

Author version

Published as:

Cao, P., Hwang, J. T., Bewley, T. R., & Kuester, F. (2022). Mission-oriented trajectory optimization for search-and-rescue multirotor uavs in cluttered and gps-denied environments. In AIAA Aviation 2022 Forum. <https://lsdolab.github.io/lsdo-publications/publication/cao2022mission/>

BibTeX for this reference:

<https://lsdolab.github.io/lsdo-publications/publication/cao2022mission/>

```
@inproceedings{cao2022mission,  
  author = {Pengcheng Cao and John T. Hwang and Thomas R. Bewley and Falko Kuester},  
  title = {Mission-Oriented Trajectory Optimization for Search-and-Rescue Multirotor UAVs  
    in Cluttered and GPS-Denied Environments},  
  booktitle = {AIAA Aviation 2022 Forum},  
  year = {2022}  
}
```

Mission-Oriented Trajectory Optimization for Search-and-Rescue Multirotor UAVs in Cluttered and GPS-Denied Environments

Pengcheng Cao^{*}, John T. Hwang[†], Thomas R. Bewley[‡], and Falko Kuester[§]
University of California San Diego, La Jolla, CA 92093, USA

Planning Search-and-rescue (SAR) missions for UAVs in cluttered and GPS-impaired environments remains a challenging topic in both robotics and aerospace related research areas. Small multirotors, especially quadrotors, are deployed by many in indoor or confined spaces due to their structural simplicity, ability to hover and take off and land vertically, and good maneuverability. However, most of small multirotors suffer from less-than-30-min flight time on one charge imposing constraints on missions including searching for survivors and payload delivery. In this paper, we introduce a mission-oriented trajectory generation approach in order to improve the quantitative performance of multirotors by reducing single-flight energy and time consumption. In our methodology, each flight mission is initially planned using computationally cheap path planning algorithms to generate a set of waypoints as an initial guess of the trajectory. Next, a multi-phase optimal control problem is formulated with the mission-specific objective function. Apart from constraints of initial and final conditions, this multi-phase optimal control problem is also subject to dynamics constraints from both quadrotor equations of motion and rotor aerodynamics models as well as path constraints. The optimal control problem is then solved in the frame of multi-disciplinary design optimization (MDO) by setting the energy consumption, times, control inputs, and states as design variables, and it has the potential to be converted into a simultaneous design and control process when including quadrotor design parameters as new design variables. After that, simulations are performed to validate the effectiveness of proposed methodology and to compare its results with those from peer methods. Three mission scenarios are set up to validate the proposed methodology. The simulation results indicate that the proposed method can decrease the flight time by up to 6.69% or mechanical energy consumption by 3.00%, and it outperforms two other trajectory generation methods for specific mission performance.

I. Nomenclature

$\mathbf{x}(t)$	= quadrotor state variable, $\mathbf{x}(t) \in \mathbb{R}^{12 \times 1}$.
$\mathbf{u}'(t)$	= quadrotor control input, $\mathbf{u}'(t) = [n_1, n_2, n_3, n_4]^T \in \mathbb{R}^{4 \times 1}$.
t	= flight time in seconds.
$E(t)$	= mechanical energy consumption over time in Joules .
u_x	= induced airflow velocity in m/s .
Ω, ω	= angular velocity of propeller in rad/s .
D, R	= diameter and radius of a propeller and rotor disk in m .
ρ	= density of air, in this paper $\rho = 1.225 kg/m^3$
$c(\bar{r}), \theta(\bar{r})$	= radial chord and twist angle distribution of a propeller.
$V_\infty, \bar{V}_\infty$	= free-stream velocity, $\bar{V}_\infty = V_\infty/\Omega R$.
u'_x	= wake air velocity in m/s .
$\lambda(\bar{r}, \psi, t)$	= $u_x/\Omega R$ the local dimensionless induced airflow velocity at $(\bar{r}, \psi)$ of rotor disk and time t

^{*}Ph.D. candidate in Mechanical Engineering, University of California, San Diego; p5cao@eng.ucsd.edu. AIAA student member

[†]Assistant professor, Department of Mechanical and Aerospace Engineering, University of California, San Diego; jhwang@eng.ucsd.edu. AIAA member

[‡]Professor, Department of Mechanical and Aerospace Engineering, University of California, San Diego; bewley@eng.ucsd.edu.

[§]Professor, Department of Structural Engineering, University of California, San Diego; fkuester@eng.ucsd.edu.

λ_{0,r_i}	= dimensionless uniform axial induced velocity component of i -th rotor.
λ_{s,r_i}	= dimensionless side-to-side variation component of induced velocity of i -th rotor.
λ_{c,r_i}	= dimensionless fore-to-aft variation component of induced velocity of i -th rotor.
$r, \bar{r}$	= radial coordinate in rotor disk, $\bar{r} = r/R$.
ψ_r	= angular coordinate in rotor disk in radians.
α	= angle of incidence on a rotor disk in radians.
α_e	= angle of attack at a blade element in radians.
θ_e	= blade twist angle at a blade element in radians.
ϕ_e	= inflow angle at a blade element in radians.
J	= propeller advance ratio, $J = V_\infty \sin \alpha / \Omega R = V_\infty \sin \alpha / nD$.
n, n_i	= propeller rotational speed in revolutions per minute (RPM) or revolutions per second (rev/s).
C_T, T	= thrust coefficient and thrust of a propeller.
C_Q, Q	= torque coefficient and drag torque of a propeller.
C_{roll}, M_r	= roll moment coefficient and local pitch moment of a propeller.
C_{pitch}, M_p	= pitch moment coefficient and local pitch moment of a propeller.
C_{P,r_i}	= power coefficient of i -th propeller.
N_b	= number of blades, $N_b = 2$ in this paper.
$C_{dT}, C_{dQ}, C_{dL}, C_{dD}$	= local thrust, torque, lift, and drag coefficients at blade element.
ψ, θ, ϕ	= quadrotor attitude: yaw, pitch, and roll angles by Euler ZYX convention wrt inertial frame.
p, q, r	= roll, pitch, and yaw rotation rates wrt quadrotor body-fixed frame in rad/s .
$F_z, \tau_x, \tau_y, \tau_z$	= total thrust and roll, pitch, yaw torque wrt the quadrotor body-fixed frame.
f_{obj}	= the objective function.
t_f	= final time in seconds when a mission is completed.
t_{phase_i}	= time of finishing i -th phase in seconds.

II. Introduction

THE search-and-rescue (SAR) operations with robotic assistance have become an emerging topic in both academia and industry over the last several decades. The development of more robust and versatile field robots has been spurred by more frequent occurrences of tragic events and risks and challenges faced by first responders during SAR operations. Among these robots, the unmanned aerial vehicles (UAVs) have demonstrated their advantages of being agile and fast with low operating costs and capable of detecting hazards to be avoided by task force members [1]. For example, two rotary-wing UAVs were deployed in a total of 14 missions and 38 flights by a team of 3 personnel to conduct search-and-rescue tasks in cluttered and destroyed urban areas in Florida and Mississippi in the aftermath of Hurricane Katrina [2]. In addition, a multipurpose UAV was developed and deployed by Silvagni et al. [3] for mountain rescue operations in the event of avalanche. Other applications of SAR UAVs include the deployment of a unmanned helicopter to rapidly detect collapsed buildings to provide guidance for ground rescue teams after the 2013 Lushan earthquake in China [4].

As discussed above, UAVs are primarily implemented for SAR missions in outdoor urban or rural areas where GPS or other GNSS signals are available. However, more recent research efforts are focused on navigation, localization, and mission planning in cluttered and GPS-denied environments. Tomic et al. [5] developed a UAV system which is able to process an extended Kalman filter (EKF) for localization and navigate through waypoints via recognition of known objects onboard using a distributed multi-board computation platform. This configuration enables the UAV to execute both indoor and outdoor urban SAR missions with no GPS access. Also, Kulkarni et al. [6] performed the simulation where the UAV searches for and locates the victim in two indoor scenarios and then navigates with the Q-learning framework based on the received signal strength (RSS) and rewards of the reinforcement learning algorithm. Other work includes search-and-rescue missions in a forest of no GPS-access by Tian et al. [7] using collaborative simultaneous localization and mapping (CSLAM) and UAV target searching and tracking using partially observable Markov decision process (POMDP) by Vanegas et al [8].

However, most electrical rotary-wing or multirotor UAVs are known to suffer from relatively short flight time upon a single charge [9]. In addition, the payload capacity of these UAVs is known to be constrained [10] due to size and actuator limitations. The bottleneck of implementing quadrotor UAVs for SAR purposes is to achieve a balance between payload capacity, battery capacity, and flight time [10]. In an attempt to overcome this bottleneck, researchers have been formulating and solving optimal control and path planning problems to minimize certain mission objectives.

Andersen [11] described five search patterns [12] in outdoor SAR missions and compared their path coverages and travelled distances given the same search area. Cabreira et al. [13] [14] introduced an energy-aware grid-based coverage path planning algorithm (EG-CPP) in order to minimize energy consumption of UAV missions when covering irregular-shaped areas. On the other hand, Hayat et al. [15] developed a multi-objective optimization algorithm to plan paths for the UAV to locate a target and navigate in a bounded area and used a genetic algorithm to minimize the mission completion time. These approaches, however, either use an indirect method to estimate mechanical energy consumption like in [13] or linearize or over-simplify the UAV equations of motion and aerodynamics, which results in significant deviation between the computed UAV paths, time, and energy consumption and those of real experiments thus undermining the reliability of these methods in real missions.

On the other hand, multi-disciplinary design optimization (MDO) has revealed its potential in optimizing mission trajectories and modeling nonlinear dynamical systems including multirotors [16]. Furthermore, OpenMDAO [17] is developed to provide a framework for assembling complex optimization problems from many components as well as an efficient architecture for computing gradients in large-scale systems. Relying on OpenMDAO and other MDO frameworks, a number of researchers have applied MDO to solving complex aircraft or robot design and trajectory optimization problems, including Hwang et al. 's work [18] on satellite's design and operation, Kao et al. 's work [19] on aircraft mission analysis and trajectory optimization, and Yan et al. [20] 's work on optimal control and design of a ball-pitching robotic arm. In authors' previous work, we have studied the multi-objective design optimization (MOO) [21] [22] and optimal control [23] for a fully-actuated hexrotor UAV. However, the application of MDO to multirotor optimal control problems is not found by authors although MDO does have the capability of solving trajectory optimization for nonlinear dynamical systems by viewing time, controls, and states as design variables.

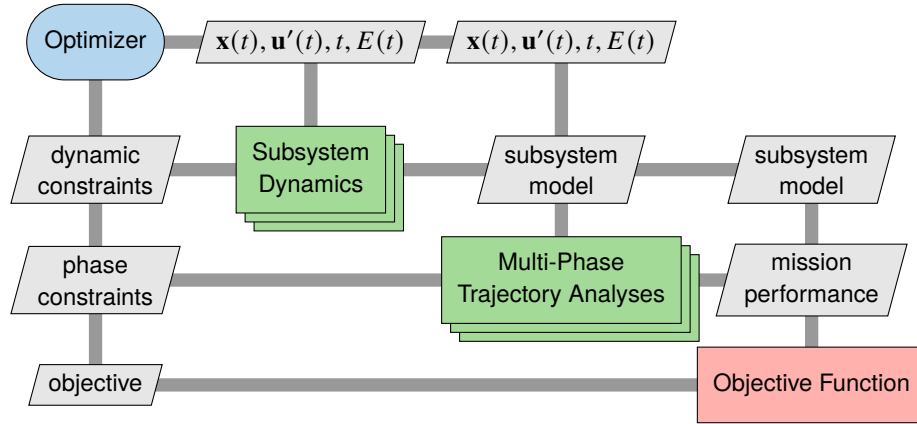


Figure 1 The extended design structure matrix (XDSM) of a multi-phase trajectory optimization problem.

In this work, the authors formulate the trajectory optimization as a multi-phase optimal control problem for quadrotors in SAR scenarios. In our problem formulation, we include nonlinear equations of motion and aerodynamics of each rotor as dynamic constraints, and all phases are constrained by initial condition, waypoints, and final condition. By viewing energy consumption, times, control inputs, and states as design variables, we are able to solve the trajectory optimization using MDO techniques [24]. The proposed method can output optimal trajectories with respect to mission objectives for SAR UAVs in GPS-denied environments. The ordinary differential equations (ODE) system structure of the proposed method is illustrated in Fig. 1. In future work, this approach has the potential to be converted into a simultaneous design and control process when including the quadrotor design parameters.

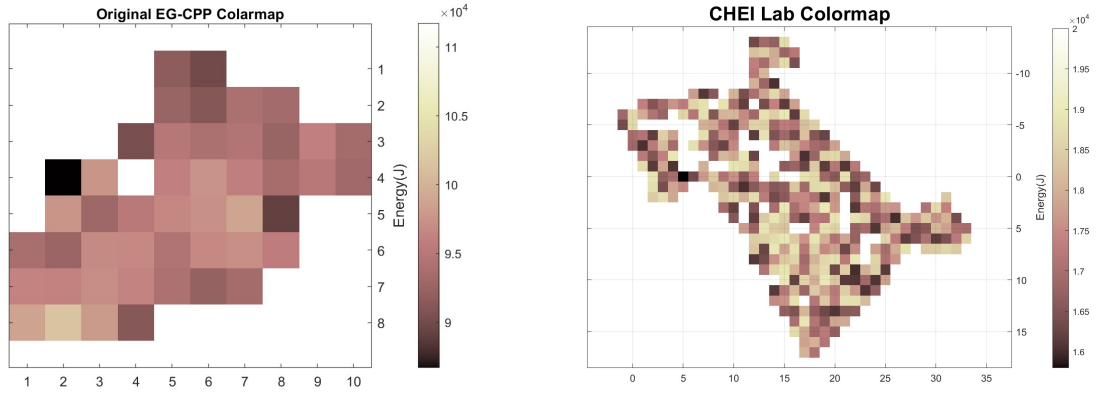
This paper is organized as follows: Section §III discusses the set of methods and models implemented by proposed trajectory optimization approach, with Section §III.A discussing the path planning methods providing waypoints and initial guesses and Section §III.B discussing the two subsystems of ODEs that builds up the dynamic model and constraints. Section §III.C and section §11c describes this paper's approach of formulating optimization problem, solving methods, and collision checks. Section §IV discusses the simulation environment and provides comparisons of the results from different trajectory generation methods. Last but not least, Section §V discuss the conclusions from this research and goals of future work.

III. Methodology

A. Flight Path Planning

Path planning is the process of generating collision-free paths from an initial state to a goal state with optimal or sub-optimal cost [25]. In this work, the initial planning of the mission path is realized using computationally inexpensive algorithms available in the literature. These path planning algorithms are expected to render a set of waypoint and initial guess for each mission flight. In this work, we implement Rapidly Exploring Random Tree Star (RRT*) algorithm [26] for minimal-time missions, and an energy-aware grid-based coverage path planning (CPP) algorithm [13] for the missions minimizing mechanical energy consumption.

1. Coverage Path Planning



(a) Original Colormap from [13].

(b) Energy-Aware Colormap of CHEI Lab of UCSD.

Figure 2 Energy Colormaps of Path Start Nodes.

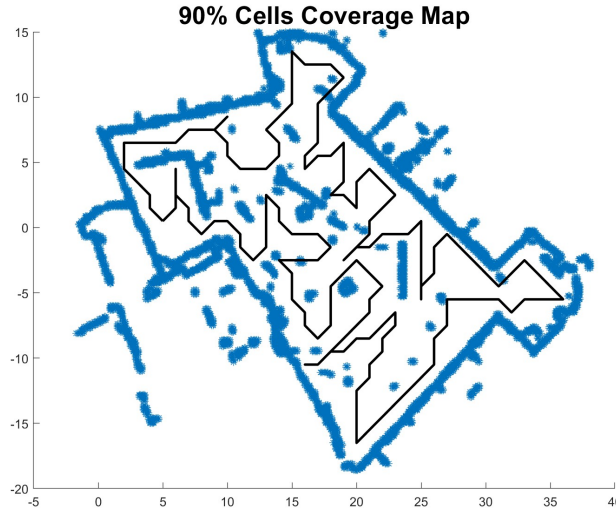


Figure 3 EG-CPP coverage path for CHEI Lab.

We define the searching-for-survivors mission as the following: In this scenario, the UAV needs to visit every collision-free square cell in the work space, for example, a grid map, via a path of minimum energy consumption. This is a simplified scenario where the UAV carries on the task of searching for victims in a known indoor space and return

their locations. In order to perform fast search for coverage path, we assume the UAV flies at the altitude of $2m$ and simplify the 3D map to be a 2D binary occupancy grid. In this grid, each cell is a $1m \times 1m$ square which the UAV needs to cover on the path. Here we assume the UAV flies at the altitude of $2m$ above ground level (AGL) and set the Z-coordinates of every waypoint to be $2m$.

In the energy-aware grid-based coverage path planning (EG-CPP) method proposed by Cabreira et al. [13], the irregular-shaped search area is discretized into a regular grid, and each cell in the grid is assigned with a coverage path starting from this cell marked with its associated energy cost. This step is also visualized as a colormap where each cell in the free configuration space (C-space) is filled with a color representing the energy consumption of the path starting from this cell as illustrated in Fig. 2a. Similarly, when planning the path for our missions, we also discretize the free C-space into grids and obtain its colormap as shown in Fig. 2b. We assume the UAV can cover a $2m \times 2m$ area surrounding it when visiting a cell. Given the colormap information, one can find that the optimal start node is $[5.0, 0.0, 2.0]$ with the minimum cost of 15,863 J. Its associated coverage path is shown in Fig 3, with a guarantee of covering 90% of the cells in free C-space. This path include in total 83 waypoints and has a length of 205.95m.

2. RRT* Path Planning

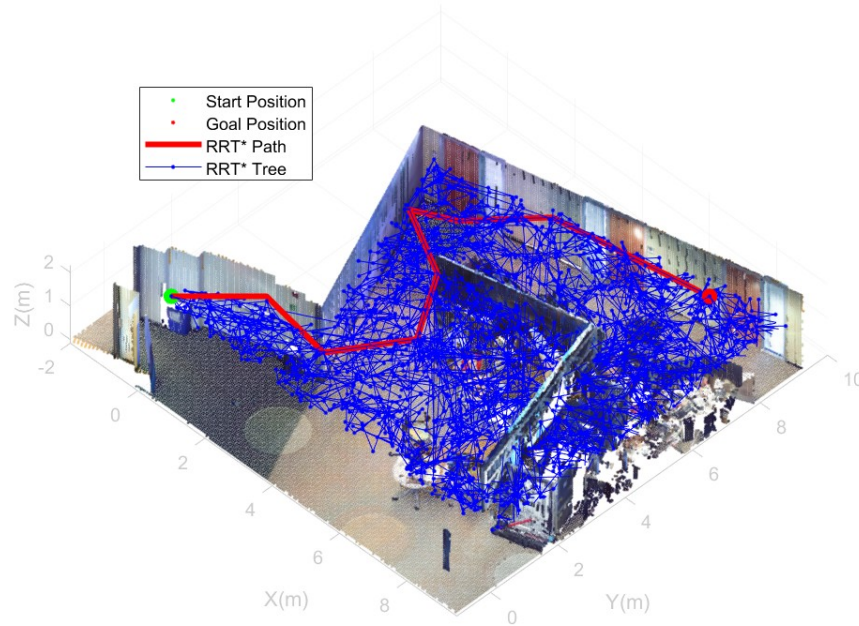


Figure 4 RRT* path planned on CHEI Lab point cloud map.

In some scenarios, the UAV will need to visit several rooms in the building to locate survivors and report their positions. In some other cases, the UAV needs to deliver important payloads like medications and life-aid devices to victims on time. These missions are typically in the interest of time such that in these missions time is set to be the objective. In order to plan the paths fast and robustly, we select RRT* [26], a renowned sampling-based path planning algorithm, to find feasible paths for our missions. RRT* builds a randomly exploring tree incrementally, but only connects the neighboring nodes along a minimum-cost path, and rewires the tree by shifting neighboring nodes to another lower-cost random neighbor at each iteration. RRT* is proven to be asymptotically optimal and computationally efficient [25].

Given a fixed maximum speed, the time consumption of a path is positively associated with its total length. Therefore, we attempt to find the optimal or sub-optimal path with minimum path length. A minimum-length path is illustrated in Fig. 4 where a UAV travels from the origin to a waypoint location $([8.0, 8.0, 2.0])$ on the CHEI Lab point cloud map.

By setting the maximum connection distance to be $2.0m$, RRT* gives a path of 8 waypoints and a total length of $17.41m$.

B. Quadrotor Dynamics Modeling

In the next three sub-sections, we will introduce both the dynamic inflow model and blade element momentum theory (BEMT) to help characterize the aerodynamics of a single rotor, and then combine aerodynamic equations of 4 rotors. And In the last sub-section, we present the quadrotor equations of motion as its flight dynamics.

Figure 5 The illustration of body-fixed frame and rotor disk coordinate systems.

Pitt and Peters [27] further developed a system of linear ODEs to describe the dynamic inflow model of a rotor disk which has become one of the most popular theories in this area during the last 4 decades. The Pitt-Peters model elegantly relates the dimensionless aerodynamic loading to dimensionless induced flow distribution. Knowing that the induced velocities distribution in a rotor disk is given by a function of polar coordinates $(\bar{r}, \psi)$ and time t :

where λ is the local dimensionless induced airflow velocity $u_x/\Omega R$, λ_0 , λ_s , and λ_c are the uniform, side-to-side, and fore-to-aft variations in induced airflow, respectively. And the dimensionless radial coordinate $\bar{r}$ is r/R where R is the propeller radius. The inflow dynamics of this rotor can be described by the following state-space model where only first-harmonic terms are kept to render the linear first-order state-space representation:

$$\begin{bmatrix} \dot{\lambda}_0 \\ \dot{\lambda}_s \\ \dot{\lambda}_c \end{bmatrix} = -\mathbf{M}^{-1}\mathbf{L}^{-1} \begin{bmatrix} \lambda_0 \\ \lambda_s \\ \lambda_c \end{bmatrix} + \mathbf{M}^{-1} \begin{bmatrix} C_T \\ C_{roll} \\ C_{pitch} \end{bmatrix},$$

$$\text{where } \mathbf{L} = \frac{1}{\bar{V}_\infty} \begin{bmatrix} \frac{1}{2} & 0 & \frac{15\pi}{64} \sqrt{\frac{1-\sin\alpha}{1+\sin\alpha}} \\ 0 & -\frac{4}{1+\sin\alpha} & 0 \\ \frac{15\pi}{64} \sqrt{\frac{1-\sin\alpha}{1+\sin\alpha}} & 0 & -\frac{4\sin\alpha}{1+\sin\alpha} \end{bmatrix}, \mathbf{M} = \begin{bmatrix} \frac{128}{75\pi} & 0 & 0 \\ 0 & \frac{-16}{45\pi} & 0 \\ 0 & 0 & \frac{-16}{45\pi} \end{bmatrix}.$$

Here $\bar{V}_\infty$ is the dimensionless free-stream velocity $V_\infty/\Omega R$ and α is the angle of incidence. In the scope of this work, we compute the magnitude of free-stream velocity to be the norm of current flight velocity vector $[\dot{x}, \dot{y}, \dot{z}]$, and angle of incidence α to be complimentary angle of the angle between the free-stream velocity and axial inflow velocity, i.e. the normal vector of the rotor disk. The rotor disk coordinates is shown in Fig. 5, and the equations to compute $\bar{V}_\infty$ and α is given as follows:

$$\bar{V}_\infty = \frac{\sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2}}{2\pi n R}, \alpha = \pi/2 - \arccos\left(\frac{[\dot{x}, \dot{y}, \dot{z}] \cdot \mathbf{n}_{rotordisk}}{\sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2} \cdot 1}\right), \quad (2)$$

such that we will need the input of quadrotor flight states here to evaluate the inflow states of each propeller at different times.

2. Blade Element Momentum Theory

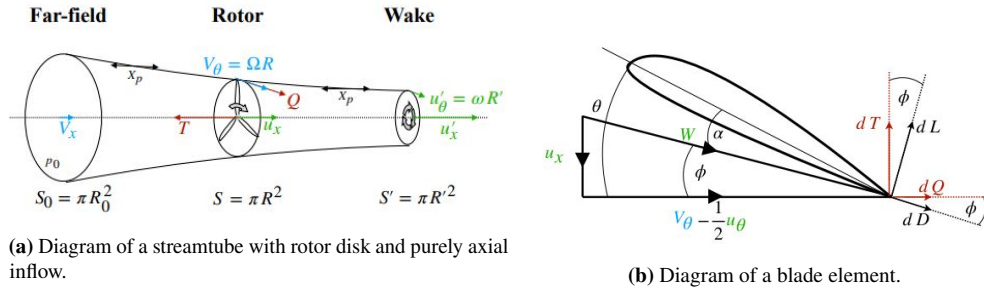


Figure 6 Illustration of BEMT parameters [28].

In order to provide the non-dimensional aerodynamic loads, we need a fast and robust approach to estimate the thrust, moment, and power coefficients. These aerodynamic loads are usually computed using Blade Element Theory with the angle of attack on the blade element estimated by either induced velocity from either momentum theory or other inflow models. As illustrated in Fig. 6b, the angle of attack α_e exerted on this element is computed by

$$\alpha_e = \theta_e - \phi_e, \quad (3)$$

where θ_e is the blade twist angle at the element location and ϕ_e is called inflow angle computed by the axial and in-plane components of induced velocity at this blade element. The approach to compute the inflow angle with momentum theory only considering uniform and steady axial and in-plane inflows is known as Blade Element Momentum Theory (BEMT) [29].

Some literature points out that this approach results in non-negligible errors in predicting aerodynamics loads when advance ratio is smaller or equal to than 0.3 ($J < 0.3$) [30] or at lower Reynolds numbers ($Re < 3 \times 10^5$). While for our trajectory optimization problem setting, the range of $J < 0.3$ is replaced by the experimental data measured with thrust testing apparatus one can find in [21], and low speed assumption holds for us to neglect complicated aerodynamic effects [31]. However, the implementation of BEMT still provides a good estimate for $J > 0.3$, and thus we create the direct mapping from the advance ratio of a propeller to its corresponding thrust, pitch, roll, and yaw moment coefficients.

Algorithm 1 BEMT Estimation

```

1: Input  $c(\bar{r})$ ,  $\theta(\bar{r})$ ,  $D$  (propeller geometry)
2:
3: for  $n \in [n_{min}, n_{max}]$  rev/s do
4:   for  $V_\infty \in [0.01, V_{max}]$  m/s do
5:
6:      $J = \frac{V_\infty \sin \alpha}{nD}$ , store to  $J$ -array
7:
8:     if  $0.01 \leq J \leq 1.00$  and  $J \notin J$ -array then
9:       Solve for  $\phi$  iteratively via  $\tan \phi = \frac{J}{\pi} \frac{1 + a_x}{1 - a_r}$  where  $a_x = \frac{1}{\frac{4 \sin^2 \phi}{\sigma C_{dT}} - 1}$ ,  $a_r = \frac{1}{\frac{4 \sin \phi \cos \phi}{\sigma C_{dQ}} + 1}$ 
10:      Compute  $a_x$  and  $a_r$  with current  $\phi$ 
11:      Compute  $T$ ,  $Q$ ,  $M_r$ ,  $M_p$  using (4 - 7), respectively
12:       $C_T = \frac{T}{\rho n^2 D^4}$ , store to  $C_T$ -array
13:       $C_Q = \frac{Q}{\rho n^2 D^5}$ , store to  $C_Q$ -array
14:       $C_{roll} = \frac{M_r}{\rho n^2 D^5}$ , store to  $C_{roll}$ -array
15:       $C_{pitch} = \frac{M_p}{\rho n^2 D^5}$ , store to  $C_{pitch}$ -array
16:    end if
17:  end for
18: end for

```

In the BEMT method, the propeller is simplified as a rotor disk model fixed within a stream tube of the airflow as shown in Fig. 6a. The stream tubes are constructed differently for a UAV rotor during hovering and forward flight [32]. Thus we introduce the angle of incidence α to help describe the non-axial inflow such that the advance ratio $J = V_\infty \sin \alpha / nD$. Therefore, we estimate the rotor thrust T , drag torque Q , pitch moment M_p , and roll moment M_r using the following 4 equations:

$$T = \frac{N_b}{2\pi} \int_0^{2\pi} \int_{r_{hub}}^R \frac{1}{2} \rho V_\infty^2 \sin^2 \alpha c \frac{(1 + a_x)^2}{\sin^2 \phi} C_{dT} dr d\psi_e, \quad (4)$$

$$Q = -\frac{N_b}{2\pi} \int_0^{2\pi} \int_{r_{hub}}^R \frac{1}{2} \rho V_\infty \sin \alpha c \omega r^2 \frac{(1 + a_x)(1 - a_r)}{\sin \phi \cos \phi} C_{dT} dr d\psi_e, \quad (5)$$

$$M_p = \frac{N_b}{2\pi} \frac{(1 + a_x)^2}{\sin^2 \phi} \left(\int_0^\pi \int_{r_{hub}}^R \frac{1}{2} \rho V_\infty^2 \sin^2 \alpha c r \sin \psi_e C_{dT} dr d\psi_e - \int_\pi^{2\pi} \int_{r_{hub}}^R \frac{1}{2} \rho V_\infty^2 \sin^2 \alpha c r \sin \psi_e C_{dT} dr d\psi_e \right), \quad (6)$$

$$M_r = \frac{N_b}{2\pi} \frac{(1 + a_x)^2}{\sin^2 \phi} \left(\int_{\pi/2}^{3\pi/2} \int_{r_{hub}}^R \frac{1}{2} \rho V_\infty^2 \sin^2 \alpha c r \cos \psi_e C_{dT} dr d\psi_e - \int_{-\pi/2}^{\pi/2} \int_{r_{hub}}^R \frac{1}{2} \rho V_\infty^2 \sin^2 \alpha c r \cos \psi_e C_{dT} dr d\psi_e \right), \quad (7)$$

where V_∞ is the free stream velocity, c is the chord length of the blade element, N_b is the number of blades, a_x and a_r are the axial and radial induction factor, respectively [33], and ϕ is the inflow angle as indicated in Fig. 6b. And C_{dT} and C_{dQ} are the element thrust and torque coefficients obtained by projecting the element lift and drag coefficients via

$$\begin{bmatrix} C_{dT} \\ C_{dQ} \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} C_{dL} \\ C_{dD} \end{bmatrix} \quad (8)$$

Also the power coefficient of each rotor need to be recomputed and stored at each time step with the aerodynamic loads and inflow states with the following analytical equation [34][35]:

$$C_{P,r_i} = \frac{1.15C_T}{2\sqrt{\lambda_0^2 + J^2}} + \frac{\sigma C_{d0}}{8}(1 + 4.6J^2) + \frac{1}{8}\frac{f}{A}J^3 \quad (9)$$

where σ is the blade solidity computed by (blade area b)/(rotor area A), C_{d0} is the profile drag coefficient with a typical value of 0.008. f/A is the equivalent flat plate area f correlated with rotor area A which can be approximated as 1 for a quadrotor [35]. Since the power coefficient C_P is equivalent to C_Q , and we learned from experimental data that C_Q computed by BEMT predictions can possibly deviate from the real-world data in higher range of propeller RPM, we replace the C_Q with the C_P value computed here if BEMT C_Q predictions differs from C_P by over 5%.

3. State-Space Model of Rotor Aerodynamics

After computing thrust and torque coefficients using Algorithm 1, we are able to sum up the thrust and torques of the 4 rotors and also predicts the dynamic inflow states of each rotor which will help us recompute the power coefficient and efficiency they deviate significantly from the BEMT predictions [29]. The output vector in rotor aerodynamics is $\mathbf{y}_{aero} = [F_z, \tau_x, \tau_y, \tau_z]^T$. F_z is the sum of vertical thrusts of 4 motors, τ_x is the sum of moments with respect to x -axis of body-fixed frame. Similarly, τ_y and τ_z are sum of moments with respect to its y - and z axes, respectively. These four output states are the inputs to the equations of motion flight dynamics. Since each rotor has its individual inflow dynamics, we formulate the state vector $\mathbf{x}_0 \in \mathbb{R}^{16 \times 1}$ to be

$$\mathbf{x}_0 = [\lambda_{0,r_1}, \lambda_{s,r_1}, \lambda_{c,r_1}, \dots, \lambda_{0,r_4}, \lambda_{s,r_4}, \lambda_{c,r_4}, F_z, \tau_x, \tau_y, \tau_z]^T, \quad (10)$$

where $\lambda_{0,r_i}, \lambda_{s,r_i}, \lambda_{c,r_i}$ are the inflow states of i -the rotor. Also we consider the square of each motor's rotary speed as the control $\mathbf{u}_0 \in \mathbb{R}^{4 \times 1}$, and the vector of aerodynamic loads as disturbance vector $\mathbf{v}_0 \in \mathbb{R}^{12 \times 1}$, with

$$\begin{aligned} \mathbf{u}_0 &= [n_1^2, n_2^2, n_3^2, n_4^2]^T \\ \mathbf{v}_0 &= [C_{T,r_1}, C_{roll,r_1}, C_{pitch,r_1}, \dots, C_{T,r_4}, C_{roll,r_4}, C_{pitch,r_4}]^T. \end{aligned} \quad (11)$$

Therefore, we can now conclude the derivation of rotor aerodynamics subsystem. The state equations of the rotor aerodynamics can be written as follows:

$$\dot{\mathbf{x}}_0 = \mathbf{A}_0 \cdot \mathbf{x}_0 + \mathbf{D}_0 \cdot \mathbf{v}_0 + \mathbf{B}_0 \cdot \mathbf{u}_0 \quad (12)$$

$$\text{with } \mathbf{A}_0 = \begin{bmatrix} -\mathbf{M}^{-1}\mathbf{L}^{-1} & & & \\ & -\mathbf{M}^{-1}\mathbf{L}^{-1} & & \\ & & -\mathbf{M}^{-1}\mathbf{L}^{-1} & \\ & & & -\mathbf{M}^{-1}\mathbf{L}^{-1} \\ & & & & -\mathbf{I}^{4 \times 4} \end{bmatrix},$$

$$\mathbf{B}_0 = \begin{bmatrix} \mathbf{0}^{12 \times 1} & \mathbf{0}^{12 \times 1} & \mathbf{0}^{12 \times 1} & \mathbf{0}^{12 \times 1} \\ b_1 & b_2 & b_3 & b_4 \\ b_1 \cdot \frac{\sqrt{2}}{2} l_{arm} + k_{roll} & -(b_2 \cdot \frac{\sqrt{2}}{2} l_{arm} + k_{roll}) & -(b_3 \cdot \frac{\sqrt{2}}{2} l_{arm} + k_{roll}) & b_4 \cdot \frac{\sqrt{2}}{2} l_{arm} + k_{roll} \\ b_1 \cdot \frac{\sqrt{2}}{2} l_{arm} + k_{pitch} & b_2 \cdot \frac{\sqrt{2}}{2} l_{arm} + k_{pitch} & -(b_3 \cdot \frac{\sqrt{2}}{2} l_{arm} + k_{pitch}) & -(b_4 \cdot \frac{\sqrt{2}}{2} l_{arm} + k_{pitch}) \\ k_{yaw} & -k_{yaw} & k_{yaw} & -k_{yaw} \end{bmatrix},$$

$$\mathbf{D}_0 = \begin{bmatrix} -\mathbf{M}^{-1} & & & \\ & -\mathbf{M}^{-1} & & \\ & & -\mathbf{M}^{-1} & \\ & & & -\mathbf{M}^{-1} \\ \mathbf{0}^{4 \times 1} & \mathbf{0}^{4 \times 1} & \mathbf{0}^{4 \times 1} & \mathbf{0}^{4 \times 1} \end{bmatrix}.$$

where $b_1 = b_2 = b_3 = b_4 = C_T \rho D^4$, $k_{roll} = C_{roll} \rho D^5$, $k_{pitch} = C_{pitch} \rho D^5$, $k_{yaw} = C_Q \rho D^5$.

4. Quadrotor Flight Dynamics

The other sub-system of the quadrotor ODE model is its equations of motion (EoM) in 6 degrees of freedom. We set the inertial frame of the indoor space with north-west-up convention where x -axis points north, y -axis points west, and z -axis points upwards. The UAV body-fixed frame uses the same convention and has its three axes coincident with those of the inertial frame if starting from the origin. Therefore, we can organize the state vector of a quadrotor to be

$$\mathbf{x} = [x \ y \ z \ \psi \ \theta \ \phi \ \dot{x} \ \dot{y} \ \dot{z} \ p \ q \ r]^T \quad (13)$$

which represents x , y , z displacement components, ψ , θ , ϕ rotational displacement components, $\dot{x}$, $\dot{y}$, $\dot{z}$ velocity components in the inertial frame and p , q , r rotational rates with respect to x , y , z -axes of the body-fixed frame, respectively. The control input of quadrotor EoMs is

$$\mathbf{u} = [F_z \ \tau_x \ \tau_y \ \tau_z]^T. \quad (14)$$

The 6-DoF EoMs of quadrotor have been well studied and documented in the literature [36], such that we only give the final State-Space form of the quadrotor EoMs as the second sub-system. We use $s(\phi)$, $c(\phi)$, $s(\theta)$, $c(\theta)$, $t(\theta)$ to represent $\sin(\phi)$, $\cos(\phi)$, $\sin(\theta)$, $\cos(\theta)$, and $\tan(\theta)$, respectively, such that:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \quad (15)$$

$$\text{where } \mathbf{f}(\mathbf{x}) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ q \frac{s(\phi)}{c(\theta)} + r \frac{c(\phi)}{c(\theta)} \\ q \cdot c(\phi) - r \cdot s(\phi) \\ p + q \cdot s(\phi)t(\theta) + r \cdot c(\phi)t(\theta) \\ 0 \\ 0 \\ -g \\ \frac{I_y - I_z}{I_x} qr \\ \frac{I_z - I_x}{I_y} pr \\ \frac{I_x - I_y}{I_z} pq \end{bmatrix}, \mathbf{g}(\mathbf{x}) = \begin{bmatrix} \mathbf{0}^{6 \times 1} & \mathbf{0}^{6 \times 1} & \mathbf{0}^{6 \times 1} & \mathbf{0}^{6 \times 1} \\ g_1 & 0 & 0 & 0 \\ g_2 & 0 & 0 & 0 \\ g_3 & 0 & 0 & 0 \\ 0 & 1/I_x & 0 & 0 \\ 0 & 0 & 1/I_y & 0 \\ 0 & 0 & 0 & 1/I_z \end{bmatrix} \text{ and} \quad (16)$$

$$\begin{aligned}
g_1 &= -\frac{1}{m} [s(\phi)s(\psi) + c(\phi)c(\psi)s(\theta)] \\
g_2 &= \frac{1}{m} [c(\psi)s(\phi) - c(\phi)s(\psi)s(\theta)] \\
g_3 &= -\frac{1}{m} [c(\phi)c(\theta)]
\end{aligned} \tag{17}$$

C. Mission-Oriented Trajectory Optimization

Finding optimal solutions for dynamic system inputs given a task constitutes an optimal control problem. As can be seen from Equations (15) - (17), the EoM of the quadrotor is not a linear model. Both $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ are coupled with components of the state vector. Therefore when we attempt to find a solution trajectory minimizing a mission-specific objective function, the ODE system provides non-linear constraints converting the optimal control problem in to a non-linear programming problem which can be rapidly and robustly solved using multidisciplinary design optimization (MDO) methods. In the mean time, this optimal control problem is also expected to have the potential to become a simultaneous control and design optimization given we have modern computational tools like SNOPT [37] to run large-scale constrained optimization. Therefore, it is possible to run optimization over the free configuration and control space over the entire mission by viewing numerous discrete control inputs, trajectory segments, and travelling time as design variables in an MDO problem.

1. Design Variables

Optimal control problem can be solved in the context of the multidisciplinary design, analysis, and optimization (MDAO) [17]. As an initial step to formulate a simultaneous design and optimization problem and The UAV design and trajectory optimization can be coupled in a monolithic approach with the state variable $\mathbf{x}(t)$ and control variable $\mathbf{u}'(t) = [n_1, n_2, n_2, n_4]^T$ as part of the design variables [16]. In addition, we add the mechanical power consumption $E(t)$ and time consumption through all phases t as other two design variables due to the need for fulfilling missions objectives.

The state vector $\mathbf{x}(t)$ and control input vector $\mathbf{u}(t)$ are given as

$$\mathbf{x}(t) = [x \ y \ z \ \psi \ \theta \ \phi \ \dot{x} \ \dot{y} \ \dot{z} \ p \ q \ r]^T, \tag{18a}$$

$$\mathbf{u}'(t) = [n_1, n_2, n_2, n_4]^T, \tag{18b}$$

respectively, and the mechanical power consumption $E(t)$ is computed by

$$E(t) = 2\pi\rho D^5 \int_0^t C_{P,r_1} \cdot n_1^3 + C_{P,r_2} \cdot n_2^3 + C_{P,r_3} \cdot n_3^3 + C_{P,r_4} \cdot n_4^3 dt. \tag{19}$$

2. Problem Formulation

In a multi-phase mission of SAR quadrotor, the objective function is to be minimized with respect to the entire mission although it is evaluated at the final time t_f in the MDO method we use. Objective functions are considered as one of design variables and is kept record of and simulated throughout the entire mission. They can be arbitrarily constructed while the only two mission objectives we investigate in this paper are time and energy consumption of the mission. During each phase, the UAV is expected to travel from $i - 1$ -th waypoint to i -th waypoint, and this phase terminates once the UAV arrives within the proximity of the i -th waypoint. And We denote the terminal time of this phase as t_{phase_i} . The design variables here are state variable $\mathbf{x}(t)$, control variable $\mathbf{u}'(t)$, mechanical energy $E(t)$, and phase time t . In Section III.C.3 and III.C.4, we will propose an approach to solve the multi-phase optimal control problem using a nonlinear programming approach within the MDAO context. The formulation of the optimal control problem to optimize the trajectories of an SAR quadrotor UAV within the MDAO context is given as follows:

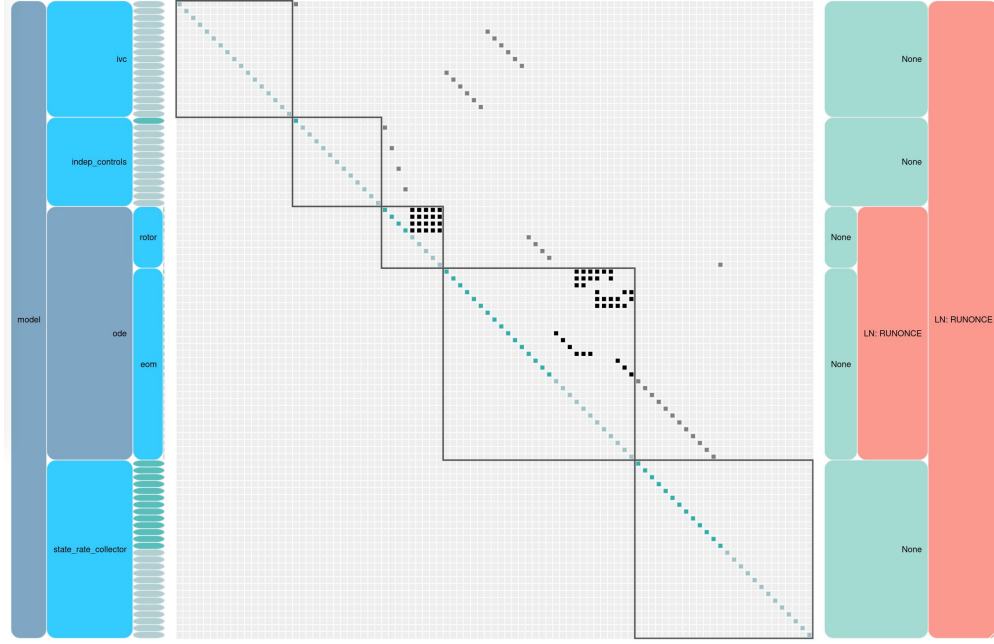


Figure 7 N^2 diagram of the ODE system with partial derivative components structure in OpenMDAO.

$$\begin{aligned}
 &\text{minimize:} && J = f_{obj}(\mathbf{x}(t), \mathbf{u}'(t), E(t), t)|_{t=t_f}; \\
 &\text{with respect to:} && \mathbf{x}(t), \mathbf{u}'(t), E(t), t; \\
 &\text{subject to dynamics:} && \dot{\mathbf{x}}_0 = \mathbf{A}_0 \cdot \mathbf{x}_0 + \mathbf{D}_0 \cdot \mathbf{v}_0 + \mathbf{B}_0 \cdot \mathbf{u}_0 \text{ (Eq. 12)} \\
 &&& \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x}) \cdot \mathbf{u} \text{ (Eq. 15)} \\
 &\text{and initial conditions:} && x(t=0) = x_0 \\
 &&& y(t=0) = y_0 \\
 &&& z(t=0) = z_0 \\
 &&& \psi(t=0) = \psi_0 \\
 &&& \text{all linear and angular velocity components} = 0 \\
 &\text{and final conditions:} && x(t=t_f) = x_f \\
 &&& y(t=t_f) = y_f \\
 &&& z(t=t_f) = z_f \\
 &&& \psi(t=t_f) = \psi_f \\
 &&& \text{all linear and angular velocity components} = 0 \\
 &\text{and path constraints:} && x(t=t_{phase_i}) = x_{phase_i} \\
 &&& y(t=t_{phase_i}) = y_{phase_i} \\
 &&& z(t=t_{phase_i}) = z_{phase_i} \\
 &&& \psi(t=t_{phase_i}) = \psi_{phase_i} \\
 &&& 0 \leq n_i \leq 350 \text{ rev/s} \\
 &&& x, y, z, \psi \text{ within the free C-space.}
 \end{aligned} \tag{20}$$

3. Pseudo-Spectral Methods

In order to minimize the computational time when computing the partial derivatives and changes of the design variables over time, we select to use implicit collocation techniques which are well-suited to gradient-based optimization with analytically computed derivatives [16]. Compared to time-marching, implicit collocation schemes can achieve same level of accuracy while utilize fewer grid points needed for analysis since the number of grid points is fixed during the optimization.

One implicit collocation used in this paper is high-order Legendre-Gauss-Lobatto (LGL) transcription which helps solve multi-phase optimal control problem. It discretizes each phase into segments at the LGL nodes such that the total number of nodes within each segment is odd. Nodes with an even index are the state discretization nodes. And controls are provided at both collocation nodes and state discretization nodes, both nodes are called as control discretization nodes. The states and state-rates are then interpolated to the collocation nodes using Hermite-interpolation [38]. We then obtain the collocation defects by subtracting interpolated values from ODE evaluated values. In the transcribed nonlinear programming problem, we constrain the collocations defects to be zero.

Another implicit collocation method used in this paper is called Radau Pseudo-Spectral Method (RPM). The RPM uses the Legendre-Gauss-Radau (LGR) nodes to discretize the problem within each segment. Discretized states and controls are provided at each LGR nodes. Different from LGL phases, the ODEs are evaluated at all nodes simultaneously, and the state rates are approximated using a Lagrange differentiation matrix [39].

4. Adjoint Method to Evaluate Derivatives

Analytic-derivative methods to evaluate model derivatives can be more efficient than numerical approaches since they solve for the total derivatives leveraging only inexpensive partial derivatives of internal model calculations [16]. Adjoint methods are one type of analytic-derivative methods and prove to be more efficient when number of input variables n_x is greater than that of output variables n_f [40]. We first start with the total derivative Jacobian of the objective function as

$$\frac{df_{obj}}{d\mathbf{X}} = \frac{\partial f_{obj}}{\partial \mathbf{X}} + \frac{\partial f_{obj}}{\partial \mathbf{Y}} \frac{d\mathbf{Y}}{d\mathbf{X}}. \quad (21)$$

The two partial derivative terms in Eqn. (21) are relatively cheap to compute, but the total derivative term $d\mathbf{Y}/d\mathbf{X}$ is expensive to compute directly. Instead, the governing equations $R(\mathbf{X}, \mathbf{Y}) = 0$ can be implemented to compute $d\mathbf{Y}/d\mathbf{X}$ indirectly with much lower computational cost.

$$R(\mathbf{X}, \mathbf{Y}) = 0, \quad (22)$$

$$\frac{dR}{d\mathbf{X}} = \frac{\partial R}{\partial \mathbf{X}} + \frac{\partial R}{\partial \mathbf{Y}} \frac{d\mathbf{Y}}{d\mathbf{X}} = 0 \Rightarrow \frac{d\mathbf{Y}}{d\mathbf{X}} = -\left[\frac{\partial R}{\partial \mathbf{Y}}\right]^{-1} \frac{\partial R}{\partial \mathbf{X}} \quad (23)$$

With Eqns. (22) and (23), one can derive the adjoint method in the form of

$$\frac{df_{obj}}{d\mathbf{X}} = \frac{\partial f_{obj}}{\partial \mathbf{X}} + \Psi^T \frac{\partial R}{\partial \mathbf{X}} \quad (24)$$

$$\text{where } \Psi = -\left[\frac{\partial R}{\partial \mathbf{Y}}\right]^{-1} \left[\frac{\partial f_{obj}}{\partial \mathbf{Y}}\right]^T. \quad (25)$$

5. Solver Software

In order to solve this optimal control problem with proposed methodology, we first implement the quadrotor ODE system in OpenMDAO [17] as shown in Fig. 7. OpenMDAO is a handy tool in building systems of ordinary differential equations and differential algebraic equations, and is also a widely-used open-source python framework for multi-disciplinary design, analysis, and optimization which is also great at solving nonlinear programming problems. Recently it has also been used for simultaneous design and control optimization of mechanical systems [20]. In addition, we install the Dymos [41] library on the basis of OpenMDAO to solve for multi-phase trajectory optimization problem with aforementioned set of methods. Moreover, Dymos implements single unified derivative equation (UDE) approach proposed by Hwang and Martins [42] to evaluate derivatives. This outperforms sole adjoint method by using a pair of direct and adjoint methods to solve for derivatives and generalizing them in an UDE. Another N2 diagram showing the structure of phase 24 of one solution trajectory is illustrated in Fig. 8. The ODE system and trajectory optimizer code can be found on the author's github page <https://github.com/p5cao/quadTrajOpt>.

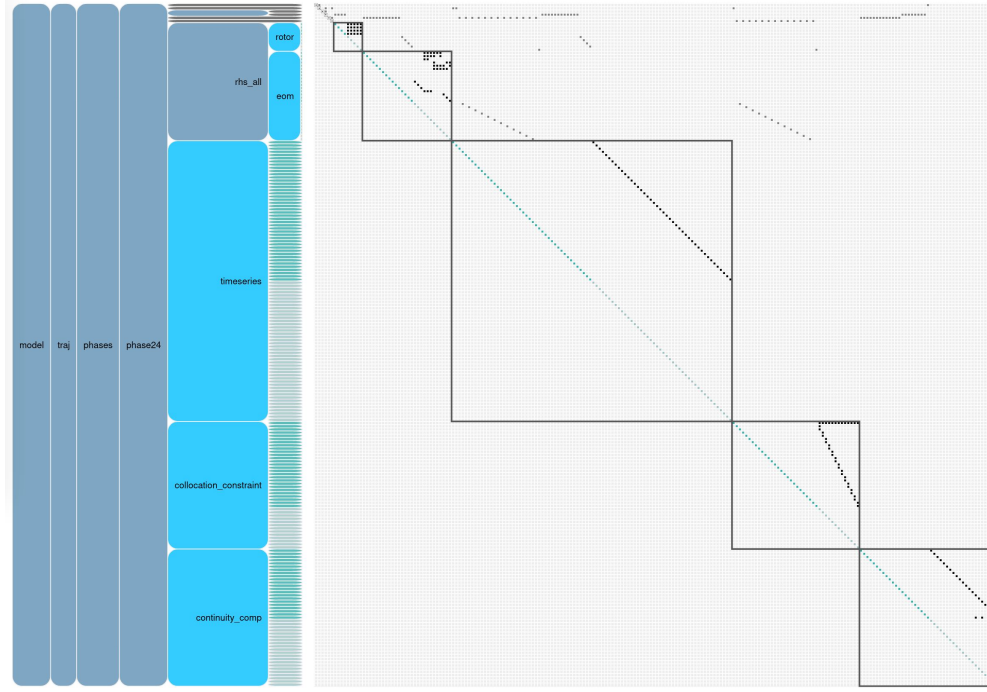


Figure 8 N^2 diagram of structure of the 24th phase's timeseries and collocation constraints of a solution trajectory.

D. Collision Check and Trajectory Modification

Once a solution trajectory is obtained our OpenMDAO program, we will need to guarantee it does not collide with the walls or obstacles. Also, any of the points in this trajectories should not exceed the indoor space boundary. The reason we need to revisit this step is that we do not add collision-free constraints to this multi-phase trajectory problem. Although the path planning algorithm does guarantee collision-free paths, the optimized trajectory needs to be checked and modified if necessary.

We first obtain the occupancy map of our indoor space as shown in Fig. 9, and create the state space based on this occupancy map. Then we run a `fixCollision()` algorithm to sequentially recompute the segments with invalid states by adding intermediate waypoints and attempting to find minimal-time or minimal energy sub-trajectory between these waypoints. This `fixCollision()` function also recompute the adjacent segments by varying their initial or final conditions. We iterate through every segment with invalid states until all the states on the trajectory become valid. After finishing the trajectory modifications, we recompute the mechanical energy consumption and time consumption of the modified trajectory and compare it with that the original trajectory result. We found that the variations are usually less than 5 % of the original objective function values.

IV. Results and Simulations

In this section, we present the results for three mission scenarios and compare the effectiveness of our mission-oriented optimal control solutions with those from other methods. At first, We show the optimized result of a quadrotor to fly from origin to the alcove to display its ability of computing the rotor RPM profile (i.e. rotary speed profile of each rotor). Next, we simulate the scenario where the UAV needs to travel across each of the waypoints obtained by the coverage path planning algorithm. Last but not least, we execute the room visiting paths and compare the time consumption between proposed methods and results of other trajectory generation methodology.

A. Quadrotor Parameters and Simulation Settings

The model of the quadrotor UAV we use in this case study is an ModalAI M500 drone. It has a propeller diameter of 6 inches (0.1524m), an arm distance (body center to propeller center) of 0.225m, and gross take-off weight without payload to be 0.69kg. In addition, we manage to measure its moments of inertia around its three body-fixed frame axes

Collision Free Trajectory on Occupancy Map

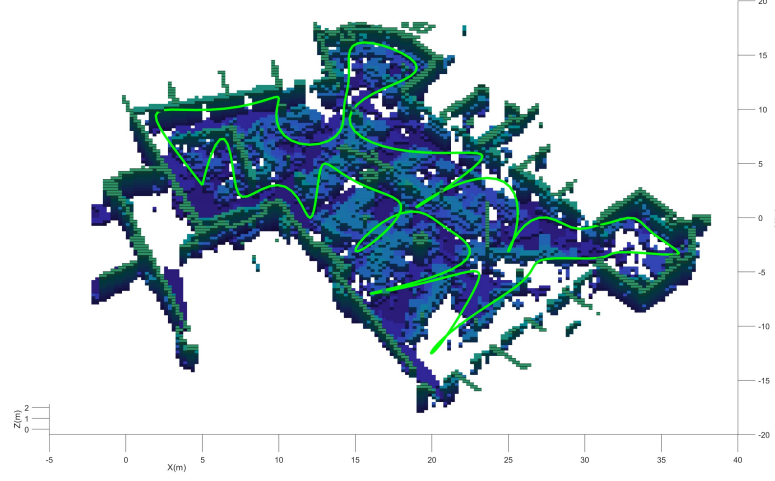


Figure 9 Illustration of the collision check function.

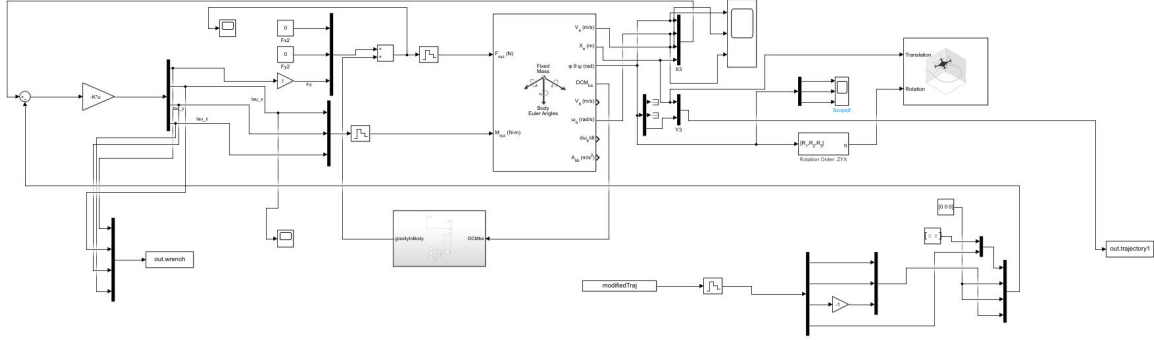


Figure 10 Simulink setup to simulate various trajectories for quadrotor UAVs.

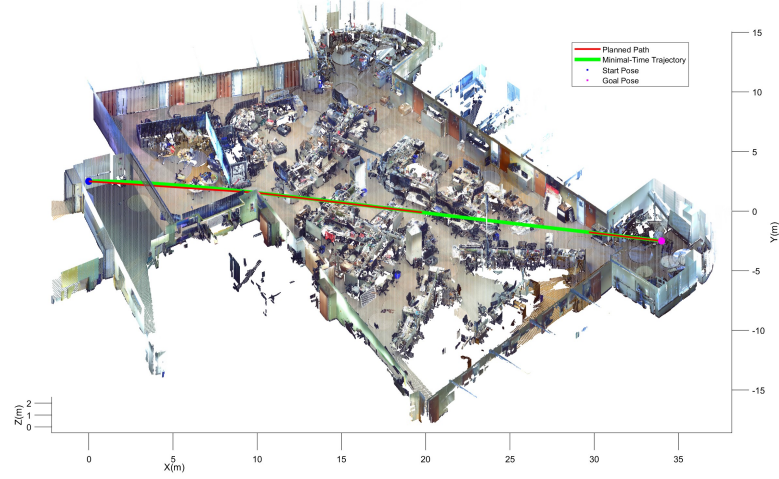
to be $I_x = 4.69 \times 10^{-2} \text{ kg} \cdot \text{m}^{-2}$, $I_y = 3.58 \times 10^{-2} \text{ kg} \cdot \text{m}^{-2}$, $I_z = 6.73 \times 10^{-2} \text{ kg} \cdot \text{m}^{-2}$, respectively. And the moment of inertia of each rotor is found to be $I_r = 3.357 \times 10^{-5} \text{ kg} \cdot \text{m}^{-2}$. In addition, we assume the air density to be uniformly $\rho = 1.225 \text{ kg/m}^3$, and gravitational acceleration to be $g = 9.80 \text{ m} \cdot \text{s}^{-2}$. With these parameters, we are able to set up the rigid body dynamics in the simulink program in Fig. 10, and have the optimizer to compute the UAV trajectory accordingly.

In order to validate and compare various quadrotor trajectories in a virtual setting, we set up the simulation environment in Simulink with the help of its UAV toolbox [43]. This setup helps execute the UAV missions with our control solutions in various virtual 3D maps while computing power consumption and realistic UAV paths to compare with other trajectories.

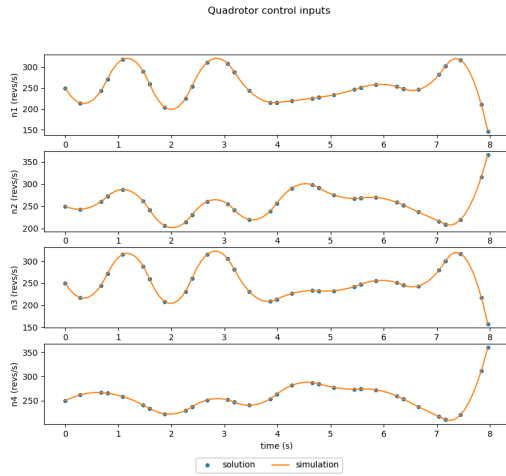
B. Origin-to-Alcove Path

The first solution trajectory of the quadrotor flying from origin to the alcove is illustrated in Fig. 11a. The maximum air speed is set to be 5 m/s , and this mission takes the UAV 7.97 seconds to fly from start position $([0.0, 0.0, 2.0])$ to the goal position $([34.0, -5.0, 2.0])$ in the alcove with the minimal time trajectory. In comparison, the minimum-snap trajectory generation algorithm [44] renders a flight time of 8.43 seconds, and a linear-quadratic-regulator based path

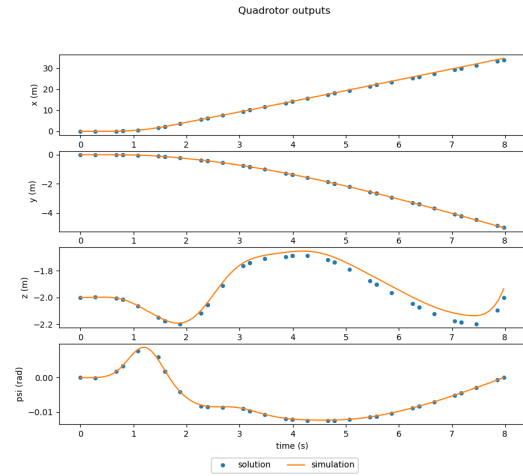
Origin-to-Alcove Minimal-Time Trajectory



(a) Minimal-time trajectory from origin to alcove.



(b) The rotor speeds profile of the trajectory.



(c) Quadrotor output of the trajectory.

Figure 11 Origin-to-alcove minimal-time trajectory.

follower will take 8.53 seconds to arrive at the goal position. For this flight mission, the proposed optimizer decreases the flight time by 5.45% from the minimum-snap trajectory. In addition, the `fixCollision()` function detects no collision on this trajectory.

C. Coverage Path – Open Space Search Mission

The search mission with coverage path of the lab space case study is based on the 90% cells coverage path from Fig. 3. The scenario of this mission is to search for victims within the space or cells covered by the path. In this mission, we set the objective to be mechanical or rotor energy consumption. In order to make the trajectory optimizer converge faster, we smooth the path in Fig. 3 to get a set of 30 waypoints. Thus, our mission-oriented trajectory optimizer partition the mission into 30 phases and manages to find a trajectory with associated control inputs through out the path. The initial solution is shown in Fig. 11c with invalid state points. Thus `fixCollision()` function is run to modify the path and eliminate the invalid states. The original invalid trajectory has a total mechanical energy consumption of 12,392 J,

Initial and Collision-Free Optimized Search Trajectory

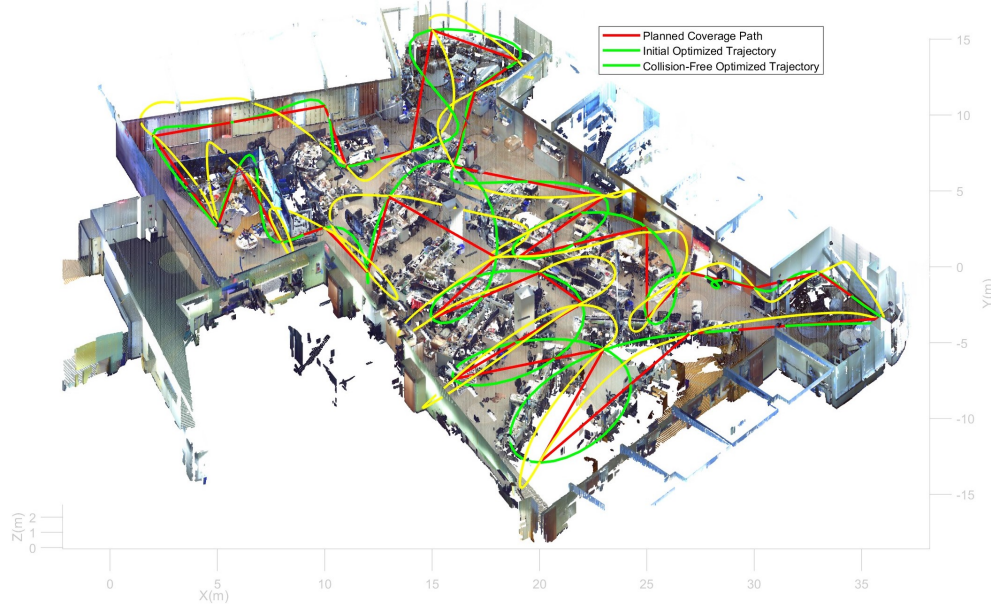


Figure 12 Initial and collision-free open space search trajectory obtained by proposed optimizer.

and the energy consumption increases to 13,074 J after running `fixCollision()`. For this flight mission, the proposed optimizer decreases the mechanical energy consumption by 3.00% from the minimum-snap trajectory. The illustration of the invalid and modified trajectory is shown in Fig. 12.

We compare the total flight time, mission energy consumption, trajectory length, and computational time of three trajectory solvers in Table 1. Although the effectiveness of the proposed approach is validated, its computational time is considerably longer than the other two, and the energy saving is not significant compared to the minimum snap path generator. The future goal of this optimizer can be to make the software more light-weight and eliminate the need to apply `fixCollision()` algorithm to save computational time.

Table 1 Comparison of proposed optimizer, Minimum snap and linear quadratic path tracker performance

	t_f (s)	$E(t_f)$ (J)	Trajectory length (m)	Computational time (s)
Proposed optimizer	77.40	13,074	183.74	55 (optimizer+fixCollision())
Minimum snap trajectory generator	76.30	13,479	188.32	41
Linear-quadratic path tracker	92.35	15,967	203.34	real-time

D. Multi-Room Search Path – Rooms Visiting Mission

The last mission trajectory interests the authors is the task where the quadrotor needs to visit 3 rooms connected to the open lab space to confirm the presence of victims in any of the three rooms. The objective of this mission is time consumption since saving time is paramount in search-and-rescue tasks and it is desirable to be aware of numbers and locations of victims within a shortest possible time. In this mission simulation we remove all door obstacles in the free configuration space such that the quadrotor is able to enter rooms through the doors. The UAV is required to start from $[0.0, 0.0, 2.0]$ and visit room 1 at $[8.0, 12.0, 2.0]$, room 3 at $[31.0, -1.25, 2.0]$, and at last arrive at room 5 ($[13.0, 10.0, 2.0]$).

The trajectory generated by proposed optimizer is shown in Fig. 13. Both the length and time consumption of the

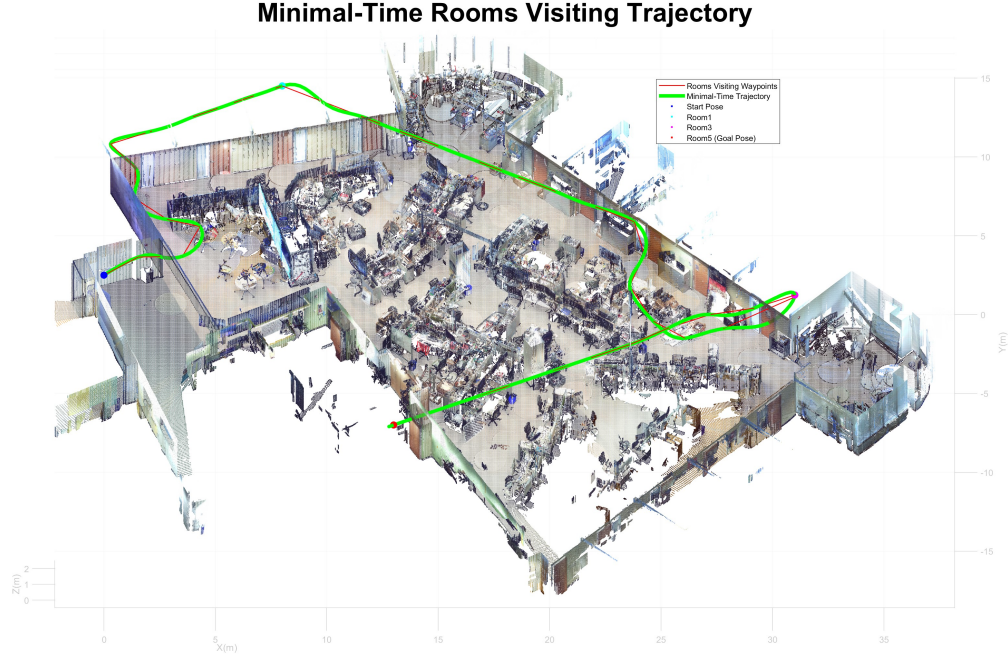


Figure 13 Minimal-time trajectory in rooms visiting mission.

proposed minimal-time trajectory and standard minimum-snap output do not deviate significantly from each other. With $V_{max} = 3m/s$, the optimizer-rendered trajectory has a total length of $68.82m$ and time consumption of $25.22s$, and minimum-snap trajectory has a total length of $67.34m$ and time consumption of $27.03s$. For this flight mission, the proposed optimizer decreases the flight time by 6.69% . In addition, the `fixCollision()` function detects no collision on this trajectory.

V. Conclusions

This paper proposes a novel approach to optimize trajectories with respect to mission specific objective functions. The optimal control problem is solved using nonlinear programming approach which is widely used for solving MDO problems. The equivalence of optimal control and MDO is discussed where the control input, objective functions, and trajectory output of the entire mission can be viewed as design variables to obtain an optimum for a quantitative performance. As expected, the MDO approach is able to minimize time or energy consumption of pre-designed missions to greater extent than do the common control approaches but with higher computational time.

The up-to-date progress only allow for minimizing mechanical energy and time consumption throughout the flight mission, but more types of objective functions can will be explored in the future work. For example, one can minimize the cumulative errors between flight trajectory and waypoints to improve path tracking, or average magnitude of UAV body rotation rates to better stabilize the camera pose for aerial imaging.

This trajectory optimization methodology is planned as stage 1 of a simultaneous design and control optimization for quadrotor UAVs serving search and rescue purposes. Future work will involve the formulation of a large-scale design optimization problem to optimize both the design parameters and mission trajectories for SAR-deployable quadrotors. In addition, the authors would want to improve the current algorithm to lower its computational cost. Last but not least, both simulation and experimental validation of the proposed trajectory optimizer are to be performed to evaluate the future results and be compared with peers' solutions.

VI. Acknowledgements

This publication is based on work supported by the US Army Corps of Engineers under research Cooperative Agreement W912HZ-17-2-0024 and under the auspices of the U.S. Department of Energy by Lawrence Livermore

National Laboratory under Contract DE-AC52-07NA27344 and by the LLNL-LDRD Program under Project No. 20-SI-005. The authors thank all collaborators at the Qualcomm Institute and the Contextual Robotics Institute of UC San Diego, as well as those at Flow Control & Coordinated Robotics Labs (FCCR) and Large-Scale Design Optimization Lab (LSDO) of UC San Diego to ideas, suggestions and comments. Opinions, findings, and conclusions from this study are those of the authors and do not necessarily reflect the opinions of the research sponsors.

References

- [1] Waharte, S., and Trigoni, N., "Supporting search and rescue operations with UAVs," *2010 International Conference on Emerging Security Technologies*, IEEE, 2010, pp. 142–147.
- [2] Murphy, R. R., Pratt, K. S., and Burke, J. L., "Crew roles and operational protocols for rotary-wing micro-UAVs in close urban environments," *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, 2008, pp. 73–80.
- [3] Silvagni, M., Tonoli, A., Zenerino, E., and Chiaberge, M., "Multipurpose UAV for search and rescue operations in mountain avalanche events," *Geomatics, Natural Hazards and Risk*, Vol. 8, No. 1, 2017, pp. 18–33.
- [4] Qi, J., Song, D., Shang, H., Wang, N., Hua, C., Wu, C., Qi, X., and Han, J., "Search and rescue rotary-wing uav and its application to the lushan ms 7.0 earthquake," *Journal of Field Robotics*, Vol. 33, No. 3, 2016, pp. 290–321.
- [5] Tomic, T., Schmid, K., Lutz, P., Domel, A., Kassecker, M., Mair, E., Grixia, I. L., Ruess, F., Suppa, M., and Burschka, D., "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE robotics & automation magazine*, Vol. 19, No. 3, 2012, pp. 46–56.
- [6] Kulkarni, S., Chaphekar, V., Chowdhury, M. M. U., Erden, F., and Guvenc, I., "Uav aided search and rescue operation using reinforcement learning," *2020 SoutheastCon*, Vol. 2, IEEE, 2020, pp. 1–8.
- [7] Tian, Y., Liu, K., Ok, K., Tran, L., Allen, D., Roy, N., and How, J. P., "Search and rescue under the forest canopy using multiple UAVs," *The International Journal of Robotics Research*, Vol. 39, No. 10-11, 2020, pp. 1201–1221.
- [8] Vanegas, F., Campbell, D., Eich, M., and Gonzalez, F., "UAV based target finding and tracking in GPS-denied and cluttered environments," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 2307–2313.
- [9] Jung, S., Jo, Y., and Kim, Y.-J., "Flight time estimation for continuous surveillance missions using a multirotor UAV," *Energies*, Vol. 12, No. 5, 2019, p. 867.
- [10] Villa, D. K., Brandao, A. S., and Sarcinelli-Filho, M., "A survey on load transportation using multirotor UAVs," *Journal of Intelligent & Robotic Systems*, Vol. 98, No. 2, 2020, pp. 267–296.
- [11] Andersen, H. L., "Path planning for search and rescue mission using multicopters," Master's thesis, Institutt for teknisk kybernetikk, 2014.
- [12] Champagne, L., Carl, R. G., and Hill, R., "Agent models II: Search theory, agent-based simulation, and U-boats in the Bay of Biscay," *Proceedings of the 35th conference on Winter simulation: driving innovation*, Citeseer, 2003, pp. 991–998.
- [13] Cabreira, T. M., Ferreira, P. R., Di Franco, C., and Buttazzo, G. C., "Grid-based coverage path planning with minimum energy over irregular-shaped areas with UAVs," *2019 international conference on unmanned aircraft systems (ICUAS)*, IEEE, 2019, pp. 758–767.
- [14] Cabreira, T. M., Di Franco, C., Ferreira, P. R., and Buttazzo, G. C., "Energy-aware spiral coverage path planning for uav photogrammetric applications," *IEEE Robotics and automation letters*, Vol. 3, No. 4, 2018, pp. 3662–3668.
- [15] Hayat, S., Yanmaz, E., Brown, T. X., and Bettstetter, C., "Multi-objective UAV path planning for search and rescue," *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2017, pp. 5569–5574.
- [16] Falck, R. D., and Gray, J. S., "Optimal control within the context of multidisciplinary design, analysis, and optimization," *AIAA Scitech 2019 Forum*, 2019, p. 0976.
- [17] Gray, J. S., Hwang, J. T., Martins, J. R., Moore, K. T., and Naylor, B. A., "OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization," *Structural and Multidisciplinary Optimization*, Vol. 59, No. 4, 2019, pp. 1075–1104.
- [18] Hwang, J. T., Lee, D. Y., Cutler, J. W., and Martins, J. R., "Large-scale multidisciplinary optimization of a small satellite's design and operation," *Journal of Spacecraft and Rockets*, Vol. 51, No. 5, 2014, pp. 1648–1663.

- [19] Kao, J., Hwang, J., Martins, J. R., Gray, J. S., and Moore, K. T., "A modular adjoint approach to aircraft mission analysis and optimization," *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2015, p. 0136.
- [20] Yan, J., Li, N., Luo, T., Tolley, M. T., and Hwang, J. T., "Optimal control and design of an underactuated ball-pitching robotic arm using large-scale multidisciplinary optimization," *AIAA Aviation 2019 Forum*, 2019, p. 3450.
- [21] Strawson, J., Cao, P., Tran, D., Bewley, T., and Kuester, F., "Monocoque Multirotor Airframe Design with Rotor Orientations Optimized for Direct 6-DoF UAV Flight Control," *AIAA AVIATION 2021 FORUM*, 2021, p. 2431.
- [22] Strawson, J., Cao, P., Bewley, T., and Kuester, F., "Rotor orientation optimization for direct 6 degree of freedom control of multirotors," *2021 IEEE Aerospace Conference (50100)*, IEEE, 2021, pp. 1–12.
- [23] Cao, P., Strawson, J., Bewley, T., and Kuester, F., "Decoupled translational and rotational flight control designs of canted-rotor hexacopters," *AIAA Scitech 2021 Forum*, 2021, p. 1058.
- [24] Martins, J. R., and Lambe, A. B., "Multidisciplinary design optimization: a survey of architectures," *AIAA journal*, Vol. 51, No. 9, 2013, pp. 2049–2075.
- [25] Noreen, I., Khan, A., Habib, Z., et al., "Optimal path planning using RRT* based approaches: a survey and future directions," *Int. J. Adv. Comput. Sci. Appl.*, Vol. 7, No. 11, 2016, pp. 97–107.
- [26] Karaman, S., and Frazzoli, E., "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, Vol. 30, No. 7, 2011, pp. 846–894.
- [27] Pitt, D. M., *Rotor dynamic inflow derivatives and time constants from various inflow models*, Washington University in St. Louis, 1980.
- [28] Ruh, M. L., and Hwang, J. T., "Robust modeling and optimal design of rotors using blade element momentum theory," *AIAA AVIATION 2021 FORUM*, 2021, p. 2598.
- [29] MacNeill, R., and Verstraete, D., "Blade element momentum theory extended to model low Reynolds number propeller performance," *The Aeronautical Journal*, Vol. 121, No. 1240, 2017, pp. 835–857.
- [30] Gur, O., and Rosen, A., "Propeller performance at low advance ratio," *Journal of aircraft*, Vol. 42, No. 2, 2005, pp. 435–441.
- [31] Zuo, Z., "Trajectory tracking control design with command-filtered compensation for a quadrotor," *IET control theory & applications*, Vol. 4, No. 11, 2010, pp. 2343–2355.
- [32] Davoudi, B., and Duraisamy, K., "A hybrid blade element momentum model for flight simulation of rotary wing unmanned aerial vehicles," *AIAA Aviation 2019 Forum*, 2019, p. 2823.
- [33] Glauert, H., "Airplane propellers," *Aerodynamic theory*, Springer, 1935, pp. 169–360.
- [34] Kim, K. C., "Analytical Calculations of Helicopter Torque Coefficient (CQ) and Thrust Coefficient (CT) Values for the Helicopter Performance (HELPE) Model." Tech. rep., ARMY RESEARCH LAB ABERDEEN PROVING GROUND MD, 1999.
- [35] Davoudi, B., Taheri, E., Duraisamy, K., Jayaraman, B., and Kolmanovsky, I., "Quad-rotor flight simulation in realistic atmospheric conditions," *AIAA Journal*, Vol. 58, No. 5, 2020, pp. 1992–2004.
- [36] Sabatino, F., "Quadrotor control: modeling, nonlinear control design, and simulation," , 2015.
- [37] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM review*, Vol. 47, No. 1, 2005, pp. 99–131.
- [38] De Boor, C., Höllig, K., and Sabin, M., "High accuracy geometric Hermite interpolation," *Computer Aided Geometric Design*, Vol. 4, No. 4, 1987, pp. 269–278.
- [39] Ross, I. M., and Fahroo, F., "Legendre pseudospectral approximations of optimal control problems," *New trends in nonlinear dynamics and control and their applications*, Springer, 2003, pp. 327–342.
- [40] Martins, J. R., and Hwang, J. T., "Review and unification of methods for computing derivatives of multidisciplinary computational models," *AIAA journal*, Vol. 51, No. 11, 2013, pp. 2582–2599.
- [41] Falck, R., Gray, J. S., Ponnappalli, K., and Wright, T., "dymos: A Python package for optimal control of multidisciplinary systems," *Journal of Open Source Software*, Vol. 6, No. 59, 2021, p. 2809.

- [42] Hwang, J. T., and Martins, J. R., “A computational architecture for coupling heterogeneous numerical models and computing coupled derivatives,” *ACM Transactions on Mathematical Software (TOMS)*, Vol. 44, No. 4, 2018, pp. 1–39.
- [43] Mathworks, “UAV toolbox,” , 2021. URL <https://www.mathworks.com/products/uav.html>.
- [44] Mellinger, D., and Kumar, V., “Minimum snap trajectory generation and control for quadrotors,” *2011 IEEE international conference on robotics and automation*, IEEE, 2011, pp. 2520–2525.