MANCHESTER
1824
The University of Manchester

Formation Control of Nonholonomic Wheeled Mobile Robots Using Consensus-Based Distributed Model Predictive Control and Input-Output Linearisation

Third Year Individual Project - Final Report

Year of submission
2025

Student ID
10949254

School of Engineering

Contents

Word count: 10366

List of figures

List of tables

List of Acronyms

Acronym	Definition
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
AGV	Automated Guided Vehicle
AUV	Autonomous Underwater Vehicles
MRS	Multi-robot Systems
WMRS	Wheeled Mobile Robots
NMPC	Nonlinear Model Predictive Control
DLPC	Distributed Learning-based Predictive Control
CLF	Control Lyapunov Function
DTCBF	Discrete-time Control Barrier Function
NSB	Null-Space-based Behavioural
APF	Artificial Potential Fields
MPC	Model Predictive Control
MPOMDPs	Multi-agent Partially Observable Markov Decision Processes
MHE	Moving Horizon Estimation
DEKF	Delayed Extended Kalman Filter
QP	Quadratic Programming
ECBF	Exponential Control Barrier Function
CAGR	Compound Annual Growth Rate
IAE	Integral Absolute Error

Abstract

This dissertation addresses distributed formation control of nonholonomic wheeled mobile robots (WMRs), with applications in autonomous vehicles, swarm robotics, and industrial automation. A consensus-based nonlinear model predictive control (NMPC) framework is proposed, enhanced through input-output linearisation and integrated with discrete-time control barrier functions (DTCBFs) for safety. The framework ensures real-time trajectory tracking, obstacle avoidance, and collision-free operation while maintaining formation geometry.

Mathematical modelling captures the nonholonomic dynamics of individual WMRs. A distributed consensus algorithm enables coordination without central control, while DTCBFs enforce safety constraints throughout motion. The NMPC controller generates optimal control inputs over a predictive horizon, balancing tracking precision and constraint satisfaction.

The proposed approach is evaluated through simulations across linear and curved trajectories in dynamic environments. Results demonstrate strong formation-keeping performance, responsiveness to disturbances, and reliable obstacle clearance. Comparative analysis with a distributed learning-based predictive control (DLPC) method highlights that while DLPC offers lower computational load, the consensus-based NMPC delivers superior formation accuracy and formal safety guarantees.

This work contributes to the theoretical foundation of distributed multi-robot control by unifying consensus theory, NMPC, and barrier functions in a single framework. It also provides practical insights for deploying autonomous robot teams in complex, real-world environments.

Declaration of originality

I hereby confirm that this dissertation is my own original work unless referenced clearly to the contrary, and that no portion of the work referred to in the dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Intellectual property statement

i The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights (the “Copyright”), and he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made *only* under the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, per licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii The ownership of certain Copyright, patents, designs, trademarks and other intellectual property (the “Intellectual Property”) and any reproductions of copyright works in the thesis, for example, graphs and tables (“Reproductions”), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see `http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=24420`), in any relevant Dissertation restriction declarations deposited in the University Library, and The University Library’s regulations (see `http://www.library.manchester.ac.uk/about/regulations/_files/Library-regulations.pdf`).

Acknowledgements

I would like to thank my mother—without her unwavering support and belief in me, I would not have reached this stage.

I am deeply grateful to my supervisor, Dr. Joaquin Carrasco, for his outstanding guidance and support throughout the past year.

I also want to thank Dr. Wei Pan for generously granting me access to his lab’s resources, which were invaluable to my research.

Finally, I would like to thank Weishu Zhan for the insightful discussions on advanced control techniques, which greatly enriched my work.

1 Introduction

1.1 Background and motivation

The formation control of multiple robots has garnered significant research attention due to its wide-ranging applications in autonomous vehicles, swarm robotics, and industrial automation [1], [2]. Multi-robot systems are being increasingly deployed in complex, dynamic environments where cooperation and coordination are essential. Achieving and maintaining a desired formation while ensuring global system stability is critical in applications such as unmanned aerial vehicle (UAV) swarms, unmanned ground vehicle (UGV) formations, and industrial robotic fleets [3], [4].

According to [5], ”Global factory automation & industrial controls market size was USD 124.3 Billion in 2024, and the market is projected to touch 178.84 Billion by 2033, exhibiting a CAGR (Compound Annual Growth Rate) of 4.1% during the forecast period.” This growth highlights the increasing reliance on autonomous robotic systems for industrial automation.

Various industries, including logistics, agriculture, and defence, leverage formation control for tasks such as autonomous convoys, drone swarms, and robotic surveillance. For example, Amazon warehouses utilise fleets of Automated Guided Vehicles (AGVs) to optimise inventory movement and reduce human workload [6]. Precise formation maintenance is essential for coordinated sensing and communication in satellite constellations [7]. Similarly, Autonomous Underwater Vehicles (AUVs) rely on formation control for efficient deep-sea exploration and environmental monitoring [8]. In industrial settings, consensus-based control strategies enable AGV fleets to synchronise warehouse logistics and streamline production lines, significantly improving operational efficiency.

However, real-time formation control presents several challenges that must be addressed to enable robust and scalable deployment. A significant difficulty arises from time delays in wireless communication networks, which can degrade system performance and stability [9]. Existing consensus algorithms often struggle with scalability due to increasing communication overhead and computational complexity [8]. Additionally, conventional delay-compensation strategies may not be well-suited for real-time applications with unpredictable network latencies. Furthermore, real-world disturbances such as sensor noise, actuator failures, and dynamic obstacles necessitate robust and adaptive control strategies [10]. Safe coordination in uncertain environments is critical, particularly in human-robot interactions and hazardous operational conditions.

To address these limitations, advanced control frameworks incorporating predictive modelling, distributed optimisation, and robustness to uncertainties are required [11]. By developing scalable and delay-tolerant formation control strategies, multi-robot systems can achieve higher efficiency, adaptability, and resilience in real-world applications.

1.2 Aims and objectives

This project aims to contribute to the development of safe and adaptable navigation in multi-robot systems capable of handling unforeseen obstacles and dynamically changing conditions. The goal is to ensure the safety and reliability of future autonomous vehicles in challenging environments. Specifically, the project will develop a consensus-based, nonlinear model predictive formation controller implementation using input-output linearisation, evaluate its suitability for the desired application, and compare its performance to a recently developed, computationally fast and efficient distributed learning-based predictive control framework that integrates a receding horizon optimisation strategy into policy updates [12]. The main objectives of the project are summarised below.

- To develop a mathematical model of the wheeled mobile robot, considering its kinematics using nonholonomic constraints and dynamics using the Lagrangian formulation.
- To develop a consensus algorithm based on nonlinear model predictive control (NMPC) for path planning and formation control using input-output linearisation.
- To enforce safety formally in the control problem using Discrete-time Control Barrier Functions.
- To evaluate the performance of the consensus algorithm by comparing its performance with a distributed NMPC implementation without the consensus algorithm.
- To compare the developed consensus-based NMPC implementation using numerical solvers with the DLPC framework [12] for WMRs.
- To assess the trade-offs between safety, computational efficiency, and formation control in the integrated framework.

In this project, the collective goal is to navigate a terrain of obstacles while maintaining an arbitrary geometric formation.

1.3 Report structure

This report is organised as follows. Section 2 reviews existing literature on the formation control of multi-robot systems (MRS). Section 3 discusses preliminary concepts fundamental to the report. The project's methodology is considered in Section 4. The simulation results and a discussion on the performance evaluation are presented in Section 5. Finally, the conclusion and a consideration of future research avenues can be found in Section 6.

2 Literature review

The formation control of multiple mobile robot systems has wide applications in both military and civilian fields [13], [14], including target tracking, cooperative search and rescue, exploration, distributed manipulation, and patrol [15]. In such missions, a group of robots offers several advantages over a single robot, including increased efficiency, adaptability, and robustness [14], ultimately saving time and reducing operational costs [13].

Formation control involves multiple agents moving toward a specific target while maintaining a predetermined geometric shape relative to each other and adapting to environmental constraints such as obstacles and narrow passages [16]. Formation control methods can be categorised in various ways. One categorisation, based on sensing capabilities, includes position-based, displacement-based, and distance-based control [4]. Additionally, modern formation control strategies can be broadly grouped into four categories: the leader-follower paradigm [8], virtual structure methodology [17], consensus-based strategies [18], and behaviour-based methods [19], each with its strengths and limitations.

The leader-follower method, one of the earliest and most straightforward approaches to formation control, establishes hierarchical relationships where followers track the leader robots' positions. This method is advantageous due to its simplicity, lower computational load, and ease of real-time implementation. However, the leader represents a single point of failure; if it fails or deviates from the expected trajectory, the formation will break. The method also suffers from accumulated tracking errors in multi-robot chains, and it is difficult to maintain precise inter-robot spacing. Recent improvements, such as the use of an embedded control technique for leader-follower formation control in wheeled mobile robots (WMRs) [8], ensure robustness despite disturbances. Furthermore, an extended state observer has been applied to leader-following formation control to compensate for inaccuracies in global positioning systems [14], improving real-time tracking for nonholonomic robots. Despite these improvements, a significant drawback remains: if the leader fails, the entire formation collapses. Some recent work, such as the integrated controller combining sliding mode control and a force function for obstacle avoidance proposed by [16], has attempted to mitigate these issues by incorporating additional controllers.

The second method of formation control, virtual structure control, is based on the concept of a virtual rigid body whose centre of mass is tracked to ensure precise formation maintenance. This approach does not require the selection of a leader and offers an intuitive way to define the formation. Robots maintain fixed positions relative to a moving reference frame, which is ideal for assembly tasks requiring rigid formations. However, when one of the robots experiences a disturbance, the formation can break apart, and the method often requires global information, reducing scalability and increasing communication demands. A high-precision formation control method for mobile robots using the virtual structure framework has been proposed [17], where robots track predefined formation trajectories with minimal deviation. This approach has demonstrated fault-tolerant movement with graceful performance degradation under disturbances.

Behaviour-based control focuses on creating complex behaviours from simpler, modular actions, drawing inspiration from biological systems to promote adaptability and robustness in dynamic environments. One example is the Null-Space-based Behavioural (NSB) control method [20], which integrates multiple prioritised behaviours using a projection mechanism, ensuring that lower-priority behaviours do not interfere with higher-priority ones. This strategy facilitates coordinated control in multi-robot systems, especially in complex environments. Another example is a behaviour-based formation control strategy that enables multi-robot teams to maintain their formations while navigating and avoiding obstacles [21].

Another popular method is consensus-based control, a distributed approach that allows robots to agree on their velocities and positions through local communication without requiring centralised control. It is a subset of graph-theoretical approaches where robot interactions are modelled through directed/undirected graphs, which enable formal stability analysis. One of the key advantages of consensus-based methods is their ability to handle disturbances and input constraints effectively. For example, a consensus formation control strategy for nonholonomic mobile robots ensures robust tracking performance despite mixed disturbances and input constraints [9]. Consensus-based control has been extended by designing a control law specifically for nonholonomic robots using Artificial Potential Fields (APFs) and providing a stability analysis to guarantee convergence [22]. Furthermore, a consensus-based formation control approach incorporates time synchronisation to address the challenges posed by communication delays, ensuring coordinated motion in distributed systems [18].

Learning-based control is a more recent strategy for formation control. It integrates advanced algorithms like reinforcement learning and actor-critic networks with model predictive control (MPC) to overcome multi-robot systems' scalability, safety, and uncertainty challenges. A recent learning-based approach is to use a DLPC framework that leverages fast policy learning to manage scalability in multi-robot formations, enabling real-time coordination with minimal computational effort [12]. In a different approach, the policy synthesis process for multi-agent partially observable Markov decision processes (MPOMDPs) embeds discrete-time barrier functions to ensure that safety constraints are met despite system uncertainties [23]. Additionally, a method enhances sample efficiency and mitigates uncertainty within the context of learning-based MPC for aerial robots, improving the robustness and performance of the system [24].

The development of formation control methods for multi-robot systems has made significant progress over the years, with each approach offering unique advantages and addressing different challenges. While leader-follower and virtual structure methods are well-established, more recent approaches, such as consensus-based and behaviour-based control, provide decentralised, scalable, and adaptable solutions particularly suited for dynamic and uncertain environments.

Despite the advances in these methods, challenges remain, especially in environments with communication delays or unpredictable disturbances. For example, a moving horizon estimation (MHE) method using a delayed extended Kalman filter (DEKF)-based update law has been developed to accurately estimate the positions of unmanned underwater vehicles in the presence of delays [25], highlighting the need for robust estimation and control techniques when implementing network topologies on real systems.

In this report, the focus is placed on the problem of trajectory tracking for multiple WMRs while maintaining the desired geometric formation and avoiding obstacles. The consensus-based control strategy helps to guarantee formation stability, while the trajectory tracking problem is addressed using an NMPC-β controller [11]. NMPC-β is chosen because it ensures stability by incorporating a terminal cost derived from a CLF, eliminating the need for explicit terminal constraints while maintaining recursive feasibility and real-time implementability.

3 Preliminaries

This section briefly reviews the formal definitions of Control Lyapunov Functions (CLFs), Discrete-time Control Barrier Functions (DTCBFs), Input-Output Linearisation, and Algebraic Graph Theory, which serve as the foundation for the consensus and nonlinear control strategies discussed in Section 4. CLFs are used as the terminal cost in the NMPC-β formulation, and DTCBFs are used to enforce safety in the formation control and trajectory tracking problems.

3.1 Mathematical Notation

This section defines the key mathematical symbols, sets, and functions used throughout the report for clarity and consistency. The notation is designed to provide a rigorous framework for understanding the mathematical formulations and algorithms presented.

3.1.1 General Symbols

- $\mathbb{R}$: The set of real numbers.
- $\mathbb{R}_+ = \{x \in \mathbb{R} \ \mid x > 0\}$: The set of positive real numbers.
- $\mathbb{Z}$: The set of integers.
- $\mathbb{N}$: The set of natural numbers.
- $\mathbb{R}^n$: The n-dimensional Euclidean space.
- $\mathbb{R}^{n \times m}$: The space of $n \times m$ real matrices.
- $\mathbb{S}^n_{++} \subset \mathbb{R}^{n \times n}$: the set of positive-definite $n \times n$ real matrices.

3.1.2 Vectors and Matrices

- **Boldface symbols**: Represent vectors or matrices.
  - Example: $\mathbf{x} = [x_1, x_2, \ldots, x_n]^\top$ is a vector in $\mathbb{R}^n$.
- $\mathbf{I}_n$: the $n \times n$ identity matrix.
- $\mathbf{0}_{n \times m}$: the $n \times m$ zero matrix.

3.1.3 Functions and Operators

- diag($\mathbf{a}$): A diagonal matrix with the elements of vector $\mathbf{a} = [a_1, a_2, \ldots, a_n]^\top$ as its diagonal entries.
  - Example: For $\mathbf{a} = [1, 2, 3]^\top$, diag($\mathbf{a}$) $= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$.
- $\| \cdot \|_p$: The p-norm of a vector or matrix.
  - Example: for a vector ($\mathbf{x}$) $= [x_1, x_2, \ldots, x_n]^\top$, the Euclidean norm is $\|\mathbf{x}\|_2 = (\sum_{i=1}^n x_i^2)^{1/2}$.
- Int($\mathcal{S}$): The *interior* of the set $\mathcal{S} \subseteq \mathbb{R}^n$, defined as the set of all points $x \in \mathcal{S}$ for which there exists an open ball $B_\epsilon(x) = \{y \in \mathbb{R}^n \mid \|y - x\|_2 < \epsilon\}$ entirely contained in $\mathcal{S}$.
- $\partial\mathcal{S}$: The *boundary* of the set $\mathcal{S} \subseteq \mathbb{R}^n$, defined as the set of points $x \in \mathbb{R}^n$ such that every open ball $B_\epsilon(x)$ contains both points in $\mathcal{S}$ and in its complement $\mathbb{R}^n \setminus \mathcal{S}$.

3.2 Control Lyapunov Functions

Consider a state space $\mathbb{X} \subset \mathbb{R}^n$ and a control input space $\mathbb{U} \subset \mathbb{R}^m$, where $\mathbb{X}$ is assumed to be path-connected and contains the origin in its interior. The dynamics of a nonlinear control-affine system are given by:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \tag{1}$$

where $\mathbf{x} \in \mathbb{X}$, $\mathbf{u} \in \mathbb{U}$, and $\mathbf{f} : \mathbb{X} \to \mathbb{R}^n$ and $\mathbf{g} : \mathbb{X} \to \mathbb{R}^{n \times m}$ are Lipschitz continuous on $\mathbb{X}$.

Assumption 1. *Assume that the origin, i.e., $\mathbf{f}(0) = 0$, is an equilibrium point of the system.*

To analyse system stability, the concept of class $\mathcal{K}$ functions is introduced, which are commonly used to express stability conditions.

Definition 1. *(Class $\mathcal{K}$ Function [11]). A continuous function $\alpha : [0, a) \to \mathbb{R}_+$, with $a > 0$, belongs to class $\mathcal{K}$ if it satisfies:*

1. $\alpha(0) = 0$,

2. $\alpha(r)$ is strictly monotonically increasing for all $r \in [0, a)$.

If $a = \infty$ and $\lim_{r \to \infty} \alpha(r) = \infty$, then α belongs to the class $\mathcal{K}_\infty$.

A class $\mathcal{K}$ function can be interpreted as a nonlinear gain function, where larger values of r result in larger function values. A simple example is the linear function $\alpha(r) = \gamma r$, with $\gamma > 0$, which satisfies the definition.

Given this, CLFs, which belong to $\mathcal{K}_\infty$, are defined as introduced in [26].

Definition 2. *(Control Lyapunov Functions [26]). A continuously differentiable function $V : \mathbb{X} \to \mathbb{R}_+$ is a CLF for system (1) if there exist class $\mathcal{K}_\infty$ functions $\alpha_1, \alpha_2, \alpha_3$ such that for all $\mathbf{x} \in \mathbb{X}$:*

$$\begin{aligned} \alpha_1(\|\mathbf{x}\|) \leq V(\mathbf{x}) \leq \alpha_2(\|\mathbf{x}\|), \\ \inf_{\mathbf{u} \in \mathbb{U}} \dot{V}(\mathbf{x}, \mathbf{u}) \leq -\alpha_3(\|\mathbf{x}\|). \end{aligned} \tag{2}$$

The existence of a CLF ensures the possibility of designing a continuous (except potentially at $\mathbf{x} = 0$) state-feedback controller $\mathbf{k} : \mathbb{X} \to \mathbb{U}$ that guarantees global asymptotic stability of the origin [26]. If the functions $\alpha_1, \alpha_2, \alpha_3$ take the form $\alpha_i(r) = c_i r^2$, with $c_i > 0$, then the system exhibits global exponential stability, with the state norm satisfying:

$$\|\mathbf{x}(t)\| \leq M \|\mathbf{x}(0)\| e^{-\gamma t}, \tag{3}$$

for some constants $M, \gamma \in \mathbb{R}_+$.

Furthermore, given a CLF, the set of admissible control inputs that stabilise the system is:

$$\mathcal{U}_{\text{CLF}}(\mathbf{x}) = \{\mathbf{u} \in \mathbb{U} \mid \dot{V}(\mathbf{x}, \mathbf{u}) \leq -\alpha_3(\|\mathbf{x}\|)\}. \tag{4}$$

A feedback controller $\mathbf{k}(\mathbf{x})$ is then defined such that $\mathbf{k}(\mathbf{x}) \in \mathcal{U}_{\text{CLF}}(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{X}$, ensuring stability.

Theorem 1. *(Stability via CLFs). Let $V : \mathbb{X} \to \mathbb{R}_+$ be a continuously differentiable CLF satisfying the conditions in Definition 2. If there exists a locally Lipschitz continuous control law $\mathbf{k} : \mathbb{X} \to \mathbb{U}$ such that:*

$$\dot{V}(\mathbf{x}, \mathbf{k}(\mathbf{x})) \leq -\alpha_3(\|\mathbf{x}\|), \tag{5}$$

for all $\mathbf{x} \in \mathbb{X}$, then:

1. *The origin $\mathbf{x} = 0$ is globally asymptotically stable,*
2. *$\alpha_1,\ \alpha_2,\ \alpha_3$ are quadratic functions, then the system is globally exponentially stable.*

Importantly, the set $\mathcal{U}_{\mathsf{CLF}}(\mathbf{x})$ is described by an affine inequality in $\mathbf{u}$ due to the control-affine structure of the system dynamics:

$$\dot{V}(\mathbf{x}, \mathbf{u}) = (\nabla_{\mathbf{x}} V)^\top \dot{\mathbf{x}} = \frac{\partial V}{\partial \mathbf{x}}(\mathbf{x})(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}). \tag{6}$$

This structure allows for the formulation of an optimisation-based control strategy using quadratic programming [11]. A standard quadratic program for determining the control input $\mathbf{k}(\mathbf{x})$ is given by:

$$\begin{aligned} \mathbf{k}(\mathbf{x}) = \arg\min_{\mathbf{u}\in\mathbb{U}} \ & \frac{1}{2}\mathbf{u}^\top \mathbf{H}\mathbf{u} + \mathbf{f}^\top \mathbf{u} + C \\ \text{s.t.} \ & \frac{\partial V}{\partial \mathbf{x}}(\mathbf{x})(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}) \leq -\alpha_3(\|\mathbf{x}\|), \end{aligned} \tag{7}$$

where $\mathbf{H}$ is positive definite, C is a constant independent of $\mathbf{u}$, and $\mathbb{U}$ is assumed to be a convex polytope. The feasibility of this optimisation problem is directly guaranteed by the CLF condition (2).

3.3 Discrete-time Control Barrier Functions

Consider the discrete-time system:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \tag{8}$$

where $\mathbf{x}_k \in \mathbb{X} \subset \mathbb{R}^n$ is the system state at time step $k \in \mathbb{N}$, $\mathbf{u}_k \in \mathbb{U} \subset \mathbb{R}^m$ is the control input, and $\mathbf{f} : \mathbb{X} \times \mathbb{U} \to \mathbb{X}$ is a continuous function representing the system dynamics.

A key consideration in control design is enforcing safety, ensuring that the state $\mathbf{x}_k$ remains within a safe set $\mathcal{C} \subset \mathbb{X}$ defined as:

$$\begin{aligned} \mathcal{C} &:= \{\mathbf{x} \in \mathbb{X} \ \mid h(\mathbf{x}) \geq 0\}, \\ \mathsf{Int}(\mathcal{C}) &:= \{\mathbf{x} \in \mathbb{X} \ \mid h(\mathbf{x}) > 0\}, \\ \partial\mathcal{C} &:= \{\mathbf{x} \in \mathbb{X} \ \mid h(\mathbf{x}) = 0\}, \end{aligned} \tag{9}$$

where $h : \mathbb{R}^n \to \mathbb{R}$ is a continuous function defining the boundary and interior of $\mathcal{C}$.

Definition 3. *The set $\mathcal{C}$ is forward invariant under* (8) *if, for any initial condition $\mathbf{x}_0 \in \mathcal{C}$, the trajectory satisfies $\mathbf{x}_k \in \mathcal{C}$ for all $k \in \mathbb{N}$.*

Definition 4. *(Discrete-Time Barrier Function [23]). A function $h : \mathbb{R}^n \to \mathbb{R}$ is a discrete-time barrier function (DTBF) for system* (8) *if there exists a class $\mathcal{K}$ function α satisfying $\alpha(r) < r$ for all $r > 0$ such that:*

$$\Delta h(\mathbf{x}_k, \mathbf{u}_k) \geq -\alpha(h(\mathbf{x}_k)), \quad \forall\, \mathbf{x}_k \in \mathbb{X}, \tag{10}$$

where the difference operator is defined as:

$$\Delta h(\mathbf{x}_k, \mathbf{u}_k) = h(\mathbf{x}_{k+1}) - h(\mathbf{x}_k). \tag{11}$$

As shown in [27], if h satisfies (10), then the safe set $\mathcal{C}$ remains forward invariant. A controller can then be designed to satisfy (10), ensuring safety via discrete-time Control Barrier Functions.

Definition 5. *(Discrete-time CBF [23]). A function $h : \mathbb{R}^n \to \mathbb{R}$ is a DTCBF if it is a discrete-time barrier function and can be used to enforce safety constraints via control design.*

Theorem 2. *([23]) Let $\mathcal{C} \subseteq \mathbb{X} \subset \mathbb{R}^n$ be defined as in* (9). *If h is a discrete-time CBF, then any control input $\mathbf{u}_k$ satisfying* (10) *ensures that $\mathcal{C}$ remains forward invariant, thus guaranteeing safety.*

The function α in (10) can be chosen as a linear function $\alpha(h) = \gamma h$ with $0 < \gamma \leq 1$. Substituting this into (10), the following is obtained:

$$h(\mathbf{x}_k) \geq (1-\gamma)^k h(\mathbf{x}_0), \quad \forall\, k \in \mathbb{N}. \tag{12}$$

For $0 < \gamma \leq 1$, this ensures exponential convergence of $h(\mathbf{x}_k)$ towards zero, guaranteeing that the system remains within the safe set $\mathcal{C}$. This formulation corresponds to the discrete-time exponential Control Barrier Function (ECBF), as discussed in [28].

3.4 Input-Output Linearisation

Input-output linearisation is a nonlinear control technique that transforms the output dynamics of a system into a stable linear form [29]. The following formalism is based on standard results in nonlinear control theory from [30].

Consider a configuration space $\mathcal{Q} \subseteq \mathbb{R}^n$ and an input space $\mathcal{U} \subseteq \mathbb{R}^m$.

Assumption 2. *Assume that $\mathcal{Q}$ is a non-empty, path-connected manifold.*

A general control-affine system is described by:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \underbrace{\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q})}_{\mathbf{H}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}} = \mathbf{B}\mathbf{u}, \tag{13}$$

where $\mathbf{q} \in \mathcal{Q}$ represents the generalized coordinates, $\dot{\mathbf{q}} \in \mathbb{R}^n$ are the velocity states, and $\mathbf{u} \in \mathcal{U}$ is the control input. The inertia matrix $\mathbf{M} : \mathcal{Q} \to \mathbb{S}^n_{++}$ is symmetric positive definite, while $\mathbf{C} : \mathcal{Q} \times \mathbb{R}^n \to \mathbb{R}^{n\times n}$ encapsulates Coriolis and centrifugal effects. The gravitational forces are denoted by $\mathbf{G} : \mathcal{Q} \to \mathbb{R}^n$, and the static input matrix $\mathbf{B} \in \mathbb{R}^{n\times m}$ determines actuation constraints.

For a system output defined as $\mathbf{y} : \mathcal{Q} \to \mathbb{R}^k$, with $k \leq m$, the following assumption is imposed:

Assumption 3. *Each output has a relative degree of 2 in a domain $\mathcal{R} \subseteq \mathcal{Q}$ [31].*

Given a desired trajectory $\mathbf{y}_d : \mathcal{I} \to \mathbb{R}^k$ over a time interval $\mathcal{I} = [t_0, t_f]$, where $t_f > t_0$, the tracking error state is defined as:

$$\boldsymbol{\eta}(\mathbf{q}, \dot{\mathbf{q}}, t) = \begin{bmatrix} \mathbf{y}(\mathbf{q}) - \mathbf{y}_d(t) \\ \dot{\mathbf{y}}(\mathbf{q}, \dot{\mathbf{q}}) - \dot{\mathbf{y}}_d(t) \end{bmatrix}, \tag{14}$$

with $\mathbf{r}(t) = \begin{bmatrix} \mathbf{y}_d(t)^\top & \dot{\mathbf{y}}_d(t)^\top \end{bmatrix}^\top$.

Differentiating this equation and substituting the system dynamics produces:

$$\dot{\boldsymbol{\eta}}(\mathbf{q}, \dot{\mathbf{q}}, t) = \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}) - \dot{\mathbf{r}}(t) + \mathbf{g}(\mathbf{q})\mathbf{u}, \tag{15}$$

where the drift and input matrices are given by:

$$\mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} \frac{\partial \mathbf{y}}{\partial \mathbf{q}} \dot{\mathbf{q}} \\ \frac{\partial \dot{\mathbf{y}}}{\partial \mathbf{q}} \dot{\mathbf{q}} - \frac{\partial \mathbf{y}}{\partial \mathbf{q}} (\mathbf{M}^{-1}\mathbf{H}) \end{bmatrix}, \quad \mathbf{g}(\mathbf{q}) = \begin{bmatrix} \mathbf{0}_{k \times m} \\ \frac{\partial \mathbf{y}}{\partial \mathbf{q}} \mathbf{M}^{-1}\mathbf{B} \end{bmatrix}. \tag{16}$$

For all $\mathbf{q} \in \mathcal{Q}$, the relative degree assumption ensures that $\mathbf{g}(\mathbf{q})$ is full rank. Defining the feedback transformation:

$$\mathbf{u}(\mathbf{q}, \dot{\mathbf{q}}, t) = \widetilde{\mathbf{g}}(\mathbf{q})^\dagger \big(-\widetilde{\mathbf{f}}(\mathbf{q}, \dot{\mathbf{q}}) + \ddot{\mathbf{y}}_d(t) + \boldsymbol{\nu}(\mathbf{q}, \dot{\mathbf{q}}, t) \big), \tag{17}$$

$$\widetilde{\mathbf{f}}(\mathbf{q}, \dot{\mathbf{q}}) = \frac{\partial \dot{\mathbf{y}}}{\partial \mathbf{q}} \dot{\mathbf{q}} - \frac{\partial \mathbf{y}}{\partial \mathbf{q}} (\mathbf{M}^{-1}\mathbf{H}), \tag{18}$$

$$\widetilde{\mathbf{g}}(\mathbf{q}) = \frac{\partial \mathbf{y}}{\partial \mathbf{q}} \mathbf{M}^{-1}\mathbf{B}, \tag{19}$$

where $\boldsymbol{\nu} \in \mathbb{R}^k$ is an auxiliary stabilising input, generates a linearised output system:

$$\dot{\boldsymbol{\eta}} = \mathbf{F}\boldsymbol{\eta} + \mathbf{G}\boldsymbol{\nu}, \tag{20}$$

where

$$\mathbf{F} = \begin{bmatrix} \mathbf{0}_{k \times k} & \mathbf{I}_k \\ \mathbf{0}_{k \times k} & \mathbf{0}_{k \times k} \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \mathbf{0}_{k \times k} \\ \mathbf{I}_k \end{bmatrix}. \tag{21}$$

Since $(\mathbf{F}, \mathbf{G})$ form a controllable pair, a feedback gain $\mathbf{K} = \begin{bmatrix} \mathbf{K}_p & \mathbf{K}_d \end{bmatrix}$ can be selected, where $\mathbf{K}_p,\ \mathbf{K}_d \in \mathbb{S}^k_{++}$, to define:

$$\boldsymbol{\nu} = -\mathbf{K}\boldsymbol{\eta}. \tag{22}$$

The resulting closed-loop system is given by:

$$\dot{\boldsymbol{\eta}} = \mathbf{A}_{cl}\boldsymbol{\eta}, \quad \mathbf{A}_{cl} = \mathbf{F} - \mathbf{G}\mathbf{K}, \tag{23}$$

where $\mathbf{A}_{cl}$ is Hurwitz, ensuring exponential stability of $\mathbf{y}_d$. Consequently, a Control Lyapunov Function (CLF) can be constructed using the Continuous Algebraic Riccati Equation (CARE) [29], guaranteeing stability in the control design.

3.5 Algebraic Graph Theory

The following formalism follows from [9]. Interactions among robots are represented using graphs. Let $G = (V, E, \mathbf{A})$ be a directed graph, where $V = 1, 2, \ldots, n$ is the set of nodes, $E \subseteq V \times V$ is the set of directed edges and $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the adjacency matrix with non-negative elements. An edge (j, i) indicates that a node i can access node j. The element of the adjacency matrix $\mathbf{A} = [a_{ij}]_{n\times n}$ is defined as follows: $a_{ij} > 0 \iff (j, i) \in E$, otherwise $a_{ij} = 0$.

Definition 6. *([9]) The robot i intake is defined as $d_i = \sum_{j=1}^{n} a_{ij}$, and $\mathbf{D} \in \mathbb{R}^{n\times n}$ is a diagonal matrix made up of d_i. Thus, the Laplacian matrix is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{L} \in \mathbb{R}^{n\times n}$.*

The virtual leader leading the formation is denoted by R_L, while the follower i is denoted by R_i. A leader-follower adjacency vector **b** describes the interaction between the leader and the followers—if the leader is in the neighbourhood of the follower i, then $b_i > 0$, otherwise $b_i = 0$; that is, the follower's access to the virtual leader's information can be expressed by a vector of weights, i.e., $\mathbf{b} = [b_1, b_2, \ldots, b_n]^\top$, and the modified Laplacian matrix $\mathbf{\Gamma} \in \mathbb{R}^{n\times n}$ can be defined as $\mathbf{\Gamma} = \mathbf{L} + \text{diag}(\mathbf{b})$.

4 Methodology

4.1 Wheeled Mobile Robot Modelling

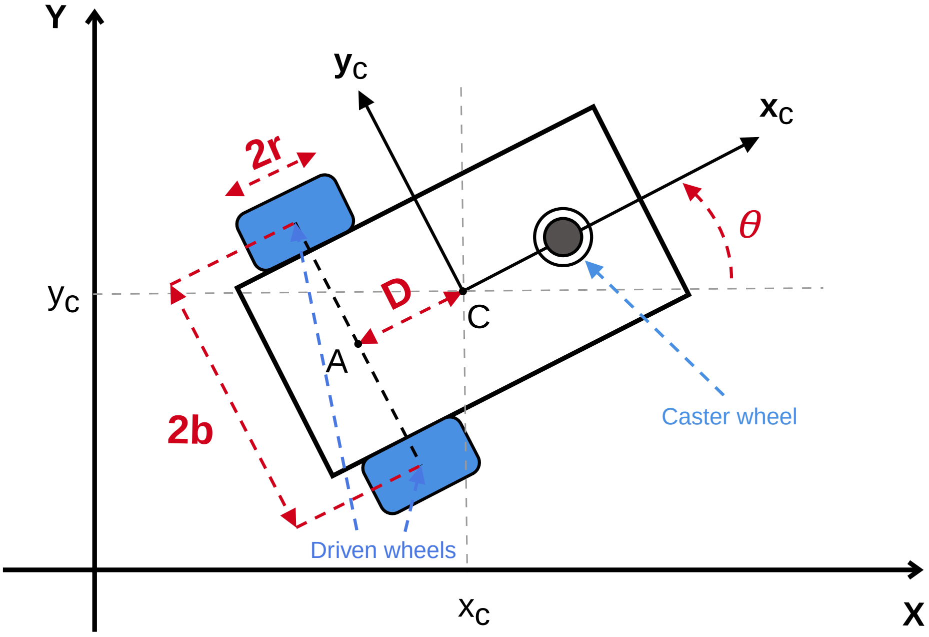


Fig. 1. Nonholonomic WMR

This section presents the kinematic and dynamic equations of a single differential drive mobile robot.

Assumption 4. *The WMR has two active wheels mounted on one axis and a front passive wheel.*

The vehicle configuration is described by three generalised coordinates: the Cartesian coordinates $(x_c,\ y_c)$ of the centre of mass measured in the fixed reference frame and the angle θ describing the platform's orientation relative to the x-axis (see Fig. 1). The state vector is thus $q = [x_c,\ y_c,\ \theta]^\top$.

4.1.1 Kinematics Model

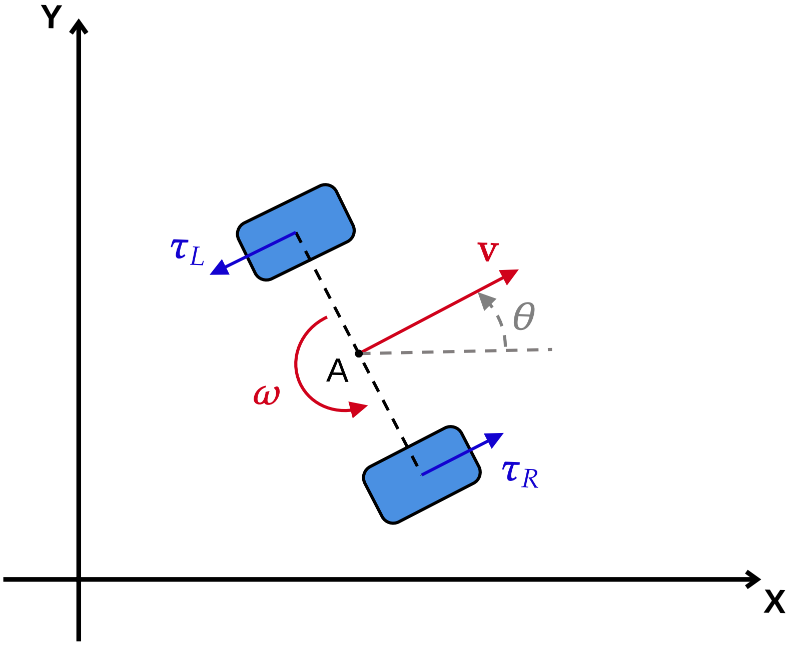


Fig. 2. Kinematic model of differential drive robot

This section is based on the work from [15].

Assumption 5. *The robot can only move along the vehicle's longitudinal axis, and the wheels roll without slipping in the horizontal plane.*

Assumption 5 describes a kinematic constraint called a nonholonomic constraint. Let the mobile robot be in the configuration described by $q(k) = [x_c(k),\ y_c(k),\ \theta(k)]^\top$. The pure rolling and non-slipping constraint can be expressed in the Pfaffian form [32] as:

$$\dot{x}_c \sin(\theta) - \dot{y}_c \cos(\theta) + D\dot{\theta} = 0, \tag{24}$$

where $\dot{x}_c$ and $\dot{y}_c$ are the robot velocity components in the Cartesian plane, $\dot{\theta}$ is the angular velocity, and D is the distance between A (the midpoint of the segment joining the centres of the two wheels) and C (the robot's centre of mass) (see Fig. 1).

The above equations can be written in the form:

$$\mathbf{A}(\mathbf{q})\dot{\mathbf{q}} = 0, \tag{25}$$

where $\mathbf{A}(\mathbf{q}) = \begin{bmatrix}\sin(\theta) & -\cos(\theta) & D\end{bmatrix}$ and $\dot{\mathbf{q}} = \begin{bmatrix}\dot{x}_c & \dot{y}_c & \dot{\theta}\end{bmatrix}^\top$ denotes the velocity vector. Thus, the kinematic model of the system can be written as the following:

$$\dot{\mathbf{q}}(k) = \mathbf{S}(\mathbf{q})\mathbf{\Phi}, \tag{26}$$

where $\mathbf{\Phi} = \begin{bmatrix}\dot{\phi}_r & \dot{\phi}_l\end{bmatrix}^\top$ is the vector of the angular velocities of the right and left wheels (see Fig. 2).

To derive the matrix $\mathbf{S}(\mathbf{q})$, the position of the centre of mass relative to A can be written as follows:

$$\begin{aligned} x_c(k) &= x_a + D\cos(\theta), \\ y_c(k) &= y_a + D\sin(\theta), \\ \theta(k) &= \theta(k). \end{aligned} \tag{27}$$

Differentiating Eq. (27) relative to time yields:

$$\dot{\mathbf{q}}(k) = \begin{bmatrix}\cos(\theta) & -D\sin(\theta) \\ \sin(\theta) & D\cos(\theta) \\ 0 & 1\end{bmatrix}\begin{bmatrix}v \\ \omega\end{bmatrix}, \tag{28}$$

where v and ω are the linear and angular velocities of the robot, respectively. According to [32], the linear and angular velocities of the robot in its local frame can be obtained as:

$$v = r\frac{\dot{\phi}_r + \dot{\phi}_l}{2}, \tag{29}$$

$$\omega = r\frac{\dot{\phi}_r - \dot{\phi}_l}{2b}, \tag{30}$$

where r is the radius of the driving wheels, and b is half the distance between the two wheel centres.

Substituting equations (29) in (28), the kinematics model of the WMR is expressed by the following:

$$\dot{\mathbf{q}}(k) = \frac{r}{2b}\begin{bmatrix}b\cos(\theta) - d\sin(\theta) & b\cos(\theta) + d\sin(\theta) \\ b\sin(\theta) + d\cos(\theta) & b\sin(\theta) - d\cos(\theta) \\ 1 & -1\end{bmatrix}\mathbf{\Phi}. \tag{31}$$

4.1.2 Dynamics Model

This section is based on the work from [9], [15].

The Lagrange formulation is used to obtain the dynamic model of the WMR system subject to the kinematic constraints in Eq. (25). A mechanical system's Lagrangian $\mathcal{L}$ is the difference between its kinetic and potential energy. The Lagrange equations, in this case, are:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}}\right) - \frac{\partial \mathcal{L}}{\partial \mathbf{q}} = \mathbf{B}(\mathbf{q})\boldsymbol{\tau} + \mathbf{A}^\top(\mathbf{q})\lambda, \tag{32}$$

where $\mathbf{B}(\mathbf{q})$ is the input transformation matrix, $\boldsymbol{\tau} = [\tau_r,\ \tau_l]^\top$ represents the actuating torques of the right and left wheels, and λ is a vector of Lagrangian multipliers. The term $\mathbf{A}^\top(\mathbf{q})\lambda$ represents the vector of reaction forces at the generalised coordinate level. The kinetic energy of the system can be written as:

$$T = \frac{1}{2}\dot{\mathbf{q}}^\top \mathbf{M}(\mathbf{q})\dot{\mathbf{q}} \tag{33}$$

where

$$\mathbf{M}(\mathbf{q}) = \text{diag}([m, m, J]),$$

$$m = (m_p + 2m_w),$$

$$J = J_{zp} + 2(J_{zw} + m_w(D^2 + b^2)),$$

where m is the total mass of the robot, m_p is the mass of the platform without the driven wheels and actuators, m_w is the mass of the wheels and their actuators, J is the moment of inertia of the robot about the centre of mass, J_{zp} is the moment of inertia of the platform, and J_{zw} is the moment of inertia of the wheels.

The robot cannot move along the z-axis; thus, the system's potential energy is defined to be zero, and the Lagrangian function is equal to the kinetic energy.

By solving the Lagrangian, the following matrix equation of motion of the WMR is obtained:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{B}(\mathbf{q})\boldsymbol{\tau} + \mathbf{A}^\top(\mathbf{q})\lambda, \tag{34}$$

where

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{0}_{3\times3}, \quad \mathbf{B}(\mathbf{q}) = \frac{1}{r_G}\begin{bmatrix} \cos(\theta) & \cos(\theta) \\ \sin(\theta) & \sin(\theta) \\ D & -D \end{bmatrix}, \quad r_G = \frac{r}{N_G},$$

$$\mathbf{A}^\top(\mathbf{q}) = \begin{bmatrix} \sin(\theta) \\ -\cos(\theta) \\ D \end{bmatrix},$$

and N_G is the actuator gear ratio.

The columns of $\mathbf{S}(\mathbf{q})$ form the basis for the null space of $\mathbf{A}(\mathbf{q})$, thus $\mathbf{S}^\top(\mathbf{q})\mathbf{A}^\top(\mathbf{q}) = \mathbf{0}$. Moreover, the Lagrange multiplier vector can be eliminated by multiplying both sides of Eq. (34) by $\mathbf{S}^\top(\mathbf{q})$. The reduced dynamic model is expressed as:

$$\overline{\mathbf{M}}(\mathbf{q})\dot{\Phi} + \overline{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\Phi = \overline{\mathbf{B}}(\mathbf{q})\boldsymbol{\tau}, \tag{35}$$

where

$$\overline{\mathbf{M}}(\mathbf{q}) = \mathbf{S}^\top(\mathbf{q})\mathbf{M}(\mathbf{q})\mathbf{S}(\mathbf{q}) = \left(\frac{r}{2b}\right)^2 \begin{bmatrix} m(b^2 + D^2) + J & m(b^2 - D^2) - J \\ m(b^2 - D^2) - J & m(b^2 + D^2) + J \end{bmatrix},$$

$$\overline{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^\top(\mathbf{q})\Big(\mathbf{M}(\mathbf{q})\dot{\mathbf{S}}(\mathbf{q}) + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{S}(\mathbf{q})\Big) = -2mbdD\left(\frac{r}{2b}\right)^2 \mathbf{S}_z(\omega),$$

$$\overline{\mathbf{B}}(\mathbf{q}) = \mathbf{S}^\top(\mathbf{q})\mathbf{B}(\mathbf{q}) = \mathbf{I}_2,$$

where $\mathbf{S}_z(\omega)$ is a skew-symmetric matrix representing the cross-product operation.

By considering a new state variable,

$$x = \begin{bmatrix} x_c \\ y_c \\ \theta \\ \dot{\phi}_r \\ \dot{\phi}_l \end{bmatrix} = \begin{bmatrix} \mathbf{q} \\ \mathbf{\Phi} \end{bmatrix},$$

the equation of motion in the control-affine, nonlinear state-space form can be represented as follows:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \tag{36}$$

where

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \mathbf{S}(\mathbf{q}) \\ -\overline{\mathbf{M}}(\mathbf{q})^{-1}\overline{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) \end{bmatrix} \mathbf{\Phi}, \quad \mathbf{g}(\mathbf{x}) = \begin{bmatrix} \mathbf{0}_{3\times 2} \\ \overline{\mathbf{M}}(\mathbf{q})^{-1}\overline{\mathbf{B}}(\mathbf{q}) \end{bmatrix}, \quad \mathbf{u} = \boldsymbol{\tau},$$

for all $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^5$ and $\mathbf{u} \in \mathcal{U} \subseteq \mathbb{R}^2$.

4.2 Formation Control

Formation control ensures that a group of mobile robots maintains a desired geometric configuration relative to a virtual leader while accounting for communication constraints, collision avoidance, and dynamic feasibility. The proposed consensus-based approach integrates input-output linearisation with constrained NMPC to achieve robust formation tracking.

The virtual leader in the mobile robot system follows a kinematic model analogous to Eq. (26), given by:

$$\dot{\mathbf{q}}_L(k) = \mathbf{R}(\mathbf{q}_L)\mathbf{v}_L, \tag{37}$$

where $\mathbf{q}_L = [x_L, y_L, \theta_L]^\top$ represents its position and orientation in Cartesian coordinates, and $\mathbf{v}_L$ denotes the vector of its linear and angular velocities.

Assumption 6. *The communication topology forms a directed spanning tree, where the virtual leader is the root node.*

Assumption 7. *There is no communication delay between the robots.*

Assumption 8. *At least one follower robot directly receives information from the virtual leader.*

The objective of the formation control strategy is to regulate the positions of the mobile robots $\mathbf{P}_i$ such that they conform to a predefined geometric configuration relative to the virtual leader. The desired formation shape is represented by the relative displacements $\Delta_i = [\Delta_{ix}, \Delta_{iy}]^\top$ for each robot $i = 1, 2, \ldots, n$ relative to the virtual leader.

The proposed control scheme assumes a hierarchical structure where a subset of followers has direct access to leader information while others rely on local communication with neighbouring robots. A formation reference system is established, incorporating collision avoidance constraints to prevent deadlock when combined with the dynamics controller. Let $\tilde{\mathbf{P}}_i = [\tilde{x}_i, \tilde{y}_i]^\top$ denote the optimal position of the i-th robot in the formation, computed by the solver. Similarly, $\mathbf{P}_L = [x_L, y_L]^\top$ represents the virtual leader's position, and $\mathbf{P}_j = [x_j, y_j]^\top$ encapsulates the positions of all neighbouring robots distributed through the communication network.

Assumption 9. *Robots within the same communication neighbourhood share their state information and optimal control inputs as computed by the consensus algorithm at the first node.*

Over a given time interval $\mathcal{I} = [t_0, t_f]$ with $t_f > t_0$, the distributed formation tracking error for robot i is formulated as:

$$\mathbf{e}_i(t) = \sum_{j=1}^{n}\{a_{ij}([\tilde{\mathbf{P}}_i(t) - \mathbf{P}_j(t)] - \Delta_{ij}(t))\} + b_i([\tilde{\mathbf{P}}_i(t) - \mathbf{P}_L(t)] - \Delta_i(t)), \tag{38}$$

where $\Delta_{ij}(t) = \Delta_i(t) - \Delta_j(t)$ represents the desired inter-robot relative displacements in the formation.

Assumption 10. *Assuming smooth leader and follower trajectories, the formation error $\mathbf{e}_i(t)$ is a differentiable function over some domain $\mathcal{E} \subseteq \mathbb{R}^2$.*

Consequently, the error dynamics evolve as follows:

$$\dot{\mathbf{e}}_i(t) = -\left[\sum_{j=1}^{n}\{a_{ij}(\dot{\mathbf{P}}_j(t) + \dot{\Delta}_{ij}(t))\} + b_i(\dot{\mathbf{P}}_L(t) + \dot{\Delta}_i(t))\right] + \left[b_i + \sum_{j=1}^{n} a_{ij}\right]\dot{\tilde{\mathbf{P}}}_i(t), \tag{39}$$

where a_{ij} and b_i denote the adjacency weights for inter-robot and leader-follower interactions, respectively, as described in Section 3.5.

The resulting system is expressed as a nonlinear control-affine model:

$$\dot{\mathbf{e}}_i(t) = \mathbf{h}_i^{dis}(t) + \mathbf{g}_i^{dis}(t)\dot{\tilde{\mathbf{P}}}_i(t), \tag{40}$$

where $\mathbf{h}_i^{dis}(t) = -\left[\sum_{j=1}^{n}\{a_{ij}(\dot{\mathbf{P}}_j(t) + \dot{\Delta}_{ij}(t))\} + b_i(\dot{\mathbf{P}}_L(t) + \dot{\Delta}_i(t))\right]$, $\mathbf{g}_i^{dis}(t) = b_i + \sum_{j=1}^{n} a_{ij}$, and $\dot{\tilde{\mathbf{P}}}_i(t)$ is the control input.

Assumption 11. *Over the finite horizon, the virtual leader and neighbouring robots are assumed to have a constant trajectory $\dot{\mathbf{P}}_j$ and $\dot{\mathbf{P}}_L$, respectively. Thus, $\mathbf{h}_i^{dis}$ and $\mathbf{g}_i^{dis}$ are time-invariant over the finite prediction horizon.*

By Assumption 10 and since the system is fully actuated, the formation error dynamics (40) is input-output linearisable as in Section 3.4. Thus, the input-output linearising control input [29], [30] can be defined as:

$$\dot{\tilde{\mathbf{P}}}_i(\mathbf{e}_i, t) = \mathbf{g}_i^{dis}(t)^\dagger \Bigg[-\mathbf{h}_i^{dis}(t) + \mathbf{z}_i(\mathbf{e}_i, t) \Bigg]. \tag{41}$$

By substituting this into the error dynamics, the auxiliary system $\dot{\mathbf{e}}_i(t) = \mathbf{z}_i(\mathbf{e}_i, t)$ is obtained. To ensure stability, the optimal feedback control law is defined as $\mathbf{z}_i(\mathbf{e}_i, t) = -\mathbf{K}_i\mathbf{e}_i(t)$, where $\mathbf{K}_i$ is a stabilising gain matrix that renders the auxiliary system Hurwitz.

A performance criterion is established through the cost function:

$$J_i^e = \int_0^\infty [\mathbf{e}_i^\top \mathbf{Q}_e \mathbf{e}_i + \mathbf{z}_i^\top \mathbf{R}_e \mathbf{z}_i]\, dt, \tag{42}$$

which leads to the Continuous Algebraic Riccati Equation (CARE):

$$\mathbf{Q}_e = \mathbf{P}_e \mathbf{R}_e^{-1} \mathbf{P}_e, \tag{43}$$

where $\mathbf{Q}_e, \mathbf{R}_e, \mathbf{P}_e \in \mathbb{S}^2_{++}$.

A constrained NMPC-β approach is used to transform this continuous optimal control problem into a convex optimisation problem. The continuous-control signal $\mathbf{z}_i(\mathbf{e}_i, t)$ in the auxiliary state space is parametrised over subintervals of the prediction horizon $[0,\ T]$. This parametrisation creates a fixed grid of nodes $k \in \{0, \dots, N\}$ defining control times t_k separated by uniform duration intervals $\delta t = T/(N-1)$. A zero-order hold discretisation is considered in this report. Denoting $\mathbf{e}_i^k = \mathbf{e}_i(t_k)$ and $\mathbf{z}_i^k = \mathbf{z}_i(\mathbf{e}_i^k, t_k)$ and integrating the continuous linearised dynamics over an interval leads to the following discrete-time representation of the dynamics:

$$\mathbf{e}_i^{k+1} = \overline{\mathbf{H}}_e \mathbf{e}_i^k + \overline{\mathbf{G}}_e \mathbf{z}_i^k, \tag{44}$$

$$\overline{\mathbf{H}}_e = \mathbf{0}_{2\times 2}, \tag{45}$$

$$\overline{\mathbf{G}}_e = \mathbf{I}_2, \tag{46}$$

where $\overline{\mathbf{H}}_e$ is the discrete-time state transition matrix, and $\overline{\mathbf{G}}_e$ is the discrete-time input matrix, both of which form a controllable pair.

To develop a stabilising control input $\mathbf{z}_i^k$, an NMPC-β controller is synthesized using elements of CLFs [11]. $\mathbf{Q}_e \in \mathbb{S}^2_{++}$ can be used to define the stage cost on the auxiliary system state. $\mathbf{P}_e \in \mathbb{S}^2_{++}$ can be used in the terminal state, which is defined as a quadratic Lyapunov Function $V : \mathcal{E} \times \mathcal{I} \rightarrow \mathbb{R}$ given by $V(\mathbf{e}_i^k, t_k) = (\mathbf{e}_i^k)^\top \mathbf{P}_e \mathbf{e}_i^k$ with a negative-definite time derivative $\dot{V}(\mathbf{e}_i^k, t_k) = -(\mathbf{e}_i^k)^\top \mathbf{Q}_e \mathbf{e}_i^k$.

Let us define the robot coordinate selection matrix $\mathbf{C}_{CBF} = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0}_{2\times 1} \end{bmatrix}$ for the robot i at time k. To guarantee safety, a DTCBF can be designed in quadratic form as:

$$h(\mathbf{q}_i(k)) = \|\mathbf{C}_{CBF}\mathbf{q}_i(k) - \mathbf{x}_{obj}^j(k)\|_2 - d_{safe}, \tag{47}$$

where $\mathbf{x}_{obj}^j(k)$ is the global position of the j-th obstacle at a time k, $d_{safe} = R + r_{obs}$ is the safe distance, R is the radius of the minimum bounding circle of the robot, and r_{obs} is the obstacle radius. This DTCBF certifies the system's safety if Eq. (10) is satisfied.

The safety condition is enforced through a DTCBF as the following constraint:

$$h_{CBF}(\mathbf{q}_i(k)) = \Delta h(\mathbf{q}_i(k)) + \hat{\gamma} h(\mathbf{q}_i(k)) \geq 0, \tag{48}$$

where $\hat{\gamma} \in [0, 1)$ is the CBF parameter.

The formation control Quadratic Programming (QP) problem in parametric form is presented below:

$$\begin{aligned}
\mathbf{Z}^* = \arg\min_{\mathbf{Z}} \quad & \beta V(\mathbf{e}_i^N, t_N) + \sum_{k=0}^{N-1} (\mathbf{e}_i^k)^\top \mathbf{Q}_e \mathbf{e}_i^k + (\Delta \mathbf{z}_i^k)^\top \mathbf{S}_e (\Delta \mathbf{z}_i^k) + \rho(\mathbf{v}_i^k - \overline{\mathbf{v}})^2, \\
s.t. \quad & \mathbf{e}_i^0 = \hat{\mathbf{e}}_i, \\
& \mathbf{e}_i^{k+1} = \overline{\mathbf{H}}_e \mathbf{e}_i^k + \overline{\mathbf{G}}_e \mathbf{z}_i^k, \quad k = 0, \dots, N-1, \\
& \overline{\mathbf{v}} = \|(\mathbf{g}_i^{dis})^\dagger (-\mathbf{h}_i^{dis})\|_2, \\
& \mathbf{v}_i^k = \|(\mathbf{g}_i^{dis})^\dagger (-\mathbf{h}_i^{dis} + \mathbf{z}_i^k)\|_2, \quad k = 0, \dots, N-1, \\
& \Delta \mathbf{z}_i^k = \mathbf{z}_i^k - \mathbf{z}_i^{k-1}, \quad k = 0, \dots, N-1, \\
& h_{CBF}(\mathbf{q}_i^k) \geq 0, \quad k = 0, \\
& \mathbf{v}_i^{lb} \leq \mathbf{v}_i^k \leq \mathbf{v}_i^{ub}, \quad k = 0, \dots, N-1,
\end{aligned} \tag{49}$$

where $\mathbf{Z} = [(\mathbf{z}_i^0)^\top, \dots, (\mathbf{z}_i^{N-1})^\top]^\top$, $\Delta \mathbf{z}_i^k$ is the control rate, $S_e \in \mathbb{S}_{++}^2$ penalises sudden changes in control inputs, ensuring smooth actuation, $\mathbf{v}_i^k$ is the magnitude of the velocity of the i-th robot at the k-th node, $\overline{\mathbf{v}}$ is the average velocity of neighbouring robots in the i-th robot's neighbourhood, $\rho \in \mathbb{R}_+$ is a weighting term that penalises deviations of the i-th robot's velocity from local velocity consensus, $\mathbf{v}_i^{lb}, \mathbf{v}_i^{ub}$ are the lower and upper bounds on the robot's velocity, respectively, over the robot's finite horizon, and $\beta \in \mathbb{R}_+$ is a parameter that scales the terminal cost. As noted in [11], if β is selected large enough, stability can be achieved without specifying a terminal state constraint.

Interior-point methods (IP) and Sequential Quadratic Programming (SQP) are two popular families of algorithms for solving general optimisation problems [33]. Additionally, the sparsity of Eq. 49 can be exploited to obtain solutions in real time and at a high sampling rate. The IPOPT solver provided in the MATLAB CasADi Library is used to solve the Quadratic Programming problem to get a practical numerical implementation in this report.

This approach extends previous formation control methods [9] by incorporating explicit safety guarantees via DTCBFs [27] while maintaining computational efficiency.

4.3 Trajectory Tracking

To ensure that each robot accurately follows the trajectory prescribed by the formation controller, a trajectory tracking NMPC is designed to minimise the deviation from the desired reference path $\widetilde{P}_i(t)$.

Let us consider the nonlinear control-affine state space model described in Eq. (36) defining the system's dynamics. Define the system output $\mathbf{y}_i : \mathcal{Y} \to \mathbb{R}^2$ as the i-th robot's position $\mathbf{y} = [x_c,\ y_c]^\top$ and the twice differentiable time-dependent desired outputs $\mathbf{y}_d : \mathcal{I} \to \mathbb{R}^2$. Assume the system satisfies the relative degree condition and is twice-differentiable on the domain $\mathcal{Y} \subseteq \mathbb{R}^2$ as described in Section 3.4, and thus the method of input-output linearisation may be used to produce an auxiliary system with linear dynamics. To linearise the dynamics of $\boldsymbol{\Phi}_i$, define the control input $\boldsymbol{\tau}(\mathbf{x}_i, t) = \overline{\mathbf{M}}[2mbd(\frac{r}{2b})^2\overline{\mathbf{M}}^{-1}\mathbf{S}_z(\omega_i)^\top\boldsymbol{\Phi}_i + \boldsymbol{\nu}(\mathbf{x}_i, t)]$, which $\boldsymbol{\nu}(\mathbf{x}_i, t)$ is the auxiliary input that will be designed to stabilise the linearised system. Substituting the expression $\boldsymbol{\tau}(\mathbf{x}_i, t)$ into the $\boldsymbol{\Phi}_i$ dynamics gives the following auxiliary system:

$$\dot{\boldsymbol{\Phi}}_i = \boldsymbol{\nu}(\mathbf{x}_i, t). \tag{50}$$

The system output dynamics are defined as:

$$\dot{\mathbf{y}}_i = \mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i, \tag{51}$$

where $\mathbf{S}_y(\mathbf{q}_i) = \mathbf{S}_{1:2,:}(\mathbf{q}_i)$ is the first two rows of the kinematics matrix defined in Eq. (31). Meanwhile, the second derivative of the system output is defined as:

$$\begin{aligned}\ddot{\mathbf{y}}_i &= \dot{\mathbf{S}}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i + \mathbf{S}_y(\mathbf{q}_i)\dot{\boldsymbol{\Phi}}_i,\\ &= \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i + \mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\nu}(\boldsymbol{\Phi}_i, t).\end{aligned} \tag{52}$$

The error between the outputs and the desired outputs in the i-th robot's local frame is defined as follows:

$$\mathbf{e}_i = \mathbf{R}(\theta_c^i)^\top(\mathbf{y}_d - \mathbf{y}_i), \tag{53}$$

where $\mathbf{R}(\theta_c^i)$ is a standard 2D rotation matrix in the counter-clockwise direction. This error allows us to define the auxiliary state $\boldsymbol{\eta}(\mathbf{x}_i, t) = [\mathbf{e}_i^\top,\ \dot{\mathbf{e}}_i^\top]^\top$, which leads to the following auxiliary system dynamics:

$$\begin{aligned}\dot{\boldsymbol{\eta}} &= \frac{d}{dt}\begin{bmatrix}\mathbf{e}_i\\ \dot{\mathbf{e}}_i\end{bmatrix}\\ &= \underbrace{-\begin{bmatrix}\mathbf{S}_z(\omega_i) & \mathbf{0}_{2\times 2}\\ \mathbf{S}_z^2(\omega_i) & 2\mathbf{S}_z(\omega_i)\end{bmatrix}\begin{bmatrix}\mathbf{e}_i\\ \dot{\mathbf{e}}_i\end{bmatrix}}_{\mathbf{l}(\mathbf{x}_i,t)} + \underbrace{\begin{bmatrix}\mathbf{R}(\theta_c^i)^\top(\dot{\mathbf{y}}_d - \mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i)\\ \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i)\end{bmatrix}}_{\mathbf{m}(\mathbf{x},t)}\\ &\quad + \underbrace{\begin{bmatrix}\mathbf{0}_{2\times 2}\\ -(\frac{r}{2b})\mathbf{S}_z(1)\mathbf{e}_i\begin{bmatrix}1\\ -1\end{bmatrix}^\top - \mathbf{R}(\theta_c^i)^\top\mathbf{S}_y(\mathbf{q}_i)\end{bmatrix}}_{\mathbf{n}(\mathbf{x}_i,t)}\boldsymbol{\nu}(\mathbf{x}_i, t).\end{aligned} \tag{54}$$

The complete derivation of the above dynamics is presented in the Appendix A.

According to Section 3.4, Eq. (54) is input-output linearisable. Let us define:

$$\begin{aligned}\widetilde{\mathbf{l}}(\mathbf{x}_i, t) &= \begin{bmatrix}\mathbf{S}_z^2(\omega_i) & 2\mathbf{S}_z(\omega_i)\end{bmatrix}\boldsymbol{\eta}(\mathbf{x}_i, t),\\ \widetilde{\mathbf{m}}(\mathbf{x}_i, t) &= \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i),\end{aligned}$$

$$\widetilde{\mathbf{n}}(\mathbf{x}_i, t) = -(\frac{r}{2b})\mathbf{S}_z(1)\mathbf{e}_i \begin{bmatrix} 1 \\ -1 \end{bmatrix}^\top - \mathbf{R}(\theta_c^i)^\top \mathbf{S}_y(\mathbf{q}_i).$$

The input-output linearising control input can then be defined as follows:

$$\boldsymbol{\nu}(\mathbf{x}_i, t) = \widetilde{\mathbf{n}}(\mathbf{x}_i, t)^\dagger \big(-\widetilde{\mathbf{l}}(\mathbf{x}_i, t) - \widetilde{\mathbf{m}}(\mathbf{x}_i, t) + \boldsymbol{\xi}(\mathbf{x}_i, t) \big), \tag{55}$$

with auxiliary control input $\boldsymbol{\xi}(\mathbf{x}_i, t) \in \mathbb{R}^2$ for all $\mathbf{x}_i \in \mathcal{X}$, $t \in \mathcal{I}$.

The controller, after substituting the $\boldsymbol{\nu}(\mathbf{x}_i, t)$, generates linear output dynamics as described below:

$$\dot{\boldsymbol{\eta}}(\mathbf{x}_i, t) = \underbrace{\begin{bmatrix} \mathbf{0}_{2\times2} & \mathbf{I}_2 \\ \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} \end{bmatrix}}_{\overline{\mathbf{H}}_c} \boldsymbol{\eta}(\mathbf{x}_i, t) + \underbrace{\begin{bmatrix} \mathbf{0}_{2\times2} \\ \mathbf{I}_2 \end{bmatrix}}_{\overline{\mathbf{G}}_c} \boldsymbol{\xi}(\mathbf{x}_i, t), \tag{56}$$

where $(\overline{\mathbf{H}}_c, \overline{\mathbf{G}}_c)$ form a controllable pair. To generate stable dynamics, define a feedback-stabilising input as $\boldsymbol{\xi}(\mathbf{x}_i, t) = -\mathbf{K}\boldsymbol{\eta}(\mathbf{x}_i, t)$. $\mathbf{K}$ is then obtained using standard LQR control by defining the following objective function:

$$J_i^c = \int_0^\infty [\boldsymbol{\eta}_i^\top \mathbf{Q}_c \boldsymbol{\eta}_i + \boldsymbol{\xi}_i^\top \mathbf{R}_c \boldsymbol{\xi}_i] \, dt, \tag{57}$$

where $\mathbf{Q}_c \in \mathbb{S}_{++}^4$, and $\mathbf{R}_c \in \mathbb{S}_{++}^2$, which allows us to obtain a positive definite matrix $\mathbf{P}_c \in \mathbb{S}_{++}^4$ by defining the following CARE:

$$\mathbf{Q}_c + \overline{\mathbf{H}}_c^\top \mathbf{P}_c + \mathbf{P}_c \overline{\mathbf{H}}_c - \mathbf{P}_c \overline{\mathbf{G}}_c^\top \mathbf{R}_c^{-1} \overline{\mathbf{G}}_c \mathbf{P}_c = 0. \tag{58}$$

Now, $\mathbf{P}_c$ can be used to define the quadratic Lyapunov function $V : \mathcal{X} \times \mathcal{I} \to \mathbb{R}$ given by $V(\boldsymbol{\eta}_i, t) = \boldsymbol{\eta}_i^\top \mathbf{P}_c \boldsymbol{\eta}_i$ with negative-definite time derivative $\dot{V}(\boldsymbol{\eta}_i, t) = -\boldsymbol{\eta}_i^\top \mathbf{Q}_c \boldsymbol{\eta}_i$.

Like Section 4.2, a zero-order hold parametrisation of the input is used over the prediction horizon $[0, T]$ to obtain the following discrete-time representation of the auxiliary system dynamics:

$$\boldsymbol{\eta}_i^{k+1} = \widehat{\mathbf{H}}_c \boldsymbol{\eta}_i^k + \widehat{\mathbf{G}}_c \boldsymbol{\xi}_i^k, \tag{59}$$

where $\widehat{\mathbf{H}}_c$ is the discrete-time state transition matrix, and and $\widehat{\mathbf{G}}_c$ is the discrete-time input matrix.

Theorem 3. *(Implicit Stability of Zero Dynamics in Trajectory Tracking)*

Consider the auxiliary system dynamics:

$$\dot{\boldsymbol{\eta}}(\mathbf{x}_i, t) = \overline{\mathbf{H}}_c \boldsymbol{\eta}(\mathbf{x}_i, t) + \overline{\mathbf{G}}_c \boldsymbol{\xi}(\mathbf{x}_i, t), \tag{60}$$

where $\boldsymbol{\xi}(\mathbf{x}_i, t)$ is the auxiliary control input. Suppose there exists a state feedback control law of the form:

$$\boldsymbol{\xi}(\mathbf{x}_i, t) = -\mathbf{K}\boldsymbol{\eta}(\mathbf{x}_i, t), \tag{61}$$

such that the closed-loop system matrix $\mathbf{A}_{cl} = \overline{\mathbf{H}}_c - \overline{\mathbf{G}}_c \mathbf{K}$ is Hurwitz. Then the following conclusions hold:

1. ***Stability of the Zero Dynamics:*** *If the system satisfies the minimum-phase condition (i.e., the zero dynamics are internally stable), stabilising the auxiliary system ensures implicit wheel velocity and heading direction stability.*
2. ***Propagation of Stability through the Control Mapping:*** *The diffeomorphic transformation* $\boldsymbol{\nu} = \widetilde{\mathbf{n}}(\mathbf{x}_i, t)^{\dagger}(-\widetilde{\mathbf{l}}(\mathbf{x}_i, t) - \widetilde{\mathbf{m}}(\mathbf{x}_i, t) + \boldsymbol{\xi}(\mathbf{x}_i, t))$ *preserves stability, ensuring that* $\boldsymbol{\nu}$ *is also a stabilizing input. Since* $\boldsymbol{\nu}$ *determines the torque input* $\boldsymbol{\tau}$ *through another diffeomorphic transformation, stability propagates to the wheel velocity dynamics.*
3. ***Convergence of Tracking Errors:*** *The auxiliary state* $\boldsymbol{\eta}(\mathbf{x}_i, t)$ *asymptotically converges to zero, ensuring that tracking errors vanish over time.*
4. ***Global Stability in the Local Frame:*** *Since the tracking error is defined in the local frame, the asymptotic convergence implies that the heading error also asymptotically converges to zero, guaranteeing overall trajectory tracking stability.*

Proof. (Proof Sketch) By construction, the auxiliary system is a fully actuated linear system controlled via state feedback. If the closed-loop system matrix $\mathbf{A}_{cl}$ is Hurwitz, then $\boldsymbol{\eta}(\mathbf{x}_i, t)$ asymptotically converges to zero. The diffeomorphic mapping ensures stability propagates through the transformation, meaning that $\boldsymbol{\nu}$ and ultimately τ remain stabilising inputs [29], [31], [34]. Since the displacement error converges to zero, the heading error must also converge, completing the proof. □

Remark 1. *The validity of this theorem relies on the assumption that the system satisfies the* ***minimum-phase condition****, meaning that the zero dynamics (which represent the internal states not directly controlled by the output feedback) are either asymptotically stable or at least not unstable. Additionally, the diffeomorphic transformation between control inputs must be well-defined, continuously differentiable, and invertible to ensure stability propagates correctly through the hierarchy of control laws.*

Now, let us define another robot coordinate selection matrix $\overline{\mathbf{C}}_{CBF} = \begin{bmatrix} I_{2\times 2} & 0_{2\times 3} \end{bmatrix}$ for the robot i at time k. To guarantee safety, a similar DTCBF to Eq. (47) is defined:

$$\overline{h}(\mathbf{x}_i(k)) = \|\overline{\mathbf{C}}_{CBF}\mathbf{x}_i(k) - \mathbf{x}^j_{obj}(k)\|_2 - d_{safe}, \tag{62}$$

certifying the system's safety if Eq. (10) is satisfied. The safety condition can now be enforced through the following DTCBF constraint:

$$\overline{h}_{CBF}(\mathbf{x}_i(k)) = \Delta\overline{h}(\mathbf{x}_i(k)) + \gamma\overline{h}(\mathbf{x}_i(k)) \geq 0. \tag{63}$$

The overall NMPC-β for the trajectory tracking system is defined as follows:

$$\begin{aligned}
[\boldsymbol{\Xi}^*,\ \gamma^*] = \operatorname*{arg\,min}_{\boldsymbol{\Xi},\ \gamma} \quad & \beta V(\boldsymbol{\eta}_i^N, t_N) + \zeta(\gamma - \overline{\gamma})^2 + \sum_{k=0}^{N-1} (\boldsymbol{\eta}_i^k)^\top \mathbf{Q}_c \boldsymbol{\eta}_i^k + (\boldsymbol{\xi}_i^k)^\top \mathbf{R}_c (\boldsymbol{\xi}_i^k), \\
s.t. \quad & \boldsymbol{\eta}_i^0 = \hat{\boldsymbol{\eta}}_i, \\
& \boldsymbol{\eta}_i^{k+1} = \widehat{\mathbf{H}}_c \boldsymbol{\eta}_i^k + \widehat{\mathbf{G}}_c \boldsymbol{\xi}_i^k, \quad k = 0, \ldots, N-1, \\
& h_{CBF}(\mathbf{x}_i^k) \geq 0, \quad k = 0, \\
& \boldsymbol{\nu}_i^k = \widetilde{\mathbf{n}}(\mathbf{x}_i^k, k)^{\dagger}\big[-\widetilde{\mathbf{l}}(\mathbf{x}_i^k, k) - \widetilde{\mathbf{m}}(\mathbf{x}_i^k, k) + \boldsymbol{\xi}_i^k\big], \quad k = 0, \ldots, N-1,
\end{aligned}$$

$$
\begin{aligned}
&\boldsymbol{\tau}_i^k = \overline{\mathbf{M}}\left[2mbd\left(\frac{r}{2b}\right)^2\overline{\mathbf{M}}^{-1}\mathbf{S}_z(\omega_i^k)^\top\boldsymbol{\Phi}_i^k + \boldsymbol{\nu}_i^k\right], \quad k = 0,\dots,N-1,\\
&0 \leq \gamma \leq 1,\\
&\boldsymbol{\tau}_i^{lb} \leq \boldsymbol{\tau}_i^k \leq \boldsymbol{\tau}_i^{ub}, \quad k = 0,\dots,N-1,
\end{aligned} \tag{64}
$$

where $\boldsymbol{\Xi} = [(\boldsymbol{\xi}_i^0)^\top,\dots,(\boldsymbol{\xi}_i^{N-1})^\top]^\top$, $\zeta \in \mathbb{R}_+$ is a weighting term that penalises deviations of the CBF parameter γ from the desired value $\overline{\gamma}$, and $\beta \in \mathbb{R}_+$ is a parameter that scales the terminal cost.

4.4 Algorithmic Implementation of Formation and Trajectory Control for WMRs

This section provides a high-level overview of the interaction between formation control and trajectory tracking in the proposed framework. The formation control module maintains a predefined geometric structure among robots, while the trajectory tracking module ensures individual robots follow their designated reference paths. These two modules coordinate through the structured communication network and the discussed control strategy.

4.4.1 Interaction Between Formation Control and Trajectory Tracking

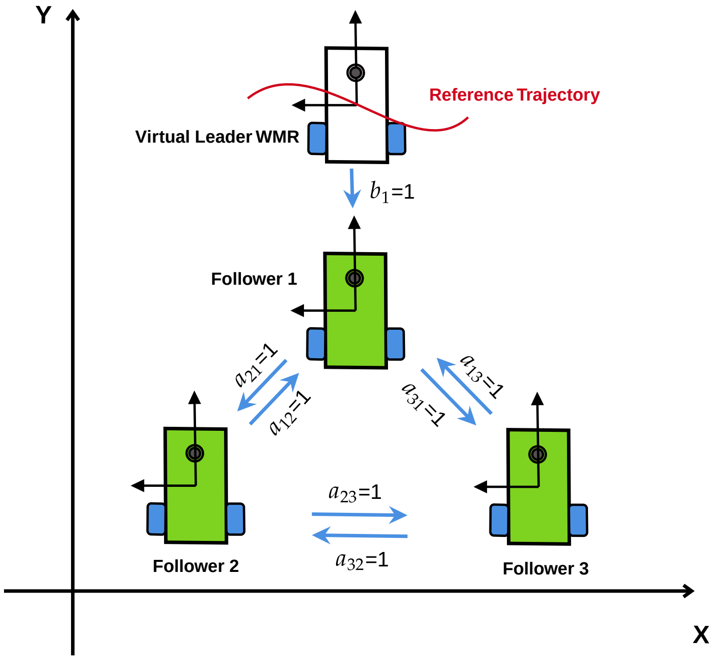


Fig. 3. Communication interaction between robots

The formation controller determines a reference trajectory for each robot, ensuring they maintain a specified formation relative to a virtual leader. However, direct application of the consensus-based formation velocities to the trajectory tracking controller results in undesirable oscillations in the velocity and acceleration profiles due to the interaction with the robot's dynamics.

To mitigate this, a virtual interaction layer is introduced that decouples the formation control from the dynamics of the robots. In this approach, the formation controller generates velocity references based on the actual positions of the robots. Still, these velocities are applied to a virtual array of ideal follower positions rather than the actual robot states. These virtual positions evolve smoothly without being affected by the disturbances introduced by the dynamics controller. The desired reference positions for each robot are then extracted from this virtual layer and passed to the trajectory-tracking NMPC.

This structure effectively filters high-frequency oscillations from the consensus-based formation velocities, preventing them from propagating into the control loop. This method ensures smooth reference trajectories by leveraging a two-time scale separation between the virtual and real positions, improving stability while maintaining formation accuracy. Conceptually, this approach resembles an observer-based filtering technique akin to a Kalman filter, where the virtual states estimate an ideal, disturbance-free trajectory.

A diagram illustrating the mobile communication topology between the robots is provided in Fig. 3.

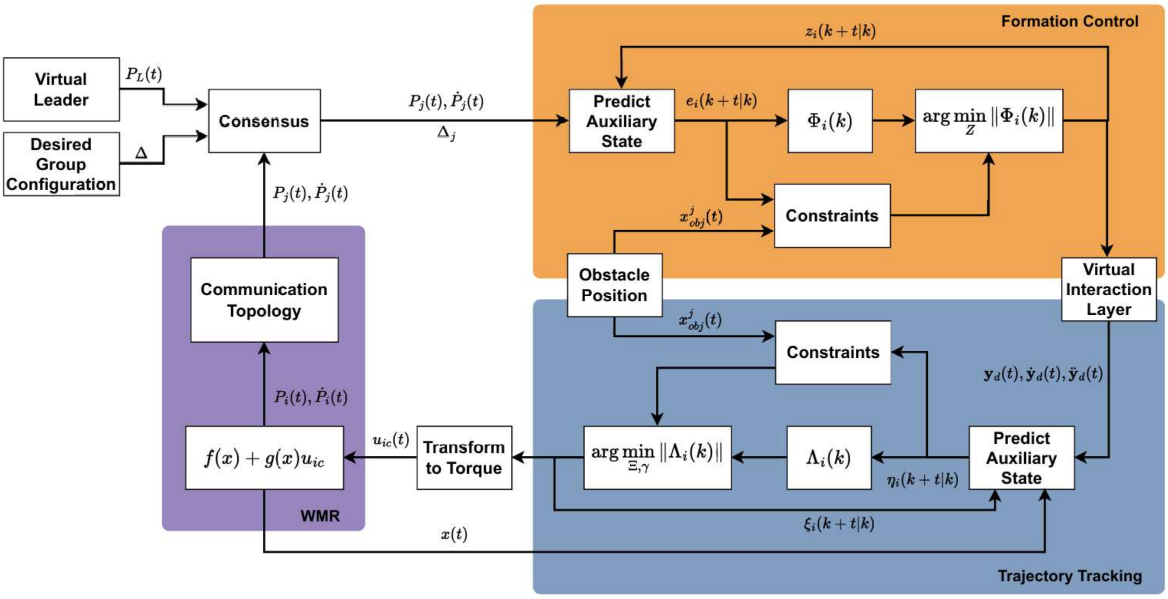


Fig. 4. Consensus NMPC Structure

A diagram illustrating the overall algorithm structure is provided in Fig. 4. The diagram highlights the information flow between the formation control and trajectory tracking modules and the distributed communication between the robots.

The following algorithm outlines the sequential execution of the formation and trajectory tracking controllers, incorporating the virtual layer.

Algorithm 1 Hierarchical NMPC for Formation Control and Trajectory Tracking

```
Require: Initial robot states $\mathbf{x}_i(0)$, virtual positions $\mathbf{P}_i^v(0) = \mathbf{y}_i(0)$
Require: Control and cost parameters $\mathbf{Q}_e, \mathbf{S}_e, \mathbf{Q}_c, \mathbf{R}_c, \beta, \zeta, \rho$
Require: Velocity and control torque constraints $\mathbf{v}_i^{lb}, \mathbf{v}_i^{ub}, \boldsymbol{\tau}_i^{lb}, \boldsymbol{\tau}_i^{ub}$
Require: CBF parameters $\overline{\gamma}, \hat{\gamma}$
$t \leftarrow 0$
while Goal not reached do
    for each robot $i$ do
        $\widehat{\mathbf{x}}_i(t) \leftarrow$ StateEstimation()                          ▷ Estimate real state
        $\mathbf{P}_j, \dot{\mathbf{P}}_j \leftarrow$ Communication()                     ▷ Exchange states with neighbours
        Formation Control (Virtual Layer Update)
        $\mathbf{e}_i(t) \leftarrow$ ComputeFormationError($\widehat{\mathbf{x}}_i, \mathbf{P}_j$)
        $\mathbf{Z}^* \leftarrow$ SolveFormationQP($\mathbf{e}_i$)                         ▷ Obtain velocity reference
        $\dot{\tilde{\mathbf{P}}}_i(t) \leftarrow$ ExtractFirstInput($\mathbf{Z}^*$)
        $\mathbf{P}_i^v(t+1) \leftarrow \mathbf{P}_i^v(t) + \dot{\tilde{\mathbf{P}}}_i(t)\Delta t$                ▷ Update virtual positions
        Trajectory Tracking
        $\boldsymbol{\eta}_i(t) \leftarrow$ ComputeTrackingError($\widehat{\mathbf{x}}_i, \mathbf{P}_i^v(t+1)$)
        $[\Xi^*, \gamma^*] \leftarrow$ SolveTrackingQP($\boldsymbol{\eta}_i$)                 ▷ Generate optimal control
        $\mathbf{u}_{ic} \leftarrow$ ExtractFirstInput($\Xi^*$)
        $\mathbf{f}(\mathbf{x}_i) + \mathbf{g}(\mathbf{x}_i)\mathbf{u} \leftarrow$ApplyControl($\mathbf{u}_{ic}$)
    end for
    $t \leftarrow t + 1$
end while
```

5 Results and discussion

This section presents the simulation results and discusses their implications. The simulation was conducted in MATLAB, where a group of WMRs navigates from an initial position, following a trajectory defined by a virtual leader. The proposed method enables the robots to track their target paths, avoid obstacles, and maintain the desired formation. Three case studies are presented below. Results are also compared against the DLPC framework introduced in Section 1.2 [12].

5.1 Formation Evaluation Criteria

Formation stability is assessed using a modified version of the consensus error defined in Eq. (38). Here, the leader-follower component is omitted to isolate formation behaviour. The total absolute formation error is calculated as:

$$\mathbf{e}_i(t) = \left| \sum_{j=1}^{n} \{a_{ij}([\mathbf{P}_i(t) - \mathbf{P}_j(t)] - \Delta_{ij}(t))\} \right|. \tag{65}$$

The Integral Absolute Error (IAE) is a performance metric used to quantify the total accumulated error of a system over time. In discrete-time systems, the IAE is calculated as the sum of the absolute values of the error at each time step: $\text{IAE}_k = T_s \sum_{k=0}^{N} |e_i(k)|$, where T_s is the sampling period, and e_i is the previously discussed absolute formation error. In this report, the IAE is used as the performance metric for analysing the performance of the control systems in the case studies. It provides a clear indication of how well the system is able to minimize deviations from the desired trajectories over time.

5.2 Case study 1: illustration of the proposed method for a linear trajectory

The system consists of three nonholonomic differential drive WMRs. The three robots are identical, each with parameters shown in Table 1.

Table 1. Definition of parameters

Variable	Value
m_p	$4.385\ kg$
m_w	$0.0575\ kg$
J_{zp}	$0.05\ kg \cdot m^2$
J_{zw}	$0.002\ kg \cdot m^2$
b	$0.125\ m$
D	$0.11\ m$
r	$0.05\ m$
N_g	1

Table 2. Obstacle Specifications 1

Obstacle	Position	Radius
Obs1	$[3.975;\ 1.1]$	0.25

Table 3. Constraints on inputs and states

Parameter	Lower Limit	Upper Limit
$\tau\ (N \cdot m)$	-0.25	0.25
v (rad/s)	-90	90

The virtual leader and follower initial values are set as $\mathbf{q}_L(0) = [1.5, 1.2, 0]^\top$, $\mathbf{q}_1(0) = [0, 0.7, 0]^\top$, $\mathbf{q}_2(0) = [-2.5, 2, 0]^\top$, and $\mathbf{q}_3(0) = [-1.8, -1.2, 0]^\top$, respectively.

The communication interactions between the virtual leader and the followers are implemented as in Fig. 3. The corresponding adjacency matrix and leader-follower adjacency vector for the communication network are shown below:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}. \tag{66}$$

The desired geometrical formation that the followers maintain relative to the virtual leader is set as follows:

$$\boldsymbol{\Delta} = \begin{bmatrix} \Delta_1 & \Delta_2 & \Delta_3 \end{bmatrix} = \begin{bmatrix} -1.5 & -3.0 & -3.0 \\ 0 & 1.5 & -1.5 \end{bmatrix}.$$

To enforce the linear trajectory, the velocities of the leader are set to $v_L(k) = 0.25$ m/s and $\omega_L(k) = 0$ rad/s.

The specifications for the obstacles are shown in Table 2, and the torque and velocity constraints of the robots are shown in Table 3.

The three robots are simulated using the defined parameters to maintain a triangle formation. Each robot should maintain the formation while following the trajectory prescribed by the virtual leader. Two cases are discussed in this section to demonstrate the effectiveness of the proposed control method:

- **Case 1.1:** The robots use the control method without formation control by implementing the following communication network:

$$\mathbf{A} = \mathbf{0}_{3\times3}, \quad \mathbf{b} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}.$$

- **Case 1.2:** The robots use the control method with the consensus formation control by implementing the communication network described in Equation (66).

The total simulation time for this case study is $40\ s$, and the sampling time is $T_s = 0.1$ s. To have a fair comparison between the two scenarios, the control parameters for each case are given as follows:

- **Case 1.1 and 1.2:** with/without formation control:

$$\mathbf{Q}_e = \text{diag}([5,5]), \quad \mathbf{S}_e = \text{diag}([50,50]), \quad \mathbf{R}_e = \text{diag}([10,10]), \quad \mathbf{P}_e = \text{diag}([7.07,7.07]),$$
$$\mathbf{Q}_c = \text{diag}([5,5,5,5]), \quad \mathbf{R}_c = \text{diag}([2.5,2.5]), \quad \mathbf{P}_c = \begin{bmatrix} 7.7689\,\mathbf{I}_2 & 3.5355\,\mathbf{I}_2 \\ 3.5355\,\mathbf{I}_2 & 5.4934\,\mathbf{I}_2 \end{bmatrix}$$
$$\mathbf{M} = 10, \quad \beta = 10, \quad \overline{\gamma} = 0.5, \quad \rho = 50, \quad \zeta = 100, \quad \hat{\gamma} = 0.2$$

5.2.1 Case 1.1: Without Formation Control

In the first case, when the consensus algorithm is not considered, the robots do not receive formation feedback from each other. This case is essentially a leader-follower system as discussed in Section 2. Fig. 5 illustrates the simulation results for the first case. In the absence of any obstacles, the three robots can track the trajectories prescribed by their neighbours and asymptotically converge and maintain the correct formation. However, the simulation results indicate that the formation error of the three robots is relatively large in the interval when robot 1 encounters an external obstacle. In this interval, as there is no feedback information between the robots, the formation error (both longitudinal and lateral) increases, and the formation cannot keep the desired shape in the presence of the obstacle, which acts as an external disturbance.

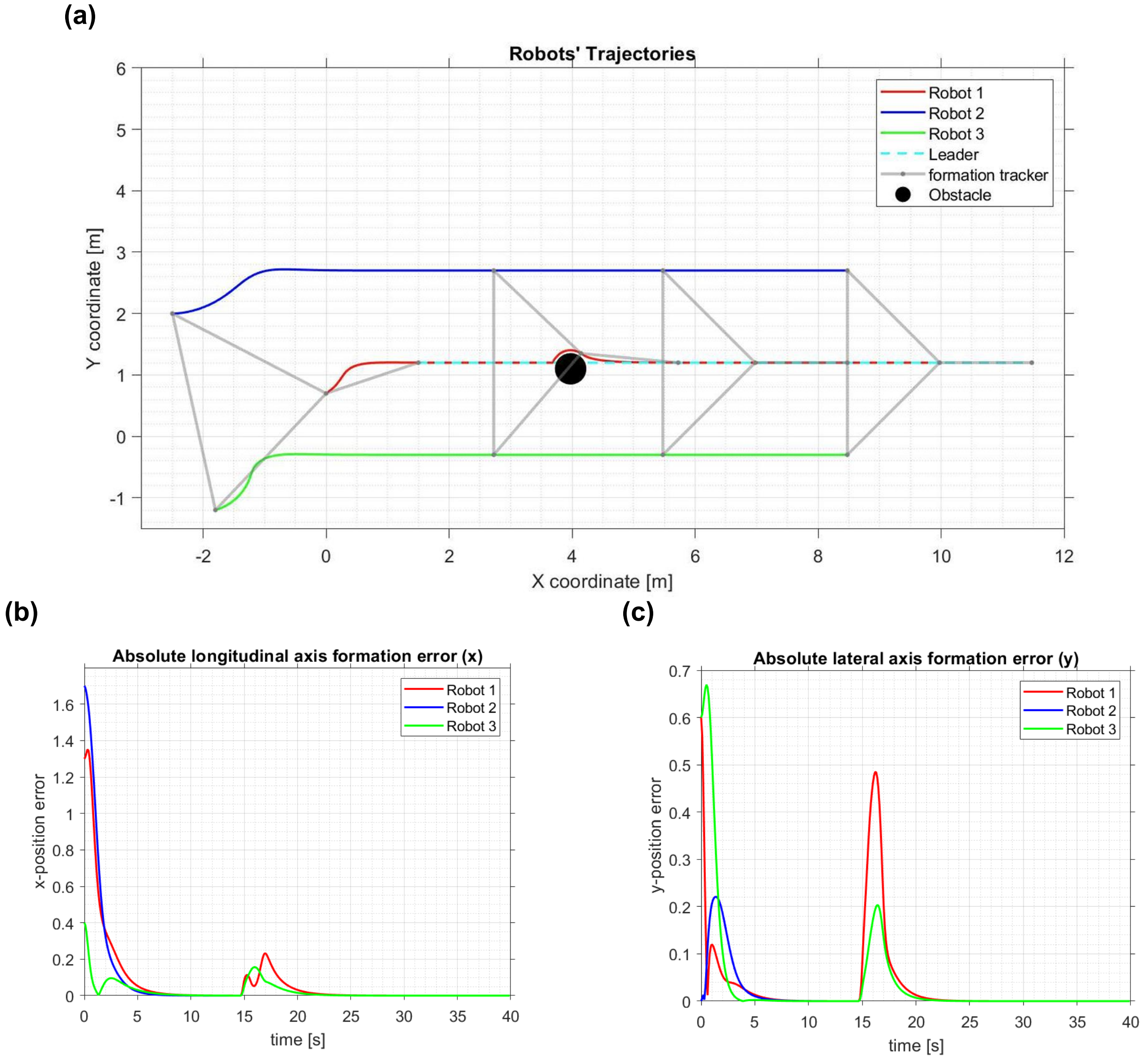


Fig. 5. Proposed method without formation control. (a) Virtual Leader and robot's trajectories. (b) The absolute longitudinal axis formation error of the robots. (c) The absolute lateral axis formation error of the robots.

Nevertheless, outside the influence of the obstacle (disturbance), the formation errors converge to zero, demonstrating the system's stability following a linear trajectory with no formation feedback.

5.2.2 Case 1.2: With Formation Control

In the second case, when the consensus algorithm is considered, the robots receive formation feedback from each other. This has similar elements to behaviour-based and virtual-structure systems, as discussed in Section 2. Fig. 6 illustrates the simulation results for the second case. Without any obstacles, the three robots can track the trajectories prescribed by their neighbours and asymptotically converge and maintain the correct formation. However, unlike the first case, the simulation results indicate that the formation error of the three robots is relatively tiny in the interval when robot 1 encounters an external obstacle. In this interval, as there is feedback information between the robots, the formation error (both longitudinal and lateral) increases slightly, and the formation keeps the desired shape in the presence of the obstacle while deviating somewhat from the intended trajectory. Both controllers' integrated absolute errors are compared and reported in

Table 4. It can be observed that the formation tracking errors of the proposed method were smaller with formation control than without formation control, as expected.

Table 4. Integral Absolute Error of x, y, case study 1.

	Robot 1		**Robot 2**		**Robot 3**	
	No Consensus	**Consensus**	**No Consensus**	**Consensus**	**No Consensus**	**Consensus**
e_x	2.98	**1.92**	2.94	**1.97**	0.91	**0.49**
e_y	1.34	**0.67**	0.92	**0.48**	1.40	**0.76**

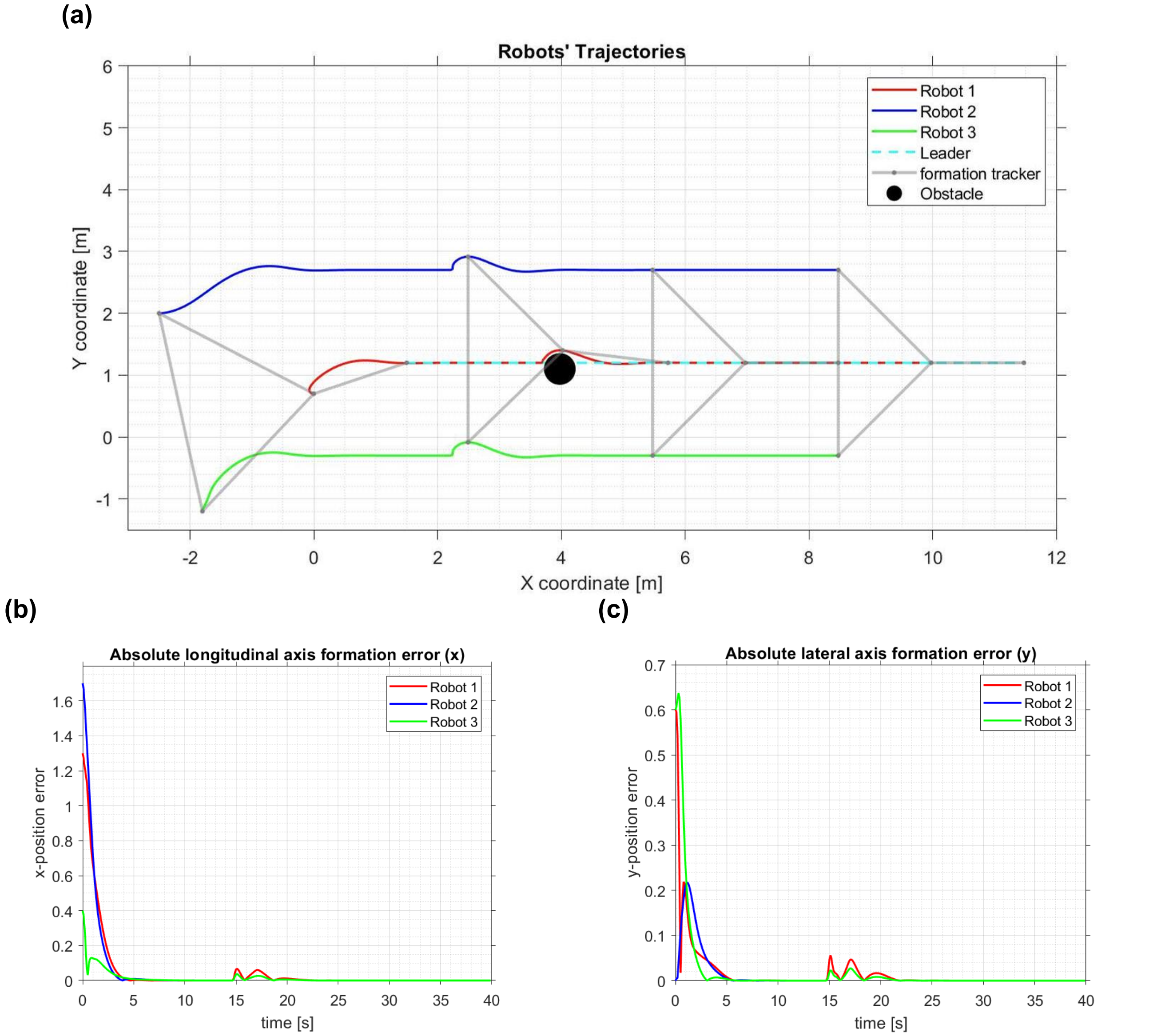


Fig. 6. Proposed method with formation control. (a) Virtual Leader and robot's trajectories. (b) The absolute longitudinal axis formation error of the robots. (c) The absolute lateral axis formation error of the robots.

5.3 Case study 2: illustration of the proposed method for a curved trajectory

In this case study, the simulation is run for a curved virtual leader trajectory and the same formation geometry as the previous section, in the presence of obstacles. The performance of the proposed method is again shown with and without formation control feedback. The same controller parameters as the previous cases were used in this scenario. However, the obstacle positions were changed. The position and radius for the new obstacles are presented in Table 5.

Table 5. Obstacle Specifications 2

Obstacle	Position	Radius
Obs1	$[3.975;\ 2.0]$	0.25

5.3.1 Case 2.1: Without Formation Control

In this case, when the consensus algorithm is not considered, as before, the robots do not receive formation feedback from each other. Fig. 7 illustrates the simulation results for the first case. In the absence of any obstacles, the three robots can track the curved trajectories prescribed by their neighbours and asymptotically converge and maintain the correct formation. However, the simulation results indicate that the formation error of the three robots is relatively large in the interval when robot 1 encounters an external obstacle. In this interval, as there is no feedback information between the robots, the formation errors (longitudinal and lateral) increase, and the formation cannot keep the desired shape in the presence of the obstacle, corroborating the results of Case 1.1.

5.3.2 Case 2.2: With Formation Control

In this case, when the consensus algorithm is considered, the robots receive formation feedback from each other, as before. Fig. 8 illustrates the simulation results for this case. Without any obstacles, the three robots can track the trajectories prescribed by their neighbours and asymptotically converge and maintain the correct formation. Similar to Case 1.2, the simulation results indicate that the formation error of the three robots is relatively tiny in the interval when robot 1 encounters an external obstacle. In this interval, as there is feedback information between the robots, the formation errors (longitudinal and lateral) increase slightly, and the formation keeps the desired shape in the presence of the obstacle while deviating somewhat from the intended trajectory. Both controllers' integrated absolute errors are compared and reported in Table 6, where the case with formation control consistently had a smaller IAE than the case without the formation control. It can be observed that this case corroborates the findings of Case 1.2, confirming that the proposed formation controller is capable of rejecting external disturbances to maintain the formation.

5.4 Case study 3: comparison between proposed controller and DLPC framework

This section compares the DPLC framework developed in [12] and the proposed control method. The main advantages of the DPLC method are that it is computationally fast, efficient, and highly scalable, which are all desirable in a scenario with multiple robots. It has also been demonstrated that the DLPC method scales up to 10000 units [12]. The DPLC framework and the proposed

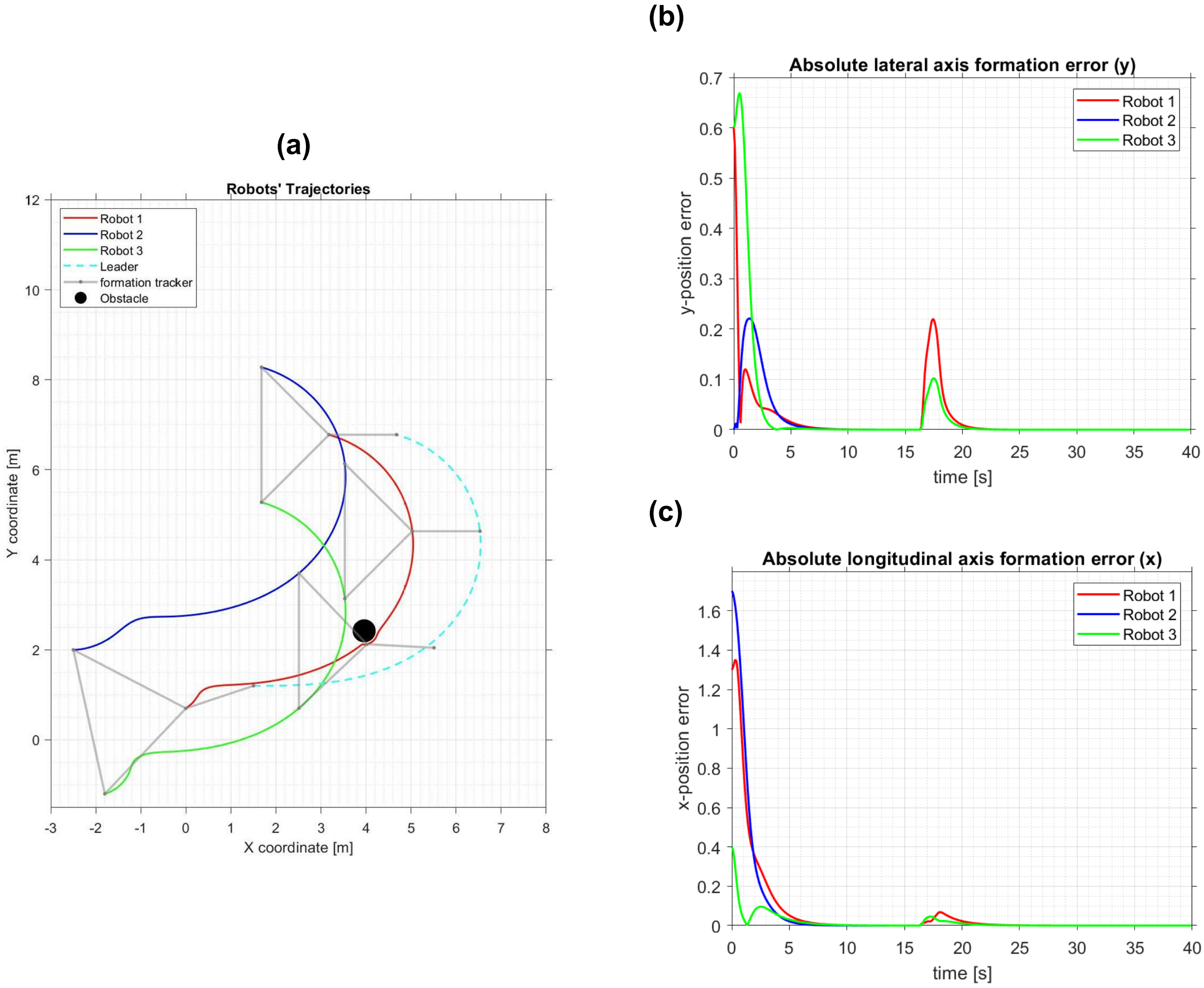


Fig. 7. Proposed method without formation control. (a) Virtual Leader and robot's trajectories. (b) The absolute longitudinal axis formation error of the robots. (c) The absolute lateral axis formation error of the robots.

Table 6. Integral Absolute Error of x, y, case study 2.

	Robot 1		**Robot 2**		**Robot 3**	
	No Consensus	**Consensus**	**No Consensus**	**Consensus**	**No Consensus**	**Consensus**
e_x	2.55	**1.81**	2.65	**1.92**	0.62	**0.44**
e_y	0.81	**0.81**	0.68	**0.56**	1.16	**0.83**

controller are applied to a scenario where two robots form a ring communication network with the virtual leader. The corresponding adjacency matrix and leader-follower adjacency vector for the communication network are shown below:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 & 1 \end{bmatrix}. \tag{67}$$

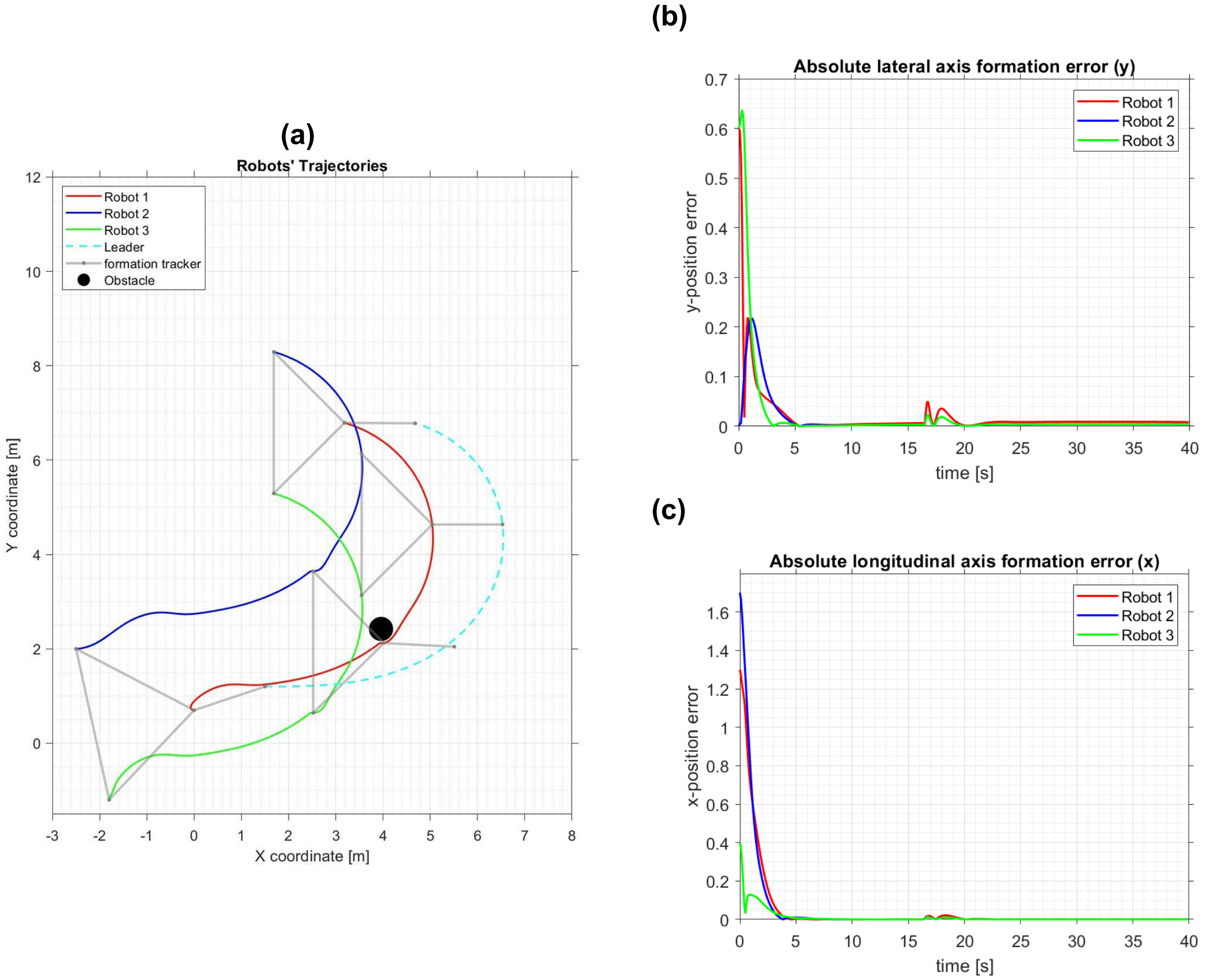


Fig. 8. Proposed method with formation control. (a) Virtual Leader and robot's trajectories. (b) The absolute longitudinal axis formation error of the robots. (c) The absolute lateral axis formation error of the robots.

To make a fair comparison, only the formation controller is considered in this case, hence the trajectory tracker is not used. The virtual robot moves in a straight line parallel to the global x-axis at a constant velocity. The virtual leader and follower initial values are set as $\mathbf{q}_L(0) = [0, 1, 0]^\top$, $\mathbf{q}_1(0) = [0, 1, 0]^\top$, and $\mathbf{q}_2(0) = [0, 0, 0]^\top$, respectively. To enforce the linear trajectory, the velocities of the leader are set to $v_L(k) = 1.0$ m/s and $\omega_L(k) = 0$ rad/s.

The desired geometrical formation that the followers maintain relative to the virtual leader is set as follows:

$$\boldsymbol{\Delta} = \begin{bmatrix} \Delta_1 & \Delta_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}.$$

The specifications for the obstacles are shown in Table 7.

Table 7. Obstacle Specifications 3

Obstacle	Position	Radius
Obs1	$[3;\ 1]$	0.1

The total simulation time for this case study is 9 s, and the sampling time is $T_s = 0.05$ s. To have a fair comparison between the two scenarios, the control parameters for each case are given as follows:

- **Case 3.1:** Proposed Controller:

$$\mathbf{Q}_e = \text{diag}([100, 20]), \quad \mathbf{S}_e = \text{diag}([50, 50]), \quad \mathbf{R}_e = \text{diag}([1, 1]), \quad \mathbf{P}_e = \text{diag}([10, 4.47]),$$
$$\mathbf{M} = 10, \quad \beta = 80, \quad \rho = 50, \quad \hat{\gamma} = 0.2$$

- **Case 3.2:** DPLC framework:

  This case uses the same parameters described in [12].

5.4.1 Case 3.1: Proposed Controller

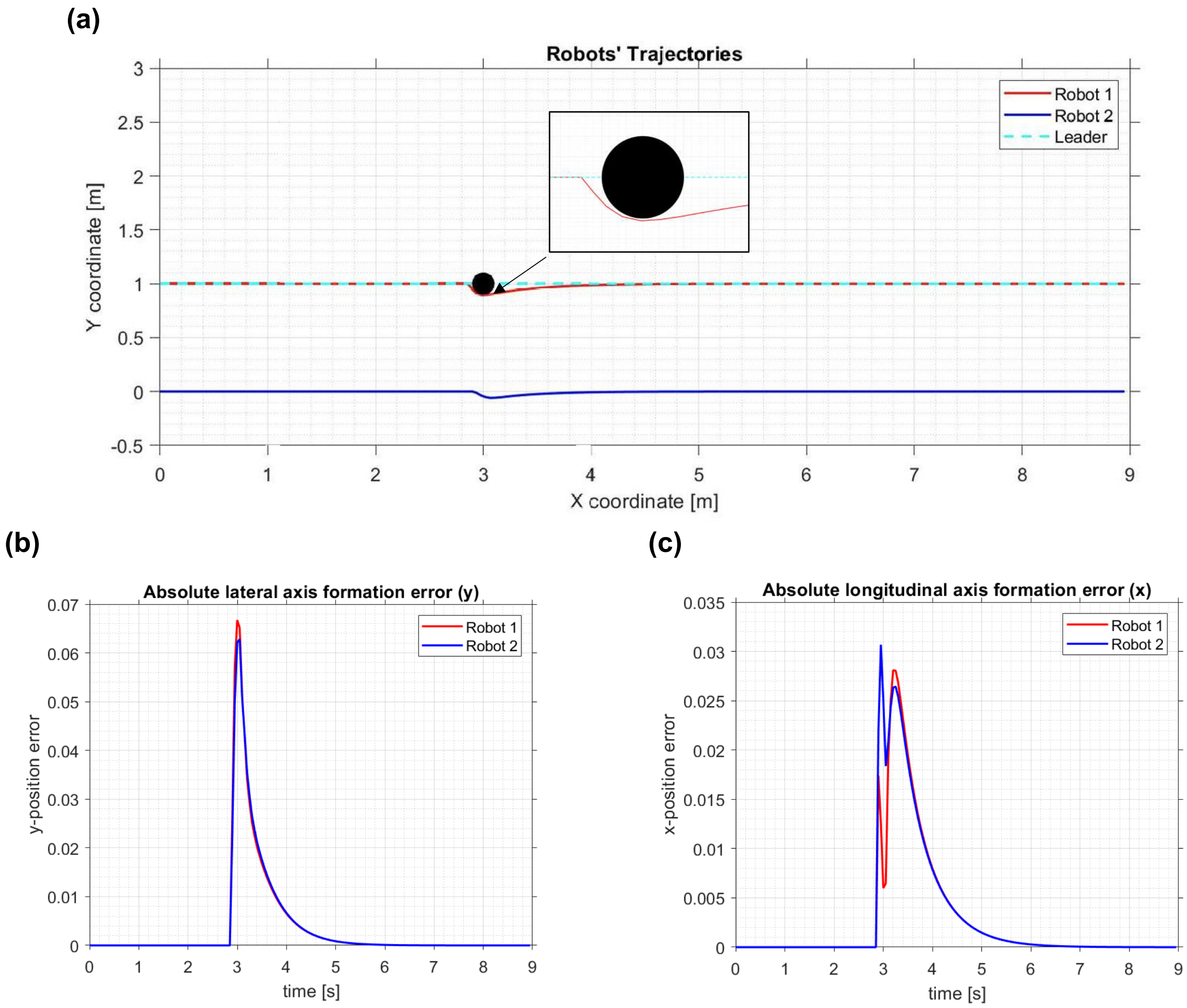


Fig. 9. Proposed method. (a) Virtual Leader and robot's trajectories. (b) The absolute longitudinal axis formation error of the robots. (c) The absolute lateral axis formation error of the robots.

Fig. 9 illustrates the simulation results for this case. As observed, the formation error between the robots is initially zero, and the controller can maintain the formation when no external disturbances are present. At about $t = 3\ s$, robot 1 encounters an obstacle, and the controller navigates around the obstacle while maintaining minimal deviation. However, a relatively large spike in the formation error can be observed at that instant. This spike is because robot 2 also has access to the leader trajectory; as such, it is not prioritising maintaining the formation only, leading to a delay in synchronising robot 1 and robot 2 until robot 1 is no longer under the influence of the obstacle. After robot 1 has navigated around the obstacle, it is observed that the formation error asymptotically converges to zero within 2 seconds. The formation error can further be minimised using the following communication network, instead, allowing the robots to prioritise maintaining the formation over following the virtual leader:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{68}$$

5.4.2 Case 3.2: DPLC Framework

Table 8. Integral Absolute Error of x, y, case study 3.

	Robot 1		**Robot 2**	
	Consensus	**DLPC**	**Consensus**	**DLPC**
e_x	**0.024**	0.067	**0.026**	0.067
e_y	**0.032**	0.092	**0.032**	0.092

Fig. 10 illustrates the simulation results for this case. The DPLC framework can also maintain the formation in the presence of external disturbances. However, unlike the proposed method, the DPLC framework begins avoiding the obstacle much earlier. This is because the DPLC framework uses a force field-inspired barrier function framework for collision avoidance instead of the DTCBF used in this report. Thus, leading to the obstacle influencing the robot's trajectory much earlier, and the influence increases as the robot gets closer to the obstacle, contrasted by the DTCBF method, which has a minimal effect until the robot is on a direct collision course with the obstacle. Thus, the proposed method outperforms the DPLC method for this particular scenario. This is further demonstrated by the IAE of both methods shown in Table 8. The IAE is smaller for the proposed method in all robots than the DPLC method.

6 Conclusions and future work

This report has developed and evaluated a novel consensus-based model predictive formation control framework for nonholonomic WMRs using input-output linearisation. This report has successfully addressed critical challenges in multi-robot formation control, particularly focusing on distributed coordination, safety guarantees, and computational efficiency.

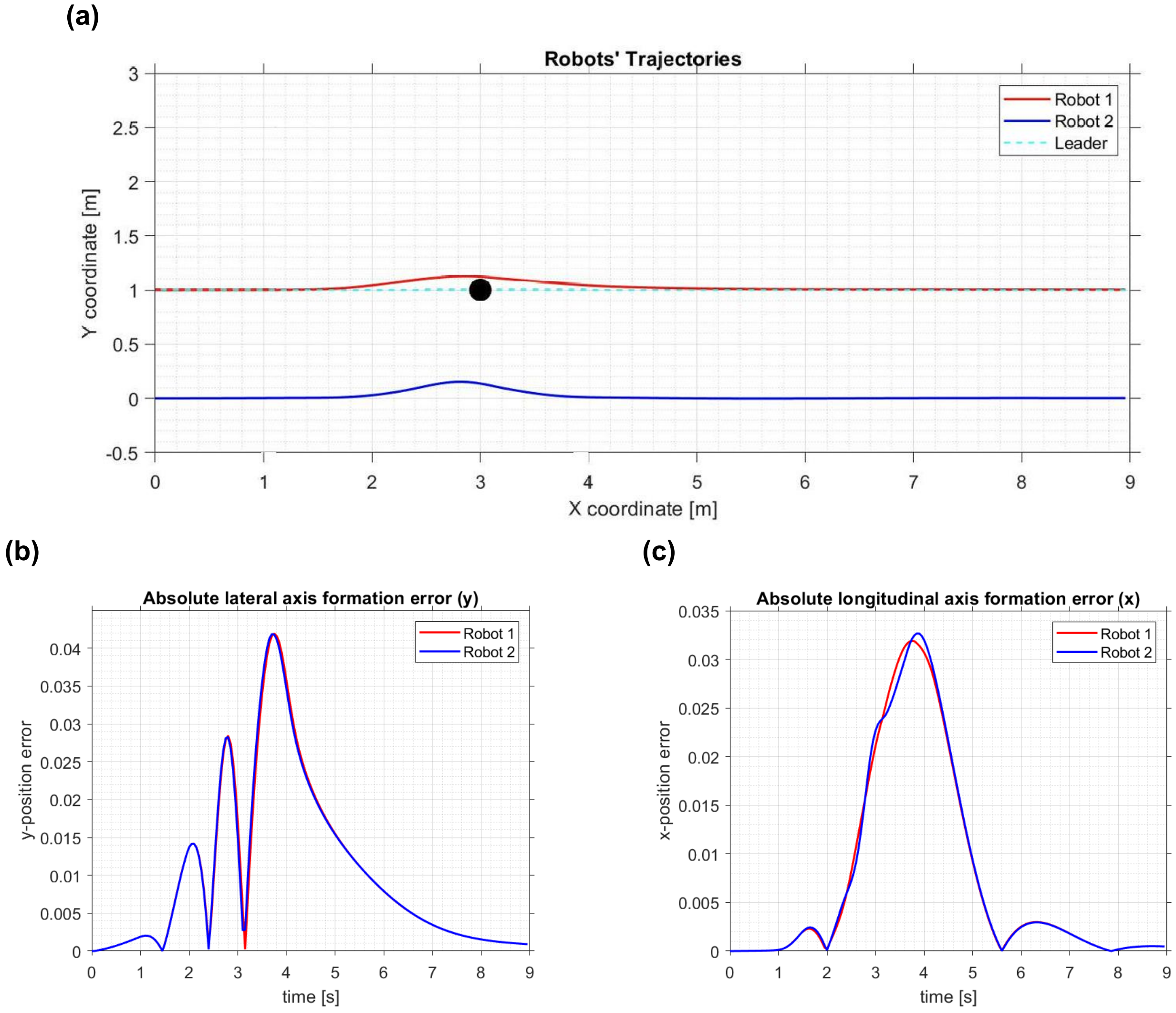


Fig. 10. DPLC Framework. (a) Virtual Leader and robot's trajectories. (b) The absolute longitudinal axis formation error of the robots. (c) The absolute lateral axis formation error of the robots.

6.1 Conclusions

This report has accomplished the aims of contributing to the development of safe and adaptable navigation in multi-robot systems. The mathematical modelling of wheeled mobile robots incorporating nonholonomic constraints was thoroughly developed, providing a robust foundation for the controller design. The proposed consensus-based NMPC framework, enhanced with input-output linearisation techniques, demonstrated exceptional performance in maintaining precise formations while navigating complex trajectories.

The integration of DTCBFs into the control architecture was found to be highly effective in formally ensuring safety during operation. This approach successfully avoided obstacles while maintaining formation integrity, even in challenging scenarios.

The comparative analysis between the proposed consensus-based NMPC and the standard distributed NMPC without consensus mechanisms revealed substantial improvements in formation stability, convergence rates, and disturbance rejection capabilities. The consensus algorithm enabled robots to rapidly agree on formation parameters while adapting to environmental changes, demonstrating superior coordination compared to non-consensus approaches.

6.1.1 Performance Evaluation and Significance

The case studies presented in this report comprehensively validated the proposed methodology in various operational scenarios. The controller exhibited excellent tracking performance for linear trajectories with minimal steady-state errors and rapid convergence to the desired formation. The formation error metrics remained consistently small, demonstrating the effectiveness of the consensus mechanism in maintaining geometric constraints.

For curved trajectories, which present significantly greater challenges due to the nonholonomic constraints of WMRs, the controller demonstrated remarkable adaptability and precision. The dynamic reconfiguration capabilities allowed the formation to navigate tight turns while preserving the relative positions between robots, showcasing the controller's ability to handle complex manoeuvres.

The comparison with the DLPC framework revealed interesting trade-offs. Although the proposed approach demonstrated superior performance in terms of formation accuracy and safety guarantees, the DLPC framework exhibited advantages in computational efficiency. This finding is still inconclusive and requires further research to confirm.

From a practical standpoint, the controller implementation demonstrated feasibility for real-time applications, with computational requirements compatible with embedded systems typically found in mobile robots. The distributed nature of the algorithm makes it particularly suitable for large-scale deployments where centralised coordination is impractical or undesirable.

6.1.2 Limitations and Challenges

Despite the promising results, several limitations were identified during this research. The assumption of perfect state information presents a simplification that may not hold in real-world scenarios with sensor noise and measurement uncertainties. Furthermore, the communication topology was assumed to be fixed, whereas practical deployments could experience intermittent connectivity and varying network topologies.

The computational complexity of solving the nonlinear optimisation problem in real-time remains a challenge, particularly for systems with limited processing capabilities. While the input-output linearisation technique and the hierarchical structure significantly reduced complexity compared to full nonlinear optimisation, further efficiency improvements are necessary for very large robot groups.

6.2 Future work

Several promising directions for future research emerge from this report. First, extending the framework to incorporate state estimation techniques would enhance robustness against sensor noise and measurement uncertainties, making the system more suitable for real-world deployment. Developing adaptive consensus gains that adjust based on formation errors and environmental conditions could further improve performance in dynamic environments.

Incorporating learning-based approaches to predict disturbances and optimise control parameters would leverage the strengths of both model-based and data-driven methods. This hybrid approach could potentially reduce computational demands while maintaining performance guarantees. In addition, exploring time-varying communication topologies and developing resilience against communication failures would address critical practical considerations for robust deployment.

Testing the framework on heterogeneous multi-robot systems with varying dynamic capabilities would expand its applicability across diverse robotic platforms. Implementing the approach on physical robotic systems would provide critical validation of the simulation results and identify practical implementation challenges.

Finally, extending the methodology to three-dimensional formations would enable applications in aerial and underwater robotics, significantly broadening the impact of this research. The theoretical foundations established in this report provide a solid platform for these future developments, paving the way for increasingly autonomous and coordinated multi-robot systems capable of operating safely and efficiently in complex environments.

In conclusion, this report has made significant contributions to the multi-robot formation control field by developing a comprehensive consensus-based model predictive control framework that effectively balances formation accuracy, safety guarantees, and computational efficiency. The methodologies and results presented provide valuable insights for theoretical advancement and practical implementation of coordinated control strategies for autonomous robot teams.

References

[1] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and Cooperation in Networked Multi-Agent Systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007, Conference Name: Proceedings of the IEEE, ISSN: 1558-2256. DOI: 10.1109/JPROC.2006.887293. Accessed: Mar. 24, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/4118472 (cited on p. 1).

[2] W. Ren, R. Beard, and E. Atkins, “A survey of consensus problems in multi-agent coordination,” in *Proceedings of the 2005, American Control Conference, 2005.*, ISSN: 2378-5861, Jun. 2005, 1859–1864 vol. 3. DOI: 10.1109/ACC.2005.1470239. Accessed: Mar. 24, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/1470239 (cited on p. 1).

[3] Y. Cao, W. Yu, W. Ren, and G. Chen, “An Overview of Recent Progress in the Study of Distributed Multi-Agent Coordination,” *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 427–438, Feb. 2013, Conference Name: IEEE Transactions on Industrial Informatics, ISSN: 1941-0050. DOI: 10.1109/TII.2012.2219061. Accessed: Mar. 24, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6303906 (cited on p. 1).

[4] K.-K. Oh, M.-C. Park, and H.-S. Ahn, “A survey of multi-agent formation control,” *Automatica*, vol. 53, pp. 424–440, Mar. 2015, ISSN: 0005-1098. DOI: 10.1016/j.automatica.2014.10.022. Accessed: Mar. 5, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109814004038 (cited on pp. 1, 2).

[5] *Factory Automation & Industrial Controls Market Size [2033] Report*. Accessed: Mar. 24, 2025. [Online]. Available: https://www.businessresearchinsights.com/market-reports/factory-automation-industrial-controls-market-107084 (cited on p. 1).

[6] P. R. Wurman, R. D’Andrea, and M. Mountz, “Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses,” en, *AI Magazine*, vol. 29, no. 1, pp. 9–19, 2008, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1609/aimag.v29i1.2082, ISSN: 2371-9621. DOI: 10.1609/aimag.v29i1.2082. Accessed: Mar. 24, 2025. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1609/aimag.v29i1.2082 (cited on p. 1).

[7] R. Beard, T. McLain, D. Nelson, D. Kingston, and D. Johanson, “Decentralized Cooperative Aerial Surveillance Using Fixed-Wing Miniature UAVs,” *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1306–1324, Jul. 2006, Conference Name: Proceedings of the IEEE, ISSN: 1558-2256. DOI: 10.1109/JPROC.2006.876930. Accessed: Mar. 25, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1677946 (cited on p. 1).

[8] W. Liu, X. Wang, and S. Li, “Formation Control for Leader–Follower Wheeled Mobile Robots Based on Embedded Control Technique,” *IEEE Transactions on Control Systems Technology*, vol. 31, no. 1, pp. 265–280, Jan. 2023, ISSN: 1558-0865. DOI: 10.1109/TCST.2022.3173887. Accessed: Feb. 5, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/9775002/?arnumber=9775002 (cited on pp. 1–3).

[9] J. Peng, H. Xiao, G. Lai, and C. Philip Chen, “Consensus formation control of wheeled mobile robots with mixed disturbances under input constraints,” en, *Journal of the Franklin Institute*, vol. 361, no. 17, p. 107 300, Nov. 2024, ISSN: 00160032. DOI: 10.1016/j.jfranklin.2024.107300. Accessed: Feb. 3,

2025. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S001600322400721X (cited on pp. 1, 3, 10, 12, 17).

[10] X. Tan and D. V. Dimarogonas, “Distributed Implementation of Control Barrier Functions for Multi-agent Systems,” *IEEE Control Systems Letters*, vol. 6, pp. 1879–1884, 2022, ISSN: 2475-1456. DOI: 10.1109/LCSYS.2021.3133802. Accessed: Dec. 16, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/9642050 (cited on p. 1).

[11] R. Grandia, A. J. Taylor, A. Singletary, M. Hutter, and A. D. Ames, “Nonlinear Model Predictive Control of Robotic Systems with Control Lyapunov Functions,” in *Robotics: Science and Systems XVI*, Jul. 2020. DOI: 10.15607/RSS.2020.XVI.098. Accessed: Dec. 22, 2024. [Online]. Available: http://arxiv.org/abs/2006.01229 (cited on pp. 1, 4, 6, 7, 16, 17).

[12] X. Zhang et al., “Toward Scalable Multirobot Control: Fast Policy Learning in Distributed MPC,” *IEEE Transactions on Robotics*, vol. 41, pp. 1491–1512, 2025, Conference Name: IEEE Transactions on Robotics, ISSN: 1941-0468. DOI: 10.1109/TRO.2025.3531818. Accessed: Mar. 5, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10847886 (cited on pp. 2, 4, 23, 28, 31).

[13] Z. Lin and H. H. Liu, “Topology-based distributed optimization for multi-UAV cooperative wildfire monitoring,” en, *Optimal Control Applications and Methods*, vol. 39, no. 4, pp. 1530–1548, 2018, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/oca.2424, ISSN: 1099-1514. DOI: 10.1002/oca.2424. Accessed: Mar. 5, 2025. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/oca.2424 (cited on p. 2).

[14] L. Li, C. Kuang, Y. Xia, and J. Qiang, “Formation control of nonholonomic mobile robots with inaccurate global positions and velocities,” en, *International Journal of Robust and Nonlinear Control*, vol. 32, no. 18, pp. 9776–9790, 2022, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rnc.6392, ISSN: 1099-1239. DOI: 10.1002/rnc.6392. Accessed: Mar. 5, 2025. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.6392 (cited on pp. 2, 3).

[15] N. Nfaileh, K. Alipour, B. Tarvirdizadeh, and A. Hadi, “Formation control of multiple wheeled mobile robots based on model predictive control,” en, *Robotica*, vol. 40, no. 9, pp. 3178–3213, Sep. 2022, ISSN: 0263-5747, 1469-8668. DOI: 10.1017/S0263574722000121. Accessed: Mar. 1, 2025. [Online]. Available: https://www.cambridge.org/core/journals/robotica/article/formation-control-of-multiple-wheeled-mobile-robots-based-on-model-predictive-control/AC68F68C04C5F9E6FE193254BF902478 (cited on pp. 2, 11, 12).

[16] J. Wang, Y. Peng, S. Wen, H. Wang, and Y. Wan, “Formation control for multiple nonholonomic wheeled mobile robots based on integrated sliding mode controller and force function,” en, *Transactions of the Institute of Measurement and Control*, p. 01 423 312 241 291 093, Dec. 2024, ISSN: 0142-3312. DOI: 10.1177/01423312241291093. Accessed: Feb. 5, 2025. [Online]. Available: https://doi.org/10.1177/01423312241291093 (cited on pp. 2, 3).

[17] M. A. Lewis and K.-H. Tan, “High Precision Formation Control of Mobile Robots Using Virtual Structures,” en, *Autonomous Robots*, vol. 4, no. 4, pp. 387–403, Oct. 1997, ISSN: 1573-7527. DOI: 10.1023/A:1008814708459. Accessed: Mar. 5, 2025. [Online]. Available: https://doi.org/10.1023/A:1008814708459 (cited on p. 3).

[18] M. Siwek, “Consensus-Based Formation Control with Time Synchronization for a Decentralized Group of Mobile Robots,” *Sensors (Basel, Switzerland)*, vol. 24, no. 12, p. 3717, Jun. 2024, ISSN: 1424-8220. DOI: 10.3390/s24123717. Accessed: Feb. 18, 2025. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11207745/ (cited on pp. 3, 4).

[19] F. Chen and W. Ren, *On the Control of Multi-Agent Systems: A Survey*. Now Foundations and Trends, 2019, vol. 1. DOI: 10.1561/2600000019 (cited on p. 3).

[20] G. Antonelli, F. Arrichiello, and S. Chiaverini, “The NSB control: A behavior-based approach for multi-robot systems,” en, *Paladyn, Journal of Behavioral Robotics*, vol. 1, no. 1, pp. 48–56, Mar. 2010, Publisher: De Gruyter Open Access Section: Paladyn, ISSN: 2081-4836. DOI: 10.2478/s13230-010-0006-0. Accessed: Mar. 6, 2025. [Online]. Available: https://www.degruyter.com/document/doi/10.2478/s13230-010-0006-0/html?utm_source=chatgpt.com (cited on p. 3).

[21] T. Balch and R. Arkin, “Behavior-based formation control for multirobot teams,” *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 926–939, Dec. 1998, Conference Name: IEEE Transactions on Robotics and Automation, ISSN: 2374-958X. DOI: 10.1109/70.736776. Accessed: Mar. 6, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/736776 (cited on p. 3).

[22] K. D. Listmann, M. V. Masalawala, and J. Adamy, “Consensus for formation control of nonholonomic mobile robots,” in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 3886–3891. DOI: 10.1109/ROBOT.2009.5152865. Accessed: Feb. 5, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/5152865 (cited on p. 4).

[23] M. Ahmadi, A. Singletary, J. W. Burdick, and A. D. Ames, “Safe Policy Synthesis in Multi-Agent POMDPs via Discrete-Time Barrier Functions,” in *2019 IEEE 58th Conference on Decision and Control (CDC)*, Dec. 2019, pp. 4797–4803. DOI: 10.1109/CDC40024.2019.9030241. Accessed: Dec. 23, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/9030241 (cited on pp. 4, 8).

[24] K. Y. Chee, T. C. Silva, M. A. Hsieh, and G. J. Pappas, *Enhancing Sample Efficiency and Uncertainty Compensation in Learning-based Model Predictive Control for Aerial Robots*, Aug. 2023. DOI: 10.48550/arXiv.2308.00570. Accessed: Dec. 16, 2024. [Online]. Available: http://arxiv.org/abs/2308.00570 (cited on p. 4).

[25] W. Gao, J. Yang, J. Liu, H. Shi, and B. Xu, “Moving Horizon Estimation for Cooperative Localisation with Communication Delay,” en, *The Journal of Navigation*, vol. 68, no. 3, pp. 493–510, May 2015, ISSN: 0373-4633, 1469-7785. DOI: 10.1017/S037346331400085X. Accessed: Dec. 16, 2024. [Online]. Available: https://www.cambridge.org/core/journals/journal-of-navigation/article/moving-horizon-estimation-for-cooperative-localisation-with-communication-delay/55872A43B94BF061AAA195ABBDCC4FB3# (cited on p. 4).

[26] Z. Artstein, “Stabilization with relaxed controls,” en, *Nonlinear Analysis: Theory, Methods & Applications*, vol. 7, no. 11, pp. 1163–1173, Jan. 1983, ISSN: 0362546X. DOI: 10.1016/0362-546X(83)90049-4. Accessed: Dec. 22, 2024. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0362546X83900494 (cited on p. 6).

[27] Y. Xiong, D.-H. Zhai, M. Tavakoli, and Y. Xia, “Discrete-Time Control Barrier Function: High-Order Case and Adaptive Case,” *IEEE Transactions on Cybernetics*, vol. 53, pp. 1–9, May 2022. DOI: 10.1109/TCYB.2022.3170607 (cited on pp. 8, 17).

[28] A. Agrawal and K. Sreenath, “Discrete Control Barrier Functions for Safety-Critical Control of Discrete Systems with Application to Bipedal Robot Navigation,” en, in *Robotics: Science and Systems XIII*, Robotics: Science and Systems Foundation, Jul. 2017, ISBN: 978-0-9923747-3-0. DOI: 10.15607/RSS.2017.XIII.073. Accessed: Dec. 23, 2024. [Online]. Available: http://www.roboticsproceedings.org/rss13/p73.pdf (cited on p. 8).

[29] H. K. Khalil, *Nonlinear Systems*, en. Prentice Hall, 2002, ISBN: 978-0-13-067389-3 (cited on pp. 8, 9, 15, 20).

[30] A. J. Taylor, V. D. Dorobantu, H. M. Le, Y. Yue, and A. D. Ames, “Episodic Learning with Control Lyapunov Functions for Uncertain Robotic Systems,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov. 2019, pp. 6878–6884. DOI: 10.1109/IROS40897.2019.8967820. Accessed: Dec. 22, 2024. [Online]. Available: http://arxiv.org/abs/1903.01577 (cited on pp. 8, 15).

[31] S. Sastry, *Nonlinear Systems* (Interdisciplinary Applied Mathematics), J. E. Marsden, L. Sirovich, and S. Wiggins, Eds. New York, NY: Springer, 1999, vol. 10, ISBN: 978-1-4757-3108-8. DOI: 10.1007/978-1-4757-3108-8. Accessed: Mar. 22, 2025. [Online]. Available: http://link.springer.com/10.1007/978-1-4757-3108-8 (cited on pp. 9, 20).

[32] R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*, en. MIT Press, 2004, Google-Books-ID: gUbQ9_weg88C, ISBN: 978-0-262-19502-7 (cited on pp. 11, 12).

[33] J. Nocedal and S. J. Wright, *Numerical Optimization* (Springer Series in Operations Research and Financial Engineering), en. Springer New York, 2006, ISBN: 978-0-387-30303-1. DOI: 10.1007/978-0-387-40065-5. Accessed: Mar. 10, 2025. [Online]. Available: http://link.springer.com/10.1007/978-0-387-40065-5 (cited on p. 17).

[34] J.-J. E. Slotine and W. Li, *Applied Nonlinear Control*, en. Prentice-Hall, 1991, Google-Books-ID: HddxQgAACAAJ, ISBN: 978-0-13-040049-9 (cited on p. 20).

[35] L. Wang, *Model Predictive Control System Design and Implementation Using MATLAB®* (Advances in Industrial Control), en. London: Springer, 2009, ISSN: 1430-9491, ISBN: 978-1-84882-331-0. DOI: 10.1007/978-1-84882-331-0. Accessed: Mar. 9, 2025. [Online]. Available: http://link.springer.com/10.1007/978-1-84882-331-0 (cited on p. 42).

Appendices

A Full Derivation of Auxiliary Tracking System Dynamics

The error between the outputs and the desired outputs in the i-th robot's local frame is defined as follows:

$$\mathbf{e}_i = \mathbf{R}(\theta_c^i)^\top(\mathbf{y}_d - \mathbf{y}_i), \tag{A.69}$$

where $\mathbf{R}(\theta_c^i)$ is a standard 2D rotation matrix in the counter-clockwise direction.

The first derivative of the trajectory tracking error can be defined as follows:

$$\begin{aligned}
\dot{\mathbf{e}}_i &= \frac{\partial}{\partial t}\big(\mathbf{R}(\theta_c^i)^\top(\mathbf{y}_d - \mathbf{y}_i)\big) \\
&= \frac{\partial}{\partial t}\big(\mathbf{R}(\theta_c^i)^\top\big)(\mathbf{y}_d - \mathbf{y}_i) + \mathbf{R}(\theta_c^i)^\top(\dot{\mathbf{y}}_d - \dot{\mathbf{y}}_i) \\
&= -\mathbf{S}_z(\omega_i)\mathbf{R}(\theta_c^i)^\top(\mathbf{y}_d - \mathbf{y}_i) + \mathbf{R}(\theta_c^i)^\top(\dot{\mathbf{y}}_d - \dot{\mathbf{y}}_i) \\
&= -\mathbf{S}_z(\omega_i)\mathbf{e}_i + \mathbf{R}(\theta_c^i)^\top(\dot{\mathbf{y}}_d - \mathbf{S}_y(\mathbf{q}_i)\mathbf{\Phi}_i).
\end{aligned} \tag{A.70}$$

The second derivative of the trajectory tracking error can be defined as follows:

$$\begin{aligned}
\ddot{\mathbf{e}}_i &= \frac{\partial}{\partial t}(\dot{\mathbf{e}}_i) \\
&= \frac{\partial}{\partial t}\big(-\mathbf{S}_z(\omega_i)\mathbf{e}_i + \mathbf{R}(\theta_c^i)^\top(\dot{\mathbf{y}}_d - \mathbf{S}_y(\mathbf{q}_i)\mathbf{\Phi}_i)\big) \\
&= -\dot{\mathbf{S}}_z(\omega_i)\mathbf{e}_i - \mathbf{S}_z(\omega_i)\dot{\mathbf{e}}_i + \frac{\partial}{\partial t}\big(\mathbf{R}(\theta_c^i)^\top\big)(\dot{\mathbf{y}}_d - \mathbf{S}_y(\mathbf{q}_i)\mathbf{\Phi}_i) + \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \dot{\mathbf{S}}_y(\mathbf{q}_i)\mathbf{\Phi}_i - \mathbf{S}_y(\mathbf{q}_i)\dot{\mathbf{\Phi}}_i) \\
&= -\mathbf{S}_z(\dot{\omega}_i)\mathbf{e}_i - \mathbf{S}_z(\omega_i)\dot{\mathbf{e}}_i - \mathbf{S}_z(\omega_i)\mathbf{R}(\theta_c^i)^\top(\dot{\mathbf{y}}_d - \mathbf{S}_y(\mathbf{q}_i)\mathbf{\Phi}_i) + \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \dot{\mathbf{S}}_y(\mathbf{q}_i)\mathbf{\Phi}_i - \mathbf{S}_y(\mathbf{q}_i)\dot{\mathbf{\Phi}}_i) \\
&= -\mathbf{S}_z(\dot{\omega}_i)\mathbf{e}_i - \mathbf{S}_z(\omega_i)\dot{\mathbf{e}}_i - \mathbf{S}_z(\omega_i)\big(\dot{\mathbf{e}}_i + \mathbf{S}_z(\omega_i)\mathbf{e}_i\big) + \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\mathbf{\Phi}_i - \mathbf{S}_y(\mathbf{q}_i)\dot{\mathbf{\Phi}}_i) \\
&= -\mathbf{S}_z(\dot{\omega}_i)\mathbf{e}_i - 2\mathbf{S}_z(\omega_i)\dot{\mathbf{e}}_i - \mathbf{S}_z^2(\omega_i)\mathbf{e}_i + \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\mathbf{\Phi}_i - \mathbf{S}_y(\mathbf{q}_i)\dot{\mathbf{\Phi}}_i).
\end{aligned} \tag{A.71}$$

The angular acceleration of the i-th robot's platform can be expressed as follows:

$$\begin{aligned}
\dot{\omega}_i &= \frac{d}{dt}\omega_i \\
&= \frac{d}{dt}\left(\frac{r}{2b}(\dot{\phi}_r - \dot{\phi}_l)\right) \\
&= \frac{r}{2b}(\ddot{\phi}_r - \ddot{\phi}_l) \\
&= \frac{r}{2b}\begin{bmatrix} 1 \\ -1 \end{bmatrix}^\top \dot{\mathbf{\Phi}}_i
\end{aligned} \tag{A.72}$$

Using Eq. (A.72), Eq. (A.71) can be further simplified:

$$\ddot{\mathbf{e}}_i = -\mathbf{S}_z(\dot{\omega}_i)\mathbf{e}_i - 2\mathbf{S}_z(\omega_i)\dot{\mathbf{e}}_i - \mathbf{S}_z^2(\omega_i)\mathbf{e}_i + \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\mathbf{\Phi}_i - \mathbf{S}_y(\mathbf{q}_i)\dot{\mathbf{\Phi}}_i)$$

$$
\begin{aligned}
&= -\frac{r}{2b}\mathbf{S}_z(1)\mathbf{e}_i \begin{bmatrix} 1 \\ -1 \end{bmatrix}^\top \dot{\boldsymbol{\Phi}}_i - 2\mathbf{S}_z(\omega_i)\dot{\mathbf{e}}_i - \mathbf{S}_z^2(\omega_i)\mathbf{e}_i + \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i - \mathbf{S}_y(\mathbf{q}_i)\dot{\boldsymbol{\Phi}}_i) \\
&= -\mathbf{S}_z^2(\omega_i)\mathbf{e}_i - 2\mathbf{S}_z(\omega_i)\dot{\mathbf{e}}_i + \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i) \\
&\quad - \left(\frac{r}{2b}\mathbf{S}_z(1)\mathbf{e}_i \begin{bmatrix} 1 \\ -1 \end{bmatrix}^\top + \mathbf{R}(\theta_c^i)^\top\mathbf{S}_y(\mathbf{q}_i)\right)\dot{\boldsymbol{\Phi}}_i.
\end{aligned} \tag{A.73}
$$

This allows us to define the auxiliary state $\boldsymbol{\eta}(\mathbf{x}_i, t) = [\mathbf{e}_i^\top,\ \dot{\mathbf{e}}_i^\top]^\top$ and subsequently, the auxiliary system dynamics:

$$
\begin{aligned}
\dot{\boldsymbol{\eta}} &= \frac{d}{dt}\begin{bmatrix} \mathbf{e}_i \\ \dot{\mathbf{e}}_i \end{bmatrix} \\
&= \begin{bmatrix} \dot{\mathbf{e}}_i \\ \ddot{\mathbf{e}}_i \end{bmatrix} \\
&= -\begin{bmatrix} \mathbf{S}_z(\omega_i) & \mathbf{0}_{2\times 2} \\ \mathbf{S}_z^2(\omega_i) & 2\mathbf{S}_z(\omega_i) \end{bmatrix}\begin{bmatrix} \mathbf{e}_i \\ \dot{\mathbf{e}}_i \end{bmatrix} + \begin{bmatrix} \mathbf{R}(\theta_c^i)^\top(\dot{\mathbf{y}}_d - \mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i) \\ \mathbf{R}(\theta_c^i)^\top(\ddot{\mathbf{y}}_d - \mathbf{S}_z(\omega_i)\mathbf{S}_y(\mathbf{q}_i)\boldsymbol{\Phi}_i) \end{bmatrix} \\
&\quad - \begin{bmatrix} \mathbf{0}_{2\times 2} \\ \left(\frac{r}{2b}\right)\mathbf{S}_z(1)\mathbf{e}_i \begin{bmatrix} 1 \\ -1 \end{bmatrix}^\top + \mathbf{R}(\theta_c^i)^\top\mathbf{S}_y(\mathbf{q}_i) \end{bmatrix}.
\end{aligned} \tag{A.74}
$$

B Tuning

The performance and stability of the proposed formation and trajectory tracking controllers are highly dependent on appropriate parameter tuning. Given the hierarchical nature of the control architecture, the tuning process must ensure smooth integration between the formation control, the virtual interaction layer, and the trajectory tracking NMPC. The objective is to balance accuracy, smoothness, and robustness while maintaining feasibility and constraint satisfaction.

The first step in the tuning process involves selecting appropriate weighting parameters in the cost functions of both the formation control and trajectory tracking optimisation problems. The matrices $\mathbf{Q}_e$, $\mathbf{Q}_c$ penalise tracking errors, thereby influencing the aggressiveness of the control response. Larger values of these matrices improve tracking accuracy, but can lead to high-frequency oscillations and excessive control effort if not correctly balanced. Similarly, the matrices $\mathbf{S}_e$, $\mathbf{R}_c$ penalise rapid changes in control inputs to ensure smooth actuation. Excessively low values may result in abrupt changes in the control commands, while excessively high values may lead to overly conservative control responses.

The prediction horizons M and N serve as critical tuning parameters. Selecting prediction horizons requires balancing computational feasibility and system stability[35]. A horizon that is too short results in a system with limited predictive capabilities, potentially compromising stability guarantees. Conversely, excessively long horizons may introduce numerical feasibility issues in the numerical solver due to high-dimensional optimisation complexity. Thus, the prediction horizons are empirically chosen to ensure a trade-off between stability and computational efficiency.

A crucial aspect of the formation controller is the velocity consensus term, which is regulated by the weighting parameter ρ. This term ensures that the velocities of neighbouring robots remain coordinated, thereby preserving formation cohesion. However, suppose ρ is set too high. In that case, the system may over-prioritise velocity matching at the expense of formation flexibility, potentially leading to infeasibility where individual robots must deviate from the average velocity to satisfy constraints. Conversely, if ρ is too low, individual robots may prioritise their trajectories over formation cohesion, leading to poor formation maintenance.

The tuning process also requires careful adjustment of system constraints. The velocity bounds $\mathbf{v}_i^{lb}$ and $\mathbf{v}_i^{ub}$, as well as the motor torque limits $\boldsymbol{\tau}_i^{lb}$ and $\boldsymbol{\tau}_i^{ub}$, must be chosen to reflect the physical limitation of the robots, while ensuring adequate feasibility margins.

C Initial Project Outline

C.1 Background and Motivation

Reinforcement Learning (RL) is a machine learning approach where an agent learns to make decisions by interacting with its environment and receiving feedback through rewards or penalties. Over time, the agent improves decision-making by optimising a policy that maximises cumulative rewards. RL algorithms such as Q-learning and Deep Q-Networks (DQN) commonly tackle decision-making problems in continuous or discrete state-action spaces. At the same time, more advanced methods like Proximal Policy Optimization (PPO) are better suited for handling complex, continuous environments like those faced by autonomous robots.

On the other hand, pathfinding algorithms like A* and Dijkstra's algorithm are widely used in robotics to find the shortest path in known environments. A* improves search efficiency by using a heuristic to estimate the distance to the goal, while Dijkstra's algorithm guarantees the shortest path by exploring all possible routes. These algorithms perform well in static, fully observable environments where the terrain is mapped in advance. However, they can struggle in unpredictable, dynamically changing environments, as they need to recompute paths when new obstacles appear.

The motivation for this project stems from the increasing demand for autonomous exploration systems that can operate in environments unavailable or inhospitable to humans. In fields such as underwater and space exploration, robots must be able to navigate complex, uncharted terrains while avoiding obstacles and making real-time decisions without human intervention. These robots must operate autonomously and perform reliably in extreme and unpredictable environments, making Reinforcement Learning a perfect research avenue.

C.2 Aims and Objectives

This project aims to contribute to the development of intelligent and adaptable navigation systems capable of handling unforeseen obstacles and dynamically changing conditions. The goal is to ensure the safety and reliability of future autonomous vehicles in challenging environments. Specifically, the project will focus on designing and implementing an RL-based 2D model of a mobile robot that can traverse unfamiliar virtual terrains populated with randomly generated obstacles and successfully reach a predetermined goal.

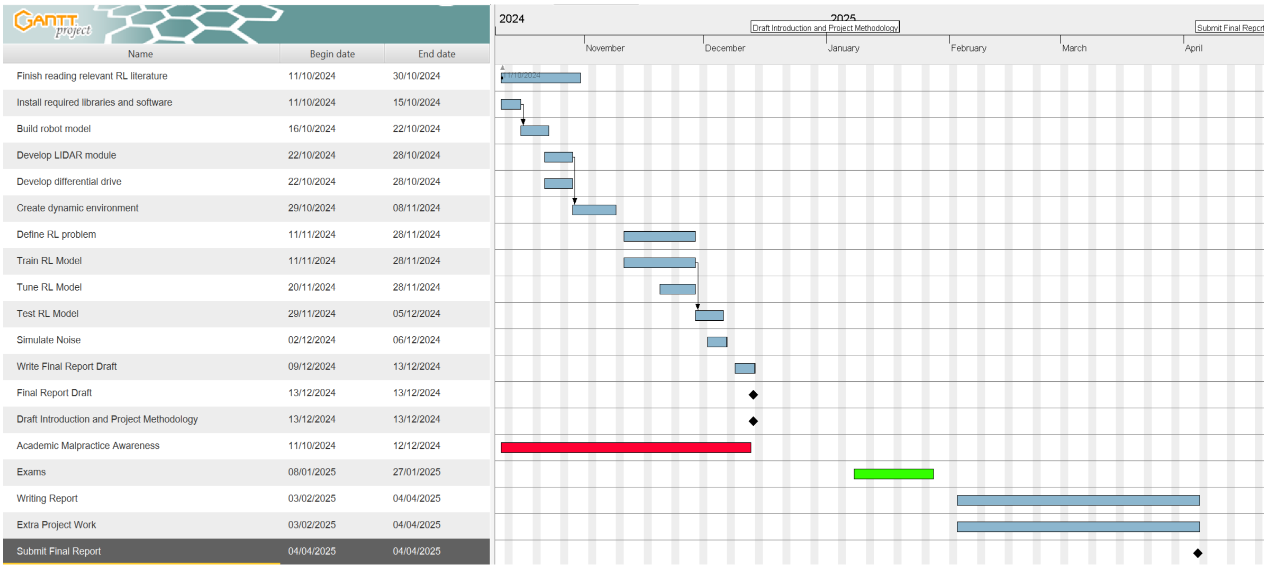
In addition to developing the RL-based navigation system, the project will compare its performance against traditional pathfinding algorithms, such as the A* algorithm. This comparison will assess RL's efficiency and adaptability in navigating unknown and unpredictable environments, highlighting its advantages or limitations relative to classical approaches.

Finally, the system's ability to adapt to dynamically changing conditions and obstacles will be tested to ensure real-time decision-making capabilities during navigation.

The main objectives are:

- **Understand and apply Reinforcement Learning techniques**: Gain a thorough understanding of RL methodologies by reviewing relevant literature and applying this knowledge to represent the robot's navigation problem as an RL task, including defining state space, action space, and reward functions.
- **Design and develop the robot and its environment**: Build a simulated mobile robot model with a LIDAR module for obstacle detection. Create a dynamic grid-based environment with randomly generated obstacles and goals to simulate real-world conditions. Improve the grid-based environment to a continuous environment and a differential drive system for realistic motion.
- **Train and optimise the RL model**: Train the RL model on multiple iterations of the environment, adjusting hyper-parameters such as learning rate and exploration strategy to enhance performance and generalisation.
- **Evaluate and compare performance**: Test the RL model's effectiveness by measuring time to goal, number of collisions, and overall rewards. Using statistical analysis, compare the performance of the RL-based approach to traditional pathfinding algorithms.
- **Analyse robustness and document results**: Assess the RL model's robustness by simulating sensor noise and testing its reliability under imperfect conditions. Organise and present the findings in a final report, highlighting key insights and results.

D Initial Project Plan



Name	Begin date	End date
Finish reading relevant RL literature	11/10/2024	30/10/2024
Install required libraries and software	11/10/2024	15/10/2024
Build robot model	16/10/2024	22/10/2024
Develop LIDAR module	22/10/2024	28/10/2024
Develop differential drive	22/10/2024	28/10/2024
Create dynamic environment	29/10/2024	08/11/2024
Define RL problem	11/11/2024	28/11/2024
Train RL Model	11/11/2024	28/11/2024
Tune RL Model	20/11/2024	28/11/2024
Test RL Model	29/11/2024	05/12/2024
Simulate Noise	02/12/2024	06/12/2024
Write Final Report Draft	09/12/2024	13/12/2024
Final Report Draft	13/12/2024	13/12/2024
Draft Introduction and Project Methodology	13/12/2024	13/12/2024
Academic Malpractice Awareness	11/10/2024	12/12/2024
Exams	08/01/2025	27/01/2025
Writing Report	03/02/2025	04/04/2025
Extra Project Work	03/02/2025	04/04/2025
Submit Final Report	04/04/2025	04/04/2025

E Software Code

Consensus-Based MPC for Multi-Robot Formation Control

Introduction

This repository contains the MATLAB implementation of a **Consensus-Based Model Predictive Control (MPC)** algorithm for multi-robot formation control. The code simulates a distributed multi-robot system where robots maintain a desired formation while following a leader robot's trajectory. The implementation incorporates **control barrier functions (CBFs)** for obstacle avoidance and ensures consensus among robots to achieve desired trajectories.

The code simulates a system of three robots following a virtual leader while avoiding obstacles. Each robot operates under constraints defined by its kinematics, dynamics, and inter-robot communication network.

System Overview

- **Leader-follower architecture:** The leader robot generates a trajectory, and the followers maintain a predefined formation relative to the leader.
- **Distributed control:** Each robot computes its control inputs locally using consensus-based MPC.
- **Obstacle avoidance:** Control barrier functions ensure safety by maintaining a minimum distance from obstacles.
- **Simulation visualization:** The code generates plots to visualize robot trajectories, velocities, formation errors, wheel torques, and other key parameters.

Installation Instructions

To run the simulation and visualize the results, follow these steps:

1. **Install MATLAB:** Ensure that MATLAB is installed on your system. This code was tested on MATLAB R2023b but should work on other versions.
2. **Install CasADi:**
   - Download the CasADi toolbox from CasADi official website
   - Add the CasADi toolbox to your MATLAB path.
3. **Clone the Repository** :

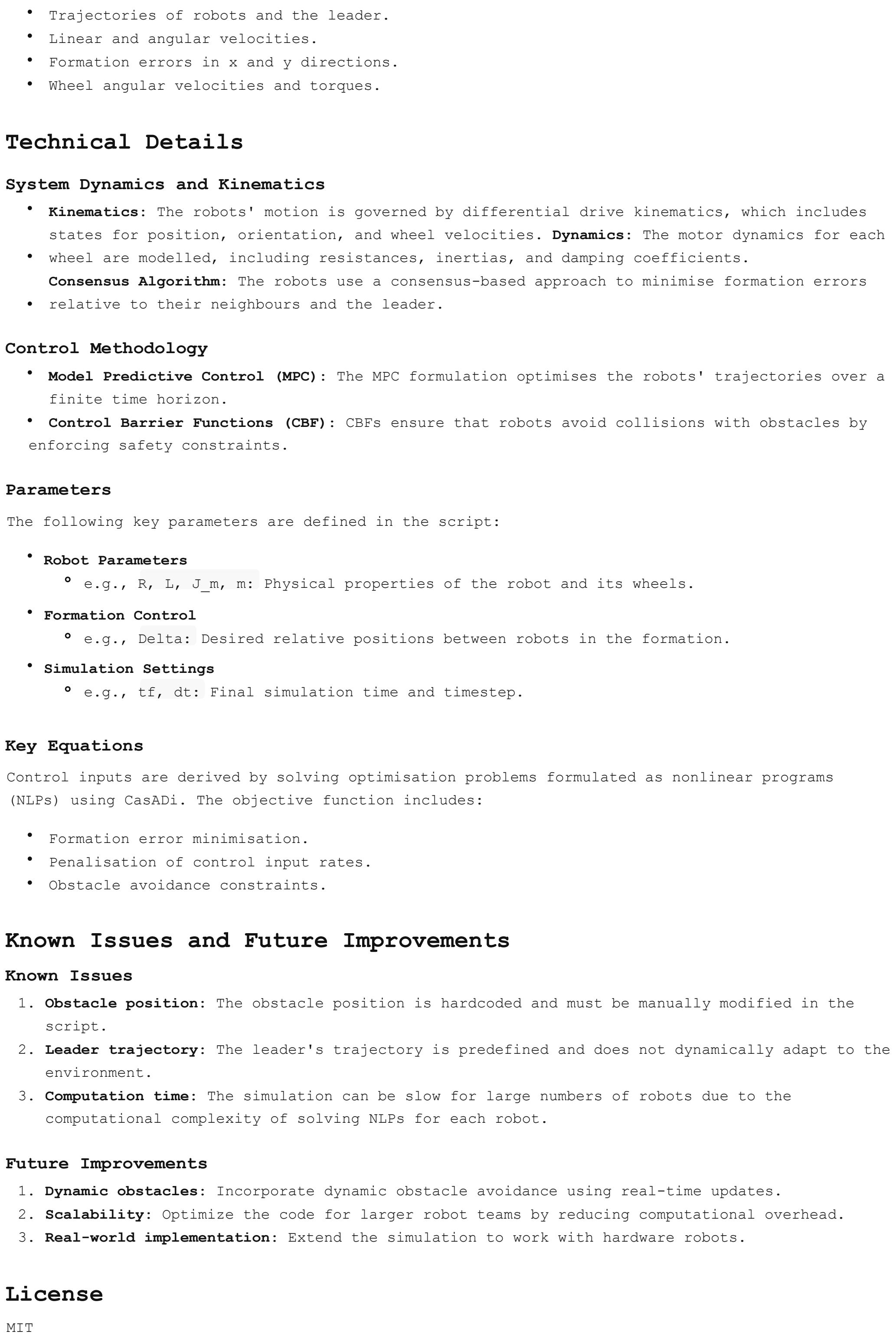
   ```
   bash
   git clone https://github.com/your-username/Consensus-MPC-Formation-Control.git
   cd Consensus-MPC-Formation-Control
   ```

4. **Run the code** :
   - Open MATLAB.
   - Navigate to the folder containing the code file, e.g., Consensus_MPC_formation_control_straight.m.
   - Run the script: `run('Consensus_MPC_formation_control_straight.m')`

How to Run the Code

1. Open the script, e.g., `Consensus_MPC_formation_control_straight.m`, in MATLAB.
2. Modify parameters if necessary:
   - `activate_formation_control`: Set to `true` to enable formation control with consensus.
   - **Obstacle Position**: Adjust `p_obs` to change the obstacle's position.
   - **Simulation time**: Modify t0, tf, and dt for the desired simulation duration and timestep.
3. Run the script and view the generated plots.

The script will produce the following visualisations:



- Trajectories of robots and the leader.
- Linear and angular velocities.
- Formation errors in x and y directions.
- Wheel angular velocities and torques.

Technical Details

System Dynamics and Kinematics

- **Kinematics:** The robots' motion is governed by differential drive kinematics, which includes states for position, orientation, and wheel velocities. **Dynamics:** The motor dynamics for each
- wheel are modelled, including resistances, inertias, and damping coefficients. **Consensus Algorithm:** The robots use a consensus-based approach to minimise formation errors
- relative to their neighbours and the leader.

Control Methodology

- **Model Predictive Control (MPC):** The MPC formulation optimises the robots' trajectories over a finite time horizon.
- **Control Barrier Functions (CBF):** CBFs ensure that robots avoid collisions with obstacles by enforcing safety constraints.

Parameters

The following key parameters are defined in the script:

- **Robot Parameters**
  - e.g., `R, L, J_m, m`: Physical properties of the robot and its wheels.
- **Formation Control**
  - e.g., `Delta`: Desired relative positions between robots in the formation.
- **Simulation Settings**
  - e.g., `tf, dt`: Final simulation time and timestep.

Key Equations

Control inputs are derived by solving optimisation problems formulated as nonlinear programs (NLPs) using CasADi. The objective function includes:

- Formation error minimisation.
- Penalisation of control input rates.
- Obstacle avoidance constraints.

Known Issues and Future Improvements

Known Issues

1. **Obstacle position:** The obstacle position is hardcoded and must be manually modified in the script.
2. **Leader trajectory:** The leader's trajectory is predefined and does not dynamically adapt to the environment.
3. **Computation time:** The simulation can be slow for large numbers of robots due to the computational complexity of solving NLPs for each robot.

Future Improvements

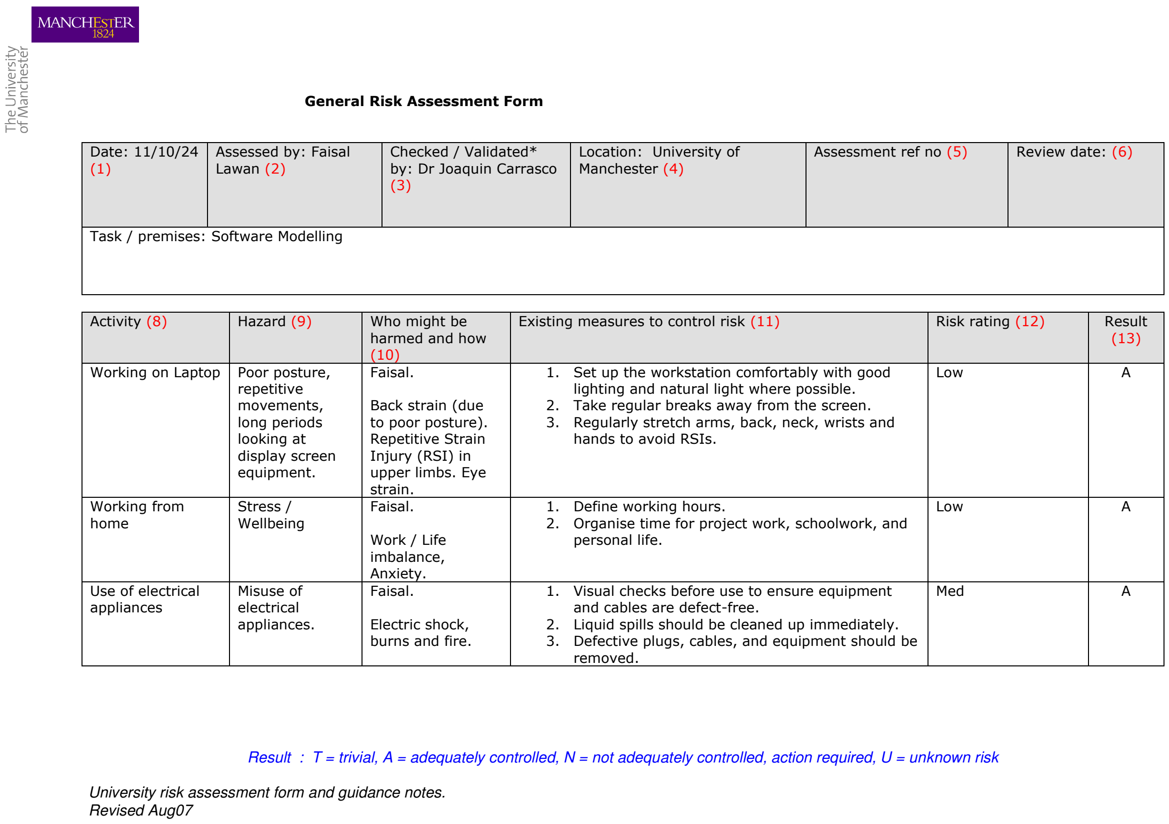
1. **Dynamic obstacles:** Incorporate dynamic obstacle avoidance using real-time updates.
2. **Scalability:** Optimize the code for larger robot teams by reducing computational overhead.
3. **Real-world implementation:** Extend the simulation to work with hardware robots.

License

MIT

[0] You can access the repository at: `https://github.com/FHL-08/Consensus_based_formation_NMPC`.

F Health and Safety Risk Assessment



General Risk Assessment Form

Date: 11/10/24 (1)	Assessed by: Faisal Lawan (2)	Checked / Validated* by: Dr Joaquin Carrasco (3)	Location: University of Manchester (4)	Assessment ref no (5)	Review date: (6)
Task / premises: Software Modelling					

Activity (8)	Hazard (9)	Who might be harmed and how (10)	Existing measures to control risk (11)	Risk rating (12)	Result (13)
Working on Laptop	Poor posture, repetitive movements, long periods looking at display screen equipment.	Faisal. Back strain (due to poor posture). Repetitive Strain Injury (RSI) in upper limbs. Eye strain.	1. Set up the workstation comfortably with good lighting and natural light where possible. 2. Take regular breaks away from the screen. 3. Regularly stretch arms, back, neck, wrists and hands to avoid RSIs.	Low	A
Working from home	Stress / Wellbeing	Faisal. Work / Life imbalance, Anxiety.	1. Define working hours. 2. Organise time for project work, schoolwork, and personal life.	Low	A
Use of electrical appliances	Misuse of electrical appliances.	Faisal. Electric shock, burns and fire.	1. Visual checks before use to ensure equipment and cables are defect-free. 2. Liquid spills should be cleaned up immediately. 3. Defective plugs, cables, and equipment should be removed.	Med	A

Result : T = trivial, A = adequately controlled, N = not adequately controlled, action required, U = unknown risk

University risk assessment form and guidance notes.
Revised Aug07

Activity (8)	Hazard (9)	Who might be harmed and how (10)	Existing measures to control risk (11)	Risk rating (12)	Result (13)
Moving around the lab, home	Obstructions and trip hazards	Faisal. Slips, trips and falls causing physical injury.	1. Trailing cables positioned neatly away from walkways. 2. Cabinet drawers and doors should be closed when not used.	Med	A
Working from home	Fire	Faisal. Risks of burns, smoke inhalation, asphyxiation.	1. All waste, including combustible waste, should be removed regularly. 2. Avoid daisy chaining and do not overload extension leads. 3. Leave the building immediately in the event of a fire and call the fire brigade.	Med	A

Result : T = trivial, A = adequately controlled, N = not adequately controlled, action required, U = unknown risk

University risk assessment form and guidance notes.
Revised Aug07

Action plan (14)				
Ref No	**Further action required**	**Action by whom**	**Action by when**	**Done**

Result : T = trivial, A = adequately controlled, N = not adequately controlled, action required, U = unknown risk

Notes to accompany General Risk Assessment Form

This form is the one recommended by Health & Safety Services, and used on the University's risk assessment training courses. It is strongly suggested that you use it for all new assessments, and when existing assessments are being substantially revised. However, its use is not compulsory. Providing the assessor addresses the same issues; alternative layouts may be used.

(1) **Date** : Insert date that assessment form is completed. The assessment must be valid on that day, and subsequent days, unless circumstances change and amendments are necessary.

(2) **Assessed by** : Insert the name and signature of the assessor. For assessments other than very simple ones, the assessor should have attended the University course on risk assessments (link to STDU)

(3) **Checked / Validated* by** : delete one.

Checked by : Insert the name and signature of someone in a position to check that the assessment has been carried out by a competent person who can identify hazards and assess risk, and that the control measures are reasonable and in place. The checker will normally be a line manager, supervisor, principal investigator, etc. Checking will be appropriate for most risk assessments.

Validated by : Use this for higher risk scenarios, eg where complex calculations have to be validated by another "independent" person who is competent to do so, or where the control measure is a strict permit-to-work procedure requiring thorough preparation of a workplace. The validator should also have attended the University's risk assessment course or equivalent, and will probably be a chartered engineer or professional with expertise in the task being considered. Examples of where validation is required include designs for pressure vessels, load-bearing equipment, lifting equipment carrying personnel or items over populated areas, and similar situations.

(4) **Location** : insert details of the exact location, ie building, floor, room or laboratory etc

(5) **Assessment ref no** : use this to insert any local tracking references used by the school or administrative directorate

(6) **Review date** : insert details of when the assessment will be reviewed as a matter of routine. This might be in 1 year's time, at the end of a short programme of work, or longer period if risks are known to be stable. Note that any assessment must be reviewed if there are any significant changes – to the work activity, the vicinity, the people exposed to the risk, etc

(7) **Task / premises** : insert a brief summary of the task, eg typical office activities such as filing, DSE work, lifting and moving small objects, use of misc electrical equipment. Or, research project [title] involving the use of typical laboratory hardware, including fume cupboards, hot plates, ovens, analysis equipment, flammable solvents, etc.

(8) **Activity** : use the column to describe each separate activity covered by the assessment. The number of rows is unlimited, although how many are used for one assessment will depend on how the task / premises is sub-divided. For laboratory work, activities in one particular lab or for one particular project might include; use of gas cylinders, use of fume cupboard, use of computer or other electrical equipment, use of lab ovens, hot plates or heaters, use of substances hazardous to health, etc

(9) **Hazard** : for each activity, list the hazards. Remember to look at hazards that are not immediately obvious. For example, use of a lathe will require identification of the machine hazards, but also identification of hazards associated with the use of cutting oils (dermatitis), poor lighting, slipping on oil leaks, etc. The same activity might well have several hazards associated with it. Assessment of simple chemical risks (eg use of cleaning chemicals in accordance with the instructions on the bottle) may be recorded here. More complex COSHH

assessments eg for laboratory processes, should be recorded on the specific COSHH forms (link).

(10) **Who might be harmed and how** : insert everyone who might be affected by the activity and specify groups particularly at risk. Remember those who are not immediately involved in the work, including cleaners, young persons on work experience, maintenance contractors, Estates personnel carrying out routine maintenance and other work. Remember also that the risks for different groups will vary. Eg someone who needs to repair a laser may need to expose the beam path more than users of the laser would do. Vulnerable groups could include children on organised visits, someone who is pregnant, or employees and students with known disabilities or health conditions (this is not a definitive list).

For each group, describe how harm might come about, eg an obstruction or wet patch on an exit route is a hazard that might cause a trip and fall; use of electrical equipment might give rise to a risk of electric shock; use of a ultraviolet light source could burn eyes or skin.

(11) **Existing measures to control the risk** : list all measures that already mitigate the risk. Many of these will have been implemented for other reasons, but should nevertheless be recognised as means of controlling risk. For example, restricting access to laboratories or machine rooms for security reasons also controls the risk of unauthorised and unskilled access to dangerous equipment. A standard operating procedure or local rules (eg for work with ionising radiation, lasers or biological hazards) will often address risks. Some specific hazards may require detailed assessments in accordance with specific legislation (eg COSHH, DSEAR, manual handling, DSE work). Where this is the case, and a detailed assessment has already been done in another format, the master risk assessment can simply cross-reference to other documentation. For example, the activity might be use of a carcinogen, the hazard might be exposure to hazardous substances, the existing control measures might all be listed in a COSHH assessment. Controls might also include use of qualified and/or experienced staff who are competent to carry out certain tasks; an action plan might include training requirements for other people who will be carrying out those tasks.

(12) **Risk Rating** : the simplest form of risk assessment is to rate the remaining risk as high, medium or low, depending on how likely the activity is to cause harm and how serious that harm might be.

The risk is **LOW** - if it is most unlikely that harm would arise under the controlled conditions listed, and even if exposure occurred, the injury would be relatively slight.
The risk is **MEDIUM** - if it is more likely that harm might actually occur and the outcome could be more serious (eg some time off work, or a minor physical injury.
The risk is **HIGH** - if injury is likely to arise (eg there have been previous incidents, the situation looks like an accident waiting to happen) and that injury might be serious (broken bones, trip to the hospital, loss of consciousness), or even a fatality.

Schools or administrative directorates may choose to use other rating systems. Typical amongst these are matrices (of 3x3, 4x4, 5x5 or even more complex) which require the assessor to select a numerical rating for both "likelihood that harm will arise" and "severity of that harm". These may give a spurious sense of accuracy and reliability – none are based on quantitative methods. There are methods of estimating risk quantitatively, and these may be appropriate for complex design of load bearing structures and the like. Advice on methods of risk assessment is available from HSS. Whatever system of assessment is adopted, it is **essential** that the assessor has received suitable training and is familiar with the meaning of the terms (or numbers) used.

(13) **Result** : this stage of assessment is often overlooked, but is probably the most important. Assigning a number or rating to a risk does not mean that the risk is necessarily adequately controlled. The options for this column are:

T = trivial risk. Use for very low risk activities to show that you have correctly identified a hazard, but that in the particular circumstances, the risk is insignificant.

A = adequately controlled, no further action necessary. If your control measures lead you to conclude that the risk is low, and that all legislative requirements have been met (and University policies complied with), then insert A in this column.

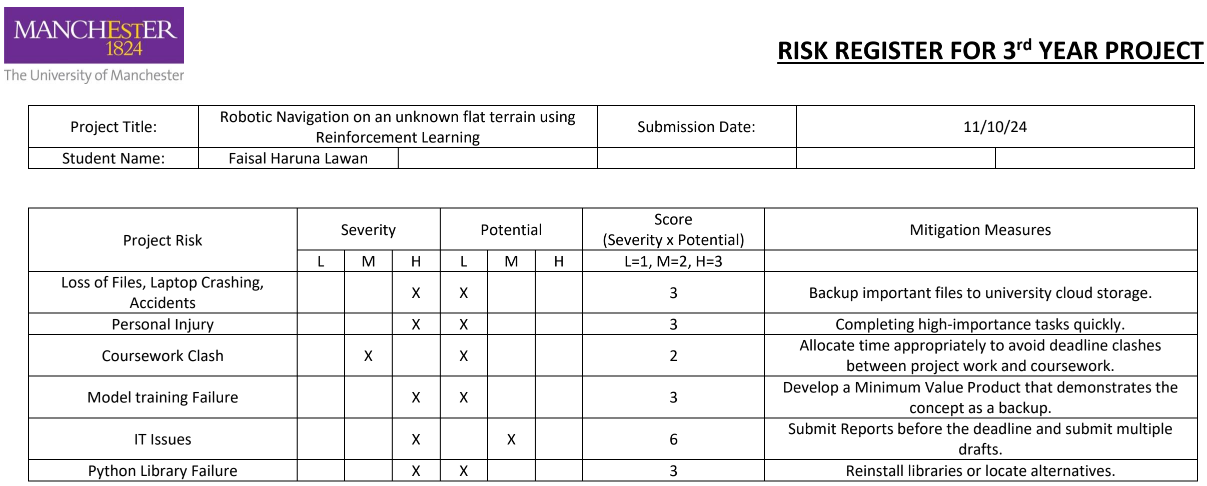
N = not adequately controlled, actions required. Sometimes, particularly when setting up new procedures or adapting existing processes, the risk assessment might identify that the risk is high or medium when it is capable of being reduced by methods that are reasonably practicable. In these cases, an action plan is required. The plan should list the actions necessary, who they are to be carried out by, a date for completing the actions, and a signature box for the assessor to sign off that the action(s) has been satisfactorily completed. Some action plans will be complex documents; others may be one or two actions that can be completed with a short timescale.

U = unable to decide. Further information required. Use this designation if the assessor is unable to complete any of the boxes, for any reason. Sometimes, additional information can be obtained readily (eg from equipment or chemicals suppliers, specialist University advisors) but sometimes detailed and prolonged enquiries might be required. Eg is someone is moving a research programme from a research establishment overseas where health and safety legislation is very different from that in the UK.

For T and A results, the assessment is complete.
For N or U results, more work is required before the assessment can be signed off.

(14) **Action Plan**. Include details of any actions necessary in order to meet the requirements of the information in Section 11 'Existing measures to control the risk'. Identify someone who will be responsible for ensuring the action is taken and the date by which this should be completed. Put the date when the action has been completed in the final column.

G Risk Register

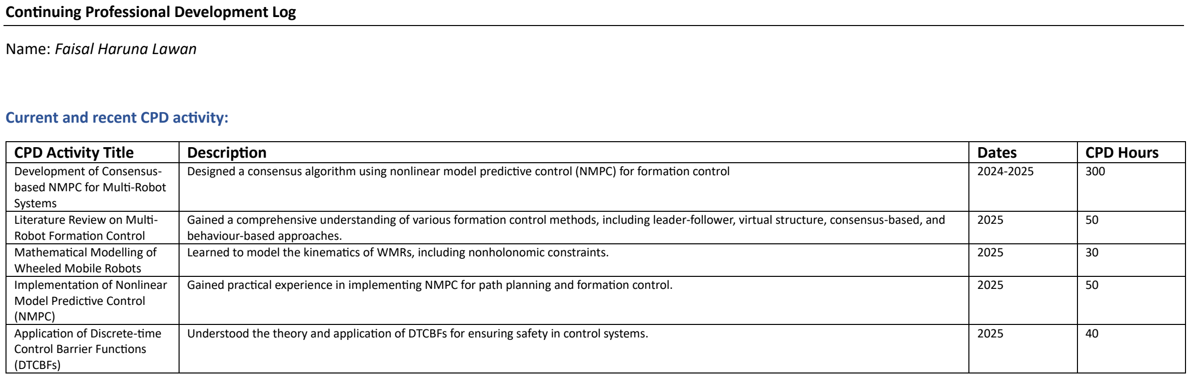


RISK REGISTER FOR 3rd YEAR PROJECT

Project Title:	Robotic Navigation on an unknown flat terrain using Reinforcement Learning		Submission Date:	11/10/24	
Student Name:	Faisal Haruna Lawan				

Project Risk	Severity			Potential			Score (Severity x Potential)	Mitigation Measures
	L	M	H	L	M	H	L=1, M=2, H=3	
Loss of Files, Laptop Crashing, Accidents			X	X			3	Backup important files to university cloud storage.
Personal Injury			X	X			3	Completing high-importance tasks quickly.
Coursework Clash		X		X			2	Allocate time appropriately to avoid deadline clashes between project work and coursework.
Model training Failure			X	X			3	Develop a Minimum Value Product that demonstrates the concept as a backup.
IT Issues			X		X		6	Submit Reports before the deadline and submit multiple drafts.
Python Library Failure			X	X			3	Reinstall libraries or locate alternatives.

H Continuing Professional Development



Continuing Professional Development Log

Name: *Faisal Haruna Lawan*

Current and recent CPD activity:

CPD Activity Title	Description	Dates	CPD Hours
Development of Consensus-based NMPC for Multi-Robot Systems	Designed a consensus algorithm using nonlinear model predictive control (NMPC) for formation control	2024-2025	300
Literature Review on Multi-Robot Formation Control	Gained a comprehensive understanding of various formation control methods, including leader-follower, virtual structure, consensus-based, and behaviour-based approaches.	2025	50
Mathematical Modelling of Wheeled Mobile Robots	Learned to model the kinematics of WMRs, including nonholonomic constraints.	2025	30
Implementation of Nonlinear Model Predictive Control (NMPC)	Gained practical experience in implementing NMPC for path planning and formation control.	2025	50
Application of Discrete-time Control Barrier Functions (DTCBFs)	Understood the theory and application of DTCBFs for ensuring safety in control systems.	2025	40