# Kinodynamic Motion Planning for a Team of Multirotors Transporting a Cable-Suspended Payload in Cluttered Environments

Khaled Wahba[1], Joaquim Ortiz-Haro[1,2], Marc Toussaint[1], and Wolfgang Hönig[1]

***Abstract*— We propose a motion planner for cable-driven payload transportation using multiple unmanned aerial vehicles (UAVs) in an environment cluttered with obstacles. Our planner is kinodynamic, i.e., it considers the full dynamics model of the transporting system including actuation constraints. Due to the high dimensionality of the planning problem, we use a hierarchical approach where we first solve the geometric motion planning using a sampling-based method with a novel sampler, followed by constrained trajectory optimization that considers the full dynamics of the system. Both planning stages consider inter-robot and robot/obstacle collisions. We demonstrate in a software-in-the-loop simulation and real flight experiments that there is a significant benefit in kinodynamic motion planning for such payload transport systems with respect to payload tracking error and energy consumption compared to the standard methods of planning for the payload alone. Notably, we observe a significantly higher success rate in scenarios where the team formation changes are needed to move through tight spaces.**

## I. Introduction

Uncrewed aerial vehicles (UAVs) are ideal for tasks that involve accessing remote locations, which makes them valuable collaborators in a variety of scenarios. Cable-driven payload transportation using multiple UAVs is well suited for collaborative assistance in construction sites such as carrying tools [1] or transporting materials.

The field of control methods for payload transport systems has witnessed significant advancements. In particular, control algorithms [2], have been devised to solve the transport problem with stability guarantees. However, they neglect inter-robot and robot/obstacle collisions. Conversely, alternative methods employ a nonlinear optimization framework to account for inter-robot collisions, which require intricate online and on-board computations.

A common limitation of the current methods for payload transport systems is that they assume a provided feasible reference trajectory that can be tracked. However, generating such reference trajectories for nonlinear, high-dimensional systems in cluttered environments is a recurring challenge and still an open problem. Traditionally, reference trajectories have been computed using either linear interpolation, planning for simplified dynamical models, or planning only for the payload. However, when the controller attempts to track these dynamically unfeasible trajectories in cluttered environments, it is likely to fail from either motor saturation

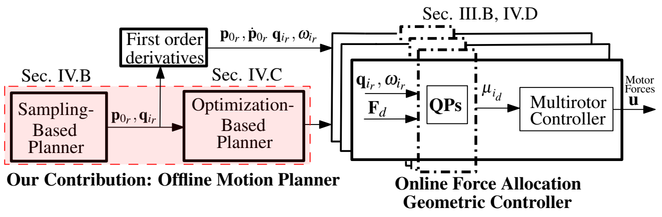
Sec. IV.B
Sec. IV.C
Sec. III.B, IV.D
First order derivatives
$\mathbf{p}_{0_r}, \dot{\mathbf{p}}_{0_r}\ \mathbf{q}_{i_r}, \omega_{i_r}$
Sampling-Based Planner
$\mathbf{p}_{0_r}, \mathbf{q}_{i_r}$
Optimization-Based Planner
$\mathbf{q}_{i_r}, \omega_{i_r}$
$\mathbf{F}_d$
QPs
$\mu_{i_d}$
Multirotor Controller
Motor Forces
$\mathbf{u}$
Our Contribution: Offline Motion Planner
Online Force Allocation Geometric Controller

Fig. 1. Highlighted in the red box is our full kinodynamic motion planning algorithm. The geometric output of a sampling-based motion planner is used to initialize an optimizer, which generates the full feasible reference trajectory of the payload $\mathbf{p}_{0_r}$ and the cable states $\mathbf{q}_{i_r}$. One can also use our sampling-based motion planner and compute the first order derivatives of the geometric states $(\dot{\mathbf{p}}_{0_r}, \boldsymbol{\omega}_{i_r})$ to provide a reference trajectory. Each reference trajectory is then tracked by our controller [3].

or collisions caused by high tracking errors. These effects are more notable if agile maneuvers are desired, or robots with low thrust-to-weight ratio are employed. To the best of our knowledge, there is no kinodynamic motion planner for cable-suspended payload transport. However, we show that such a planner has significant advantages as it is possible to construct feasible trajectories for the entire system's state, including not only the payload but also the UAVs and cable states. These trajectories account for inter-robot, robot/obstacle, cable/obstacle collisions, and the actuation limits of the motors. Overall, tracking such trajectories with existing controllers can lead to more reliable operation (success rate), higher predictability (lower tracking error), and lower energy consumption as flight time can be reduced.

In this paper, we extend our previous work [3] by proposing an offline hierarchical kinodynamic motion planner (see Fig. 1) to generate feasible reference trajectories to transport a point mass with multiple aerial multirotors in environments cluttered with obstacles. We use an enhanced version of our prior geometric sampling-based motion planner [4] as an initial guess for a nonlinear optimizer. The optimizer generates feasible energy-efficient reference trajectories that take inter-robot, cable/cable, cable/obstacle and robot/obstacle collisions into account. To the extent of our knowledge, this is the first work that solves the full kinodynamic motion planning problem for the cable-suspended payload system with multirotors. We evaluate the planner by tracking reference trajectories using our highly-efficient controller [3] and compare it using a realistic software-in-the-loop (SITL) simulation and several real flights with two baselines: the geometric cable-payload planner and the payload-only planning baseline. We consider three different environments, vary the number of robots and report key metrics such as energy efficiency, tracking accuracy, and success rate.

[1] Faculty of Electrical Engineering and Computer Science, Technical University of Berlin, Berlin, Germany k.wahba@tu-berlin.de.
[2] Machines in Motion Laboratory, New York University, USA
Code: https://github.com/IMRCLab/coltrans-planning
Website: https://imrclab.github.io/coltrans-planning/
The research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 448549715.

## II. Related Work

The **dynamic model** of the cable-suspended payload system with multiple multirotors can be described through Lagrangian mechanics [2, 5–7]. **Control algorithms** of the payload transport system include centralized controllers, employing a cascading reactive approach with stability guarantees [2, 8, 9] and decentralized controllers [10, 11]. There are still practical obstacles to overcome, including the requirement to measure noisy payload accelerations and considering inter-robot collisions. Optimization-based controllers can directly include some constraints. One approach involves using iterative gradient-based solvers, but they are susceptible to local minima [12, 13]. Nonlinear model predictive control (NMPC) offers an alternative for payload control and collision avoidance [14, 15]. However, its high computational costs and limited scalability with multiple robots make it unsuitable for resource-constrained microcontrollers. Moreover, these methods do not directly integrate the dynamic model of the multirotors or the actuation limits of their motors into their online optimization formulation. They rely only on cable tension constraints and state bounds, which are only suitable for slow-varying trajectories. Our previous work [3, 4] combines advantages of prior work by leveraging QPs that can handle inter-robot, robot/obstacle and cable/cable collision constraints. We proposed a QP-force allocation geometric controller that are executed on compute-constrained multirotors in realtime efficiently.

Other methods employ offline **motion planners** [16–18] for inter-robot and robot/obstacle collision avoidance. These planners use sampling-based or control-based approaches, but often overlook multirotor actuation limits, state bounds, and payload distribution, potentially suggesting impractical configurations for the controller. From a broader robotics perspective, kinodynamic motion planning can rely on search or sampling [19–22]. Yet, these methods scale exponentially with the dimensionality of the state space, and thus fail when planning for a cable-suspended payload system.

In contrast, constrained trajectory **optimization** methods [23–27] are more suitable for planning in high dimensional state spaces, with polynomial complexity on the state size. Kinodynamic motion planning using nonlinear optimization has shown great success in different robotics fields, from sequential manipulation planning [28] to legged locomotion [29–31] and flying robots [32, 33]. However, optimization methods require a good initial guess of the solution trajectory, as they can only optimize the trajectory locally. In our method, we combine a novel geometric sampling-based motion planner for cable-suspended payload systems with multiple multirotors with subsequent nonlinear trajectory optimization for full kinodynamic planning, resulting in a state-of-the-art integrated system.

## III. Background

This section provides necessary background for the dynamic model and the used control design, see [3] for details.

### A. System Description

Consider a team of $n$ multirotors transporting a cable-suspended payload. The payload is a point mass with mass $m_0$ and the cables are massless rigid rods each with length $l_i$, where $i \in \{1, \ldots, n\}$. Each UAV is modeled as a floating rigid body with mass $m_i$ and diagonal moment of inertia $\mathbf{J}_i$.

The state space vector of the payload is defined by the position and velocity vectors $\mathbf{p}_0 \in \mathbb{R}^3$ and $\dot{\mathbf{p}}_0 \in \mathbb{R}^3$. While the cable states are composed of the cable unit vector $\mathbf{q}_i \in \mathbb{S}^2$ directed from the multirotor towards the payload, where $\mathbb{S}^2 = \{\mathbf{q} \in \mathbb{R}^3 | \|\mathbf{q}\| = 1\}$, and the cable angular velocity $\boldsymbol{\omega}_i \in \mathbb{R}^3$. Moreover, the multirotor position and velocity state vectors are $\mathbf{p}_i \in \mathbb{R}^3$ and $\dot{\mathbf{p}}_i \in \mathbb{R}^3$ with respect to the global frame of reference. As the cables are modeled rigidly, the position of each multirotor can be computed as

$$\mathbf{p}_i = \mathbf{p}_0 - l_i \mathbf{q}_i. \tag{1}$$

The attitude states of the $i$-th multirotor is comprised of the rotation matrix $\mathbf{R}_i \in SO(3)$ and body angular velocity $\boldsymbol{\Omega}_i \in \mathbb{R}^3$. In summary, the manifold configuration $\mathcal{C}$ of the presented system is defined by $\mathbb{R}^3 \times (\mathbb{S}^2 \times SO(3))^n$, where the state space vector of the full system is defined by

$$\boldsymbol{x} = (\mathbf{p}_0, \dot{\mathbf{p}}_0, \mathbf{q}_1, \boldsymbol{\omega}_1, \mathbf{R}_1, \boldsymbol{\Omega}_1, \ldots, \mathbf{q}_n, \boldsymbol{\omega}_n, \mathbf{R}_n, \boldsymbol{\Omega}_n)^T. \tag{2}$$

The output wrench of the $i$-th robot is defined by the collective thrust and the torques $\boldsymbol{\eta}_i = (f_{ci}, \mathbf{M}_i)^T$, where $\mathbf{M}_i = (\tau_{xi}, \tau_{yi}, \tau_{zi})^T$. This wrench vector is linearly related to the control input motor forces $\boldsymbol{u}^i$ of the multirotor by $\boldsymbol{\eta}_i = \boldsymbol{B}_0 \boldsymbol{u}^i$, where $\boldsymbol{B}_0$ is the actuation matrix and the motor force command is $\boldsymbol{u}^i = (f_{i_1}, f_{i_2}, f_{i_3}, f_{i_4})^T$, The control input vector of the full system is defined as

$$\boldsymbol{u} = (\boldsymbol{u}^1, \ldots, \boldsymbol{u}^n) \tag{3}$$

and the system kinematics and dynamics are

$$\begin{aligned}
&\dot{\mathbf{q}}_i = \boldsymbol{\omega}_i \times \mathbf{q}_i, \qquad (4)\\
&\mathbf{M_t}(\ddot{\mathbf{p}}_0 + g\mathbf{e_3}) = \sum_{i=1}^{n} (f_{ci}\mathbf{R}_i\mathbf{e_3} - m_i l_i \|\boldsymbol{\omega}_i\|^2 \mathbf{q}_i),\\
&m_i l_i \dot{\boldsymbol{\omega}}_i = m_i \hat{\mathbf{q}}_i (\ddot{\mathbf{p}}_0 + g\mathbf{e_3}) - f_{ci}\hat{\mathbf{q}}_i \mathbf{R}_i \mathbf{e_3},\\
&\dot{\mathbf{R}}_i = \mathbf{R}_i \hat{\boldsymbol{\Omega}}_i, \quad \mathbf{J}_i \dot{\boldsymbol{\Omega}}_i = \mathbf{J}_i \boldsymbol{\Omega}_i \times \boldsymbol{\Omega}_i + \mathbf{M}_i,
\end{aligned}$$

where $\ddot{\mathbf{p}}_0$ is the acceleration of the payload, $g$ is the gravitational acceleration constant, $\mathbf{e_3} = (0, 0, 1)^T$ and $\mathbf{M}_t = (m_0 + \sum_{i=1}^{n} m_i)\mathbf{I}_{3\times3}$. $\hat{(\cdot)}$ denotes the skew-symmetric mapping $\mathbb{R}^3 \to \mathfrak{so}(3)$.

### B. Controller Overview

Our control design (see Fig. 1) presents an efficient optimization-based cable force allocation of a geometric controller [2] for cable-suspended payload transportation that is aware of neighboring robots to avoid collisions. Consider the desired control forces that track the payload reference trajectory as $\mathbf{F}_d$. This method reformulates the cable forces allocation optimization problem as three consecutive quadratic programs (QPs). The QPs solve for the desired cable forces $\boldsymbol{\mu}_{i_d}$, taking into account the inter-robot collisions, and track $\mathbf{F}_d$. Thus, $\boldsymbol{\mu}_{i_d}$ is tracked by the $i$-th multirotor with a low-level controller [8].

## IV. Approach

### A. Problem Statement

Consider the system described in Section III-A. The state space vector is defined by (2). Let $\boldsymbol{X} = \langle \boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_T \rangle$ be a sequence of states sampled at time $0, \Delta t, \ldots, T\Delta t$ and $\boldsymbol{U} = \langle \boldsymbol{u}_0, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{T-1} \rangle$ be a sequence of controls applied to the system for times $[0, \Delta t), [\Delta t, 2\Delta t) \ldots, [(T-1)\Delta t, T\Delta t)$, where $\Delta t$ is a small timestep and the controls are constant during this timestep. We denote the start state as $\boldsymbol{x}_s$, the goal state as $\boldsymbol{x}_g$ and the collision-free configuration space as $\mathcal{C}_{\text{free}} \subset \mathcal{C}$, which accounts for collisions between the robots, payload, and the cables as well as collisions against the environment. Then our goal is to transport the payload from a start to a goal state in the minimal time $T$, which can be framed as the following optimization problem

$$\min_{\boldsymbol{X}, \boldsymbol{U}, T} \quad J(\boldsymbol{X}, \boldsymbol{U}, T), \tag{5}$$

$$\text{s.t.} \begin{cases} \boldsymbol{x}_{k+1} = \text{step}(\boldsymbol{x}_k, \boldsymbol{u}_k) \quad \forall k \in \{0, \ldots, T-1\}, \\ \boldsymbol{u}_k \in \mathcal{U} \quad \forall k \in \{0, \ldots, T-1\}, \\ \boldsymbol{x}_0 = \boldsymbol{x}_s, \;\; \boldsymbol{x}_T = \boldsymbol{x}_f, \\ \boldsymbol{x}_k \in \mathcal{C}_{\text{free}} \subset \mathcal{C} \quad \forall k \in \{0, \ldots, T\}, \end{cases}$$

where the cost function can be set to minimize $T$ and other task objectives (e.g., energy). The function $\text{step}(\boldsymbol{x}_k, \boldsymbol{u}_k)$ is the time-discretized version of the dynamic model of the system and the second constraint limits the control input within a feasible control space $\mathcal{U}$ (e.g., actuator limits). The third set of constraints ensures that the motion connects the start and the goal states. The last constraint ensures a collision-free path for the full state while avoiding any cable entanglements.

### B. Geometric Motion Planning (Offline)

Given a start state $\boldsymbol{x}_s$ and a goal state $\boldsymbol{x}_g$ in an environment with obstacles, we propose to use a sampling-based motion planner to plan a collision-free geometric path $\boldsymbol{X}_{\text{geom}}$ for the following state vector

$$\boldsymbol{x}_{\text{geom}} = (\mathbf{p}_{0_r}, \mathbf{q}_{1_r}, \ldots, \mathbf{q}_{n_r})^T, \tag{6}$$

where $\boldsymbol{x}_{\text{geom}} \in \mathbb{R}^3 \times \mathbb{R}^{3n}$. Consequently, $\boldsymbol{X}_{\text{geom}}$ is interpolated and the first order derivatives are computed by numerical differentiation to define the full reference trajectory that can be tracked by a controller or used as an initial guess for an optimizer.

*1) State Space Representation:* We propose to reduce the state space size directly by using different representations for part of the state space. In particular, a possible local parametrization for the unit vector $\mathbf{q}_i \in \mathbb{R}^3$ are the azimuth $\alpha_i$ and elevation $\gamma_i$ angles, such that

$$\mathbf{q}_i = -\big(\cos(\alpha_i)\cos(\gamma_i),\; \sin(\alpha_i)\sin(\gamma_i),\; \sin(\gamma_i)\big)^T, \tag{7}$$

where $\alpha_i \in [0, 2\pi)$ and $\gamma_i \in [0, \pi/2)$. Thus, the reduced state vector can be represented by

$$\boldsymbol{x}_{geom} = (\mathbf{p}_{0_r}, \alpha_{1_r}, \gamma_{1_r}, \ldots, \alpha_{n_r}, \gamma_{n_r})^T \in \mathbb{R}^3 \times \mathbb{R}^{2n}. \tag{8}$$

*2) Cost Function:* Given two consecutive states as $\boldsymbol{x}_{geom}(k)$ and $\boldsymbol{x}_{geom}(k+1)$, we use a cost function that minimizes the integral of the energy between the two states. First, let us define the $\mathbf{q}_{i_{\text{base}}} = (0, 0, -1)^T$ as the desired cable direction to carry the payload statically. Consider the required force magnitude to carry a unit payload mass with respect to the static case (i.e., all cables point towards $\mathbf{q}_{i_{\text{base}}}$) as

$$F(k) = -\frac{1}{n}\sum_{i=1}^{n} \frac{1}{\mathbf{q}_{i_{\text{base}}} \cdot \mathbf{q}_i(k)}, \tag{9}$$

where $(\cdot)$ is the dot product and $n$ is the number of cables. We assume a trapezoidal energy profile between two consecutive states. Thus, the cost function is

$$c = \frac{F(k) + F(k+1)}{2}\Big(\beta \;\; \mathbf{p}_{0_\delta} + (1-\beta)\sum_{i=1}^{n} \mathbf{p}_{i_\delta}\Big), \tag{10}$$

where $\beta = 0.5$ is a weight, $\mathbf{p}_{0_\delta} = \|\mathbf{p}_0(k) - \mathbf{p}_0(k+1)\|$ and $\mathbf{p}_{i_\delta} = \|\mathbf{p}_i(k) - \mathbf{p}_i(k+1)\|$ are the travel distances of the payload position and each UAV, respectively. The sampler will converge to the minimum cost solution over time. The accepted samples by the collision checker ensure that the current formation distributes the load quasi-statically over the cables.

*3) Sampling Strategy:* Sampling $(\alpha_i, \gamma_i)$ uniformly and iid creates two major challenges: i) the probability of sampling a configuration that is collision-free without tangling of the cables shrinks exponentially with $n$, and ii) the number of cable permutations that result in the same relative formation grows factorial in $n$. To mitigate this curse of dimensionality we propose a custom sampler, see Algorithm 1. The sampler uses a preprocessing step to compute a set of witness cable configurations (*formations*) that can be reached from the given valid start state $\boldsymbol{x}_s$. During the actual sampling-based search, we rely on these witness formations.

For the preprocessing we initialize the witness set $\mathcal{S}$ with the initial state $\boldsymbol{x}_s$ (Line 2). For $M-1$ subsequent witnesses, we randomly choose a base state from the set (Line 4) and uniformly sample a formation (Line 5). Then we solve an optimal assignment problem that minimizes the sum of the distances that the UAVs would have to move to change the formation from $\boldsymbol{x}^b$ to $\boldsymbol{x}^r$ (Line 6). For the cost, we extract the position of the UAVs as

$$\text{Pos}(\alpha_i, \gamma_i) = \mathbf{p}_0 - l_i \mathbf{q}_i(\alpha_i, \gamma_i), \tag{11}$$

where $\mathbf{q}_i(\alpha_i, \gamma_i)$ is given in (7) and $\mathbf{p}_0$ is the position of the payload at the initial state $\boldsymbol{x}_s$ which is known to be collision-free. The assignment problem can be solved optimally in polynomial time, for example by using the Hungarian Method (Line 7). We re-arrange the cables (Line 8) and add the new witness if the whole formation change motion from $\boldsymbol{x}^b$ is collision-free (Line 10).

In the online search phase, we randomly pick a witness formation from the pre-computed set, add random noise to it, and augment it with a payload position uniformly drawn from the workspace $\mathcal{W} \subset \mathbb{R}^3$ (Lines 13 to 16).

**Algorithm 1:** Sampling Strategy

```
1  Function Preprocessing($M$, $\boldsymbol{x}_s$, $\sigma$):
2      $\mathcal{S} = \{\boldsymbol{x}_s = (\mathbf{p}_0, \alpha_1, \gamma_1, \dots, \alpha_n, \gamma_n)^T\}$
3      while $|\mathcal{S}| < M$ do
4          $\boldsymbol{x}^b \leftarrow$ ChooseUniform($\mathcal{S}$)  ▷ pick a base state
5          $\boldsymbol{x}^r \leftarrow$ SampleUniform($\mathcal{C}$)  ▷ sample a new state
           /* Permute cables in $\boldsymbol{x}^r$ s.t. UAVs move minimally from $x^b$ */
6          $c[i,j] \leftarrow \|\text{Pos}(\alpha_i^b, \gamma_i^b) - \text{Pos}(\alpha_j^r, \gamma_j^r)\| \;\; \forall i,j \in \{1,\dots,N\}$
7          $A \leftarrow$ OptimalAssignment($c$)
8          $\boldsymbol{x} \leftarrow (\mathbf{p}_0, \alpha^r_{A[1]}, \gamma^r_{A[1]}, \dots, \alpha^r_{A[n]}, \gamma^r_{A[n]})^T$
9          if $\boldsymbol{x} \in \mathcal{C}_{\text{free}} \wedge (\boldsymbol{x}, \boldsymbol{x}^b) \in \mathcal{C}_{\text{free}}$ then
10             $\mathcal{S} = \mathcal{S} \cup \{\boldsymbol{x}\}$
11     return $\mathcal{S}$
12 Function SampleState($\mathcal{S}$, $\sigma$):
13     $\boldsymbol{x} \leftarrow$ ChooseUniform($\mathcal{S}$)
14     $(\_, \alpha_1, \gamma_1, \dots, \alpha_n, \gamma_n) \leftarrow$ SampleGaussian($\mathcal{C}, \boldsymbol{x}, \sigma$)
15     $\mathbf{p}_0 \leftarrow$ SampleUniform($\mathcal{W}$)
16     return $(\mathbf{p}_0, \alpha_1, \gamma_1, \dots, \alpha_n, \gamma_n)$
```

For the goal we only check if the payload reached the desired state. We use goal biasing and sample goal states in a similar fashion as in the online search, except that the payload part is the user provided desired state rather than sampled as in Line 15.

*4) Reference Trajectory (Offline):* Since the geometric planner generates only $\boldsymbol{x}_{geom}$, the rest of the state vector $\boldsymbol{x}$ in (2) needs to be recovered. The payload velocity vector $\dot{\mathbf{p}}_0$ is computed with numerical differentiation over small time steps. The $\boldsymbol{\omega}_i$ is set to $\mathbf{0}$, since we found that the numerical differentiation of $\mathbf{q}_i$ is a noisy signal. Finally, the attitude states of the *i*-th multirotor $(\mathbf{R}_i, \boldsymbol{\Omega}_i)$ are set to $(\mathbf{I}_3, \mathbf{0})$, respectively, where $\mathbf{I}_3$ is the identity matrix.

## C. Nonlinear Trajectory Optimization (Offline)

The objective of the optimization step is to solve the full kinodynamic motion planning problem using the output of the geometric planner as initial guess. While the geometric planner only plans for the payload and cable states, trajectory optimization considers the full state of the systems $\boldsymbol{x}$ and the motor forces $\boldsymbol{u}$ expressed in (2) and (3).

To generate a valid initial guess, the geometric solution $\mathbf{X}_{\text{geom}}$ is interpolated with a small time discretization ($\Delta t = 0.01$ s), setting the velocity components to zero (which empirically results in a better initial guess) and using a default orientation for the robots. For the initial guess on the controls sequence, we use the constant motor forces that are required to hover each quadrotor individually. The optimization problem is formulated as a nonlinear trajectory optimization

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{U},\Delta t} \quad & \sum_k (\Delta t - \Delta t_0)^2 + \beta_1 ||\boldsymbol{u}_k||^2 + \\ & \beta_2 ||\ddot{\boldsymbol{x}}(\boldsymbol{x}_k, \boldsymbol{u}_k)||^2, \end{aligned} \tag{12}$$

$$\text{s.t.} \begin{cases} \boldsymbol{x}_{k+1} = \boldsymbol{x}_k \oplus f(\boldsymbol{x}_k, \boldsymbol{u}_k)\Delta t \quad \forall k \in \{0, \dots, T-1\}, \\ \boldsymbol{u}_k \in \mathcal{U}, \quad \boldsymbol{x}_k \in \mathcal{C} \quad \forall k \in \{0, \dots, T\}, \\ \boldsymbol{x}_0 = \boldsymbol{x}_s, \quad \boldsymbol{x}_T = \boldsymbol{x}_f, \\ g(\boldsymbol{x}_k) \geq 0 \quad \forall k \in \{0, \dots, T\}. \end{cases}$$

The decision variables are the sequence of controls $\mathbf{U} = \langle \boldsymbol{u}_0, \dots, \boldsymbol{u}_{T-1} \rangle$ , the sequence of states $\mathbf{X} = \langle \boldsymbol{x}_0, ..., \boldsymbol{x}_T \rangle$ and the time-duration $\Delta t \in \mathbb{R}_+$ of the intervals in the time discretization. The number of steps $T$ is defined by the initial guess. The collision distance between the robots, payload, cables and environment is computed by the signed distance function $g(\boldsymbol{x}_k)$.

In the dynamics constraints, the continuous dynamics $f(\boldsymbol{x}_k, \boldsymbol{u}_k)$ are now discretized using Euler integration with a time interval subject to optimization. We use the notation $\oplus$ to highlight that some components of the state space $\boldsymbol{x}_k \in \mathcal{C}$ lie on manifolds (e.g. quaternions or unit vectors).

The dynamics of the system is highly nonlinear, and the collision constraints define a non-convex feasible set, resulting in a very challenging nonlinear problem. To improve the robustness and success of trajectory optimization, and to ensure good local convergence towards a locally optimal trajectory, we combine three terms in the objective function.

The term $(\Delta t - \Delta t_0)^2$, minimizes the time duration of the trajectory, with a small $\Delta t_0$ acting as a proximal regularization. The term $||\boldsymbol{u}_k||^2$ minimizes the control effort. The term $||\ddot{\boldsymbol{x}}(\boldsymbol{x}_k, \boldsymbol{u}_k)||^2$ minimizes the acceleration of the system and is required to generate smooth trajectories and to ensure a good convergence of the solver. The coefficients $\beta_1, \beta_2 > 0$ are used to weigh the three terms. Together, these three terms combine the original objective of time minimization with a regularization that provides smooth gradients for the nonlinear optimizer to converge successfully.

The nonlinear trajectory optimization problem is solved using Differential Dynamic Programming (DDP) [23, 34], which ensures that the nonlinear dynamics constraints are always fulfilled. Goal constraints, states and control bounds are included with a squared penalty in the cost using the squared penalty method [35].

## D. Desired Cable Forces (Online)

A significant advantage of the QPs formulation presented in [3] is that a user is able to specify a desired formation configuration based on another task objective (e.g., obstacle avoidance). Let us define $\boldsymbol{\mu}_{i_r}$ as prespecified reference cable forces. The cost function in the QPs can be modified as

$$c = \frac{1}{2} \|\boldsymbol{\mu}_{i_d}\|^2 + \lambda \|\boldsymbol{\mu}_{i_r} - \boldsymbol{\mu}_{i_d}\|^2, \tag{13}$$

where the first term minimizes the sum of norms of the cable forces. The second term minimizes the difference between the cable forces and the preferred ones $\boldsymbol{\mu}_{i_r}$ with $\lambda$ as a weighting factor. Our previous work [3] proposed this method for a teleoperation task where the operator can switch to a predefined line formation to pass between two obstacles. Here, we compare two new motion planning algorithms that plan for the payload poses, the desired cable forces $\boldsymbol{\mu}_{i_r}$ and the $n$ multirotors states and motor control input vector $\boldsymbol{u}$.

## E. Reference Cable Forces Computation (Offline)

In order to determine the reference cable forces $\boldsymbol{\mu}_{i_r}$, the reference motors forces $\boldsymbol{u}^i$ and the reference cable unit vectors $\mathbf{q}_{i_r}$ are used. Particularly, the reference cable forces

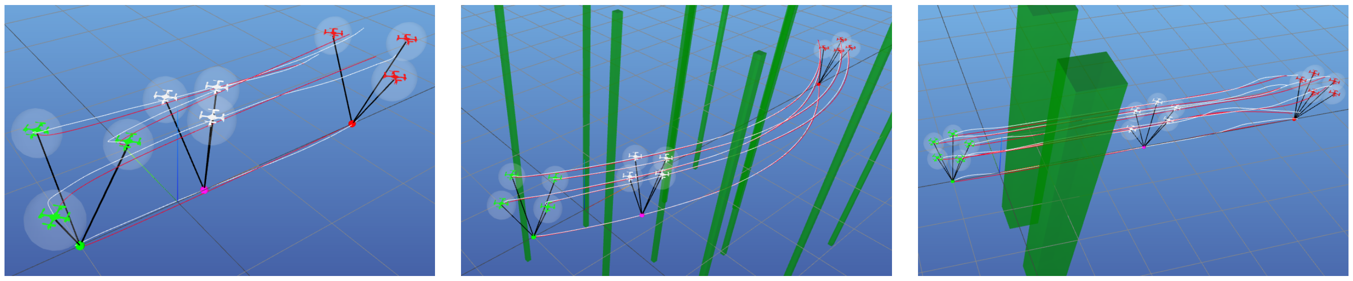
Fig. 2. Validation scenarios. From left to right: *empty* (3 robots), *forest* (4 robots), *window* (5 robots). Green UAVs (left on each picture) show the initial state, red UAVs the desired state. The red line represents the reference trajectory, and the white line is the tracked trajectory by our controller. For *window*, the obstacles necessitate a formation change to pass through a narrow passage.

$\boldsymbol{\mu}_{i_r}$ are derived from $\mathbf{q}_{i_r}$, such that $\boldsymbol{\mu}_{i_r} = -T_i\mathbf{q}_{i_r}$, where $T_i$ is the tension of the $i$-th cable. Then, the tension of the $i$-th cable is computed from the reference motor forces $\boldsymbol{u}^i$ as

$$T_i = \|m_i(\ddot{\mathbf{p}}_{i_r} + g\mathbf{e_3}) - f_{ci_r}\mathbf{R}_{i_r}\mathbf{e_3}\|, \tag{14}$$

where $\ddot{\mathbf{p}}_{i_r}$ and $\mathbf{R}_{i_r}$ are the reference acceleration and orientation of the $i$-th multirotor. $f_{ci_r}$ is the collective thrust magnitude computed from the actuation matrix $\boldsymbol{B}_0$, as shown in Section III-A. Differentiating the position of the multirotor from (1) twice yields the acceleration of the $i$-th multirotor as

$$\ddot{\mathbf{p}}_{i_r} = \ddot{\mathbf{p}}_{0_r} - l_i(\dot{\boldsymbol{\omega}}_{i_r} \times \mathbf{q}_{i_r} + \boldsymbol{\omega}_{i_r} \times \dot{\mathbf{q}}_{i_r}), \tag{15}$$

where the acceleration of the payload and the $i$-th cable $\ddot{\mathbf{p}}_{0_r}$, $\dot{\boldsymbol{\omega}}_{i_r}$ can be computed using (4).

In the geometric case, we have $\boldsymbol{\omega}_i = \boldsymbol{\Omega} = \mathbf{0}$ and $\mathbf{R}_i = \mathbf{I}_3$, see Section IV-B.4. Note that $\dot{\boldsymbol{\omega}}_{i_r}$ is not generally $\mathbf{0}$ and still computed using (4). For the control input, we assume that each multirotor is hovering, i.e., $\boldsymbol{\eta}_i = (m_i g\mathbf{e}_3, 0, 0, 0)^T$. Then (14) and (15) apply directly, but the resulting $\boldsymbol{\mu}_{i_r}$ may violate the dynamics. Nevertheless, since $\boldsymbol{\mu}_{i_r}$ is added as a soft constraint in (13), the controller will compute feasible $\boldsymbol{\mu}_{i_d}$ to track.

## V. Experimental Results

To validate the performance of our method, we provide several real flight and simulation experiments. In particular, we implement the sampling-based motion planner using OMPL [36], a widely used C++ library and rely on RRT* as sampling motion planning algorithm. For optimization we extend Dynoplan [22], an optimization-based motion planner based on Crocoddyl [23], to include the system dynamics defined in (4). Both geometric and optimization-based planners rely on FCL (Flexible Collision Library) [37] for collision checking. We use sympy [38] to compute the analytical Jacobians of the dynamics and to generate efficient C code.

### A. Simulation Results

For validation, we use a software-in-the-loop simulation where the flight controller code [3] that runs on Bitcraze Crazyflie 2.1 multirotors is directly executed, together with the physics model that we implement in Dynobench [22]. Due to the agile nature of multirotors, we use a small $\Delta t$ of $0.01\,\mathrm{s}$ for both optimization and simulation. All results are validated to fulfill the constraints in (5).

We test our motion planning approach on three different scenarios, see Fig. 2 and the supplemental video: obstacle-free (i.e., Empty), a random forest-like environment, and a window environment, where the payload is transported through a narrow passage between two columns. The gap between each column and the origin linearly increases from $0.4\,\mathrm{m}$ to $0.5\,\mathrm{m}$ as the number of robots increases. For each scenario, we evaluate five different problems increasing the number of robots, from two to six robots.

All scenarios for the motion planning experiments were solved on a workstation (AMD Ryzen Threadripper PRO 5975WX @ 3.6 GHz, 64 GB RAM, Ubuntu 22.04) and repeated 10 times. The runtime of the geometric motion planner was limited to $300\,\mathrm{s}$.

*1) Baseline Comparison:* We consider three different approaches for payload transport: ***Payload*** uses geometric planning for the payload alone and the resulting trajectory is tracked using the specialized payload transport controller. ***Geom*** uses geometric planning for the payload, cables, and UAVs as described in Section IV-B. The same controller can then use the augmented cost function (13) to track both the cables and the payload, allowing formation changes to pass through passages. ***Opt*** uses the proposed pipeline for kinodynamic motion planning: We first generate a geometric plan for the cable-payload system, (Section IV-B) and then we use the nonlinear trajectory optimization to generate feasible trajectories (Section IV-C).

We analyze tracking error, energy usage, and success rate in 15 different settings. The results are shown in Table I. We consider an execution a success if i) a reference trajectory is successfully computed, ii) no collisions occurred when the controller is tracking this reference trajectory, and iii) the controller reaches the goal. For the empty environment, all algorithms are successful. In the forest environment, the simple *payload* approach results in collisions since the UAVs are ignored during planning, and the planned path tends to be too close to obstacles. Here, the *geom* approach works in the majority of the tested cases. For the *window* case, only our kinodynamic planner is highly successful, since the geometric planner often produces desired cable states that cannot be tracked accurately when considering

TABLE I

SIMULATION RESULTS. SHOWN ARE MEAN VALUES FOR THE TRACKING ERROR AND ENERGY USAGE OF THE FINAL SOLUTION OVER 50 RUNS WITH A TIMELIMIT OF 300 s FOR THE GEOMETRIC PLANNER. STANDARD DEVIATION IS SMALL GRAY. PERCENTAGES ARE SUCCESS RATES.

| Environment | Metrics | 2 robots | 3 robots | 4 robots | 5 robots | 6 robots | success [%] |
|---|---|---|---|---|---|---|---|
| empty | Error payload [m] | **0.02** 0.01 **100 %** | 0.04 0.02 **100 %** | 0.02 0.01 **100 %** | **0.01** 0.01 14 % | — | 63 |
| | Error geom [m] | 0.03 0.01 **100 %** | 0.05 0.03 **100 %** | 0.03 0.01 98 % | 0.02 0.01 40 % | — | 68 |
| | Error opt [m] | **0.02** 0.03 98 % | **0.01** 0.02 **100 %** | **0.01** 0.01 **100 %** | **0.01** 0.01 **98 %** | **0.02** 0.01 **98 %** | 99 |
| | Energy payload [Wh] | 0.05 0.00 | 0.08 0.00 | 0.10 0.00 | 0.12 0.00 | — | |
| | Energy geom [Wh] | 0.05 0.00 | 0.08 0.00 | 0.10 0.00 | 0.12 0.00 | — | |
| | Energy opt [Wh] | **0.04** 0.00 | **0.05** 0.00 | **0.07** 0.00 | **0.11** 0.01 | **0.13** 0.01 | |
| forest | Error payload [m] | — | — | — | — | — | 0 |
| | Error geom [m] | 0.08 0.06 46 % | 0.10 0.06 66 % | 0.10 0.06 62 % | 0.11 0.06 16 % | 0.20 0.08 2 % | 38 |
| | Error opt [m] | **0.02** 0.04 **100 %** | **0.01** 0.01 **100 %** | **0.01** 0.00 **96 %** | **0.03** 0.02 **88 %** | **0.03** 0.02 **94 %** | 96 |
| | Energy payload [Wh] | — | — | — | — | — | |
| | Energy geom [Wh] | **0.05** 0.00 | 0.08 0.00 | 0.10 0.00 | **0.12** 0.00 | 0.15 0.00 | |
| | Energy opt [Wh] | **0.05** 0.00 | **0.07** 0.00 | **0.09** 0.00 | **0.12** 0.01 | **0.14** 0.01 | |
| window | Error payload [m] | 0.04 0.02 4 % | 0.06 0.04 12 % | 0.04 0.02 2 % | — | — | 4 |
| | Error geom [m] | 0.10 0.09 34 % | 0.10 0.06 14 % | — | — | — | 10 |
| | Error opt [m] | **0.02** 0.02 **98 %** | **0.01** 0.01 **100 %** | **0.02** 0.01 **90 %** | **0.04** 0.03 **54 %** | **0.05** 0.03 **56 %** | 80 |
| | Energy payload [Wh] | **0.05** 0.00 | 0.08 0.00 | 0.10 0.00 | — | — | |
| | Energy geom [Wh] | **0.05** 0.00 | 0.08 0.00 | — | — | — | |
| | Energy opt [Wh] | **0.05** 0.00 | **0.07** 0.00 | **0.09** 0.01 | **0.13** 0.01 | **0.15** 0.01 | |

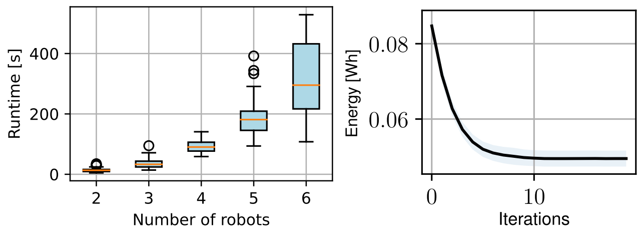

Fig. 3. Left: Computational effort in seconds for the optimization to compute a solution in the forest environment over different numbers of robots. Right: Solution quality (in terms of energy) when sequentially solving the kinodynamic optimization over multiple iterations.

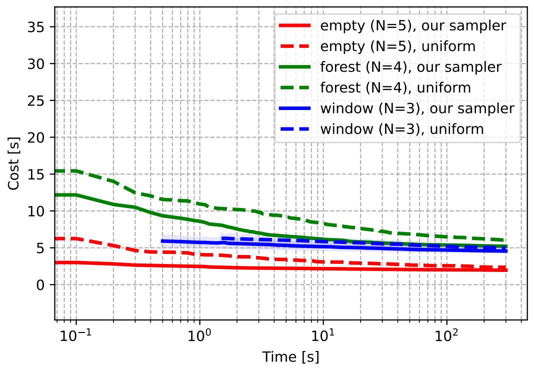

Fig. 4. Examples for the sampling-based geometric planner using different environment and sampling strategies. The plot shows the mean and standard deviation (shaded) for cost convergence over runtime (log-scale), if the success rate is over 50% (50 trials).

the kinodynamic constraints. The average tracking error in all settings is significantly lower for our approach *opt* (up to 9 times lower compared to the geometric solution), since the full system states are considered. We also compute the expected energy consumption of the flight using a model for the Bitcraze Crayflie 2.X multirotors [1]. Here, the energy linearly depends on the flight time as well as the forces $\boldsymbol{u}^i$ that the propellers create. Not surprisingly, the energy for *payload* and *geom* are almost identical, while our method, *opt*, reaches a reduction of around 10 %. However, the energy is reduced by almost 50 % with iterative optimization (see Section V-A.3). Moreover, the energy usage almost linearly increases with the number of robots, which is not surprising as multirotors require a lot of energy just to stay airborne.

[1]https://bitcraze.io/documentation/repository/crazyflie-firmware/master/functional-areas/pwm-to-thrust/

TABLE II

REAL FLIGHTS RESULTS. SHOWN ARE MEAN VALUES FOR THE TRACKING ERROR, ENERGY USAGE AND THE PLANNING TIME OF EACH APPROACH OF 10 FLIGHT EXPERIMENTS EXECUTED ON-BOARD OF THE CFS TO TRACK THE REFERENCE TRAJECTORIES. STANDARD DEVIATION IS SMALL GRAY. PERCENTAGES ARE SUCCESS RATES.

| Env. | Metric | 2 robots | 3 robots |
|---|---|---|---|
| window | Error geom [m] | 0.062 0.04 80 % | 0.121 0.06 60 % |
| | Error opt [m] | **0.056** 0.03 100 % | **0.085** 0.04 100 % |
| | Energy geom [Wh] | 0.022 0.00 | 0.027 0.00 |
| | Energy opt [Wh] | **0.016** 0.00 | **0.020** 0.00 |
| | Plan time geom [s] | 0.4 0.42 | 0.1 0.05 |
| | Plan time opt [s] | 26.8 9.45 | 46.0 12.5 |
| forest | Error geom [m] | 0.092 0.08 60 % | — |
| | Error opt [m] | **0.057** 0.04 100 % | **0.118** 0.05 100 % |
| | Energy geom [Wh] | 0.017 0.00 | — |
| | Energy opt [Wh] | **0.012** 0.00 | **0.024** 0.01 |
| | Plan. time geom [s] | 0.3 0.43 | 168.4 105.6 |
| | Plan. time opt [s] | 22.2 3.71 | 57.8 16.13 |

2) *Computational Effort:* We analyze the runtime of our two planning phases, geometric planning and optimization, separately. For the geometric planner, we show that our new sampling strategy (Algorithm 1) compared to the uniform sampling, significantly reduces the time until a low-cost solution is reached, see Fig. 4. These results are more significant for the environments with more open space (*empty* and *window*). Moreover, our results show that the sampling strategy effectively reduces the standard deviation of the cost, making it more likely that we find a high-quality solution quickly.

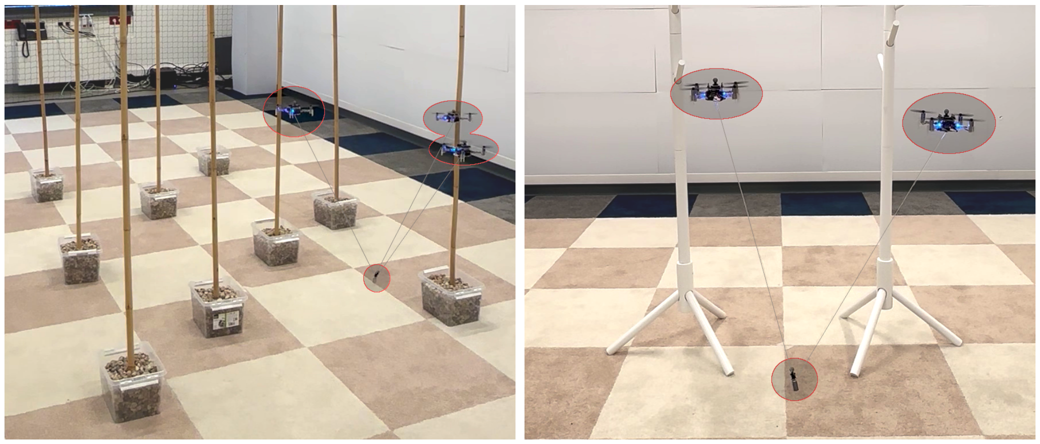
Fig. 5. Real flights validation scenarios. left: forest (3 robots), right: window (2 robots). The payload in both scenarios is modeled as a $10\,\mathrm{g}$ point mass.

For the optimization, we are interested in the scalability with the number of robots. The number of decision variables is linear in the time horizon and the number of robots. Theoretically, trajectory optimization using DDP scales cubically with the state dimensionality (i.e., the number of robots) and linearly with the time horizon. However, we observe that adding robots also results in more nonlinear iterations, increasing the overall computation time, see Fig. 3.

*3) Iterative Optimization (Offline):* To minimize the trajectory duration, we solve a sequence of optimization problems for decreasing values of $\Delta t_0$ (12) using the solutions of each problem as initial guess for the next one. Examples of this approach are apparent in sequential convex programming (SCP) [24] as well as prior motion planning techniques for multirotors [39]. We observe the benefits in our application, as shown in Fig. 3 for the *forest* environment with 4 robots, the estimated energy consumption of the team is reduced by almost 50 % just by repeating the optimization ten times.

## B. Real Flights Results

*1) Physical Setup:* To provide concrete validation to our simulation results, we conduct several real flight tests. For the real platform, we use multirotors of type Bitcraze Crazyflie 2.1 (CF), where we use the same flight controller code [3] running on-board as in SITL. These are small (9 cm rotor-to-rotor) and lightweight ($34\,\mathrm{g}$) products that are commercially available. Controller and extended Kalman filter for state estimation run directly on-board the STM32-based flight controller (168 MHz, 192 kB RAM). For all scenarios, we use dental floss as cables with length $0.5\,\mathrm{m}$ and the payload mass is $10\,\mathrm{g}$. We use magnets to connect the cables and the payload/multirotors to be easily repaired. On the host side, we use Crazyswarm2, which is based on Crazyswarm [40] but uses ROS 2 [41] to control and send commands for multiple CFs. In particular, we equip each multirotor and the payload with a single reflective marker for position tracking at $100\,\mathrm{Hz}$ using an OptiTrack motion capture system in a $7.5 \times 4 \times 2.75\ \mathrm{m}^3$ flight space.

*2) Results:* We validate the functionality and the quality of our kinodynamic planner *opt* against *geom* through especially designed environments, where the state-of-the-art motion planners fail to generate feasible trajectories for the full system that can be tracked by the controller [3]. As shown in Fig. 5, we test the experiments in the window and forest environments for two and three robots. Similar to our prior work [3], we only fly up to three robots due to the computationally constrained microcontroller.

We can generate solutions within a few minutes, see Table II, where in most cases the optimization is the slower part. In the forest case with 3 robots, the geometric planning is computationally expensive due to a high obstacle density. Note that both window and forest examples are different from the simulation results to match our flight space constraints.

As in the simulation evaluation, we compare the tracking error and the energy usage for ten executed flights. Moreover, we compare the success rate of each method in tracking the reference trajectory to the goal state $\boldsymbol{x}_g$ while avoiding inter-robot or robot/obstacle collisions. As shown in Table II, *opt* is succeeding in all cases. However, when tracking the plans generated by the geometric planner we observed failures either by the team crashing into obstacles or the whole team becoming unstable due to infeasibility of the provided reference. In the 3 robot forest case, such tracking failures could even be observed in simulation, explaining the $0\,\%$ success rate.

## VI. Conclusion

We propose a hierarchical kinodynamic motion planning algorithm for the cable-suspended payload transport system. Our method directly considers obstacles, inter-robot collisions, the full dynamics, the actuation limits and allows us to plan feasible reference trajectories that can be tracked accurately by our controller in realtime. We compare our method with the state-of-the-art baselines in multiple simulation and real experiments. In all cases, we achieve high success rates and enhance the energy consumption of the executed motions, thereby maximizing effectiveness.

In the future, we would like to extend our method to the rigid payload and enable realtime planning with an obstacle-aware controller.

## References

[1] C. Gabellieri, M. Tognon, L. Pallottino, and A. Franchi, “A study on force-based collaboration in flying swarms,” in *International Conference on Swarm Intelligence (ICSI)*, 2018, pp. 3–15.

[2] T. Lee, “Geometric control of quadrotor uavs transporting a cable-suspended rigid body,” *IEEE Transactions on Control Systems Technology*, vol. 26, no. 1, pp. 255–264, 2017.

[3] K. Wahba and W. Hönig, “Efficient optimization-based cable force allocation for geometric control of a multirotor team transporting a payload,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 9, no. 4, pp. 3688–3695, 2024.

[4] K. Wahba and W. Hönig, “Motion planning for cable-suspended payload transportation with a team of multirotors in cluttered environments,” *Future of Construction Workshop at ICRA*, 2023.

[5] C. Masone, H. H. Bülthoff, and P. Stegagno, “Cooperative transportation of a payload using quadrotors: A reconfigurable cable-driven parallel robot,” in *International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 1623–1630.

[6] P. O. Pereira and D. V. Dimarogonas, “Control framework for slung load transportation with two aerial vehicles,” in *IEEE Conference on Decision and Control (CDC)*, 2017, pp. 4254–4259.

[7] 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.

[8] T. Lee, K. Sreenath, and V. Kumar, “Geometric control of cooperating multiple quadrotor uavs with a suspended payload,” in *IEEE Conference on Decision and Control (CDC)*, 2013, pp. 5510–5515.

[9] D. Six, S. Briot, A. Chriette, and P. Martinet, “The kinematics, dynamics and control of a flying parallel robot with three quadrotors,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 1, pp. 559–566, 2017.

[10] A. Tagliabue, M. Kamel, R. Siegwart, and J. Nieto, “Robust collaborative object transportation using multiple mavs,” *The International Journal of Robotics Research (IJRR)*, vol. 38, no. 9, pp. 1020–1044, 2019.

[11] M. Tognon, C. Gabellieri, L. Pallottino, and A. Franchi, “Aerial co-manipulation with cables: The role of internal force for equilibria, stability, and passivity,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 3, pp. 2577–2583, 2018.

[12] 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,” *Journal of Intelligent & Robotic Systems*, vol. 105, no. 3, p. 68, 2022.

[13] 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 *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2020, pp. 158–165.

[14] G. Li and G. Loianno, “Nonlinear model predictive control for cooperative transportation and manipulation of cable suspended payloads with multiple quadrotors,” *arXiv preprint arXiv:2303.06165*, 2023.

[15] S. Sun and A. Franchi, “Nonlinear mpc for full-pose manipulation of a cable-suspended load using multiple uavs,” in *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2023, pp. 969–975.

[16] M. Manubens, D. Devaurs, L. Ros, and J. Cortés, “Motion planning for 6-d manipulation with aerial towed-cable systems,” in *Robotics: science and systems (RSS)*, 2013, 8p.

[17] H. G. De Marina and E. Smeur, “Flexible collaborative transportation by a team of rotorcraft,” in *International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1074–1080.

[18] Y. Zhang, J. Xu, C. Zhao, and J. Dong, “If-based trajectory planning and cooperative control for transportation system of cable suspended payload with multi uavs,” in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2023, pp. 635–642.

[19] Y. Li, Z. Littlefield, and K. E. Bekris, “Asymptotically optimal sampling-based kinodynamic planning,” *The International Journal of Robotics Research (IJRR)*, vol. 35, no. 5, pp. 528–564, 2016.

[20] M. Pivtoraiko and A. Kelly, “Kinodynamic motion planning with state lattice motion primitives,” in *International Conference on Robotics and Automation (ICRA)*, 2011, pp. 2172–2179.

[21] D. J. Webb and J. v. d. Berg, “Kinodynamic RRT*: Optimal motion planning for systems with linear differential constraints,” *arXiv preprint arXiv:1205.5088*, 2012.

[22] W. Hönig, J. Ortiz-Haro, and M. Toussaint, “db-A*: Discontinuity-bounded search for kinodynamic mobile robot motion planning,” in *International Conference on Intelligent Robots and Systems (IROS)*, Code Available at: https://github.com/quimortiz/dynoplan and https://github.com/quimortiz/dynobench, 2022, pp. 13 540–13 547.

[23] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard, “Crocoddyl: An efficient and versatile framework for multi-contact optimal control,” in *International Conference on Robotics and Automation (ICRA)*, 2020, pp. 2536–2542.

[24] D. Malyuta, T. P. Reynolds, M. Szmuk, T. Lew, R. Bonalli, M. Pavone, and B. Açıkmeşe, “Convex optimization for trajectory generation: A tutorial on generating dynamically feasible trajectories reliably and efficiently,” *IEEE Control Systems Magazine*, vol. 42, no. 5, pp. 40–113, 2022.

[25] J. Schulman, Y. Duan, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, “Motion planning with sequential convex optimization and convex collision checking,” *The International Journal of Robotics Research (IJRR)*, vol. 33, no. 9, pp. 1251–1270, 2014.

[26] T. A. Howell, B. E. Jackson, and Z. Manchester, “Altro: A fast solver for constrained trajectory optimization,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 7674–7679.

[27] D. Pardo, L. Möller, M. Neunert, A. W. Winkler, and J. Buchli, “Evaluating direct transcription and nonlinear optimization methods for robot motion planning,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 1, no. 2, pp. 946–953, 2016.

[28] M. A. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum, “Differentiable physics and stable modes for tool-use and manipulation planning,” in *Robotics: science and systems (RSS)*, 2018.

[29] B. Ponton, A. Herzog, A. Del Prete, S. Schaal, and L. Righetti, “On time optimization of centroidal momentum dynamics,” in *International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5776–5782.

[30] J. Carpentier and N. Mansard, “Multicontact locomotion of legged robots,” *IEEE Transactions on Robotics (T-RO)*, vol. 34, no. 6, pp. 1441–1460, 2018.

[31] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, “Gait and trajectory optimization for legged systems through phase-based end-effector parameterization,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 3, pp. 1560–1567, 2018.

[32] P. Foehn, D. Falanga, N. Kuppuswamy, R. Tedrake, and D. Scaramuzza, “Fast trajectory optimization for agile quadrotor maneuvers with a cable-suspended payload,” *Robotics: science and systems (RSS)*, 2017.

[33] M. Geisert and N. Mansard, “Trajectory generation for quadrotor based systems using numerical optimal control,” in *International Conference on Robotics and Automation (ICRA)*, 2016, pp. 2958–2964.

[34] W. Li and E. Todorov, “Iterative linear quadratic regulator design for nonlinear biological movement systems,” in *International Conference on Informatics in Control, Automation and Robotics*, 2004, pp. 222–229.

[35] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer, 1999.

[36] I. A. Sucan, M. Moll, and L. E. Kavraki, “The open motion planning library,” *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[37] J. Pan, S. Chitta, and D. Manocha, “FCL: A general purpose library for collision and proximity queries,” in *International Conference on Robotics and Automation (ICRA)*, 2012, pp. 3859–3866.

[38] A. Meurer *et al.*, “Sympy: Symbolic computing in python,” *PeerJ Computer Science*, vol. 3, e103, 2017.

[39] W. Hönig, J. A. Preiss, T. K. S. Kumar, G. S. Sukhatme, and N. Ayanian, “Trajectory planning for quadrotor swarms,” *IEEE Transactions on Robotics (T-RO)*, vol. 34, no. 4, pp. 856–869, 2018.

[40] J. A. Preiss, W. Hönig, G. S. Sukhatme, and N. Ayanian, “Crazyswarm: A large nano-quadcopter swarm,” in *International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3299–3304.

[41] 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.