2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Downwash-aware Configuration Optimization for Modular Aerial Systems

Mengguang Li and Heinz Koeppl

***Abstract*— This work proposes a framework that generates and optimally selects task-specific assembly configurations for a large group of homogeneous modular aerial systems, explicitly enforcing bounds on inter-module downwash. Prior work largely focuses on planar layouts and often ignores aerodynamic interference. In contrast, firstly we enumerate non-isomorphic connection topologies at scale; secondly, we solve a nonlinear program to check feasibility and select the configuration that minimizes control input subject to actuation limits and downwash constraints. We evaluate the framework in physics-based simulation and demonstrate it in real-world experiments.**

I. INTRODUCTION

Recent advances in modular reconfigurable robotic systems show promising results, due to their high design flexibility compared with conventional systems [1]. By reassembling and reusing a group of repeated modules, these systems enable robust, resilient and cost-efficient solutions, thereby offering versatility and scalability for task-specific adaptation.

In aerial robotics, many modular systems have emerged over the last decade. In this work, we focus on rotor-based actuation. Early work on modular multirotors demonstrated assemblies built from single-rotor modules [2] and mid-air self-assembly of standard quadrotors [3]. These elegant designs have limitations though, as the assemblies are largely planar, rotor orientations remain collinear, and therefore the overall system is underactuated, which restricts the application scope. In general, a common strategy is to tilt rotors to break collinearity. Rotors can be either fixed-tilt [4] or actively tilting during flight [5].

When tilting the rotors, the aerodynamic effect of downwash [6], which is the deflection of air under the propellers, needs to be considered, as neighboring wakes may interfere. Airflow interference between rotors can reduce efficiency and even induce system instability, effects that are pronounced especially in modular designs where rotors are often in close proximity. Downwash has, for example, already been considered in the design of aerial platforms [7]. Researchers have also proposed control allocation strategies to reduce the downwash effect in overactuated aerial systems [8]. A recent study shows that the quadrotor-induced combined airflow can be well-approximated as a turbulent jet [9].

*This work has been funded by the LOEWE initiative (Hesse, Germany) within the emergenCITY center [LOEWE/1/12/519/03/05.001(0016)/72].

The authors are with the Department of Electrical Engineering and Information Technology, Technische Universität Darmstadt, 64287 Darmstadt, Germany. {mengguang.li, heinz.koeppl}@tu-darmstadt.de

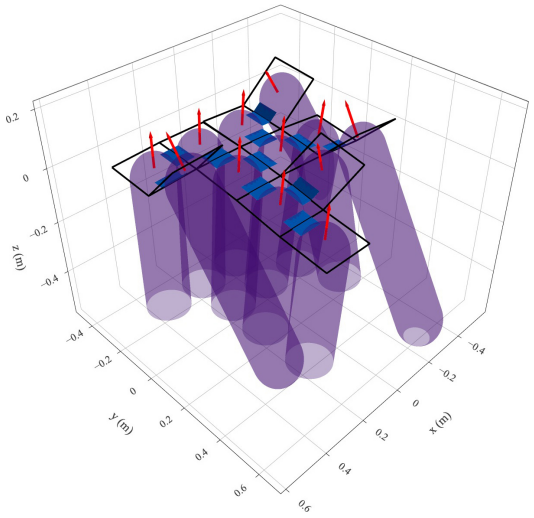


Fig. 1. The optimal assembly of 12 modules, given the target wrench set defined in Section IV. Each black square represents one module, each blue rectangle represents a rigid connection, each red arrow represents the module frame z-axis, and the purple capsules indicate the collision-free downwash volume for each module.

However, in the design of modular aerial systems, the downwash effect is rarely explored or sometimes ignored completely. Interference can occur within a single module or between modules. For instance, in [10], each module is a multirotor with four fixed-tilt rotors, where inter-module downwash interference is ignored, potentially affecting system performance. In [11], a standard quadrotor is attached via a passive hinge joint to a cage to form a module, thereby extending the overall system actuation capability. This additional one degree of freedom (DOF) may cause downwash interference within the configuration but was not considered. Likewise, in [12], each quadrotor-based module is extended with a 2-DOF passive gimbal, further enlarging the actuation space, again without considering downwash effects.

Therefore, this paper proposes a design and optimization framework that addresses these gaps. We first use the standard quadrotor as the building block of our base module so that within each module, the rotor downwash interaction can be neglected. Inspired by [13], we allow non-planar assemblies to explore fully 3D arrangements while respecting each module's airflow envelope. Given a task-specific wrench requirement, our method automatically determines the number of modules needed, the inter-module connection topology, and their 3D spatial arrangement to satisfy the requirement

with minimum total effort. The pipeline explicitly accounts for rotor downwash interference by enforcing collision-free spatial volume, yielding physically realizable layouts. An example is shown in Fig. 1.

This paper is organized as follows. Section II defines the applicable module within this framework and states the problem. Section III presents the enumeration algorithm for non-isomorphic configurations. Section IV details configuration optimization and Section V describes the control. Physics-based simulation and real-world experiments are reported in Section VI, and Section VII concludes this work.

II. MODULES AND PROBLEM STATEMENT

Specifically, we use an X-layout quadrotor with collinear, unidirectional propellers as the module's building block. This platform is widely used both commercially and in research, including agile vision-based flight [14], swarm exploration [15], and even aerial physical interaction despite its underactuated nature [16]. Consequently, commercially available components can be readily adapted to our method.

A. Module and assembly

Each module has the same arm length and is equipped with four identical connection ports as shown in Fig. 2. Two modules can rigidly connect to each other using one of their connectors by aligning the contact areas, as indicated by the arrow on the connector. For each connector on the module v, the angle $\theta_{c,v}$ of connector $c_v \in \{1,2,3,4\}$ can be adjusted within the prescribed range $[\pi/4, 3\pi/4]$. One realization of the connector is shown in Fig. 2, where the four white circles represent permanent magnets, as in standard modular systems [3]. Other position-adjustable hinge joints can also be used, as in [13] with a 3D-printed connector for each fixed angle. Note that when all angles $\theta_{c,v}$ are fixed at $\pi/2$, the system resembles that in [3]. The advantage of this type of connection is the significant enlargement of configuration space compared with purely planar docking, by modifying the connector angle. By adjusting the connection angle, the system can break the collinearity of the rotor orientation, resembling the standard fixed-tilt rotor design and, at the same time, the downwash effect within a single module is not affected; only inter-module effects need to be considered. With this, the structure is able to reconfigure into non-planar layouts and adapt to various tasks. Once the task-specific configuration is given, the connector angles are fixed for that structure, and the whole system can be considered as a rigid body, which does not suffer from the common drawbacks of actuated tiltrotor design, such as additional actuation complexity, servo-motor induced rotational limitations, and singularities [5]. We consider a modular aerial system composed of n identical base modules. We assume that assemblies are acyclic, which have no kinematic loops, and that there is at most one physical connection between any pair of modules. Excluding cycles avoids coupled connector angles that overconstrain the structure.

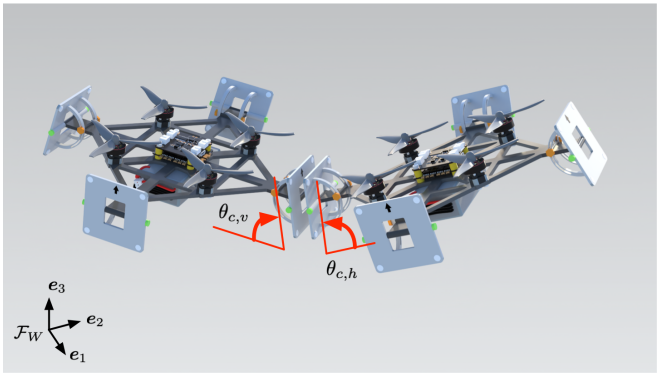


Fig. 2. An illustration of two modules with four connection ports on each frame. Each connector can rotate along the green axis, where the orange screw can fix the connector. The white circles on the connector are permanent magnets.

B. Problem statement

We denote a configuration as a tuple $\mathcal{C} := (n, \mathcal{G}, \Lambda)$, where $\mathcal{G} = (V, E)$ is a finite, edge-labeled undirected tree with $|V| = n$, each vertex $v \in V$ is a module instance, and each edge $e = \{v, h\} \in E$ represents exactly one physical connection. The edge label is denoted as $\Lambda(e) = (c_v, c_h, \theta_e)$, where c_v, c_h are the connectors of the two modules connected by edge e, and $\theta_e = (\theta_{c,v} + \theta_{c,h}) \in [\pi/2, 3\pi/2]$ is the relative angle between the two module frames. We set the angle $\theta_{c,v} = \theta_{c,h}$ for each connection.

We model the task requirements as a finite set of K target wrenches

$$\mathcal{W} = \{\boldsymbol{b}_k \in \mathbb{R}^6 | k \in \{1, \dots, K\}\},$$

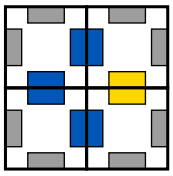
each entry $\boldsymbol{b}_k$ is a wrench vector, composed of a force vector $\boldsymbol{b}_{k,f} \in \mathbb{R}^3$ and a torque vector $\boldsymbol{b}_{k,\tau} \in \mathbb{R}^3$, that the system must generate within its actuation limits. Our objective is to find, for a given set of target wrenches $\mathcal{W}$, a configuration $\mathcal{C}$ that satisfies the wrench set while minimizing the overall control input.

The problem stated above is not trivial to solve. The number of tree topologies grows exponentially in n, and connector port choices further increase the search space by an additional exponential factor. Furthermore, each edge on each topology carries a continuous angle θ_e, yielding a mixed discrete–continuous design space that further complicates direct solution.

To manage this complexity, we use a two-stage pipeline. Firstly, we set all connector angles temporarily fixed at $\pi/2$, enumerate all non-isomorphic configurations for n modules. Secondly, given a task wrench set $\mathcal{W}$, we evaluate each configuration by searching the connector angle space to check feasibility and compute cost of all the control input. We then select the configuration with the smallest cost.

III. ENUMERATION OF NON-ISOMORPHIC CONFIGURATIONS

In this section, we solve the first part stated above. Setting all the connection angles to $\pi/2$ reduces the task



(a)

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 6 & 2 & 6 & 1 \\ 0 & 2 & 0 & 3 & 0 \\ 1 & 6 & 2 & 6 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

(b)

Fig. 3. An example of a four-module assembly and its matrix representation, where a black square represents a single module, the four rectangles within each square are the connectors. Blue indicates a valid connection, yellow indicates a blocked connection, and gray indicates an available connector.

to the classical problem of square-lattice modular robots enumeration, a problem that has been widely studied. We extend the matrix-based approach in [17] to handle identically shaped assemblies that may differ in their topological interconnections, as in our case. To address scalability, we also introduce a sampling-based heuristic that trades completeness for tractability when the number of modules becomes large.

A. Exhaustive enumeration for small n

We represent each module as a 3×3 matrix as in [17], where the module center is coded as `6`, and each available connector on the cardinal edges is coded as `1`. Two modules may be joined when opposing edge entries (both `1`) are aligned; a valid connection is recorded by summing the interface entries and coded as `2`. As only one connection between two modules is allowed, introducing one new module may block other available connectors. Any edge rendered unusable by the connection is recoded as `3`. An illustration of the matrix representation is given in Fig. 3 where four modules are connected together. For enumeration, we start with a single module and add a new module to all available connectors. For each newly generated configuration, we identify whether it is already known or isomorphic to known configurations. For our system, we consider configurations under the symmetric point group C_4 to be isomorphic, i.e., the four rotations around the module's local frame z-axis, $\{0°, 90°, 180°, 270°\}$. As each module produces directional thrust along its positive body frame z-axis, flip symmetry does not apply here. With this, all non-isomorphic configurations can be found by visiting all available connectors. Unlike [17], with blocked connections the matrix representation is able to distinguish structures with the same shape but different connection topologies, as changes in the connection angles can result in different configurations for the same assembly shape. An illustration of all non-isomorphic configurations with 5 modules is shown in Fig. 4. A table of all non-isomorphic configurations up to 11 modules is given in Table I. Experiments are conducted on Gentoo Linux (kernel `6.12.21-gentoo`) with an AMD EPYC 7702P processor. Runtime is measured single-threaded using one logical CPU, 2.00 GHz. The algorithm can be generalized to other module families by adapting the rule of blocked connections and the symmetry group used for isomorphism check. By defining a configuration signature to represent a set of isomorphic configurations [18], the runtime can be improved, but the exponential scaling remains the same and the exhaustive procedure for large n becomes computationally impractical.

TABLE I

EXHAUSTIVE ENUMERATION OF NON-ISOMORPHIC CONFIGURATIONS FOR CONNECTOR ANGLES ALL FIXED TO $\pi/2$.

n	Number of available configurations	Number of non-isomorphic configurations	CPU Time (s)
1	1	1	0.0000
2	4	1	0.0006
3	6	2	0.0007
4	16	7	0.0017
5	68	24	0.0072
6	272	97	0.0261
7	1242	401	0.1147
8	5740	1772	0.5488
9	27 960	7930	2.7293
10	136 628	36 335	14.1008
11	678 204	168 249	78.5251

B. Heuristic sampling for large n

For large n, we explore the physical characteristics of the system and propose a sampling-based enumeration that trades off completeness for computational feasibility.

In general, for the same number of modules n, all the configurations have the same total weight, but the radius of gyration can vary depending on how widely the modules are distributed. A smaller radius of gyration indicates a lower moment of inertia, which leads to a lower control input for rotational dynamics, resulting in a compact structure with higher maneuverability. For our aerial system, we therefore encourage compact configurations, where the mass is distributed not too far from the center of mass (COM). As has been shown, a chain-like configuration reaches actuation saturation earlier [19]. The matrix representation already encodes the spatial information of the structure. We consider all entries equal to `6` in the matrix, which represent the center of each module. It is then straightforward to compute the COM for a given configuration as $\bar{\boldsymbol{g}}_m \in \mathbb{R}^2_{\geq 0}$, which is the average coordinates of all the `6` in the matrix. To scale up, we sample configurations with lower radius of gyration with higher probability. For n modules, we sample $N_n \in \mathbb{N}_{>0}$ from the M_n available non-isomorphic configurations $\mathcal{S}_n = \{\mathcal{C}_1(n), \ldots, \mathcal{C}_{M_n}(n)\}$. If $M_n \leq N_n$, we keep all configurations without sampling. For $M_n > N_n$, we calculate the radius of gyration with

$$g_m = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \left\| \boldsymbol{g}_{m,i} - \bar{\boldsymbol{g}}_m \right\|^2}, \qquad \bar{\boldsymbol{g}}_m = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{g}_{m,i},$$

where $\{\boldsymbol{g}_{m,i}\}_{i=1}^{n}$ are the module-center coordinates of the m-th configuration $\mathcal{C}_m(n)$, $m \in \{1, \ldots, M_n\}$. Let $g_{m,0} := \min_{1 \leq m \leq M_n} g_m$, we set for each m-th configuration a Gaus-

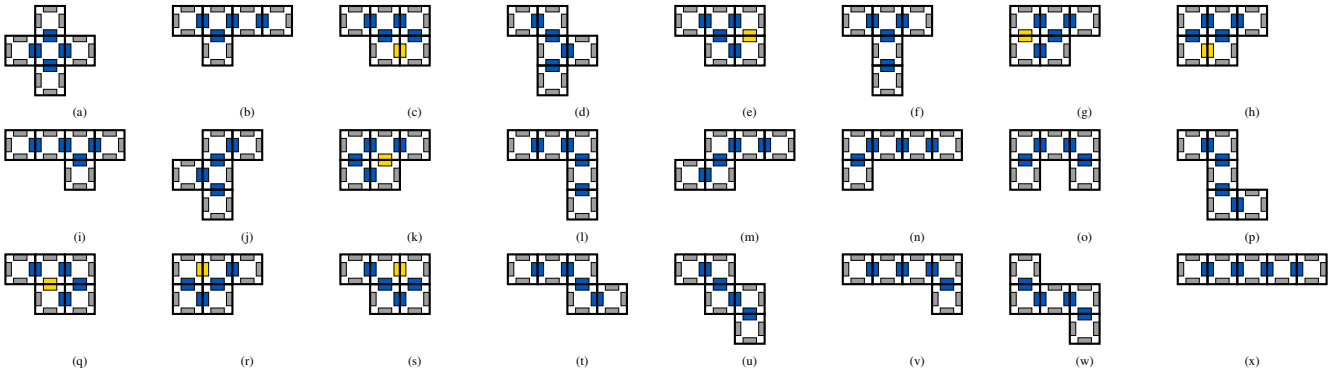


Fig. 4. All non-isomorphic configurations under the symmetric point group C_4 using 5 modules.

Algorithm 1 Sampling-based enumeration of non-isomorphic configurations

Init: n, $\{N_i\}_{i=2}^{n}$, $\{\sigma_i\}_{i=2}^{n}$, $\mathcal{S}_1 \leftarrow \{\mathcal{C}_1(1)\}$, $M_1 \leftarrow 1$
for $i = 2$ **to** n **do**
  for $m = 1$ **to** M_{i-1} **do**
    for connector c coded as 1 in $\mathcal{C}_m(i-1)$ **do**
      $\mathcal{C}'(i) \leftarrow$ attach one new module at c
      if $\mathcal{C}'(i)$ (up to symmetry group C_4) $\notin \mathcal{S}_i$ **then**
        $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{\mathcal{C}'(i)\}$
  if $M_i \leq N_i$ **then**
    return $\mathcal{S}_i$
  else
    compute $\bar{\boldsymbol{g}}_m, g_m, g_{m,0}, w_m, P_m$
    $\mathcal{S}_i' \leftarrow \textsc{Sample}(\mathcal{S}_i, N_i; P_m)$ *(without replacement)*
    return $\mathcal{S}_i \leftarrow \mathcal{S}_i'$
return $\mathcal{S}_2, \ldots, \mathcal{S}_n$

sian weight w_m and its sampling probability P_m as

$$w_m := \exp\Big(-\frac{(g_m - g_{m,0})^2}{2\sigma_n^2}\Big), \quad P_m := \frac{w_m}{\sum_{m'=1}^{M_n} w_{m'}}.$$

We can now draw N_n configurations without replacement from the set $\mathcal{S}_n$ using the probability P_m. After sampling, we proceed with the same steps as above in the exhaustive search. The algorithm is summarized in Algorithm 1.

With this, configurations with a large number of modules can be explored within a manageable runtime. The computation time per number of modules is given in Fig. 5, and the hardware is the same as in Section III-A with one CPU-core. As can be seen, the method is scalable, as the run time grows approximately linearly with a growing number of modules. The trade-off is completeness, for larger n, the fraction of sampled assemblies relative to the rapidly expanding set of all non-isomorphic configurations drops approximately exponentially. By changing the variance σ_n of the Gaussian weights, the aggressiveness of sampling can be adjusted. To retain useful coverage, we bias configurations by the radius of gyration of the COM, where smaller g_m receive higher probability based on the Gaussian kernel. This prioritizes compact assemblies while still exploring diverse topologies.

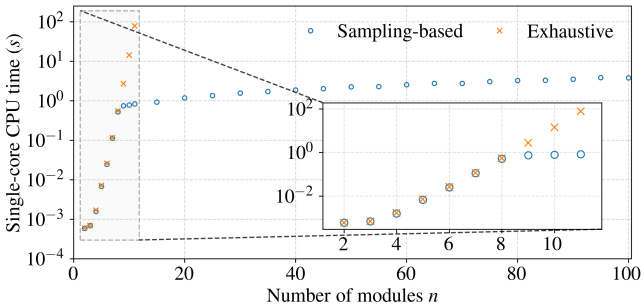


Fig. 5. Comparison of exhaustive and sampling-based enumeration. The sample count is fixed at $N_n = 500$ for all n. For $n > 11$, CPU time is shown only for every fifth value of n.

For other robotic systems, different sampling schemes can be tailored to robot-specific properties. For example, uniform sampling explores growth without added bias, or reversed sampling as ours by defining $g_{m,0} := \max_{1 \leq i \leq M_n} g_m$, which encourages chain-like or more spatially spread out configurations.

IV. Configuration Optimization

From the previous section, we obtain, for each n, a set of non-isomorphic candidate configurations, where all connector angles are set to $\pi/2$. We now propose an optimization procedure that, given a set of task wrenches, selects connector angles and input allocations to meet the wrench requirements while accounting for downwash interactions. Concretely, we impose geometric constraints that prevent harmful overlap of the rotor downwash regions between modules, and we solve for feasible, low control input configurations that satisfy the tasks. Afterward, we compare across the configurations to select the best variant among them.

A. Configuration kinematics

We first derive the kinematics for any non-isomorphic configuration of n modules, using the matrix representation introduced in Section III. Note that relaxing the connector angle from the fixed $\pi/2$ to $\theta_e \in [\pi/2, 3\pi/2]$ does not change the underlying topology $\mathcal{G}$ of each configuration

$\mathcal{C}$. We exploit the fact that $\mathcal{G}$ is a tree, where the path between any two vertices is unique, and every non-root vertex has exactly one parent [20]. Consequently, a traversal algorithm such as Breadth-First Search (BFS) visits each vertex exactly once, allowing us to assign the connector angle on each edge sequentially and express the full 3D configuration relative to a chosen root module. Since a full traversal touches each vertex and edge a constant number of times, the choice of root affects only the visitation order and not the computational complexity. Therefore, for simplicity we choose the first upper-left module in the representation matrix as the root module.

We now formalize this. As shown in Fig. 2, the inertial frame $\mathcal{F}_W$ is denoted as the space spanned by three unit basis vectors $\boldsymbol{e}_1 = [1,0,0]^\top$, $\boldsymbol{e}_2 = [0,1,0]^\top$, $\boldsymbol{e}_3 = [0,0,1]^\top$. We place the root module temporarily at the origin $\boldsymbol{p}_1 = [0,0,0]^\top$. Its orientation relative to the world frame can be represented as $\boldsymbol{R}_1 = \boldsymbol{R}_x(\alpha_0)\boldsymbol{R}_y(\alpha_1)$, where $\alpha_0, \alpha_1 \in [-\pi/2, \pi/2]$, and $\boldsymbol{R}_x(\alpha_i)$ denotes a basic rotation by an angle α_i around the x-axis. By applying BFS, all modules' world frame coordinates and orientations can be calculated according to their parent vertices as

$$\boldsymbol{R}_i = \begin{cases} \boldsymbol{R}_{i-1}\boldsymbol{R}_{\boldsymbol{x}}(\alpha_i), & \text{if } c_v \text{ or } c_h \in \{1,3\}, \\ \boldsymbol{R}_{i-1}\boldsymbol{R}_{\boldsymbol{y}}(\alpha_i), & \text{otherwise,} \end{cases}$$
$$\boldsymbol{p}_i = \boldsymbol{p}_{i-1} + l_c\boldsymbol{R}_{i-1}\boldsymbol{t}_i + l_c\boldsymbol{R}_i\boldsymbol{t}_i,$$

where l_c is the distance from the COM to a connector, and $\boldsymbol{t}_i$ is a unit vector from the module COM to its connector in its body frame, which depends on c_v. With this, the set of $\{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n\}$ and $\{\boldsymbol{R}_1, \ldots, \boldsymbol{R}_n\}$ can be computed. Once every module is visited once, we compute the COM position $\boldsymbol{p}_0 = 1/n \sum_{i=1}^{n} \boldsymbol{p}_i$ and move the structure to the origin, so that each module i position is $\boldsymbol{o}_i = \boldsymbol{p}_i - \boldsymbol{p}_0$. Each module has four rotors, where the relative position of the four rotors is the same in each module body frame; their positions $\boldsymbol{o}_{iq}, q \in \{1,2,3,4\}$ can be computed using $\boldsymbol{p}_i, \boldsymbol{R}_i$ and module arm length l_{arm}. For a given angular velocity Ω_{iq} of each rotor, a thrust of $f_{iq} = c_f\Omega_{iq}^2$ and a torque of $\tau_{iq} = (-1)^q c_m \Omega_{iq}^2$ can be generated, where c_f and c_m are rotor constants. The overall achievable wrench of the system $\boldsymbol{W} \in \mathbb{R}^6$, comprising the thrust $\boldsymbol{F}$ and the torque $\boldsymbol{\tau}$, is

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{\tau} \end{bmatrix} = \begin{bmatrix} \sum_{iq} f_{iq}\boldsymbol{R}_i\boldsymbol{e}_3 \\ \sum_{iq} f_{iq}\boldsymbol{o}_{iq} \times \boldsymbol{R}_i\boldsymbol{e}_3 + \tau_{iq}\boldsymbol{R}_i\boldsymbol{e}_3 \end{bmatrix}.$$

For simplicity, we denote by $\boldsymbol{u}_c = \left[\Omega_{11}^2, \ldots, \Omega_{n4}^2\right]^\top$ the control input. Then the overall wrench $\boldsymbol{W}$ can be expressed as $\boldsymbol{W} = \boldsymbol{A}\boldsymbol{u}_c$, with $\boldsymbol{A} = [\boldsymbol{A}_{11}, \ldots, \boldsymbol{A}_{n4}]$, where

$$\boldsymbol{A}_{iq} = \begin{bmatrix} c_f\boldsymbol{R}_i\boldsymbol{e}_3 \\ c_f\boldsymbol{o}_{iq} \times \boldsymbol{R}_i\boldsymbol{e}_3 + (-1)^q c_m\boldsymbol{R}_i\boldsymbol{e}_3 \end{bmatrix}.$$

B. Optimization for a single configuration

We adapt the wrench requirement method proposed in [10], for a set of wrenches, if those wrenches are within the achievable polytope of the system, we consider that the wrench requirement is fulfilled. Mathematically, given a set of task wrenches $\{\boldsymbol{b}_k\}_{k=1}^K$, this can be formulated as

$$\boldsymbol{A}\,\boldsymbol{u}_k = \boldsymbol{b}_k,$$

with element-wise input bounds $0 \le \boldsymbol{u}_k \le u_{max}$, where $\boldsymbol{u}_k$ is the input vector used to realize task wrench $\boldsymbol{b}_k$. The total control effort for all tasks can be formulated as $J_{cost} = \sum_{k=1}^K \lambda_k \|\boldsymbol{u}_k\|_2^2$. The weights $\lambda_k \ge 0$ reflect task importance, for example, rare tasks may receive smaller weights. Without loss of generality, we set $\lambda_k = 1$ for all $k = 1, \ldots, K$.

To avoid downwash interference, we require a clearance between any two modules $i \ne j$. We propose to use a capsule, which is a cylinder with two hemispherical ends, as a convex hull to enclose the critical collision-free downwash airflow volume for each module: $\boldsymbol{d}_i = \boldsymbol{o}_i - \mu\boldsymbol{R}_i\boldsymbol{e}_3,\ \mu \in [a,b]$, where $\boldsymbol{d}_i$ is a point on the line segment along the i-th module's body frame z-axis. We use a capsule for its simplicity for collision checking, as the shortest Euclidean distance between two capsules is the minimum distance between their capsule axes, which are two line segments. Although the capsule does not completely capture the fully induced airflow under a quadrotor, by setting the fixed line-segment range $[a,b]$ and the radius r for the hemispherical ends, a wide range of noncollision requirements can be satisfied. For each pair of modules $i \ne j$, we impose

$$\|\boldsymbol{d}_i - \boldsymbol{d}_j\|_2^2 \ge 4r^2, \qquad 1 \le i < j \le n,$$

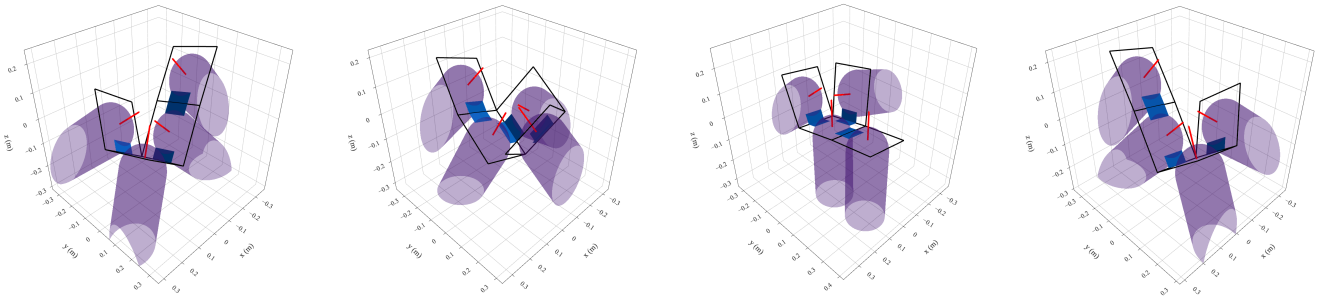
which yields $n(n-1)/2$ pairwise constraints for an n-module configuration. Let $\boldsymbol{\alpha} := [\alpha_0, \ldots, \alpha_n]^\top \in \mathbb{R}^{n+1}$, the vector of the root module orientation angles and all relative angles on each connection. Together with the input constraints, we can pose this as a nonlinear optimization problem:

$$\begin{aligned} \min_{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_k,\,\boldsymbol{\alpha}} \quad & \sum_{k=1}^{K} \lambda_k \|\boldsymbol{u}_k\|_2^2 \\ \text{s.t.} \quad & \|\boldsymbol{d}_i - \boldsymbol{d}_j\|_2^2 \ge 4r^2, \quad \forall 1 \le i < j \le n \\ & \boldsymbol{A}(\boldsymbol{\alpha})\boldsymbol{u}_k = \boldsymbol{b}_k, \quad \forall k \in K \\ & 0 \le \boldsymbol{u}_k \le u_{max}, \quad \forall k \in K \\ & -\pi/2 \le \boldsymbol{\alpha} \le \pi/2 \end{aligned} \tag{1}$$

where $\boldsymbol{u}_k, \boldsymbol{\alpha}$ are bounded element-wise. We note that, although the objective is quadratic, the equality-constraint matrix $\boldsymbol{A}$ depends on the decision vector $\boldsymbol{\alpha}$, which makes the problem a nonconvex nonlinear program. Global optimization can therefore be computationally expensive. We thus solve it using the gradient-based solver `Ipopt` within the `CasADi` toolbox [21]. `Ipopt` implements a large-scale interior-point method with a filter line-search and reliably finds locally optimal solutions even for large n [22].

C. Selection across configurations

Now we provide the final step for our pipeline. Since a target wrench set $\mathcal{W}$ is typically not directly related to the number of modules, we add the module's weight mg scaled by n to each wrench in the f_z direction to compensate for gravity, and we include one more wrench $[0,0,nmg,0,0,0]^\top$



(a) $J_{cost} = 56.62$ (b) $J_{cost} = 60.79$ (c) $J_{cost} = 54.14$ (d) $J_{cost} = 55.49$

Fig. 6. All optimal configurations with control costs J_{cost} using 4 modules for the wrench set specified in Section IV-C.

to emphasize the assembly hover capability. With the modified wrench target $\mathcal{W}'$, we start at one module and solve the nonlinear programming from Section IV-B. For each n, we solve all configurations in the given list from Section III, if there is no feasible solution, we increment to $n+1$. When a feasible solution is found, we compare all the costs within this number of modules and select the configuration with the smallest objective value and report the corresponding design. The algorithm is given in Algorithm 2.

Algorithm 2 Selection across configurations

Init: $n, \{\mathcal{S}_1, \ldots, \mathcal{S}_n\}, J_{best} = +\infty, \mathcal{C}_{best} = \{\emptyset\}$
for $i = 1$ **to** n **do**
  if $J_{best} < +\infty$ **then**
    return optimal solution found: $\mathcal{C}_{best}$
  for $m = 1$ **to** N_n **do**
    solve Eq. (1) for $\mathcal{C}_m(i)$
    if feasible solution exist **then**
      if $J_{cost} < J_{best}$ **then**
        $J_{best} \leftarrow J_{cost}$, $\mathcal{C}_{best} \leftarrow \mathcal{C}_m(i)$
return no feasible solution found within n.

As an illustration, we solve for a simple wrench set $\mathcal{W}$ of three target wrenches, where all $\boldsymbol{b}_{k,\tau} = \mathbf{0}$, with a force set $\{[0.5nmg, 0, nmg]^\top, [0, 0.5nmg, nmg]^\top, [0, 0, nmg]^\top\}$ for 4 modules. We use the parameters in Table II, the same as in the Section VI real world experiment. We choose the capsule variable $\mu \in [r/2, r/2 + 10l_{arm}]$, where the line segment is 10 times the arm length, for a large volume covering the downwash airflow. All configurations with feasible solutions are shown in Fig. 6. As can be seen, the airflow interference constraints are satisfied in all configurations. The cost, which is the sum of equally weighted control input magnitudes to generate all target wrenches, differs across non-isomorphic configurations, which demonstrates the necessity of our two-step approach. The best configuration for $n = 12$ is shown in Fig. 1, using the target force set $\{[0.1nmg, 0, nmg]^\top, [0, 0.1nmg, nmg]^\top, [0, 0, nmg]^\top\}$ with $\boldsymbol{b}_{k,\tau} = \mathbf{0}$.

To better illustrate how the overall achievable wrench of the configuration generated by Algorithm 2 behaves, we analyze the feasible force polytope [23], which is the set of reachable forces while maintaining zero torque. As can be seen, the target wrenches lie within the boundary of the polytope, which confirms the feasibility of the configuration.

TABLE II
SINGLE MODULE PARAMETERS USED IN THIS WORK.

m	c_f	c_m	l_{arm}	l_c	r
0.24 kg	3.87×10^{-7}	1.06×10^{-8}	0.06 m	0.11 m	0.095 m

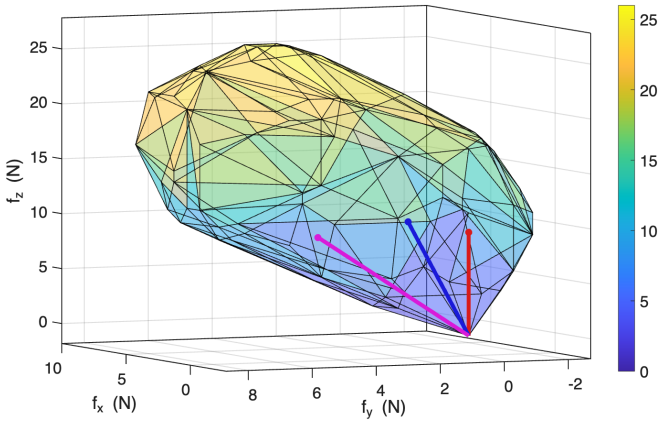


Fig. 7. Achievable force polytope for the best configuration in Fig. 6 (c); the blue, red, and purple vectors are projections onto the force subspace at zero torque and all lie within the polytope.

We also evaluate the computation time for solving Eq. (1). We randomly select 10 target wrenches with $f_x, f_y \in [-0.2nmg, 0.2nmg]$, $f_z \in [0.9nmg, 1.1nmg]$ and target torque components in $[-0.1l_cnmg, 0.1l_cnmg]$. We do not directly apply the wrench set modification in Section IV-C, as a target wrench set not related to n might be too large for small n and too trivial for large n. The total computation time is given in Fig. 8. As an example, for $n = 90$ with 10 target wrenches, the NLP comprises 3691 decision variables, 60 equality constraints, and 4005 inequality constraints. The median single-core solve time is 5460.71 s, demonstrating the scalability of the proposed pipeline. The hardware is the same as in Section III, and with modern hardware the runtime can be further reduced.

V. CONTROL

We now derive the control for an optimized configuration with n modules. For its COM position and orientation

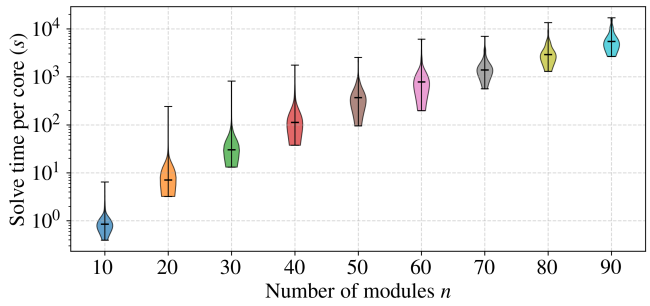


Fig. 8. Computation time, averaged per CPU core, with 10 randomly selected wrenches for $n \in \{10, 20, \ldots, 90\}$.

$(\boldsymbol{p}, \boldsymbol{R}) \in SE(3)$, given a desired trajectory $(\boldsymbol{p}_d, \boldsymbol{R}_d)$, we apply a geometric controller [24], where the desired force and torque can be expressed as

$$\boldsymbol{F}'_d = K_P(\boldsymbol{p}_d - \boldsymbol{p}) + K_D(\dot{\boldsymbol{p}}_d - \dot{\boldsymbol{p}}) + m\ddot{\boldsymbol{p}}_d + mg\boldsymbol{e}_3,$$
$$\boldsymbol{\tau}_d = -K_R\boldsymbol{e}_R - K_\omega\boldsymbol{e}_\omega + \boldsymbol{\omega} \times \boldsymbol{J}\boldsymbol{\omega},$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ is the angular velocity, K_P, K_D, K_R, K_ω are diagonal matrices with positive gains. The error of rotation matrix and the error of angular velocity can be obtained as $\boldsymbol{e}_R = \frac{1}{2}[\boldsymbol{R}_d^\top \boldsymbol{R} - \boldsymbol{R}^\top \boldsymbol{R}_d]^\vee$ with $\boldsymbol{e}_\omega = \boldsymbol{\omega} - \boldsymbol{R}^\top \boldsymbol{R}_d \boldsymbol{\omega}_d$, where $[\cdot]^\vee$ is the inverse of $[\cdot]_\times$, mapping a skew-symmetric matrix to a vector in $\mathbb{R}^3$. The desired thrust in the body frame is given by $\boldsymbol{F}_d = \boldsymbol{R}^\top \boldsymbol{F}'_d$. For configurations which are not fully actuated, the desired orientation $\boldsymbol{R}_d$ and desired torque $\boldsymbol{\tau}_d$ can be adjusted based on $\boldsymbol{F}'_d$ [13]. The desired wrench can be expressed as $\boldsymbol{W}_d = [\boldsymbol{T}_d^\top, \boldsymbol{\tau}_d^\top]^\top$. The mapping from $\boldsymbol{W}_d$ to rotor inputs can then be formulated as a regularized optimization problem

$$\min_{\boldsymbol{u}_c} \|\boldsymbol{A}\boldsymbol{u}_c - \boldsymbol{W}_d\|_2^2 + \delta\|\boldsymbol{u}_c\|_2^2,$$

where $\delta > 0$ is the regularization constant. The control input has the closed form

$$\boldsymbol{u}_c = \boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top + \delta\boldsymbol{I})^{-1}\boldsymbol{W}_d,$$

where $\boldsymbol{I} \in \mathbb{R}^{6\times6}$ is the identity matrix. Since the motor thrust is limited, we bound the control input to the motor limits.

VI. Experiments

In this section, we first verify the flight performance of two generated configurations from Fig. 8 using the real-time physics-based simulation engine `Bullet`. We then present a real world toy example to further confirm the feasibility of the proposed pipeline.

A. Simulation experiments

Using the controller in Section V, we are able to control a configuration to follow a given trajectory. We choose two of the optimal configurations with 30 and 60 modules, obtained in Fig. 8 using the 10 randomly generated target wrench set. The flight results are shown in Fig. 9. For clarity, we plot the 3D illustration with black squares only. Both assemblies track a circular trajectory, with relatively low overall orientation error within 3°. The tracking performance

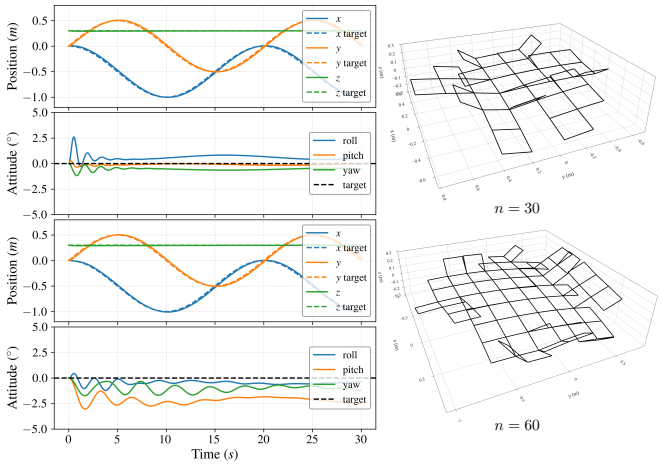


Fig. 9. Two optimal configurations from Fig. 8 with $n = 30$ and $n = 60$, tracking a circular trajectory with fixed identity orientation.

can be further improved by fine-tuning the control parameters in Section V.

B. Real world toy case

We also conduct a real world toy example. We define a simple push task along the horizontal y-axis, with a target force of $0.36\,\mathrm{N}$ along the y-axis, we set $\boldsymbol{b}_{k,\tau} = \boldsymbol{0}$, and the modified force set is $\{[0, 0.36, nmg]^\top, [0, -0.36, nmg]^\top, [0, 0, nmg]^\top\}$. By solving Eq. (1) as in Algorithm 2, a feasible solution exists with $n = 2$, with optimal angles $\alpha_0 = 0°, \alpha_1 = -22.06°, \alpha_2 = 44.12°$. We assemble two modules as in Fig. 10, as there is only one connection edge, we 3D-print both connectors for simplicity. Note that it is not necessary to have all 6 controllable DOF to perform aerial physical interaction [16], as in this case, two modules have 5 controllable DOF and track a 5-DOF trajectory, while the last DOF is dynamically coupled with the translational dynamics. During physical interaction, the feedback controller is responsible for keeping the system stable. In Fig. 11 we show two free-flight trajectory tracking results of the toy-case prototype. These experiments are performed indoors using a VICON motion capture system for localization. Communication between the aerial system and the VICON system is managed using the Crazyswarm framework [25]. In both flights, the 2-module structure follows a 3D figure-8 trajectory $\boldsymbol{p}_d(t) = [l_0 \sin(2\pi t/t_c) \cos(2\pi t/t_c), l_0 \sin(2\pi t/t_c), h_0 - l_0/3 \sin(2\pi t/t_c)]^\top$ with fixed roll and yaw angles. As the pitch angle is coupled with the translational dynamics, it is therefore not tracked independently. As the overall structure has 5 controllable DOF, it can maintain the roll angle at a fixed target. The experiments with $0°$ and $-5°$ are given in Fig. 11. The tracking results indicate that the system maintains its horizontal orientation at $0°$, enabling physical interaction along the horizontal y-axis as the target wrench set requires.

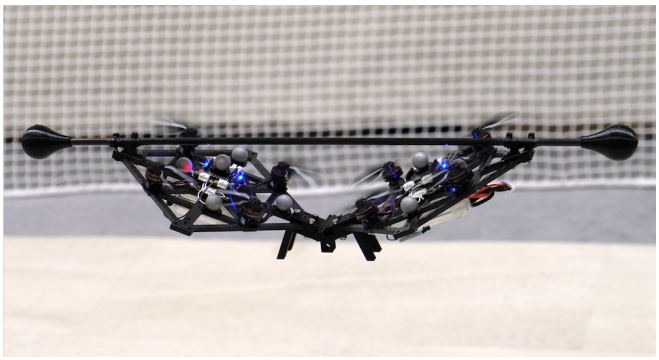


Fig. 10. The 2-module configuration hovering in the air.

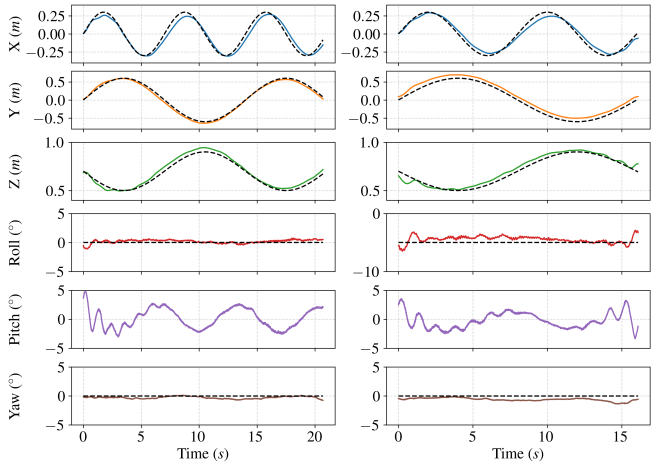


Fig. 11. Experiment results of the system in free flight when tracking a 3D figure-8 trajectory while maintaining the target roll angle at $0°$(left) and $-5°$(right). The black dashed lines are the target trajectories.

VII. CONCLUSIONS

In this work, we propose a novel pipeline to design optimal configurations for modular aerial systems. We use homogeneous quadrotor-based modules and assemble them into non-planar structures while ensuring collision-free inter-module downwash constraints, to achieve efficient, physically feasible optimal designs. The proposed approach is scalable and general, and can be applied to other modular robotic systems. In the future, we plan to explore configurations with more modules and construct various assemblies to perform aerial manipulation tasks.

REFERENCES

[1] J. Seo, J. Paik, and M. Yim, "Modular reconfigurable robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, no. Volume 2, 2019, pp. 63–88, 2019.

[2] R. Oung, F. Bourgault, M. Donovan, and R. D'Andrea, "The distributed flight array," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 601–607.

[3] D. Saldaña, B. Gabrich, G. Li, M. Yim, and V. Kumar, "ModQuad: The flying modular structure that self-assembles in midair," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 691–698.

[4] M. Ryll, G. Muscio, F. Pierri, E. Cataldi, G. Antonelli, F. Caccavale, D. Bicego, and A. Franchi, "6D interaction control with aerial robots: The flying end-effector paradigm," *The International Journal of Robotics Research*, vol. 38, no. 9, pp. 1045–1062, 2019.

[5] M. Allenspach, K. Bodie, M. Brunner, L. Rinsoz, Z. Taylor, M. Kamel, R. Siegwart, and J. Nieto, "Design and optimal control of a tiltrotor micro-aerial vehicle for efficient omnidirectional flight," *The International Journal of Robotics Research*, vol. 39, no. 10-11, pp. 1305–1325, 2020.

[6] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed," *IEEE Robotics & Automation Magazine*, vol. 17, no. 3, pp. 56–65, 2010.

[7] A. Nikou, G. C. Gavridis, and K. J. Kyriakopoulos, "Mechanical design, modelling and control of a novel aerial manipulator," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 4698–4703.

[8] 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*, vol. 8, no. 10, pp. 6187–6194, 2023.

[9] L. Bauersfeld, K. Muller, D. Ziegler, F. Coletti, and D. Scaramuzza, "Robotics meets fluid dynamics: A characterization of the induced airflow below a quadrotor as a turbulent jet," *IEEE Robotics and Automation Letters*, vol. 10, no. 2, pp. 1241–1248, 2025.

[10] J. Xu and D. Saldaña, "Finding optimal modular robots for aerial tasks," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 11 922–11 928.

[11] B. Gabrich, D. Saldaña, and M. Yim, "Finding structure configurations for flying modular robots," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 6970–6976.

[12] Y. Su, Z. Jiao, Z. Zhang, J. Zhang, H. Li, M. Wang, and H. Liu, "Flight structure optimization of modular reconfigurable UAVs," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 4556–4562.

[13] M. Li, K. Cui, and H. Koeppl, "A modular aerial system based on homogeneous quadrotors with fault-tolerant control," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 8408–8414.

[14] P. Foehn, E. Kaufmann, A. Romero, R. Penicka, S. Sun, L. Bauersfeld, T. Laengle, G. Cioffi, Y. Song, A. Loquercio, and D. Scaramuzza, "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *Science Robotics*, vol. 7, no. 67, p. eabl6259, 2022.

[15] K. Cui, M. Li, C. Fabian, and H. Koeppl, "Scalable task-driven robotic swarm control via collision avoidance and learning mean-field control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1192–1199.

[16] B. Yüksel, C. Secchi, H. H. Bülthoff, and A. Franchi, "Reshaping the physical properties of a quadrotor through ida-pbc and its application to aerial physical interaction," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 6258–6265.

[17] J. Liu, Y. Wang, S. Ma, and Y. Li, "Enumeration of the non-isomorphic configurations for a reconfigurable modular robot with square-cubic-cell modules," *International Journal of Advanced Robotic Systems*, vol. 7, no. 4, p. 31, 2010.

[18] K. Stoy and D. Brandt, "Efficient enumeration of modular robot configurations and shapes," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 4296–4301.

[19] B. Gabrich, G. Li, and M. Yim, "ModQuad-DoF: A novel yaw actuation for modular quadrotors," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 8267–8273.

[20] J. A. Bondy and U. S. R. Murty, *Graph theory with applications*. Macmillan London, 1976, vol. 290.

[21] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[22] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[23] M. Hamandi, F. Usai, Q. Sablé, N. Staub, M. Tognon, and A. Franchi, "Design of multirotor aerial vehicles: A taxonomy based on input allocation," *The International Journal of Robotics Research*, vol. 40, no. 8-9, pp. 1015–1044, 2021.

[24] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 5420–5425.

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