Real-time Dynamic-consistent Motion Planning for Over-actuated UAVs

Yao Su[1*], Jingwen Zhang[1*], Ziyuan Jiao[1], Hang Li[1], Meng Wang[1], and Hangxin Liu[1†]

***Abstract*— Existing motion planning approaches for over-actuated unmanned aerial vehicle (UAV) platforms can achieve online planning without considering dynamics. However, in many envisioned application areas such as aerial manipulation, payload delivery, and moving target tracking, it is critical to ensure dynamic consistency in the generated trajectory. The dynamics of these platforms introduce a high nonlinearity, leading to a substantial increase in computational burden. This paper presents an efficient method to plan motions that are consistent with the dynamics of over-actuated UAVs. With a hierarchical control structure, the dimension of the optimization problem is greatly reduced with synthesized wrench commands. Additionally, by exploring the dynamics of over-actuated UAVs, the complex planning process is decoupled into two simpler sub-problems. As a result, the proposed planner can be solved as two small quadratic programmings (QPs) and deployed in real-time. The computational efficiency and dynamic consistency of the proposed method are verified through both simulations and experiments, including comparison with other approaches and dynamic target tracking.**

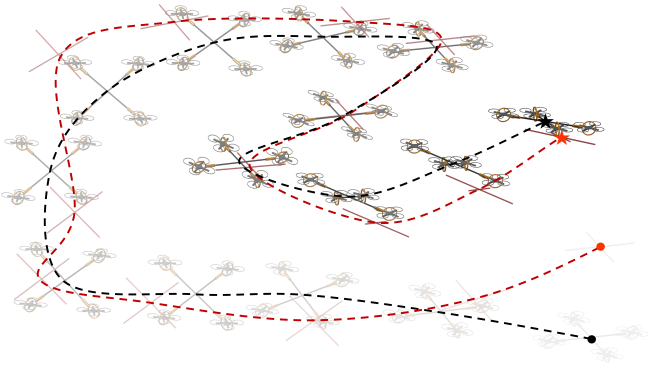

Fig. 1: **Utilizing the proposed dynamic-consistent motion planner, the over-actuated UAV platform successfully tracks a dynamic target in real-time.** The trajectories of the target and over-actuated UAV are plotted in red and black dash lines, respectively. Corresponding initial and final configurations are plotted as dots and stars, respectively.

I. Introduction

With the ability to generate thrust and torque in an arbitrary direction, over-actuated UAV platforms [1–3] emerged in the past decade to overcome the under-actuation issue of traditional collinear UAVs. Extensive research has been conducted on these platforms, including hardware design [2–5], dynamics modeling [6–8], control algorithms [9–11], aerial manipulation integration [12–15] and modularization [16, 17] *etc*. Despite the decoupling of position and attitude control by these over-actuated UAV platforms, providing more controllable degree-of-freedoms (DoFs) for motion planners [18–20], the methods to generate feasible trajectories within their dynamics constraints have not been extensively investigated yet.

The existing motion planning frameworks of over-actuated UAV can be categorized into two groups according to how they handle the **highly nonlinear dynamic constraints**. The first group [18–20] formulates the planning problem as a QP, utilizing polynomial motion primitives for computational efficiency improvement with **N**o **D**ynamic constraints. This approach referred to as the **ND** planner in this paper, typically employs conservative input constraints for feasibility assurance and therefore fails to fully exploit the dynamic capabilities of over-actuated UAV platforms. The other group [21] directly includes all constraints into consideration and formulates the planning problem as a nonlinear programming (NLP), which is referred to as the **NLP** planner, which unfortunately suffers from low computational efficiency even with state-of-the-art optimizers [22].

In this paper, we investigate the motion planning problem of over-actuated UAV using a customized platform proposed in [4]. The platform integrates four novel omni-directional thrust generators, each consisting of a mini-quadcopter connected to a 2-DoF passive gimbal mechanism. Based on the dynamics of this platform, we first review and analyze the formulations of the existing ND planner and NLP planner. Then we introduce a dynamic-consistent real-time motion planner via **T**wo-step **Q**Ps, referred to as the **TQ** planner, which (i) replaces low-level control inputs with total wrench commands, reducing the dimension of the optimization problem and (ii) decouples the original problem into two sub-problems, eliminating the nonlinearity of the rotation matrix and reducing the size of decision variables. Taken together, the proposed planner offers high-frequency solvability and incorporates dynamics constraints, thus combining the merits of two existing planners.

By implementing the three planners in both realistic simulation and real-world experiments, we validate that the proposed TQ planner successfully combines the merits of both the ND and NLP planners while mitigating their limitations. Specifically, the TQ planner successfully includes nonlinear dynamics constraints along with integration, boundary, and actuator constraints in the formulation while performing QP-level solving efficiency. As a result, it can be applied in both feed-forward and real-time scenarios, such as dynamic object tracking (demonstrated in Fig. 1), enlarging the application scope of over-actuated UAVs from static to dynamic environments.

* Yao Su and Jingwen Zhang contributed equally to this work. Emails: {suyao, zhangjingwen}@bigai.ai. † Corresponding author. Email: liuhx@bigai.ai. [1] National Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence (BIGAI).

A. Related Work

In recent years, **optimization-based motion planning and control frameworks** have been proposed, as they can consider kinematics, dynamics, physical and contact constraints through the optimization process. These frameworks have found success in various mobile robots, such as humanoid robots [23–25], quadruped robots [26, 27], and hexapod robots [28, 29], and quadcopters [30, 31]. However, the high nonlinearity of constraints impedes the solving efficiency and often necessitates sacrificing certain features in this framework to enhance computational speed. For example, the complete multi-body dynamics of robots is simplified as a single rigid body in [32, 33]; the planning/control horizon is limited to less than 10 steps in [24, 34]; the system input constraints are neglected by [35]; and certain motion primitives are given in [21], *etc*. This study aims to develop a motion planning framework for over-actuated UAVs that allows independent planning and control of the six DoF of the mainframe. This capability enables more agile maneuvering within the UAV's dynamical capability.

Existing motion planners of **over-actuated UAV** primarily focus on the computing efficiency [19], collision avoidance [20] and time optimality [18] without considering the dynamical capability of the platforms. Specifically, Brescianini *et al*. [19] employed polynomial motion primitives to plan the motion of the over-actuated UAV with a specified time horizon, which features a closed-form solution and enables real-time trajectory generation. Liu *et al*. initially formulated a QP problem for spatial trajectory generation, followed by a Convex Second Order Cone Programming (SOCP) for temporal trajectory generation with minimum time. The generation of collision-free trajectory was investigated in [20] which constructed whole-body safety constraints based on the safe flight corridor of a convex polyhedron. Morbidi *et al*. [21] introduced the only planning framework that incorporates the full dynamics constraints of over-actuated UAVs and formulates an NLP problem. Although energy optimality is ensured, the computation speed is dramatically sacrificed. In this work, we proposed a real-time approach for the nonlinear formulation, where the low-level control inputs are replaced by the total wrench commands for decision variable minimizatiohn, and the nonlinearity is eliminated by decoupling into two subsequent QP problems.

B. Overview

We organize the remainder of the paper as follows. Sec. II describe the dynamics and control of the over-actuated UAV system. Sec. III presents the planning framework. Secs. IV and V show the simulation and experiment results of the proposed motion planning framework with comprehensive evaluations. Finally, we conclude the paper in Sec. VI.

II. Platform Dynamics & Control

The over-actuated UAV system (see Fig. 2) discussed in this paper adopts regular quadcopters with 2-DoF passive gimbal mechanism, serving as omni-directional thrust generators [4, 7]. Each thrust generator i outputs 3-DoF to the system: the magnitude of thrust T_i, tilting angle α_i, and twisting angle β_i, resulting in a system with 12 DoFs in total.

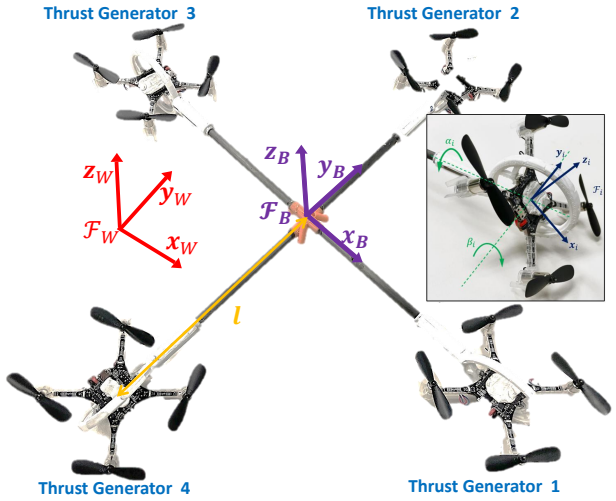

Fig. 2: **The coordination system of the over-actuated UAV platform.** Four normal quadcopters are mounted on the 2-DoF gimbal mechanisms to provide omni-directional thrust generation capability.

A. System Frames Definition & Notation

Let $\mathcal{F}_W$ denote the world coordinate frame and attach the platform frame $\mathcal{F}_B$ to its geometric center. We define the central position of the main frame as $\boldsymbol{X} = [x,\, y,\, z]^\mathsf{T}$, the attitude in the roll-pitch-yaw convention as $\boldsymbol{\Theta} = [\phi,\, \theta,\, \psi]^\mathsf{T}$, and the platform angular velocity in $\mathcal{F}_B$ as $\boldsymbol{\Omega} = [p,\, q,\, r]^\mathsf{T}$. Thrust generator frames $\mathcal{F}_i$s are attached to the geometric center of the ith omni-directional thrust generator. $\boldsymbol{d}_i = [x_i,\, y_i,\, 0]^\mathsf{T}$ denotes the vector from thrust generator frame $\mathcal{F}_i$ to platform frame $\mathcal{F}_B$. l is the arm length from each thrust generator to the mainframe.

B. Platform Dynamics

The dynamics model of this over-actuated UAV platform can be described as [4]:

$$\begin{bmatrix} m\,{}^{W}\ddot{\boldsymbol{X}} \\ {}^{B}\boldsymbol{J}\,{}^{B}\dot{\boldsymbol{\Omega}} \end{bmatrix} = \begin{bmatrix} {}^{W}_{B}\boldsymbol{R} & 0 \\ 0 & \boldsymbol{I}_3 \end{bmatrix} \boldsymbol{u} + \begin{bmatrix} mg\hat{\boldsymbol{z}} \\ {}^{B}\boldsymbol{\tau}_g \end{bmatrix} + {}^{\text{ext}}\boldsymbol{u}, \tag{1}$$

where m and $\boldsymbol{J}$ are the total mass and inertia matrix of the platform, respectively. $\ddot{\boldsymbol{X}}$ and $\dot{\boldsymbol{\Omega}}$ are the linear and angular acceleration of the mainframe, respectively. g is the acceleration due to gravity, ${}^{B}\boldsymbol{\tau}_g$ is the gravity torque due to the displacement of its center of mass (CoM) from the geometric center [36], $\hat{\boldsymbol{z}} = [0,\, 0,\, 1]^\mathsf{T}$, and total wrench

$$\boldsymbol{u} = \begin{bmatrix} \sum_{i=1}^{N} {}^{B}_{i}\boldsymbol{R}\, T_i \hat{\boldsymbol{z}} \\ \sum_{i=1}^{N} (\boldsymbol{d}_i \times {}^{B}_{i}\boldsymbol{R}\, T_i \hat{\boldsymbol{z}}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_X(\boldsymbol{\alpha}, \boldsymbol{\beta}) \\ \boldsymbol{J}_\Omega(\boldsymbol{\alpha}, \boldsymbol{\beta}) \end{bmatrix} \boldsymbol{T}, \tag{2}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]^\mathsf{T}$, $\boldsymbol{\beta} = [\beta_1, \beta_2, \beta_3, \beta_4]^\mathsf{T}$, $\boldsymbol{T} = [T_1, T_2, T_3, T_4]^\mathsf{T}$. $\boldsymbol{J}_X$ and $\boldsymbol{J}_\Omega$ are two nonlinear mapping matrices whose detailed definitions can be found in our previous work [9]. ${}^{\text{ext}}\boldsymbol{u}$ is the external force/torque disturbances [7].

C. Control

A hierarchical control architecture is designed (see Fig. 3): (i) The high-level control of the platform is centralized which

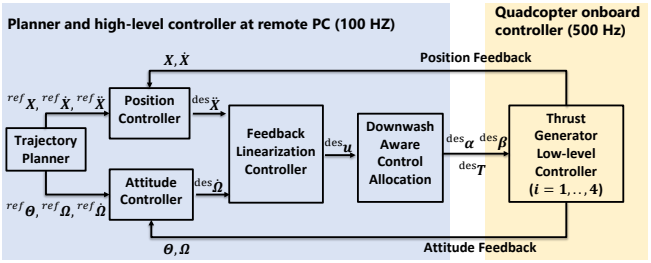

Fig. 3: **Hierarchical control architecture of the over-actuated UAV platform**. The high-level controller of the over-actuated UAV (i) calculates the desired wrench command ${}^{\text{des}}\boldsymbol{u}$ for trajectory tracking and (ii) allocates it to desired thrusts and joint angles of the thrust generators. Each quadcopter has its onboard low-level controller to regulate the joint angles and thrust to desired values.

first calculates the desired total wrench command with the position and attitude controller, the wrench command is then allocated as desired tilting and twisting angles and thrust forces of the thrust generators; (ii) The low-level quadcopter control is decentralized which regulates the attitude and magnitude of thrust force as required with fast response [7, 11].

a) High-level Trajectory Tracking Control: According to Eq. (1), a feedback-linearization controller is designed:

$$ {}^{\text{des}}\boldsymbol{u} = \begin{bmatrix} {}^{\text{des}}\boldsymbol{u}_X \\ {}^{\text{des}}\boldsymbol{u}_\Omega \end{bmatrix} = \begin{bmatrix} m\, {}^{W}_{B}\boldsymbol{R}^{\mathsf{T}} \left({}^{\text{des}}\ddot{\boldsymbol{X}} - g\hat{\boldsymbol{z}} \right) \\ {}^{B}\boldsymbol{J}\, {}^{\text{des}}\dot{\boldsymbol{\Omega}} - {}^{B}\boldsymbol{\tau}_g \end{bmatrix}, \tag{3} $$

where the superscript des indicates the desired values. This controller transfers the dynamics of the platform into a simple double integrator [37]. The desired linear and angular accelerations are designed as follows:

$$ \begin{aligned} {}^{\text{des}}\ddot{\boldsymbol{X}} &= {}^{\text{ref}}\ddot{\boldsymbol{X}} + \boldsymbol{K}_{X1}\dot{\boldsymbol{e}}_X + \boldsymbol{K}_{X2}\boldsymbol{e}_X + \boldsymbol{K}_{X3}\int \boldsymbol{e}_X dt, \\ {}^{\text{des}}\dot{\boldsymbol{\Omega}} &= {}^{\text{ref}}\dot{\boldsymbol{\Omega}} + \boldsymbol{K}_{\Omega1}\boldsymbol{e}_\Omega + \boldsymbol{K}_{\Omega2}\boldsymbol{e}_\Theta + \boldsymbol{K}_{\Omega3}\int \boldsymbol{e}_\Theta dt, \end{aligned} \tag{4} $$

where $\boldsymbol{K}_{Xi}$ and $\boldsymbol{K}_{\Omega i}$ are constant gain matrices, the superscript ref indicates the reference value from motion planner, the error terms are defined as introduced in [10, 37].

b) Control Allocation & Low-level Control: The control allocation of over-actuated UAV platforms determines the low-level commands $({}^{\text{des}}\alpha_i,\ {}^{\text{des}}\beta_i,\ {}^{\text{des}}T_i)$ for each omni-directional thrust generator based on the desired total wrench ${}^{\text{des}}\boldsymbol{u}$ through a nonlinear mapping (see Eq. (2)). Here, we implement the downwash-aware control allocation framework proposed by Su *et al.* in [7, 9] to avoid the downwash effect and maintain high thrust efficiency. In low-level control of each quadcopter, the desired tilting and twisting angles, and thrust force are regulated with a faster control loop [4].

III. Motion Planning

This section starts with outlining two existing motion planning approaches of over-actuated UAVs: (i) the **NLP** planner with a single complete formulation [21] and (ii) the **ND** planner without dynamics [19]. Their advantages and disadvantages are discussed, which motivates us to propose the **TQ** planner, a dynamic-consistent real-time planning framework that combines the merits of these two formulations.

A. The *NLP* Formulation

1) Decision Variables: To ensure the smoothness of the trajectory, the decision variables are determined as:

$$ \begin{aligned} \boldsymbol{\Gamma} = \{ &\boldsymbol{X}_{[k]}, \dot{\boldsymbol{X}}_{[k]}, \ddot{\boldsymbol{X}}_{[k]}, \dddot{\boldsymbol{X}}_{[k]}, \boldsymbol{\Theta}_{[k]}, \boldsymbol{\Omega}_{[k]}, \dot{\boldsymbol{\Omega}}_{[k]}, \\ &\boldsymbol{\alpha}_{[k]}, \boldsymbol{\beta}_{[k]}, \boldsymbol{T}_{[k]}, \dot{\boldsymbol{\alpha}}_{[k]}, \dot{\boldsymbol{\beta}}_{[k]} \quad \forall k = 1, \cdots, N \} \in \mathbb{R}^{41 \times N} \end{aligned} \tag{5} $$

where $\dddot{\boldsymbol{X}}$ is the jerk of translational movement [18, 19], the subscript $[k]$ indicates the kth timestep of a variable,

2) Constraints: This formulation considers four types of constraints: integration constraints, boundary constraints, physical constraints, and dynamics constraints to ensure trajectory consistency, the required initial and final configurations, and the tracking feasibility of the generated trajectory.

Dynamics Constraints ($\forall k = 1, \cdots, N$)**:**

$$ \begin{bmatrix} \ddot{\boldsymbol{X}}_{[k]} \\ \dot{\boldsymbol{\Omega}}_{[k]} \end{bmatrix} = \begin{bmatrix} \frac{1}{m}\boldsymbol{R}(\boldsymbol{\Theta}_{[k]}) & 0 \\ 0 & \boldsymbol{J}^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{J}_X(\boldsymbol{\alpha}_{[k]}, \boldsymbol{\beta}_{[k]}) \\ \boldsymbol{J}_\Omega(\boldsymbol{\alpha}_{[k]}, \boldsymbol{\beta}_{[k]}) \end{bmatrix} \boldsymbol{T}_{[k]} + \begin{bmatrix} g\hat{\boldsymbol{z}} \\ \boldsymbol{J}^{-1}\boldsymbol{\tau}_g \end{bmatrix} \tag{6} $$

Integration Constraints ($\forall k = 2, \cdots, N$)**:**

$$ \Delta\boldsymbol{X}_{[k]} = \dot{\boldsymbol{X}}_{[k]}dt, \Delta\dot{\boldsymbol{X}}_{[k]} = \ddot{\boldsymbol{X}}_{[k]}dt, \Delta\ddot{\boldsymbol{X}}_{[k]} = \dddot{\boldsymbol{X}}_{[k]}dt, \tag{7a} $$

$$ \boldsymbol{R}_{[k]} = \boldsymbol{R}_{[k-1]} e^{\hat{\boldsymbol{\Omega}}_{[k]}dt}, \tag{7b} $$

$$ \Delta\boldsymbol{\Omega}_{[k]} = \dot{\boldsymbol{\Omega}}_{[k]}dt, \tag{7c} $$

$$ \Delta\boldsymbol{\alpha}_{[k]} = \dot{\boldsymbol{\alpha}}_{[k]}dt, \Delta\boldsymbol{\beta}_{[k]} = \dot{\boldsymbol{\beta}}_{[k]}dt, \tag{7d} $$

where $\Delta[\cdot]$ is the difference w.r.t. previous timestep of a variable, $[\cdot]$ is the mapping from $\mathbb{R}^3$ to $SO(3)$, and dt is the time interval between adjacent timesteps.

Boundary Constraints:

$$ \boldsymbol{X}_{[1]} = \boldsymbol{X}_{[s]},\ \dot{\boldsymbol{X}}_{[1]} = \dot{\boldsymbol{X}}_{[s]},\ \ddot{\boldsymbol{X}}_{[1]} = \ddot{\boldsymbol{X}}_{[s]}, \tag{8a} $$

$$ \boldsymbol{X}_{[N]} = \boldsymbol{X}_{[e]},\ \dot{\boldsymbol{X}}_{[N]} = \dot{\boldsymbol{X}}_{[e]},\ \ddot{\boldsymbol{X}}_{[N]} = \ddot{\boldsymbol{X}}_{[e]}, \tag{8b} $$

$$ \boldsymbol{\Theta}_{[1]} = \boldsymbol{\Theta}_{[s]},\ \boldsymbol{\Omega}_{[1]} = \boldsymbol{\Omega}_{[s]},\ \dot{\boldsymbol{\Omega}}_{[1]} = \dot{\boldsymbol{\Omega}}_{[s]}, \tag{8c} $$

$$ \boldsymbol{\Theta}_{[N]} = \boldsymbol{\Theta}_{[e]},\ \boldsymbol{\Omega}_{[N]} = \boldsymbol{\Omega}_{[e]},\ \dot{\boldsymbol{\Omega}}_{[N]} = \dot{\boldsymbol{\Omega}}_{[e]}, \tag{8d} $$

$$ \boldsymbol{\alpha}_{[1]} = \boldsymbol{\alpha}_{[s]},\ \boldsymbol{\beta}_{[1]} = \boldsymbol{\beta}_{[s]}, \tag{8e} $$

where the subscript $[s]$ and $[e]$ indicate the given initial and final value of a variable, respectively.

Actuator Constraints ($\forall k = 1, \cdots, N$)**:**

$$ 0 \le \boldsymbol{T}_{[k]} \le \boldsymbol{T}_{\max},\ -\dot{\boldsymbol{\alpha}}_{\max} \le \dot{\boldsymbol{\alpha}}_{[k]} \le \dot{\boldsymbol{\alpha}}_{\max},\ -\dot{\boldsymbol{\beta}}_{\max} \le \dot{\boldsymbol{\beta}}_{[k]} \le \dot{\boldsymbol{\beta}}_{\max}, \tag{9} $$

where the subscript max refers to the maximum value of a variable. It should be noted that the thrust control $\boldsymbol{T}$ is assumed to be sufficiently fast, therefore $\dot{\boldsymbol{T}}$ is not included as a decision variable.

3) Complete Formulation: The complete NLP formulation is summarized as follows:

$$ \min_{\boldsymbol{\Gamma}} \sum_{k=1}^{N} \gamma_1 \left\| \dddot{\boldsymbol{X}}_{[k]} \right\|^2 + \gamma_2 \left\| \dot{\boldsymbol{\Omega}}_{[k]} \right\|^2 + \gamma_3 \left\| \boldsymbol{T}_{[k]} \right\|^2 \tag{10a} $$

$$ s.t. \quad Eqs.\ (6) to\ (9) \tag{10b} $$

where γ_{1-3} are the weighting parameters. In this formulation, all the dynamic constraints are included to ensure the feasibility of the generated trajectory. However, incorporating low-level control variables and their derivatives $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{T}, \dot{\boldsymbol{\alpha}}, \dot{\boldsymbol{\beta}})$ as decision variables increases the dimension of the solution space. Moreover, highly nonlinear dynamic constraints Eq. (6) dramatically enhance the complexity of the formulated optimization problem. Therefore, this formulation is impractical in real-time applications.

B. The ND Formulation

On the other hand, the motion planning framework without considering dynamic constraints is formulated as:

$$\min_{\mathbf{\Gamma}_p} \quad \sum_{k=1}^{N} \lambda_1 \left\| \dddot{\boldsymbol{X}}_{[k]} \right\|^2 + \lambda_2 \left\| \dot{\boldsymbol{\Omega}}_{[k]} \right\|^2 \tag{11a}$$

$$s.t. \quad \dot{\boldsymbol{X}}_{[k]} \leqslant \dot{\boldsymbol{X}}_{\max}, \ddot{\boldsymbol{X}}_{[k]} \leqslant \ddot{\boldsymbol{X}}_{\max}, \forall k = 1, \cdots, N \tag{11b}$$

$$\boldsymbol{\Omega}_{[k]} \leqslant \boldsymbol{\Omega}_{\max}, \dot{\boldsymbol{\Omega}}_{[k]} \leqslant \dot{\boldsymbol{\Omega}}_{\max}, \forall k = 1, \cdots, N \tag{11c}$$

$$Eqs.\ (7a) to\ (7c) and\ (8a) to\ (8d), \tag{11d}$$

where λ_{1-2} are the weighting parameters, and

$$\mathbf{\Gamma}_p = \{\boldsymbol{X}_{[k]}, \dot{\boldsymbol{X}}_{[k]}, \ddot{\boldsymbol{X}}_{[k]}, \dddot{\boldsymbol{X}}_{[k]}, \boldsymbol{\Theta}_{[k]}, \boldsymbol{\Omega}_{[k]}, \dot{\boldsymbol{\Omega}}_{[k]}, \forall k = 1, \cdots, N\}. \tag{12}$$

By neglecting the dynamic constraints Eq. (6), this formulation [19] exhibits reduced dimensionality and nonlinearity compared to the **NLP** formulation. Consequently, it can be solved as a QP with conservative velocity and acceleration constraints Eqs. (11b) and (11c) to ensure feasible tracking. However, this **ND** formulation may produce infeasible trajectories within a restricted time horizon due to the ignorance of the platform's real dynamic capabilities.

C. The Proposed Real-time TQ Formulation

To leverage the advantages of both **NLP** and **ND** formulations, we first replace the thrust generator commands $(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{T}, \dot{\boldsymbol{\alpha}}, \dot{\boldsymbol{\beta}})$ with the total wrench command $\boldsymbol{u}$ for variable minimization. This substitution effectively eliminates the nonlinear mapping Eq. (2) from the dynamics constraints Eq. (6). Then we decouple the rotational and translational DoFs with a two-step optimization formulation which further reduces the dimensionality of the problem and eliminates the nonlinearity introduced by the rotation matrix.

1) Decision Variables: We utilize the wrench command as control input and decouple the decision variables into two groups according to translational and rotational DoFs as:

$$\begin{aligned} \mathbf{\Gamma}_X &= \{\boldsymbol{X}_{[k]}, \dot{\boldsymbol{X}}_{[k]}, \ddot{\boldsymbol{X}}_{[k]}, \dddot{\boldsymbol{X}}_{[k]}, \boldsymbol{u}_{X[k]} \quad \forall k = 1, \cdots, N\}, \\ \mathbf{\Gamma}_\Omega &= \{\boldsymbol{\Theta}_{[k]}, \boldsymbol{\Omega}_{[k]}, \dot{\boldsymbol{\Omega}}_{[k]}, \boldsymbol{u}_{\Omega[k]} \quad \forall k = 1, \cdots, N\}. \end{aligned} \tag{13}$$

2) Two Steps Optimization: In the dynamics of over-actuated UAV platform (Eq. (1)), the rotational dynamics are not affected by the translational dynamics, whereas the translational dynamics are affected by the rotational dynamics through the rotation matrix $\boldsymbol{R}(\boldsymbol{\Theta})$. Therefore, we first solve the rotational trajectory planning problem:

$$\min_{\mathbf{\Gamma}_\Omega} \quad \left\| \begin{bmatrix} \boldsymbol{\Theta}_{[N]} - \boldsymbol{\Theta}_{[e]} \\ \boldsymbol{\Omega}_{[N]} - \boldsymbol{\Omega}_{[e]} \\ \dot{\boldsymbol{\Omega}}_{[N]} - \dot{\boldsymbol{\Omega}}_{[e]} \end{bmatrix} \right\|_{\boldsymbol{W}_1}^2 + \sum_{k=1}^{N} \left(\left\| \dot{\boldsymbol{\Omega}}_{[k]} \right\|_{\boldsymbol{Q}_1}^2 + \left\| \boldsymbol{u}_{\Omega[k]} \right\|_{\boldsymbol{R}_1}^2 \right) \tag{14a}$$

$$s.t. \quad \dot{\boldsymbol{\Omega}}_{[k]} = \boldsymbol{J}^{-1} \boldsymbol{u}_{\Omega[k]} + \boldsymbol{J}^{-1} \boldsymbol{\tau}_g \quad \forall k = 1, \cdots, N \tag{14b}$$

$$l\, T_{\max} \begin{bmatrix} -2 \\ -2 \\ -4 \end{bmatrix} \leqslant \boldsymbol{u}_{\Omega[k]} \leqslant l\, T_{\max} \begin{bmatrix} 2 \\ 2 \\ 4 \end{bmatrix} \quad \forall k = 1, \cdots, N \tag{14c}$$

$$Eqs.\ (7b), (7c) and\ (8c) \tag{14d}$$

where $\|\cdot\|_{\boldsymbol{W}}$ represents the weighted Euclidean norm of a vector, $\boldsymbol{W}_1$, $\boldsymbol{Q}_1$ and $\boldsymbol{R}_1$ are weighting matrices. Note that the original hard constraints Eq. (8d) are treated as soft constraints and included in the cost function Eq. (14a) as an additional terminal cost for numerical stability. Specifically, if the commanded final state is unreachable given the limits of control inputs, the formulation Eq. (14) would find the optimal trajectory to the closest state instead of reporting infeasibility. As a result, real-time consistency can be guaranteed.

With the optimized rotational trajectory $\boldsymbol{\Theta}^*$, we calculate the corresponding rotation matrix $\boldsymbol{R}(\boldsymbol{\Theta}^*)$. And then solve the following translational planning problem with the predescribed modification on the cost function:

$$\min_{\mathbf{\Gamma}_X} \quad \left\| \begin{bmatrix} \boldsymbol{X}_{[N]} - \boldsymbol{X}_{[e]} \\ \dot{\boldsymbol{X}}_{[N]} - \dot{\boldsymbol{X}}_{[e]} \\ \ddot{\boldsymbol{X}}_{[N]} - \ddot{\boldsymbol{X}}_{[e]} \end{bmatrix} \right\|_{\boldsymbol{W}_2}^2 + \sum_{k=1}^{N} \left(\left\| \ddot{\boldsymbol{X}}_{[k]} \right\|_{\boldsymbol{Q}_2}^2 + \left\| \boldsymbol{u}_{X[k]} \right\|_{\boldsymbol{R}_2}^2 \right) \tag{15a}$$

$$s.t. \quad \ddot{\boldsymbol{X}}_{[k]} = \frac{1}{m} \boldsymbol{R}(\boldsymbol{\Theta}^*_{[k]}) \boldsymbol{u}_{X[k]} + g\hat{\boldsymbol{z}} \quad \forall k = 1, \cdots, N \tag{15b}$$

$$-4T_{\max} \leqslant \boldsymbol{u}_{X[k]} \leqslant 4T_{\max} \quad \forall k = 1, \cdots, N \tag{15c}$$

$$Eqs.\ (7a) and\ (8a) \tag{15d}$$

Similarly, the original hard constraints Eq. (8a) are treated as soft constraints and included in the cost function Eq. (15a), $\boldsymbol{W}_2$, $\boldsymbol{Q}_2$ and $\boldsymbol{R}_2$ are weighting matrices. Both Eq. (14c) and Eq. (15c) are actuator constraints in which the lower and upper bounds are estimated from Eq. (2) by sweeping all possible tilting and twisting angles. With $\boldsymbol{R}(\boldsymbol{\Theta}^*)$ known, this problem can be efficiently solved as a QP problem.

After reformulating the original problem in Sec. III-A with the wrench command $\boldsymbol{u}$ and decoupling it into two steps, the nonlinearity only remains in the Eq. (7b) which is the integration of the rotation matrix. To meet the real-time requirement, Eq. (7b) is linearized based on the variation on the non-Euclidean *SO* (3) manifold in Sec. III-C.3.

3) Linearization of the Rotational Dynamics: Nonlinear dynamics can be linearized around a fixed operating point which requires extra consideration to minimize the accumulated approximated errors. Due to the excellent tracking performance of the controller described in Sec. II-C for our system, we choose to linearize around the reference trajectory to simplify the implementation. Meanwhile, we choose to use the rotation matrix representation to avoid the singularity problem of Euler angles and the unwinding phenomenon of quaternion representation. The nonlinear dynamic equation of the rotation matrix is given as:

$$\dot{\boldsymbol{R}} = \boldsymbol{R}\hat{\boldsymbol{\omega}}. \tag{16}$$

Assuming the predicted rotation matrix is close to the reference point, the variation could be approximated by $\delta\boldsymbol{R}$ using the derivative of the error function on *SO*(3) as in [38]. Then the discrete integral Eq. (7b) can be further approximated with the first-order Taylor expansion of the matrix exponential map assuming that $\delta\boldsymbol{R}$ is small,

$$\boldsymbol{R}_{[k]} \approx \boldsymbol{R}_{[k-1]} e^{\delta \boldsymbol{R}_{[k]}} \approx \boldsymbol{R}_{[k-1]} (\mathbf{I}_{3\times 3} + \delta \boldsymbol{R}_{[k]}). \tag{17}$$

Following the manner of [39], the discrete integration of the variation can be expressed as:

$$\delta \boldsymbol{R}_{[k]} = \delta \boldsymbol{R}_{[k-1]} + {}^{\text{ref}}\boldsymbol{R}^{\mathsf{T}} \dot{\boldsymbol{R}}_{[k]} dt, \tag{18}$$

and the derivative of the rotation matrix is derived as

$$\dot{\boldsymbol{R}}_{[k]} = {}^{\text{ref}}\boldsymbol{R}\, {}^{\text{ref}}\hat{\boldsymbol{\Omega}} + {}^{\text{ref}}\boldsymbol{R}\, {}^{\text{ref}}\hat{\boldsymbol{\Omega}} \delta \boldsymbol{R}_{[k]} + {}^{\text{ref}}\boldsymbol{R} \widehat{\delta \boldsymbol{\Omega}_{[k]}}. \tag{19}$$

However, for practical implementation, it is difficult to use matrix variables in the standard QP formulation. Vectorization is performed with a vector $\boldsymbol{\xi} \in \mathbb{R}^3$ and $\hat{\boldsymbol{\xi}} = \delta\boldsymbol{R}$ as in [40]. With the constant mapping matrix $\boldsymbol{N} \in \mathbb{R}^{9\times 3}$ in

$vec(\delta \boldsymbol{R}) = \boldsymbol{N}\boldsymbol{\xi}$, Eq. (18) can be rewritten as

$$\boldsymbol{\xi}_{[k]} = \boldsymbol{\xi}_{[k-1]} + dt\boldsymbol{N}^{\star}(\mathbf{I} \otimes {}^{\text{ref}}\boldsymbol{R}^{\mathsf{T}})(\boldsymbol{C}_1 + \boldsymbol{C}_2\boldsymbol{\xi}_{[k]} + \boldsymbol{C}_3\boldsymbol{\Omega}_{[k]}) \quad (20)$$

where $\otimes$ denotes the Kronecker tensor operator and constant matrices are defined as:

$$\begin{aligned} \boldsymbol{C}_1 &= vec({}^{\text{ref}}\boldsymbol{R}\,{}^{\text{ref}}\hat{\boldsymbol{\Omega}}) - (\mathbf{I} \otimes {}^{\text{ref}}\boldsymbol{R}^{\mathsf{T}})\boldsymbol{N}\,{}^{\text{ref}}\boldsymbol{\Omega}, \\ \boldsymbol{C}_2 &= (\mathbf{I} \otimes {}^{\text{ref}}\boldsymbol{R}\,{}^{\text{ref}}\hat{\boldsymbol{\Omega}})\boldsymbol{N} - (\mathbf{I} \otimes {}^{\text{ref}}\boldsymbol{R}^{\mathsf{T}})\boldsymbol{N}\,{}^{\text{ref}}\hat{\boldsymbol{\Omega}}, \\ \boldsymbol{C}_3 &= (\mathbf{I} \otimes {}^{\text{ref}}\boldsymbol{R}^{\mathsf{T}})\boldsymbol{N}. \end{aligned} \quad (21)$$

Refer to [40] for detailed derivation. To rewrite the problem Eq. (14) as a QP, the original nonlinear constraint Eq. (7b) is replaced with the linearized Eq. (20) and the variation of the rotation matrix $\boldsymbol{\xi}$ is included as alternative decision variables to represent the attitude.

IV. Simulation

A. Simulation Setup

A simulation platform is developed in Matlab Simulink to evaluate and characterize the proposed real-time motion planning framework of over-actuated UAVs [7]. Three cases are designed to study the performance of our proposed **TQ** planner and compare it with the **NLP** planner (Sec. III-A), which can be solved as nonlinear programming, and the **ND** formulation (Sec. III-B) but with linearized rotation dynamics as introduced in Sec. III-C.3. The three cases are as follows: (i) **Case 1:** we study the computation speed of three planners versus the size of the planning horizon N; (ii) **Case 2:** we compare the feasibility of trajectories generated by the three planners. This involves evaluating the tracking performance within physical limitations of the over-actuated UAV on the developed simulator; (iii) **Case 3:** we test the real-time performance by dynamically changing the target.

B. Simulation Results

Computation speed: In this case, we run three planners on a desktop (AMD Ryzen9 5950X CPU, 64 GB RAM) in MATLAB 2022b and record the computing time while the size of the planning horizon N spans from 1 to 96 with the increment as 5. For the NLP planner, the SNOPT solver [22] is used in favor of sparsity. With the linearized techniques described in Sec. III-C.3, both the ND and TQ planners are solved as QPs with the OSQP solver [41]. The computing time of the NLP planner ranges from tens of seconds to hundreds of seconds, which is significantly higher than that of the TQ and NP planners which are around milliseconds to tens of milliseconds. In Fig. 4, the TQ planner is slightly slower than the ND planner since two QPs must be solved each time. However, ignorance of the dynamics would increase the infeasible portion of the reference trajectory, which can be seen in the following study.

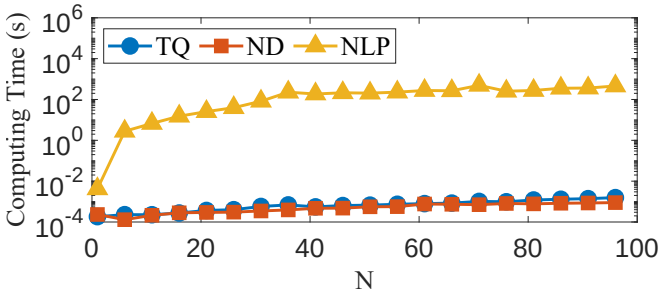

Fig. 4: **Case 1: Computation speed study of three planners with increasing planning horizon.** The proposed TQ planner performs significantly faster than the NLP planner, while its computation speed is comparable with the simplest ND planner.

Feasibility of the reference trajectory: For comparison, the over-actuated UAV starts at the origin from rest and the aggressive terminal states are set as $\boldsymbol{X}_{[e]} = [1,2,3]^{\mathsf{T}}$, $\dot{\boldsymbol{X}}_{[e]} = [2,3,0]^{\mathsf{T}}$, $\boldsymbol{\Theta}_{[e]} = [3\pi/4, \pi/2, -\pi/4]^{\mathsf{T}}$, and $\ddot{\boldsymbol{X}}_{[e]}, \boldsymbol{\Omega}_{[e]}, \dot{\boldsymbol{\Omega}}_{[e]} = \mathbf{0}$ for all planners. The planning horizon is

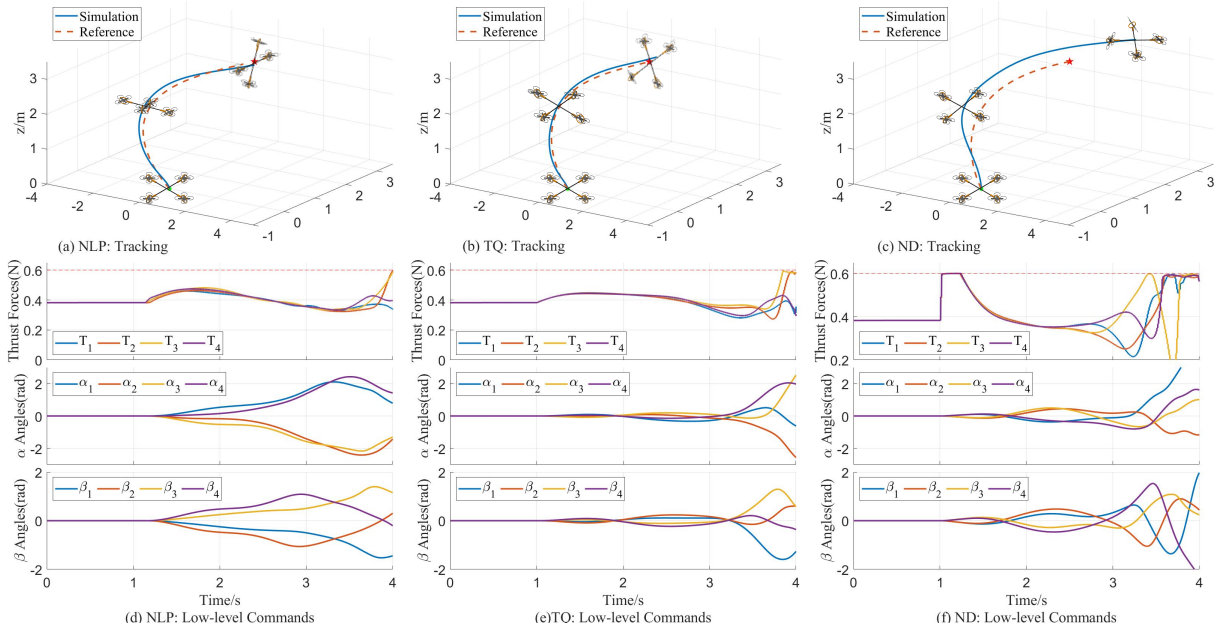

Fig. 5: **Case 2: Feasibility study of the reference trajectory**. The motion starts around 1s while the green dot and red star indicate the start and target reference points. The low-level commands are for the 4 thrust generators as indicated in Fig. 2.

TABLE I: **Accumulated RMS Errors in Case 2**

Algorithm	x(m)	y(m)	z(m)	roll(rad)	pitch(rad)	yaw(rad)
NLP	0.0041	0.0022	0.0007	0.1066	0.0020	0.0591
TQ	0.0107	0.0030	0.0002	0.1494	0.0004	0.0882
ND	0.1907	0.5905	0.4419	0.1238	0.0201	0.0920

fixed as $3s$ with $dt = 0.01s$. From Fig. 5, our hierarchical controller in Sec. II-C can properly track the reference trajectories of the NLP and TQ planner. The ND planner can only generate a spatial trajectory with relaxed upper bounds in Eqs. (11b) and (11c), otherwise it will be infeasible given the aggressive terminal states. However, the tracking performance degenerates due to the satuartion of the thrust forces which comes from the lack of dynamics constraints on the motor inputs. Note that, the saturation also happens near the end of the trajectory for our proposed TQ planner since the approximated input constraints Eqs. (14c) and (15c) are used for convexification.

For visualization, only 3 poses of the over-actuated UAV are shown in Fig. 5 while the RMS errors are reported in Tab. I. Although the approximated linearization of the attitude representation affects the attitude trajectory, the optimal solution of our proposed TQ planner is still dynamically consistent with the platform. To report the weights used, $[\gamma_1, \gamma_2, \gamma_3] = [1e^{-1}, 1e^{-1}, 1e^{-3}]$ is used to encourage smoothness of the trajectory for the NLP planner, $\lambda_1 = 1e^{-1}, \lambda_2 = 1e^{-1}$ for the ND planner while for the TQ planner, $\boldsymbol{W}_{1,2} = \mathbf{I}$, $\boldsymbol{Q}_{1,2} = 1e^{-4}\mathbf{I}$, and $\boldsymbol{R}_{1,2} = 1e^{-2}\mathbf{I}$.

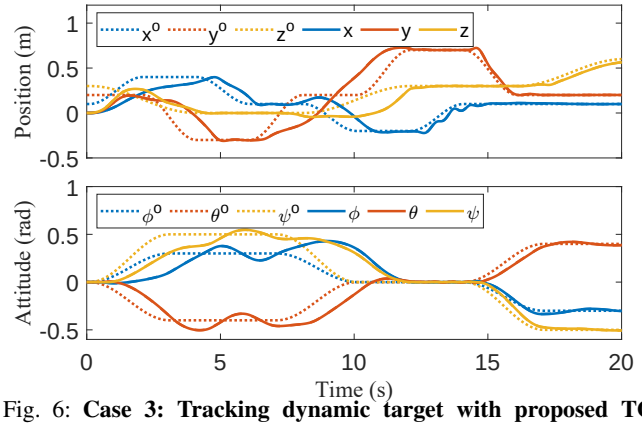

Fig. 6: **Case 3: Tracking dynamic target with proposed TQ planner.** The superscript o indicates the trajectory of the target in each DoF.

Dynamic target tracking: To further evaluate the real-time performance, the proposed TQ planner is implemented in an MPC manner to track a dynamic target. Similar to the scheme in [34], the trajectory planning problem is solved in 100 Hz with the feedback signal as initial states, which determines the control sequences over a receding prediction horizon into the future. Due to our hierarchical control framework design as shown in Fig. 3, the next optimal 6-DoF states are fed into the controller instead of the wrench command $\boldsymbol{u}$. As shown in Fig. 1, the target is dynamically changing its position and attitude. With the prediction horizon as $5s(N = 500)$, the over-actuated UAV can quickly converge to the target pose and accurately follow a dynamic trajectory as seen in Fig. 6.

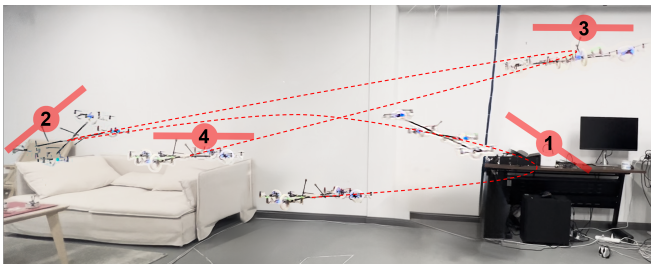

Fig. 7: **Hardware experiment**. The red marks with numbers represent the commanded target poses in order, and the dashed line approximates the real trajectory of the over-actuated UAV for visualization. More details can be seen in the accompanied video.

V. Experiment

A. Experiment Setup

In the experiment, we use the VICON motion capture system to measure the position and attitude of the central frame. The high-level controller and motion planner run on a remote PC, which communicates with the motion capture system through Ethernet. The high-level controller calculates the desired thrust, tilting, and twisting angles for all quadcopter modules. The communication between the remote PC and each quadcopter is achieved by Crazy Radio PA antennas. Each quadcopter is embedded with an onboard IMU module, estimating the rotation angle given the attitude of the central frame. Meanwhile, the onboard controller regulates the tilting and twisting angles to desired values and provides the required thrust.

B. Experiment Results

To mimic the dynamic target in the simulation experiment, four target poses are defined that include positions and altitudes, as shown in Fig. 7. Note that the target pose is updated every 2 secs in sequence online. Instead of following a complete reference trajectory passing through these 4 targets, the proposed TQ planner has to solve for the optimal trajectory online once the command is received. From Fig. 7, the over-actuated UAV can track the dynamic target while maintaining its stability. To track the latest received target, the over-actuated UAV has to update the trajectory, resulting in an inability to fully reach the previous target.

VI. Conclusion

In this paper, we propose a fast and efficient motion planning strategy for the over-actuated UAV. Being consistent with the hierarchical control architecture, the dimension of the optimization problem can be reduced by replacing input variables. We further explore the dynamics of the UAV platform and decouple it into two smaller sub-problems. With linearization of the rotational dynamics, the proposed planner can be solved via two-step QPs. As a result, its computation cost is significantly reduced compared to solving in a single NLP. And our simulation and hardware experiments verify that, unlike planners that completely ignore dynamics, our planner can guarantee dynamical consistency. Due to the fact that it can be solved in a real-time manner, we also demonstrate that it can be used to track a dynamic target. In the future, we aim to include obstacle avoidance for the over-actuated UAV applications in dynamic environments.

REFERENCES

[1] T. Anzai, M. Zhao, X. Chen, F. Shi, K. Kawasaki, K. Okada, and M. Inaba, "Multilinked multirotor with internal communication system for multiple objects transportation based on form optimization method," in *IEEE/RAS International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[2] M. Ryll, H. H. Bülthoff, and P. R. Giordano, "A novel overactuated quadrotor unmanned aerial vehicle: Modeling, control, and experimental validation," *IEEE Transactions on Control Systems Technology (TCST)*, vol. 23, no. 2, pp. 540–556, 2014.

[3] M. Kamel, S. Verling, O. Elkhatib, C. Sprecher, P. Wulkop, Z. Taylor, R. Siegwart, and I. Gilitschenski, "The voliro omniorientational hexacopter: An agile and maneuverable tiltable-rotor aerial vehicle," *IEEE Robotics and Automation Magazine (RA-M)*, vol. 25, no. 4, pp. 34–44, 2018.

[4] P. Yu, Y. Su, M. J. Gerber, L. Ruan, and T.-C. Tsao, "An over-actuated multi-rotor aerial vehicle with unconstrained attitude angles and high thrust efficiencies," *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 4, pp. 6828–6835, 2021.

[5] C. Pi, L. Ruan, P. Yu, Y. Su, S. Cheng, and T. Tsao, "A simple six degree-of-freedom aerial vehicle built on quadcopters," in *IEEE Conference on Control Technology Applications (CCTA)*, 2021.

[6] Y. Su, L. Ruan, P. Yu, C.-H. Pi, M. J. Gerber, and T.-C. Tsao, "A fast and efficient attitude control algorithm of a tilt-rotor aerial platform using inputs redundancies," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 2, pp. 1214–1221, 2021.

[7] Y. Su, C. Chu, M. Wang, J. Li, L. Yang, Y. Zhu, and H. Liu, "Downwash-aware control allocation for over-actuated uav platforms," in *IEEE/RAS International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[8] H.-N. Nguyen, S. Park, J. Park, and D. Lee, "A novel robotic platform for aerial manipulation using quadrotors as rotating thrust generators," *IEEE Transactions on Robotics (T-RO)*, vol. 34, no. 2, pp. 353–369, 2018.

[9] Y. Su, P. Yu, M. Gerber, L. Ruan, and T.-C. Tsao, "Nullspace-based control allocation of overactuated uav platforms," *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 4, pp. 8094–8101, 2021.

[10] Y. Su, P. Yu, M. Gerber, L. Ruan, and T. Tsao, "Fault-tolerant control of an over-actuated uav platform built on quadcopters and passive hinges," *IEEE/ASME Transactions on Mechatronics (TMECH)*, vol. 29, no. 1, pp. 602–613, 2024.

[11] P. Yu, Y. Su, L. Ruan, and T.-C. Tsao, "Compensating aerodynamics of over-actuated multi-rotor aerial platform with data-driven iterative learning control," *IEEE Robotics and Automation Letters (RA-L)*, vol. 8, no. 10, pp. 6187–6194, 2023.

[12] Y. Su, J. Li, Z. Jiao, M. Wang, C. Chu, H. Li, Y. Zhu, and H. Liu, "Sequential manipulation planning for over-actuated unmanned aerial manipulators," in *IEEE/RAS International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[13] M. Zhao, K. Kawasaki, X. Chen, S. Noda, K. Okada, and M. Inaba, "Whole-body aerial manipulation by transformable multirotor with two-dimensional multilinks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[14] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, present, and future of aerial robotic manipulators," *IEEE Transactions on Robotics (T-RO)*, 2021.

[15] M. Ryll and R. K. Katzschmann, "Smors: A soft multirotor uav for multimodal locomotion and robust interaction," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[16] Y. Su, Z. Jiao, Z. Zhang, J. Zhang, H. Li, M. Wang, and H. Liu, "Flight structure optimization of modular reconfigurable uavs," in *International Conference on Intelligent Robots and Systems (IROS) (submitted)*, 2024.

[17] J. Xu, D. S. D'Antonio, and D. Saldaña, "Modular multi-rotors: From quadrotors to fully-actuated aerial vehicles," *arXiv preprint arXiv:2202.00788*, 2022.

[18] K. Liu, L. Ma, H. Zhou, S. Li, K. Zhang, D. Huang, B. Li, and D. Zhao, "Optimal time trajectory generation and tracking control for over-actuated multirotors with large-angle maneuvering capability," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 3, pp. 8339–8346, 2022.

[19] D. Brescianini and R. D'Andrea, "Computationally efficient trajectory generation for fully actuated multirotor vehicles," *IEEE Transactions on Robotics (T-RO)*, vol. 34, no. 3, pp. 555–571, 2018.

[20] P. Liu, F. Quan, Y. Liu, and H. Chen, "Collision-free 6-dof trajectory generation for omnidirectional multi-rotor aerial vehicle," *arXiv preprint arXiv:2209.06764*, 2022.

[21] F. Morbidi, D. Bicego, M. Ryll, and A. Franchi, "Energy-efficient trajectory generation for a hexarotor with dual-tilting propellers," in *IEEE/RAS International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[22] P. E. Gill, W. Murray, and M. A. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM review*, vol. 47, no. 1, pp. 99–131, 2005.

[23] J. Zhang, J. Shen, Y. Liu, and D. W. Hong, "Design of a jumping control framework with heuristic landing for bipedal robots," in *IEEE/RAS International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[24] J. Luo, Y. Su, L. Ruan, Y. Zhao, D. Kim, L. Sentis, and C. Fu, "Robust bipedal locomotion based on a hierarchical control structure," *Robotica*, vol. 37, no. 10, pp. 1750–1767, 2019.

[25] Z. He, J. Wu, S. Zhang, J. Zhang, L. Sun, Y. Su, and X. Leng, "Cdm-mpc: An integrated dynamic planning and control framework for bipedal robots jumping," *IEEE Robotics and Automation Letters (RA-L) (submitted)*, 2024.

[26] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," in *IEEE/RAS International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[27] Z. Zhou, B. Wingo, N. Boyd, S. Hutchinson, and Y. Zhao, "Momentum-aware trajectory optimization and control for agile quadrupedal locomotion," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 3, pp. 7755–7762, 2022.

[28] M. Wang, Y. Su, H. Liu, and Y. Xu, "Walkingbot: Modular interactive legged robot with automated structure sensing and motion planning," in *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2020.

[29] X. Lin, H. Krishnan, Y. Su, and D. W. Hong, "Multi-limbed robot vertical two wall climbing based on static indeterminacy modeling and feasibility region analysis," in *IEEE/RAS International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[30] N. Rikovitch and I. Sharf, "Kinodynamic motion planning for uavs: A minimum energy approach," in *AIAA Guidance, Navigation, and Control (GNC) Conference*, 2013.

[31] L. Chen, I. Mantegh, T. He, and W. Xie, "Fuzzy kinodynamic rrt: a dynamic path planning and obstacle avoidance method," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2020.

[32] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," in *IEEE/RAS International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[33] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.

[34] M. Jacquet, G. Corsini, D. Bicego, and A. Franchi, "Perception-constrained and motor-level nonlinear mpc for both underactuated and tilted-propeller uavs," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[35] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, and A. Del Prete, "Optimization-based control for dynamic legged robots," *IEEE Transactions on Robotics (T-RO)*, vol. 40, pp. 43–63, 2024.

[36] M. J. Gerber and T.-C. Tsao, "Twisting and tilting rotors for high-efficiency, thrust-vectored quadrotors," *Journal of Mechanisms and Robotics*, vol. 10, no. 6, p. 061013, 2018.

[37] L. Ruan, C.-H. Pi, Y. Su, P. Yu, S. Cheng, and T.-C. Tsao, "Control and experiments of a novel tiltable-rotor aerial platform comprising quadcopters and passive hinges," *Mechatronics*, vol. 89, p. 102927, 2023.

[38] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se (3)," in *IEEE conference on decision and control (CDC)*, 2010.

[39] G. Wu and K. Sreenath, "Variation-based linearization of nonlinear systems evolving on $so(3)$ and $\mathbb{S}^2$," *IEEE Access*, vol. 3, pp. 1592–1604, 2015.

[40] Y. Ding, A. Pandala, C. Li, Y.-H. Shin, and H.-W. Park, "Representation-free model predictive control for dynamic motions in quadrupeds," *IEEE Transactions on Robotics (T-RO)*, vol. 37, no. 4, pp. 1154–1171, 2021.

[41] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: an operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.