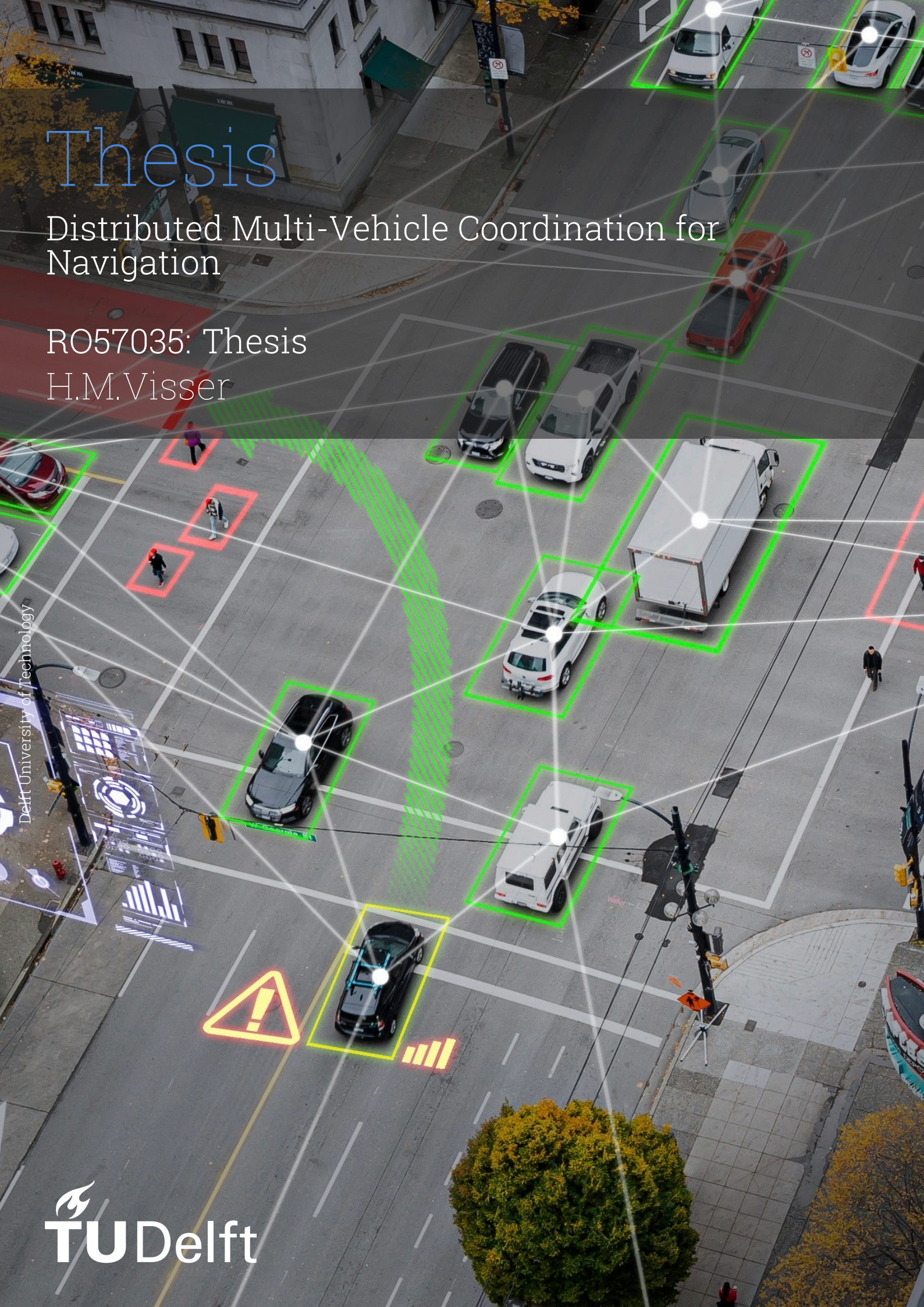


Thesis

Distributed Multi-Vehicle Coordination for Navigation

RO57035: Thesis

H.M.Visser

Delft University of Technology

TU Delft

Thesis

Distributed Multi-Vehicle Coordination for Navigation

by

H.M.Visser

Student Name	Student Number
Hubert Visser	4901843

Instructor: Dr. Laura Ferranti
Project Duration: July, 2024 - July, 2025
Faculty: Faculty of Mechanical Engineering, Delft

Cover: Connected autonomous vehicles by Trimble Applanix (Modified)
Style: TU Delft Report Style, with modifications by Daan Zwaneveld

TU Delft

Abstract

As self-driving vehicles progress toward real-world deployment, efficient and reliable motion planning in dynamic multi-agent environments becomes increasingly essential. This work addresses this challenge by advancing the field of nonlinear distributed model predictive control (NMPC) for autonomous multi-vehicle coordination. The approach focuses on complex, interaction-dense scenarios where space is limited. To model vehicle shapes and collision boundaries efficiently it uses polytopic set representations to approximate vehicle geometries and collision boundaries instead of conventional shapes such as ellipsoids.

We propose a novel distributed NMPC approach for navigation in tight environments. The goal is to enable coordinated motion planning for multiple autonomous vehicles in dense traffic scenarios. This can be easily modelled with centralised formulations, however, considering the scale of the road network they become computationally intractable as the number of agents grows. In distributed approaches instead each vehicle solves its local part of the problem which scales linearly and is therefore better suited for these kind of environments. So building on an existing distributed method we enhance its structure to improve scalability, safety, and performance in cooperative autonomous driving tasks. The research is guided by three objectives: (RO1) to reproduce a known distributed baseline algorithm using the DART (Delft's Autonomous-driving Robotic Testbed) vehicle model [10], the Model Predictive Contouring Control (MPCC) [7] tracking objective, and the ACADOS solver framework [14]; (RO2) to develop a novel distributed MPC-based algorithm that improves inter-vehicle spacing and tracking performance while generalizing beyond predefined reference trajectories; and (RO3) to prepare the algorithm for validation on a real-world robotic testbed.

The primary contributions of this work include: (C1) successful reproduction of the distributed baseline using open-source tools and realistic vehicle modelling; (C2) development of two enhanced distributed algorithms, Distributed Model Predictive Contouring Control with Relaxed Collision Avoidance (DMPC-RCA) and DMPC-RCA with consensus (DMPC-RCA-C), that demonstrate superior performance in tracking accuracy and safety margins; and (C3) integration of these algorithms with the DART hardware platform for future experimental validation. The repository of these approaches can be found in *https://github.com/HubertVisser/multi-vehicle-coordination-algorithm.git*

Performance is evaluated in two representative scenarios, a merging situation and a T-junction, with the centralised NMPC approach serving as the performance benchmark. To evaluate the performance of the proposed designs, three key metrics are used: accumulated tracking cost, computation time, and minimum inter-agent distance. The results show that both proposed methods achieve tracking costs comparable to the centralised controller, while significantly outperforming the distributed baseline method. Notably, the inclusion of a consensus term yields no substantial improvement in performance over the non-consensus version.

To conclude, the proposed approaches offer strong potential for scalable, safe, and efficient multi-agent motion planning, moving one step closer to the deployment of fully cooperative autonomous driving on public roads.

Contents

1

Introduction

Autonomous driving technology is becoming an increasingly integral part of modern transportation, gradually making its way into our daily lives. Many new vehicles are now equipped with advanced driver assistance systems (ADAS), which provide features like adaptive cruise control, lane-keeping assistance, and automated parking, enhancing safety and convenience for drivers. As this technology advances, fully autonomous taxis have started operating in select cities on a small scale [9], offering a glimpse into a future where transportation is safer, more accessible, and less reliant on human intervention. These developments hold transformative potential across economic, environmental, social and ethical domains [1]. Economically, self-driving vehicles can significantly increase road capacity by enabling smoother traffic flow and reducing congestion, ultimately saving time for travellers. Additionally, autonomous systems can optimize fuel consumption, leading to cost savings and reduced reliance on non-renewable resources. Environmentally, the improved efficiency of autonomous vehicles leads to lower emissions, contributing to better air quality and addressing climate change. The increased road capacity reduces the need for extensive road infrastructure, requiring less land for parking and road surfaces. Socially, the reduction in human error associated with autonomous vehicles is expected to lead to fewer accidents, which not only improves public safety but also reduces medical expenses and the societal impact of road injuries. Ethically, autonomous driving offers mobility to individuals who may currently face limitations, such as the elderly and disabled, fostering greater independence and accessibility. Together, these benefits illustrate the far-reaching societal relevance of autonomous driving technology.

To realize a future where autonomous vehicles operate safely and efficiently in complex urban environments, the development of advanced control systems is essential. Centralised controllers, while effective in small-scale scenarios, face significant challenges when scaled to large fleets of self-driving cars, including computational intractability and communication bottlenecks that make real-time optimisation infeasible. As a result, there is a growing need for distributed optimisation schemes that can decentralize decision-making and leverage local information. These methods offer various trade-offs between computational efficiency, scalability, and optimality, making them suitable for different aspects of decentralised planning in multi-agent systems. State-of-the-art motion planning techniques for multi-vehicle systems include sampling-based algorithms such as Rapidly-exploring Random Tree (RRT) [8], graph-based methods like A* [5], optimisation-based approaches such as Model Predictive Control (MPC) [11] and Model Predictive Path Integral (MPPI) control [15], as well as artificial potential field methods [4], [13]. While these methods have demonstrated effectiveness in structured or low-density environments, they face significant limitations in online coordination for dense traffic or tight scenarios. [16] Sampling-based and graph-based planners often suffer from high computational complexity and slow replanning, making real-time operation challenging as the number of vehicles increases [12]. optimisation-based approaches, including MPC and MPPI, though capable of handling constraints and vehicle dynamics, become computationally intractable for large fleets and are often centralised, which is impractical for distributed traffic [17]. Potential field methods can struggle with local minima and lack explicit coordination mechanisms, leading to deadlocks or suboptimal behaviours

in crowded environments [6]. Furthermore, many existing approaches assume perfect communication and synchronization, which is rarely achievable in real-world scenarios where delays, packet loss, and asynchrony are common [2]. These limitations highlight the need for scalable, distributed, and robust planning frameworks that can handle real-time coordination and uncertainty in practical multi-vehicle traffic environments. This research addresses this need by proposing a distributed Model Predictive Control (MPC)-based framework for coordinated multi-vehicle navigation, particularly in constrained or tight environments where cooperation and adaptability are crucial.

1.1. Research Objectives and Contributions

This work is guided by the following research objectives, which aim to advance the field of distributed multi-vehicle control through performance improvement and broader applicability.

- **RO1:** Can a known baseline approach for nonlinear distributed multi-vehicle control be successfully reproduced using the DART [10] vehicle model, the Model Predictive Contouring Control (MPCC) [7] tracking objective, and the open-source ACADOS [14] solver?
- **RO2:** Can a distributed Model Predictive Control (MPC)-based algorithm be developed that not only improves inter-vehicle spacing and trajectory tracking accuracy over the baseline, but also generalises beyond reliance on predefined reference trajectories for use in diverse and dynamic environments?
- **RO3:** Can the improved algorithm be validated on a real-world testbed beyond simulation?

In accordance with the stated research objectives, the primary contributions of this work are outlined as follows:

- **C1:** The baseline distributed control algorithm was successfully reproduced using the DART [10] vehicle model, the Model Predictive Contouring Control (MPCC) [7] tracking objective and the open-source ACADOS [14] solver.
- **C2:** A novel distributed control algorithm was developed and implemented, demonstrating improved performance over the distributed baseline in terms of trajectory tracking accuracy and inter-vehicle spacing, while also generalizing beyond the use of predefined reference trajectories to enhance applicability in diverse and dynamic environments.
- **C3:** The algorithm has been prepared for deployment and testing on the DART experimental platform, supporting future validation in real-world scenarios.

1.2. Outline

This thesis is structured to guide the reader through the development and evaluation of distributed control algorithms for autonomous vehicle coordination. The outline of the chapters is as follows:

The first chapter, Preliminaries 2, introduces the foundational concepts and tools used throughout the research. It presents the formulation of the Model Predictive Contouring Control (MPCC) tracking cost, describes both the centralised and distributed baseline algorithms, and introduces the DART platform, a modular framework developed for testing autonomous control algorithms, including details of the vehicle model employed. Additionally, the open-source optimisation solver ACADOS, which is used for real-time implementation of the controllers, is discussed.

Chapter 3, Proposed Approach, elaborates on the design choices made in implementing the baseline algorithms and the rationale behind them. It then introduces the proposed distributed control algorithms, detailing their conceptual motivation and expected performance improvements.

In chapter 4 Results, the performance of the proposed algorithms is rigorously evaluated. A consistent set of metrics is applied to compare the new approaches against the baseline, including tracking performance, computation time, and inter-vehicular distance. The algorithms are tested in two distinct scenarios to demonstrate robustness and generalizability.

The Discussion, chapter 5, reflects on the findings, and identifies limitations.

Finally, the Conclusion, chapter 6, provides a concise overview of the research, offers directions for future work and concludes the thesis.

2

Preliminaries

The definitions needed in the remainder of the thesis are given below.

2.1. Model Predictive Contouring Control

Model Predictive Contouring Control (MPCC) [7] is a state-of-the-art technique for reference path tracking in autonomous vehicle control. It provides a flexible trade-off between tracking accuracy and traversal speed, which is regulated through the weighting of different objective components, specifically, the contouring and lag errors. This makes MPCC particularly suitable for tasks requiring precise navigation in dynamic or constrained environments.

To better understand how the MPCC controller evaluates tracking performance, consider the geometric relationship illustrated in Fig. 2.1. The black dot at (x_k, y_k) represents the vehicle's current position, while the black curve denotes the reference path, which is parametrized by arc length θ_k. The orientation of the path at a given point is indicated by the angle $\phi(\theta_k)$, which corresponds to the tangent direction of the path at θ_k. The errors are computed and approximated with respect to the nearest point on the path, $(x_d(\theta_r), y_d(\theta_r))$, and are decomposed into two components: the **contouring error** ϵ_k^c, representing the lateral deviation from the path, and the **lag error** ϵ_k^l, representing the longitudinal deviation along the path. The errors are approximated via projection onto the local path frame and are denoted by $\hat{\epsilon}_k^c$ and $\hat{\epsilon}_k^l$.

The MPCC controller minimizes a composite objective that balances speed, accuracy, and control smoothness. The cost function comprises four main components:

1. **Velocity Deviation:** The deviation of the current velocity v_k from a reference velocity v_{ref} is penalized to ensure speed regulation:
$$v_k - v_{ref} \tag{2.1}$$

2. **Contouring Error ε^c:** This term captures the orthogonal (normal) deviation from the vehicle's position to the desired path (see Fig 2.1). It is approximated as:
$$\hat{\epsilon}^c(\xi_k, \theta_k) = \sin\phi(\theta_k)(x_k - x_d(\theta_k)) - \cos\phi(\theta_k)(y_k - y_d(\theta_k)) \tag{2.2}$$
where (x_k, y_k) is the vehicle position, $(x_d(\theta_k), y_d(\theta_k))$ is the reference point on the path, and $\phi(\theta_k)$ is the orientation of the path at arc length parameter θ_k.

3. **Lag Error ε^l:** This measures the longitudinal deviation (along the path) between the vehicle's current position and the corresponding reference point (see Fig 2.1) and is approximated as:
$$\hat{\varepsilon}^l(\xi_k, \theta_k) = -\cos\phi(\theta_k)(x_k - x_d(\theta_k)) - \sin\phi(\theta_k)(y_k - y_d(\theta_k)) \tag{2.3}$$
The reference path parameter θ_k is updated based on the current velocity:
$$\theta_{k+1} = \theta_k + v_k, \quad v_k \in [0, v_{max}], \quad v_{max} > 0 \tag{2.4}$$

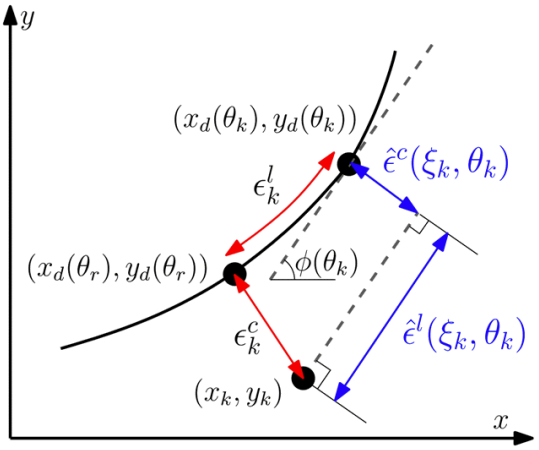


Figure 2.1: Illustration of the contouring error ϵ_k^c and lag error ϵ_k^l at time step k. (Adapted from [7])

4. **Control Effort:** To promote smooth control inputs and reduce actuator usage, the squared norm of the control input vector $\mathbf{u}$ is included in the cost. This penalizes large or abrupt control inputs via a quadratic term:

$$\mathbf{u}^\top \mathbf{Q}_u \mathbf{u}$$

Here, $\mathbf{Q}_u \in \mathbb{R}^{m \times m}$ is a symmetric positive semi-definite weighting matrix that assigns relative importance to each control input component. For example, in this research, the control input is defined as $\mathbf{u} = \begin{bmatrix} th & st \end{bmatrix}^\top$, representing throttle and steering input. Then, $\mathbf{Q}_u$ could be defined as:

$$\mathbf{Q}_u = \begin{bmatrix} q_{th} & 0 \\ 0 & q_{st} \end{bmatrix}$$

where q_{th} and q_{st} are scalar weights that penalize excessive throttle and steering effort, respectively.

The complete objective function minimized by the MPCC controller at each time step k is given by:

$$J_k = q_1(v_k - v_{ref})^2 + q_2(\hat{\varepsilon}_k^c)^2 + q_3(\hat{\varepsilon}_k^l)^2 + \mathbf{u}_k^\top \mathbf{Q}_u \mathbf{u}_k \tag{2.5}$$

Having defined the MPCC framework for single-vehicle path tracking, we now shift focus to multi-agent coordination. The following section introduces the baseline distributed algorithm upon which our proposed method builds.

2.2. Baseline: Distributed Multi-Vehicle Coordination Algorithm

The baseline for this research is the distributed multi-vehicle coordination algorithm proposed by Firoozi et al. [3]. This algorithm provides a scalable framework for coordinating multiple autonomous vehicles by decomposing a centralised optimisation problem into distributed subproblems, making it suitable for real-world deployment where computational scalability is critical.

The core idea behind the algorithm is to partition a centralised coordination problem into smaller, agent-specific optimisations. Each vehicle independently solves two optimisation problems at every time step: a Nonlinear Model Predictive Control (NMPC) problem for reference trajectory tracking, and a Collision Avoidance (CA) problem that ensures safety in multi-agent scenarios. The difference between the centralised structure and the distributed structure is shown in figure 2.2. This distributed structure allows computational complexity to grow linearly with the number of agents, significantly improving scalability compared to centralised approaches.

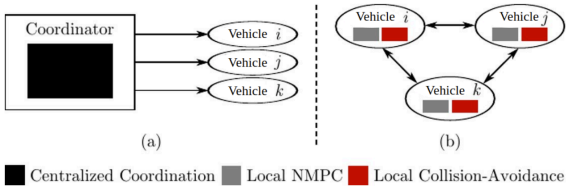


Figure 2.2: Centralised design (a) vs. distributed design (b) for multi-vehicle coordination. (Adapted from [3])

A key component of the algorithm is the reformulation of the collision avoidance constraint based on strong duality theory. As detailed in Section 2.2.1, this reformulation enables the decomposition of the centralised problem into two interdependent but separately solvable optimisation problems:

1. **The NMPC subproblem**, which optimizes the tracking performance for each vehicle using a local objective (e.g., MPCC-based).
2. **The CA subproblem**, which updates the dual variables associated with the collision avoidance constraints to ensure inter-vehicle safety.

Furthermore, the use of polytopic sets to represent vehicle shapes in the collision avoidance formulation (see Section 2.2.2) makes the algorithm particularly well-suited for navigation in constrained environments. For instance, in road-like scenarios where vehicles must remain between the lanes, this representation provides an efficient and non-conservative approximation for safe interaction among agents.

This distributed structure forms the basis upon which the proposed algorithms in this thesis are built, with the goal of improving both, tracking performance and general applicability.

The following subsections present the reformulation of the collision avoidance constraint and the definition and use of polytopic sets to model the vehicles. Both are required to construct the centralised and distributed formulations.

2.2.1. Reformulation of the Collision Avoidance Constraint

Let P_1 and P_2 be two convex polytopic sets, defined respectively as:

$$P_1 = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}_1\mathbf{x} \leq \mathbf{b}_1\}, \quad P_2 = \{\mathbf{y} \in \mathbb{R}^n \mid \mathbf{A}_2\mathbf{y} \leq \mathbf{b}_2\}.$$

The Euclidean distance between these sets is given by the following optimisation problem:

$$\text{dist}(P_1, P_2) = \min_{\mathbf{x},\mathbf{y}} \left\{ \|\mathbf{x} - \mathbf{y}\|^2 \mid \mathbf{A}_1\mathbf{x} \leq \mathbf{b}_1,\ \mathbf{A}_2\mathbf{y} \leq \mathbf{b}_2 \right\}.$$

Incorporating a minimum distance constraint $\text{dist}(P_1, P_2) \geq d_{\min}$ directly into a Nonlinear Model Predictive Control (NMPC) formulation is computationally expensive due to the nested nature of this optimisation. To reduce complexity, the approach proposed by Firoozi et al. [3] leverages strong duality theory to reformulate this constraint using its dual representation. The resulting dual problem is given by:

$$\begin{aligned}
\text{dist}(P_1, P_2) := \max_{\lambda_{12},\lambda_{21},\mathbf{s}} \quad & -\mathbf{b}_1^\top\lambda_{12} - \mathbf{b}_2^\top\lambda_{21} && \text{(2.6a)}\\
\text{subject to} \quad & \mathbf{A}_1^\top\lambda_{12} + \mathbf{s} = 0, && \text{(2.6b)}\\
& \mathbf{A}_2^\top\lambda_{21} - \mathbf{s} = 0, && \text{(2.6c)}\\
& \|\mathbf{s}\|_2 \leq 1, && \text{(2.6d)}\\
& \lambda_{12} \geq 0, \quad \lambda_{21} \geq 0. && \text{(2.6e)}
\end{aligned}$$

Here, λ_{12} and λ_{21} are dual variables corresponding to the polytopic constraints of P_1 and P_2, and dual variable s captures the separating hyperplane direction between the sets. This dual formulation

maintains the same optimal value as the primal distance problem and allows the collision avoidance constraint to be imposed efficiently within a distributed framework.

This reformulation plays a critical role in enabling the decomposition of the centralised NMPC problem into smaller, agent-level problems, as detailed in Section 2.2.

To effectively handle tight-space scenarios, the vehicles are represented as polytopic sets, described in the following section.

2.2.2. Polytopic Sets

To model the DART robots within the collision avoidance framework, an oriented rectangular polytope is used. The polytopic definition used in this research is specified below.

A polytopic set is defined as:

$$P = \{\mathbf{p} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{p} \leq \mathbf{b}\}$$

The orientation of the vehicle is incorporated through a rotation matrix defined by its heading angle θ:

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{2.7}$$

Using this rotation matrix, the constraint matrix $A(\theta)$ is constructed as:

$$A(\theta) = \begin{bmatrix} R(\theta)^\top \\ -R(\theta)^\top \end{bmatrix} \tag{2.8}$$

The corresponding vector $\mathbf{b}$ defining the bounds of the polytope is given by:

$$\mathbf{b} = \begin{bmatrix} l/2 \\ w/2 \\ l/2 \\ w/2 \end{bmatrix} + A(\theta) \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.9}$$

Here, l and w denote the length and width of the DART robot, respectively (see Section 2.4). The construction of polytopic set yields a rotated rectangular polytope that accurately captures the robot's shape and orientation in the world frame. These polytopic sets are used in the collision avoidance constraints to ensure safe multi-vehicle coordination in constrained environments.

2.2.3. Centralised Approach

In the centralised formulation, a single optimisation problem coordinates the trajectories of all agents simultaneously. This approach has the advantage of full information but suffers from scalability issues due to the rapid growth in problem size with the number of agents. Next, the objectives and the constraints are presented.

Objective Modules:

- *Low-level planning objective*. To minimize deviation from a higher-level reference.
- *Minimization of dual variables* $\lambda_{ij}, \lambda_{ji}, s_{ij}$: Encourages tight but safe distances between vehicles while contributing to the dual-based collision avoidance.

Constraint Modules:

- *Vehicle dynamics constraints*: Ensure that each agent's motion complies with its respective nonlinear dynamics model.
- *State and input feasibility constraints*: Ensures each trajectory respects system limits.
- *Collision avoidance constraints*: Enforced using the dual formulation from Section 2.2.1.

The resulting centralised optimisation problem is formulated as:

$$\min_{\substack{\mathbf{u}^i(\cdot|t),\,\lambda_{ij}(\cdot|t),\\ \lambda_{ji}(\cdot|t),\,s_{ij}(\cdot|t)}} \quad \sum_{i=1}^{M} J^i(\mathbf{z}^i, \mathbf{u}^i) \tag{2.10a}$$

$$\text{subject to} \quad \mathbf{z}^i(k+1|t) = f(\mathbf{z}^i(k|t), \mathbf{u}^i(k|t)), \tag{2.10b}$$

$$\mathbf{z}^i(0|t) = \mathbf{z}^i(t), \tag{2.10c}$$

$$\mathbf{z}^i(k|t) \in \mathcal{Z}, \quad \mathbf{u}^i(k|t) \in \mathcal{U}, \tag{2.10d}$$

$$-\mathbf{b}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) - \mathbf{b}^j(\mathbf{z}^j(k|t))^\top \lambda_{ji}(k|t) \geq d_{\min}, \tag{2.10e}$$

$$\mathbf{A}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) + s_{ij}(k|t) = 0, \tag{2.10f}$$

$$\mathbf{A}^j(\mathbf{z}^j(k|t))^\top \lambda_{ji}(k|t) - s_{ij}(k|t) = 0, \tag{2.10g}$$

$$\lambda_{ij}(k|t) \geq 0, \quad \lambda_{ji}(k|t) \geq 0, \quad \|s_{ij}(k|t)\|_2 \leq 1, \tag{2.10h}$$

$$\forall i \in \mathcal{V},\ j \in \mathcal{N}_i,\ k \in \{1, 2, \ldots, N\}. \tag{2.10i}$$

In the formulation above:

- $f(\cdot)$ in (2.10b) represents the nonlinear system dynamics of each agent.
- $\mathcal{Z}$ and $\mathcal{U}$ denote the state and control input feasible sets, respectively.
- d_{min} is the set constrained minimum distance between the agents.
- $\mathcal{V}$ is the set of all agents, and $\mathcal{N}_i$ represents the set of neighbours of agent i.

This centralised approach serves as a benchmark for assessing the scalability and performance of the distributed alternatives proposed in this thesis.

2.2.4. Distributed Approach

In the distributed formulation, each agent $i \in \mathcal{V}$ independently solves its own optimisation problem based on local information and shared variables with its neighbouring agents $j \in \mathcal{N}_i$.

The optimisation is split into two subproblems for each agent: one Nonlinear Model Predictive Control (NMPC) problem for trajectory tracking, and one Collision Avoidance (CA) problem to ensure the collision avoidance constraints are satisfied through dual variable updates.

NMPC Subproblem

Each agent solves the following NMPC optimisation, assuming the dual variables from the previous iteration are fixed:

$$\min_{\mathbf{u}^i(\cdot|t)} \quad \sum_{i=1}^{M} J^i(\mathbf{z}^i, \mathbf{u}^i) \tag{2.11a}$$

$$\text{subject to} \quad \mathbf{z}^i(k+1|t) = f(\mathbf{z}^i(k|t), \mathbf{u}^i(k|t)), \tag{2.11b}$$

$$\mathbf{z}^i(0|t) = \mathbf{z}^i(t), \tag{2.11c}$$

$$\mathbf{z}^i(k|t) \in \mathcal{Z}, \quad \mathbf{u}^i(k|t) \in \mathcal{U}, \tag{2.11d}$$

$$-\mathbf{b}^i(\mathbf{z}^i(k|t))^\top \bar{\lambda}_{ij}(k|t) - \mathbf{b}^j(\mathbf{z}^j(k|t))^\top \bar{\lambda}_{ji}(k|t) \geq d_{\min}, \tag{2.11e}$$

$$\mathbf{A}^i(\mathbf{z}^i(k|t))^\top \bar{\lambda}_{ij}(k|t) + \bar{s}_{ij}(k|t) = 0, \tag{2.11f}$$

$$\forall k \in \{1, 2, \ldots, N\}$$

Here, the bar notation $\bar{(\cdot)}$ denotes variables treated as constants during this optimisation, passed from the CA module of the previous time step.

CA Subproblem

After each NMPC optimisation, each agent updates its collision avoidance dual variables by solving the following maximization problem:

$$\max_{\lambda_{ij}(\cdot|t),\, \lambda_{ji}(\cdot|t),\, s_{ij}(\cdot|t)} \quad -\overline{\mathbf{b}}^i(\overline{\mathbf{z}}^i(k|t))^\top \lambda_{ij}(k|t) - \overline{\mathbf{b}}^j(\overline{\mathbf{z}}^j(k|t))^\top \lambda_{ji}(k|t) \tag{2.12a}$$

$$\text{subject to} \quad \overline{\mathbf{A}}^i(\overline{\mathbf{z}}^i(k|t))^\top \lambda_{ij}(k|t) + s_{ij}(k|t) = 0, \tag{2.12b}$$

$$\overline{\mathbf{A}}^j(\overline{\mathbf{z}}^j(k|t))^\top \lambda_{ji}(k|t) - s_{ij}(k|t) = 0, \tag{2.12c}$$

$$-\overline{\mathbf{b}}^i(\overline{\mathbf{z}}^i(k|t))^\top \lambda_{ij}(k|t) - \overline{\mathbf{b}}^j(\overline{\mathbf{z}}^j(k|t))^\top \lambda_{ji}(k|t) \geq d_{\min}, \tag{2.12d}$$

$$\|s_{ij}(k|t)\|_2 \leq 1, \tag{2.12e}$$

$$\lambda_{ij}(k|t) \geq 0, \quad \lambda_{ji}(k|t) \geq 0, \tag{2.12f}$$

$$\forall k \in \{1, 2, \ldots, N\}.$$

This dual update ensures the safety constraints in the primal NMPC formulation are respected. The decoupling into two alternating subproblems enables distributed computation, reducing the centralised computational burden while maintaining coordination via dual variable exchange between neighbours.

2.2.5. Cost Function

The cost function used in the NMPC formulation in [3] is presented below. This objective function penalizes deviation from a reference trajectory, control effort, and control input rate changes. It is defined as:

$$J(\mathbf{z}, \mathbf{u}) = \sum_{k=t}^{t+N} \|\mathbf{z}(k|t) - \mathbf{z}_{\text{ref}}(k|t)\|^2_{\mathbf{Q}_z} + \sum_{k=t}^{t+N-1} \left(\|\mathbf{u}(k|t)\|^2_{\mathbf{Q}_u} + \|\Delta\mathbf{u}(k|t)\|^2_{\mathbf{Q}_{\Delta u}} \right), \tag{2.13}$$

where:

- $\mathbf{z}(k|t)$ is the predicted state at step k given current time t,
- $\mathbf{z}_{\text{ref}}(k|t)$ is the reference trajectory state at step k,
- $\mathbf{u}(k|t)$ is the control input at step k,
- $\Delta\mathbf{u}(k|t) = \mathbf{u}(k|t) - \mathbf{u}(k-1|t)$ represents the control input rate,
- $\mathbf{Q}_z$, $\mathbf{Q}_u$, and $\mathbf{Q}_{\Delta u}$ are symmetric positive semi-definite weighting matrices that encode the relative importance of state tracking, control effort, and control smoothness, respectively, in the cost function.

This quadratic cost formulation ensures smooth and accurate trajectory tracking while avoiding aggressive control actions that might result in instability or actuator saturation.

The following section presents the distributed baseline algorithm, which sequentially solves the NMPC and CA problems in a structured and distributed manner.

2.2.6. Original Algorithm

Algorithm 1 Distributed Coordination Algorithm

Description: This algorithm alternates between two optimisation stages: the NMPC optimisation (2.11) and the Collision Avoidance optimisation (2.12).
Initialize: $[s_{ij}(1), \ldots, s_{ij}(N)]$, $[\lambda_{ij}(1), \ldots, \lambda_{ij}(N)]$, $[\lambda_{ji}(1), \ldots, \lambda_{ji}(N)]$, for all $i, j \in \mathcal{N},\ i \neq j$.
for $t = 0, 1, 2, \ldots$ **do**
  for all vehicles $i \in \mathcal{V}$ **(in parallel) do**
    Solve NMPC problem (2.11).
    Compute the shifted state and interpolate the terminal state:

$$[\mathbf{z}^i(1), \ldots, \mathbf{z}^i(N), \bar{\mathbf{z}}_T^i]$$

    Compute associated polytopic sets:

$$\left[\mathbf{A}^i(1), \ldots, \mathbf{A}^i(N), \mathbf{A}_T^i\right], \quad \left[\mathbf{b}^i(1), \ldots, \mathbf{b}^i(N), \mathbf{b}_T^i\right]$$

    Communicate polytopic sets to all neighbouring vehicles $j \in \mathcal{N}_i$.
    for all $j \in \mathcal{N}_i$ **do**
      Solve Collision Avoidance problem (2.12).
    end for
    Apply first control input $\mathbf{u}_{\text{MPC}}^i$.
  end for
end for

2.3. ACADOS

ACADOS is a free and open-source software package designed for fast embedded optimisation, offering high-performance solvers with a high-level Python interface [14]. Its open-source nature makes it accessible to researchers and developers, fostering transparency and collaboration. ACADOS is especially convenient for distributed multi-agent experimentation, as it avoids the licensing constraints commonly associated with commercial optimisation solvers. Its solver architecture supports real-time performance and is particularly well-suited for model predictive control (MPC) and nonlinear programming (NLP) in robotics and automotive applications.

2.4. Delft's Autonomous-driving Robotic Testbed (DART)

The algorithms proposed in this research enable complex multi-vehicle motion planning through a distributed approach. While promising, these methods require substantial further development and validation before they can be reliably deployed in real-world traffic scenarios. The Delft Autonomous-driving Robotic Testbed (DART) [10] provides a cost-effective and easily accessible platform to support this development. DART is a small, car-like robotic platform, approximately 1/10th the size of a standard vehicle, equipped with a comprehensive suite of sensors suited for research in cooperative and autonomous driving.

The kinematic and geometric model of the DART vehicle forms the foundation for the distributed Model Predictive Control (MPC) formulation used in this work. To streamline the development process, a high-fidelity DART simulator is employed to evaluate the real-time performance of the distributed controller in a controlled and reproducible environment. The simulator replicates all relevant sensor outputs and enables rapid prototyping, significantly reducing the time between algorithm design and validation.

The control architecture operates within a closed-loop system using the Robot Operating System (ROS). Controller nodes compute throttle and steering commands and transmit them to the simulator node. The simulator then updates the vehicle state based on these inputs and returns the updated state to the controller nodes. This interaction loop, illustrated in figure 2.3, facilitates thorough testing of distributed decision-making and control strategies under realistic communication and feedback conditions.

While testing on a 1/10 scale vehicle provides valuable insights, it also introduces limitations. Physical

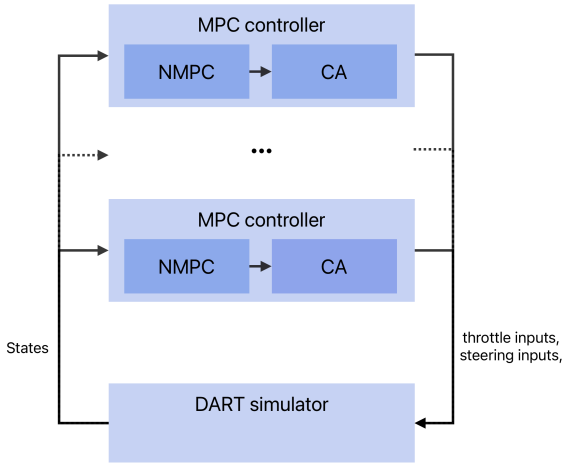


Figure 2.3: The closed-loop interaction between the controllers and the DART simulator

dynamics such as friction, inertia, and actuation delays scale nonlinearly and may not fully reflect the behaviour of full-size vehicles. Additionally, real-world challenges, such as GPS inaccuracies, road surface variability, and large-scale communication issues, are not entirely captured in the DART setup. These are important considerations for future work, where transitioning to full-scale autonomous platforms will be essential for validating practical deployment. However, for now, the DART platform offers an efficient and accurate tool for early-stage testing, allowing researchers to develop, debug, and refine distributed planning and control strategies before moving to more complex, real-world environments.

3

Proposed Approach

This chapter begins by presenting a set of design modifications and enhancements to the general optimisation problems defined in Chapter 2. Specifically, Equations (2.10), (2.11), and (2.12). These modifications are foundational to the methods evaluated experimentally in subsequent chapters. The chapter proceeds by detailing the baseline algorithms, which incorporate these design choices, forming a reference point for evaluating the improvements brought by the proposed methods. Finally, the proposed algorithms are introduced. For each, we provide the motivation, design rationale, and algorithmic modifications, with a focus on illustrating the concrete advantages they offer over the baseline.

To enhance the performance and practicality of the distributed coordination scheme, several key adjustments have been proposed in this work. These modifications aim to address computational challenges, improve generalizability, and facilitate future deployment on physical testbeds.

First, the cost function employed in the optimisation was reformulated based on contouring control objectives, as described in Section 2.1. In contrast to [3], which relies on a carefully designed reference trajectory that implicitly encodes merging timing, the proposed approach leaves the timing of interactions to be determined by the algorithm itself. This provides increased flexibility of the solution and avoids the need for hand-crafted references, but also introduces additional complexity for the optimizer. By leveraging contouring control, agents can follow spatial references more generically, enabling broader applicability in scenarios where exact reference timing is unknown or undesired.

Second, the system dynamics used in the NMPC formulation are replaced with the DART vehicle model introduced in Section 2.4. The DART platform is specifically designed for autonomous driving experiments and includes all necessary sensors for perception and control. By using this model, the proposed controller can be tested in simulation with high fidelity and readily transferred to physical robots in the lab, providing a practical pathway for real-world validation. For an illustration of the setup see figure 2.3.

Another important change concerns the dual variable $\mathbf{s}$, which appears in the collision avoidance (CA) formulation. The original problem includes a constraint $\|\mathbf{s}\|_2 \leq 1$, which represents a ball constraint in two dimensions. In figure 3.1 an illustration is shown which visualises the 2D ball constraint, where the axes correspond to the components of the $\mathbf{s_{ij}}$ vector. It was observed, however, that this constraint posed difficulties for the numerical solver, often resulting in infeasibilities or poor convergence. To alleviate this, the unit ball constraint was approximated using an octagon inscribed within the ball (see figure 3.1). This polytopic approximation preserves the core geometric property of the constraint while improving solver compatibility. It is noted that this relaxation may be solver-dependent and could require tuning of the number of polygon sides for best performance.

Additionally, the original algorithm does not minimize the dual variables in the CA optimisation. In this work, the objective function of the CA solver was extended to include a regularization term that penalizes the norm of the dual variables. This modification encourages tighter and more efficient interactions between agents and was empirically shown to improve the spatial coordination, resulting in smaller

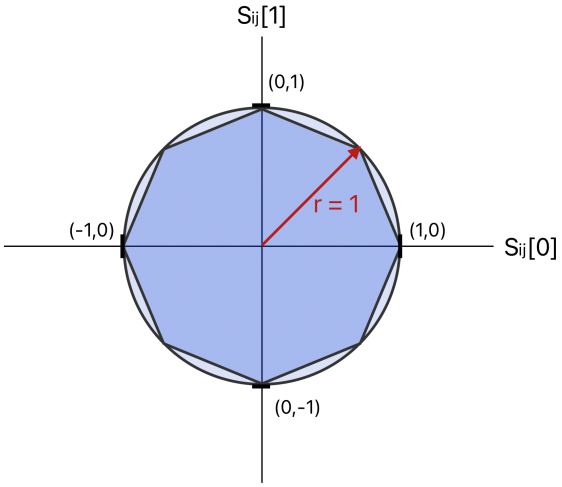


Figure 3.1: Visualisation of the adjusted s_{ij}-norm constraint. The axes represent the two entries of the s_{ij} dual variable

and more consistent inter-agent distances during operation.

Finally, the iteration structure of the algorithm was revisited. The distributed baseline version performs one NMPC optimisation followed by one CA optimisation at each timestep. In contrast, the proposed algorithm allows for multiple alternations between NMPC and CA at each timestep, giving the agents more opportunities to refine their plans in response to updated collision avoidance solutions. It was observed that this additional iterative refinement improves overall convergence and robustness.

Together, these enhancements aim to make the distributed coordination algorithm more general, practical, and reliable. The full details of the modified algorithmic implementation are provided in the following section.

3.1. Baseline Proposed Design

Based on the design choices outlined above, the proposed baseline is presented in the following centralised and distributed formulations.

3.1.1. Centralised Model Predictive Contouring Control

In this section, based on the centralised problem in 2.2.3, we formulate a centralised Model Predictive Contouring Control (MPCC) problem for coordinating multiple autonomous vehicles. The contouring control approach is advantageous in scenarios where the trajectory is defined in a spatial frame rather than over time, allowing each agent to follow a reference path without a predefined timing, which is particularly beneficial for cooperative manoeuvres such as merging or intersection negotiation.

The objective function in the centralised MPCC formulation includes several terms. The tracking term ensures the vehicle velocity remains close to a reference speed v_{ref}. The contouring error ε^c and lag error ε^{lag} terms penalize deviations from the spatial reference path, ensuring the agent stays near the intended trajectory. The control effort is minimized via a weighted quadratic penalty on the control input $\mathbf{u}$. In a similar manner, the decision variables λ_{ij}, λ_{ji}, and s_{ij} are also included as optimisation variables and minimized accordingly.

This results in the following centralised optimal control problem:

$$\min_{\substack{\mathbf{u}^i(\cdot|t),\,\lambda_{\mathbf{ij}}(\cdot|t),\\ \lambda_{\mathbf{ji}}(\cdot|t),\,\mathbf{s_{ij}}(\cdot|t)}} \quad \sum_{i=1}^{M} \Big[\|\mathbf{v}(k|t) - v_{ref}\|^2_{\mathbf{Q}_{v_{ref}}} + \|\varepsilon^{\mathbf{c}}(k|t)\|^2_{\mathbf{Q}_{\varepsilon^c}} + \left\|\varepsilon^{\mathsf{lag}}(k|t)\right\|^2_{\mathbf{Q}_{\varepsilon^{\mathsf{lag}}}}$$

$$+ \|\mathbf{u}(k|t)\|^2_{\mathbf{Q}_u} + \|\lambda_{\mathbf{ij}}(k|t)\|^2_{\mathbf{Q}_{\lambda_{ij}}} + \|\lambda_{\mathbf{ji}}(k|t)\|^2_{\mathbf{Q}_{\lambda_{ji}}} + \|\mathbf{s_{ij}}(k|t)\|^2_{\mathbf{Q}_{s_{ij}}} \Big] \tag{3.1a}$$

subject to

$$\mathbf{z}^i(k+1|t) = f(\mathbf{z}^i(k|t), \mathbf{u}^i(k|t)), \tag{3.1b}$$

$$\mathbf{z}^i(0|t) = \mathbf{z}^i(t), \tag{3.1c}$$

$$\mathbf{z}^i(k|t) \in \mathcal{Z}, \quad \mathbf{u}^i(k|t) \in \mathcal{U}, \tag{3.1d}$$

$$-\mathbf{b}^i(z^i(k|t))^\top \lambda_{ij}(k|t) - \mathbf{b}^j(z^j(k|t))^\top \lambda_{ji}(k|t) \geq d_{\min}, \tag{3.1e}$$

$$\mathbf{A}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) + s_{ij}(k|t) = 0, \tag{3.1f}$$

$$\mathbf{A}^j(\mathbf{z}^j(k|t))^\top \lambda_{ji}(k|t) - s_{ij}(k|t) = 0, \tag{3.1g}$$

$$\lambda_{ij}(k|t),\, \lambda_{ji}(k|t) \geq 0, \quad \|s_{ij}(k|t)\|_2 \leq 1, \tag{3.1h}$$

$\forall i \in \mathcal{V},\ j \in \mathcal{N}_i,$ and $k \in \{1, 2, \ldots, N\}$.

3.1.2. Distributed Model Predictive Contouring Control (DMPC)

To enable scalability and reduce computational burden, the centralised MPCC problem (3.1) can be reformulated in a distributed fashion, enabling each vehicle to optimize its trajectory independently while still accounting for nearby vehicles through information exchange.

As with the distributed scheme in Section 2.2.4, the MPCC formulation is split into two subproblems: the local NMPC problem and the collision avoidance (CA) problem. The local NMPC step focuses on optimizing trajectory tracking and control effort, based on the latest predictions of other agents. The CA step, in turn, updates the shared dual variables required to enforce inter-agent separation constraints. In the remainder of this thesis, we will refer to this approach as **Distributed Baseline** or **DMPC**.

The alternating scheme between NMPC and CA is coordinated through the algorithm described in Algorithm 1. The algorithm allows agents to refine their decisions iteratively, sharing their trajectories as polytopic sets to support local optimisation.

The distributed NMPC subproblem for each agent i is defined as follows:

$$\min_{\mathbf{u}^i(\cdot|t)} \quad \sum_{i=1}^{M} \Big[\|\mathbf{v}(k|t) - v_{ref}\|^2_{\mathbf{Q}_{v_{ref}}} + \|\varepsilon^{\mathbf{c}}(k|t)\|^2_{\mathbf{Q}_{\varepsilon^c}} + \left\|\varepsilon^{\mathsf{lag}}(k|t)\right\|^2_{\mathbf{Q}_{\varepsilon^{\mathsf{lag}}}} + \|\mathbf{u}(k|t)\|^2_{\mathbf{Q}_u} \Big] \tag{3.2a}$$

subject to $(3.1b), (3.1c), (3.1d),$

$$-\mathbf{b}^i(\mathbf{z}^i(k|t))^\top \overline{\lambda}_{ij}(k|t) - \mathbf{b}^j(\mathbf{z}^j(k|t))^\top \overline{\lambda}_{ji}(k|t) \geq d_{\min}, \tag{3.2b}$$

$$\mathbf{A}^i(\mathbf{z}^i(k|t))^\top \overline{\lambda}_{ij}(k|t) + \overline{s}_{ij}(k|t) = 0, \tag{3.2c}$$

for all $k \in \{1, 2, \ldots, N\}$.

The dual variables $\overline{\lambda}_{ij}, \overline{\lambda}_{ji}$ and $\overline{s}_{ij}$ are fixed during this NMPC phase and updated in the subsequent CA phase equal to optimisation problem 2.12. This modular decomposition enables each agent to compute its optimal input trajectory using only local information and shared geometric constraints from its neighbours.

3.2. Proposed Alternative Algorithms

Building on the distributed baseline formulation, we now introduce two alternative algorithms that relax coordination constraints to improve flexibility and convergence.

The two algorithms proposed in this chapter are used for an in-depth analysis of their performance, found in the next chapter: Results chapter 4.

In addition to these two main alternatives, also other variants were explored, discussed in the appendix in Sections A.2.1 and A.2.2.

3.2.1. Distributed Model Predictive Contouring Control with Relaxed CA (DMPC-RCA)

In the DMPC formulation, the dual variable λ_{ij} used in the NMPC phase are fixed based on the outputs of the collision avoidance (CA) solver. While this enforces consistent inter-agent constraints, it also significantly restricts the flexibility of the NMPC solver, resulting in conservative trajectory plans and smaller feasible regions.

This section proposes a relaxed formulation that allows greater flexibility in planning by treating the dual variables λ_{ij} as decision variables in the NMPC optimisation step. In the remainder of this thesis, we will refer to this proposed method as **DMPC-RCA**. The key insight is that the same constraints governing λ_{ij} in the centralised problem can be directly applied in the distributed NMPC step, without compromising the underlying dual structure of the formulation.

By making λ_{ij} a decision variable in the NMPC phase, the optimizer gains the ability to adapt the supporting hyperplanes defining collision avoidance constraints dynamically within the prediction horizon. This leads to less conservative plans and allows agents to explore a larger region of feasible trajectories.

Proposed DMPC-RCA Algorithm

The overall structure of the algorithm remains unchanged from Algorithm 1. The modification lies in the NMPC solver, where the dual variable λ_{ij} is now optimized alongside the control inputs $\mathbf{u}^i$. This modified NMPC problem is defined below:

$$\begin{aligned}
\min_{\mathbf{u}^i(\cdot|t),\, \lambda_{\mathbf{ij}}(\cdot|t)} \quad & \sum_{i=1}^{M} \Big[\|\mathbf{v}(k|t) - v_{ref}\|^2_{\mathbf{Q}_{v_{ref}}} + \|\varepsilon^{\mathbf{c}}(k|t)\|^2_{\mathbf{Q}_{\varepsilon^c}} + \left\|\varepsilon^{\mathsf{lag}}(k|t)\right\|^2_{\mathbf{Q}_{\varepsilon^{\mathsf{lag}}}} + \|\mathbf{u}(k|t)\|^2_{\mathbf{Q}_u} \\
& \quad + \|\lambda_{\mathbf{ij}}(k|t)\|^2_{\mathbf{Q}_{\lambda_{ij}}} \Big] && \text{(3.3a)} \\
\text{subject to} \quad & (3.1b), (3.1c), (3.1d), \\
& - \mathbf{b}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) - \mathbf{b}^j(\mathbf{z}^j(k|t))^\top \overline{\lambda}_{ji}(k|t) \geq d_{\min}, && \text{(3.3b)} \\
& \mathbf{A}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) + \overline{s}_{ij}(k|t) = 0, && \text{(3.3c)} \\
& \text{for all } k \in \{1, 2, \ldots, N\}.
\end{aligned}$$

Here, the dual variable λ_{ij} is treated as a local decision variable by agent i, while $\overline{\lambda}_{ji}$ remains fixed from the CA solver. This introduces asymmetry in the constraint handling, but it is progressively resolved as both agents update their respective duals in an alternating fashion in multiple iterations.

The flowchart in figure 3.2 illustrates the exchange of decision variables between the algorithm stages and consecutive time steps. Dotted indicate the exchange of predicted state trajectories between agents, while solid lines denote the exchange of dual variables computed in the CA optimisation. Between the NMPC and CA solvers, the trajectories are shifted forward by one time step to align with the prediction horizon of the next time instant.

3.2.2. Distributed Model Predictive Contouring Control with Relaxed CA and Consensus (DMPC-RCA-C)

Building on the DMPC-RCA approach presented earlier, this section proposes an enhancement to improve coordination and convergence between agents. In the remainder of this thesis, we will refer to this proposed method as **Consensus Algorithm** and **DMPC-RCA-C**.

In the previously introduced DMPC-RCA approach, the dual variables λ_{ij} are independently computed in both the NMPC and CA solvers. While this design increases flexibility in trajectory planning, it may also lead to discrepancies between agents' planned paths. As a result, the CA solver can struggle

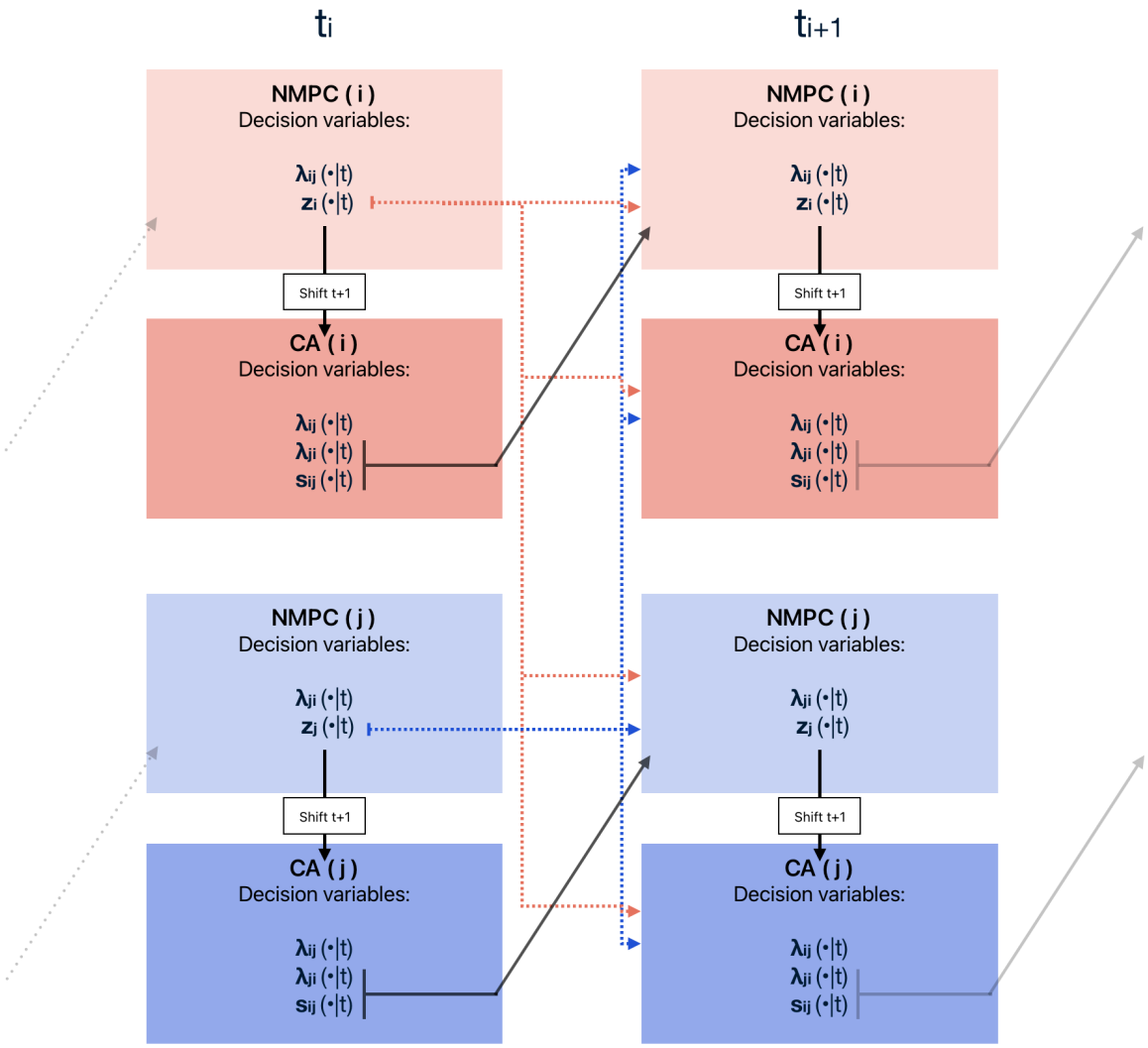


Figure 3.2: Flowchart for the shared variables for DMPC-RCA Algorithm. The orange colour scheme belongs to vehicle i, and the blue colour scheme belongs to vehicle j. The black and gray arrows represent the exchange of the dual variables (λ_{ji}, s_{ij}) between the CA and NMPC modules. The dashed coloured arrows represent the exchanged computed trajectories.

to find feasible solutions, particularly when trajectories diverge significantly due to overly independent NMPC decisions.

To address this, we propose the DMPC-RCA-C algorithm, which introduces a consensus-regularization term in the CA cost function. This encourages agreement between the dual variables computed by neighbouring agents, without explicitly enforcing equality constraints. The regularization improves consistency across agents and supports convergence while preserving the flexibility of DMPC-RCA. This consensus term penalizes the deviation between λ_{ji}^{i}, the local CA decision variable, and $\overline{\lambda}_{ji}^{j}$, the corresponding value received from the neighbouring agent j, as shown in Equation (3.4).

$$\rho \cdot \left\| \lambda_{ji}^{i}(k|t) - \overline{\lambda}_{ji}^{j}(k|t) \right\| \quad \text{for } k = 0, \ldots, N \tag{3.4}$$

Here, ρ is a tunable penalty parameter that determines the strength of the consensus enforcement. The inclusion of this term introduces asymmetry into the CA optimisation, as agents i and j no longer solve identical problems and each uses a locally received estimate from its neighbour. However, this asymmetry is expected to diminish over successive iterations, as the consensus term promotes alignment between agents' solutions.

By including this consensus term, the CA solver is guided to maintain alignment with the dual variables computed in the NMPC step of neighbouring agents. This mechanism effectively serves as a form of warm-starting and promotes convergence by reducing inter-agent discrepancy in the collision avoidance constraints.

Proposed DMPC-RCA-C Algorithm

The structure of this algorithm follows the general framework of Algorithm 1 and the NMPC problem definition is equal to the definition in problem 3.3. The modifications for this approach are introduced in the CA solver. The adjusted CA problem, incorporating the consensus term, is defined in Equation (3.5).

$$\begin{aligned}
\max_{\lambda_{ij}(\cdot|t),\, \lambda_{ji}(\cdot|t),\, s_{ij}(\cdot|t)} \quad & -\overline{\mathbf{b}}^{i}(\overline{\mathbf{z}}^{i}(k|t))^{\top}\lambda_{ij}(k|t) - \overline{\mathbf{b}}^{j}(\overline{\mathbf{z}}^{j}(k|t))^{\top}\lambda_{ji}(k|t) - \rho \cdot \left\| \lambda_{ji} - \overline{\lambda_{ji}^{j}} \right\| && \text{(3.5a)} \\
\text{subject to} \quad & \overline{\mathbf{A}}^{i}(\overline{\mathbf{z}}^{i}(k|t))^{\top}\lambda_{ij}(k|t) + s_{ij}(k|t) = 0, && \text{(3.5b)} \\
& \overline{\mathbf{A}}^{j}(\overline{\mathbf{z}}^{j}(k|t))^{\top}\lambda_{ji}(k|t) - s_{ij}(k|t) = 0, && \text{(3.5c)} \\
& -\overline{\mathbf{b}}^{i}(\overline{\mathbf{z}}^{i}(k|t))^{\top}\lambda_{ij}(k|t) - \overline{\mathbf{b}}^{j}(\overline{\mathbf{z}}^{j}(k|t))^{\top}\lambda_{ji}(k|t) \geq d_{\min}, && \text{(3.5d)} \\
& \|s_{ij}(k|t)\|_2 \leq 1, && \text{(3.5e)} \\
& \lambda_{ij}(k|t) \geq 0, \quad \lambda_{ji}(k|t) \geq 0, && \text{(3.5f)} \\
& \forall k \in \{1, 2, \ldots, N\}.
\end{aligned}$$

The flowchart in figure 3.3 illustrates the exchange of decision variables between algorithm stages across consecutive time steps, following a similar structure to figure 3.2. In contrast to the earlier version, this diagram incorporates the *consensus mechanism* in the CA solver. The black lines represent the additional exchange of dual variables introduced by the consensus term, which ensures agreement on shared variables between agents.

The resulting algorithm is presented in Algorithm 2.

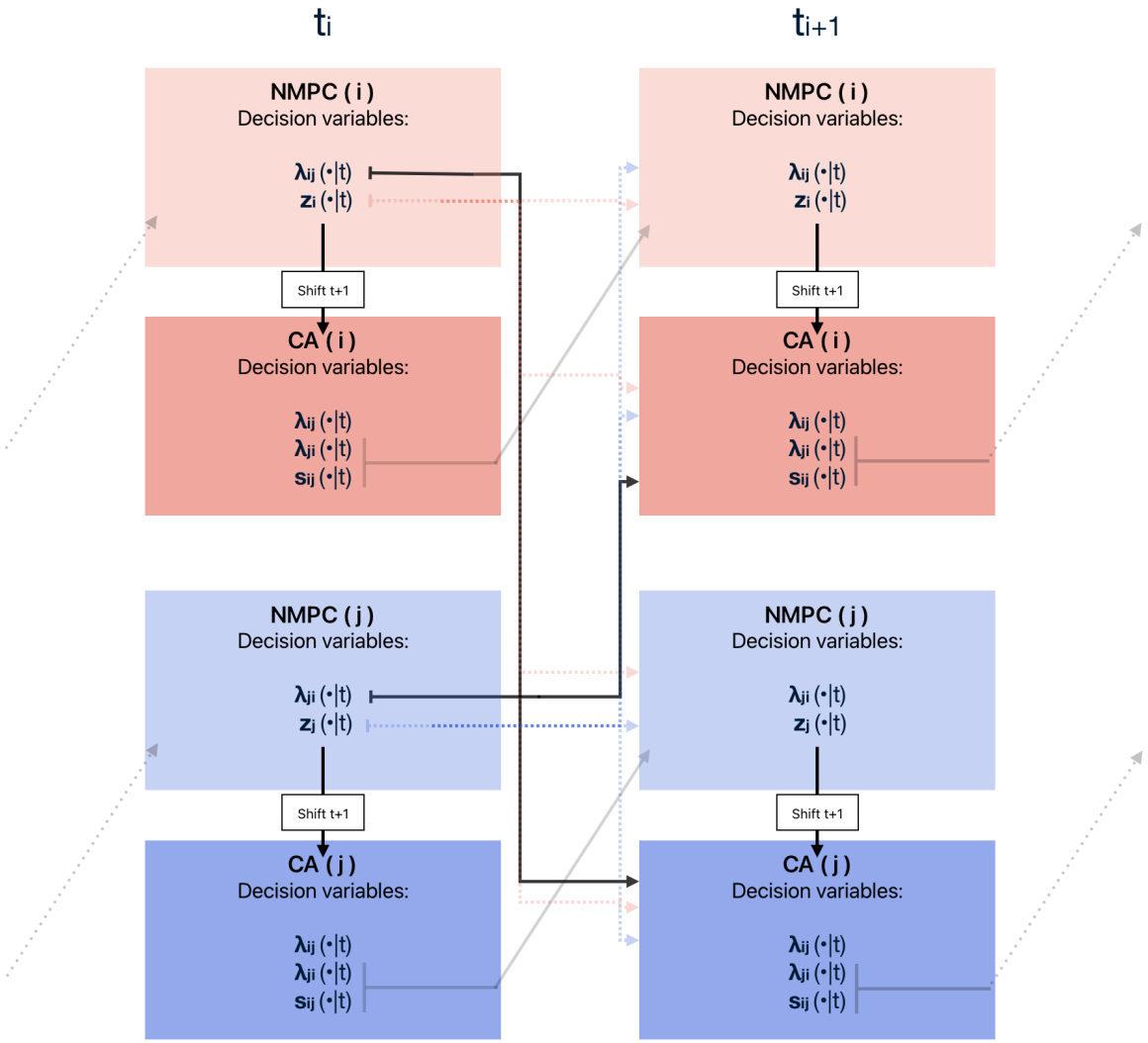


Figure 3.3: Flowchart of the decision variables for the DMPC-RCA-C Algorithm. The orange colour scheme belongs to vehicle i, and the blue colour scheme belongs to vehicle j. The connection arrows explained in figure 3.2 are made transparent for readability. The black arrows represent the exchange of the, in the NMPC module computed, dual variable $\lambda_{ij}/\lambda_{ji}$ for the consensus term in the CA objective.

Algorithm 2 Distributed Coordination Algorithm

Description: This algorithm alternates between two optimisation stages: the NMPC optimisation (3.3) and the Collision Avoidance optimisation (3.5).
Initialize: $[s_{ij}(1), \ldots, s_{ij}(N)]$, $[\lambda_{ij}(1), \ldots, \lambda_{ij}(N)]$, $[\lambda_{ji}(1), \ldots, \lambda_{ji}(N)]$, for all $i, j \in \mathcal{N},\ i \neq j$.
for $t = 0, 1, 2, \ldots$ **do**
    for all vehicles $i \in \mathcal{V}$ **(in parallel) do**
        Solve NMPC problem (3.3).
        Compute the shifted state and interpolate the terminal state:

$$[\mathbf{z}^i(1), \ldots, \mathbf{z}^i(N), \bar{\mathbf{z}}_T^i]$$

        Compute associated polytopic sets:

$$\left[\mathbf{A}^i(1), \ldots, \mathbf{A}^i(N), \mathbf{A}_T^i\right], \quad \left[\mathbf{b}^i(1), \ldots, \mathbf{b}^i(N), \mathbf{b}_T^i\right]$$

        Communicate polytopic sets and dual variables λ_{ij} to all neighbouring vehicles $j \in \mathcal{N}_i$.
        for all $j \in \mathcal{N}_i$ **do**
            Solve Collision Avoidance problem (3.5).
        end for
        Apply first control input $\mathbf{u}_{\text{MPC}}^i$.
    end for
end for

4

Results

4.1. Experimental Design

To evaluate the performance of the proposed algorithms in comparison to the centralised method and the distributed baseline, a set of simulation experiments was conducted. All four variants: centralised (3.1), distributed baseline (section 3.1.2), distributed DMPC-RCA (section 3.2.1), and distributed DMPC-RCA-C (section 3.2.2) were tested across two representative traffic scenarios: a merging scenario and a T-junction scenario.

Merging Scenario. In this scenario, both vehicles begin in separate lanes and are required to merge into a single shared lane, see figure 4.1. The primary focus of analysis is the resulting merging order and the interplay between collision avoidance behaviour and reference trajectory tracking.

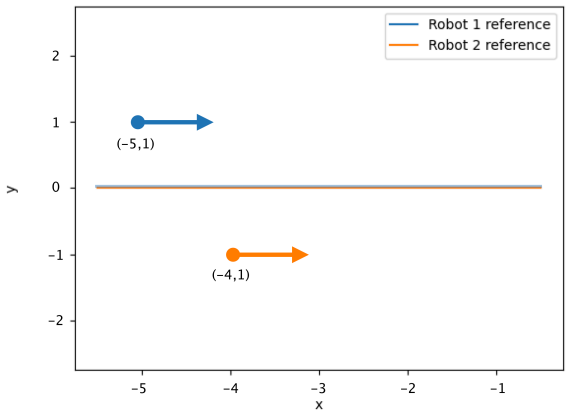


Figure 4.1: Reference path for the merging scenario. The vehicles' starting positions are denoted by the arrows and chosen to simulate their placement in adjacent lanes.

T-junction Scenario. Here, the two vehicles approach a T-junction from different directions and must coordinate a safe crossing, see figure 4.2. Although scheduling-based strategies may be more naturally suited to this problem, the scenario is employed in this work to evaluate how well the algorithms balance reference tracking with collision avoidance under more complex spatial interactions.

4.1.1. Slack Handling

To ensure robustness and feasibility of the optimisation problems, all algorithms incorporate constraint relaxation through slack variables. These slacks are penalized both linearly and quadratically in the objective function to discourage constraint violation while allowing flexibility when needed.

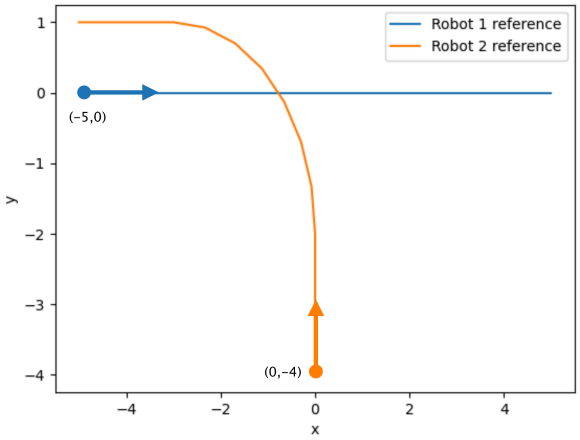


Figure 4.2: Reference path for the T-junction scenario. The vehicles starting positions are denoted by the arrows

The penalties are formulated as follows:

$$\text{Quadratic penalty:} \quad \frac{1}{2} s(t)^\top Z\, s(t), \tag{4.1}$$

$$\text{Linear penalty:} \quad z^\top s(t), \tag{4.2}$$

where $s(t)$ denotes the slack variable, Z is a diagonal matrix of quadratic penalties, and z contains the linear penalty weights. These penalties are chosen to be 2 to 10 times larger than the other weights in the objective function, depending on the experiment.

4.1.2. Evaluation Metrics

The algorithms are evaluated on the following criteria:

- **Accumulated Tracking Cost Compared to Centralised.** To evaluate and compare the performance of all four approaches, the accumulated cost is computed using the centralised objective function without slack penalty cost. This ensures a consistent and fair comparison across methods. The objective function, detailed in equation (4.3), includes contouring error, reference velocity tracking, and the minimization of control inputs and dual variables. The total cost is the sum of the total prediction cost for every control update (denoted in the equation by variable K).
- **Computation Time.** One of the core motivations of this work is to identify distributed methods that match the centralised solution in performance but scale better in terms of computational cost. Computation time is measured as the average time per prediction step. For the centralised method, this corresponds to the time required to solve problem (3.1) at each control step. For the distributed variants, it refers to the time taken by Algorithm 1 to complete two full iterations. While the number of iterations can be adjusted, it was observed in these test cases that two iterations were sufficient.
- **Inter-Vehicular Distance.** The minimum distance between vehicles serves as a proxy for solution optimality: smaller distances indicate more efficient use of the shared space. This metric computes the minimum distance between the polytopes representing each vehicle, accounting for orientation and shape.

$$J = \sum_{0}^{K} \sum_{k=1}^{N} \mathbf{z}_k^\top Q \mathbf{z}_k \tag{4.3}$$

$$\mathbf{z}_k = \begin{bmatrix} \epsilon_k^{\text{contour}} \\ \epsilon_k^{\text{lag}} \\ v_k - v_{\text{ref}} \\ u_k^{\text{throttle}} \\ u_k^{\text{steering}} \\ \boldsymbol{\lambda}_k \\ \mathbf{s}_k \end{bmatrix}$$

$$Q = \text{diag}\left(w_{\text{contour}}, w_{\text{lag}}, w_{\text{velocity}}, w_{\text{throttle}}, w_{\text{steering}}, \underbrace{w_\lambda, \ldots, w_\lambda}_{n_\lambda}, \underbrace{w_s, \ldots, w_s}_{n_s} \right)$$

4.1.3. Scenario: Merging

	Accumulated Cost Centralised Objective (without slack cost)	Computation Time Mean (ms)	Minimum Inter-Vehicular Distance
Centralised MPCC	24.08	37.9	7.40e-1
DMPC Algorithm (Distributed Baseline)	50.44	0.5	5.80e-1
DMPC-RCA Algorithm	24.11	0.4	4.50e-1
DMPC-RCA-C Algorithm	24.03	0.5	4.50e-1

Table 4.1: The table shows the performance comparison on the four algorithms evaluated in this research on the merging scenario. Metrics include accumulated tracking cost, computation time mean and distance between the vehicles.

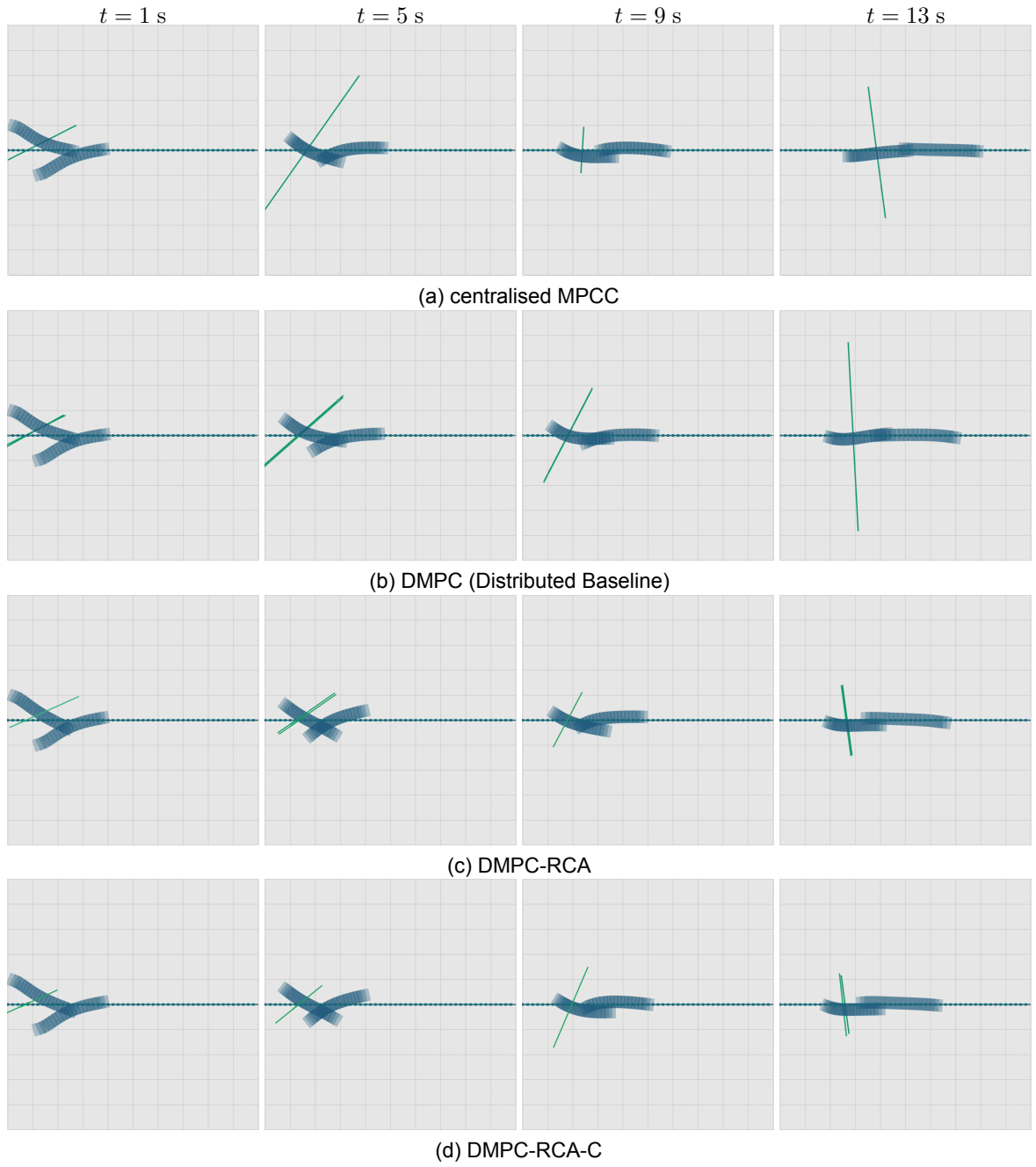
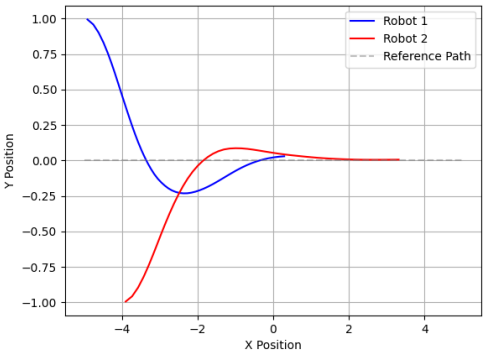
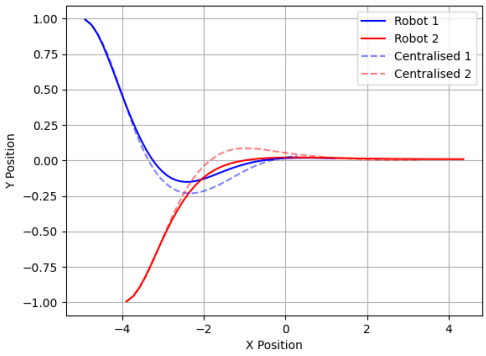


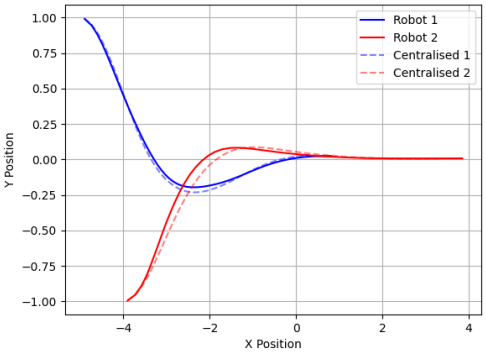
Figure 4.3: Snapshots at selected time steps illustrating the behaviour of the four algorithms in the merging scenario. The images highlight differences in agent interaction across the algorithms. Predicted trajectories are shown as sequences of blue rectangles, which visualises the polytopic sets at each time step. The first rectangle indicates each agent's current position. The green bars represent the separating hyperplanes computed by the agents for collision avoidance. Both agents maintain a local copy of the separating hyperplane, which is plotted and may occasionally overlap.



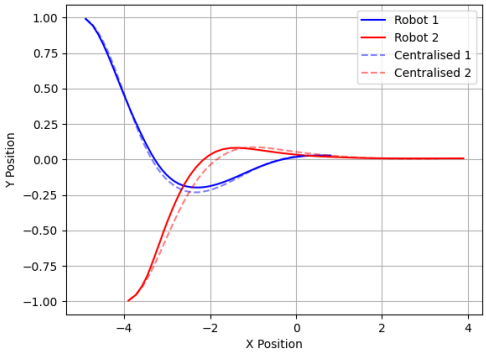
(a) Centralised



(b) Distributed Algorithm DMPC



(c) Distributed Algorithm DMPC-RCA



(d) Distributed Algorithm DMPC-RCA-C

Figure 4.4: Trajectories of the agents (solid lines) compared to the centralised trajectories (dashed lines) for all four algorithms in the merging scenario.

4.1.4. Scenario: T-junction

	Accumulated Cost Centralised Objective (without slack penalties)	Mean Computation Time (ms)	Minimum Inter-Vehicular Distance
Centralised MPCC	16.05	14.5	2.20e-1
DMPC Algorithm (Distributed Baseline)	64.79	0.5	4.40e-4
DMPC-RCA Algorithm	20.94	0.5	9.40e-2
DMPC-RCA-C Algorithm	24.03	0.5	9.40e-2

Table 4.2: The table shows the performance comparison on the four algorithms evaluated in this research on the T-junction scenario. Metrics include accumulated tracking cost, computation time mean and distance between the vehicles.

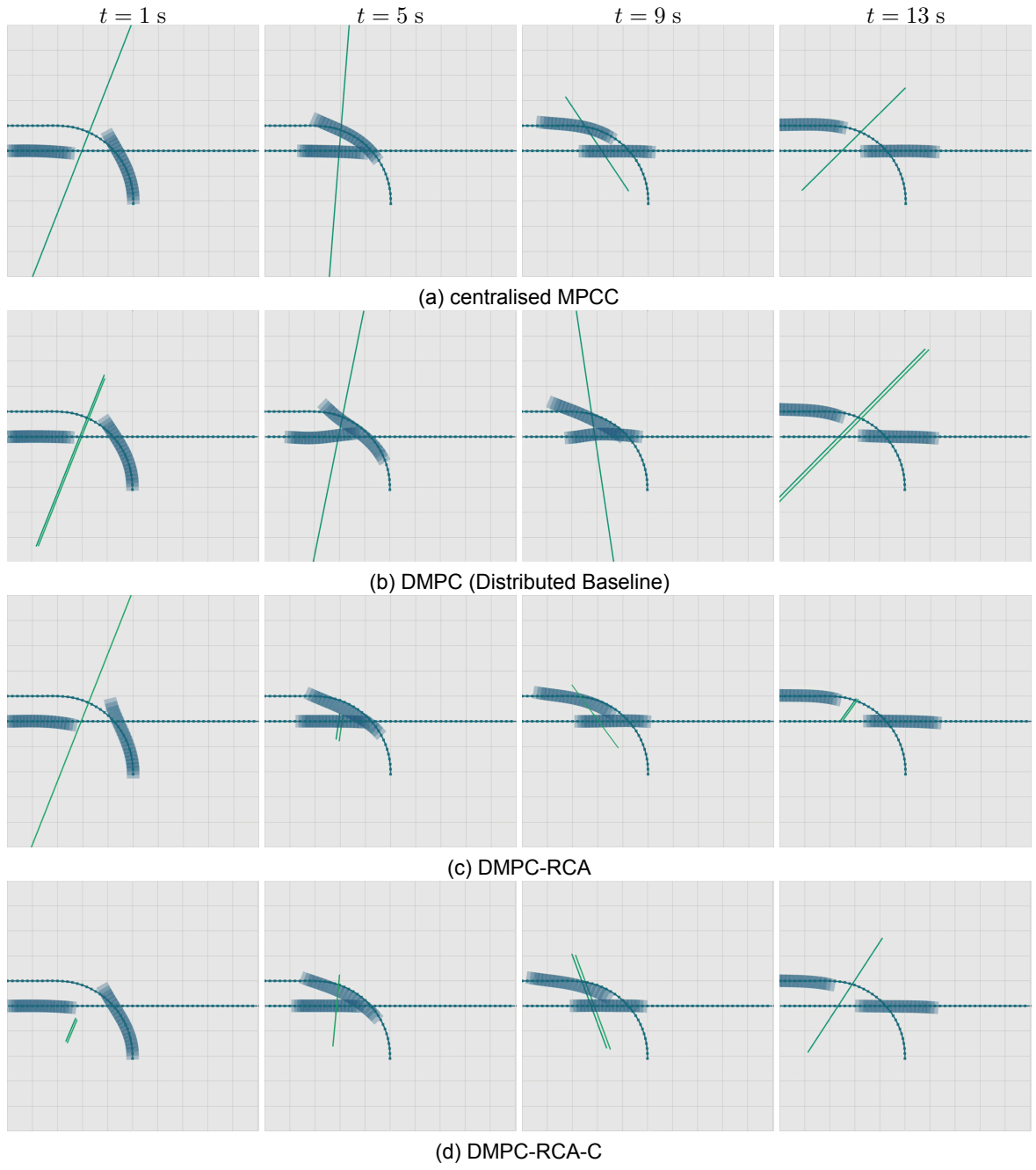
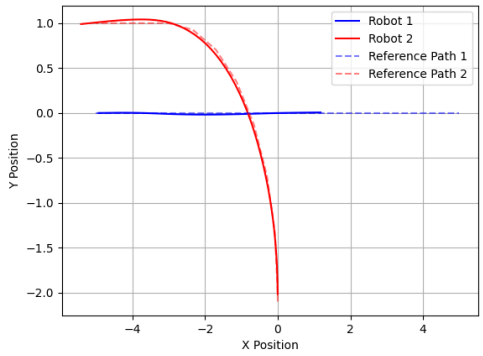
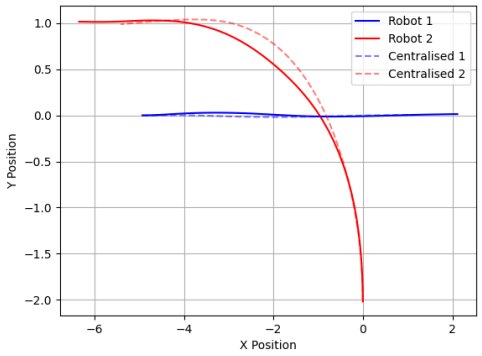


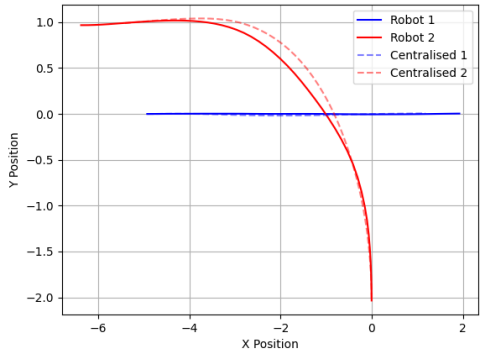
Figure 4.5: Snapshots at selected time steps illustrating the behaviour of the four algorithms in the T-junction scenario. The images highlight differences in agent interaction across the algorithms. Predicted trajectories are shown as sequences of blue rectangles, which visualises the polytopic sets at each time step. The first rectangle indicates each agent's current position. The green bars represent the separating hyperplanes computed by the agents for collision avoidance. Both agents maintain a local copy of the separating hyperplane, which is plotted and may occasionally overlap.



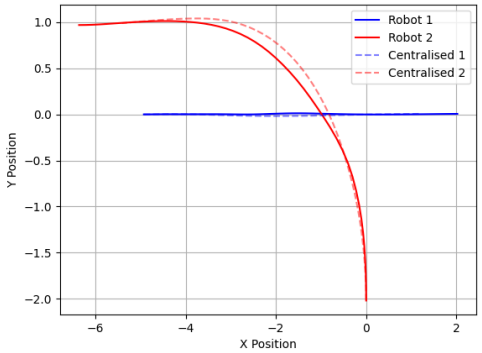
(a) Centralised



(b) Distributed Algorithm DMPC



(c) Distributed Algorithm DMPC-RCA



(d) Distributed Algorithm DMPC-RCA-C

Figure 4.6: Trajectories of the agents (solid lines) compared to the centralised trajectories (dashed lines) for all four algorithms in the T-junction scenario.

Final parameter values for the proposed algorithms can be found in appendix A.1.

5

Discussion

This chapter discusses the observed behaviours and performance characteristics of the proposed distributed NMPC algorithms, particularly in comparison to a centralised approach, for a merging and a T-junction scenario.

5.1. Performance in the Merging Scenario

Accumulated Tracking Cost

From Table 4.1, it can be observed that in terms of deviation from the centralised cost, both the DMPC-RCA algorithm and the DMPC-RCA with consensus perform similarly well to the centralised approach. In contrast, the Distributed Baseline shows the highest recomputed cost, which is approximately twice that of the centralised method. This result highlights the limitations of the Distributed Baseline approach, where the static dual structure restricts the nonlinear model predictive control from effectively following the reference path and target velocity. The improved performance of DMPC-RCA and DMPC-RCA-C illustrates the benefit of introducing additional flexibility in the optimisation problem. By relaxing the dual variables, the feasible solution space is enlarged, allowing trajectories to more closely align with those produced by the centralised solution. When comparing DMPC RCA with its consensus-enhanced version, the results indicate that the inclusion of a consensus mechanism does not significantly improve tracking performance.

Computation Time

The results in Table 4.1 show that the distributed approaches achieve a significant reduction in computation time, approximately two orders of magnitude faster than the centralised approach. This computational advantage is expected to become even more pronounced as the number of vehicles increases, due to the improved scalability of the distributed framework. It could be expected that the consensus algorithm would improve the computation time as it assists the CA solver in converging more efficiently to a solution for the dual variables.

Inter-Vehicular Distance

All algorithms successfully respect the minimum required inter-agent distance of 0.1, as enforced by the collision avoidance constraints and seen in table 4.1. It is noteworthy, however, that the centralised approach results in the largest minimum distance. This could be attributed to the highly nonlinear nature of the problem and the possibility of local minima affecting different approaches differently. Despite using relaxation techniques, all distributed methods ensured collision-free behaviour. Additionally, the results for the distributed baseline approach show a more conservative minimum distance compared to the relaxed versions (DMPC-RCA and DMPC-RCA-C), which is expected. Relaxing the algorithm allows for less conservative, more flexible solutions.

Visual Behaviour

Figure 4.4 presents the trajectories of the two agents under all four control approaches. The dashed lines indicate the trajectories generated by the centralised controller, which serves as the benchmark for performance comparison. Notably, the trajectory of the DMPC algorithm (figure 4.4b) exhibits the

smallest deviation from the reference path among the distributed approaches. Interestingly, despite this visual advantage, the numerical results in table 4.1 show that DMPC yields the highest Accumulated Tracking Cost. This apparent contradiction can be attributed to the algorithm's conservative behaviour: by maintaining a larger inter-agent distance, the DMPC solution provides more flexibility in tracking the reference trajectory accurately, albeit at the expense of overall efficiency.

5.2. Performance in the T-Junction Scenario

Accumulated Tracking Cost

In Table 4.2, the first column shows that both the DMPC-RCA and DMPC-RCA-C algorithms achieve accumulated costs comparable to that of the centralised approach, while the baseline DMPC incurs a tracking cost that is approximately three times higher. Unlike the merging scenario, the DMPC-RCA algorithm performs notably better than its consensus-enhanced counterpart. This difference may be attributed to the sharp 90-degree turn required by the bottom agent in the T-junction scenario. In this context, the dual variables (lambdas), which represent supporting hyperplanes perpendicular to the minimum inter-agent distance, must undergo a rapid 180-degree flip midway through the maneuver. The consensus formulation hinders this transition, as it attempts to minimize deviations from previously computed lambdas. This effectively reduces the algorithm's responsiveness and limits its flexibility. The observed performance gap highlights the importance of balancing convergence stability with adaptability in dynamic multi-agent environments. Furthermore, the clear improvement of DMPC-RCA over the distributed baseline approach underscores the benefits of increased flexibility introduced by its enhanced NMPC structure.

Computation Time

The distributed algorithms demonstrate superior computational performance, achieving solution times that are approximately one to two orders of magnitude faster than the centralised approach. It can also be observed that adding the consensus term to the CA objective does not decrease the overall computation time. One might expect the consensus term to support the solver by guiding the optimisation process toward coordinated trajectories. In this way, it could act as a regularizing component that helps the solver converge more efficiently. However, this supportive effect is not clearly reflected in the measured computation times. Overall, the computational advantage of the distributed approaches compared to the centralised method is substantial and is expected to become even more pronounced as the number of agents increases.

Inter-Vehicular Distance

In the last column of table 4.2, the minimum distance between agents is reported. The results indicate that only the centralised approach maintains a distance greater than the required minimum of 0.1. In contrast, the distributed baseline results in a collision, as the two agents violate the minimum distance constraint. This outcome suggests that the slack penalties in the DMPC formulation overly relax the collision avoidance constraints, preventing convergence to a collision-free solution in this specific scenario. While the DMPC-RCA algorithms avoid actual collisions, they do not strictly adhere to the minimum distance threshold of 0.1, indicating some level of constraint violation as a result of the introduced slack. Nevertheless, these algorithms still converge to a collision-free trajectory for this specific scenario.

Visual Behaviour

From the snapshot of the DMPC algorithm in figure 4.5, it can be observed that in the second frame at $t = 5$ s, the agent trajectories bend upward in contrast to those produced by the other algorithms. This deviation likely contributes to the higher accumulated tracking cost and the near-zero inter-vehicular distance observed. The altered manoeuvrer may stem from the conservative structure of the distributed baseline approach. In particular, the hyperplanes generated by the collision avoidance (CA) solver, represented by the dual variables, are enforced more rigidly within the NMPC solver. This strict treatment could explain the agents' trajectories bending away from the separating hyperplane as visualized in the snapshot.

5.3. Discussion

The results of both scenarios demonstrate that the RCA-based algorithms clearly outperform the distributed baseline approach in terms of tracking accuracy and collision avoidance. By introducing flexibility through the relaxation of dual variables, these distributed algorithms significantly reduce conservativeness in their behaviour while still maintaining safety. This allows for trajectories that more closely resemble the centralised, and therefore optimal, solution without incurring its computational burden. While the consensus-based formulation was hypothesized to enhance coordination and possibly guide the solver toward improved performance, its practical impact was limited. Across all metrics, including accumulated tracking cost, inter-vehicular distance, and computation time, the addition of a consensus term did not lead to substantial improvements. Therefore, although the RCA framework proves highly effective, the consensus enhancement does not yet justify its added complexity in the scenarios evaluated.

5.4. Limitations and Assumptions

While the proposed distributed NMPC framework demonstrates promising results in simulation, several limitations and assumptions should be acknowledged. First, the current study assumes perfect inter-agent communication, with no delays, packet loss, or bandwidth constraints. In real-world deployments, communication imperfections can significantly affect coordination and convergence, potentially leading to degraded performance or safety violations. Second, the simulations neglect sensor noise and model uncertainties, which are inevitable in practical scenarios and may impact both state estimation and control accuracy. Additionally, the vehicle and environment models are idealized, omitting factors such as actuator dynamics, friction variations, and unexpected disturbances. Addressing these limitations in future work, by incorporating realistic communication models, sensor noise, and hardware-in-the-loop experiments will be essential to validate the robustness and applicability of the proposed methods in real-world multi-vehicle systems.

6 Conclusion

The proposed algorithms demonstrate increased robustness and flexibility in multi-agent scenarios, especially in complex situations such as merging or negotiating intersections. Compared to the baseline distributed approach, both the DMPC-RCA and the consensus-based formulation handle a wider range of initial conditions and promote safer and more optimal behaviours. Additionally, they offer a significant computational advantage over the centralised nonlinear model predictive control, making them more suitable for real-time applications in autonomous multi-agent systems. This research highlights the strong potential of the proposed methods in the field of distributed cooperative motion planning and marks a meaningful step toward the realization of fully cooperative autonomous driving on public roads.

6.1. Future Work

While the current implementation demonstrates promising results, several areas remain open for further investigation and enhancement. These can be categorized into short-term and long-term goals as follows:

Short-term goals:

- **Re-initialization strategies**: Introducing mechanisms to recover from standstill situations or deadlocks can immediately enhance robustness, particularly in scenarios where local infeasibility arises.
- **Real hardware validation**: Testing the algorithms on real hardware, such as the DART platform, including the effects of communication delays and random faults, is a critical next step to bridge the gap between simulation and practical deployment.
- **Heterogeneous robot systems**: The current architecture can be extended to heterogeneous robots by employing different polytopic constraint sets. However, tuning becomes more complex due to the need for robot-specific weights and parameters

Long-term goals:

- **Higher-level scheduling**: Integrating a global scheduling layer to improve performance in highly dynamic environments or scenarios with complex task dependencies.
- **Formal convergence analysis**: Although empirical performance is promising, a theoretical proof of convergence—possibly by leveraging existing results from ADMM literature—would significantly strengthen the theoretical foundation of the approach.

References

[1] D. J. Fagnant and K. Kockelman, "Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations," *Transportation Research Part A: Policy and Practice*, vol. 77, pp. 167–181, Jul. 2015, ISSN: 0965-8564. DOI: 10.1016/j.tra.2015.04.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0965856415000804 (visited on 06/29/2024).

[2] L. Ferranti, L. Lyons, R. R. Negenborn, T. Keviczky, and J. Alonso-Mora, "Distributed Nonlinear Trajectory Optimization for Multi-Robot Motion Planning," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 2, pp. 809–824, Mar. 2023, Conference Name: IEEE Transactions on Control Systems Technology, ISSN: 1558-0865. DOI: 10.1109/TCST.2022.3211130. [Online]. Available: https://ieeexplore.ieee.org/document/9920538 (visited on 06/19/2024).

[3] R. Firoozi, L. Ferranti, X. Zhang, S. Nejadnik, and F. Borrelli, "A Distributed Multi-Vehicle Coordination Algorithm for Navigation in Tight Environments," en, *IEEE Transactions on Vehicular Technology*, pp. 1–11, 2024, ISSN: 0018-9545, 1939-9359. DOI: 10.1109/TVT.2024.3409687. [Online]. Available: https://ieeexplore.ieee.org/document/10549815/ (visited on 06/18/2024).

[4] Y. Gao, D. Li, Z. Sui, and Y. Tian, "Trajectory Planning and Tracking Control of Autonomous Vehicles Based on Improved Artificial Potential Field," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 9, pp. 12 468–12 483, Sep. 2024, Conference Name: IEEE Transactions on Vehicular Technology, ISSN: 1939-9359. DOI: 10.1109/TVT.2024.3389054. [Online]. Available: https://ieeexplore.ieee.org/document/10502308 (visited on 11/25/2024).

[5] P. Hart, N. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968, ISSN: 0536-1567. DOI: 10.1109/TSSC.1968.300136. [Online]. Available: http://ieeexplore.ieee.org/document/4082128/ (visited on 11/25/2024).

[6] H. Hewawasam, M. Ibrahim, and G. Appuhamillage, "Past, Present and Future of Path-Planning Algorithms for Mobile Robot Navigation in Dynamic Environments," *IEEE Open Journal of the Industrial Electronics Society*, vol. 3, pp. 353–365, 2022. DOI: 10.1109/OJIES.2022.3179617.

[7] D. Lam, C. Manzie, and M. Good, "Model predictive contouring control," in *49th IEEE Conference on Decision and Control (CDC)*, ISSN: 0191-2216, Dec. 2010, pp. 6137–6142. DOI: 10.1109/CDC.2010.5717042. [Online]. Available: https://ieeexplore.ieee.org/document/5717042 (visited on 04/14/2025).

[8] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 1, Detroit, MI, USA: IEEE, 1999, pp. 473–479, ISBN: 978-0-7803-5180-6. DOI: 10.1109/ROBOT.1999.770022. [Online]. Available: http://ieeexplore.ieee.org/document/770022/ (visited on 11/25/2024).

[9] Lee, J. Zinkula, and N. Naughton, *Your robot has arrived*, en-US, Aug. 2024. [Online]. Available: https://www.businessinsider.com/waymo-self-driving-taxis-chipotle-robots-future-of-service-2024-8 (visited on 10/29/2024).

[10] L. Lyons, T. Niesten, and L. Ferranti, *DART: A Compact Platform For Autonomous Driving Research*, arXiv:2402.07602 [cs], Feb. 2024. DOI: 10.48550/arXiv.2402.07602. [Online]. Available: http://arxiv.org/abs/2402.07602 (visited on 04/22/2025).

[11] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, Jun. 2000, ISSN: 0005-1098. DOI: 10.1016/S0005-1098(99)00214-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109899002149 (visited on 11/13/2024).

[12] M. Reda, A. Onsy, A. Y. Haikal, and A. Ghanbari, “Path planning algorithms in the autonomous driving system: A comprehensive review,” *Robotics and Autonomous Systems*, vol. 174, p. 104 630, Apr. 2024, ISSN: 0921-8890. DOI: 10.1016/j.robot.2024.104630. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889024000137 (visited on 10/29/2024).

[13] E. Rimon and D. Koditschek, “Exact robot navigation using artificial potential functions,” *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, Oct. 1992, Conference Name: IEEE Transactions on Robotics and Automation, ISSN: 2374-958X. DOI: 10.1109/70.163777. [Online]. Available: https://ieeexplore.ieee.org/document/163777/?arnumber=163777 (visited on 11/07/2024).

[14] R. Verschueren, G. Frison, D. Kouzoupis, *et al.*, *Acados: A modular open-source framework for fast embedded optimal control*, arXiv:1910.13753 [math], Nov. 2020. DOI: 10.48550/arXiv.1910.13753. [Online]. Available: http://arxiv.org/abs/1910.13753 (visited on 04/15/2025).

[15] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, “Information-Theoretic Model Predictive Control: Theory and Applications to Autonomous Driving,” en, *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, Dec. 2018, ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2018.2865891. [Online]. Available: https://ieeexplore.ieee.org/document/8558663/ (visited on 11/20/2024).

[16] T. Xia and H. Chen, “A Survey of Autonomous Vehicle Behaviors: Trajectory Planning Algorithms, Sensed Collision Risks, and User Expectations,” en, *Sensors*, vol. 24, no. 15, p. 4808, Jan. 2024, Number: 15 Publisher: Multidisciplinary Digital Publishing Institute, ISSN: 1424-8220. DOI: 10.3390/s24154808. [Online]. Available: https://www.mdpi.com/1424-8220/24/15/4808 (visited on 06/26/2025).

[17] S. Yu, M. Hirche, Y. Huang, H. Chen, and F. Allgöwer, “Model predictive control for autonomous ground vehicles: A review,” *Autonomous Intelligent Systems*, vol. 1, no. 1, 2021. DOI: 10.1007/s43684-021-00005-z.

A

Final Settings and Other Approaches

A.1. Final Code Settings

Listing A.1: Final parameter configuration merging scenario

```
N: 20                          # Prediction horizon length
integrator_step: 0.3           # Integration step size
skip_solver_generation: false   # Skip solver code generation (assume precompiled)
add_slack: true                # Add slack variable for feasibility
slack_value: 1.0e0             # Slack penalty magnitude
number_of_robots: 2            # Total number of robots

robot_1:
  start_x: -5.0                # Initial x-position
  start_y: 1.0                 # Initial y-position
  start_theta: 0.0             # Initial heading (radians)

robot_2:
  start_x: -4.0
  start_y: -1.0
  start_theta: 0.0

solver_settings:
  iterations_centralised: 2    # Iterations for centralised solver
  iterations_distributed: 2    # Iterations for distributed NMPC-CA loop
  solver_type: SQP_RTI         # Solver type: SQP_RTI (default), SQP
  control_frequency: 5.0       # Control update frequency (Hz)
  braking_acceleration: 0.4    # Emergency braking acceleration

polytopic:
  d_min: 0.1                   # Minimum separation distance
  length: 0.5                  # Robot bounding box length
  width: 0.5                   # Robot bounding box width

contouring:
  num_segments: 5              # Segments for contouring path

weights:
  throttle: 0.1                # Throttle effort penalty
  steering: 0.1                # Steering effort penalty
  lambda: 0.1                  # Penalty on dual variables
  s_dual: 0.1                  # Penalty on slack variables s
  dmin_objective: 0.1          # Penalty on proximity constraint
```

```
  velocity: 0.6                 # Velocity tracking penalty
  reference_velocity: 0.5       # Reference speed penalty
  lag: 0.7                      # Lag error penalty
  contour: 0.7                  # Contour error penalty
```

Listing A.2: Final parameter configuration T-junction scenario

```
N: 20                            # Prediction horizon length
integrator_step: 0.3            # Integration step size
skip_solver_generation: false    # Skip solver code generation (assume precompiled)
add_slack: true                 # Add slack variable for feasibility
slack_value: 1.0e0              # Slack penalty magnitude
number_of_robots: 2             # Total number of robots

robot_1:
  start_x: -5.0                 # Initial x-position
  start_y: 0.0                  # Initial y-position
  start_theta: 0.0              # Initial heading (radians)

robot_2:
  start_x: 0.0
  start_y: -2.0=1
  start_theta: 0.5

solver_settings:
  iterations_centralised: 2     # Iterations for centralised solver
  iterations_distributed: 2     # Iterations for distributed NMPC-CA loop
  solver_type: SQP_RTI          # Solver type: SQP_RTI (default), SQP
  control_frequency: 5.0        # Control update frequency (Hz)
  braking_acceleration: 0.4     # Emergency braking acceleration

polytopic:
  d_min: 0.1                    # Minimum separation distance
  length: 0.5                   # Robot bounding box length
  width: 0.5                    # Robot bounding box width

contouring:
  num_segments: 5               # Segments for contouring path

weights:
  throttle: 0.1                 # Throttle effort penalty
  steering: 0.1                 # Steering effort penalty
  lambda: 0.1                  # Penalty on dual variables
  s_dual: 0.1                  # Penalty on slack variables s
  dmin_objective: 0.1           # Penalty on proximity constraint
  velocity: 0.7                 # Velocity tracking penalty
  reference_velocity: 0.5       # Reference speed penalty
  lag: 0.2                      # Lag error penalty
  contour: 0.7                  # Contour error penalty
```

A.2. Other Distributed Approaches

A.2.1. Approach: s-Only CA

In this approach the NMPC optimiser has the freedom to compute the path including optimising for λ_{ij}. Variable λ_{ij} is then communicated and neighbouring agents use this lambda as parameter λ_{ji} in their NMPC and CA solver.

The resulting optimisation problems can be found below.

Result: the CA solver does not find a solution for the trajectories and lambda's given.

Possible cause: The agents operate too much independent of one another and there is no solution possible in the form of separating hyperplane parallel to the supporting vectors defined by λ_{ij} and λ_{ji}.

$$\begin{aligned}
\min_{\mathbf{u}^i(\cdot|t),\, \lambda_{\mathbf{ij}}(\cdot|t)} \quad & \sum_{i=1}^{M} \Big[\|\mathbf{v}(k|t) - v_{ref}\|^2_{\mathbf{Q}_{v_{ref}}} + \|\varepsilon^{\mathbf{c}}(k|t)\|^2_{\mathbf{Q}_{\varepsilon^c}} + \left\|\varepsilon^{\mathsf{lag}}(k|t)\right\|^2_{\mathbf{Q}_{\varepsilon^{\mathsf{lag}}}} + \|\mathbf{u}(k|t)\|^2_{\mathbf{Q}_u} \\
& \quad + \|\lambda_{\mathbf{ij}}(k|t)\|^2_{\mathbf{Q}_{\lambda_{ij}}} \Big] && \text{(A.1a)} \\
\text{subject to} \quad & (3.1b), (3.1c), (3.1d), \\
& - \mathbf{b}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) - \mathbf{b}^j(\mathbf{z}^j(k|t))^\top \overline{\lambda}_{ji}(k|t) \geq d_{\min}, && \text{(A.1b)} \\
& \mathbf{A}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) + \overline{s}_{ij}(k|t) = 0, && \text{(A.1c)} \\
& \text{for all } k \in \{1, 2, \ldots, N\}.
\end{aligned}$$

$$\begin{aligned}
\min_{s_{ij}(\cdot|t)} \quad & \|\mathbf{s_{ij}}(k|t)\|^2_{\mathbf{Q}_{s_{ij}}} && \text{(A.2a)} \\
\text{subject to} \quad & \overline{\mathbf{A}}^i(\overline{\mathbf{z}}^i(k|t))^\top \overline{\lambda}_{ij}(k|t) + s_{ij}(k|t) = 0, && \text{(A.2b)} \\
& \overline{\mathbf{A}}^j(\overline{\mathbf{z}}^j(k|t))^\top \overline{\lambda}_{ji}(k|t) - s_{ij}(k|t) = 0, && \text{(A.2c)} \\
& \|\mathbf{s_{ij}}(k|t)\|_2 \leq 1, && \text{(A.2d)} \\
& \forall k \in \{1, 2, \ldots, N\}.
\end{aligned}$$

A.2.2. Approach: Fixed-λ_{ij} CA

In this approach the λ_{ij} is optimised in the NMPC optimiser and used as a parameter in the CA solver.

motivation: the NMPC solver has a larger solution space to find a solution for the trajectory while optimising λ_{ij}.

result: Ether agent one or agent two finds a solution.

Possible cause: λ_{ij} and λ_{ji} are optimised independent of each other, so or the separating hyperplane, s, is aligned with the trajectory of agent 1 or agent 2.

$$\begin{aligned}
\min_{\mathbf{u}^i(\cdot|t),\, \lambda_{\mathbf{ij}}(\cdot|t)} \quad & \sum_{i=1}^{M} \Big[\|\mathbf{v}(k|t) - v_{ref}\|^2_{\mathbf{Q}_{v_{ref}}} + \|\varepsilon^{\mathbf{c}}(k|t)\|^2_{\mathbf{Q}_{\varepsilon^c}} + \left\|\varepsilon^{\mathsf{lag}}(k|t)\right\|^2_{\mathbf{Q}_{\varepsilon^{\mathsf{lag}}}} + \|\mathbf{u}(k|t)\|^2_{\mathbf{Q}_u} \\
& \quad + \|\lambda_{\mathbf{ij}}(k|t)\|^2_{\mathbf{Q}_{\lambda_{ij}}} \Big] && \text{(A.3a)} \\
\text{subject to} \quad & (3.1b), (3.1c), (3.1d), \\
& - \mathbf{b}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) - \mathbf{b}^j(\mathbf{z}^j(k|t))^\top \overline{\lambda}_{ji}(k|t) \geq d_{\min}, && \text{(A.3b)} \\
& \mathbf{A}^i(\mathbf{z}^i(k|t))^\top \lambda_{ij}(k|t) + \overline{s}_{ij}(k|t) = 0, && \text{(A.3c)} \\
& \text{for all } k \in \{1, 2, \ldots, N\}.
\end{aligned}$$

$$\max_{\lambda_{ji}(\cdot|t),\, s_{ij}(\cdot|t)} \quad -\overline{\mathbf{b}}^i(\overline{\mathbf{z}}^i(k|t))^\top \overline{\lambda}_{ij}(k|t) - \overline{\mathbf{b}}^j(\overline{\mathbf{z}}^j(k|t))^\top \lambda_{ji}(k|t) \tag{A.4a}$$

subject to

$$\overline{\mathbf{A}}^i(\overline{\mathbf{z}}^i(k|t))^\top \overline{\lambda}_{ij}(k|t) + s_{ij}(k|t) = 0, \tag{A.4b}$$

$$\overline{\mathbf{A}}^j(\overline{\mathbf{z}}^j(k|t))^\top \lambda_{ji}(k|t) - s_{ij}(k|t) = 0, \tag{A.4c}$$

$$-\overline{\mathbf{b}}^i(\overline{\mathbf{z}}^i(k|t))^\top \overline{\lambda}_{ij}(k|t) - \overline{\mathbf{b}}^j(\overline{\mathbf{z}}^j(k|t))^\top \lambda_{ji}(k|t) \geq d_{\min}, \tag{A.4d}$$

$$\|s_{ij}(k|t)\|_2 \leq 1, \tag{A.4e}$$

$$\lambda_{ji}(k|t) \geq 0, \tag{A.4f}$$

$$\forall k \in \{1, 2, \ldots, N\}.$$