REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DE MEDEA



FACULTE DE TECHNOLOGIE

THESE DE DOCTORAT

Présentée en vue de l'obtention du diplôme de

Docteur en : Automatique

Spécialité : Automatique et Systèmes

Présentée par : GUENDOUZ Asmaa

Thème

Contribution à l'ordonnancement et au contrôle d'exécution d'un Véhicule Autonome

Présentée devant le jury composé de :

Nom & Prénom	**Grade**	**Institution d'affiliation**	**Désignation**
Fethia HAMIDIA	Professeur	Université de Médéa	Présidente
Mustapha HATTI	Directeur de recherche	UDES, Tipaza	Directeur de thèse
Abdelhalim TLEMCANI	Professeur	Université de Médéa	Co- Directeur de thèse
Noureddine HENINI	Professeur	Université de Médéa	Examinateur
Amel ABBADI	Professeur	Université de Médéa	Examinatrice
Katia KOUZI	Professeur	Université de Laghouat	Examinatrice

Année académique : 2025/2026

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY YAHIA FARES OF MEDEA



FACULTY OF TECHNOLOGY

THESIS

To obtain the degree of

DOCTOR in: Automatic Control

Specialty: Automatic Control and Systems

Presented by: GUENDOUZ Asmaa

Thesis title

Contribution to the Scheduling and Task Execution Control of an Autonomous Vehicle

Defended in front of a jury composed of:

Last and first name	Grade	Institution of affiliation	Designation
Fethia HAMIDIA	Professor	University of Medea	President
HATTI Mustapha	Research Director	UDES, Tipaza	Supervisor
TLEMCANI Abdelhalim	Professor	University of Medea	Co-supervisor
Noureddine HENINI	Professor	University of Medea	Examiner
Amel ABBADI	Professor	University of Medea	Examiner
Katia KOUZI	Professor	University of Laghouat	Examiner

Academic year : 2025/2026

كلمة شكر وامتنان

أحمد الله تعالى وأشكره على نعمه التي لا تُعد ولا تُحصى، وعلى أن وفّقني وهداني ويسر لي سُبل العلم، ومنّ عليّ بالصبر والعزيمة حتى بلغت هذه المرحلة من مساري الأكاديمي.

أتقدّم بجزيل الشكر والعرفان إلى **جامعة يحيى فارس بالمدية** على ما وفرته من بيئة علمية ملائمة ودعم متواصل مكّنني من إنجاز هذا العمل. كما أتوجه بخالص التقدير والامتنان إلى مشرفيّ الكريمين: **الأستاذ حتّي** و**الأستاذ تلمساني**، لما قدّماه من توجيهات سديدة، ومتابعة دقيقة، ودعم بناء طيلة مراحل هذا البحث.

كما أرفع أسمى آيات الشكر والتقدير إلى أعضاء لجنة المناقشة الموقّرين، الذين أتشرف بعرض هذا العمل أمامهم، وأثمّن عالياً وقتهم الكريم وملاحظاتهم المستقبلية.

وبقلوب يغمرها الامتنان، أخصّ بالشكر والاعتراف بالجميل **والديّ العزيزين**، تاج رأسي ونبراس دربي، اللذين لولاهما بعد الله لما كنت حيث أنا اليوم. دعاؤهما، تضحياتهما، صبرهما، وإيمانهما بي، كانت الزاد الروحي والدافع الأكبر لي في هذا الطريق. فلهما مني كل الامتنان والتقدير مدى الحياة.

ولا يفوتني أن أتقدّم بالشكر إلى **والديّ زوجي**، وسائر أفراد العائلة الممتدة، على دعمهم وتشجيعهم المتواصل.

ثم أخص بالشكر **زوجي**، الذي كان سنديًا وشارك معي هذه الرحلة بكل صبر وتفهم.

كما أخص بالشكر **أختي وأخوتي الأعزاء**، ولكل فرد من عائلاتهم، على دعمهم المعنوي المتواصل وتشجيعهم الدائم.

وأعرب عن بالغ امتناني للدكتورة **هند سعيداني سكوت** على دعمها السخي، وتوجيهاتها القيمة التي كان لها أثر كبير في مسيرتي العلمية.

كما أتقدّم بالشكر إلى **الدكتور كمال مسعودان**، على وقته الذي خصّصه لي.

وأشكر كذلك **الدكتور زكرياء توكال** على مبادرته.

وأثمّن تجربتي التدريسية في **المدرسة العليا للأساتذة بالقبة**، التي منحتني خبرة مهنية مثرية.

ولا أنسى أن أعبّر عن شكري العميق لأستاذي السابق الدكتور **عبد الرحمن بوهنا**، الذي أشرف عليّ في مرحلتي الهندسة والماستر بالمدرسة الوطنية المتعددة التقنيات بوهران ، وكان له الفضل في غرس مبادئ البحث العلمي في مساري الأكاديمي.

وفي الختام، أتوجّه بالشكر الصادق لكل من قدّم لي يد العون، من قريب أو من بعيد، طيلة هذا المشوار العلمي. فلكل من ساعدني بكلمة، أو دعاء، أو تشجيع، أو معلومة، أرفع أسمى عبارات التقدير والامتنان من أعماق القلب.

إهداء

إلى أبي... قد كنتَ حلمي الأول، وقدوتي التي اهتديتُ بها في طريقي، كنتُ أطمح أن أبلغ هذه الدرجة العلمية لتفخر بي كما كنتَ أفخر بك، لكنّ إرادة الله سبقت، فرحلتَ قبل أن ترى ثمرة هذا الجهد. رحمك الله يا أبي، فقد كنتَ النور الذي أضاء دربي، ولا يزال أثرك حيًّا في كل خطوة أخطوها.

اللهم ارحمه رحمةً واسعة، واغفر له مغفرةً تامّة، اللهم اجعل قبره روضةً من رياض الجنة، ونوّر له فيه، ووسّع مدخله، وآنِس وحشته، واجعل عمله الصالح شفيعًا له.

اللهم ارفع درجته في عليين، واجمعني به في الفردوس الأعلى من الجنة، واجعل هذا العمل صدقةً جاريةً له، ونورًا يصل إليه في قبره، ولا تحرمني أجر برّه بعد موته، واغفر لنا وله يا رب العالمين.

ملخص

يمثل هذا العمل مساهمةً في تطوير استراتيجيات التحكم الخاصة بالمركبات الكهربائية الذاتية، من خلال دمج بنى تحكم كلاسيكية وتقنيات تحسين ذكية مستندة إلى الخوارزميات الاسترشادية، إلى جانب آليات تحكم تكيفي.
تم إجراء دراسة مقارنة لمسارات السرعة، عبر صياغة وتقييم مسارات من نوع منحنى ومسار جيبي، بهدف الاستجابة لمتطلبات السلامة المرورية وضمان راحة الركاب. وقد تم اعتماد المتحكم التناسبي-التكاملي-التفاضلي الكلاسيكي لتتبع السرعة، قبل تحسين أدائه عن طريق ضبط غير مباشر باستخدام خوارزميتين استرشاديتين مستوحاتين من الطبيعة، وهما الخوارزمية الجينية وخوارزمية بحث قنديل البحر، حيث أظهرت خوارزمية بحث قنديل البحر فعالية أعلى. وللتغلب على محدوديات أنظمة التحكم الثابتة في البيئات الديناميكية، تم تطوير خوارزمية تحكم تكيفي وفق نموذج مرجعي، استناداً إلى نظرية الاستقرار لليابونوف ومعيار الاستقرار الفائق لبوبوف. وقد تم تقييم هذه المقاربة تحت عدة سيناريوهات، حيث أظهرت مقاربة ليابونوف مستوى عالياً من المتانة والدقة والقدرة على التكيف.
تمت المصادقة على الاستراتيجيات المقترحة من خلال محاكاة موسعة باستخدام بيئة ماتلاب\سيمولينك، مما أكد فعالية التكامل بين التحكم الكلاسيكي، والتحسين الذكي، والتحكم التكيفي. وقد أبرزت النتائج أداء كل مقاربة، من المتحكم التناسبي-التكاملي-التفاضلي الكلاسيكي، والمتحكم المحسن عبر الخوارزميات الاسترشادية، إلى التحكم التكيفي عبر سيناريوهات متعددة. كما مكّنت التحاليل المقارنة من تحديد دقيق للظروف التي يتفوق فيها كل متحكم .وتُعد هذه النتائج مرجعاً مهماً لتوجيه اختيار استراتيجيات التحكم المناسبة في أنظمة المركبات الكهربائية الذاتية، مما يساهم في تطوير تقنيات القيادة الذكية والتكيفية.

الكلمات المفتاحية: المركبات الكهربائية الذاتية القيادة، التحكم التكيفي المعتمد على لياپونوف، خوارزمية قنديل البحر، الخوارزمية الجينية، معيار الاستقرار الفائق لبوبوف، أنظمة التحكم الذكية.

Abstract

This dissertation offers a modest yet meaningful contribution toward advancing control strategies for Electric Autonomous Vehicles (EAVs), through a comprehensive framework that integrates classical control structures, intelligent optimization, and adaptive control mechanisms.
Initially, a comparative study of longitudinal speed profiles is conducted, formulating and evaluating S-curve and Sinusoidal trajectories to meet road safety and passenger comfort standards. The classical Proportional Integral Derivative (PID) controller is implemented for longitudinal speed tracking, with its performance subsequently enhanced through offline tuning via two nature-inspired Metaheuristic algorithms, Genetic Algorithm (GA) and the Jellyfish Search (JS) optimizer. Results demonstrated the superior optimization capabilities of Metaheuristics, with the JS optimizer exhibiting higher efficiency.
To address the limitations of fixed-gain controllers in dynamic environments, Model Reference Adaptive Control (MRAC) strategies based on Lyapunov stability theory and Popov's Hyperstability criterion are developed and evaluated under scenarios involving parametric uncertainties and external disturbances. The Lyapunov-based MRAC consistently delivered superior robustness.
Extensive simulations within a MATLAB/Simulink environment were conducted to validate the proposed control strategies. The results highlighted the performance of each approach, classical PID control, optimized PID control via Metaheuristic algorithms, and adaptive control across different test scenarios. The comparative analysis clearly identified the conditions under which each controller excelled. These findings provide valuable insights for the deployment of suitable control methods in Electric Autonomous Vehicle systems, contributing to the advancement of adaptive and intelligent driving technologies.

Key words: *Electric Autonomous Vehicles (EAVs), Lyapunov-based Adaptive Control, Jellyfish Search Algorithm, Genetic Algorithm, Popov's Hyperstability criterion, Speed Profiling, Intelligent Control Systems.*

Résumé

Ce travail constitue une contribution à la fois modeste et significative à l'avancement des stratégies de commande des Véhicules Autonomes Électriques, intègrant des structures de commande classiques, des techniques d'optimisation intelligentes basées sur les algorithmes métaheuristiques, ainsi que des mécanismes de commande adaptative.
Une étude comparative des profils de vitesse est d'abord réalisée, à travers la formulation et l'évaluation de trajectoires de type S-curve et sinusoïdale, dans l'objectif de satisfaire aux exigences de sécurité routière tout en assurant le confort des passagers. La commande classique PID est mise en œuvre pour le suivi de vitesse, puis améliorée via un réglage hors ligne fondé sur deux algorithmes métaheuristiques inspirés de la nature, Algorithme Génétique (GA) et le Jellyfish Search (JS). Les résultats ont mis en évidence la supériorité des approches métaheuristiques. Cependant, le JS a démontré une efficacité supérieure.
Afin de pallier les limitations des contrôleurs à gains fixes en environnement dynamique, une commande adaptative à modèle de référence (MRAC), basée sur la théorie de la stabilité de Lyapunov et le critère d'Hyperstabilité de Popov, a été développée et évaluée sous divers scénarios, incluant des incertitudes paramétriques et des perturbations externes. La commande basée sur Lyapunov a démontré une robustesse, une précision et une adaptabilité systématiquement supérieures.
Des simulations approfondies réalisées sous MATLAB/Simulink ont validé les approches proposées, confirmant les avantages d'une synergie entre commande classique, optimisation intelligente et commande adaptative. Les résultats ont mis en évidence les performances de chaque approche, le PID classique, le PID optimisé via les algorithmes métaheuristiques et la commande adaptative dans différents scénarios. L'analyse comparative a clairement identifié les conditions dans lesquelles chaque contrôleur a excellé. Assurant ainsi, des recommandations stratégiques pour l'implémentation de méthodes de commande appropriées dans les systèmes de véhicules électriques autonomes, contribuant ainsi à l'avancement des technologies de conduite adaptative et intelligente.

Mots-clés : *Véhicules autonomes, Commande Adaptative basée sur Lyapunov, Algorithme Jellyfish Search, Algorithme Génétique, Critère d'Hyperstabilité de Popov, Génération de profil de vitesse, Commandes intelligentes.*

Contents

List of Figures xi

List of Tables xiii

List of Abbreviations xiv

Notations xv

Scientific Publications and Communications xvii

General Introduction 1

0.1 Problem Statement . . . 1
0.2 Main Contributions . . . 3
0.3 Thesis Outline . . . 3

1 Literature Review 5

1.1 Introduction . . . 6
1.2 Electric Autonomous Vehicles History and Trends . . . 6
1.3 Control Systems and Trajectory Planning in EAVs . . . 9
1.4 Optimization in Control Systems of EAVs . . . 10
1.5 Robust Control Strategies . . . 11
1.6 Identified Gap in the Literature . . . 11
1.7 Justification of the Present Work . . . 12
1.8 Conclusion . . . 13

2 Autonomous Electric Vehicles Modelling 14

2.1 Introduction . . . 15
2.2 Electric Vehicles . . . 15
2.2.1 Powertrain Configuration . . . 17

2.3 Autonomous Vehicles . . . 20
2.4 Vehicle Modelling . . . 21
2.4.1 Kinematic Bicycle Model . . . 22
2.4.2 Dynamic Bicycle Model . . . 25
2.4.3 Implementation of the Kinematic Bicycle Model . . . 28
2.5 Conclusion . . . 29

3 Trajectory Generation and Controller Design 30
3.1 Introduction . . . 31
3.2 S–Curve Profile . . . 31
3.3 Sinusoidal Profile . . . 36
3.4 Controller Design . . . 40
3.5 Lateral Control . . . 40
3.6 Longitudinal Control . . . 42
3.7 Conclusion . . . 44

4 Metaheuristics in Control and Parameter Optimization 45
4.1 Introduction . . . 46
4.2 Metaheuristic Algorithms . . . 46
4.2.1 Classification of Metaheuristics . . . 47
4.2.2 Genetic Algorithms . . . 51
4.2.3 Jellyfish Search Optimizer . . . 53
4.2.4 Objective Function Design and Optimization Criteria . . . 57
4.3 Multi-Objective Optimization in PID Tuning through Metaheuristics . . . 59
4.3.1 Genetic Algorithm-Based PID Tuning . . . 60
4.3.2 Jellyfish Algorithm-Based PID Tuning . . . 63
4.3.3 Results and Discussion . . . 66
4.4 Conclusion . . . 73

5 Adaptive Control Theory and Implementation 74
5.1 Introduction . . . 75
5.2 Adaptive Control . . . 75
5.2.1 Gain Scheduling . . . 76
5.2.2 Self-Tuning Controllers . . . 76
5.2.3 Model Reference Adaptive Control . . . 77
5.3 Synthesis Methods for MRAC Algorithms . . . 78

5.3.1 The Gradient Method . . . 79
5.3.2 Lyapunov's Direct Method . . . 80
5.3.3 Popov's Hyperstability Method . . . 84
5.4 Comparative Evaluation of Adaptive Speed Control Strategies . . . 88
5.4.1 Hyperstability Criterion on Longitudinal Adaptive Control . . . 88
5.4.2 Lyapunov-based Longitudinal Adaptive Control . . . 91
5.4.3 Results and Discussion . . . 94
5.5 Conclusion . . . 97

6 Performance Summary and Insights **98**
6.1 Introduction . . . 99
6.2 System Configuration . . . 99
6.3 Test Scenarios and Results . . . 100
6.4 Conclusion . . . 109

General Conclusion **110**

Bibliography **112**

A S-function

B Jellyfish Algorithm

List of Figures

1.1 City traffic in the 1890s. . . . 7
1.2 Timeline illustrating the evolution of electric autonomous vehicles. . . . 8

2.1 Powertrain configurations . . . 18
2.2 Bicycle model representation . . . 22
2.3 Body fixed reference frame . . . 23
2.4 Kinematic bicycle model . . . 24
2.5 Simulink block diagram for the kinematic bicycle model. . . . 28

3.1 S-curve profiles . . . 35
3.2 Graphical result of the Sinusoidal and S-curve profiles comparison . . . 39
3.3 Steering angle control block diagram . . . 41
3.4 Steering angle simulation result . . . 41
3.5 PID control system . . . 43
3.6 Simulation result for the speed control system . . . 43

4.1 Historical timeline of metaheuristic algorithms development . . . 49
4.2 Classification of nature-inspired metaheuristics . . . 51
4.3 General flowchart illustrating the main steps of a genetic algorithm. . . . 53
4.4 Conceptual illustration of jellyfish behaviour. . . . 54
4.5 Flowchart of the Jellyfish Search optimizer mechanism. . . . 56
4.6 Flowchart of the GA-based PID tuning process. . . . 61
4.7 Flowchart of the JS algorithm-based PID tuning process. . . . 65
4.8 Optimized PID longitudinal control. . . . 67
4.9 Optimized sinusoidal speed profile. . . . 68
4.10 Genetic Algorithm optimization results through MOO. . . . 69
4.11 Speed output response . . . 72
4.12 Error response . . . 73

5.1 Adaptive Control block diagram. 76
5.2 MRAC block diagram. 78
5.3 Hyperstability-based MRAC block diagram 90
5.4 Hyperstability-based MRAC angular velocity results 91
5.5 Lyapunov-based MRAC block diagram 93
5.6 Lyapunov-based MRAC angular speed results 93
5.7 Overall control system with hyperstability-based adaptive controller 94
5.8 Overall control system with Lyapunov-based adaptive controller 95
5.9 Speed tracking performance comparison. 96

6.1 Position tracking along the circular trajectory 101
6.2 Yaw angle evolution 101
6.3 Steering angle tracking 102
6.4 Speed tracking performance before and after resistance variation 104
6.5 Speed tracking error under parameter variation 104
6.6 Evolution of internal parameters $c_0(t)$ and $k_a(t)$. 105
6.7 Adaptation gains and the control input $u(t)$ evolution. 105
6.8 Speed response of the system under a 10% load disturbance. 107
6.9 Speed error response. 107
6.10 Evolution of the adaptation gains and the control input $u(t)$. 108
6.11 Evolution of internal parameters $c_0(t)$ and $k_a(t)$. 108

List of Tables

3.1 Parameters initialization for the S-curve profile 31
3.2 ISO 2631-1 standard comfort caution ranges 33
3.3 Comparison of parameters for different values of Γ 34
3.4 Parameters initialization for the sinusoidal profile 36
3.5 DC Motor parameters 42

4.1 Genetic Algorithm optimization results using ISE 62
4.2 GA parameters 67
4.3 JS parameters 67
4.4 Genetic Algorithm optimization results through MOO. 69
4.5 Comparative study of PID gains 71
4.6 Performance criteria comparison 72

5.1 Linear gain parameters and their expressions. 89
5.2 Hyperstability-based MRAC parameters 90
5.3 Identified parameters 92
5.4 Lyapunov-based MRAC parameters 92
5.5 Quantitative comparison of control strategies. 96

6.1 Vehicle parameters 100

List of Abbreviations

EAVs:	Electric Autonomous Vehicles
AV:	Autonomous Vehicle
EV:	Electric Vehicle
MRAC:	Model Reference Adaptive Control
BLDC:	Brushless Direct Current Motor
PMSM:	Permanent Magnet Synchronous Motor
IM:	Induction Motor
SRM:	Switched Reluctance Motor
IAFPM:	Ironless Axial Flux Permanent Magnet Motor
AWD:	All-Wheel Drive
IMU:	Inertial Measurement Unit
GNSS:	Global Navigation Satellite System
LiDAR:	Light Detection and Ranging
RADAR:	Radio Detection and Ranging
PI:	Proportional-Integral
PID:	Proportional-Integral-Derivative
EMF:	Electromotive Force
DC:	Direct Current
GA	Genetic Algorithm
GP	Genetic Programming
ES	Evolutionary Strategies
PSO	Particle Swarm Optimization
ACO	Ant Colony Optimization
JSO	Jellyfish Search Optimizer
EAs	Evolutionary Algorithms

Notations

Symbol	Description	Unit
x, y	Vehicle position in global coordinate frame	m
θ	Yaw angle (vehicle orientation)	rad
δ	Steering angle (front wheel)	rad
L	Wheelbase (distance between front and rear axles)	m
$\dot{x}$	Time derivative of x (velocity along x)	m/s
$\dot{y}$	Time derivative of y (velocity along y)	m/s
$\dot{\theta}$	Yaw rate (angular velocity of θ)	rad/s
F_x	Longitudinal force	N
F_y	Lateral force	N
m	Vehicle mass	kg
R	Tire radius	m
V_m	Maximum speed	m/s
A_m	Maximum acceleration	m/s^2
D	Total travelled distance	m
τ	Time constant for motion profile transitions	s
J_m	Maximum jerk value (limit)	m/s^3
Γ	S-curve shaping parameter (smoothness factor)	–
ω	Angular frequency (for sinusoidal profile)	rad/s
R_a	Armature resistance	Ω
i_a	Armature current	A
L_a	Armature inductance	H
k	Back EMF constant (motor constant)	$V \cdot s$
Ω	Angular velocity of motor shaft	rad/s
T_{em}	Electromagnetic torque produced by the motor	Nm
k_{em}	Torque constant	Nm/A

Symbol	Description	Unit
J	Moment of inertia of the motor (and load)	$\mathrm{kg} \cdot \mathrm{m}^2$
f	Coefficient of viscous friction	$\mathrm{Nm} \cdot \mathrm{s}$
T_L	Load torque (external resisting torque)	Nm
K_P	Proportional gain in PID controller	–
K_I	Integral gain in PID controller	–
K_D	Derivative gain in PID controller	–
y, y_m	System output and reference model output	–
$a_i(t), b_j$	Time-varying and constant system coefficients	–
$K_s(t), K_m$	Time-varying and constant system gains	–
$e(t)$	Output tracking error	–
$\gamma_i(t)$	Adjustable parameters in Lyapunov control law	–
$k_s(t), k_a(t), k_0(t)$	Control gain factors	–
$c_i(t)$	Time-varying adjustment coefficients	–
x_i	State variables in error dynamics	–
A, B, C	System, input, and output matrices	–
$u(t)$	Control input signal	–
P, Q	Positive definite symmetric matrices	–
V	Lyapunov candidate function	–
λ_i	Positive weighting constants	–
A_m, B_m, C_m	Reference model matrices	–
K_e, K_p, K_u	Linear adaptation gains	–
A_e	Error dynamics matrix	–
K_x, K_w, K_θ	Nonlinear adaptation gains	–
M, G, Q, S, T, U	Positive adaptation matrices	–
L, F, P	Semi-positive definite matrices	–
k_1, k_2	Positive and semi-positive adaptation constants	–
$f(Z)$	Objective function	–
n_{pop}	Number of jellyfish in the population	–
I_{max}	Maximum number of iterations	–
Z_i	Position of the i^{th} jellyfish in search space	–
$c(t)$	Time control function	–
Z^*	Position of the jellyfish with highest food quantity (best solution)	–

Scientific Publications and Communications

Journal Publications

1. **Longitudinal Control of Autonomous Electric Vehicles via Genetic Algorithm Tuned PID and Lyapunov-based Adaptive Control: A Robustness Evaluation Under Dynamic Uncertainties**.
   Journal of Electrical Systems (JES), **Vol. 21**, No. 01, 2025.
   Published: 25 May 2025.
   DOI: https://doi.org/10.52783/jes.8806

2. **Optimized PID Tuning in Longitudinal Control of Electric Autonomous Vehicles: A Comparative Study of Jellyfish Search and Genetic Algorithm**.
   Journal of Engineering and Technology for Industrial Applications (ITEGAM-JETIA), **Vol. 11**, No. 53, pp. 186–192, May/June 2025.
   Published: 30 June 2025.
   DOI: https://doi.org/10.5935/jetia.v11i53.1672

Book Chapters

1. **Hyperstability Criterion in Model Reference Adaptive Control: Enhancing Induction Motor Efficiency for Autonomous Electric Vehicles**.
   In: *IoT-Enabled Energy Efficiency Assessment of Renewable Energy Systems and Microgrids in Smart Cities*, Springer, 2023.
   DOI: https://doi.org/10.1007/978-3-031-60629-8_24

Conference Communications

1. **Longitudinal Adaptive Control and Speed Profiling: Enhancing Safe, Efficient Driving for Autonomous Electric Vehicles**
   7th International Conference on Applied Engineering and Natural Sciences (ICAENS 2025)
   Date: 15–16 May 2025
   Location: Konya, Turkey

2. **Self-Tuning Adaptive Control Regulators for Electric Motors: Enhancing DC Motor Efficiency for Autonomous Vehicles**
   First National Conference on Energy and Its Applications (NCEA'2024)
   Date: 18 December 2024
   Location: Medea, Algeria

3. **Adaptive Control Models for Electric Motors: Enhancing Efficiency in Autonomous Driving through Hyperstability Criterion**
   3rd International Conference on Recent Academic Studies (ICRAS 2024)
   Date: 04 December 2024
   Location: Konya, Turkey

4. **Hyperstability Criterion in Model Reference Adaptive Control: Enhancing Induction Motor Efficiency for Autonomous Electric Vehicles**
   7th International Conference on Artificial Intelligence in Renewable Energetic Systems (ICAIRES 2023)
   Date: 26-28 November 2023
   Location: Tipasa, Algeria

General Introduction

0.1 Problem Statement

Rising global demand for safer, more efficient, and sustainable transportation is driving the rapid development of EAVs. By combining autonomous driving capabilities with electric propulsion, they strive to enhance road safety, optimise traffic flow, and reduce environmental impact, all while ensuring greater passenger comfort and operational dependability [1]. Over recent decades, autonomous vehicle research has advanced from 1980s pioneers like Carnegie Mellon University [2] to intelligent systems capable of perceiving, planning, and acting in complex environments. Despite significant advancements in perception and decision-making modules, the effectiveness of autonomous vehicles is fundamentally linked to the precision and robustness of their control systems, which play a crucial role in ensuring safe and reliable operation.

Longitudinal control, the module responsible for governing the acceleration and deceleration of a vehicle to track a desired speed or maintain safe distances, is one of the fundamental components of autonomous driving. Among the various control strategies, the Proportional-Integral-Derivative (PID) controller remains widely adopted in industrial and academic settings due to its simplicity and intuitive design. However, the tuning of its gains is a non-trivial task, especially in systems exhibiting non-linearities, parameter uncertainties, and external disturbances.

To address the limitations of manual tuning methods, Metaheuristic optimisation algorithms are introduced. Metaheuristics are high-level optimisation strategies, designed to efficiently explore large and complex search spaces in order to find near-optimal solutions to problems, where classical methods often fail or become computationally expensive. In the context of control systems tuning, they offer robust alternatives to traditional trial-and-error approaches, particularly when the search landscape is poorly understood [3]. To evaluate the effectiveness of the tuning process, appropriate performance metrics must be defined.

In this study, a multi-objective performance index is constructed by combining the Integral of Squared Error (ISE) and the Integral of Time-weighted Absolute Error (ITAE), with weighted coefficients to prioritise time-domain accuracy and transient behaviour [4]. This composite cost function serves as the optimisation target for the metaheuristic algorithm. Following this formulation, two prominent nature-inspired techniques are employed: the Jellyfish Search (JS) algorithm and the Genetic Algorithm (GA), aiming to optimise controller parameters. These methods are selected for their ability to balance exploration and exploitation, which is essential for achieving high-quality solutions.

Although such optimisation approaches significantly improve the performance of PID controllers, they inherently retain a fixed structure and remain sensitive to changes in system dynamics. To overcome this limitation, adaptive control techniques are introduced. Model Reference Adaptive Control (MRAC) scheme is selected and designed using Direct Lyapunov method, providing formal guarantees for the asymptotic stability of the closed-loop system. The MRAC framework dynamically adjusts controller parameters in real-time, making it well-suited for handling model uncertainties and external perturbations. In addition, a comparative study is conducted with an alternative MRAC formulation based on Popov's Hyperstability method [5], offering a complementary perspective on adaptive control design and further reinforcing the robustness of the proposed approaches.

Complementary to the control architecture, this work also addresses the motion planning problem by focusing on the generation and evaluation of longitudinal speed profiles, which serve as reference trajectories for the control system. Specifically, two types of profiles S-curve and sinusoidal are developed and analysed in terms of their suitability for autonomous driving. These profiles are constructed to satisfy comfort and safety constraints derived from road standards and human perception thresholds. A comparative study highlights compromises between these profiles and identifies conditions under which each is best suited, contributing to a holistic approach to longitudinal motion control.

The methodological framework developed in this thesis is validated through extensive simulations in the MATLAB/Simulink environment. The simulations demonstrate the superiority of adaptive control over classical methods in dynamic scenarios and confirm the effectiveness of metaheuristic algorithms in PID tuning. Moreover, the proposed speed profiling methods contribute to smoother and more efficient vehicle trajectories.

0.2 Main Contributions

The main contributions of this thesis to the field of longitudinal control in electric autonomous vehicles are outlined as follows.

- **Development and comparison of speed profiles**
- **Metaheuristic optimisation of PID-based longitudinal control**
- **Performance comparison between GA and JS optimization techniques**
- **Design and validation of a Model Reference Adaptive Control for EAVs**
- **Integration and evaluation of a Lyapunov and hyperstability-based MRAC approach**

In summary, this thesis presents a comprehensive and integrated approach to longitudinal control for electric autonomous vehicles. It combines **classical and adaptive control techniques**, **intelligent optimisation methods**, and **realistic speed profile generation** to address key challenges in safety, efficiency, and passenger comfort. By bridging the gap between control design and motion planning, this research offers valuable insights and practical solutions for the future development of robust and intelligent autonomous driving systems.

0.3 Thesis Outline

The remainder of this manuscript is organised as follows:

- **Chapter 1** presents a comprehensive literature review on electric autonomous vehicles, focusing on the evolution of control strategies, motion planning techniques, and the integration of optimisation algorithms in vehicle control.
- **Chapter 2** introduces the vehicle modelling approach, including the longitudinal dynamics and the kinematic bicycle model used as the foundation for control design.
- **Chapter 3** explores the design and generation of S-curve and sinusoidal speed profiles, including their mathematical formulation and comparative performance analysis.
- **Chapter 4** addresses the controller design for electric autonomous vehicles , focusing on both lateral and longitudinal control strategies.

- **Chapter 5** explores the formulation and integration of Metaheuristic optimisation algorithms to enhance PID controller performance.
- **Chapter 6** is dedicated to adaptive control strategies.
- **Chapter 7** presents a performance summary of the developed control strategies.
- **A General Conclusion**, is finally provided to summarise the key findings and contributions of the thesis and outline future research directions.

Chapter 1

Literature Review

1.1 Introduction

This chapter presents a comprehensive review of the current state of research and development in the field of motion planning for electric autonomous vehicles. It begins by outlining the historical overview to contextualize the evolution of both electric vehicles and autonomous driving technologies, along with fundamental concepts of EAVs, emphasizing the convergence of electrification and automation in modern transportation. Subsequent sections explore the key methodologies used for motion control, starting with conventional approaches such as PID and Model Predictive Control. The discussion then shifts to adaptive control strategies, particularly Model Reference Adaptive Control (MRAC), and examines how these techniques address the challenges posed by dynamic and uncertain environments. Optimization through Metaheuristic algorithms is also investigated as a means to enhance controller performance. By identifying critical limitations in current literature, such as the lack of adaptability in fixed-structure controllers and insufficient testing under variable conditions, this chapter justifies the need for research into adaptive control solutions. The final section highlights how this thesis contributes to the field, by advocating for robust adaptive speed control as an essential component in the deployment of efficient and resilient EAV systems.

1.2 Electric Autonomous Vehicles History and Trends

The idea of autonomy has deep historical roots, stemming from the Greek word autonomos, which means "self-governing". Initially, it described the independence of individuals and political entities, but over time, its meaning expanded. With advancements in technology, autonomy became linked to machines and systems that could operate without human intervention. The Industrial Revolution marked a turning point, introducing automated machinery that transformed industries. By the mid-20th century, early programmable computers pushed technological autonomy even further, laying the groundwork for the sophisticated autonomous systems in the present day [6]. The intersection of autonomy and electrification emerged prominently in the 2010s, resulting in the rise of EAVs [7].

Electric vehicles (EVs) and autonomous vehicles (AVs) are converging to shape the future of sustainable and intelligent transportation systems. While EVs address environmental and energy concerns through electrified powertrain, AVs aim to revolutionize mobility by reducing human intervention in driving tasks. The integration of these two technologies forming electric autonomous vehicles, has become a central research and development priority in both academia and industry. Although widely perceived as a modern innovation, the

electric vehicle has origins dating back to the early 19th century. Inventors such as Ányos Jedlik in Hungary and Robert Anderson in Scotland built early electric motors and primitive vehicles, demonstrating the feasibility of electricity-driven motion [8]. By the late 1800s, urban congestion and environmental issues caused by horse-drawn transport led to a search for cleaner alternatives. Electric vehicles emerged as a promising solution due to their quiet operation and ease of use compared to steam or gasoline counterparts.



Figure 1.1: City traffic in the 1890s.

The key milestones in the evolution are:

- **1830s–1850s**: Early electric motors and rudimentary vehicles were demonstrated in Europe.
- **1890s–1910s**: The commercialisation of EVs began, with the Electrobat (1894) among the first purpose-built electric cars. By 1900, EVs outsold Internal Combustion Engine (ICE) vehicles in the United States.

To better understand the progress and future outlook of EAVs, a chronological timeline is presented in Figure 1.2, summarising historical breakthroughs, commercial launches, and projected goals.

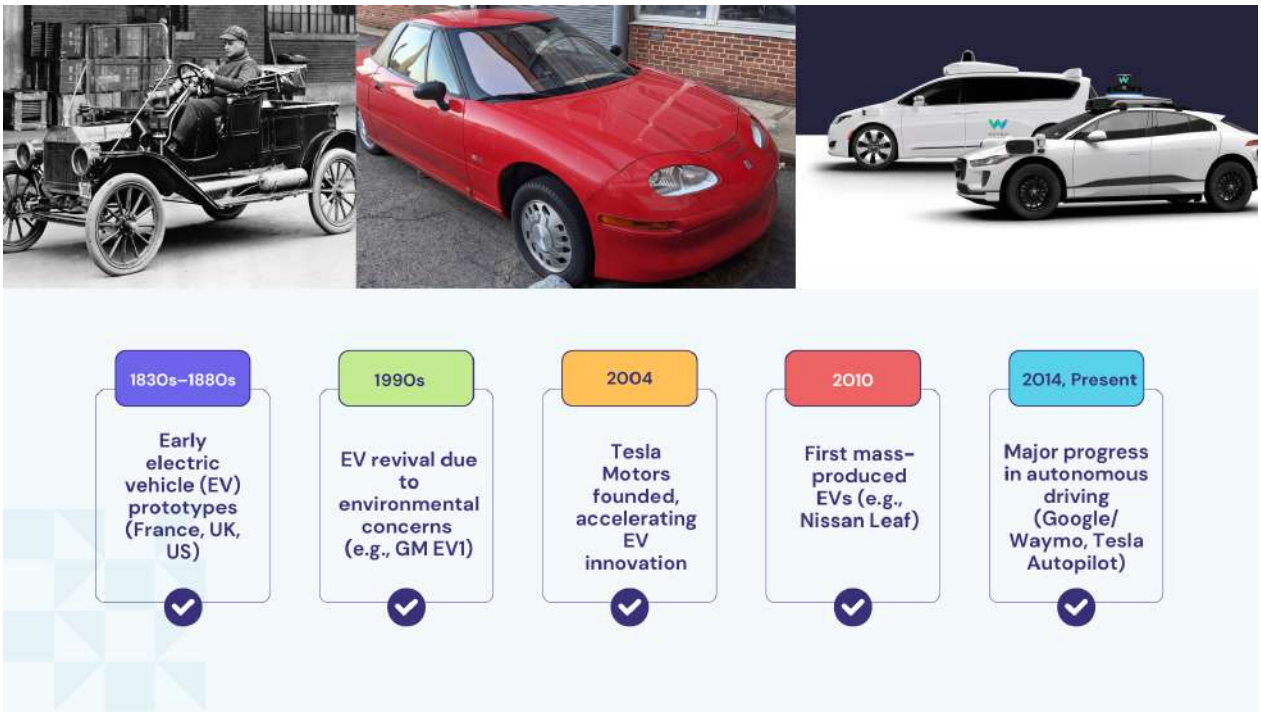


Figure 1.2: Timeline illustrating the evolution of electric autonomous vehicles.

The mass production of ICE vehicles, notably Ford's Model T, and the discovery of abundant petroleum resources led to a significant decline in EV adoption. By the 1920s, ICE vehicles dominated due to their extended range, low fuel cost, and quicker refuelling. The revival of interest in EVs began in the 1970s, driven by oil crises and increasing environmental awareness. Vehicles such as the General Motors EV1 and the Toyota Prius hybrid marked a turning point. Over the following decades, improvements in battery technology, especially lithium-ion cells, made modern EVs viable. Furthermore, autonomous vehicle technologies have made significant progress, enabled by advances in sensors, artificial intelligence, and control systems. The fusion of EV and AV technologies into electric autonomous vehicles represents a natural evolution. These vehicles rely on electric drivetrains for efficiency and sustainability, while leveraging autonomy to enhance safety, reduce traffic accidents, and increase mobility access for non-drivers [9].

Advantages of electric autonomous Vehicles

- **Environmental Impact**: Zero tailpipe emissions from EV powertrain.
- **Operational Efficiency**: Intelligent routing and smooth driving patterns reduce energy consumption.

- **Safety and Comfort**: Advanced driver assistance systems (ADAS) and full autonomy reduce human error and provide a more relaxed travel experience.
- **Integration with Smart Cities**: EAVs can be part of intelligent transportation systems, enhancing urban mobility through connectivity and automation.

The evolution of electric vehicles from early experimental prototypes to cutting-edge autonomous platforms, reflects the continuous effort to improve mobility while addressing environmental and societal challenges. Electric autonomous vehicles are not merely a technological merger, but a transformative solution for future transportation systems. As research continues, EAVs are poised to become central to smart, clean, and connected cities. In addition to electrification, autonomy is based on advanced algorithms that perceive the environment, estimate vehicle position, plan trajectories, and execute control commands. While perception and planning systems have received significant research attention, the control layer faces persistent challenges due to terrain variations, uncertain system parameters, and external disturbances.

1.3 Control Systems and Trajectory Planning in EAVs

Electric autonomous vehicles rely heavily on sophisticated control systems to ensure safe and efficient operation in dynamic traffic environments. Among these, decoupled control architectures have emerged as a robust solution to manage the complex interactions between longitudinal and lateral vehicle dynamics. In particular, the longitudinal control system is responsible for regulating speed, maintaining safe distances from other vehicles, and executing smooth acceleration or deceleration during maneuvers such as lane changes [10]. In parallel, trajectory planning remains a cornerstone in enabling EAVs to navigate safely and smoothly through various traffic scenarios. It involves generating feasible and dynamically consistent paths that comply with road geometry, environmental constraints, and the vehicle's kinematic and dynamic limits. When integrated with a decoupled control framework, trajectory planning enables the separation of lateral and longitudinal paths, facilitating independent optimization and control of each. This decomposition not only improves computational efficiency but also enhances the system's flexibility in responding to rapidly changing environments. As seen in Kim et al. [11], while a longitudinal-trajectory-free method was adopted to cope with variable traffic flow conditions, the system still relied on spatial references and predictive modelling to guide the lateral motion robustly. Such integration of adaptive trajectory planning with decoupled control mechanisms, plays a crucial role in achieving both

safety and efficiency in autonomous driving.

In conjunction with trajectory generation, speed planning is a critical component of longitudinal control in EAVs. It involves determining an optimal velocity profile along the planned trajectory, taking into account safety constraints, dynamic feasibility, and comfort. Many scholars have carried out related research in response to the above subject. Antonio et al.[12] led a comparative study of a jerk-limited, time-optimal speed planning method for arbitrary paths. Their approach ensures not only kinematic feasibility but also minimizes jerk to enhance ride comfort, which is essential for real-world deployment of EAVs in complex driving environments. In the study by Lu et al. [13], a real-time decoupling trajectory planning method is proposed for on-road autonomous driving scenarios, with particular attention given to reference speed selection as a key element in achieving smooth and safe navigation. The authors introduce a reference speed selector.

Most existing control strategies are built around PID control laws and modular (piecewise) planning frameworks, which remain popular due to their straightforward design and proven effectiveness in structured environments.

1.4 Optimization in Control Systems of EAVs

Metaheuristic algorithms such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Jellyfish Search (JS) have been widely used for automatic tuning of control parameters. These methods aim to optimize performance criteria like the Integral of Squared Error (ISE) or the Integral of Time-weighted Absolute Error (ITAE), resulting in improved tracking and energy efficiency [14]. Several studies have shown that using metaheuristics to optimize PID parameters leads to superior results compared to manual tuning. For instance, Nassef et al. [15] conducted a systematic and comprehensive review of metaheuristic-based strategies for tuning fractional-order controllers, further confirming the versatility and effectiveness of these algorithms in diverse control systems beyond traditional structures. In particular, the field of speed control and motion. A comprehensive review by Oladipo, Sun, and Wang [16], supports this observation, presenting a detailed analysis of how various metaheuristic algorithms, such as Particle Swarm Optimization (PSO), Genetic Algorithms (GA), Ant Colony Optimization (ACO), and Grey Wolf Optimizer (GWO), have been successfully employed to fine tune PID controllers for DC motor drives. In a more applied context, Naranjo and colleagues [17] focused specifically on speed control for autonomous vehicles. Their work highlights how metaheuristic optimization using methods like PSO and Differen-

tial Evolution (DE) can improve the adaptability and precision of speed regulation. Similarly, Costa et al. [18] explore the application of metaheuristic algorithms for optimizing speed control in autonomous vehicles, demonstrating notable improvements in system responsiveness and energy efficiency. Despite performance improvements, these optimized controllers remain limited by their fixed structure, reducing adaptability in dynamic real-time conditions.

1.5 Robust Control Strategies

Robust control techniques play a crucial role in ensuring accurate speed regulation for dynamic systems subject to uncertainties, external disturbances, and non-linear characteristics. Traditional methods such as H∞ control focus on minimizing the impact of worst-case disturbances, offering solid stability margins, as demonstrated in the robust speed regulation of electric vehicles using the LMI-based H∞ framework [19]. Sliding Mode Control (SMC) is another widely adopted approach, known for its resilience against matched uncertainties through enforcing state convergence to a designed sliding surface, though it often encounters chattering effects. Its effectiveness has been highlighted in autonomous vehicle path tracking applications, where improved SMC variants enhance robustness under dynamic conditions [20]. Meanwhile, Adaptive Control schemes adjust control parameters in real time to maintain system stability despite parameter variations. Researchers have proposed various MRAC formulations, from basic gradient-based approaches to more sophisticated Lyapunov-stable designs. These approaches are capable of compensating for unknown parameters and disturbances [21].

1.6 Identified Gap in the Literature

Although adaptive control techniques are well established in other industrial sectors, their application in EAV speed control, particularly under dynamically varying parameters is relatively limited. Most existing works tend to:

- Focus on fixed-structure controllers like PID or MPC without incorporating mechanisms for online adaptation ;
- Apply adaptive control in isolated settings, often without benchmarking against optimized fixed-structure counterparts ;
- Overlook scenarios involving non-constant system dynamics or external disturbances.

This creates a methodological gap where adaptive controllers are not rigorously evaluated against up-to-date, well-tuned conventional strategies, potentially underestimating their relative effectiveness under realistic, time-varying conditions. Given the inherent nonlinearity and uncertainty in EAV propulsion systems, especially when driven by electric motors with parameter variations, such gaps limit the generalizability and practical relevance of existing findings.

In addition, speed references used in many studies are often simplified. Step inputs are commonly employed for ease of analysis, while trajectory planning typically relies on piecewise functions or curve-based forms such as polynomials, splines, Bézier curves, or clothoids [22]. Although these trajectories offer smoothness and computational efficiency, they do not always capture the continuous variability present in real-world driving scenarios. As a result, the robustness of controllers under realistic, dynamically varying profiles, such as sinusoidal or S-curve speed trajectories remains insufficiently explored in the literature.

1.7 Justification of the Present Work

This research addresses the previously mentioned limitations by highlighting the advantages and drawbacks of optimized PID controllers. While, metaheuristic techniques significantly enhance performance in stable conditions, the fixed nature of these controllers limits their usefulness in environments with changing parameters. Gain scheduling, often used as an intermediary solution, adjusts controller parameters based on operating conditions but still lacks the real-time adaptability needed for highly dynamic scenarios. Therefore, this work:

- Demonstrates the strengths of Metaheuristic-based optimized PID controllers in nominal operating conditions ;
- Illustrates their limitations when vehicle dynamics vary over time ;
- Advocates for the adoption of adaptive control as a necessary solution to ensure robust performance ;
- Emphasizes the value of benchmarking controllers using dynamic reference profiles and variable conditions.

By establishing a clear distinction between parameter optimization and structural adaptability, this thesis supports the deployment of adaptive control in the longitudinal control of EAVs and paves the way for future enhancements using intelligent control techniques.

1.8 Conclusion

This chapter reviewed the evolution of electric autonomous vehicles and key control strategies, including PID, metaheuristic optimization, and adaptive control. Furthermore, this literature review has highlighted some critical gaps in the current research on motion control for electric autonomous vehicles. Motivated by these shortcomings, this thesis focuses on fulfilling these gaps.

Chapter 2

Autonomous Electric Vehicles Modelling

2.1 Introduction

This chapter provides a comprehensive overview of Electric Autonomous Vehicles (EAVs), highlighting their powertrain systems and the role of electric motors in achieving efficient and high performance mobility. As a fusion of electrification and advanced automation, EAVs represent a leading edge technology in the evolution of modern transportation. To support the development of effective control strategies for these vehicles, the chapter also outlines the key methodologies for mathematically modelling vehicle motion dynamics. Central to this discussion is the bicycle model, which is widely used in autonomous vehicle applications due to its ability to simplify complex vehicle dynamics while preserving essential characteristics for control and analysis. Both kinematic and dynamic bicycle models are presented, with detailed mathematical formulations and illustrative figures to clarify the spatial relationships and dynamic behaviour of the vehicle. These visual aids and equations facilitate a deeper understanding of the models' geometric and physical principles. Finally, the chapter demonstrates how these models can be implemented using MATLAB simulation tools, offering practical insights into the setup, execution, and interpretation of simulation results, bridging the gap between theoretical modelling and practical application.

2.2 Electric Vehicles

Electric vehicles (EVs) are powered by electric motors instead of internal combustion engines. Rather than using traditional fuels like gasoline or diesel, electricity is employed as the primary energy source to recharge the batteries in EVs, offering an effective means of reducing greenhouse gas emissions and dependence on petroleum-based transportation [23].
To fully understand how electric vehicles operate, it is essential to explore their key components:

Battery Pack: the main energy source, storing the electricity needed to power the motor driving the vehicle.

Auxiliary battery: provides power to operate the vehicle's accessories.

Charge port: It serves as the link between an external power supply and the traction battery pack.

DC/DC converter: transforms high-voltage DC power from the traction battery pack into lower-voltage DC power necessary for operating vehicle accessories and recharging the auxiliary battery.

On board charger: Converts the alternating current (AC) from the charge port into

direct current (DC) to charge the traction battery, while monitoring battery characteristics such as voltage, current, temperature, and state of charge.

Power electronics controller: a unit that manages the flow of electrical energy delivered by the traction battery, controlling the speed of the electric traction motor and the torque it produces.

Thermal system (cooling): Maintains a proper operating temperature range for the motor, power electronics and the other components.

Transmission: Transfers mechanical power from the electric traction motor to drive the vehicle's wheels.

Electric traction motor: drives the vehicle wheels using the battery pack's power, some vehicles use motor generators that perform both driving and regeneration functions.
Electric motors are the core of electric vehicles, converting electrical energy into mechanical motion to drive the vehicle. Electrical energy from the battery is converted into a rotating magnetic field, which interacts with the rotor to produce motion that ultimately drives the wheels [24].
For propulsion, various types of motors are used, each offering specific advantages and drawbacks depending on the application. The seven main used motors in EVs are Brushed DC Motor, Brushless DC Motor (BLDC), Permanent Magnet Synchronous Motor (PMSM), Induction Motor (IM), Switched Reluctance Motor (SRM), Synchronous Reluctance Motor and Axial Flux Ironless Permanent Magnet Motor [25].

Brushed DC Motor

The brushed DC motor is renowned for its simplicity and effectiveness. Once DC voltage is applied, current flows through the armature windings, creating a magnetic field. Hence, a rotating force is generated due to the interaction between the armature's magnetic field and the stator's magnetic field. In addition of its simple design, the DC motor provides maximum torque at low speeds without requiring a controller to operate or vary speed. However, its bulky structure and heat generation at the brushes leads to a drop in efficiency.

Brushless DC Motor (BLDC)

It is named brushless after the elimination of the commutator and the brushes. Therefore, the commutation is done electronically. Sensors like Hall effect sensors are used to detect the position of the rotor. The latter is then used by an electronic controller to switch the current in the motor windings, ensuring the motor's motion. This modified structure makes it maintenance-free and highly efficient despite the high cost and electronic requirements.

Permanent Magnet Synchronous Motor (PMSM)

It is a motor similar to the brushless DC, featuring permanent magnets on the rotor. It exhibits traction characteristics such as high-power density and high efficiency. Nonetheless, it comes with a high cost.

Induction Motor (IM)

The Induction motor is a popular choice for performance-oriented electric vehicles due to its long lifespan, low maintenance and cost-effectiveness. The Tesla Model S is a prime example of the high-performance capability of induction motors despite the complexity and control challenges.

Switched Reluctance Motors (SRM)

The SRM is noted for its simple and robust design, making it increasingly recognized for its potential in the EVs industry. Unlike brushed DC motors, Switched Reluctance Motors (SRMs) deliver power to the windings in the stator rather than the rotor, which simplifies the mechanical design yet adding complexity to the electrical design along with the noise issues due to torque ripple[26].

Synchronous Reluctance Motor

Characterized by a rotor without windings or permanent magnets, it relies on geometric magnetic reluctance variation. This design eliminates rotor cage losses and enhances torque, though it is less commonly used and may be more expensive [27].

Axial Flux Ironless Permanent Magnet Motor

The Ironless Axial Flux Permanent Magnet (IAFPM) motor has gained interest due to its efficiency, power density, and dynamic performance, enabled by its axial configuration and iron-free stator. However, it remains complex and costly to manufacture [28].

2.2.1 Powertrain Configuration

The powertrain is the heart of an EV, leading the conversion of electrical energy into mechanical energy to drive the vehicle. The powertrain configuration plays a crucial role in determining the performance, efficiency, and overall driving experience of an EV. Several configurations have been developed to optimize these aspects, each with its unique advantages and applications [29], see Figure 2.1.

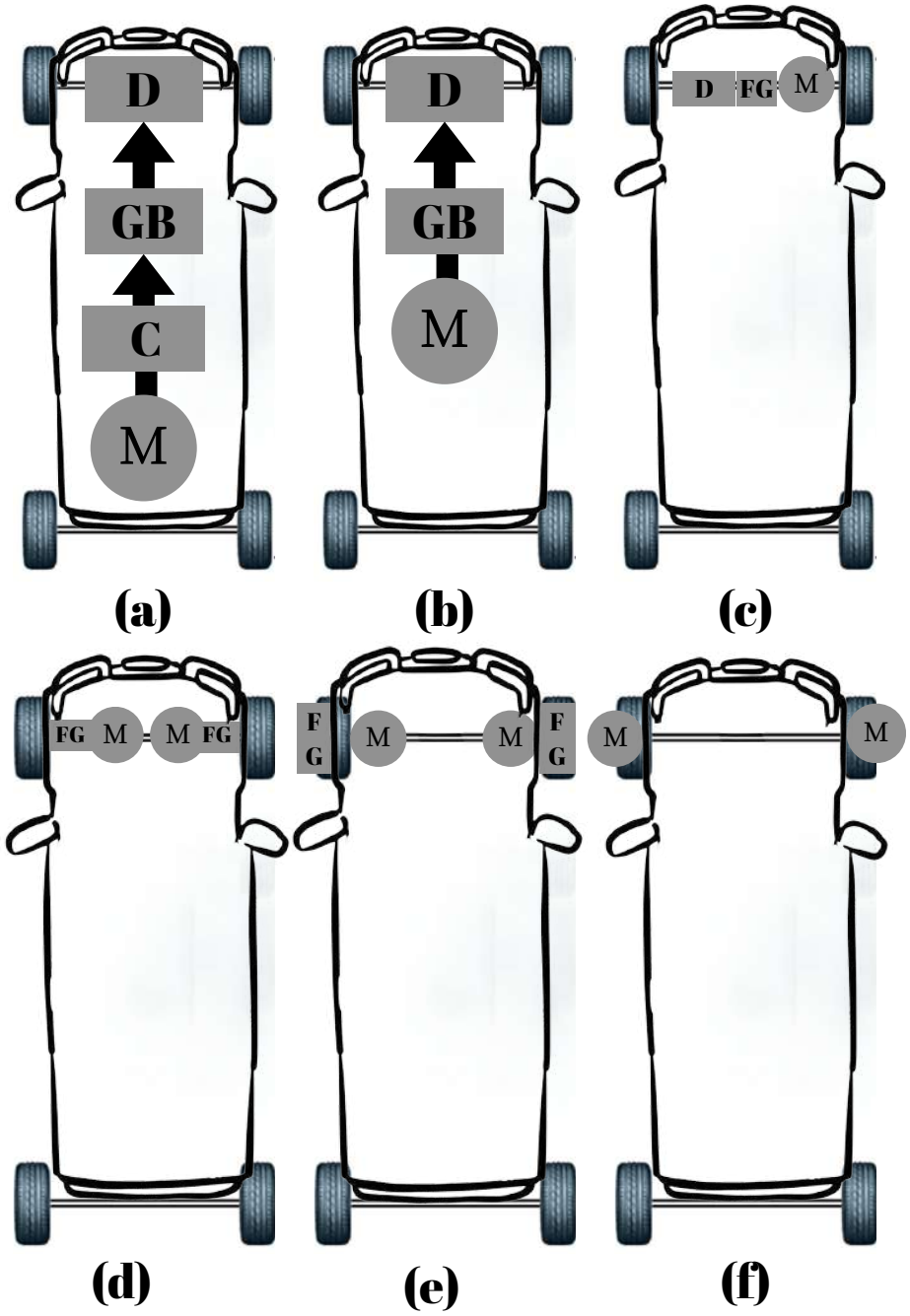


Figure 2.1: Powertrain configurations

Figure 2.1a shows the first configuration that includes an electric motor (M), a clutch(C), a gearbox (GB), and a differential(D). This latter distributes power to the wheels, allowing them to rotate at different speeds, which is essential for smooth cornering. This setup, while effective, is relatively complex and heavier due to the inclusion of the clutch and gearbox.
By removing the clutch and replacing the gearbox with a fixed gear (FG) ensuring a direct connection between the motor and the differential, a modern configuration is created in order to reduce the weight and complexity, see Figure 2.1b.
Then comes the third configuration illustrated in Figure 2.1c that integrates the electric motor, fixed gear, and differential into a single assembly, reducing the overall weight and size of the powertrain, which makes it the most preferred architecture by EV manufacturers in modern models for its optimal balance of efficiency, performance, and ease of use.
The dual-motor configuration is shown in Figure 2.1d, featuring two electric motors, each connected to a fixed gear and driving one of the vehicle's axles, which separates the control of the front and rear wheels, providing enhanced traction and stability. This configuration is particularly beneficial for all-wheel-drive (AWD) systems.
Next comes the In-Wheel Drive Configuration, where the electric motors are placed directly within the wheels, eliminating the need for a traditional drivetrain. Each wheel is equipped with its own motor and a fixed epicyclic gearing system to reduce the motor's output speed to the appropriate level. Offering high-speed reduction ratio, compact design, and improved efficiency (Figure 2.1e).
Finally, the direct drive motors configuration is given in Figure 2.1f which simplifies the mechanical setup by eliminating the gearbox and differential. The electric motors are directly connected to the wheels, providing a direct transfer of power. Enhancing reliability and efficiency. As EV technology advances, these configurations are expected to undergo further refinement, improving efficiency, performance, and the overall driving experience of electric vehicles.
In summary, electric vehicles represent a transformative shift in the automotive sector, offering a sustainable alternative to conventional internal combustion engine vehicles. With continuous advancements in battery systems, power electronics, and energy management, EVs are becoming increasingly efficient and accessible. They are essential in efforts to reduce greenhouse gas emissions and promote cleaner urban mobility. Moreover, EV technology lays a solid foundation for future innovations in autonomous driving and smart mobility solutions.

2.3 Autonomous Vehicles

Technological progress is enabling the transition from conventional, fully human-driven vehicles toward autonomous, mostly electric vehicles. Autonomous vehicles (AVs), are self-driving cars built and populated by and for humans, operating without human intervention in unpredictable, uncertain, and open traffic environments [30].
In recent years, various proposals have been made for the classification of the levels of autonomy for these vehicles. The categorization proposed by SAE International (2016) seems to be emerging as the recognized standard in international discussions. This classification outlines six levels (0 to 5) of autonomy, determined by the degree of human intervention and monitoring needed in different situations [31].

Level 0 (No Automation): The human driver is fully in control of the vehicle at all times. Even when assistance features are activated, the latter provide alerts such as automatic emergency braking, blind spot warnings, and lane departure warnings.

Level 1 (Driver Assistance): While the driver remains in control, the vehicle can assist with either steering or acceleration/deceleration.

Level 2 (Partial Automation): The vehicle can control both steering and acceleration/braking under specific conditions such as lane centring and adaptive cruise control. However, the driver must still monitor the environment and be ready to intervene.

Level 3 (Conditional Automation): The automated vehicle handles all aspects of driving under specific conditions, but the driver must be available to take over when requested.

Level 4 (High Automation): The vehicle is able to perform all driving tasks in specific environments without human intervention. However, if conditions fall outside of its operational limits, human control is required.

Level 5 (Full Automation): The vehicle is fully autonomous, being able to operate under all conditions and all environments without any human intervention.

In summary, Levels 0-2 are systems that assist the driver while being primarily driven by humans enhancing safe driving, Levels 3-4 are Self-driving systems representing increasing degrees of vehicle automation with the ability to safely transition control back to the human driver when necessary. Finally, with Level 5 being fully autonomous, the driverless phase is reached.
To allow an autonomous car to take control, it must function like a human driver, utilizing sensors often referred to as the vehicle's "eyes and ears", allowing autonomous vehicles to interpret their surroundings. Sensors are classified into two categories based on their

operational principle, Proprioceptive Sensors and Exteroceptive Sensors [32].

Proprioceptive Sensors: detect the internal state of the vehicle, such as motion, position, force, angular rate, and wheel load. Examples of these sensors include inertial measurement units (IMU), encoders, inertial sensors like gyroscopes and magnetometers, as well as location sensors such as Global Navigation Satellite System (GNSS) receivers.

Exteroceptive Sensors: external state sensors that gather information from the external environment, such as distance or object detection. These include: Cameras, LiDAR (Light Detection and Ranging), RADAR (Radio Detection and Ranging), and Ultrasonic Sensors.

Sensors are additionally classified based on their transmission range:

1. Short-range sensors (ultrasonic)
2. Medium-range sensors (radar)
3. Long-range sensors (LiDAR and high-range radar)

These sensors are central to the technological capabilities of AVs, enabling key functions such as path planning, obstacle detection, and accurate vehicle localization. They provide critical data for real-time decision-making, allowing autonomous vehicles to navigate complex road conditions and dynamic environments safely. By continuously monitoring both external surroundings and internal vehicle states, these systems ensure the safety, efficiency, and reliability of AV operations.

2.4 Vehicle Modelling

Bicycle models are the most commonly used models for trajectory generation and control laws generation for vehicle control. These models simplify the dynamics of a four-wheeled vehicle by representing it as a system with two virtual wheels. The two front wheels are combined into one at the front axle center, and the two rear wheels are lumped into one at the rear axle center, as shown in Figure 2.2. In this approach, roll (lateral inclination) and pitch (forward-backward tilting motion) are neglected to reduce computational complexity while maintaining sufficient accuracy for many applications. It is important to note that two main variants of these models are distinguished: Kinematic Bicycle Model and Dynamic Bicycle Model.

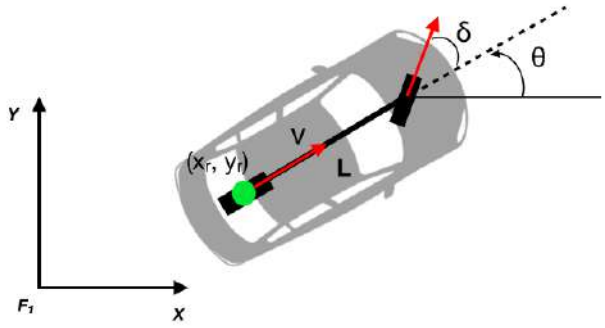


Figure 2.2: Bicycle model representation

2.4.1 Kinematic Bicycle Model

The kinematic bicycle model is a simplified model used to describe the motion of robotics and autonomous vehicles in order to perform path planning or trajectory generation.
The study of motion is a fundamental concept in mechanics, it can be divided into two primary branches: kinematics and kinetics (dynamics). These latter address distinct aspects of motion, and differentiating between them is essential for effectively analysing and understanding any dynamical system.
Kinematics presents the geometric description of motion without considering the forces or masses that cause it. Its main focus is on the positions, velocities, and accelerations of point masses or rigid bodies over time, often in relation to different reference frames or coordinate systems.
The key concepts in Kinematics are:

Position: A point or body's location in space is described using a reference frame (e.g., Cartesian, polar, or spherical coordinates).

Velocity: The rate of change of position with respect to time, represented as a vector quantity.

$$v = \frac{d\mathbf{r}}{dt},$$

where $\mathbf{r}$ is the position vector.

Acceleration: The rate of change of velocity over time.

$$a = \frac{d\mathbf{v}}{dt} = \frac{d^2\mathbf{r}}{dt^2}.$$

Figure 2.3 illustrates a rigid body RB, in motion relative to a primary reference frame taken from [33].

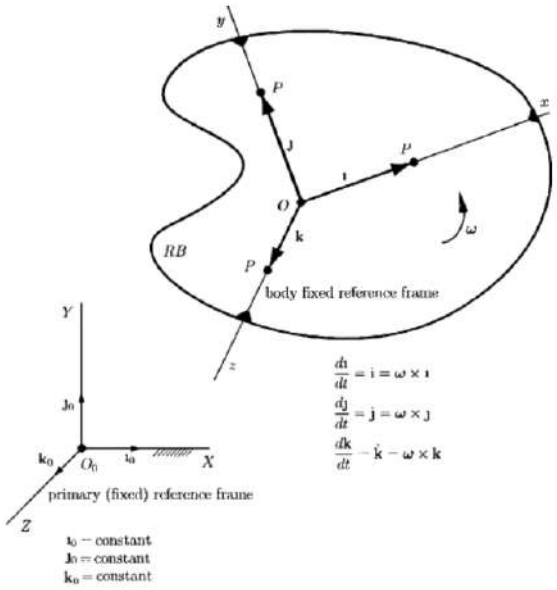


Figure 2.3: Body fixed reference frame

Reference frames are divided into :

- **Inertial Frames:** Frames of reference where Newton's laws are valid without correction.
- **Non-Inertial Frames:** Accelerating reference frames requiring fictitious forces such as centrifugal force.

Kinematics presents the geometry of motion in the kinematic bicycle model, describing motion without considering the forces behind it. In this model, a vehicle is represented as a rigid body with three main components:

- **Front Wheel:** Responsible for steering.
- **Rear Wheel:** The reference point for motion.
- **Wheelbase:** The distance between the front and rear axles.

The model uses a global coordinate system to describe the position and orientation of the vehicle (x, y, θ), while a local coordinate system is used to express motion relative to the

vehicle's frame. The governing kinematic equations are:

$$\dot{x} = v \cos\theta \tag{2.1}$$

$$\dot{y} = v \sin\theta \tag{2.2}$$

$$\dot{\theta} = \frac{v \tan\delta}{L} \tag{2.3}$$

Where (x, y) are the vehicle's position, v is the longitudinal velocity, δ is the steering angle, and θ is the yaw angle, as shown in Figure 2.4. These equations illustrate the evolution of the vehicle's position and orientation over time, under the assumptions of no slip and perfect traction. The steering is restricted to the front wheel [34].

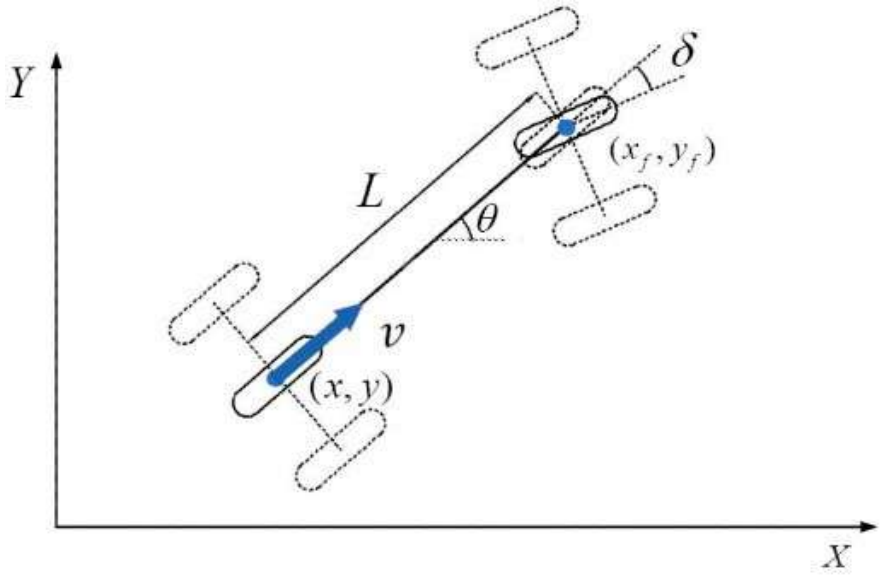


Figure 2.4: Kinematic bicycle model

Limitations

While the kinematic bicycle model is powerful, its reliance solely on kinematics introduces certain limitations:

- It assumes perfect traction (no slipping).
- It ignores dynamic effects like mass distribution, inertia, and tire deformation.
- It is less accurate at higher speeds where lateral forces and tire slip dominate.

2.4.2 Dynamic Bicycle Model

While kinematics describes how a vehicle moves, kinetics (or dynamics) explains the causes of motion by studying the relationship between forces, torques, masses, and accelerations. It extends the geometric description provided by kinematics to include the laws that govern the behaviour of physical systems. The absence of kinetics in the kinematic bicycle models makes it suitable for low-speed scenarios where dynamic effects like tire slip and inertia are negligible. However, for high-speed conditions, dynamic models become necessary [35].

Key Principles of Kinetics

Newton's Laws of Motion:

- **First Law:** A body remains at rest or moves with constant velocity unless acted upon by an external force.
- **Second Law:** The force acting on a body is equal to its mass multiplied by its acceleration:

$$F = ma.$$

- **Third Law:** For every action, there is an equal and opposite reaction.

Work-Energy Theorem: The work done on a system by external forces equals the change in its kinetic energy.

Impulse-Momentum Principle: Impulse applied to a body is equal to the change in its momentum.

Rotational Dynamics: Torque τ and angular acceleration α are related by:

$$\tau = I\alpha,$$

where I is the moment of inertia. The dynamic bicycle model uses Newton's second law to describe the motion of the vehicle [36]. The equations of motion are given below.

Longitudinal Dynamics:

$$\dot{v}_x = \frac{F_x}{m} - v_y \cdot \omega \tag{2.4}$$

where:

- v_x: Longitudinal velocity.
- F_x: Longitudinal force.
- m: Vehicle mass.
- v_y: Lateral velocity.
- ω: Yaw rate.

Lateral Dynamics:

$$\dot{v}_y = \frac{F_y}{m} + v_x \cdot \omega \tag{2.5}$$

where:

- v_y: Lateral velocity.
- F_y: Lateral force.

Yaw Dynamics:

$$\dot{\omega} = \frac{M_z}{I_z} \tag{2.6}$$

where:

- ω: Yaw rate.
- M_z: Moment about the center of mass.
- I_z: Moment of inertia about the yaw axis.

Position and Heading: The inertial coordinates (x, y) and heading angle θ evolve as:

$$\dot{x} = v_x \cos(\theta) - v_y \sin(\theta) \tag{2.7}$$

$$\dot{y} = v_x \sin(\theta) + v_y \cos(\theta) \tag{2.8}$$

$$\dot{\theta} = \omega \tag{2.9}$$

The forces F_x and F_y depend on the tire-road interaction, which includes Lateral forces which are often modelled using the Pacejka "Magic Formula" or linear tire models [37], such as:

$$F_y = C\delta \tag{2.10}$$

where:

- C: Tire cornering stiffness.
- δ: Steering angle.

Longitudinal tire forces are determined by engine torque, braking torque, and road resistance, and can be expressed as:

$$F_x = F_{\text{engine}} - F_{\text{braking}} - F_{\text{resistance}} \tag{2.11}$$

where:

- F_{engine}: Force generated by the engine.
- F_{braking}: Braking force.
- $F_{\text{resistance}}$: Road resistance (including friction and drag).

Frictional forces play a significant role in the slip dynamics at the tires. These forces are a function of the normal forces on the tires and the friction coefficients.

Limitations of the Dynamic Bicycle Model

While the dynamic bicycle model provides a more realistic and comprehensive representation of vehicle behavior, especially at higher speeds and under aggressive maneuvers, it also presents certain limitations:

- **Increased Complexity:** The model includes nonlinear differential equations that require careful numerical integration and parameter tuning, increasing computational burden.
- **Parameter Sensitivity:** Accurate simulation relies on precise values for vehicle mass, tire stiffness, moment of inertia, and other physical parameters, which may vary with loading conditions or tire wear.
- **Tire Modeling Assumptions:** Many dynamic models rely on simplified linear or semi-empirical tire models (e.g., Pacejka), which may not capture complex tire behaviors like combined slip or transient response accurately.
- **Limited Use at Low Speeds:** At low velocities, the dynamic effects become negligible, making the model unnecessarily complex compared to simpler alternatives like the kinematic model.

2.4.3 Implementation of the Kinematic Bicycle Model

The kinematic bicycle model is chosen over the dynamic model for this study. Since the study primarily focuses on longitudinal control, the dynamic intricacies of lateral forces and moments are not critical. Additionally, the kinematic model provides a solid foundation for integrating longitudinal controllers, allowing straightforward testing and validation of speed control strategies that will be discussed in the next chapters. The simulation model is implemented using two distinct approaches, a block-based Simulink model and a custom S-function model written in MATLAB code.

Simulink Block-Based Implementation

The block-based implementation is constructed using Simulink graphical interface, employing standard blocks to represent the equations of motion while offering quick setup and testing of control algorithms, such as PID controllers, without requiring in-depth coding knowledge. The block diagram of the Simulink implementation is shown in Figure 2.5.

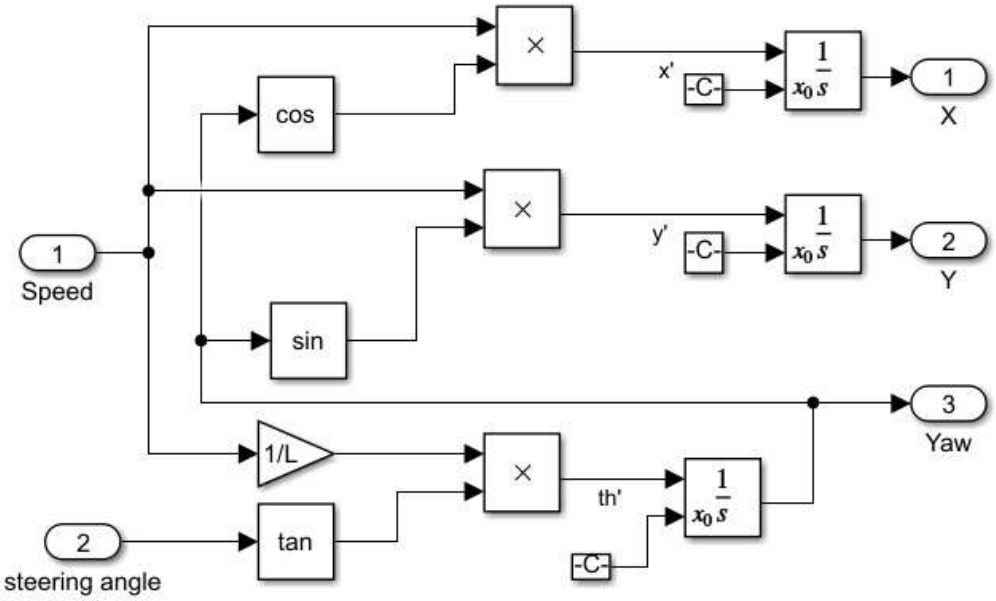


Figure 2.5: Simulink block diagram for the kinematic bicycle model.

S-Function Implementation

The S-function implementation involves a custom MATLAB code to define the vehicle's differential equations of motion in a program. It allows for advanced customization, making it

suitable for implementing more sophisticated control strategies requiring custom integration of model parameters .
The MATLAB code for the S-function implementation is provided in Appendix A (Section A).

2.5 Conclusion

This chapter began with an overview of autonomous electric vehicles, highlighting the integration of electric propulsion and autonomous technologies as a cornerstone of modern transportation. The efficiency, responsiveness, and environmental advantages of the electric powertrain were examined, establishing a technological foundation for developing intelligent, sustainable mobility systems.
Building on this foundation, the application of bicycle models in vehicle modelling was explored, with both kinematic and dynamic representations discussed. The kinematic bicycle model was selected for its simplicity and suitability in capturing essential motion characteristics relevant to this study. Modelling approaches were developed to implement simulation models, comparing the flexibility of block diagrams with the computational precision of S-functions in Simulink. The block diagram approach proved effective for prototyping and testing basic control strategies, while the S-function enabled high-performance, dynamic simulations. Together, these elements form a solid groundwork for the next chapters, where advanced control methodologies will be employed to further enhance vehicle performance and robustness.

Chapter 3

Trajectory Generation and Controller Design

3.1 Introduction

Motion planning is an important research topic within the automotive field, it includes path planning that involves defining a geometric path, and trajectory planning (also known as trajectory generation). The latter goes beyond defining a path; it involves creating smooth and continuous motion profiles by optimizing abrupt changes in acceleration. The implementation methods for this task range from polynomial-based approaches to optimization techniques [38].

3.2 S–Curve Profile

An S-curve profile is a type of function characterized by a piecewise construct that generalizes different families of profiles using a dimensionless parameter Γ, which varies from 0 to 1, allowing profile selection. The parameters involved in the calculation of this profile are given in Table 3.1, along with the initialization sequence. The equations used are based on the work of [39].

Parameter	Initialization
Midpoint position Q_s	$\frac{D}{2}$
Auxiliary position Q_{aux}	$\frac{(1+\Gamma)}{2} \cdot \frac{V_m^2}{A_m}$
Adjusted position Q_a	$\begin{cases} Q_s, & Q_s \leq Q_{\text{aux}} \\ Q_{\text{aux}}, & Q_s > Q_{\text{aux}} \end{cases}$
Effective Velocity V_w	$\begin{cases} \sqrt{\frac{Q_s \cdot A_m}{\frac{1}{2}(1+\Gamma)}}, & Q_s \leq Q_{\text{aux}} \\ V_m, & Q_s > Q_{\text{aux}} \end{cases}$
Auxiliary constant T_o	$\frac{V_m}{A_m}$
Transition time T_a	$T_o \cdot (1+\Gamma)$
Intermediate time T_m	$\frac{T_a}{2}$
Time constant τ	$\Gamma \cdot T_o$
Constant velocity period T_k	$2 \cdot \frac{Q_s - Q_a}{V_m}$
Midpoint time T_s	$T_a + \frac{T_k}{2}$
Total Time T_t	$2 \cdot T_s$

Table 3.1: Parameters initialization for the S-curve profile

The key parameters are:

- V_m: the maximum speed
- A_m: the maximum acceleration
- D: the total travelled distance

The position $q(t)$, speed $v(t)$, and acceleration $a(t)$ of the vehicle are described using piecewise functions, which account for the vehicle's behaviour over different time intervals and conditions.
Position $q(t)$ is defined as:

$$q(t) = \begin{cases} 0, & t \leq 0 \\ \frac{1}{2} \cdot \frac{V_w}{T_o} \cdot \frac{t^3}{3 \cdot \tau}, & 0 < t < \tau \\ \frac{1}{2} \cdot \frac{V_w}{T_o} \cdot \left((t - \frac{\tau}{2})^2 + \frac{\tau^2}{12} \right), & \tau < t \leq T_o \\ Q_s + V_w \cdot (t - T_s) + q(T_a - t), & T_o < t \leq T_s \\ Q_f - q(T_t - t), & t > T_s \end{cases} \tag{3.1}$$

Speed $v(t)$, the derivative of position with respect to time, is given by:

$$v(t) = \begin{cases} 0, & t \leq 0 \\ \frac{A_m}{2} \cdot \frac{t^2}{\tau}, & 0 < t < \tau \\ \frac{A_m}{2} \cdot (2 \cdot t - \tau), & \tau < t \leq T_o \\ V_w - v(T_a - t), & T_m < t \leq T_s \\ v(T_t - t), & t > T_s \end{cases} \tag{3.2}$$

Acceleration $a(t)$, the derivative of speed with respect to time, is expressed as:

$$a(t) = \begin{cases} 0, & t \leq 0 \\ \frac{A_m}{\tau} \cdot t, & 0 < t < \tau \\ A_m, & \tau < t \leq T_o \\ a(T_a - t), & T_m < t \leq T_s \\ -a(T_t - t), & t > T_s \end{cases} \tag{3.3}$$

The jerk $j(t)$ (change in acceleration) is similarly obtained as follows:

$$j(t) = \begin{cases} 0, & t \leq 0 \\ J_m, & 0 < t < \tau \\ 0, & \tau < t \leq T_o \\ -j(T_a - t), & T_m < t \leq T_s \\ j(T_t - t), & t > T_s \end{cases} \tag{3.4}$$

Where J_m is the jerk limit, an optimized value that ensures comfort and smooth changes in acceleration.

The graphs presented in Figure 3.1 depict the S-curve profile for different values of Γ. The chosen scenario for this simulation is based on Intelligent Transport Systems standards [40], where the speed limit V_m is set to $28.8km/h(8m/s)$ suitable for urban environments, and the road is a segment of length $D = 2km$.

Referring to the ISO 2631-1 standards for comfort perception (Table 3.2) [41], the acceleration range of [0.315 m/s^2 - 0.63 m/s^2] is selected, and considering urban travel, while maintaining safety, smooth driving, and passenger comfort, an exact acceleration value of $A_m = 0.4m/s^2$ is chosen.

ACCELERATION RANGE	**COMFORT PERCEPTION**
<0.315 m/s^2	Not uncomfortable
0.315 m/s^2 - 0.63 m/s^2	A little uncomfortable
0.50 m/s^2 - 1.00 m/s^2	Fairly uncomfortable
0.80 m/s^2 - 1.60 m/s^2	Uncomfortable
1.25 m/s^2 - 2.5 m/s^2	Very uncomfortable
>2 m/s^2	Extremely uncomfortable

Table 3.2: ISO 2631-1 standard comfort caution ranges

For $\Gamma = 0$, the results present a 3-segment profile, also known as the trapezoidal profile [42], where the velocity graph increases linearly until it reaches the maximum value $V_w = 8\text{m/s}$, which remains constant for $T_k = 229.6\text{s}$, and then decreases linearly to zero. This results in a trapezoidal shape on the velocity-time graph, characterized by its linear increase and decrease with a flat, constant section in the middle. Meanwhile, the acceleration plot remains constant during the acceleration and deceleration phases and null in between, creating a piecewise constant graph with two distinct levels of acceleration. These instantaneous changes in acceleration lead to an infinite jerk, represented by discontinuities at the points of change in acceleration.

For $0 < \Gamma \leq 1$, a 7-segment profile (S-curve profile) is introduced. The acceleration slope is reduced from being a vertical line, which results in a continuous, smooth velocity curve that is free of sharp corners, providing a more realistic and comfortable motion profile. Overall, it is observed that the resulting S-curve profiles feature smooth, gradual changes in acceleration and deceleration ensuring continuity, eliminating abrupt changes which results in a finite jerk. Table 3.3 illustrates the comparison between profile parameters corresponding to different Γ values. It is evident that as Γ increases, jerk value decreases which leads to smoother transitions.

Γ	0.25	0.5	0.75	1.0
$J_m(\text{m/s}^2)$	0.08	0.04	0.0267	0.02
$T_a(\text{s})$	25	30	35	40
$T_t(\text{s})$	275	280	285	290

Table 3.3: Comparison of parameters for different values of Γ

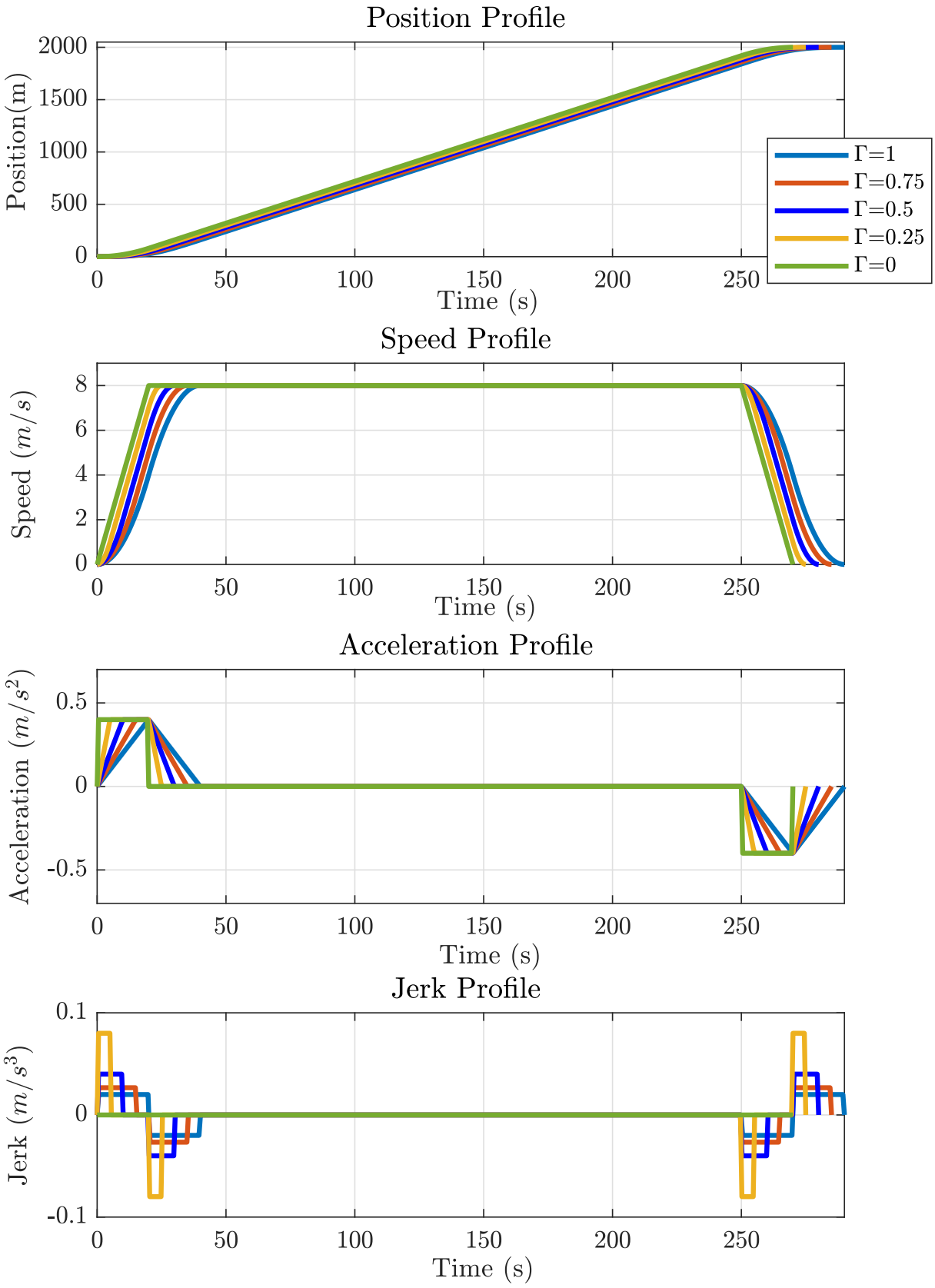


Figure 3.1: S-curve profiles

In summary, while the trapezoidal profile is simple and fast, its sudden changes, abrupt transitions, and high jerk make it unsuitable for ensuring safe and comfortable motion. Hence, S-curve profiles with higher Γ values are more suitable for autonomous vehicles. However, the specific selection of $\Gamma = 0.5$ offers a compromise between efficiency and smoothness. As observed from the table, $\Gamma = 0.5$ represents a moderate jerk value of 0.04m/s^2, which is significantly lower than the jerk values for smaller Γ values. This reduction in jerk ensures smoother transitions without excessively prolonging the acceleration and deceleration phases. Furthermore, it strikes an optimal balance by providing a reasonable acceleration time ($T_a = 30\text{s}$) and total time ($T_t = 280\text{s}$), making it an efficient and practical choice for the desired motion profile.

3.3 Sinusoidal Profile

The S-curve profile represents a smooth and comfortable motion design. However its rate of change in acceleration, and the presence of rectangular jerk pulses are a drawback. To mitigate this issue, the Sinusoidal profile is introduced. This profile uses sinusoidal waveforms to ensure continuity in acceleration, which effectively minimizes spikes in jerk for a smoother passenger experience [43].
The initialization sequence for this profile is given in Table 3.4. Q_s, Q_a, T_o, T_k, T_s, and T_t remain unchanged from Table 3.1.

Parameter	**Initialization**
Auxiliary position Q_{aux}	$Q_{\text{aux}} = \frac{V_m^2}{A_m}$
Effective Velocity V_w	$V_w = \begin{cases} \sqrt{Q_s \cdot A_m} & \text{if } Q_s \leq Q_{\text{aux}} \\ V_m & \text{if } Q_s > Q_{\text{aux}} \end{cases}$
Transition time T_a	$T_a = 2 \cdot T_o$
Angular frequency ω	$\omega = \frac{2\pi}{T_a}$
Scaling constant K_s	$K_s = \frac{T_a \cdot V_w}{4\pi^2}$

Table 3.4: Parameters initialization for the sinusoidal profile

The position $q(t)$, speed $v(t)$, and acceleration $a(t)$ are given by:

$$q(t) = \begin{cases} 0, & t \leq 0 \\ \frac{A_m}{4} \cdot t^2 + K_s \cdot (\cos(\omega \cdot t) - 1), & 0 < t \leq T_a \\ Q_s + V_w \cdot (t - T_s), & T_a < t \leq T_s \\ D - q(T_t - t), & t > T_s \end{cases} \tag{3.5}$$

$$v(t) = \begin{cases} 0, & t \leq 0 \\ K_s \cdot \omega \cdot (\omega \cdot t - \sin(\omega \cdot t)), & 0 < t \leq T_a \\ V_w, & T_a < t \leq T_s \\ v(T_t - t), & t > T_s \end{cases} \tag{3.6}$$

$$a(t) = \begin{cases} 0, & t \leq 0 \\ \frac{A_m}{2} \cdot (1 - \cos(\omega \cdot t)), & 0 < t \leq T_a \\ 0, & T_a < t \leq T_s \\ -a(T_t - t), & t > T_s \end{cases} \tag{3.7}$$

The jerk $j(t)$ is as follows:

$$j(t) = \begin{cases} 0, & t \leq 0 \\ \frac{A_m}{2} \cdot (\omega \cdot \sin(\omega \cdot t)), & 0 < t \leq T_a \\ 0, & T_a < t \leq T_s \\ j(T_t - t), & t > T_s \end{cases} \tag{3.8}$$

Figure 3.2 presents the results of a comparative study between the sinusoidal profile and the S-curve profile, with the parameters set to $\Gamma = 0.5$, while maintaining the conditions from the previous scenario: a distance of $D = 2km$, a maximum velocity of $V_m = 8\,\mathrm{m/s}$, and an acceleration of $A_m = 0.4\,\mathrm{m/s^2}$. The velocity responses for both profiles demonstrate smooth transitions, indicating that both profiles effectively manage acceleration and deceleration without abrupt changes. However, a notable distinction lies in the total transition times. The S-curve profile completes its transition in $T_a = 30\,\mathrm{s}$, while the sinusoidal profile takes a longer time of $T_a = 40\,\mathrm{s}$, indicating that the sinusoidal profile leads to a slower overall response.

The S-curve acceleration profile is inherently designed to change in a more gradual manner. Its acceleration follows a curve that starts at zero, ramps up smoothly to the maximum value, and then decelerates smoothly back to zero. This produces a trapezoidal acceleration profile, where the rate of change of acceleration (jerk) is nonlinear but controlled, avoiding any sharp changes. Nevertheless, the jerk in the S-curve profile exhibits some abrupt shifts, leading to minor discontinuities. In contrast, the sinusoidal profile ensures continuous changes in acceleration, following a smooth oscillating function. As a result, the jerk is also continuous and smooth, without the sharp transitions. The sinusoidal profile typically exhibits a more harmonic content in its jerk, meaning that it oscillates in a regular manner, producing smoother transitions with less abruptness.

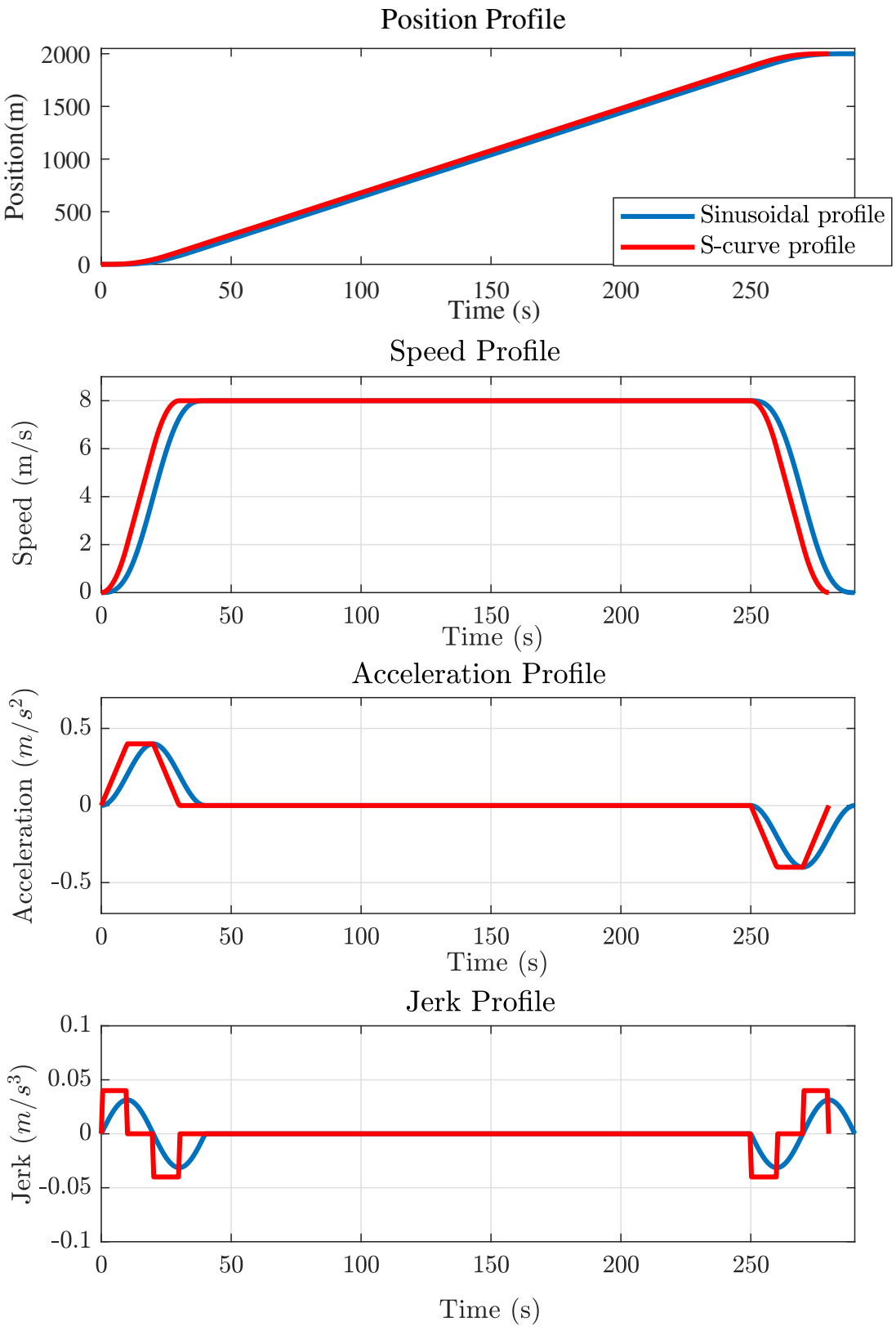


Figure 3.2: Graphical result of the Sinusoidal and S-curve profiles comparison

In summary, the choice between the S-curve and sinusoidal motion profiles depends on the specific demands of the autonomous vehicle's trajectory. The S-curve profile is optimal for efficiency and minimizing abrupt motion changes over extended and relatively straight paths, offering a well-balanced trade-off between smoothness and computational load. In contrast, the sinusoidal profile is preferable in contexts where maximum smoothness and ride comfort are critical, such as navigating tight corners or operating in sensitive environments. Ultimately, the selection of the most appropriate profile should consider the operational context, weighing performance, comfort, and computational cost.

3.4 Controller Design

This section defines the general structure of a control system for autonomous vehicle motion, focusing on the two primary control categories: lateral and longitudinal control. These distinct but interdependent control aspects are fundamental for achieving reliable and efficient autonomous driving performance, as they handle steering and speed regulation, respectively. A Proportional-Integral (PI) controller is chosen for the steering control, as its robustness and simplicity make it well-suited for achieving precise lateral control, effectively addressing path-following and stability requirements. This control approach is implemented throughout the remainder of this thesis. For longitudinal control, a PID controller is selected to regulate the vehicle speed. This choice is made due to its simplicity and effective response to speed tracking under varying conditions. However, in the remainder of this study, the performance of the PID controller is compared to alternative control techniques to evaluate its effectiveness relative to more advanced methods in longitudinal dynamics.

3.5 Lateral Control

The yaw angle is the main parameter in lateral control, which is managed by adjusting the steering angle to maintain a desired trajectory. An appropriate actuator must be selected for the steering control system to accurately execute the required adjustments to the steering angle. The actuator chosen for this section is a stepper motor characterized by a transfer function, as defined in [44]. Figure 3.3 illustrates the control system, where the PI controller is implemented with the following tuned parameters: $P = 1$, $I = 2.9$.

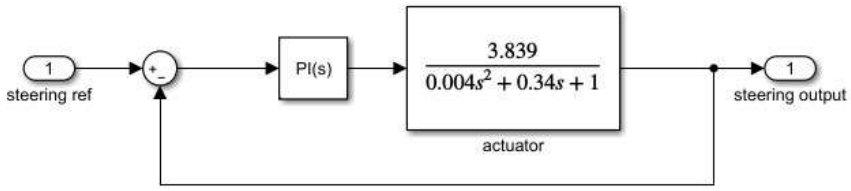


Figure 3.3: Steering angle control block diagram

This control system is evaluated on a trajectory featuring a single turn, defined by a steering angle of $\delta = \frac{\pi}{12}$ radians. The graphical outputs of this test are presented in Figure 3.4, the steering angle output follows the steering reference with high precision. Although a slight delay of 0.26 seconds is observed in the response, the system demonstrates robust performance, with no noticeable overshoot.

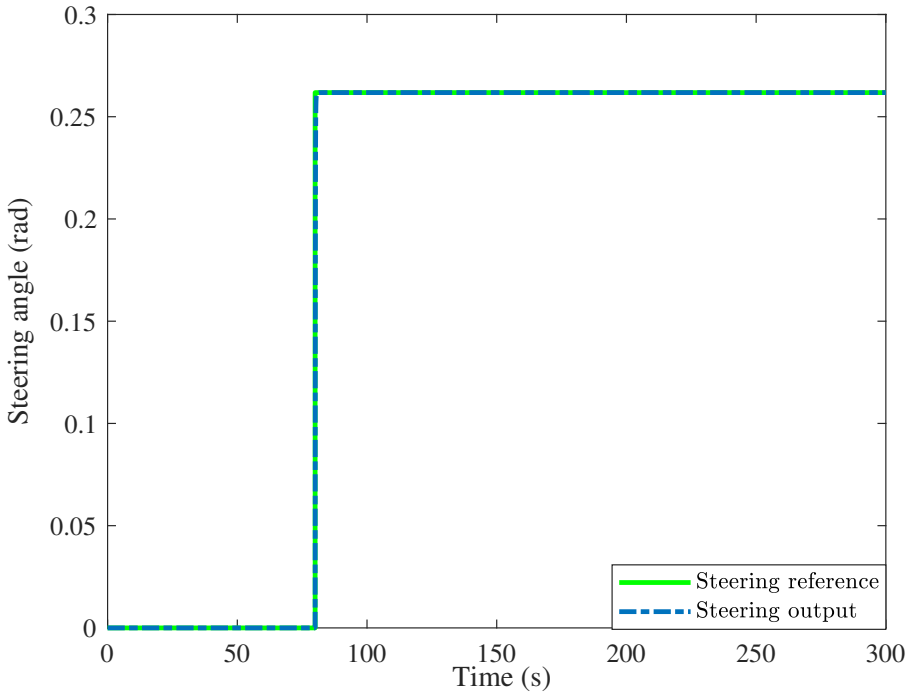


Figure 3.4: Steering angle simulation result

3.6 Longitudinal Control

Longitudinal control is responsible for the vehicle's speed control along a straight or a curved path. The primary objective is to closely track a velocity profile designed to decouple longitudinal and lateral dynamics, enhancing the stability of the overall control system. For the actuation of longitudinal control, a DC motor is selected, with the mathematical modelling and parameter specifications provided below [25].

$$V = R_a \cdot i_a + e + L_a \cdot \frac{di_a}{dt} \tag{3.9}$$

$$e = k \cdot \Omega \tag{3.10}$$

$$J \cdot \frac{d\Omega}{dt} = T_{em} - f \cdot \Omega - T_L \tag{3.11}$$

$$T_{em} = k_{em} \cdot i_a \tag{3.12}$$

Where:

- V: the input voltage.
- e: the back electromotive force (EMF).
- i_a: the current in the motor winding.
- T_{em}: the electromagnetic torque.
- Ω: the angular velocity.

The parameters are chosen based on similarity with the Tesla Model S, with a torque $T_L = 430\,\text{Nm}$, as referenced in [25]. See Table 3.5.

Table 3.5: DC Motor parameters

Parameter	**Value**
Armature resistance R_a	0.193 Ω
Armature inductance L_a	0.00383 H
Back EMF constant k	2.332232 $V.s$
Torque constant k_{em}	2.1717 $N.m/A$
Moment of inertia J	0.6 $kg.m^2$
Coefficient of friction f	2.632177$N.m.s$

A PID controller is employed for the velocity control system, where the controller gains are carefully tuned to achieve an optimized dynamic response. $K_P = 19$, $K_I = 100$, and $K_D = 0.5$. The configuration of the complete control system is shown in Figure 3.5.

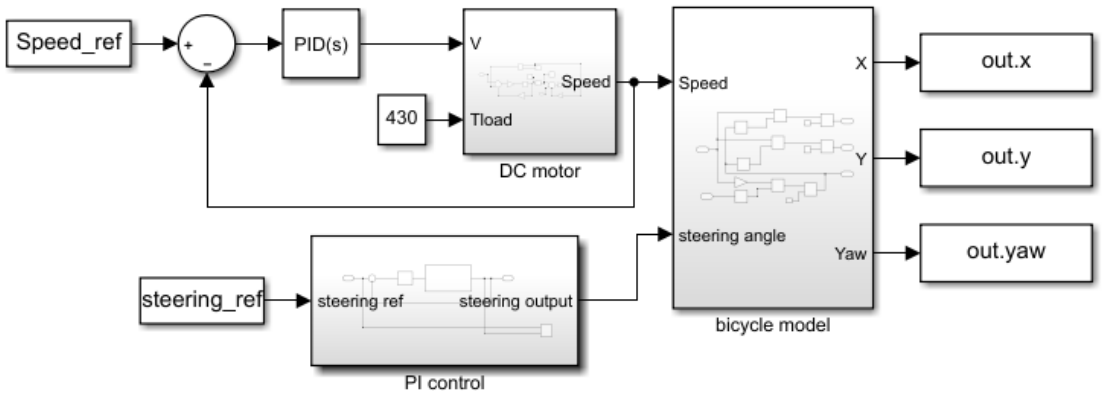


Figure 3.5: PID control system

The speed response of the PID control system is illustrated in Figure 3.6 . The reference speed is the sinusoidal velocity profile discussed earlier. It is observed that the suggested controller shows a good stability and tracking performance. However, a noticeable delay is present, which could affect the response time and overall performance of the control system in more demanding scenarios.

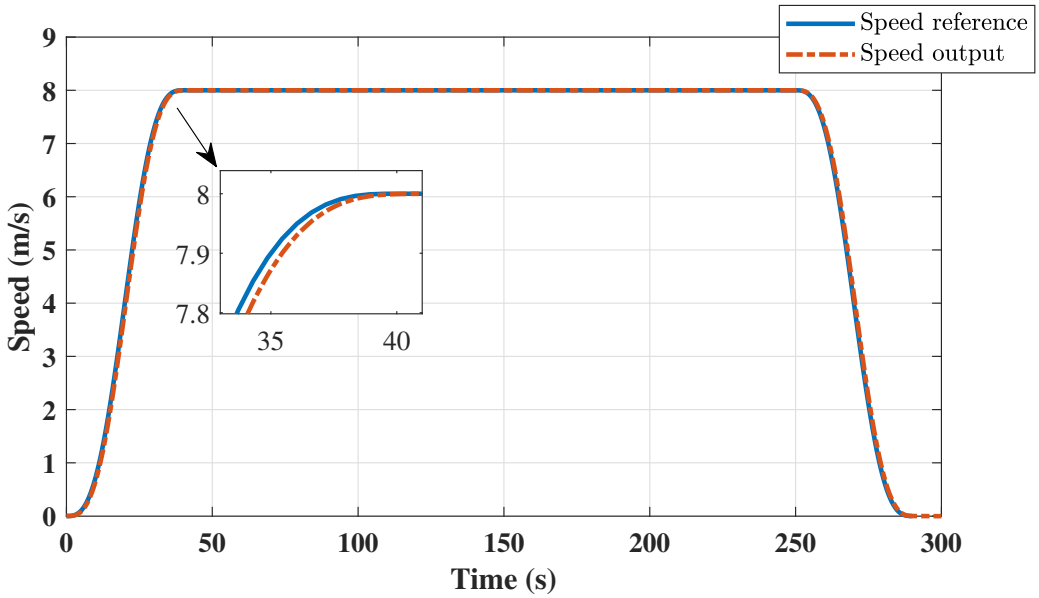


Figure 3.6: Simulation result for the speed control system

3.7 Conclusion

Following the design and implementation of both longitudinal and lateral control systems, the results demonstrate the effectiveness of classical control approaches under predefined conditions. The PI controller employed for lateral control, demonstrated satisfactory performance in tracking. Given its stability and simplicity, it will be retained for the remainder of this thesis.

In contrast, although the PID controller applied to longitudinal dynamics provided acceptable stability and speed tracking performance, it exhibited a noticeable response delay. This limitation suggests that traditional PID control may not be sufficient for more dynamic or uncertain environments. Consequently, the next phase of this work will explore the integration of advanced control strategies, such as metaheuristic optimization methods, to enhance longitudinal control accuracy and responsiveness.

Chapter 4

Metaheuristics in Control and Parameter Optimization

4.1 Introduction

This chapter presents a synthesis of the fundamental principles underlying Metaheuristic algorithms. This synthesis examines their objectives, methodologies, and findings, aiming to provide the general structure of such algorithms, and discusses their advantages and limitations. This synthesis not only consolidates current knowledge but also identifies potential directions for future research and practical applications. Furthermore, the specific metaheuristics used in this work are introduced. Their application to parameter tuning and control system optimization is also highlighted, demonstrating their relevance and effectiveness in addressing complex real-world applications.

4.2 Metaheuristic Algorithms

Heuristics are problem-solving strategies based on trial and error to achieve satisfactory solutions. The term "meta" means beyond or higher-level. Combined together, they form the origin of metaheuristics, high-level algorithms that guide and adapt subordinate heuristics to discover solutions that go beyond local optimality. These approaches integrate local search techniques with randomization in a structured manner, enabling them to efficiently generate high-quality solutions for complex optimization problems within a reasonable time frame [14]. The key characteristics that contribute to their robustness and adaptability are the following.

- Problem-independence;
- Stochastic behaviour;
- Exploration and exploitation balance;
- No gradient requirement;
- Adaptability and flexibility;
- Master strategy role.

One of the most important traits is their problem-independence. Due to their low need for substantial modification, they can be applied to a wide range of optimization problems. Additionally, they typically incorporate stochastic behaviour, relying on randomness during the search process to help escape local optima and enhance global exploration, which explains their ability to balance exploration and exploitation. Furthermore, they do not require gradient information, making them suitable for functions that are non-differentiable,

noisy, or black-box in nature, which extends their adaptability and flexibility, as they can be hybridized with other methods or tailored to specific problem domains. Importantly, metaheuristics serve as master strategies, coordinating the behaviour of lower-level heuristics to guide the optimization process effectively.

The process of metaheuristics is fundamentally iterative, working on a given measure of quality to improve a candidate solution. Rather than focusing on precise mathematical modelling, these algorithms are built on procedural logic and heuristic rules, making them effective even when dealing with incomplete or uncertain information. Their design allows them to be adapted to diverse application areas and encourages integration with other optimization strategies (hybrid optimization). By incorporating feedback and adaptive mechanisms, metaheuristics can overcome local optima and more effectively navigate complex or poorly characterized search spaces. Consequently, they are widely adopted as practical tools for solving optimization problems that are challenging or infeasible for traditional, exact, or gradient-based methods due to complexity or computational cost [45].

4.2.1 Classification of Metaheuristics

Metaheuristics has evolved over multiple decades, characterized by ongoing advancements in the creation of new algorithms and the enhancement of existing ones, in order to improve optimization performance. Heuristics and even metaheuristics were used, but not formally studied in the 1940s. This phase of the timeline is called the pre-theoretical period. Since optimization has always been a part of human and animal behaviour, long before its discovery as a field of study. In daily life, optimization problems are introduced and solved without intention, such as choosing the quickest route to work, scheduling daily tasks, or deciding how to invest money for the best return. These are examples of classic optimization problems that are treated with advanced techniques. However, humans and also animals instinctively tackle these problems without formal training or awareness of concepts such as finding exact solutions compared to approximate ones, or distinguishing between straightforward and challenging problems. This natural intuition evolved due to the need to increase the chances of survival. The human brain is naturally wired to respond decisively in critical situations, instinctively applying decision-making strategies that closely resemble modern optimization techniques. Much like the physiological systems of the human body are built for survival, the mind has evolved to prioritize quick, practical solutions over perfect ones.

This ability to act under pressure by favouring speed and adequacy mirrors the core philosophy behind contemporary heuristic and metaheuristic approaches, which seek satisfactory

solutions within limited time and resources rather than exhaustively searching for an exact optimum. For example, when confronted with a novel problem, individuals often instinctively search for similar issues they have previously resolved, applying or adapting those solutions. This is known as learning by analogy. Another instinctive method is means-end analysis, where one continually moves from the current state to a state closer to the desired goal until reaching a satisfactory solution, much like local search strategies in modern optimization algorithms. Interestingly, while heuristic reasoning has been an integral part of life for millennia, exact optimization methods only emerged relatively recently. In the early 1950s, stochastic optimization algorithms such as Monte Carlo methods were studied, introducing probabilistic techniques for finding good solutions in complex search spaces. Evolutionary algorithms emerged in the following decade, inspired by the principles of natural selection and biological evolution. These evolutionary strategies laid the groundwork for the widespread adoption of Genetic Algorithms (GA) during the 1980s, a subset of evolutionary algorithms that has emerged as one of the most widely adopted metaheuristics for addressing a broad range of optimization challenges. Swarm Intelligence-based metaheuristics emerged in the 1990s, introducing optimization techniques inspired by the collective behaviour of decentralized, self-organized systems such as ant colonies, bird flocks, and fish schools. These approaches, including Particle Swarm optimization, have since become fundamental methods for solving complex computational problems. Since the 2000s, research has increasingly focused on hybrid metaheuristics and multi-objective optimization algorithms to solve problems involving conflicting objectives [46]. Figure 4.1 illustrates the historical development, summarizing the key milestones in the evolution of metaheuristic optimization algorithms and providing a clearer perspective on their progression.

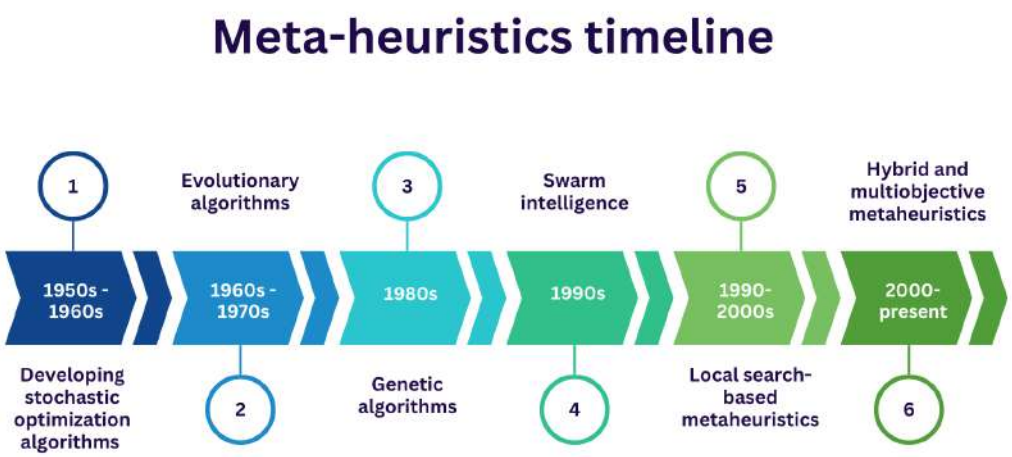


Figure 4.1: Historical timeline of metaheuristic algorithms development

Metaheuristic algorithms can be broadly classified according to various criteria such as:

- Search strategy (single-solution vs. population-based);
- Inspiration source (nature or non-nature);
- Memory usage;
- Neighbourhood and search movement;
- Solution update mechanism (deterministic vs. stochastic).

One of the most common classifications is the search strategy, where the algorithms are divided into two main categories: single-solution-based and population-based algorithms. As the name suggests, single-solution-based metaheuristics focus on exploring and improving a single candidate solution throughout the search process. In contrast, population-based metaheuristics maintain and evolve a set of candidate solutions simultaneously, enabling a more diverse exploration of the search space. Furthermore, metaheuristics can be nature-inspired or non-nature-inspired. Their classification is based on whether their inspiration is drawn from natural phenomena or other abstract principles. Nature-inspired metaheuristics are developed by drawing analogies from processes and behaviours observed in nature. This category includes a wide range of algorithms, and can be classified into four principal groups: evolutionary algorithms, swarm-based algorithms, physics-based algorithms, and

human-based algorithms. The classification is illustrated in Figure 4.2. Evolutionary algorithms (EAs) draw inspiration from the principles of natural selection and genetics. They operate on a population of potential solutions, applying processes analogous to biological evolution such as selection, crossover (recombination), mutation, and survival of the fittest. These evolutionary operations progressively enhance the quality of solutions over successive generations, facilitating optimal problem-solving. Prominent examples of EAs include Genetic Algorithms, Genetic Programming (GP), and Evolutionary Strategies (ES).

Swarm-based algorithms, as the name suggests, are inspired by the swarm intelligence observed in nature, where swarms, such as bird flocks, fish schools, and ant colonies, exhibit coordinated, self-organizing behaviour without centralized control. This principle is based on the fact that simple local interactions among individual agents can lead to sophisticated global solutions. Notable examples of swarm-based metaheuristics include Particle Swarm Optimization (PSO), Ant Colony Optimization, and Jellyfish Search Optimizer (JSO). The source of physics-based metaheuristics is inspired by physical laws and phenomena governing natural systems, where candidate solutions evolve based on fundamental physical principles such as energy minimization, gravitational attraction, and motion dynamics. By drawing analogies from natural forces that govern the universe, these metaheuristics simulate how particles, masses, or energetic states interact within a system, gradually converging toward optimal solutions. Examples of such algorithms include Simulated Annealing (SA), which imitates the annealing process in metallurgy, Gravitational Search Algorithm (GSA) based on Newtonian gravity, Harmony Search (HS) inspired by musical improvisation, and Central Force Optimization (CFO) simulating central force fields [47]. Among these nature-inspired metaheuristics, this work primarily investigates two algorithms: Genetic Algorithms and Jellyfish Search Optimizer.

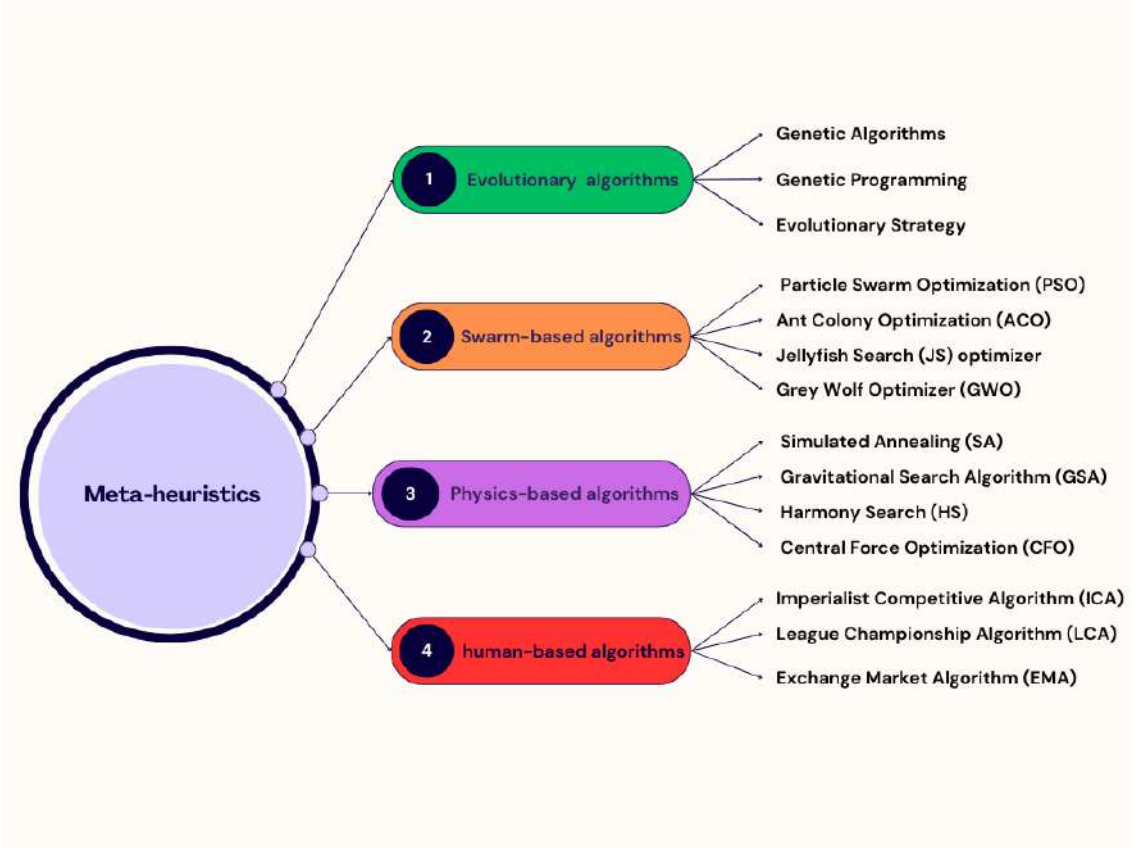


Figure 4.2: Classification of nature-inspired metaheuristics

4.2.2 Genetic Algorithms

Genetic Algorithms are evolutionary metaheuristic algorithms grounded in population dynamics and inspired by the concepts of natural selection and evolutionary processes. They progressively improves potential solutions through selection, crossover, and mutation, ultimately converging on optimal parameters. Rooted in the idea that evolution is one of nature's most effective problem-solving processes, GAs simulate this adaptive mechanism to explore and exploit complex search spaces. While natural evolution produces highly adaptive lifeforms capable of sophisticated behaviours in ever-changing environments, GAs translate these concepts into computational procedures that iteratively refine potential solutions. A key distinction between evolutionary algorithms like GA and other nature-inspired metaheuristics lies in their foundational analogy while systems like swarm intelligence draw from the collective behaviours of animals such as birds or fish, GAs emulate the evolutionary process itself. The real strength of GAs lies in their use of three central ideas: preserving a diverse and robust solution pool, integrating existing solutions through recombination to produce new

candidates, and strategically regulating diversity to prevent premature convergence [45].

Selection is the decision factor on which solutions will contribute to producing the next generation. Inspired by natural selection, individuals with better fitness scores are more likely to be chosen, while less fit solutions are occasionally retained to maintain diversity rather than prematurely settling into local optima. Common techniques include tournament selection and roulette wheel selection.

Crossover, also called recombination, generates new candidate solutions by combining features from two or more parent solutions. This process mimics the exchange of genetic material in nature, in order to inherit desirable traits from parents and produce potentially better offspring. Typical crossover methods include one-point, two-point, and uniform crossover.

Mutation introduces stochastic variations into candidate solutions to preserve population diversity and enhance exploration. In genetic algorithms, this typically entails modifying one or more elements of a solution with a low probability. Mutation serves to prevent excessive similarity among solutions, reducing the risk of convergence to local optima and strengthening the algorithm's overall search efficiency [48].

To better illustrate the operational workflow of the genetic algorithm, a general flowchart is presented in Figure 4.3. This diagram outlines the key steps of the algorithm, from initial population generation to the iterative processes of selection, crossover, mutation, and final convergence [49].

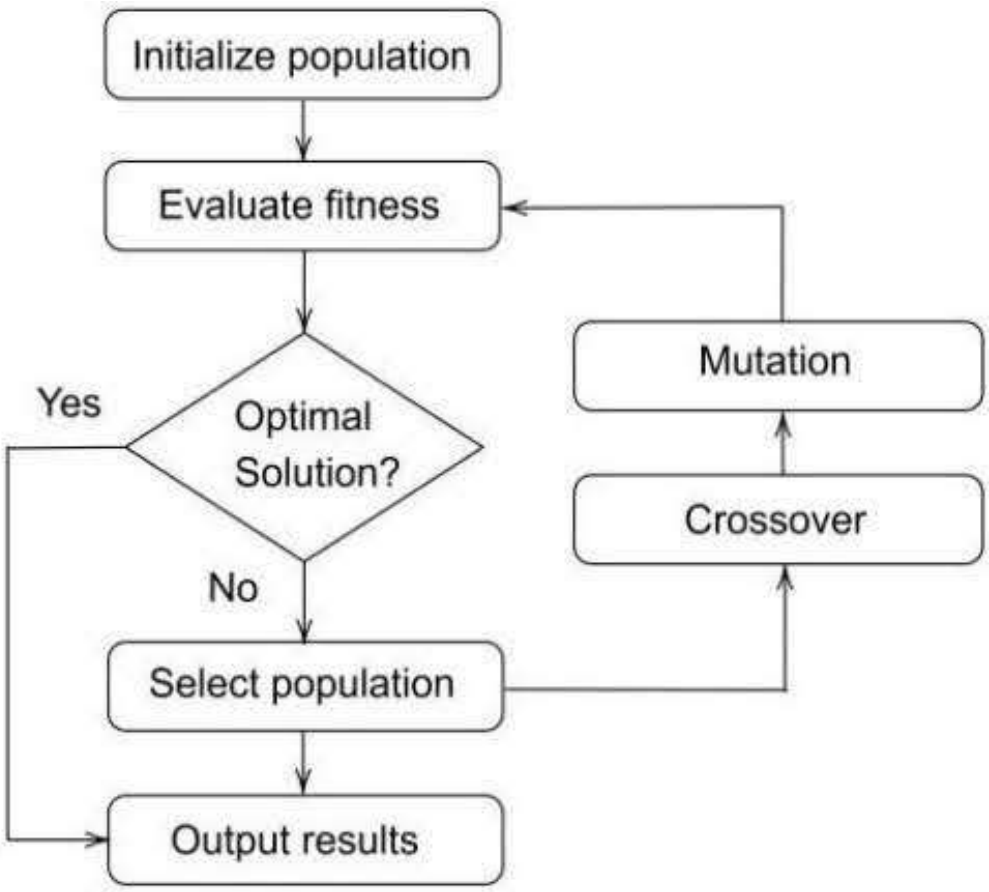


Figure 4.3: General flowchart illustrating the main steps of a genetic algorithm.

4.2.3 Jellyfish Search Optimizer

The Jellyfish Search optimizer is a recent metaheuristic optimization algorithm developed by Jui-Sheng Chou et al. [50]. It is built based on the behaviour of the jellyfish food-seeking process in the ocean. The first step involves following the ocean current to find regions rich in food resources (optimal solutions), this step is called the exploration phase. Over time, a jellyfish swarm is created, and each jellyfish pursues the motion within the swarm to discover a more optimal location, which is translated as active and passive motions, representing the exploitation phase. Meanwhile, time control is introduced to identify the varied motion types and regulate the transition, which makes the optimum phase. The particularity of this algorithm is its adaptive mechanism that alternates between exploration and exploitation phases, dynamically regulated by a time control function. A scheduling mechanism that adjusts the algorithm's behaviour as the optimization progresses. Its main job is to decide when the algorithm should focus on exploration and when it should switch to exploitation.

During the initial phases, the algorithm prioritizes exploration to extensively survey the solution space. As optimization progresses, the focus gradually shifts toward exploitation, refining candidate solutions and enhancing convergence reliability. As depicted in Figure 4.4 [51], the natural movement of jellyfish serves as a guiding metaphor for balancing exploration and exploitation in the optimization process.

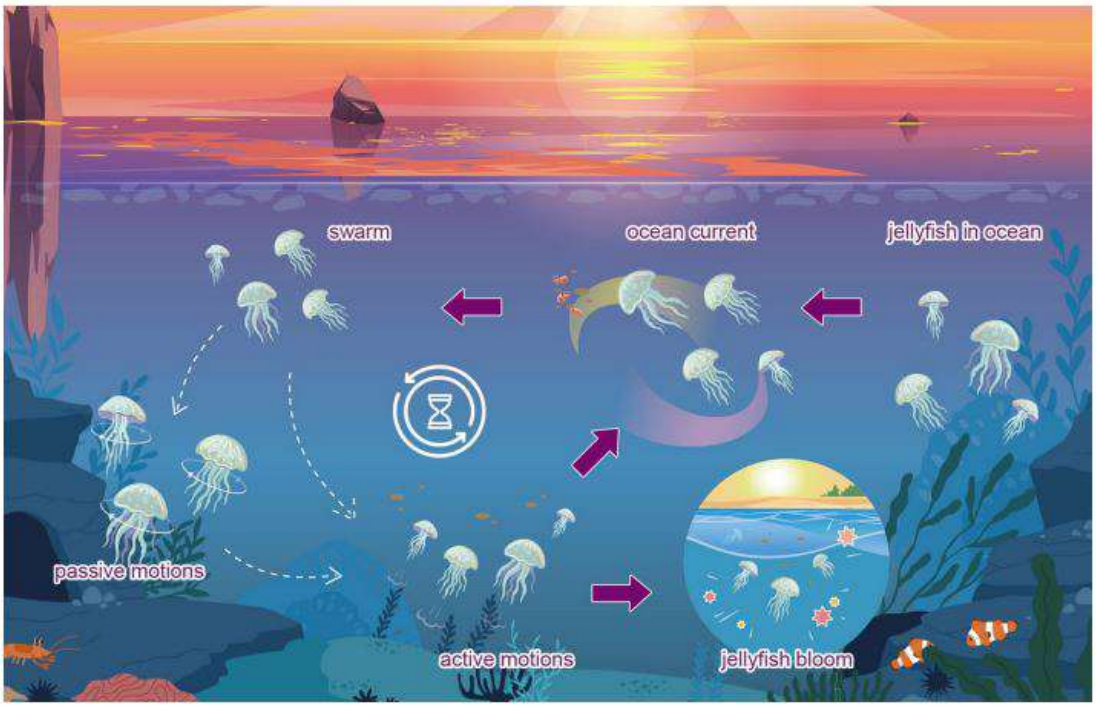


Figure 4.4: Conceptual illustration of jellyfish behaviour.

The mechanism process of the Jellyfish Search (JS) optimizer is governed by a series of mathematical equations, as detailed in [50]. These are presented below.

Trend calculation:

$$\vec{tr} = Z^* - \beta \times \text{rand}(0,1) \times \mu \tag{4.1}$$

Active motion update:

$$Z_i(t+1) = Z_i(t) + \text{rand}(0,1) \times (Z^* - \beta \times \text{rand}(0,1) \times \mu) \tag{4.2}$$

Passive motion update:

$$Z_i(t+1) = Z_i(t) + \gamma \times \text{rand}(0,1) \times (U_b - L_b) \tag{4.3}$$

Direction calculation:

$$\vec{D_r} = \begin{cases} Z_j(t) - Z_i(t), & \text{if } f(Z_i) \geq f(Z_j) \\ Z_i(t) - Z_j(t), & \text{if } f(Z_i) < f(Z_j) \end{cases} \tag{4.4}$$

Position update:

$$Z_i(t+1) = Z_i(t) + \vec{Step} \tag{4.5}$$

with:

$$\vec{Step} = \text{rand}(0,1) \times \vec{D_r} \tag{4.6}$$

Time control function:

$$c(t) = \left| \left(1 - \frac{t}{I_{\max}}\right) \times (2 \times \text{rand}(0,1) - 1) \right| \tag{4.7}$$

Where:

- $\vec{tr}$ is the trend vector representing the direction of the ocean current.
- $Z_i(t)$ is the location of the i^{th} jellyfish at iteration t.
- U_b and L_b are the upper and lower bounds of the search space.
- μ is the mean location of all jellyfish.
- β and γ are distribution and motion coefficients, respectively.
- $I_{\max}$ is the maximum number of iterations.

The flowchart illustrating the overall mechanism of the Jellyfish Search optimizer is shown in Figure 4.5.

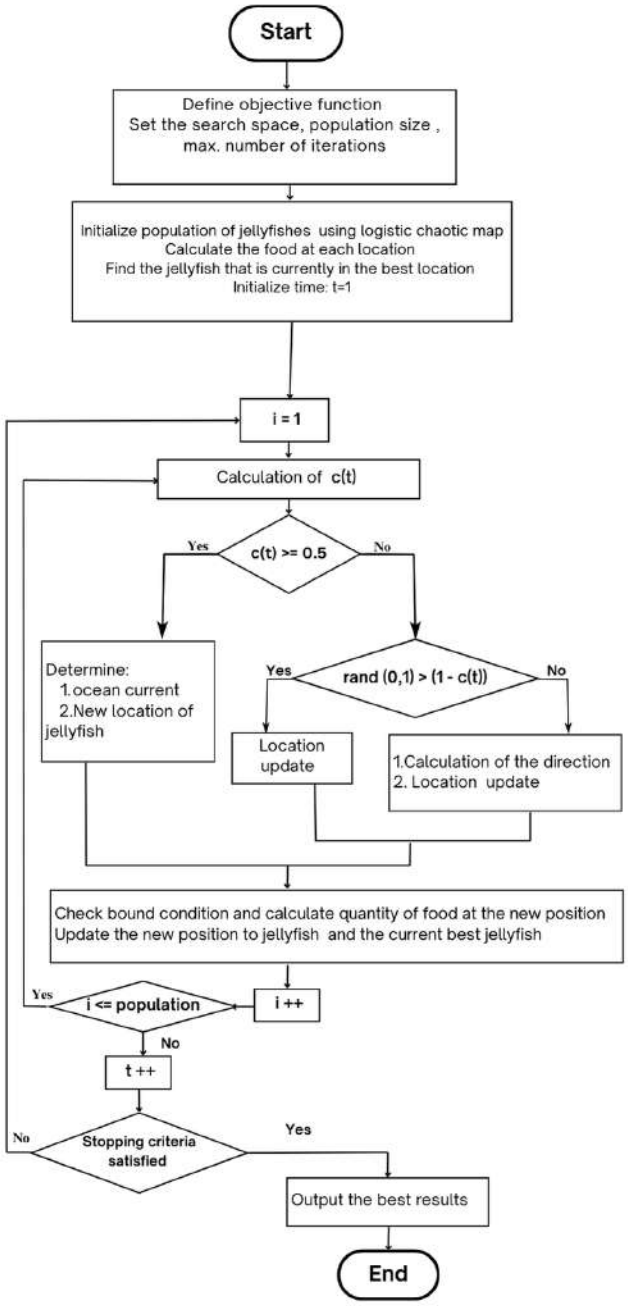


Figure 4.5: Flowchart of the Jellyfish Search optimizer mechanism.

4.2.4 Objective Function Design and Optimization Criteria

In optimization and mathematical modelling, the objective function is a fundamental concept. It is a function defined by a mathematical expression that quantitatively represents the goal of the optimization problem, which could involve either maximizing or minimizing a particular value such as a performance criterion. The design and formulation of an objective function are crucial and directly affect the behaviour and effectiveness of the optimization algorithm in action.

Initially, in its most basic form, an optimization problem is described by an objective function denoted $f(x)$, where x is a vector of decision variables. Decision variables are controllable inputs in a mathematical model that influences the outcome of an optimization problem, subject to a set of constraints. The aim is to determine the values of these variables that result in the best possible value of the objective function [52]. Some typical examples include:

- **Business:** determining the optimal quantity of products to manufacture or stock in order to maximize the overall profit while considering production costs and market demand.
- **Machine Learning:** tuning model parameters, such as learning rate and regularization coefficients, to reduce prediction error on unseen data.
- **Logistics:** selecting the most efficient transportation routes for delivering goods to minimize travel time, fuel consumption, and overall operational costs.
- **Control systems:** tuning the parameters of a PID (Proportional-Integral-Derivative) controller to achieve desired system performance in dynamic systems such as motor speed control, temperature regulation, or vehicle trajectory tracking.

Optimization problems can be divided into two groups, single-objective optimization and multi-objective optimization. They are classified based on the number of objectives they seek to improve.

Single-objective optimization deals with only one criterion, its focus is on optimizing a single criterion whether it is minimizing costs, maximizing efficiency, or finding the best possible value for a given function. The problem is typically formulated as follows.

$$\text{Minimize or Maximize } f(x), \text{ subject to constraints on } x.$$

However, many engineering challenges in the real world involve multiple competing goals, which are inherently multi-dimensional and involve several conflicting objectives that must

be balanced. Problems that have more than one objective are referred to as **Multi-Objective Optimization (MOO)** problems. The goal of this latter is to optimize two or more objective functions simultaneously instead of optimizing just a single factor, where the landscape of possibilities needs to be navigated, to achieve a well-rounded solution. A typical formulation for multi-objective problems is:

$$\text{Minimize or Maximize } (f_1(x), f_2(x), \ldots, f_n(x)), \text{ subject to constraints on } x.$$

Solving MOO problems involves two main strategies: the Pareto method and the scalarization method. These approaches differ in how they handle multiple objectives and prioritize solutions. The **Pareto method** focuses on identifying a set of solutions known as the **Pareto Optimal Front (POF)**, where no objective can be improved without worsening at least one other. Solutions are classified as either dominated or non-dominated based on whether they are inferior or superior across the objectives. The concept of Pareto optimality ensures that the final solutions represent the best possible compromises. Key terms in this method include the Anchor Point (the best individual values for each objective) and the Utopia Point (a theoretical ideal that combines the best of all objectives). Optimization algorithms iteratively search for non-dominated solutions to construct the POF, often using adaptive or evolving populations. On the other hand, the **scalarization method** transforms multiple objectives into a single aggregate objective by assigning weights to each one. The resulting weighted sum is then optimized using conventional single-objective techniques. The choice of weights significantly influences the solution by reflecting the relative importance of each objective. Three common strategies for setting these weights are equal weighting, Rank Order Centroid (ROC) weighting, and Rank Sum (RS) weighting. In addition, normalization is commonly applied to balance objectives with different scales or units [53].

Objective functions can be expressed in various forms, depending on the nature of the problem. Common examples include cost functions, error metrics, time-based criteria, and efficiency indices. In control applications, objective functions are often formulated based on performance metrics such as Integral of Absolute Error (IAE), Integral of Squared Error (ISE), Integral of Time-weighted Absolute Error (ITAE), and Integral of Time-weighted Squared Error (ITSE).

$$\textbf{Integral Absolute Error} \qquad \text{IAE} = \int |e(t)|\, dt \tag{4.8}$$

$$\textbf{Integral Square Error} \qquad \text{ISE} = \int e^2(t)\, dt \tag{4.9}$$

Integral Time-weighted Absolute Error $$\text{ITAE} = \int t|e(t)|\,dt \tag{4.10}$$

Integral Time-weighted Squared Error $$\text{ITSE} = \int te^2(t)\,dt \tag{4.11}$$

$e(t)$ is the error signal, defined as the difference between the desired (reference) value and the actual system output atntime t:

$$e(t) = r(t) - y(t) \tag{4.12}$$

Where:

- $r(t)$ is the reference (desired) value at time t.
- $y(t)$ is the actual system output at time t.

Overall, a careful formulation of the objective function is a key step in any optimization problem. It defines the aim of the problem, directs the search process, and ultimately determines the quality of the obtained solution. Whether optimizing a single objective or balancing multiple conflicting objectives, the objective function remains the cornerstone of the optimization framework.

4.3 Multi-Objective Optimization in PID Tuning through Metaheuristics

PID controllers constitute the foundation of most control systems, offering a simple yet powerful framework for system regulation. The transfer function describing a PID controller is given in Equation 4.13. In this expression, K_P, K_I, and K_D represent the proportional, integral, and derivative gains, respectively [54].

$$G(s) = K_P + \frac{K_I}{s} + K_D \cdot s \tag{4.13}$$

The performance of a PID controller is highly dependent on the appropriate selection of its gain parameters. However, conventional tuning methods often yield suboptimal results, particularly regarding disturbance rejection and overall robustness. This limitation underscores the growing need for advanced optimization strategies, such as metaheuristic algorithms, to enhance controller performance in complex and dynamic environments.

4.3.1 Genetic Algorithm-Based PID Tuning

Genetic Algorithms are particularly suitable for solving complex optimization problems such as PID tuning. In this work, a GA is employed to optimally select the proportional (K_P), integral (K_I), and derivative (K_D) gains of the PID controller. The general formulation involves the following steps:

1. **Initialization:** generate an initial population of candidate solutions, where each individual represents a set of PID gains (K_P, K_I, K_D).
2. **Fitness evaluation:** evaluate the performance of each individual using the objective function.
3. **Selection:** select parent individuals for reproduction based on their fitness values. Methods such as roulette wheel selection or tournament selection can be applied.
4. **Crossover:** combine selected parents to produce offspring using crossover operations, which exchange segments of the PID gain parameters between parents.
5. **Mutation:** introduce small random changes to some of the offspring to maintain population diversity and avoid premature convergence.
6. **Replacement:** form a new population by selecting the best individuals from both parents and offspring.
7. **Termination:** repeat steps 2 to 6 until the stop criterion is met.

The corresponding flowchart is illustrated in Figure 4.6, adapted from [55]. Initially, the Integral of Squared Error (ISE) is employed as the objective function, denoted by $\boldsymbol{f}$, in conjunction with the Genetic Algorithm to optimize controller parameters.

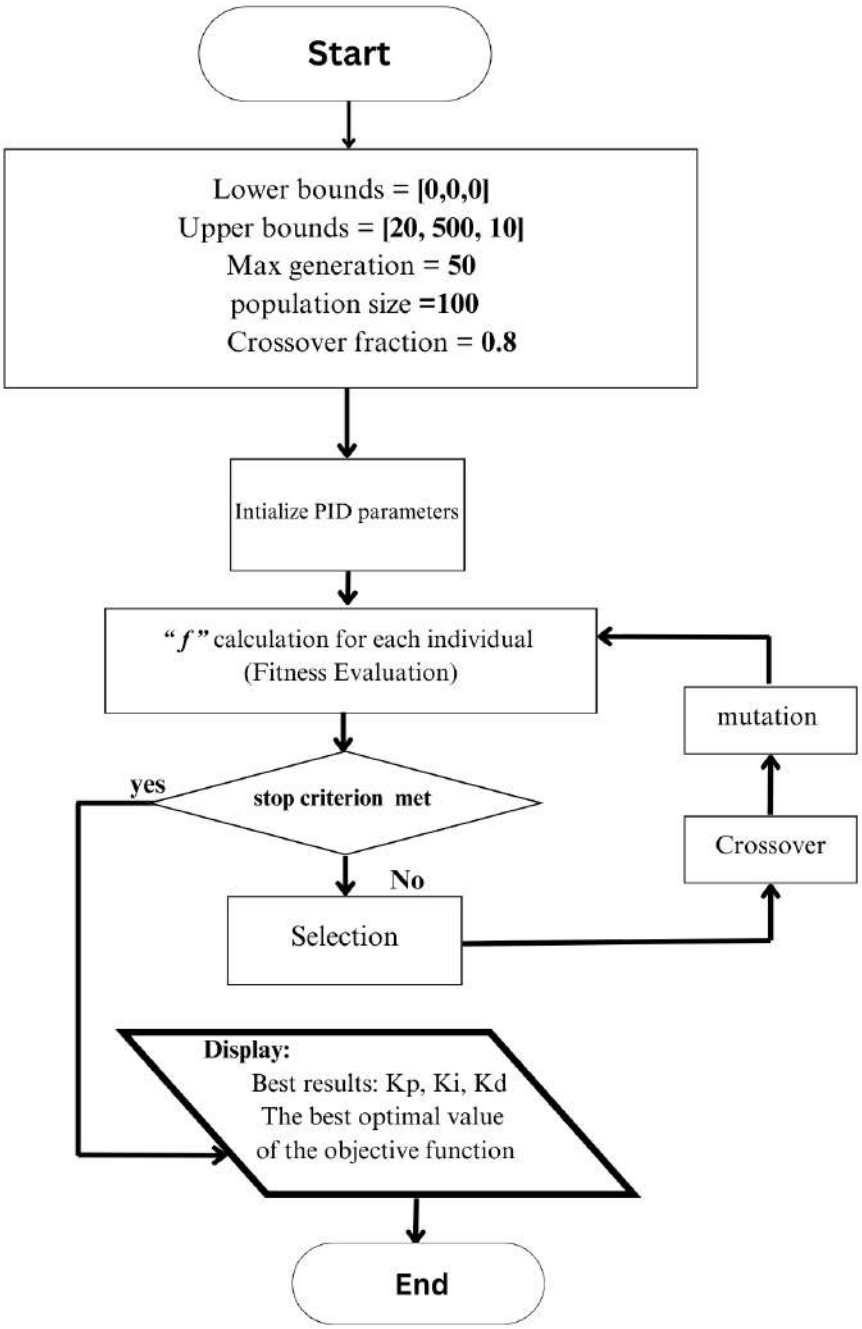


Figure 4.6: Flowchart of the GA-based PID tuning process.

Table 4.1 presents the statistical indices of the algorithm's performance during the tuning process. Over **50** generations, the algorithm shows a consistent decrease in the mean objec-

tive value, indicating improved population convergence. The best objective value stabilizes around **0.001777** by generation **23**, with the resulting PID parameters: $K_p = 19.9595, K_i = 499.9999$,and $K_d = 0.1158$, suggesting early convergence. Stall generations begin to accumulate more frequently after this point, confirming a plateau in improvement. These results validate the GA's effectiveness in minimizing the ISE. However, further optimization methods may be required to enhance performance beyond this local optimum.

Table 4.1: Genetic Algorithm optimization results using ISE

Generation	**Func-count**	**Best f(x)**	**Mean f(x)**	**Stall Generations**
1	200	0.001784	1.422	0
2	295	0.001784	0.01696	0
3	390	0.001784	0.005337	1
4	485	0.001780	0.004413	0
5	580	0.001780	0.002221	1
6	675	0.001780	0.002059	0
7	770	0.001780	0.001975	1
8	865	0.001780	0.001789	2
9	960	0.001779	0.001784	0
10	1055	0.001779	0.001782	0
11	1150	0.001779	0.001781	1
12	1245	0.001779	0.001780	2
13	1340	0.001778	0.001779	0
14	1435	0.001778	0.001779	0
15	1530	0.001778	0.001779	0
16	1625	0.001778	0.001779	1
17	1720	0.001778	0.001779	2
18	1815	0.001778	0.001778	0
19	1910	0.001778	0.001778	0
20	2005	0.001778	0.001779	1
21	2100	0.001778	0.001779	2
22	2195	0.001778	0.001778	3
23	2290	0.001777	0.001778	0
24	2385	0.001777	0.001778	0
25	2480	0.001777	0.001778	1

Generation	Func-count	Best f(x)	Mean f(x)	Stall Generations
26	2575	0.001777	0.001778	0
27	2670	0.001777	0.001778	1
28	2765	0.001777	0.001778	2
29	2860	0.001777	0.001778	0
30	2955	0.001777	0.001778	0
31	3050	0.001777	0.001778	1
32	3145	0.001777	0.001778	2
33	3240	0.001777	0.001778	3
34	3335	0.001777	0.001778	0
35	3430	0.001777	0.001778	0
36	3525	0.001777	0.001778	0
37	3620	0.001777	0.001778	1
38	3715	0.001777	0.001778	2
39	3810	0.001777	0.001778	3
40	3905	0.001777	0.001777	0
41	4000	0.001777	0.001778	1
42	4095	0.001777	0.001777	0
43	4190	0.001777	0.001777	0
44	4285	0.001777	0.001777	1
45	4380	0.001777	0.001777	2
46	4475	0.001777	0.001777	3
47	4570	0.001777	0.001777	0
48	4665	0.001777	0.001777	1
49	4760	0.001777	0.001777	2
50	4855	0.001777	0.001777	0

4.3.2 Jellyfish Algorithm-Based PID Tuning

The JS algorithm is adopted to tune the PID controller gains (K_P, K_I, K_D) by minimizing a multi-objective performance index. The performance of the proposed Jellyfish Search-based PID tuning will be compared with that of the Genetic Algorithm in the following section. The formulation involves:

1. **Initialization:** initialize the population of candidate solutions, where each individual

represents a set of PID gains (K_P, K_I, K_D). A logistic chaotic map is used during initialization to enhance population diversity.

2. **Objective function evaluation:** calculate the value of the objective function for each individual, which combines the ITAE and ISE performance indices.

3. **Movement decision:** at each iteration, calculate the time control function $c(t)$. Based on its value, jellyfish either follow ocean currents or move within the swarm.

4. **Location update:**

   - If $c(t) \geq 0.5$, individuals update their positions based on ocean current influence.
   - If $c(t) < 0.5$, a random check determines whether an individual moves randomly within the swarm or toward better positions.

5. **Bound and fitness check:** after each movement, positions are checked against boundary constraints, and new objective function values are computed.

6. **Update best solution:** update the global best position Z^* based on the best objective function value obtained.

7. **Termination:** Repeat the above steps until the stopping criterion is met. Finally, report the optimal PID gains and the corresponding best objective function value.

The complete process is summarized in the flowchart shown in Figure 4.7.

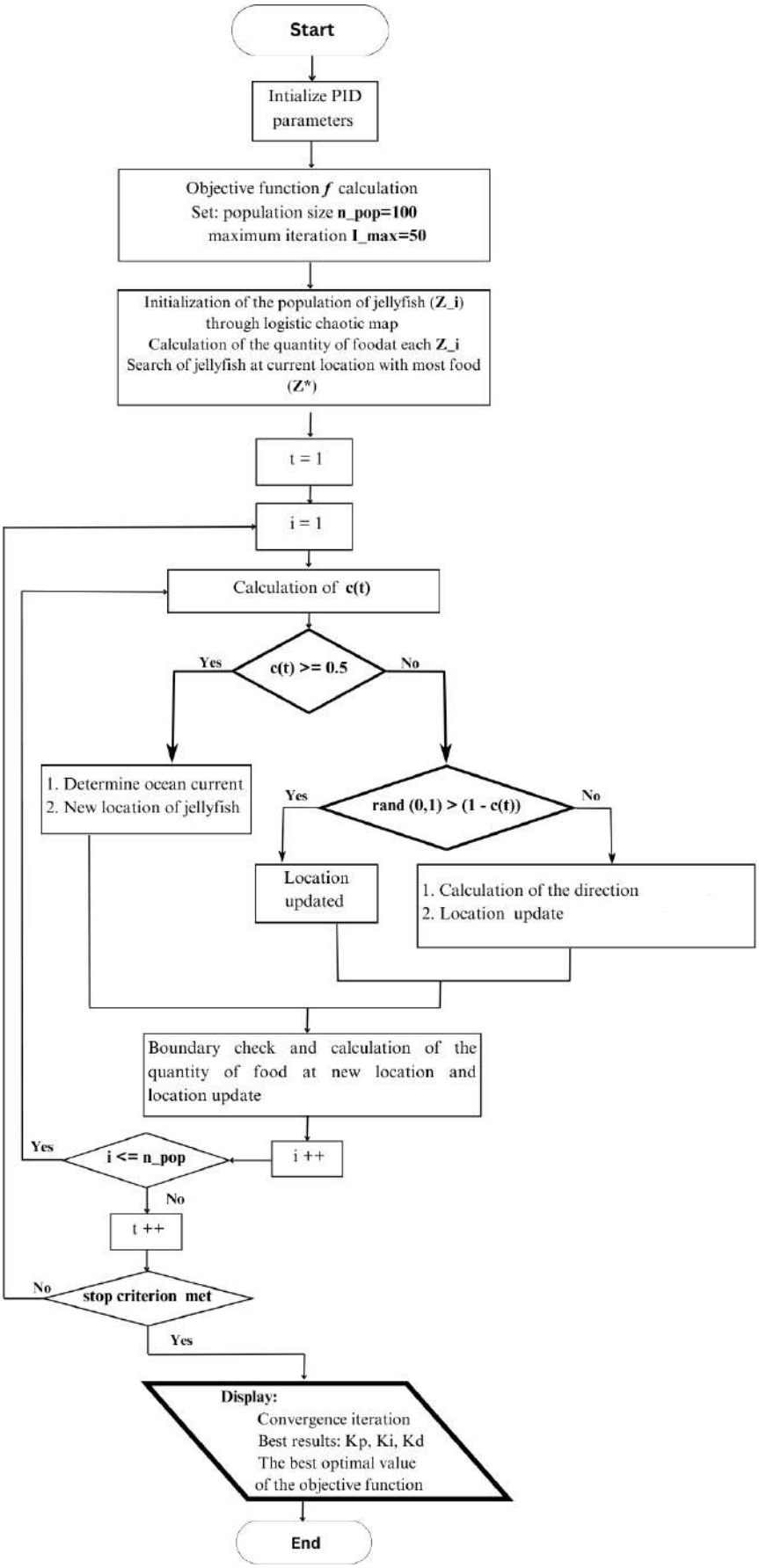


Figure 4.7: Flowchart of the JS algorithm-based PID tuning process.

The pseudo-code of the proposed JS algorithm-based PID tuning method is provided in Appendix B.

4.3.3 Results and Discussion

In order to further enhance the performance criteria, including minimal overshoot, rapid settling time, and low steady-state error, a cost function is formulated based on the performance metrics ITAE and ISE combined. For instance, ITAE penalizes sustained errors more than transient ones, whereas ISE emphasizes large errors due to the squaring operation. These indices are combined through scalarization, forming a single aggregated fitness function:

$$f(K_p, K_i, K_d) = w_1 IAE + w_2 ISE \tag{4.14}$$

where w_1 and w_2 are predefined weights representing the priority assigned to each performance criterion. In this work, $w_1 = 0.2$ and $w_2 = 0.8$.

The designed PID longitudinal control system is evaluated on a 2 km road segment. Initially, the PID gains are determined using the conventional trial-and-error method. The performance of this approach is then compared to the optimized PID gains via the Jellyfish Search optimizer and the Genetic Algorithm through MOO. The overall control system is depicted in Figure 4.8. The speed input selected for this scenario is the optimized sinusoidal profile (Figure 4.9), characterized by the following parameters: a speed limit of $V_m = 28.8\,\text{km/h}$ (i.e., $8\,\text{m/s}$), suitable for urban environments as per the ISO 2631 standard, and an acceleration limit of $A_m = 0.4\,\text{m/s}^2$, chosen based on the ISO 22737 standard. This speed input is designed to reflect urban travel conditions while maintaining safety, smooth driving, and passenger comfort.

The initialization parameters for both the Genetic Algorithm and the Jellyfish Search algorithm are provided in Tables 4.2 and 4.3, respectively. The corresponding GA results are illustrated in Figure 4.10, while the statistical indices summarizing the algorithm's performance are presented in Table 4.4.

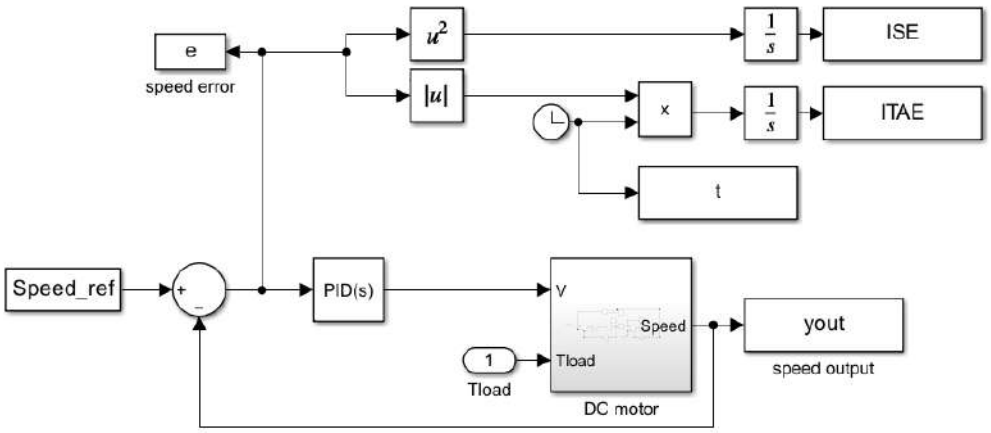


Figure 4.8: Optimized PID longitudinal control.

Table 4.2: GA parameters

Parameter	**Value**
Number of variables	3
Lower bounds	[0, 0, 0]
Upper bounds	[20, 500, 10]
Max generations	50
Population size	100
Crossover fraction	0.8
Mutation	Adaptive feasible mutation

Table 4.3: JS parameters

Parameter	**Value**
Dimensions	3
Lower bounds	[0, 0, 0]
Upper bounds	[20, 500, 10]
Max iterations	50
Population size	100
Stopping condition	Convergence check
Initialization method	Logistic map

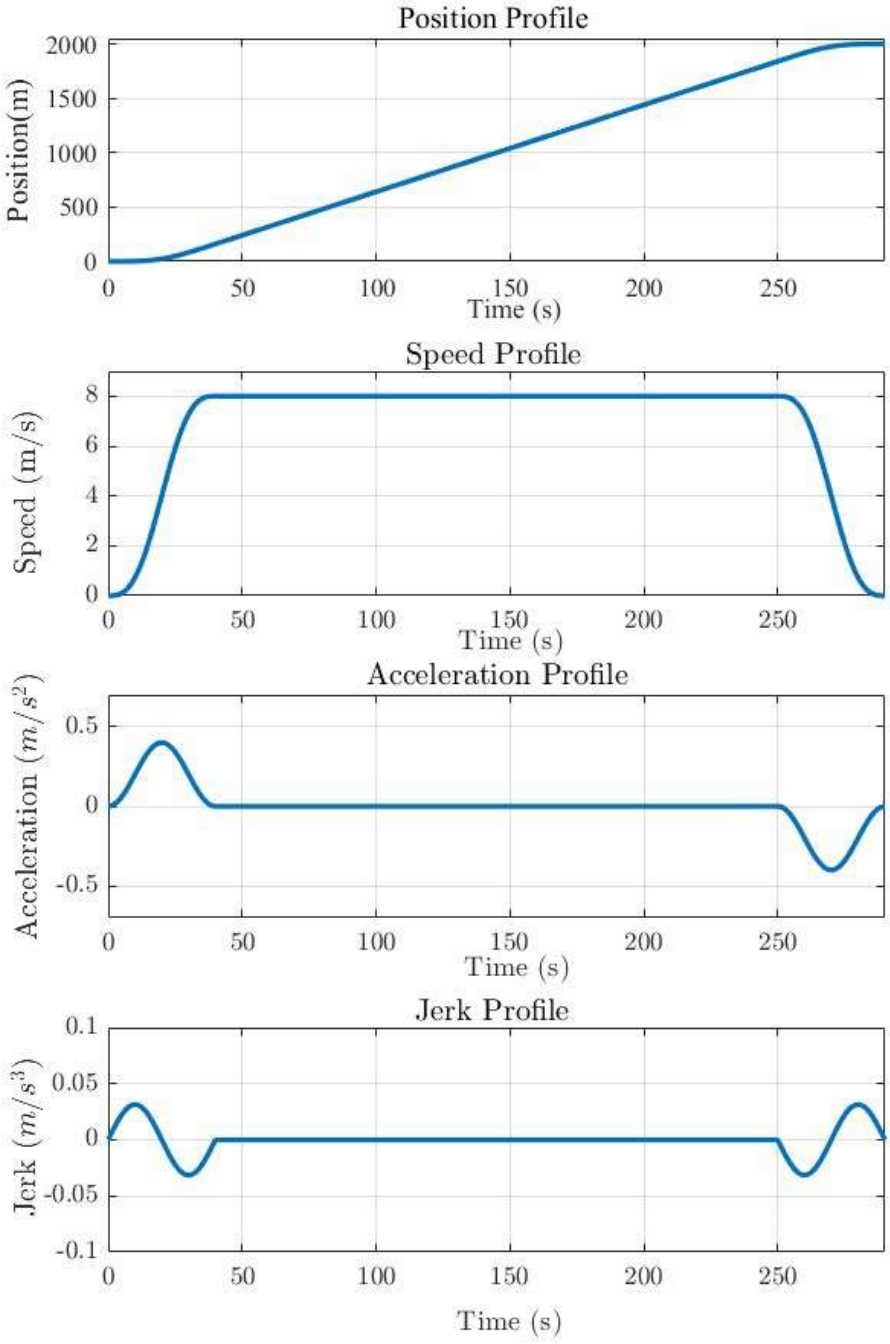


Figure 4.9: Optimized sinusoidal speed profile.

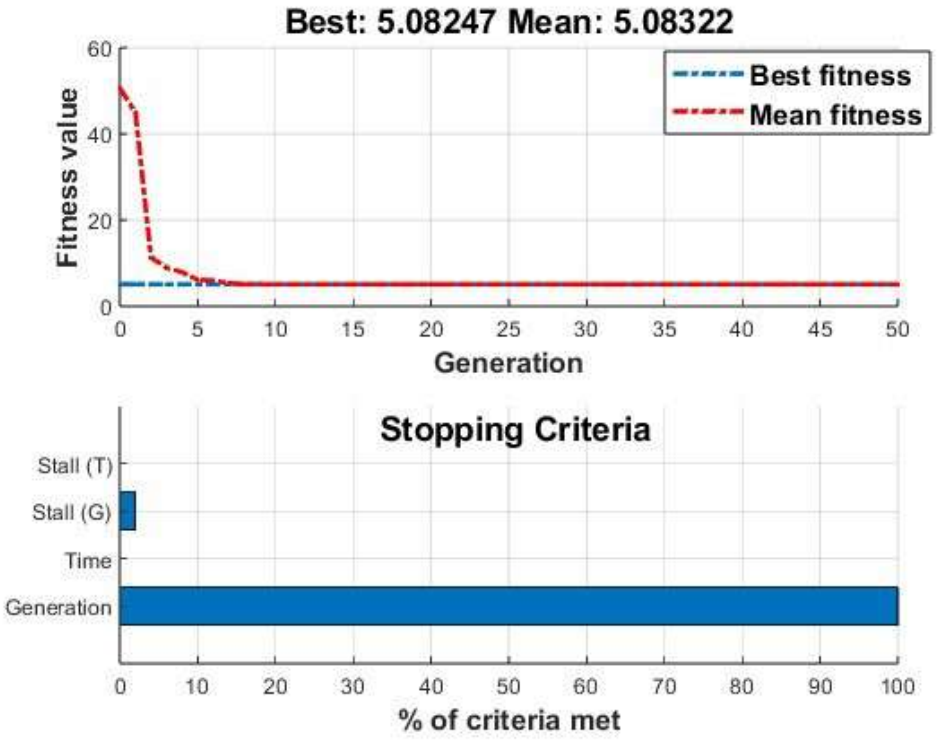


Figure 4.10: Genetic Algorithm optimization results through MOO.

Table 4.4: Genetic Algorithm optimization results through MOO.

Generation	**Func-count**	**Best f(x)**	**Mean f(x)**	**Stall Generations**
1	200	5.103	44.930	0
2	295	5.094	11.300	0
3	390	5.092	8.905	0
4	485	5.092	7.892	0
5	580	5.092	6.107	1
6	675	5.087	5.937	0
7	770	5.087	5.328	0
8	865	5.087	5.097	0
9	960	5.087	5.095	0
10	1055	5.087	5.094	0
11	1150	5.087	5.093	0
12	1245	5.087	5.093	1
13	1340	5.084	5.092	0
14	1435	5.084	5.091	1

Generation	**Func-count**	**Best f(x)**	**Mean f(x)**	**Stall Generations**
15	1530	5.084	5.087	0
16	1625	5.084	5.087	1
17	1720	5.084	5.086	2
18	1815	5.083	5.086	0
19	1910	5.083	5.086	0
20	2005	5.083	5.086	1
21	2100	5.083	5.085	0
22	2195	5.083	5.084	1
23	2290	5.083	5.084	2
24	2385	5.083	5.083	3
25	2480	5.083	5.083	0
26	2575	5.083	5.083	0
27	2670	5.083	5.083	0
28	2765	5.083	5.084	1
29	2860	5.083	5.083	2
30	2955	5.083	5.083	3
31	3050	5.083	5.083	4
32	3145	5.083	5.083	0
33	3240	5.083	5.083	0
34	3335	5.083	5.083	1
35	3430	5.083	5.083	2
36	3525	5.083	5.083	3
37	3620	5.083	5.083	0
38	3715	5.083	5.083	1
39	3810	5.082	5.083	0
40	3905	5.082	5.083	1
41	4000	5.082	5.083	2
42	4095	5.082	5.083	0
43	4190	5.082	5.083	0
44	4285	5.082	5.083	0
45	4380	5.082	5.083	0
46	4475	5.082	5.083	0
47	4570	5.082	5.083	0

Generation	Func-count	Best f(x)	Mean f(x)	Stall Generations
48	4665	5.082	5.083	0
49	4760	5.082	5.083	0
50	4855	5.082	5.083	1

The Jellyfish Search (JS) algorithm demonstrates strong effectiveness in PID gain optimization, achieving convergence by the 7^{th} iteration with a best cost function value of 5.083. Table 4.5 presents a comparative analysis of the PID gains obtained using the JS algorithm, Genetic Algorithm, and conventional tuning.

Table 4.5: Comparative study of PID gains

Technique	K_p	K_i	K_d	Objective Function
Conventional PID	19.000	100.000	0.500	44.7791
JS optimizer	4.3455	499.998	9.7863	5.083
GA	0.00087	499.999	4.2312	5.082

In terms of optimizing the objective function, both metaheuristic algorithms JS and GA significantly enhance PID tuning compared to conventional methods. While their final performances are quite similar, the JS algorithm offers a slightly more balanced distribution of gains. Additionally, the JS optimizer reached convergence as early as the 7^{th} iteration, whereas the GA required around 39 iterations to converge. This demonstrates the JS algorithm's advantage in computational efficiency, achieving high-quality optimization with reduced time cost.

The speed outputs and their corresponding error plots of the Jellyfish algorithm, Genetic algorithm, and conventional PID are depicted in Figures 4.11 and 4.12, respectively. All methods exhibit good tracking performance. However, the zoomed-in inset highlights an overshoot in the response of the conventional PID controller, while the GA and JS optimizer generate smoother outputs with less deviation, indicating improved control performance. Figure 4.12 further shows that the conventional PID struggles during the transient phase, presenting larger errors and reduced steady-state accuracy. In contrast, the GA and JS methods demonstrate their effectiveness by offering more stable and accurate responses. Table 4.6 highlights the results discussed earlier, where both the JS optimizer and GA significantly outperform the conventional PID tuning approach. Notably, both metaheuristic methods achieve zero overshoot, shorter settling times, and markedly reduced Steady-State Errors (SSE). Among them, the JS optimizer exhibits a slight edge in terms of rapidity, indicating

faster system responsiveness. These findings suggest that nature-inspired optimization algorithms, such as JS and GA, offer substantial improvements in control precision and stability. The enhanced performance metrics especially in minimizing transient errors and achieving faster convergence, make these approaches more suitable for dynamic systems requiring high reliability and efficiency.

Table 4.6: Performance criteria comparison

Technique	**Overshoot (%)**	**Settling Time (s)**	**Rise Time (s)**	**SSE (m/s)**
Conventional PID	0.52	41.64	34.236	10^{-4}
JS	0.00	38.93	34.153	10^{-5}
GA	0.00	40.41	34.156	10^{-5}

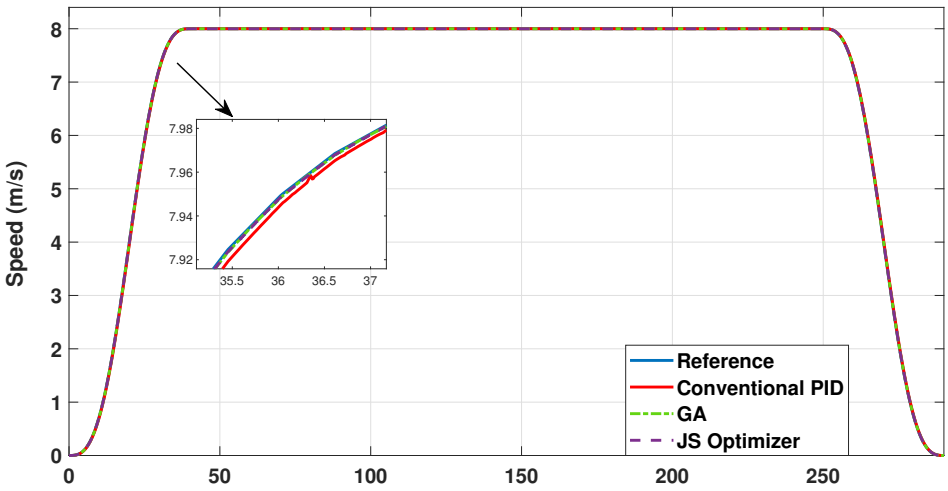


Figure 4.11: Speed output response

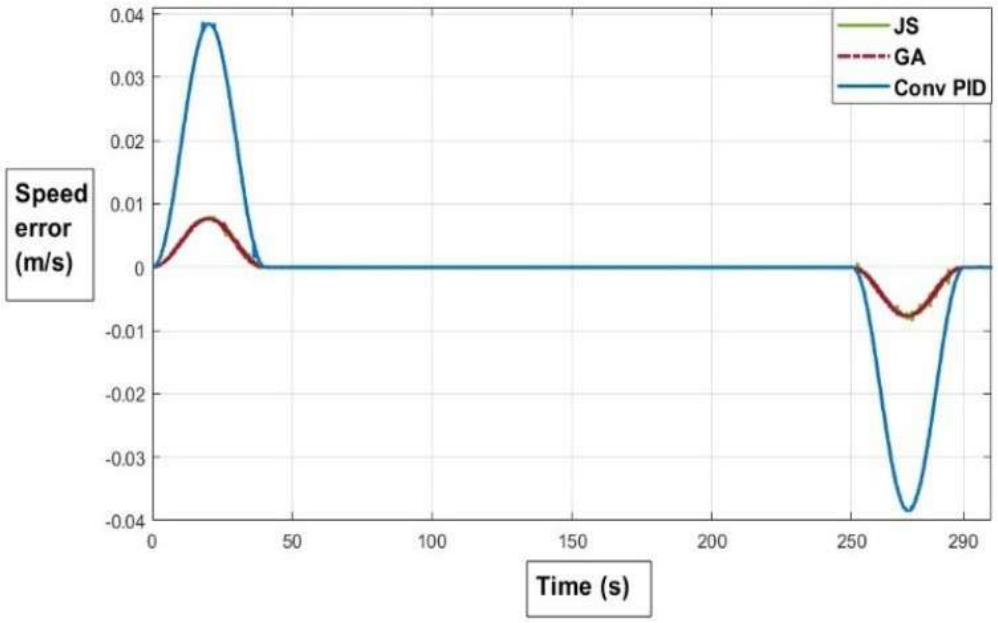


Figure 4.12: Error response

4.4 Conclusion

This chapter has provided a comprehensive investigation of metaheuristic algorithms, focusing on their application to PID tuning. Through detailed analysis and comparison, both the Genetic Algorithm and the Jellyfish Search optimizer demonstrated significant improvements over conventional PID tuning methods, particularly in terms of accuracy, convergence speed, and tracking performance. Among them, the JS algorithm achieved slightly better dynamic response and faster convergence, making it well-suited for real-time control scenarios. However, despite their effectiveness under ideal conditions, these optimizers rely on static parameter settings and may struggle with dynamic uncertainties. These limitations underscore the need for adaptive and robust control strategies, which will be addressed in the following chapters.

Chapter 5

Adaptive Control Theory and Implementation

5.1 Introduction

This chapter discusses adaptive control, emphasizing its importance in systems where traditional control laws may be insufficient. Traditional control laws are highly effective for linear systems with constant parameters. However, when these parameters become variable, the effectiveness of these control laws diminishes.
The development of traditional control algorithms assumes a fixed (invariant) model, which means that the controller's model is also fixed. Consequently, as long as the system does not experience variations, both the model and the regulator remain valid. However, there are cases where the system undergoes variations due to parametric variations of the model, inaccuracies in modelling, and changes in the characteristics of the input signal. Under the influence of these factors, the controller's model is no longer valid, which has led to the search for other controllers that can adapt to the variations of a system.

5.2 Adaptive Control

The adaptive approach has been explored towards the late 1940s and continues to be an area of intense activity. The term adaptive control referred primarily to a regulator's ability to adjust its parameters to both dynamic and static behaviour of a process.
It is a control method that allows the regulator to self-adjust to changes in the process through:

- The introduction of control actions;
- Changes in the system structure;
- Changes in the control algorithm.

The goal of automatically adjusting the regulator's parameters in real time is to achieve satisfactory performance when the process parameters vary over time [56].
The general block diagram for the adaptive control is shown in Figure 5.1. The control loop is divided into two loops, a main (internal) loop that includes the regulator block (controller), as well as the process block (plant), and a secondary (external) loop that responds to variations in the process parameters through an adaptation mechanism that makes the system adaptive [57].

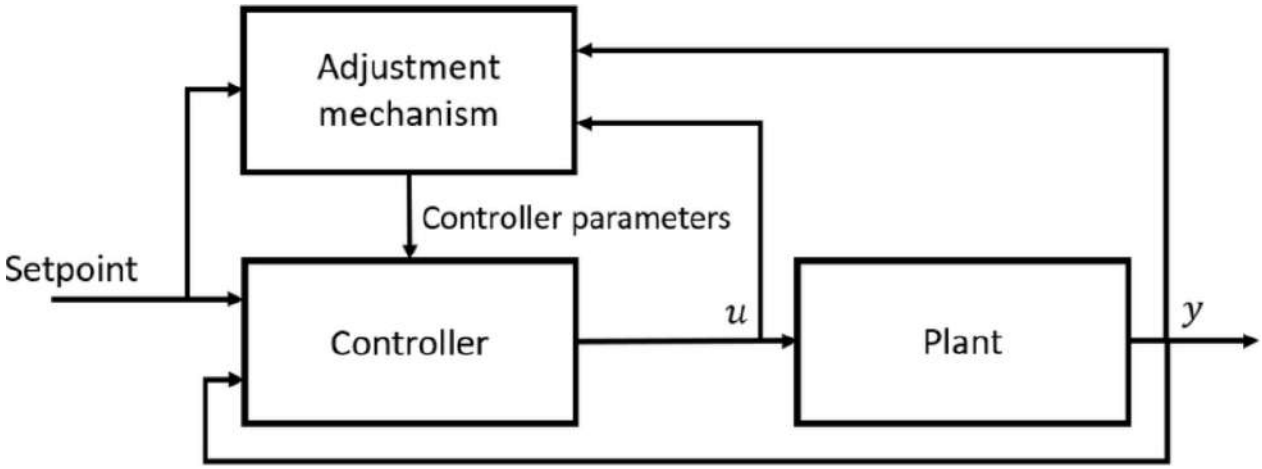


Figure 5.1: Adaptive Control block diagram.

Classification of Adaptive Control Systems

The structure used to update regulator parameters online based on the variations of the process to be controlled differs from one control system to another. Three main classes of adaptive control are distinguished [58]:

- Gain scheduling;
- Self-tuning controllers;
- Model Reference Adaptive control.

5.2.1 Gain Scheduling

This method assumes that the nonlinearities are known, as there is no correction to compensate for incorrect programming (open-loop operation). However, it has the advantage of quickly adjusting the controller parameters during rapid changes in the process dynamics. The main drawbacks of this method are:

- The effects of disturbances due to inaccuracies are not compensated in open-loop;
- The cost of the regulator is high due to the complexity of modeling the relationships between disturbance measurements and process dynamics.

5.2.2 Self-Tuning Controllers

This method's principle is based on online parameter estimation, which means that the system parameters are assumed unknown and not directly used. At each iteration, a new model

is obtained, and the regulator ensures the satisfaction of nominal specifications. Identification and control design are treated independently based on the principles of separation and certainty equivalence. Only the control criterion is involved in the synthesis of the regulator, without considering the performance of the estimation, and vice versa. Furthermore, the principle of certainty equivalence reflects the fact that any error in identification is not taken into account in the design of the control law.

5.2.3 Model Reference Adaptive Control

Model Reference Adaptive Control (MRAC) first appeared in 1958 in an article published by M.P. Whitaker, J. Yamron, and A. Keser. The basic idea is to use a reference model, which takes an input reference and provides the desired output. This model indicates how the process should behave in response to a control signal [59].
Its basic principles are as follows:

- The system output must follow the reference model output;
- The tracking error is given by: $e(t) = y_m(t) - y(t)$;
- The closed-loop system is based on a feedback control law;
- The control system includes a controller (internal loop) and an adjustment mechanism (external loop).

$y(t)$ and $y_m(t)$ are the process and model reference outputs, respectively.
The basic structure of the reference model control systems is represented by the following diagram 5.2 [60]. The structure consists of two loops:

- The correction loop;
- The adjustment loop.

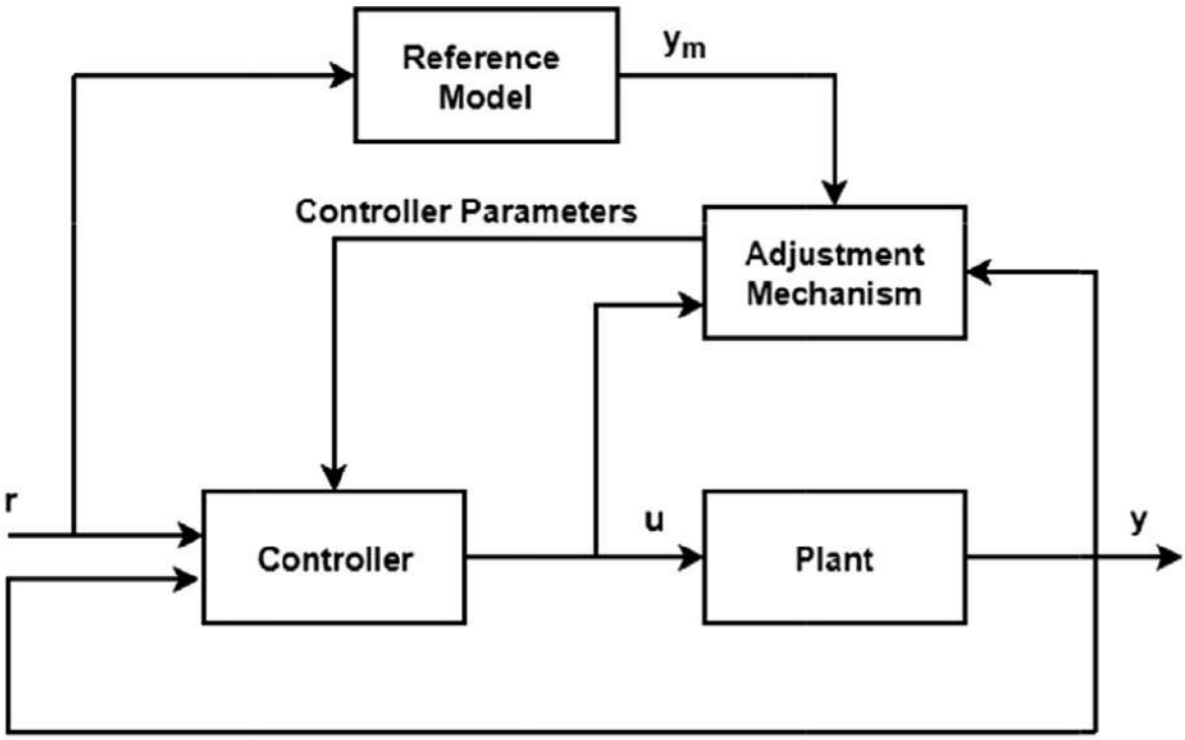


Figure 5.2: MRAC block diagram.

The correction loop: It includes the system to be controlled and the regulator, whose parameters are automatically adjusted through an adjustment mechanism.
The adjustment loop: It includes the adjustment mechanism, which takes as input the tracking error between the reference model output and the process output, with the goal of designing the adjustment law for the controller parameters.
This method offers the following advantages:

- Stability and control performance;
- Simplicity of implementation;
- Reliability, which explains its widespread use in various application areas.

5.3 Synthesis Methods for MRAC Algorithms

Concept of Performance Index
It is a criterion on which the development of the adjustment mechanism is based. Its main objective is to reduce/eliminate the error $e(t)$ between the output of the reference model and the output signal of the real system. The performance criterion depends on the quality of

the reference model and the self-tuning criterion. The latter helps optimize the adjustment of the internal loop system relative to the reference model, thus determining the quality of the control. The self-tuning criterion is defined by a function J that depends on the error $e(t)$.

$$J(e) = J(y_m(t) - y(t)) \tag{5.1}$$

The most commonly used performance criteria are:

- Linear criterion: $J_1 = \int_0^\infty |e(t)|\, dt$;
- Quadratic criterion: $J_2 = \int_0^\infty e(t)^2\, dt$;
- Weighted linear criterion: $J_3 = \int_0^\infty t|e(t)|\, dt$.

The primary synthesis methods include:

- Gradient method;
- Direct Lyapunov method;
- Popov's Hyperstability method.

5.3.1 The Gradient Method

This method employs the minimization of the performance index based on the gradient of the parameters. It does not require global stability, but rather a balance between stability and response speed.
Let $J(e)$ be the given quality criterion:

$$grad\ J(e) = \left\{ \frac{\partial J}{\partial b_1}, \frac{\partial J}{\partial b_2}, \ldots, \frac{\partial J}{\partial b_n} \right\} \tag{5.2}$$

This vector represents the gradient of the criterion J with respect to the vector B_r of the adjustable parameters $b_1, \ldots, b_n$.
The gradient method adjusts the parameters so that their rate of change is proportional to the elements of the gradient of J, i.e.:

$$b_i'(t) = -\gamma_i \frac{\partial J}{\partial b_i} \tag{5.3}$$

Once the function J is selected, the vector B_r can be extracted, allowing the determination of the adaptive control law that depends on the adjustable parameters b_i [61].

5.3.2 Lyapunov's Direct Method

The direct Lyapunov method is commonly applied in the field of control engineering. This approach relies on the concept of energy. It determines system stability by constructing a scalar function of the system's states and analysing its time-dependent variation. The method follows this logic:
If the process error is bounded, then the model reference control system is stable.
The structure of the adjustment loop is designed according to the Lyapunov stability condition. The control system is described in Cauchy form (Equation (5.4)) [59].

$$\dot{y} = F_i(y_1, y_2, \ldots, y_n, t) \tag{5.4}$$

A Lyapunov function is selected in the following form:

$$V = V(y_1, y_2, \ldots, y_n) \tag{5.5}$$

V is a defined function with a specific sign, and its derivative with respect to time must have the opposite sign to ensure asymptotic stability. Thus, this function is crucial for studying the stability of a system. However, there is no general law to obtain the precise formula for any given system.
Consider the equation of the main system (5.6) and that of the reference model (5.7):

$$y^{(n)} + \sum_{i=0}^{n-1} a_i(t) y^{(i)} = K_s(t) y_c(t) \tag{5.6}$$

$$y_m^{(N)} + \sum_{j=0}^{N-1} b_j y_m^{(j)} = K_m y_c(t) \tag{5.7}$$

where $a_i(t)$ and $K_s(t)$ are time-varying coefficients, while b_j and K_m are the constant coefficients of the model.
As defined by the error:

$$e^{(i)} = y_m^{(i)} - y^{(i)} \tag{5.8}$$

By substituting the two previous Equations 5.6 and 5.7, as well as $\sum_{j=0}^{N-1} b_j y_m^{(j)}$ into both sides, the following result is obtained:

$$y_m^{(N)} - y^{(n)} + \sum_{j=0}^{N-1} b_j y_m^{(j)} - \sum_{j=0}^{N-1} b_j y^{(j)} = [K_m - K_s(t)] y_c(t) + \sum_{i=0}^{n-1} a_i(t) y^{(i)} - \sum_{j=0}^{N-1} b_j y^{(j)} \tag{5.9}$$

Let $N = n$ and $i = j$:

$$e^{(n)} + \sum_{i=0}^{n-1} b_i e^{(i)} = \sum_{i=0}^{n-1} [a_i(t) - b_i] y^{(i)} + [K_m - K_s(t)] y_c \tag{5.10}$$

Equation (5.10) is used for synthesizing the law for changing the adjustable parameters $\gamma_i(t)$ of the main loop, the adjustment mechanism is designed to modify the coefficients of the main loop so that the error is zero, i.e.:

$$\lim_{t\to+\infty} e(t) = 0 \Rightarrow \lim_{t\to+\infty} K_s(t) = K_m \; and \lim_{t\to+\infty} a_i(t) = b_i \tag{5.11}$$

The adjustment of the coefficients takes place over a time interval $t \in [t_k, t_k + T_k]$.
Let:

$$k_s(t) = k_a(t)k_0(t) = k_a(t)k_0(T_k) \tag{5.12}$$

$$a_i(t) = a_i(T_k) + k_0(T_k)c_i(t) \tag{5.13}$$

Thus, Equation (5.10) becomes:
$e^{(n)} + \sum_{i=0}^{n-1} b_i e^{(i)} = [k_m - k_a(t)k_0(T_k)]y_c(t) - \{b_{n-1} - [a_{n-1}(T_k) + k_0(T_k)c_{n-1}]\}y^{(n-1)}$
$- \ldots - \{b_0 - [a_0(T_k) + k_0(T_k)c_0]\}y$
Defining the adjustable parameters denoted $\gamma_i(t)$ as follows:

$$\gamma_{n+1} = k_m - k_a(t)k_0(T_k) \tag{5.14}$$

$$\gamma_n = b_{n-1} - [a_{n-1}(T_k) + k_0(T_k)c_{n-1}] \tag{5.15}$$

$$\ldots \tag{5.16}$$

$$\gamma_1 = b_0 - [a_0(T_k) + k_0(T_k)c_0] \tag{5.17}$$

The result is:

$$e^{(n)} + \sum_{i=0}^{n-1} b_i e^{(i)} = \gamma_{n+1}y_c(t) - \gamma_n y^{(n-1)} - \gamma_{n-1}y^{(n-2)} - \ldots - \gamma_1 y \tag{5.18}$$

If the dynamic error is written as $e^{(i)} = x_{i+1}$, the following system of equations is obtained:

$$\dot{x}_1 = x_2, \tag{5.19}$$

$$\dot{x}_2 = x_3, \tag{5.20}$$

$$\ldots \tag{5.21}$$

$$\dot{x}_{n-1} = x_n, \tag{5.22}$$

$$\dot{x}_n = -b_0x_1 - b_1x_2 - \ldots - b_{n-1}x_n + u_0(t); \tag{5.23}$$

It follows:

$$u_0(t) = \gamma_{n+1}y_c(t) - \gamma_n y^{(n-1)} - \gamma_{n-1}y^{(n-2)} - \ldots - \gamma_1 y \tag{5.24}$$

The previous system of equations can be written in a condensed form:

$$\dot{x} = Ax + u \tag{5.25}$$

where :

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & 0 \\ \vdots & \dots & \dots & \dots & 1 \\ -b_0 & -b_1 & -b_2 & \dots & -b_{n-1} \end{bmatrix} \tag{5.26}$$

$$u = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ u_0 \end{bmatrix} \tag{5.27}$$

The positive-definite Lyapunov function in Equation (5.28) is chosen based on the concept of energy that satisfies the negative (semi) definiteness of its time derivative [62].

$$V = x^T P x + \sum_{i=1}^{n+1} \lambda_i \gamma_i^2 \tag{5.28}$$

λ_i are positive weighting constants and P is a positive definite symmetric matrix.

$$P = \begin{bmatrix} p_{11} & p_{12} & \dots & p_{1n} \\ p_{21} & p_{22} & \dots & p_{2n} \\ \dots & \dots & \dots & \dots \\ p_{n1} & p_{n2} & \dots & p_{nn} \end{bmatrix} \tag{5.29}$$

Assuming that the matrix A is regular (invertible), the Lyapunov function (5.28) is differentiated to obtain:

$$\dot{V} = \dot{x}^T P x + x^T P \dot{x} + \sum_{i=1}^{n+1} 2\lambda_i \gamma_i \dot{\gamma}_i = (Ax + u)^T P x + x^T P (Ax + u) + \sum_{i=1}^{n+1} 2\lambda_i \gamma_i \dot{\gamma}_i \tag{5.30}$$

Taking into account the matrix product transposition rule: $(AB)^T = B^T A^T$; $(ABCD)^T = D^T C^T B^T A^T$, Equation (5.30) can be expressed as:

$$\dot{V} = x^T (A^T P + P A) x + 2 x^T P u + \sum_{i=1}^{n+1} 2\lambda_i \gamma_i \dot{\gamma}_i \tag{5.31}$$

Let $A^T P + P A = -Q$, where Q is a diagonal matrix with elements $q_{ij} > 0$ for $i = j$ and $q_{ij} = 0$ for $i \neq j$.

$$Q = \begin{bmatrix} q_{11} & 0 & \dots & 0 \\ 0 & q_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & q_{nn} \end{bmatrix} \tag{5.32}$$

The determination of the law for changing the adjustable parameters $\gamma_i(t)$ is based on the stability condition of the self-adjusting process.

Stability condition:

The derivative of the Lyapunov function must be negative definite ($\dot{V} < 0$).

It is established that the self-adjusting process is stable if the following condition is satisfied (5.33).

$$2x^T Pu + \sum_{i=1}^{n+1} 2\lambda_i \gamma_i \dot{\gamma}_i \leq 0 \tag{5.33}$$

This inequality is satisfied if $\dot{\gamma}_i$ are chosen in the following form:

$$\dot{\gamma}_1 = \frac{1}{\lambda_1}(p_{n1}x_1 + p_{n2}x_2 + \ldots + p_{nn}x_n)y \tag{5.34}$$

$$\dot{\gamma}_2 = \frac{1}{\lambda_2}(p_{n1}x_1 + p_{n2}x_2 + \ldots + p_{nn}x_n)y^{(1)} \tag{5.35}$$

$$\vdots \tag{5.36}$$

$$\dot{\gamma}_n = \frac{1}{\lambda_n}(p_{n1}x_1 + p_{n2}x_2 + \ldots + p_{nn}x_n)y^{(n-1)} \tag{5.37}$$

$$\dot{\gamma}_{n+1} = \frac{1}{\lambda_{n+1}}(p_{n1}x_1 + p_{n2}x_2 + \ldots + p_{nn}x_n)y_c(t) \tag{5.38}$$

Hence, the adjustment algorithm can be deduced.

$$\dot{k}_a = \frac{1}{k_0(T_k)\lambda_{n+1}}(p_{n1}x_1 + p_{n2}x_2 + \ldots + p_{nn}x_n)y_c \tag{5.39}$$

$$\dot{c}_{n-1} = -\frac{1}{k_0(T_k)\lambda_n}(p_{n1}x_1 + p_{n2}x_2 + \ldots + p_{nn}x_n)y^{(n-1)} \tag{5.40}$$

$$\vdots \tag{5.41}$$

$$\dot{c}_0 = -\frac{1}{k_0(T_k)\lambda_1}(p_{n1}x_1 + p_{n2}x_2 + \ldots + p_{nn}x_n)y \tag{5.42}$$

Considering $e^{(i)} = x_{i+1}$, the adjustment algorithm can be expressed in a condensed form as follows.

$$\dot{k}_a(t) = \frac{\sum_{i=1}^{n} p_{ni}e^{(i)}y_c}{k_0(T_k)\lambda_{n+1}} \tag{5.43}$$

$$\dot{c}_{n-1}(t) = \frac{\sum_{i=1}^{n} p_{ni}e^{(i)}y^{i-1}}{k_0(T_k)\lambda_i} \tag{5.44}$$

This algorithm is employed to develop model reference adaptive systems, and the parameter calculations are carried out within defined self-adjustment intervals, with the values p_{ni} chosen according to the conditions of the matrices P and Q.

5.3.3 Popov's Hyperstability Method

This method was developed to address certain limitations identified in the Lyapunov method. It was further refined by researchers such as Landau for the synthesis of model reference adaptive control systems. The aim is to find a mechanism that can both eliminate the error and ensure system stability. Since the dynamics of physical systems are often governed by complex, strongly coupled nonlinear differential equations, finding the explicit solution to these equations to study stability is typically a challenging task. To overcome this issue, Popov's hyperstability theory has become widely adopted within the control systems community [63].

5.3.3.1 Concept of Hyperstability

The concept of hyperstability was introduced by Popov for the stability analysis of dynamic systems with nonlinear feedback. Popov's stability criterion is based on the idea that total energy in the system must always be non-decreasing. This can be expressed by the inequality: **Energy supplied + Initial energy ≥ Stored energy**.
Mathematically, this relationship is formulated as follows.

$$\int_{t_0}^{t} u^T(\tau)y(\tau)d\tau + \frac{1}{2}x^T(t_0)x(t_0) \geq \frac{1}{2}x^T(t)x(t) \tag{5.45}$$

This inequality expresses the principle of energy conservation in linear passive systems, ensuring stability. Since then, Popov's work has focused on extending this principle to real passive systems, that is, non-linear systems [63].

Hyperstability of Linear Systems

The hyperstability of linear systems, also known as the concept of positivity, involves studying local stability around an equilibrium point. In contrast, Popov's integral inequality, based on the concept of energy, determines the global stability of the system within its operating domain.

Hyperstability of Nonlinear Systems

From inequality (5.45), the hyperstability of a nonlinear system, as demonstrated by Popov, is obtained. A nonlinear system is weakly hyperstable if, for ($t \geq t_0$) with ($t_0 \geq 0$), $u(\tau)$ satisfies the inequality:

$$\int_{t_0}^{t} u^T(\tau)y(\tau)d\tau \geq -{\gamma_0}^2 \tag{5.46}$$

Here, γ_0 is a time-independent constant. This definition is particularly important for the analysis of the stability of nonlinear systems.

5.3.3.2 Study of Model Reference Adaptive Control through Hyperstability

Consider the nonlinear dynamic system:

$$\dot{x}(t) = Ax(t) + Bu(t) + f_{NL}(x,t) \tag{5.47}$$

$$y(t) = Cx(t) \tag{5.48}$$

Where:
A: matrix of size $n \times n$
B: input matrix of size $m \times n$
$x(t)$: state vector of dimension n
$u(t)$: control vector of dimension m
$f_{NL}(x,t)$: time-varying nonlinear term.
The vector y and the matrix C represent the output variable and matrix, respectively, of appropriate dimensions.
The reference model is given by:

$$\dot{x}_m(t) = A_m x_m(t) + B_m r(t) \tag{5.49}$$

$$y_m(t) = C_m x_m(t) \tag{5.50}$$

The error vector is defined by the following.

$$e(t) = x_m(t) - x(t) \tag{5.51}$$

The objective is to achieve a zero steady-state error, $\lim_{t\to+\infty} e(t) = 0$, by generating a control law through a regulator. This control law is calculated based on measured quantities using linear adaptation gains. The latter is given by [63]:

$$u(t) = -K_p x(t) + K_u r(t) + K_e e(t) \tag{5.52}$$

where K_e,K_u, K_p are linear adaptation gains.
K_e: for closed-loop stabilization;
K_u, K_p: for reference tracking.
The dynamic error equation is given by:

$$\dot{e}(t) = A_m x_m(t) + (-A + BK_p)x(t) + (B_m - BK_u)r(t) - BK_e e(t) - f_{NL}(x,t) \tag{5.53}$$

By adding $A_m x(t)$ to both sides:

$$\dot{e}(t) + A_m x(t) = A_m x_m(t) + (-A + BK_p)x(t) + (B_m - BK_u)r(t) - BK_e e(t) - f_{NL}(x,t) + A_m x(t) \tag{5.54}$$

$$\dot{e}(t) = (A_m - A + BK_p)x(t) + (B_m - BK_u)r(t) + (A_m - BK_e)e(t) - f_{NL}(x,t) \tag{5.55}$$

To obtain **a linear time-invariant system**, it is necessary that:

$$f_{NL} = 0 \tag{5.56}$$
$$(B_m - BK_u) = 0 \tag{5.57}$$
$$(A_m - A + BK_p) = 0 \tag{5.58}$$

Thus, the variation of the error becomes:

$$\dot{e}(t) = (A_m - BK_e)e(t) \tag{5.59}$$

$(A_m - BK_e) = A_e$, represents the dynamic error matrix, where A_e is chosen such that it forms a Hurwitz matrix. A matrix is classified as Hurwitz if all of its eigenvalues have strictly negative real parts, which ensures the system's stability. This property directly leads to the satisfaction of condition $\lim_{t\to+\infty} e(t) = 0$, meaning that the error tends to zero as time approaches infinity. In addition to A_e being the chosen matrix and using Equation (5.58), the following results are achieved.

$$K_e = (B^TB)^{-1}B^T(A_m - A_e) \tag{5.60}$$
$$K_p = (B^TB)^{-1}B^T(A - A_m) \tag{5.61}$$
$$K_u = (B^TB)^{-1}B^TB_m \tag{5.62}$$

The equations for K_u and K_p cannot be satisfied unless the following **Erzberger conditions** are met:

$$(I_n - B(B^TB)B^T)(A - A_m) = 0 \tag{5.63}$$
$$(I_n - B(B^TB)B^T)B_m = 0 \tag{5.64}$$

5.3.3.3 Nonlinear Time-Varying System $f_{NL}(x,t) \neq 0$

The control developed in the previous part, where $f_{NL}(x,t) = 0$, yields satisfactory performance only when the system is linear and invariant in time. To address this, it is necessary to introduce a nonlinear control to compensate for the parametric variations of the system. The error equation becomes the following.

$$\dot{e}(t) = (A_m - BK_e)e(t) - f_{NL}(x,t) \tag{5.65}$$

The control law is the sum of a linear component and a nonlinear component.

$$u(t) = u_1(t) + u_2(t) \tag{5.66}$$

With :

$$u_1(t) = -K_p x(t) + K_u r(t) + K_e e(t) \tag{5.67}$$
$$u_2(t) = K_x x(t) + K_w r(t) + K_\theta e(t) \tag{5.68}$$

Here, K_x, K_w, and K_θ are nonlinear adaptation gains.
$x(t)$ is the output signal, $r(t)$ is the reference signal and $e(t)$ is the error between the output signal and the output of the reference model.

5.3.3.4 Hyperstability conditions of a system

- The linear part must be strictly positive and real;
- The nonlinear part must satisfy the Popov inequality (5.46).

Since the Popov inequality is not easy to solve, Landau proposed a proportional-integral adaptation law to compute the nonlinear adaptation gains [63].

$$K_x = \int_0^t M v x^T S dt + L v x^T S \tag{5.69}$$
$$K_w = \int_0^t G v r^T T dt + F v r^T T \tag{5.70}$$
$$K_\theta = \int_0^t Q v e^T U dt + P v e^T U \tag{5.71}$$

With:
$v = De$: the output vector of the linear block.
The matrices M, G, Q, S, T, and U are defined as positive, while the matrices L, F, and P are semi-positive definite. These equations have been simplified to find a solution to the Popov inequality as follows.

$$K_x = \int_0^t k_1 v x^T dt + k_2 v x^T \tag{5.72}$$
$$K_w = \int_0^t k_1 v r^T dt + k_2 v r^T \tag{5.73}$$
$$K_\theta = \int_0^t k_1 v e^T dt + k_2 v e^T \tag{5.74}$$
$$\tag{5.75}$$

With: $k_1 > 0$, $k_2 \geq 0$, and they must satisfy:

$$S = T = U = 1 \tag{5.76}$$
$$M = G = Q = k_1 I_n \tag{5.77}$$
$$L = F = P = k_2 I_n \tag{5.78}$$

The choice of k_1 and k_2 depends on the system being controlled and the adaptation error.

5.4 Comparative Evaluation of Adaptive Speed Control Strategies

The primary objective of this section is to compare Lyapunov-based and hyperstability-based model reference adaptive control approaches for autonomous vehicle speed regulation, a critical requirement for ensuring safety, comfort, and efficiency. Adaptive control techniques have demonstrated remarkable robustness in handling system uncertainties and external disturbances.

5.4.1 Hyperstability Criterion on Longitudinal Adaptive Control

This section presents the implementation of hyperstability-based model reference adaptive control for the vehicle's longitudinal dynamics, where a DC motor is used as the speed control actuator. Mechanical dynamics of the latter are used as the main system to construct the adaptive mechanism. This approach is particularly crucial in the context of autonomous electric vehicles, where precise and reliable speed regulation is critical for safe and efficient operation [5].

Implementation

The control law applied on the actuator's dynamics is given by:

$$u(t) = u_1(t) + u_2(t) \tag{5.79}$$

Where:

$$u_1(t) = -K_p\Omega + K_u\Omega_{\text{ref}} + K_e e \tag{5.80}$$

$$u_2(t) = K_x\Omega + K_w\Omega_{\text{ref}} + K_\theta e \tag{5.81}$$

The equation representing the motor's dynamics is given by:

$$J\frac{d\Omega}{dt} = T_{\text{em}} - f\Omega - T_L \tag{5.82}$$

The chosen reference model is as follows.

$$\tau\frac{d\Omega_m}{dt} + \Omega_m = k\Omega_{\text{ref}} \tag{5.83}$$

The tracking error (between the reference model and the motor's model) is defined as follows.

$$e = \Omega_m - \Omega \tag{5.84}$$

Applying identification with Equations 5.49 and 5.47, the following parameters are obtained:

- A_m: $-\frac{1}{\tau}$
- B_m: $\frac{k}{\tau}$
- A: $-\frac{f}{J}$
- B: $\frac{1}{J}$

Hence, the linear gains are obtained and given in Table 5.1.

Parameter	K_u	K_p	K_e
Expression	$\frac{Jk}{\tau}$	$\frac{J}{\tau} - f$	$J\left(1 - \frac{1}{\tau}\right)$

Table 5.1: Linear gain parameters and their expressions.

The nonlinear gains are computed as follows:

$$K_x = \int_0^t M \cdot v \cdot \Omega \, dt + L \cdot v \cdot \Omega \tag{5.85}$$

$$K_w = \int_0^t G \cdot v \cdot \Omega_{\text{ref}} \, dt + f \cdot v \cdot \Omega_{\text{ref}} \tag{5.86}$$

$$K_\theta = \int_0^t Q \cdot v \cdot e \, dt + P \cdot v \cdot e \tag{5.87}$$

Where:

- $v = D.e$
- $S = T = U = 1$
- $F = L = P = k_1$
- $G = M = Q = k_2$

Table 5.2: Hyperstability-based MRAC parameters

Parameter	**Value**
k	1
τ	1×10^{-4} s
k_1	0.01
k_2	0.01
D	5

The model reference adaptive control system based on hyperstability criterion is illustrated in Figure 5.3, the corresponding parameters are given in Table 5.2.

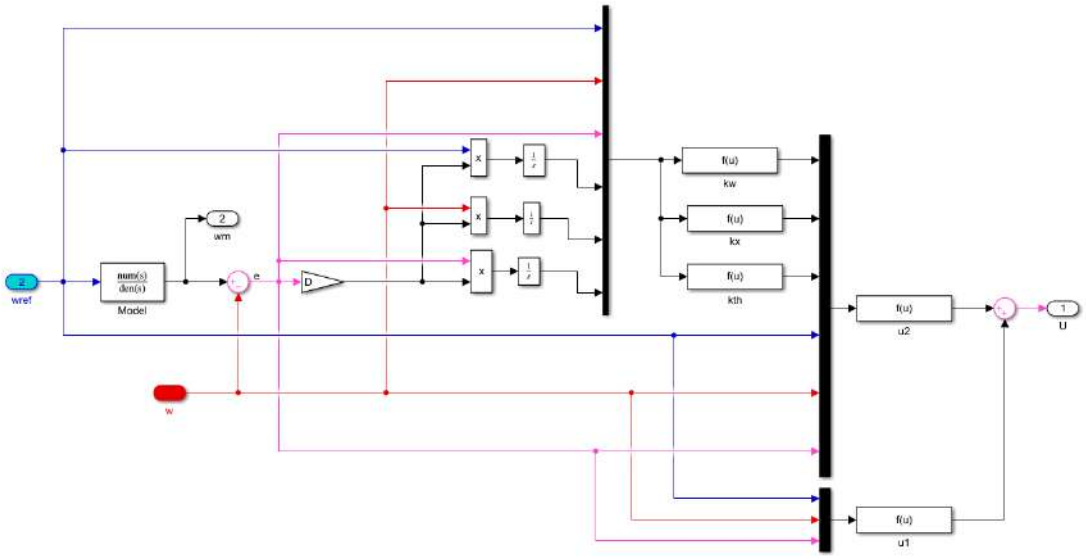


Figure 5.3: Hyperstability-based MRAC block diagram

To test the effectiveness of the proposed algorithm, the angular velocity of the DC motor is plotted. In this test, a sinusoidal profile is used as a reference input for the angular velocity. Figure 5.4 compares the reference input with the output of the reference model and the actual output. As observed, the actual velocity successfully tracks the reference velocity. Furthermore, it performs a high degree of alignment with the model reference output, demonstrating the effectiveness of the proposed adaptive control mechanism.

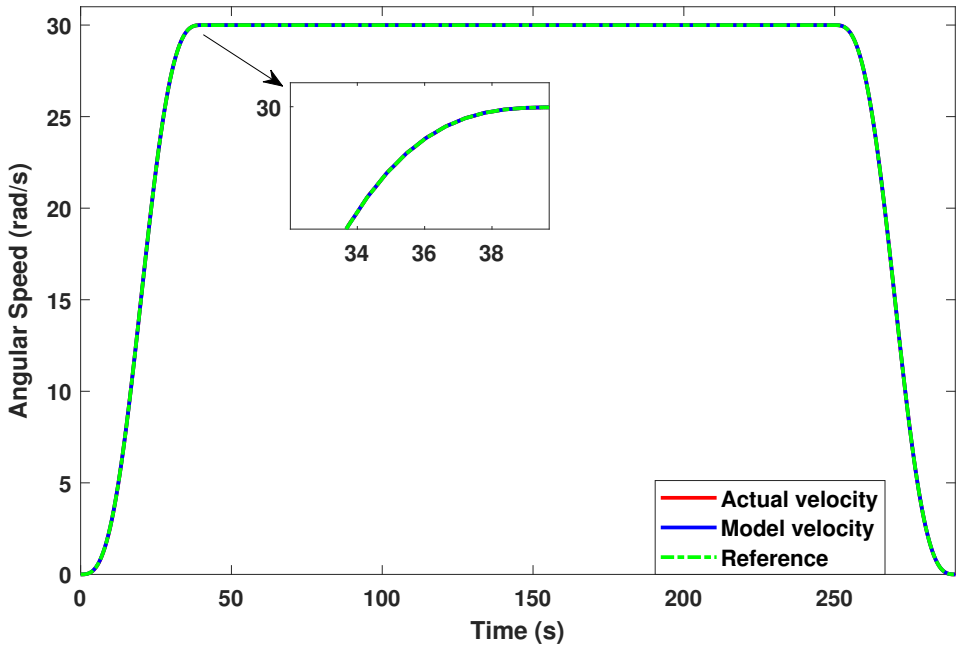


Figure 5.4: Hyperstability-based MRAC angular velocity results

5.4.2 Lyapunov-based Longitudinal Adaptive Control

This section presents the implementation of Lyapunov-based model reference adaptive control for autonomous vehicle speed regulation. .

Implementation

The reference model is defined as a first-order system, given in Equation 5.88.

$$W_m(s) = \frac{\Omega_m}{\Omega_{\text{ref}}} = \frac{1}{\tau s + 1} \tag{5.88}$$

Ω_m represents the model's speed, while Ω_{ref} denotes the speed reference, and τ is a time constant.

The transfer function for the main system is obtained from Equation 3.11 and is expressed as follows.

$$W_0(s) = \frac{\Omega}{T_{\text{em}} - T_{\text{L}}} = \frac{\frac{1}{J}}{s + \frac{f}{J}} \tag{5.89}$$

Applying identification with Equations 5.6 and 5.7, the following parameters are obtained (Table 5.3).

The coefficients to be adjusted are reduced to:

$$\gamma_1 = b_0 - [a_0 + K_0 c_0(t)] \tag{5.90}$$

Table 5.3: Identified parameters

Parameter	**Expression**
a_0	$\frac{f}{J}$
k_0	$\frac{1}{J}$
$k_m = b_0$	$\frac{1}{\tau}$

$$\gamma_2 = k_m - k_a(t)k_0 \tag{5.91}$$

Hence, the adjustment mechanism is deduced:

$$\begin{aligned} \dot{k_a} &= \frac{1}{k_0 \cdot \lambda_2} \cdot P \cdot e \cdot \Omega_{\text{ref}}, \\ \dot{c_0} &= -\frac{1}{k_0 \cdot \lambda_1} \cdot P \cdot e \cdot \Omega \end{aligned} \tag{5.92}$$

Considering that the controlled system is a first-order system, A, P, and Q are scalars. $A = -b_0$ and $P = \frac{Q}{2b_0}$. The tracking error and the control law are given by Equations 5.93 and 5.94, respectively.

$$e = \Omega_m - \Omega \tag{5.93}$$

$$u_0 = \gamma_2 \Omega_{\text{ref}} - \gamma_1 \Omega \tag{5.94}$$

The Lyapunov-based MRAC system is illustrated in Figure 5.5, the corresponding parameters are given in Table 5.4.

Table 5.4: Lyapunov-based MRAC parameters

Parameter	**Value**
τ	$5 \cdot 10^{-5}\,\text{s}$
λ_1	0.1
λ_2	0.1
Q	0.001

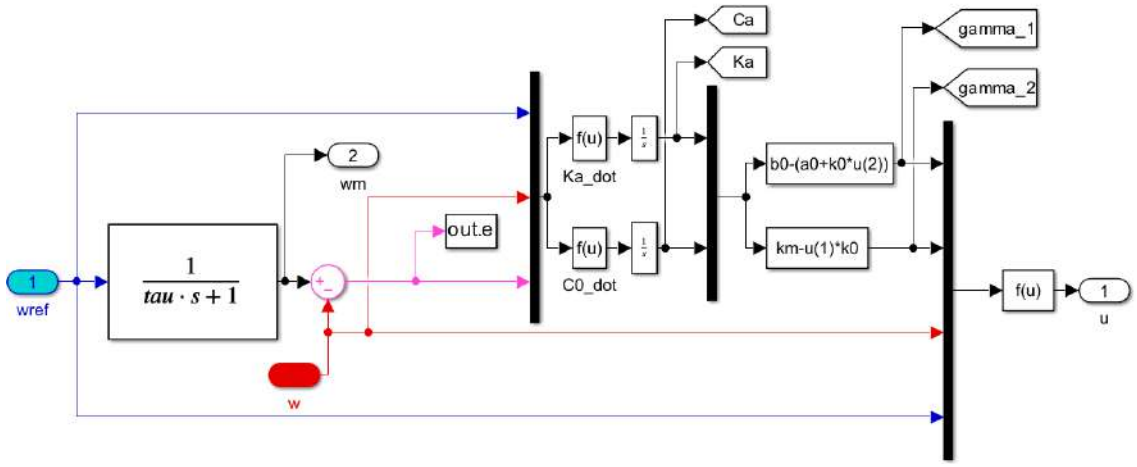


Figure 5.5: Lyapunov-based MRAC block diagram

To assess the effectiveness of the proposed algorithm, the actuator's angular velocity is depicted in Figure 5.6, where a sinusoidal signal is used as a reference input. As shown, the actual velocity closely follows the reference input and exhibits strong agreement with the reference model output. These results, obtained using the Lyapunov-based approach, are consistent with those previously achieved using the hyperstability-based criterion. Thereby, reinforcing the validity and robustness of the proposed adaptive control strategy.

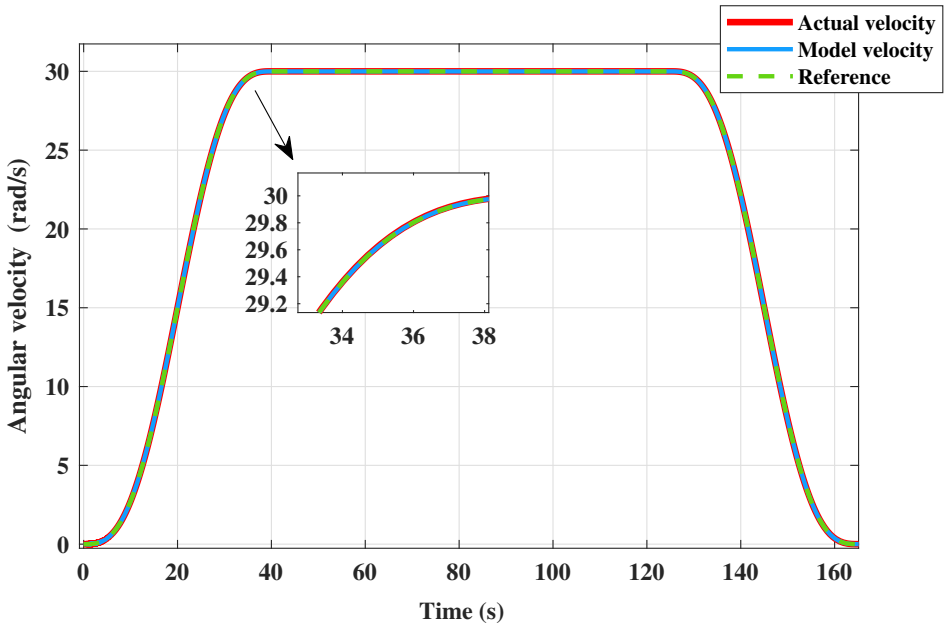


Figure 5.6: Lyapunov-based MRAC angular speed results

5.4.3 Results and Discussion

In this section, the performance of the two adaptive control strategies is comparatively evaluated in speed regulation. Both control systems are tasked with accurately tracking a predefined speed reference profile. The reference input chosen for this study is a sinusoidal speed profile, designed to continuously vary between 0 m/s and 8 m/s over time. This type of reference profile allows for assessing the controllers abilities to handle both acceleration and deceleration phases, making it a suitable benchmark for adaptive speed control applications. Two distinct configurations of the overall control system are considered. In both cases, the core structure of the speed control system includes a speed reference generator, an actuator (DC motor), a feedback loop for measuring the actual motor speed, and a controller responsible for adjusting the control input based on the tracking error.
The key difference between the two configurations lies in the design methodology of the adaptive controller.

The first configuration employing hyperstability-based adaptive controller is given in Figure 5.7, and the configuration with Lyapunov-based adaptive controller is shown in Figure 5.8. Both control systems are implemented and tested under the same operating conditions. Their performance is then compared in terms of transient response, steady-state accuracy, and overall robustness during both acceleration and deceleration phases. This comparative analysis is crucial, as it allows for identifying the control strategy that delivers the most reliable and consistent results for the intended application. The selected controller from this study will be used as the baseline solution for the remaining parts of this thesis, which will address more advanced control scenarios. Figure 5.9 presents the speed tracking performance along with the corresponding error plots of both controllers.

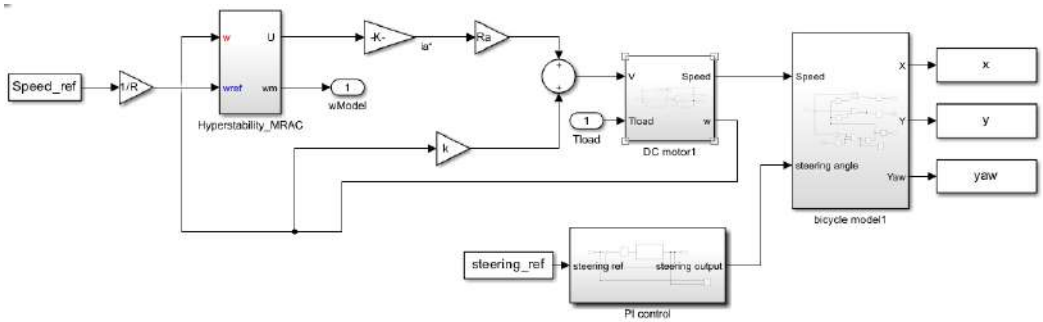


Figure 5.7: Overall control system with hyperstability-based adaptive controller

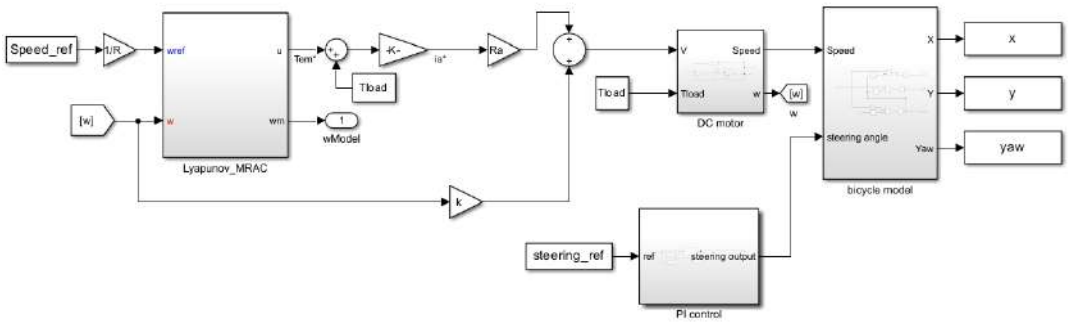


Figure 5.8: Overall control system with Lyapunov-based adaptive controller

The speed responses of the controllers closely follow the speed reference as shown in the speed plot in Figure 5.9. During the acceleration and deceleration phases, both controllers demonstrate rapid convergence with minimal overshoot. The speed error plot highlights subtle differences in the transient tracking behaviour. The Lyapunov-based controller achieves a slightly tighter adherence to the reference trajectory, with its error converging more effectively to zero, whereas the hyperstability-based controller exhibits a less accurate transient response with a marginally larger tracking error.

Key observations include:

- **Initial transient phase (0–20 s):** The hyperstability controller exhibits a higher peak error, approximately $5 \times 10^{-3}\,\mathrm{m/s}$, while the Lyapunov controller maintains a lower peak error of around $2 \times 10^{-4}\,\mathrm{m/s}$.
- **Error convergence:** Both errors decay rapidly within 30 s, with the Lyapunov controller achieving slightly faster convergence.
- **Steady-state performance:** Both controllers maintain negligible steady-state errors, on the order of $10^{-4}\,\mathrm{m/s}$.
- **Deceleration phase (after 250 s):** The hyperstability controller shows slightly more oscillatory behaviour, while the Lyapunov controller achieves lower amplitude oscillations with quicker settling.

Table 5.5 summarizes key quantitative performance metrics.

Table 5.5: Quantitative comparison of control strategies.

Performance metric	**Lyapunov-based control**	**Hyperstability-based control**
Peak transient error (m/s)	2×10^{-4}	5×10^{-3}
Steady-state error (m/s)	$< 10^{-4}$	$< 10^{-4}$
Settling time (s)	30	35
Deceleration phase overshoot	Lower	Higher

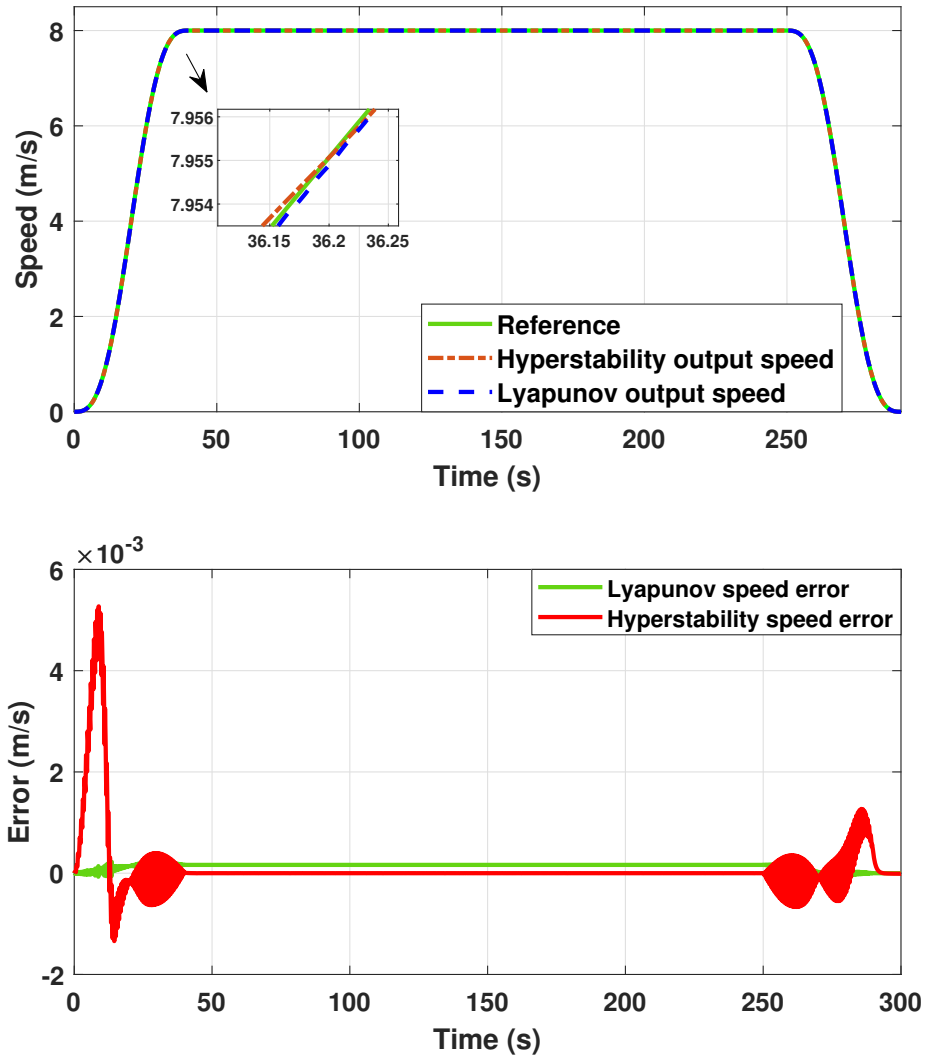


Figure 5.9: Speed tracking performance comparison.

Overall, both adaptive control strategies exhibit robust and precise speed tracking capabilities. However, the Lyapunov-based adaptive controller outperforms the hyperstability-based controller across several important aspects of transient performance. Specifically, it achieves lower peak errors during rapid changes in the reference speed, demonstrating a more disciplined and stable transient response. Additionally, the Lyapunov-based controller exhibits faster error convergence, with the tracking error diminishing more quickly to zero compared to its counterpart. Another notable advantage is the reduction of oscillations during deceleration phases, where the hyperstability-based controller tends to display minor fluctuations. These performance improvements can be attributed to the Lyapunov-based controller's formal guarantee of global asymptotic stability, achieved through the careful design of a suitable Lyapunov function. This approach allows for more consistent and effective adaptive gain tuning, resulting in tighter error regulation and improved overall tracking accuracy throughout the entire speed profile.

5.5 Conclusion

This chapter provided a comprehensive overview of adaptive control principles, highlighting its key mechanisms and structural configurations. Adaptive control systems are designed to modify their behaviour in response to changes in system dynamics and external disturbances, ensuring consistent performance even under uncertain conditions. The general architecture of an adaptive control system consists of two interconnected loops: a conventional feedback loop comprising the process and regulator, and an adaptation loop responsible for adjusting control parameters in real-time. The nature of this adaptation mechanism varies based on the chosen control approach, whether it involves preprogrammed adjustments, model reference frameworks, or self-tuning regulators. In this work, the focus was directed toward Model Reference Adaptive Systems (MRAS), with two specific adaptive control strategies implemented and evaluated: a hyperstability-based adaptive controller and a Lyapunov-based adaptive controller. Both configurations were tested using a sinusoidal speed reference profile, designed to assess their transient and steady-state tracking capabilities under varying conditions. Comparative analysis revealed that while both controllers achieved satisfactory steady-state accuracy and rapid error convergence, the Lyapunov-based adaptive controller consistently outperformed its counterpart. In light of these findings, the Lyapunov-based adaptive controller has been selected as the primary control strategy for the subsequent chapters of this thesis.

Chapter 6

Performance Summary and Insights

6.1 Introduction

This chapter summarizes the main contributions and results achieved during the progressive development and evaluation of the autonomous vehicle control system. The control system is structured into two modules:

- **Lateral control**: A fixed PI controller is used throughout all experiments to regulate the steering angle. Its performance is evaluated through specific lateral test scenarios.
- **Longitudinal control**: Several controllers are investigated and compared.

The initial implementation for longitudinal control employed a classical PID controller to regulate vehicle speed. To enhance performance, offline optimization using Genetic Algorithm and Jellyfish Search was applied. The JS-based controller demonstrated superior accuracy and faster convergence compared to GA-PID and baseline PID. Subsequently, a Model Reference Adaptive Control (MRAC) approach was introduced to ensure online adaptability. Two variants were considered:

- Hyperstability-based MRAC
- Lyapunov-based MRAC

Simulation results highlighted the Lyapunov-based MRAC's superiority in terms of tracking precision and robustness under dynamic conditions.

6.2 System Configuration

The system performance is assessed under four tests to evaluate both lateral and longitudinal behaviour. The methodology for longitudinal control follows a progressive enhancement from classical control to optimization-based techniques, and finally to adaptive control, as outlined below. Each control strategy is tested under realistic vehicle dynamics. The progression includes:

1. Classical PID controller
2. Optimized PID controller using Jellyfish Search
3. Lyapunov-based Model Reference Adaptive Controller

The physical and dynamic parameters of the vehicle used in the simulations are summarised in Table 6.1.

Table 6.1: Vehicle parameters

Parameter	**Value**
Wheelbase L	2.96 m
Tire radius R	0.2667 m
Speed limit V_m	8 m/s
Acceleration limit A_m	0.4 m/s^2

6.3 Test Scenarios and Results

6.3.0.1 Steady-State Circular Test

To evaluate the lateral control performance under steady-state conditions, a circular trajectory of radius $r = 120\,\text{m}$ is selected. A constant steering input is applied, calculated according to Ackermann steering geometry [64]:

$$\delta = \tan^{-1}\left(\frac{L}{r}\right)$$

L denotes the vehicle's wheelbase. This setup allows for a consistent curvature, ensuring that the vehicle remains on a circular path throughout the test.
Figure 6.1 illustrates the vehicle's position tracking along a circular trajectory on the XY plane. As observed, the vehicle's path closely follows the reference trajectory, demonstrating the lateral control system's ability to maintain accurate curvature tracking over time. The trajectory remains superimposed throughout the test, indicating precise steering angle regulation and minimal lateral deviation. In Figure 6.2, the yaw angle response provides further insight into the system's stability. The angle exhibits a smooth and monotonic increase, ultimately converging to approximately 6.28 rad, which corresponds to a full rotation around the 120-meter radius circular path (i.e., 2π radians). The absence of high-frequency oscillations or instability in the yaw response confirms the effectiveness of the lateral PI controller in maintaining directional stability under steady-state conditions.

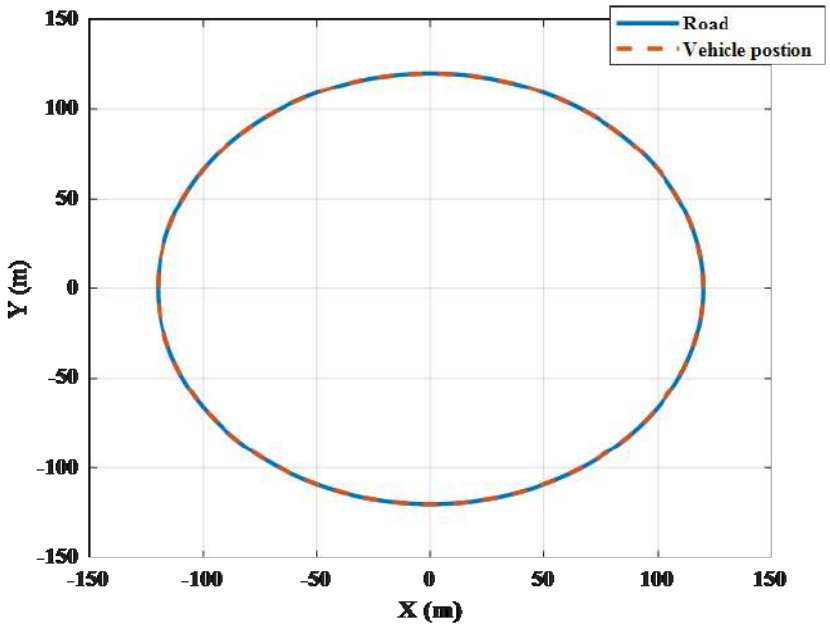


Figure 6.1: Position tracking along the circular trajectory

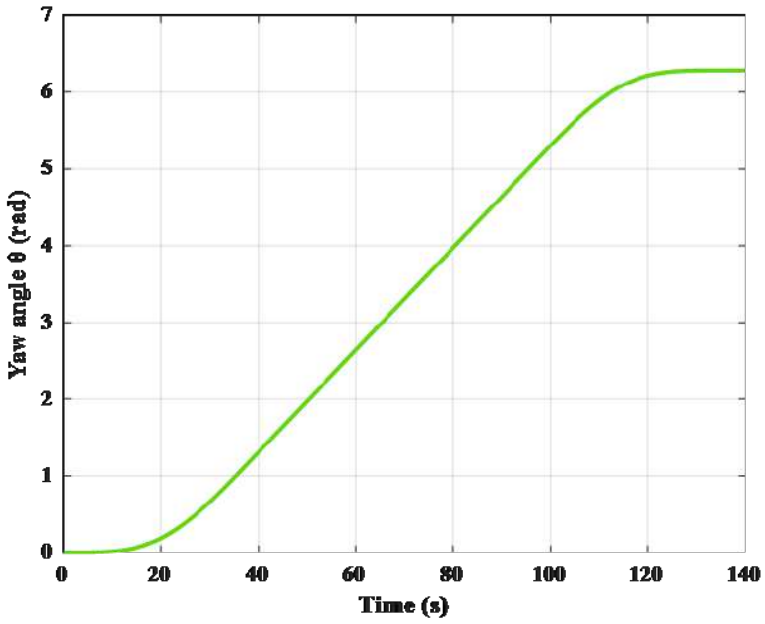


Figure 6.2: Yaw angle evolution

6.3.0.2 Transient Step-Steer Test

To assess the system's transient performance under rapid directional changes, the ISO 7401 step-steer input is applied. This standard test simulates a sudden and significant change in the steering angle, typically encountered during emergency manoeuvres or sharp turns. The objective is to evaluate the responsiveness and stability of the lateral control loop, particularly in handling abrupt inputs without inducing instability or excessive oscillations.
Figure 6.3 displays the steering angle output in response to the ISO step input. As observed, the controller enables the system to track the reference with high precision and negligible lag. The response curve rises sharply, closely following the reference input profile, and stabilizes without overshoot or delay. This behaviour underscores the efficacy of the PI-based lateral controller in delivering prompt and stable steering actions. Moreover, the minimal steady-state error and the smooth transition in the steering profile highlight the controller's ability to preserve lateral stability even under aggressive input conditions. Such performance is essential for ensuring safe and predictable vehicle handling in real-world driving scenarios, especially when operating at higher speeds or navigating tight curves.

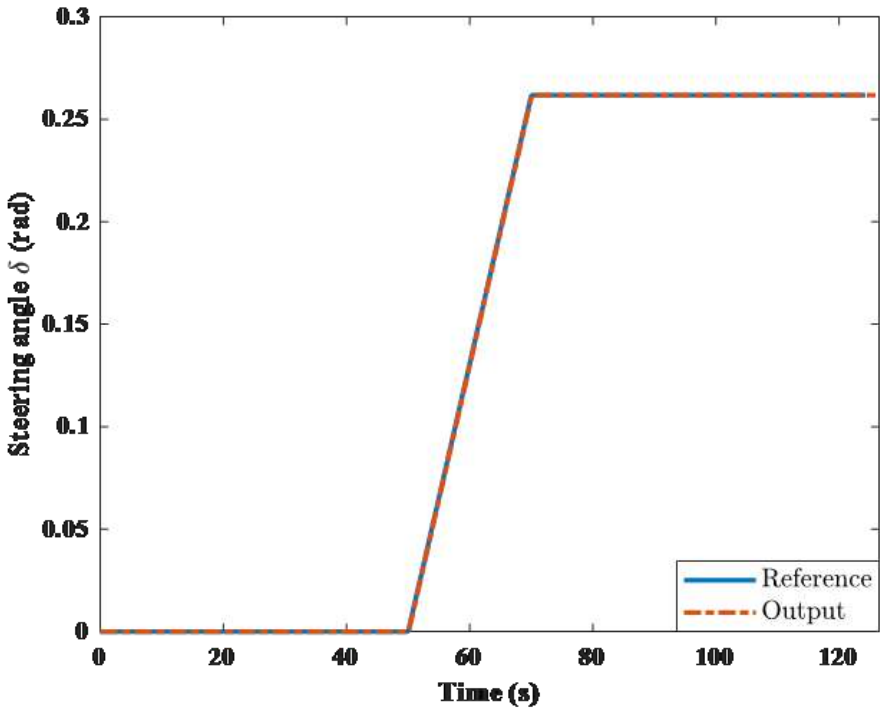


Figure 6.3: Steering angle tracking

6.3.0.3 Parametric Uncertainty Test

To assess robustness, the armature resistance R_a is increased by 200%. Speed tracking results under different controllers are shown below. This test aims to assess the robustness and adaptability of different control strategies when the system is subjected to sudden dynamic changes.

Figure 6.4 illustrates the vehicle's speed response before and after introducing resistance variation, specifically an increase in internal resistance. As observed, the conventional PID controller exhibits a perceptible tracking delay following resistance variation, deviating from the reference signal by approximately 0.1 seconds. This lag is indicative of the controller's limited ability to adapt to real-time parameter shifts, highlighting its dependency on fixed gain values tuned under nominal conditions. The JS-optimized PID controller, while demonstrating better adaptability, still shows a minor deviation of roughly 0.02 seconds. This improvement can be attributed to the jellyfish algorithm's ability to optimize controller gains for improved performance across a broader operating range. However, its response remains sensitive to abrupt disturbances, revealing a trade-off between robustness and pre-optimized performance. In contrast, the MRAC (Model Reference Adaptive Control) exhibits outstanding robustness to resistance variation. Its output remains tightly aligned with the reference trajectory, both before and after the parameter change. These observations are further confirmed in the error plot shown in Figure 6.5. The PID controller exhibits a significant increase in peak error, from approximately 0.036 m/s to 0.048 m/s—a relative increase of about 33.3%. Similarly, the JS-PID controller's overshoot increases from 0.007 m/s to 0.009 m/s, marking an approximate 28.6% rise. In contrast, MRAC exhibits a flat error profile both before and after the variation, confirming its superior robustness and adaptability.

Figures 6.6 and 6.7 illustrate the evolution of adaptive parameters during the control process. All parameters converge smoothly and remain bounded, indicating that the controller quickly adapts to system dynamics and achieves stable performance under variation in armature resistance R_a, with the parameters adjusting in real time to accommodate the internal changes. The smooth convergence observed highlights the effectiveness of the adaptation law, which preserves the stability of the closed-loop system despite the presence of internal perturbations.

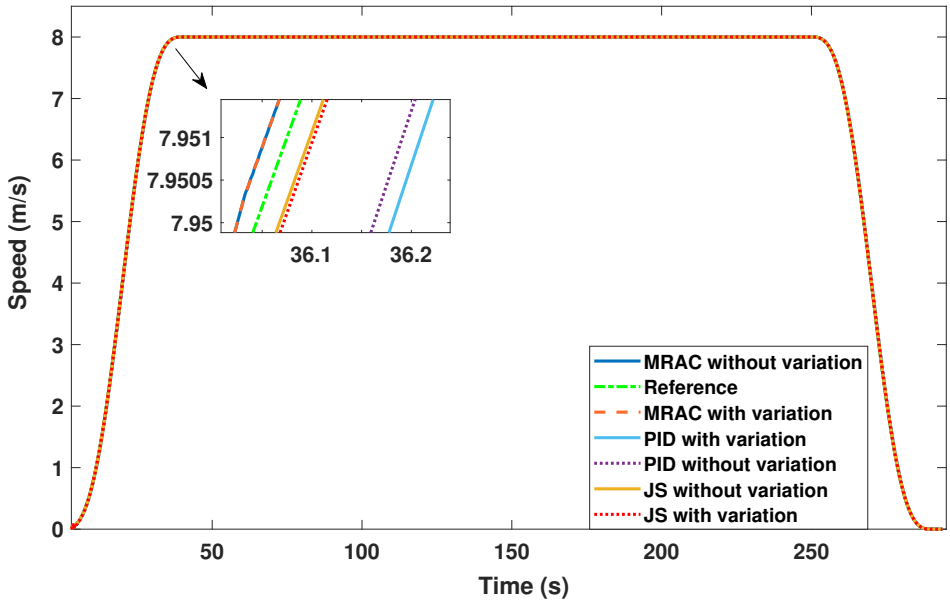


Figure 6.4: Speed tracking performance before and after resistance variation

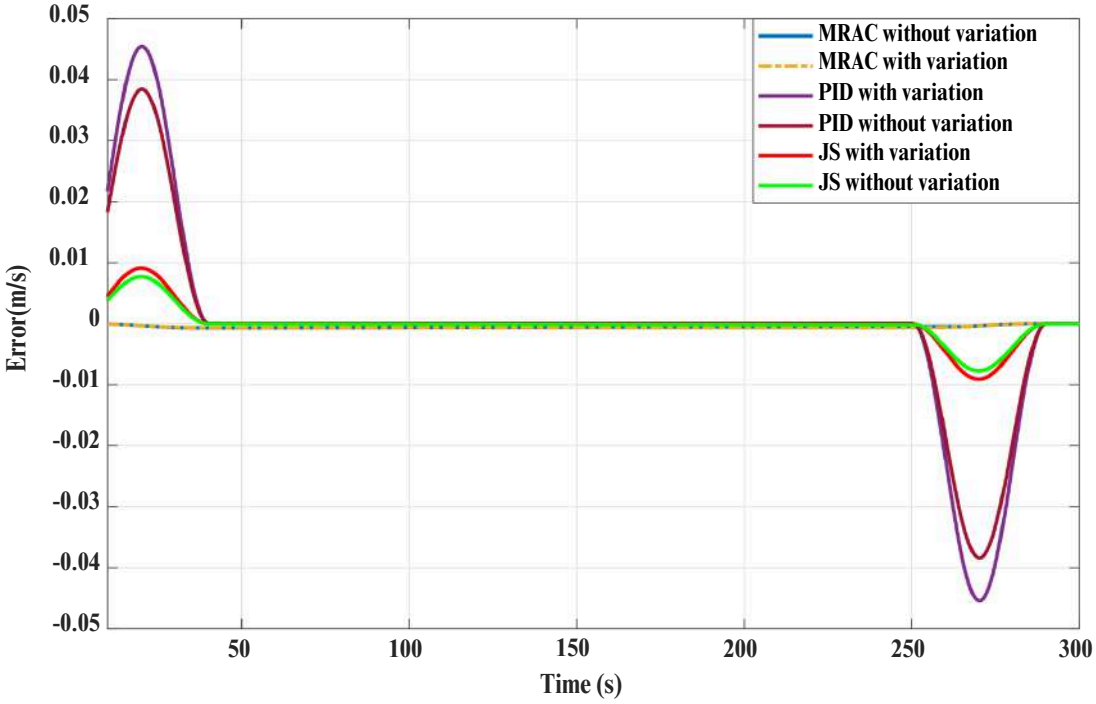


Figure 6.5: Speed tracking error under parameter variation

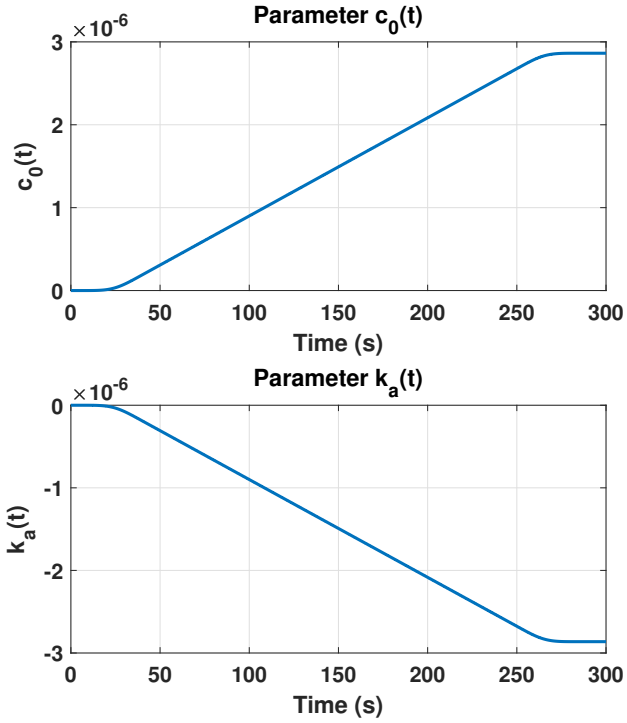


Figure 6.6: Evolution of internal parameters $c_0(t)$ and $k_a(t)$.

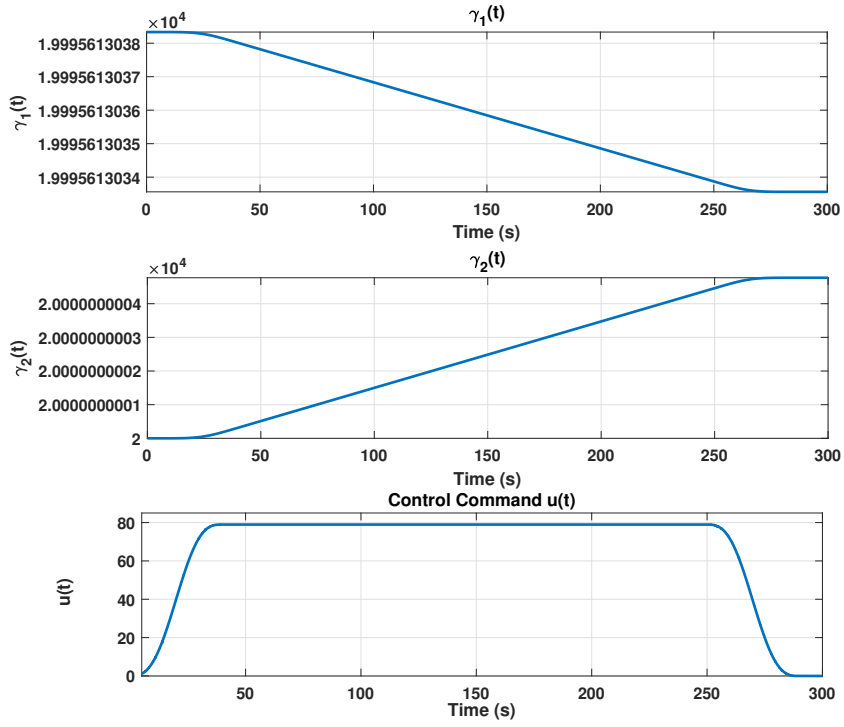


Figure 6.7: Adaptation gains and the control input $u(t)$ evolution.

6.3.0.4 Disturbance Rejection Test

In this test, the system is subjected to an external load torque disturbance to assess the robustness of the controllers. Initially, the load torque is set to its nominal value of 430 N·m. At time $t_1 = 50$ s, the load torque is increased by 10%, and it returns to its nominal value at $t_2 = 100$ s.

Figure 6.8 illustrates the speed response of the system under the control of three different strategies: the Model Reference Adaptive Control, the Jellyfish Search-optimized PID (JS-PID), and the conventional PID controller. During the disturbance interval, the conventional PID controller exhibits a pronounced overshoot and a visibly delayed recovery to the reference trajectory. The JS-PID controller achieves partial mitigation of the disturbance effects, but still shows a noticeable overshoot and oscillations. In contrast, the MRAC controller maintains a smooth and accurate speed profile, effectively rejecting the disturbance with no significant deviation from the reference. The corresponding speed error profiles are presented in Figure 6.9. Upon the introduction of the disturbance at $t_1 = 50$ s, the conventional PID controller displays a peak error of approximately 0.16 m/s. The JS-PID controller improves upon this with a reduced error magnitude of around 0.04 m/s. Nevertheless, both controllers suffer from delayed convergence back to the desired speed, indicating limited disturbance rejection capabilities. On the other hand, the MRAC controller demonstrates superior performance, characterized by a minimal and transient deviation from the reference. The system under MRAC rapidly re-aligns with the desired speed trajectory, evidencing high robustness and adaptive behaviour in the presence of external disturbances.

Figure 6.10 provides insight into the time evolution of the MRAC controller's internal parameters. The adaptation gains $\gamma_1(t)$ and $\gamma_2(t)$ are shown to evolve smoothly over time. Specifically, $\gamma_1(t)$ slightly decreases while $\gamma_2(t)$ increases, both maintaining large positive values, indicating that the adaptation mechanism remains active and responsive throughout the simulation. This adaptive behaviour enables the controller to compensate for system uncertainties and external disturbances effectively. The control input $u(t)$ shown in the bottom subplot of Figure 6.10 reacts sharply at the instants of disturbance ($t = 50$ s and $t = 100$ s), with two short impulsive changes that enable the system to reject the load variation quickly. The control signal remains bounded and stabilizes immediately after the disturbance is mitigated, showcasing good transient and steady-state behaviour. Similarly, Figure 6.11 displays the evolution of two internal parameters, $c_0(t)$ and $k_a(t)$. The parameter $c_0(t)$ increases steadily from zero to approximately 3×10^{-6}, while $k_a(t)$ decreases symmetrically from 10^{-6} to -3×10^{-6}. This inverse and smooth evolution suggests a well-balanced adaptation pro-

cess. The adaptive tuning of these parameters ensures that the control law remains effective in the face of non-linearities and disturbances, further reinforcing the controller's ability to maintain performance under variable conditions.

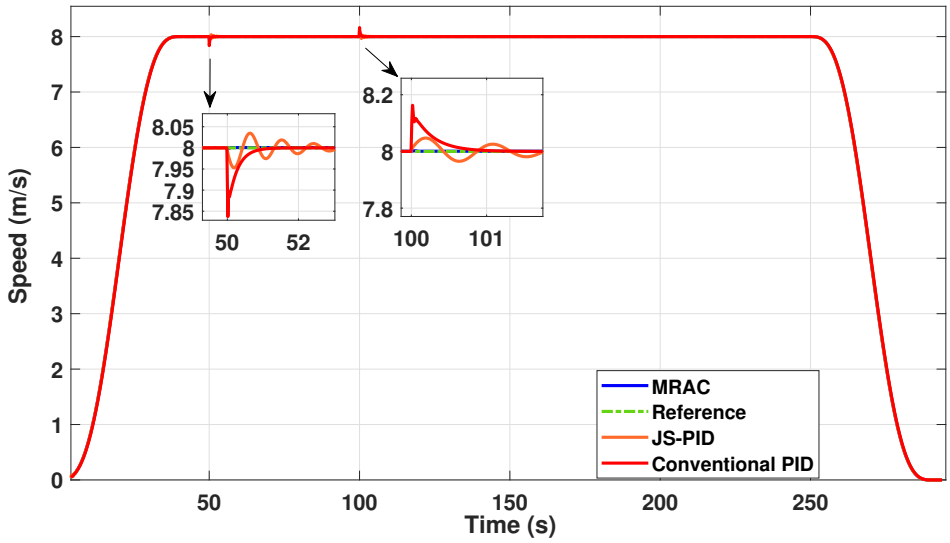


Figure 6.8: Speed response of the system under a 10% load disturbance.

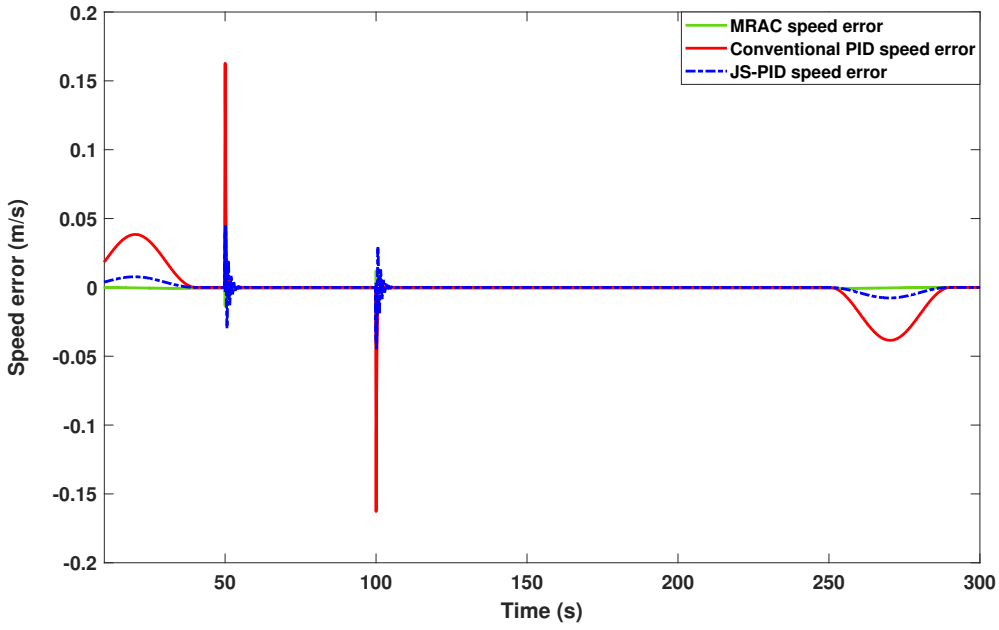


Figure 6.9: Speed error response.

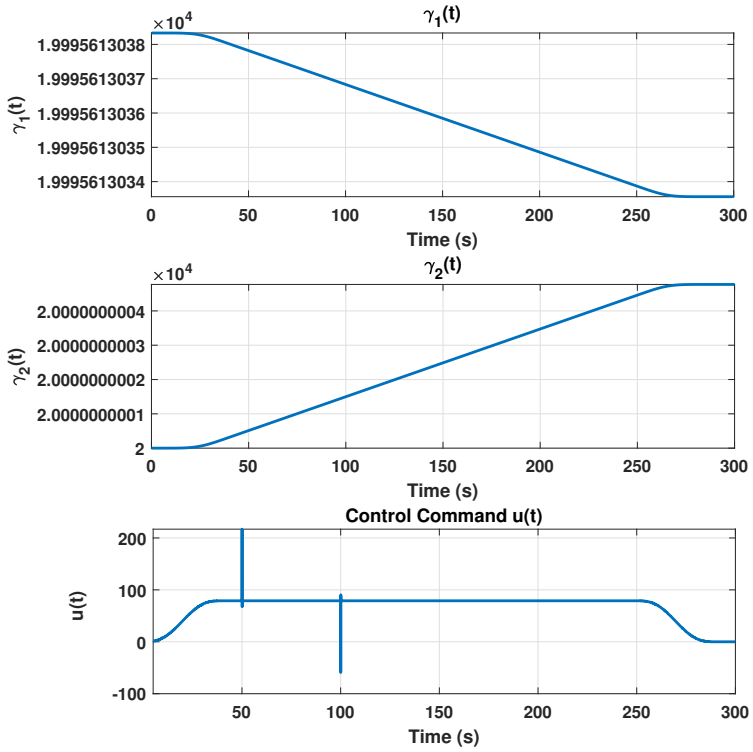


Figure 6.10: Evolution of the adaptation gains and the control input $u(t)$.

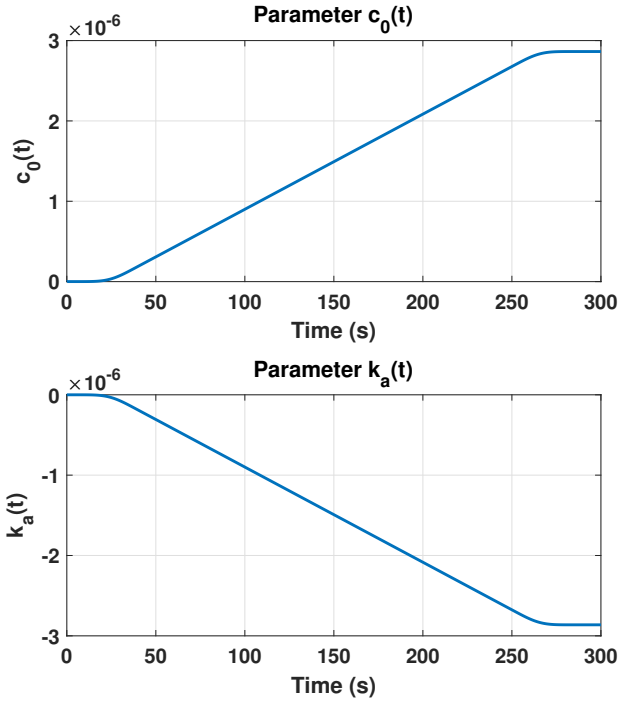


Figure 6.11: Evolution of internal parameters $c_0(t)$ and $k_a(t)$.

6.4 Conclusion

This chapter has provided a comprehensive analysis of the performance of the proposed control strategies for autonomous vehicle navigation, addressing both lateral and longitudinal dynamics. For lateral control, a fixed PI controller was employed. Through steady-state and transient tests, it demonstrated excellent tracking precision, stability, and robustness, even under abrupt steering inputs. Its ability to maintain accurate curvature following and minimal yaw oscillations underscores its suitability for typical navigation tasks. In the longitudinal domain, a progressive evaluation was conducted. The baseline PID controller, though simple, revealed limitations under dynamic uncertainties. Optimization using the Jellyfish Search algorithm significantly enhanced its responsiveness and accuracy. However, true robustness and adaptability were achieved with the implementation of a Model Reference Adaptive Control system. Among its variants, the Lyapunov-based MRAC exhibited superior performance in handling both parameter uncertainties and external disturbance rejection scenarios. Across all test cases, the MRAC approach consistently maintained minimal tracking error, stable control inputs, and smooth parameter adaptation. These findings confirm that leading a comparative study of adaptive control and bio-inspired optimization offers a powerful and resilient framework for autonomous vehicle control.

General Conclusion

This thesis has presented the development, gain scheduling optimization, and comprehensive evaluation of advanced control strategies for electric autonomous vehicles, addressing both lateral and longitudinal dynamics within the context of increasingly demanding transportation systems. Motivated by the global pursuit of safer, more efficient, and sustainable mobility solutions, an integrated control framework combining classical controllers, intelligent metaheuristic optimization techniques, adaptive control architectures, and realistic motion planning strategies was proposed.

For lateral control, a Proportional-Integral (PI) controller was adopted for trajectory tracking. Its performance was systematically validated under steady-state and transient scenarios, where it consistently demonstrated smooth yaw response, effective stability, and reliable path tracking, even under aggressive steering conditions. This confirmed its suitability for real-time navigation in nominal driving situations.

For longitudinal control, the study began by developing and evaluating two distinct speed profiles, namely the S-curve and sinusoidal profiles, which were constructed to meet road safety standards and human comfort thresholds. The comparative analysis of these profiles provided valuable insight into their suitability for various operational scenarios, establishing an essential foundation for subsequent control design.

A baseline Proportional-Integral-Derivative (PID) controller was initially deployed for longitudinal speed tracking. While effective under nominal conditions, its performance exhibited limitations. To enhance its performance, offline tuning using two nature-inspired metaheuristic optimization algorithms, the Genetic Algorithm (GA) and the Jellyfish Search (JS) optimizer, were implemented. Comparative results showed that both algorithms effectively handled complex nonlinear search spaces, with JS outperforming GA in some cases.

Recognizing the limitations of fixed-gain controllers in dynamic environments, this thesis progressed to the design of Model Reference Adaptive Control (MRAC) architectures, incorporating two distinct formulations. Lyapunov-based MRAC and a Popov Hyperstability-based MRAC. Through extensive simulation tests incorporating parametric uncertainties,

abrupt load disturbances, and variable operational conditions, the Lyapunov-based MRAC consistently emerged as the most robust and precise solution. It exhibited minimal tracking error, smooth parameter adaptation, and superior resilience to system variations. The adaptive mechanism effectively compensated for internal parameter changes and external perturbations, confirming its practical reliability for dynamically uncertain autonomous driving scenarios.

Overall, the methodological contributions of this thesis are significant. A progressive control framework has been developed, combining classical control, intelligent optimization techniques, and realistic motion planning, while extending the study toward adaptive control strategies. This approach provides a structured and scalable way to address control challenges in electric autonomous vehicle systems.

The results demonstrated that while classical and optimization-based fixed-gain controllers achieve satisfactory performance under nominal conditions, adaptive control strategies particularly those based on Lyapunov theory exhibit superior resilience, precision, and reliability in dynamic and uncertain environments, as evidenced through simulation-based analysis.

Looking forward, several promising research directions emerge from this work:

- Real-time implementation of the proposed control strategies to assess their practical applicability.
- Investigating the integration of metaheuristic optimization techniques for offline and online enhancement of adaptive control laws.
- Conducting an in-depth exploration of metaheuristic algorithms, including their theoretical advancements, hybridization strategies, and performance evaluation metrics.

In conclusion, this dissertation makes a robust, comprehensive, and forward-looking contribution to the field of control for electric autonomous vehicles. By bridging classical control, intelligent optimization, and adaptive methodologies within a unified framework, it lays a solid foundation for the development of safer, smarter, and more resilient autonomous transportation systems, offering valuable solutions and directions for future research and practical deployment.

Bibliography

[1] Pantelis Kopelias, Elissavet Demiridi, Konstantinos Vogiatzis, Alexandros Skabardonis, and Vassiliki Zafiropoulou. Connected & autonomous vehicles–environmental impacts–a review. *Science of the total environment*, 712:135237, 2020.

[2] R. Wallace, A. Stentz, C. E. Thorpe, et al. First results in robot road following. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 18–23, California, USA, 1985.

[3] H. R. Moshtaghi, A. T. Eshlaghy, and M. R. Motadel. A comprehensive review on meta-heuristic algorithms and their classification with novel approach. *Journal of Applied Research on Industrial Engineering*, 8(2):151–174, 2021.

[4] R. T. Marler and J. S. Arora. The weighted sum method for multi-objective optimization: new insights. *Structural and Multidisciplinary Optimization*, 41:853–862, 2010.

[5] A. Guendouz and A. Bouhenna. Hyperstability criterion in model reference adaptive control: Enhancing induction motor efficiency for autonomous electric vehicles. In Mustapha Hatti, editor, *IoT-Enabled Energy Efficiency Assessment of Renewable Energy Systems and Micro-grids in Smart Cities*, pages 231–242, Cham, 2024. Springer Nature Switzerland.

[6] Gerard Rosich. *Autonomy In and Between Polities: A Political Philosophy of Modernity.* PhD thesis, University of Barcelona, 2015.

[7] Antonio Alonso-Cepeda, Raquel Villena-Ruiz, Andrés Honrubia-Escribano, and Emilio Gómez-Lázaro. A review on electric vehicles for holistic robust integration in cities: History, legislation, meta-analysis of technology and grid impact. *Applied Sciences*, 14(16):7147, 2024.

[8] VĂSCAN Iulia and SZABÓ Loránd. A brief history of electric vehicles. *Journal of Computer Science & Control Systems*, 15(1), 2022.

[9] Jose A. Sanguesa, Victor Torres-Sanz, Pablo Garrido, Francisco J. Martinez, and Johann M. Marquez-Barja. A review on electric vehicles: Technologies and challenges. *Smart Cities*, 4(1):372–404, 2021.

[10] Thomas Gillespie. *Fundamentals of vehicle dynamics*. SAE international, 2021.

[11] Jinyong Kim et al. Decoupled longitudinal and lateral vehicle control based autonomous lane change system adaptable to various surrounding conditions. *IEEE Access*, 9:4315–4331, 2021.

[12] A. Artuñedo, J. Villagra, and J. Godoy. Jerk-limited time-optimal speed planning for arbitrary paths. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):8194–8208, 2022.

[13] Yongkang Lu, Xuanzhe Zhang, Xiaosong Hu, Jian Wu, and Jiaqing Duan. A real-time decoupling trajectory planning method for on-road autonomous driving. *IET Control Theory & Applications*, 17(13):1800–1812, 2023.

[14] Saman M. Almufti, Awaz Ahmad Shaban, Rasan Ismael Ali, and Jayson A. Dela Fuente. Overview of metaheuristic algorithms. *Polaris Global Journal of Scholarly Research and Trends*, 2(2):10–32, 2023.

[15] Ahmed M. Nassef, Mahmoud A. Abdelkareem, Hossam M. Maghrabie, and Ali Baroutaji. Metaheuristic-based algorithms for optimizing fractional-order controllers—a recent, systematic, and comprehensive review. *Fractal and Fractional*, 7(7):553, 2023.

[16] Stephen Oladipo, Yanxia Sun, and Zenghui Wang. Optimization of pid controller with metaheuristic algorithms for dc motor drives: Review. *International Review of Electrical Engineering (IREE)*, 15(5):352–381, 2020.

[17] José Eugenio Naranjo, Francisco Serradilla, and Fawzi Nashashibi. Speed control optimization for autonomous vehicles with metaheuristics. *Applied Sciences*, 10(6):2215, 2020.

[18] Bruno Leandro Galvão Costa, Clayton Luiz Graciola, Bruno Augusto Angélico, Alessandro Goedtel, and Marcelo Favoretto Castoldi. Metaheuristics optimization applied to pi controllers tuning of a dtc-svm drive for three-phase induction motors. *Applied Soft Computing*, 62:776–788, 2018.

[19] F Oudjama, A Boumediene, K Saidi, and D Boubekeur. Robust speed control in nonlinear electric vehicles using h-infinity control and the lmi approach. *Journal of Intelligent Systems and Control*, 2(3):170–182, 2023.

[20] Yong Zhang, Kangting Liu, Feng Gao, and Fengkui Zhao. Research on path planning and path tracking control of autonomous vehicles based on improved apf and smc. *Sensors*, 23(18):7918, 2023.

[21] Raghavv Goel and Sayan Basu Roy. Closed-loop reference model based distributed model reference adaptive control for multi-agent systems. In *2021 American Control Conference (ACC)*, pages 1082–1087. IEEE, 2021.

[22] Zhaolun Li. *Autonomous Lateral Maneuvers for Self-driving Vehicles in Complex Traffic Environment*. PhD thesis, Loughborough University, 2023.

[23] Madhav Kumar, Kaibalya Prasad Panda, Ramasamy T Naayagi, Ritula Thakur, and Gayadhar Panda. Comprehensive review of electric vehicle technology and its impacts: Detailed investigation of charging infrastructure, power management, and control techniques. *Applied Sciences*, 13(15):8919, 2023.

[24] Mehrdad Ehsani, Yimin Gao, Stefano Longo, and Kambiz Ebrahimi. *Modern electric, hybrid electric, and fuel cell vehicles*. CRC press, 2018.

[25] L.Arturo Torres-Romero, Luis González-Jiménez, and Riemann Ruiz-Cruz. Dc electric machines review and modeling for electric vehicles part i, 06 2020.

[26] Muhammed A Ibrahim and Ahmed Nasser B Alsammak. Switched reluctance motor drive challenges-a review. *Al-Iraqia Journal for Scientific Engineering Research*, 2(4):9–23, 2023.

[27] Hamidreza Heidari, Anton Rassõlkin, Ants Kallaste, Toomas Vaimann, Ekaterina Andriushchenko, Anouar Belahcen, and Dmitry V Lukichev. A review of synchronous reluctance motor-drive advancements. *Sustainability*, 13(2):729, 2021.

[28] KW Sun, TJ Summers, and CE Coates. Ironless axial flux permanent magnet motor control with multilevel cascaded h-bridge converter for electric vehicle applications. In *2016 IEEE 2nd Annual Southern Power Electronics Conference (SPEC)*, pages 1–6. IEEE, 2016.

[29] Hicham El Hadraoui, Mourad Zegrari, Ahmed Chebak, Oussama Laayati, and Nasr Guennouni. A multi-criteria analysis and trends of electric motors for electric vehicles. *World Electric Vehicle Journal*, 13(4):65, 2022.

[30] Milan Todorovic, Milan Simic, and Arun Kumar. Managing transition to electrical and autonomous vehicles. *Procedia computer science*, 112:2335–2344, 2017.

[31] Patrick M Bösch. *Autonomous Vehicles-The next Revolution in Mobility*. PhD thesis, ETH Zurich, 2018.

[32] Henry Alexander Ignatious, Manzoor Khan, et al. An overview of sensors in autonomous vehicles. *Procedia Computer Science*, 198:736–741, 2022.

[33] Dan B. Marghitu, Bogdan O. Ciocirlan, and Cristian I. Diaconescu. 2 - dynamics. In Dan B. Marghitu, editor, *Mechanical Engineer's Handbook*, Academic Press Series in Engineering, pages 51–118. Academic Press, San Diego, 2001.

[34] K. Zhao et al. Research for nonlinear model predictive controls to laterally control unmanned vehicle trajectory tracking. *Applied Sciences*, 10(17), 2020.

[35] R. Rajamani. *Fundamentals of Vehicle Dynamics*. SAE International, 2021.

[36] Shoaib Azam, Farzeen Munir, and Moongu Jeon. Dynamic control system design for autonomous car. In *VEHITS*, pages 456–463, 2020.

[37] Mike Blundell and Damian Harty. Chapter 5 - tyre characteristics and modelling. In Mike Blundell and Damian Harty, editors, *The Multibody Systems Approach to Vehicle Dynamics (Second Edition)*, pages 335–450. Butterworth-Heinemann, Oxford, second edition edition, 2015.

[38] Torsten Kröger and Torsten Kröger. *Literature survey: Trajectory generation in and control of robotic systems*. Springer, 2010.

[39] Victor M Arevalo, A Opticos, and BC Ensenada. Sinusoidal velocity profiles for motion control. *Proceedings of the ASPE Control of Precision Systems, Philadelphia, PA, USA*, pages 18–20, 2001.

[40] International Organization for Standardization. *ISO 22737:2021 - Intelligent transport systems — Low-speed automated driving (LSAD) systems for predefined routes — Performance requirements, system requirements and performance test procedures*. International Organization for Standardization, 2021.

[41] International Organization for Standardization. *ISO 2631-1:1997 - Mechanical vibration and shock — Evaluation of human exposure to whole-body vibration — Part 1: General requirements*. International Organization for Standardization, 1997.

[42] Hong-Jun Heo, Yungdeug Son, and Jang-Mok Kim. A trapezoidal velocity profile generator for position control using a feedback strategy. *Energies*, 12(7):1222, 2019.

[43] Yi Fang, Guo-Niu Zhu, Yudi Zhao, and Chaochen Gu. Design procedure for motion profiles with sinusoidal jerk for vibration reduction. *Applied Sciences*, 13(24):13320, 2023.

[44] Aniket B Kabde and A Dominic Savio. Position control of stepping motor. *Int J Adv Res Electr Electron Instrum Eng*, 3(4):8974–8981, 2014.

[45] Michel Gendreau and Jean-Yves Potvin, editors. *Handbook of Metaheuristics*. Springer, Cham, Switzerland, 3 edition, 2019.

[46] Kenneth Sörensen, Marc Sevaux, and Fred Glover. *A History of Metaheuristics*, pages 791–808. Springer International Publishing, 2018.

[47] Trinav Bhattacharyya, Bitanu Chatterjee, Pawan Kumar Singh, Jin Hee Yoon, Zong Woo Geem, and Ram Sarkar. Mayfly in harmony: A new hybrid meta-heuristic feature selection algorithm. *IEEE Access*, 8:2031718, 2020.

[48] Annu Lambora, Kunal Gupta, and Kriti Chopra. Genetic algorithm- a literature review. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pages 380–384.

[49] Y. Kebbati, N. Ait-Oufroukh, V. Vigneron, D. Ichalal, and D. Gruyer. Optimized self-adaptive pid speed control for autonomous vehicles. In *2021 26th International Conference on Automation and Computing (ICAC)*, pages 1–6, Portsmouth, United Kingdom. IEEE.

[50] J.-S. Chou and D.-N. Truong. A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean. *Applied Mathematics and Computation*, 389:125535, 2020.

[51] Jui-Sheng Chou and Asmare Molla. Recent advances in use of bio-inspired jellyfish search algorithm for solving optimization problems. *Scientific Reports*, 12(1):19157, 2022.

[52] Ali Ramezani, Seyed Mohammad Ali Khatami Firouzabadi, and Maghsoud Amiri. A review of methods for estimating coefficients of objective functions and constraints in mathematical programming models. *International Journal of Nonlinear Analysis and Applications*, 15(7):325–336, 2024.

[53] Nyoman Gunantara. A review of multi-objective optimization: Methods and its applications. *Cogent Engineering*, 5(1):1502242, 2018.

[54] S. Singh, V. Singh, A. Rani, and J. Yadav. Optimization of PID controller based on various tuning methods. In *2023 International Conference on Power, Instrumentation, Energy and Control (PIECON)*, pages 1–6, Aligarh, India, 2023. IEEE.

[55] Asmaa Guendouz, Mustapha Hatti, Abdelhalim Tlemçani, and Hind Saidani-Scott. Longitudinal control of autonomous electric vehicles via genetic algorithm tuned pid and lyapunov-based adaptive control: A robustness evaluation under dynamic uncertainties. *Journal of Electrical Systems*, 21(01):28–41, 2025.

[56] Anuradha M Annaswamy and Alexander L Fradkov. A historical perspective of adaptive control and learning. *Annual Reviews in Control*, 52:18–41, 2021.

[57] Thomas Chaffre, Julien Moras, Adrien Chan-Hon-Tong, Julien Marzat, Karl Sammut, Gilles Le Chenadec, and Benoit Clement. Learning-based vs model-free adaptive control of a mav under wind gust. In *Informatics in Control, Automation and Robotics: 17th International Conference, ICINCO 2020 Lieusaint-Paris, France, July 7–9, 2020, Revised Selected Papers*, pages 362–385. Springer, 2022.

[58] Ioan Doré Landau, Rogelio Lozano, Mohammed M'Saad, and Alireza Karimi. *Adaptive control: algorithms, analysis and applications*. Springer Science & Business Media, 2011.

[59] Zelmat M. *Automatisation des processus industriels Tome2,Commande modale et adaptative*. Office des Publications Universitaires,Algérie, 2001.

[60] R Rajesh and SN Deepa. Design of direct mrac augmented with 2 dof pidd controller: An application to speed control of a servo plant. *Journal of King Saud University-Engineering Sciences*, 32(5):310–320, 2020.

[61] Guy Albert Dumont and Mihai Huzmezan. Concepts, methods and techniques in adaptive control. In *Proceedings of the 2002 American control conference (IEEE Cat. No. CH37301)*, volume 2, pages 1137–1150. IEEE, 2002.

[62] Nhan T Nguyen and Nhan T Nguyen. *Model-reference adaptive control*. Springer, 2018.

[63] I Landau. A generalization of the hyperstability conditions for model reference adaptive systems. *IEEE Transactions on Automatic Control*, 17(2):246–247, 1972.

[64] Donghao Sui and Yulai Zhang. Analyzation of the application scenarios of ackerman geometry based on vehicle steering model. *Highlights in Science, Engineering and Technology*, 46, 2023.

Appendix A

S-function

S-Function Implementation for the Kinematic Bicycle Model

This appendix provides the MATLAB S-function code used to implement the kinematic bicycle model in Simulink. The code allows for efficient customization and simulation of the model.

Listing A.1: MATLAB S-Function for the Kinematic Bicycle Model

```
function [sys,x0,str,ts,simStateCompliance] = bm(t,x,u,flag)

switch flag
    % Initialization
    case 0
        [sys,x0,str,ts,simStateCompliance]=mdlInitializeSizes;
    % Derivatives
    case 1
        sys=mdlDerivatives(t,x,u);
    % Update
    case 2
        sys=mdlUpdate(t,x,u);
    % Outputs
    case 3
        sys=mdlOutputs(t,x,u);
```

```
    % GetTimeOfNextVarHit
    case 4
        sys=mdlGetTimeOfNextVarHit(t,x,u);
    % Terminate
    case 9
        sys=mdlTerminate(t,x,u);
    % Unexpected flags
    otherwise
        DAStudio.error('Simulink:blocks:unhandledFlag', num2str(flag));
end

% Initialization Function
function [sys,x0,str,ts,simStateCompliance]=mdlInitializeSizes
sizes = simsizes;
sizes.NumContStates  = 3;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 3;
sizes.NumInputs      = 2;
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = [114.0758 131.2616 0.33];
str = [];
ts  = [0 0];
simStateCompliance = 'UnknownSimState';

% Derivatives Function
function sys=mdlDerivatives(t,x,u)
L = 2.96;
xd1 = u(1)*cos(x(3));
xd2 = u(1)*sin(x(3));
xd3 = (u(1)/L) * tan(u(2));
sys = [xd1 xd2 xd3];
```

```
% Update Function
function sys=mdlUpdate(t,x,u)
sys = [];

% Outputs Function
function sys=mdlOutputs(t,x,u)
sys = [x(1) x(2) x(3)];

% GetTimeOfNextVarHit Function
function sys=mdlGetTimeOfNextVarHit(t,x,u)
sampleTime = 1; % Example: set the next hit to be one second later.
sys = t + sampleTime;

% Terminate Function
function sys=mdlTerminate(t,x,u)
sys = [];
```

Appendix B

Jellyfish Algorithm

JS PID Tuning Pseudo-code

1. Begin
2. Initialize PID parameters
3. Calculate objective function $f(Z)$
4. Set population size $n_{pop} = 100$, maximum iteration $I_{max} = 50$
5. Initialize population of jellyfish Z_i $(i = 1, 2, \ldots, n_{pop})$ using logistic chaotic map
6. Calculate food quantity at each Z_i, $f(Z_i)$
7. Identify jellyfish with the highest food quantity (Z^*)
8. Initialize time $t = 1$
9. Repeat:
   (a) For each jellyfish $i = 1$ to n_{pop}:
   - Calculate time control function $c(t)$
   - If $c(t) \geq 0.5$, pursue ocean current:
     - Determine ocean current
     - Calculate new location
   - Else, with probability $(1 - c(t))$:
     - Passive motion: update location

  - Active motion: calculate direction and update location
- Perform boundary check and recalculate food quantity
- Update positions (Z_i, Z^*)

(b) Update time $t \leftarrow t + 1$

10. Until stop criterion is met
11. Output: convergence iteration, best results K_P, K_I, K_D, and optimal objective value
12. End