Navigating the Urban Jungle: A Study on UAV Trajectory Optimization in Cities

Yingqi Shen, Emma Yan
syq@umich.edu,eyan@umich.edu

December 14, 2023

Abstract

The purpose of this project is to study the UAV parcel delivery problem in which the trajectory is mainly constrained by a way point and an optional infeasible region. Based on previous work on the quadcopter UAV model and the developed OpenMDAO Gray et al. (2019) optimization framework, we consider the continuous system under different mission and environment settings. This report presents the derivation of the equations of motion for the model, problem formulation, implementation process, simulation results, and the ability of OpenMDAO to handle complex trajectory optimization problems.

1 Background and Motivation

In recent years, rapid urbanization has led to a significant increase in the need for efficient and timely deliveries within cities. This growing demand faces the challenge of congested city streets, which often result in delays and inefficiencies in ground transportation. The limitations of these ground-based systems have become increasingly apparent, prompting a search for alternative delivery methods that can overcome these urban constraints. Unmanned Aerial Vehicles (UAVs), commonly known as drones, have emerged as a promising solution, offering the potential to revolutionize urban logistics by utilizing the skies to enhance delivery efficiency.

However, integrating UAVs into the complex urban environment poses its own set of challenges. Navigating the maze of high-rise buildings, dealing with varied terrain, and adhering to strict regulatory frameworks require optimized trajectory planning for drones. Noor et al. (2018) This study focuses on exploring UAV trajectory optimization in urban settings, aiming to address the technological, regulatory, and practical challenges involved. By optimizing drone flight paths, we can not only improve efficiency but also pave the way for a future where UAVs play a crucial role in the evolving landscape of urban delivery services.

2 Problem Formulation

2.1 UAV six degree of freedom model

In recent years, the technological revolution in UAVs has happened. The UAVs are mainly categorized based on their types of wings and rotors, such as fixed-wing, single-rotor, and hybrid. The quadcopter, as a quad helicopter or as a multi-rotor helicopter with four rotors, has features that make it superior to other types of UAVs in maneuverability including unconstrained motion, vertical take-off, and landing. Considering the specific circumstances of our proposed problem of delivering parcels, the quadcopter is chosen as the UAV model to study and simulate for the rest of the study. The dynamic model of the quadcopter and its derived equations of motion are studied and applied based on Saif & Eminoğlu (2022).

Before defining the mathematical models, the most important mission is to define frames and reference coordinates so that the location, structure, and movement of the vehicle can be studied in detail. While trajectory and flight dynamics are considered, the earth frame (E-frame) and the body frame (B-frame) are the two references used in the derivation to determine the location and the position or direction of the quadcopter respectively.

The quadcopter is an aircraft with four rotating wings that are symmetrical and equally spaced from the center of the mass. The rotors are all connected to propellers which rotate clockwise (right and left) and counter-clockwise (front and back). The dynamics of the system are described using Euler angles. Let the linear position in the E-frame be presented by X, Y, Z. Let the rotation of the quadcopter around these three axes be denoted by the angle of rotation (ϕ), the angle of the pitch (θ), and the angle of yaw (ψ) separately. Now the linear position and the attitude of quadcopter in the E-frame can be expressed by $\xi_E = [x, y, z]$ and $\eta_E = [\phi, \theta, \psi]$. In the same way, the linear velocities and the angular rate velocities in the B-frame can be represented as $\nu_B = [u, v, w]$ and $\omega = [p, q, r]$.

The rotation matrix that transits between coordinate frames to be able to transform variables. The transformation between the E-frame and the B-frame about the inertial axes is presented below:

$$R_{DCM} = \begin{bmatrix} c\theta c\psi & c\psi s\theta s\phi - s\psi c\phi & s\phi s\psi + c\phi c\psi s\theta \\ s\psi c\theta & c\psi c\phi + s\psi s\theta s\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\phi c\theta \end{bmatrix} \tag{1}$$

where c and s refer to cosine and sine respectively.

To obtain the equations of motion for the quadcopter model, a study of the dynamics of the system needs to be performed, especially on the forces and the moments exerted on the vehicle. When the rotating propeller on the quadcopter works, there is an upward force produced and is proportional

to the square of the angular velocity of each motor. Thus the total thrust applied to the quadcopter is the sum of all the upward forces. This can be expressed in the equation:

$$T_t = k_t(\omega_{M1}^2 + \omega_{M2}^2 + \omega_{M3}^2 + \omega_{M4}^2) \tag{2}$$

where k_t is a thrust coefficient. In the mean time, the rotating propellers also produce a torque that is opposite to the upward thrust forces. Let k_m be the moment constant, and the force produced in the three axes can be given by:

$$M_i = k_m \omega_{Mi}^2 \tag{3}$$

$$\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} (f_4 - f_2)l \\ (f_3 - f_1)l \\ \sum_{i=1}^{4} M_i \end{bmatrix} \tag{4}$$

where the four forces (f_1, f_2, f_3, f_4) are generated by the change in the velocity of the four motors, and thus a vector containing control inputs can be determined on the basis of the square of the angular velocity of the propellers. The control inputs are represented by:

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} k_t(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \\ k_t(\omega_4^2 - \omega_2^2) \\ k_t(\omega_3^2 - \omega_1^2) \\ k_t(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \tag{5}$$

The total influential force that affects the quadcopter is the sum of the force of gravity, the drag force, and the thrust force. Assume that the quadcopter flies with a relatively low velocity and thus the drag force is ignored in this case. The total forces can be computed and presented in a vector:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} -mgsin\theta \\ mgsin\phi cos\theta \\ mgcos\phi cos\theta + F_{motors} \end{bmatrix} \tag{6}$$

The derivation of the equations of motion starts with Newton's second law which describes the relation between mass and acceleration:

$$\begin{aligned} F_E &= m(\frac{d\nu}{dt}) = m(\frac{d\xi_E^2}{dt}) \\ &= m \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} \end{aligned} \tag{7}$$

Now the equation with both sides multiplied by the transformation matrix can be converted to:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \frac{1}{m}(R_{DCM})^{-1} \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} \tag{8}$$

Now that the equations of motion has been derived, the rotational response of the body needs to be derived based on the relationship between the inertia matrix in E-frame and the angular moment and the Coriolis theorem. The assumptions are made that the friction is neglected in all movements except yaw movement, and the Euler angles are small enough to be ignored. The total momentum vector is given by

$$\begin{bmatrix} M_x \\ M_y \\ Mz \end{bmatrix} = \begin{bmatrix} I_{xx} & -I_{xy} & I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \times \begin{bmatrix} I_{xx} & -I_{xy} & I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{9}$$

where (p, q, r) are considered as the body angular rates, which is assumed to be equal to the first-order derivative of the Euler angles (ϕ, θ, ψ). Based on the previous assumptions, a dynamic model for the quadcopter can be obtained as:

$$\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \frac{l}{Ixx}U_2 \\ \frac{l}{Iyy}U_3 \\ \frac{l}{Izz}U_4 \end{bmatrix} \tag{10}$$

2.2 Design Variables

Based on the equations of motion we derived in the previous part, the state variables of the vehicle are listed in the Table 1.

Variable	Description
x	The linear location on the x-axis
$\dot{x}$	The linear velocity on the x-axis
y	The linear location on the y-axis
$\dot{y}$	The linear velocity on the y-axis
z	The linear location on the z-axis
$\dot{z}$	The linear velocity on the z-axis
ϕ	Roll Angle
$\dot{\phi}$	Roll Angular Velocity
θ	Pitch Angle
$\dot{\theta}$	Pitch Angular Velocity
ψ	Yaw Angle
$\dot{\psi}$	Yaw Angular Velocity

Table 1: State Variables

Accordingly, there are control variables that we use to control the trajectory of the vehicle. The control variables are also listed in the Table 2.

Control Variable	Description
ω_1	The angular velocity of Motor 1
ω_2	The angular velocity of Motor 2
ω_3	The angular velocity of Motor 3
ω_4	The angular velocity of Motor 4

Table 2: Control Variables

2.3 Constraint Function

The optimal control problem can be limited by path constraints and state constraints at specific time points. According to Betts (2010), a set of constraints related to this problem is defined.

2.3.1 Initial and Final State Constraint

At the beginning and the end of this trajectory, the vehicle is expected to be at a completely static phase with different linear locations on the map. Thus, the initial and final state constraints are defined in Table 3 as

State	Initial Constraints	Final Constraints
x	$x_0 = x_i$	$x_T = x_f$
y	$y_0 = y_i$	$y_T = y_f$
z	$z_0 = z_i$	$z_T = z_f$
$\dot{x}$	$\dot{x}_0 = 0$	$\dot{x}_T = 0$
$\dot{y}$	$\dot{y}_0 = 0$	$\dot{y}_T = 0$
$\dot{z}$	$\dot{z}_0 = 0$	$\dot{z}_T = 0$
ϕ	$\phi_0 = 0$	$\phi_T = 0$
θ	$\theta_0 = 0$	$\theta_T = 0$
ψ	$\psi_0 = 0$	$\psi_T = 0$
$\dot{\phi}$	$\dot{\phi}_0 = 0$	$\dot{\phi}_T = 0$
$\dot{\theta}$	$\dot{\theta}_0 = 0$	$\dot{\theta}_T = 0$
$\dot{\psi}$	$\dot{\psi}_0 = 0$	$\dot{\psi}_T = 0$
ω_1	$\omega_{1,0} = 0$	$\omega_{1,T} = 0$
ω_2	$\omega_{2,0} = 0$	$\omega_{2,T} = 0$
ω_3	$\omega_{3,0} = 0$	$\omega_{3,T} = 0$
ω_4	$\omega_{4,0} = 0$	$\omega_{4,T} = 0$

Table 3: Initial and Final State Constraints

where the initial and final location of the vehicle are denoted by (x_0, y_0, z_0) and (x_T, y_T, z_T) respectively. The desired locations with determined values for initial state and final state are denoted by (x_i, y_i, z_i) and (x_f, y_f, z_f).

2.3.2 Way Point Constraint

The quadcopter is expected to take off at the starting point, deliver the parcel to a waypoint with a complete stop, and finally to the destination. This intuitively separates the flight trajectory into two different phases and thus a set of state constraints with respect to the waypoint needs to be defined in Table 4.

Way Point Constraints	
$x = x_{wp}$	$\dot{x} = 0$
$y = y_{wp}$	$\dot{y} = 0$
$z = z_{wp}$	$\dot{z} = 0$
$\phi = 0$	$\dot{\phi} = 0$
$\theta = 0$	$\dot{\theta} = 0$
$\psi = 0$	$\dot{\psi} = 0$

Table 4: Way Point Constraints

where (x_{wp}, y_{wp}, z_{wp}) is the determined way point coordinate.

2.3.3 Obstacle Constraint

In the problem set, we define an obstacle point in the 3D map, create a semi-sphere from this point, and command the vehicle to avoid this obstacle at a safe distance. This leads to an infeasible semi-spherical region in the map where the vehicle is forbidden to travel through. To address this problem, an obstacle constraint was made in the way of

$$\sqrt{(x - x_{obs})^2 + (y - y_{obs})^2 + (z - z_{obs})^2} - r_{obs} \geq d_{safe} \tag{11}$$

where the obstacle point is defined as $(x_{obs}, y_{obs}, z_{obs})$, the radius of the semi-sphere is r_{obs} and the safe distance is d_{safe}.

2.3.4 Boundary Constraint

As the generated map has limited space, there is a set of boundary constraints at all times defined as

$$x_{lower} \leq x \leq x_{upper} \tag{12}$$

$$y_{lower} \leq y \leq y_{upper} \tag{13}$$

$$z_{lower} \leq z \leq z_{upper} \tag{14}$$

where the map's lower and upper boundary are defined as $(x_{lower}, y_{lower}, z_{lower})$ and $(x_{upper}, y_{upper}, z_{upper})$.

2.3.5 Maneuverability Constraint

Considering the maneuverability of the vehicle, the constraints are given by

$$-\pi \leq \phi \leq \pi \tag{15}$$

$$-\pi \leq \theta \leq \pi \tag{16}$$

$$-\pi \leq \psi \leq \pi \tag{17}$$

2.4 Objective Function

The main purpose of the optimization is to find out the minimum total time needed for delivery so that the objective function can be represented by

$$\min t_T = t_{phase0} + t_{phase1} \tag{18}$$

which is at all times subject to

$$0 \leq t \leq T \tag{19}$$

2.5 Optimization Problem Formulation

In the following optimization problem, we will use physical parameters from Table 5 Saif & Eminoğlu (2022) as our parameter for the quad-copter.

Parameter	Definition	Value	Unit
m	Mass of the quadcopter	2.0	kg
l	Arm length	0.3	m
I_{xx}	Moment of inertia about x-axis	0.02	kg m^2
I_{yy}	Moment of inertia about y-axis	0.02	kg m^2
I_{zz}	Moment of inertia about z-axis	0.03	kg m^2
g	Gravitational force	9.81	m/s^2
d	Drag coefficient	7.5×10^{-7}	Nms^2
k_t	Thrust coefficient	1.7×10^{-5}	N s

Table 5: Physical Parameters of the Quadcopter Model

We set the boundary constraint defined in subsubsection 2.3.4 of the 3D space witch specific value as shown in Table 6. We will use this boundary constraint case for all cases of optimization problems.

Phase	Coordinate	Lower Limit	Upper Limit
Phase 0	z	0 m	100 m
	x	0 m	50 m
	y	0 m	50 m
Phase 1	z	0 m	100 m
	x	0 m	50 m
	y	0 m	50 m

Table 6: Boundary Constraint of the 3D Space

By incorporating all constraint functions and objective function, we can formulate our optimization problem as Equation 20.

$$
\begin{aligned}
&\text{minimize } t_T \\
&\text{with respect to} \\
&\qquad \omega_1, \omega_2, \omega_3, \omega_4 \\
&\text{subject to} \\
&\qquad 0 \leq (\omega_1, \omega_2, \omega_3, \omega_4) \leq 1200\,\text{rad/s} \\
&\qquad -\pi \leq \phi, \theta, \psi \leq \pi \\
&\qquad 0 \leq z \leq 100\,\text{m} \\
&\qquad 0 \leq x, y \leq 50\,\text{m} \\
&\qquad x_0 = x_i, \quad x_T = x_f \\
&\qquad y_0 = y_i, \quad y_T = y_f \\
&\qquad z_0 = z_i, \quad z_T = z_f \\
&\qquad \dot{x}_0 = \dot{y}_0 = \dot{z}_0 = 0, \quad \dot{x}_T = \dot{y}_T = \dot{z}_T = 0 \\
&\qquad \phi_0 = \theta_0 = \psi_0 = \dot{\phi}_0 = \dot{\theta}_0 = \dot{\psi}_0 = 0, \\
&\qquad \phi_T = \theta_T = \psi_T = \dot{\phi}_T = \dot{\theta}_T = \dot{\psi}_T = 0 \\
&\qquad \omega_{1,0} = \omega_{2,0} = \omega_{3,0} = \omega_{4,0} = 0, \\
&\qquad \omega_{1,T} = \omega_{2,T} = \omega_{3,T} = \omega_{4,T} = 0
\end{aligned}
\tag{20}
$$

2.5.1 Case 1: Simple Trajectory

The first case we want to optimize is a trajectory with one way point and one destination point as shown in Table 7.

Point	**x (m)**	**y (m)**	**z (m)**
Start	0	0	0
Waypoint	8	10	0
End	30	15	0

Table 7: Coordinates of Points

This is a simple trajectory where only state constraints, control constraints, and boundary constraints are applied to the vehicle. The objective is to minimize the total time of the entire flight.

2.5.2 Case 2: Trajectory With Obstacle

In case 2, we want to solve a problem of trajectory with one way point, one obstacle between the starting point and the way point, and one destination point as shown in Table 7. The obstacle point is defined using the parameters in Table 8. This problem is similar to case 1 except it has an

additional obstacle constraint. We expect this additional constraint to increase the complexity of the optimization and compare its performance to evaluate the optimizer.

Parameter	Value	Units
x_{obs}	5	m
y_{obs}	5	m
z_{obs}	0	m
r_{obs}	3	m

Table 8: Obstacle Parameters

2.6 Problem Classification

The proposed problem can be categorized based on two aspects: problem formulation and objective and constraint function characteristics.

1) *Problem Formulation:* The design variables are continuous and the objective is single. It is a constrained optimization problem as well.

2) *Objective and Constraint Function Characteristics:* The functions are continuous, mostly linear, unimodal, mostly convex, and deterministic.

The analysis of the problem statement along with the requirement to optimize the trajectory leads to our choice of the optimizer and its application, which is explained in detail in the next section.

2.7 Optimizer Selection

In our optimization problem, we will select the optimizer with the best performance from pyOptSparse Wu et al. (2020). The evaluation criteria are straightforward. We will choose the optimizer that converges to the optimum in the fewest iterations of calculation. We will compare the performance between two gradient-based optimizers, namely *IPOPT* and *SLSQP*, and one gradient-free optimizer, which is *NSGA2*.

Although gradient-based optimizers are generally more efficient than gradient-free optimizers Martins & Ning (2022), we will still compare their performance to see if we can make any new discoveries. We will execute the optimization problem and record the *Duration time* for each iteration. Subsequently, we can create a plot of *Duration time vs. Iteration* to observe the convergence rate and the optimum for each algorithm.

3 Result and Discussion

The Figure 1, displays the N2 diagram of our optimization problem. On the left-hand side of the N2 diagram, we have listed all subsystems of the optimization problem. The *motion_ode*

subsystem represents the equation of motion of the quad-copter, and the $Traj$ subsystem represents the simulation of the quad-copter's trajectory. In the center of the N2 diagram is the sparsity matrix, which represents the connections between the different subsystems and individual elements. The right-hand side shows the solvers we used an their responsible for converging the set of equations for each element. In this optimization problem, we primarily used linear solvers, and these linear solvers are linked to phase 0 and phase 1. Phase 0 and phase 1 represent the two phases of the trajectory: from the starting point to the first waypoint and from the first waypoint to the endpoint.

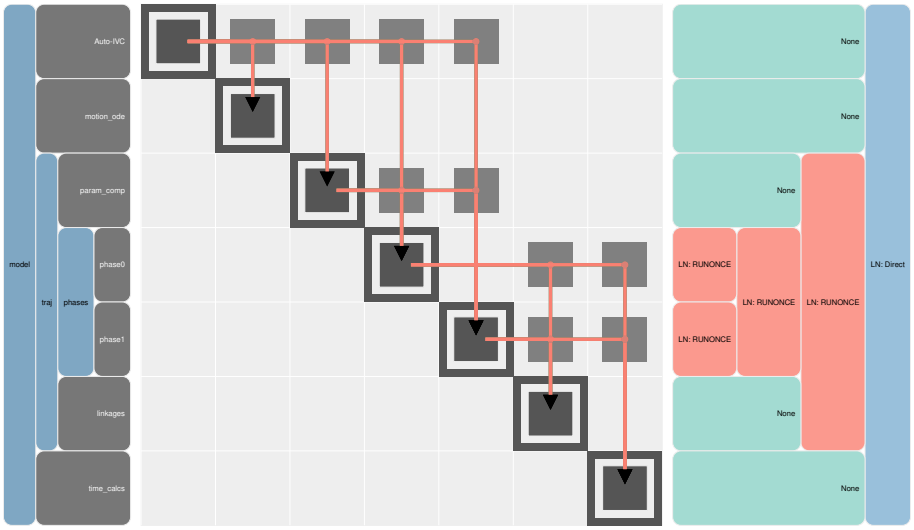

Figure 1: N2 Diagram of the Optimization Problem

3.1 Optimizer Selection

In subsection 2.7, we mentioned our intention to compare the performance of *IPOPT*, *SLSQP*, and *NSGA2*. However, we were only able to evaluate the performance of *IPOPT* and *NSGA2* because the optimizer encountered a singular matrix error during optimization. Consequently, we will focus our comparison and selection solely on *IPOPT* and *NSGA2* as potential optimizers.

In Figure 2, we conducted a comprehensive performance analysis of two optimizers, specifically *IPOPT*, in the context of a specific optimization problem outlined in subsubsection 3.2.2.

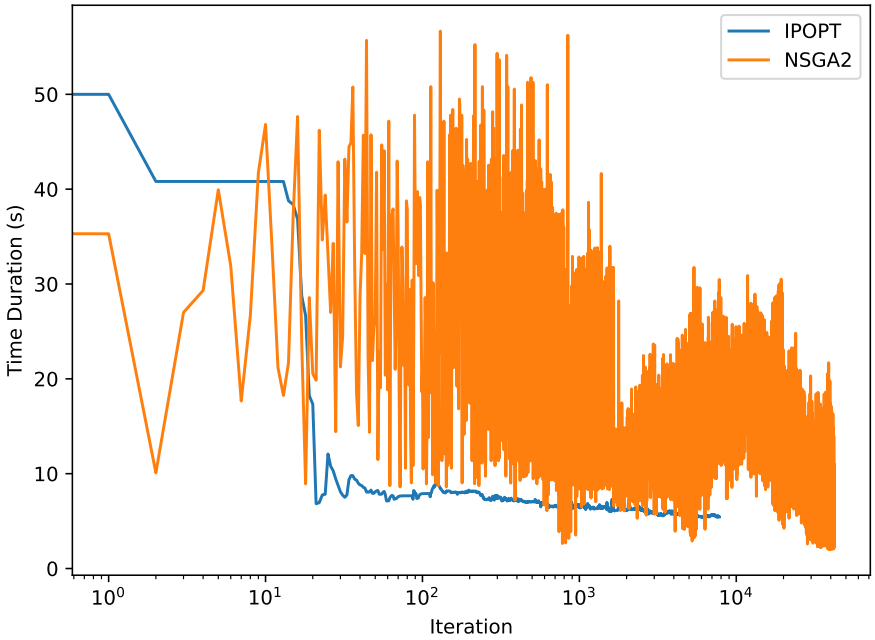

Figure 2: Performance Comparison: *Time Duration vs. Iteration*

IPOPT exhibited significantly higher efficiency compared to *NSGA2*, particularly in terms of convergence rate. Notably, *IPOPT* achieved substantial optimization results within approximately 10^2 iterations, while *NSGA2* required about an order of magnitude more iterations (around 10^3 iterations) to achieve similar outcomes. This discrepancy in iteration counts underscores *IPOPT*'s superior efficiency in navigating the solution space and rapidly converging to optimal solutions.

Furthermore, although *NSGA2* occasionally displayed shorter time durations for specific optimization results, its performance exhibited inconsistency. The plot reveals significant fluctuations and abrupt changes in *NSGA2*'s time duration between iterations. This variability may indicate *NSGA2*'s sensitivity to particular problem configurations or its inherent stochastic nature, potentially resulting in less predictable performance across diverse problem sets.

In contrast, *IPOPT* demonstrated a more stable and predictable time duration trajectory, aligning with the expected behavior of a robust and efficient nonlinear optimizer. This stability is of paramount importance in scenarios where predictability and reliability are critical, especially in the context of large-scale or complex optimization problems.

Based on these observations, we have chosen *IPOPT* as the preferred optimizer for Case 2. Its superior convergence rate, consistent performance, and time efficiency make it better suited to our specific requirements, providing a reliable and efficient solution for our optimization challenges.

3.2 Results and Interpretations

In this section, we present the results and interpretations of our analysis. The following figures provide visual representations of the trajectory and key variables associated with our study. The trajectory comprises two distinct phases: the gray section represents phase 0, which extends from the starting point to the first waypoint, while the red section represents phase 1, covering the segment from the first waypoint to the endpoint. Waypoints serve as connection points between these trajectory phases.

3.2.1 Case 1: Simple Trajectory

In this case, we examine a straightforward trajectory with no obstacle points, allowing for greater flexibility in optimizing the flight path. The figures below showcase the trajectory, state variables, and control variables over time.

The trajectory in 3D space, along with its 2D projection, is illustrated in Figure 3 and Figure 4.

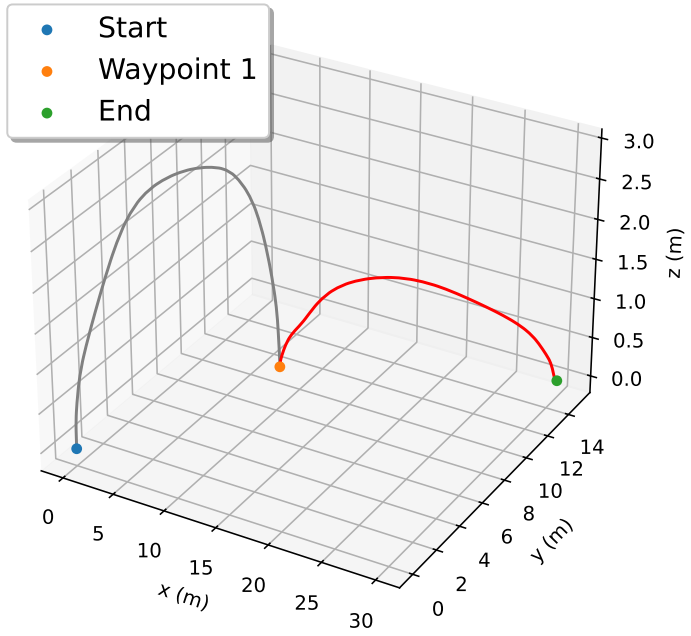

Figure 3: 3D Trajectory

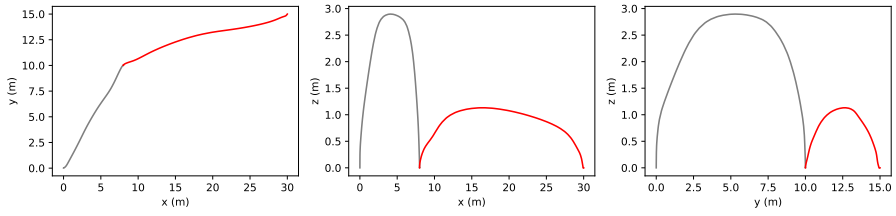

Figure 4: 2D Trajectory

Figures Figure 5 and Figure 6 present the variations in state variables (x, y, z) and linear velocities ($\dot{x}$, $\dot{y}$, $\dot{z}$) over time.

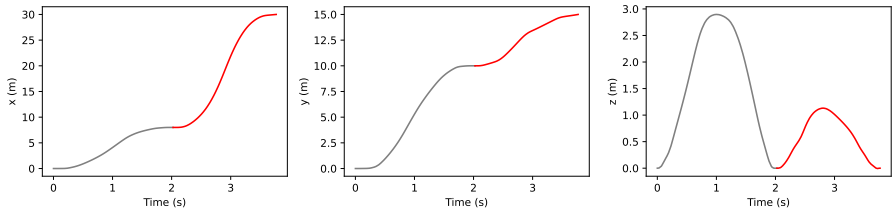

Figure 5: Linear Location (x, y, z) of Quad-Copter vs. Time

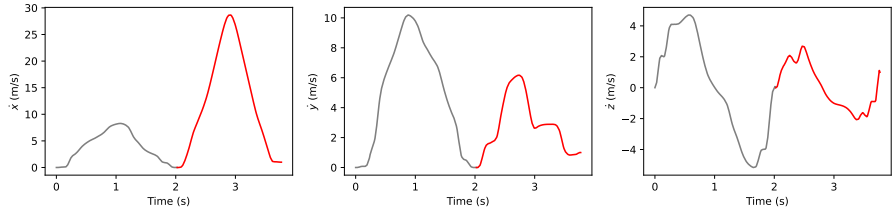

Figure 6: Linear Velocity ($\dot{x}$, $\dot{y}$, $\dot{z}$) of Quad-Copter vs. Time

Figure Figure 5 shows that the quad-copter started at (0,0,0) m, reached the first waypoint at (8,10,0) m in approximately 2 seconds, and arrived at the endpoint at (30,15,0) m in about 1.5 seconds, totaling approximately 3.5 seconds. Additionally, in Figure 6, we observe that the initial velocity of the quad-copter was (0,0,0) m/s. As it approached the first waypoint, its velocity decreased to zero, consistent with the constraint established in subsection 2.3, where we aimed to ensure complete stoppage at the first waypoint.

Figures Figure 7 and Figure 8 depict the changes in Euler angles (ϕ, θ, ψ) and angular velocities ($\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$) over time.

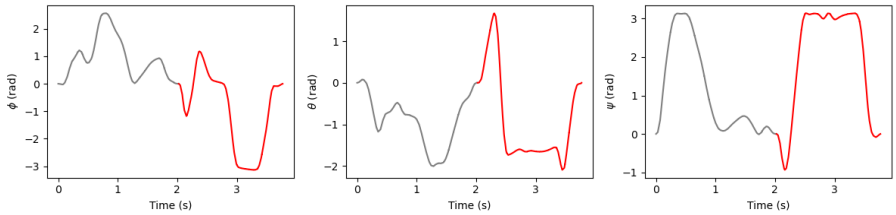

Figure 7: Euler Angles (ϕ, θ, ψ) of Quad-Copter vs. Time

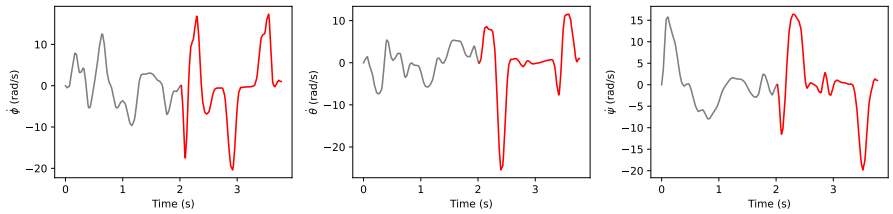

Figure 8: Euler Angular Velocity ($\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$) of Quad-Copter vs. Time

In Figure 7, the quad-copter's Euler angles start at (0,0,0) rad and return to this position at the first waypoint due to an equality constraint specifying that Euler angles must be (0,0,0) rad at this point. The time taken for phase 0 matches the time for the quad-copter to travel from (0,0,0) m to the first waypoint at (8,10,0) m since the changes in Euler angles are correlated with the quad-copter's positional changes. After reaching the first waypoint, the quad-copter takes off again with Euler angles set to (0,0,0) rad. In phase 1, the quad-copter undergoes further Euler angle changes, eventually returning to (0,0,0) rad at the endpoint, as dictated by the boundary constraint.

Figure Figure 9 displays the control inputs ω_1, ω_2, ω_3, ω_4 over time. In subsection 2.3, we imposed constraints on the angular velocity of motor outputs, limiting it to a range of 0 to 1200 rad/s. At the start, first waypoint, and endpoint, the angular velocity of all four rotors is zero due to the equality constraint ensuring complete stoppage at these points. The control inputs are not very stable, since we did not set up the lower bound and upper bound of the angular acceleration for four rotors.

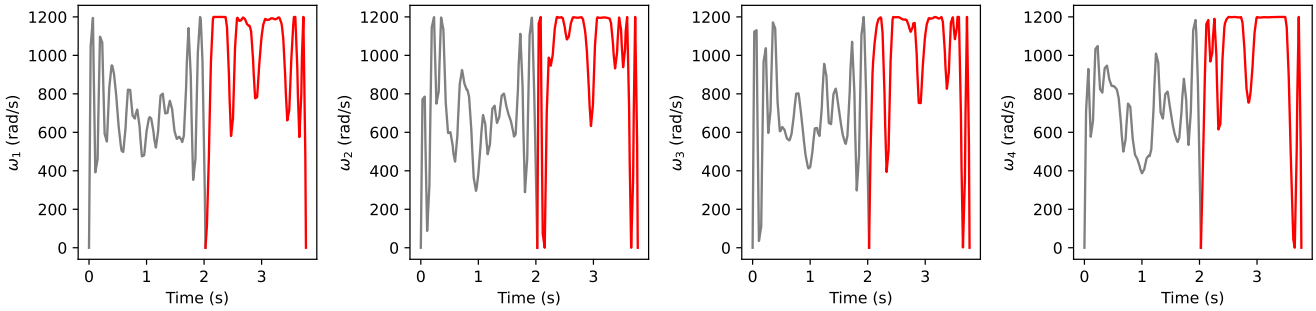

Figure 9: Control Inputs (ω_1, ω_2, ω_3, ω_4) of Quad-Copter vs. Time

3.2.2 Case 2: Trajectory With Obstacle

In Case 2, we define an obstacle point on the map and command the quad-copter to avoid this point at a safe distance of 2 meters. Recall that the obstacle point is defined in subsubsection 2.3.3. We apply the obstacle constraint defined in Equation 11, resulting in the simulation of a new trajectory.

The 2D and 3D trajectories are shown in Figure 11 and Figure 10. These two figures clearly depict the flight path in which the quad-copter avoids the obstacle. This also demonstrates that the

optimizer effectively handles this problem with an additional obstacle constraint.

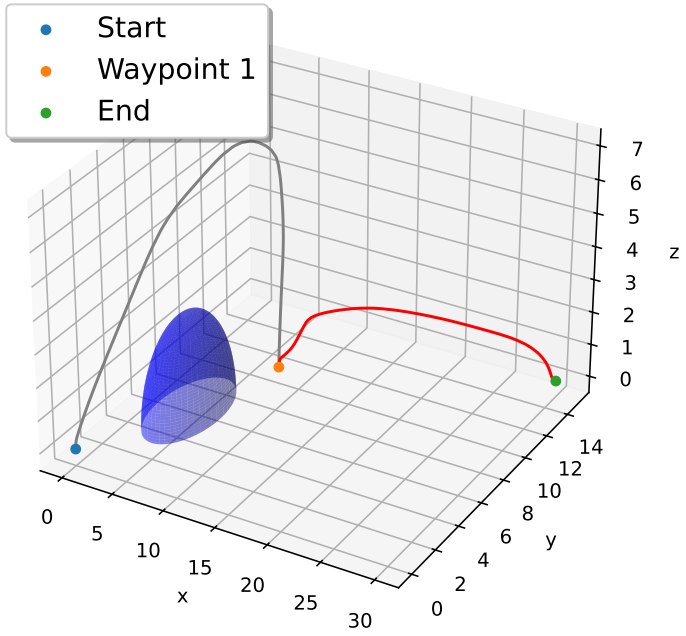

Figure 10: 3D Trajectory

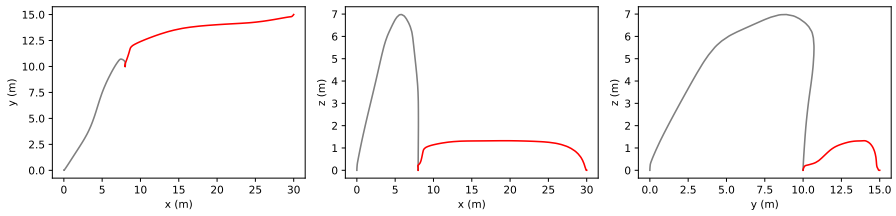

Figure 11: 2D Trajectory

Figure 12 shows the flight path of the quad-copter throughout the entire trajectory. It is evident that the vehicle successfully avoids the obstacle and maintains a safe distance from it, satisfying our most critical constraint. We can observe that the z-axis reaches a height of 7 meters in phase 0, while it only goes up to 3 meters in phase 0 in Case 1. To understand the detailed maneuver of the quad-copter, the figure depicting its linear velocity is presented in Figure 13. The quad-copter takes off and experiences a drastic increase in velocity along both the y-axis and z-axis, mainly due to the presence of the obstacle. Similarly, with the waypoint limit, the velocities in all directions drop to zero as it is commanded to come to a complete stop at the end of phase 0. In phase 1, as indicated by the red lines, the quad-copter maintains a relatively high speed along the x-axis and eventually drops to zero at the very last point.

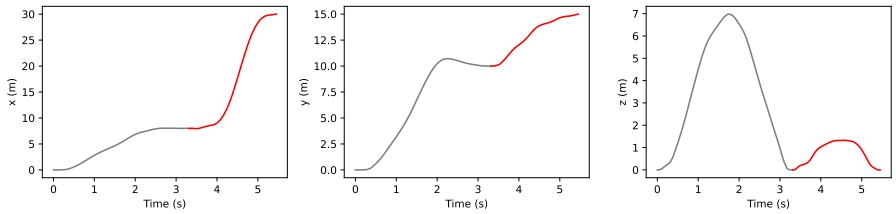

Figure 12: Linear Location (x, y, z) of Quad-copter vs. Time

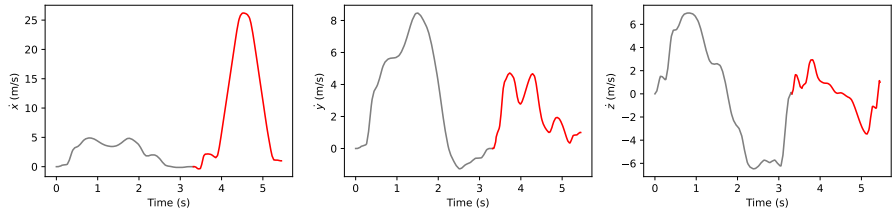

Figure 13: Linear Velocity ($\dot{x}$, $\dot{y}$, $\dot{z}$) of Quad-copter vs. Time

The position of the quad-copter can be understood using Figure 14 and Figure 15, which present the Euler angles and angular velocities, respectively. The Euler angles and angular velocities are all constrained to zero at the beginning and end of each phase due to state constraints.

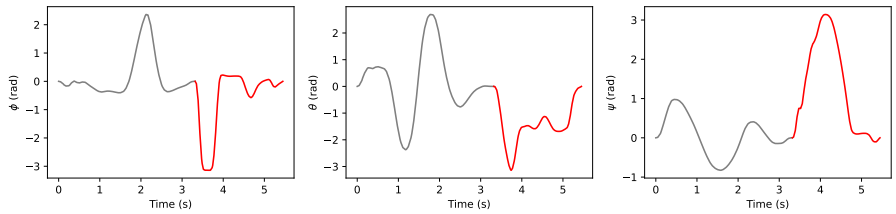

Figure 14: Euler Angles (ϕ, θ, ψ) of Quad-copter vs. Time

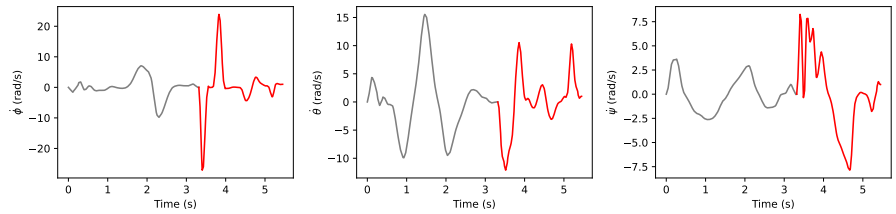

Figure 15: Euler Angular Velocity ($\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$) of Quad-copter vs. Time

The control inputs are also presented in Figure 16. It can be observed that the control inputs are not very stable, as analyzed in subsubsection 3.2.1. This instability is primarily due to the absence of bounds on the angular acceleration of the four rotors.

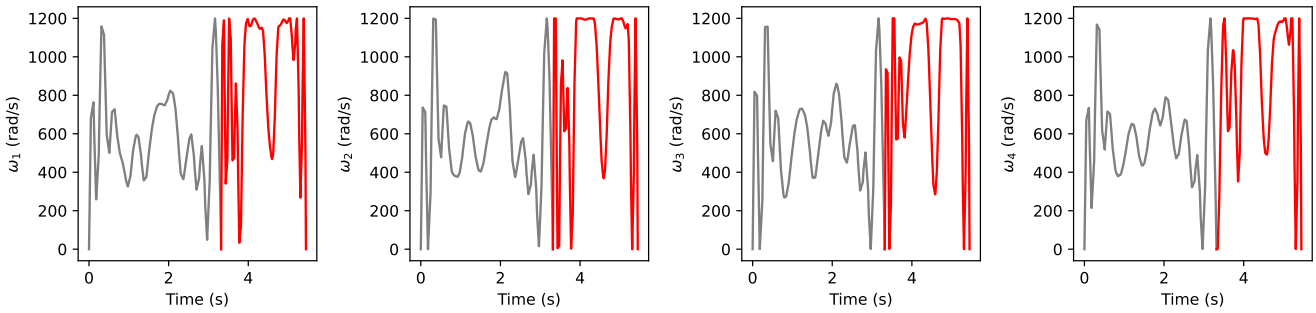

Figure 16: Control Inputs (ω_1, ω_2, ω_3, ω_4) of Quad-copter vs. Time

3.3 Case Comparison

In Figure 17, we have plotted the Time Duration vs. Iteration for both cases. From the figure, we can see that the convergence rate of both cases is similar for the first 110 iterations. However, after 110 iterations, the time duration curve for Case 2 did not decrease significantly, while the time duration was still decreasing for Case 1. After 210 iterations, we observed that the time duration of Case 2 was still decreasing at a slower rate, whereas the decrease in time duration for Case 1 was almost negligible.

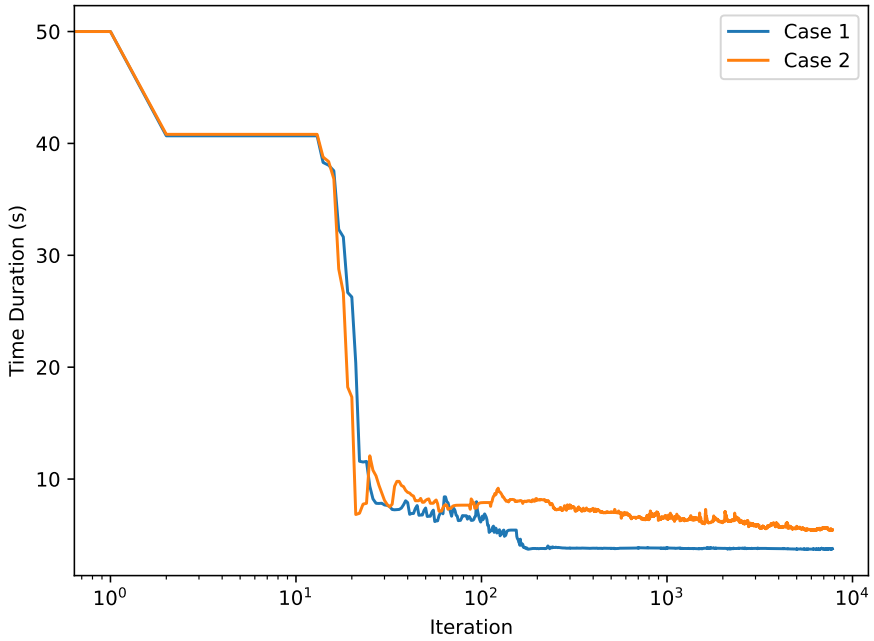

Figure 17: Time Duration vs. Iteration for Two Cases

In Table 9, we have extracted the optimum values for both cases. We can observe that Case 1 took approximately 3.77 seconds for the entire journey from the starting point to the end point, whereas Case 2 took about 5.44 seconds for the same journey. By comparing these results, we can

conclude that Case 2 took approximately 1.67 seconds longer than Case 1. The main reason for this difference is that during Case 2, the quad-copter had to avoid obstacles in its flight path, which required more time compared to Case 1, where there were no obstacles.

Case	Total Time Duration (s)
1	3.77060192
2	5.44869617

Table 9: Total Time Durations for Both Cases

4 Conclusion

In this project, we harnessed the power of the PyOptSparse library Wu et al. (2020) to tackle challenging trajectory optimization problems. Our investigation involved the evaluation of three distinct optimizers from the library: two gradient-based optimizers, *IPOPT* and *SLSQP*, and one gradient-free optimizer, *NSGA2*. Through comprehensive testing and comparison, we meticulously assessed their respective performances. Ultimately, our analysis revealed that *IPOPT* emerged as the optimal choice, exhibiting superior overall performance in addressing our trajectory optimization challenges.

Our project featured two trajectory optimization problems. The first was a straightforward problem, involving a single waypoint. The second, a modified version of the first, introduced an obstacle into the trajectory. In this complex scenario, we controlled the continuous angular velocity variables for four rotors (ω_1, ω_2, ω_3, ω_4) over time to minimize the total time required to traverse the trajectory from the initial point to the waypoint and then from the waypoint to the endpoint. The results of our optimization efforts were remarkable. We achieved a swift and efficient execution, with the quadcopter completing the entire trajectory in 3.77 seconds for the simple scenario and 5.44 seconds in the presence of an obstacle.

Despite the significant progress made in this project, we recognize numerous avenues for future exploration and improvement. Expanding the map size and increasing the obstacle count can enhance the realism of our simulations, better mirroring urban environments. Additionally, incorporating a more comprehensive energy consumption model during flight can provide a more realistic perspective. Lastly, diversifying our objectives to include minimizing power consumption during flight represents a promising direction for future research.

In conclusion, our utilization of PyOptSparse and the successful optimization of challenging trajectories underscore the potential for further advancements in this field. This project serves as a stepping stone toward more sophisticated simulations and optimizations in the realm of urban aerial navigation.

References

Betts, J. T. (2010). *4.1.1 dynamic constraints.* Society for Industrial and Applied Mathematics (SIAM). Retrieved from `https://app.knovel.com/hotlink/khtml/id:kt007T6NZ5/practical-methods-optimal/dynamic-constraints`

Gray, J. S., Hwang, J. T., Martins, J. R. R. A., Moore, K. T., & Naylor, B. A. (2019). OpenMDAO: An Open-Source Framework for Multidisciplinary Design, Analysis, and Optimization. *Structural and Multidisciplinary Optimization*, *59*, 1075-1104. doi: 10.1007/s00158-019-02211-z

Martins, J. R. R. A., & Ning, A. (2022). *Engineering design optimization.* Cambridge, UK: Cambridge University Press. Retrieved from `https://mdobook.github.io` doi: 10.1017/9781108980647

Noor, N. M., Abdullah, A., & Hashim, M. (2018, jun). Remote sensing uav/drones and its applications for urban areas: a review. *IOP Conference Series: Earth and Environmental Science*, *169*(1), 012003. Retrieved from `https://dx.doi.org/10.1088/1755-1315/169/1/012003` doi: 10.1088/1755-1315/169/1/012003

Saif, E., & Eminoğlu, (2022). Modelling of quad-rotor dynamics and hardware-in-the-loop simulation. *The Journal of Engineering*, *2022*(10), 937-950. Retrieved from `https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/tje2.12152` doi: https://doi.org/10.1049/tje2.12152

Wu, N., Kenway, G., Mader, C. A., Jasa, J., & Martins, J. R. R. A. (2020). pyoptsparse: A python framework for large-scale constrained nonlinear optimization of sparse systems. *Journal of Open Source Software*, *5*(54), 2564. doi: 10.21105/joss.02564