IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Efficient Optimization-based Cable Force Allocation for Geometric Control of a Multirotor Team Transporting a Payload

Khaled Wahba and Wolfgang Hönig

***Abstract*— We consider transporting a heavy payload that is attached to multiple multirotors. The current state-of-the-art controllers either do not avoid inter-robot collision at all, leading to crashes when tasked with carrying payloads that are small in size compared to the cable lengths, or use computational demanding nonlinear optimization. We propose an efficient optimization-based cable force allocation for a geometric payload transport controller to effectively avoid such collisions, while retaining the stability properties of the geometric controller. Our approach introduces a cascade of carefully designed quadratic programs that can be solved efficiently on highly constrained embedded flight controllers.**

We show that our approach exceeds the state-of-the-art controllers in terms of scalability by at least an order of magnitude for up to 10 robots. We demonstrate our method on challenging scenarios with up to three small multirotors with various payloads and cable lengths, where our controller runs in realtime directly on a microcontroller on the robots.

I. Introduction

Aerial vehicles can access remote areas, making them suitable for collaborative tasks at a construction site, rubble removal in search-and-rescue scenarios, or nuclear power plant decommissioning. Cable-driven payload transportation using multi-UAVs is beneficial as it avoids carrying manipulators or grippers onboard, enabling the transportation of heavier objects [1], especially tools or supplies [2].

State-of-the-art controllers for a team of multirotors to transport a cable-suspended payload include geometric controllers [3, 4] as well as nonlinear model predictive control [5, 6]. These existing works either ignore possible collisions between robots [3], or use a nonlinear optimization framework that necessitates complex on-board computation [5, 7]. Collision between robots can easily occur for payloads that are small in size compared to the robot's size and the cable length which poses a safety risk.

We propose a new distributed cable force allocation method for the geometric controller by [3] that provides the team with new capabilities, see Fig. 1. In particular, we directly consider safety distances between robots and allow for user-preferred formations. Our approach unifies the handling of point mass [4] and rigid body [3] payload types. We use a cascaded design of three low-dimensional quadratic programs (QPs), which have global minima that can be found within milliseconds even on highly resource-limited microcontrollers. In fact, this approach allows us to execute the control law in a distributed fashion on physical robots. Effectively, our method allows users to teleoperate a team of robots by simply commanding the payload, independent of the payload size or type. Our contributions are as follows:

1) Algorithmically, we provide a novel distributed and payload-type-agnostic QP-based force allocator that can be deployed on microcontrollers.
2) Empirically, we demonstrate our distributed controller on a team of up to three small (34 g) multirotors carrying different payloads. We also show a superior scalability with the number of robots compared to existing state-of-the-art controllers.

We note that we are the first to demonstrate the geometric controller on a highly compute-constrained flight controller.

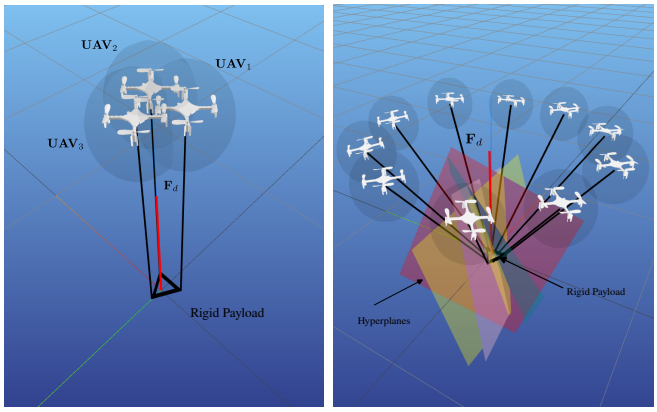

Fig. 1. We contribute an efficient payload-type-agnostic approach to compute desired cable forces that avoids collisions. Left: Three multirotors carrying a rigid body payload with the baseline controller [3] . The desired cable forces are allocated with (5), causing inter-robot collisions. Right: Our approach with computed hyperplanes that define a safe convex set for the cable forces, while considering the desired motion of the payload.

II. Related Work

The dynamics of the cable-suspended payload system with multi-UAVs can be described explicitly through the interaction forces between the payload, cables, and the UAVs [1]. In general, the dynamics are expressed using Newton's equations of motion [8–10], while augmenting the rotational dynamics of the UAVs [3].

Manuscript received: August 11, 2023; Revised: November 11, 2023; Accepted: December 11, 2023. This paper was recommended for publication by Editor M. Ani Hsieh upon evaluation of the Associate Editor and Reviewers' comments.

All authors are with Faculty of Electrical Engineering and Computer Science, Technical University of Berlin, Berlin, Germany {k.wahba, hoenig}@tu-berlin.de.

The work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 448549715.

Video: https://youtu.be/3OfnUIiu3bs

Code: https://github.com/IMRCLab/col-trans

Digital Object Identifier (DOI): see top of this page.

There has been some advancement in simulation, planning, and control of payload collaborative transport. One method involves simulating the system dynamics [11] and uses offline motion planners [12, 13] to generate references for states, sacrificing reactivity during offline planning.

Conversely, **centralized controllers** have been devised for the entire system in a cascaded reactive fashion [3, 4, 14, 15]. These are based on force or kinematic analysis and often ensure exponential stability. However, practical challenges remain, such as the need to measure noisy payload accelerations. Estimates using disturbance observers [16] are tested in simulation, only. Moreover, these methods do not take inter-robot collisions and cable tangling into account.

Decentralized controllers enhance robustness and scalability [17, 18], but have only been tested in simulations or assume the presence of reliable communication. Other methods do not rely on communication [9, 19]. Nevertheless, inter-robot collisions are not considered by these methods.

Optimization-based controllers may use iterative gradient-based solvers, but they are prone to local minima [20, 21]. Another approach is to constrain the cables into predefined convex regions [22]. However, all these methods lack direct consideration of inter-robot collisions. Nonlinear model predictive control (NMPC) can control the payload and avoid inter-robot collisions [5, 6]. However, these methods are either limited to simulations or computationally costly for highly constrained microcontrollers, and scalability with the number of robots remains poor.

The closest methods to our approach allocate cable forces by employing either null-space methods with closed-form solutions [7] or nonlinear optimization [7, 23], including inter-robot collisions. In contrast, our method uses convex optimization and supports novel use cases, such as different cable lengths, incorporating load distribution and inter-robot collision constraints given a desired formation. Thus, these two prior methods are unsuitable for direct comparison to our work. Our approach utilizes a cascade of solving consecutive convex programs where constraints are defined by cutting off parts of the non-convex solution in exchange of finding a solution quicker. Thus, our method is not equivalent to any existing non-convex formulation [7, 23]. We note that we have not seen cases where our convexification caused infeasibility. On the other hand, non-convex formulations might fail without a good initial guess.

This paper builds on existing controllers [3, 4] and enhances them by incorporating consecutive quadratic programs to optimize the cable force allocation that take into account various constraints for different use cases while retaining stability. Moreover, we demonstrate that our method surpasses the state-of-the-art controller [6] in terms of scalability and computational costs. It is noteworthy that [6] was the only work that provided a detailed scalability analysis.

III. Background

A. Single Multirotor Dynamics

The dynamics of a single multirotor is modeled as a 6 degrees-of-freedom floating rigid body with mass m and diagonal moment of inertia $\mathbf{J}$. The multirotor's state comprises of the global position $\mathbf{p} \in \mathbb{R}^3$, global velocity $\mathbf{v} \in \mathbb{R}^3$, attitude rotation matrix $\mathbf{R} \in SO(3)$ and body angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$. The dynamics can be expressed using Newton-Euler [24] equations of motion as follows

$$\dot{\mathbf{p}} = \mathbf{v}, \qquad m\dot{\mathbf{v}} = m\mathbf{g} + \mathbf{R}\mathbf{f}_u, \tag{1a}$$

$$\dot{\mathbf{R}} = \mathbf{R}\hat{\boldsymbol{\omega}}, \qquad \mathbf{J}\dot{\boldsymbol{\omega}} = \mathbf{J}\boldsymbol{\omega} \times \boldsymbol{\omega} + \boldsymbol{\tau}_u, \tag{1b}$$

where $\hat{\cdot}$ denotes a skew-symmetric mapping $\mathbb{R}^3 \to \mathfrak{so}(3)$; $\mathbf{g} = (0, 0, -g)^T$ is the gravity vector; $\boldsymbol{f}_u = (0, 0, f)^T$ and $\boldsymbol{\tau}_u = (\tau_x, \tau_y, \tau_z)^T$ are the total thrust and body torques from the rotors, respectively. The total wrench vector applied on the center-of-mass (CoM) of the multirotor's body is defined as $\boldsymbol{\eta} = (f, \tau_x, \tau_y, \tau_z)^T$. The total wrench vector $\boldsymbol{\eta}$ is linearly related to the squared motor rotational rate (i.e., propeller speed) for k motors $\boldsymbol{\omega_m} = (w_1^2, \ldots, w_k^2)^T$, such that $\boldsymbol{\eta} = \mathbf{B}_0\boldsymbol{\omega_m}$, where $\mathbf{B}_0$ is the actuation matrix [24].

B. Full System Dynamics

Consider a team of *n* multirotors that are connected to a payload through massless, and non-extensible cables (see Fig. 1). In the following, we summarize results presented in [3, 4] using our own notation for clarity. Throughout this work, the variables related to the payload are denoted by the subscript 0, and the variables for the *i*-th multirotor are denoted by the subscript $i \in \{1, \ldots, n\}$. The payload is described as a rigid body with mass m_0 and moment of inertia matrix $\mathbf{J}_0$. The cables are assumed to be always taut (i.e., modeled as rigid rods) each with length l_i. The states can be described as the global position $\mathbf{p}_0 \in \mathbb{R}^3$ and the velocity $\dot{\mathbf{p}}_0 \in \mathbb{R}^3$ of the payload; the attitude rotation matrix $\mathbf{R}_0 \in SO(3)$ and the body angular velocity $\boldsymbol{w}_0 \in \mathbb{R}^3$ of the payload; the unit directional vector $\mathbf{q}_i \in \mathbb{R}^3$ and the angular velocity $\boldsymbol{\omega}_i \in \mathbb{R}^3$ of each *i*-th cable, where $\mathbf{q}_i$ points from multirotor *i* towards the payload. Hence, the state space has dimension $13 + 6n$.

Let us define the attachment point vectors between each cable and the payload as ${}^0\mathbf{p}_{a_i}$ expressed in the payload reference frame. Given the states of the full system, the position of each multirotor expressed in the fixed global frame is

$$\mathbf{p}_i = \mathbf{p}_0 + \mathbf{R}_0{}^0\mathbf{p}_{a_i} - l_i\mathbf{q}_i. \tag{2}$$

Thus, the kinematics of the full system is described as

$$\dot{\mathbf{q}}_i = \boldsymbol{\omega}_i \times \mathbf{q}_i, \qquad \dot{\mathbf{p}}_i = \dot{\mathbf{p}}_0 + \dot{\mathbf{R}}_0{}^0\mathbf{p}_{a_i} - l_i\dot{\mathbf{q}}_i$$
$$\dot{\mathbf{R}}_0 = \mathbf{R}_0\hat{\boldsymbol{\omega}}_0, \tag{3}$$

and the nonlinear dynamics of the full system can be expressed using Euler-Lagrange equations [3, 4].

C. Control Design

1) Single Multirotor without Payload: The goal for a single multirotor controller is to compute propeller speeds such that a given a tuple reference trajectory $\langle \mathbf{p}_r, \dot{\mathbf{p}}_r, \ddot{\mathbf{p}}_r, \boldsymbol{\psi}_r \rangle$ for the multirotor CoM is tracked. Here, $\mathbf{p}_r$ and $\boldsymbol{\psi}_r$ are the position and the heading of the multirotor, respectively. The controller in [25] consists of a 2-*loop* cascaded design, where

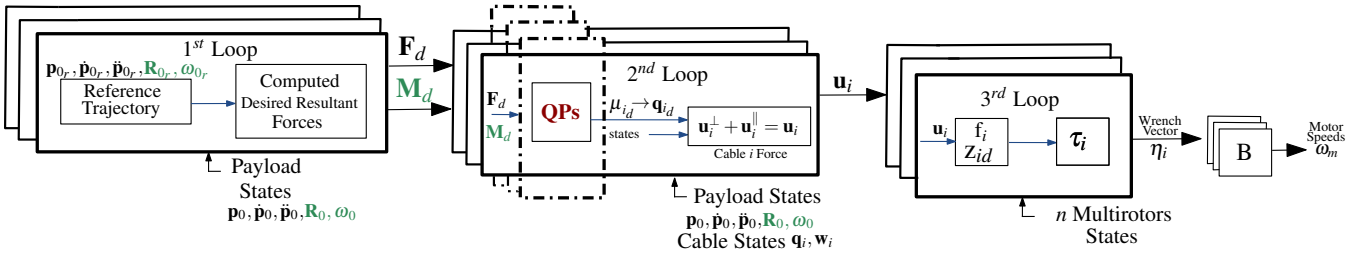

Fig. 2. Controller architecture of the state-of-the-art geometric controller for payload transport. We contribute an optimization-based cable force allocation (shown in red). Green states are only needed when transporting a rigid body rather than a point mass.

the outer loop computes the desired force control vector, which then computes f and the desired third body z-axis z_d. Thus, using differential flatness along with the reference heading $\boldsymbol{\psi}_r$, the desired rotational states are computed and $\boldsymbol{\tau}_u$ is used to track those desired states [24, 25].

2) Control Design for Rigid Payload Dynamics: For the multi-UAVs payload transport system, a new controller design is needed since the single control design does not include the dynamics of the cables.

The geometric controller by [3] solves the multi-UAVs payload transport problem, where the control problem is defined as follows. The input is a tuple reference trajectory $\langle \mathbf{p}_{0_r}, \dot{\mathbf{p}}_{0_r}, \ddot{\mathbf{p}}_{0_r}, \mathbf{R}_{0_r}, \boldsymbol{\omega}_{0_r} \rangle$, where $\mathbf{p}_{0_r}, \mathbf{R}_{0_r}, \boldsymbol{\omega}_{0_r}$ are the position, rotation matrix and angular velocity of the payload, respectively. The output is a decentralized controller that computes the motor signals for each robot.

a) Desired Payload Forces and Moments: This controller consists of a 3-*loop* cascaded design as shown in Fig. 2. The first step in the first loop computes the desired resultant payload control forces, $\mathbf{F}_d$, and moments, $\mathbf{M}_d$, to track the payload reference trajectory. Then, the first loop outputs the desired cable unit directional vectors $\mathbf{q}_{i_d}$ in order to achieve $\mathbf{F}_d$ and $\mathbf{M}_d$. Let $\boldsymbol{\mu}_{i_d}$ be the desired i-th cable force and $\mathbf{P} \in \mathbb{R}^{6\times 3n}$ be a matrix that maps the forces and torques of the payload CoM to desired cable forces $\boldsymbol{\mu}_{i_d}$ on the attachment points $\mathbf{p}_{a_i}$. Thus, $\mathbf{P}$ is a constant matrix structured from $\mathbf{p}_{a_i}$, refer to [3] for details. The cable force allocation relation between $\boldsymbol{\mu}_{i_d}$ and $\mathbf{F}_d, \mathbf{M}_d$ is defined as

$$\mathbf{P}\left((\mathbf{R}_0^T\boldsymbol{\mu}_{1_d})^T \dots (\mathbf{R}_0^T\boldsymbol{\mu}_{n_d})^T\right) = \left((\mathbf{R}_0^T\mathbf{F}_d)^T \quad \mathbf{M}_d^T\right). \quad (4)$$

b) Cable Force Allocation: For any $\mathbf{F}_d$, $\mathbf{M}_d$ and $n \geq 3$, there exist desired cable forces $\boldsymbol{\mu}_{i_d}$. These desired cable forces can be computed using the Moore-Penrose inverse of $\mathbf{P}$ to achieve a least-square solution:

$$\boldsymbol{\mu}_{i_d} = \text{diag}(\mathbf{R}_0, \dots, \mathbf{R}_0)\mathbf{P}^T(\mathbf{P}\mathbf{P}^T)^{-1}\begin{pmatrix}\mathbf{R}_0^T\mathbf{F}_d \\ \mathbf{M}_d\end{pmatrix}. \quad (5)$$

3) Control Design for Point Mass Dynamics: Let us consider the special case of the payload being a point mass with $n \geq 2$, then the controller architecture retains the same structure, see Fig. 2. However, [4, 14] solve the allocation of the cable forces $\boldsymbol{\mu}_{i_d}$ by providing a user-specified team formation which is then projected on the null space of the cable forces to satisfy the following

$$\mathbf{P}\begin{pmatrix}\boldsymbol{\mu}_{1_d}^T & \dots & \boldsymbol{\mu}_{n_d}^T\end{pmatrix}^T = \mathbf{F}_d. \quad (6)$$

4) Control Inputs of the UAVs: Let the control force applied by each i-th multirotor on its cable be $\mathbf{u}_i = \mathbf{R}_i\mathbf{f}_{u_i}$. Denote that $\mathbf{u}_i^{\parallel} \in \mathbb{R}^3$ and $\mathbf{u}_i^{\perp} \in \mathbb{R}^3$ are the orthogonal projection of $\mathbf{u}_i$ along $\mathbf{q}_i$ and to the plane normal to $\mathbf{q}_i$, respectively, i.e., $\mathbf{u}_i = \mathbf{u}_i^{\parallel} + \mathbf{u}_i^{\perp}$. We compute $\mathbf{q}_{i_d}$ using the desired cable forces $\boldsymbol{\mu}_{i_d}$ by applying the definition

$$\mathbf{q}_{i_d} = \frac{\boldsymbol{\mu}_{i_d}}{\|\boldsymbol{\mu}_{i_d}\|}. \quad (7)$$

The second loop computes the control force $\mathbf{u}_i$ applied by each multirotor on the payload. In particular, $\mathbf{u}_i^{\parallel}$ is first computed by projecting $\boldsymbol{\mu}_{i_d}$ on the current cable force vector $\boldsymbol{\mu}_i$, in addition to non-linear terms that linearize the translational dynamics of the payload. In other words, when $\boldsymbol{\mu}_i \to \boldsymbol{\mu}_{i_d}$, then $\mathbf{F}_d, \mathbf{M}_d$ are tracked. In order to track the desired cable forces $\boldsymbol{\mu}_{i_d}$, the vector $\mathbf{u}_i^{\perp}$ is used to track the desired unit directional vector, i.e., $\mathbf{q}_i \to \mathbf{q}_{i_d}$, of the cables. When $\mathbf{q}_i = \mathbf{q}_{i_d}$ then the resultant force and moment acting on the payload become identical to their desired values.

After computing $\mathbf{u}_i$, the final third loop computes the thrust magnitude f (i.e., $\mathbf{f}_u$) and the desired third body vector $\mathbf{z}_d$. Thus the attitude control input $\boldsymbol{\tau}_u$ can be computed using $\mathbf{f}_u$ as presented in Section III-C.1.

D. Quadratic Programs and Hyperplanes

A quadratic program (QP) is an optimization problem with a quadratic objective and affine equality and inequality constraints. The QP is formulated as

$$\min_{\boldsymbol{x}} \quad \frac{1}{2}\boldsymbol{x}^T\boldsymbol{P}\boldsymbol{x} + \boldsymbol{q}^T\boldsymbol{x} \quad (8)$$
$$\text{s.t.} \left\{\underline{\boldsymbol{u}} \leq \boldsymbol{A}\boldsymbol{x} \leq \overline{\boldsymbol{u}} \quad ,\right.$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the decision variable vector. The objective function is defined by a positive semi-definite matrix $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{q} \in \mathbb{R}^n$. The linear constraints are defined by matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and the lower and upper bound vectors are $\underline{\boldsymbol{u}}$ and $\overline{\boldsymbol{u}}$ respectively, such that $\underline{u}_i \in \mathbb{R} \cup \{-\infty\}$ and $\overline{u}_i \in \mathbb{R} \cup \{+\infty\}$ for all $i \in \{1, \dots, m\}$. QPs are convex and thus have a global minima.

A hyperplane $\mathcal{H}$ in $\mathbb{R}^d$ can be formulated by a normal vector $\mathbf{n}$ and an offset a as $\mathcal{H} = \{\boldsymbol{x} \in \mathbb{R}^d \mid \mathbf{n}^T\boldsymbol{x} - a = 0\}$. A half-space $\tilde{\mathcal{H}}$ in $\mathbb{R}^d$ is a subset of $\mathbb{R}^d$ that is bounded by a hyperplane such that $\tilde{\mathcal{H}} = \{\boldsymbol{x} \in \mathbb{R}^d \mid \mathbf{n}^T\boldsymbol{x} - a \leq 0\}$. The intersection of multiple hyperspaces creates a polyhedra. Hyperplanes can be used as a linear constraints for a QP formulation.

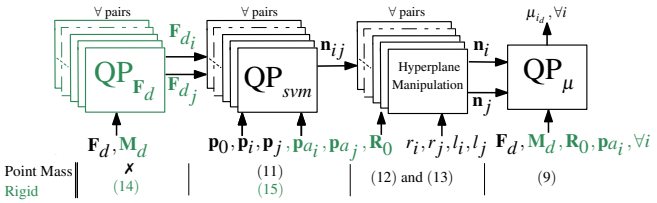

Fig. 3. A cascade of three QPs allocate the desired cable forces $\boldsymbol{\mu}_{i_d}$ for $n \geq 2$ robots. The final QP, QP$_{\boldsymbol{\mu}}$, computes $\boldsymbol{\mu}_{i_d}$ by constraining the cable forces to fulfill the allocation constraints (6) (point mass) or (4) (rigid body) and limiting them to be inside a polyhedra. This volume is defined by a set of hyperspaces, which are constructed by two other QPs that are solved for each robot pair, QP$_{svm}$ and QP$_{\mathbf{F}_d}$. Black represents both payload types and green is exclusively used for the rigid payload.

IV. Approach

A. Overview

We augment the existing geometric controller described in Section III-C with a custom force allocation method (replacing the standard approach presented in Section III-C.2.b) to achieve trajectories for the multirotors that avoid inter-robot collisions. Our approach employs a unifying framework that is capable of handling point mass and rigid payloads. As such, we provide a drop-in replacement for (5), that is we use the desired force on the payload, $\mathbf{F}_d$, and the desired moments for the payload, $\mathbf{M}_d$, as input and compute the desired cable forces, $\boldsymbol{\mu}_{i_d}$, while taking inter-robot collision avoidance into account. The nonlinear optimization problem formulation can be defined as,

$$\min_{\boldsymbol{\mu}_{i_d}} \ \frac{1}{2}\|\boldsymbol{\mu}_{i_d}\|^2 \tag{9}$$
$$\text{s.t.} \begin{cases} \text{(4) or (6)} \\ \|\mathbf{p}_i(\boldsymbol{\mu}_{i_d}, \cdot) - \mathbf{p}_j(\boldsymbol{\mu}_{j_d}, \cdot)\| \geq r_i + r_j, \quad \forall i \neq j, \end{cases}$$

where $\mathbf{p}_i(\boldsymbol{\mu}_{i_d}, \cdot)$ and $\mathbf{p}_j(\boldsymbol{\mu}_{j_d}, \cdot)$ are the positions of each robot pair (see (2),(7)) and r_i, r_j are the safety radii. Consider the tracking of the desired cable directions $\mathbf{q}_{i_d}$ is fast enough such that $\mathbf{q}_i = \mathbf{q}_{i_d}$. The first constraint enforces the force allocation constraints for the point mass or rigid body payload case, respectively. The second constraint ensures that the safety distance between each pair of i and j robots is at least $r_i + r_j$. Without the second constraint, the optimal solution for the desired cable forces is the same as in (5). Instead of using a nonlinear optimization that solves for $\boldsymbol{\mu}_{i_d}$ subject to the second nonconvex constraint, we use a cascaded design of three quadratic programs (QPs) to link these variables implicitly, see Fig. 3. The first three blocks in Fig. 3 run in a decentralized manner on each multirotor to compute all hyperplanes for all robot pairs. Thus, for n multirotors, these blocks are executed $n(n-1)/2$ times.

Our key insight is to construct hyperplanes that separate each pair of multirotors. These hyperplanes should also take the desired motion of the payload (i.e., $\mathbf{F}_d, \mathbf{M}_d$) into account. For each pair we use the first block, QP$_{\mathbf{F}_d}$, to provide heuristics of desired cable forces ($\mathbf{F}_{d_i}$ and $\mathbf{F}_{d_j}$) that approximate the actual solution to achieve $\mathbf{F}_d, \mathbf{M}_d$, which will guide the hyperplane generation. The next block, QP$_{svm}$, constructs a hyperplane to separate the two corresponding cables with a preference for the approximate solutions $\mathbf{F}_{d_i}$ and $\mathbf{F}_{d_j}$ (i.e., rotating or offsetting the hyperplane). Since the resulting hyperplane between each pair has a small safety margin, we use geometric hyperplane manipulation (third block) to construct two new hyperplanes, one for each multirotor of the pair, that possess the desired safety distance and constrains the desired cable forces $\boldsymbol{\mu}_{i_d}$. Note that this geometric manipulation is computationally better compared to solving one optimization problem per hyperplane.

Finally, the last block QP$_{\boldsymbol{\mu}}$ runs once on each robot to allocate the desired cable forces $\boldsymbol{\mu}_{i_d}$ for the full cable system, and each multirotor extracts its own solution from QP$_{\boldsymbol{\mu}}$. This QP always finds the same solution for the same problem input when $n \geq 2$, since it has one global minima only.

B. QP For Desired Cable Forces (QP$_{\boldsymbol{\mu}}$)

The minimum-norm solution of (4) or (6) provides a solution for $\boldsymbol{\mu}_{i_d}$. We add additional (hard) constraints to the optimization. The resulting optimization problem is a QP due to the linear constraints and can be solved efficiently:

$$\min_{\boldsymbol{\mu}_{i_d}} \ \frac{1}{2}\|\boldsymbol{\mu}_{i_d}\|^2 \tag{10}$$
$$\text{s.t.} \begin{cases} \text{(4) or (6)} \\ \begin{pmatrix} \mathbf{n}_1^T & \dots & 0 \\ 0 & \ddots & 0 \\ 0 & \dots & \mathbf{n}_m^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\mu}_{1_d} \\ \vdots \\ \boldsymbol{\mu}_{n_d} \end{pmatrix} - \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \leq \mathbf{0}_{m \times 1} \end{cases},$$

where $\mathbf{n}_i \in \mathbb{R}^{3\times 1}$ and $a_i \in \mathbb{R}$ for $i \in 1, \dots, m$ define the half-spaces and m is the number of hyperplanes with $m \geq n$.

C. Point Mass Model

In the QP$_{\boldsymbol{\mu}}$ formulation, our inequality constraints are formulated using hyperplanes. We compute these hyperplanes by solving another QP followed by geometric manipulation to construct a constraint for the cable forces. Consider each pair of robots i and j. Their positions are ${}^0\mathbf{p}_i$ and ${}^0\mathbf{p}_j$ in the payload frame of reference. For better readability, we omit the superscripts of the robot position vectors in the payload frame, and use them as $\mathbf{p}_i$ and $\mathbf{p}_j$.

1) QPs for Hyperplanes (QP$_{svm}$): For each pair of robots we generate two hyperplanes $\mathbf{n}_i$ and $\mathbf{n}_j$, see Fig. 4 on the left. In order to generate $\mathbf{n}_i, \mathbf{n}_j$, we first compute for each pair the hyperplane $\mathbf{n}_{ij}$ that separates both robots while being as close as possible to $\mathbf{F}_d$. Thus, $\mathbf{n}_{ij}$ maximizes the safety margin between robots positions while regularizing for $\mathbf{F}_d$. This regularization term guarantees that $\boldsymbol{\mu}_{i_d}$ can be tracked well by allowing the cables to move in the same direction of $\mathbf{F}_d$. Afterwards, $\mathbf{n}_{ij}$ is used to compute both $\mathbf{n}_i$ and $\mathbf{n}_j$ to be used in QP$_{\boldsymbol{\mu}}$. We adopt a variation of multi-objective hard-margin support vector machines (SVM) with additional regularization for $\mathbf{F}_d$ as QP$_{svm}$:

$$\min_{\mathbf{n}_{ij}} \ \|\mathbf{n}_{ij}\|^2 + \lambda_s(\mathbf{n}_{ij}^T\mathbf{F}_d)^2 \tag{11}$$
$$\text{s.t.} \begin{cases} \mathbf{n}_{ij}^T\mathbf{p}_i \leq -1, \quad \mathbf{n}_{ij}^T\mathbf{p}_j \geq 1 \end{cases}.$$

Here, λ_s is a weighting factor for the soft constraint that acts as a trade-off between the safety margin and the hyperplane of $\mathbf{n}_{ij}$ being as close to $\mathbf{F}_d$. In fact, a high λ_s might result in a safety margins that are close to zero. Our formulation implicitly constrains the plane to pass through the payload by not including an offset a in the QP$_{svm}$. If such an offset is included, then the hyperplanes might be shifted and consequently, this might render QP$_{\boldsymbol{\mu}}$ infeasible.

2) Hyperplane Manipulation: After computing the separating hyperplane $\mathbf{n}_{ij}$, we need to compute $\mathbf{n}_i$ and $\mathbf{n}_j$ normals of the hyperplanes as constraints for the QP in (10), see Fig. 4. It is worth noting that the $\mathbf{n}_{ij}$ vector is always directed towards $\mathbf{p}_i$. Given the radius of each robot r_i and r_j and the length of each cable l_i and l_j, we first compute the angles α_i and α_j by using the properties of the resulting isosceles triangle as

$$\alpha_i = 2\arcsin\left(\frac{r_i}{2l_i}\right), \quad \alpha_j = 2\arcsin\left(\frac{r_j}{2l_j}\right). \tag{12}$$

Let us define $\mathbf{e}_3 = (0,0,1)^T$ and define quaternions $\mathbf{q}_i$ and $\mathbf{q}_j$ by converting the axis-angle representation to a quaternion using the axis $\mathbf{n}_{ij} \times \mathbf{e}_3$ with the angles α_i and $-\alpha_j$. Finally, we compute the normals $\mathbf{n}_i$ and $\mathbf{n}_j$ by tilting $\mathbf{n}_{ij}$ with the computed $\mathbf{q}_i$ and $\mathbf{q}_j$. Let us define $\odot$ as the quaternion rotation operator, then $\mathbf{n}_i$ and $\mathbf{n}_j$ are

$$\mathbf{n}_i = \mathbf{q}_i \odot \mathbf{n}_{ij}, \quad \mathbf{n}_j = -\mathbf{q}_j \odot \mathbf{n}_{ij}. \tag{13}$$

Note that it is required by QP$_{\boldsymbol{\mu}}$ in (10) that both normals must be pointing inwards (i.e., towards each other), which explains the negative sign added to (13) for the $\mathbf{n}_j$ normal vector computation.

D. Rigid Body Model

Similar to the point mass model (see Section IV-C), our objective for the rigid payload is to find the hyperplanes $\mathbf{n}_i$ and $\mathbf{n}_j$ (Fig. 4 on the right) that set the desired safety distances between the robots. However, in the rigid payload case the hyperplane $\mathbf{n}_{ij}$ (which computes $\mathbf{n}_i$ and $\mathbf{n}_j$) needs to not only take into account being close to $\mathbf{F}_d$, but also the desired moments $\mathbf{M}_d$. Thus, we propose a special QP for the rigid payload case, QP$_{\mathbf{F}_d}$ (see Fig. 3). Here, QP$_{\mathbf{F}_d}$ includes $\mathbf{M}_d$ to generate the $\mathbf{n}_{ij}$ hyperplane.

1) QPs for Force Pairs (QP$_{\mathbf{F}_d}$): We first solve for each pair of robots QP$_{\mathbf{F}_d}$, which computes virtual heuristic forces $\mathbf{F}_{d_i}$ and $\mathbf{F}_{d_j}$ to be applied on the attachment points $\mathbf{p}_{a_i}$ and $\mathbf{p}_{a_j}$. Recall that the attachment points of each cable are $\mathbf{p}_{a_i}, \mathbf{p}_{a_j}$ in the payload reference frame. The sum of these forces must achieve $\mathbf{F}_d$ while producing moments close to $\mathbf{M}_d$. Let us define $\hat{\mathbf{p}}_{a_i}$ and $\hat{\mathbf{p}}_{a_j}$ as the skew-symmetric mapping of $\mathbf{p}_{a_i}$ and $\mathbf{p}_{a_j}$, then $\mathbf{F}_{d_i}$ and $\mathbf{F}_{d_j}$ can be solved as

$$\begin{aligned} \min_{\mathbf{F}_{d_i},\mathbf{F}_{d_j}} \ & \|\mathbf{F}_{d_i}\|^2 + \|\mathbf{F}_{d_j}\|^2 + \\ & \|\mathbf{M}_d - (\hat{\mathbf{p}}_{a_i}\mathbf{R}_0^T\mathbf{F}_{d_i} + \hat{\mathbf{p}}_{a_j}\mathbf{R}_0^T\mathbf{F}_{d_j})\|^2 \\ \text{s.t.} \ & \mathbf{F}_{d_i} + \mathbf{F}_{d_j} = \mathbf{F}_d. \end{aligned} \tag{14}$$

Since the output of QP$_{\mathbf{F}_d}$ are heuristic virtual forces $\mathbf{F}_{d_i}$ and $\mathbf{F}_{d_j}$, we do not need to track $\mathbf{M}_d$ as a hard constraint. In particular, the mapping between $(\mathbf{F}_d, \mathbf{M}_d)$ and $(\mathbf{F}_{d_i}, \mathbf{F}_{d_j})$ might be infeasible depending on the attachment points positions $(\mathbf{p}_{a_i}, \mathbf{p}_{a_j})$. Thus, we prioritize to solve for $\mathbf{F}_d$ by considering it as a hard constraint and add a soft constraint to have a solution as close as possible to $\mathbf{M}_d$.

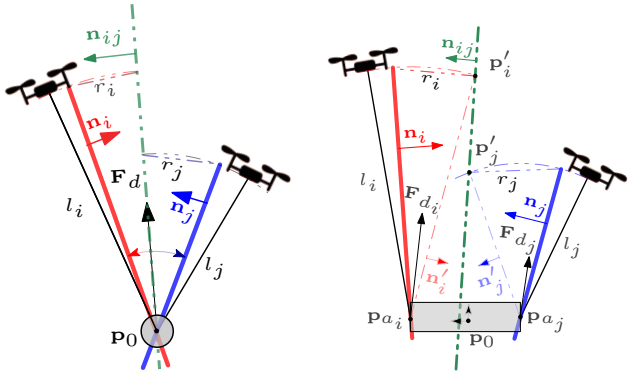

Fig. 4. 2D projection of the hyperplane manipulation, where QP$_{svm}$ computed $\mathbf{n}_{ij}$. Left: Point mass case, where $\mathbf{n}_{ij}$ is rotated around $\mathbf{p}_0$ such that the distance to $\mathbf{n}_{ij}$ is r_i or r_j. Right: Rigid body case, where new intermediate hyperplanes n'_i and n'_j are constructed followed by the same rotation as in the point mass case.

2) QPs for Hyperplanes (QP$_{svm}$): After solving for $\mathbf{F}_{d_i}$ and $\mathbf{F}_{d_j}$, we use a hybrid soft-hard margin SVM. In particular, we solve for the normal of the hyperplane $\mathbf{n}_{ij}$ and an offset a. Since there is no particular desired intersection point between the $\mathbf{n}_{ij}$ hyperplane and the rigid payload, a is a decision variable. Accordingly, we propose QP$_{svm}$ as

$$\min_{\mathbf{n}_{ij},a,s_1,s_2} \ \|\mathbf{n}_{ij}\|^2 + \lambda_s(s_1+s_2) \tag{15}$$
$$\text{s.t.} \begin{cases} \mathbf{n}_{ij}^T\mathbf{p}_i - a \le -1, \quad \mathbf{n}_{ij}^T\mathbf{p}_j - a \ge 1 \\ \mathbf{n}_{ij}^T\mathbf{p}_{a_i} - a \le -1, \quad \mathbf{n}_{ij}^T\mathbf{p}_{a_j} - a \ge 1 \\ \mathbf{n}_{ij}^T(\mathbf{p}_{a_i} + \mathbf{F}_{d_i} - a) \le -1 + s_1 \\ \mathbf{n}_{ij}^T(\mathbf{p}_{a_j} + \mathbf{F}_{d_j} - a) \ge 1 - s_2 \\ s_1 \ge 0, \quad s_2 \ge 0 \end{cases}.$$

Here, the first four constraints separate the cables like in a hard-margin SVM. The next four constraints assure that the hyperplane $\mathbf{n}_{ij}$ minimizes the projection of both $\mathbf{F}_{d_i}$ and $\mathbf{F}_{d_j}$ after being shifted on the attachment points $\mathbf{p}_{a_i}, \mathbf{p}_{a_j}$, respectively. We use two slack variables s_1 and s_2 that are minimized to relax the hard constraints imposed on the projection of $\mathbf{F}_{d_i}$ and $\mathbf{F}_{d_j}$ on the hyperplane of $\mathbf{n}_{ij}$. This part is similar to a soft-margin SVM. Similar to the point mass case, λ_s also acts as a trade-off between the safety margin and being close to both forces. The hyperplane does not have to pass through $\mathbf{p}_0$ since a is a decision variable.

3) Hyperplane Manipulation: We use the same method for tilting as in the point mass case (Section IV-C.2) with extra steps. Let us define two points $\mathbf{p}'_i$ and $\mathbf{p}'_j$ as shown in Fig. 4 on the right. We first compute the intersection between the sphere that describes the motion of each cable and the hyperplane $\mathbf{n}_{ij}$, which is a circle on $\mathbf{n}_{ij}$. We use the points of the circle with the highest z-coordinate as $\mathbf{p}'_i$ and $\mathbf{p}'_j$. Afterwards, two new hyperplanes are computed as $\mathbf{n}'_i$ and $\mathbf{n}'_j$ by using vector $\mathbf{n}_{ij} \times \mathbf{e}_3$ and two points for each

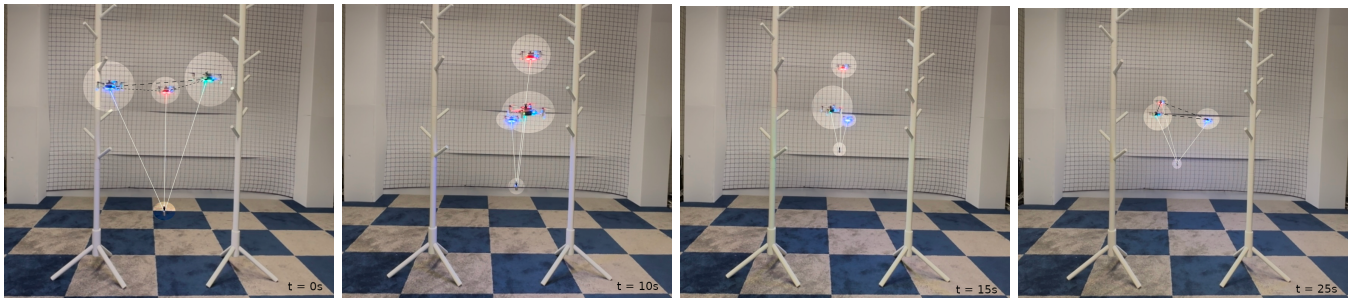

Fig. 5. Four frames (manually highlighted for better visibility) of three multirotors carrying a point mass payload in a teleoperation use-case while avoiding inter-robot collisions. There are two obstacles forming a narrow passage. The operator controls the payload position as well as the desired formation. Here, a line configuration as shown in $t = 10s$ and $t = 15s$ is used. After passing the obstacles, the operator disables the manual formation control and our approach computes to a more energy-efficient triangle formation ($t = 25s$).

plane, which are the intersection points $\mathbf{p}'_i$ and $\mathbf{p}'_j$ and the attachment points $\mathbf{p}_{a_i}, \mathbf{p}_{a_j}$, respectively. Finally, we apply the same steps from the point mass case to tilt $\mathbf{n}'_i$ and $\mathbf{n}'_j$ based on the radii r_i and r_j of each robot model to generate $\mathbf{n}_i$ and $\mathbf{n}_j$ using (12) and (13).

E. Desired Cable Forces

One of the main advantages of the $\text{QP}_{\boldsymbol{\mu}}$ formulation and in particular the choice of representing the inequality constraints are the extensions that provide new use-cases. Here, we provide one such example to provide desired cable forces. There are two motivations behind supporting this feature. The first motivation is to steer the optimization problem to have consecutive solutions that avoid sudden jumps. The second motivation is to allow the user to specify a desired formation configuration, for example to pass through a narrow passage.

Let us define $\boldsymbol{\mu}_{i_0}$ as preferable desired cable forces, as either the solution from the previous optimization or the user-specified desired values. In the latter case, we can rescale $\boldsymbol{\mu}_{i_0}$ such that $\sum_i \boldsymbol{\mu}_{i_0} = \mathbf{F}_d$. The cost function in $\text{QP}_{\boldsymbol{\mu}}$ (10) can be modified to minimize the difference between the desired cable forces and the preferred ones with a weighting factor λ as

$$c = \frac{1}{2}\|\boldsymbol{\mu}_{i_d}\|^2 + \lambda\|\boldsymbol{\mu}_{i_0} - \boldsymbol{\mu}_{i_d}\|^2. \tag{16}$$

V. Experiments

We outline our experimental setup and first describe our steps to bridge the sim-to-real gap. For the real platform, we use multirotors of type Bitcraze Crazyflie 2.1 (CF). These are small ($9\,\text{cm}$ rotor-to-rotor) and lightweight ($34\,\text{g}$) products that are commercially available. The physical parameters are identified in prior work [26]. We use the existing extended Kalman filter for state estimation and implement our control algorithm in *C* to run directly on-board the STM32-based flight controller ($168\,\text{MHz}$, $192\,\text{kB}$ RAM).

We use a four-staged **iterative development method** to reduce challenges regarding the sim-to-real gap. In the first stage, we implement our experiments in simulation using only *Python*. For our controller, we write the QP program using an open source Python-embedded modeling language for convex optimization problems, CVXPY [27]. In the second stage, we port our controller from *Python* to *C* and add it to the multirotor's firmware. We verify the semantic equivalence in our simulator by generating *Python* bindings (Software-in-The-Loop, SITL). In the third stage, we port our own controller, which requires to solve multiple QPs (10) on-board the STM32 microcontroller. The primary challenge is to migrate our QP formulations from CVXPY to *C*, for which we rely on OSQP [28], which can generate code that is optimized for embedded systems. As before, we validate this implementation in simulation in a SITL-fashion. In the fourth stage, we optimize the code to be able to run in realtime on the physical hardware. To this end, we limit ourselves to single-precision floating point operations, solve QPs asynchronously from the control loop, and warm start the optimization with the previous results.

We **validate** the functionality of our proposed approach through especially designed physical flight experiments, which highlight that a fatal crash would occur in the absence of our approach that accounts for inter-robot collisions and small in size payloads, while tracking the payload trajectory.

We verify our approach in physical flight tests with up to three Crazyflies. Each multirotor has its own on-board controller, solving the QPs and using the computed desired cable forces for itself to generate its own control input for the motors. Such a distributed execution requires that all robots compute identical values. Since our QPs have a global minimum that the solver reaches, we just need to ensure that all multirotors use the same input data. On the host side, we use Crazyswarm2, which is based on Crazyswarm [29] but uses ROS 2 [30] to control and send commands for multiple Crazyflies. Crazyswarm2 heavily relies on broadcast communication for state estimation and control commands, which ensures that all robots receive the data at the same time. In particular, we equip each multirotor with a single reflective marker for position tracking at $100\,\text{Hz}$ using an OptiTrack motion capture system. For the payload, we use a single marker for the point mass or four markers to track the rigid body payload. Multirotors and payload states are sent with broadcasts to all robots. For commands such as takeoff, land, or trajectory execution we also rely on broadcasts to ensure consistent desired states between multirotors.

The states of each multirotor are estimated on-board using an extended Kalman filter and the first derivatives of the

TABLE I
MEAN AND STANDARD DEVIATION VALUES (SMALL GRAY) OF THE CONTROL LOOP RUNTIME IN MILLISECONDS.

Robots	3	6	8	10
Distributed $t_{\mathbf{total}}$	0.1 0.0	1.0 0.2	2.0 0.3	3.6 0.9
Centralized $t_{\mathbf{c}}$	0.1 0.0	3.9 0.6	10.9 0.6	20.8 3.4
NMPC [6]: $t_{\mathbf{c}}$	7.0	30.0	55.0	100.0

payload state using numeric differentiation in combination with a complimentary filter. For the payload, we use a calibration weight as point mass, cardboard and 3D-printed objects as rigid bodies, and dental floss as cables. We use magnets to connect the cables and the payload/multirotors to be easily repaired. Our approach does not require measuring or estimating the cable forces, since the controller tracks $\mathbf{q}_{i_d}$.

We highlight through our simulation results the superiority of our method in scalability and computational efficiency compared to state-of-the-art controllers.

A. Experimental Results

We conduct several real flight experiments, see Table II. For each case, we report the runtime of our optimization (as executed on-board the microcontroller) and the payload pose error. To the best of our knowledge, existing optimization-based controllers (e.g., [5, 6]) that can operate in similar use-cases have not been implemented on a highly constrained computing platform, such as the STM32-based flight controller. They require more sophisticated solvers, which makes it almost impossible to do so. Moreover, executing controllers off-board for comparison introduces significant latencies, invalidating the results. Consequently, a direct performance comparison with such controllers is currently not possible.

We consider two different types of reference motions: a polynomial figure-8 trajectory (reaching $0.5\,\mathrm{m/s}$ for the point mass and $0.4\,\mathrm{m/s}$ for the rigid body cases), and teleoperation with position and velocity commands for the payload. For the teleoperation experiment, we put obstacles as shown in Fig. 5 such that the robots switch to a line configuration (see Section IV-E) to avoid collisions. The operator was able to switch to a predefined configuration by pressing a button. For the other experiments, we use two types of rigid payloads and different number of Crazyflies and cable lengths.

Overall, we are able to compute desired cable forces at $30\,\mathrm{Hz}$ on-board the microcontroller, which is sufficient for robust pose tracking of the payload. Surprisingly, the overhead of setting up the QPs takes also a significant amount of time (e.g., $13\,\mathrm{ms}$ in the 3 UAVs triangle case). The obtained pose errors are consistent with prior work using bigger multirotors [7], for positions in the range of the rotor-to-rotor size of the UAV. The yaw error is higher in the 2 UAVs rod case, because of the challenging nature of our yaw setpoints, that require rotating the formation while flying the figure-8 motion.

B. Scalability

To assess the team size scalability, we record the computation time on a laptop (i7-1165G7, $2.8\,\mathrm{GHz}$) of the three QPs in Fig. 3 for up to 10 robots. Compared to [6], we report the runtimes that were obtained on a comparable laptop. Our first metric is the total time of the full control loop running in distributed fashion as $t_{\mathbf{total}} = t_{\mathbf{svm}} + t_{\mathbf{F}_d} + t_{\boldsymbol{\mu}} + t_{\mathbf{ctrl}}$, where $t_{\mathbf{ctrl}}$ is the time of all the other computations. As shown in Table I, the distributed time scales roughly quadratically with the number of robots and the overall runtime is at least 1.4 orders of magnitude faster than the results in [6].

Additionally, since [6] is a centralized optimization, we consider the centralized version of our method as the second metric. We compute the time over n robots over the QPs as $t_{\mathbf{c}} = nt_{\mathbf{svm}} + nt_{\mathbf{F}_d} + t_{\boldsymbol{\mu}}$. We are still significantly faster than [6] with a runtime reduction by a factor of five.

C. Challenges

Implementing the control framework presented in Fig. 2 in simulation posed a significant challenge. The key difficulty was in identifying the control gains that lie in one of the local optima that is robust enough to balance between the control performance and addressing physical uncertainties upon transfer to the actual platform. Moreover, executing the full control framework on the physical multirotors in real flights had additional challenges. In particular, the geometric controller [3] contained several parts that are numerically unstable for physical flights. The second loop of the controller computes $\mathbf{u}_i$, which requires either measuring or estimating the acceleration of the payload. The numeric estimate of this value is very noisy and causes the multirotors to crash. Instead, we rely on the reference acceleration of the payload. Similarly, the change of the cable unit direction $\dot{\mathbf{q}}_{i_d}$ is too noisy in practice and we use $\mathbf{0}$ instead.

We also found that tuning the gains of the cascaded design is very challenging, even with access to all relevant tracking errors. The gains are very sensitive and depend on the number of UAVs and the type of payload. Moreover, recovering from the disturbances that occur during takeoff poses another challenge especially for tuning. We note that switching from the single multirotor controller to our controller midflight is a possible alternative, but it is not practical for non-uniform cable lengths. Some of these challenges might be easier to overcome on platforms with a higher thrust-to-weight ratio than ours with a low ratio of about 1.4, although our data indicates that motors did only occasionally saturate during takeoff in our experiments. In contrast, the two new gains our method introduces, λ and λ_s, are easy to tune. Both hyperparameters are tuned in our iterative development workflow from sim-to-real in different settings.

VI. CONCLUSION AND FUTURE WORK

We present an efficient optimization-based cable force allocation of a geometric controller for cable-suspended payload transportation that is aware of neighboring robots to avoid collisions. Unlike previous work that relies on nonlinear optimization, we use a cascaded sequence of small and efficiently solvable quadratic programs. The stability analysis of the geometric controller still holds since we only operate in the nullspace of the force allocation. We show

TABLE II

FLIGHT TEST RESULTS. SHOWN ARE MEAN VALUES OVER TIME FOR A SINGLE FLIGHT EACH WITH STANDARD DEVIATION (SMALL GRAY).

	2 UAVs, point mass	3 UAVs, point mass	2 UAVs, rod	3 UAVs, triangle	3 UAVs, point mass
Payload	point mass	point mass	rod	triangle	point mass
Trajectory	figure 8 (13 s)	figure 8 (13 s)	figure 8 (15 s)	figure 8 (15 s)	teleoperation
Mass [g]	10	10	8	10	10
Dimension [cm]	-	-	15	8	-
Cables [cm]	25, 50	25, 50, 75	50, 50	50, 50, 50	50, 50, 50
QP runtime total [ms]	5.7 1.5	23.1 5.8	13.5 2.1	33.1 4.3	21.8 6.2
QP runtime Fd [ms]	0.0 0.0	0.0 0.0	2.4 0.7	6.9 1.3	0.0 0.0
QP runtime SVM [ms]	2.0 1.2	14.7 4.8	2.7 1.3	8.4 2.9	13.0 4.8
QP runtime $\boldsymbol{\mu}$ [ms]	1.8 0.8	3.9 3.0	3.0 1.2	4.7 2.7	4.4 3.7
Payload x, y, z error [cm]	2.5 1.3, 3.0 2.2, 2.8 1.4	3.6 3.0, 6.3 3.4, 4.0 2.3	4.3 3.4, 3.1 2.4, 4.2 1.7	2.4 1.6, 2.9 1.9, 3.6 1.6	3.0 3.0, 6.8 4.5, 5.9 4.1
Payload r, p, y error [deg]	-	-	6.3 4.5, 6.9 3.9, 20.0 7.3	10.3 4.3, 5.9 3.8, 3.4 1.3	-

that our method scales well with the number of robots with runtime reduction of an order of magnitude compared to the state-of-the-art controllers. We demonstrate through different physical experiments that our approach can be executed on compute-constrained multirotors in realtime.

An exciting future avenue lies in planning the desired motions and cable forces for time-optimal and collision-free navigation. It would also be an interesting future research direction to analyze the solution quality of our approach with respect to the non-convex formulations. Furthermore, a future research direction for the system's model and control is to allow unknown physical parameters of the payload and non-taut cables, rather than the rigid rod assumption.

REFERENCES

[1] C. Masone, H. H. Bülthoff, and P. Stegagno, "Cooperative transportation of a payload using quadrotors: A reconfigurable cable-driven parallel robot," in *Int. Conf. Intell. Robots Syst.*, 2016, pp. 1623–1630.

[2] C. Gabellieri, M. Tognon, L. Pallottino, and A. Franchi, "A study on force-based collaboration in flying swarms," in *Int. Conf. Swarm Intell.*, 2018, pp. 3–15.

[3] T. Lee, "Geometric control of quadrotor UAVs transporting a cable-suspended rigid body," *IEEE Trans. Control Syst. Tech.*, vol. 26, no. 1, pp. 255–264, 2017.

[4] T. Lee, K. Sreenath, and V. Kumar, "Geometric control of cooperating multiple quadrotor UAVs with a suspended payload," in *Conf. Decis. Control*, 2013, pp. 5510–5515.

[5] G. Li and G. Loianno, "Nonlinear model predictive control for cooperative transportation and manipulation of cable suspended payloads with multiple quadrotors," in *Int. Conf. Intell. Robots Syst.*, 2023, pp. 5034–5041.

[6] S. Sun and A. Franchi, "Nonlinear MPC for full-pose manipulation of a cable-suspended load using multiple UAVs," in *Int. Conf. on Unm. Air. Sys.*, 2023, pp. 969–975.

[7] X. Liu, G. Li, and G. Loianno, "Safety-aware human-robot collaborative transportation and manipulation with multiple MAVs," *CoRR*, vol. abs/2210.05894, 2022.

[8] P. O. Pereira and D. V. Dimarogonas, "Control framework for slung load transportation with two aerial vehicles," in *Conf. Decis. Control*, 2017, pp. 4254–4259.

[9] M. Tognon, C. Gabellieri, L. Pallottino, and A. Franchi, "Aerial co-manipulation with cables: The role of internal force for equilibria, stability, and passivity," *Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2577–2583, 2018.

[10] E. Tuci, M. H. Alkilabi, and O. Akanyeti, "Cooperative object transport in multi-robot systems: A review of the state-of-the-art," *Frontiers in Robotics and AI*, vol. 5, p. 59, 2018.

[11] G. Li, X. Liu, and G. Loianno, "RotorTM: A flexible simulator for aerial transportation and manipulation," *IEEE Trans. Robot.*, pp. 1–20, 2023.

[12] M. Manubens, D. Devaurs, L. Ros, and J. Cortés, "Motion planning for 6-d manipulation with aerial towed-cable systems," in *Rob.: Sci. and Syst.*, 2013.

[13] H. G. De Marina and E. Smeur, "Flexible collaborative transportation by a team of rotorcraft," in *Int. Conf. Robot. Autom.*, 2019, pp. 1074–1080.

[14] K. Sreenath and V. Kumar, "Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots," in *Rob.: Sci. and Syst.*, 2013.

[15] D. Six, S. Briot, A. Chriette, and P. Martinet, "The kinematics, dynamics and control of a flying parallel robot with three quadrotors," *Robot. Autom. Lett.*, vol. 3, no. 1, pp. 559–566, 2017.

[16] K. Zhao and J. Zhang, "Composite disturbance rejection control strategy for multi-quadrotor transportation system," *Robot. Autom. Lett.*, vol. 8, no. 8, pp. 4697–4704, 2023.

[17] B. E. Jackson, T. A. Howell, K. Shah, M. Schwager, and Z. Manchester, "Scalable cooperative transport of cable-suspended loads with UAVs using distributed trajectory optimization," *Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3368–3374, 2020.

[18] A. Sharma and N. K. Sinha, "Decentralized aerial transportation and manipulation of a cable-slung payload with swarm of agents," *CoRR*, vol. abs/2306.12331, 2023.

[19] A. Tagliabue, M. Kamel, R. Siegwart, and J. Nieto, "Robust collaborative object transportation using multiple MAVs," *Int. J. Robotics Res.*, vol. 38, no. 9, pp. 1020–1044, 2019.

[20] A. Jiménez-Cano, D. Sanalitro, M. Tognon, A. Franchi, and J. Cortés, "Precise cable-suspended pick-and-place with an aerial multi-robot system: A proof of concept for novel robotics-based construction techniques," *J. of Int. & Rob. Sys.*, vol. 105, no. 3, p. 68, 2022.

[21] A. Petitti, D. Sanalitro, M. Tognon, A. Milella, J. Cortés, and A. Franchi, "Inertial estimation and energy-efficient control of a cable-suspended load with a team of UAVs," in *Int. Conf. on Unm. Air. Sys.*, 2020, pp. 158–165.

[22] J. Geng and J. W. Langelaan, "Cooperative transport of a slung load using load-leading control," *J. Guid., Control, Dyn.*, vol. 43, no. 7, pp. 1313–1331, 2020.

[23] Z. Li, J. F. Horn, and J. W. Langelaan, "Coordinated transport of a slung load by a team of autonomous rotorcraft," in *AIAA Guid., Nav., Contr. Conf.*, 2014.

[24] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Int. Conf. Robot. Autom.*, 2011, pp. 2520–2525.

[25] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Conf. Decis. Control*, 2010, pp. 5420–5425.

[26] J. Förster, "System identification of the crazyflie 2.0 nano quadrocopter," B.S. thesis, ETH Zurich, 2015.

[27] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *J. of Mach. Lear. Res.*, vol. 17, no. 83, pp. 1–5, 2016.

[28] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Math. Prog. Comp.*, vol. 12, no. 4, pp. 637–672, 2020.

[29] J. A. Preiss, W. Hönig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *Int. Conf. Robot. Autom.*, 2017, pp. 3299–3304.

[30] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, 2022.