Project 101: Simulation and Guidance of Parafoils for Rocket Recovery

MECH5080M Team Project – Individual Report

Project 101: Implementing Active Recovery Control and Monitoring Systems for Rocket Landing

Author: Tyler Green, 201537563

Supervisor: Dr Jongrae Kim

Industrial Mentor:

Examiner: Marlene Mengoni

Date: 09/05/2025

SCHOOL OF MECHANICAL ENGINEERING

UNIVERSITY OF LEEDS

MECH5080M TEAM PROJECT 45 credits

TITLE OF PROJECT

Project 101: Simulation and Guidance of Parafoils for Rocket Recovery

PRESENTED BY

Tyler Green

OBJECTIVES OF PROJECT

1. 1. Conduct a comprehensive literature review of parafoil modelling and guidance methods.
2. Develop a model of the parafoil-payload system.
3. Conduct simulations to predict vehicle response.
4. Develop a method of aerodynamic coefficient identification.
5. Implement a guidance algorithm.
6. Validate system performance through comparison to real-world flight

IF THE PROJECT IS INDUSTRIALLY LINKED TICK THIS BOX AND PROVIDE DETAILS BELOW

COMPANY NAME AND ADDRESS:

INDUSTRIAL MENTOR:

THIS PROJECT REPORT PRESENTS OUR OWN WORK AND DOES NOT CONTAIN ANY UNACKNOWLEDGED WORK FROM ANY OTHER SOURCES.

SIGNED  DATE 9/05/2025

Table of Contents

Abstract

This project presents the development and validation of a simulation and guidance framework for a steerable parafoil recovery system designed for rocket payloads. A nonlinear six-degree-of-freedom (6-DoF) model was constructed to simulate the dynamic and aerodynamic behaviour of a parafoil-payload system. Aerodynamic coefficients were identified using a Two-Step Genetic Algorithm (TSGA), with improvements introduced to enhance convergence and efficiency. Achieving an error of 0.25m average error from the expected flight path. A multiphase path planning guidance algorithm, integrated with real-time wind estimation via Recursive Least Squares (RLS), was implemented to enable precise landings. The simulation was validated under variable wind conditions, demonstrating a landing error of less than 20m.

Chapter 1 Introduction

1.1 Introduction

1.1.1 Motivation

Leeds University Rocket Association (LURA) is a student-led rocketry association based at the University of Leeds that designs, builds, and flies sub-orbital rockets. They, along with commercial organisations, face the problem of recovering expensive rocket bodies safety down to earth and current recovery methods use a simple parachute where the eventual landing site is subject to the prevailing wind conditions and landscape. At the student scale, parafoil rocket recovery is a simpler, cheaper way to recover rockets compared to other actively controlled methods [1], allowing precise control over the rocket's downward trajectory and landing zone, maximising safety and reusability.

This group project, "Project Icarus," aims to develop and demonstrate this capability. My contribution lies in the development of a simulation model and the implementation of a path planning algorithm to test and verify parafoil guidance and control performance under realistic conditions. These tools will allow for the design and evaluation of control strategies before physical flight testing, significantly reducing development risk and cost.

More specifically, this report focuses on the development of a nonlinear 6-degree-of-freedom (6-DoF) model of the parafoil-payload system, the identification of aerodynamic coefficients via a Genetic Algorithm, and the implementation of a real-time multi-path planning strategy.

1.1.2 Background

Parafoils are aerodynamic decelerators that generate lift via a flexible canopy, typically inflated during descent. Through differential control of trailing-edge flaps, a parafoil can achieve turning (skid or roll steering) and flaring manoeuvres. These dynamic capabilities offer a degree of control akin to that of a fixed-wing glider, enabling landing at a specified location rather than a probable drift point of regular parachutes.

However, accurately predicting and controlling parafoil behaviour is non-trivial. Their light weight, flexible structure make them very susceptible to environmental factors like wind.

A Genetic algorithm (GA) is an optimization technique inspired by natural selection that evolves a population of candidate solutions toward better results through selection, crossover, and mutation [2]. A fitness function is required which informs how well an

individual is, a higher fitness, more likely is this individual to pass on its genes in the next generation of the algorithm. The randomness of this algorithm makes it resilient to local minima and excels at navigating complex fitness landscapes to achieve the global minima.

1.2 Aim

To develop a simulation of a guided parafoil system and implement a path planning algorithm that enables it to follow a predefined trajectory.

1.3 Objectives

1. Conduct a comprehensive literature review of parafoil modelling and guidance methods.
2. Develop a model of the parafoil-payload system.
3. Conduct simulations to predict vehicle response.
4. Develop a method of aerodynamic coefficient identification.
5. Implement a guidance algorithm.
6. Validate system performance through comparison to real-world flight data.

1.4 Report Structure

The report is structured as follows:

Chapter 1 – Introduces the project and background before outlining the Individual Aims and Objectives.

Chapter 2 - Covers the literature review of parafoil dynamics, system identification, wind estimation, and guidance algorithms.

Chapter 3 - Details the derivation and implementation of the model.

Chapter 4 - Discusses the methodology and results of aerodynamic coefficient calibration.

Chapter 5 - Outlines the chosen path planning strategy and its implementation.

Chapter 6 - Presents simulation results and their analysis.

Chapter 7 - Summarises the contributions made in this report and highlights areas for future work.

Chapter 2 Literature review

2.1 Parafoil modelling

A range of models have been developed to simulate parafoil dynamics, each varying in complexity and fidelity. Early work in the 1970s by Godrick introduced a 3-degree-of-freedom (3-DoF) model based on experimental data to describe the longitudinal motion of a parafoil [3]. Since then, both 3- and 4-DoF models have been widely used to support guidance, navigation, and control (GNC) applications [4], [5]. By modelling lower-order models, only fundamental aspects of motion are captured and are insufficient for accurately simulating the full dynamics and response of a parafoil system for detailed control purposes. To address these limitations, more comprehensive 6-degree-of-freedom (6-DoF) models have been extensively employed [6]. This typically represents the parafoil and payload as a single rigid body, tracking Euler angle orientation and three-dimensional position relative to the intended point of impact (IPI).

The apparent mass describes the additional inertial forces arising from a body's movement through a fluid medium [7]. Due to a parafoils low weight, parafoils are particularly sensitive to this. Whilst this improves accuracy, it shifts the moment of inertia outside the longitudinal plane of symmetry requiring this matrix to be computed at every time step, adding computation cost [8]. Tonglia at el. [9] compared models with and without apparent mass and observed noticeable differences in steady-state roll and pitch angles, particularly during manoeuvring. However, the model also excluded the moment generated by aerodynamic forces, which likely influenced these discrepancies. Despite these differences in orientation, positional trajectories remained largely consistent across models.

More sophisticated models extend to 8 and 9-DoF representations, incorporating relative motion between the canopy and payload [9], [10] [11] as well as structural dynamics such as suspension line elasticity and joint mechanics [12]. Toglia et al. [9] demonstrated that such effects introduced notable discrepancies between simpler and higher-fidelity models, particularly in dynamic conditions, largely attributed to the coupling between the payload dynamics and the parafoil. While these high-order models improve accuracy, their nonlinearity and computational cost make them unsuitable for real-time GNC.

A common approach to simplify the model is linearisation [13], [14]. By assuming near-

constant local air velocity around the parafoil, the system can be treated as quasi-static, allowing the use of a reduced state vector. Linearisation techniques such as the Observer/Kalman Filter Identification (OKID) method have been successfully applied to 8-DoF models, yielding adequate performance [15]. Pena et al. [16] employed a Genetic Algorithm (GA) to identify the parameter matrix for the time-invariant linear model of an aircraft's longitudinal dynamics using only 66 data samples. The GA converged within 22 minutes, achieving a mean squared error (MSE) of 4.39×10^{-4} m^2 compared to the original complex nonlinear model, demonstrating that linearisation can maintain the accuracy.

A major challenge in constructing physics models lies in the number of required aerodynamic coefficients. Forces such as the lift and drag, as well as the moments about the roll, pitch, and yaw axes, depend on variables like angle of attack, sideslip angle and vehicle orientation. Each is characterised by an additional coefficient, leading to approximately 14 distinct coefficients for some 6-DoF models [17]. Consequently, Ward et al. [18] laterally split the parafoil wing up into segments and only considered the lift and drag forces, reducing the number of coefficients down to 8. The lateral and rotational motion of the parafoil arose from the different orientation and displacements of each segment when considering the parafoil as a whole. Validation against flight data showed this approach could accurately predict core parafoil behaviour.

An alternative to physics-based modelling is to train black-box models directly on flight data. These models bypass the assumptions required by physical equations. Li et al. [19] for example, trained a specialised neural network for non-linear systems on simulated airdrop data. Results showed that black box model had a lower average error compared to a previous 6-DoF model previously developed. However, a thorough understanding of system cannot be attained from this method. In addition, as flight data is required, this defeats the purpose of using a simulation to estimate flight performance.

Despite extensive literature on general parafoil systems, studies specifically modelling a rocket as a payload remain limited. Due to their elongated shape, rockets typically exhibit a lower centre of pressure and centre of mass compared to the compact payloads usually typically considered. Consequently, this can induce significant oscillatory behaviour [20], consistent with observations from drop tests conducted by Aenishanslin et al. [21], who reported oscillations around 1.5 Hz in the suspended rocket payload beneath the parafoil making lower fidelity 6-DoF models less accurate.

2.2 System Identification

To ensure that physics-based models of parafoil systems accurately represent real-world behaviour, precise values describing the system are needed, most notably the aerodynamic coefficients. Analytical methods, such as lifting line theory [22] , provide initial estimates but are typically used as starting points for more refined techniques. Computational Fluid Dynamics (CFD) has been explored as a solution [23], though it is computationally intensive and still subject to inaccuracies due to the challenges of modelling real-world environmental uncertainties, especially due to the flexible nature of the parafoil canopy.

A widely used, empirical, alternative is system identification. This involves minimizing the discrepancy between simulated and measured flight data to estimate aerodynamic parameters. Barrown et al. [22] compared analytical (lifting line theory), CFD, and empirical methods for estimating the longitudinal coefficients C_d and C_L (drag and lift, respectively). The empirical method used the glide ratio (horizontal vs. vertical travel) to derive the coefficients. A minimum error of 16% between empirical and CFD estimates can largely be attributed to the difficulties associated with extracting accurate position data, particularly from video sources.

Ling L. et al [24] used GPS data to minimize the sum of squared residuals by computing the partial derivatives of each parameter via a system response method. This involved comparing nominal to perturbed trajectories to estimate the aerodynamic coefficients, but due to error accumulation, short intervals of translational motion (<10s) were only considered. To facilitate convergence, the inclusion of the moment coefficients was omitted and success was highly dependent on initial parameter guesses. Ward et al. [18] similarly used GPS data to find the both the force and moment coefficients of a simplified model, considering only the steady-state portions of the flight. The problem was framed as an optimisation problem of non-linear least squares and utilised the Levenberg-marquardt algorithm to minimise error between simulated and calculated lift, drag and turn rate. The use of the refined coefficients successfully improved simulation accuracy. The results were compared to independent Inertial Measurement Unit (IMU) sensor data and the error to be below 3 times the average IMU noise. Recursive Weighted Least Squares was applied in [13] and [14] to a simplified 6-DoF linear model. These studies showed that while optimized coefficients could predict core trajectories well, finer details such as oscillations and minor deviations were missed.

This was attributed the limits of the simplified model used or the failure to accurately measure and counter the wind from the drop test data.

Another widely used online mean square error (MSE) minimiser is the Kalman filter known for its exceptional reliance to noise. An extension, the adaptive Unscented Kalman filter was highlighted by Majeed et al. [25] to estimate aerodynamic coefficients of an aircraft in the presence of dynamic process noise. The UKF converged in 7.55s demonstrating its suitability for online parameter estimation.

Less attention has been given to parameter estimation in higher DoF models. Jaiswal et al [26] developed a 9-DoF model and used maximum likelihood estimation, a statistical based method, to estimate the required 12 aerodynamic coefficients. While convergence was achieved, the study did not report how closely the final simulation matched real flight data.

Genetic Algorithms (Gas) have also shown promise in this field. A Two-Step Genetic Algorithm (TSGA) in [27] showed very promising results identifying a nonlinear model of an aircraft, identifying 20 aerodynamic coefficients. This method involved first optimising each individual parameter before combining the optimised parameters and optimising the entire parameter set as a whole. However, for each iteration, the simulation must be run for every agent, making this method computationally expensive.

2.3 Wind Effects

Wind estimation is another crucial aspect for parameter estimation, guidance and simulation as often the parafoil velocity is similar magnitude to the wind, gravely effecting the parafoils motion. While pilot tubes offer the most accurate wind readings, they are often impractical due to weight and volume constraints. The vehicle's Course Over Ground (COG) velocity is often used to estimate wind by assuming constant relative air velocity and utilising the wind triangle. Ward et al. [18] extended this further and discovered that the wind estimation error whilst turning was proportional to the heading traversed during the manoeuvre, resulting in a minimum estimate error equal to the GPS error, assuming constant wind. The Recursive Least Squares (RLS) method was used to iteratively update the estimate and results showed that the reconstructed flight from the estimated wind followed the GPS measurement of the flight accurately.

This agrees with Luo et al. [28] who compared estimated winds at a range of altitudes to the RLS calculated estimates. The RLS estimate had a maximum error of 3.5 degrees and 0.75m/s. However, the wind is not constant, and guidance manoeuvres

means wind estimation cannot be done throughout the entire flight. This is why Gao et al. [29] also used RLS but added an atmospheric model of the wind to predict wind at lower altitudes in the remainder of the flight. Simulations were successful and a sensitivity study on the position of the flap deflection for optimum wind identification, 20% was found to be the best with 0.96% error. This approach is similar to Cristi et al. [30] which implemented a logarithmic model of the wind and showed that the corresponding guidance was more accurate.

Slow convergence is often an issue with the above methods. To address slow acquisition time, Yu et al. [31] used deep Q-learning to identify the wind vector from comparing flight data to simulated data from a dynamic model. This method was compared directly to calculating the wind using the wind triangle and the corresponding flight path error was reduced by 80-90%. In addition, the completed deep learning model had a computation time of two seconds making this a good candidate for online wind estimation. However, this method relies on an accurate physics model which is why Ling et al. [24] combined system identification with wind estimation so both variables could be estimated at the same time. This allows the system to mitigate errors such as partially deployed parafoil as system parameters are updated to mitigate this.

2.4 Guidance Methods

With parafoil behaviour modelled and wind estimation complete, guidance algorithms can now generate a desired path for the control algorithm to follow. Overall, there are two main methodologies, optimal guidance and multiphase path planning. Optimal guidance aims to minimise a cost function to produce the most efficient path. In contrast, multiphase path planning is typically simpler, consisting of discrete behavioural phases which are triggered based on criteria met during flight.

2.4.1 Multiphase Path Planning

The typical phases of Multipath Planning include:

1. Launch – stabilises from deployment and conducts wind estimates.
2. Homing – travels towards the Intended Point of Impact (IPI)
3. Energy management circle (EMC) – the parafoil repeats a similar glide pattern to reduce altitude whilst maintaining lateral position.
4. Final approach – aligns the vehicle against the wind before using a flaring manoeuvre to provide a soft landing.

The vehicle is commonly modelled as a 3 or 4-DoF particle model assuming constant horizontal and vertical velocity. These simplifications allow for time-based estimations to determine when key altitudes or positions are reached. Such estimations can then be used to update the geometry of the path or trigger transitions between flight phases.

The two fundamental geometries used are the Orbit Geometry [32] or the T-Approach [33]. The Orbit Geometry has fixed approach and exit angles for different phases of the flight. Consequently, the radius of the EMC circle is often the variable which is changed throughout flight as new predictions made [21]. On the other hand, for the T-Approach the distances between waypoints are the variable which is changed [33]. One advantage of the Orbiting approach is its failure mode – since it is continuously rotating, if an avionics failure occurs the parafoil will continue to rotate and travel with the wind. As the T-Approach can have linear paths, the parafoil could continue in that direction.

No matter which method is used, the final approach is critical. A large proportion of the landing error has been attributed to the last 100m of descent due to wind variability [34]. Slengers et al. [35] approached this problem by using optimal control to continuously update the geometry of the final turn, ensuring that the parafoil enters the final approach with the correct altitude and heading. This was achieved by modelling the final turn as a two-point boundary value problem. Simulations conducted had a circular error probability (CEP) of 55ft (16.7m) primarily due to downrange error. Later drop tests [34] validated this approach, estimating a CEP of 35m from 5 drop tests. From both the simulations and drop tests, it is apparent that downrange accuracy is the primary challenge. To tackle this, ward et al. [36] extended the T-Approach by introducing a virtual offset IPI. By dynamically switching between the actual IPI and the virtual one, the final approach path was effectively lengthened or shortened allowing the altitude to be tuned producing a CEP of 20.1m.

2.4.2 Optimal path planning

Looking at optimal guidance methods, Zhang et al. [37] used the guass pseudospectral to minimise path error and control effort. Whilst simulation results were promising, each one requires 11s to compute, posing challenges for deployment on embedded processors. In contrast, Leeman et al [5], reformatted the problem so sequential Convex Programming can be used. The iterative framework allows a feasible trajectory to be produced at each time step, with an average compute time of 1s, online guidance is feasible. Results from 600 Monte Carlo simulations demonstrated its superior accuracy compared a typical Multipath guidance method [32].

Li et al. [38] optimised the flight path by initially generating multiple unconnected path segments. These segments are then linked into a tree structure from which the shortest viable path is calculated. Compared to the compound optimisation random tree algorithm, the distance of the path was shorter for similar acceleration and velocity inputs making the proposed algorithm more effective. What's more, it considered threats by generating a threat model of the hills enabling collision avoidance.

Chapter 3 6-DoF model

Whilst larger, more complex models may be more accurate as indicated in the literature review, a six degree of freedom (6-DoF) model was chosen as it represents a good baseline and a balance between complexity and accuracy. Overall, there were 4 assumptions made to simulation:

1. Apparent mass is negligible.
2. Moments generated by the aerodynamic forces are negligible.
3. Centre of pressure (Cp) in parafoil is positioned directly above centre of mass (CoM) of payload. I.e. there is no horizontal distance between Cp and CoM.
4. Mass of parafoil negligible so CoM of system can assume to be CoM of payload.

3.1 Model derivation

The full derivation for this model can be found in Appendix A. Presented below is a summary of the core aerodynamics. Equation 3.1 and 3.2 outline translational and rotational dynamics of the system to get the resulting linear and angular accelerations:

$$\dot{v}_b = \frac{1}{m}\left(F_A + F_g - m(\omega \times v_b)\right) \tag{3.1}$$

$$\dot{\omega} = \boldsymbol{I}^{-1}\left(M_A - \omega \times (\boldsymbol{I}\omega)\right) \tag{3.2}$$

Where v_b is the body velocity, m is the mass of the system, F_A is the aerodynamic forces in the body frame, F_g is the gravitational force in the body frame, ω is the body angular velocity, $\boldsymbol{I}$ is the bodies moment of inertia and, M_A represents the aerodynamic moments. The aerodynamic forces and moments are defined in Equations 3.3 and 3.4 below:

$$F_A = -\boldsymbol{R}_w^{\mathcal{B}} \frac{1}{2}\rho \|v_a\|^2 S \begin{bmatrix} C_D \\ C_Y \\ C_L \end{bmatrix} \tag{3.3}$$

$$M_A = \frac{1}{2}\rho \|v_a\|^2 S \begin{bmatrix} C_l b \\ C_m \bar{c} \\ C_n b \end{bmatrix} \tag{3.4}$$

Where $\boldsymbol{R}_w^{\mathcal{B}}$ is the rotation from the wind frame to the body frame, ρ is the air density, v_a is the relative air velocity, S is the relative Surface area of the parafoil, b is the parafoils wingspan, $\bar{c}$ is the parafoils mean chord length. C_D, C_Y, C_L and C_l, C_m, C_n are summed aerodynamic effect relevant to each axis of motion.

3.2 Simulation implementation

The dynamics was written into a simulation Class in Python to allow future implementation into rocketPy, a python library used by LURA for rocket trajectory simulations [39]. The iterative 4th order Runge-Kutta approach was chosen to update the state vector with its derivatives as it has comparably low accumulation errors [40]. The full physics simulation can be found on GitHub [41].

Chapter 4 Coefficient calibration

As discussed in the literature review, accurately determining the aerodynamic coefficients is essential for realistic parafoil simulation. Traditional methods such as model linearisation and steady state response are highly specific to a given model and become time-consuming when adapting to new configurations. In contrast, Genetic Algorithms (GA) have demonstrated strong adaptability across different models and varying levels of complexity. For this reason, a Genetic Algorithm was selected in this project to estimate the aerodynamic coefficients. The methodology presented by Ulinowics, et al [27] will serve as a baseline for an adapted approach to be built from. The Two Step Genetic Algorithm (TSGA) consists of two steps; within the first step, individual parameters evolved independently before a regular GA in the second step optimises the parameters further together. Bounds are used ensure that the GA searches relevant realistic values.

4.1 TSGA Improvements

Three improvements were chosen to be the focus of this work:

- Implement elitism – It has been demonstrated that the global optimum is not guaranteed if this is not implemented [42]. This is where the best performing agents are carried across to the next generation without mutation or crossover.
- First step single parameter estimation may be hindering performance as coupling between parameters are not considered. Thus implementing subsets of similar coefficients, the algorithm could converge more efficiently.

4.2 Implementation

The implementation of this method was developed in Python using the powerful DEAP library [43]. This allowed multiple simulations to run in parallel, drastically reducing computation time. Within the TSGA implementation, the overall mutation rate was modelled as a linear decay as shown in equation:

$$P_{mut} = P_{base} + P_{delta} \times (1 - g_i/G) \quad (4.1)$$

Where P_{mut} is the probability of mutation, P_{base} is the minimum mutation probability, P_{bias} is the extra probability to mutate at the start of the evolution, g_i is the current generation and G is the total number of generations.

Once an individual was chosen to be mutated, the actual method for mutation was Simulated Binary Crossover as this allowed easy integration with parameter bounds. Selection was done using a tournament-based style approach. The fitness was initially set to the Root Means Squared Error (RMSE) of the simulation positional error however after initial testing, it was changed to the following equation 4.2:

$$fitness = \left(\sqrt{\frac{1}{N^*} \sum_{i=0}^{N^*} \sqrt{\left\| X_i - X_i^* \right\|^2}} \right) \times (1 + N - N^*) \quad (4.2)$$

Where the N is the expected number of values, $N^*$is the actual number of values, X_i is the i^{th} simulated values, X_i^* is the i^{th} ideal value. This still incorporated the RMSE as seen on the left of the equation, but it also allowed the simulation to finish early if errors such as unrealistic dynamics or motion were found to occur and factored this into the fitness.

4.3 Methodology

The following targeted genomes were used for the proposed method, aiming to split longitudinal, lateral and coefficients only effecting forces:

$$\begin{gathered} \left[C_{L_0},\ C_{L_\alpha},\ C_{L_{\delta,s}},\ C_{D_0},\ C_{D_\alpha},\ C_{D_{\delta,s}},\ C_{m,0},\ C_{m,\alpha}\alpha,\ C_{m,q}\right] \\ [C_{l,\beta},\ C_{l,p},\ C_{l,r},\ C_{l,\delta_a},\ C_{n,\beta},\ C_{n,p},\ C_{n,r},\ C_{n,\delta_a}, C_{Y,\beta}] \\ \left[C_{L_0},\ C_{L_\alpha},\ C_{D_0},\ C_{D_\alpha},\ C_{m,0}\right] \end{gathered} \quad (4.3)$$

Any coefficients not used in the GA process were set to the coefficients of Snowflake to ensure consistency. In reality, these could be seen to arise from Analytical methods or previous knowledge. These targeted genomes need to be combined together to form a population for the second step of the algorithm. This is done by randomly selecting coefficients form each partial genome to create an individual with all required coefficients ready for the normal GA to finish optimizing.

To generate the desired flap deflections for the simulation, a dynamic approach using the simulation run-time was used. This set deflections periods based on how far though the run the simulation was using T, the period and t, the current time:

$$f = \begin{cases} [0.2, 0.2] & for\ 0 \le t < 0.2T \\ [0.4, 0] & for\ 0.2T \le t < 0.4T \\ [0, 0] & for\ 0 \le t < 0.6T \\ [0,0.8] & for\ 0 \le t < 0.8T \\ [0.2, 0] & otherwise \end{cases} \quad (4.4)$$

Finally, to make the simulation more realistic, sensor imperfection was modeled as a zero-mean Gaussian noise with a standard deviation of 0.1 was independently added to each position as to mimic the expected positional GPS accuracy [44].

The search space of the coefficients for the GA were defined as the continuous space from zero to double the value of the Snowflake coefficient.

The implementation of the proposed TSGA algorithm can be found on GitHub [41].

Chapter 5 Path Planning

Low deployment height and limited computational power on the vehicle means that complex optimal path planning algorithms are not suitable for this application in the scope of this project. Thus, a simple multi-phase path planning method was chosen instead.

5.1 Method

A simplified particle model is used determine and update the path, commonly used in GNC applications [35]. For these assumptions to be correct: the side slip β is assumed to be zero and the vertical (v_{sink}) and horizontal (v_h) velocities constant, leading the motion described by Equation 5.1:

$$\begin{aligned} v_x &= \cos(\psi)\, V_h + w_x \\ v_y &= \sin(\psi)\, V_h + w_y \\ v_z &= V_{sink} \\ \dot{\psi} &= \lim\,(\psi_c - \psi_d) \end{aligned} \tag{5.1}$$

Where ψ_c is the current heading, ψ_d is the desired heading and $\lim$ clips the heading error to the maximum yaw rate.

Overall, the proposed path planning method can be broken down into 4 steps as seen in Figure 5.1 below.

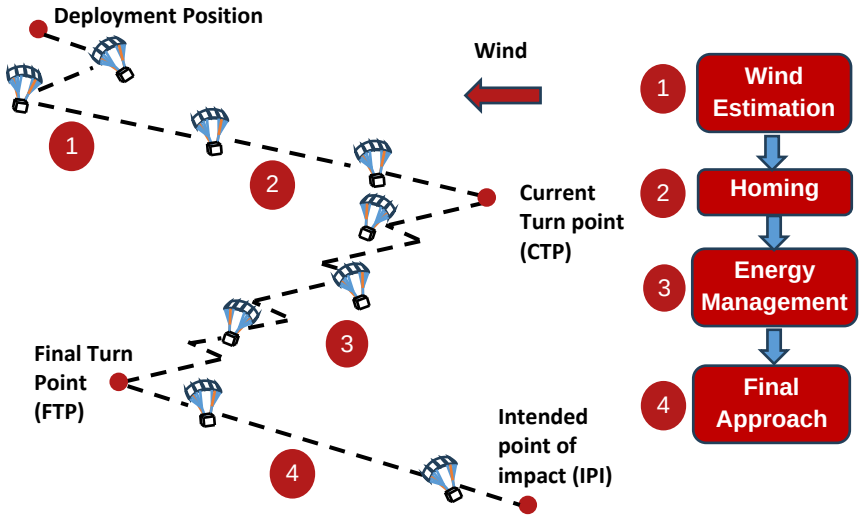


Figure 5.1: The proposed multi-phase algorithm

5.1.1 Wind estimation

By considering iterative measurements of the current velocity and following the method proposed by Ward et al. [18], an overdetermined system of linear equations can be found which RLS can be used to solve to get an estimate for the wind. This method was chosen as it is fast and it allows the wind estimate to be updated throughout the flight. The implementation of this can be found in Appendix B. For a fast estimate and

to minimise the wind drift, this phase is kept to only one spiral. Equation 5.2 was used to update the desired heading $\psi_{desired}$ using the current heading $\psi_{current}$, the update rate of the algorithm t_{upate} and the desired radius of the spiral $R_{spirial}$.

$$\psi_{desired} = \psi_{current} + \frac{v_{horizontal}}{R_{spirial}} \times t_{upate} \tag{5.2}$$

5.1.2 Homing

Since the energy management phase will drift downwind, the optimal location for the parafoil to begin spiralling to reach the final target point (FTP) must be determined. To simplify computation, the current turn point (CTP) is continuously estimated at the current altitude. This will guarantee that this position will be reached. The process is governed by equation 5.3:

$$\begin{aligned} T_{FTP} &= \frac{z_i - z_{FTP}}{v_{sink}} \\ CTP &= FTP - T_{FTP} \times V_{wind} \end{aligned} \tag{5.3}$$

Where T_{CTP} is the estimated time before the FTP is reached vertically, z_i is the current height and z_{FTP} is the height of the FTP. Knowing this and the wind vector, a compensated heading can be calculated as implemented in Appendix C .

5.1.3 Energy management

The deployment point of the parafoil will typically be upwind of the launch pad and intended impact point (IPI), as the natural tendency of rockets is to weathercock into the wind. To address this, a constant-radius spiral energy management phase, drifting downwind, was chosen as it offers two key advantages: reduced control effort and the ability to continuously update estimated wind conditions close to the critical final approach. To calculate the desired heading during this phase, equation 5.2 is reused.

5.1.4 Final approach

The final approach allows the parafoil to line up with the wind to limit its cross-range error and conduct a flaring manoeuvre to enable a soft landing. To calculate the position of the FTP relative to the IPI, Equation 5.4 is used to position it downwind:

$$FTP = FTP - T_{FTP} \times (v_{horz} - \|V_{wind}\|) \times \hat{v}_{wind} \tag{5.4}$$

To ensure that the parafoil aligns with the wind vector, a Pure-pursuit-controller was used to enable a smooth transition into the wind [45]. This looks ahead along a desired path allowing it to do transition between large headings smoothly and gradually align. The implementation of the Line Following method is shown in Appendix D.

Chapter 6 Results & Discussion

6.1 Simulation model

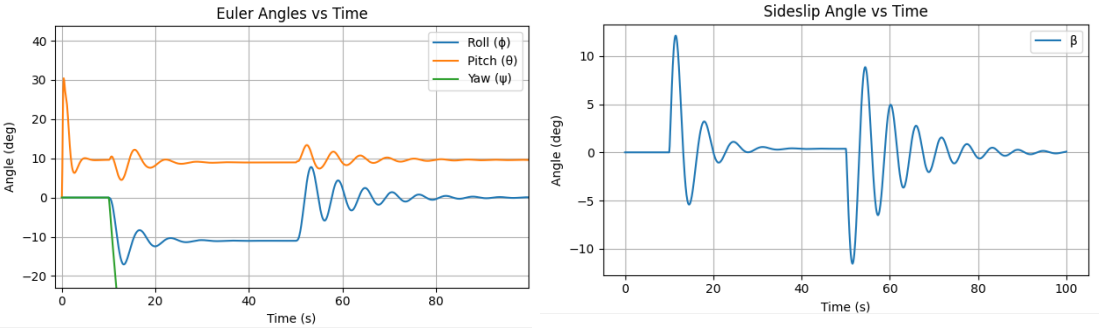
For initial testing and to provide a baseline to validate that the simulation functions correctly, aerodynamic and system data from the Snowflake vehicle was used [34]. Snowflake was chosen as it has an identical surface area relative to the parafoil which is used in this project (1m^2). Appendix E outlines Snowflake's parameters.

Once the physics simulation has been validated with a known parafoil model, to get a better understanding of how modelling a rocket as a payload differs to a typical dense payload such as the one used in the Snowflake model, a second model was derived, the 'Icarus' model. This models the rocket which this project planned to use as a payload in its expected configuration during recovery. However, as no aerodynamic data could be found about the selected parafoil to be used in this project, the aerodynamic coefficients and parafoil geometry were kept from the Snowflake model. The detailed derivation of the Icarus model can be found in Appendix E .

For all the following simulations, the initial values of the simulations were set as follows: deployment position: $P_{deploy} = [0, 0, 100]$ m; initial Euler Angles: $[\phi, \theta, \psi] = [0,0,0]$; initial body velocity: $v_b = [10,0,3]$ m/s and initial angular velocity: $\omega = [0, 0, 0]\ rad/s$.

6.1.1 Asymmetric flap control

The following simulation aims to quantify the parafoil behavior when an asymmetric flap deflection is applied to cause steering. Thus, a step input of 0.3 was applied to the left flap at 10s, lasting for 30 seconds. The initial delay was to ensure that the parafoil was in a steady state before entering the maneuver. The magnitude of 0.3 was chosen as this represents a good compromise of seeing permutations without pushing the system to the limit. The results are shown in Figure 6.1.



a) Euler Angles *b) Sideslip angles*

***Figure 6.1** – Asymmetric Flap Deflection Responses*

As seen from graph a) in Figure 6.1, the parafoil yaw decreases rapidly at 10 seconds. This indicates that the system is turning left, demonstrating that the parafoil is skid steering which agrees with literature [18]. The -10 degrees magnitude of the roll is also consistent [9]. However, the oscillations seen for both the sideslip angle of graph b) and the roll of graph a) whilst expected, are comparably large. Toglia et al. [9] uses a larger parafoil and mass which will be inherently more stable, explaining the difference, but this doesn't align with the smaller parafoil literature. One explanation could be the instantaneous nature of the transition between flap deflections which is not representative of the delayed servo actuation. This could cause a large instantaneous moment resulting in the large oscillations seen. However, when this was implemented, as shown in Appendix G, there was no change.

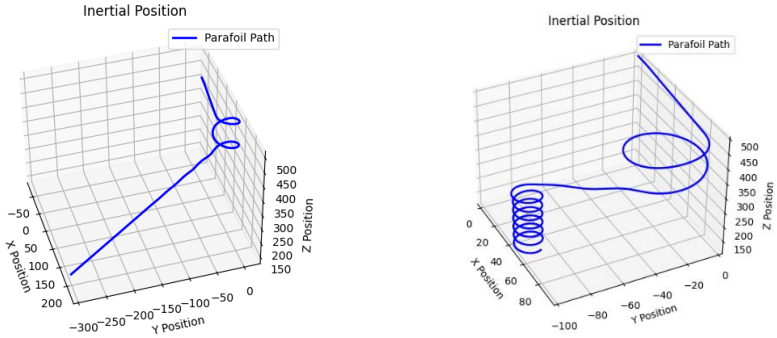
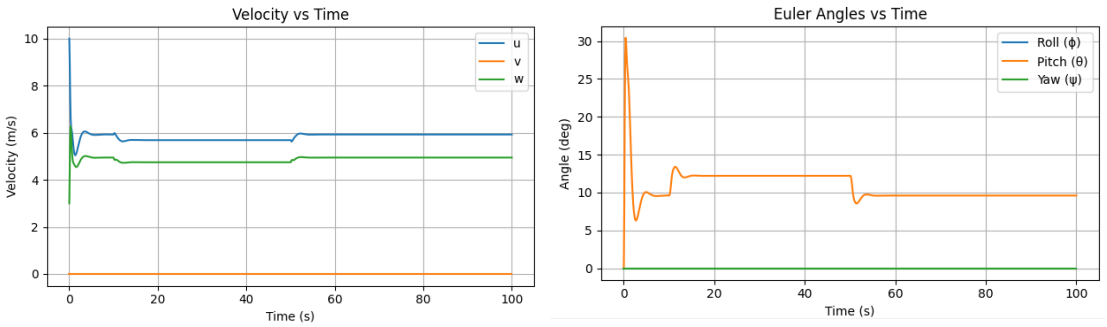


Figure 6.2: *Example simulated Parafoil Paths*

Overall, Figure 6.2 shows that whilst the oscillations seen do produce small permutations in the parafoil path, the overall parafoil motion can clearly be modelled.

6.1.2 Symmetric flap control

This simulation aimed to determine the effect of symmetric flap control to quantify the effects of flaring. Both flaps were set to 0.3 at 10 seconds, holding until 50 seconds in simulation time was reached. The results can be seen in Figure 6.3



a) Velocities *b) Euler angles*

Figure 6.3*: Simulated Symmetric flap deflection*

Due to this control input, the pitch Euler angle increases to 13 Degrees as depicted on the graph b). At the same time, graph a) shows that both the horizontal and vertical body velocities decrease by 0.2 m/s. This makes sense as the flap deflection reduces the efficiency of the parafoil, slowing it down [9]. This demonstrates that flaring at small flap deflections has a small effect. This agrees with Ward et al. [36].

6.1.3 Icarus comparison

The Icarus simulation aims to understand the possible effects of having a larger rocket as a payload. The asymmetric flap deflection simulation was repeated but for the Icarus model. The results are presented in Figure 6.4.

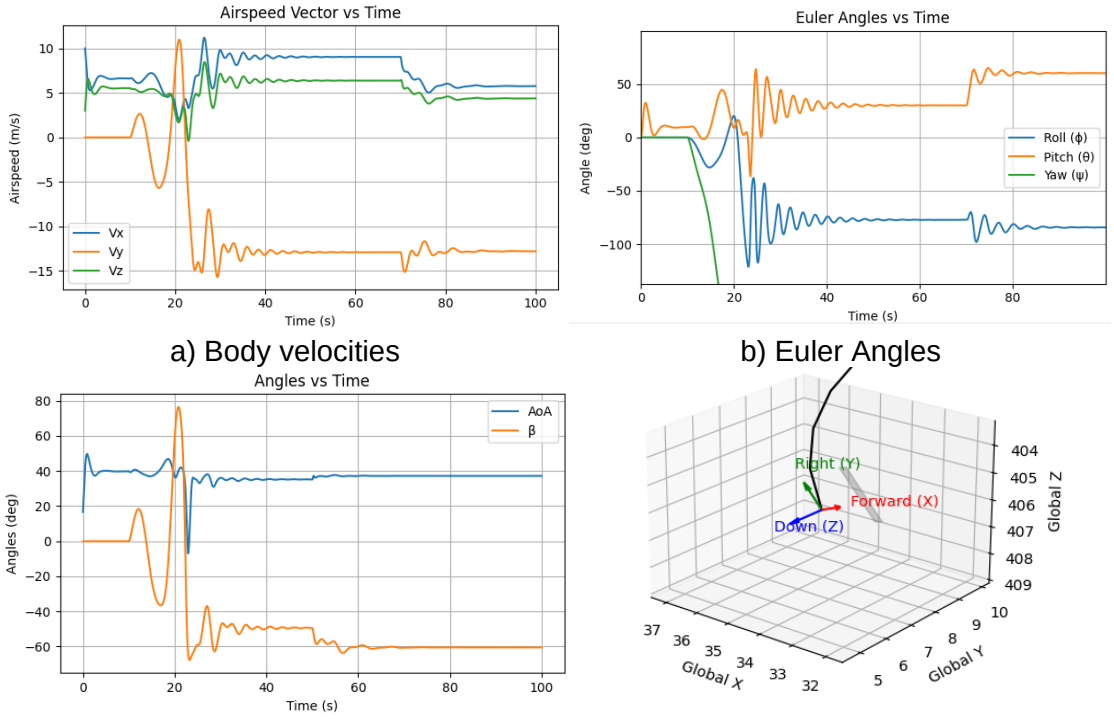


***Figure 6.4**: Simulated Asymmetric flap deflection using the Icarus model.*

As can be seen, the results are not what was expected. As the mass and moment of inertia have increased, theory would suggest that the parafoil motion should be more reserved as larger forces and moments are needed to attain the same motion.

Looking at graph a), the initial longitudinal steady state velocities are larger than the Snowflake model. This is expected as the larger mass will increase the descent rate which will consequently increase lift and thus the horizontal velocity. However, once the flap deflection is initiated, the side velocity increases as expected, but rather than settling as seen in the previous simulation, it increases. This can be explained from the longitudinal velocities which both decrease to 0. This suggests that the parafoil has

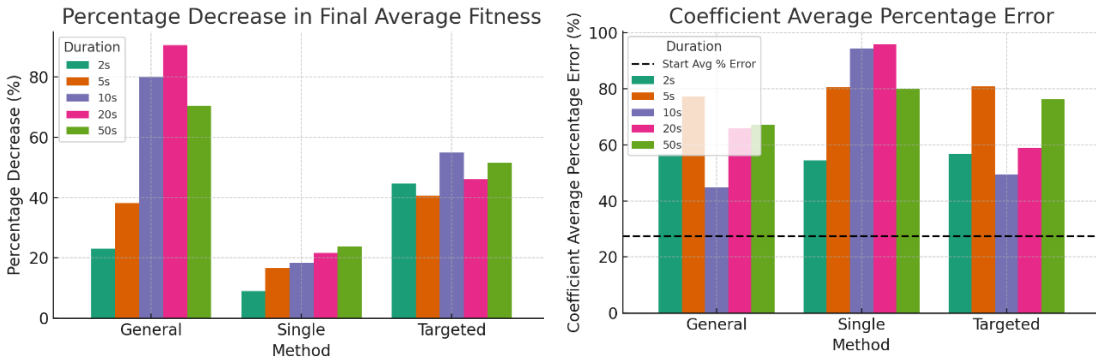
stalled, which can be seen from the angle of attack (AoA) in graph c) – it goes negative at 23 seconds.

Between 20s and 32s, the parafoil can be seen recovering from this stall before entering a steady state turn. The orientation of this turn can be seen in graph d). When the flap deflection is returned at 70s, the parafoil continues to spiral with large sideslip, yaw and roll angles as . There are two reasons for this.:

- Firstly, as the Snowflake coefficients were attained from drop test data, these coefficients characterize the entire system aerodynamics rather than just the parafoil. For these to be decoupled, the aerodynamic coefficients for the payload would also need to be modelled.
- Secondly, from these results, the assumptions made about the center of mass need to be reconsidered. The large roll and pitch values seen are unrealistic as the payload would generate resolving moment and prevent these large errors, especially at a low flap deflection of 0.3. This would also help mitigate the stall seen at the start of the maneuver.

6.2 Coefficient Calibration

The bounds and ideal coefficients can be seen in Appendix H. A population of 10 was chosen to determine the limits of the algorithm. Initial testing compared the first stage of the proposed method against the original method and a normal GA using 5 different simulation runtimes. In Figure 6.7a) represents the percentage decrease seen between the starting and final finesses in the simulations. The second graph represents the decrease in average absolute error between the ideal coefficients and the best fit individual in the final generation.

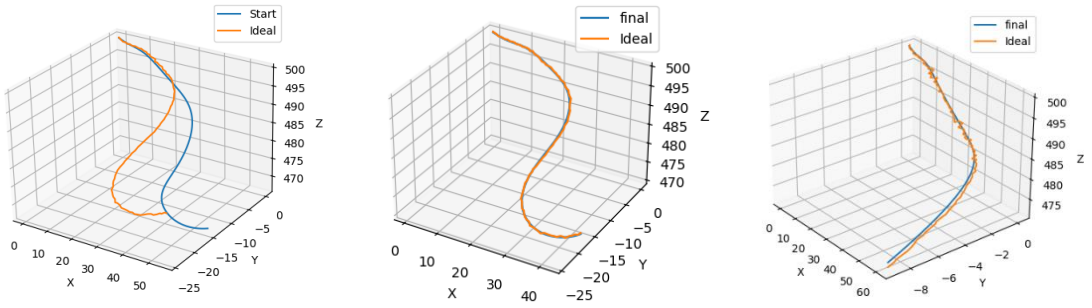


a) Percentage decrease in fitness *b) Percentage Decrease in fitness seen*

***Figure 6.7**: GA results comparing different simulation times*

From figure a) All methods minimize fitness. The typical General GA seen on the left most bars improve the fitness the most but is also the most sensitive to the duration size, with larger populations minimizing the fitness the most. The original 1st TSGA step, single parameter also improves with population size however the magnitude of

improvement is small compared to the other results. This is due to the limited ability of one parameter to influence fitness. The proposed targeted method remains constant across simulation size compared to the other methods. This indicates that this method is more suited for small sample sizes, where it outperforms a typical GA, at larger sizes a general GA is preferred. However, as seen in the second graph, the average percentage error worsens after running these methods. Still, the targeted and GA method perform better than the original TSGA method. This highlights the non-linearity and complexity of the search space resulting in local minima very close to the actual result.



a) starting Snowflake flight path

b) final path of proposed method

c) Validation flight, minimal overfitting

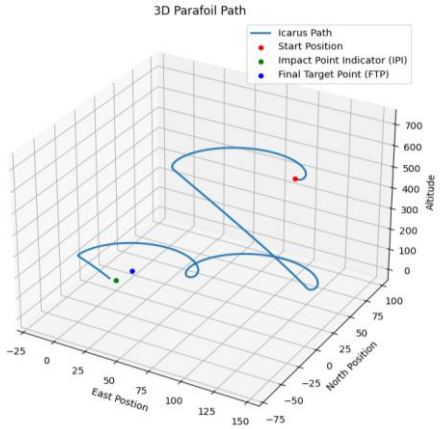
***Figure 6.8**: Proposed Method vs ideal flight paths.*

Combining this with the second step of using a GA, figure 6.8 a) and b) represent the flight simulation between the starting Snowflake coefficients and the final coefficients derived from the proposed method. An average error of 0.25 m is seen throughout the flight, an improvement from the initial 7.5m error seen at the start. A validation flight as seen in graph c) had an error of 0.75 m demonstrating the overfitting was minimal. The other methods path errors seen were identical. The overall time for completion was 24 minutes, however further optimizing the multithreading of evaluations within populations could half this time.

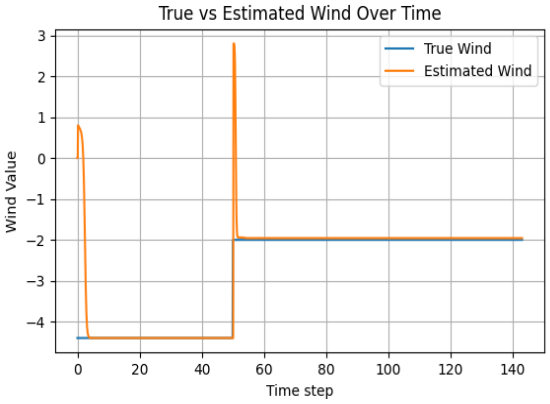
6.3 Guidance

The initial conditions of where the path planning would commence were attained by simulating the rockets ascent path as detailed in Appendix I. This led to the following initial position $[130, 0, 715]$ m and wind velocity of $[-4.4, 0, 0]$ m/s. The parafoil was assumed to have a horizontal and vertical velocity of 6m/s and 5 m/s respectively, attained from steady state velocities seen in Figure 6.3. Values of 0.7 and 1000 for λ and δ in the RLS algorithm were selected from iteratively fine tuning the results from different simulations.

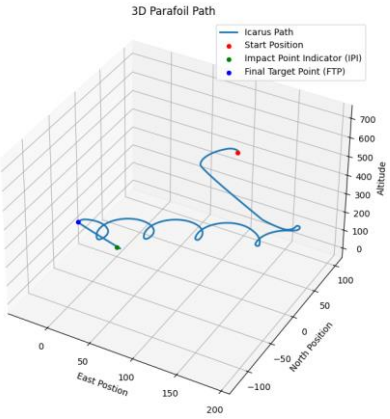
This case was tested, and the 3D path can be seen in Figure 6.9, graph a). It demonstrates that the algorithm can recover the Icarus rocket assuming ideal constant wind conditions, landing with a downrange error of 5.4 m.



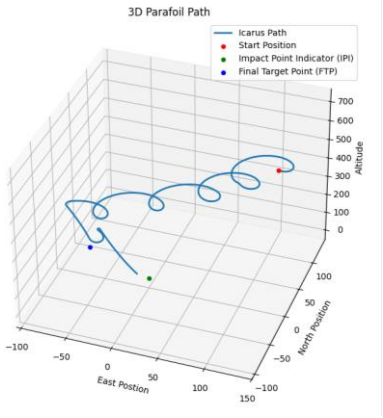
a) Rocket Simulation



b) Variable wind with estimations



c) With variable wind



d) Turning error example

***Figure 6.8**: Guidance Simulations*

The next simulation aimed to test the both the accuracy of the wind estimation and the resilience of the algorithm to variable wind. This was done by updating the magnitude of the wind from -4.4m/s to -2 m/s at 55 seconds as in graph b) of Figure 6.8. The wind estimation line closely follows the actual wind, demonstrating that the wind estimation can accurately estimate the wind to within an average error of 0.1 m. The large spikes seen are due to the large heading and thus velocity vector differences between the current and previous phases where the wind was updated. In the future, this could be mitigated by resetting the RLS algorithm between phases allowing ensuring that the velocity data used is up to date. Figure 6.7, graph c) shows that the wind change didn't affect the accuracy of the algorithm, it achieved a IPI error of 1.2m.

Graph d) shows one limitation of this algorithm - as the radius of the turnings are not considered in the calculation of FTP, this means that the parafoil is further away than expected resulting in an error of 18.2m. To mitigate this, the FTP could be set closer and the calculation using more refined final approach methods such as [36].

Chapter 7 Conclusion

7.1 Conclusion

This project successfully developed a nonlinear 6-DoF model, capable of capturing key aerodynamic and dynamic behaviours of a parafoil system. The analysis of rocket payload indicated that current assumptions commonly made to simplify parafoil modelling cannot be used when considering this application.

The proposed GA was evaluated, and the positional error was shown to be reduced to 0.25m accuracy however this did not translate to a reduction in the error seen when comparing the coefficient values.

A multiphase path planning algorithm was developed using a simplified particle model, with an integrated wind estimation technique based on Recursive Least Squares (RLS). The guidance algorithm was tested under various conditions, including variable wind scenarios, and demonstrated that the core trajectory guidance is possible. However, only simple test cases were conducted, More complex cases such as changing wind direction need to be validated against and due to the simple loose rules-based approach, in reality, this method would display worse results.

7.2 Achievements and Impact

- A 6-DoF model for parafoil-payload dynamics was successfully implemented.
- The first step of the TSGA Method was improved by using targeted groups of coefficients to get initial estimates.
- Integration of Recursive Least Squares for online wind estimation.
- A complete, computationally efficient guidance algorithm suitable for real-time applications.

7.3 Future work

To further build upon the developments presented in this report:

- Refine the dynamic model to include payload dynamics and areo-dynamics.
- Research on the relative role between the parafoil and payload remains to be characterised.
- Investigate flight patterns to optimise aerodynamic coefficient identification.
- Validate model predictions with actual flight test data.
- Integrate the guidance algorithm onto the avionics board to determine online feasibility and key challenges in a real-world environment.

References

[1] C. Dek *et al.*, “A recovery system for the key components of the first stage of a heavy launch vehicle,” *Aerosp Sci Technol*, vol. 100, May 2020, doi: 10.1016/j.ast.2020.105778.

[2] M. Mitchell, *An Introduction to Genetic Algorithms*. The MIT Press, 1996. doi: 10.7551/mitpress/3927.001.0001.

[3] T. GOODRICK, “Theoretical study of the longitudinal stability of high-performance gliding airdrop systems,” in *5th Aerodynamic Deceleration Systems Conference*, Reston, Virigina: American Institute of Aeronautics and Astronautics, Nov. 1975. doi: 10.2514/6.1975-1394.

[4] L. Zhang, H. Gao, Z. Chen, Q. Sun, and X. Zhang, “Multi-objective global optimal parafoil homing trajectory optimization via Gauss pseudospectral method,” *Nonlinear Dyn*, vol. 72, no. 1–2, pp. 1–8, Apr. 2013, doi: 10.1007/s11071-012-0586-9.

[5] A. Leeman, V. Preda, I. Huertas, and S. Bennani, “Autonomous parafoil precision landing using convex real-time optimized guidance and control,” *CEAS Space Journal*, vol. 15, no. 2, pp. 371–384, Mar. 2023, doi: 10.1007/s12567-021-00406-z.

[6] O. A. Yakimenko, “5.1.5 Six-DoF Model,” 2015, *American Institute of Aeronautics and Astronautics/Aerospace Press (AIAA)*. [Online]. Available: https://app.knovel.com/hotlink/khtml/id:kt010RIQP6/precision-aerial-delivery/six-dof-model

[7] G. Kowaleczko, “APPARENT MASSES AND INERTIA MOMENTS OF THE PARAFOIL,” 2014.

[8] G. Bonaccorsi, F. Braghin, P. di Milano Mentor, and M. B. Quadrelli, “Guidance & Control of a Parafoil-Based Landing on Titan,” 2018.

[9] C. Toglia, M. Vendittelli, C. Toglia, and M. Vendittelli, “Modeling and motion analysis of autonomous paragliders.”

[10] J. Tao, Q. lin Sun, P. long Tan, Z. qiang Chen, and Y. ping He, “Autonomous homing control of a powered parafoil with insufficient altitude,” *ISA Trans*, vol. 65, pp. 516–524, Nov. 2016, doi: 10.1016/j.isatra.2016.08.016.

[11] O. A. Yakimenko, “On the Development of a Scalable 8-DoF Model for a Generic Parafoil-Payload Delivery System.”

[12] G. Strickert, “Untersuchung der Relativbewegung von Gleitfallschirm-Last-Systemen,” *Aerosp Sci Technol*, vol. 8, no. 6, pp. 479–488, Sep. 2004, doi: 10.1016/j.ast.2004.04.003.

[13] L. Zhao, J. Tao, H. Sun, and Q. Sun, “Dynamic modelling of parafoil system based on aerodynamic coefficients identification,” *Automatika*, vol. 64, no. 2, pp. 291–303, 2023, doi: 10.1080/00051144.2022.2145697.

[14] N. Slegers and M. Costello, “Model Predictive Control of a Parafoil and Payload System.” [Online]. Available: https://digitalcommons.georgefox.edu/mece_fac

[15] G. B. Hur and J. Valasek, “System identification of powered parafoil-vehicle from flight test data,” in *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 2003. doi: 10.2514/6.2003-5539.

[16] R. Peña-García, R. D. Velázquez-Sánchez, C. Gómez-Daza-Argumedo, J. O. Escobedo-Alva, R. Tapia-Herrera, and J. A. Meda-Campaña, “Physics-Based Aircraft Dynamics Identification Using Genetic Algorithms,” Dec. 20, 2023. doi: 10.20944/preprints202312.1547.v1.

[17] O. A. Yakimenko, “5.1.6 Implementations of Six-DoF Model,” 2015, *American Institute of Aeronautics and Astronautics/Aerospace Press (AIAA)*. [Online]. Available: https://app.knovel.com/hotlink/khtml/id:kt010RIQQ5/precision-aerial-delivery/implementations-six-dof

[18] M. Ward, M. Costello, and N. Slegers, “Specialized System Identification for Parafoil and Payload Systems.” [Online]. Available: https://digitalcommons.georgefox.edu/mece_fac

[19] Z. Li, H. Sun, and Q. Sun, “Data-driven Hybrid Modeling of Parafoil Systems Based on NARX.”

[20] S. Farì and D. Grande, “Vector Field-based Guidance Development for Launch Vehicle Re-entry via Actuated Parafoil,” 2021.

[21] I. Aenishanslin, “Experimental research on rocket re-usability by GNC parafoil,” 2021, doi: 10.13009/EUCASS2022-7672.

[22] J. A. Bowman, F. L. Haan, J. M. Vogel, and J. P. Basart, “Determination of parafoil lift and drag coefficients using both three-dimensional modeling and experimental methods,” 2004.

[23] W. Wu, Q. Sun, S. Luo, M. Sun, Z. Chen, and H. Sun, “Accurate calculation of aerodynamic coefficients of parafoil airdrop system based on computational fluid dynamic,” *Int J Adv Robot Syst*, vol. 15, no. 2, Mar. 2018, doi: 10.1177/1729881418766190.

[24] L. Ling, “Method for Extracting Aerodynamic Coefficients and Winds On-the-Fly Using Real-Time GPS Data.”

[25] M. Majeed and I. N. Kar, “Aerodynamic parameter estimation using adaptive unscented Kalman filter,” *Aircraft Engineering and Aerospace Technology*, vol. 85, no. 4, pp. 267–279, 2013, doi: 10.1108/AEAT-Mar-2011-0038.

[26] R. Jaiswal, O. Prakash, and S. K. Chaturvedi, “A preliminary study of parameter estimation for fixed wing aircraft and high endurability parafoil aerial vehicle,” *INCAS Bulletin*, vol. 12, no. 4, pp. 95–109, 2020, doi: 10.13111/2066-8201.2020.12.4.9.

[27] M. Ulinowicz and J. Narkiewicz, “Aircraft Parameter Identification using Genetic Algorithms,” *29th Congress of the International Concil of the Aeronutical Sciences*, 2014.

[28] S. Luo, P. Tan, Q. Sun, W. Wu, H. Luo, and Z. Chen, “In-flight wind identification and soft landing control for autonomous unmanned powered parafoils,” *Int J Syst Sci*, vol. 49, no. 5, pp. 929–946, Apr. 2018, doi: 10.1080/00207721.2018.1433245.

[29] H. Gao *et al.*, “In-flightwind field identification and prediction of parafoil systems,” *Applied Sciences (Switzerland)*, vol. 10, no. 6, Mar. 2020, doi: 10.3390/app10061958.

[30] R. Cristi, “Pose and Wind Estimation for Autonomous Parafoils,” 2014.

[31] Z. Yu, H. Sun, Q. Sun, J. Tao, and Z. Chen, “Wind-field identification for parafoils based on deep Q-learning iterative inversion,” *Inf Sci (N Y)*, vol. 610, pp. 571–591, Sep. 2022, doi: 10.1016/j.ins.2022.07.185.

[32] J. M. Stein’, C. M. Ads, and A. L. Strahans, “An Overview of the Guided Parafoil System Derived from X-38 Experience.” [Online]. Available: https://ntrs.nasa.gov/search.jsp?R=20070026249

[33] O. A. Yakimenko, “7.3.3 Precision Placement Guidance Algorithms,” 2015, *American Institute of Aeronautics and Astronautics/Aerospace Press (AIAA)*. [Online]. Available: https://app.knovel.com/hotlink/khtml/id:kt010RIROC/precision-aerial-delivery/precision-placement-guidance

[34] O. A. Yakimenko, N. Slegers, R. A. Tiaden, and N. J. Slegers, “Development and Testing of the Miniature Aerial Delivery System Snowflake.” [Online]. Available: https://digitalcommons.georgefox.edu/mece_fac

[35] N. Slegers, O. A. Yakimenko, and N. J. Slegers, “Optimal Control for Terminal Guidance of Autonomous Parafoils.” [Online]. Available: https://digitalcommons.georgefox.edu/mece_fac

[36] M. Ward and M. Costello, “Adaptive glide slope control for parafoil and payload aircraft,” *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 4, pp. 1019–1034, 2013, doi: 10.2514/1.59260.

[37] L. Zhang, H. Gao, Z. Chen, Q. Sun, and X. Zhang, “Multi-objective global optimal parafoil homing trajectory optimization via Gauss pseudospectral method,” *Nonlinear Dyn*, vol. 72, no. 1–2, pp. 1–8, Apr. 2013, doi: 10.1007/s11071-012-0586-9.

[38] Z. Li and Y. Nan, “Optimal Path Planning and Tracking Control Methods for Parafoil,” *Applied Sciences (Switzerland)*, vol. 13, no. 14, Jul. 2023, doi: 10.3390/app13148115.

[39] “RocketPy: By Rocketeers, for Rocketeers.” Accessed: May 01, 2025. [Online]. Available: https://docs.rocketpy.org/en/latest/user/index.html

[40] R. L. Burden and D. F. J., *Numerical Analysis*, 9th ed. Cengage Learning, 2010.

[41] G. Tyler, “Level_4_project,” 2025. Accessed: May 07, 2025. [Online]. Available: https://github.com/el21tg/Level_4_project

[42] G. Rudolph, “Convergence Analysis of Canonical Genetic Algorithms,” 1994.

[43] F.-A. Fortin, U. Marc-André Gardner, M. Parizeau, and C. Gagné, “DEAP: Evolutionary Algorithms Made Easy François-Michel De Rainville,” 2012. [Online]. Available: http://deap.gel.ulaval.ca,

[44] Ublox, “MAX-M10S Standard precision GNSS module Datasheet.” Accessed: May 10, 2025. [Online]. Available: https://content.u-blox.com/sites/default/files/MAX-M10S_DataSheet_UBX-20035208.pdf

[45] R. Coulter Craig, “Implementation of the Pure Pursuit Path Tracking Algorithm,” 1992.

[46] S. Niskanen, “Development of an Open Source model rocket simulation software.”

Appendix A 6-DoF Model Derivation

The three frames of reference used throughout this work are the Inertial Earth Frame, the Body Frame and the Canopy Frame. The origin and axis definitions can be found in Table A1 below.

Table A1 : Model frame origin and axis definitions

Inertial Earth Frame $\{\boldsymbol{\mathcal{R}}\}$	**Body Frame** $\{\boldsymbol{b}\}$	**Canopy Frame** $\{c\}$
Origin: Fixed on the earth surface (typically IPI).	**Origin**: Vehicles centre of Mass	**Origin**: Canopy centre of pressure, in longitudinal z-x plane of $\{b\}$.
$\boldsymbol{x}$ **axis**: Positive in the direction of True North.	$\boldsymbol{x_b}$ **axis**: Positive in the direction of travel in the longitudinal axis of PADs in plane of symmetry.	$\boldsymbol{x_c}$ **axis**: Positive in the direction towards where the local air velocity is coming from.
$\boldsymbol{y}$ **axis**: Positive in the direction of True East.	$\boldsymbol{y_b}$ **axis**: perpendicular to the longitudinal plane, positive determined by right-hand-rule.	$\boldsymbol{y_c}$ **axis**: perpendicular to the longitudinal plane, positive determined by right-hand-rule.
$\boldsymbol{z}$ **axis**: Positive towards centre of the Earth	$\boldsymbol{z_b}$ **axis**: Positive downwards from centre of canopy passing through COG of payload.	$\boldsymbol{z_c}$ **axis**: Positive facing downwards, perpendicular

Within the Earth Frame, the inertial position, ${X_b}^{\mathcal{R}}$ and wind velocity vector $w^{\mathcal{R}}$ are defined. In addition, the Euler angles representing the body frames roll, pitch and yaw orientations are also defined. Equations A1,A2, A3 define these quantities respectively.

$$ {X_b}^{\mathcal{R}} = [X, Y, Z]^T \tag{A1} $$

$$ w^{\mathcal{R}} = \left[w_x, w_y, w_z\right]^T \tag{A2} $$

$$ [\phi, \theta, \psi]^T \tag{A3} $$

Within the Body Frame, the body angular and linear velocity are described by Equations A4,A5 and A6.

$$ {v_b}^{\boldsymbol{\mathcal{B}}} = [u, v, w]^T \tag{A4} $$

$$ \omega^{\boldsymbol{\mathcal{B}}} = [p, q, r]^T \tag{A5} $$

$$ {v_a}^{\mathcal{B}} = [v_x \quad v_y \quad v_z]^T \tag{A6} $$

Three consecutive rotations are used to convert from the Body Frame to the reference frame using the Euler angles. Here the ZYX order of rotation has been adopted, multiplied out, this produces the direction cosine matrix, $\boldsymbol{R}_{\mathcal{B}}^{\mathcal{R}}$ of Equation A7.

$$v_b{}^{\mathcal{R}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} cos\theta\ cos\psi & sin\phi\ sin\theta\ cos\psi - cos\phi\ sin\psi & cos\phi\ sin\theta\ cos\psi + sin\phi\ sin\psi \\ cos\theta\ sin\psi & sin\phi\ sin\theta\ sin\psi + cos\phi\ cos\psi & cos\phi\ sin\theta\ sin\psi - \sin\phi\ cos\psi \\ -sin\theta & \sin\phi\ cos\theta & cos\phi\ cos\theta \end{bmatrix} v_b \quad \text{(A7)}$$

$$v_b{}^{\mathcal{R}} = \boldsymbol{R}_{\mathcal{B}}^{\mathcal{R}}\, v_b$$

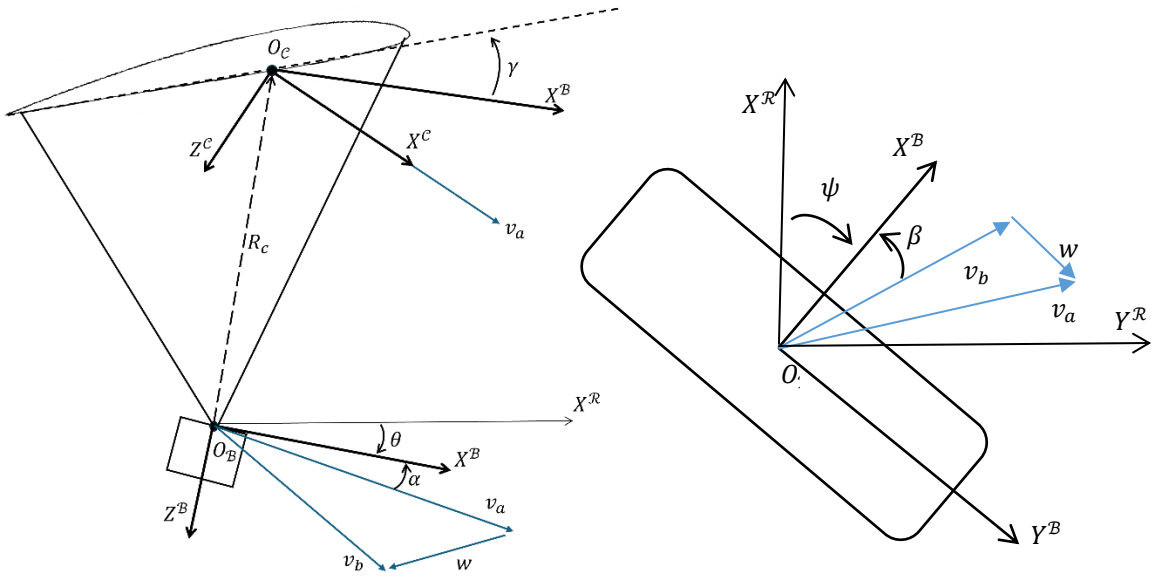
To convert from the Canopy Frame to the Body Frame two angles are required, these are the angle of attack, α and the sideslip angle, β. Combining these rotations sequentially, forms the rotation matrix $\boldsymbol{R}_{\mathcal{B}}^{\mathcal{C}}$ (from B to C), Equation A8

$$\boldsymbol{R}_{\mathcal{B}}^{\mathcal{C}} = R_\beta \cdot R_\alpha = \begin{bmatrix} cos\beta & -sin\beta & 0 \\ sin\beta & cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\alpha & 0 & sin\alpha \\ 0 & 1 & 0 \\ -sin\alpha & 0 & cos\alpha \end{bmatrix} = \begin{bmatrix} cos\alpha cos\beta & -sin\beta & sin\alpha cos\beta \\ cos\alpha sin\beta & cos\beta & sin\alpha sin\beta \\ -sin\alpha & 0 & cos\alpha \end{bmatrix} \quad \text{(A8)}$$

The formal definitions of these angles are presented in the next section.

A1 Parafoil definitions

Here the geometry and characteristics of the parafoil is defined. Figure A1 represents the definitions of the canopy and body frame and key angles involved.



a) Longitudinal Geometry a) Lateral Geometry

Figure A1 : Parafoil geometry

A summary of these definitions depicted in the Figure A1 can be can be found in Table A2 below.

Table A2 : Parafoil geometry definitions

Quantity	Definition
Rigging angle	γ
Parafoil wingspan	b
Parafoil average Chord length	c
Parafoil Surface Area	S
Angle of Attack	$\alpha = \tan^{-1}\left(\frac{v_z}{v_x}\right)$
Sideslip angle	$\beta = tan^{-1}\left(\frac{v_y}{\sqrt{u^2+w^2}}\right)$
Relative Air Velocity	${v_a}^{\mathcal{B}} = [v_x \quad v_y \quad v_z]^T = v_b - \boldsymbol{R}_{\mathcal{R}}^{\mathcal{B}} w$
Position vector between canopy and Body	${R_c}^{\mathcal{B}} = \left[{}^{c}/_{4} \quad 0 \quad -R_{pc} - R_{lc}\right]^T$

The flap deflection inputs can be considered as a value between 0 and 1, representing the ratio between the current deflection and the maximum deflection. The left and right deflections are referred to as δ_l and δ_r. However, for simplicity, they are to re-cast into symmetrical and asymmetrical deflections as these better represent longitudinal and lateral dynamic effects. These are represented by Equations A9 and A10.

$$\delta_s = \frac{1}{2}(\delta_r + \delta_l) \tag{A9}$$

$$\delta_a = \delta_r - \delta_l \tag{A10}$$

A1.1 Kinematic equations

To convert from angular velocities in the body frame to the Euler angle rates, Equation A11 can be used:

$$\begin{bmatrix}\dot{\psi}\\ \dot{\theta}\\ \dot{\phi}\end{bmatrix} = \frac{1}{cos\theta}\begin{bmatrix}0 & sin\phi & cos\phi\\ 0 & cos\phi\, cos\theta & -sin\phi\, cos\theta\\ cos\theta & sin\phi & cos\phi\, sin\theta\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix} = \mathcal{L}(\psi,\theta,\phi,)\begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{A11}$$

As the parafoil is a rotating rigid body, additional fictitious derivatives must also be considered using the transport theorem of Equation A12:

$$\frac{d^{\mathcal{R}}}{dt}X^{\mathcal{R}} = \frac{d^{\mathcal{B}}}{dt}X^{\mathcal{B}} + \omega^{\mathcal{B}} \times X^{\mathcal{B}} \tag{A12}$$

Or simply, omitting the superscript, Equation A13:

$$\frac{d^{\mathcal{R}}}{dt}X = \dot{X} + \omega \times X \tag{A13}$$

Secondly, the cross product of the angular velocity can be represented as the following matrix $\boldsymbol{S_\omega}$, named the skew symmetric matrix of ω, Equation A14 defines this.

$$\boldsymbol{S_\omega} = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & -p & 0 \end{bmatrix} \quad \text{(A14)}$$

Substituting A14 into A13, the Equation A15 is found:

$$\frac{d^{\mathcal{R}}}{dt}X = \dot{X} + \boldsymbol{S_\omega}X \quad \text{(A15)}$$

A1.2 Dynamics

The forces which act on the body are gravity, the aerodynamic forces and moments produced by the canopy. These are calculated there and then can be rotated to the body frame.

A1.2.1 Aerodynamic forces

Aerodynamic surfaces produce a drag force, F_D, a side force F_Y and a lifting force F_L. These act at the centre of pressure of the wing and are defined relative to the local airspeed velocity by Equations A16:

$$F_D = \frac{1}{2}\rho\|v_a\|^2 C_D S \qquad F_Y = \frac{1}{2}\rho\|v_a\|^2 C_Y S \qquad F_L = \frac{1}{2}\rho\|v_a\|^2 C_L S \quad \text{(A16)}$$

Where ρ is the air density and the aerodynamic force coefficients C_D, C_Y and C_L are defined below in Equations A17:

$$C_Y = C_{Y,\beta}\beta \quad \text{(A17)}$$

$$C_L = C_{L_0} + C_{L_\alpha}(\alpha + \gamma) + C_{L_{\delta,s}}\delta_s \quad \text{(A18)}$$

$$C_D = C_{D_0} + C_{D_\alpha}(\alpha + \gamma) + C_{D_{\delta,s}}\delta_s \quad \text{(A19)}$$

The first terms in A18 represent the __, the second terms represent

In Vector form, the aerodynamic force in the canopy frame, Equation A20:

$$F_A{}^{\mathcal{C}} = -[F_D \quad F_Y \quad F_L]^T \quad \text{(A20)}$$

A1.2.2 Linear momentum

Looking at the body frame, only the weight of the system F_g is defined in here, as shown in Equation A21:

$$F_g = m.g[-sin\theta \quad cos\theta sin\phi \quad cos\theta cos\phi]^T \quad \text{(A21)}$$

The aerodynamic forces, Equation 19 must also be in the body frame producing Equation A22.

$$F_A = \boldsymbol{R}_w^{\mathcal{B}} F_A{}^{\mathcal{C}} \tag{A22}$$

Equation A23 represents Newtons second law. However, as this is in the inertial frame and the body velocity is described in the rotating body frame, the transport theorem (Equation A15) can be applied, resulting in equation A24.

$$\sum F = m\frac{d^{\mathcal{R}}}{dt} v_b \tag{A22}$$

$$F = m(\dot{v}_b + \omega \times v_b) \tag{A23}$$

Summating aerodynamic force (A20) and weight (A21), rearranging equation A23 and substituting the angular symmetric skew matrix (A14), the final Equation A24 can be found explaining the linear dynamics of the model.

$$\dot{v}_b = \frac{1}{m}\left(F_A + F_g - m(\boldsymbol{S}_{\boldsymbol{\omega}} v_b)\right) \tag{A24}$$

A1.2.3 Angular moment consideration

From the assumptions stated, there is only one angular moment to consider, this is the aerodynamic moment M_A. This can be broken down into its components: The Rolling L, Pitching M and Yawing N moments as detailed in Equation A25.

$$L = \frac{1}{2}\rho\|v_a\|^2 S C_l b \qquad M = \frac{1}{2}\rho\|v_a\|^2 S C_m c \qquad N = \frac{1}{2}\rho\|v_a\|^2 C_n S b \tag{A25}$$

Where aerodynamic coefficients can be defined in Equation A26, A27, A28:

$$C_l = C_{l,\beta}\beta + C_{l,p}\frac{b}{2\|v_a\|}p + C_{l,r}\frac{b}{2\|v_a\|}r + C_{l,\delta_a}\delta_a \tag{A26}$$

$$C_m = C_{m,0} + C_{m,\alpha}\alpha + C_{m,q}\frac{b}{2\|v_a\|}q \tag{A27}$$

$$C_n = C_{n,\beta}\beta + C_{n,p}\frac{b}{2\|v_a\|}p + C_{n,r}\frac{b}{2\|v_a\|}r + C_{n,\delta_a}\delta_a \tag{A28}$$

Similarly to Equation A20 in the previous section, the moments can be combined together, to form Equation A29:

$$M_A = [L \quad M \quad N]^T \tag{A29}$$

Taking the dynamic pressure out of A24, equation A25 now becomes:

$$M_A = \begin{bmatrix} L \\ M \\ N \end{bmatrix} = \frac{1}{2}\rho\|v_a\|^2 S \begin{bmatrix} C_l b \\ C_m \bar{c} \\ C_n b \end{bmatrix} \tag{A30}$$

Now the total angular momentum, Equation A31, can be found, again using the transport theorem (A15) to consider the rotating body frame.

$$M = \frac{d}{dt}H + \omega \times H \quad \text{(A31)}$$

Where H is the angular momentum.

Expanding and differentiating the angular momentum, and substituting A14 results in Equation A32:

$$M = \boldsymbol{I}\dot{\omega} + \boldsymbol{S_\omega}(\boldsymbol{I}\omega) \quad \text{(A32)}$$

Consequently, the lateral dynamics of the system can now be expressed as Equation A33:

$$\dot{\omega} = \boldsymbol{I}^{-1}\left(M_A - \boldsymbol{S_\omega}(\boldsymbol{I}\omega)\right) \quad \text{(A33)}$$

Appendix B Recursive Least Squares (RLS) of the wind

The RLS algorithm is derived off by Ward et al. [18] and inspiration for the following code was taken from [28]. Overall, the method works knowing that the parafoil typically travels at a constant velocity. It can then compare the current and previous velocity magnitudes and components to estimate the current error in the wind estimate. To update the wind estimate, the Kalman gain is used. This uses two variables, $lambda$ which determines how quickly the algorithm forgets the effects of previous estimates and $delta$ which is used to set the confidence of previous estimates.

```python
class WindRLS:      You, 9 minutes ago • Uncommitted changes
    """
    Follows derivation from:
    Luo, S., Tan, P., Sun, Q., Wu, W., Luo, H., & Chen, Z. (2018).
    In-flight wind identification and soft landing control for autonomous unmanned powered parafoils.
    International Journal of Systems Science, 49(5), 929–946. https://doi.org/10.1080/00207721.2018.1433245

    """
    def __init__(self, lambda_=0.99, delta=1000.0):
        self.lambda_ = lambda_
        self.theta = np.zeros((2, 1))  # [V_Wx, V_Wy]
        self.P = delta * np.eye(2)
        self.Vx_prev = 0
        self.Vy_prev = 0

    def update(self, Vx_k, Vy_k):
        # Compute y(k)
        V_k = np.sqrt(Vx_k**2 + Vy_k**2)
        V_k_1 = np.sqrt(self.Vx_prev**2 + self.Vy_prev**2)
        y_k = 0.5 * (V_k**2 - V_k_1**2)

        # Compute phi(k)
        phi_k = np.array([[Vx_k - self.Vx_prev],
                          [Vy_k - self.Vy_prev]])

        # Kalman gain K(k)
        numerator = self.P @ phi_k
        denominator = self.lambda_ + (phi_k.T @ self.P @ phi_k)
        K_k = numerator / denominator

        # Update theta
        error = y_k - (phi_k.T @ self.theta)
        self.theta = self.theta + K_k * error

        # update velocity
        self.Vx_prev = Vx_k
        self.Vy_prev = Vy_k

        # Update P
        self.P = (np.eye(2) - K_k @ phi_k.T) @ self.P / self.lambda_

    def get_wind_estimate(self):
        return self.theta.flatten()
```

Figure B1: RLS algorithm in code

Appendix C Wind compensated heading calculation

```python
def compute_required_heading(wind_vector, airspeed, target_vector):
    """
    Calculate the required heading angle to follow the desired vector in wind.
    Uses wind triangle, taking norms and solving quadratic for ground speed along
    desired vector.
    Args:
        wind_vector (list): (w_x, w_y), wind vector in m/s.
        airspeed (float): Airspeed of the parafoil  in m/s.
        target_vector (list): (d_x, d_y), vector pointing to the desired target.

    Returns:
        float: Required heading angle in radians.
        np.ndarray: Air velocity vector (v_a_x, v_a_y).
    """      You, 2 weeks ago • Inital_path_planning_completed …

    # format and normalise
    w = np.array(wind_vector)
    d = np.array(target_vector)
    d_hat = d / np.linalg.norm(d)

    # Solve for Vg using quadratic equation
    a = 1
    b = -2 * np.dot(d_hat, w)
    c = np.dot(w, w) - airspeed**2

    discriminant = b**2 - 4 * a * c
    if discriminant < 0:
        raise ValueError("No solution: desired vector cannot be achieved with given airspeed and wind.")

    Vg = (-b + np.sqrt(discriminant)) / (2 * a)

    # Compute air velocity vector
    v_g = Vg * d_hat
    v_a = v_g - w

    # Compute heading angle
    heading = np.arctan2(v_a[1], v_a[0])

    return heading, v_a
```

Figure C1: Heading calculation code

Appendix D Line Following Algorithm

```python
def smooth_heading_to_line_with_wind(position, line_point,
                                     line_direction, lookahead_distance, wind_vector, airspeed):
    """
    Compute the heading required to move toward the lookahead point, considering wind.
    """
    p = np.array(position)
    a = np.array(line_point)
    d = np.array(line_direction)
    d = d / np.linalg.norm(d)

    # Project position onto the line to find the closest point
    ap = p - a
    t = np.dot(ap, d)
    closest_point = a + t * d

    # Lookahead point on the line
    lookahead_point = closest_point + d * lookahead_distance

    # Desired ground track vector
    desired_track = lookahead_point - p
    desired_track /= np.linalg.norm(desired_track)

    # Solve for air vector that, when combined with wind, gives the desired track
    air_vector = desired_track * airspeed - wind_vector

    # Compute heading from air vector
    required_heading = np.arctan2(air_vector[1], air_vector[0])
    return required_heading
```

Figure D1: Line heading code

Appendix E Snowflake Parafoil Parameters

Table E.1 Showing the Snowflake Parafoil parameters used in this work [17]

TABLE 5.3 IDENTIFIED PARAMETERS OF A SIX-DOF MODEL OF SNOWFLAKE PADS

Parameter	Value
System mass m	2.4 kg (5.2 lb)
Canopy span b, chord c, thickness t	1.35 m (4.45 ft), 0.75 m (2.47 ft), 0.075 m (0.25 ft)
Canopy reference area S	1 m^2 (11.0 ft^2)
Rigging angle μ	−12 deg
Steady-state aerodynamic velocity V_a	7.3 m/s (14.2 kt)
Inertia matrix **I** elements	$\mathbf{I} = \begin{bmatrix} 0.42 & 0 & 0.03 \\ 0 & 0.40 & 0 \\ 0.03 & 0 & 0.053 \end{bmatrix}$ kg · m^2, $\mathbf{I} = \begin{bmatrix} 9.97 & 0 & 0.71 \\ 0 & 9.5 & 0 \\ 0.71 & 0 & 1.26 \end{bmatrix}$ lb · ft^2
Apparent mass matrix	$\mathbf{I}_{a.m.} = \text{diag}([0.012, 0.032, 0.42])$ kg, $\mathbf{I}_{a.m.} = \text{diag}([0.027, 0.071, 0.93])$ lb
Apparent inertia matrix	$\mathbf{I}_{a.i.} = \text{diag}([0.054, 0.14, 0.0024])$ kg · m^2, $\mathbf{I}_{a.i.} = \text{diag}([1.29, 3.34, 0.06])$ lb · ft^2
Vector from the mass center to the apparent mass center	$\mathbf{r}_{BM} = [0.046,\ 0,\ -1.11]^T$ m, $\mathbf{r}_{BM} = [0.15,\ 0,\ -3.64]^T$ ft
Maximum brake deflection $\delta_{s\,\max}$	0.13 m (0.42 ft or 5 in.)
Aerodynamic coefficients	$C_{D0} = 0.25$, $C_{D\alpha^2} = 0.12$, $C_{Y\beta} = -0.23$, $C_{L0} = 0.091$, $C_{L\alpha} = 0.90$, $C_{m0} = 0.35$, $C_{m\alpha} = -0.72$, $C_{mq} = -1.49$, $C_{l\beta} = -0.036$, $C_{lp} = -0.84$, $C_{lr} = -0.082$, $C_{l\delta_a} = -0.0035$, $C_{n\beta} = -0.0015$, $C_{np} = -0.082$, $C_{nr} = -0.27$, $C_{n\delta_a} = 0.0115$

Appendix F Icarus Model Formation

Appendix G Servo delay estimation

To model the delay due to the servos, the following function was created. This requires the flap time constant T_{flap} which represents the time required to pull the flap from zero to max. This can be calculated by the following Equation G1:

$$T_{flap} = \frac{x_{max}}{\pi \times D_{spool}} \times S_{rpm} \quad \text{(G1)}$$

Where x_{max} is the maximum control line travel, S_{rpm} is the maximum revolutions per minute that the servo can operate at and D_{spool} is the diameter of the control line spools. $x_{max} = 0.13\ m$ was attained from the Snowflake table outlined in Appendix E . As the spool radius of Snowflake was not directly specified, the diameter was estimated from the pictures to be 1.5 inches (0.0381 m) [34]. Unfortunately, no data about the servos was found so this was estimated to be 1 rotation a second. Consequently, Equation G2:

$$T_{flap} = \frac{0.13}{2\pi \times 0.0381 \times 1} = 1.08 \quad \text{(G2)}$$

The actual flap deflection can then be updated following Figure G1

```
def update_flaps(self,dt):
    """updates"""
    max_rate = dt/self.flap_time_constant
    self.flap_l += np.clip(self.flap_l_desired - self.flap_l, -max_rate, max_rate)
    self.flap_r += np.clip(self.flap_r_desired - self.flap_r, -max_rate, max_rate)
    # calculate the flap deflection angles
    self.delta_a = self.flap_r - self.flap_l  # asymmetric flap deflection
    self.delta_s = 0.5*(self.flap_l + self.flap_r) # symmetric flap deflection
```

Figure G1: Simulated Asymmetric flap deflection

Overall, these yielded responses shown in Figure G2 which are identical to the simulations without this feature:

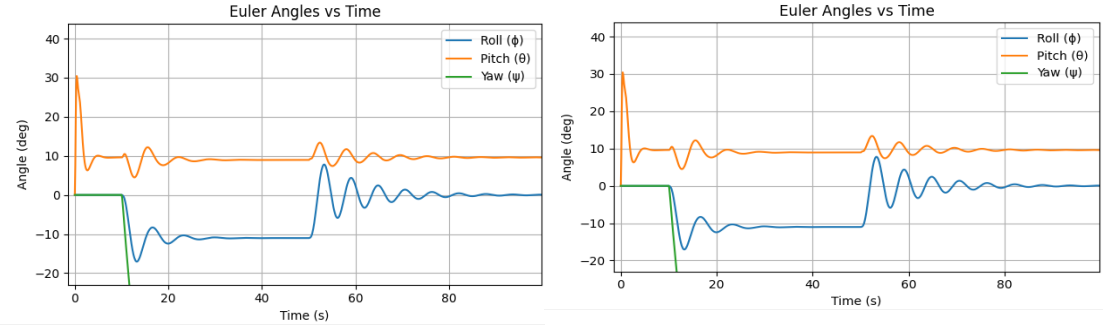


Figure G2: Servo Asymmetric flap deflection Response

Appendix H TSGA Coefficient Values

Coefficient	Desired Value	Lower bound	Upper bound
C_{L_0}	0.12	0	0.2
C_{L_α}	0.9	0	1.8
$C_{L_{\delta,s}}$	0.3	0	0.6
C_{D_0}	0.3	0	0.5
C_{D_α}	0.05	0	0.25
$C_{D_{\delta,s}}$	0.25	0	0.5
$C_{Y,\beta}$	-0.15	-0.5	0
$C_{l,\beta}\beta$	-0.07	-0.1	0
$C_{l,p}$	-0.8	-1.7	0
$C_{l,r}$	-0.082	-0.2	0
C_{l,δ_a}	-0.004	0.01	0
$C_{m,0}$	0.3	0	0.7
$C_{m,\alpha}$	-0.5	-1.5	0
$C_{m,q}$	-1.7	-3	0
$C_{n,\beta}$	-0.002	-0.1	0
$C_{n,p}$	-0.082	-0.2	0
$C_{n,r}$	-0.27,	0.5	0
C_{n,δ_a}	0.02	0	0.03

Appendix I Estimation of Rocket Deployment Position

To estimate the Rocket path and deployment altitude, the widely used open rocket simulation package was used [46]. The flight was modelled with 10 mph winds from the east, using a J400 Rocket motor achieving apogee of 760m.

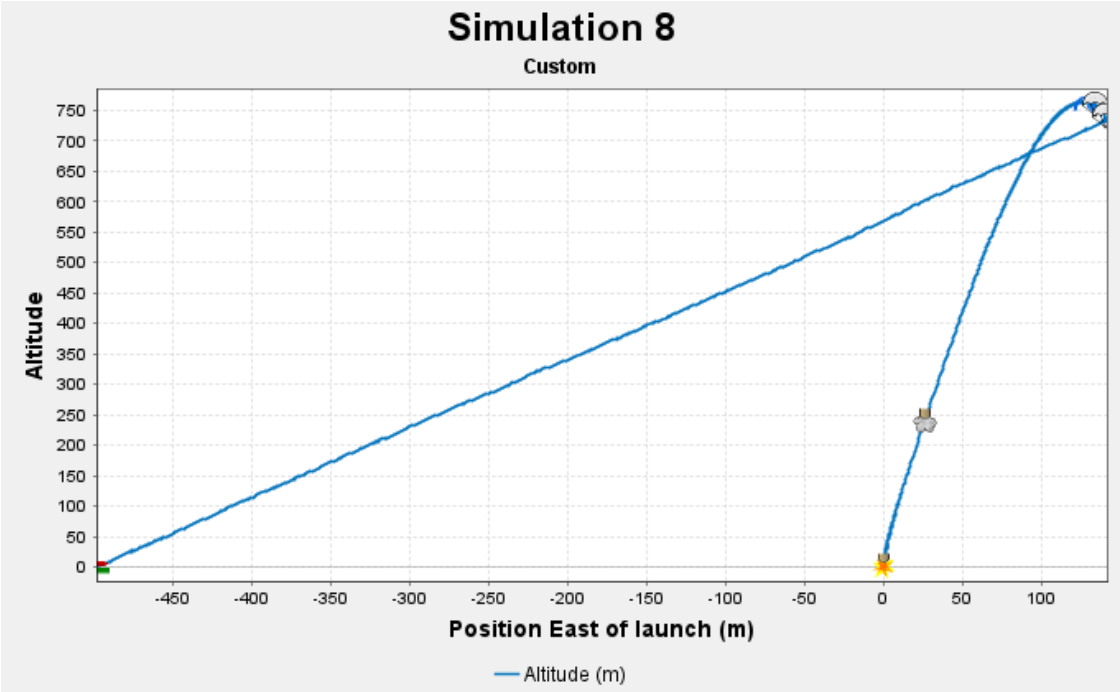


Figure H1 : Estimated rocket path and parafoil deployment

As seen in Figure H1, the separation position, where the ejection charge separates the rocket, allowing the parachutes to exit is at [150, 0, 750] m. As a drogue is deployed first to reduce the initial velocity and also get the parafoil to deploy from above, it can be assumed that the parafoil deploys once the system reaches the steady state terminal velocity of the drogue.

This leads to an actual deployment altitude of [130, 0, 715].