Distributed Model Predictive Control for Multi-Agent Systems: Conception and Design for Dynamic Routing Task

Fransiska
DFKI GmbH
Kaiserslautern, Germany
fransiska.fransiska@dfki.de

Dennis Salzmann
DFKI GmbH
Kaiserslautern, Germany
dennis.salzmann@dfki.de

Hans D. Schotten
RPTU Kaiserslautern-Landau
Kaiserslautern, Germany
schotten@rptu.de

***Abstract*—This paper presents a sequential Distributed Model Predictive Control (DMPC) solution for multi-robot navigation. The method targets scenarios where multiple agents must reach designated goals while avoiding collisions in a shared environment. Unlike centralized Model Predictive Control (MPC), which suffers from scalability and communication bottlenecks, the proposed solution decomposes the problem into local optimizations solved sequentially in a fixed cyclic order. Collision avoidance is enforced through dynamically activated constraints, which are introduced only when predicted inter-agent distances fall below a threshold, thereby reducing conservatism while ensuring safety. The solution is implemented in Robot Operating System (ROS) using the ChoiRbot library [1] and extended to support unicycle dynamics, two-dimensional motion, and soft Manhattan distance constraints. Simulation results in Gazebo with TurtleBot3 robots demonstrate collision-free convergence in both sparse two-agent and dense four-agent intersection scenarios. These findings highlight the potential of sequential DMPC as a scalable and communication-efficient solution for safe multi-robot coordination.**

I. Introduction

Multi-robot systems are now becoming common in industry and research [2], [3], [4]. However, when robots differ in capabilities, sensing, or tasks, coordinating their motion in a shared environment becomes non-trivial [5]. This creates the need for control strategies that ensure both safety and efficiency. Addressing this challenge is critical for deploying robust and scalable autonomous systems in real-world conditions.

Multi-agent coordination strategies range from classical formation control to predictive optimization methods. Although formation control has been widely applied in swarms, convoys, and underwater exploration, it faces critical limitations in heterogeneous and obstacle-rich environments [6]. MPC and its distributed variant (DMPC) address these challenges by jointly planning trajectories, handling constraints, and integrating path planning with path following in a unified optimization framework [7], [8]. These properties make MPC-based methods particularly promising for scalable, resilient multi-robot coordination under real-world conditions [9].

This work addresses decentralized multi-robot navigation using a sequential DMPC approach for ground robots with unicycle dynamics. Each agent solves a local MPC problem at every control cycle, accounting for its kinematic constraints and collision avoidance, while exchanging trajectory information with neighboring robots in a ring communication topology. The system is implemented and validated in ROS2 and Gazebo, with TurtleBot3 Burger robots as test agents. Inter-agent communication and distributed control are handled using ChoiRbot, extended here to support two-dimensional motion, trajectory coordination, and collision-avoidance constraints.

The primary contribution of this work is a practical and scalable DMPC architecture that enables decentralized collision avoidance with minimal communication overhead. We validated the solution's effectiveness through a realistic simulation testbed built with ROS 2 and Gazebo. In this setup, TurtleBot3 robots coordinate their planned paths by exchanging trajectory information with their neighbors over a predefined ring communication topology. This system demonstrates the feasibility of the sequential DMPC approach for robust, decentralized trajectory planning and serves as a testbed for scalable multi-robot coordination algorithms.

The remainder of this paper is organized as follows. Section II provides the necessary background, reviewing the principles of Model Predictive Control and the relevant literature on multi-robot coordination. Section III details the proposed methodology, including the unicycle robot model, the specific formulation of the sequential DMPC controller, and the communication architecture. Section IV presents the experimental setup and discusses the simulation results, and finally, Section V concludes the paper with an outline of future work.

II. Background

A. Model Predictive Control

MPC is a receding-horizon strategy that predicts the system's evolution over a finite horizon, computes the optimal control sequence, and applies only the first input before repeating the process at the next step [10]. This predictive mechanism enables MPC to adapt continuously to disturbances and interactions. A key advantage is its ability to explicitly handle constraints on inputs, states, and inter-agent distances, making it highly suitable for robotic navigation tasks [11], [12]. In multi-robot systems, these features allow MPC to

unify path planning and path following in a single optimization framework, ensuring both convergence to goals and collision-free coordination.

At each control step, the MPC problem is formulated as minimizing a cost function that penalizes deviations from the reference trajectory and the use of control effort [13]:

$$\min_{U(k)} J = \sum_{t=0}^{T-1} \left(\|x(k+t) - x_{\text{ref}}(k+t)\|_Q^2 + \|u(k+t)\|_R^2 \right), \tag{1}$$

subject to the system dynamics and constraints

$$\text{State constraints:} \quad x_{\min} \leq x(k+t) \leq x_{\max}, \tag{2}$$

$$\text{Input constraints:} \quad u_{\min} \leq u(k+t) \leq u_{\max}, \tag{3}$$

$$\text{System dynamics:} \quad x(k+1) = f(x(k), u(k)), \tag{4}$$

Here, $x(k+t) \in \mathbb{R}^n$ represents the predicted system state, $x_{\text{ref}}(k+t)$ is the reference trajectory, $u(k+t) \in \mathbb{R}^m$ is the control input, and $Q \succeq 0$, $R \succ 0$ are weighting matrices that prioritize tracking accuracy and control effort, respectively.

The optimization is subject to the system dynamics and constraints summarized in Eqs. (3)–(4). The dynamics in Eq. (4) ensure that the predicted trajectory respects the system model. The state constraints in Eq. (2) limit the predicted states to admissible ranges, while the input constraints in Eq. (3) restrict control inputs to feasible values.

By solving this optimization at every control step and applying only the first input $u(k)$, MPC implements a receding horizon strategy that adapts to disturbances and model uncertainties, ensuring feasible and smooth trajectories while respecting constraints.

While MPC is effective for controlling a single robot, scaling to multi-robot systems requires coordination across agents. A centralized MPC approach can jointly optimize all trajectories, but this quickly becomes computationally demanding as the number of robots increases and requires global state information.

Distributed MPC (DMPC) addresses these limitations by allowing each agent to solve its own local problem while sharing predictions with neighbors, improving scalability and modularity [12]. However, many distributed schemes rely on iterative exchanges per control cycle, which may not be feasible in real time [14].

B. Related Work

Early research in multi-robot coordination focused on *formation control*, where agents maintain fixed spatial relationships to move as a group. While effective for synchronized motion in UAV swarms and convoys, these methods rely on strong assumptions of homogeneity and static environments. As noted by Liu et al. [6], formation control struggles with heterogeneous agents and obstacle-rich environments, where rigid formations must reconfigure or break, limiting flexibility and scalability.

To overcome these limitations, predictive approaches based on *Model Predictive Control* (MPC) have gained prominence [15], [16]. MPC predicts the system's evolution over a finite horizon, computes the optimal control sequence, and applies only the first input before repeating the process [10], [17]. Its ability to explicitly handle input, state, and distance constraints makes it well suited for robotic navigation and coordination [11], [12]. By combining short-horizon planning and feedback control, MPC can unify path planning and path following, ensuring both goal convergence and collision avoidance [18], [19].

However, centralized MPC architectures scale poorly with the number of robots and depend on global communication [9]. Distributed MPC (DMPC) addresses these issues by allowing each agent to solve a local optimization problem while exchanging predicted trajectories with neighbors [12], [20]. Sequential and iterative DMPC variants improve scalability and coordination: sequential approaches reduce communication by optimizing agents in order [21], while iterative schemes enhance performance through repeated exchanges at higher computational cost [22]–[24]. Hierarchical MPC further improves scalability by separating high-level planning from low-level control [25], [26], though at the cost of greater system complexity.

A critical design aspect in MPC-based navigation is model selection. Kinematic models, which capture geometric motion while ignoring dynamics, are computationally efficient and enable convex formulations [27]–[29]. Dynamic models offer higher fidelity but introduce nonlinearities that hinder real-time performance [17]. Comparative studies confirm that linearized kinematic MPC provides an excellent trade-off between efficiency and accuracy for moderate-speed mobile robots [27], [28].

In navigation, MPC bridges the gap between global path planning and local control. Classical planners such as A* and RRT generate feasible global paths [30], [31], while local controllers like Pure Pursuit or Stanley ensure path following [32]. MPC integrates both roles, continuously optimizing trajectories with built-in obstacle avoidance and goal-tracking [18], [19]. Convex relaxations such as Manhattan-distance surrogates [23] are commonly used to retain feasibility with solvers like CVXOPT under tight constraints.

Overall, the literature reflects a progression from rigid formation control toward predictive, distributed methods such as DMPC. Among these, sequential DMPC achieves a practical balance between coordination quality, real-time feasibility, and scalability, making it an effective approach for multi-robot navigation in dynamic environments.

III. Methodology

To balance tractability and coordination, we adopt a *sequential DMPC* strategy. In this scheme, agents update their trajectories one after another in a fixed cyclic order, each incorporating the most recent predictions of its predecessors. This reduces communication overhead compared to iterative schemes, while still ensuring that coupling constraints such as collision avoidance are respected [21]. The approach is implemented using the ROS 2-based ChoiRbot framework [1],

extended here to support unicycle dynamics and on-demand collision-avoidance constraints.

ChoiRbot Framework

The proposed approach is implemented using the ChoiRbot framework [1], a ROS 2-compatible library for distributed optimization and cooperative control. ChoiRbot provides a lightweight infrastructure for defining local MPC problems and exchanging trajectories among agents, making it well suited for testing DMPC strategies. In this work, we extend the framework from its original single-integrator model to support unicycle dynamics, two-dimensional motion, and dynamic collision-avoidance constraints, enabling realistic multi-robot navigation experiments with TurtleBot3 robots in Gazebo.

A. Robot Model and Dynamics

Each agent is modeled as a unicycle robot with state $x(k) = [p_x(k),\, p_y(k),\, \theta(k)]^\top$, where $p_x(k)$ and $p_y(k)$ denote the planar position, and $\theta(k)$ is the heading angle. The input is defined as $u(k) = [v(k),\, \omega(k)]^\top$, with $v(k)$ being the linear velocity along the robot's heading and $\omega(k)$ the angular velocity. The discrete-time dynamics are expressed as

$$x(k+1) = A(\theta(k))\, x(k) + B(\theta(k))\, u(k),$$

with

$$A(\theta(k)) = \begin{bmatrix} 1 & 0 & -dt\, v(k)\, \sin(\theta(k)) \\ 0 & 1 & dt\, v(k)\, \cos(\theta(k)) \\ 0 & 0 & 1 \end{bmatrix}, \tag{5}$$

$$B(\theta(k)) = \begin{bmatrix} dt\cos(\theta(k)) & 0 \\ dt\sin(\theta(k)) & 0 \\ 0 & dt \end{bmatrix}, \tag{6}$$

where dt is the sampling time.

The matrix $A(\theta(k))$ in Eq. (5) captures the heading-dependent state evolution, while $B(\theta(k))$ in Eq. (6) maps the control inputs to state changes. This linearized form retains the nonlinearity of the unicycle model while maintaining convexity for efficient MPC optimization.

B. Controller Formulation

At each control cycle k, agent i solves a finite-horizon quadratic program (QP) based on the general MPC formulation in Eqs. (3)–(4), with linearized dynamics to track its reference while enforcing safety through *dynamically activated* collision-avoidance constraints and a terminal *guiding* constraint. Let $t \in \{0, \ldots, T-1\}$ denote prediction steps and use the shorthand $x_t^{(i)} = x^{(i)}(k+t)$, $u_t^{(i)} = u^{(i)}(k+t)$.

Objective: The agent-specific cost function extends the general cost in Eq. (1) by including collision-avoidance and slack penalties:

$$J^{(i)}(k) = \sum_{t=0}^{T-1} \Big(\underbrace{\|x_t^{(i)} - x_{\text{ref},t}^{(i)}\|_Q^2}_{\text{state tracking}} + \underbrace{\|u_t^{(i)}\|_R^2}_{\text{control effort}} + \lambda_s \sum_{j \in \mathcal{N}_i} s_t^{(i,j)} + \lambda_a \sum_{j \in \mathcal{N}_i} \big(a_{x,t}^{(i,j)} + a_{y,t}^{(i,j)}\big) \Big), \tag{7}$$

Subject to constraints (all $t = 0, \ldots, T-1$ unless stated): These constraints correspond to the general bounds and dynamics in Eqs. (3)–(4), specialized for agent i:

$$\text{Dynamics:} \quad x_{t+1}^{(i)} = A x_t^{(i)} + B(\theta_t^{(i)})\, u_t^{(i)}, \tag{8}$$

$$\text{Input bounds:} \quad u_{\min} \le u_t^{(i)} \le u_{\max}, \tag{9}$$

$$\text{State bounds:} \quad x_{\min} \le x_t^{(i)} \le x_{\max}, \tag{10}$$

$$\text{Initial condition:} \quad x_0^{(i)} = x_{\text{current}}^{(i)}. \tag{11}$$

Here, $s_t^{(i,j)} \ge 0$ is a slack variable acting as a soft safety margin, and $a_{x,t}^{(i,j)}, a_{y,t}^{(i,j)} \ge 0$ are Manhattan auxiliaries for collision avoidance. The matrices Q and R weight deviations from the reference state and control effort, while λ_s and λ_a penalize reliance on slack or auxiliary variables. The heading reference θ_{ref} is computed from the goal direction and de-weighted when the agent is within 0.5 m of the target to reduce oscillations.

In summary, this agent-specific QP can be seen as a direct specialization of the generic MPC problem in Eqs. (1)–(4), enriched with safety and coordination terms for the multi-agent scenario.

Terminal guiding constraint: To encourage steady progress toward the goal, a terminal *guiding constraint* is imposed at the end of the prediction horizon. At time step k, an advancing subgoal is defined via linear interpolation with parameter $\alpha \in (0, 1)$:

$$x_{\text{term}}^{(i)}(k) = x^{(i)}(k) + \alpha\big(x_{\text{goal}}^{(i)} - x^{(i)}(k)\big),$$
$$y_{\text{term}}^{(i)}(k) = y^{(i)}(k) + \alpha\big(y_{\text{goal}}^{(i)} - y^{(i)}(k)\big).$$

At the terminal prediction step $t = T$, the agent's predicted position is constrained to remain within a tolerance box around the subgoal:

$$x_{\text{term}}^{(i)}(k) - \epsilon(k) \le x_T^{(i)} \le x_{\text{term}}^{(i)}(k) + \epsilon(k),$$
$$y_{\text{term}}^{(i)}(k) - \epsilon(k) \le y_T^{(i)} \le y_{\text{term}}^{(i)}(k) + \epsilon(k), \tag{12}$$

where the tolerance $\epsilon(k)$ decreases with the current distance to the goal and is clamped to predefined bounds to ensure feasibility on short horizons while promoting consistent convergence.

The resulting optimization problem is solved using CVXOPT [33], a QP solver integrated into the DISROPT library within CHOIRBOT [1]. As CVXOPT supports only linear equality and inequality constraints, the problem formulation in Eqs. (8)–(12) is maintained in convex form. However, when numerous constraints are simultaneously active, numerical conditioning may degrade, potentially leading to reported infeasibility despite theoretical feasibility. To mitigate these issues, collision-avoidance constraints are expressed in a convex Manhattan formulation and are *dynamically activated* only when potential conflicts are predicted, thereby improving both numerical robustness and real-time tractability.

Dynamic collision avoidance (on-demand Manhattan form): For each neighboring agent $j \in \mathcal{N}_i$, the predicted pairwise distance is computed as

$$d_t^{(i,j)} = \left\| [\, x_t^{(i)} - x_t^{(j)},\, y_t^{(i)} - y_t^{(j)} \,]^\top \right\|_2.$$

The collision-avoidance constraint is *dynamically activated* when the predicted minimum distance $\min_t d_t^{(i,j)}$ falls below a predefined activation threshold d_{activate}. This activation strategy limits constraint enforcement to relevant time steps, improving numerical efficiency and scalability.

To preserve convexity of the overall Quadratic Program (QP), the Euclidean separation constraint $d_t^{(i,j)} \geq d_{\min}$ is relaxed into a linear Manhattan form:

$$|x_t^{(i)} - x_t^{(j)}| \leq a_{x,t}^{(i,j)}, \qquad |y_t^{(i)} - y_t^{(j)}| \leq a_{y,t}^{(i,j)}, \tag{13}$$

$$a_{x,t}^{(i,j)} + a_{y,t}^{(i,j)} + s_t^{(i,j)} \geq d_{\min}, \qquad s_t^{(i,j)} \in [0,\, s_{\max}], \tag{14}$$

where $a_{x,t}^{(i,j)}$ and $a_{y,t}^{(i,j)}$ are nonnegative auxiliary variables that bound the absolute position differences in the x- and y-directions, respectively, and $s_t^{(i,j)} \geq 0$ is a slack variable that relaxes the constraint to ensure feasibility under close interactions. The parameter $d_{\min}$ defines the minimum allowable distance between any two agents, and $s_{\max}$ caps the permissible violation range.

The penalty terms associated with $s_t^{(i,j)}$ and $a_{x,t}^{(i,j)}, a_{y,t}^{(i,j)}$ appear in the local objective function in Eq. (7), weighted by coefficients $\lambda_s \gg \lambda_a \geq 0$. A large λ_s enforces near-hard safety by discouraging slack activation, while smaller λ_a values maintain convexity and numerical stability. Together, Eqs. (13)–(14) define a convex surrogate that approximates circular collision boundaries with Manhattan geometry, ensuring tractable and feasible multi-agent optimization even in dense environments.

C. Communication and Execution Flow

The agents interact over an undirected communication graph defined by an adjacency matrix. A ring topology is adopted, where each agent communicates with its two immediate neighbors. For the four-agent setup, the adjacency matrix is:

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

This ensures that agent i exchanges trajectories with agents $(i-1)$ and $(i+1)$, enabling distributed coordination based on local information only. Communication is assumed instantaneous unless otherwise specified.

Execution follows a sequential scheme: each agent waits until it receives the updated trajectory from its predecessor before solving its local MPC problem. After optimization, the agent forwards its predicted trajectory to the next neighbor, and the first control inputs are applied.

Algorithm 1 Sequential DMPC Execution per Agent

```
Receive predicted trajectories from neighbors
Compute predicted inter-agent distances (Euclidean)
if any predicted distance < d_activate then
    Initialize MPC with motion and collision constraints (Manhattan form)
else
    Initialize MPC with basic motion constraints only
end if
Solve the local MPC optimization problem
Apply first control input and shift the prediction horizon
Share updated predicted trajectory with neighbors
Loop to next control cycle
```

The procedure is summarized in Algorithm 1. At each cycle, agents incorporate the latest shared trajectories, predict inter-agent distances, and activate collision-avoidance constraints only if a violation is imminent. This enables efficient on-demand collision avoidance without introducing unnecessary conservatism.

IV. Experiment and Results

The evaluation is designed to verify that the proposed DMPC system enables all agents to reach their assigned goals while avoiding inter-agent collisions. Two performance aspects are considered: convergence and safety. Convergence is assessed by monitoring the distance-to-goal over time for each agent, which shows whether agents consistently approach their targets. Safety is quantified by the minimum pairwise distance between agents,

$$d_{\min}(k) = \min_{i \neq j} \|x_i(k) - x_j(k)\|_2,$$

which must remain above the safety threshold d_{safe} throughout the simulation. In addition to these quantitative measures, qualitative trajectory plots are used to illustrate how the agents interact and avoid collisions under different scenarios.

A. First Case: Two-Agent Crossing Scenario

To validate the collision avoidance mechanism, we begin with a simple two-agent setup. The agents start at opposite corners of the environment and are assigned diagonally opposite goals, which forces their paths to cross near the center, as implemented in [23]. This configuration directly tests whether the DMPC controller can ensure safety while both agents converge to their targets.

As shown in Fig. 1, the agents briefly deviate from their straight paths when the predicted inter-agent distance approaches the safety margin. This deviation originates from the optimization problem itself: the collision-avoidance constraint activates only when necessary and deactivates once separation is restored. After passing the point of closest encounter, both agents resume direct goal-seeking motion.

Fig. 2 confirms that the minimum pairwise distance never falls below the threshold. The constraint activates preemptively, resolving the potential conflict without oscillations or

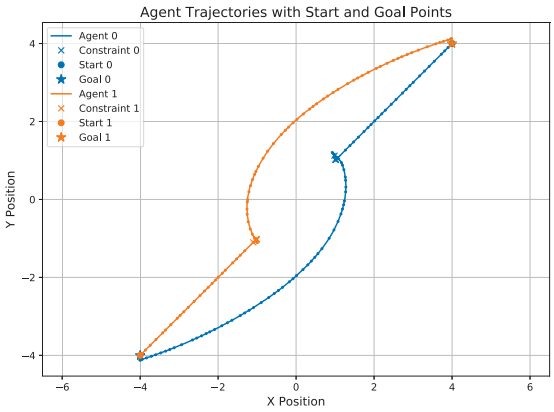

Fig. 1. Two-agent crossing scenario: planned trajectories. Both robots reach their goals without collisions

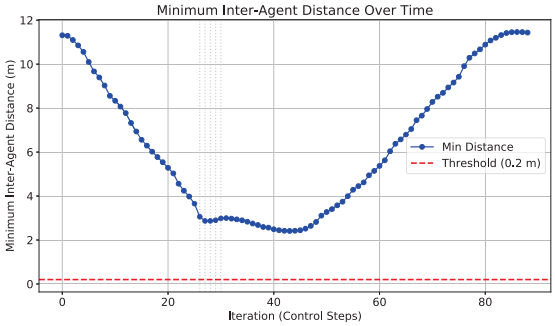

Fig. 2. Two-agent crossing scenario: minimum inter-agent distance over time. The red dashed line indicates the safety threshold $d_{\text{safe}} = 0.2$ m

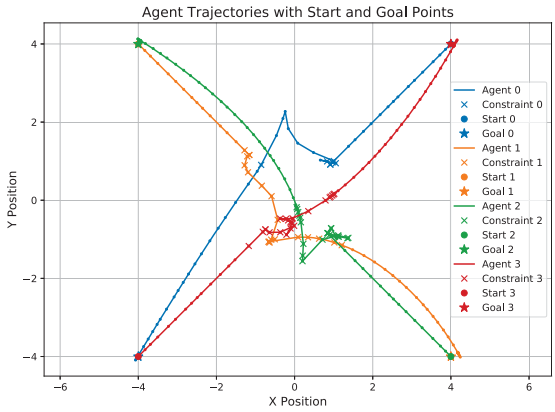

Fig. 3. Four-agent intersection scenario: planned trajectories. All robots pass through the central region without collisions

Fig. 4. Four-agent intersection scenario: minimum inter-agent distance over time. The red dashed line marks the safety threshold $d_{\text{safe}} = 0.2$ m

deadlocks. Both robots reach their goals, demonstrating that the distributed controller ensures collision-free convergence even in direct crossing encounters.

B. Second Case: Four-Agent Fully Intersecting Scenario

To test robustness under dense interactions, a scenario with four agents and fully intersecting trajectories was simulated. Each agent starts at a corner of the environment and is assigned the diagonally opposite goal, creating simultaneous conflicts in the center.

As shown in Fig. 3, the agents deviate smoothly from their straight paths as they approach the intersection zone. Multiple avoidance constraints activate in parallel, yet all agents maintain progress toward their goals. The avoidance maneuvers are transient and anticipatory: once safe spacing is restored, each robot resumes a direct route to its target. This demonstrates that the distributed formulation can handle overlapping interactions without centralized coordination.

Fig. 4 confirms that the minimum pairwise distance never falls below the safety threshold, despite multiple agents entering the central region simultaneously. The lowest value is observed around iteration 20–30, corresponding to peak congestion, but safety is preserved throughout. All four agents converge successfully, showing that the proposed DMPC framework ensures feasibility, collision avoidance, and convergence even under high-density, multi-agent interactions.

Both scenarios confirm that the controller achieves collision-free convergence under sparse and dense interactions, without centralized coordination.

V. Conclusion

This work presented an implementation of a sequential distributed Model Predictive Control (DMPC) method for multi-robot navigation, developed in ROS 2 using the ChoiRbot library and evaluated in Gazebo simulations. The approach combines predictive planning with dynamic, on-demand collision-avoidance constraints, allowing multiple agents to coordinate their motion without centralized control. Results from two benchmark scenarios, a two-agent crossing and a four-agent full intersection, demonstrated collision-free convergence to their respective goals. These results confirm the effectiveness of sequential DMPC for safe and scalable multi-robot coordination.

Several directions remain for future work. First, the scalability of the sequential DMPC approach should be further investigated, particularly regarding how communication delays and fleet size affect performance. Second, systematic methods for tuning cost weights and penalty parameters could

improve robustness across diverse environments. Third, while the current study focused on homogeneous agents, the formulation naturally extends to heterogeneous robots with differing dynamics, sensing capabilities, and task priorities. Finally, incorporating nonlinear MPC formulations and validating the system on physical robot platforms represent promising steps toward real-world deployment.

ACKNOWLEDGMENT

The authors acknowledge the financial support by the German *Federal Ministry of Research, Technology and Space (BMFTR)* within the project »Twin4Trucks« {13IK010F}.

REFERENCES

[1] A. Testa, A. Camisa, and G. Notarstefano, "ChoiRbot: A ROS 2 Toolbox for Cooperative Robotics," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2714–2720, 2021.

[2] A. Krnjaic, R. D. Steleac, J. D. Thomas, G. Papoudakis, L. Schäfer, A. W. Keung To, K.-H. Lao, M. Cubuktepe, M. Haley, P. Börsting, and S. V. Albrecht, "Scalable multi-agent reinforcement learning for warehouse logistics with robotic and human co-workers," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 677–684.

[3] J. Sang, D. Ma, X. Xie *et al.*, "Group-aggregation of hierarchical containment control for homogeneous multi-agent systems in precision agriculture," *International Journal of Control, Automation and Systems*, vol. 22, pp. 1400–1408, 2024.

[4] Z. Liu, J. Zhang, E. Shi, Z. Liu, D. Niyato, B. Ai, and X. Shen, "Graph neural network meets multi-agent reinforcement learning: Fundamentals, applications, and future directions," *IEEE Wireless Communications*, vol. 31, no. 6, pp. 39–47, 2024.

[5] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems: A survey," *ACM Comput. Surv.*, vol. 52, no. 2, Apr. 2019. [Online]. Available: https://doi.org/10.1145/3303848

[6] Y. Liu, J. Liu, Z. He, Z. Li, Q. Zhang, and Z. Ding, "A survey of multi-agent systems on distributed formation control," *Unmanned Systems*, vol. 12, no. 05, pp. 913–926, 2024. [Online]. Available: https://doi.org/10.1142/S2301385024500274

[7] Z. Zuo, X. Yang, Z. Li, Y. Wang, Q. Han, L. Wang, and X. Luo, "Mpc-based cooperative control strategy of path planning and trajectory tracking for intelligent vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 3, pp. 513–522, 2021.

[8] K. Zhang, J. Sprinkle, and R. G. Sanfelice, "A hybrid model predictive controller for path planning and path following," in *Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems*, ser. ICCPS '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 139–148. [Online]. Available: https://doi.org/10.1145/2735960.2735966

[9] L. Dai, Y. Ma, R. Gao, J. Wu, and Y. Xia, "Cloud-based computational model predictive control using a parallel multiblock admm approach," *IEEE Internet of Things Journal*, vol. 10, no. 12, pp. 10 326–10 343, 2023.

[10] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109899002149

[11] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Santa Barbara, California: Nob Hill Publishing, 2020.

[12] R. Negenborn and J. Maestre, "Distributed model predictive control: An overview and roadmap of future research opportunities," *IEEE Control Systems Magazine*, vol. 34, no. 4, pp. 87–97, 2014.

[13] E. F. Camacho and C. Bordons, *Model Predictive Control*, 1st ed., ser. Advanced textbooks in control and signal processing. Springer, 1999.

[14] J. Liu, X. Chen, D. M. Muñoz de la Pena, and P. D. Christofides, "Iterative distributed model predictive control of nonlinear systems: Handling asynchronous, delayed measurements," *IEEE Transactions on Automatic Control*, vol. 57, no. 2, pp. 528–534, 2012.

[15] X. Zhang, W. Pan, C. Li, X. Xu, X. Wang, R. Zhang, and D. Hu, "Toward scalable multirobot control: Fast policy learning in distributed mpc," *IEEE Transactions on Robotics*, vol. 41, pp. 1491–1512, 2025.

[16] A. da C. Vangasse, E. J. R. Freitas, G. V. Raffo, and L. C. A. Pimenta, "Safe navigation on path-following tasks: A study of mpc-based collision avoidance schemes in distributed robot systems," *Journal of Intelligent & Robotic Systems*, vol. 110, no. 4, p. 166, 2024. [Online]. Available: https://doi.org/10.1007/s10846-024-02202-3

[17] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Santa Barbara, California: Nob Hill Publishing, 2020.

[18] E. Schöneberg, M. Schröder, D. Görges, and H. D. Schotten, "Trajectory planning with model predictive control for obstacle avoidance considering prediction uncertainty," 2025. [Online]. Available: https://arxiv.org/abs/2504.19193

[19] M. Mohaghegh, S.-A. Saeedinia, and Z. Roozbehi, "Optimal predictive neuro-navigator design for mobile robot navigation with moving obstacles," *Frontiers in Robotics and AI*, vol. Volume 10 - 2023, 2023. [Online]. Available: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2023.1226028

[20] A. Richards and J. P. H. and, "Robust distributed model predictive control," *International Journal of Control*, vol. 80, no. 9, pp. 1517–1531, 2007. [Online]. Available: https://doi.org/10.1080/00207170701491070

[21] K. S. Narkhede, A. M. Kulkarni, D. A. Thanki, and I. Poulakakis, "A sequential mpc approach to reactive planning for bipedal robots," 2022. [Online]. Available: https://arxiv.org/abs/2205.00156

[22] J. Liu, X. Chen, D. Pena, and P. Christofides, "Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. part i: Theory," 08 2010, pp. 3148 – 3155.

[23] C. E. Luis and A. P. Schoellig, "Trajectory generation for multiagent point-to-point transitions via distributed model predictive control," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 375–382, 2019.

[24] J. Wu, L. Dai, and Y. Xia, "Iterative distributed model predictive control for nonlinear systems with coupled non-convex constraints and costs," *International Journal of Robust and Nonlinear Control*, vol. 34, no. 11, pp. 7220–7244, 2024. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.7341

[25] H. Li, R. J. Frei, and P. M. Wensing, "Model hierarchy predictive control of robotic systems," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3373–3380, 2021.

[26] M. Kögel, M. Ibrahim, C. Kallies, and R. Findeisen, "Safe hierarchical model predictive control and planning for autonomous systems," *International Journal of Robust and Nonlinear Control*, vol. 35, no. 7, pp. 2658–2676, 2025. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.6808

[27] X. Liu and V. Atanassov, "Safe model predictive control approach for non-holonomic mobile robots," 2022. [Online]. Available: https://arxiv.org/abs/2207.12878

[28] K. Mondal, A. Rodriguez, S. Manne, N. Das, and B. Wallace, "Comparison of kinematic and dynamic model based linear model predictive control of non-holonomic robot for trajectory tracking: Critical trade-offs addressed," 12 2019.

[29] M. Bozza, "Kinematic-based model predictive control for space exploration rovers," Master's thesis, Politecnico di Milano, Scuola di Ingegneria Industriale e dell'Informazione, Milano, Italy, Jul. 2024, academic year 2023–2024. [Online]. Available: https://hdl.handle.net/10589/223133

[30] G. Gugan and A. Haque, "Path planning for autonomous drones: Challenges and future directions," *Drones*, vol. 7, no. 3, 2023. [Online]. Available: https://www.mdpi.com/2504-446X/7/3/169

[31] M. Jones, S. Djahel, and K. Welsh, "Path-planning for unmanned aerial vehicles with environment complexity considerations: A survey," *ACM Comput. Surv.*, vol. 55, no. 11, Feb. 2023. [Online]. Available: https://doi.org/10.1145/3570723

[32] J. Liu, Z. Yang, Z. Huang, W. Li, S. Dang, and H. Li, "Simulation performance evaluation of pure pursuit, stanley, lqr, mpc controller for autonomous vehicles," in *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, 2021, pp. 1444–1449.

[33] M. S. Andersen, J. Dahl, and L. Vandenberghe, "CVXOPT: A python package for convex optimization, version 1.3.2," https://cvxopt.org/, Aug. 2023, version as of August 9,2023.