Wind Sensing and Rejection for Agile Fixed-Wing Unmanned Aerial Vehicles

Jackson Empey, Department of Mechanical Engineering
McGill University, Montreal
July, 2022

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of

Master of Science

Abstract

Agile fixed-wing unmanned aerial vehicles (UAVs) are an emerging class of autonomous aircraft that combine the efficient distance flight of conventional fixed-wings with the maneuverability, and ability to hover, of multi-rotor platforms. These characteristics make agile fixed-wings compelling candidates for a wide variety of missions from search and rescue to surveillance. The remarkable maneuverability of these platforms is largely due to their low weight, and powerful thrusters, which drive reliable and fast slipstream airflow over the control surfaces, keeping them effective during low-speed flight. While their low weight is critical to the agility of this class of UAV, it also makes them highly susceptible to wind and can present challenges for precision flight outdoors.

The objective of this thesis is to reject wind disturbances on agile fixed-wing platforms while minimizing any reductions to the performance of the agile fixed-wing. To achieve this objective, a modular, feedforward, wind rejection controller, relying on a wind estimate, was developed to improve the performance of existing agile fixed-wing controllers in outdoor environments. To provide a wind estimate to this controller while maintaining the platform's agility, only one additional sensor, an airspeed sensor placed near the platform's body is required. Since this sensor location is impacted by propeller slipstream, a model-based approach to removing slipstream flow from airspeed measurements is used as part of a larger wind estimation architecture. Simulations and experimental flights were conducted to validate both the feedforward controller and model-based slipstream removal.

Abrégé

Les véhicules aériens sans pilote (UAV) à voilure fixe agiles sont une classe émergente d'avions autonomes qui combinent le vol d'avancement efficace des voilures fixes conventionnelles avec la maniabilité et la capacité de vol stationnaire des plates-formes multirotor. Ces caractéristiques font des voilures fixes agiles des candidats convaincants pour une grande variété de missions, tels le sauvetage et la surveillance. La maniabilité remarquable de ces plates-formes est en grande partie due à leur faible poids et à leurs propulseurs puissants, qui entraînent un flux d'air en sillage fiable et rapide sur les surfaces de commande, les gardant efficaces pendant le vol à basse vitesse. Malgré que leur faible poids soit essentiel à l'agilité de cette classe d'UAV, cela les rend également très sensibles au vent et peut présenter un défi pour un vol de précision à l'extérieur.

L'objectif de cette thèse est de rejeter les perturbations du vent sur ces plates-formes tout en minimisant la réduction en performance de vol. Pour atteindre cet objectif, un précompensateur modulaire de rejet du vent, reposant sur une estimation du vent, a été développé pour améliorer la performances des correcteurs agiles pour les avions à voilure fixe en environnement extérieur. Pour fournir une estimation du vent à ce contrôleur tout en conservant l'agilité de la plate-forme, un seul capteur supplémentaire, un anémomètre placé près du corps de la plate-forme, est nécessaire. Étant donné que le placement de ce capteur est influencé par le sillage de l'hélice, une approche basée sur un modèle du flux de sillage est utilisée dans le cadre d'une architecture d'estimation du vent pour supprimer les effets sur les mesures de vitesse anémométrique. Des simulations et des vols expérimentaux ont été effectués pour valider à la fois le contrôleur prédictif et l'estimation du vent basée sur le modèle de sillage de l'hélice.

Acknowledgements

First and foremost, I would like to thank my supervisor, Professor Meyer Nahon. This work would not have been possible without his careful guidance and thoughtful insight.

I would also like to thank Professor Jovan Nedić for the use of the venturi constriction calibration tool, used in this thesis for calibration of airspeed sensors.

Next, I would like to thank all of my labmates at the Aerospace Mechatronics Lab, whether it was going on our near-daily bagel run, or getting beaten at ultimate frisbee, I enjoyed every moment I spent with you all, you made my experience at McGill what it was. I would like to especially thank: Juan Carlos Hernández Ramírez for always pressing me to remain optimistic even when flight tests went wrong, Jad Wehbeh for his support and help, Zihao Zhuo, and Eitan Bulka.

I would like to acknowledge my parents Peter Empey and Betty-Ann Franssen, and my sister Kitt Empey who have always provided endless encouragement throughout all my academic pursuits. I would also like to thank my girlfriend Harriet Chorney, for her ever-present compassion and support. Lastly, this work could not have been completed without the support of my friends, I don't think I can express how much of a positive impact you all have had on me, but I am incredibly grateful. In particular, I would like to thank Jack Williamson, Zachary May, and Liam O'Sullivan, for getting me through the COVID-19 lockdowns.

Contents

Abstract i
Abrégé ii
Acknowledgements iii

Contents **iv**

List of Figures **vii**

List of Tables **xiii**

1 Introduction **1**
1.1 Background 1
1.2 Literature review 3
1.2.1 Wind rejection 4
1.2.2 Wind estimation 5
1.2.3 Propeller slipstream 7
1.3 Thesis Objectives 9
1.4 Thesis Organization 9

2 Propeller slipstream **10**
2.1 Propeller slipstream model 10
2.1.1 Near-field model (momentum theory) 12
2.1.2 Efflux plane (near-field to far-field transition) 13
2.1.3 Far-field model (semi-empirical) 14
2.2 Effect of freestream velocity on slipstream 17
2.2.1 Thruster model 18
2.2.2 Calculation of induced flow at the propeller plane 20
2.2.3 Total flow from induced flow 20

2.2.4 Effect of freestream on slipstream . . . 21
2.3 In-flight validation of the propeller slipstream model . . . 23
2.3.1 Methodology . . . 24
2.3.2 Sensors . . . 24
2.3.3 Flight profile . . . 25
2.3.4 Measurement locations . . . 26
2.4 Results . . . 27
2.4.1 Constant throttle tests . . . 28
2.4.2 Throttle variation tests . . . 28
2.4.3 Discussion . . . 28

3 Synthetic airspeed sensing 33
3.1 Introduction . . . 33
3.2 Preliminaries . . . 35
3.2.1 Proposed estimation architecture . . . 35
3.2.2 Effect of oblique flow conditions on pitot tubes . . . 36
3.2.3 Extended Kalman Filter . . . 38
3.3 Observability . . . 40
3.4 Synthetic airspeed sensor . . . 42
3.4.1 Extended Kalman Filter . . . 42
3.5 Results . . . 44
3.5.1 Offline testing . . . 45
3.5.2 Experiment . . . 47
3.6 Use of slipstream measurement in control allocation . . . 50

4 Wind rejection 54
4.1 Feedback position control . . . 55
4.1.1 Vehicle dynamics . . . 56
4.1.2 Position controller . . . 56
4.1.3 Attitude controller . . . 58
4.2 Feedforward wind rejection controller . . . 58
4.2.1 Attitude feedforward . . . 60
4.2.2 Thrust feedforward I . . . 61
4.2.3 Thrust feedforward II . . . 63
4.3 Simulation results . . . 64

4.3.1 Straight and level tracking . . . 66
4.3.2 Circular tracking . . . 68
4.3.3 Frequency tests . . . 71
4.3.4 Wind rejection in 3-dimensions . . . 75
4.4 Experiments . . . 77
4.4.1 Direct freestream measurement . . . 78
4.4.2 Synthetic airspeed sensor . . . 79

5 Conclusion **84**
5.1 Summary of Conclusions . . . 84
5.2 Suggestions for future work . . . 85

References **88**

A Sensor validation and calibration **95**
A.1 Airspeed sensors . . . 95
A.1.1 Airspeed reference . . . 95
A.1.2 MS4525DO validation . . . 97
A.1.3 SDP33 calibration . . . 98
A.1.4 Airspeed sensor results . . . 100
A.2 RPM sensor . . . 103

B PX4 implementation of synthetic airspeed sensor **105**

C Exponential filter synthetic airspeed sensor **109**

List of Figures

1.1 Multi-rotor UAV 2
1.2 Fixed-wing UAV 2
1.3 Agile fixed-wing UAV 3

2.1 Propeller frame definition showing axial and radial directions $\hat{x}$ and $\hat{r}$ respectively and the test point location $\underline{P}_d$ with components x_d and r_d. 11
2.2 Induced flow ratio $\gamma(x)$ as function of normalized axial distance from the propeller plane. 13
2.3 Flow velocity distribution in the slipstream far-field, showing parameters $V_{max}(x)$ and $R_{max}(x)$ 15
2.4 Half cross-section of static slipstream flow showing near-field, far-field, far-field sections and efflux plane. Propeller conditions are $V_\infty = 0\frac{m}{s}$, $8700\ RPM$ 18
2.5 Thrust coefficient C_t used in this work as a function of advance ratio J . . . 19
2.6 Slipstream changing with freestream speed. $n_{rpm} = 8700\ RPM$, $V_\infty = 0, 5, 10, 15$ m/s 22
2.7 Slipstream model results of maximum induced flow, and flow velocity versus axial distance for a propeller operating at 8700 RPM with differing freestream speeds. The top figure shows the static case, the middle figure shows the case where $V_\infty = 5\frac{m}{s}$, and the bottom figure shows the case where $V_\infty = 15\frac{m}{s}$. In the top subfigure, V_{max} and w_{max} are equal. 23
2.8 Experimental platform used for flight tests, showing the freestream airspeed sensor placed on the starboard wingtip and the slipstream sensor locations. 24
2.9 Pitot probe 'cradle' mounts. 27
2.10 Airfame mounting point showing differential pressure sensor and tubing. The test location shown here is C2. 27

2.11 Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location A1. 29
2.12 Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location B1. 29
2.13 Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location C1. 29
2.14 Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location C2. 29
2.15 Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location D1. 30
2.16 Root mean squared errors of the semi-empirical model and classical momentum theory model for the 10-second level flight segments shown in Figures 2.11 through 2.15 30
2.17 Slipstream velocity recordings and model estimates at test point A1 during rapid throttle change (top), and RPM measurements (bottom). 30
2.18 Slipstream velocity recordings and model estimates at test point B1 during rapid throttle change (top), and RPM measurements (bottom). 30
2.19 Slipstream velocity recordings and model estimates at test point C1 during rapid throttle change (top), and RPM measurements (bottom). 31
2.20 Slipstream velocity recordings and model estimates at test point C2 during rapid throttle change (top), and RPM measurements (bottom). 31
2.21 Slipstream velocity recordings and model estimates at test point D1 during rapid throttle change (top), and RPM measurements (bottom). 31
2.22 Root mean squared errors of the semi-empirical and classical momentum theory models for throttle variation flight segments shown in figures 2.19 through 2.21 31

3.1 Simulation of static slipstream over agile fixed-wing platform. 34
3.2 Block diagram overview of proposed estimator architecture. The airspeed sensor measures v_d inside the slipstream, which is a combination of induced propeller flow and freestream. The synthetic airspeed sensor extracts the freestream component of this measurement, and sends it to a conventional wind estimator. While not shown, both the synthetic airspeed sensor and wind estimator also rely on groundspeed and attitude states. . . 36
3.3 Pitot-static probe diagram 37

3.4 Observability of v_∞ inside the slipstream at a variety of propeller speeds. The observability metric shown here is the average value for c, $\bar{c}$. A black triangle marks the location $\underline{P}_d = [0.62\ 0.12]\ \mathrm{m}$. 41
3.5 Agile fixed-wing UAV instrumented with wingtip and tail airspeed sensors measuring freestream and slipstream respectively 45
3.6 The top plot shows measured v_∞ and v_d from outdoor flight test, compared to v_∞ estimated using the EKF based synthetic airspeed sensor. The bottom plot shows the estimated variance. 47
3.7 Estimated wind magnitude and direction using the traditional wind estimator (blue), and the cascaded wind estimator (orange). The traditional wind estimator relies on the measured v_∞, whereas the cascaded wind estimator uses the EKF based synthetic wind estimator to estimate v_∞. 48
3.8 Experimental platform configuration for on-line wind estimation flight test, a single sensor, near the platform's tail, at $\underline{P}_d = [0.62\ 0.12]\ \mathrm{m}$, is used. 49
3.9 Slipstream measurement and freestream estimate for online cascaded wind estimation flight test. The platform is launched at $t \sim 0$ 50
3.10 Wind estimate results for online cascaded wind estimation flight test. The platform is launched at $t \sim 0$ 50
3.11 Pitch reference and measurements for the ATA maneuver using momentum theory control allocation and slipstream measurement control allocation. 52
3.12 Roll reference and measurements for the ATA maneuver using momentum theory control allocation and slipstream measurement control allocation. . . 52
3.13 Experimental result of ATA maneuver using direct slipstream measurement in the control allocation 53
3.14 Experimental result of ATA maneuver using momentum theory control allocation 53

4.1 Inertial, $\mathcal{F}_i$, and body, $\mathcal{F}_b$, frame definitions. The inertial frame vectors are aligned with the North, East, and Down vectors respectively. 56
4.2 Block diagram of feedforward/feedback controller. Feedback elements are shown blue and feedforward elements are shown in green. 59
4.3 Wing triangle definition 60
4.4 T_{ff_1} compensating for the change in thrust vector due to the attitude feedforward 62
4.5 Visualization of on-track and cross-track errors, $e_{\|}$ and $e_{\perp}$ respectively. . . . 65

4.6 Visualizations of wind directions examined in straight and level simulations where the platform is commanded to track the black dashed line. Vectors point along the wind direction for each test. 67
4.7 Path relative errors for a series of straight and level $10\ \mathrm{m/s}$ trajectory tracking cases with varying wind directions, with a mean wind of $5\ \mathrm{m/s}$ beginning at $t = 5\ \mathrm{s}$. 67
4.8 Error along path for a series of straight and level $10\ \mathrm{m/s}$ trajectory tracking cases with varying wind directions, with a mean wind of $5\ \mathrm{m/s}$ beginning at $t = 5\ \mathrm{s}$. 67
4.9 Straight and level tracking performance of the feedforward controller with $\boldsymbol{k_\gamma = 0}$ in a $5\ \mathrm{m/s}$ wind field. 69
4.10 Straight and level tracking performance of the feedforward controller with $\boldsymbol{k_\gamma = 0.5}$ in a $5\ \mathrm{m/s}$ wind field. 69
4.11 Straight and level tracking performance of the feedforward controller with $\boldsymbol{k_\gamma = 2}$ in a $5\ \mathrm{m/s}$ wind field. 69
4.12 Straight and level tracking performance of the feedforward controller with $\boldsymbol{k_\gamma = 3}$ in a $5\ \mathrm{m/s}$ wind field. 69
4.13 Simulated flight of base-controller and feedforward tracking a $30\ \mathrm{m}$ radius circle at a $10\ \mathrm{m/s}$ ground speed in a uniform $5\ \mathrm{m/s}$ wind field with a steady Eastern direction. The reference trajectory in this figure travels clockwise. . 70
4.14 Sample data segment from the oscillating wind test where $f = 0.1\ \mathrm{Hz}$, amplitude and mean of base-controller and feedforward $e_\perp$ signals are shown. 72
4.15 $e_\perp$ as a function of wind frequency for the magnitude variation simulation configuration in Table 4.3 73
4.16 $e_\parallel$ as a function of wind frequency for the magnitude variation simulation configuration in Table 4.3 73
4.17 $e_\perp$ as a function of wind frequency for the direction variation simulation configuration in Table 4.4 75
4.18 $e_\parallel$ as a function of wind frequency for the direction variation simulation configuration in Table 4.4 75

4.19 Simulation tracking a circular trajectory under a wind with an eastern component and a downdraft component. The top plot shows trajectories in the North-East plane, and the middle plot shows the vertical errors. The bottom plot shows the recorded roll angle. The platform begins on the left side of the figure and travels right, the circular reference rotates clockwise. . . . 76
4.20 Sample of wind rejection setup showing airspeed sensor location 78
4.21 The wind estimate for experimental flight test. A dashed line separates wind estimates logged before the feedforward elements are disabled. The wind direction is defined similar to a compass rose, but with the y-axis replacing North and the x-axis replacing East. 79
4.22 Experimental flight test tracking a $25\ \mathrm{m}$ radius circle at $10\ \mathrm{m/s}$ with feedforward enabled. Average wind direction and magnitude are shown with the black arrow and text respectively. 80
4.23 Experimental flight test tracking a $25\ \mathrm{m}$ radius circle at $10\ \mathrm{m/s}$ using only the base-controller. Average wind direction and magnitude are shown with the black arrow and text respectively. 80
4.24 Racetrack reference trajectory. 80
4.25 Wind rejection setup showing airspeed senor location for use with synthetic airspeed sensor 81
4.26 Flight test results tracking a racetrack trajectory with feedforward wind rejection and wind estimation using a synthetic airspeed sensor. The average wind direction and magnitude estimated during the flight test is shown with a black arrow and black text respectively. 82
4.27 Wind estimate during racetrack flight test using the synthetic airspeed sensor. The wind direction is again defined similar to a compass rose, but with the y-axis replacing North and the x-axis replacing East. 82

A.1 Schematic of the airspeed source used for sensor calibration. D_{dwn} represents the tube diameter at the downstream pressure tap and D_{up} represents the tube diameter at the upstream pressure tap. 96
A.2 Venturi constriction device being used to validate the MS4525 sensor. 97
A.3 $\beta(\Delta p_{sensor})$ measurements compared with Δp_{sensor}, used for calibration of the SDP33 airspeed sensor 100

A.4 Validation tests results for the calibrated SDP33 sensor. Green and blue lines indicate the bounds on the airspeed reference and the dashed black line indicates the average measurement from the airspeed sensor. 102
A.5 Validation tests results for the MS4525DO sensor. Red and yellow lines indicate the bounds on the airspeed reference and the dashed black line indicates the average measurement from the airspeed sensor. 102
A.6 Standard deviation of airspeed from pitot sensors as a function of average calculated airspeed they record. Results are shown for the MS4525DO based airspeed sensor and the SDP33 based airspeed sensor. 102
A.7 Kotleta20 RPM validation test showing ESC measurements and optical measurements (top). The error between these two measurements is shown in the bottom subfigure. 104

B.1 High-level overview of cascaded estimator as it is implemented in PX4. Components unchanged from PX4 are coloured blue and our additions are highlighted in red. 106

C.1 Relationship between $\alpha(\Phi),\ \alpha_{min},\ \alpha_{max},\ \text{and},\ \Phi_{\min}$ for Eq. (C.5). 111
C.2 The top plot shows measured v_∞ and v_d from outdoor flight test, compared to estimated freestream using the EF-SAS. The bottom plot shows the value of α; spikes in α attempt to reject inaccurate airspeed measurements under oblique flow conditions. 112
C.3 Estimated wind magnitude and direction using the traditional wind estimator (blue), and the cascaded wind estimator (orange). The traditional wind estimator relies on the measured v_∞, whereas the cascaded wind estimator uses the EF-SAS to estimate v_∞. 113

List of Tables

2.1 Axial boundaries of sections in the far-field model in terms of efflux plane location and diameter 16
2.2 Table of coefficients used in the semi-empirical slipstream model 17
2.3 Test point locations 26

3.1 Cascaded wind estimator configuration parameters used in offline EKF-SAS testing. 46

4.1 Feedforward configurations table used simulation tests 65
4.2 RMS of path relative errors for four controller configurations in simulated tracking of a $30\ \mathrm{m}$ radius circle at $10\ \mathrm{m/s}$ in a $5\ \mathrm{m/s}$ wind-field. 71
4.3 Magnitude variation test configuration 72
4.4 Direction variation test configuration 74

A.1 True airspeed source data for the SDP33 validation test shown in Figure A.4 101
A.2 True airspeed source data for the MS4525DO validation test shown in Figure A.5 101

C.1 Cascaded wind estimator configuration parameters used in offline EF-SAS testing. 112

Chapter 1

Introduction

1.1 Background

Interest in unmanned aerial vehicles (UAVs) has grown sharply in recent years with autonomous airborne platforms being developed for a wide variety of missions, from aiding in search and rescue, to inspecting construction sites, or even keeping technicians away from harm by monitoring infrastructure in dangerous environments [1].

Several classes of UAV exist, and are specialized for different applications. One popular class is the multi-rotor, shown in Figure 1.1, these platforms fly using only thrusters without additional lift from aero-surfaces. Multi-rotors can hover in place, maneuver in tight spaces, and perform impressive feats of agility. These characteristics make them excellent performers for missions covering a small area where only relatively low speed flight is required. To some degree, the antithesis of the multi-rotor is the fixed-wing UAV platform, shown in Figure 1.2. Conventional fixed-wing UAVs rely on wings for lift, meaning they must maintain a certain airspeed to remain airborne, and as such they often require runways for takeoff and landing and, of course, cannot hover. Once these platforms are aloft however, they can fly quickly and efficiently to a destination, making them ideal for missions covering a large area, where hover is not required [2].

Figure 1.1: Multi-rotor UAV

Source: [3]

Figure 1.2: Fixed-wing UAV

Source: [4]

One UAV class, the agile fixed-wing, shown in Figure 1.3, takes inspiration from both multi-rotor and conventional fixed-wing platforms to merge the best performance characteristics of both. These fixed-wings are light weight, with large control surfaces capable of large deflections, and powerful thrusters, making them capable of agile maneuvers comparable to multi-rotors [5]. Additionally, they can hover by pointing their thruster upward and using their strong propeller flow to keep their control surfaces effective, even when stationary. They are also capable of relatively efficient distance flight when flying similarly to a conventional fixed-wing.

These platforms originate from indoor aerobatic competitions, where they are operated by pilots over remote-control. Recent work has focused on automating the agile fixed-wing UAVs by first studying their complex dynamics [6], and then proceeding to design aerobatics-capable controllers for the platform [7]. With their agility and efficient distance flight, these platforms may be ideal for missions requiring low-speed or stationary inspection of multiple targets which are a long distance apart. An agile fixed-wing could fly to a target, stop and hover, carry out a task, and then fly quickly and efficiently to the next target.

Figure 1.3: Agile fixed-wing UAV

Lightweight platforms such as agile fixed-wings are, however, highly susceptible to wind disturbances [8]. While this is not an issue for indoor flight, it may be problematic as agile fixed-wings are increasingly used for outdoor autonomous missions. As such, compensating for wind disturbances on agile fixed-wing UAVs is a worthwhile problem to investigate, and is the topic of this thesis.

1.2 Literature review

We now present a literature review covering topics relevant to this thesis. The review is partitioned into three sections focusing on the three major topics of this work: wind rejection, wind estimation, and propeller slipstream modelling.

These topics may initially seem somewhat isolated, however, we will show in the remainder of this section how they are deeply interconnected and naturally lead to each other from the central problem of this thesis, wind disturbance rejection for agile fixed-wings. The logical starting point to investigate this problem begins with the wind rejection topic, and studying how a controller should be modified to account for the effects of wind. From this investigation, we find that wind estimation will likely be required since many wind rejection approaches use explicit wind knowledge. Furthermore, many wind

estimation approaches require some airspeed measurement which, as is discussed in Section 1.2.2, can be challenging on platforms strongly influenced by propeller slipstream, hence the investigation and modelling of slipstream flow.

1.2.1 Wind rejection

The approach used to reject wind on UAVs varies substantially across platforms and controllers. For multi-rotor platforms, disturbance accommodating control has been demonstrated experimentally to reject wind in [9], and Non-Linear Dynamic Inversion (NLDI) control augmented with an artificial neural network is also shown to be effective at wind rejection in [10].

Tailsitter platforms can be especially susceptible to wind disturbances when in their hover state, since they often expose their entire wing planform to the wind [11]. Rejection of wind on an agile platform in hover is studied in [12], where the authors design a disturbance observer based-controller using $\mathcal{H}_\infty$ synthesis, validating the controller experimentally indoors with artificial wind.

Wind rejection for fixed-wing UAVs is well studied, and different approaches are used depending on the control objective. One objective, path-following, has seen a large amount of interest for wind rejection. In path-following, the platform is commanded to track a geometric path with success often measured by cross-track error, the distance between the platform and the reference path [13]. Wind rejection for path following has been studied in [14–18]. Both [17] and [14] conduct path-following using a Lyapanov Guidance Vector Field (LGVF); to reject wind disturbances, they augment the vector field using a wind estimate. Cohen and Forbes [15] design an Invariant Extended Kalman Filter within a matrix Lie group framework for the estimation of wind and vehicle states on a tailsitter platform. The authors then use their wind estimate to improve path following performance by incorporating it into their attitude control and control allocation layers. In the attitude control layer, the wind estimate is used to properly coordinate a balanced turn according to a reference yaw rate, regardless of the platform's heading relative to the wind. In the control allocation layer, the wind estimate is used to more accurately calculate the authority of the control surfaces. In [16], the authors reject wind disturbances for a path-following controller by using a wind estimate to calculate pitch and yaw trim angles, which they then use directly in their guidance law. Finally, Pfeifle and Fichter [18] use an incremental-NLDI controller which natively tracks aerodynamic angles in the attitude layer to provide wind rejection qualities without the need for a separate wind estimator.

Wind disturbance rejection for altitude and airspeed tracking has also been investigated. The authors of [19] pair a NLDI controller with a disturbance observer to track these states in a windy environment. In [20], altitude and airspeed are tracked using a non-linear model predictive (NMPC) controller. Using a wind estimator and supplying the result to the NMPC controller for use in the model was shown to reject wind disturbances in simulation. Lu et al. [21] track sideslip and roll in addition to airspeed and altitude, using a disturbance observer and back-stepping controller to reject both wind and the disturbance from unmodeled aerodynamic effects.

In [22], Brezoescu et al. study cross-wind disturbance rejection for a trajectory following controller. The authors design an adaptive controller and show its ability to reject disturbances from an unknown wind field through simulation.

Since a large portion of the wind disturbance rejection approaches discussed above require a wind estimate, we also conduct a review surveying wind rejection techniques for fixed-wing UAVs.

1.2.2 Wind estimation

To aid in wind measurement and estimation, conventional large-scale aircraft can take advantage of multi-port pitot tubes and vane-type sensors to measure directional airspeed, angle of attack (AoA) and angle of side-slip (AoS) [14] [23]. The weight of these sensors is negligible when compared to the mass of a large aircraft, so their addition does not generally impact flight performance. For small UAVs however, below about $4\ \mathrm{kg}$, this additional mass is no longer negligible, providing limited options for wind estimation [24]. This restriction, along with the increased research interest in lightweight UAVs, has created a drive to estimate wind with the limited range of sensors included on these platforms. The limited sensor suite usually includes: IMU, GPS, and an airspeed sensor [23], research on wind estimation with this sensor package is ongoing, but has progressed far enough to migrate to popular autopilot platforms such as the PX4 [25]. Wind estimation solutions are platform and mission-dependant, as an example, reliance on airspeed data from a pitot tube may work well for a UAV in level flight [26], but is less effective when the aircraft pitches up to perform a hover maneuver [24].

A large body of work exists for on-line wind estimation on lightweight conventional fixed-wing UAVs, which generally do not perform aggressive maneuvering at high angles of attack and sideslip. Some of these methods are briefly described below.

In Rautenberg et al. [26] three wind estimation methods are investigated. The first of these collects data from a single port pitot probe, IMU, and GPS over time and estimates wind velocity and direction with a least squares method. Secondly, the authors present a 'no-flow' sensor algorithm which assumes a constant airspeed and windspeed, wind observability comes through the change in groundspeed as the platform changes heading. Lastly, a five-port directional pitot probe is used to calculate the airspeed vector in the platform's body frame which, when paired with GPS measurements, allows calculation of the wind vector. Experimental flight tests were conducted by the authors, with results showing similar performance between the single and multi-port wind estimation methods, and lower performance from the 'no-flow' algorithm. Chan et al. [8] also compare three wind estimation approaches, but instead focus on Kalman Filter frameworks. They study wind estimation using an Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and finally direct calculation of windspeed from three-axis airspeed measurements. The authors perform experimental flight tests and show similar performance between the EKF and UKF, with the direct calculation showing the worst noise performance. While all three methods predict windspeeds that are relatively consistent with ground truth measurements, this work does require the UAV have onboard AoA and AoS sensors. In [27], Johansen et al. use a kinematics-based Kalman Filter fusing: GPS, IMU, and airspeed measurements to estimate wind velocity, angle of attack and sideslip. The authors achieve good convergence in experimental flight tests on multiple platforms. Brezoescu et al. [28] studied wind estimation for fixed-wing UAVs using lateral, 'vehicle response', models of the UAV along with position, and velocity measurements through GPS. On-line parameter estimation is used to augment performance, and simulation tests are used to validate the wind estimator.

Since the airspeed sensors often used on light-weight UAVs are uni-directional, meaning they can only measure the component of air flowing directly into the sensor, wind estimation for hovering UAVs can be challenging. This is particularly true for platforms such as tailsitters or agile fixed-wings which hover vertically and transition to a horizontal state for level flight. If an airspeed sensor is mounted along the platform's longitudinal axis, it may provide good airspeed measurements in level flight but will be highly inaccurate when the aircraft transitions to hover. Estimating wind on these hovering platforms is an active area of research.

In [11], the authors utilize a 'gray-box' dynamics model of their tailsitter platform with a Kalman Filter framework to estimate wind in a hover state. The dynamics model

is tuned in a zero-wind environment using parameter identification, and the authors validate their estimator in experimental flight tests. Sun et al. [14] also use a dynamics model to estimate wind for a tailsitter platform in hover, developing a 'synthetic airspeed sensor' to estimate airspeed when the platform is in hover, synthetic airspeed was then used to estimate the wind velocity. Their platform also includes an airspeed sensor mounted along the platform's longitudinal axis to measure airspeed in level flight. When in level flight, wind is primarily estimated using the conventional airspeed sensor, and when in hover, the synthetic sensor is used, enabling wind estimation for all flight states of the agile platform.

A large majority of wind estimation methods, including those described above, require some airspeed measurement to function. Typically, this measurement must be unaffected by the aerodynamics of the platform and should be an accurate representation of the velocity that the platform is moving with respect to a coordinate system fixed to the surrounding air. We will denote this ideal measurement as a freestream measurement.

For many fixed-wing UAV platforms, particularly larger ones, freestream measurements are easy to obtain, as most of the platform's airframe is relatively free of disturbances to the freestream and even if a section of the airframe is disturbed, by the propeller for example, many other placement locations are available with minimal impacts on performance. This is not true for agile fixed-wings however, which face two opposing issues that make freestream measurement especially challenging. Firstly, due to their powerful thrusters, most of the airframe is covered in fast moving slipstream flow from the propeller, which can cause airspeed measurements on the body of the platform to be much higher than the true freestream. The wingtips of the platform are void of slipstream flow, making them theoretically ideal for freestream measurement, however on platforms as light as agile-fixed wings placing even light airspeed sensors this far away from the centre of mass can greatly impact agility. Since it is clear that propeller slipstream will impact wind estimation on agile fixed-wings, we conduct a literature review to survey previous work on this type of flow.

1.2.3 Propeller slipstream

The high speed flow behind propellers, or propeller slipstream, enables many of the incredible maneuvers that agile UAV platforms are capable of, but it is also a region of complex aerodynamics with research still ongoing to fully characterize this type of flow.

The flow close behind a propeller has been studied in the literature, as this is a region of particular interest for wing-mounted propellers. Classical approaches, such as momentum theory and a coupling of blade element and lifting line theories, have been shown to provide useful estimates of slipstream in this region [29] [30]. Farther downstream, these methods are expected to become less accurate, as the slipstream diffuses and mixes with the freestream air.

Particle Image Velocimetry (PIV), has been used to conduct detailed surveys of propeller slipstream in [31] and [32] and records the complex vortical structures that are present in this region. Computational Fluid Dynamics (CFD) is also used to examine flow inside the slipstream region, such as in [33] which studies the influence of slipstream on a twin turboprop aircraft. The authors of [31] also develop a CFD model to investigate the interaction between slipstream and the wing in [34], validating their model using PIV measurements.

Wind tunnel slipstream measurements behind a variety of small UAV propellers are recorded by Deters and Selig in [35]. These measurements are taken through a large axial and radial sweep of the slipstream, with and without interference from a flat-plate airfoil. A similar study is carried out by the Leng et al. in [36], where the authors focus on slipstream-wing interference at a variety of advancing flow conditions.

An adjacent field, marine propulsion, has studied propeller slipstream extensively. The powerful slipstream behind ship propellers is the primary factor of accelerated erosion effecting underwater structures in ports, hence the large research interest in investigating this flow [37]. A common, and suitably accurate, approach for modelling marine slipstream employs a semi-empirical model to capture the complex diffusion effects far downstream of the propeller. These semi-empirical models usually assume that the flow velocity downstream of the propeller follows a Gaussian profile in the radial direction at a given distance along the propeller's rotational axis. The shape of the Gaussian profile is controlled by parameters that are fit through extensive experimental measurements [38]. While most work on marine slipstream considers static conditions, semi-empirical propeller models have been adapted for use with marine turbines powered by advancing flow in [39] and [40].

For propellers operating in air, recent work by Khan and Nahon in [41] uses classical momentum theory to model slipstream in the region close to the propeller, and an adaptation of the semi-empirical methods used for marine propellers to model the region far downstream. The authors report good accuracy for this model after an extensive se-

ries of measurements throughout the slipstream while the propeller is operating in static conditions.

1.3 Thesis Objectives

The goal of this thesis is to reject wind disturbances that affect agile fixed-wing aircraft in outdoor flight.
The approach used in this thesis rejects wind using an explicit wind estimate, similar to the architecture used by many works reviewed in Section 1.2.1. This decomposes wind disturbance rejection into two coupled problems, wind estimation and wind rejection. Due to the unusual aerodynamics of agile fixed-wings and the advanced controllers that are designed to automate them, wind disturbance rejection for this class of UAV presents unique and interesting challenges for both the wind estimation and rejection problems. Wind estimation is challenging on agile fixed-wing platforms due to propeller slipstream, as such this thesis also includes work on the modelling of validation of this flow.

Key objectives of this thesis are as follows:

1. Modelling and validation of propeller slipstream in advancing flow conditions.
2. Wind estimation on platforms where propeller slipstream covers a large portion of the airframe.
3. Feedforward wind rejection using an explicit wind estimate that can be integrated with acrobatic controllers already developed for agile fixed-wing UAVs.

1.4 Thesis Organization

This thesis is composed of three main sections, each covering a key objective, beginning with Chapter 2. Chapter 2 is focused on propeller slipstream modelling and in-flight validation. This model is then used extensively in Chapter 3 for wind estimation. Chapter 4 uses this wind estimate for wind rejection. Chapter 5 concludes this thesis and suggests topics for future study.

Each chapter contains simulation and experimental results specific to its topic, with Chapter 4 presenting final results utilizing the work developed in Chapters 2 though 4.

Chapter 2

Propeller slipstream

This section will discuss propeller slipstream modelling on an agile fixed-wing UAV in forward flight conditions. Validation of the model is performed in-flight using an instrumented agile fixed-wing platform.

The flight dynamics of agile fixed-wings can be largely dominated by propeller slipstream. Slipstream often drives a majority of flow over the platform's surfaces, enabling control authority at low speeds or while hovering. As such, propeller slipstream models are often used as part of a larger vehicle dynamics model, such as in [6]. In this thesis, we instead look to use the slipstream model to conduct wind estimation using an airspeed sensor submerged in slipstream, which is the focus of Chapter 3. Moreover, while other work considers slipstream in a static case, analogous to hover, it is critical that we model and validate slipstream in forward flight conditions.

This chapter will first introduce a propeller slipstream model developed for static conditions in prior work, then extend this model to forward flight conditions. The effect of forward flight on the slipstream will be investigated using the model, and finally validation in forward flight will be done using an instrumented vehicle.

2.1 Propeller slipstream model

We begin by introducing the base propeller slipstream model used in this work, which was developed by Khan and Nahon in [41]. This model propagates an induced flow velocity w_p at the propeller plane to some location downstream, denoted $\underline{P}_d$. The propeller reference frame used in this work, along with $\underline{P}_d$, can be seen in Figure 2.1. $\underline{P}_d$ has orthogonal components x_d and r_d. As Figure 2.1 indicates, the radial direction, along $\hat{r}$, is

a vector normal to the propeller's rotational axis, and the axial direction, $\hat{x}$, is along the propeller's rotational axis. Slipstream is often assumed to be symmetric about $\hat{x}$, hence the use of only two components to localize a position inside the slipstream.

With regard to notation, in this thesis, vectors are denoted with an underbar as $\underline{(\cdot)}$, unit-length vectors are denoted with the standard $\hat{(\cdot)}$ notation and neglect the underbar, and matrices are generally indicated using bold type.

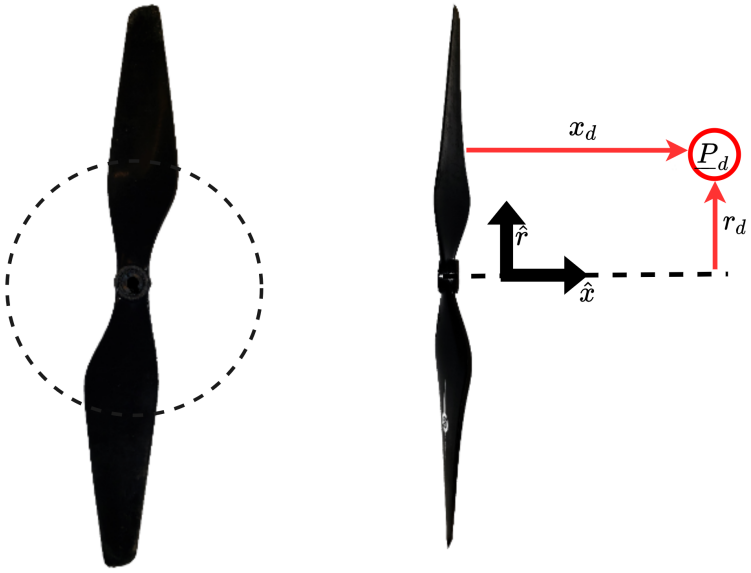

Figure 2.1: Propeller frame definition showing axial and radial directions $\hat{x}$ and $\hat{r}$ respectively and the test point location $\underline{P}_d$ with components x_d and r_d.

Before discussing the propeller slipstream model, it is worthwhile to briefly mention the meaning of induced flow in the context of slipstream. Classical slipstream models, such as momentum theory, evaluate induced flow, which is the difference between the flow velocity at some location, and the flow velocity far upstream. The downstream propagation of an induced flow velocity from the propeller plane, consists of analyzing the increase in flow speed compared to the freestream, and how this changes with axial distance away from the propeller. The induced flow velocity itself is not necessarily useful for our purposes here, and as such it needs to be superimposed with the freestream velocity to compute the true flow speed. The true flow speed is what one would measure using an ideal airspeed sensor fixed to the aircraft at the given location.

The slipstream model used in this thesis, developed in [41], uses classical momentum theory to model the flow in a region close to the propeller, and a semi-empirical model adapted from work on marine propeller slipstream to model the region far away from

the propeller plane. Classical momentum theory assumes that flow is driven by acceleration forces, gradually and indefinitely contracting as it travels downstream. While this assumption can work reasonably well for flow near the propeller plane, its accuracy will eventually fade as the slipstream diffuses and expands, which is not modelled by momentum theory. The region where momentum theory is valid, due to a lack of flow diffusion, is labelled the 'near-field'. The 'far-field' region is downstream of this and is characterized by the diffusion of slipstream in the freestream air. A plane normal to $\hat{x}$ represents the transition from near-field to far-field, with the exact location of this transition being established later in this section. The remainder of this section will provide an overview of the model developed in [41], the modifications which enable this model to be used for forward flight will be presented in the following section.

2.1.1 Near-field model (momentum theory)

Classical momentum theory assumes that a propeller can be modelled as an actuator disk, a thin plane which has a pressure discontinuity across it with no discontinuity of flow velocity [42]. Momentum theory also assumes incompressible flow and neglects swirl, with swirl being flow normal to both $\hat{r}$ and $\hat{x}$. Since swirl is not considered, mixing and diffusion are likewise unmodeled, hence the indefinite contraction of the slipstream.

McCormick discusses propagation of an induced flow at the propeller plane, w_p, to a location x downstream using momentum theory in [43], and introduces an equation to calculate induced flow w as a function of axial distance, using R_p which is the radius of the propeller. McCormick's equation is decomposed below into induced flow velocity as a function of $w(x)$ and a parameter $\gamma(x)$, which represents the ratio of induced flow at the propeller plane to induced flow at axial location x.

$$w_x = w(x) = w_p \gamma(x) \tag{2.1}$$

$$\gamma(x) = \left[1 + \frac{x/R_p}{\sqrt{1 + (x/R_p)^2}} \right] \tag{2.2}$$

As can be in Eq. (2.2), $\gamma(\infty) = 2$ and $\gamma(-\infty) = 0$, which implies that far upstream of the propeller no additional flow velocity is induced and, less intuitively, infinitely far downstream the induced flow is double that at the propeller plane. This implies that half of the induced velocity caused by the propeller is generated upstream of the pro-

peller plane, and the other half is generated downstream. The change from $\gamma(-\infty) = 0$ to $\gamma(\infty) = 2$ happens in a very localized region near the propeller, with the vast majority of flow development occurring within a five propeller diameter long axial range. A visualization of $\gamma(x)$ as a function of normalized axial distance from the propeller plane is shown in Figure 2.2.

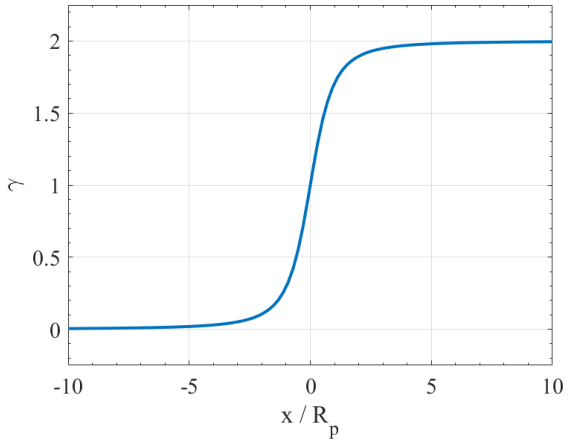

Figure 2.2: Induced flow ratio $\gamma(x)$ as function of normalized axial distance from the propeller plane.

It should also be noted that Eq. (2.1) is not a function of radial distance, implying a uniform slipstream velocity in $\hat{r}$ at a given location x. The diameter of the slipstream, D_s, is then calculated through conservation of mass, assuming that, at $x = 0$, $D_s = D_p$, where D_p is the diameter of the propeller. D_s can be calculated as a function of axial distance as

$$D_s = D_p \sqrt{\frac{1}{\gamma(x)}} \tag{2.3}$$

2.1.2 Efflux plane (near-field to far-field transition)

Momentum theory models acceleration forces in the flow, which causes the slipstream to contract in order to maintain flow continuity, assuming incompressibility. In reality, however, diffusion also plays a dominant role in the slipstream, and has the opposite effect of slowing and expanding the propeller flow. These two forces are present throughout the slipstream, with acceleration dominating near the propeller, and diffusion dominating far away from it [41]. The plane where diffusion begins dominating over acceleration is called the efflux plane and is the transition from the near-field to the far-field.

Since acceleration contracts the slipstream, and diffusion expands it, the efflux plane is located where the slipstream diameter is smallest. The axial location of the efflux plane, x_e, must be measured experimentally by sweeping through the slipstream to find the location where D_s is smallest. The authors of [41] perform this procedure in static conditions and measure x_e to be $0.764D_p$ downstream of the propeller plane.

The flow conditions at the efflux plane will be important for modelling slipstream in the far-field, as such we must calculate the induced velocity at the efflux plane w_e, and the diameter of the efflux plane D_e. Since the near-field model is applicable up to the efflux plane, D_e and w_e can be calculated using momentum theory, as follows

$$w_e = w_p\gamma(x_e) \tag{2.4}$$

$$D_e = D_p\sqrt{\frac{1}{\gamma(x_e)}} \tag{2.5}$$

2.1.3 Far-field model (semi-empirical)

Past the efflux plane, diffusion becomes a dominant force in the slipstream, expanding and slowing the flow, and reducing the accuracy of the near-field model. Diffusion can be complex to model analytically, which has led to the development of semi-empirical models to predict the slipstream in this region. While these models are commonly used for marine propellers, beginning with studies on propeller jets by Albertson [44] in 1950, it has been until recently that they have been adapted for use in air. Khan and Nahon [41] conducted this adaption, re-fitting the semi-empirical models to predict the slipstream behind small UAV propellers in air with good accuracy.

The far-field model, developed in [41], assumes that flow far downstream of the propeller is axisymmetric about $\hat{x}$, and that the flow velocity profile at a given axial distance is a Gaussian function of radial location. At a given axial distance, the shape of the Gaussian is controlled by three parameters, the maximum velocity $V_{max}(x)$, the radial location where this maximum occurs $R_{max}(x)$, and the standard deviation of the profile $\sigma(x)$, which are all functions of axial location.

These parameters can then be used to calculate the induced velocity in the slipstream at a location $\underline{P}_d = [x_d,\ r_d]$ using

$$w(x_d, r_d) = w_d = V_{max}(x_d)\ \exp\left(\frac{(r_d - R_{max}(x_d))^2}{-2\sigma(x_d)^2}\right) \tag{2.6}$$

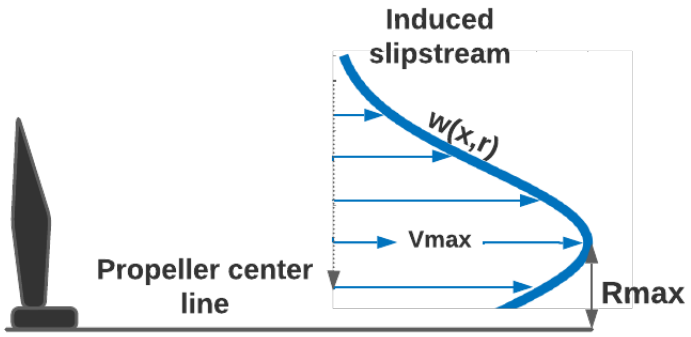

Figure 2.3: Flow velocity distribution in the slipstream far-field, showing parameters $V_{max}(x)$ and $R_{max}(x)$

V_{max}, R_{max}, and σ are observed to change relatively linearly with respect to axial distance in three distinct sections of the far-field by the authors of [41], motiving a partitioning of the far-field into smaller sections to use linear models of these parameters.

2.1.3.1 Sectioning the far-field

The far-field is partitioned into three sections, labelled 1-3, with section 1 beginning at the efflux plane and section 3 being the furthest downstream. In sections 1 and 2 a low velocity core near the propeller's rotational axis causes the peak of the slipstream velocity distribution to be offset from $\hat{x}$. This results in the flow velocity distribution appearing as two Gaussian curves mirrored about $\hat{x}$. In marine propulsion research, sections 1 and 2 are called the "zone of flow establishment" [38]. Figure 2.3 shows the flow velocity distribution for a half cross-section of the slipstream, as the flow travels downstream peaks on each side of $\hat{x}$ gradually move closer to the propeller's rotational axis, until they eventually meet and form a single Gaussian curve. The point where the peaks merge marks the beginning of section 3 which, in the marine propulsion literature, is called the "zone of established flow" [38]. An analogous way to define section 3 is that it is the region of the far-field where $R_{max}(x) = 0$.

While the $R_{max}(x) = 0$ condition clearly defines the bounds of section 3, the border between section 1 and 2 is less clear. The division between these zones is established in [41] as the authors observed that, before section 3, $V_{max}(x)$, $R_{max}(x)$, and $\sigma(x)$ appeared to be linear in two distinct regions, which motivated the split into separate sections. As such, the boundary between section 1 and section 2 was selected by the authors through examination of experimental measurements and can be seen in Table 2.1.

Table 2.1: Axial boundaries of sections in the far-field model [41] in terms of efflux plane location and diameter

Section	Beginning of section	End of section
1	x_e	$1.7D_e + x_e$
2	$1.7D_e + x_e$	$4.35D_e + x_e$
3	$4.35D_e + x_e$	∞

While these boundaries were measured for a particular propeller, the authors of [41] show that they are valid for a variety of propellers and propeller configurations.

2.1.3.2 Empirical parameters and tuning

Fitting of the far-field model was done in [41] by linearly fitting each parameter: $V_{max}(x)$, $R_{max}(x)$, and $\sigma(x)$ in sections 1-3.

Sections 1 and 2: The equations governing $V_{max}(x)$, $R_{max}(x)$, and $\sigma(x)$ in sections 1 and 2 are now considered. The symbol $R_{max,0}$ denotes the value for R_{max} at the efflux plane, and it is represented as a function of the diameter of the efflux plane D_e and the radius of the propeller hub R_{hub} [41] [45]

$$R_{max,0} = 0.67\left(\frac{D_e}{2} - R_{hub}\right) \tag{2.7}$$

$$V_{max}(x) = w_e\left(a_{1,i} - b_{1,i}\frac{x - x_e}{D_p}\right) \tag{2.8}$$

$$R_{max}(x) = R_{max,0}\left(a_{2,i} - b_{2,i}\frac{x - x_e}{D_e}\right) \tag{2.9}$$

$$\sigma(x) = \frac{\sqrt{2}}{2}\left[a_{3,i}R_{max,e} + b_{3,i}(x - x_e - R_e)\right] \tag{2.10}$$

As can be seen above, the empirical equations for $V_{max}(x)$, $R_{max}(x)$, and $\sigma(x)$ require six parameters in sections 1 and 2, denoted: $[a_{1,i},\ a_{2,i},\ a_{3,i},\ b_{1,i},\ b_{2,i},\ b_{3,i}]$. The second subscript i of each parameter refers to the section which it is used in.

Section 3: Since the flow velocity in section 3 can be described as a single Gaussian across the entire slipstream diameter, the empirical equation for $\sigma(x)$ must be slightly

modified to describe this section. $V_{max}(x)$ and $R_{max}(x)$ however, can still be modelled using Eq. (2.8) and (2.9) respectively.

$$\sigma(x) = \frac{\sqrt{2}}{2}\left[b_{3,i}(x - x_e)\right] \tag{2.11}$$

Since the Eq. (2.11) uses one less parameter than (2.10), only five parameters are required to fit the model in section 3.

Khan and Nahon fit the 17 parameters experimentally though extensive probing of the slipstream behind an Electrifly 10×4.5 propeller in [41]. The tuned parameters are summarized in Table 2.2.

Table 2.2: Table of coefficients used in the semi-empirical slipstream model [41]

Parameter	Section 1	Section 2	Section 3
i	1	2	3
$a_{1,i}$	1.24	1.37	0.89
$b_{1,i}$	0.0765	0.1529	0.04
$a_{2,i}$	1	1.3	0
$b_{2,i}$	0.1294	0.3059	0
$a_{3,i}$	0.8839	0.5176	N/A
$b_{3,i}$	0.1326	0.2295	0.2411

An annotated simulation of the semi-empirical slipstream model discussed in this section is shown in Figure 2.4 for a particular case in which the freestream velocity, V_∞, is zero and the propeller angular velocity expressed in $\mathrm{rev/s}$, n, is set to $n = 145\ \mathrm{rev/s}\ (8700\ \mathrm{rpm})$. The variable n_{rpm} in Figure 2.4 refers to the propeller angular velocity in explicit units of RPM. The effect of both V_∞ and n on the slipstream will be discussed in detail in the following section. Figure 2.4 shows the contraction of the near-field under momentum theory, and the gradual expansion past the efflux plane due to diffusion.

2.2 Effect of freestream velocity on slipstream

The slipstream model discussed in Section 2.1, and the original work presented in [41], only considers propeller slipstream under static conditions, where the freestream velocity, V_∞, is zero. In this thesis however, we need to use the model in forward flight conditions

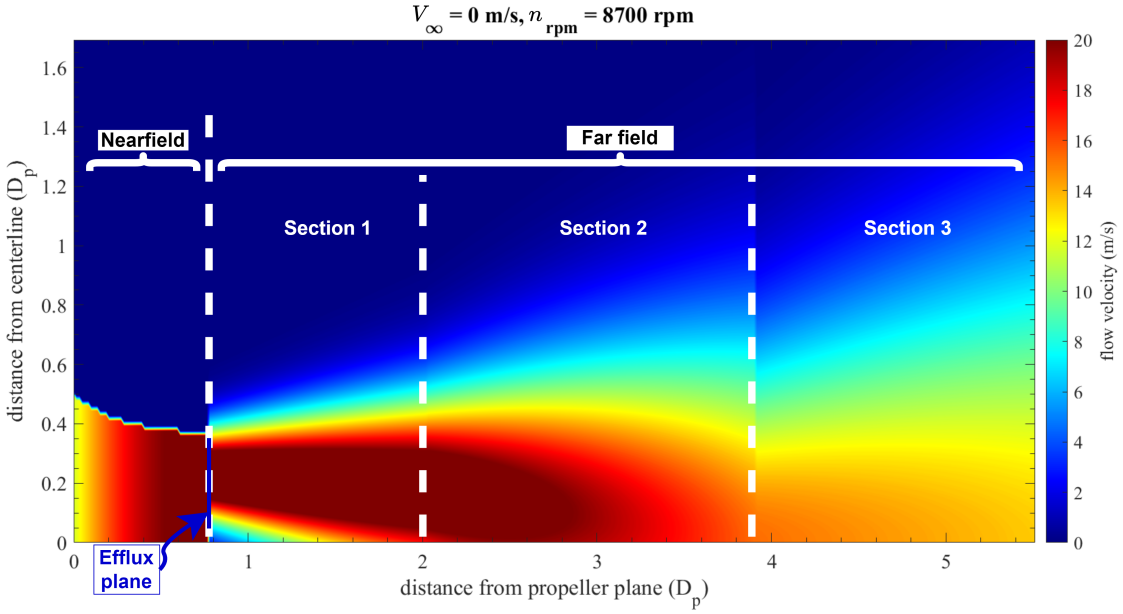

Figure 2.4: Half cross-section of static slipstream flow showing near-field, far-field, far-field sections and efflux plane. Propeller conditions are $V_\infty = 0\frac{m}{s}$, $8700\ RPM$

where freestream airspeed becomes a critical component of flow inside the slipstream. This section will examine how the model presented in Section 2.1 can be adapted for use with a nonzero freestream flow, and how the propagated induced flow velocity w_d can be converted to a flow velocity relative to the propeller frame, shown in Figure 2.1.

As will be detailed in the remainder of this section, the slipstream is impacted by freestream flow through three mechanisms, the first being the reduction in thrust with an increased advance ratio, the second being the reduction in w_p with freestream, and the final mechanism being the superposition of induced flow and freestream flow to calculate total slipstream flow. These three mechanisms are discussed in the remainder of this section, and precede an investigation of the effect of freestream velocity on slipstream, which concludes this section.

2.2.1 Thruster model

To examine the first way in which V_∞ impacts slipstream, we must first examine the thrust coefficient of the propeller. The thrust coefficient, denoted by C_t, is a normalized form of thrust given by the equation below [46]

$$C_t = \frac{T}{\rho n^2 D_p^4} \tag{2.12}$$

The thrust coefficient is usually dependent on the advance ratio, J, which is a function of V_∞ and the propeller's rational speed. J, is defined as follows [47] [48]

$$J = \frac{V_\infty}{nD_p} \tag{2.13}$$

Neither C_t nor J are dimensionless, with units of $1/\text{rev}^2$ and $1/\text{rev}$ respectively. The thrust coefficient C_t has a strong effect on the induced flow at the propeller plane w_p, additionally prior work studying small UAV propellers has shown that C_t decreases as J increases [48]. The implication of these relations is that for a constant propeller speed n, any increase in V_∞ will lead to a decrease in C_t which then leads to a decrease in w_p.

To calculate C_t in this thesis, we use the Blade Element Momentum Theory (BEMT) model developed for small UAV propellers in [49]. The $C_t(J)$ curve generated by this model can be seen in Figure 2.5, which demonstrates the drop in C_t with an increase in J.

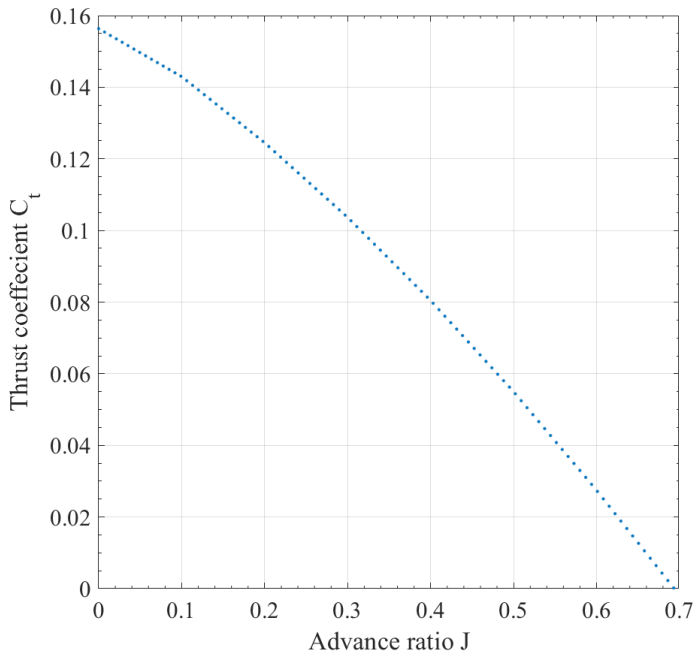

Figure 2.5: Thrust coefficient C_t used in this work as a function of advance ratio J, generated from the BEMT model developed in [49].

2.2.2 Calculation of induced flow at the propeller plane

In Section 2.1, it was assumed that the input to the slipstream model, induced flow at the propeller plane, w_p, is a known value. In reality, however, w_p is generally unknown and must instead be calculated using C_t and the propeller state. We now show how w_p is calculated in forward flight conditions.

The momentum theory actuator disk model can be used to calculate thrust based on V_∞ and w_p using the following equation [42] [43]

$$T = 2\pi\rho R_p^2(V_\infty + w_p)w_p \tag{2.14}$$

The thrust in Eq. (2.14) can then be replaced with Eq. (2.12) to incorporate C_t as

$$C_t\rho n^2 D_p^4 = 2\rho A(V_\infty + w_p)w_p \tag{2.15}$$

In the original work on this slipstream model [41], static conditions were assumed, implying $V_\infty = 0$. Allowing w_p to be easily solved for as

$$w_p = \sqrt{\frac{2C_t}{\pi}} nD_p \tag{2.16}$$

This approach is not valid for forward flight conditions, however, where $V_\infty \neq 0$. Eq. (2.16) instead becomes a quadratic equation in w_p and can be solved as

$$w_p = \sqrt{\frac{2C_t D_p^2 n^2}{\pi} + \frac{V_\infty^2}{4}} - \frac{V_\infty}{2} \tag{2.17}$$

A unique solution for Eq. (2.17) is generated by enforcing $V_\infty, w_p > 0$.

Eq. (2.17) incorporates two dependencies on V_∞, the first of which is the direct use of V_∞ in the equation and the second comes from the thrust coefficient's dependence on V_∞, as discussed previously.

2.2.3 Total flow from induced flow

The final way in which V_∞ affects the slipstream comes through the calculation of total flow from induced flow. In this context, total flow represents the flow velocity relative to the propeller's coordinate system. If one were to insert an airspeed sensor, rigidly attached to an aircraft, into the propeller slipstream, they would measure the total flow velocity. The slipstream model detailed in Section 2.1 outputs an induced flow velocity

w_d at the test location $\underline{P}_d$. We now look to convert this into a total flow velocity at $\underline{P}_d$, denoted V_d.

In [43] and [42], the induced velocity w_p at the propeller plane is dependent on V_∞ and the total flow velocity at the propeller plane V_p, and is calculated as follows

$$w_p = V_p - V_\infty \tag{2.18}$$

If we extend this to the arbitrary location $\underline{P}_d$, we see that V_d is simply the superposition of V_∞ and w_d, yielding the relation [6]

$$V_d = w_d + V_\infty \tag{2.19}$$

2.2.4 Effect of freestream on slipstream

Now that the slipstream model is extended for use in forward flight, we proceed to use the model to investigate how V_∞ affects the results of the slipstream model. To do this, we run a series of simulations on the slipstream model, holding propeller RPM constant while changing V_∞. These simulations are compiled into a single plot in Figure 2.6, for a radial range of $\pm 0.83 D_p$ and an axial range of $0 - 3.74 D_p$. The results for the semi-empirical slipstream model are shown in the top half of each subfigure. For comparison, the bottom half of each subplot showcases the results for the classical momentum theory slipstream model, i.e. continuing to use the near-field model past the efflux plane.

Figure 2.6 shows that V_∞ does not notably change the shape of the slipstream, but does have a strong influence on the velocity gradient. Comparing the $V_\infty = 15 \frac{m}{s}$ case with the static $V_\infty = 0$ case demonstrates this effect well, with the static case showing a slower slipstream with a more noticeable velocity gradient, and the $V_\infty = 15 \frac{m}{s}$ case showing an overall faster slipstream that is much more uniform.

The superposition of freestream flow and induced flow, is the cause of this reduced velocity gradient at higher freestream speeds. The low induced velocity at the propeller plane w_p caused by high freestream speeds leads to a low propagated induced velocity w_d. When V_d is calculated using Eq. (2.18), most of the flow velocity comes from V_∞ due to the low w_d. Since w_d is the element which contains the velocity gradient, having a low influence from w_d results in a relatively uniform slipstream. The opposite of this happens in static conditions, when the propeller's induced flow it at its strongest, and freestream is nonexistent. This phenomenon is more clearly illustrated in Figure 2.7,

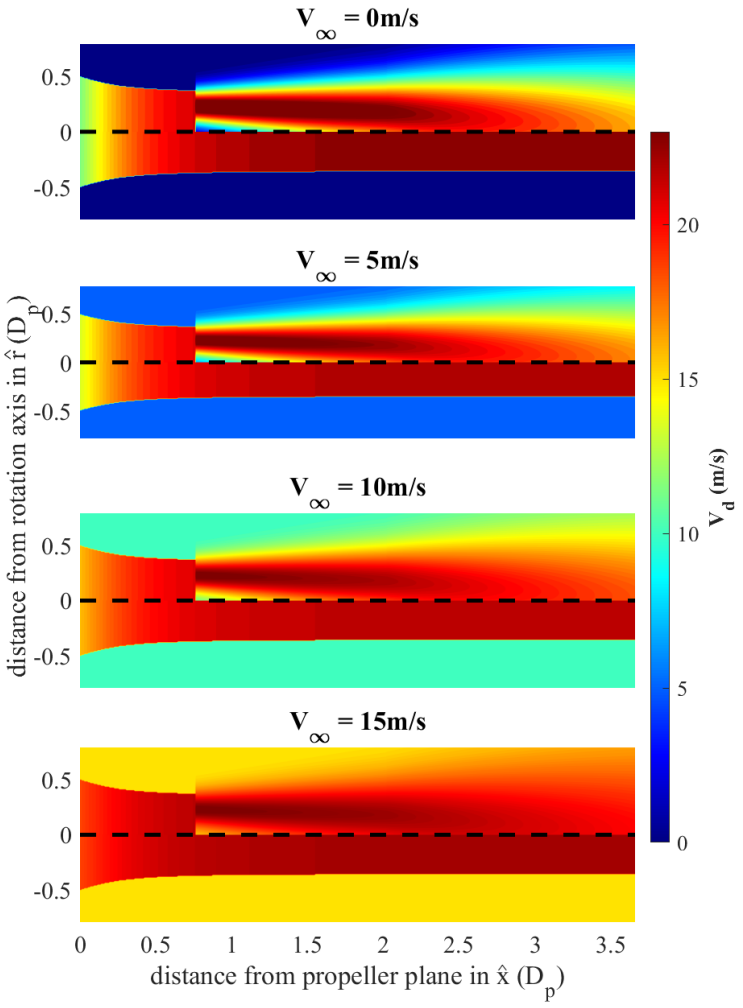

Figure 2.6: Slipstream changing with freestream speed. $n_{rpm} = 8700\ RPM$, $V_\infty = 0, 5, 10, 15$ m/s

which plots $V_{max}(x_d)$, $w_{max}(x_d)$, and V_∞ along $\hat{x}$, for three freestream speeds, all with the same propeller RPM. Figure 2.7 shows the faster V_∞ leading to a lower induced flow and a lower gradient in the slipstream. It may also be observed that at certain locations, such as $x_d/D_p \approx 2.75$, V_d appears to be somewhat constant regardless of freestream velocity. This is an interesting effect where freestream and induced flow balance each other and will be elaborated on further in Chapter 3.

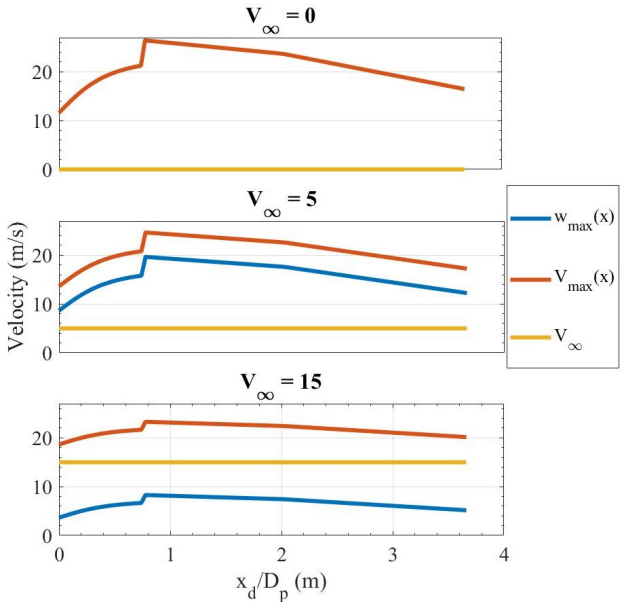

Figure 2.7: Slipstream model results of maximum induced flow, and flow velocity versus axial distance for a propeller operating at 8700 RPM with differing freestream speeds. The top figure shows the static case, the middle figure shows the case where $V_\infty = 5\frac{m}{s}$, and the bottom figure shows the case where $V_\infty = 15\frac{m}{s}$. In the top subfigure, V_{max} and w_{max} are equal.

2.3 In-flight validation of the propeller slipstream model

Now that the slipstream model for forward flight conditions is established, we proceed to validate this model in experimental flight tests using an instrumented vehicle to evaluate its performance. In doing so, we also evaluate the performance of the classical momentum theory slipstream model and compare the two. Tests are conducted with both constant and varying propeller speed.

2.3.1 Methodology

Model validation is performed by instrumenting an agile fixed-wing platform to measure propeller RPM, freestream velocity V_∞, and airspeed at some location inside the slipstream V_d. Propeller RPM and freestream speed are the inputs to the semi-empirical slipstream model, used to calculate an expected slipstream velocity from the model which is compared to the measured V_d to evaluate model accuracy.

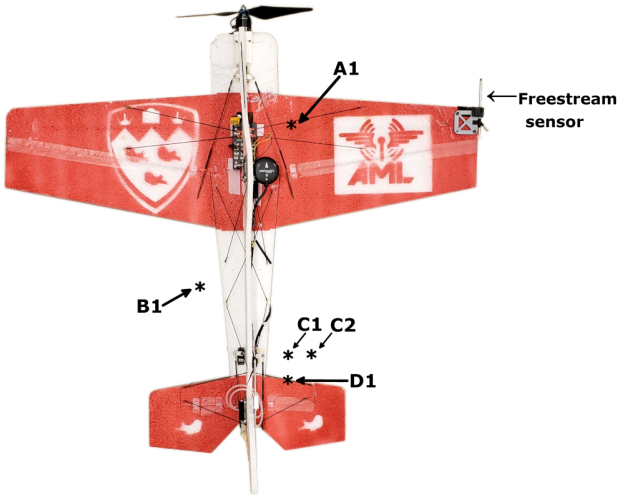

Figure 2.8: Experimental platform used for flight tests, showing the freestream airspeed sensor placed on the starboard wingtip and the slipstream sensor locations.

The agile fixed-wing platform used in these flight tests is a West Michigan Park Flyers McFoamy, fitted with a *Master Airscrew MR Series - 10×4.5* propeller. The *MR - 10×4.5* is not the same propeller used in the original work on the semi-empirical slipstream model [41], but does have the same diameter and pitch. The onboard flight computer is a PixRacer running PX4 and is responsible for semi-autonomous control and sensor data-logging. The instrumented platform can be seen in Figure 2.8.

2.3.2 Sensors

Three sensors, two airspeed sensors and one RPM sensor, are required to record the key inputs and outputs of the slipstream model, n, V_∞, and V_d. One airspeed sensor is dedicated to measurement of V_∞, and the other measures V_d.

A UAVCAN-enabled Electronic Speed Controller (ESC), the Kotleta20, was used for measurement of propeller RPM. This ESC contains its own microprocessor, both to control

the motor, and to log performance data. Data is regularly reported to the main flight computer over a UAVCAN bus. By default, the Kotleta20 measures and reports motor RPM, so no modifications were required.

Airspeed is measured using differential pressure sensors connected to pitot probes. To avoid conflicting addresses on the flight computer's I^2C databus, two types of differential pressure sensor were used. A MS4525DO as part of the mRobotics mRo I^2C Airspeed Sensor system was used to measure freestream flow and a Sensirion SDP33 as part of the Drotek SDP3x Airspeed Sensor package was used to measure airspeed inside the slipstream.

To ensure the three sensors were functioning correctly and producing accurate measurements, all three were validated in advance of experimental flights. Due to the measurement mechanism of the SDP33 sensor, calibration of this sensor was required. Further detail relating to the validation and calibration of these sensors can be found in Appendix A.

The results of the sensor validation shows that all three sensors have suitable accuracy. The MS4525DO was seen to be able to measure airspeeds above $6\ \mathrm{m/s}$ with an error less than $0.4\ \mathrm{m/s}$. The SDP33 had substantially better performance, with measurements being within the accuracy bounds of the airspeed reference for nearly all airspeed values tested. The RPM sensor in the Kotleta20 also showed very good performance, with a RMSE of about $30\ \mathrm{RPM}$.

2.3.3 Flight profile

The flight profile used in the flight test campaign requires smooth and level flight to ensure accurate airspeed readings and flight segments with constant throttle and variable throttle to examine the accuracy of the model under both these conditions. To achieve smooth flight with throttle control, the platform is flown semi-autonomously via remote with PX4 configured in fixed-wing stabilized mode. This mode allows the pilot to command an attitude which is tracked by the flight computer, enabling the pilot to issue smooth attitude commands, avoiding high levels of sideslip or angle of attack, while still enabling attitude disturbance rejection. While the attitude commands in this mode are used as references for the PX4 attitude controller, the throttle command is directly passed through to the ESC.

The procedure for each flight test begins with the launching of the platform, after which it is flown to a safe altitude where the pilot commands the aircraft to follow a level and approximately straight flight path.

2.3.4 Measurement locations

Ground-based measurements of propeller slipstream, such as those carried out in [35] and [41], do not need to consider the impact of their measurement mount on airworthiness. Unfortunately, for in-flight slipstream measurements on a lightweight platform, the weight of the sensor mount is of critical consideration. Since automated stages capable of sequentially sampling airspeed at many locations in the slipstream would be far too heavy to implement on an agile fixed-wing platform, we instead measure slipstream using stationary mounts at a select number of carefully chosen measurement points. We choose to measure slipstream at five locations, one near the wing of the platform, one behind the aileron, and three near the tail.

The mounting hardware consists of stationary mounts bonded to the airframe which enable measurements at four axial distances. These mounting points can be paired with one of two sensor cradles which offset the pitot-probe from the stationary mount. The location of the stationary mount determines the axial placement of the probe, and the cradle used determines the approximate radial location. This hardware can be seen in Figures 2.9 and 2.10. The mount for location D1 is slightly different as it is swept backward to measure an axial distance far downstream without interfering with the rudder.

To establish a descriptive and compact notation for the sensor locations, each test point is denoted by an alphanumeric code where a letter $A-D$ represents the approximate axial location, and the number $1-2$ represents the approximate radial location. These locations are summarized in Table 2.3 and can be seen overlaid on the test platform in Figure 2.8.

Table 2.3: Test point locations

Axis	A1	B1	C1	C2	D1
$\frac{\hat{x}}{D_p}$	0.81	1.99	2.48	2.47	2.68
$\frac{\hat{r}}{D_p}$	0.35	0.29	0.33	0.48	0.32

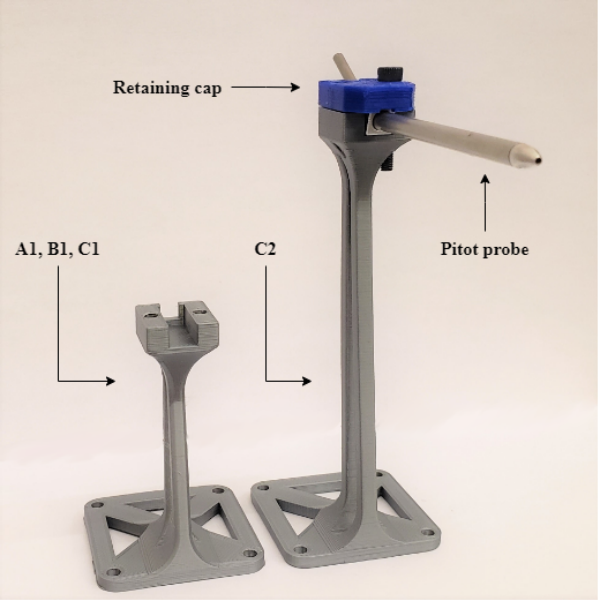

Figure 2.9: Pitot probe 'cradle' mounts.

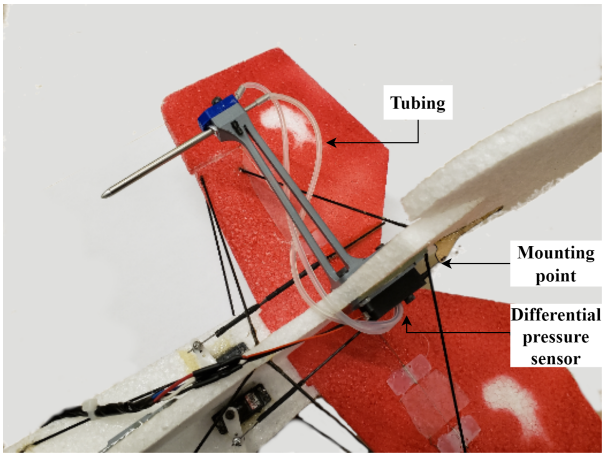

Figure 2.10: Airfame mounting point showing differential pressure sensor and tubing. The test location shown here is C2.

2.4 Results

Multiple flights measuring freestream flow velocity, propeller RPM, and slipstream flow velocity are conducted with segments of constant and variable throttle. This validation work examines smooth level flight, validation with a constant throttle level is presented first, and followed by validation of tests with throttle variations. Since the flight test was operated semi-autonomously, it is important when comparing data from different flights to only compare flight segments where conditions are similar. This is accomplished by selecting 10 s data segments from each flight which represent straight and level conditions, with low yaw, pitch, and roll rates to avoid maneuvering which can cause inaccurate airspeed measurements. Selection of data segments for the constant throttle results are constrained by the additional criteria that the throttle must be constant within approximately 300 RPM, and the throttle must be greater than 90% of maximum. Since throttle was controlled directly by the pilot, the throttle command was never the exact same between flights, which made matching flight segments for the throttle variation trials not possible. As such, the throttle variation trials are not completely similar between different segments, and must be assessed more individually for each location.

2.4.1 Constant throttle tests

Results for the constant throttle segments are shown in Figures 2.11 through 2.15. Each plot shows the recorded freestream, and slipstream data, along with the results expected from the semi-empirical and momentum theory models. Figure 2.16 shows the root mean squared error (RMSE) between each model and the measured data for each test location.

It should be noted that since test point C2 is outside the influence of momentum theory's slipstream, and as such the velocity expected by momentum theory at this location is always equal to V_∞. This is clearly seen in Figure 2.14, as the momentum theory model result and freestream velocity completely overlap each other.

2.4.2 Throttle variation tests

Results for the variable throttle segments can be seen in Figures 2.17 through 2.21, with RMSE summarized in Figure 2.22.

2.4.3 Discussion

Discussion of the in-flight validation results is broken down into discussion of the results near the tail section, at locations C1, C2, D1, and discussion of the results near the wing section A1, B1. This is done as the characteristics of the results from locations near the tail are substantially different from the results near the wing.

Tail region: Results near the tail region show good agreement between the semi-empirical slipstream model and experimental measurements, with similar error between constant throttle segments and throttle variation segments. These errors are similar to those observed in the original work on the slipstream model in [41], but slightly larger. The increased error is likely due to a few factors, firstly the semi-empirical slipstream model was not fitted for the specific propeller used in this work, even though it has the same pitch and diameter as the propeller used in [41], the exact blade profile likely has a non-negligible effect on the slipstream. Secondly, the model does not account for interactions between the slipstream and the platform's airframe or wing. Both Deters and Selig in [35] and Leng et al. in [36] found that a flat plate airfoil can substantially interfere with the slipstream and cause a violation of the axi-symmetric assumption that is core to the semi-empirical model. Finally, the location of the efflux plane is critical to the semi-empirical

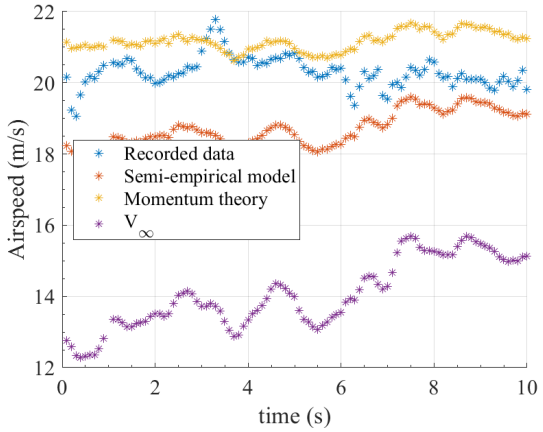

Figure 2.11: Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location A1.

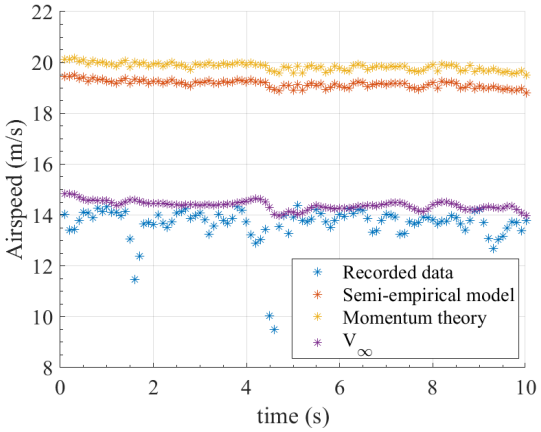

Figure 2.12: Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location B1.

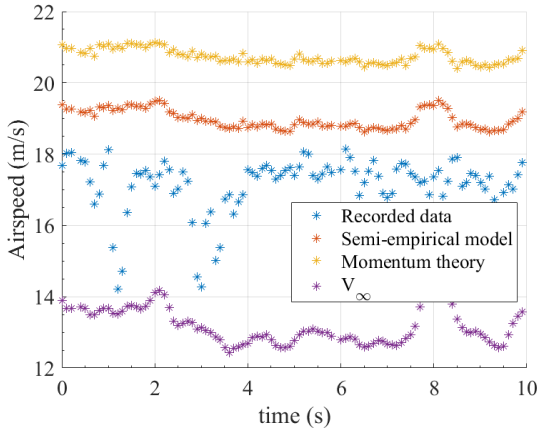

Figure 2.13: Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location C1.

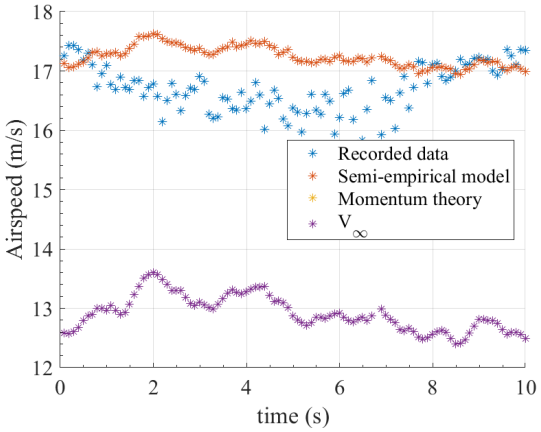

Figure 2.14: Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location C2.

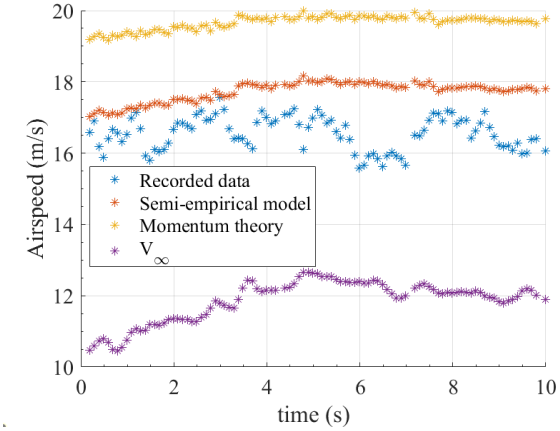

Figure 2.15: Velocity measurement, semi-empirical model and momentum theory model estimates for the steady flight case at test location D1.

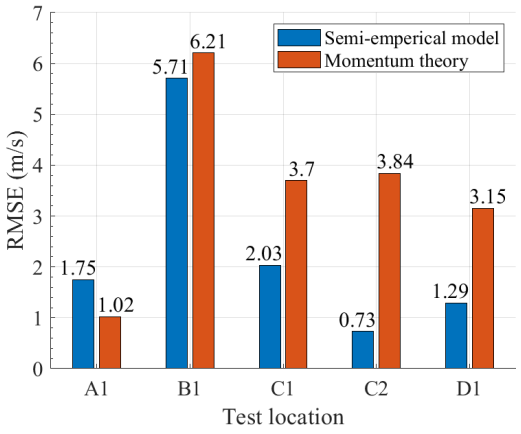

Figure 2.16: Root mean squared errors of the semi-empirical model and classical momentum theory model for the 10-second level flight segments shown in Figures 2.11 through 2.15

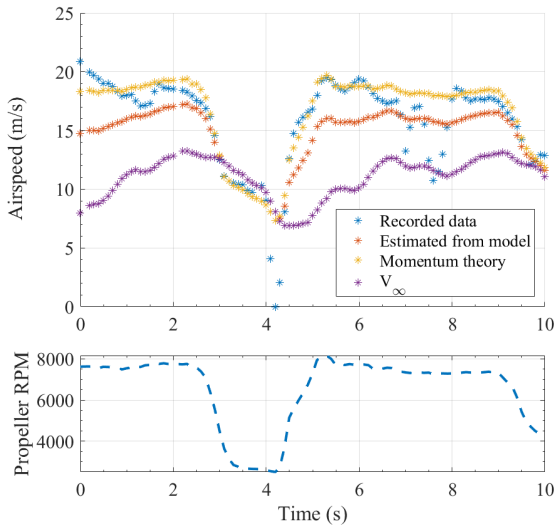

Figure 2.17: Slipstream velocity recordings and model estimates at test point A1 during rapid throttle change (top), and RPM measurements (bottom).

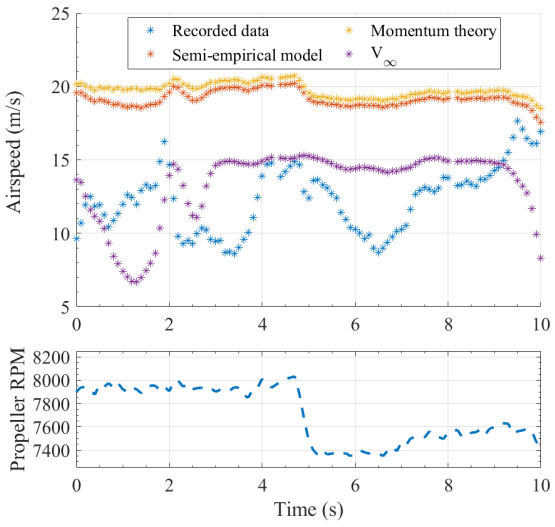

Figure 2.18: Slipstream velocity recordings and model estimates at test point B1 during rapid throttle change (top), and RPM measurements (bottom).

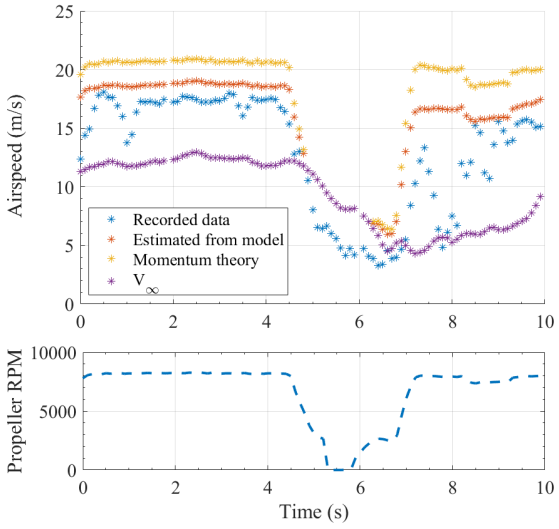

Figure 2.19: Slipstream velocity recordings and model estimates at test point C1 during rapid throttle change (top), and RPM measurements (bottom).

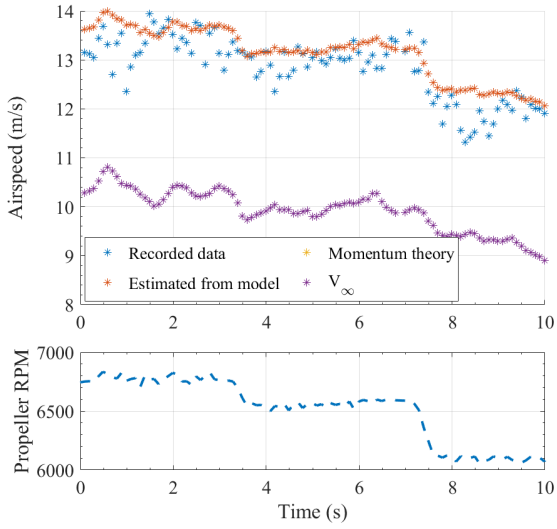

Figure 2.20: Slipstream velocity recordings and model estimates at test point C2 during rapid throttle change (top), and RPM measurements (bottom).

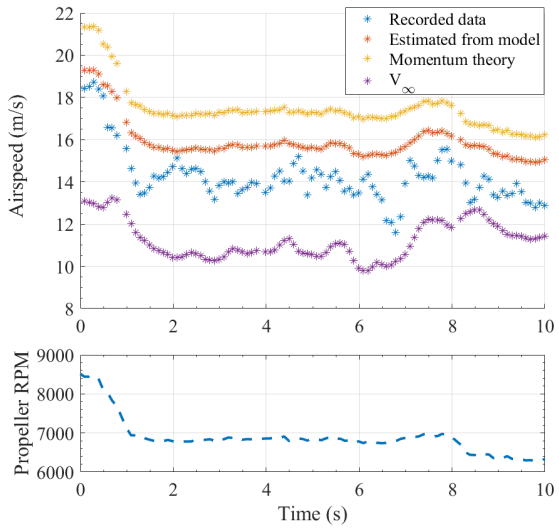

Figure 2.21: Slipstream velocity recordings and model estimates at test point D1 during rapid throttle change (top), and RPM measurements (bottom).

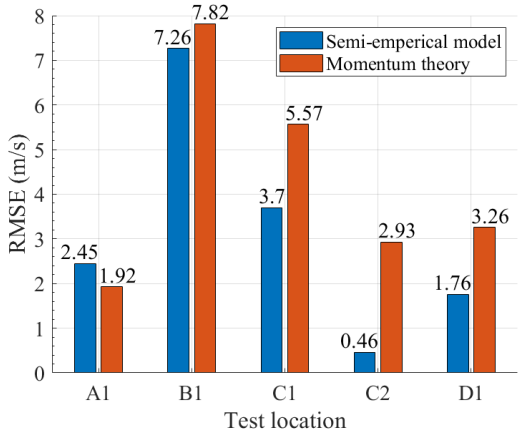

Figure 2.22: Root mean squared errors of the semi-empirical and classical momentum theory models for throttle variation flight segments shown in figures 2.19 through 2.21

slipstream model, but is only measured in static tests in [41]. It is unknown how the efflux plane may change in flight, but any deviations from the static case can easily cause a notable impact on the accuracy of the semi-empirical model.

While experimental results show slightly higher error than the original work on the semi-empirical model, the accuracy is still concluded to be very good, with the semi-empirical model providing much better results than momentum theory at all test locations in the tail region.

Wing region: Results from test locations A1 and B1 contrast the results from the tail region with poorer performance from both the semi-empirical model and momentum theory. Performance between the constant throttle and throttle variation tests is similar, with the throttle variation tests displaying slightly worse accuracy. Measurements at location A1 show better performance from momentum theory compared to the semi-empirical model, though this difference is relatively slight.

The most interesting location is B1, which shows a substantial deviation from both models, and is very similar to the freestream measurement, not showing any notable influence from the slipstream. This measurement location being directly behind the port aileron may be responsible for the discrepancy, as strong interference from the wing may be present. Due to the unexpected nature of measurements in this location, the B1 flight test was repeated additional times to ensure accuracy of the results.

The results of this chapter indicate that the semi-empirical slipstream model provides accurate flow prediction results near the tail of the agile fixed-wing platform in forward flight conditions. Results were shown for measurements with constant throttle and with throttle variations present, with both having similar performance.

Now that the slipstream model is validated, we proceed to use it in the following chapter to remove the effect of propeller slipstream on wind-estimation for agile fixed-wing platforms.

Chapter 3

Synthetic airspeed sensing

3.1 Introduction

The objective of wind estimation is to use the suite of sensors onboard a UAV to determine both the magnitude and direction of the surrounding wind. The problem can be particularly complex on lightweight UAVs due to their limited wind sensing capabilities, in particular the use of uni-directional airspeed sensors. Calculation of wind properties can be done using the difference between airspeed and groundspeed, and while the complete groundspeed vector is available through GPS and barometric sensors, only having one component of airspeed available means that the complete wind vector cannot be directly calculated. Instead, only a component of wind is observable, and the platform must change its heading, thus changing the direction of the airspeed sensor, to gradually build up a complete picture of wind magnitude and direction.

Thus, wind estimation can be challenging even if the measured airspeed is undisturbed by any external aerodynamics. Expressed differently, wind estimation is non-trivial even if the required freestream measurement is available. If the measurement is impacted by propeller slipstream, the wind estimation problem inherits an additional layer of complexity. Initial tests were conducted which used airspeed measurements inside the slipstream directly with a conventional wind estimator, without any compensation for the propeller's disturbance to the airflow. The results of this testing showed very poor performance of the wind estimator, as would be expected due to the magnitude of the propeller induced flow seen in Chapter 2.

As is briefly discussed in Chapter 1 and more extensively in Chapter 2, freestream properties can be challenging to measure on an agile fixed-wing platform due to the sub-

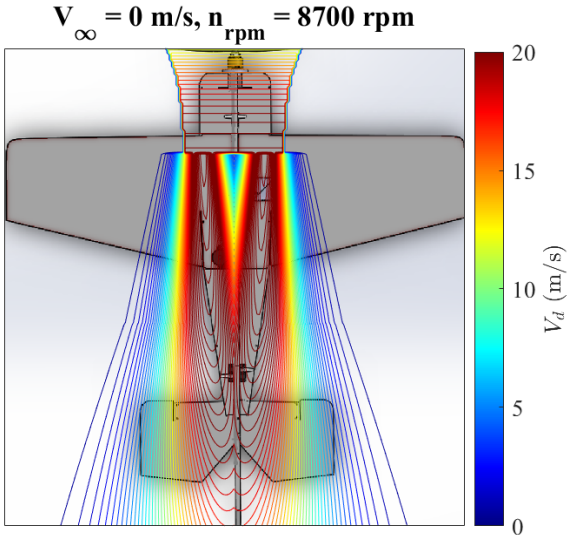

Figure 3.1: Simulation of static slipstream over agile fixed-wing platform.

stantial impact of propeller slipstream flow. Figure 3.1 presents a simulation of the propeller slipstream model overlaid on an agile fixed-wing platform, showing that very few locations on the platform's airframe are unaffected by slipstream.

While the wing appears to be undisturbed by slipstream, it is not an ideal location for sensor mounting due to the low inertia of the platform. It was observed in early experimental flight tests that sensor placement on the wing or at the wing tip led to substantially reduced agility, which is an expected result of placing a sensor far away from the centre of mass of a very lightweight platform. Our main goal in this chapter is to use the slipstream model from Chapter 2 to estimate wind without sacrificing maneuverability by placing the airspeed sensor near the center of mass, well inside the slipstream.

A cascaded approach is used to estimate wind from an airspeed sensor inside the slipstream. The first step in this cascade takes a reading from an airspeed sensor and filters out propeller-induced flow, leaving only the freestream. We call this component the 'synthetic airspeed sensor'. The second component is a traditional wind estimator, which uses synthetic airspeed and other vehicle states to determine the wind vector. The traditional wind estimator is not the focus of this thesis, and we design the cascaded architecture to allow 'drop-in' compatibility with various wind estimators.

The remainder of this chapter begins with an introduction to the cascaded wind estimation architecture, after which preliminaries relating to estimation and airspeed sensors will be discussed. Following this we examine observability of freestream, and describe

the synthetic airspeed sensor we develop. Results for both offline and online estimation are then shown.

3.2 Preliminaries

As a few topics arise frequently throughout this chapter, we present them here first as preliminaries to increase the clarity of the remainder of this chapter. First, we discuss the cascaded wind estimation architecture that serves as the basis of this chapter, then we discuss airspeed sensor measurements, and the Extended Kalman Filter.

3.2.1 Proposed estimation architecture

A key consideration for estimating wind from a sensor submerged in the propeller slipstream is the architecture of the estimator. One option might be to design a monolithic wind estimator which includes slipstream in its sensor measurement model. From a theoretical standpoint, this approach benefits from simplicity, as a slight modification to the measurement model of a wind estimator is required, however, since it requires either modification to an existing wind estimator, or development of a new wind estimator, it has the practical disadvantage of needing to reimplement the wind estimator on flight hardware. Additionally, since this approach directly uses slipstream measurements, an estimate of undisturbed freestream is still unavailable to the system for other applications, such as stall detection.

Instead of the monolithic wind estimation solution, this thesis proposes a modular architecture to estimate wind from an airspeed sensor immersed in slipstream without a modified wind estimator. The approach is inspired by the work of Sun et al. which develops a wind estimator for tailsitter UAV's in hover [14]. Their approach first uses a synthetic airspeed sensor to estimate freestream using a dynamics model. The freestream estimate is then sent to a relatively conventional wind estimator.

In this work, the synthetic airspeed sensor computes a freestream estimate, v_∞, from a measurement of airspeed inside the slipstream, v_d, using the slipstream model presented in Chapter 2. The synthetic freestream estimate can then be used with any freestream-based wind estimator, making it 'drop-in' portable between different systems. We also notice that since wind is being calculated, and the relationship between wind and airspeed should obey the wind-triangle, we can use the estimated wind vector to reduce the noise of the freestream estimate. This uses the assumption that wind should change

slowly compared to airspeed inside the slipstream, allowing us to use it for filtering. The wind-triangle is the kinematic relation between freestream, wind, and groundspeed, written as follows [50]

$$\underline{V}_i = \underline{V}_{wind} + \underline{V}_\infty \tag{3.1}$$

A block diagram overview of the proposed cascaded estimation architecture can be seen in Figure 3.2.

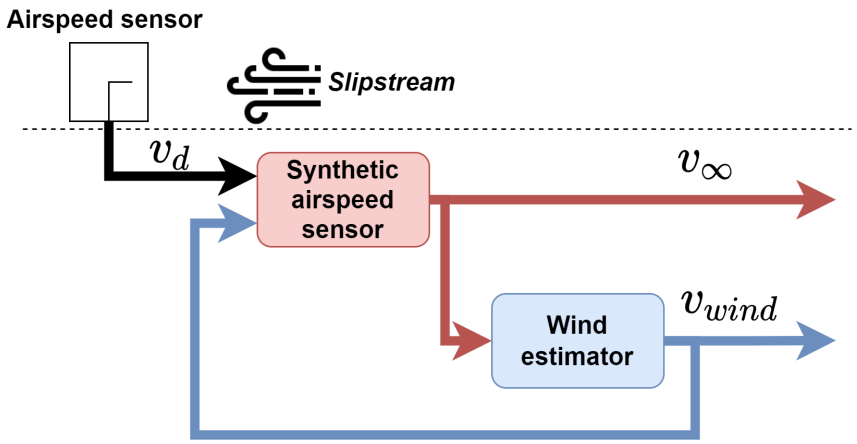

Figure 3.2: Block diagram overview of proposed estimator architecture. The airspeed sensor measures v_d inside the slipstream, which is a combination of induced propeller flow and freestream. The synthetic airspeed sensor extracts the freestream component of this measurement, and sends it to a conventional wind estimator. While not shown, both the synthetic airspeed sensor and wind estimator also rely on groundspeed and attitude states.

3.2.2 Effect of oblique flow conditions on pitot tubes

Since all of the results we present in this chapter use some experimental airspeed measurements, whether the estimator is running offline with flight data or online on the platform's flight computer, it is worthwhile to discuss airspeed measurements using pitot tubes. While the topic was briefly discussed in previous sections of this thesis, we provide additional details here as the work in this chapter features more aggressive flights which momentarily exceed the measurement capabilities of inexpensive pitot probes. The additional context is required for interpretation of sensor data beyond these limits.

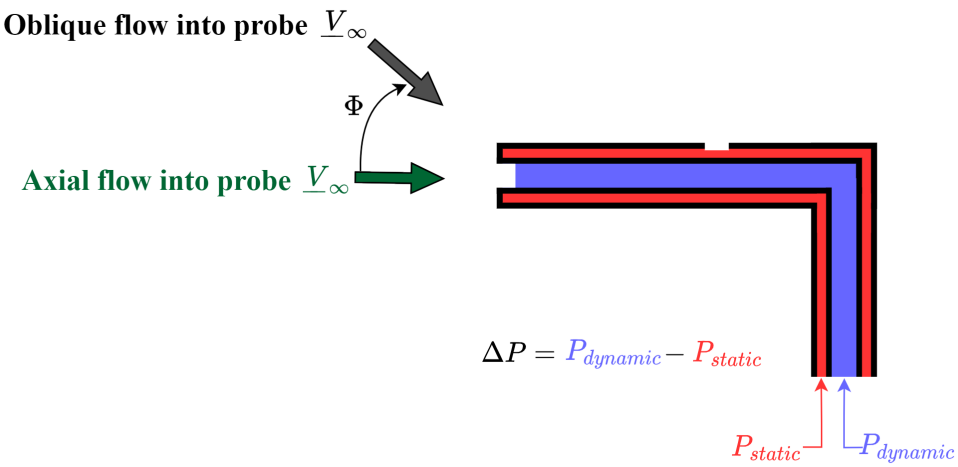

Figure 3.3: Pitot-static probe diagram

Uni-directional pitot-static probes work by measuring the difference in pressure between two sources. One source, the static source, should be pressure equalized with the ambient air, as such it is usually oriented to prevent flow from being driven into it. In contrast, the dynamic source should be aligned with the flow and completely stop it, raising the pressure of the dynamic source according to Bernoulli's Equation. The difference between the pressure of the static and dynamic sources is measured by a differential pressure sensor and is used to calculate flow velocity. Figure 3.3 presents a diagram of a pitot-static probe.

This type of airspeed measurement is very popular on fixed-wing aircraft, but is most reliable when the airflow is travelling directly into the dynamic port. We will denote the angle between the pitot probe's dynamic source tube and the airspeed vector as the oblique flow angle Φ. As the oblique flow angle increases, the calculated flow velocity from the pitot probe drops [51]. This drop is expected, as the airspeed component parallel to the probe is reduced. However, the magnitude of the drop observed during aggressive maneuvers, is less expected. It is suspected that during these events, air enters the static port depending on the orientation of the airspeed vector. The sharp drop in Δp during these events, leads to a drop in calculated airspeed v_d, as will be shown in experimental data in Section 3.5.1, but the measurement recovers quickly.

All this is to say that the pitot-static tube, while being a lightweight and inexpensive method to measure airspeed, does have some limitations at high oblique flow angles. This is important to note as we process experimental data later in this chapter.

Since, unlike Chapter 2, this chapter considers oblique flow, the airspeed sensor will not always measure V_∞ or V_d in their entirety, instead only measuring the component along the airspeed sensor's measurement axis, indicated similarly using lower-case as v_d and v_∞.

3.2.3 Extended Kalman Filter

Since it will be used at multiple points throughout this chapter, we now provide a brief description of the Extended Kalman Filter (EKF).

Before migrating to the EKF, it is worthwhile to mention the Kalman filter itself. The Kalman filter is a way to optimally estimate the state of a linear system, and it works on the notion of prediction-correction. The system model is used to predict its behaviour in the future, and this prediction is corrected with a measurement. Critically, neither the model nor the measurement are assumed to be perfect, and noise is assumed for both. Even if a measurement is expected to be very noisy, a Kalman filter can still use its information to increase its performance. The optimality of the filter, relates to its minimization of error variance [52].

Since the Kalman filter only works for linear systems, the EKF is developed to 'extend' the Kalman filter for use with non-linear systems. This is done by linearizing the system about its current operating point [53].

To describe the mathematical process of the EKF, we can use a relevant example system, the wind triangle used by many wind estimators. Here, our state $\underline{q}$ is the wind vector, and we assume that we have a uni-directional airspeed sensor which measures true freestream mounted on a fixed-wing UAV. To maintain common notation, we also label our v_∞ measurement as y, our inertial velocity is denoted $\underline{V}_i$ and the measurement axis of the airspeed sensor is assumed to be aligned with the platform's longitudinal axis $\hat{b}_1$. The notation and equations used for the EKF in this section and throughout the remainder of this chapter are heavily drawn from [54]. A priori values are denoted using $\check{(\cdot)}$ and a posteriori values are indicated using $\tilde{(\cdot)}$.

The discrete system, based on the wind triangle, and assuming constant wind, can be written as follows, where $\underline{w}_k$ and η_k represent the process and measurement noise respectively

$$\underline{q}_k = f\left(\underline{q}_{k-1}\right) = \underline{q}_{k-1} + \underline{w}_k \tag{3.2a}$$
$$y_k = h\left(\underline{q}_k\right) = \hat{b}_1^T\left(\underline{V}_{i_k} - \underline{q}_k\right) + \eta_k \tag{3.2b}$$

As an aside: Eq. (3.2) is already nearly linear, however it does include a constant term due to the $\underline{V}_{i_k}$ in the measurement model, meaning it can't be rewritten in the form $y_k = \boldsymbol{H}\underline{q}_k$. This could likely be compensated for in the linear Kalman filter framework by modifying the equation for the measurement residual. We instead show the EKF approach here, since we use a simple wind estimator for offline tests later in this chapter, and we also use an EKF for synthetic airspeed sensing. Using the EKF framework with Eq. (3.2) allows us to introduce both simultaneously.

In Eq. (3.2a), $f\left(\underline{q}_{k-1}\right)$ is commonly known as the process model and $h\left(\underline{q}_k\right)$ in Eq. (3.2b) is referred to as the measurement model. We can now assemble the EKF as shown below, where $\boldsymbol{Q}$ is the model covariance and R is the noise variance.

Jacobians:

$$\boldsymbol{F_{k-1}} = \left.\frac{\partial f(\underline{q}_{k-1})}{\partial \underline{q}_{k-1}}\right|_{\tilde{\underline{q}}_k} = \left.\frac{\partial \underline{q}_k}{\partial \underline{q}_{k-1}}\right|_{\tilde{\underline{q}}_k} = \boldsymbol{I}$$
$$\underline{H}_k = \left.\frac{\partial h(\underline{q}_k)}{\partial \underline{q}_k}\right|_{\check{\underline{q}}_k} = \left.\frac{\partial\left[\hat{b}_1^T\left(\underline{V}_{i_k} - \underline{q}_k\right)\right]}{\partial \underline{q}_k}\right|_{\check{\underline{q}}_k} = -\hat{b}_1^T$$

Prediction:

$$\check{\boldsymbol{P}}_{\boldsymbol{k}} = \boldsymbol{F_{k-1}}\tilde{\boldsymbol{P}}_{\boldsymbol{k-1}}\boldsymbol{F_{k-1}}^T + \boldsymbol{Q} \tag{3.3}$$
$$\check{\underline{q}}_k = \tilde{\underline{q}}_{k-1}$$
$$\underline{K}_k = \check{\boldsymbol{P}}_{\boldsymbol{k}}\underline{H}_k^T\left(\underline{H}_k\check{\boldsymbol{P}}_{\boldsymbol{k}}\underline{H}_k^T + R\right)^{-1}$$

Correction:

$$\check{y}_k = \hat{b}_1^T\left(\underline{V}_{i_k} - \check{\underline{q}}_k\right)$$
$$\tilde{\boldsymbol{P}}_{\boldsymbol{k}} = \left(\boldsymbol{I} - \underline{K}_k\,\underline{H}_k\right)\check{\boldsymbol{P}}_{\boldsymbol{k}}$$
$$\tilde{\underline{q}}_k = \check{\underline{q}}_k + \underline{K}_k\left(y_k - \check{y}_k\right)$$

The best estimate from the EKF at k is then $\check{\underline{q}}_k = \underline{V}_{wind_k}$. This simple wind estimator uses the same process and measurement model as the PX4 wind estimator, but is not cou-

pled to a larger state estimator. We will use this estimator later in this chapter to conduct offline tests of our cascaded wind estimator, as such, in the remainder of this chapter we will refer to the wind estimator shown in Eq. (3.3) as the 'basic wind estimator'.

3.3 Observability

Before designing the synthetic airspeed sensor, we first examine the observability of v_∞ from a measurement location inside the slipstream v_d. This is critical since, as mentioned in Chapter 2, there are locations where the slipstream airspeed is largely invariant with changes in the freestream. This is due to the propeller induced flow rising as freestream drops, with the two effects counteracting each other. First, v_d can be described as a function, $h(\cdot)$, of v_∞ using the semi-empirical slipstream model in Chapter 2. The propeller speed n is not explicitly included as an input to the function, since we conduct observability tests at a constant propeller speed. The model, summarized as $h(v_\infty)$, can then be linearized about a nominal condition $\overline{v_\infty}$ to more easily examine observability

$$\begin{aligned} v_d &= h(v_\infty) \\ \delta v_d + \overline{v_d} &= h(\delta v_\infty + \overline{v_\infty}) \\ \text{let } \; c &= \frac{\delta h(v_\infty)}{\delta v_\infty} \\ \delta v_d + \overline{v_d} &\approx c\, \delta v_\infty + h(\overline{v_\infty}) \\ \delta v_d &\approx c\, \delta v_\infty \end{aligned} \tag{3.4}$$

With this formulation, and assuming that the process dynamics of v_∞ are decoupled from v_d, v_∞ could be considered observable for any $c \neq 0$. This does not fully characterize the observability of v_∞ however, as small values of c will make estimation of v_∞ challenging, and locations inside the slipstream which provide better signal-to-noise relationships will have a higher value of c. The highest value of c seen in this work is far away from the propeller when $c = 1$, in this location, an airspeed sensor measures freestream directly. To analyze observability of v_∞ from inside the slipstream, we calculate an average value for c throughout the freestream range of $v_\infty \in [0\ 20]$ m/s, denoted $\bar{c}$. Each $\bar{c}$ calculation is at a specific propeller speed and location. To create the 2-dimensional observability maps we will later show, we calculate $\bar{c}$ across a variety of propeller speeds and locations. At a given propeller speed, a grid consisting of 400 points is constructed, covering the

slipstream far-field from $0 - 0.3$ m in the radial $\hat{r}$ direction, and $0.2 - 1$ m in the axial $\hat{x}$ direction. At each location, the freestream range is stepped through at 1 m/s increments, with each increment denoted using the subscript i as v_{∞_i}. The value of c at these conditions is calculated as

$$c_i = \frac{h(v_{\infty_i}) - h(v_{\infty_{i-1}})}{v_{\infty_i} - v_{\infty_{i-1}}} \tag{3.5}$$

where $v_{\infty_i} - v_{\infty_{i-1}} = 1$ m/s. $\bar{c}$ is then simply the average of all c values at a given location across the freestream range.

The results of this observability analysis are shown for six propeller speeds in Figure 3.4, with each subplot showing a portion of the slipstream where the propeller plane is located at $x = 0$ and the propeller rotates about $\hat{x}$.

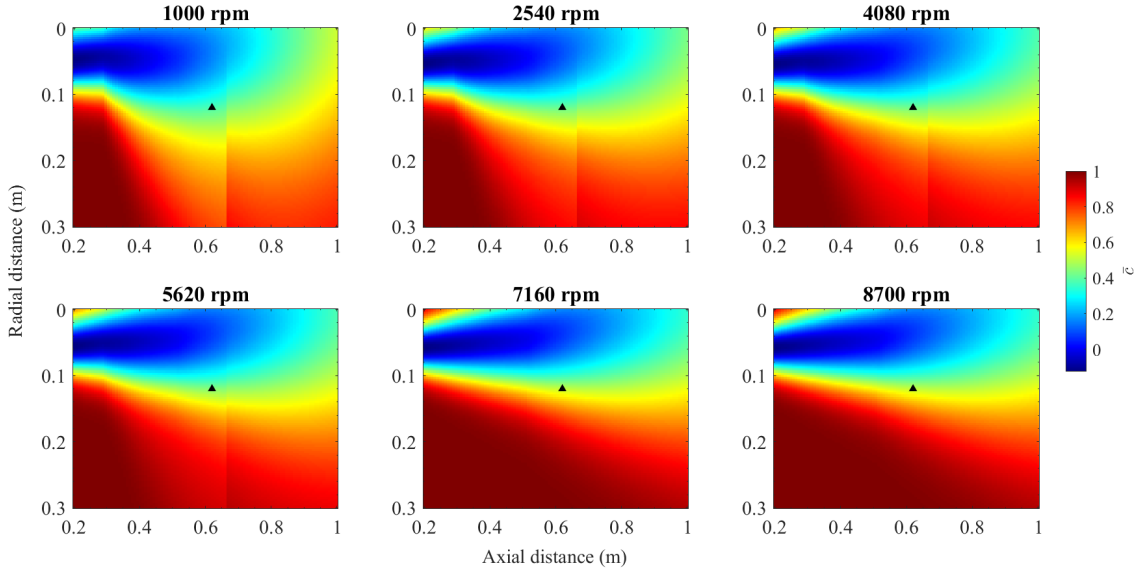

Figure 3.4: Observability of v_∞ inside the slipstream at a variety of propeller speeds. The observability metric shown here is the average value for c, $\bar{c}$. A black triangle marks the location $\underline{P}_d = [0.62\ 0.12]$ m.

As can be seen in Figure 3.4, the observability of v_∞ is relatively constant at propeller speeds higher than 4000 RPM, and there is a distinct region inside the slipstream where v_∞ is very challenging to observe. This occurs close to the propeller plane, and near the propeller's rotational axis. As such, we conclude that the synthetic airspeed sensor approach can be used to estimate v_∞ from a large majority of the slipstream, so long as the v_d sensor is not placed too close to the propeller.

Using this observability analysis while also considering sensor mounting and distance from the platform's centre of mass, we select a good measurement location as $\underline{P}_d = [0.62\ 0.12]\ \mathrm{m}$. This location is marked with a triangle in Figure 3.4, and shown with respect to the platform in Figure 3.8. While a location further downstream of the propeller may seem ideal, the location $\underline{P}_d = [0.62\ 0.12]\ \mathrm{m}$, is essentially the furthest downstream a sensor could be mounted before it interferes with the rudder.

3.4 Synthetic airspeed sensor

Now that v_∞ is seen to be observable from inside the slipstream, we proceed to design a synthetic airspeed sensor to estimate v_∞ from measurements of v_d.

The proposed approach uses an Extended Kalman Filter to estimate freestream. In this case, the process model is based on the wind triangle, and the measurement model is based on the semi-empirical slipstream model.

3.4.1 Extended Kalman Filter

This Extended Kalman Filter implementation of the synthetic airspeed sensor is referred to as the 'EKF-SAS', and is detailed below.

The EKF-SAS begins with the system model. To match notation with Section 3.2.3, let us set the state variable q as $q = v_\infty$ and the measurement y as our measurement inside the slipstream $y = v_d$. It is once again assumed that the airspeed sensor is aligned with the aircraft's longitudinal axis. We can now express the system in discrete-time using the slipstream model and wind triangle as follows, with process and model noise w_k and η_k respectively

$$\begin{aligned} q_k &= \hat{b}_1^T \left(\underline{V}_{i_k} - \underline{V}_{wind_k}\right) + w_k \\ y_k &= h(q_k, n) + \eta_k \end{aligned} \tag{3.6}$$

In Eq. (3.6), the function $h(\cdot)$ is once again the slipstream model, which requires freestream and propeller speed n as inputs, and the process model is directly a result of the wind triangle. It should be noted that q_k and y_k are both scalar, simplifying the estimation problem. For clarity, the same notation used in Section 3.2.3 is employed here.

Before the EKF can be assembled, the measurement model must be linearized. Traditionally, the process model may also be linearized, however due to the cascaded nature of

the synthetic slipstream sensor, q_k is not explicitly dependent on q_{k-1}, instead this dependence comes through the wind estimate. The measurement model is linearized about the a priori state estimate, $\check{q}_k$, with the expected measurement at this location being $\check{y}_k = h(\check{q}_k)$. The partial derivative of $h(\check{q}_k)$, labelled H_k, is assessed as follows

$$H_k = \left.\frac{\delta h(q_k)}{\delta q_k}\right|_{\check{q}_k} \tag{3.7}$$

Instead of directly using the slipstream model presented in Chapter 2, we compute H_k using a polynomial fit of the slipstream model since it is globally smooth. A quadratic surface is fit to the slipstream model within our operating range, with a dependence on v_∞ and n. This approach is taken for two reasons, firstly the slipstream model incorporates the thruster model, which has some discontinuities and switching behaviour in aggressive flight regimes, fitting a surface to the model guarantees a smooth surface avoiding any discontinuity issues. Secondly, the slipstream model was seen to be extremely well captured by a low-order surface, as such minimal performance loss would be expected using the fitted surface instead of the complete model.

Fitting is done at the sensor location $\underline{P}_d = [0.62\ 0.12]\ \mathrm{m}$ with inputs v_∞ and n. The objective is to fit the slipstream model output, v_d, to v_∞ and n. We only fit the model within the expected operating range of the agile fixed-wing platform, approximately $v_\infty \in [0\ 15]\ \mathrm{m/s}$ and $n \in [0\ 145]\ \mathrm{rev/s}$. A grid of $10,000$ points covering this operating range is constructed and provided to the slipstream model, the output of the slipstream model is then fit to the input using MATLAB's curve fitting toolbox. A good fit is observed with a polynomial that is quadratic with respect to v_∞ and linear with respect to n, given by

$$v_d(v_\infty, n) \approx 0.040 + 0.744n + 0.528v_\infty + 0.002v_\infty n + 0.010\,(v_\infty)^2 \tag{3.8}$$

The fitted surface captures the slipstream model very well at location $\underline{P}_d = [0.62\ 0.12]\ \mathrm{m}$ across our expected flight regime, with an R^2 value of 0.9998 and $RMSE$ of $0.0420\ \mathrm{m/s}$.
Now that the slipstream model is fit as a polynomial surface, H_k can be assessed at the a priori freestream velocity estimate $\check{q}_k$ using Eq. (3.8) as follows

$$
\begin{aligned}
H_k &= \left.\frac{\delta\left[0.010\, q_k^2 + (0.528 + 0.002n)\, q_k + (0.744n + 0.040)\right]}{\delta q_k}\right|_{\check{q}_k} \\
H_k &= 0.020\, \check{q}_k + 0.528 + 0.002n
\end{aligned}
\tag{3.9}
$$

Since the q_k is not directly dependent on q_{k-1}, the state transition variable F_{k-1} is forcibly set to 1 to ensure that the variance value does not drop to zero between iterations of the estimator. This represents the assumption of a constant variance between iterations. While a more sophisticated approach, accounting for the presence of the wind estimator, such as the sigma point transformation approach used in [55], may provide better performance, the simple approach used in this thesis it is shown to work well both offline, and online in experimental flight tests.

The remainder of the filter is assembled using the steps detailed in Section 3.2.3. The process and model noise variance, labelled Q and R respectively, can be challenging to measure for the physical system. The approach taken in this work is to tune the two parameters using offline estimation tests on flight data. The initial value of R comes from a measurement of noise in the airspeed sensor reading. This tuning is relatively simple for the EKF-SAS since Q and R are scalar, and while it is not ideal, we show that it leads to a cascaded wind estimator that provides suitable performance for our wind rejection needs.

Other implementations of the synthetic airspeed sensor have been shown to work well both in offline and online testing. One alternative developed alongside the EKF-SAS is based on an exponential filter and is denoted the EF-SAS. More details about the EF-SAS, including experimental results, can be found in Appendix C, but that implementation was not pursued further in the present work.

3.5 Results

Now that the synthetic airspeed sensor is developed, we proceed to examine its effectiveness when used in the cascaded wind estimator architecture.

Two types of tests are conducted, offline tests and online tests, with the naming convention referring to whether the cascaded wind estimator is run on the flight computer in experiments or using experimental data after the flight test is complete. Offline tests offer us the opportunity to easily tune the estimator and examine its performance, however

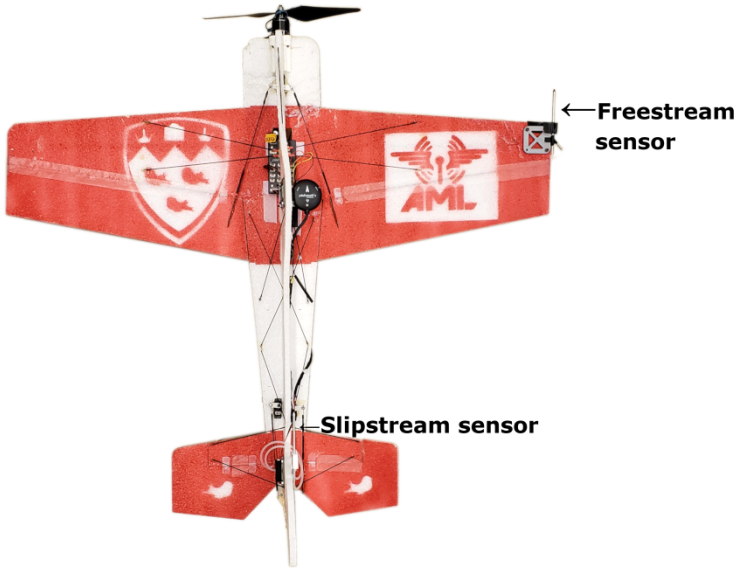

Figure 3.5: Agile fixed-wing UAV instrumented with wingtip and tail airspeed sensors measuring freestream and slipstream respectively

online tests are still required to ensure that the cascaded estimator works well on flight hardware. It should be noted that the offline test does not use any simulated sensors, all measurements are experimental.

There is a critical difference between the offline and online tests, aside from whether they are run during flight or post-flight, that is the wind estimator block they use. As discussed in Section 3.2.1, the cascaded wind estimator is designed to be easily portable between different systems, which may have premade wind estimators. Our chosen flight computer autopilot, PX4, incorporates its own wind estimator which is tightly coupled to its state estimator, since we assume this estimator is well-made, we wish to use it over our basic wind estimator designed in Section 3.2.3. As such, the wind estimator block seen in Figure 3.2 is occupied by the PX4 EKF2 wind estimator in online tests, and the basic wind estimator in offline tests. The basic wind estimator is used for offline tests out of necessity, as the PX4 estimator is not available in our offline testing environment.

3.5.1 Offline testing

Offline testing requires in-flight measurements of v_d, v_∞, and n, which are recorded in flights similar to those conducted in Chapter 2 using the instrumented platform shown in Figure 3.5. Since this testing also measures freestream, we are able to run the wind

estimator block using both the synthetic airspeed sensor and the freestream sensor to evaluate the performance of the synthetic sensor. Ideal performance would result in a matching wind estimate between the freestream sensor and the synthetic airspeed sensor, however, this is not expected. The offline estimator runs at an effective rate of 100 Hz. Key parameters for the cascaded wind estimator in this configuration are given in Table 3.1.

Table 3.1: Cascaded wind estimator configuration parameters used in offline EKF-SAS testing.

EKF parameters	value
Q	$0.1\ \mathrm{m}^2/\mathrm{s}^2$
R	$0.25\ \mathrm{m}^2/\mathrm{s}^2$
Wind estimator parameters	
$\boldsymbol{Q}$	$\mathrm{diag}(0.01, 0.01, 0.01)\ \mathrm{m}^2/\mathrm{s}^2$
R	$2\ \mathrm{m}^2/\mathrm{s}^2$

The results of this testing, including the slipstream measurement, freestream measurement, and freestream estimate are overlaid in Figure 3.6, as is the estimated variance from the EKF-based synthetic sensor.

Figure 3.6 shows that EKF-SAS estimates freestream reasonably well, with reasonable noise performance compared to the slipstream measurement. However, there is an issue with sudden drops in airspeed that is apparent when examining the test results. It is expected that these sudden drops occur under oblique flow, when airflow may enter the pitot probe's static port. The estimate is affected as the sharp decrease in the airspeed measurement under oblique flow is not modelled in the estimator.

Figure 3.7 shows the offline testing results for the cascaded estimator using the EKF-SAS and the traditional wind estimator using the freestream measurement. Wind direction appears to match the traditional wind estimator reasonably well, but greater error is seen in the magnitude. It is also clear that the wind estimate is affected by the oblique flow airspeed sensor readings.

The performance of the EKF-SAS, while suffering from high noise, which is to be expected when using propeller slipstream measurements, appears to be adequate for our expected wind estimation needs.

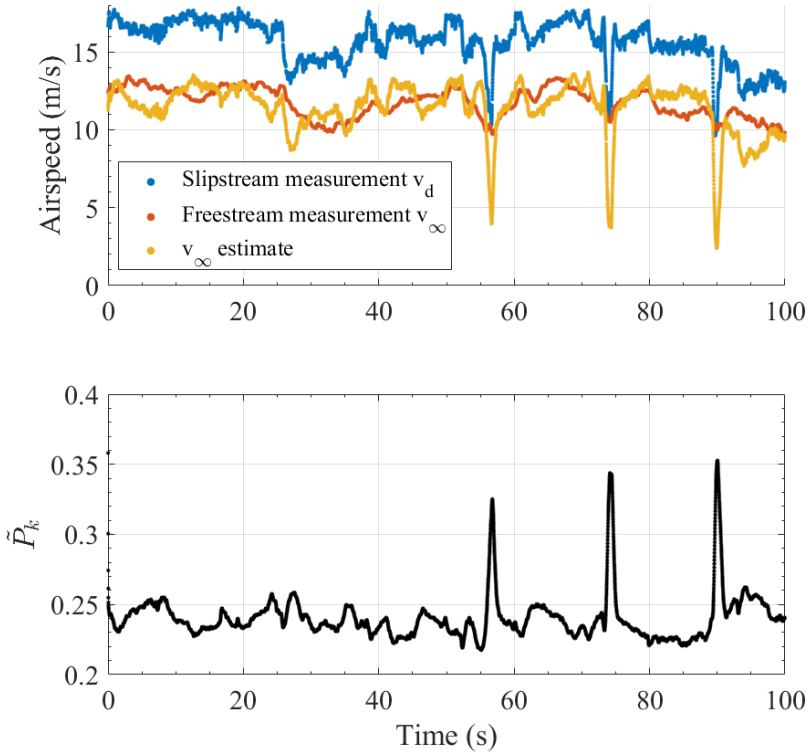

Figure 3.6: The top plot shows measured v_∞ and v_d from outdoor flight test, compared to v_∞ estimated using the EKF based synthetic airspeed sensor. The bottom plot shows the estimated variance.

3.5.2 Experiment

Now that reasonable performance is observed in offline tests, we proceed to test the cascaded wind estimator in experimental flight, with all estimation running online. This flight test uses only a single airspeed sensor, placed in the slipstream at $\underline{P}_d = [0.62\ 0.12]$ m. The platform configuration can be seen in Figure 3.8. The synthetic airspeed sensor is implemented inside of PX4 so that it requires a minimal amount of new firmware development, and can share its freestream estimate with other stock firmware components. The firmware implementation of the synthetic airspeed sensor is elaborated on in Appendix B.

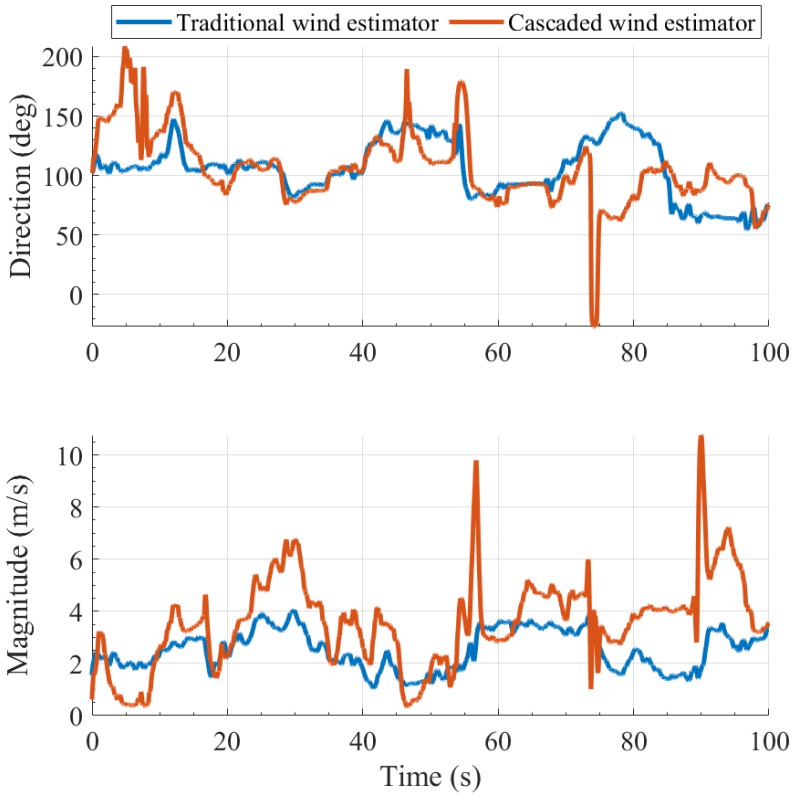

Figure 3.7: Estimated wind magnitude and direction using the traditional wind estimator (blue), and the cascaded wind estimator (orange). The traditional wind estimator relies on the measured v_∞, whereas the cascaded wind estimator uses the EKF based synthetic wind estimator to estimate v_∞.

3.5.2.1 Flight test results

This flight test uses the EKF-SAS, with $Q, R = 10\ \mathrm{m}^2/\mathrm{s}^2$, and the PX4 state estimator with default settings to compose the cascaded wind estimator. The platform is piloted semi-autonomously, in the same manner as the tests conducted in Chapter 2, and the cascaded estimator is run completely online. Since this test is focused on the wind estimate, agile manoeuvring is avoided to prevent inaccurate airspeed readings under oblique flow conditions.

This test does not include a separate wingtip freestream sensor, as such we focus on the results of the wind estimator instead of the freestream estimate. We look for apparent convergence in the wind estimate at a reasonable wind speed to indicate good performance.

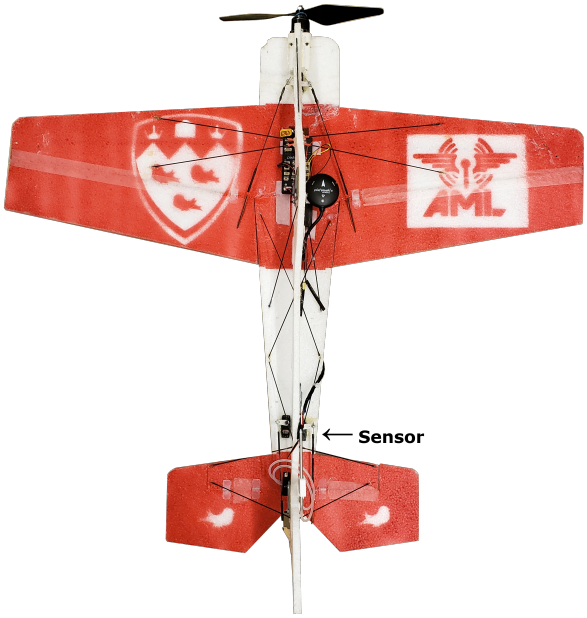

Figure 3.8: Experimental platform configuration for on-line wind estimation flight test, a single sensor, near the platform's tail, at $\underline{P}_d = [0.62\ 0.12]\ \mathrm{m}$, is used.

Figure 3.9 shows the slipstream measurement and estimated freestream speed from the EKF-SAS after the platform was launched at $t \sim 0$ s. The results in this figure show that the freestream estimate is still noisy, particularly before $t \sim 40$ s, but behaviour and the average airspeed appears reasonable for this platform.

The wind estimate, shown in Figure 3.10, appears to show better performance when compared to earlier offline tests. While no ground truth wind data was recorded, the wind magnitude and direction estimates both appear to converge after approximately $t \sim 40$ s. The better behaviour compared to offline test results using the basic wind estimator is expected to be due to the PX4 wind estimator. This estimator is both more sophisticated, and fuses synthetic sideslip estimates to increase wind estimation performance. Synthetic sideslip is not solely responsible for the good wind estimation performance, however. Tests using no airspeed sensor and only synthetic sideslip were observed to be very noisy and inconsistent with expected wind behaviour, featuring sharp magnitude and direction changes. As such, it likely that this additional synthetic sensor only serves to increase the performance of the estimate. We deem this to be acceptable performance of the online cascaded wind estimator.

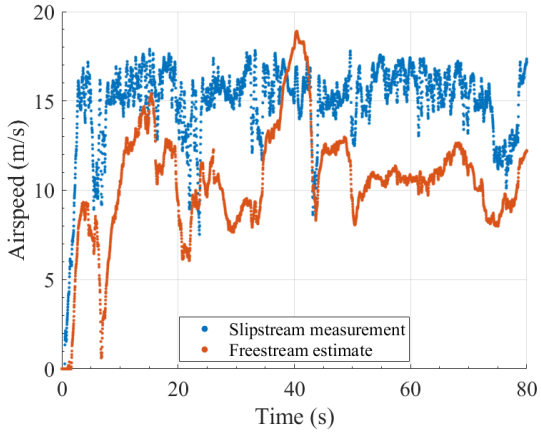

Figure 3.9: Slipstream measurement and freestream estimate for online cascaded wind estimation flight test. The platform is launched at $t \sim 0$

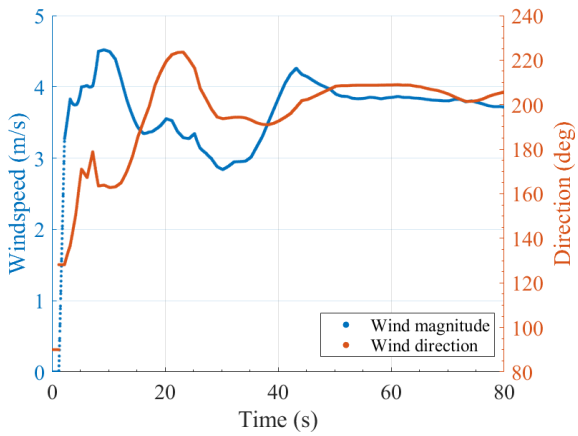

Figure 3.10: Wind estimate results for online cascaded wind estimation flight test. The platform is launched at $t \sim 0$

Before proceeding to wind disturbance rejection, we also demonstrate a potential advantage from placement of the airspeed sensor inside the slipstream, the use of slipstream in the control allocator.

3.6 Use of slipstream measurement in control allocation

Since our sensor location is inside the slipstream and near the tail of the platform, we also demonstrate the ability to use the airspeed measurement for control allocation.

Control allocation on this platform involves the challenge of mapping a requested moment from an attitude controller to deflections of the control surfaces. The primary challenge here is that control surface effectiveness is dependent on airflow, meaning that at low airspeeds, much higher control deflections are required to generate a given moment on the vehicle than at high airspeeds. Coupling this with the influence of propeller slipstream on flow around the control surfaces and the correct relation between deflection and moment can be challenging to compute.

In [56], this mapping is computed for an agile fixed-wing platform by first estimating slipstream velocity, V_{d_s}, near the tail using momentum theory, and assuming complete flow development (where γ, introduced in Chapter 2, is $\gamma = 2$)

$$V_{d_s} = \sqrt{V_\infty^2 + \frac{2T}{\rho \pi R_p^2}} \tag{3.10}$$

This V_{d_s} is then used to calculate the moment $\underline{m}_s$ generated by the surface using the deflection δ_s as

$$\underline{m}_s = \underline{l}_s \times c_s V_{d_s}^2 \delta_s \hat{d}_s \tag{3.11}$$

Where c_s is a coefficient measured in ground tests, $\hat{d}_s$ is the direction of force generated by the surface for a positive deflection, and $\underline{l}_s$ is the distance from the surface's aerodynamic centre to the platform's centre of mass.

We propose instead to use the measurement inside the slipstream, which is already being used for wind estimation, with Eq. (3.11) instead of Eq. (3.10). This is shown purely as a demonstration, and more work must be undertaken to fully evaluate its performance compared to the approach used in [56]. In addition, there are some important considerations regarding this approach, firstly, the slipstream will change across the control surface which isn't accounted for, secondly the slipstream model is not used here, only direct measurement, meaning we are not calculating the true flow velocity at the surface, only measuring the velocity near it. Finally, this approach works due to the mounting location chosen, as it is near the platform's tail surfaces, and is not universal to all airspeed measurements inside the slipstream. This method shows that needing to mount an airspeed sensor inside the slipstream can be potentially beneficial, improving the platform's attitude tracking, and not only detrimental, requiring a synthetic airspeed sensor for practical uses.

To demonstrate this approach we show the platform tracking an 'aggressive turn around' (ATA) attitude maneuver, tracked by the attitude controller detailed in Section 4.1.3. The pitch reference can be seen in Figure 3.11 and the roll reference in Figure 3.12. This maneuver consists of a pitch up to vertical where the platform then follows a gradual 180° roll maneuver with a pitch down. The result of this maneuver is that the platform ends the maneuver with the same pitch and roll it started with, but at a 180° yaw difference. The measured pitch using both momentum theory and direct measurement control allocation can be seen in Figure 3.11. The measured roll for both the momentum theory and direct measurement approaches can be seen alongside the reference in Figure 3.12. It should be noted that this maneuver reaches an Euler angle singularity at a pitch

of 90° causing discontinuities in roll. As such, the roll data in 3.12 required additional post-processing.

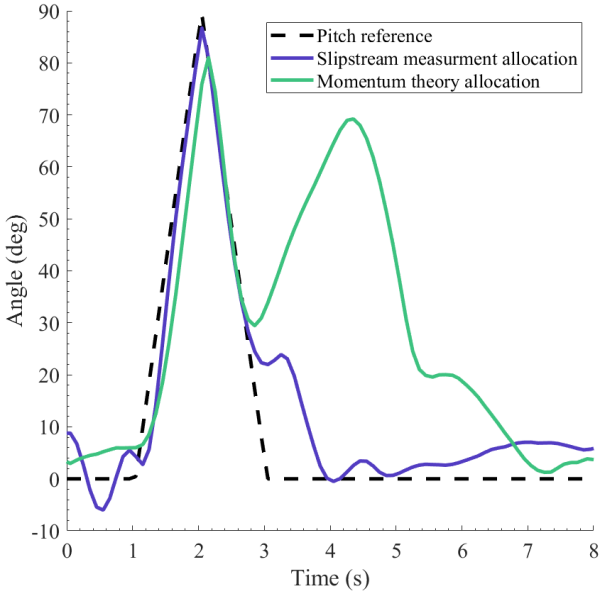

Figure 3.11: Pitch reference and measurements for the ATA maneuver using momentum theory control allocation and slipstream measurement control allocation.

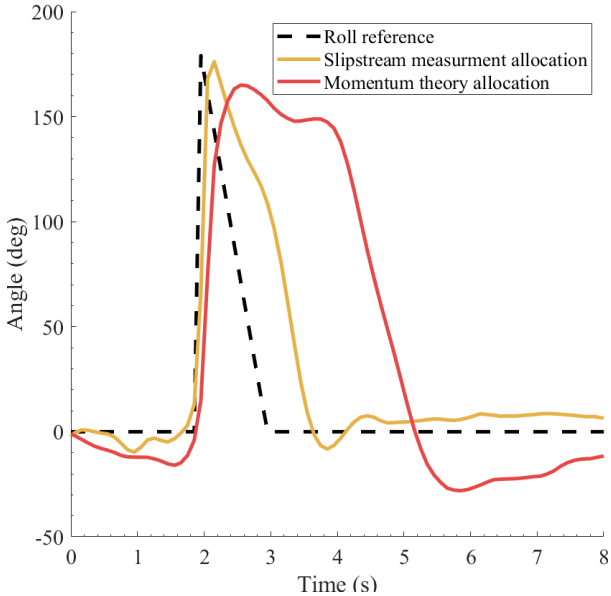

Figure 3.12: Roll reference and measurements for the ATA maneuver using momentum theory control allocation and slipstream measurement control allocation.

The maneuvers are also plotted in a 3-dimensional visualization, Figure 3.13 shows the experimental flight test results of this maneuver using direct slipstream measurement, and Figure 3.14 shows the results using the momentum theory method.

Figures 3.11 through 3.14 clearly show that performance is improved when using the direct slipstream measurement, however to draw any conclusions, a tuning of attitude gains would need to be carefully performed.

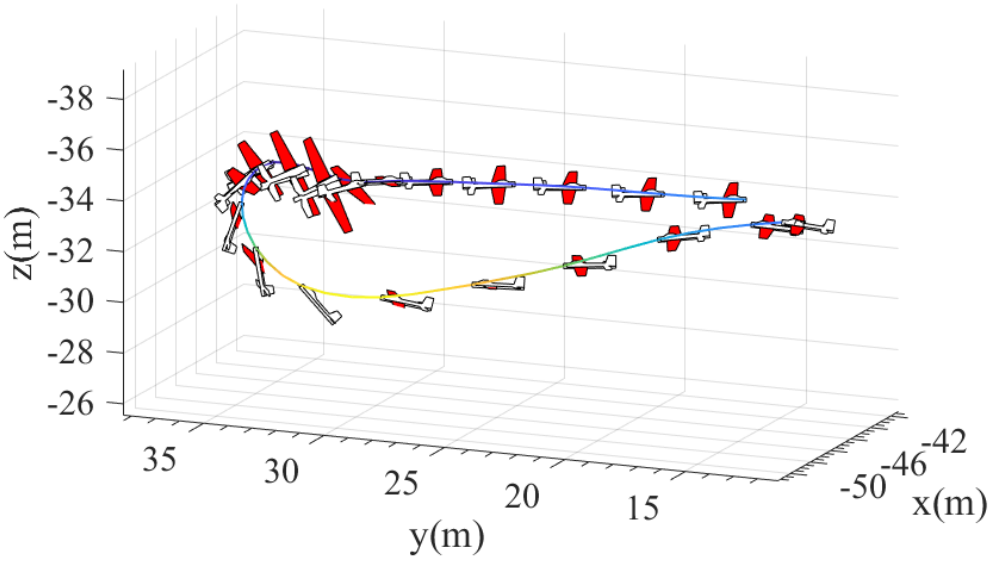

Figure 3.13: Experimental result of ATA maneuver using direct slipstream measurement in the control allocation

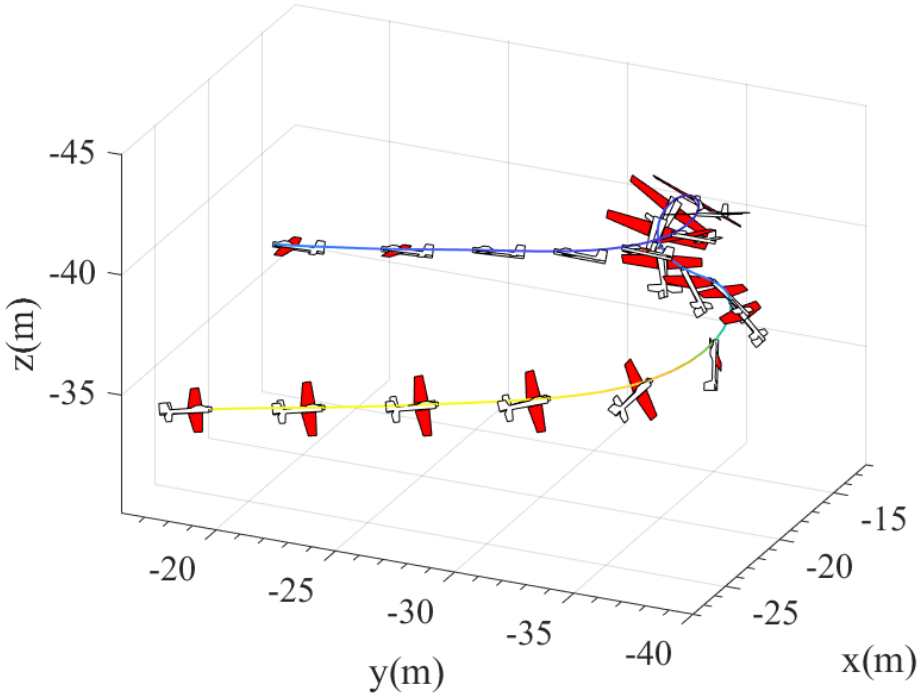

Figure 3.14: Experimental result of ATA maneuver using momentum theory control allocation

Chapter 4

Wind rejection

Now that we have access to a well-performing, on-line wind estimator aboard the agile fixed-wing platform, we continue on to use this estimate for wind disturbance rejection.

The scope of this chapter is restricted to wind rejection, and does not consider general position control of an agile platform. Position tracking controllers for agile platforms are already established and show good performance, such as the controllers developed in [57] and [5], as such our objective in this section is to develop a wind rejection method that can be easily integrated with these established position controllers. The controllers designed in [57] and [5] use feedback control, i.e., using some function of error from the desired state as the controller input. In contrast, the wind rejection approach we propose here is based on feedforward control. The feedforward wind rejection controller directly augments the signals from the position controller based on a wind estimate, without explicit consideration of error. Due to this design, we require a 'base' position controller to run alongside our wind rejection feedforward controller. The base-controller is responsible for tracking a position reference, while the feedforward controller increases performance by reducing the effect that wind disturbance has on flight. This feedforward approach must be applied carefully, so that it does not reduce the performance of the base-controller or make it unstable, but we show in this chapter that it is an effective approach to reject wind disturbances.

The remainder of this chapter organized as follows: first we provide a brief description of the base-position controller, then the design of the feedforward wind rejection controller will be detailed, finally, a series of simulation and experimental results will be shown. Experimental results will be first shown using a completely stock wind estimator

in PX4 before we show results using the synthetic airspeed sensor developed in Chapter 3.

4.1 Feedback position control

This section will summarize the trajectory tracking feedback controller developed in [57]. In this thesis we use this controller as the 'base-controller' for both simulation and experimental flight tests. The base-controller is minimally modified from [57] for use in the present work.

Fully autonomous control of agile fixed-wing vehicles presents considerable challenges for conventional fixed-wing UAV controllers. Traditional controllers are generally designed for a single flight state, for example level-flight *or* hover, this imposes limitations on agile vehicles which need to transition freely between hover and level flight [56]. To combat this limitation, new control methods relying on thrust-pointing have been developed for trajectory tracking which are valid in all phases of flight from low speed hover manoeuvres to acrobatics and level flight.

The trajectory tracking objective of these controllers is slightly different from the path-following control objective presented in Chapter 1. In path-following, the control objective is to converge to a geometric path in space, while for trajectory tracking, the control objective is to 'chase' after a time parametrized reference point. The main difference of the latter approach is the time dependence, as the platform is commanded to reach a certain position at a certain time. While trajectory tracking is often used for multi-rotors, such as in [58] and [59], it is less commonly used for fixed-wings which must maintain a minimum airspeed to remain airborne. However, the unconventional abilities of agile fixed-wing UAVs, particularly their ability to hover, enables the use of trajectory tracking.

The agile fixed-wing trajectory tracking controller used in this thesis, developed by Hernández and Nahon in [57], is selected since it has very good performance, does not require attitude references, and was validated on the same West Michigan Park Flyers platform used in the present work. The remainder of this section will summarize the controller developed in [57], which is denoted the base-controller or feedback controller in later sections of this thesis.

Before discussing the base-controller we first provide a summary of the vehicle dynamics derived in [57].

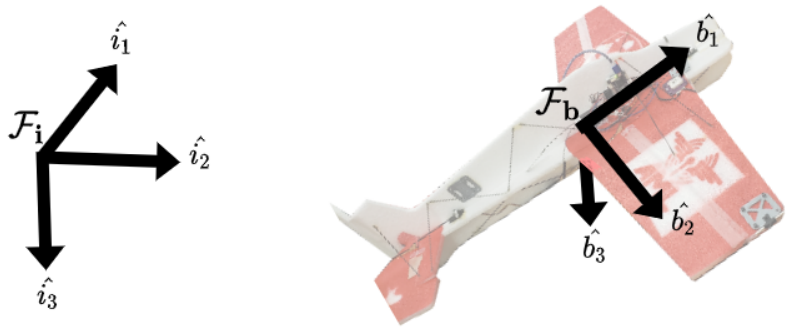

Figure 4.1: Inertial, $\mathcal{F}_i$, and body, $\mathcal{F}_b$, frame definitions. The inertial frame vectors are aligned with the North, East, and Down vectors respectively.

4.1.1 Vehicle dynamics

For the inertial and body frames defined in Figure 4.1, the platform position and velocity resolved in the inertial frame can be denoted as $\underline{P}_i$ and $\underline{V}_i$ respectively, and the translational dynamics of the platform can then be described as

$$m\underline{\ddot{P}}_i = T\hat{b}_1 + mg\hat{i}_3 \tag{4.1}$$

Where, as in Chapter 2, T represents thrust from the platform's single propeller, g is the acceleration due to gravity and m is the mass of the aircraft. Since the dynamics of the platform are assumed to be dominated by the thruster, aerodynamic forces are excluded from Eq. (4.1) [57].

The rotation from $\mathcal{F}_i$ to $\mathcal{F}_b$ is represented by the Direction Cosine Matrix (DCM) $\boldsymbol{C_{bi}}$, which can be composed using the unit vectors of $\mathcal{F}_b$ as $\boldsymbol{C_{bi}} = [\hat{b}_1\ \hat{b}_2\ \hat{b}_3]^T$. Eq. (4.1) can then be rewritten using $\boldsymbol{C_{bi}}$ as

$$\underline{\ddot{P}}_i = \frac{T}{m}\boldsymbol{C_{bi}}^T \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + g\hat{i}_3 \tag{4.2}$$

4.1.2 Position controller

The errors used for position control in [57] to track a time varying position reference $\underline{P}_r$ and velocity reference $\underline{\dot{P}}_r$ are defined as follows

$$\underline{e} = \underline{P}_i - \underline{P}_r \tag{4.3}$$

$$\dot{\underline{e}} = \underline{V}_i - \dot{\underline{P}}_r \tag{4.4}$$

Hernández and Nahon then propose a controller which outputs a reference attitude $\boldsymbol{C_{ri}^{fb}} \in SO(3)$ and thrust T_{fb} so that $\underline{e}$ and $\dot{\underline{e}}$ converge. $\boldsymbol{C_{ri}^{fb}}$ can be decomposed into a reference body frame defined by orthogonal unit vectors $\hat{r}_1^{fb},\ \hat{r}_2^{fb},\ \hat{r}_3^{fb}$, as $\boldsymbol{C_{ri}^{fb}} = [\hat{r}_1^{fb}\ \hat{r}_2^{fb}\ \hat{r}_3^{fb}]^T$. The reference nose vector is $\hat{r}_1^{fb}$ while $\hat{r}_2^{fb}$ points along the reference for the right wing, and $\hat{r}_3^{fb}$ represents the reference belly vector, which completes the right-handed frame. The controller is defined using gains $\boldsymbol{K_p}$ and $\boldsymbol{K_d}$ as

$$\frac{T_{fb}}{m}\hat{r}_1^{fb} = \boldsymbol{K_p}\underline{e} + \boldsymbol{K_d}\dot{\underline{e}} - g\hat{i}_3 \tag{4.5}$$

$$\frac{T_{fb}}{m}\left(\boldsymbol{C_{ri}^{fb}}\right)^T \begin{bmatrix}1\\0\\0\end{bmatrix} = \boldsymbol{K_p}\underline{e} + \boldsymbol{K_d}\dot{\underline{e}} - g\hat{i}_3 \tag{4.6}$$

The controller in Eq. (4.6) outputs the unit vector $\hat{r}_1^{fb}$ as the reference direction of thrust, since the propeller is assumed to be aligned with the platform's nose vector. $\hat{r}_1^{fb}$ alone cannot be used to define a unique reference attitude, $\boldsymbol{C_{ri}^{fb}}$, so the authors of [57] develop a vector projection method to allow a roll command to be used to complete the attitude reference when the platform is flying as a conventional fixed-wing. Since much of our focus in this chapter compares the performance of the base-controller running with our feedforward to the base-controller running alone, we command roll to be zero to ensure the most consistent comparison.

A slight modification from the original controller presented in [57] can be found in Eq. (4.6). This modification removes a $\ddot{\underline{P}}_r$ term that represents the acceleration of the reference trajectory. This was done for two reasons: firstly many of the trajectories used with this controller are steady, where $\ddot{\underline{P}}_r$ would not affect the controller even if present, and secondly, the inclusion of $\ddot{\underline{P}}_r$ was seen to cause a slight increase in path relative error for circular trajectories.

The base-controller does demonstrate impressive performance in all agile fixed-wing flight regimes, however, since it is only error driven in the $\hat{i}_1 - \hat{i}_2$ (North-East) plane, rejection of wind disturbances will require the platform to take on additional error. Our goal is to correct this using the feedforward controller developed and tested in the remainder of this chapter.

4.1.3 Attitude controller

To track the attitude reference, we use the geometric attitude controller developed in [57]. For reference, a brief overview of this controller is provided below.

The attitude error is defined geometrically by the authors of [57] as

$$\underline{e}_{att} = \frac{1}{2}\left(\boldsymbol{C_{bi}}\boldsymbol{C_{ri}}^T - \boldsymbol{C_{ri}}\boldsymbol{C_{bi}}^T\right)^\vee \tag{4.7}$$

The operator $(\cdot)^\vee$ is the inverse of the cross operator $(\cdot)^\times$. In this case, $(\cdot)^\vee$ maps a skew symmetric matrix in $Skew(\mathbb{R})^{3\times3}$ onto $\mathbb{R}^3$. A sample of the behaviour of the $(\cdot)^\times$ operator is shown below [60] [54].

$$\begin{aligned} \underline{\phi} &= \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} \\ (\underline{\phi})^\times &= \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix} \\ \left((\underline{\phi})^\times\right)^\vee &= \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} \end{aligned} \tag{4.8}$$

The control law below is then proposed in [57] to drive $\underline{e}_{att}$ to zero and track the reference body rate $\underline{\omega}_r$, where $\underline{m}_d \in \mathbb{R}^3$ is the desired moment on the vehicle in the body frame, $\boldsymbol{K_{ap}}$ and $\boldsymbol{K_{ad}}$ are symmetric gain matrices, and $\underline{\omega}$ is the platform's current body rate.

$$\underline{m}_d = \boldsymbol{K_{ap}}\underline{e}_{att} + \boldsymbol{K_{ad}}\left(\boldsymbol{C_{bi}}\boldsymbol{C_{ri}}^T\underline{\omega}_r - \underline{\omega}\right) \tag{4.9}$$

$\underline{m}_d$ can then be mapped to the control surface deflections of the platform, $\underline{\delta}_a$.

4.2 Feedforward wind rejection controller

The main objective of the feedforward wind rejection controller is to make the closed-loop dynamics of the platform in a windy environment as close as possible to the closed-loop dynamics in still air. Said another way, wind impacts the closed loop dynamics of

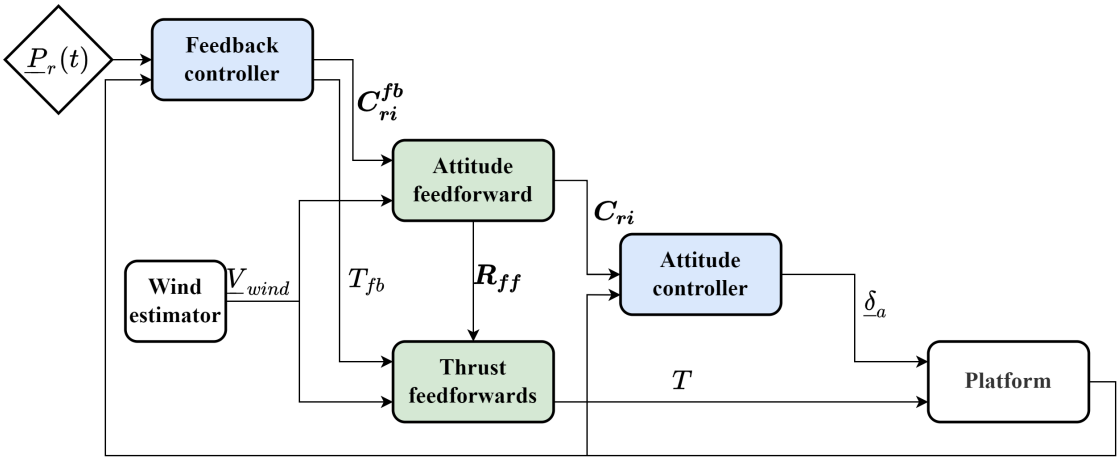

Figure 4.2: Block diagram of feedforward/feedback controller. Feedback elements are shown blue and feedforward elements are shown in green.

base-controller, so we supplement the base-controller with a feedforward wind rejection controller to minimize this impact as much as possible.

The wind rejection controller, also referred to as the feedforward controller, is a modular controller using a wind estimate as an input. The input is then used to modify the control commands from the base-controller in some way to minimize the effect of wind on flight. In a sense, the feedforward controller acts as an intermediary between the base-controller and either the platform's actuators or the inner-loop attitude controller. When the wind estimate is zero, it has no impact on the signals from the base-controller, but when wind is present it will modify them to increase performance.

The feedforward controller is formed out of three elements, the first of which is an attitude feedforward which calculates a trim attitude to reject cross-path wind effects. The trim attitude is then combined with the attitude signal from the base controller. The second component is a feedforward acting on thrust, which coordinates an increased thrust based on the rotation applied by the attitude feedforward. The final component is a second feedforward on thrust, which compensates for wind disturbances along the aircraft's longitudinal axis.

The block diagram in Figure 4.2 shows the complete feedback/feedforward control architecture. Elements in blue denote the feedback components and are developed in [57], elements in green are feedforward components and are developed in this work. Elements shown in white are either references, the plant, or abstracted components such as estimators.

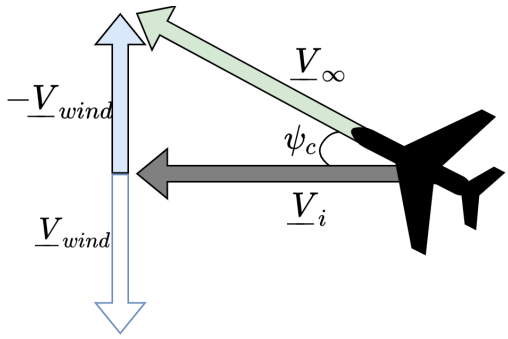

Figure 4.3: Wing triangle definition

While the feedforward controller is only expected to be used experimentally with wind estimators that do not provide a vertical wind estimate, we design all three components so that they can easily be used to reject up or downdrafts if one is available. Additionally, this wind rejection method is primarily intended for flight close to level, with the motivation being to reject wind disturbances during the large amount of time an agile fixed-wing may spend in level flight, travelling between destinations. A different approach, more akin to multi-rotor or tailsitter wind rejection would likely be needed for wind rejection of an agile platform near the hover state.

4.2.1 Attitude feedforward

The attitude feedforward's objective is to output an attitude DCM, $\boldsymbol{R}_{\not ff}$, which allows the platform to maintain a desired ground velocity direction when wind is present. The efficacy of this method is most obvious in straight and level flight, but simulation and experiments show it is effective even when the controller is following a more complex trajectory.

$\boldsymbol{R}_{\not ff}$ is calculated kinematically, using the wind triangle, meaning no aerodynamic modelling of the vehicle is required. The wind triangle has been mentioned in previous chapters but will be detailed further here as it is critical to the feedforward controller. The wind triangle represents the relationship between the three velocities: $\underline{V}_\infty$, $\underline{V}_i$, and $\underline{V}_{wind}$. Figure 4.3 shows the interaction between vectors $\underline{V}_\infty$, $\underline{V}_i$, and $\underline{V}_{wind}$. $\underline{V}_\infty$ can be written as [50]

$$\underline{V}_\infty = \underline{V}_i - \underline{V}_{wind} \tag{4.10}$$

Assuming that the platform is flying without sideslip, $\underline{V}_\infty$ should be aligned with the platform's nose vector $\hat{b}_1$. This restriction allows Eq. (4.10) to be rewritten as

$$\hat{b}_1 = \frac{(\underline{V}_i - \underline{V}_{wind})}{|\underline{V}_\infty|} \tag{4.11}$$

The attitude feedforward rotation, $\boldsymbol{R_{ff}}$, can now be built as a rotation that aligns the unit vectors $\hat{V}_i$ and $\hat{V}_\infty$. In this thesis, $\boldsymbol{R_{ff}}$ is computed using the axis-angle formulation, the calculation for which can be seen below [60] [54]. A similar approach is used by Liu et al. in [16] where the authors incorporate pitch and yaw wind trim points into their guidance controller.

$$\hat{V}_i \times \hat{V}_\infty = \sin(\psi_c)\hat{a} \tag{4.12}$$

$$\boldsymbol{R_{ff}} = \cos(\psi_c)\mathbf{I} + (1 - \cos(\psi_c))\,\hat{a}\hat{a}^T + \sin(\psi_c)\hat{a}^\times \tag{4.13}$$

ψ_c represents the angle between $\hat{b}_1$ and $\underline{V}_i$, if we restrict $\underline{V}_\infty$, $\underline{V}_i$, and $\underline{V}_{wind}$ to the North-East plane, ψ_c becomes what is known in aircraft mechanics as the crab angle.

The attitude reference from the base-controller $\boldsymbol{C_{ri}^{fb}}$ is then rotated by $\boldsymbol{R_{ff}}$ to generate the final reference attitude $\boldsymbol{C_{ri}}$, which is sent to the attitude controller.

$$\boldsymbol{C_{ri}} = \boldsymbol{C_{ri}^{fb}}\boldsymbol{R_{ff}} \tag{4.14}$$

$\boldsymbol{C_{ri}}$ can be broken down into the orthogonal unit vectors $\hat{r}_1,\ \hat{r}_2,\ \hat{r}_3$ as $\boldsymbol{C_{ri}} = [\hat{r}_1\ \hat{r}_2\ \hat{r}_3]^T$.

4.2.2 Thrust feedforward I

The objective of our first thrust feedforward, denoted T_{ff_1}, is to compensate for any effects that the attitude feedforward may have on the dynamics of the base-controller.

Of course, it can be expected that changing the base-controller's attitude reference with an external component like $\boldsymbol{R_{ff}}$ can have unintended effects on performance. This is because rotating the reference attitude changes the reference thrust vector $\hat{r}_1^{fb}$ without changing the reference thrust T_{fb}. This causes the base-controller to not receive the thrust that it requests in the direction it originally requested it, which is critical for tracking performance. When the attitude feedforward is applied, instead of the base-controller receiving a force T_{fb} in direction $\hat{r}_1^{fb}$, it will receive T_{fb} along $\hat{r}_1$. The thrust along the original direction, $\hat{r}_1^{fb}$ will be reduced by the attitude feedforward based on ψ_c. To avoid

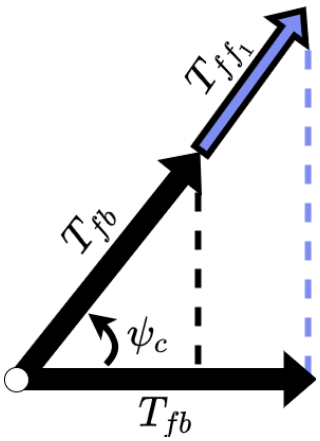

Figure 4.4: T_{ff_1} compensating for the change in thrust vector due to the attitude feedforward

this reduction, we introduce a thrust feedforward term T'_{ff_1} which ensures that the final thrust along $\hat{r}_1^{fb}$ is always equal to T_{fb}.
The thrust reduction along $\hat{r}_1^{fb}$, and how T'_{ff_1} compensates, is visualized in Figure 4.4. To satisfy the design goal of T'_{ff_1} it must satisfy the following relationship

$$T_{fb} = \left((T_{fb} + T'_{ff_1})\hat{r}_1\right) \cdot \hat{r}_1^{fb} \tag{4.15}$$

$$T_{fb} = \left((T_{fb} + T'_{ff_1})(\boldsymbol{R_{ff}}^T \hat{r}_1^{fb})\right) \cdot \hat{r}_1^{fb} \tag{4.16}$$

$$\zeta' = \left[\frac{1}{\hat{r}_1^{fb^T} \boldsymbol{R_{ff}}^T \hat{r}_1^{fb}} - 1\right] \tag{4.17}$$

$$T'_{ff_1} = \zeta' \, T_{fb} \tag{4.18}$$

The constraint for T'_{ff_1} is shown in Eq. (4.15), with a T'_{ff_1} satisfying this condition being solved for in Eq. (4.18). It is clearly seen that the equation for ζ', Eq. (4.16), contains a singularity. If $\hat{r}_1^{fb}$ is orthogonal to $\hat{r}_1$, ζ' is undefined. This is easily avoided by saturating ζ' if it exceeds $\pm\zeta_{max}$. When this saturation is employed, $\pm\zeta_{max} = \sec(\pi/4) - 1$, which is rarely reached in simulation or experimental tests, and only activates when $\psi_c > 45°$.

The saturated ζ' is denoted ζ, and results in a saturated T'_{ff_1}, similarly denoted T_{ff_1}. The calculation of T_{ff_1} is as follows

$$\zeta = \begin{cases} \zeta' & |\zeta'| < \sec(\pi/4) - 1 \\ \sec(\pi/4) - 1 & |\zeta'| > \sec(\pi/4) - 1 \end{cases} \tag{4.19}$$

$$T_{ff_1} = \zeta \, T_{fb} \tag{4.20}$$

4.2.3 Thrust feedforward II

Since the base-controller is error driven in the North-East plane, rejecting wind disturbances using it alone will always require a change in tracking error, where tracking error is the distance between the platform and the reference point. Both the attitude feedforward and T_{ff_1} alleviate some of this disturbance, but are most effective in a cross-wind, with their wind rejection authority dropping to zero as $\underline{V}_i$ becomes collinear with $\underline{V}_{wind}$. Many common flight paths, such as racetracks and circular holding patterns, are closed loops and as such will at some point require flight directly up or down wind. This motivates the development of an additional feedforward term to compensate for the effect of wind in this flight condition, where wind disturbances align with the aircraft's longitudinal axis.

The base-controller is capable of tracking a constant inertial velocity while travelling along the wind, however to do so it must settle on a greater tracking error compared to a zero-wind case when flying upwind and a lower error while flying downwind. Even though these effects are undesirable in straight and level flight, the errors are also primarily contained to be along the reference track. Presuming the reference path is designed to avoid obstacles, on-track errors such as these are less concerning than cross-track errors, which cause the aircraft to fly in an environment of unknown safety. For a more complex trajectory, however, which may contain a turn aligning the platform with the wind, the time required to build up enough error to compensate for wind in the longitudinal direction can have a detrimental impact on flight performance and lead to cross-track error.

To compensate for longitudinal wind disturbances, we introduce another thrust feedforward term, denoted T_{ff_2}. T_{ff_2} is purposefully structured similarly to the velocity component of the base-controller in Eq. (4.6), where k_γ is an additional gain to tune the feedforward.

$$T_{ff_2} = m k_\gamma \boldsymbol{K_d} (\underline{V}_\infty - \underline{V}_i) \cdot \hat{r}_1^{fb} \tag{4.21}$$

While the platform is flying parallel to the wind, incorperating T_{ff_2} into Eq. (4.6) generates a combined feedback/feedforward controller where terms containing the velocity error gain $\boldsymbol{K_d}$ can be grouped into a new pseudo-error rate $\dot{\underline{e}}'$ which has a dependence on $\underline{V}_\infty$ that is not present in Eq. (4.6). This dependence on airspeed is what helps compensate for the longitudinal wind disturbance. The actual control law of the base-controller is of course not modified, but examining T_{ff_2} using this approach provides some additional insight into its behaviour.

$$\dot{\underline{e}}' = \underline{V}_i(1 - k_\gamma) + k_\gamma \underline{V}_\infty - \dot{\underline{P}}_r \tag{4.22}$$

As can be seen in Eq. (4.22) a gain of $k_\gamma = 1$ will result in an $\dot{\underline{e}}'$ without a dependence on $\underline{V}_i$, leaving only $\underline{V}_\infty$. When k_γ is set to zero, eliminating T_{ff_2}, $\dot{\underline{e}}' = \dot{\underline{e}}$. k_γ can be chosen relatively arbitrarily, and good results were observed for a wide range of values with a very reliable gain being $k_\gamma = 1$. In the remainder of this thesis, excluding Section 4.3.1.1, simulation tests use $k_\gamma = 1$. Experimental flights use either $k_\gamma = 1$ or $k_\gamma = 1/0.45 \approx 1/m$. Further examination of k_γ will be discussed in Section 4.3.1.1.

After both T_{ff_1} and T_{ff_2} are computed, they can simply be summed with T_{fb} to generate the final thrust reference T as follows,

$$T = T_{fb} + T_{ff_1} + T_{ff_2} \tag{4.23}$$

4.3 Simulation results

This section will show a series of simulation tests which validate and examine the performance of the wind rejection controller. Since the PX4 wind estimator, used for wind estimation in experiments, only provides a horizontal wind estimate, most of the simulation tests presented here restrict the feedforward controller to two dimensions with no vertical component. This is done by projecting vectors $\underline{V}_\infty$, $\underline{V}_i$, and $\underline{V}_{wind}$ onto the North-East plane. Simulation results for a three-dimensional wind rejection case will be presented near the end of this section, but unless otherwise noted, the feedforward controller is operating only on the North-East plane.

The simulation environment uses the agile fixed-wing dynamics model developed throughout [6, 61–63] to model the flight behaviour of a West Michigan Park Flyers McFoamy platform. This model consists of the slipstream and thruster models discussed

Table 4.1: Feedforward configurations table used simulation tests

Configuration	Base-controller	Attitude	T_{ff_1}	T_{ff_2}
$\mathcal{A}$	×	×		
$\mathcal{B}$	×	×	×	
$\mathcal{C}$	×	×		×
$\mathcal{D}$	×	×	×	×

in Chapter 2, along with a component breakdown-based aerodynamics model capable of predicting aerodynamic forces and moments on the aircraft in all flight regimes.

Since the feedforward controller is modular, part of our testing examines its performance while some elements are deactivated. To prevent confusion when referring to different feedforward controller configurations, we assign each configuration a character from $\mathcal{A}-\mathcal{D}$. The feedforward elements active in each configuration are then denoted with an $\times$ in Table 4.1.

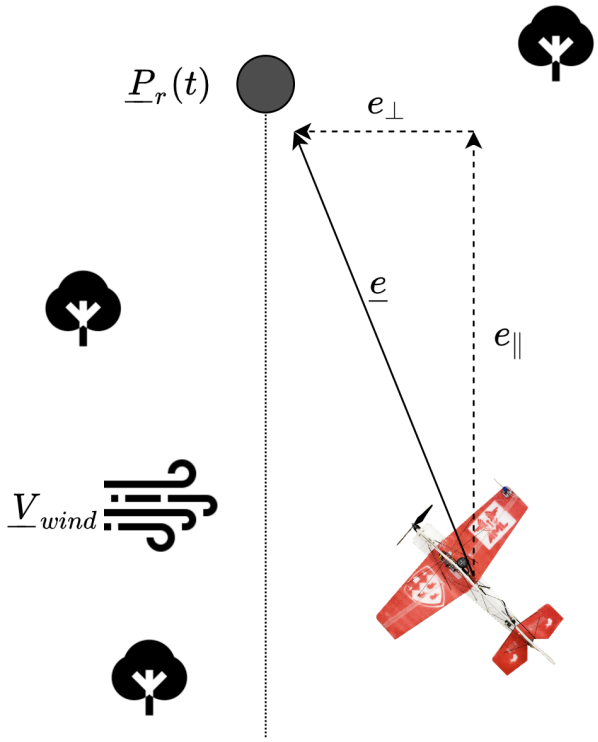

Figure 4.5: Visualization of on-track and cross-track errors, $e_\parallel$ and $e_\perp$ respectively.

Wind rejection performance is assessed by examining the errors along the reference track, denoted $e_\parallel$, and path relative errors $e_\perp$. These errors essentially represent a decom-

position of the tracking error e based on the reference trajectory. There are two primary benefits to using $e_\perp$ and $e_\parallel$ over $\underline{e}$. The first reason is that the base-controller is purely error driven in the North-East plane, meaning if $\underline{e} = 0$ no horizontal thrust force will be requested, as such, some steady state error will always be required to track a reference and overcome the drag disturbance. This steady state error can be substantial and is often largely in $e_\parallel$, investigating $e_\parallel$ and $e_\perp$ individually allows us to run test cases which can determine which portion of error is due to wind and which portion is due to the nature of the base-controller. Secondly, as mentioned earlier, for an agile platform which may fly close to obstacles, $e_\perp$ and $e_\parallel$ may not be equally undesirable as, provided that the reference path does not collide with obstacles, only $e_\perp$ poses additional risk of collision. These error definitions are also shown in Figure 4.5.

It should be noted that in simulation tests, no wind estimation is used. Wind is known precisely as we explicitly set it in the simulation environment. This exact wind value is then passed to the feedforward controller.

4.3.1 Straight and level tracking

We begin our series of simulation tests by evaluating the performance of both the base-controller and base-controller with feedforward while tracking a straight, level, and steady reference in a windy environment. Feedforward configuration $\mathcal{D}$ is used for all tests in this series.

The test series is composed of six simulation runs, to examine how effective the feedforward controller is at rejecting wind striking the platform in different directions. Each simulation follows the same reference, which travels due North at $10\ \mathrm{m/s}$, and has the same wind magnitude, a constant $V_{wind} = 5\ \mathrm{m/s}$. Additionally, the wind field is uniform, meaning it is the same at every point in the simulation space.

The wind direction is changed by 30° about $\hat{i}_3$ between each of the six simulation. The first run in the series has the platform disturbed by a Southern 180° headwind, and the final run applies a wind travelling 330° North-West. The wind directions applied in this test series with respect to the reference track are visualized in Figure 4.6.
Each simulation begins with the platform flying at an equilibrium state in still air, after which wind is instantaneously applied at $t = 5\ \mathrm{s}$. Results for $e_\perp$ and $e_\parallel$ are shown in Figures 4.7 and 4.8 respectively and demonstrate a substantial cross-track error reduction due to the feedforward controller, with the largest steady state $e_\perp$ being $0.14\ \mathrm{m}$ when feedforward is applied, compared to $3.3\ \mathrm{m}$ from the base controller operating alone.

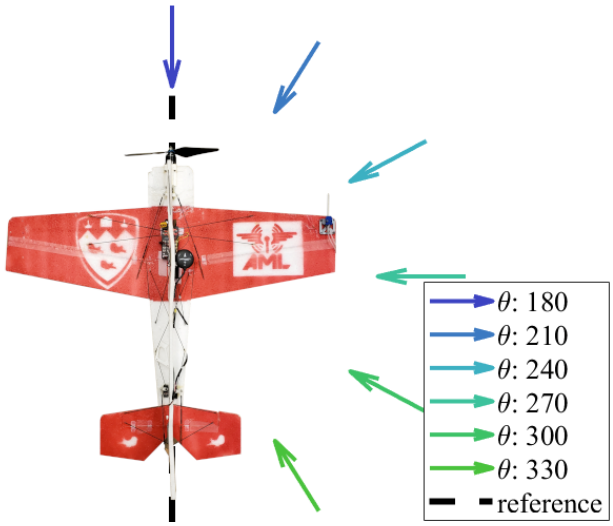

Figure 4.6: Visualizations of wind directions examined in straight and level simulations where the platform is commanded to track the black dashed line. Vectors point along the wind direction for each test.

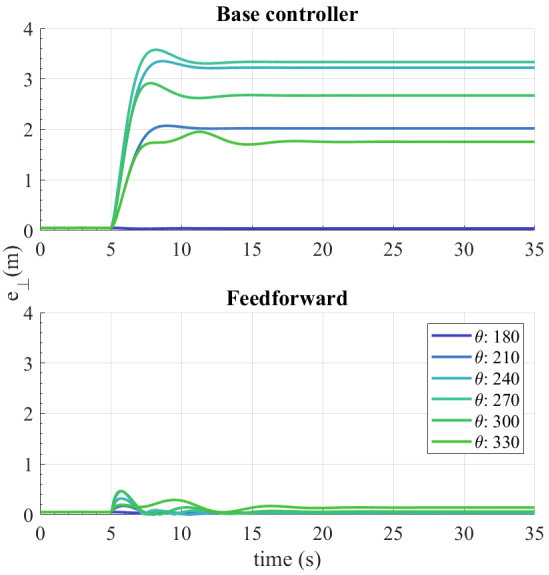

Figure 4.7: Path relative errors for a series of straight and level $10\ \mathrm{m/s}$ trajectory tracking cases with varying wind directions, with a mean wind of $5\ \mathrm{m/s}$ beginning at $t = 5\ \mathrm{s}$.

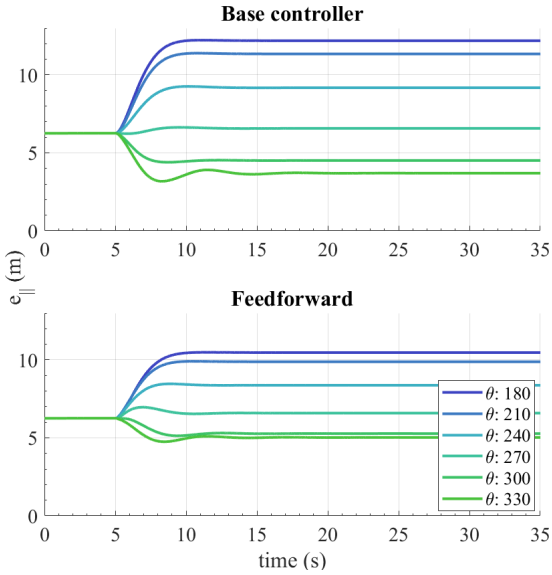

Figure 4.8: Error along path for a series of straight and level $10\ \mathrm{m/s}$ trajectory tracking cases with varying wind directions, with a mean wind of $5\ \mathrm{m/s}$ beginning at $t = 5\ \mathrm{s}$.

Performance improvements for on-track error, $e_{\parallel}$, also seen in Figures 4.7 and 4.8, are still notable, but clearly much smaller than for $e_{\perp}$. This performance can be modified by tuning of k_{γ}, however as mentioned above, errors in $e_{\parallel}$ are considered less concerning than $e_{\perp}$ errors.

4.3.1.1 T_{ff_2} tuning

Straight and level tracking tests are also used to explore how the tuning of k_{γ}, and thus T_{ff_2}, affects performance. In this thesis, outside this section, simulation tests with the feedforward controller use $k_{\gamma} = 1$ to demonstrate the replacement of inertial velocity with airspeed in the base-controller, detailed in Eq. (4.22), and experimental flight tests use either $k_{\gamma} = 1/0.45$ or $k_{\gamma} = 1$. Other values of k_{γ} are interesting to study, however, and indicate potential avenues for increased wind rejection performance.
Figures 4.9 to 4.12 show the straight and level tracking test for a variety of k_{γ} values from $k_{\gamma} = 0$ to $k_{\gamma} = 3$. The feedforward controller in configuration $\mathcal{D}$ is active for all tests in this series, however it should be noted that the $k_{\gamma} = 0$ test forces $T_{ff_2} = 0$ and as such is akin to configuration $\mathcal{B}$. Since conditions are the same as the previous straight and level tracking test, the results shown in this section can be compared to the base controller and feedforward controller with $k_{\gamma} = 1$ by referencing Figures 4.8 and 4.7

As can be seen in Figure 4.12, a gain of $k_{\gamma} = 3$ can lead to a large $e_{\parallel}$ performance increase. While the remainder of this thesis uses $k_{\gamma} = 1$ for simulation tests and either $k_{\gamma} = 1/0.45$ or $k_{\gamma} = 1$ for experimental flights, the results of this section show that further improvements in the performance of the feedforward controller are available by tuning k_{γ}. However, further work would be required to fully characterize the effect of higher k_{γ} values, and identify any potential disadvantages.

4.3.2 Circular tracking

We now proceed to examine the performance of the feedforward controller while tracking a more complex trajectory, a constant groundspeed circle. For this test case, the simulation once again begins with the platform in equilibrium, tracking a straight and level Northern reference at $10\ \mathrm{m/s}$ in still air. After $10\ \mathrm{s}$ of level equilibrium flight a constant and uniform wind of $5\ \mathrm{m/s}$ due East is instantaneously applied. This condition is held for $10\ \mathrm{s}$, allowing transients from the application of wind to settle, after which the reference begins to follow a circular trajectory with a $30\ \mathrm{m}$ radius at $10\ \mathrm{m/s}$. Since the wind direction in

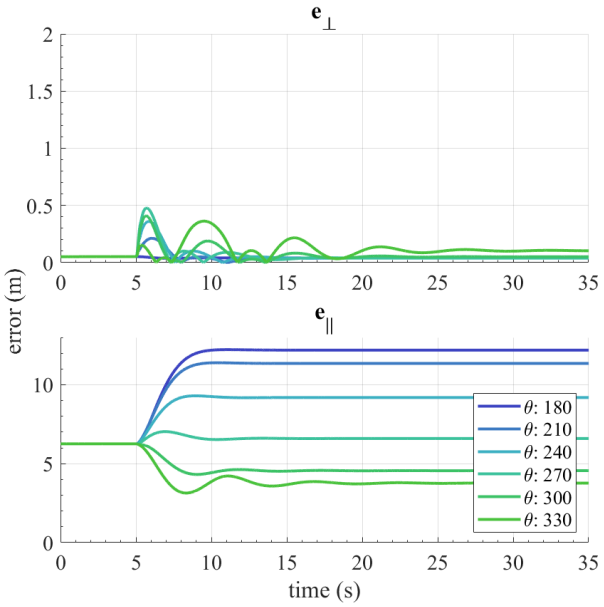

Figure 4.9: Straight and level tracking performance of the feedforward controller with $\boldsymbol{k_\gamma = 0}$ in a 5 m/s wind field.

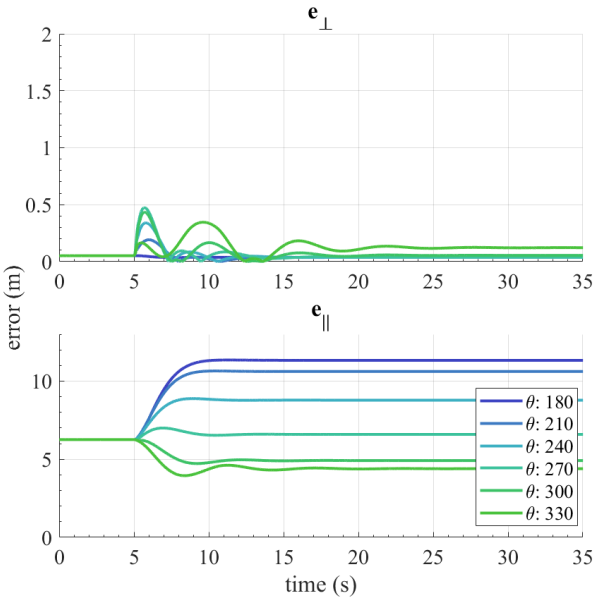

Figure 4.10: Straight and level tracking performance of the feedforward controller with $\boldsymbol{k_\gamma = 0.5}$ in a 5 m/s wind field.

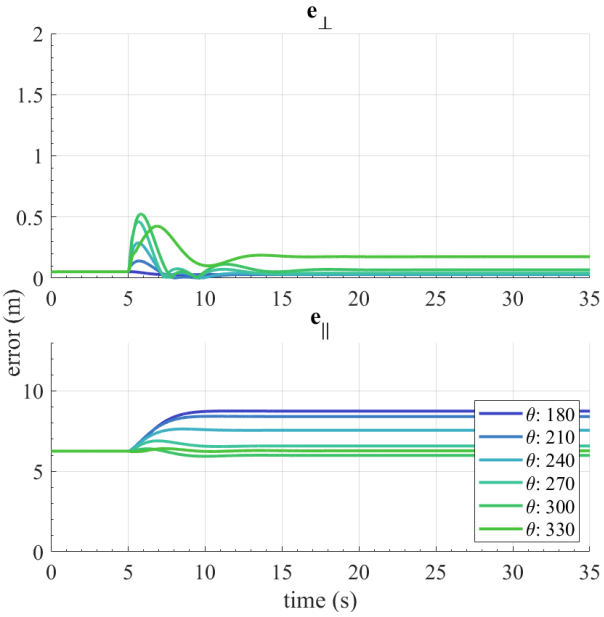

Figure 4.11: Straight and level tracking performance of the feedforward controller with $\boldsymbol{k_\gamma = 2}$ in a 5 m/s wind field.

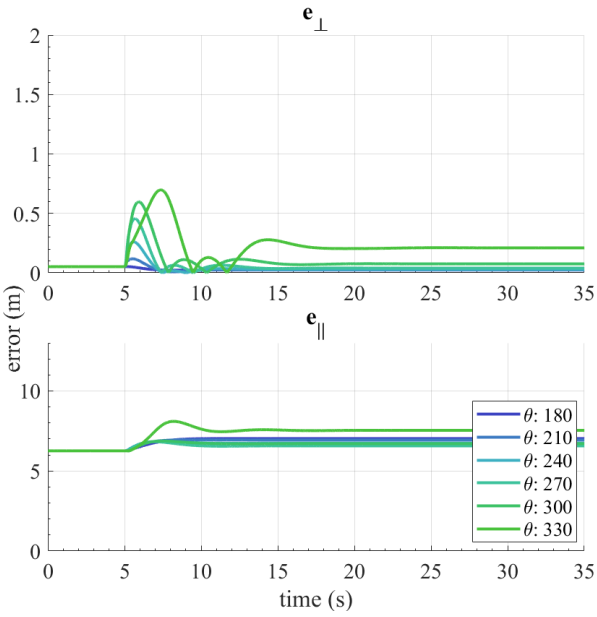

Figure 4.12: Straight and level tracking performance of the feedforward controller with $\boldsymbol{k_\gamma = 3}$ in a 5 m/s wind field.

this test is constant, and the platform tracks a circular trajectory, the platform must reject a wind disturbance that is constantly changing relative to the platform's body frame, introducing an additional layer of complexity compared to straight and level tracking. Simulation results showing the base-controller and feedforward configuration $\mathcal{D}$ can be seen in Figure 4.13

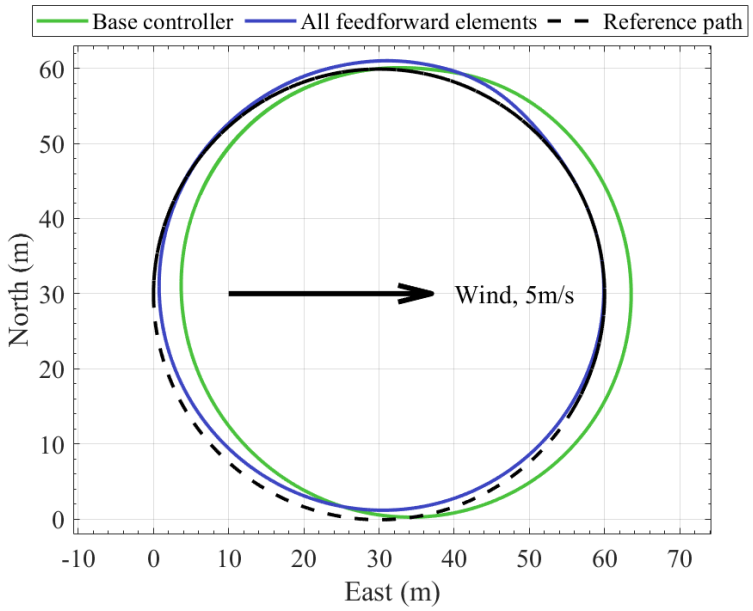

Figure 4.13: Simulated flight of base-controller and feedforward tracking a $30\ \mathrm{m}$ radius circle at a $10\ \mathrm{m/s}$ ground speed in a uniform $5\ \mathrm{m/s}$ wind field with a steady Eastern direction. The reference trajectory in this figure travels clockwise.

As can be seen in Figure 4.13, the presence of wind causes the base-controller's trajectory to be offset from the reference. This disturbance is compensated for by the feedforward controller. We use this test case to examine the impact of deactivating some components of the feedforward controller, re-running the simulation with feedforward configurations $\mathcal{A}$ through $\mathcal{C}$. We then examine the cross-track $e_\perp$ error for configurations: $\mathcal{A}$ to $\mathcal{D}$ along with the base controller to assess performance. For this circular trajectory, $e_\perp$ is defined at each time step as the minimum distance between the platform's position and the path. For each configuration, the RMS of $e_\perp$, referred to as the RMSE, is listed in Table 4.2.

Table 4.2 shows the clear performance improvement from the feedforward controller, even in a complex trajectory. Additionally, it also shows that most of the increased perfor-

Table 4.2: RMS of path relative errors for four controller configurations in simulated tracking of a $30\ \mathrm{m}$ radius circle at $10\ \mathrm{m/s}$ in a $5\ \mathrm{m/s}$ wind-field.

Configuration	Base-controller	$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$
RMSE (m)	2.54	1.00	0.94	0.89	0.84

mance comes from the attitude feedforward, with small improvements arising from both T_{ff_1} and T_{ff_2}.

4.3.3 Frequency tests

Previous tests have only considered flight in a constant wind. This is unrealistic and unrepresentative of the outdoor environment, so we conduct a series of tests with a variable wind at a variety of frequencies to further validate the feedforward controller.

The frequency tests are preformed in batches which measure the errors $e_\perp$ and $e_\parallel$ as the frequency of the wind is changed through some series of values. There are two versions of these tests, one labelled 'magnitude variation' and the other labelled 'direction variation'. In the magnitude variation tests, the platform tracks a straight and level trajectory due North at $10\ \mathrm{m/s}$ and is disturbed by a constant direction wind with a sinusoidal magnitude. The direction variation test tracks the same reference, however, in this case, the wind magnitude is constant and the vector's direction changes. Essentially, the wind direction in this test case rotates about $\hat{i}_3$ at some frequency. For both test cases, the wind field is uniform in space.

All frequency tests use the feedforward controller in configuration $\mathcal{D}$.

4.3.3.1 Magnitude variation

In the magnitude variation case, the wind direction is constant, and the magnitude changes sinusoidally. The entire wind magnitude does not change sinusoidally, as the rapid wind reversals this would present are not physical and as such are less interesting to investigate. Instead, the wind magnitude consists of some constant component $\bar{V}_{wind}$ and some variable component with a peak-to-peak amplitude of $2V^{\delta}_{wind}$. V^{δ}_{wind} varies sinusoidally at a frequency f, and is always directed Eastward, allowing the wind velocity to be expressed as

$$\underline{V}_{wind} = \left[V^{\delta}_{wind} \sin\left(2\pi f t\right) + \bar{V}_{wind}\right] \cdot \hat{i}_2 \tag{4.24}$$

Since $\underline{V}_{wind}$ always points due East, and the platform tracks a Northern reference, this test case represents a cross-wind scenario. Simulations are run at a linearly spaced series of frequencies f generated from a minimum and maximum frequency target. At each frequency, the peak-to-peak amplitudes of $e_{\perp}$ and $e_{\parallel}$ are logged.

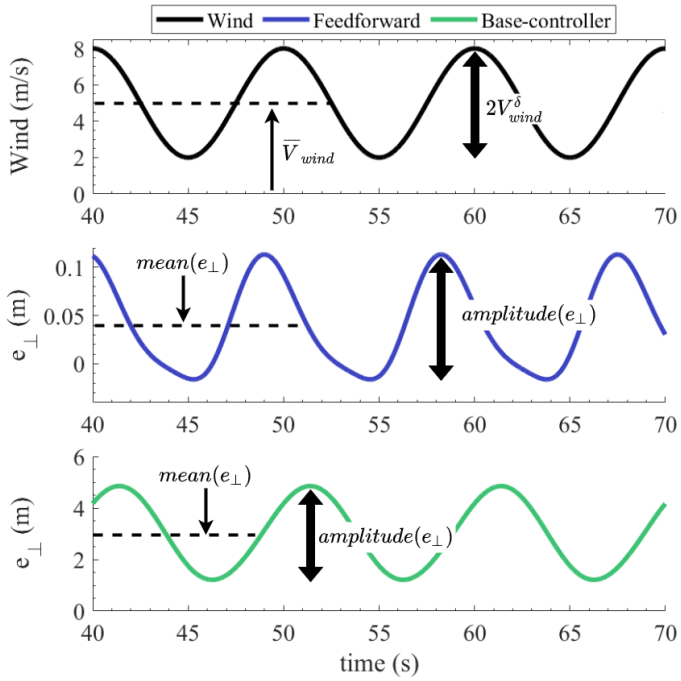

Figure 4.14: Sample data segment from the oscillating wind test where $f = 0.1$ Hz, amplitude and mean of base-controller and feedforward $e_{\perp}$ signals are shown.

A sample of the measurement of $e_{\perp}$ and $e_{\parallel}$ can be found in Figure 4.14, which shows the two error signals over time at a frequency $f = 0.1$ Hz. The configuration for the magnitude variation tests shown here is summarized in Table 4.3.

Table 4.3: Magnitude variation test configuration

Property	$f(Hz)$	V^{δ}_{wind}(m/s)	$\bar{V}_{wind}$(m/s)
Value	$0.05 - 1$	3	5

The results from the test series are assembled to form a Bode-like chart where $e_\perp$ and $e_\parallel$ are shown with respect to f. Results relating to $e_\perp$ are shown in Figure 4.15 and results for $e_\parallel$ can be found in Figure 4.16

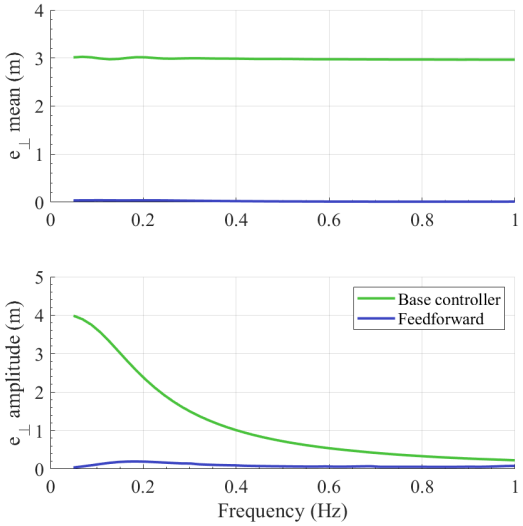

Figure 4.15: $e_\perp$ as a function of wind frequency for the magnitude variation simulation configuration in Table 4.3

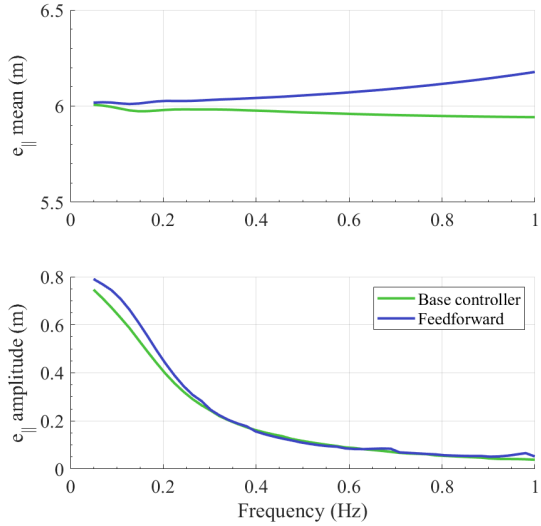

Figure 4.16: $e_\parallel$ as a function of wind frequency for the magnitude variation simulation configuration in Table 4.3

The results of this testing shows that the feedforward controller substantially attenuates cross-track wind disturbances when compared to the base-controller. This is particularly true at low frequencies where the difference is greatest, at higher frequencies both $e_\perp$ amplitudes shrink, likely due to the platform's inertia limiting its response to high frequencies. The results for on-track error, $e_\parallel$, shows that the frequency response of the base-controller is negligibly impacted by the feedforward controller in a direction undisturbed by the wind, which is desirable. This essentially shows that the feedforward controller is effective at reducing changing wind disturbances while minimizing any impacts on the tracking ability of the base-controller, satisfying one of our core design objectives. T_{ff_1} is mainly responsible for the similarity seen between the base-controller's response and the feedforward in Figure 4.16.

4.3.3.2 Direction variation

Direction variation tests follow a similar procedure to the magnitude variation tests, but instead, hold magnitude constant and rotate the wind direction about $\hat{i}_3$ at a certain fre-

quency. These tests are not performed to examine the effect that a spiralling wind has on the vehicle, but instead serve as a way of examining how the changing wind direction (relative to the aircraft) seen in circular flight impacts both the base-controller and the feedforward controller. This test is performed at a variety of frequencies to model the platform tracking small circles, where the wind vector changes quickly relative to the body frame, and large circles with a slowly changing wind vector relative to the body frame. To perform this test, the wind velocity is calculated as a function of f and time, t, as follows

$$\underline{V}_{wind} = \bar{V}_{wind} \begin{bmatrix} \cos(2\pi f t) \\ \sin(2\pi f t) \\ 0 \end{bmatrix} \tag{4.25}$$

The configuration parameters of the direction variation test case, including the maximum and minimum values for f, are summarized in Table 4.4. For reference to the case of circular flight, while traveling at a $10\ \mathrm{m/s}$ groundspeed, the maximum frequency $f = 0.3183\ \mathrm{Hz}$ would be encountered while tracking a $5\ \mathrm{m}$ radius circle, and the minimum frequency $f = 0.0159\ \mathrm{Hz}$ would be encountered while tracking a circle of $100\ \mathrm{m}$ radius. The remainder of the process for each test, including measurement of the amplitudes of $e_\perp$ and $e_\parallel$ is carried out the same as the magnitude test case. Peak-to-peak error amplitudes at each recorded virtual radius is shown for $e_\perp$ in 4.17 and for $e_\parallel$ in 4.18.

Table 4.4: Direction variation test configuration

Property	$f(Hz)$	$\bar{V}_{wind}(\ \mathrm{m/s})$
Value	$0.0159 - 0.3183$	5

Results for this test case show much improved performance compared to the base-controller, once again particularly in $e_\perp$, but the improvement is less dramatic than was seen in the magnitude variation case. We do see substantial improvements in $e_\parallel$ in this test case, as the wind vector is rotating and does occasionally align with the platform's longitudinal axis. These results are promising and do suggest that the feedforward controller is able to reject a wind vector disturbing the platform at a varying angle reasonably well. While the oscillations in the $e_\perp$ and $e_\parallel$ mean values for the base-controller case are

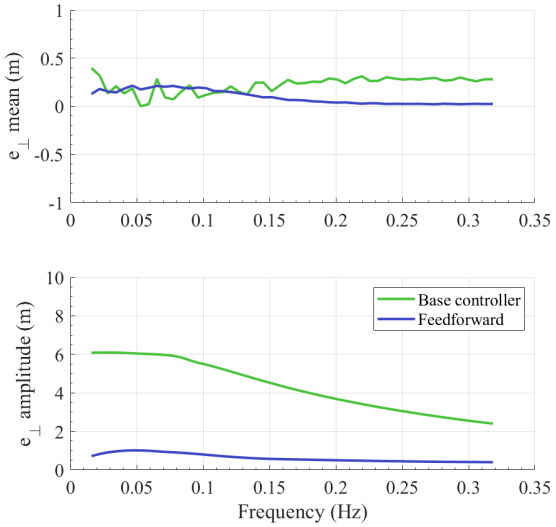

Figure 4.17: $e_{\perp}$ as a function of wind frequency for the direction variation simulation configuration in Table 4.4

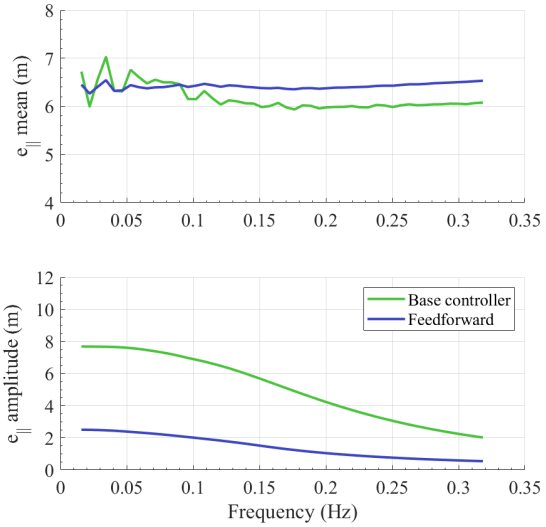

Figure 4.18: $e_{\parallel}$ as a function of wind frequency for the direction variation simulation configuration in Table 4.4

slightly unusual, since its magnitude is small we assume that this is a complex interaction between the base-controller and the rapidly changing wind vector.

4.3.4 Wind rejection in 3-dimensions

We now demonstrate the ability of the feedforward controller to reject an arbitrary wind vector, which may contain up or down drafts. No changes are required to the feedforward controller, we simply do not restrict $\underline{V}_{\infty}$, $\underline{V}_{i}$, and $\underline{V}_{wind}$ to the North-East plane. This is possible in simulation, as the true wind is known by the feedforward controller, but would require an advanced wind estimator capable of estimating vertical winds to demonstrate experimentally.

It is worth noting that strong vertical winds disturbances are not expected for flights of agile fixed-wing platforms. These platforms normally fly very low to the ground, where the vertical component of wind turbulence is significantly lower than its horizontal counterpart [64]. As such, this test case is shown less for its practical applications, and more as a demonstration of the capabilities of the feedforward controller.

Here, we show a simulation where the platform is tracking the same reference described in Section 4.3.2, but instead of a constant and uniform wind due East, the platform is disturbed by the following wind vector, resolved in the NED inertial frame

$$\underline{V}_{wind} = \begin{bmatrix} 0 \\ 5 \\ 5 \end{bmatrix} \text{ m/s}$$

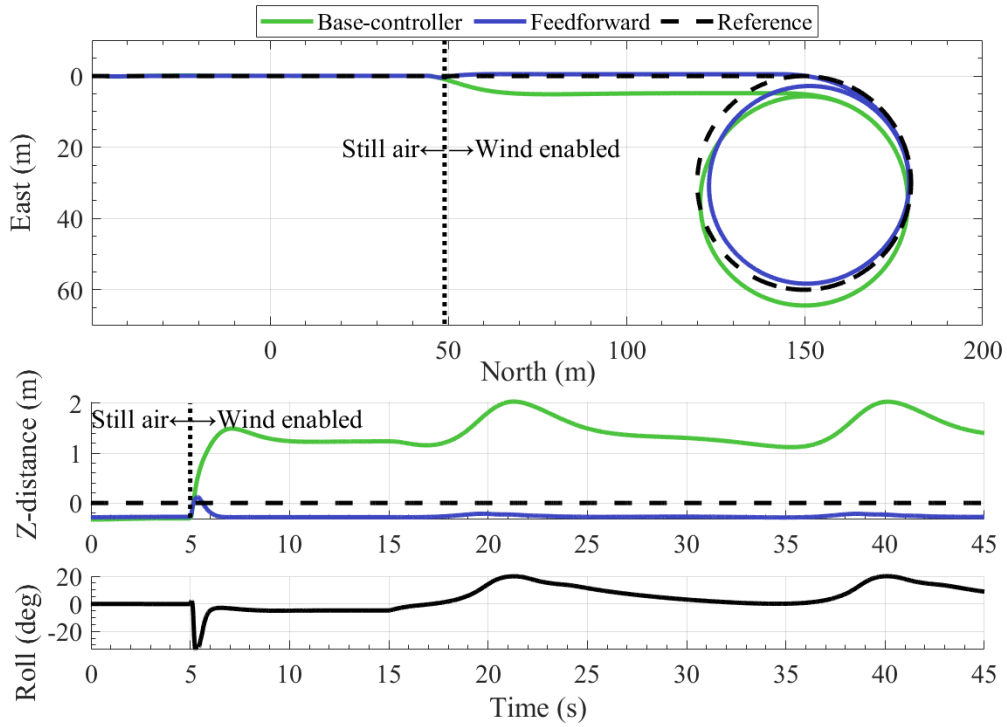

Figure 4.19: Simulation tracking a circular trajectory under a wind with an eastern component and a downdraft component. The top plot shows trajectories in the North-East plane, and the middle plot shows the vertical errors. The bottom plot shows the recorded roll angle. The platform begins on the left side of the figure and travels right, the circular reference rotates clockwise.

The results of this simulation can be seen in Figure 4.19 which displays the North-East trajectory in the top sub-figure, altitude in the middle sub-figure, and roll in the bottom sub-figure. The results show good performance rejecting the vertical wind component, as, after wind is applied, the platform travels at essentially the same altitude it did when wind was not present. The altitude error even with feedforward doesn't converge to zero, however. This is expected as aerodynamics are neglected in the base-controller, Eq. (4.6), causing the lift from the platform's wings to be a disturbance upward (a negative disturbance in the plot as the inertial frame is aligned to NED). Horizontal wind rejection also appears very good in the straight section before the circle. The circular tracking while

rejecting vertical wind is notably worse however when compared to the tests conducted in Figure 4.13. This appears to be due to the attitude feedforward imparting unintentional roll on the platform. When the commanded roll is zero, the base-controller ensures that $\boldsymbol{C}_{ri}^{fb}$ has no roll, it appears that when the wind vector has a vertical component, $\boldsymbol{R}_{ff}$ can induce some roll in the final attitude reference $\boldsymbol{C}_{ri}$. This issue, and potential fixes, require further study. One solution to removing roll from the final attitude reference would be to use the geometric vector projection method proposed in [57], however a solution on $\boldsymbol{R}_{ff}$ may be preferred. It should be noted that this issue does not arise in the two-dimensional tests, when the elements of the wind triangle are restricted to the horizontal plane.

Three-dimensional wind tests demonstrate reasonable wind rejection performance, but further work is needed to fix some of the remaining issues with rejection of an arbitrary wind.

4.4 Experiments

With good performance observed in simulation, we now proceed to evaluate the performance of the wind rejection feedforward in experimental flight tests. Since these flight tests are outdoors, and a 'true' wind value is not known, we must utilize a wind estimator to provide a wind estimate to the feedforward controller. We first use the stock wind estimator in PX4 for this task with an airspeed sensor far outside of the slipstream. This is done before advancing to use of the synthetic airspeed sensor to isolate the results of the feedforward controller from the synthetic airspeed sensor. Since this airspeed sensor placement effects the agile performance of the platform, we then utilize the synthetic airspeed sensor developed in Chapter 3 to enable wind estimation from a sensor mounted close to the aircraft's body. We demonstrate wind rejection using this wind estimation set-up.

Both experiments, using direct freestream measurement and the synthetic airspeed sensor, use a West Michigan Park Flyers McFoamy platform and the SDP33 sensor for airspeed measurement. They also both use a PixHawk 4 flight computer running PX4, with the native controller replaced by the feedback/feedforward controller described earlier in this chapter. The launch procedure is the same for both experiments. First the platform is launched, and flown semi-autonomously in a circle for a brief time to build up the wind estimate, after which full control of the platform is transferred to the feedback/feedforward controller.

In some experimental tests, a coordinate system, $x - y - z$, is defined relative to the reference path for easier visualization of performance. In this case, the *x-direction* is defined instantly when the feedback/feedforward controller is engaged along the aircraft's heading at that time. The *z-direction* aligns with $\hat{i}_3$ and the *y-direction* completes the right-handed frame.

4.4.1 Direct freestream measurement

The test presented here is designed only to assess the performance of the wind rejection feedforward controller, as such an airspeed sensor placed outside of the slipstream is used for wind estimation. This essentially uses all stock processes in PX4 for wind estimation. While sensor placement in this location will impact agile performance, the tests shown here do not require any aggressive maneuvering. The airspeed sensor location can be seen in Figure 4.20. Feedforward configuration $\mathcal{D}$, with $k_\gamma = 1$, was used for this flight test.

Figure 4.20: Sample of wind rejection setup showing airspeed sensor location

The reference path that the platform is commanded to track is very similar to the path used in Section 4.3.2, beginning with straight and level flight before the reference transitions into a circle. The primary difference here is the diameter of the circle, in this case it is $25\ \mathrm{m}$ instead of the $30\ \mathrm{m}$ used in simulation. The platform begins following the reference as soon as it is switched into its fully autonomous mode, where the feedback/feedforward controller is fully responsible for the platform's actuators. The feedforward controller is

enabled until the platform completes approximately 1.5 laps of the circular reference, after which it is deactivated, leaving only the base-controller to track the reference.

The wind estimate during this experiment can be seen in Figure 4.21, and the platform's measured trajectory relative to the reference, both when the feedforward is active and when the base-controller alone is used, can be seen in Figures 4.22 and 4.23

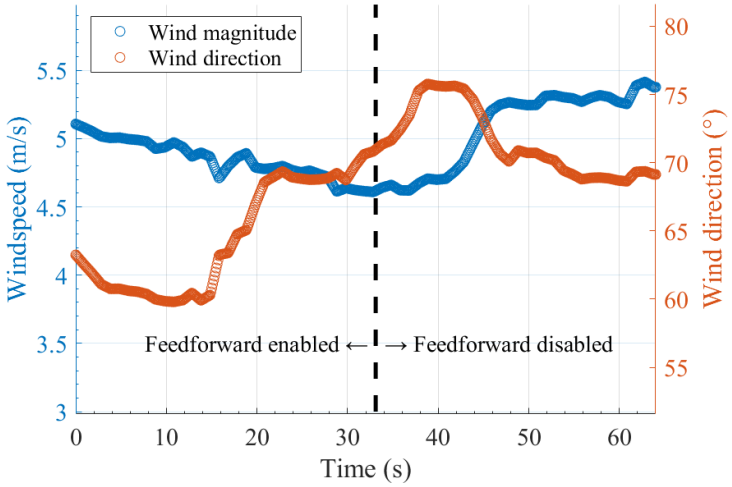

Figure 4.21: The wind estimate for experimental flight test. A dashed line separates wind estimates logged before the feedforward elements are disabled. The wind direction is defined similar to a compass rose, but with the y-axis replacing North and the x-axis replacing East.

The results of this flight test show good performance for the feedforward controller, especially compared to the base-controller alone. While the average estimated wind magnitude during the section of the test where the feedforward was disabled was slightly higher than the section with feedforward enabled, the difference is very unlikely to have caused the performance increase observed in the results, especially since the feedforward results appear to show it rejects most offset due to the wind.

4.4.2 Synthetic airspeed sensor

The experimental flight test using the synthetic airspeed sensor is largely similar to the previous flight test with direct freestream measurement, but with a different reference trajectory. For this test, a 'racetrack' type flight path is used here instead of the circular flight path shown previously, the straight segments of the racetrack have a vertical component and are set at an angle of 22° to demonstrate the vertical flight aspects of the

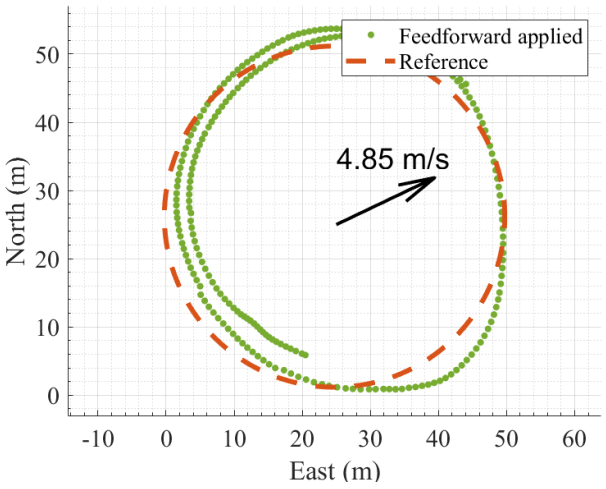

Figure 4.22: Experimental flight test tracking a 25 m radius circle at 10 m/s with feedforward enabled. Average wind direction and magnitude are shown with the black arrow and text respectively.

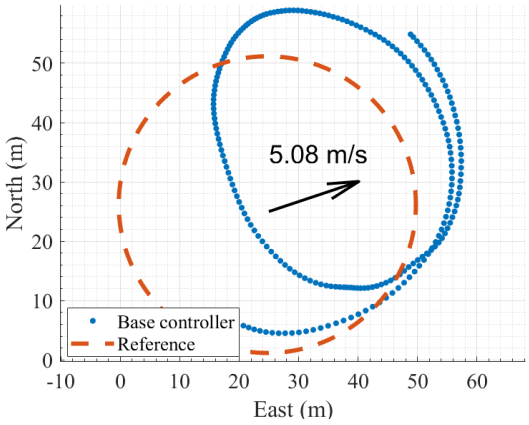

Figure 4.23: Experimental flight test tracking a 25 m radius circle at 10 m/s using only the base-controller. Average wind direction and magnitude are shown with the black arrow and text respectively.

platform. This reference trajectory can be seen in Figure 4.24. The same airspeed sensor location and synthetic airspeed sensor settings as used in Section 3.5.2.1 are used here, the sensor location is once again shown in Figure 4.25. Feedforward configuration $\mathcal{C}$, with $k_\gamma = 1/0.45$, is used for this flight test.

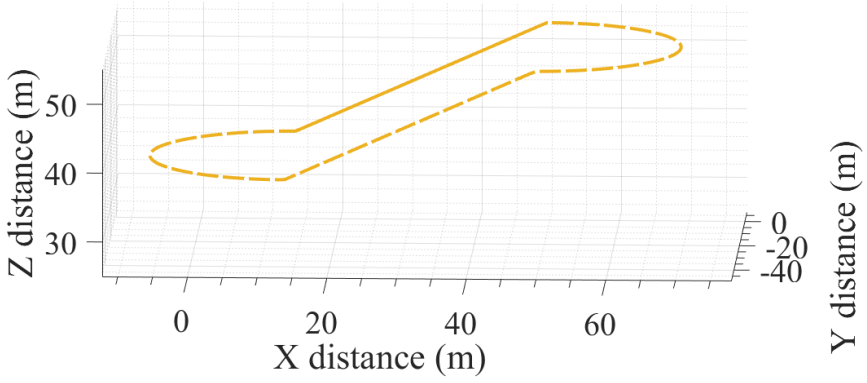

Figure 4.24: Racetrack reference trajectory.

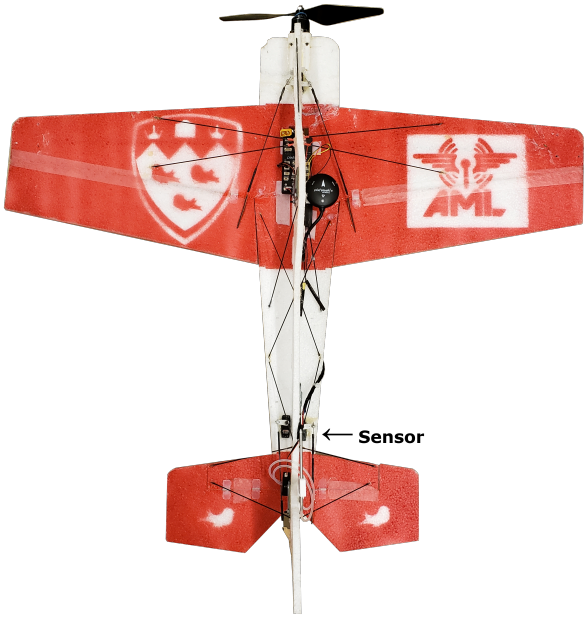

Figure 4.25: Wind rejection setup showing airspeed senor location for use with synthetic airspeed sensor

The autonomous flight mode is once again initialized with the feedforward active, with it being disabled after one lap of the racetrack. The reference trajectory using only the base-controller, and trajectory using the feedforward can be seen in Figure 4.26 and the wind estimate can be seen in Figure 4.27.

An interesting feature can be noted near the bottom right side of the flight trajectories shown in Figure 4.26. This feature appears to occur due to loss of control authority while travelling downwind. On the right downwind leg, the platform requires very little throttle to match the velocity of the reference point, as propeller RPM drops substantially during this leg so does airflow over the control surfaces, reducing control authority. Control authority only returns once the target begins moving along the lower straight path and the tracking error increases substantially, rapidly increasing the throttle command. As it occurs both with the feedforward active and inactive, it is not of substantial concern for evaluating the feedforward controller's performance, however it is clearly a complex disturbance on the base controller due to wind that the feedforward controller is not fully capable of compensating for. This phenomenon will be revisited in the future work section of Chapter 5 as it is an interesting effect to correct.

The downwind disturbance aside, we do see good performance of the wind rejection feedforward controller here, even when the wind estimate is computed using measure-

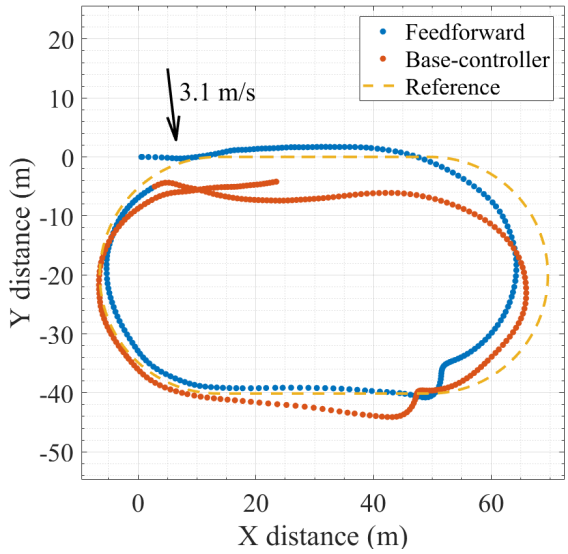

Figure 4.26: Flight test results tracking a racetrack trajectory with feedforward wind rejection and wind estimation using a synthetic airspeed sensor. The average wind direction and magnitude estimated during the flight test is shown with a black arrow and black text respectively.

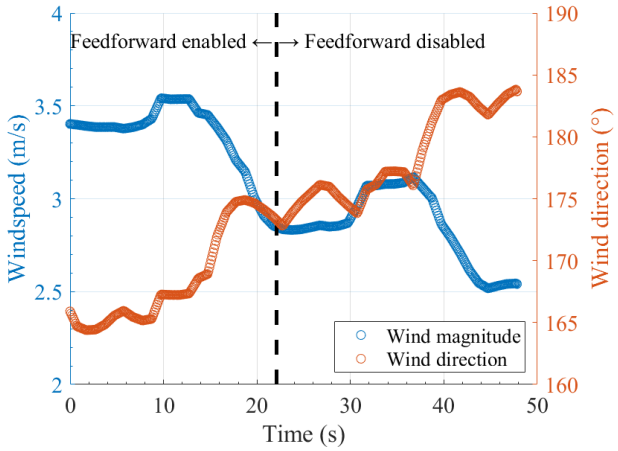

Figure 4.27: Wind estimate during racetrack flight test using the synthetic airspeed sensor. The wind direction is again defined similar to a compass rose, but with the y-axis replacing North and the x-axis replacing East.

ments from inside the propeller slipstream. It can be seen in Figure 4.26 that the feedforward controller prevents the offset in trajectory along the wind direction that is observed when the base-controller is operating alone. Future flight tests will be performed to validate the feedforward controller using synthetic airspeed sensing in configuration $\mathcal{D}$, though the performance differences from this are expected to be relatively minor.

Chapter 5

Conclusion

We conclude this thesis by first providing an overview of its key conclusions, before discussing potential opportunities for future work.

5.1 Summary of Conclusions

This thesis presents a method for wind rejection on agile fixed-wing platforms which only requires the addition of a standard airspeed sensor and is minimally disruptive to the platform's agility.

The approach used for wind disturbance rejection in this thesis relies on an explicit wind estimate to reject wind. As such, the first two objectives of this thesis investigate how wind can be estimated on a platform which is aerodynamically dominated by propeller flow. Wind estimation is performed with minimal effects on maneuverability by directly compensating for airflow disturbances due to the propeller by through a model-based approach. This wind estimate is then used to reject the wind disturbance.

The first objective of this thesis was to validate the propeller slipstream model to ensure its accuracy before using it to estimate wind from an airspeed sensor inside the slipstream. A series of validation test flights were conducted to assess accuracy of the slipstream model at a variety of locations within the slipstream. The results of this test campaign showed very good accuracy of the slipstream model at most locations tested, both under constant and varying propeller speeds. Particularly good accuracy was observed near the tail of the aircraft.

After the model was validated, the second objective of this thesis, estimation of wind from an airspeed sensor immersed in slipstream, was investigated. To achieve this ob-

jective, a model-based synthetic airspeed sensor was designed to remove the effect of propeller flow from an airspeed measurement, to obtain only freestream velocity. To do this, we first presented an observability analysis to select a location for the airspeed sensor that would balance freestream observability and the maneuverability of the platform. We then designed a synthetic airspeed sensor based on the Extended Kalman Filter. The synthetic airspeed sensor was cascaded with a conventional wind estimator, to provide simultaneous estimates of freestream velocity and wind speed. Freestream estimates were validated using in-flight measurements, and we showed that the wind estimate converges when run on-line in experimental testing. In addition to demonstrating the estimation of freestream velocity from an airspeed sensor located inside the slipstream, we also showed the potential for improved attitude control by using the unfiltered slipstream measurement to better allocate the platform's control surfaces during agile flight.

Once a wind estimate was available onboard the agile fixed-wing platform, the final objective of this thesis, was pursued, which focused on the rejection of wind disturbances using a wind estimate. A modular, feedforward, wind disturbance rejection controller was developed, which was designed to work in tandem with an existing agile fixed-wing trajectory tracking controller to improve its performance in windy conditions. The wind rejection controller works on attitude and thrust references to reject wind disturbances impacting the platform in an arbitrary direction. The wind rejection controller was tested extensively in simulation, showing very good performance rejecting both constant and oscillating wind while tracking straight and circular references. Experimental flight tests were also conducted to validate the performance of this controller, both using a conventional wind estimator with wingtip freestream measurement, and also using the synthetic airspeed sensor with an airspeed sensor placed inside the slipstream. Wind rejection performance was assessed to be very good for both tests.

In the process of meeting the objectives of this thesis, several interesting opportunities for future research were identified.

5.2 Suggestions for future work

Potential topics for future work are listed below, these involve both improvements to the experiments conducted in this work, and avenues for new projects.

- To better validate the slipstream model, it may be of interest to perform additional in-flight measurements inside the slipstream. These measurements should focus on

investigating the readings observed at location B1, and may involve changing the direction of the propeller to see if swirl is the cause of the low recordings at this location.

- Further development could be carried out for the synthetic airspeed sensor, particularly the EKF-SAS, to improve performance when the airspeed sensor experiences oblique flow. An interesting approach may be to couple oblique flow angle estimation into the EKF-SAS to account for the loss of airspeed sensor accuracy at high oblique flow angles.

- The use of direct slipstream measurement for control allocation on agile fixed-wing platforms was only partially explored in Section 3.6. While a substantial performance increase was observed when using this method when compared to traditional momentum theory control allocation methods, further work should be undertaken to fully characterize the performance of direct measurement control allocation in a wide variety of agile maneuvers.

- To more completely validate the wind rejection controller, feedforward configuration $\mathcal{D}$ should be tested experimentally for the EKF-SAS based wind estimator case. Additionally, it may be of interest to evaluate the performance of the wind rejection controller with a greater variety of reference trajectories.

- All tests for the wind rejection controller in Chapter 4 command a zero roll angle in the base-controller for consistency. It may be of interest to examine how a separate wind rejection roll controller could be used to provide additional performance improvements in the presence of wind disturbances.

- The attitude feedforward controller was seen to induce undesired roll when rejecting a wind vector with a vertical component. Developing a method of eliminating this roll could drastically increase performance under this wind condition, and as such may be worth additional investigation.

- The wind rejection controller presented in this thesis is mainly intended for level flight, when sideslip and angle of attack are small. This could be extended to more agile maneuvers using sideslip and angle of attack estimates, to increase the useful operating range of the wind rejection controller.

- To avoid the downwind performance issue discussed at the end of Chapter 4, it may be interesting to use a predictive approach for the wind rejection controller. As an example of why this may be useful, when tracking a downwind trajectory, a conventional feedback position controller can easily reduce propeller speed low enough that propeller slipstream is insufficient to maintain attitude control. While the controller may see this reduction in thrust as beneficial for minimizing tracking error at a given moment, over the long term, the loss in attitude control often results in substantial tracking errors. A predictive approach may be able to avoid the large jumps in tracking error described in Chapter 4, by accepting increases in downwind tracking error in the short term, to maintain attitude control and prevent more severe increases in tracking error in the long term. This would require a reference that is known ahead of time, which may not always be available if the platform is being sent commands remotely or piloted semi-autonomously.

References

[1] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, "Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges," *IEEE Access*, vol. 7, pp. 48572–48634, 2019.

[2] M. Stewart and S. Martin, "Unmanned aerial vehicles: fundamentals, components, mechanics, and regulations," *Unmanned Aerial Vehicles*, 2021.

[3] J. Blackeye, "White drone mid-flight." `https://commons.wikimedia.org/wiki/File:White_drone_mid-flight_(Unsplash).jpg`, 2016. Accessed: 2022-3-7.

[4] JoeB52, "Inview UAV flyover." `https://commons.wikimedia.org/wiki/File:Inview_UAV_flyover.jpg`, 2010. Accessed: 2022-3-7.

[5] E. Bulka and M. Nahon, "Automatic control for aerobatic maneuvering of agile fixed-wing UAVs," *Journal of Intelligent & Robotic Systems*, vol. 93, pp. 85–100, Feb. 2018.

[6] W. Khan, *Dynamics modeling of agile fixed-wing unmanned aerial vehicles*. PhD Thesis, McGill University (Canada), 2016.

[7] E. Bulka, *Control and Obstacle Avoidance for Agile Fixed-Wing Aircraft*. PhD Thesis, McGill University (Canada), 2021.

[8] W. L. Chan, C. S. Lee, and F. B. Hsiao, "Real-time approaches to the estimation of local wind velocity for a fixed-wing unmanned air vehicle," *Measurement Science and Technology*, vol. 22, p. 105203, Aug. 2011.

[9] J. Bannwarth, Z. Chen, K. Stol, and B. MacDonald, "Disturbance accomodation control for wind rejection of a quadcopter," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 695–701, IEEE, 2016.

[10] J. Verberne and H. Moncayo, "Robust control architecture for wind rejection in quadrotors," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, June 2019.

[11] Y. Demitrit, S. Verling, T. Stastny, A. Melzer, and R. Siegwart, "Model-based wind estimation for a hovering VTOL tailsitter UAV," in *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3945–3952, Institute of Electrical and Electronics Engineers Inc., 7 2017.

[12] X. Lyu, J. Zhou, H. Gu, Z. Li, S. Shen, and F. Zhang, "Disturbance observer based hovering control of quadrotor tail-sitter VTOL UAVs using H_∞ synthesis," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2910–2917, 2018.

[13] P. Sujit, S. Saripalli, and J. B. Sousa, "Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicles," *IEEE Control Systems*, vol. 34, pp. 42–59, Feb. 2014.

[14] S. Sun, K. Dong, C. Guo, and D. Tan, "A wind estimation based on unscented Kalman filter for standoff target tracking using a fixed-wing UAV," *International Journal of Aeronautical and Space Sciences*, vol. 22, pp. 366–375, July 2020.

[15] M. R. Cohen and J. R. Forbes, "Navigation and control of unconventional VTOL UAVs in forward-flight with explicit wind velocity estimation," *IEEE Robotics and Automation Letters*, vol. 5, pp. 1151–1158, 4 2020.

[16] C. Liu, O. McAree, and W.-H. Chen, "Path-following control for small fixed-wing unmanned aerial vehicles under wind disturbances," *International Journal of Robust and Nonlinear Control*, pp. 1682–1698, Dec. 2012.

[17] D. Shin, Y. Song, J. Oh, and H. Oh, "Nonlinear disturbance observer-based standoff target tracking for small fixed-wing UAVs," *International Journal of Aeronautical and Space Sciences*, vol. 22, pp. 108–119, Apr. 2020.

[18] O. Pfeifle and W. Fichter, "Cascaded incremental nonlinear dynamic inversion for three-dimensional spline-tracking with wind compensation," *Journal of Guidance, Control, and Dynamics*, vol. 44, pp. 1559–1571, Aug. 2021.

[19] C. Liu and W.-H. Chen, "Disturbance rejection flight control for small fixed-wing unmanned aerial vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 39, pp. 2810–2819, Dec. 2016.

[20] H. Nobahari and A. Sharifi, "Wind compensation in trajectory tracking of a fixed wing UAV using a nonlinear model predictive controller based on the particle swarm optimization," in *2019 7th International Conference on Robotics and Mechatronics (ICRoM)*, pp. 138–143, IEEE, 2019.

[21] H. Lu, Y. Zhen, and M. Hao, "Nonlinear autopilot design for fixed-wing UAV using disturbance observer based backstepping," in *2020 Chinese Automation Congress (CAC)*, IEEE, Nov. 2020.

[22] A. Brezoescu, T. Espinoza, P. Castillo, and R. Lozano, "Adaptive trajectory following for a fixed-wing UAV in presence of crosswind," *Journal of Intelligent & Robotic Systems*, vol. 69, pp. 257–271, Aug. 2012.

[23] A. Wenz, T. A. Johansen, and A. Cristofaro, "Combining model-free and model-based angle of attack estimation for small fixed-wing UAVs using a standard sensor suite," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 624–632, 2016.

[24] J. Sun, B. Li, C. Y. Wen, and C. K. Chen, "Model-Aided Wind Estimation Method for a Tail-Sitter Aircraft," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, pp. 1262–1278, 4 2020.

[25] PX4 Autopilot, "Using the ECL EKF." `https://docs.px4.io/master/en/advanced_config/tuning_the_ecl_ekf.html#airspeed`, 2021. Accessed: Mar 2021.

[26] A. Rautenberg, M. Graf, N. Wildmann, A. Platis, and J. Bange, "Reviewing wind measurement approaches for fixed-wing unmanned aircraft," *Atmosphere*, vol. 9, p. 422, Oct. 2018.

[27] T. A. Johansen, A. Cristofaro, K. Sorensen, J. M. Hansen, and T. I. Fossen, "On estimation of wind velocity, angle-of-attack and sideslip angle of small UAVs using standard sensors," in *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, June 2015.

[28] A. Brezoescu, P. Castillo, and R. Lozano, "Wind estimation for accurate airplane path following applications," *Journal of Intelligent & Robotic Systems*, vol. 73, pp. 823–831, Oct. 2013.

[29] S. Shkarayev, J.-M. Moschetta, and B. Bataille, "Aerodynamic design of micro air vehicles for vertical flight," *Journal of Aircraft*, vol. 45, pp. 1715–1724, Sept. 2008.

[30] D. Hunsaker and D. Snyder, "A lifting-line approach to estimating propeller/wing interactions," in *24th AIAA Applied Aerodynamics Conference*, p. 3466, 2006.

[31] E. W. M. Roosenboom, A. Heider, and A. Schröder, "Investigation of the propeller slipstream with particle image velocimetry," *Journal of Aircraft*, vol. 46, pp. 442–449, Mar. 2009.

[32] R. de Vries, N. van Arnhem, T. Sinnige, R. Vos, and L. L. Veldhuis, "Aerodynamic interaction between propellers of a distributed-propulsion system in forward flight," *Aerospace Science and Technology*, vol. 118, p. 107009, Nov. 2021.

[33] X. Q. Gong, M. S. Ma, J. Zhang, and J. Tang, "Investigation on propeller slipstream by using an unstructured rans solver based on overlapping grids," *Journal of Mechanics*, vol. 34, pp. 89–101, July 2017.

[34] E. W. M. Roosenboom, A. Stürmer, and A. Schröder, "Advanced experimental and numerical validation and analysis of propeller slipstream flows," *Journal of Aircraft*, vol. 47, pp. 284–291, Jan. 2010.

[35] R. W. Deters, G. K. Ananda, and M. S. Selig, "Slipstream measurements of small-scale propellers at low reynolds numbers," in *33rd AIAA Applied Aerodynamics Conference*, American Institute of Aeronautics and Astronautics, June 2015.

[36] Y. Leng, M. Bronz, T. Jardin, and J.-M. Moschetta, "Comparisons of different propeller wake models for a propeller-wing combination," in *Proceedings of the 8th European Conference for Aeronautics and Space Sciences. Madrid, Spain*, 2019.

[37] M. Wei, Y.-M. Chiew, and N.-S. Cheng, "Recent advances in understanding propeller jet flow and its impact on scour," *Physics of Fluids*, vol. 32, p. 101303, Oct. 2020.

[38] W. Lam, G. Hamil, Y. Song, D. Robinson, and S. Raghunathan, "A review of the equations used to predict the velocity distribution within a ship's propeller jet," *Ocean Engineering*, vol. 38, pp. 1–10, Jan. 2011.

[39] W.-H. Lam and L. Chen, "Equations used to predict the velocity distribution within a wake from a horizontal-axis tidal-current turbine," *Ocean Engineering*, vol. 79, pp. 35–42, Mar. 2014.

[40] W.-H. Lam, L. Chen, and R. Hashim, "Analytical wake model of tidal current turbine," *Energy*, vol. 79, pp. 512–521, Jan. 2015.

[41] W. Khan and M. Nahon, "Development and validation of a propeller slipstream model for unmanned aerial vehicles," *Journal of Aircraft*, vol. 52, pp. 1985–1994, Nov. 2015.

[42] B. McCormick, *Aerodynamics, Aeronautics and Flight Mechanics*. Wiley, 1979.

[43] B. McCormick, *Aerodynamics of V/STOL Flight*. Dover Publications, 1999.

[44] M. Albertson, Y. Dai, R. A. Jensen, and H. Rouse, "Diffusion of submerged jets," *Transactions of the American Society of Civil Engineers*, vol. 115, no. 1, pp. 639–664, 1950.

[45] G. Hamill, D. Ryan, and C. Kee, "Three-dimension efflux velocity characteristics of marine propeller jets," *Proceedings of the ICE - Maritime Engineering*, vol. 168, pp. 62–75, June 2015.

[46] R. Deters and M. Selig, "Static testing of micro propellers," in *26th AIAA applied aerodynamics conference*, p. 6246, 2008.

[47] O. Gur and A. Rosen, "Propeller performance at low advance ratio," *Journal of Aircraft*, vol. 42, pp. 435–441, Mar. 2005.

[48] J. Brandt and M. Selig, "Propeller performance data at low reynolds numbers," in *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, p. 1255, 2011.

[49] W. Khan and M. Nahon, "A propeller model for general forward flight conditions," *International Journal of Intelligent Unmanned Systems*, vol. 3, pp. 72–92, May 2015.

[50] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft*. Princeton University Press, Dec. 2012.

[51] M. A. Amini and M. Ayati, "Performance of low-cost air-data sensors for airspeed and angle of attack measurements in a flapping-wing robot," *Journal of Aerospace Engineering*, vol. 32, p. 04019018, May 2019.

[52] P. S. Maybeck, *Stochastic models, estimation, and control*. Academic press, 1982.

[53] G. Bishop, G. Welch, *et al.*, "An introduction to the Kalman filter," *Proc of SIGGRAPH, Course*, vol. 8, no. 27599-23175, p. 41, 2001.

[54] T. D. Barfoot, *State estimation for robotics*. Cambridge University Press, 2017.

[55] M. Shalaby, C. C. Cossette, J. Le Ny, and J. R. Forbes, "Cascaded filtering using the sigma point transformation," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4758–4765, 2021.

[56] E. Bulka and M. Nahon, "A unified control strategy for autonomous aerial vehicles," *Autonomous Robots*, vol. 45, pp. 859–883, Sept. 2021.

[57] J. C. H. Ramirez and M. Nahon, "Nonlinear vector-projection control for agile fixed-wing unmanned aerial vehicles," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2020.

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

[59] J. Moreno-Valenzuela, R. Pérez-Alcocer, M. Guerrero-Medina, and A. Dzul, "Nonlinear PID-type controller for quadrotor trajectory tracking," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 5, pp. 2436–2447, 2018.

[60] K. W. Spring, "Euler parameters and the use of quaternion algebra in the manipulation of finite rotations: A review," *Mechanism and Machine Theory*, vol. 21, pp. 365–373, Jan. 1986.

[61] W. Khan and M. Nahon, "A propeller model for general forward flight conditions," *International Journal of Intelligent Unmanned Systems*, 2015.

[62] W. Khan and M. Nahon, "Development and validation of a propeller slipstream model for unmanned aerial vehicles," *Journal of Aircraft*, vol. 52, no. 6, pp. 1985–1994, 2015.

[63] W. Khan and M. Nahon, "Real-time modeling of agile fixed-wing UAV aerodynamics," in *2015 international conference on unmanned aircraft systems (ICUAS)*, pp. 1188–1195, IEEE, 2015.

[64] B. Etkin, "Turbulent wind and its effect on flight," *Journal of Aircraft*, vol. 18, pp. 327–345, May 1981.

[65] F. White, *Fluid mechanics*. New York, N.Y: McGraw Hill, 2011.

[66] Sensirion, *Compensation of pressure drop in a hose*, 7 2016. Version 0.2.

[67] L. Meier, D. Honegger, and M. Pollefeys, "PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 6235–6240, IEEE, 2015.

[68] A. Watts, "On exponential smoothing of discrete time series (corresp.)," *IEEE Transactions on Information Theory*, vol. 16, no. 5, pp. 630–630, 1970.

Appendix A

Sensor validation and calibration

This Appendix presents a validation and, when necessary, calibration of the sensors used for the in-flight slipstream model validation shown in Chapter 2. First validation and calibration results for the two airspeed sensors, the SDP33 and MS4525DO, are shown, followed by validation results for the RPM sensor.

A.1 Airspeed sensors

A.1.1 Airspeed reference

In order to validate the measurements from the airspeed sensors, we require an accurate and trusted airspeed source to serve as a base-line reference. In this thesis, the trusted airspeed source we used is a calibration tool which can provide an accurate and constant airspeed by ducting airflow from a blower fan into an instrumented venturi tube. Each end of the venturi tube contains a pressure tap leading to a Furness Control FCO432 differential pressure sensor, which is used to calculate the flow velocity at the outlet of the venturi tube. A diagram of this apparatus can be seen in Figure A.1, and an image of the apparatus being used to calibrate an airspeed sensor is shown in A.2.

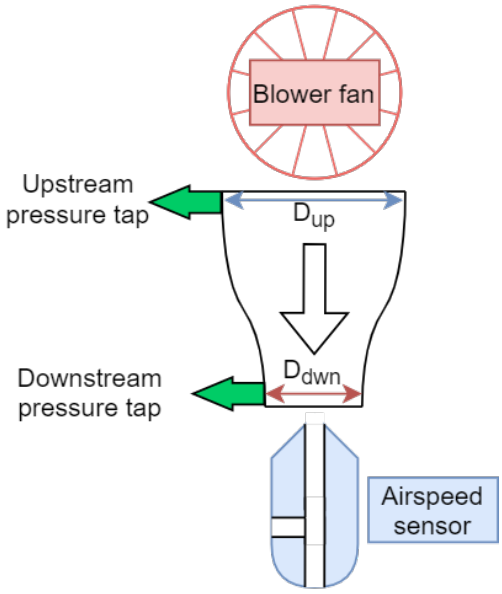

Figure A.1: Schematic of the airspeed source used for sensor calibration. D_{dwn} represents the tube diameter at the downstream pressure tap and D_{up} represents the tube diameter at the upstream pressure tap.

Flow velocity at the exit of the venturi tube can be calculated using the differential pressure between the upstream and downstream taps, Δp, and the air density, ρ, as follows [65]

$$V_{dwn} = \sqrt{\frac{2\Delta p}{\rho\left(1 - (\frac{D_{dwn}}{D_{up}})^4\right)}} \tag{A.1}$$

Eq. (A.1) relies on the known diameter at the downstream and upstream pressure taps, D_{dwn} and D_{up} respectively, which are measured using calipers to be $D_{dwn} = 10.53$ mm and $D_{up} = 27.06$ mm. When validating an airspeed sensor, the pitot probe is placed in the centre of the outlet of the venturi constriction, so that the air velocity entering the pitot is equal to V_{dwn}.

The speed of the outlet flow is controlled manually by changing the voltage of the blower fan's power supply, and the FCO432 differential pressure sensor provides a digital readout which is logged manually by the operator. Of course, since the operator cannot record all readings displayed by the FCO432 during a given time frame, only the maximum and minimum Δp readings from the FCO432 are recorded, denoted Δp_r^+ and Δp_r^- respectively. Using Eq. (A.1), Δp_r^+ and Δp_r^- can be converted to airspeed values,

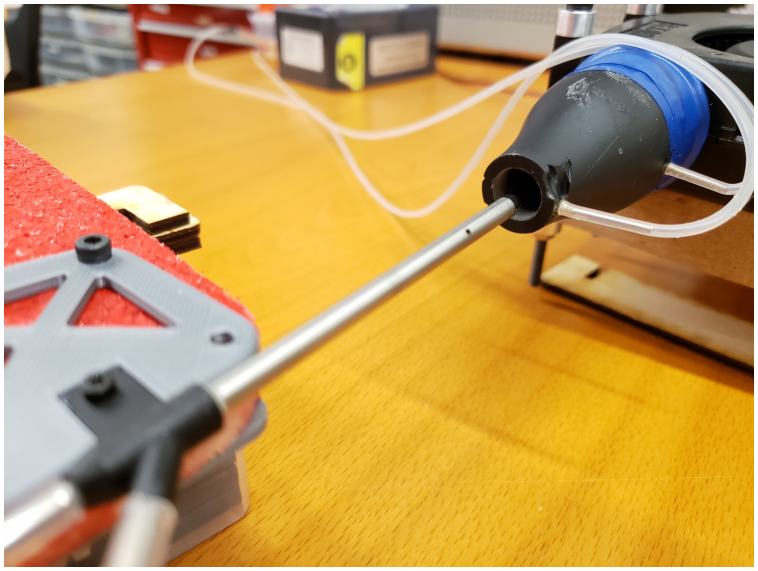

Figure A.2: Venturi constriction device being used to validate the MS4525 sensor.

which are denoted similarly as $V^+_{a_{ref}}$ and $V^-_{a_{ref}}$. For a given measurement period $V^+_{a_{ref}}$ and $V^-_{a_{ref}}$ essentially provide bounds on the reference airspeed, when validating an airspeed sensor, the measurements from the sensor should fall within these bounds if the sensor is accurate. It should be noted that the gap between $V^+_{a_{ref}}$ and $V^-_{a_{ref}}$ is quite narrow, generally no more than $0.1\ \mathrm{m/s}$ and often around $0.05\ \mathrm{m/s}$. As it is used for calibration of the SDP33, the reference airspeed in the middle of these bounds is denoted $\overline{V}_{a_{ref}}$ and is calculated as the average of $V^+_{a_{ref}}$ and $V^-_{a_{ref}}$. The trusted airspeed source is limited to a maximum output airspeed of approximately $14\ \mathrm{m/s}$, due to the maximum speed of the blower fan.

A.1.2 MS4525DO validation

The MS4525DO sensor used for freestream measurement is a digital, factory-calibrated, membrane-style differential pressure sensor. Since it is a membrane-type sensor, there is no route for flow to travel between the static and dynamic ports of the pitot probe, and, as we are not concerned about measuring high-frequency transients in the flow, we can reasonably assume that the mass flow in the tubing section, between the pitot probe and differential pressure sensor, is negligible for this sensor. Since the mass flow is negligible, we can also assume that tube pressure loss in the tubing assembly is likewise negligible, meaning we do not need to calibrate this airspeed sensor. We do still validate it against the trusted airspeed source at six reference airspeeds from $0\ \mathrm{m/s}$ to $12\ \mathrm{m/s}$ to

ensure it is functioning correctly. Reference airspeeds are approximately evenly spread out throughout the range of $0\ \mathrm{m/s}$ to $12\ \mathrm{m/s}$, and each airspeed reference is held for $15\ \mathrm{s}$. During this validation test, the readings from the FCO432 are logged manually by the operator, and the measurements from the MS4525DO are recorded electronically by the PixRacer flight computer, in the same configuration as is used for flight tests in Chapter 2. Results for this validation will be shown in the airspeed sensor results section of this appendix, alongside the results for the SDP33 sensor. Although only the results of one test are shown, two validation tests were conducted for the MS4525DO, both confirming similar levels of performance.

A.1.3 SDP33 calibration

The SDP33 microthermal differential pressure sensor, used to measure slipstream, is more sophisticated than the MS4525DO, with zero-drift and low noise being advertised by the manufacturer, but does require calibration. This sensor uses a different mechanism to measure differential pressure: instead of employing a membrane which blocks flow, it uses a small amount of flow through the sensor to measure differential pressure [66]. This poses an issue when connecting the sensor to tubing for use with a pitot probe, as the flow in the tubing will cause a small pressure drop between the pitot and the sensor. While the SDP33 may accurately measure differential pressure, the pressure drop in the tubing means that this differential pressure at the sensor will be slightly less than at the pitot probe. The manufacturer recommends compensating for this loss using [66]

$$\Delta p_{eff} = \frac{\Delta p_{sensor}}{1+\epsilon} \tag{A.2a}$$

$$\epsilon = \nu_{air} \frac{-64\, L\, m_c}{\pi D^4 \Delta p_{sensor}} \left(\sqrt{1 + \frac{8 \Delta p_{sensor}}{\Delta p_c}} - 1 \right) \tag{A.2b}$$

In Eq. (A.2), Δp_{sensor} is the differential pressure measured by the sensor, Δp_{eff} is the differential pressure that would be measured if there were no tube pressure loss, and ν_{air} is the kinematic viscosity of air. The objective of this correction is to determine the differential pressure that would be measured by the SDP33 if tube pressure loss were not present, Δp_{eff}, as some function of the actual differential pressure measured by the sensor, Δp_{sensor}. In Eq. (A.2b), the parameters: L, D, m_c and Δp_c are all constants relating to the flow characteristics of the tubing and sensor. The manufacturer lists nominal values for these parameters, however since we have access to a trusted airspeed source, we in-

stead perform a calibration so that the relation between ϵ and Δp_{sensor} is accurate for our particular tubing and pitot configuration. To do this, we note that ϵ can be written as an arbitrary function of Δp_{sensor}, denoted $\beta(\Delta p_{sensor})$, that is multiplied by ν_{air}. This allows us to rewrite ϵ as

$$\epsilon = \nu_{air}\beta(\Delta p_{sensor}) \tag{A.3}$$

We then fit the function $\beta(\Delta p_{sensor})$ using experimental data. Given experimental measurements of $\overline{V}_{a_{ref}}$, Δp_{eff} can be calculated using the following equation for measuring airspeed with a pitot-static probe [65]

$$\Delta p_{eff} = \frac{1}{2}\rho\overline{V}^2_{a_{ref}} \tag{A.4}$$

Substituting Eq. (A.4) into (A.2a) then yields

$$\frac{1}{2}\rho\overline{V}^2_{a_{ref}} = \frac{\Delta p_{sensor}}{1+\epsilon} \tag{A.5}$$

Using the relation shown in Eq. (A.3), $\beta(\Delta p_{sensor})$ can be solved for as a function of experimental measurements Δp_{sensor} and $\overline{V}_{a_{ref}}$ as follows

$$\beta(\Delta p_{sensor}) = \frac{1}{\nu_{air}}\left(\frac{\Delta p_{sensor}}{\frac{1}{2}\rho\overline{V}^2_{a_{ref}}} - 1\right) \tag{A.6}$$

To summarize, Eq. (A.6) provides the results of $\beta(\Delta p_{sensor})$ as a function of measurements from the SDP33 sensor and the trusted airspeed source. This enables us to fit $\beta(\Delta p_{sensor})$ using experimental results specific to our SDP33 sensor and related tubing. To increase the accuracy of the calibration, four runs were performed, gathering data for calibration in an airspeed range of $0\ \mathrm{m/s}$ to $12\ \mathrm{m/s}$.

The measurements showing $\beta(\Delta p_{sensor})$ as a function of Δp_{sensor} can be seen in Figure A.3 for all four calibration runs of the SDP33. We find that $\beta(\Delta p_{sensor})$ is reasonably well represented by a reciprocal function of Δp_{sensor}, estimated using MATLAB, and resulting in the following relationship

$$\beta(\Delta p_{sensor}) \approx \frac{1}{-5.018\times10^{-7}\Delta p_{sensor} - 4.372\times10^{-5}} \tag{A.7}$$

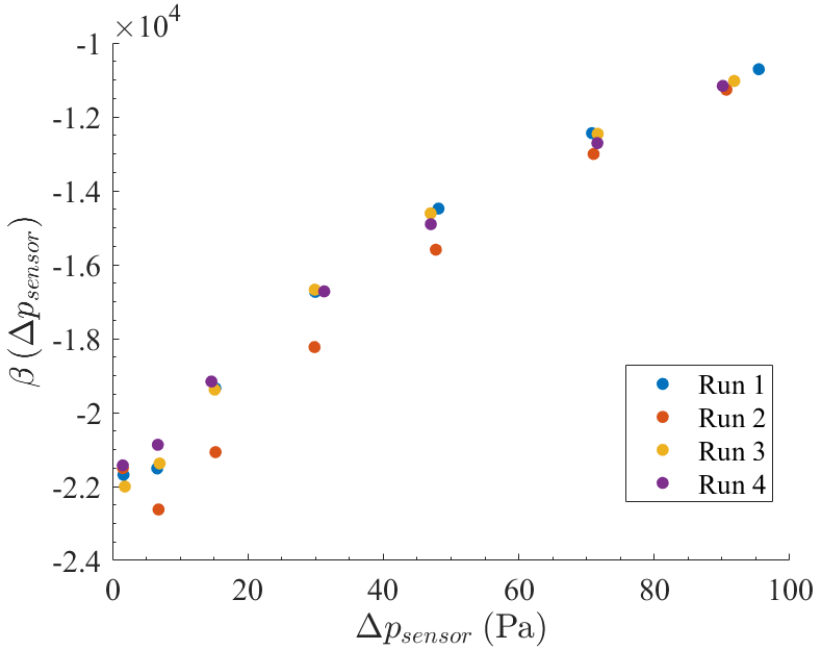

Figure A.3: $\beta(\Delta p_{sensor})$ measurements compared with Δp_{sensor}, used for calibration of the SDP33 airspeed sensor

Substituting Eq. (A.7) into (A.2) yields the following equation, where Δp_{eff} can then be calculated using only Δp_{sensor} and ν_{air} as

$$\Delta p_{eff} = \frac{\Delta p_{sensor}}{1 + \frac{\nu_{air}}{-5.018\times10^{-7}\Delta p_{sensor} - 4.372\times10^{-5}}} \tag{A.8}$$

Once Δp_{eff} is calculated, it can then be used with Eq. (A.4) to calculate airspeed from a pitot-based differential pressure measurement using the SDP33. This calibration allows us to accurately account for the tube pressure loss when using the SDP33 to measure airspeed, using only the differential pressure measurement from the SDP33.

After the SDP33 sensor is calibrated, we perform the same validation test on it as was done for the MS4525DO sensor. In this case, however, four validation tests were performed using the SDP33 sensor, with all tests once again yielding similar results. The results of the validation testing are shown in the following section.

A.1.4 Airspeed sensor results

Now that the SDP33 sensor is calibrated, validation using the trusted airspeed source can be conducted for both sensors. The reference airspeed, from the trusted airspeed source,

for the SDP33 validation test can be seen in Table A.1. Likewise, the reference airspeed data recorded during the MS4525DO test can be found in Table A.2.

Table A.1: True airspeed source data for the SDP33 validation test shown in Figure A.4

Δp_r^+ (Pa)	Δp_r^- (Pa)	$V_{a_{ref}}^+$ (m/s)	$V_{a_{ref}}^-$ (m/s)
0	0	0	0
2.5	2.3	2.1	2.0
9.8	9.6	4.1	4.1
21.4	21.0	6.1	6.0
39.7	39.3	8.3	8.3
61.0	60.3	10.3	10.2
86.1	85.3	12.2	12.2
112.3	111.3	14.0	13.9

Table A.2: True airspeed source data for the MS4525DO validation test shown in Figure A.5

Δp_r^+ (Pa)	Δp_r^- (Pa)	$V_{a_{ref}}^+$ (m/s)	$V_{a_{ref}}^-$ (m/s)
0	0	0	0
2.7	2.4	2.2	2.0
10.7	10.4	4.3	4.3
22.2	21.7	6.2	6.1
38.7	38.0	8.2	8.1
60.0	59.1	10.2	10.1
86.3	85.2	12.2	12.2
108.6	107.3	13.7	13.7

Figure A.4 shows the results of the validation run for the SDP33 sensor. The validation results for the MS4525DO sensor can be seen in Figure A.5. The plots in Figures A.4 and A.5 show the bounds $V_{a_{ref}}^+$ and $V_{a_{ref}}^-$, of the trusted airspeed reference at the six test velocities, and the average airspeed measurement from the SDP33 and the MS4525DO, respectively. For ideal airspeed sensor performance, the average airspeed measurement should be inside the bounds created by $V_{a_{ref}}^+$ and $V_{a_{ref}}^-$.

Since taking the mean value of our airspeed sensor readings hides sensor noise, we also show the standard deviation over the 15 s sample duration at each interval in Figure A.6. This is shown for both airspeed sensors.

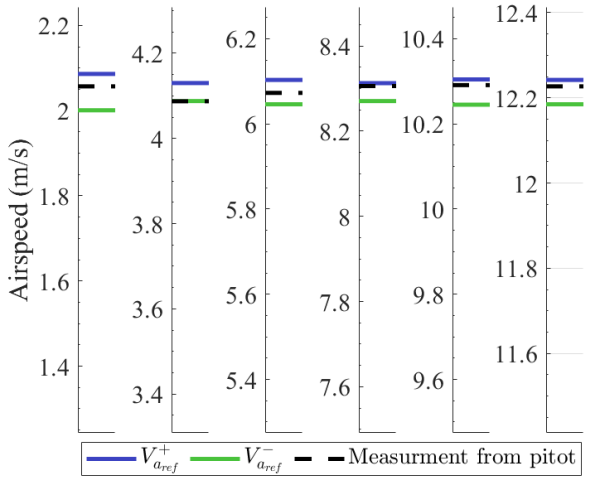

Figure A.4: Validation tests results for the calibrated SDP33 sensor. Green and blue lines indicate the bounds on the airspeed reference and the dashed black line indicates the average measurement from the airspeed sensor.

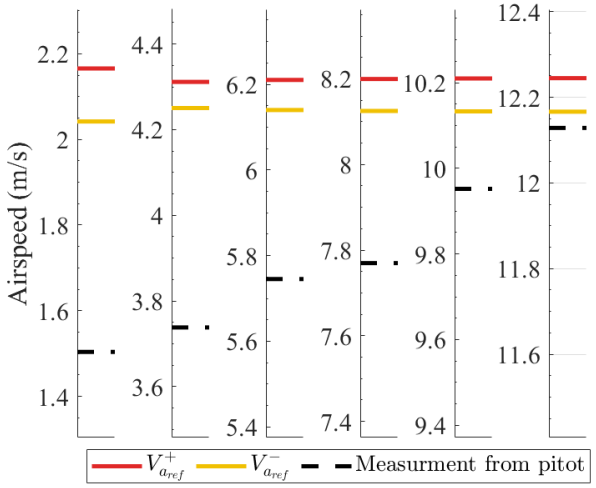

Figure A.5: Validation tests results for the MS4525DO sensor. Red and yellow lines indicate the bounds on the airspeed reference and the dashed black line indicates the average measurement from the airspeed sensor.

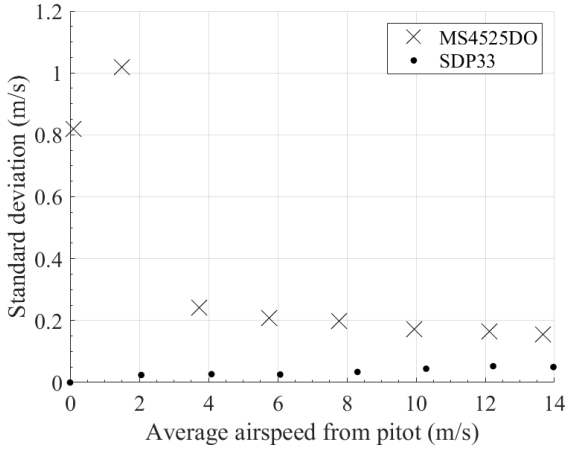

Figure A.6: Standard deviation of airspeed from pitot sensors as a function of average calculated airspeed they record. Results are shown for the MS4525DO based airspeed sensor and the SDP33 based airspeed sensor.

As can be seen in Figures A.4 and A.5, the airspeed measurements from the pitot-based sensors match the reference airspeed well, with large error at lower airspeeds for the MS4525DO sensor. At higher airspeeds, above $6\ \mathrm{m/s}$ the error of the MS4525DO becomes relatively small, below $0.4\ \mathrm{m/s}$. While the SDP33 requires calibration, it is also capable of exceptional performance, remaining within the reference airspeed bounds at nearly all airspeeds tested. The SDP33 also has far better noise performance than the MS4525DO sensor, due to its more precise differential pressure sensor, even at low airspeeds, though, at higher airspeeds the noise of the MS4525DO does decrease. The lower noise at high airspeed is an expected result of the conversion between differential pressure and airspeed. Even if a differential pressure sensor has no change in measurement noise as Δp increases, since $V_a \propto \sqrt{\Delta p}$, small changes in Δp when Δp is large lead only to minor changes in V_a.

A.2 RPM sensor

The Kotleta20 measures RPM explicitly as it drives the BLDC motor. Since propeller RPM has a profound effect on slipstream, it is necessary to confirm that this RPM measurement is accurate. For this validation, we mount the motor and propeller on a RCbenchmark Series 1580 test stand and use an optical RPM probe to serve a reference RPM measurement. The motor is controlled by the Kotleta20 ESC in the same configuration as is used in flight, RPM is recorded by the PX4 flight computer, and motor speed commanded manually by an operator. The motor validation test examines a steady RPM increase, and results can be seen in Figure A.7.

Optical RPM data is recorded by a personal computer connected to the thrust stand, whereas ESC RPM data is logged by the Pixracer flight computer. This does present a challenge when comparing the two curves as the data is not synchronized and is recorded at slightly different rates. The Kotleta20 data is logged at approximately, $50\ \mathrm{Hz}$ whereas the optical data is logged at $42.3\ \mathrm{Hz}$. To synchronize the two sets of RPM data, the operator commands a series of sharp throttle increases and decreases, the peaks of which are used in post-processing to synchronize the two datasets. It should be noted that this approach still cannot determine if the ESC based data lags the true propeller speed, but this effect is assumed to be negligible. The primary goal of this sensor validation was to confirm that the Kotleta20 sensor produced readings of the same magnitude as the optical sensor. Analysis of error also requires us to account for the different logging rates, to do so the

Kotleta20 data is linearly interpolated so that every data point in the optical dataset has a unique associate in the interpolated Kotleta20 dataset.

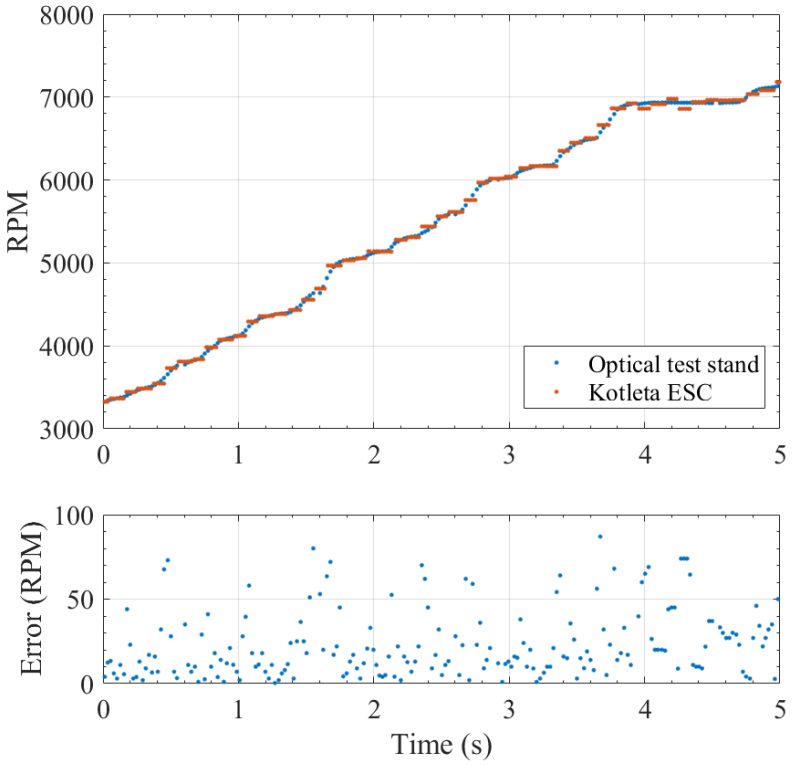

Figure A.7: Kotleta20 RPM validation test showing ESC measurements and optical measurements (top). The error between these two measurements is shown in the bottom subfigure.

Results of the RPM validation show that the embedded ESC measurements match well with the test stand's optical sensor. The root mean squared error (RMSE) between the Kotleta20 and optical RPM curves is $30.32\ \mathrm{RPM}$.

Appendix B

PX4 implementation of synthetic airspeed sensor

Our implementation of the cascaded wind estimator inside the flight computer has the benefit of providing wind and airspeed to all programs running on the system without additional modification. Since it may be of interest for future work, we provide a brief description of the software implementation in this appendix.

The experimental test flight uses a flight computer running PX4 for state estimation, process management, and control of actuators. The PX4 state estimator estimates wind and is used in experimental flights as part of the cascaded wind estimator. Essentially, to build the cascaded estimator, the synthetic airspeed estimator is added to the PX4 firmware, and its output is directed to the default PX4 wind estimator, which does not require modification. This demonstrates the rationale for using a cascaded estimator, as the synthetic airspeed sensor can be developed once and easily ported to different platforms using different native wind estimators.

PX4 is separated into multiple layers to facilitate high-speed sensor communication and actuator outputs at a low-level, while also providing a convenient node based structure for high-level applications [67]. Two layers are most relevant to the cascaded wind estimator, one which contains the sensor drivers and the other which contains applications on the system, also known as modules in PX4. Modules can be isolated processes that communicate between each other using the publish/subscribe message broker known as μORB. Modules are often non-blocking, allowing high-speed applications, such as an attitude controller, to work in tandem with low-speed applications such as a waypoint manager. μORB manages message communication between applications running on the

system at different speeds. The sensor driver layer operates at a lower-level and a high frequency, they also directly control the interfaces of the flight computer, communicating with sensors over SPI, I^2C, or UART. Sensor drivers publish their data to applications in the module layer through a μORB message.

Limiting the scope of PX4 to the components used in the cascaded wind estimator narrows our focus to a few key processes, which are visualized in Figure B.1.

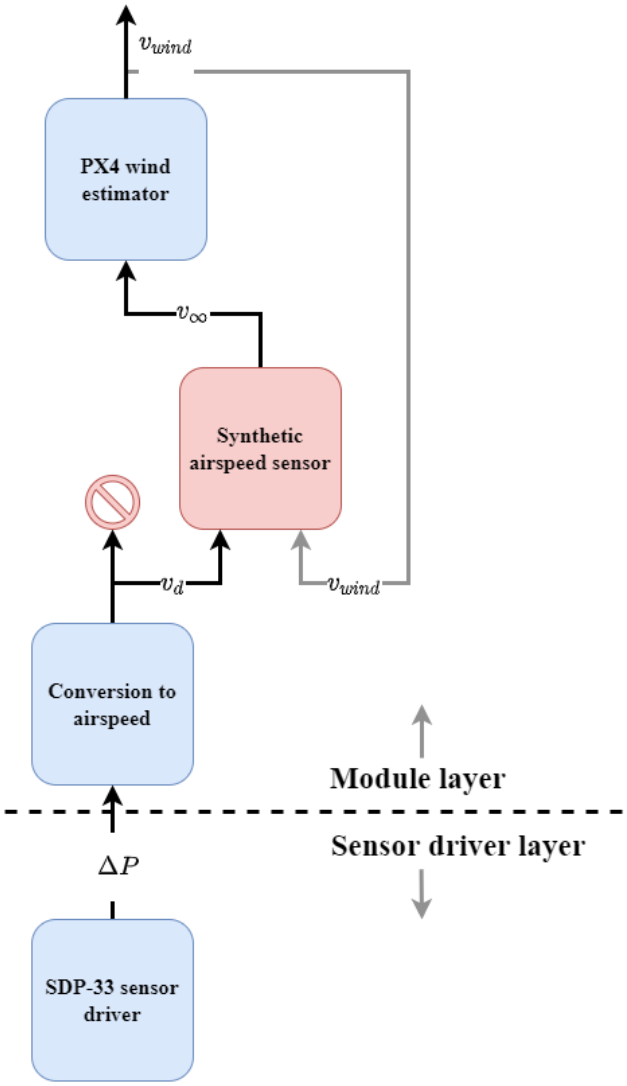

Figure B.1: High-level overview of cascaded estimator as it is implemented in PX4. Components unchanged from PX4 are coloured blue and our additions are highlighted in red.

In the stock PX4 firmware, the flow of information leading to the wind estimate is as follows

1. Sensor driver communicates with differential pressure sensor over I^2C

2. Sensor driver publishes differential pressure to system
3. An application in the module layer listens to the differential pressure topic and converts it to airspeed
4. This application then publishes the calculated airspeed to the system
5. The PX4 state estimator listens to the airspeed topic and, if available, fuses it to generate a wind estimate
6. The wind estimate is published to the system by the PX4 state estimator

To transform this into the cascaded wind estimator, which can estimate wind from a measurement inside the slipstream, a new module, the synthetic airspeed sensor, is created. This module listens to a new airspeed topic related to the slipstream, and a wind estimate if one is available. It fuses this information into a freestream estimate using either the EKF-SAS, described in Chapter 3, or the EF-SAS, designed in Appendix C. The freestream estimate is then published to the system under the stock airspeed topic, which is natively listened to by the PX4 state estimator. The flow of data for implemented cascaded estimator is as follows

1. Sensor driver communicates with differential pressure sensor over I^2C
2. Sensor driver publishes differential pressure to system
3. An application in the module layer listens to the differential pressure topic and converts it to airspeed
4. The application publishes the calculated airspeed to the system under a new topic name
5. The synthetic airspeed estimator listens to the new topic as well as the wind estimate, if both are available it fuses them into a freestream estimate
6. The freestream estimate is then published to the system under the airspeed topic
7. The PX4 state estimator listens to the airspeed topic and, if available, fuses it to generate a wind estimate
8. The wind estimate is published to the system by the PX4 state estimator

Since the synthetic airspeed sensor is published to the system under the general airspeed topic, all other applications which listen to airspeed will use the results of the synthetic airspeed sensor without further modification. While other uses of freestream on agile platforms are out of the scope of this work, it further highlights the advantage of the cascaded approach. Though it is not measured in flight, ground tests showed a freestream estimate publication rate of ~ 100 Hz on a PixHawk 4 flight computer, which was deemed sufficiently fast for use in this work.

Appendix C

Alternative synthetic airspeed sensor (exponential filter)

One potential alternative to the EKF-SAS, designed in Chapter 3, uses a simple exponential filter to estimate freestream speed. This implementation of the synthetic airspeed sensor is covered in this appendix.

The exponential-type filter implementation of the synthetic airspeed sensor (EF-SAS) requires a measurement inside the slipstream v_d, location of the measurement $\underline{P}_d$, the platform's longitudinal vector $\hat{b}_1$, inertial velocity $\underline{V}_i$, and wind estimate $\underline{V}_{wind}$. The motivation behind the EF-SAS is that, provided a wind estimate, freestream can be calculated using either the slipstream model or a wind estimate. While the freestream velocity calculated through the wind estimate is expected to produce low noise results, they are also expected to have poor accuracy, suffering from drift over time. Conversely, the freestream velocity estimate from the slipstream model is expected to be noisy but have good accuracy. The EF-SAS is composed by low-pass filtering the freestream velocity calculated through the slipstream model. Traditionally the exponential filter uses the results from the previous filter output, instead we use the wind triangle based freestream calculation in its place. This is done for two reasons, firstly, wind is expected to be much more constant than airspeed, secondly, it is also expected to be more consistent as the platform changes heading. The wind estimate and freestream estimate develop simultaneously.

The first step in the exponential-type filter approach is to calculate v_∞ using both the wind triangle and slipstream model, labelled v_∞^w and v_∞^m respectively.

v_∞^w can be calculated from the wind triangle as

$$v_\infty^w = \hat{b}_1^T \cdot (\underline{V}_i - \underline{V}_{wind}) \tag{C.1}$$

Calculation of v_∞^m requires more careful consideration, since the slipstream model must be inverted. For this, we once again use the polynomial fit of the slipstream model generated in Chapter 3. Inversion of Eq. (3.8) is done relatively easily as n is assumed to be known exactly from an RPM sensor, reducing (3.8) to a quadratic function of v_∞^m. The solution for v_∞^m given v_d and n is then expressed as

$$v_\infty^m = -\left(0.528 + 0.002n\right) \pm \frac{\sqrt{\left(0.528 + 0.002n\right)^2 - 0.04\left(0.744n + 0.04 - v_d\right)}}{0.02} \tag{C.2}$$

Eq. (C.2) does usually generate two solutions, however for $v_d, n > 0$ only one of them is positive which is the solution we use. In the event a solution does not exist, which may occur briefly during agile manoeuvring where measurements of v_d can be inaccurate, it is straightforward to find the closest solution or set v_d to a constant value.

Once v_∞^w and v_∞^m are calculated, the filtered airspeed estimated, v_∞ can be computed using the following equation, where α is a tuning parameter of the filter [68]

$$v_\infty = \alpha \cdot v_\infty^w + (1 - \alpha) \cdot v_\infty^m \tag{C.3}$$

When α is set high, v_∞ is heavily filtered by the wind triangle, reducing noise but making it more prone to drift and slower to respond to the true freestream value. Decreasing α has the opposite effect, increasing the noise in the v_∞ estimate, but also making it more reactive to the true freestream. To avoid poor slipstream measurements from the airspeed sensor under oblique flow conditions from adversely impacting our wind estimate, we increase α if the oblique flow angle is detected to be past a certain critical angle. A heuristic approach based-on the oblique flow angle Φ is used for this, and Φ can be calculated as follows, assuming the sensor's measurement axis is aligned with the platform's longitudinal axis $\hat{b}_1$

$$\Phi = \cos^{-1}\left(\hat{b}_1^T \frac{\underline{V}_a}{||\underline{V}_a||}\right) \tag{C.4}$$

The heuristic function mapping Φ to α is proposed to keep α at a nominal value, α_{min}, when Φ is low, then quickly increase it to some high value, α_{max}, when the oblique flow

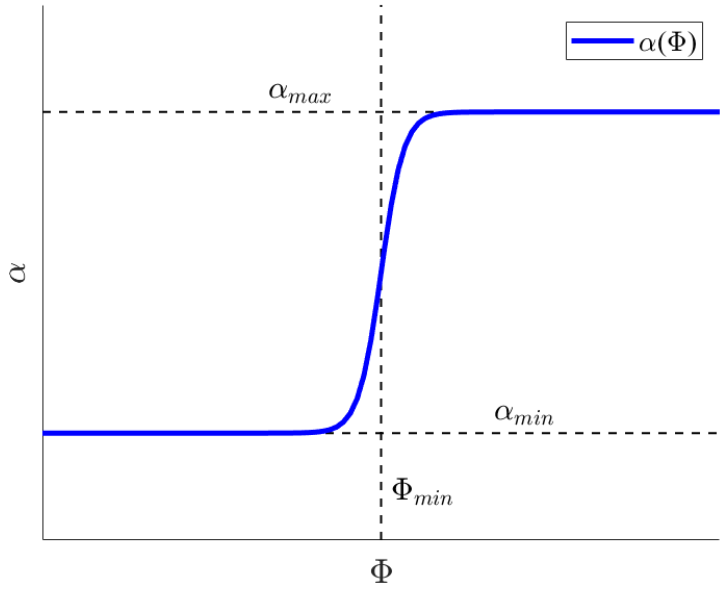

Figure C.1: Relationship between $\alpha(\Phi),\ \alpha_{min},\ \alpha_{max},\ \text{and},\ \Phi_{\text{min}}$ for Eq. (C.5).

angle is high. This switching from α_{min} to α_{max} begins at a value Φ_{min}. Essentially, when an airspeed measurement dropout is expected due to oblique flow, the impact of v_d on the freestream estimate is diminished by rapidly increasing α. The heuristic function is visualized in Figure C.1 and is defined as follows

$$\alpha(\Phi) = \frac{1}{2}\left(1 - \tanh(\Phi - \Phi_{min} - 2)\right) \cdot (\alpha_{min} - \alpha_{max}) + \alpha_{max} \tag{C.5}$$

The parameters: $\alpha_{min},\ \alpha_{max},\ \text{and},\ \Phi_{\text{min}}$ are chosen based-on offline estimator tuning. The initial and final values for α are chosen to balance noise and responsiveness in the freestream estimate, and Φ_{min} is chosen experimentally based on the observed oblique flow performance of the airspeed sensor. When the EF-SAS is used for testing, the values of $\alpha_{min},\ \alpha_{max},\ \text{and},\ \Phi_{\text{min}}$ will be listed.

The EF-SAS is also tested offline, identically to the offline EKF-SAS tests performed in Section 3.5.1. Key parameters for the cascaded wind estimator in this configuration can be seen in Table C.1.

The freestream and slipstream measurements, along with the estimated freestream from the EF-SAS can be seen in the top subplot of Figure C.2. The α parameter of the exponential filter can also be seen in Figure C.2, occupying the bottom plot.

Table C.1: Cascaded wind estimator configuration parameters used in offline EF-SAS testing.

EF-SAS parameters	value
α_{min}	0.1
α_{max}	0.9
Φ_{min}	22°
Wind estimator parameters	
$\boldsymbol{Q}$	$\mathrm{diag}(0.01, 0.01, 0.01)$ $\mathrm{m}^2/\mathrm{s}^2$
R	2 $\mathrm{m}^2/\mathrm{s}^2$

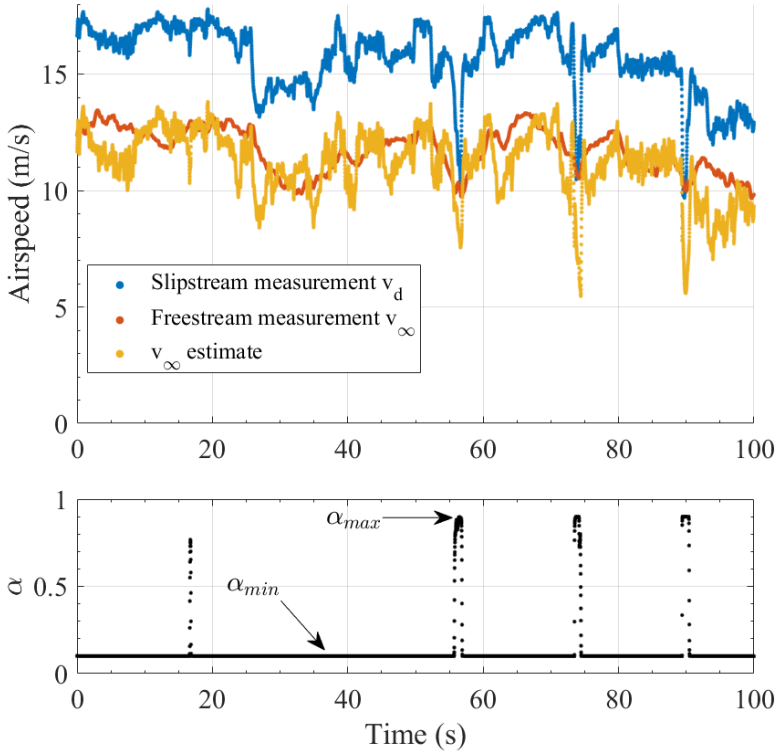

Figure C.2: The top plot shows measured v_∞ and v_d from outdoor flight test, compared to estimated freestream using the EF-SAS. The bottom plot shows the value of α; spikes in α attempt to reject inaccurate airspeed measurements under oblique flow conditions.

Figure C.2 shows that the synthetic airspeed sensor is reasonably effective at estimating freestream however there is a high level of noise. Airspeed measurement dropouts, likely due to oblique flow, can be clearly seen in this figure at approximately $t = 55,\ 75,$ and 90 s. During these oblique flow events α increases rapidly to α_{max} based on Eq. (C.5) to reduce the influence of the slipstream sensor on the freestream estimate. The reaction

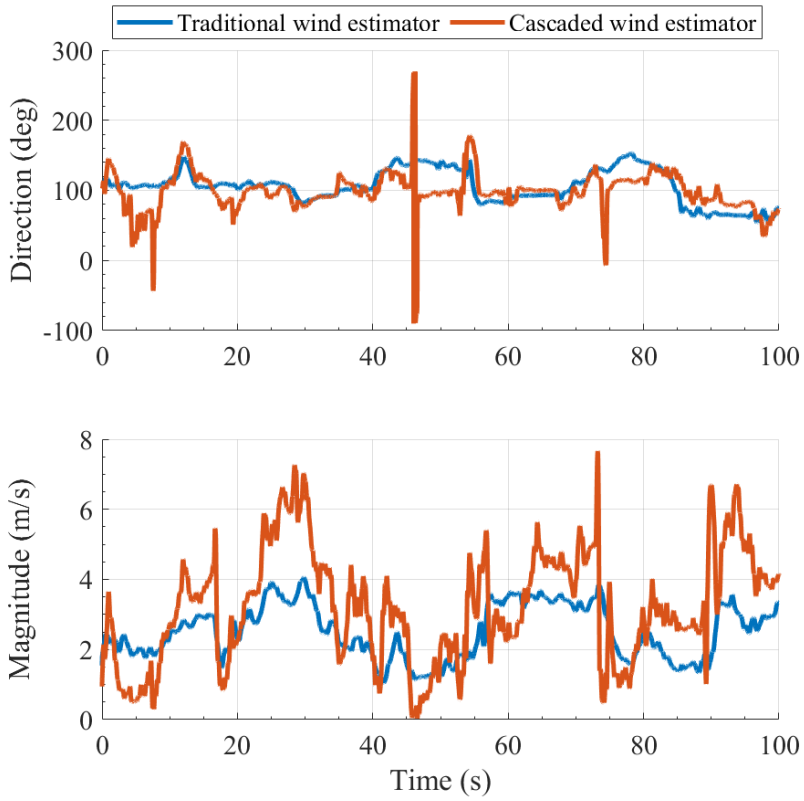

Figure C.3: Estimated wind magnitude and direction using the traditional wind estimator (blue), and the cascaded wind estimator (orange). The traditional wind estimator relies on the measured v_∞, whereas the cascaded wind estimator uses the EF-SAS to estimate v_∞.

from α is not fast enough to completely mitigate the oblique airspeed sensor readings, but is sufficient for the scope of this work. Better detection of oblique flow conditions and associated airspeed measurement dropout may be an interesting topic for future study.

The two wind estimates, one using the EF-SAS and the other using the freestream measurement, can be seen in Figure C.3. The result of the wind estimator using the freestream measurement is labelled the 'traditional' wind estimator, and the estimator using the EF-SAS is labelled the cascaded wind estimator.

As can be seen in Figure C.3, the cascaded wind estimator yields a qualitatively similar result to the traditional wind estimator, particularly in terms of wind direction, however the magnitude estimate is substantially noisier than the baseline. Additionally, the oblique flow dropouts can be seen to affect the wind estimate, with sharp changes in the estimate apparent at the instants of high oblique flow. While the wind estimate using the

synthetic sensor does not provide an excellent match with the base-line, it is still seen to provide reasonable performance.