'modelworkspace');
97
98          % Assign parameters to the Simulink model
99          gains.assignin('a', params(1));
100         gains.assignin('A1', params(2));
101         gains.assignin('A2', params(3));
102         gains.assignin('c', params(4));
103         gains.assignin('C1', params(5));
104         gains.assignin('C2', params(6));
105
106         % Simulate the Simulink model to get the outputs
107         simOut = sim(simulink_model, 'SaveOutput', 'on');
108
109         % Retrieve time vector and error signals
110         t = simOut.find('t');
111         e1 = simOut.find('e1');
112         e2 = simOut.find('e2');
113
114         % Calculate the performance index using the trapezoidal
               integration rule
115         n = numel(t);
116         ae = abs(e1) + abs(e2); % IAE
117         obj = 0;
118         for i = 2:n
119         stepsize = t(i) - t(i - 1);
120         obj = obj + 0.5 * t(i) * (ae(i - 1) + ae(i)) * stepsize; %
               ITAE
121         end
122         end
```

MATLAB Function: Tune_simulink_model_with_PSO

```
1           function Tune_simulink_model_of_segway1
2           tic % start timer
3           rng default
4           options = optimoptions('particleswarm', 'HybridFcn',
               @fmincon);
5           options.PlotFcn = 'pswplotbestf';
6           options.SwarmSize = 20;
7           options.MaxIterations = 10;
8           nvar = 6;
9           LB = [0.05 0.05 0.05]; % Lower bound of tuning parameters
10          UB = [15 15 15]; % Upper bound of tuning parameters
```

```
11          [x, fval] = particleswarm(@EvalObj, nvar, LB, UB, options)
                ;
12          end
13
14          function obj = EvalObj(para)
15          % Updates the values of C1, C2, c in Simulink model
                workspace
16          simulink_model = 'simulink_model_of_segway';
17          load_system(simulink_model);
18          gains = get_param(simulink_model, 'modelworkspace');
19          gains.assignin('c', para(1));
20          gains.assignin('C1', para(2));
21          gains.assignin('C2', para(3));
22          gains.assignin('a', para(4));
23          gains.assignin('A1', para(5));
24          gains.assignin('A2', para(6));
25
26          % Simulates the Simulink model to get the outputs
27          simOut = sim(simulink_model, 'SaveOutput', 'on');
28
29          % Retrieves time t and error e
30          t = simOut.find('t');
31          e1 = simOut.find('e1');
32          e2 = simOut.find('e2');
33
34          % Calculates a performance index using the trapezoidal
                integration rule
35          n = numel(t);
36          ae = abs(e1)+abs(e2); % IAE
37          obj = 0;
38          for i = 2:n
39          stepsize = t(i) - t(i-1);
40          obj = obj + 0.5 * t(i) * (ae(i-1) + ae(i)) * stepsize; %
                ITAE
41          end
42          end
43          toc % end timer
```