ACLM: ADMM-Based Distributed Model Predictive Control for Collaborative Loco-Manipulation

Anonymous Author(s)

***Abstract*— Collaborative transportation of heavy payloads via loco-manipulation is a challenging yet essential capability for legged robots operating in complex, unstructured environments. Centralized planning methods, e.g., holistic trajectory optimization, capture dynamic coupling among robots and payloads but scale poorly with system size, limiting real-time applicability. In contrast, hierarchical and fully decentralized approaches often neglect force and dynamic interactions, leading to conservative behavior. This study proposes an Alternating Direction Method of Multipliers (ADMM)-based distributed model predictive control framework for collaborative loco-manipulation with a team of quadruped robots with manipulators. By exploiting the payload-induced coupling structure, the global optimal control problem is decomposed into parallel individual-robot-level subproblems with consensus constraints. The distributed planner operates in a receding-horizon fashion and achieves fast convergence, requiring only a few ADMM iterations per planning cycle. A wrench-aware whole-body controller executes the planned trajectories, tracking both motion and interaction wrenches. Extensive simulations with up to four robots demonstrate scalability, real-time performance, and robustness to model uncertainty. `https://admm-clm.github.io/`**

I. Introduction

Recent robotics research has increasingly focused on enhancing loco-manipulation capabilities, aiming to automate repetitive and labor-intensive tasks while preserving all-terrain mobility. Among these tasks, collaborative transportation of heavy and oversized loads via loco-manipulation has attracted significant attention due to their prevalence in logistics, mining, construction, agriculture, and search-and-rescue operations [1]. Collaborative loco-manipulation, in which multiple legged robots equipped with manipulators cooperatively transport a shared load, offers a promising approach by distributing both the payload weight and the control effort across the robot team.

To simplify coordination, many approaches adopt hierarchical frameworks that decouple payload and robot planning [2], [3]. While scalable, these designs often rely on quasi-static assumptions or neglect dynamic and kinematic coupling between robots and the payload, leading to conservative motions. For quadrupedal manipulators, where locomotion and manipulation forces are strongly coupled, ignoring these interactions certainly degrade performance and stability especially on rough terrains (see Fig. 1), motivating planning approaches that explicitly model force and dynamic coupling.

In contrast, centralized planning frameworks can naturally account for dynamic coupling and shared constraints among the robots and the payload by solving a holistic optimal control problem (OCP) [4], [5]. However, this comes at the cost of significantly increased computational complexity.

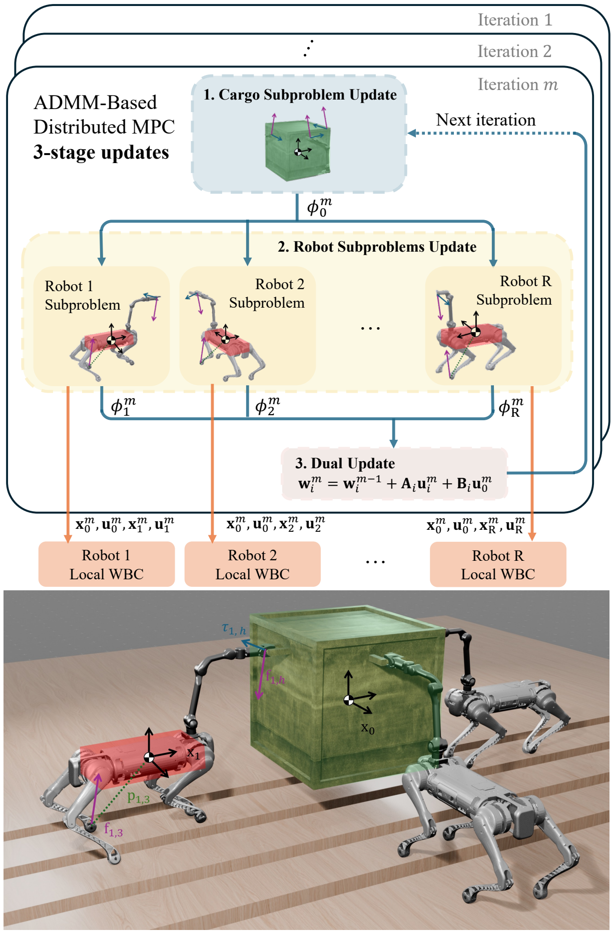
Iteration 1
Iteration 2
Iteration m
ADMM-Based
Distributed MPC
3-stage updates
1. Cargo Subproblem Update
Next iteration
ϕ_0^m
2. Robot Subproblems Update
Robot 1
Subproblem
Robot 2
Subproblem
Robot R
Subproblem
…
ϕ_1^m
ϕ_2^m
ϕ_R^m
3. Dual Update
$\mathbf{w}_i^m = \mathbf{w}_i^{m-1} + \mathbf{A}_i\mathbf{u}_i^m + \mathbf{B}_i\mathbf{u}_0^m$
$\mathbf{x}_0^m, \mathbf{u}_0^m, \mathbf{x}_1^m, \mathbf{u}_1^m$
$\mathbf{x}_0^m, \mathbf{u}_0^m, \mathbf{x}_2^m, \mathbf{u}_2^m$
$\mathbf{x}_0^m, \mathbf{u}_0^m, \mathbf{x}_R^m, \mathbf{u}_R^m$
Robot 1
Local WBC
Robot 2
Local WBC
…
Robot R
Local WBC
$\tau_{1,h}$
$f_{1,h}$
x_0
x_1
$p_{1,3}$
$f_{1,3}$

Fig. 1. ADMM-based distributed MPC framework exploiting the star-shaped coupling induced by the shared payload. The robot subproblems are solved in parallel with consensus on interaction wrenches, and the resulting trajectories are executed by local WBC.

Compared to teams of wheeled mobile manipulators or quadrotors, cooperative quadrupedal manipulators present additional challenges due to a high number of degrees of freedom (DoFs), hybrid contact dynamics, and rough terrain. As a result, the scalability of fully centralized planning with respect to the number of robots becomes a critical bottleneck, limiting its applicability for real-time planning and control.

In this work, we explicitly analyze the coupling structure between quadrupedal manipulators and a shared payload and leverage the Alternating Direction Method of Multipliers (ADMM) [6] to decompose the resulting multi-robot optimal control problem into tractable subproblems. Although the system exhibits strong coupling through the payload dynamics, each robot interacts directly only with the payload rather than with other robot teammates. This star-shaped coupling structure enables parallel optimization across robots through carefully designed consensus constraints, allowing distributed computation while preserving the essential force and dynamic interactions induced by the shared load.

We further adopt a real-time, receding-horizon implementation of the distributed planner within a model predictive control (MPC) framework. Notably, the solution from the previous MPC window provides an effective warm start for the current optimization, allowing the ADMM solver to converge to a satisfactory consensus with only a few ADMM iterations per planning cycle. At the lower level, a wrench-aware whole-body controller (WBC) tracks the planned end-effector poses, foot placements, and interaction wrenches, bridging distributed planning and execution. Although whole-body inverse dynamics is well studied [7], to our knowledge this is the first integrated distributed MPC–WBC pipeline that accounts for coupling effects in prehensile collaborative loco-manipulation with multiple legged manipulators in challenging environments.

Our contributions can be summarized as follows:

- We formulate the collaborative prehensile loco-manipulation with multiple quadrupedal manipulators as a centralized OCP and develop an ADMM-based distributed MPC that decomposes it into parallel robot-level subproblems coupled through payload dynamics.
- By exploiting the star-shaped coupling induced by the shared payload, we treat the payload as an independent subsystem and enforce consensus only on interaction wrenches while using approximated state copies to maintain compact problem size, keeping each subproblem scalable and enabling fast online planning.
- We integrate the distributed planner with a wrench-aware WBC and evaluate teams of up to four robots on diverse collaborative transportation tasks, including obstacle avoidance and rough-terrain traversal, demonstrating real-time performance independent of team size (50 Hz with perceptive inputs and 100 Hz without them) and robustness to model uncertainties.

II. Related Work

A. Collaborative Transportation and Manipulation

Multi-robot collaborative transportation has been studied on aerial and ground platforms, with approaches broadly categorized as centralized, decentralized, or leader–follower frameworks. Centralized methods scale poorly [5], [8], while decentralized approaches trade coordination optimality for scalability [9]–[11]. Leader–follower strategies [1], [12] avoid explicit group-level trajectory optimization. In contrast, collaborative loco-manipulation with legged robots remains less explored due to high DoFs and complex dynamics.

Holonomically constrained systems, using cables or rigid links, enable scalable decentralized control through learning-based or hierarchical architectures [3], [13], [14], but their applicability is limited by restrictive mechanical assumptions and reduced interaction flexibility. **Prehensile collaborative loco-manipulation** allows more flexible grasp-based interaction and has been addressed using centralized MPC with simplified models [4], hierarchical planning frameworks [2], passive-arm leader–follower designs [15], and learning-based decentralized strategies [16], [17]. However, many of these approaches depend on centralized optimization that limits scalability, requires strong hierarchy, or need complex reward design. **Non-prehensile approaches** rely on indirect manipulation such as pushing or carrying [18]–[20], typically within hierarchical planners. While suitable for long-horizon tasks, they lack direct force control and manipulation precision.

Motivated by the flexibility of prehensile manipulation and the scalability limitations of existing centralized approaches, we focus on prehensile collaborative loco-manipulation and propose a distributed MPC framework based on distributed optimization, which enables scalable subsystem updates with local communication while retaining much of the centralized solution quality and improving computational scalability.

B. ADMM for Multi-Robot Distributed Control

Many OCPs exhibit intrinsic distributed structure despite coupling effects [21]. In multi-robot systems, spatial separability enables decomposition via distributed optimization, particularly ADMM [6]. Multi-robot path finding (MAPF) is an example where these ideas have been successfully applied [22], [23]: robots maintain local copies of their own and neighboring agents' trajectories, enforce consensus over coupling constraints such as collision avoidance, and achieve scalability to large teams. However, these formulations mostly consider simplified robot models, which keep subproblem computationally tractable despite the augmented state dimension for each robot.

Collaborative manipulation also exhibits an implicit distributed structure, although the subsystems are coupled through shared object dynamics. To avoid the growing number of control inputs for the robot-object subsystem, the authors in [24] propose a decomposition in which each subsystem optimizes only the force and torque contributed by a single robot, while consensus is enforced on the object trajectory. A similar idea is extended to contact-implicit manipulation settings in [25]. In contrast, collaborative loco-manipulation with legged manipulators involves high-dimensional, multi-contact dynamics. Rather than augmenting robot states with object states, we treat the object as an independent subsystem and couple subsystems only through interaction forces and torques under consensus. This keeps subproblems compact and enables scalable distributed MPC despite nonlinear, high-dimensional dynamics.

III. Preliminaries

A. Consensus ADMM

We adopt the standard *consensus* ADMM formulation [6] for problems of the form

$$\begin{aligned} \min_{\bar{\mathbf{x}}, \{\mathbf{x}_i\}_{i=1}^{P}} \quad & \sum_{i=1}^{P} f_i(\mathbf{x}_i) + g(\bar{\mathbf{x}}) \\ \text{s.t.} \quad & \mathbf{x}_i = \bar{\mathbf{x}}, \quad i = 1, \ldots, P, \end{aligned} \tag{1}$$

where $\mathbf{x}_i$ denotes the local decision variables of subsystem i among P subsystems, $\bar{\mathbf{x}}$ is a global consensus variable, and $g(\bar{\mathbf{x}})$ is an optional regularization term. By establishing the scaled augmented Lagrangian (AL) of (1), consensus ADMM proceeds by alternating updates of the local variables, the consensus variable, and the dual variables $\{\mathbf{w}_i\}$:

$$\mathbf{x}_i^{m+1} := \arg\min_{\mathbf{x}_i} \left(f_i(\mathbf{x}_i) + \frac{\rho}{2}\|\mathbf{x}_i - \bar{\mathbf{x}}^m + \mathbf{w}_i^m\|^2 \right), \quad \forall i, \tag{2a}$$

$$\bar{\mathbf{x}}^{m+1} := \arg\min_{\bar{\mathbf{x}}} \left(g(\bar{\mathbf{x}}) + \frac{\rho}{2}\sum_{i=1}^{N} \|\mathbf{x}_i^{m+1} - \bar{\mathbf{x}} + \mathbf{w}_i^m\|^2 \right), \tag{2b}$$

$$\mathbf{w}_i^{m+1} := \mathbf{w}_i^m + \mathbf{x}_i^{m+1} - \bar{\mathbf{x}}^{m+1}, \quad \forall i. \tag{2c}$$

where m is the ADMM iteration number and $\rho > 0$ is the penalty parameter. This structure enables parallel optimization of the local subproblems (2a), followed by a lightweight consensus update (2b). The updates in (2) correspond to standard *consensus* ADMM. We use the standard *Gauss–Seidel* consensus variant, which provides better convergence than the fully parallel *Jacobi* variant while keeping the consensus subproblem inexpensive.

B. Centralized Optimization Formulation

The centralized optimization formulation for collaborative loco-manipulation is written as:

$$\min_{\mathbf{U},\mathbf{X}} \mathcal{C}(\mathbf{U},\mathbf{X}) := \sum_{k=0}^{N-1} l_k(\mathbf{x}[k],\mathbf{u}[k]) + l_N(\mathbf{x}[N]) \tag{3a}$$

$$\text{s.t.} \quad \mathbf{x}[k+1] = \mathcal{D}(\mathbf{x}[k],\mathbf{u}[k]) \tag{3b}$$

$$\mathbf{x}[0] = \mathbf{x}_{\text{init}} \tag{3c}$$

$$\mathbf{g}(\mathbf{x}[k],\mathbf{u}[k]) = 0 \tag{3d}$$

$$\mathbf{h}(\mathbf{x}[k],\mathbf{u}[k]) \leq 0 \tag{3e}$$

where $\mathbf{x}[k]$ and $\mathbf{u}[k]$ denote the state and control variables at the k-th time step over a horizon of $N+1$ steps, with the initial condition $\mathbf{x}_{\text{init}}$. The stacked state and control trajectories are defined as $\mathbf{X} := [\mathbf{x}[0]^\top, \mathbf{x}[1]^\top, \dots, \mathbf{x}[N]^\top]^\top$ and $\mathbf{U} := [\mathbf{u}[0]^\top, \mathbf{u}[1]^\top, \dots, \mathbf{u}[N-1]^\top]^\top$. At each time step, the state and control vectors are decomposed into payload and robot components: $\mathbf{x}[k] := [\mathbf{x}_0[k]^\top, \dots, \mathbf{x}_i[k]^\top, \dots, \mathbf{x}_R[k]^\top]^\top$, $\mathbf{u}[k] := [\mathbf{u}_1[k]^\top, \dots, \mathbf{u}_i[k]^\top, \dots, \mathbf{u}_R[k]^\top]^\top$, where index $i \in \{0, 1, \dots, R\}$ denotes the i-th subsystem, with $i = 0$ corresponding to the payload and $i > 0$ to the robots. The payload has no direct control input, as its motion is governed by the interaction forces and torques exerted by the robots. The payload state is defined as

$$\mathbf{x}_0[k] := [\mathbf{r}_0[k]^\top, \dot{\mathbf{r}}_0[k]^\top, \boldsymbol{\theta}_0[k]^\top, \mathbf{l}_0[k]^\top]^\top \tag{4}$$

where $\mathbf{r}$, $\dot{\mathbf{r}}$, $\boldsymbol{\theta}$, and $\mathbf{l} \in \mathbb{R}^3$ denote the position, linear velocity, Euler angles, and angular momentum of the rigid body, respectively. In this work, each robot is approximated as a single rigid body to reduce optimization complexity; however, the proposed distributed framework can readily extend to full-order models with joint-level dynamics and whole-body constraints. For each robot $i \in \{1, \dots, R\}$, the state and control variables are defined as

$$\begin{aligned} \mathbf{x}_i[k] &:= [\mathbf{r}_i[k]^\top, \dot{\mathbf{r}}_i[k]^\top, \boldsymbol{\theta}_i[k]^\top, \mathbf{l}_i[k]^\top, \mathbf{p}_i[k]^\top]^\top \\ \mathbf{u}_i[k] &:= [\mathbf{f}_i[k]^\top, \dot{\mathbf{p}}_i[k]^\top, \mathbf{f}_{i,h}[k]^\top, \boldsymbol{\tau}_{i,h}[k]^\top]^\top \end{aligned} \tag{5}$$

where $\mathbf{p}_i[k] := \left[\mathbf{p}_{i,0}[k]^\top, \dots, \mathbf{p}_{i,n_f-1}[k]^\top\right]^\top$ and $\mathbf{f}_i[k] := \left[\mathbf{f}_{i,0}[k]^\top, \dots, \mathbf{f}_{i,n_f-1}[k]^\top\right]^\top$ stack the positions ($\dot{\mathbf{p}}_i$ as velocities) and contact forces of all n_f feet, with $j \in \{0, \dots, n_f - 1\}$ indexing the feet. Here, $\mathbf{f}_{i,h}$ and $\boldsymbol{\tau}_{i,h}$ denote the manipulation force and torque applied at the arm end-effector (EE).

The coupled system dynamics $\mathcal{D}$ can be decomposed into payload and robot components $\mathcal{D}^0$ and $\mathcal{D}^i$. For clarity of presentation, the time-step superscript k is omitted in the following equations. The second-order dynamics of the payload are given by

$$\ddot{\mathbf{r}}_0 = \frac{1}{m_0}\Big(-\sum\nolimits_i \mathbf{f}_{i,h}\Big) + \boldsymbol{g}, \tag{6a}$$

$$\dot{\mathbf{l}}_0 = \sum\nolimits_i \left(\mathbf{p}_{i,h}(\mathbf{r}_0, \boldsymbol{\theta}_0) - \mathbf{r}_0\right) \times (-\mathbf{f}_{i,h}) - \boldsymbol{\tau}_{i,h}, \tag{6b}$$

where $\mathbf{p}_{i,h}(\cdot,\cdot)$ denotes the position of the i-th robot's arm EE expressed in the inertial frame, obtained by transforming a constant handle offset in the cargo frame. The second-order dynamics of the i-th robot are expressed as

$$\ddot{\mathbf{r}}_i = \frac{1}{m_i}\Big(\sum\nolimits_j \mathbf{f}_{i,j} + \mathbf{f}_{i,h}\Big) + \boldsymbol{g}, \tag{7a}$$

$$\dot{\mathbf{l}}_i = \sum\nolimits_j (\mathbf{p}_{i,j} - \mathbf{r}_i) \times \mathbf{f}_{i,j} + \tag{7b}$$

$$(\mathbf{p}_{i,h}(\mathbf{r}_0, \boldsymbol{\theta}_0) - \mathbf{r}_i) \times \mathbf{f}_{i,h} + \boldsymbol{\tau}_{i,h}, \tag{7c}$$

All continuous-time dynamics are discretized using the backward Euler method for use in the optimal control formulation. The equality and inequality constraint functions $\mathbf{g}(\cdot)$ and $\mathbf{h}(\cdot)$ are introduced later.

IV. Distributed Optimization

In the previous section, it can be seen that each dynamics equation and part of the inequality constraints still depend not only single payload/robot state but also other system's states. For the latter case, we call them inter-system constraints. Therefore, one major contribution of this work is to develop an optimization framework that further separates each subsystem so that each subproblem can be solved in parallel to improve the computational efficiency.

A. Decomposed System Dynamics

To apply ADMM, we first create the control variables $\mathbf{u}_0$ for the payload, which consists of global copies of the manipulation wrench, including force and torque variables applied by each robot:

$$\mathbf{u}_0[k] := [\bar{\mathbf{f}}_{i,h}[k]^\top, \bar{\boldsymbol{\tau}}_{i,h}[k]^\top]^\top, i \in \{1, \dots, R\} \tag{8}$$

For better illustration of each subsystem, we use $\boldsymbol{\phi}_i = [\mathbf{x}_i[0, \dots, N], \mathbf{u}_i[0, \dots, N-1]]^\top$ to express the trajectory of the concatenated state and control variables for the i-th rigid body. Then we build consensus constraint for the manipulation force and torque variables between the payload and robot states based on the Newton's law:

$$\forall i \in \{1, \dots, R\},$$
$$\mathbf{f}_{i,h}[k] + \bar{\mathbf{f}}_{i,h}[k] = 0, \tag{9a}$$
$$\boldsymbol{\tau}_{i,h}[k] + \bar{\boldsymbol{\tau}}_{i,h}[k] = 0 \tag{9b}$$

Remark 1: ***Approximated State Copy**: Instead of introducing consensus copies of both state and control variables for each robot's neighbor (the payload), we deliberately avoid enforcing state consensus to prevent unnecessary state augmentation. For a single quadrupedal manipulator, even using a simplified model, the subsystem already has a high-dimensional state (i.e., 24 states), making the computational cost highly sensitive to further state expansion. Moreover, unlike contact-implicit formulations [25], the dynamic coupling between the robots and the payload in our static grasping setup is dominated by interaction forces and torques rather than state variables. Consequently, for dynamics and inter-system constraints that involve state variables from other subsystems, we use the values from the previous ADMM iteration. This approximation has been empirically shown to have negligible impact on performance while improving computational efficiency, consistent with observations in [26].*

By establishing local copies, the payload dynamics gets rid of the dependency on $\mathbf{u}_i (i > 0)$ and replaces it with the new control variable $\mathbf{u}_0$. To consider the coupling effects from the payload state in the robot dynamics without creating payload state copies, we use the payload states from the previous ADMM iteration as described in Remark 1. The discrete dynamics equations for the payload and robots at the $m + 1^{\text{th}}$ ADMM iteration can be written as:

$$\mathbf{x}_0^{m+1}[k+1] = \mathcal{D}^0(\mathbf{x}_0^{m+1}[k], \mathbf{u}_0^{m+1}[k]) \tag{10a}$$
$$\mathbf{x}_i^{m+1}[k+1] = \mathcal{D}^i(\mathbf{x}_i^{m+1}[k], \mathbf{x}_0^{m}[k], \mathbf{u}_i^{m+1}[k]) \tag{10b}$$

Then at the $m + 1^{\text{th}}$ ADMM iteration, the states from m^{th} ADMM iteration can be treated as constant variables.

B. Constraints

We classify the constraints into independent constraints, applied locally to each subsystem, and inter-system coupling constraints, handled through ADMM.

1) Independent — ***Frictional and Contact**:* The manipulation contact wrench cone is simplified to a positive normal force constraint, a manipulation friction cone, and a bounding box $\mathcal{B}$ for the moment part defined in Eqs. (11a) - (11b). Point-foot positive normal force and frictional constraints are defined in Eqs. (11c) - (11d). We define $\mathbf{n}$ as the normal vector of the grasping point or the ground, corresponding to the payload orientation $\boldsymbol{\theta}_0$ or the foot EE tangential position $\mathbf{p}_{i,j}^{xy}$ by inquiring the terrain map. The friction cone $\mathcal{F}$ depends on the friction coefficient μ and the $\mathbf{n}$. Contact constraints force the EE velocity to be zero during stance phase and the contact force to be zero during non-contact.

$$\forall i \in \{1, \dots, R\},$$
$$\mathbf{f}_{i,h}[k] \cdot \mathbf{n}_{i,h}(\boldsymbol{\theta}_0[k]) \geq 0, \tag{11a}$$
$$\mathbf{f}_{i,h}[k] \in \mathcal{F}(\mu, \mathbf{n}_{i,h}), \boldsymbol{\tau}_{i,h} \in \mathcal{B}_{i,h} \tag{11b}$$
$$\mathbf{f}_{i,j}[k] \cdot \mathbf{n}_{i,j}(\mathbf{p}_{i,j}^{xy}[k]) \geq 0, \ \forall j \in [n_f] \tag{11c}$$
$$\mathbf{f}_{i,j}[k] \in \mathcal{F}(\mu, \mathbf{n}_{i,j}), \ \forall j \in [n_f] \tag{11d}$$

2) Independent — ***Foot Kinematics**:* The kinematics constraint limits the possible foot EE movements inside a nominal box region to ensure safety and avoid singularity.

3) Independent — ***Foot Placement**:* Foot placements are constrained to satisfy either a default terrain height or the elevation map. In perceptive mode, admissible footholds are restricted to convex safe regions extracted from the elevation map, ensuring collision-free and stable contacts. The convex regions are selected heuristically for each stance phase similar to [27].

4) Independent — ***Collision Avoidance**:* Obstacle avoidance is incorporated via a control barrier function (CBF) formulation, maintaining forward invariance of the safe set for both robots and payload. Similarly, a terrain-aware constraint is imposed on the payload to prevent collisions during terrain transitions, such as moving from flat ground to a slope.

5) Inter-System — ***Arm Kinematics**:* The arm kinematics constraint bounds the relative pose between each robot base and the grasping handle (arm EE), ensuring the grasp remains within a feasible workspace. Therefore, similar to the previous method for decomposing the system dynamics, we use the state from the previous ADMM iteration. Note that due to the fixed state assumption, we add additional constraints for the payload with fixed robot states, maintaining the constraint feasibility from both payload and robots.

$$\mathbf{r}_0[k], \boldsymbol{\theta}_0[k] \in \mathcal{R}_h(\mathbf{r}_i^m[k], \boldsymbol{\theta}_i^m[k], \mathbf{p}_h^{\max})$$
$$\forall i \in \{1, \dots, R\}, j \in [n_f]$$
$$\mathbf{r}_i[k], \boldsymbol{\theta}_i[k] \in \mathcal{R}_h(\mathbf{r}_0^m[k], \boldsymbol{\theta}_0^m[k], \mathbf{p}_h^{\max})$$

where $\mathcal{R}_h$ is defined as a 3D box constraint around a nominal arm workspace based on the anchored system's position and orientation, and constrained by a maximum deviation $\mathbf{p}_h^{\max}$.

6) Inter-System — ***Formation**:* The formation constraint keeps each robot's center of mass near its nominal offset from the payload in the payload frame. Although similar to the arm kinematics constraint, it empirically enhances team-level obstacle avoidance and mitigates the risk of individual robots or the payload getting trapped in local minima. As with other inter-system constraints, it is enforced in a distributed manner by using the other subsystem's state from the previous ADMM iteration.

C. Cost Function

The cost function is directly defined for each subsystem separately, consisting of simple tracking costs and regularization terms governed by diagonal matrices $\mathbf{Q}$ and $\mathbf{R}$.

$$l_k^i = \delta\boldsymbol{\phi}_Q[k]^T \mathbf{Q}\, \delta\boldsymbol{\phi}_Q[k] + \boldsymbol{\phi}_R[k]^T \mathbf{R} \boldsymbol{\phi}_R[k] \tag{12}$$

where $\delta\phi_Q = \mathbf{x}_i - \mathbf{x}_i^{\text{ref}}$ and $\phi_R = \mathbf{u}_i$. Tracking costs include deviation from the desired base reference trajectories. The regularization term includes minimizing the foot and arm EE velocities, and contact forces and wrenches to encourage motion smoothness.

D. Distributed Update

Now we are able to adopt *consensus* ADMM to update each subproblem separately. Based on the decomposition from previous sections, the overall optimization for each ADMM iteration now can be further written as:

$$
\begin{aligned}
\min_{\phi_0,\ldots,\phi_N} \quad & \mathcal{C} := \sum_{i=0}^{R} \Big(\sum_{k=0}^{N-1} l_k^i(\mathbf{x}_i[k], \mathbf{u}_i[k]) + l_N^i(\mathbf{x}_i[N]) \Big) && \text{(13a)} \\
\text{s.t.} \quad & \mathbf{x}_0[k+1] = \mathcal{D}^0(\mathbf{x}_0[k], \mathbf{u}_0[k]) && \text{(13b)} \\
& \mathbf{h}_0(\mathbf{x}_0[k], \mathbf{u}_0[k]) \leq 0 && \text{(13c)} \\
& \forall i \in \{1, \ldots, R\} \\
& \mathbf{x}_i[k+1] = \mathcal{D}^i(\mathbf{x}_i[k], \mathbf{u}_i[k]) && \text{(13d)} \\
& \mathbf{g}_i(\mathbf{x}_i[k], \mathbf{u}_i[k]) = 0 && \text{(13e)} \\
& \mathbf{h}_i(\mathbf{x}_i[k], \mathbf{u}_i[k]) \leq 0 && \text{(13f)} \\
& \mathbf{A}_i \mathbf{u}_i + \mathbf{B}_i \mathbf{u}_0 = 0 && \text{(13g)}
\end{aligned}
$$

where $\mathbf{A}_i$ and $\mathbf{B}_i$ denote the vectorized forms of the consensus constraints defined in Eqs. (9), which constitute the only coupling constraints among the subproblems.

Then, based on ADMM, the original problem is divided into $N+1$ subproblems which are also known as sub-blocks. Each sub-block only requires part of the aforementioned AL as the local cost function with the payload sub-block being:

$$
\begin{aligned}
\mathcal{L}_0(\cdot) = & \sum_{k=0}^{N-1} l_k^0(\mathbf{x}_0[k], \mathbf{u}_0[k]) + l_N^0(\mathbf{x}_0[N]) + \\
& \sum_{i=1}^{R} \frac{\rho}{2} \|\mathbf{A}_i \mathbf{u}_i + \mathbf{B}_i \mathbf{u}_0 + \mathbf{w}_i\|^2
\end{aligned} \tag{14}
$$

The i^{th} robot sub-block can be formulated as:

$$
\begin{aligned}
\mathcal{L}_i(\cdot) = & \sum_{k=0}^{N-1} l_k^i(\mathbf{x}_i[k], \mathbf{u}_i[k]) + l_N^i(\mathbf{x}_i[N]) + \\
& \frac{\rho}{2} \|\mathbf{A}_i \mathbf{u}_i + \mathbf{B}_i \mathbf{u}_0 + \mathbf{w}_i\|^2
\end{aligned} \tag{15}
$$

Then as shown in Fig. 1, for each ADMM iteration m, the updating sequence in a scaled form is

$$
\begin{aligned}
\phi_0^{m+1} &= \arg\min_{\phi_0} \mathcal{L}_0(\phi_0, \phi_1^m, \ldots, \phi_N^m, \mathbf{w}^m) && \text{(16a)} \\
& \quad \text{s.t. Eq. (13b), Eq. (13c)} \\
\phi_i^{m+1} &= \arg\min_{\phi_i} \mathcal{L}_i(\phi_0^{m+1}, \phi_i, \mathbf{w}^m) && \text{(16b)} \\
& \quad \text{s.t. Eq. (13d), Eqs. (13e} - \text{13f)} \\
\mathbf{w}_i^{m+1} &= \mathbf{w}_i^m + \mathbf{A}_i \mathbf{u}_i^{m+1} + \mathbf{B}_i \mathbf{u}_0^{m+1} && \text{(16c)}
\end{aligned}
$$

where for each subproblem, due to the nonlinear dynamics, we employ a Riccati-like Sequential Quadratic Programming (SQP) solver to efficiently get the reduced-size subproblem solutions. All inequality constraints are penalized with a relaxed log barrier. Lastly, based on the consensus constraints defined in (9), we define the vectorized residual as $\mathbf{s} = [\mathbf{A}_1\mathbf{u}_1 + \mathbf{B}_1\mathbf{u}_0, \ldots, \mathbf{A}_R\mathbf{u}_R + \mathbf{B}_R\mathbf{u}_0]^\top$. Similar to [28], the stopping criteria is based on a time-scaled sum squared error with a tolerance ϵ: $dt \cdot \sum_k \|\mathbf{s}[k]\|_2 < \epsilon$.

V. Real-Time Planning and Control

A. Warm-Start for Distributed MPC

The distributed MPC framework involves iterative replanning based on Eq. (16). To accelerate convergence, we employ two warm-start strategies. First, within a single MPC solve, the solution of each subproblem from the previous ADMM iteration is used to initialize the corresponding subproblem in the current iteration. Second, across successive MPC solves, the primal and dual solutions from the previous window are used to initialize the first ADMM iteration of the new window. For the initial time window, the dual variables are initialized to zero, and the reference trajectory is used as the primal initial guess.

B. Wrench-Aware Whole-Body Control

The wrench-aware WBC executes the planned motion and interaction wrenches at 500 Hz. It tracks the optimized base, foot, and arm EE poses together with the desired manipulation wrench, solving for generalized accelerations, contact forces, and joint torques. Inverse kinematics (IK) provides joint position and velocity references for both legs and arm, and the final torque command combines WBC feedforward torques with low-gain PD feedback to realize the desired motion and interaction wrenches while compensating residual tracking errors.

A central design choice is the task hierarchy. We adopt a hierarchical quadratic program (QP) [7] with three priority levels, so that safety and dynamic consistency are never traded off for tracking. In the highest priority, we enforce: the floating-base equations of motion, joint torque limits, friction cones for feet and arm contact, no-contact motion for swing legs, and the arm contact wrench tracking task given the MPC desired states. In the second level, we handle base pose, swing-leg task, and arm EE pose tracking, while in the lowest level, we track the leg contact force and regularize torque/joint-acceleration. Unlike [7], we elevate wrench tracking to the highest priority in our dynamic manipulation task, thereby guaranteeing strict wrench consistency while exploiting the remaining redundancy for pose objectives.

VI. Results

A. Experimental Setup

Unless otherwise specified, we use at most 2 ADMM iterations and 1 SQP iteration per MPC solve. The rationale is discussed in Sec. VI-C. We refer to different iteration limits as ADMM-SQP iterations. The consensus constraint tolerance is set to 5×10^{-3}. All experiments are conducted on Unitree B1-Z1 via an Intel Core i7-14650HX CPU. The implementation is majorly built upon OCS2 [29]. The perception modules consist of elevation mapping and convex segmentation implemented in [30].

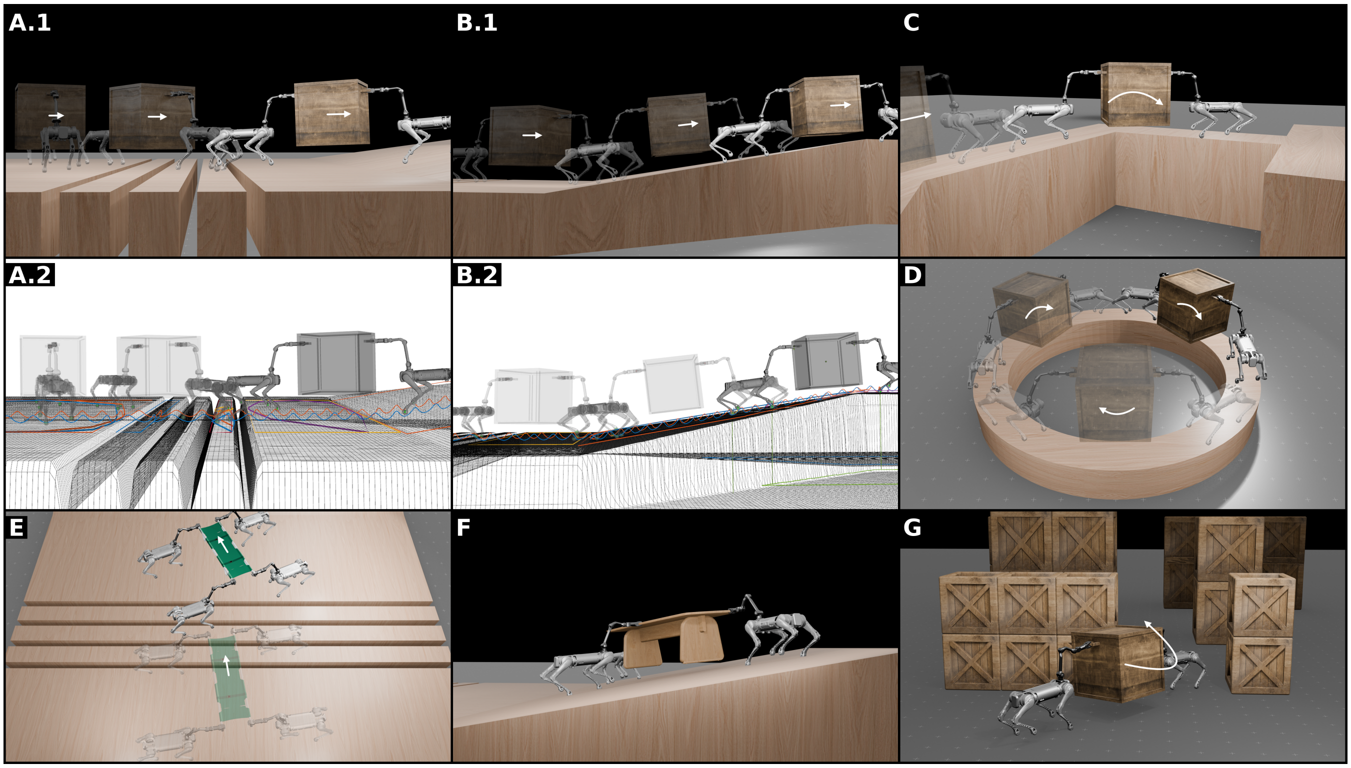

Fig. 2. Illustration of diverse terrain navigation scenarios used to evaluate collaborative loco-manipulation. **A.1–A.2**: Two robots carry a cargo across stepped terrain of uniform surface height, shown in rendered and RViz views. **B.1–B.2**: Two robots carry a cargo on a sloped surface, shown in rendered and RViz views. **C**: Two robots carry a cargo through a 90-degree turn in a narrow passage. **D**: Two robots carry a cargo following a circular path on an elevated annular platform. **E**: Four robots carry a folding stretcher across stepped terrain of uniform surface height. **F**: Three robots carry a wooden table on a sloped surface. **G**: Two robots carry a cargo performing obstacle avoidance on flat terrain.

We evaluate two aspects of the proposed framework: 1) **Scalability** and **Convergence**: Experiments are conducted in a non-physical simulator similar to the one used in [4] by rolling out the system dynamics in (6) and (7) using only the first-step MPC optimized inputs. The full-body motions are realized through IK without WBC, allowing us to isolate trajectory quality and MPC solving time from tracking errors or lower-level controller effects. 2) **Tracking performance**: Experiments are performed in a high-fidelity physical simulator with Gazebo physics engine and rendered in Blender. The MPC and WBC run asynchronously to emulate realistic multi-rate interactions between planning and control. The payloads consist of a wide range of geometries and mass including cargo box, table, and folding stretcher.

B. Rough Terrain

Fig. 2 summarizes results across rough terrain scenarios. In the Gap case (A.1–A.2), the two-robot team clears steps terrain with adaptive swing heights while maintaining stable payload transport. In the Slope case (B.1–B.2), the system traverses a 10° incline with robot base and cargo orientation compensating for terrain tilt. The Narrow Turn scenario (C) presents the most constrained configuration, requiring coordinated replanning as the team redirects the payload through a 90° turn within a narrow passage. The annular platform scenario (D) demonstrates stable formation tracking under continuous curvature on an elevated circular path.

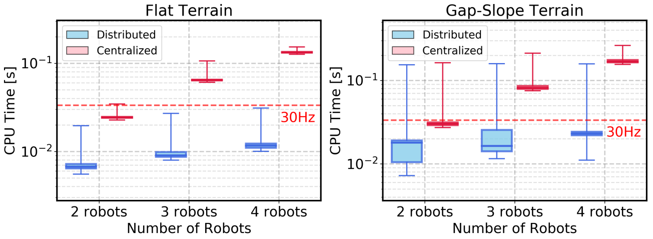

Fig. 3. CPU time scalability comparison between distributed and centralized MPC for 2–4 robots in flat and Gap-Slope terrain scenarios.

C. Scalability and Convergence Analysis

We use the flat and Gap-Slope (Fig. 2-E and F) terrain scenarios as benchmarks for scalability tests between distributed and centralized MPC. Fig. 3 shows computational advantages for distributed MPC that increase with robot team size. In flat terrain, distributed MPC achieves 3.6×, 7.1×, and 11.4× speedups for 2, 3, and 4 robots, with median CPU times of 6.73 ms to 11.63 ms compared to centralized MPC's 24.38 ms to 133.13 ms. In Gap-Slope terrain with perceptive constraints, distributed MPC achieves 1.7×, 5.0×, and 7.3× speedups with median times of 18.01 ms, 16.39 ms, and 23.03 ms. Distributed MPC remains 50 Hz (100 Hz on flat terrain) for all team sizes, whereas centralized MPC is below 30 Hz for 3 and 4 robots. The largest runtime outliers occur during the initial solve due to perception initialization and cold starts. The superior scalability of distributed MPC stems from decomposing the multi-robot optimization into smaller, parallelizable subproblems, resulting in nearly uniform com-

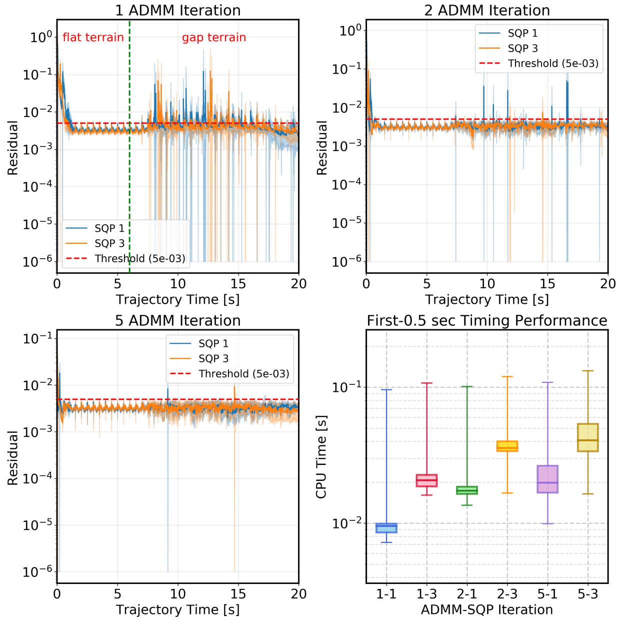

Fig. 4. Residual convergence (top-left, top-right, and bottom-left) and computation time (bottom-right) for different ADMM-SQP iterations.

putational time across team sizes, whereas centralized MPC must solve a single, linearly growing problem that becomes computationally prohibitive as team size increases.

To evaluate ADMM convergence and warm-start effects, we run 60 trials on a Gap terrain (Fig. 2-A) with different ADMM-SQP iterations. Each trial uses a waypoint sent across the gap with random offsets of ± 1 m in x-y and $\pm 90^\circ$ in yaw. As shown in Fig. 4, with 1 ADMM iteration, the residual drops more slowly initially along with trajectory time than with more ADMM iterations, and deviation increases, especially in the gap phase, where the consensus residual becomes unsuppressed. More SQP iterations slightly reduce large constraint violations. With 2 ADMM iterations, the residual stays within the threshold with rare violations; 5 iterations improve further. To capture computational differences, we record computation time for the first 0.5 s of planned trajectory after MPC starts, since the initial steps must quickly reduce the residual for consensus and the solver starts from scratch. Under the same SQP iteration, more ADMM iterations converge faster initially but are more expensive and show sparser time distributions, as more ADMM iterations are needed to converge, slowing the process initially. More SQP iterations substantially increase computation time under the same ADMM iteration. Therefore, we chose 2 ADMM iterations and 1 SQP iteration to balance constraint satisfaction and computational efficiency.

D. Obstacle Avoidance

To evaluate the obstacle avoidance capability of the proposed MPC framework, a scenario is constructed in a physical simulator as shown in Fig. 2-G and Fig. 5. The left portion of Fig. 5 illustrates the resulting trajectory, showing that the robot–cargo system safely traverses among all obstacles. With only a simple start-to-goal interpolation as the initial reference, the MPC automatically finds a feasible

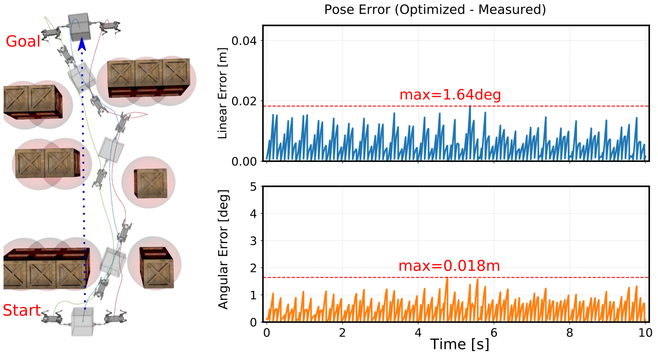

Fig. 5. Obstacle avoidance and pose tracking performance.

trajectory that turns and leverages manipulator flexibility to pass narrow passages while remaining close to the reference. The right portion of Fig. 5 shows the pose tracking error over a representative 10 s segment of the full 110 s duration, obtained from integrated MPC-WBC. The maximum linear error is 0.018 m, and the maximum angular error is 1.64°. Periodic spikes in the error correspond to ground impact events during trotting, which introduce transient disturbances but remain bounded. These results demonstrate that the proposed MPC–WBC framework achieves reliable real-time obstacle avoidance while maintaining accurate pose tracking.

E. Robustness and Ablation Study

To evaluate robustness, the robot–cargo system is commanded to track a reference trajectory consisting of a $6.5\,\mathrm{m}$ translation with a sharp 90° turn in physical simulations. Multiple trials are performed under different nominal cargo masses as well as under deliberate mass and inertia modeling errors. As shown in Fig. 6, despite rare transient peaks caused by foot–ground impacts, the median errors and interquartile ranges remain consistently low across different nominal masses and under both mass and inertia modeling error, demonstrating strong robustness of the proposed controller. The controller performance degrades to failure when modeling error is increased to approximately 67%.

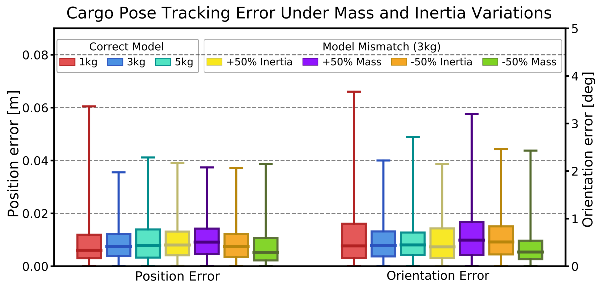

Fig. 6. Cargo pose tracking errors under mass and inertia variations.

Another important feature of our framework is that we optimize and track the full wrench, including both force and torque at the grasped handles. As an ablation study, when torque is completely disabled, the angular tracking error increases over time and eventually leads to instability, demonstrating the limitation of force-only tracking. Enabling torque tracking, particularly along the alignment axis between the robot and cargo (x-axis), significantly improves rotational stability and reduces angular error. The quantitative result is shown in Fig. 7.

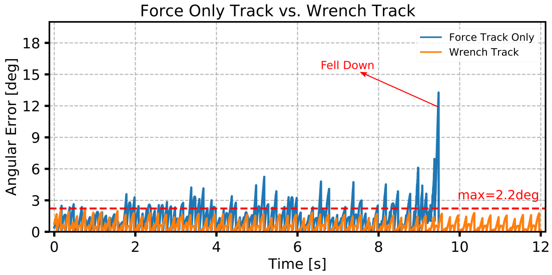

Fig. 7. Comparison between force-only tracking and full wrench tracking.

VII. Conclusion

This paper presented an ADMM-based distributed MPC framework for collaborative prehensile loco-manipulation with multiple quadrupedal manipulators. Exploiting the star-shaped coupling induced by a shared payload, the centralized optimal control problem is decomposed into parallel robot-level subproblems, preserving dynamic coupling while improving scalability. Combined with a wrench-aware whole-body controller, the framework achieves real-time, force-consistent execution. Simulations with up to four robots validate scalability, fast convergence, and robustness to model and terrain variations. Future work includes hardware validation on physical platforms, and exploring GPU implementations to guide decentralized RL training.

References

[1] H. Farivarnejad and S. Berman, "Multirobot control strategies for collective transport," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, no. 1, pp. 205–219, 2022.

[2] A. Rigo, J. Ma, N. Chun, S. K. Gupta, and Q. Nguyen, "Hierarchical control framework for collision-free collaborative loco-manipulation of large and heavy objects," in *2025 IEEE 21st International Conference on Automation Science and Engineering (CASE)*. IEEE, 2025, pp. 333–338.

[3] C. Yang, N. Guo, Z. Li, L. Yang *et al.*, "Collaborative Navigation and Manipulation of A Cable-Towed Load by Multiple Quadrupedal Robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 041–10 048, 2022.

[4] F. De Vincenti and S. Coros, "Centralized Model Predictive Control for Collaborative Loco-Manipulation," in *Robotics: Science and Systems*, vol. 2023, 2023.

[5] S. Sun, X. Wang, D. Sanalitro, A. Franchi, M. Tognon, and J. Alonso-Mora, "Agile and cooperative aerial manipulation of a cable-suspended load," *Science Robotics*, vol. 10, no. 107, p. eadu8015, 2025.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[7] C. D. Bellicoso, K. Krämer, M. Stäuble, D. Sako, F. Jenelten, M. Bjelonic, and M. Hutter, "Alma-articulated locomotion and manipulation for a torque-controllable robot," in *2019 International conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 8477–8483.

[8] A. Nikou, C. Verginis, S. Heshmati-Alamdari, and D. V. Dimarogonas, "A nonlinear model predictive control scheme for cooperative manipulation with singularity and collision avoidance," in *2017 25th Mediterranean Conference on Control and Automation (MED)*. IEEE, 2017, pp. 707–712.

[9] O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, and A. Casal, "Coordination and decentralized cooperation of multiple mobile manipulators," *Journal of Robotic Systems*, vol. 13, no. 11, pp. 755–764, 1996.

[10] C. K. Verginis, A. Nikou, and D. V. Dimarogonas, "Communication-based decentralized cooperative object transportation using nonlinear model predictive control," in *2018 European control conference (ECC)*. IEEE, 2018, pp. 733–738.

[11] P. Culbertson and M. Schwager, "Decentralized adaptive control for collaborative manipulation," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 278–285.

[12] T. G. Sugar and V. Kumar, "Control of cooperating mobile manipulators," *IEEE Transactions on robotics and automation*, vol. 18, no. 1, pp. 94–103, 2002.

[13] B. Pandit, A. Gupta, M. S. Gadde, A. Johnson *et al.*, "Learning Decentralized Multi-Biped Control for Payload Transport," *arXiv preprint arXiv:2406.17279*, 2024.

[14] J. Kim, R. T. Fawcett, V. R. Kamidi, A. D. Ames, and K. A. Hamed, "Layered Control for Cooperative Locomotion of Two Quadrupedal Robots: Centralized and Distributed Approaches," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4728–4748, 2023.

[15] G. Turrisi, L. Schulze, V. S. Medeiros, C. Semini, and V. Barasuol, "PACC: A Passive-Arm Approach for High-Payload Collaborative Carrying with Quadruped Robots Using Model Predictive Control," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 11 139–11 146.

[16] B. Pandit, A. K. Shrestha, and A. Fern, "Multi-quadruped cooperative object transport: Learning decentralized pinch-lift-move," *arXiv preprint arXiv:2509.14342*, 2025.

[17] T. An, F. De Vincenti, Y. Ma, M. Hutter, and S. Coros, "Collaborative loco-manipulation for pick-and-place tasks with dynamic reward curriculum," *arXiv preprint arXiv:2509.13239*, 2025.

[18] Y. Feng, C. Hong, Y. Niu, S. Liu, Y. Yang, and D. Zhao, "Learning multi-agent loco-manipulation for long-horizon quadrupedal pushing," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 14 441–14 448.

[19] M. Sombolestan and Q. Nguyen, "Hierarchical Adaptive Motion Planning with Nonlinear Model Predictive Control for Safety-Critical Collaborative Loco-Manipulation," *arXiv:2411.10699*, 2024.

[20] Y. Ji, B. Zhang, and K. Sreenath, "Reinforcement learning for collaborative quadrupedal manipulation of a payload over challenging terrain," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 899–904.

[21] Z. Zhao, S. Cheng, Y. Ding, Z. Zhou, S. Zhang, D. Xu, and Y. Zhao, "A survey of optimization-based task and motion planning: From classical to learning approaches," *IEEE/ASME Transactions on Mechatronics*, 2024.

[22] A. D. Saravanos, Y. Aoyama, H. Zhu, and E. A. Theodorou, "Distributed differential dynamic programming architectures for large-scale multiagent control," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4387–4407, 2023.

[23] A. Tajbakhsh, A. Saravanos, J. Zhu, E. A. Theodorou, L. T. Biegler, and A. M. Johnson, "Asynchronous distributed multi-robot motion planning under imperfect communication," *arXiv preprint arXiv:2511.18703*, 2025.

[24] O. Shorinwa and M. Schwager, "Scalable collaborative manipulation with distributed trajectory planning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 9108–9115.

[25] O. Shorinwa, M. Devlin, E. W. Hawkes, and M. Schwager, "DisCo: Distributed Contact-Rich Trajectory Optimization for Forceful Multi-Robot Collaboration," Oct. 2024, arXiv:2410.23283. [Online]. Available: http://arxiv.org/abs/2410.23283

[26] L. Amatucci, G. Turrisi, A. Bratta, V. Barasuol, and C. Semini, "Accelerating Model Predictive Control for Legged Robots through Distributed Optimization," Tech. Rep., 2022.

[27] R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter, "Perceptive locomotion through nonlinear model-predictive control," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3402–3421, 2023.

[28] Z. Zhou and Y. Zhao, "Accelerated admm based trajectory optimization for legged locomotion with coupled rigid body dynamics," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 5082–5089.

[29] F. Farshidian *et al.*, "OCS2: An open source library for optimal control of switched systems," [Online]. Available: https://github.com/leggedrobotics/ocs2.

[30] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter, "Elevation mapping for locomotion and navigation using gpu," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 2273–2280.