Model Predictive Control (MPC) for Task Allocation under Constraints in Mobile Robotics: Application to Industrial Logistics

Arnaud BELHOMME[1], François GUERIN[2]

Abstract— **This paper introduces a Model Predictive Control (MPC)-based approach for decentralized task allocation in fleets of mobile robots operating under industrial logistics constraints. The proposed framework optimally assigns tasks while accounting for external factors like operator requests. Each robot assesses its ability to perform a given task and resolves assignment conflicts through a structured decision-making process. The system is validated using ROS-based mobile robots in a laboratory setting that replicates an industrial application, showcasing the model's adaptability and efficiency. The results highlight that constraint-based MPC improves the management and flexibility of multi-robot logistics operations in dynamic industrial environments.**

I. INTRODUCTION

In recent years, industrial enterprises have continuously pursued the development of their products, processes, and organizational structures to maintain a competitive edge. In this context, industrial logistics provides essential support by optimizing the supply of tools and raw materials to production lines [1]. Mobile robots constitute a crucial component of modern logistics transportation systems and are widely deployed in industrial settings. Most mobile robots available from major manufacturers operate under centralized control, where a single controller coordinates the entire fleet. However, there is a growing shift towards decentralized control, where individual robots make autonomous decisions, enhancing the system's flexibility, management, and scalability [2].

Task allocation can be performed in either a centralized or decentralized manner. In many situations, a centralized approach becomes impractical due to factors such as infrastructure limitations, vulnerability to single points of failure, communication bottlenecks, and potential loss of connection to the central node. This issue is particularly relevant in scenarios such as military operations, disaster response, and underwater exploration, where infrastructure is unavailable, making centralized task allocation infeasible [3]. Multi-agent systems offer a decentralized alternative, where autonomous agents coordinate and collaborate to achieve a global goal [4]. In recent years, robotics has increasingly been integrated with artificial intelligence, enabling robots to make autonomous decisions [5].

[1]GREAH, University Le Havre Normandie, Le Havre, France arnaud.belhomme@univ-lehavre.fr

[2]GREAH, University Le Havre Normandie, Le Havre, France francois.guerin@univ-lehavre.fr

Model Predictive Control (MPC) is increasingly used in mobile robot task allocation due to its ability to optimize decision-making under constraints. The primary challenge in this domain is to balance computational efficiency, dynamic adaptability, and real-time responsiveness while ensuring safety and optimal performance. Researchers have explored various approaches, including Distributionally Robust Chance-Constrained MPC (DRCC-MPC), Topology-driven MPC (T-MPC), and probabilistic motion planning, to enhance predictive control models for multi-robot coordination.

[6] presents a DRCC-MPC framework designed to enhance safe robot navigation in human-populated environments. By incorporating collision probability as an interpretable risk metric, this method improves management against unpredictable human motion. utilizing conditional value at risk constraints, the approach maintains computational efficiency, enabling real-time, risk-aware navigation in dynamic crowds. [7] explores probabilistic motion planning for autonomous robots operating in dynamic environments. Traditional motion planners often struggle with uncertainty in human movement predictions and fail to adapt to sudden changes. This work introduces Scenario-based (S-MPCC, SH-MPC) and topology-driven (T-MPC) approaches, leveraging multi-modal uncertainty modeling and parallelized trajectory optimization to ensure safe, efficient, and adaptive navigation in human-populated spaces. [8] presents an MPC-based framework for mobile manipulation tasks, enabling robots to adaptively interact with unknown environments without requiring prior object modeling. By integrating online system identification and adaptive control, this approach improves force regulation and motion planning, leading to enhanced performance in tasks such as door opening and object lifting, as demonstrated on a ball-balancing manipulator. [9] introduces an MPC-based decentralized framework for multi-robot object transportation in constrained environments. By facilitating real-time local decision-making while ensuring global coordination, this approach enhances scalability, management, and adaptability in applications such as industrial automation and search and rescue. It also overcomes challenges related to communication and coordination through decentralized MPC strategies. [10] presents a Hierarchical-Task MPC (HT-MPC) framework for sequential mobile manipulation tasks, improving task execution efficiency and reactivity. By leveraging robot redundancy and nonlinear lexicographic optimization, this approach enhances trajectory tracking

by 42% and accelerates execution by 2.3 times. Validated on a 9-DOF mobile manipulator, HT-MPC enables faster, adaptive, and hierarchical task execution.

In our work, we first analysed different task allocation algorithms [11], leading to the development of a decentralized, bio-inspired task allocation method for mobile robots, based on a consensus mechanism [12]. We propose a generalized task allocation algorithm designed for a wide range of industrial logistics systems, setting it apart from existing MPC-based approaches. Unlike [6] and [7], which focus on robot navigation under uncertainty, our work addresses constrained task allocation, ensuring decentralized decision-making and conflict resolution through consensus mechanisms. Compared to [8] and [9], which emphasize environment interaction and multi-robot coordination, our anticipatory MPC-based approach optimally assigns tasks while accounting for external constraints. Moreover, unlike [10], which prioritizes hierarchical task execution, our model provides a robust decentralized framework tailored for industrial logistics, improving system-wide adaptability and efficiency.

In the "Problem Description" section, we describe a system in which a fleet of mobile robots, each equipped with unique capabilities, autonomously selects and executes tasks in an industrial setting. The "Contribution" section introduces a decentralized, constraint-aware MPC algorithm for task allocation, where each robot evaluates its suitability for a given task based on its attributes, selects the optimal task, communicates its choice, resolves potential conflicts in task assignment with other robots, and finally executes its assigned task.

II. PROBLEM DESCRIPTION

We consider a fleet of R mobile robots, with N tasks to complete. Each robot is endowed with unique abilities that are relevant to the completion of these tasks. All robots can complete the tasks. The objective is to develop an embedded algorithm for each robot allowing it to assign itself to the task for which it is most competent, due to battery level (state of charge) and time (distance, speed). Then provide a consensus mechanism in the event of a task assignment conflict between several robots. The solution proposed in this paper will be implemented in an aircraft engine nacelle assembly company, as part of a line-side supply automation project.

Consider the example in figure 1 with 3 robots and 5 workstations. Each operator has a replenishment request push button for his workstation. The robot fleet is then responsible for managing the replenishment of component bins at various workstations. The supply of full bins (using a First In First Out table) is carried out by an operator, as well as the evacuation of empty bins.

For the purpose of a modelization (figure 2), let us consider the following notations :

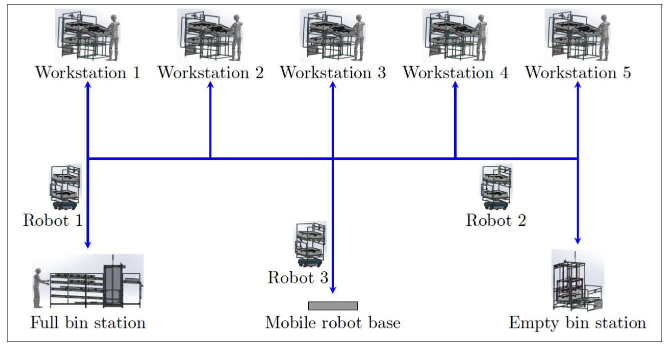

Fig. 1. Industrial application (example for 3 robots and 5 workstations)

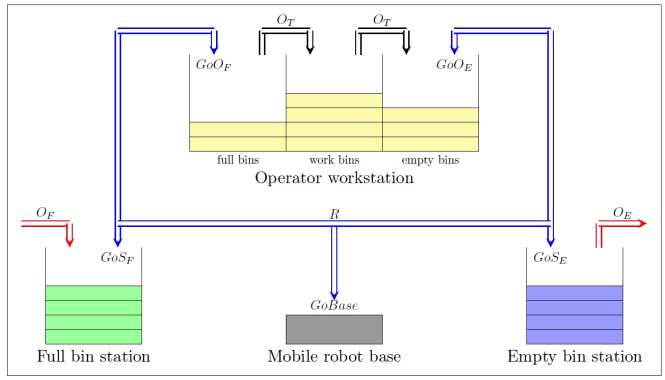

Fig. 2. Workstation modelization (one workstation represented)

- "N" operator workstations ($N \geq 1$), with 2 parts:
  - 1 : full bins => capacity = $Co_F(1)$
  - 2 : empty bins => capacity = $Co_E(1)$
- "R" mobile robots ($R \geq 1$)
- S_F : full bin station => capacity = $C_F(25)$
- S_E : empty bin station => capacity = $C_E(25)$
- O_{T_1}, ... O_{T_N} : workstation operators
- O_F : full bin station operator
- O_E : empty bin station operator
- Each workstation and station is equipped with a "level sensor" connected to an IoT network.
- Each mobile robot can transport "B_{R_1}, ... B_{R_N}" bins and perform actions:
  - GoO_{F_1}, ... GoO_{F_N} = go to the "full" operator workstation 1, ... N + bin deposit
  - GoO_{E_1}, ... GoO_{E_N} = go to the "empty" operator workstation 1, ... N + bin recovery
  - GoS_F = go to full bin station + bin recovery
  - GoS_E = go to empty bin station + bin deposit
  - $GoBase$ = return to base (battery charge)
- The navigation function of mobile robots is managed by SLAM-based navigation in ROS.

The modelization of the system (N workstations and R mobile robots) is therefore:

$$N_{OF_{1_{n+1}}} = N_{OF_{1_n}} + \sum_{i=1}^{R} B_{R_i} - \sum_{j=1}^{0} B_{T_j} \qquad (1)$$

$$\vdots$$

$$N_{OF_{N_{n+1}}} = N_{OF_{N_n}} + \sum\nolimits_{i=1}^{R} B_{R_i} - \sum\nolimits_{j=1}^{0} B_{T_j} \tag{2}$$

$$N_{OE_{1_{n+1}}} = N_{OE_{1_n}} + \sum\nolimits_{j=1}^{0} B_{T_j} - \sum\nolimits_{i=1}^{R} B_{R_i} \tag{3}$$

$$\vdots$$

$$N_{OE_{N_{n+1}}} = N_{OE_{N_n}} + \sum\nolimits_{j=1}^{0} B_{T_j} - \sum\nolimits_{i=1}^{R} B_{R_i} \tag{4}$$

$$N_{F_{n+1}} = N_{F_n} + \sum\nolimits_{j=1}^{0} B_{M_j} - \sum\nolimits_{i=1}^{R} B_{R_i} \tag{5}$$

$$N_{E_{n+1}} = N_{E_n} + \sum\nolimits_{i=1}^{R} B_{R_i} - \sum\nolimits_{j=1}^{0} B_{M_j} \tag{6}$$

With :

- N_{OF_1}, ... N_{OF_N}: Number of bins in full operator workstation
- N_{OE_1}, ... N_{OE_N}: Number of bins in empty operator workstation
- N_F: Number of bins in the full bin station
- N_E: Number of bins in the empty bin station
- B_{R_i}: Number of bins transported by the mobile robot ”i”
- B_{T_j}: Number of bins transported by operator O_{T_1}, ... O_{T_N}
- B_{F_j}: Number of bins transported by operator O_F
- B_{E_j}: Number of bins transported by operator O_E

The state of the system (workstations and mobile robots) is represented by the following discrete linear state space model. The sampling period (Ts) is implicit and provide by a synchronised clock. The sampling period (on the order of one second) is much shorter than the overall dynamics of the system (several minutes).

$$\begin{cases} X_{n+1} = F.X_n + G_{R_1}.U_{R_1} + ... + G_{R_R}.U_{R_R} + G_O.U_O \\ Y_n = C.X_n \end{cases} \tag{7}$$

With:

- F is an identity matrix $[2*N+2]$.
- X_n is the state vector $[2*N+2)]$.
- G_{R_R} is the vector of transport capacities of each robot $[2*N+2]$.
- U_{R_R} is the vector of actions to be carried out by the robots $[2*N+2)]$.
- G_O is the vector of transport capacities of the operators $[2*N+2]$.
- U_O is the vector of actions to be carried out by the operators $[2*N+2]$.
- Y_n is the output vector $[2*N+2]$.
- C is an identity matrix $[2*N+2]$.

$$\underbrace{\begin{bmatrix} N_{OF_{1_{n+1}}} \\ \vdots \\ N_{OF_{N_{n+1}}} \\ N_{OE_{1_{n+1}}} \\ \vdots \\ N_{OE_{N_{n+1}}} \\ N_{F_{n+1}} \\ N_{E_{n+1}} \end{bmatrix}}_{X_{n+1}} = \underbrace{\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}}_{F}^{\top} \cdot \underbrace{\begin{bmatrix} N_{OF_{1_n}} \\ \vdots \\ N_{OF_{N_n}} \\ N_{OE_{1_n}} \\ \vdots \\ N_{OE_{N_n}} \\ N_{F_n} \\ N_{E_n} \end{bmatrix}}_{X_n} + \underbrace{\begin{bmatrix} B_{R_1} \\ -B_{R_1} \\ \vdots \\ B_{R_1} \\ -B_{R_1} \\ -B_{R_1} \\ B_{R_1} \end{bmatrix}}_{G_{R_1}}^{\top} \cdot \underbrace{\begin{bmatrix} GoO_{F_{1_1}} \\ GoO_{E_{1_1}} \\ \vdots \\ GoO_{F_{N_1}} \\ GoO_{E_{N_1}} \\ GoS_{F_1} \\ GoS_{E_1} \end{bmatrix}}_{U_{R_1}} + ...$$

$$+ \underbrace{\begin{bmatrix} B_{R_R} \\ -B_{R_R} \\ \vdots \\ B_{R_R} \\ -B_{R_R} \\ -B_{R_R} \\ B_{R_R} \end{bmatrix}}_{G_{R_R}}^{\top} \cdot \underbrace{\begin{bmatrix} GoO_{F_{1_R}} \\ GoO_{E_{1_R}} \\ \vdots \\ GoO_{F_{N_{RT}}} \\ GoO_{E_{N_R}} \\ GoS_{F_R} \\ GoS_{E_R} \end{bmatrix}}_{U_{R_R}} + \underbrace{\begin{bmatrix} -B_{T_1} \\ B_{T_1} \\ \vdots \\ -B_{T_N} \\ B_{T_N} \\ B_F \\ -B_E \end{bmatrix}}_{G_O}^{\top} \cdot \underbrace{\begin{bmatrix} O_{T_1} \\ O_{T_1} \\ \vdots \\ O_{T_N} \\ O_{T_N} \\ O_F \\ O_E \end{bmatrix}}_{U_O} \tag{8}$$

$$\underbrace{\begin{bmatrix} N_{OF_{1_n}} \\ \vdots \\ N_{OF_{N_n}} \\ N_{OE_{1_n}} \\ \vdots \\ N_{OE_{N_n}} \\ N_{F_n} \\ N_{E_n} \end{bmatrix}}_{Y_n} = \underbrace{\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}}_{C}^{\top} \cdot \underbrace{\begin{bmatrix} N_{OF_{1_n}} \\ \vdots \\ N_{OF_{N_n}} \\ N_{OE_{1_n}} \\ \vdots \\ N_{OE_{N_n}} \\ N_{F_n} \\ N_{E_n} \end{bmatrix}}_{X_n} \tag{9}$$

III. CONTRIBUTION

A. *MPC contribution*

Model Predictive Control (MPC) optimizes future control actions by solving a constrained optimization problem at each step, ensuring both system stability and performance while adhering to input and system constraints [13] [14]. It continuously updates predictions to maintain feasibility, stability, and optimal performance.

B. *Formulation of the Problem*

Consider a set of robots $R = \{r_1, \dots r_n\}$ and a set of tasks $T = \{t_1, \dots t_m\}$. Each robot r_i has attributes including position p_i, battery state of charge (SoC) b_i, speed s_i, and payload capacity c_i.

1) **Initialisation:**
   - Define the state space model of the system.
   - Set the number of robots n and the number of tasks m.
   - Set the inputs of the state space model.
2) **MPC process**
   - **Update constraints MPC**
     - Get constraints and measured disturbances of the system.
   - **MPC calculation**
     - From inputs data (state space model of the system, number of robots, number of tasks, inputs of state model, constraints), calculate task list to be assigned to the robots.
   - **Task Assessment:**
     - For each robot r_i, obtain its position p_i, battery state of charge (SoC) b_i, speed s_i, and payload capacity c_i.
     - Each robot assesses its capability $\kappa_i(t_j)$ for each task t_j, based on d_i (distance from position p_i to task t_j), b_i, s_i, and c_i:

$$\kappa_i(t_j) = \alpha_i.\frac{d_i}{d_{i_{ref}}} * \beta_i.\frac{b_i}{b_{i_{ref}}} * \gamma_i.\frac{s_i}{s_{i_{ref}}} * \delta_i.\frac{c_i}{c_{i_{ref}}} \tag{10}$$

     with α_i, β_i, γ_i, δ_i weighting coefficients and $d_{i_{ref}}$, $b_{i_{ref}}$, $s_{i_{ref}}$, $c_{i_{ref}}$ reference values for each robot r_i and each task t_j. It should be noted that a configuration of the weighting coefficients is necessary to determine the extent of their influence on the final decision-making.
   - **Task Selection:**

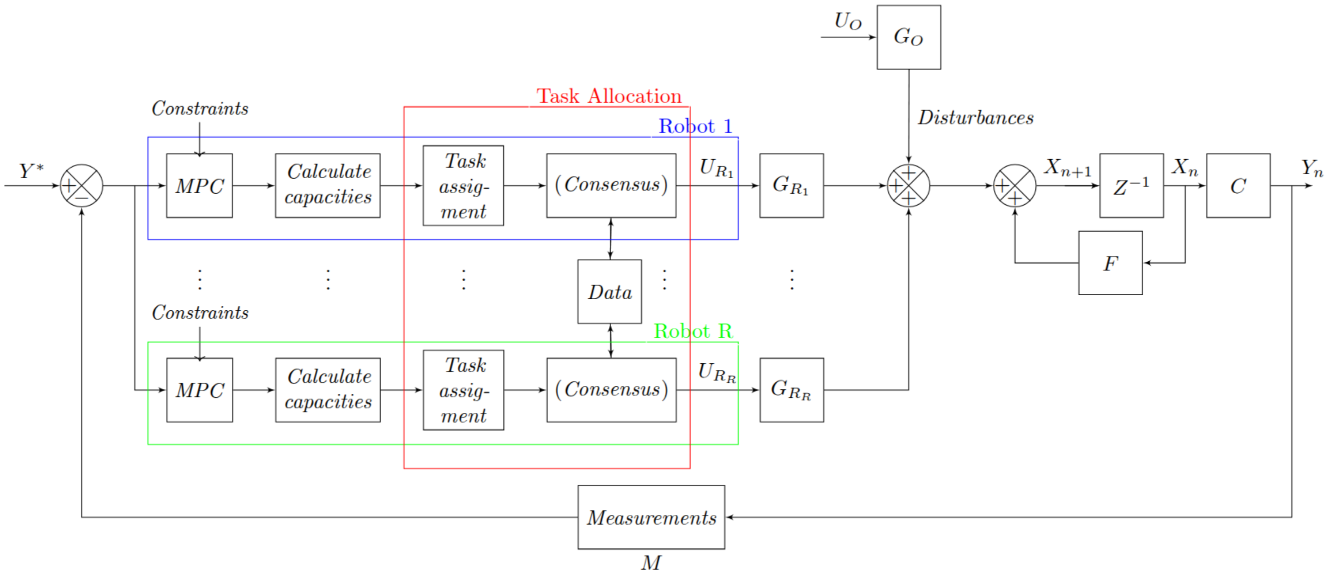

Fig. 3. System state representation

- Each robot r_i selects the task t_j for which it has the greatest capability, i.e., $t_i^* = \underset{t_j \in T}{\operatorname{argmax}} \kappa_i(t_j)$.

- **Decision Making:**
  a) **Communication:**
    - Each robot r_i communicates its selected task t_i^* to other robots, via a shared Wi-Fi network database.
    - Each robot reads all task allocations, via a shared Wi-Fi network database.

  b) **Check Conflicts:**
    - Check if two or more robots are assigned to the same task.
    - If yes, find consensus:
      * Robots communicate their capabilities $\kappa_i(t_j)$ for the contested task, via a shared Wi-Fi network database.
      * The robot with the highest $\kappa_i(t_j)$ is assigned the task.
      * In case of equal capabilities, the robot that communicated first is assigned the task (the date of writing in the database acting as referee).
    - If no conflict, continue.
- **Execution:**
  - Each robot performs its allocated task t_i^*.

The overall structure of the program is presented in algorithm 1.

C. Experimental Results

In order to test our proposed MPC-based allocation algorithm, we chose to use the *do-mpc Python* toolbox [15] along with three ROS-based mobile robots *Turtlebot3 Burger standard Raspberry PI* [16], equipped with *Robotis* actuator *Dynamixel* for driving . All programs were run on *ROS Noetic*, installed on *Ubuntu 20.04*. The movement of the robots is thus carried out by SLAM navigation, which guarantees that obstacles in the navigation paths are taken into account.

1) Experimental Setup: We replicated the industrial application case in the laboratory (figure 4) using a model built in a room (L: 2.85m x W: 2.29m), equipped with models of the different

Fig. 4. Experimental Area (laboratory)

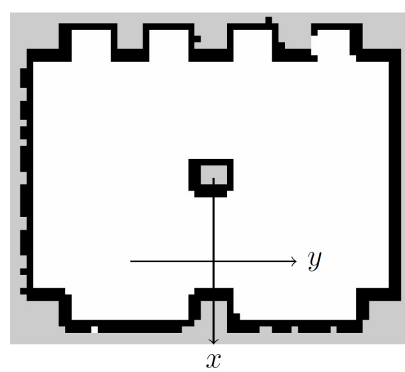

Fig. 5. Experimental Area (mapping)

workstations (L 310mm x W 220mm x H 250mm) and 3 mobile robots (L 138mm x W 178mm x H 192mm). And we mapped the experimental area thus created (figure 5).

2) Experimental Procedure: The experimental scenario involved three robots tasked with replenishing three distinct workstations. Operator requests were simulated using a Python script that generated random supply demands. The experiment followed these steps:

1) **Initialization:** Robots started from random positions with

Algorithm 1 Robot Task Allocation Process

INITIALISATION:
Define state model of the system
Set number of robots
Set number of tasks
Set inputs of state model
MPC LOOP:
for each sampling period **do**
  UPDATE CONSTRAINTS:
  for each constraint on system **do**
    Get measured disturbances
  end for
  MPC CALCULATION:
  Inputs = (State model of the system, Number of robots, Number of tasks, Inputs of state model, Constraints)
  Result = Task list to be assigned to the robots
  CALCULATE CAPACITIES:
  Get robot's battery level
  Get robot's position
  Set maxspeed of the robot
  Set payload of the robot
  for each task in task list **do**
    Evaluate capacity of the robot to perform task
  end for
  TASK ALLOCATION:
  Task assignment:
    Assign a task to the robot
    Write assigned task in database
  Check conflicts:
    Read assignments of other robots from database
    Check conflicts
  if Conflicts between robots **then**
      Find consensus
      Write new assigned task to database
  end if
  EXECUTE ACTIONS:
  Check assigned tasks
  if Task assigned to the robot **then**
    Reset the constraint related to the task in database
    EXECUTE ACTION
  end if
end for

varying battery levels.

2) **Task Allocation:** Each robot executed its MPC-based algorithm and evaluate task suitability based on its state (battery level, position, speed, payload).
3) **Consensus Mechanism:** In cases where multiple robots selected the same task, a conflict resolution process based on capability scores and time-stamped communication was triggered.
4) **Task Execution:** Once tasks were assigned, robots autonomously navigated to execute the required actions.
5) **Data Logging:** Each robot recorded its position, battery status, task assignment, and timestamps for analysis.

The task allocation process and navigation were visualized in real-time using *RViz* (Figure 6).

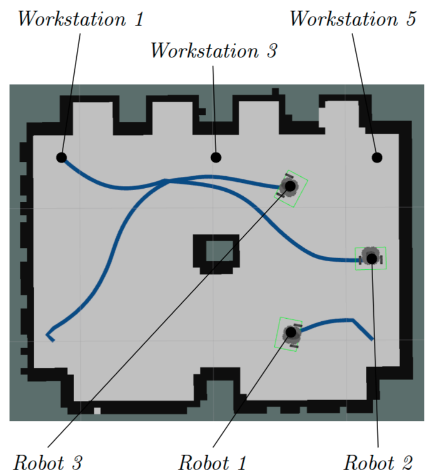

Fig. 6. *RViz* visualisation

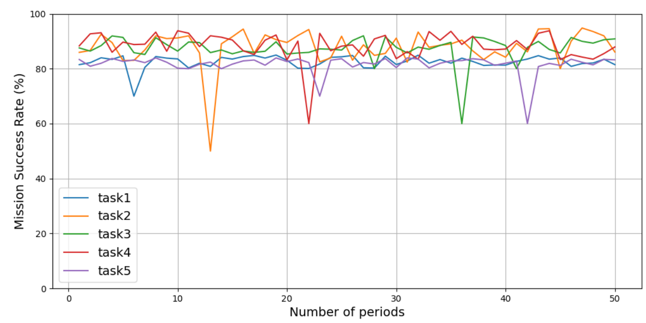

Fig. 7. Mission success rate

3) Performance Metrics: We assessed the system using the following performance indicators:

- **Mission Success Rate:** The percentage of tasks successfully completed without failures.
- **Response Time:** The time elapsed from task operator call to successful completion.

4) Results and Analysis: The mission success rate is depicted in Figure 7. The results indicate a high success rate, with all failures attributed to navigation errors—specifically SLAM-related issues within the ROS navigation function for the TurtleBot3 robot—rather than to the task allocation algorithm. This demonstrates the force of our MPC-based approach in managing task assignments effectively.

Figure 8 presents the distribution of response times for completed missions, that is, the time elapsed from task operator call to successful completion. While response times varied, this variability reflects differences in robot starting positions, battery levels, and real-time environmental conditions. Notably:

- Tasks assigned to robots with optimal proximity and battery levels exhibited shorter response times.
- The consensus mechanism effectively resolved task conflicts without introducing significant delays.

5) Conclusion of Experimental Evaluation: The experimental results validate the efficiency and reliability of the proposed MPC-based task allocation algorithm in decentralized multi-robot systems. The combination of constraint-aware optimization, real-time adaptability, and robust conflict resolution significantly enhances operational performance in industrial logistics scenarios.

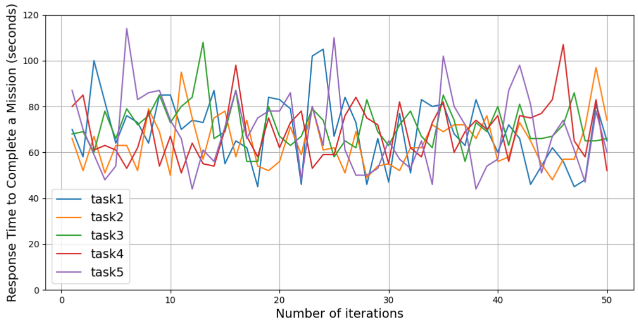

Fig. 8. Response time to complete a mission

D. Comparative analysis with our previous work

The bio-inspired approach from [17] relies on local decision-making and consensus for task allocation. Inspired by swarm intelligence, robots assign tasks based on battery level, distance, and priority. Conflict resolution is handled via a negotiation mechanism.

In contrast, this study applies Model Predictive Control (MPC), which anticipates future conditions and optimizes task assignment under constraints. Unlike the reactive bio-inspired method, MPC incorporates forecasting for better allocation.

The experimental evaluation focuses on four key performance indicators: the Task Allocation Efficiency (η_{alloc}), representing the percentage of successfully assigned tasks; the Response Time (T_{resp}), measuring the duration between task assignment and completion; the Mission Success Rate (σ_{succ}), indicating the proportion of completed tasks; and the Computation Overhead (C_{comp}), which quantifies the processing time required for task allocation. The comparison of these performance metrics is presented in Table I.

TABLE I

AVERAGE COMPARISON OF BIO-INSPIRED AND MPC-BASED ALLOCATION

Metric	Bio-Inspired	MPC-Based
η_{alloc}	89.5%	94.2%
T_{resp}	3.7 sec	2.9 sec
σ_{succ}	91.0%	96.3%
C_{comp}	Low	Moderate

The MPC-based approach enhances efficiency and reduces response time by leveraging its predictive capabilities. In contrast, the bio-inspired method is computationally lighter, making it more suitable for resource-constrained systems. While MPC achieves higher success rates in task allocation, it comes at the cost of increased processing requirements.

Bio-inspired allocation is fast and decentralized, ideal for low-power systems. MPC improves efficiency, suits dynamic environments, but demands higher computation.

IV. CONCLUSION AND PERSPECTIVE

Our research demonstrates that the proposed Model Predictive Control (MPC)-based algorithm effectively manages multi-robot task allocation under constraints while maintaining flexibility and robustness in industrial logistics systems. The experimental results show that the algorithm successfully assigns tasks to robots based on factors like battery levels, distance, and speed, ensuring an efficient and conflict-free allocation process. The built-in consensus mechanism allows for smooth conflict resolution, preventing disruptions even when multiple robots compete for the same task.

Although the approach has been validated using three robots in a laboratory setup, further evaluation is needed to assess scalability when increasing the number of robots, as communication delays and conflict resolution times may grow. Additionally, future research could explore how the algorithm performs in more dynamic or uncertain environments, such as those involving obstacle interference, real-time task generation, or communication loss, to better support claims of generalizability and operational resilience. Incorporating machine learning could also allow robots to learn from experience, further enhancing autonomous task assessment and allocation.

ACKNOWLEDGMENT

We would like to thank the company Safran Nacelles, located in Gonfreville L'Orcher France, for trusting us with the implementation of this project.

REFERENCES

[1] Y. Shang, "Design of an Integrated Approach to Industrial Logistics Information Based on Supply Chain Management," *Advances in Multimedia*, vol. 2022, pp. 1–8, May 2022.

[2] M. De Ryck, D. Pissoort, T. Holvoet, and E. Demeester, "Decentral task allocation for industrial AGV-systems with resource constraints," *Journal of Manufacturing Systems*, vol. 59, pp. 310–319, Apr. 2021.

[3] P. Mahato, S. Saha, C. Sarkar, and M. Shaghil, "Consensus-based fast and energy-efficient multi-robot task allocation," *Robotics and Autonomous Systems*, vol. 159, p. 104270, Jan. 2023.

[4] S. Teck, P. Vansteenwegen, and R. Dewil, "An efficient multi-agent approach to order picking and robot scheduling in a robotic mobile fulfillment system," *Simulation Modelling Practice and Theory*, vol. 127, p. 102789, Sept. 2023.

[5] N. Sharma, J. K. Pandey, and S. Mondal, "A Review of Mobile Robots: Applications and Future Prospect," *International Journal of Precision Engineering and Manufacturing*, vol. 24, pp. 1695–1706, Sept. 2023.

[6] K. Ryu and N. Mehr, "Integrating Predictive Motion Uncertainties with Distributionally Robust Risk-Aware Control for Safe Robot Navigation in Crowds," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, (Yokohama, Japan), pp. 2410–2417, IEEE, May 2024.

[7] O. M. de Groot, "Probabilistic Motion Planning in Dynamic Environments," *Journal of Engineering Probability*, 2024.

[8] M. V. Minniti, R. Grandia, K. Fah, F. Farshidian, and M. Hutter, "Model Predictive Robot-Environment Interaction Control for Mobile Manipulation Tasks," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, (Xi'an, China), pp. 1651–1657, IEEE, May 2021.

[9] I. Muhammed, A. A. Nada, and H. El-Hussieny, "Multi-Robot Object Transport in Constrained Environments: A Model Predictive Control Approach," in *2024 20th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA)*, (Genova, Italy), pp. 1–8, IEEE, Sept. 2024.

[10] X. Du, S. Zhou, and A. P. Schoellig, "Hierarchical Task Model Predictive Control for Sequential Mobile Manipulation Tasks," *IEEE Robotics and Automation Letters*, vol. 9, pp. 1270–1277, Feb. 2024.

[11] A. Belhomme and F. Guerin, "Task Allocation Algorithms for Mobile Robots, Application to Industrial Logistics," *Cahiers de la logistique*, pp. 85–96, Dec. 2024.

[12] A. Belhomme and F. Guerin, "Decentralised Bio-inspired Task Allocation for Mobile Robots, Application to Industrial Logistics," in *10th International Conference (IEEE-IFAC) on Control, Decision and Information Technologies (CoDIT 2024) July 01-04, 2024*, (Valetta (Malta)), 2024.

[13] E. F. Camacho and C. Bordons, *Model Predictive control*. Advanced Textbooks in Control and Signal Processing, London: Springer London, 2007.

[14] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*. Madison, Wisconsin: Nob Hill Publishing, 2nd edition ed., 2017.

[15] do mpc.com, "Model predictive control python toolbox — do-mpc 4.6.5 documentation," 2025.

[16] emanual.robotis.com, "ROBOTIS e-Manual," 2025.

[17] A. Belhomme, F. Guerin, N. Kerthe, and F. Germain, "Decentralised Bio-inspired Task Allocation for Mobile Robots, Application to Industrial Logistics," in *2024 10th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 928–933, July 2024. ISSN: 2576-3555.