2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Multi-robot Human-in-the-loop Control under Spatiotemporal Specifications

Yixiao Zhang, Victor Nan Fernandez-Ayala and Dimos V. Dimarogonas

***Abstract*— In this work, we present a coordination strategy tailored for scenarios involving multiple agents and tasks. We devise a range of tasks using signal temporal logic (STL), each earmarked for specific agents. These tasks are then imposed through control barrier function (CBF) constraints to ensure completion. To extend existing methodologies, our framework adeptly manages interactions among multiple agents. This extension is facilitated by leveraging nonlinear model predictive control (NMPC) to compute trajectories that avoid collisions. An integral aspect of our approach is the integration of a human-in-the-loop (HIL) model. This model enables real-time integration of human directives into the coordination process. A novel task allocation protocol is embedded within the framework to guide this process. We substantiate our methodology through a series of experiments, which corroborate the viability and relevance of our algorithms.**

I. Introduction

In recent years, the demand for robots to undertake intricate tasks has progressively risen. However, certain tasks, like collaborative planning and traffic management, surpass the capabilities of individual agents. These tasks necessitate the synergistic efforts of multiple agents working in tandem. Hence, the coordination of robots stands out as one of the most important challenges in multi-agent systems (MAS).

However, due to the limitation of time and space resources in the environment, collisions or deadlocks often arise in regions where agents interact frequently. Therefore, proper trajectory coordination is very important for enhancing the safety and efficiency of MAS. The solutions can be categorized into two types. The centralized approach is based on a coordination node to receive requests from agents and then plan the maneuver and trajectory for each agent [1], [2]. Distributed approaches are typically built upon a communication graph, where each agent only receives information from its neighbors for onboard real-time planning and control [3], [4].

Moreover, task definition and planning are pivotal features in MAS. Temporal logic plays a crucial role in defining complex tasks. Linear temporal logic (LTL) is primarily focused on capturing the characteristics of discrete-time sequences, specifying relationships between current and forthcoming states in a sequence of events [5]. Signal temporal logic (STL) serves as an extended version, boasting robust semantics capable of accommodating the attributes of continuous-time signals [6]. Furthermore, control barrier functions (CBFs) represent a relevant tool that satisfies security constraints during task fulfillment. They facilitate the design of control laws for dynamic systems that assure the system's operation within safe regions of the state space [7]. While the fusion of STL and CBF has found significant application in single-agent systems [8]–[11], several aspects in multi-agent scenarios are yet to be explored.

In addition, incorporating a human-in-the-loop (HIL) feature in the system introduces an extra layer of flexibility, enabling the harmonization of human judgment with automated planning. This symbiotic approach proves especially valuable in scenarios involving unforeseen circumstances, identification and rectification of undesirable robot behaviors, as well as collaborative decision-making processes. There exist several investigations on HIL. Authors in [12] designed a mixed-initiative control system to mix human non-speech, remote control input with robot input in the low-level control process. Furthermore, humans can act as the full operator of an agent or group of agents in mixed-robot environments. Autonomous agents need to recognize and adapt to the behavior of human-driven agents while completing their own tasks [13], [14]. Another type is the high-level HIL, particularly in scenarios of human-robot collaboration [15], [16]. Humans can issue real-time commands to autonomous agents. Agents must integrate these human-directed tasks with their existing objectives, prioritizing as necessary. Developing an effective method to coordinate and execute these integrated tasks is the primary aim of this study.

In this work, we first expand [8] to nonlinear control and MAS through a distributed coordination framework. Nonlinear model predictive control (NMPC) is used to obtain collision-free trajectories for completing different STL tasks. Moreover, a high-level HIL model is generated for the framework to apply online human commands to the coordination, with a novel task allocation protocol.

These algorithms are applied in the precision agriculture scenario through the EU CANOPIES project [17]. This is intertwined with grape collection in vineyards, wherein a network of intelligent agents assumes the role of aiding in harvesting and pruning activities. The experiments show the feasibility and applicability of our proposed framework.

II. Preliminaries

In this section, we first consider a single-agent system and the following system dynamics

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

This work was supported by the ERC CoG LEAFHOUND, the EU CANOPIES project, the Knut and Alice Wallenberg Foundation (KAW) and the Digital Futures Smart Construction project.

Yixiao Zhang, Victor Nan Fernandez-Ayala and Dimos V. Dimarogonas are with the Division of Decision and Control Systems, School of EECS, Royal Institute of Technology (KTH), 100 44 Stockholm, Sweden (Email: yixiaoz, vnfa, dimos@kth.se).

with agent state $\mathbf{x} \in \mathbb{X} = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| \leq d_x\}$ and control input $\mathbf{u} \in \mathbb{U} = \{\mathbf{u} \in \mathbb{R}^m : \|\mathbf{u}\| \leq d_u\}$. f, g functions are locally Lipschitz continuous. We assume that $d_x, d_u > 0$ and $g(\mathbf{x})g(\mathbf{x})^T$ is positive definite.

A. Robot Tasks

STL [6] is a formal language to define agent tasks. We can evaluate a continuously differentiable predicate function $h : \mathbb{R}^n \to \mathbb{R}$ to decide the boolean value of a predicate μ,

$$\mu = \begin{cases} \text{true}, & h(\mathbf{x}) \geq 0 \\ \text{false}, & h(\mathbf{x}) < 0 \end{cases} \tag{2}$$

STL formulas can then be recursively defined based on the STL syntax, i.e.,

$$\phi := \text{true} \,|\mu|\, \neg\phi \,|\phi_1 \wedge \phi_2|\, G_{[a,b]}\phi \,\Big|F_{[a,b]}\phi\Big|\, \phi_1 U_{[a,b]}\phi_2 \tag{3}$$

where ϕ, ϕ_1 and ϕ_2 are STL formulas, $0 \leq a \leq b$ is the time interval of the different STL operators including always G, eventually F, and until U. We denote by $(\mathbf{x}, t) \models \phi$ that the signal $\mathbf{x} : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ satisfies ϕ at time t. The STL semantics can be defined as Table I.

In this work, we consider multiple tasks for each agent, which are assumed to belong to the following STL fragment

$$\varphi_k := G_{[a_k,b_k]}\phi \mid F_{[a_k,b_k]}\phi \tag{4}$$

$$\varphi := \bigwedge_{k=1}^{n_\varphi} \varphi_k \tag{5}$$

where $\phi := \text{true}|\mu|\neg\mu$, n_φ is the number of G, F tasks and $k \in \{1, 2, ..., n_\varphi\}$ is the task index.

Remark 1. *Eq. (5) is extendable to some bigger fragments consisting of certain G, F, U tasks, since the satisfaction of $G_{[a,b]}\phi_1 \wedge F_{[b,b]}\phi_2$ implies the satisfaction of $\phi_1 U_{[a,b]}\phi_2$ [8]. In addition, disjunctions can also be included [18], however, we did not consider them in this work.*

B. CBF constraints

CBFs define conditions that the system needs to satisfy to ensure safety during task execution. A linear-time function $\gamma_k(t)$ can be used to design a CBF [8] so that its positivity for $t \geq 0$ implies satisfaction of $(\mathbf{x}, 0) \models \varphi_k$, i.e.,

$$\mathfrak{b}_k(\mathbf{x}, t) = -\gamma_k(t) + h_k(\mathbf{x}) \tag{6}$$

For the combination of multiple tasks in (5), an overall CBF can be created by the conjunction of n_φ candidate CBFs using an under-approximation of the min-operator [9]:

$$\begin{aligned} \mathfrak{b}(\mathbf{x}, t) &:= \min_{k \in \{1,\ldots,n_\varphi\}} \mathfrak{b}_k(\mathbf{x}, t) \\ &\approx -\ln\left(\sum_{k=1}^{n_\varphi} \mathfrak{o}_k(t) \exp\left(-\mathfrak{b}_k(\mathbf{x}, t)\right)\right) \end{aligned} \tag{7}$$

where $\mathfrak{o}_k(t) : \mathbb{R}_{\geq 0} \to \{0, 1\}$ is a switch function [9], i.e.,

$$\mathfrak{o}_k(t) = \begin{cases} 0 & t > b_k \text{ and } b_k < \max\{b_1, b_2, ..., b_{n_\varphi}\} \\ 1 & \text{otherwise} \end{cases} \tag{8}$$

Thus, we can switch off the candidate CBF when its temporal operator is satisfied. Then, we denote the switching sequence $\{s_0 := 0, s_1, \ldots, s_{n_{\text{sw}}}\}$ where $n_{\text{sw}} \in \{0, 1, ..., n_\varphi - 1\}$ is the total number of the switches. For $t > s_j$, we have

$$s_{j+1} := \text{argmin}_{b_k \in \{b_1,\ldots,b_{n_\varphi}\}} \zeta(b_k, t) \tag{9}$$

$$\zeta(b_k, t) := \begin{cases} b_k - t & \text{if } \; b_k - t > 0 \\ \infty & \text{otherwise} \end{cases} \tag{10}$$

Because each function $\mathfrak{b}_k(\mathbf{x}, t)$ is continuously differentiable, $\mathfrak{b}(\mathbf{x}, t)$ is continuously differentiable on $\mathbb{R}^n \times (s_j, s_{j+1})$. To guarantee safety and task completion, we need to ensure that the agent lies in the interior of $\mathcal{C}(t) = \{\mathbf{x} \in \mathbb{X} \mid \mathfrak{b}(\mathbf{x}, t) \geq 0\}$ [13].

C. NMPC Approach

Considering an NMPC problem analogous to [8] for the real-time agent control, we assume a control interval Δt, the length of the predicted horizon N_p, and the current time t_c. The planning state and control input at t_c can be represented as $\tilde{\mathbf{x}}(t|t_c)$ and $\tilde{\mathbf{u}}(t|t_c)$, $t \in [t_c, t_c + N_p\Delta t]$. Then, the NMPC problem can be represented as

$$\mathscr{P}_1 : \min_{\tilde{\mathbf{u}}, \epsilon} \int_{t_c}^{t_c + N_p\Delta t} J(\tilde{\mathbf{x}}, \tilde{\mathbf{u}}, \epsilon)\mathrm{d}t \tag{11}$$

$$\text{s.t. } \dot{\tilde{\mathbf{x}}}(t|t_c) = f\left(\tilde{\mathbf{x}}(t|t_c)\right) + g\left(\tilde{\mathbf{x}}(t|t_c)\right)\tilde{\mathbf{u}}(t|t_c) \tag{12}$$

$$\mathfrak{b}\left(\tilde{\mathbf{x}}(t \mid t_c), t\right) \geq -\epsilon, \quad \epsilon \in [0, \infty) \tag{13}$$

$$\tilde{\mathbf{x}}\left(t_c + N_p\Delta t \mid t_c\right) \in \mathcal{C}_F\left(t_c + N_p\Delta t\right) \tag{14}$$

$$\tilde{\mathbf{x}}\left(t_c \mid t_c\right) = \mathbf{x}(t_c) \tag{15}$$

$$\tilde{\mathbf{x}}\left(t \mid t_c\right) \in \mathbb{X} \tag{16}$$

$$\tilde{\mathbf{u}}\left(t \mid t_c\right) \in \mathbb{U} \tag{17}$$

(11) shows the differentiable objective function J. Equations (12) and (13) are the system dynamics and the CBF constraint with a relaxation factor ϵ respectively. (14) is designed for the terminal condition of the state, by generating a *terminal barrier function* $\mathfrak{b}_F(\mathbf{x}, t)$ with $\varphi_F := \varphi \wedge \varphi'$ in order to obtain $\mathcal{C}_F(t) := \{\mathbf{x} \in \mathbb{X} \mid \mathfrak{b}_F(\mathbf{x}, t) \geq 0\}$ [8], where $\varphi' := G_{[0, \max_k\{b_k\} + N_p\Delta t]}(d_x - \|\mathbf{x}\| \geq 0)$. (15) sets the initial state. (16) and (17) are the limitations of the state and control.

Lemma 1. *Assume that any control input $\tilde{\mathbf{u}}$ satisfying constraints (12)-(17) is continuous in $[t_c, t_c + N_p\Delta t]$. If $\mathscr{P}_1$ is feasible at $t_c = 0$, then $\mathscr{P}_1$ is recursively feasible.*

Proof. See related parts in [8, Theorem 2]. □

Lemma 2. *Consider the system dynamics (1) subject to $\mathbf{u} \in \mathbb{U}$, the STL formula φ_F and an extended class $\mathcal{K}$ function α_F. Let [9, Assumptions 1,3] hold. Define the control law*

$$\overline{\mathbf{u}} := \begin{cases} -g(\mathbf{x})^T\left(g(\mathbf{x})g(\mathbf{x})^T\right)^{-1} f(\mathbf{x}), \\ \qquad \text{if } \frac{\partial \mathfrak{b}_F(\mathbf{x}, t')}{\partial \mathbf{x}} = \mathbf{0}_n, \text{ for } t' \in [s_j, t] \\ \overline{\mathbf{u}} \quad \text{from} \quad \mathscr{P}_2, \quad \text{otherwise} \end{cases} \tag{18}$$

where $\mathscr{P}_2$ is given by $\overline{\mathbf{u}} = \text{argmin}_{\mathbf{u} \in \mathbb{U}} \, \mathbf{u}^T\mathbf{u}$, subject to:

$$\frac{\partial \mathfrak{b}_F(\mathbf{x}, t)}{\partial \mathbf{x}}(f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}) + \frac{\partial \mathfrak{b}_F(\mathbf{x}, t)}{\partial t} \geq -\alpha_F\left(\mathfrak{b}_F(\mathbf{x}, t)\right)$$

TABLE I: STL Semantics [9]

$(\mathbf{x}, t) \models \phi$	$h(\mathbf{x}(t)) \geq 0$
$(\mathbf{x}, t) \models \neg\phi$	$\neg((\mathbf{x}, t) \models \phi)$
$(\mathbf{x}, t) \models \phi_1 \wedge \phi_2$	$((\mathbf{x}, t) \models \phi_1) \wedge ((\mathbf{x}, t) \models \phi_2)$
$(\mathbf{x}, t) \models G_{[a,b]}\phi$	$\forall t_1 \in [t+a, t+b], (\mathbf{x}, t_1) \models \phi$
$(\mathbf{x}, t) \models F_{[a,b]}\phi$	$\exists t_1 \in [t+a, t+b], \text{s.t.}\, (\mathbf{x}, t_1) \models \phi$
$(\mathbf{x}, t) \models \phi_1 U_{[a,b]} \phi_2$	$\exists t_1 \in [t+a, t+b]$ s.t. $(\mathbf{x}, t_1) \models \phi_2$ $\wedge \forall t_2 \in [t, t_1], (\mathbf{x}, t_2) \models \phi_1$

for each $t \in [s_j, s_{j+1})$, $j = 0, 1, ..., n_{\text{sw}}$. *Then,* $\mathbf{x}(t), t \in [0, \max_k\{b_k\} + N_p\Delta t]$ *satisfying (1) for* $\overline{\mathbf{u}}$*, a.e. guarantees* $(\mathbf{x}, 0) \models \varphi_F$ *provided that* $\mathbf{x}(0) \in \mathcal{C}_F(0)$.

Proof. See in [9, Theorem 2] and [8, Theorem 1]. □

III. Multi-agent Coordination

In this section, N agents ($N > 1$) are existing in the scenario. For each agent $i, i \in \mathcal{I} = \{1, 2, ..., N\}$, its dynamics are $\dot{\mathbf{x}}_i = f(\mathbf{x}_i) + g(\mathbf{x}_i)\mathbf{u}_i$. In general, we consider $\mathbf{x}_i = [\mathbf{p}_i, ...]^T$, containing the Cartesian position $\mathbf{p}_i$ of the robot and other agent states such as heading, velocity and so on, depending on the type of robot involved.

For each agent i, depending on the properties of different tasks, STL formulas $\varphi_k^i, k = 1, 2, ..., n_\varphi^i$ can be reorganized into two sets, which can be described as

$$\Phi_H^i = \{\tilde{\varphi}_1^i, \tilde{\varphi}_2^i, ..., \tilde{\varphi}_{n_H^i}^i\} \tag{19}$$

$$\Phi_S^i = \{\hat{\varphi}_1^i, \hat{\varphi}_2^i, ..., \hat{\varphi}_{n_S^i}^i\} \tag{20}$$

Φ_H^i refers to hard tasks, where their specifications should strictly be satisfied for safety, like collision avoidance tasks. Φ_S^i refers to soft tasks, whose satisfaction is less stringent and can be violated or delayed in some situations. n_H^i and n_S^i are the cardinalities of the two sets, with $n_H^i + n_S^i = n_\varphi^i$. Thus, Eq. (5) can be rewritten as

$$\varphi^i = \bigwedge_{k=1}^{n_\varphi^i} \varphi_k^i = \left(\bigwedge_{k=1}^{n_H^i} \tilde{\varphi}_k^i\right)\left(\bigwedge_{k=1}^{n_S^i} \hat{\varphi}_k^i\right) \tag{21}$$

A. Agent avoidance based on priority

A priority mechanism is utilized before planning, and is particularly decided by the following ranking function M_i

$$M_i = \underbrace{w_d \left\|\mathbf{p}_i(t_c) - p_{k_c}^i\right\|^2}_{\text{Task Efficiency}} + \underbrace{w_s \int_0^{t_c} \left\|\frac{\partial^2 \mathbf{x}_i(t)}{\partial t^2}\right\|^2 \mathrm{d}t}_{\text{Path Smoothness}} \tag{22}$$

The first part is for task efficiency. Here we consider position-based tasks $\hat{\varphi}_k^i$ in Φ_S^i, where $h_k^i(\mathbf{x}) = d_k^i - \left\|\mathbf{p}_i - p_k^i\right\|$ with a constant d_k^i and target position p_k^i. Thus, the efficiency can be described by the distance between the current robot position $\mathbf{p}_i(t_c)$ and $p_{k_c}^i$ of the first incoming task $\{\hat{\varphi}_{k_c}^i | b_{k_c} = s_{j+1}\}$. This part can also be adjusted to other types of tasks. The second part reflects the smoothness of the trajectory.

Definition 1. *A place symbol* $1 \leq \delta_i \leq N, \delta_i \in \mathbb{N}_+$ *can be assigned to each agent by ordering the* M_i *values from smallest to largest. Lower* δ_i *means higher priority. Thus, we can obtain an order list* $\mathcal{L}$ *to save the priorities, i.e.,*

$$\mathcal{L} = \{(i, M_i, \delta_i) \mid i = 1, 2, ..., N\} \tag{23}$$

New hard tasks need to be created for agent i to avoid collisions with higher priority agents $\{j \mid \delta_j < \delta_i\}$ according to Definition 1. Agent i can obtain $\mathbf{p}_j$ through the communication with agent j. A safe tunnel [2] can be used to describe the occupied area of each agent j based on its estimated positions $\mathbf{p}_j(t)$, $t \in [t_c, t_c + N_p\Delta t]$. As shown in Fig. 1, the tunnel is a three-dimensional region, generated by joining continuous circular regions with a safe radius d_{safe} around $\mathbf{p}_j$ at each future time $t \in [t_c, t_c + N_p\Delta t]$. Agent i creates a G-task to stay away from the tunnel at each time step inside the predicted horizon, i.e.,

$$\tilde{\varphi}_{k,j}^i := G_{[t_c+(k-1)\Delta t,\ t_c+k\Delta t]} \left(\left\|\mathbf{p}_i - \mathbf{p}_j\left(t_c + (k-1)\Delta t\right)\right\| - d_{\text{safe}} \geq 0\right) \tag{24}$$

Then, Eq. (19) is updated by adding the new tasks for all higher-priority agents, which can represented as

$$\Phi_H^i \leftarrow \Phi_H^i \cup \left\{\tilde{\varphi}_{k,j}^i \;\middle|\; \begin{matrix} \delta_j < \delta_i,\ j \in \mathcal{I} \\ k = 1, 2, ..., N_p \end{matrix}\right\} \tag{25}$$

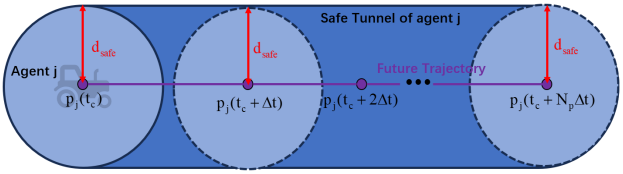


Fig. 1: Safe tunnel used to avoid collisions with agent j.

B. NMPC Optimization

After updating agent priority and Φ_H^i in Definition 1 and (25), the total CBFs $\mathfrak{b}_H^i(\mathbf{x}_i, t)$, $\mathfrak{b}_S^i(\mathbf{x}_i, t)$ can be created through (7) for the task sets Φ_H^i, Φ_S^i, respectively. The CBF constraints corresponding to (13) are

$$\mathfrak{b}_H^i(\tilde{\mathbf{x}}_i, t) \geq -\epsilon_H^i, \quad \epsilon_H^i \in [0, \infty) \tag{26}$$

$$\mathfrak{b}_S^i(\tilde{\mathbf{x}}_i, t) \geq -\epsilon_S^i, \quad \epsilon_S^i \in [0, \infty) \tag{27}$$

We design an objective function J_i with the form as

$$J_i = \underbrace{w_{u_1}\left\|\tilde{\mathbf{u}}_i(t \mid t_c)\right\|^2}_{\text{Control Magnitude}} + \underbrace{w_{u_2}\left\|\frac{\partial \tilde{\mathbf{u}}_i(t \mid t_c)}{\partial t}\right\|^2}_{\text{Control Smoothness}} + \underbrace{w_{\epsilon_H}\|\epsilon_H^i\|^2 + w_{\epsilon_S}\|\epsilon_S^i\|^2}_{\text{CBF satisfaction}} \tag{28}$$

with weights $w_{u_1}, w_{u_2}, w_{\epsilon_H}, w_{\epsilon_S}$ and $w_{\epsilon_H} >> w_{\epsilon_S}$. A finite difference method [19] can be used for discretization. The NMPC problem for each agent i is represented as

$$\mathscr{P}_3^i : \min_{\tilde{\mathbf{u}}_i, \epsilon_H^i, \epsilon_S^i} \int_{t_c}^{t_c+N_p\Delta t} J_i \quad \mathrm{d}t \tag{29}$$

$$\text{s.t. } (12), (15) - (17), (26), (27)$$

$$\tilde{\mathbf{x}}_i(t_c + N_p\Delta t \mid t_c) \in \mathcal{C}_F^i(t_c + N_p\Delta t) \tag{30}$$

with $t \in [t_c, t_c + N_p\Delta t]$. $\mathcal{C}_F^i(t) = \{\mathbf{x}_i \in \mathbb{X} | \mathfrak{b}_F^i(\mathbf{x}_i, t) \geq 0\}$ with $\varphi_F^i := \varphi^i \wedge \varphi'$. For the constraints, $\tilde{\mathbf{u}}, \tilde{\mathbf{x}}, \mathbf{x}$ is replaced with $\tilde{\mathbf{u}}_i, \tilde{\mathbf{x}}_i, \mathbf{x}_i$ in (12), (15) - (17). $\mathscr{P}_3^i$ can be solved by many off-the-shelf solvers, including IPOPT and SQP.

Assumption 1. *Assume that for each agent $i, i \in \mathcal{I}$, the tasks and task times in Φ_H^i and Φ_S^i as well as those in agents $\{j | \delta_j < \delta_i\}$ are allocated in a manner that ensures there exists a feasible solution for $\mathscr{P}_3^i$.*

Remark 2. *We also assume that the communication and computing delays are negligible.*

Theorem 1. *For each agent $i, i \in \mathcal{I}$, consider the system (1) and the STL formulas Φ_H^i, Φ_S^i. Let the assumptions of Lemmas 1-2 and Assumption 1 hold. Assume that any control input $\tilde{\mathbf{u}}_i$ satisfying all the constraints of (29) is continuous in $\left[t_c, t_c + N_p\Delta t\right]$, and the NMPC problem (29) is feasible at $t_c = 0$. Then, all $\mathscr{P}_3^i, i = 1, ..., N$ are recursively feasible.*

Proof. First, for agent $\{i, \delta_i = 1\}$, $\mathscr{P}_3^i$ is recursively feasible by Lemma 1. Then, for agent $\{i, \delta_i > 1\}$, we assume that $\{\mathscr{P}_3^j \mid \delta_j < \delta_i\}$ are recursively feasible and $\mathscr{P}_3^i$ is feasible at t_c. Through agent info sharing, Φ_H^i and φ_F^i are updated by (25). According to Assumption 1 and Lemma 2 applied to φ_F^i, there also exists an admissible control law $\overline{\mathbf{u}}_i(\mathbf{x}_i(t), t)$ that is continuous in $\mathbf{x}_i$ and t, since $\mathbf{x}_i(t)$ is absolutely time-continuous. We consider $\tilde{\mathbf{u}}_i\left(t | t_c + \Delta t\right) = \overline{\mathbf{u}}_i(t)$ when $t \in (t_c + N_p\Delta t, t_c + (N_p+1)\Delta t]$, and $\tilde{\mathbf{u}}_i\left(t | t_c + \Delta t\right) = \tilde{\mathbf{u}}_i\left(t | t_c\right)$ when $t \in [t_c + \Delta t, t_c + N_p\Delta t]$. Thus constraints (16), (17) and (30) are satisfied for $\tilde{\mathbf{u}}_i(t | t_c + \Delta t)$ with dynamics (1). For Φ_S^i and updated Φ_H^i, the violation factor ϵ_o^i can be used to achieve $\mathfrak{b}_o^i(\tilde{\mathbf{x}}_i\left(t | t_c + \Delta t\right), t) \geq \epsilon_o^i(t | t_c + \Delta t),\ o \in \{H, S\}$:

$$\epsilon_o^i(t | t_c + \Delta t) = \begin{cases} \epsilon_o^i(t | t_c), & t \in \left[t_c + \Delta t, t_c + N_p\Delta t\right] \\ \overline{\epsilon}_o^i(t), & t \in \left(t_c + N_p\Delta t, t_c + (N_p+1)\Delta t\right] \end{cases}$$

$$\overline{\epsilon}_o^i(t) = \begin{cases} 0, & \mathfrak{b}_o^i(\tilde{\mathbf{x}}_i, t) \geq 0 \\ -\mathfrak{b}_o^i(\tilde{\mathbf{x}}_i, t), & \mathfrak{b}_o^i(\tilde{\mathbf{x}}_i, t) < 0 \end{cases} \tag{31}$$

Then, according to the proof of [8, Theorem 2], we can similarly deduce that $\{\tilde{\mathbf{u}}_i(t | t_c + \Delta t), \epsilon_H^i(t | t_c + \Delta t), \epsilon_S^i(t | t_c + \Delta t) \mid t \in \left[t_c + \Delta t, t_c + (N_p + 1)\Delta t\right]\}$ also satisfies all the constraints in $\mathscr{P}_3^i$, which implies its recursive feasibility.

Therefore, NMPC problems $\mathscr{P}_3^i$ can be proven, in the increasing order of δ_i, to show the overall recursive feasibility of the proposed MAS system if the constraints hold. □

C. Distributed Implementation

For distributed implementation, we considered the undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that describes the communication links between the agents. Agent i can only share information with its neighbors $\mathcal{N}^i = \{j | (i, j) \in \mathcal{E}\}$. Each agent i will store an ordered list $\mathcal{L}^i$ locally, i.e.,

$$\mathcal{L}^i = \{(z, M_z^i, \delta_z^i) \mid z = 1, 2, ..., N\}, \quad i \in \mathcal{I} \tag{32}$$

The process of the distributed implementation is shown in Fig. 2. The first step is for exchanging M-values between neighbors, where agent i will update $M_j^i \leftarrow M_j^j, j \in \mathcal{N}^i$ and subsequently reorganize the priorities $\delta_z^i, z = 1, ..., N$ in $\mathcal{L}^i$ according to Definition 1. The second step is sharing the planned trajectories between neighbors, such that agent i can avoid higher-priority neighbors $\{j | j \in \mathcal{N}^i$ and $\delta_j^i < \delta_i^i\}$.

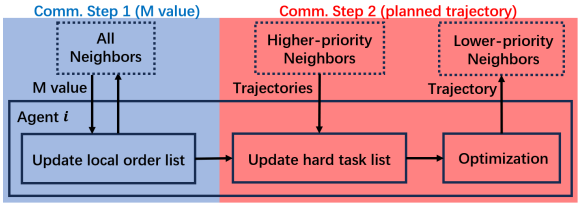


Fig. 2: Distributed Implementation.

IV. High-Level Human-in-the-Loop Control

In this section, we consider that agents are given real-time high-level human commands. A simple human command (HC) consists of three parts, including Task Command (TC), Task Priority Command (PC), and Subject Command (SC). For example, in the command "All agents follow me now", "follow me" is TC, "now" is PC, and "all agents" is SC.

For different TCs, we can create a command dictionary to map human tasks to different STL formulas, as shown in Table II. Similar to Eq. (19) and (20), human tasks at each transmission are classified into hard and soft sets Ψ_H, Ψ_S:

$$\Psi_H = \{\tilde{\psi}_1, \tilde{\psi}_2, ..., \tilde{\psi}_{n_H^{\mathrm{h}}}\} \tag{33}$$

$$\Psi_S = \{\hat{\psi}_1, \hat{\psi}_2, ..., \hat{\psi}_{n_S^{\mathrm{h}}}\} \tag{34}$$

with $\hat{\psi}_k, \tilde{\psi}_k := G_{[a_k^{\mathrm{h}}, b_k^{\mathrm{h}}]}\phi \mid F_{[a_k^{\mathrm{h}}, b_k^{\mathrm{h}}]}\phi$.

For soft tasks, we must delay the execution times of Φ_S^i, Ψ_S to get $\Phi_S'^i, \Psi_S'$, based on different PCs, which are given as follows: (i) **Now:** a_k, b_k of the unfinished original tasks $\{\hat{\varphi}_k^i | b_k > s_j, \hat{\varphi}_k^i \in \Phi_S^i\}$ should add Δt_ψ to get the new set $\Phi_S'^i$ with $\Delta t_\psi = \max\{b_k^{\mathrm{h}} | \hat{\psi}_k \in \Psi_S\}$. (ii) **Later:** $a_k^{\mathrm{h}}, b_k^{\mathrm{h}}$ of Ψ_S should add Δt_φ^i to obtain Ψ_S' with $\Delta t_\varphi^i = \max\{b_k | \hat{\varphi}_k^i \in \Phi_S^i\}$. (iii) **A specific location** (do HCs at time t_{h}): $\{a_k, b_k \mid b_k > t_{\mathrm{h}}\}$ should delay $t_{\mathrm{h}} + \Delta t_\psi$, then $a_k^{\mathrm{h}}, b_k^{\mathrm{h}}$ should delay t_{h}.

Hard tasks Φ_H^i and Ψ_H are usually unchangeable. But for the static obstacle avoidance tasks, their execution times need to be extended automatically for extra activity due to the increasing of total time, $b_k = b_k^{\mathrm{h}} \leftarrow \Delta t_\varphi^i + \Delta t_\psi$.

There are three types of SCs. The first two are "All agents" and "Agent $i \in \mathcal{N}_{\mathrm{h}}$" with subset $\mathcal{N}_{\mathrm{h}} \subset \mathcal{I}$, where human soft tasks are assigned to specific agents $\Phi_S''^i = \Phi_S'^i \cup \Psi_S'$ with $i \in \mathcal{I}$ and $i \in \mathcal{N}_{\mathrm{h}}$ respectively. The third one is "n_{h} agents" with $n_{\mathrm{h}} \in \mathcal{I}$, which only specifies the number of agents for HC. Task allocation for Ψ_S' needs to be carried out among agents. It is based on agent priority with a novel M_i function compared to equation (22), by adding two criteria with weights w_n, w_t, i.e.,

$$M_i \leftarrow M_i + \underbrace{w_n\left(\left|\Phi_S''^i\right| + \left|\Phi_H^i\right|\right)}_{\text{Task Fairness}} + \underbrace{w_t(\Delta t_\varphi^i + \Delta t_\psi)}_{\text{Total Time}} \tag{35}$$

The allocation process is shown in Algorithm 1. Human tasks are only allocated to n_{h} higher-priority agents in (36).

As for the human hard tasks, they must always be considered by all agents for safety $\Phi_H^i \leftarrow \Phi_H^i \cup \Psi_H, i \in \mathcal{I}$.

Task Commands	STL Formulas
Do **not** do task	$\neg\psi$
Do task 1 **and** task 2	$\psi_1 \wedge \psi_2$
Do task 1 **until** task 2 finished	$\psi_{\text{until},[a,b]} := \psi_1 U_{[a,b]} \psi_2 := G_{[a,b]}\psi_1 \wedge F_{[b,b]}\psi_2$
Go to (Goal)	$\hat{\psi}_{\text{go},[a,b]} := F_{[a,b]}(d_\text{r} - \|\mathbf{p} - p_\text{goal}\| \geq 0)$ with the central position p_goal and radius d_r of the goal region.
Stay in (Goal)	$\hat{\psi}_{\text{stay},[a,b]} := G_{[a,b]}(d_\text{r} - \|\mathbf{p} - p_\text{goal}\| \geq 0)$
Avoid collision with (Obstacle)	$\tilde{\psi}_{\text{ob},[a,b]} := G_{[a,b]}\left(\|\mathbf{p} - p_\text{ob}\| - d_\text{safe} \geq 0\right)$ with obstacle position p_ob and safe distance d_safe .
Follow me	$\psi_{\text{follow},[a,b]} := \hat{\psi}_{\text{go},[a,b]} \wedge \tilde{\psi}_{\text{ob},[t_c,b]} := F_{[a,b]}(d_\text{r} - \|\mathbf{p} - p_\text{h}(t_c)\| \geq 0) \wedge G_{[t_c,b]}\left(\|\mathbf{p} - p_\text{h}(t_c)\| - d_\text{safe} \geq 0\right)$ With the current human position $p_\text{h}(t_c)$, $d_\text{safe} \leq d_\text{r}$.
Pick (X)	$\hat{\psi}_\text{pick}$
Bring (X)	$\psi_{\text{bring},[a,b]} := F_{[a,b_1]}\hat{\psi}_\text{pick} \wedge \psi_{\text{follow},[a_2,b]}$ with $a \leq b_1 \leq a_2 \leq b$
...	...

TABLE II: Dictionary of human task command (TC).

Algorithm 1 Task allocation in distributed coordination

1: Initialization. Set the number of agents N, original task sets Φ_H^i, Φ_S^i, and ranking list $\mathcal{L}^i$ with $i \in \mathcal{I}$.
2: Human gives HCs to all agents.
3: **Parallel For** $i = 1$ **to** N **do in parallel**
4: **repeat**
5: Agent i receive HCs, create Ψ_H, Ψ_S based on TCs.
6: Change times in Φ_S^i, Ψ_S to get $\Phi_S'^i, \Psi_S'$ by their PC.
7: Add tasks $\Phi_H^i \leftarrow \Phi_H^i \cup \Psi_H$ and $\Phi_S''^i = \Phi_S'^i \cup \Psi_S'$.
8: Agent i calculate its M-value M_i^i based on (35).
9: Agent i exchange M-value with its neighbors $j \in \mathcal{N}^i$, then update M_j^i and $\delta_z^i, z = 1, 2, ..., N$ in $\mathcal{L}^i$.
10: Agent i receives $\mathbf{p}_j(t), t \in [t_c, t_c + N_p\Delta t]$ from higher priority neighbors and update Φ_H^i based on (25).
11: Remove human soft tasks for lower-priority agents,

$$\begin{array}{ll} \Phi_S''^i \leftarrow \Phi_S^i, & \delta_i^i > n_\text{h} \\ \Phi_S''^i \leftarrow \Phi_S'^i \cup \Psi_S', & \delta_i^i \leq n_\text{h} \end{array} \tag{36}$$

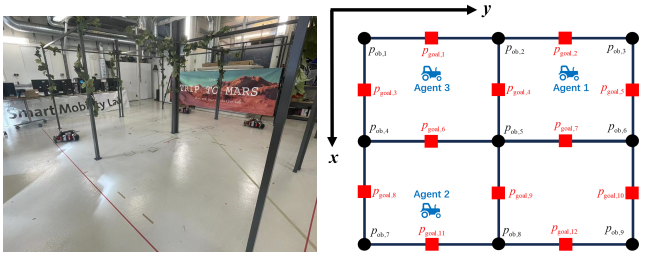
12: Construct constraints (26) and (27) with Φ_H^i, $\Phi_S''^i$.
13: Solve the optimization problem $\mathscr{P}_3^i$ for control.
14: **until** the coordination process is finished.
15: **End Parallel For**

V. Experiments

A testing scenario has been developed in connection with the EU CANOPIES project. This scenario involves configuring a laboratory setting to replicate a 3.8 m $\times$ 4.6 m artificial vineyard, which is depicted in Fig. 3a. In this setup, there are grapes positioned on the shelves that need to be harvested. The main objective of the robots is to navigate to specified locations for picking or placing, all while carefully avoiding collisions with the poles present in the environment.

Three HEBI Robotics Rosie robots [20] are used for experiments, operated by ROS-based Intel NUC computers each equipped with a two-core 4.1 GHz CPU and 16 GB of DDR4 RAM. The system dynamic of (1) is re-written as

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} \cos\theta_i & 0 \\ \sin\theta_i & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \tag{37}$$



(a) Artificial vineyard. (b) Robot task model.

Fig. 3: Laboratory settings for experiments.

where $i \in \{1, 2, 3\}$, $\mathbf{x}_i = [x_i, y_i, \theta_i]^T$ and $\mathbf{u}_i = [v_i, \omega_i]^T$, with robot position (x_i, y_i), yaw angle θ_i, forward velocity v_i and angular velocity around the z-axis ω_i. The QUALISYS system [21] tracks the positions of robots and humans in real-time. Task allocation and agent message sharing are managed through the ROS API via a wireless connection.

The graphic model of the robot tasks is also shown in Fig. 3b. There are two types of hard tasks. The first one is for obstacle avoidance with the poles, i.e.,

$$\tilde{\varphi}_k^i := G_{[0,T_\text{max}]}\left(||\mathbf{p}_i - p_{\text{ob},k}|| - 0.4 \geq 0\right), k = 1, ..., 9 \tag{38}$$

with pole positions $p_{\text{ob},k}$. The second is for agent avoidance which is added before each planning in section III-A.

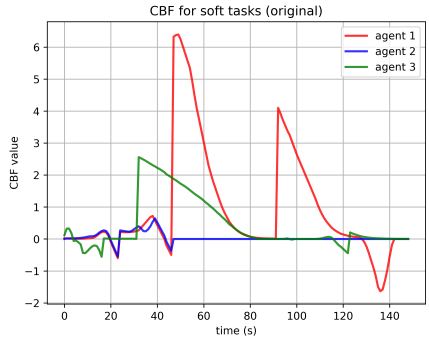
For the soft tasks, we consider the most common type of tasks, going to a shelf to pick and place, i.e.,

$$\hat{\varphi}_{[a,b]}^i(q) := F_{[a,b]}\left(0.2 - \left\|\mathbf{p}_i - p_{\text{goal},q}\right\| \geq 0\right) \tag{39}$$

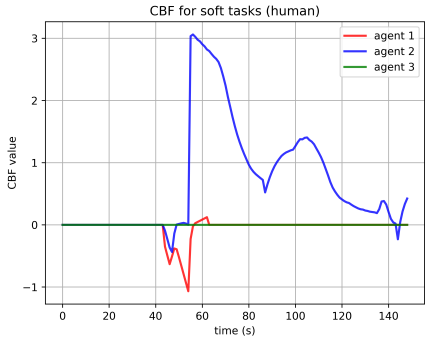
with the shelf positions $p_{\text{goal},q}$. Thus, we created soft lists,

$$\Phi_S^1 = \{\hat{\varphi}_{[15,30]}^1(11), \hat{\varphi}_{[45,60]}^1(2), \hat{\varphi}_{[75,90]}^1(11), \hat{\varphi}_{[105,120]}^1(2)\}$$
$$\Phi_S^2 = \{\hat{\varphi}_{[15,30]}^2(2), \hat{\varphi}_{[45,60]}^2(9)\}$$
$$\Phi_S^3 = \{\hat{\varphi}_{[15,30]}^3(8), \hat{\varphi}_{[45,60]}^3(1), \hat{\varphi}_{[90,120]}^3(12), \hat{\varphi}_{[120,150]}^3(1)\}$$

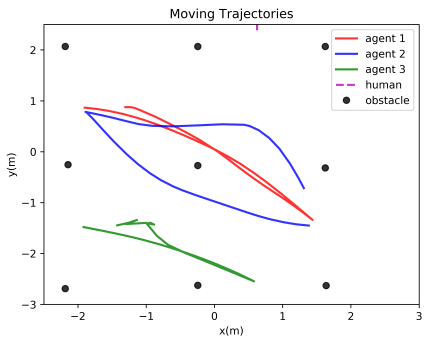
High-level HIL is also under consideration in the experiment. At around $t = 45$s, a human sequentially gives two human commands to the robots, "HC_1: Agent 1 and 2 go to my position now" and "HC_2: One agent follow me later". At the same time, there is also a hidden command that "HC_3: All agents avoid collision with the human". STL formulas for human commands can be created through Table II.



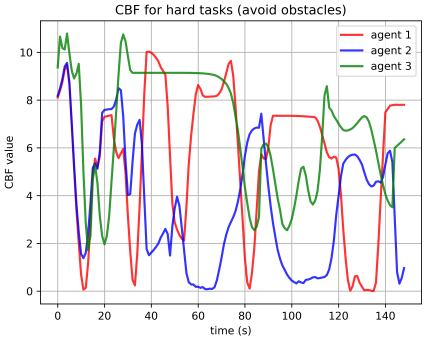
(a) CBF for original soft tasks.



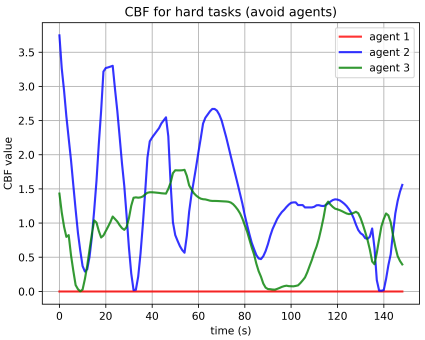
(b) CBF for human soft tasks.



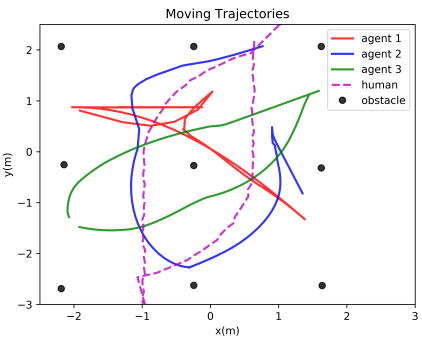
(c) Planned agent trajectories (0-45s).



(d) CBF for hard tasks (obstacle avoidance).



(e) CBF for hard tasks (agent avoidance).



(f) Planned agent trajectories (45-150s).

Fig. 4: Evolution of different types of CBFs, and moving trajectories of all agents and the human.

Each robot establishes CBFs with (6) for the received STL tasks. Then, in the NMPC optimization process, we set $\Delta t = 0.1$ s, $N_p = 30$, $w_{u_1}, w_{u_2}, w_{\epsilon_H}, w_{\epsilon_S} = 1, 10, 10^6, 10^3$ and $w_d, w_s, w_n, w_t = 100, 10, 100, 1$. Also, we give limitations to the control input $|v_i| \leq 0.5$ m/s and $|w_i| \leq 0.5$ rad/s.

The experimental results for various types of CBFs and pre-planned trajectories are depicted in Fig. 4. The motion process is demonstrated in a video available at the given link [22]. We can see that **(i)** the framework can obtain smooth feasible solutions to achieve all the STL tasks without any collisions between agents and humans. **(ii)** Fig. 4a and 4b display the CBFs for both the original and human soft tasks. Notably, it becomes evident that during certain time intervals, the soft tasks may be temporarily breached when CBF values enter the negative regions, particularly when characterized by a larger ϵ_S^i. But eventually, the CBFs returned to the non-negative regions, enabling the completion of the tasks as intended. Nonetheless, it is crucial to highlight that in Fig. 4d and 4e, the CBFs for hard tasks, which encompass obstacle avoidance and interaction with other agents, must consistently remain positive to ensure safety. This is achieved through the adoption of smaller and more stringent values for ϵ_H^i. **(iii)** Furthermore, there exists a priority hierarchy for agent collision avoidance, as shown in Fig 4e. Specifically, Agent 1 with the highest priority is exempt from the consideration, while Agent 2 is tasked with avoiding Agent 1, and Agent 3 is responsible for avoiding both 1 and 2. **(iv)** For the human tasks started from 45s, Fig. 4b and 4f show that HC_2 are allocated to agent 2 with the minimum task length and least total time, based on protocol (35).

Fig. 5 shows the cumulative distribution function (CDF) of the average computation time per planning step. The observation is that the distributed strategy typically necessitates less computation time than the centralized one since the computing is distributed among each agent.

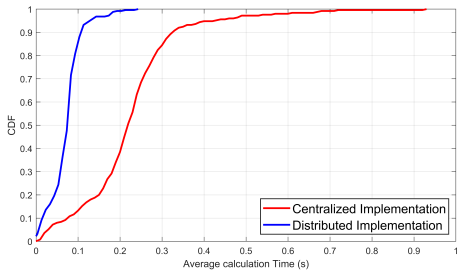


Fig. 5: Comparison on average computing time.

VI. Conclusion

This work addresses the coordination challenge in MAS for managing multiple tasks involving different robots. A distributed coordination framework is developed using CBF-based nonlinear MPC. This proposed framework efficiently produces smooth, collision-free trajectories to accomplish different STL tasks. In addition, the system is also able to manage high-level human commands by merging HIL into the coordination framework. A novel task allocation protocol is introduced to facilitate task distribution among agents. Practical cases involving grape collection within the EU CANOPIES project demonstrate the real-world applicability and viability of these algorithms.

REFERENCES

[1] P. Shi and B. Yan, "A survey on intelligent control for multiagent systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 161–175, 2021.

[2] Y. Zhang, R. Hao, T. Zhang, X. Chang, Z. Xie, and Q. Zhang, "A trajectory optimization-based intersection coordination framework for cooperative autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14 674–14 688, 2022.

[3] G. Wen, X. Yu, W. Yu, and J. Lu, "Coordination and control of complex network systems with switching topologies: A survey," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 10, pp. 6342–6357, 2021.

[4] T. Yang, Y. Zou, S. Li, and Y. Yang, "Distributed model predictive control for probabilistic signal temporal logic specifications," *IEEE Transactions on Automation Science and Engineering*, pp. 1–11, 2023.

[5] M. Kloetzer and C. Belta, "Automatic deployment of distributed teams of robots from temporal logic motion specifications," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 48–61, 2010.

[6] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*. Springer, 2004, pp. 152–166.

[7] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European control conference (ECC)*. IEEE, 2019, pp. 3420–3431.

[8] M. Charitidou and D. V. Dimarogonas, "Barrier function-based model predictive control under signal temporal logic specifications," in *2021 European Control Conference (ECC)*, 2021, pp. 734–739.

[9] L. Lindemann and D. V. Dimarogonas, "Control barrier functions for signal temporal logic tasks," *IEEE Control Systems Letters*, vol. 3, no. 1, pp. 96–101, 2019.

[10] A. Thirugnanam, J. Zeng, and K. Sreenath, "Safety-critical control and planning for obstacle avoidance between polytopes with control barrier functions," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 286–292.

[11] B. Xu and K. Sreenath, "Safe teleoperation of dynamic uavs through control barrier functions," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7848–7855.

[12] M. Guo, S. Andersson, and D. V. Dimarogonas, "Human-in-the-loop mixed-initiative control under temporal tasks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6395–6400.

[13] V. N. Fernandez-Ayala, X. Tan, and D. V. Dimarogonas, "Distributed barrier function-enabled human-in-the-loop control for multi-robot systems," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 7706–7712.

[14] J. Wang, Y. Zheng, Q. Xu, J. Wang, and K. Li, "Controllability analysis and optimal control of mixed traffic flow with human-driven and autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 12, pp. 7445–7459, 2021.

[15] S. Kumar, C. Savur, and F. Sahin, "Survey of human–robot collaboration in industrial settings: Awareness, intelligence, and compliance," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 280–297, 2021.

[16] G. Silano, A. Afifi, M. Saska, and A. Franchi, "A signal temporal logic planner for ergonomic human–robot collaboration," in *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2023, pp. 328–335.

[17] CANOPIES project, "A Collaborative Paradigm for Human Workers and Multi-Robot Teams in Precision Agriculture Systems," in *https://canopies.inf.uniroma3.it/*, european Commission under the H2020 Framework Programme. [Online]. Available: https://canopies.inf.uniroma3.it/.

[18] A. Wiltz and D. V. Dimarogonas, "Handling disjunctions in signal temporal logic based control through nonsmooth barrier functions," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 3237–3242.

[19] H. Fan, F. Zhu, C. Liu, L. Zhang, L. Zhuang, D. Li, W. Zhu, J. Hu, H. Li, and Q. Kong, "Baidu apollo em motion planner," *arXiv:1807.08048*, 2018. [Online]. Available: https://arxiv.org/abs/1807.08048

[20] Hebi robotics rosie robot: Omni-directional mobile base. [Online]. Available: https://www.hebirobotics.com/

[21] Qualisys: Motion capture system. [Online]. Available: https://www.qualisys.com/

[22] Y. Zhang, V. Nan Fernandez-Ayala, and D. V. Dimarogonas. Multi-robot human-in-the-loop control under spatiotemporal specifications. Youtube. [Online]. Available: https://youtu.be/Qnx1MjHuFEo