DISS. ETH NO. 28647

PERCEIVING THE AIR FOR SAFER AND MORE EFFICIENT FIXED-WING UAV FLIGHTS

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

FLORIAN ACHERMANN

BSc ETH in Mechanical Engineering, ETH Zurich
MSc ETH in Robotics, Systems and Control, ETH Zurich

born on 02.08.1992
citizen of Switzerland

accepted on the recommendation of

Prof. Dr. Roland Siegwart, Examiner
Prof. Dr. Jack W. Langelaan, Co-examiner

2022

Autonomous Systems Lab
Department of Mechanical and Process Engineering
ETH Zurich
Switzerland

© 2022 Florian Achermann. All rights reserved.

Abstract

Recent advances in small uncrewed aerial vehicle (sUAV) technology has caused a surge in using sUAVs in a wide range of scientific and commercial applications. Specifically, fixed-wing and hybrid sUAVs allow large-scale beyond visual line of sight (BVLOS) missions, e.g. distributing medical goods to isolated locations or remote glacier monitoring. Current BVLOS regulations require low altitude operations where the wind can exhibit complex flow patterns and chaotic velocity changes that pose safety risks to sUAVs. On the other side, the wind and thermals, pockets of hot rising air caused by temperature variations on the ground, can be exploited to extend the range and flight time of soaring sUAVs. However, current sUAVs lack the ability to remotely predict the wind and potential thermal locations. The primary goal of this thesis is to develop efficient algorithms to remotely perceive the air, specifically the low-altitude wind around terrain and thermal updrafts, for safer and more efficient sUAV flight.

In Part A we tackle the challenge of predicting the dense low-altitude wind around complex terrain. We trained deep neural networks (DNNs) to predict the wind using computational fluid dynamics (CFD)-simulated flows over realistic terrain patches. In a first version, the input to the DNN is composed of the known elevation map and wind inflow condition. We demonstrated that these wind predictions allow planning for safer and more efficient flight paths. As the inflow condition is not available onboard during flight we developed a second DNN, *WindSeer*, to predict the wind and turbulence based on the elevation map and sparse wind measurements. These sparse measurements can be collected onboard an sUAV during flight. We demonstrate zero-shot sim-to-real transfer by evaluating *WindSeer*, trained only with CFD simulated data, on real wind data without retraining. *WindSeer* accurately predicts the wind and turbulence collected at weather stations across different hills in Europe and can accurately reconstruct the wind given onboard measurements as validated by data from multiple fixed-wing sUAVs.

In Part B we address the challenge of predicting and detecting remote thermal updraft locations. We first enabled building a temperature map of the ground by developing *MultiPoint*, a DNN-based keypoint detector and descriptor for the optical and thermal infrared (TIR) spectrum. We composed a dataset of aligned optical-TIR image pairs using a mutual information (MI)-based pipeline. *MultiPoint*, trained in a multi-stage pipeline in a self-supervised fashion, significantly outperforms baseline detectors and descriptors on multi-spectral image data. In a next step, we demonstrated in a proof of concept that DNNs are able to detect schlieren, brightness and color changes due to refractive index gradients in air, using a single greyscale image. We first collect a dataset of schlieren optical flows in an ideal indoor lab setting labelled with background oriented schlieren (BOS) methods. Then we trained the DNN using the labelled flows with a mix of real imagery and synthetically generated images. Finally, we showed the performance of the model on held back data and real-world imagery.

Overall, in this thesis we presented DNN-based methods able to run onboard an sUAV perceiving the air to enable planning for safer and more efficient flight paths for BVLOS missions. We conclude the thesis with suggestions to extend this work.

Zusammenfassung

Jüngste Fortschritte in der Technologie kleiner unbemannter Luftfahrzeuge (Engl. *small Uncrewed Aerial Vehicles*, sUAVs), auch Drohnen genannt, haben zu einem Anstieg beim Einsatz von ihnen in einer Vielzahl von wissenschaftlichen und kommerziellen Anwendungen geführt. Insbesondere Starrflügler- und Senkrechtstarter-Drohnen ermöglichen grossflächige Einsätze ausserhalb der direkten Sicht, sogenannten BVLOS-Missionen (Engl. *Beyond Visual Line of Sight*). Beispiele solcher Missionen sind die Verteilung von medizinischen Gütern an abgelegene Orte oder die Fernüberwachung von Gletschern. Die aktuellen BVLOS-Vorschriften erfordern Flugbetrieb in geringer Höhe. Nahe dem Gelände kann der Wind komplexe Strömungsmuster und chaotische Geschwindigkeitänderungen aufweisen, welche ein Sicherheitsrisiko für die Drohnen darstellen. Andererseits können Wind oder Thermiken ausgenutzt werden um die Reichweite und Flugzeit von Drohnen zu verlängern. Thermiken sind Aufwinde die durch Erwärmung der Luft in Bodennähe durch Sonneneinstrahlung entstehen. Im Moment fehlt jedoch Drohnen die Fähigkeit den Wind und mögliche Thermiken vorherzusagen. Das primäre Ziel dieser Arbeit ist die Entwicklung zweckmässiger Algorithmen um die Luft aus der Ferne wahrzunehmen, insbesondere den Wind in Bodennähe und thermische Aufwinde, um sicherere und effizientere Drohnenflüge zu ermöglichen.

In Teil A stellen wir uns der Herausforderung den Wind in Bodennähe mit hoher Auflösung vorherzusagen. Um dies zu erreichen haben wir künstliche neuronale Netze (Engl. *Deep Neural Networks*, DNNs) mit Winden über reales Gelände, berechnet mit numerischer Strömungssimulation, trainiert. In einer ersten Version hat das DNN den Wind mit der bekannten Höhenkarte und dem Anströmungsprofil vorhergesagt. Wir haben gezeigt, dass diese Windvorhersagen die Planung sicherer und effizienterer Flugrouten ermöglichen. Da das Anströmungsprofil im Flug nicht verfügbar ist, haben wir eine zweite Version des DNNs, *WindSeer*, entwickelt. *WindSeer* berechnet die Wind- und Turbulenzvorhersage basierend auf der Höhenkarte und einigen wenigen Windmessungen. Die Drohne kann diese Windmessungen während dem Flug ausführen. Wir zeigen, dass *WindSeer* den realen Wind vorhersagen kann, obwohl es nur mit simulierten Strömungen trainiert wurde. Wir validieren die Wind- und Turbulenz-Vorhersagen von *WindSeer* mit Messungen von Wetterstationen über verschiedene Hügel in Europa und dem Wind gemessen von mehreren Starrflüglern-Drohnen.

In Teil B befassen wir uns mit der Aufgabe, entfernte thermische Aufwinde zu erkennen und vorherzusagen. Zunächst ermöglichen wir das Erstellen einer Temperaturkarte des Bodens mit merkmalsbasierten Verfahren durch die Entwicklung von *MultiPoint*, einem DNN zur Erkennung und Beschreibung wichtiger Kontrollpunkte von optischen und thermalen Bildern. Wir haben unter Anwendung einer auf Transinformation (Engl. *Mutual Information*, MI) basierenden Pipeline einen Datensatz mit ausgerichteten optischen-thermalen Bildpaaren erstellt. *MultiPoint*, trainiert mit einem mehrstufigen Prozess, übertrifft die Ergebnisse von existierenden Methoden zur Erkennung und Beschreibung wichtiger Kontrollpunkte in multispektralen Bildern. In einem nächsten Schritt haben wir in einem Machbarkeitsnachweis gezeigt, dass DNNs Schlieren in einem einzigen Graustufenbild erkennen können. Schlieren werden von lokalen Schwankungen des Brechungsindex verursacht und führen zu Helligkeits- und Farbänderungen im aufgenommenen Bild. Zuerst generierten wir einen Datensatz mit Bildern von Schlieren und dem assoziierten optischen Fluss in einer idealen Laborumgebung. Die optischen Flüsse werden mit hintergrundorientierter Schlierenfotografie (Engl. *Background Oriented Schlieren*, BOS) berechnet. Wir haben das Netzwerk mit einer Mischung aus realen und synthetisch generierten Bildern trainiert. Schliesslich zeigten wir die Ergebnisse der Schlierendetektion auf neu aufgenommen Innen- und Aussenaufnahmen.

Zusammengefasst haben wir in dieser Dissertation DNN-basierte Methoden vorgestellt, die in der Lage sind die Luftströmungen der Umgebung mit Messwerten von kleinen Flugdrohnen wahrzunehmen um die Planung sichererer und effizienterer Flugrouten für BVLOS-Missionen zu ermöglichen. Wir schliessen die Dissertation mit Vorschlägen zur Erweiterung dieser Arbeit ab.

Acknowledgements

During my stay at the Autonomous Systems Lab (ASL) I had the pleasure to work with and learn from extremely talented and motivated people. Their teaching, help, and support throughout this time was crucial, helped me to improve, and lead me to where I am today.

First and foremost I want to thank Prof. Roland Siegwart for giving me the chance to pursue my PhD at the ASL. You created a wonderful open-minded environment at the ASL with great collaboration between the groups that allows the freedom to even explore the craziest ideas, sometimes suggested by you. Your great support during the SOLAR^3 project and the BVLOS flight over Lake Neuchâtel was crucial to a successful outcome in both projects!

Thank you Prof. Jack Langelaan for agreeing to co-examine this thesis. Reading about your work in field of autonomous soaring and optimal planning in wind with fixed-wing UAVs was always an inspiration during the PhD. I hope that the work in this thesis will spark some inspiration in return.

A huge thanks goes to my additional PhD supervisors Dr. Nicholas Lawrance and Dr. Jen Jen Chung. Your guidance during our meetings helped me to stay on track and improve as a researcher. You always had an open ear and the time to discuss new ideas or the current progress if I was stuck. Finally, your support in the last few field test to gather the windy data in the Swiss Alps was key to success. I hope you enjoyed the field tests as much as I did.

I also want to express my gratitude towards the PIs of the projects I completed in collaboration with Intel and Microsoft: Alexey Dosovitskiy, René Ranftl, and Andrey Kolobov. Especially Andrey, I really enjoyed your support and feedback during the three years of Project Altair and meeting you in person.

To all the people in the fixed-wing team: Amir Melzer, Philipp Oettershagen, Thomas Mantel, Timo Hinzmann, Thomas Stastny, Jaeyoung Lim, David Rohr, and Nicholas Lawrance. I could learn from you about every aspect of fixed-wing and tilt-wing UAVs ranging from design, state estimation, modelling to control and path planning. I really enjoyed the time I spent with you on the field and watch the planes fly after countless hours of preparation and debugging. It was a pleasure meeting you all and working together with you!

The flight tests wouldn't have been so successful without our safety pilots that more than once rescued the planes by taking over from the autopilot: Jonas Langenegger, Tizian Steiger, Yves Allenspach, Rainer Lotz, Jayeoung Lim, David Rohr, and Thomas Stastny.

Luciana Borsatti, Cornelia Della Casa, your help in organizing events and meetings, travelling, and making sure all the paperwork is done made my life during

Figure 1: Without the support of many people successfully conducting the flight tests would not have been possible.

the PhD so much easier. Michael Riner-Kuhn, and Matthias Müller, together with all the Zivis, you ensure the lab equipment is well organized and we use the tools in a safe way. All of you are the corner stones of the lab and keep it running like a well oiled machine.

The people at the ASL are like a big family. I enjoyed the time we spent together in the lab hikes, ski weekends, or terrace barbecues and will always remember some of the crazy stories that happened during these events.

I am grateful for all the support I received from my friends and teammates in the various sports that I practice. The time I spent with you outside the lab gave me the energy to keep going during the PhD. Last, but not least my gratitude goes to my family for always supporting my journey.

December 6, 2022 *Florian Achermann*

Financial Support

This work was supported during the first year by the Intel Network on Intelligent Systems. The last three years were funded by the Microsoft Swiss Joint Research Center under Contract No. 2019-038 “Project Altair: Infrared Vision and AI Decision-Making for Longer Drone Flights”.

Contents

Abstract **i**

Zusammenfassung **iii**

Acknowledgements **v**

Preface **1**

1 Introduction **3**
- 1.1 Challenges and Approach 6

2 Contributions **9**
- 2.1 Part A: Wind Prediction 9
- 2.2 Part B: Thermal Detection 12
- 2.3 List of Publications 15
  - 2.3.1 Publications included in this thesis 15
  - 2.3.2 Other publications 16
- 2.4 List of Supervised Students 16
- 2.5 List of Open-source Software 18
- 2.6 Miscellaneous Contributions 19

3 Conclusion & Outlook **21**
- 3.1 Discussion 22
- 3.2 Future Work 25

A. WIND PREDICTION **29**

Paper I: Learning to Predict the Wind for Safe Aerial Vehicle Planning **31**
- 1 Introduction 32
- 2 Related Work 34
- 3 CFD Wind Data 35
- 4 Wind Prediction 37
- 5 Experiments 38
- 6 Conclusions 43

Paper II: WindSeer: Low-altitude real-time volumetric wind prediction over complex terrain aboard a small UAV **45**
1 Introduction 47
2 Results 50
3 Discussion 65
4 Materials and Methods 68
5 Supplementary Materials 73

B. THERMAL DETECTION **85**

Paper III: MultiPoint: Cross-spectral registration of thermal and optical aerial imagery **87**
1 Introduction 89
2 *MultiPoint* – Cross-spectral interest point detection and description 91
3 Multi-spectral aerial image dataset 94
4 Results 97
5 Conclusion 99
6 Appendix 100

Paper IV: An Outlook on Single Frame Thermal Column Detection using a CNN **109**
1 Introduction 110
2 Label Flow Generation 112
3 CNN Training for Single Frame Schlieren Detection 113
4 Experiments/Results 117
5 Conclusion and Outlook 123

Bibliography **126**

Curriculum Vitae **145**

Preface

This is a cumulative doctoral thesis and as such consists of the three relevant publications of the author during the doctoral studies and an additional unpublished technical brief. The individual papers are framed by an overarching introduction and conclusion section offering an overview about the work in this thesis and connecting the individual publications.

Chapter 1 introduces the motivation, challenges, and the goal of this research. Chapter 2 explains for each contributing paper its context to the current state of the art and how it embeds within the overall goal of the thesis and connects to the other publications. The publications attached at the end are grouped into two parts: In Part A the publications present low altitude wind prediction around complex terrain and Part B introduces two modalities to predicting thermal updraft locations. Chapter 3 closes this thesis by a summary of the achievements and provides an outlook for future research directions.

Chapter 1

Introduction

Recent advances in small uncrewed aerial vehicle (sUAV) technology has caused a surge in using sUAVs in a wide range of scientific and commercial applications such as monitoring glaciers [56], precision agriculture (smart farming) [65], search and rescue [67, 120], avalanche detection [21], and transmission line inspection [166].

The deployed sUAVs can be classified depending on their propulsion system: Rotary-wing sUAVs offer vertical take-off and landing (VTOL) capabilities and high maneuverability and thus are commonly used in indoor or cluttered environments. However, they are usually energy-inefficient with restricted travel speeds, which leads to limited flight time and range [111]. Their fixed-wing counterparts are more energy-efficient, leading to much longer flight time and range — even multi-day flight in ideal conditions [98] — but require dedicated take-off and landing runways and are nonholonomic systems, complicating the path planning. Hybrid sUAVs, also called VTOL aircraft, combine the advantages of rotary-wing and fixed-wing sUAVs by taking off vertically in rotary-wing mode and then cruising efficiently in fixed-wing mode. They achieve high endurance, just slightly below pure fixed-wing sUAV flight, but controlling the transition between the flight modes is a challenge [151].

So far, the majority of civil research and commercial application deploy rotary-wing sUAVs due to their maneuverability, ease of use and cost. Recent developments providing commercially available fixed-wing and hybrid sUAVs [167] allow for larger scale beyond visual line of sight (BVLOS) missions previously not feasible with rotary-wing sUAVs. Such examples include distributing COVID-19 test kits to remote locations in Ghana as well as collecting and returning the samples for processing at specialised test stations [69] or monitoring remote glaciers in Greenland [56].

One of the main limitations to broader deployments of sUAVs stems from flight restrictions due to national airspace regulations. Certain approval processes for BVLOS missions allow only operating at low flight altitudes, for instance the STS

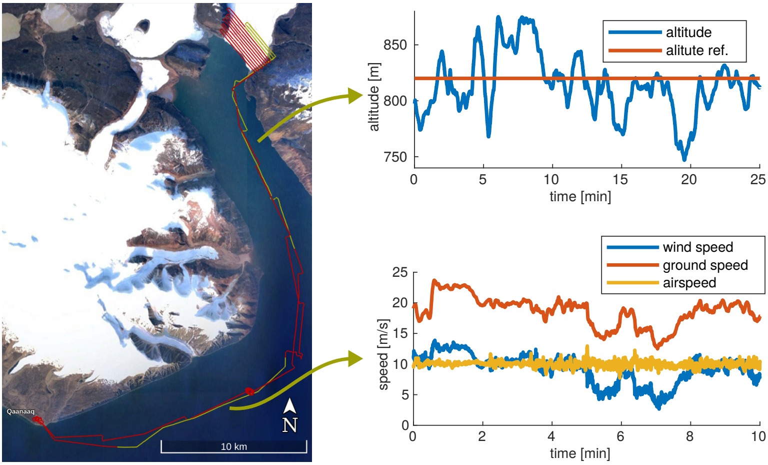


Figure 1.1: The flight path of AtlantikSolar from Qaanaaq to the Bowdoin glacier during the Sun2Ice project. During the flight the wind exceeded the airspeed of AtlantikSolar and caused altitude tracking errors up to 70 m. These strong winds were not modelled by NWP.

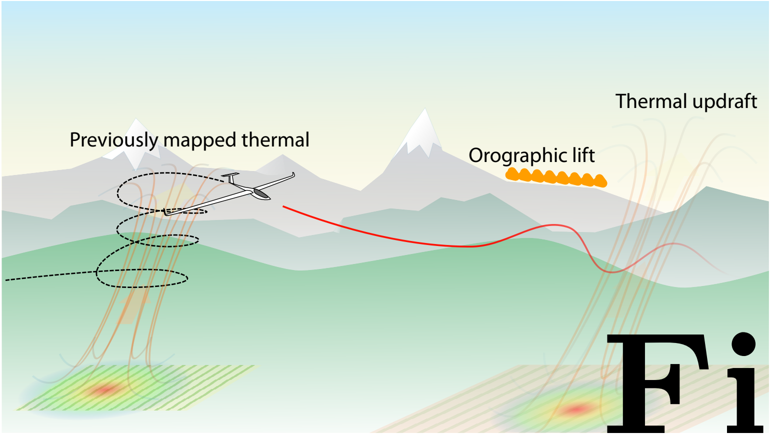
or EU STS-02 permits allow a maximum altitude of 120 m above ground. The wind at such low altitudes can negatively impact the safety of an sUAV due to the complex flow patterns and chaotic velocity changes, called turbulence, caused by the interactions of the air and terrain. In particular, winds in mountainous regions can exceed $10\,\mathrm{m\,s^{-1}}$ [125], a speed similar to the normal cruise speed of sUAVs [97] resulting in poor tracking of the planned flight path [137].

Path planning frameworks to plan safe and efficient BVLOS missions, such as MetPASS [100], use meteorological data from numerical weather prediction (NWP) models. These models provide large-scale wind predictions with grid resolutions on the order of kilometers. However, complex fluid-dynamic effects due to local steep terrain [22] cause wind at low altitudes to differ significantly from these predictions. During the Sun2Ice project[1] where AtlantikSolar [97] monitored remote Glaciers in Greenland it was subject to terrain-induced rotors and strong valley winds exceeding the aircraft's maximum airspeed not predicted by NWP, causing altitude deviations up to 70 m as shown in Fig. 1.1.

Besides being a safety risk, wind can also be viewed as an energy source that can be exploited to extend the range and duration of sUAVs. Human glider pilots and birds are keenly aware of the wind around terrain and have learnt to exploit orographic lift for efficient flight [123]. Similarly, they have also learnt to target

---

[1] https://sun2ice.ethz.ch/


Previously mapped thermal
Orographic lift
Thermal updraft
Fi

1.1 Challenges and Approach

To achieve our goal of safe and efficient sUAV flight we want to tackle the following challenges:

Limited Onboard Computing BVLOS missions over long distances usually span multiple hours, thus the sUAV must be able to react and re-plan the flight path subject to changing environmental conditions. Especially in remote areas, such as Greenland during the Sun2Ice mission, low bandwidth communication links render cloud-based computing infeasible. All algorithms must run onboard to guarantee autonomy during the whole flight even in case of communication loss. Recent developments in computer hardware have resulted in small-scale and power-efficient graphics processing units (GPUs) suitable for use onboard an sUAV, for instance the Jetson Xavier or Orin.

In this thesis we aimed to develop efficient algorithms suitable to run in real-time with onboard computing. To achieve this we use deep neural network (DNN)-based approaches that efficiently utilize the GPU of the onboard compute.

High-resolution Onboard Wind Prediction While NWP can accurately model long term and large-scale wind patterns, the local low-altitude winds might significantly differ from these predictions due to terrain features at smaller scales than are represented in the NWP models [22]. Although remote 3D wind sensing systems exist, such as multi-Doppler LiDAR [150], they are not suitable for measuring the wind onboard an sUAV due to their large mass of around 180 kg. Computational fluid dynamics (CFD) simulations can estimate high-resolution wind flows and turbulence kinetic energy (TKE) around terrain at smaller scales, on the order of meters or less, but are much too expensive to compute in real-time aboard an sUAV and require well-defined boundary conditions reflecting the overall weather situation [15, 17]. TKE is a metric for the strength of the turbulent velocity fluctuations in the wind field that is proportional to the sum of the variances in each dimension.

Onboard the sUAV, high-resolution terrain maps are available either from online mapping pipelines [127] or pre-flight map services [140, 149]. Noisy wind measurements along the flight path can be observed on an sUAV equipped with the appropriate airflow sensing equipment.

The goal of this thesis is to predict the dense wind around the sUAV by only relying on onboard wind measurements and the known elevation map. Doing so we first generated a dataset of dense labelled flows over realistic terrain patches from Switzerland using a CFD solver. We then trained a DNN solely on this synthetic data. Finally, we evaluated it on held back CFD-simulated flows, real wind data gathered in measurement campaigns [15, 17, 39, 142, 143], as well as in-situ wind measurements from multiple sUAVs.

Consistent Optical and TIR Mapping Thermal updrafts are caused by temperature variation on the ground [9, 16]. Generating a temperature map of the ground enables the sUAV to remotely detect potential updraft locations. This allows more consistent exploitation of thermal updrafts as the current state of the art relies on chance to find a thermal. Creating consistent cross-spectral maps using mapping pipelines such as Maplab [127] is currently thwarted by the poor performance of existing cross-spectral feature detectors and descriptors [6, 14, 28, 78].

In this thesis we developed a DNN-based keypoint detector and descriptor for cross-spectral image registration. We generated first a dataset of aligned pairs of optical and thermal infrared (TIR) images of agricultural and forested terrain gathered by an sUAV. The proposed network is trained in a multi-stage pipeline by first generating viewpoint invariant keypoints consistent across both spectra and then training the DNN with the generated label keypoints.

Direct Thermal Updraft Detection Depending on the atmospheric condition, different types of thermals might form. Chimney thermals are a continuous column of rising air attached to a generating spot on the ground. Bubble thermals are detached vortex rings. They form on the ground and detach once they are buoyant enough and rise in altitude over time and drift with the wind [16]. In this latter case, detecting the temperature differences on the ground is not sufficient to predict the updraft location.

Temperature differences in air cause slightly different refractive indices [26] resulting in deflections of the light rays and eventually small brightness and color changes called schlieren (from the German word for ‘streaks’) [130]. The schlieren can be made visible to the human eye using a specific lab setup [129] or a more generic setup assuming the background is known via background oriented schlieren (BOS) [109, 134]. However, BOS methods are not suitable to run onboard sUAVs as they are computationally too expensive and the undistorted background is not available.

The goal of this thesis is to detect small distortions of the schlieren using a single optical image with a convolutional neural network (CNN). In an optimized static indoor setting we recorded schlieren patterns over heat sources using BOS methods. We then trained a CNN to predict the two-dimensional flow of the schlieren using a mixture of real and synthetic samples. The real images are captured with the indoor setting and the synthetic samples are images from the Places Standard dataset [171] warped with the optical flow from the recorded schlieren patterns. We evaluate our approach on unseen flow patterns and backgrounds in indoor and outdoor environments.

Chapter 2

Contributions

This chapter outlines the contributions of each of the manuscripts presented as part of this cumulative thesis. We describe the context of the work at the time of publication, the novel contributions, and how the work relates to the rest of the thesis and other publications.

Additionally, we present an overview of all authored and co-authored papers published during the doctoral studies as well as a list of supervised student theses and software projects.

2.1 Part A: Wind Prediction

Paper I

Florian Achermann, Nicholas R.J. Lawrance, René Ranftl, Alexey Dosovitskiy, Jen Jen Chung, and Roland Siegwart, "Learning to Predict the Wind for Safe Aerial Vehicle Planning". In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

Context

Flying at low altitude in complex terrain without an accurate wind prediction imposes a risk to the aircraft's safety as the wind speeds can easily exceed the travel speed of small fixed-wing uncrewed aerial vehicles (UAVs) [125, 138]. If the wind is accurately known, existing algorithms can account for the wind to plan safe and efficient paths [23, 25, 101, 160]. However, current sources of wind data either do not offer high enough resolution (NWP: 1.1 km or more) or are computationally too expensive for real-time wind predictions (CFD: multiple hours). Recent work has successfully shown the use of DNNs in computer graphics to render time-varying flow fields in real-time [38, 91, 164]. These methods are optimized for visual appearance and not physical accuracy. The accurate steady-state flow, together with additional properties such as drag or lift coefficients, have been predicted as well by DNNs around relatively simple geometries such as cars or airfoils [12, 42, 148].

Contribution

In this paper we focused on accurately predicting simulated steady wind flow around realistic complex terrain using a DNN. We generated a dataset of CFD simulated flows over realistic 1.2 km square terrain patches using a Reynolds-averaged Navier–Stokes (RANS) solver and validated our simulation setup with the publicly available Bolund hill benchmark [15, 17]. We trained a DNN to replicate the CFD simulated flows using the known elevation map and inflow conditions. The resulting network predicted previously unobserved flows well, demonstrating the ability of a DNN to predict complex fluid flows. We showed that using the more accurate predictions of the DNN compared to baseline methods in a planning framework yields safer and more efficient paths.

The following list summarizes the technological contributions of this paper:

- Composing a dataset of simulated RANS CFD simulation flows.
- Training and evaluating a DNN to replicate the CFD simulated flows.
- Demonstrating the effect of the wind prediction on planning safe and efficient flight paths.

Interrelations

This paper presents the initial idea of predicting the wind around complex terrain by imitating CFD simulated flows. However, it relies on input information not typically available onboard an sUAV. We extended this work in Paper II to enable onboard wind prediction based on sparse measurements collected onboard an sUAV during flight.

Accurate wind predictions are not only important for safe and efficient sUAV flight, but also have direct commercial applications such as for optimising the design and operation of wind farms [168]. Traditionally, computationally expensive CFD simulation models have been used to model the flow around wind turbines [53]. More recently, data-driven methods using DNNs are replacing CFD simulations to model certain flow phenomena such as wake interaction between multiple turbines [159, 170]. So far, the modelling targets have been offshore wind farms, thus neglecting the effect of the terrain on the flow. However, for wind farms in mountainous areas, such as the Gotthardpass[1] or Mount Crosin[2], the terrain effects cannot be neglected. Thus, an immediate practical application of

---

[1] `https://www.aet.ch/DE/Gotthard-Windpark-fa0c5600`

[2] `https://www.juvent.ch/de`

our work on predicting terrain-induced flows could be to combine it with existing wake models to optimize wind farms in mountainous terrain.

More fundamentally, combining physics based solvers with DNN to speed up the computations of flows is an active research field [102, 107, 145]. Our model could provide the initial solution to a CFD simulation to potentially speed up the computation of flows that follow the Navier-Stokes equations.

Paper II

Florian Achermann, Thomas Stastny, Bogdan Danciu, Andrey Kolobov, Jen Jen Chung, Roland Siegwart, and Nicholas Lawrance, "WindSeer: Low-altitude real-time volumetric wind prediction over complex terrain aboard a small UAV". In *Science Robotics, under review*, 2022.

Context

Enabling fixed-wing sUAVs to autonomously plan safe and efficient paths requires up-to-date wind information to be available onboard the aircraft. Especially in remote mountainous areas, where having accurate wind predictions is safety critical, high bandwidth communication is likely not available [43] and low bandwidth, high-latency backup links, such as Iridium satellite communication [75], do not allow offboard wind prediction or computing the path. Therefore, we need to run the algorithm onboard to get fast and reliable online wind predictions.

Our work in Paper I showed that DNNs are able to predict simulated wind flows given a known elevation map and the inflow conditions to the prediction domain. While the terrain can be obtained from map services pre-flight [140, 149] or incrementally constructed during the flight [48, 127], accurate inflow conditions are much harder to get a hold of. The large-scale NWP models could serve as sources for prior wind estimates for the DNN. However, we show in Paper II Section 5.2 that these predictions are not necessarily an accurate representation of the measured wind, most probably due to their coarse terrain representation at 1.1 km resolution. On the other hand, sensors to remotely measure the wind, such as LIDARs [150], are simply too heavy (100 kg or more) for use aboard an sUAV. Therefore, the only reliable source available to predict the wind around an sUAV are wind estimates recorded in-situ by the vehicle itself.

Contribution

In this paper we trained a CNN, *WindSeer*, to predict the high-resolution wind and turbulence close to complex terrain solely based on onboard wind measurements and the known elevation map. We leveraged data augmentation methods to improve the quality and size of the synthetic CFD simulated flow dataset to avoid overfitting. We extend the model architecture and training pipeline from Paper I to handle the sparse input. The resulting model successfully demonstrates zero-shot sim-to-real capabilities when evaluated on real wind data without retraining. We used data collected in measurement campaigns at weather stations across different hills in Europe and wind estimates collected by multiple fixed-wing sUAVs. In the latter case we designed and calibrated airflow vanes to allow estimating the three-dimensional wind with sUAVs.

The following list summarizes the technological contributions of this paper:

- Training a CNN with synthetic flows to predict wind around complex terrain based on sparse measurements and an elevation map.
- Custom designed airflow vanes and calibration procedures to enable three-dimensional wind estimation.
- Evaluating the model on real data without retraining to demonstrate zero-shot sim-to-real transfer.

Interrelations

This paper extends the work from Paper I to enable wind and turbulence predictions from sparse onboard UAV measurements. These accurate onboard wind predictions will allow us to apply theoretical time-/energy-optimal planning algorithms [23, 25, 101, 160] to compute the optimal paths in-flight and onboard sUAVs.

More fundamentally, by predicting the dense wind using sparse input measurements, *WindSeer* builds on previous work where DNNs have been trained to predict dense depth [50, 54, 79, 81, 82, 108] or optical flow [162] based on sparse input features. *WindSeer* shows that these approaches can also predict fluid flows with even sparser input levels.

In the context of the overall goal of this thesis to perceive the air for more efficient flight the predicted wind could help to model the thermal updraft column to precisely predict the thermal location at various heights. Such a system could use the heatmap from Paper III and the wind predictions to estimate the shape of the thermal column [16, 74].

2.2 Part B: Thermal Detection

Paper III

Florian Achermann, Andrey Kolobov, Debadeepta Dey, Timo Hinzmann, Jen Jen Chung, Roland Siegwart, and Nicholas Lawrance, “MultiPoint: Cross-spectral registration of thermal and optical aerial imagery”. In *Proceedings of the 2020 Conference on Robot Learning*, 2020.

Context

Thermals are pockets of rising air masses that arise due to parts of the ground surface absorbing more solar radiation than the surrounding areas and heating up the air immediately above [9, 16]. Birds or human glider pilots have learnt to exploit such thermals to extend flight duration and range or to simply conserve energy [7, 19, 158]. Previous research has focused on utilising thermal columns that are found, primarily by chance, during flight. These methods build lift maps based on current and past observations but lack the ability to remotely map and detect thermals to deliberately plan paths targeting areas with prevailing thermals [8, 25, 30, 31, 41, 99]. While thermals themselves are invisible in both optical and TIR spectra, their generating regions on the ground can be sensed by a thermal camera on an sUAV. However, building consistent TIR maps using standard mapping pipelines [127] is thwarted by the poor performance of existing cross-spectral feature detectors and descriptors [6, 14, 28, 78]. To allow for autonomous decision making onboard the sUAV regardless of the communication bandwidth, similar to Paper II, the mapping framework should run in real-time with onboard compute, thus the feature detector must be able to process the incoming image stream in real-time.

Contribution

In this paper we developed a DNN-based multi-spectral keypoint detector and descriptor, *MultiPoint*, for the optical and TIR spectrum. To train *MultiPoint* we compiled a dataset of aligned multi-spectral images. First, the images were captured from a fixed-wing sUAV with two downward-facing cameras (optical and TIR). Small relative transformations between the optical and TIR images in each pair were present due to trigger time offsets and different exposure times. The image pairs were aligned offline with a pipeline optimizing the mutual information (MI) score. *MultiPoint* was trained in three stages: First we evaluated different base detectors used to generate keypoint labels. Then we developed and applied the concept of multi-spectral homographic adaption to generate viewpoint invariant and cross-spectral-consistent keypoint labels. Finally, we trained *MultiPoint* on synthetically warped multi- and same-spectrum image pairs with known transformations on the multi-spectral dataset with the generated keypoint labels. We showed that *MultiPoint* significantly outperforms baseline detectors and descriptors.

The following list summarizes the technological contributions of this paper:

- Composing a dataset of aligned multi-spectral image pairs (optical and TIR).
- Developing a pipeline generating consistent cross-spectral keypoint labels.
- Training and evaluating a DNN-based multi-spectral keypoint detector and descriptor.

Interrelations

The presented feature detector and descriptor, *MultiPoint*, can be utilized in a mapping framework to build a temperature map of the ground. Such a map could be used to directly detect hotter thermal-generating areas or it could be combined with a framework such as that presented by Depenbusch et al. [30] where the location of previously observed thermals is incorporated into a lift map.

Paper IV

Florian Achermann, Julian Haug, Tobias Zumsteg, Andrey Kolobov, Jen Jen Chung, Roland Siegwart, and Nicholas Lawrance, “An Outlook on Single Frame Thermal Column Detection using a CNN”. In *Unpublished*, 2022.

Context

The hot rising air of a thermal can be detected by direct or indirect clues. Indirect clues include detecting temperature variations on the ground (Paper III), observing birds soaring, or seeing specific clouds that typically form above large thermals [139]. The air inside the thermal column differs in temperature and humidity from the surrounding air resulting in a slightly different refractive index [26]. Gradients of refractive index deflect light rays causing causing brightness and color changes, called schlieren, mostly invisible to the human eye [130]. With a specific lab setup consisting of mirrors, lenses, and a point light source, the schlieren can be made visible to the human eye as a shadowgraph measuring the second derivative of the density [129]. BOS methods allow us to visualize the schlieren with a more generic setup but require a known high texture background [13, 109, 134]. First a reference image of the synthetic [109, 134] or natural [13] background is captured in absence of any refractive index gradients. Then a second measurement image is recorded with refractive index gradients, e.g. due to a heat plate or candle in between the camera and the background. The displacement between the reference and background image computed using optical flow-based algorithms reveal the schlieren locations in the image frame. Extensions of BOS also allow us to measure the flow velocities and three-dimensional position [165]. However, these algorithms are unsuitable for deployment on an sUAV. First, computing the dense optical flow in real-time using traditional methods is intractable with onboard compute. Second, the undistorted background is usually not available and aligning two successive frames results in artifacts with higher magnitude than the schlieren flow when computing the optical flows.

Contribution

In this technical brief we investigated if a CNN can predict the schlieren optical flow based on a single greyscale image. We first recorded schlieren flow patterns with a BOS approach in a static controlled indoor environment with a high-texture background. We then developed a pipeline to train the CNN on a mixture of real and synthetic samples. The real samples are images captured in the indoor environment and the synthetic samples are composed of random background images taken from the Places Standard dataset [171] warped with the recorded optical flows. Finally, we evaluate the performance of our CNN-based single image schlieren flow prediction in different indoor and outdoor environments testing the performance and limitations of the approach.

The following list summarizes the technological contributions of this paper:

- Building a dataset of flows in a static indoor BOS setup.
- Training a CNN to predict the optical flow/schlieren using a single greyscale image.
- Evaluating the CNN-based schlieren detection in various settings and environments.

Interrelations

The system presented in this paper is complementary to a model using the temperature on the ground (Paper III) and the wind (Paper I and Paper II) to detect and predict the thermal column. Both models, together with additional modalities, such as detecting the clouds or soaring birds, could be combined into a voting-based system for robust remote thermal column detection.

The simple setup using only one optical camera allows us to efficiently gather data of natural thermals to validate and possibly extend the current thermal models [16, 74].

2.3 List of Publications

The research conducted during this doctoral thesis led/contributed to the following publications, which are listed in chronological order.

2.3.1 Publications included in this thesis

[P1] **Achermann, F.**, Lawrance, N., Ranftl, R., Dosovitskiy, A., Chung, J.J. and Siegwart, R., 2019, May. Learning to predict the wind for safe aerial vehicle planning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 2311-2317.

[P2] **Achermann, F.**, Kolobov, A., Dey, D., Hinzmann, T., Chung, J.J., Siegwart, R. and Lawrance, N., 2021, November. MultiPoint: Cross-spectral registration of thermal and optical aerial imagery. In *Proceedings of the 2020 Conference on Robot Learning*, pp. 1746-1760.

[P3] **Achermann, F.**, Stastny, T., Danciu, B., Kolobov, A., Chung, J.J., Siegwart, R. and Lawrance, N., 2022. WindSeer: Low-altitude real-time volumetric wind prediction over complex terrain aboard a small UAV. In *Science Robotics*. Under review.

2.3.2 Other publications

[P4] Oettershagen, P., Müller, B., **Achermann, F.**, and Siegwart, R., 2019, March. Real-time 3D wind field prediction onboard UAVs for safe flight in complex terrain. In *2019 IEEE Aerospace Conference.*

[P5] Auf der Maur, P., Djambazi, B., Haberthür, Y., Hörmann, P., Kübler, A., Lustenberger, M., Sigrist, S., Vigen, O., Förster, J., **Achermann, F.**, Hampp, E., Katzschmann, R. and Siegwart, R., 2021, April. RoBoa: Construction and evaluation of a steerable vine robot for search and rescue applications. In *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*, pp. 15-20.

[P6] Phillips, T., Stölzle, M., Turricelli, E., **Achermann, F.**, Lawrance, N., Siegwart, R. and Chung, J.J., 2021, May. Learn to Path: Using neural networks to predict Dubins path characteristics for aerial vehicles in wind. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1073-1079.

[P7] Jeger S., Lawrance, N., **Achermann, F.**, Pang, O., Kovac, M., and Siegwart, R., 2022. Reinforcement Learning for Outdoor Balloon Navigation. In *IEEE Robotics & Automation Magazine.* Under review.

2.4 List of Supervised Students

Throughout the author's doctoral studies significant effort was dedicated to supervising students. For projects that resulted in a publication the citation is given.

Master Thesis

Master student, 6 months full time

1. Gregory Duthé (Spring 2019): "Physics-Aware Deep Learning for Wind Prediction over Complex Terrain"

2. Alex Hönger (Spring 2020): "Energy Optimal Planning in Wind for a Fixed-Wing UAV"

3. Bogdan Danciu (Spring 2020): "Incorporating On-line Measurements into Deep Neural Network Predictions for Estimating Wind near Terrain with a Fixed-wing UAV" [3]

4. Stefan Parapid (Fall 2020): "Informative Path Planning for Wind Prediction with a Fixed-Wing UAV"

5. Tobias Zumsteg (Fall 2020): "Thermal Detection using Visual Cameras"

6. Damian Lenherr (Spring 2021): “Avian-Inspired Landing Gear System Design with Walking and Perching Capabilities”
7. Nadja Kämpf (Spring 2021): “Online Local Planning for Obstacle Avoidance with Fixed-Wing UAVs”
8. Simon Jeger (Fall 2021): “Reinforcement Learning for Outdoor Balloon Navigation”
9. William Ericsson (Spring 2022): “Downscaling Weather Prediction using a Neural Network”

Semester Thesis

Master student, 3-4 months part time

10. Eric Sinner, (Fall 2018): “Uncertainty Estimates for a Neural Network Predicting the Wind around Terrain”
11. Jasmin Fischli, (Fall 2018): “Online Neural Network Based System Identification for Fixed-wing UAVs”
12. Felix Graule, (Spring 2019): “Towards Robust Cross-Spectral Optical-Thermal SLAM Onboard a Fixed-wing UAV”
13. Marko Maljkovic, (Spring 2019): “Online Neural Network Based Model Identification of a Fixed-wing UAV”
14. Tamara Farinelli, (Spring 2019): “Neural Networks for Improving Convergence and Solve Time in Computational Fluid Dynamics”
15. Yash Vyas, (Spring 2020): “Safe Path Planning under Wind for UAVs”
16. Simon Jeger, (Spring 2021): “RL-based Navigation for Balloons in Wind”
17. Julian Haug, (Fall 2021): “Single Frame Optical Flow Prediction for Thermal Detection”
18. Wiktor Hoffmann, (Fall 2021): “Informative Path Planning for Wind Prediction with a Fixed-wing UAV”

Bachelor Thesis

Bachelor student, 3-4 months part time

19. Luca Disse (Spring 2018): “System Identification for an Elephant-trunk-like Robotic Arm”

20. Felix Taubner (Fall 2018): “Motion Planning for a Soft, Worm Like Robot”
21. Yves Haberthür and Samuel Sigrist (Spring 2020): “Retraction of an Everting Tube Robot”
22. Julius Fricke (Spring 2021): “ADMM Algorithm Unrolling: Deblurring and Matting”
23. Pelayo Garcia (Spring 2021): “UAV Path Planning through Energy Optimization of Thermal Flows”
24. Jan Schiess (Spring 2022): “Stabilizing Landing Gear for Fixed-wing UAVs”

CSE Seminar in Robotics

Master student, literature review, 3-4 months part time

25. Luzius Brogli (Fall 2018): “Learning for On-line Inference”
26. Mark Frey (Fall 2020): “Thermal Updraft Detection and Prediction using Visual and Thermal Cameras”
27. Wiktor Hoffmann (Spring 2022): “Informative Path Planning”

Perception and Learning for Robotics course

2-3 Master students, 3-4 months part time

28. Trevor Phillips, Maximilian Stölzle, and Erick Turricelli (Spring 2020): “Learn to Path” [106]

Focus Project

6-8 Bachelor students, 1 year project full time

29. Proboscis (Fall 2017-2018): `https://proboscis.ethz.ch/`
30. Roboa (Fall 2019-2020): `https://roboa.ethz.ch/`, Auf der Maur et al. [10]

2.5 List of Open-source Software

This section lists all the open-source frameworks that have been implemented and publicly released over the course of the doctoral studies.

1. *MultiPoint*: A framework to train a neural network based multi-spectral keypoint detector and descriptor [2].
   `https://github.com/ethz-asl/multipoint.git`

2. *WindSeer*: Predicting the wind around complex terrain using a CNN. Generating the data to train the CNN, as well as evaluating the performance on different types of data [1, 3].
   `https://github.com/ethz-asl/WindSeer.git`

3. *FW PX4 Plottools*: Matlab utilities to process and display the content of PX4 ulog files.
   `https://github.com/ethz-asl/fw_px4_plottools.git`

4. *SatComInfrastructure*: Tools for satellite communication with the PX4 autopilot.
   `https://github.com/acfloria/SatComInfrastructure.git`

2.6 Miscellaneous Contributions

Throughout the course of this doctoral thesis the author contributed to developing a small fixed-wing UAV, SenseSoar2 shown in Fig. 2.1, a research platform with a wingspan of 3 m weighing 5.3 kg. With the use of solar cells it achieves travel distances up to 300 km. The author contributed to developing the autopilot firmware, a long range communication system based on satellite and broadband telecommunication links, and extensive field testing. SenseSoar2 was used in the Solar^3 project where the goal was to monitor crop health and status using sUAVs equipped with a hyper-spectral camera[3]. The aircraft was also used in an approved BVLOS flight over Lake Neuchâtel travelling 68 km[4]. To our knowledge this was the first BVLOS flight in Switzerland over such a long distance by a fixed-wing UAV.

---

[3] `https://business.esa.int/projects/solar3`

[4] `https://youtu.be/ks-TiJP3dxs`

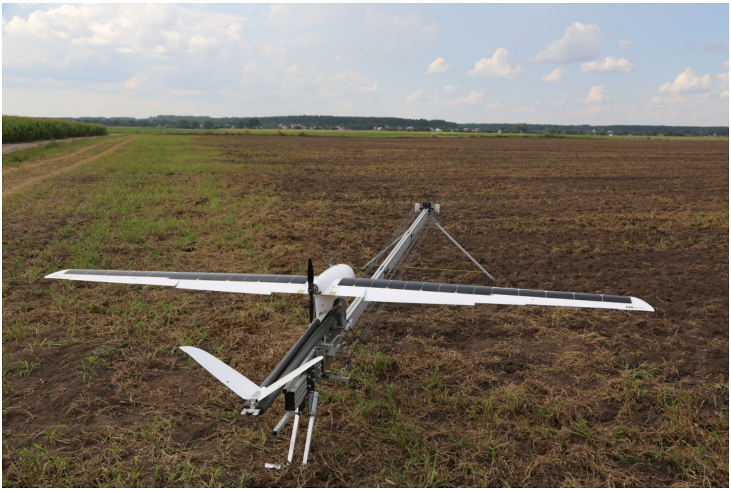


Figure 2.1: The SenseSoar2 sUAV used in the Solar^3 project and the BVLOS flight over Lake Neuchâtel. The catapult was developed to replace hand-launching the aircraft to increase safety during launch.

Chapter 3

Conclusion & Outlook

In this thesis, we developed methods to perceive and predict the invisible air flows to enable safer and more efficient low-altitude sUAV flight. We emphasized the onboard executability by leveraging computationally efficient learning-based models. The core contributions of this thesis are as follows:

Onboard Wind Prediction We developed *WindSeer*, a CNN, predicting the low-altitude wind and TKE around complex terrain. To train the network we generated a dataset of synthetically generated flows over 1.5 km × 1.5 km terrain patches from Switzerland using a RANS CFD solver. We first demonstrated that a CNN is able to predict fluid flows around complex terrain with a known inflow profile and elevation map. However, as the inflow profile is not available in flight we then substituted the inflow profile in the input by sparse onboard wind measurements, accomplishing onboard wind and TKE prediction. The latter network, *WindSeer*, was trained with noisy measurements along simulated piece-wise linear flight paths using only CFD simulated flows. The prediction accuracy of *WindSeer* was evaluated on previously unseen CFD simulated flows. We demonstrated zero-shot sim-to-real transfer by evaluating WindSeer on real wind measurements without retraining. We showed that *WindSeer* is able to predict wind and TKE of different resolutions and spatial extents, up to 30 times higher than the resolution of the training data, on historic data collected in measurement campaigns across different hills in Europe. We also evaluated the prediction performance of *WindSeer* in two different experiments using data collected by multiple fixed-wing sUAVs at three locations in Switzerland. Finally, we demonstrated that *WindSeer* can make wind predictions at around 5 Hz on sUAV grade hardware, such as the Xavier NX.

Multi-spectral Keypoint Detector and Descriptor To enable multi-spectral mapping we developed *MultiPoint* a keypoint detector and descriptor for the optical and TIR spectra. We collected a dataset of optical and TIR image pairs with a fixed-wing UAV of agricultural and lightly forested terrain and aligned the images in an offline MI-based procedure. We extended the concept of homographic adaption to the multi-spectral domain to create viewpoint invariant and multi-spectral consistent keypoint labels. *MultiPoint* was then trained on cross-spectral and same-spectrum image pairs using the generated keypoint labels. We evaluated the performance of different *MultiPoint* variants using alternative network architectures or different base detectors generating the keypoint labels. The best performing *MultiPoint* variant significantly outperforms all baseline methods in all detector and descriptor metrics as well as in estimating the homography between image pairs. *MultiPoint* performs reasonably well on the COCO dataset containing notably different images than the multi-spectral dataset, indicating that the network generalizes to a wide array of textures. Finally, we again demonstrated the real-time capability of *MultiPoint* on sUAV grade hardware.

Single Frame Schlieren Detection We investigated as a proof of concept if a CNN-based approach can detect the schlieren optical flow on a single greyscale image. To train the CNN we created a dataset with a mixture of real and synthetic samples. The real samples were recorded in an indoor environment with optimal conditions to detect the schlieren flow with a BOS algorithm. The synthetic samples are images from the Places Standard dataset warped with the recorded schlieren flows from the indoor environment. We evaluated the CNN-based schlieren detector in a series of indoor and outdoor experiments. In general the schlieren flows are well predicted in the controlled indoor environment and on the synthetically generated samples but currently the CNN struggles to generalize to real images recorded in a generic outdoor environment. Overall, we can state that we successfully proved that a CNN can predict the sub-pixel optical flows of schlieren using a single greyscale image but more work is required to generalize the approach across different outdoor environments.

3.1 Discussion

The work in this thesis focused on predicting complex real-world phenomena for safer and more efficient flights. Special attention was given to creating field-applicable solutions suitable for running onboard sUAVs. This required the development of lightweight algorithms and consideration of the available sensor input information.

Real-time Onboard Execution We achieved real-time onboard execution by deploying DNN models leveraging the GPU of the onboard compute to replace physics-based solvers. In Paper I and Paper II we replaced a physics-based CFD solver with a run-time of multiple hours with CNNs capable of predicting the flows in less than a second. *MultiPoint*, presented in Paper III, outperforms existing keypoint detectors and descriptors, such as SIFT and log-Gabor histogram descriptor (LGHD), not only in the performance metrics but also in the run-time. While the work presented in Paper IV was a proof of concept of detecting schlieren with a single optical image, and developing a light-weight network was not an objective, the CNN still yields lower inference times (1.33 s) than the dense optical flow method used to generate the flow labels (around 10 s to 20 s).

By using DNN-based surrogate models we trade off accuracy and solutions that follow physics first principles for a decrease in computation time. For example, although the flow predictions from *WindSeer* are a good approximation of the CFD-generated flows and direct wind measurements, the predictions are not guaranteed to satisfy basic physical principles such as continuity of mass, momentum and energy. To guarantee such properties we would need a mixture between first principles and DNN-based models most likely losing some computational efficiency in the process.

Deploying a DNN on sUAV grade hardware has been made possible by the recent advances in developing small scale and power efficient GPUs. A few years back the models developed in this thesis would not have run in real time on computer hardware available at that time. This trend of rapidly increasing compute has even continued over the course of this thesis. The next generation of onboard compute, the Jetson Orin NX, is around five times more powerful than the Jetson Xavier NX used in this thesis, with a similar physical and energy footprint, allowing inference with even more powerful DNNs onboard sUAVs.

Data Curation/Augmentation DNNs require a vast amount of data during training to reliably succeed at the task at hand. Consequently, a good deal of the work in this thesis was spent preparing and curating datasets. Carefully selecting the method or underlying model to generate training data can determine if the trained DNN generalizes well to unseen data and learns to predict the target well. Depending on the task the data can be curated from real world data (Paper III), simulation (Paper I and Paper II) or a combination of both (Paper IV). The generated dataset should capture all relevant phenomena or else the model cannot represent these modes. Such an example is presented in Paper II where lee side rotors were observed in the measurement campaign data but the training dataset does not contain such flow characteristics, thus the models fail to predict the flow accurately. Capturing all relevant real-world phenomena is a general limitation of training models with a simulation, especially for complex systems such as turbulent fluid flow.

Applying data augmentation techniques, such as transformations, noise, photometric changes or cropping, can significantly increase the amount of available data to avoid over-fitting to a limited dataset. This is of paramount importance if generating/collecting data is a time intensive task, e.g. CFD simulation in Paper I/Paper II or collecting real-world data as in Paper III.

The model trained in Paper I was trained with limited augmentation methods, only random flipping along the horizontal axis, and performed about an order of magnitude worse on the test set compared to the training dataset. In Paper II we deployed more sophisticated augmentation methods to randomize the location of flow features and flow directions and enhanced the dataset with zero-velocity samples, eventually eliminating the performance gap between the different datasets. *MultiPoint*, presented in Paper III, performed remarkably well on the COCO dataset. This dataset is commonly used to train DNNs for object recognition and contains a wide array of different textures and objects. *MultiPoint* was trained solely on aerial images from an sUAV but multiple homographic and photometric augmentation methods likely prevented overfitting to the limited data.

Input Data Representation The representation of the input data affects how useful it is to the DNN and if the model can be deployed in a real-world environment. For example the CNN in Paper I encoded the wind information as the boundary layer profile. While this representation was potent we do not have access to the true boundary layer profile in a real-world setting. Further developments evaluated the usage of large-scale predictions from NWP simulations as the DNN input. Given correct NWP predictions the DNN predicted the wind with high accuracy. However, flight experiments showed large errors that would be difficult to model, rendering this input definition as invalid. The final input representation for *WindSeer* (Paper II), using the noisy wind estimates along the sUAV flight path, is a trade-off between the available data and prediction performance. Similarly, the terrain for *WindSeer* is represented as a Euclidean distance field in contrast to Paper I where a boolean occupancy grid was used. The Euclidean distance field representation allows for better propagation of the terrain information and even considering terrain features outside of the current prediction domain if they affect the distance field.

Overall with the appropriate representation of the input data we can enable deployment of the model in real-world environments and allow more efficient extraction of the available information.

Modelling Complex Systems The vast expressive power of DNNs allows us to model complex systems, often outperforming more traditional non-learning-based approaches, e.g. transformer models revolutionized natural language processing [161]. *MultiPoint* is no exception, outperforming the non-learning-based baseline detectors and descriptors. Similarly, *WindSeer* predicts a dense wind field based on sparse and noisy measurements, a feat no traditional method has accomplished.

In some cases DNNs can even open up new applications previously not solvable, e.g. StyleGAN allows quickly generating realistic looking fake portraits of people in different styles [58]. The DNN developed in Paper IV managed to detect the subpixel distortions due to refractive index gradients in the air on a single image. Consequently this will enable direct thermal column detection with optical cameras.

Approach to Solving Complex Problems Throughout the thesis our approach to complex problems was always to initially keep it simple and gradually increase the complexity. This helped us to gain an understanding of the underlying challenges and principles early and enabled us to evaluate and compare different approaches. We even used simple toy examples to determine if certain problem statements are solvable in the simplest case and worth spending more effort for the general solution.

One such example is the wind prediction where we first neglected the constraint that the algorithm must only rely on information available onboard and were only interested in whether developing a DNN that predicts the wind is feasible. Only in the second iteration when developing *WindSeer* did we impose the onboard constraint. Another case was the schlieren detection CNN where we initially started with only the real indoor data and in a second iteration added the synthetically generated samples to generalize the CNN across different backgrounds.

Hardware In essence, designing and working with hardware is complex and usually takes longer than initially thought/planned.

Deploying and testing robotics systems and algorithms often requires custom hardware designs and developments. Constructing, assembling, and calibrating such systems is a tedious task often requiring more time than initially planned. One such example was developing and calibrating the airflow vanes where the calibration procedure required much more attention than initially expected. These calibration issues arose when flight testing the setup. Consequently, rigorously testing and evaluating the designs is an important but time-consuming step when developing hardware.

3.2 Future Work

Our vision in this thesis was to enable safer and more efficient BVLOS sUAV flights by predicting the airflow around the aircraft. By predicting the wind using onboard measurements, developing a multi-spectral keypoint descriptor and detector, and a DNN that detects the schlieren on a single image we believe we have made a significant step towards our vision. However, there are still many possible improvements and new research directions to explore. In the following we list our suggestions on how to extend our work.

Wind Prediction Fluid Model To enable *WindSeer* to predict more complex flow phenomena such as mountain waves [24, 33], lee-side rotors or thermal-induced winds [7], these phenomena need to be included in the training data.

By replacing the steady-state RANS with a time-varying model such as large eddy simulation (LES) in the CFD simulations we could better model the effect of turbulence in wind gusts in the input to the DNN or eventually even predict a time-varying solution. The current CFD simulation domain of $1.5\,\mathrm{km} \times 1.5\,\mathrm{km}$ restricts the terrain to mostly one single major geographic feature. Larger simulation domains could allow simulating complex wind phenomena such as lee-side rotors, which arise in the presence of multiple mountains or ridges, thus increasing the *WindSeer* prediction accuracy in mountain ranges. The air in the current data generation pipeline is modelled as an incompressible fluid with uniform temperature distribution. By including temperature differences in the compressible fluid the CFD simulation could model complex flow phenomena such as thermals, updrafts caused by temperature differences on the ground, or mountain waves that are large-scale oscillations of the wind direction and magnitude behind large ridges resulting in periodically strong up- and downdrafts.

***WindSeer* Input Representation** The extended fluid flow model will lead to more complex flows and a wider range of possible solutions. Local wind measurements may no longer contain enough information to uniquely determine the flow condition of the whole domain. Thus, *WindSeer* will probably require additional input information to accurately predict the more complex flows. This can include additional local onboard sensor data, e.g. air temperature, humidity or pressure, or remote measurements, such as a temperature map of the ground generated in flight. In the end it remains to be verified whether the CFD simulation accurately models the complex flow phenomena and that *WindSeer* is able to predict the flow patterns with the available input information.

Accurate Airflow Sensing The necessity of accurately measuring the airflow angles (angle of attack (AoA) and angle of sideslip (AoS)) to estimate the wind was underlined by our flight experiments in the Swiss Alps. Accurately calibrating our wind vanes was a tedious process and is still ongoing. While averaging the estimates over multiple minutes in a cyclic flight path can eliminate such calibration errors this is not feasible when navigating from point to point. Alternative airflow sensing methods, such as multi-hole probes [113], might simplify the calibration procedure but are still subject to aerodynamic interference from the airframe. By carefully analysing the placement of the airflow sensors with CFD to minimize the aerodynamic interference can eventually lead to more accurate airflow angle measurements producing higher quality wind estimates.

TIR Mapping for Thermal Updraft Detection Building temperature maps in real-time onboard an sUAV is now feasible thanks to *MultiPoint*. In a next step we can evaluate if the temperature map contains valuable information for a classifier predicting thermal updraft locations and which features would

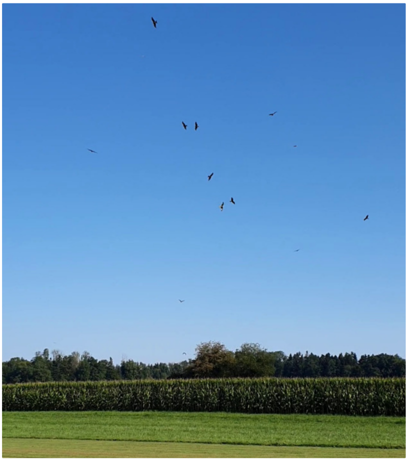


Figure 3.1: Multiple birds soaring in a thermal updraft observed during one of our flight tests.

characterize potential updraft locations, e.g. extent of the hot area or difference to ambient temperature. Depending on the amount of available data the classifier could be learning-based or hand-crafted feature-based.

Bird Tracking for Updraft Detection Birds use orographic lift or thermal updrafts [7, 19, 158] for soaring and conserving energy. Detecting and tracking birds can reveal potential updraft locations as strong thermal updrafts usually attract multiple soaring birds as shown in Fig. 3.1. Tracking a single bird and detecting whether it is soaring or flapping its wings will indicate lift sources, e.g. if the bird is gaining altitude without flapping its wings. Such a system could enable building a lift map from remote observation and complement the other work developed in this thesis.

Multi-modal Thermal Detection Combining the output of the different algorithms for updraft detection and prediction in a voting-based system can increase the robustness of the system and reduce false positives. The different modalities for such a multi-modal detection system could be the wind predicted by *WindSeer*, a temperature map of the ground, bird flight trajectories, cloud detections [139], and sensing the schlieren of the thermal column.

Onboard Reactive Energy-optimal Path Planning The wind prediction now allows deploying time-optimal or energy-optimal planning algorithms to the real world. However, such algorithms must be able to run onboard sUAV grade hardware as the predictions are only available in flight and might rapidly change with new measurements. Computing time-optimal paths is computationally more demanding than computing the pure geometric path [101]. To enable onboard execution, efficient algorithms are required, potentially trading off global optimality for a faster run-time.

Large-scale System Demonstration Finally, deploying an sUAV on large-scale BVLOS missions, dynamically predicting the wind and thermal updraft locations and reactively replanning the flight path would demonstrate the feasibility of such a complex system. Such a flight test would also allow a fair comparison of the energy efficiency to an sUAV simultaneously following a pre-planned path. Real-world flight tests will reveal shortcomings of current sUAV systems, sparking new research directions and ideas, eventually paving the way for regular sUAV BVLOS missions.

Part A

WIND PREDICTION

Paper

Learning to Predict the Wind for Safe Aerial Vehicle Planning

Florian Achermann, Nicholas R.J. Lawrance, René Ranftl, Alexey Dosovitskiy, Jen Jen Chung, and Roland Siegwart

Abstract

Obtaining an accurate estimate of the local wind remains a significant challenge for small uncrewed aerial vehicles (UAVs). Small UAVs often operate at low altitudes near terrain, where the wind environment can be more complex than at higher altitudes. Combined with their relatively low mass, this makes small UAVs particularly susceptible to wind. In this paper we present an approach for predicting high-resolution wind fields based on a terrain elevation model and known inflow conditions. Our approach uses a deep convolutional neural network (CNN) to generate 3D wind estimates. We show that our approach produces wind estimates with lower prediction error than existing methods, and that inference can be performed on an on-board computer in less than two seconds. By providing the wind estimate to a sampling-based planner we show that the improved estimates allow the planner to generate safer paths in strong wind scenarios than with alternative wind estimation techniques.

Published in:
IEEE International Conference on Robotics and Automation (ICRA), 2019
DOI: 10.1109/ICRA.2019.8793547

1 Introduction

Small uncrewed aerial vehicles (UAVs) provide the capacity to conduct inspection and monitoring tasks in challenging environments. In particular, fixed-wing UAVs enable long endurance flights (compared to multi-rotor UAVs) and can therefore be used in areas that are inaccessible to humans and potentially beyond visual line of sight. Under such circumstances, robust platform autonomy is key to mission success, and this relies on a combination of situational awareness and fast reactive (if not pre-emptive) planning. Further, the robust autonomy required for long-duration flights must also be executed on-board the aircraft, since external communication can be unreliable and have limited bandwidth, and the aircraft must be able to react to unforeseen environmental changes.

Unfortunately, small UAVs are particularly susceptible to wind because naturally-occurring wind speeds can often approach or exceed the airspeed of the UAV. For example, in mountainous areas, the yearly average wind speed can exceed $10\,\mathrm{m\,s^{-1}}$ [125], a speed similar to the normal cruise speed of a small UAV [138]. Further, small aircraft have correspondingly low rotational inertia properties, so roll or pitch moments caused by spatial variations in wind at a similar scale to the aircraft size can result in significant rotational moments. These conditions mean that small UAVs often cannot be flown safely in high wind speeds or turbulent wind conditions without risking the safety of the aircraft [98].

Currently, one of the primary sources of data for wind predictions used in UAV flight planning is from large-scale numerical weather prediction (NWP) models. These models generally incorporate observations made using satellite and ground-based sensors with forward simulations based on flow transport equations (such as Navier–Stokes) to solve for flow estimates on a discretized grid [92]. The smaller scale effects of terrain and local thermal variations are often neglected or reduced because the simulation resolution is generally of order 1 km or larger. This limitation often means that NWP predictions are not sufficiently accurate or reliable for use in small UAV operations [98].

Estimating the wind at a resolution relevant to UAV flight planning remains a challenging task, especially because true wind is a spatially and temporally varying three-dimensional vector field. The processes that drive variations in wind range from large-scale atmospheric forces such as pressure differences and Coriolis forces down to smaller-scale influences such as thermal differences and terrain. Furthermore, air is transparent to many remote sensing modalities, while in-situ measurements provide limited value since they often arrive too late for the platform to react. Previous work has shown that (in simulation), if a small UAV could accurately predict the wind, it would allow planning methods that avoid dangerous flows or even make use of favourable wind conditions to improve endurance [23, 25, 160].

In this paper, we present a learning-based method for predicting the steady (time-averaged) wind flow at meter-scale resolutions from terrain elevation data and upstream wind conditions suitable for inference on an on-board CPU. To generate sufficient training data, we leverage existing (and computationally expensive) Reynolds-averaged Navier–Stokes (RANS) computational fluid dynamics (CFD)

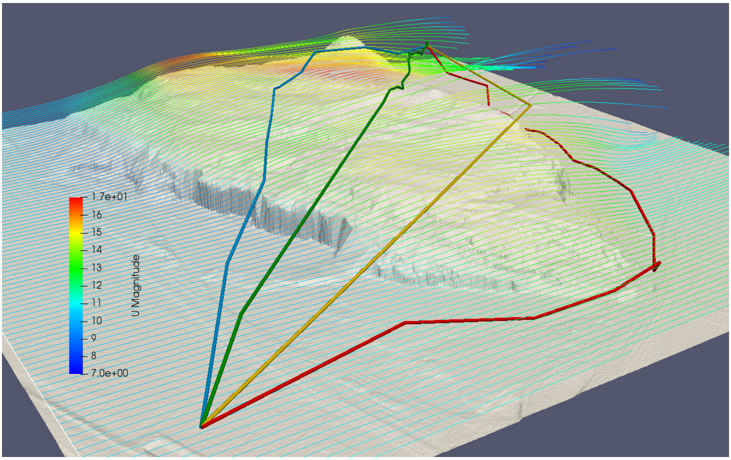


Figure 4.1: Paths generated by a sampling-based planner to minimize air-relative distance using wind fields estimated by different approaches. Our wind prediction approach allows the planner to find a safe trajectory to the goal (red path), while baseline methods (green, yellow, blue) lead to generating infeasible paths that would cause the UAV to crash due to its airspeed limitations. This highlights the value of accurate wind prediction for autonomous UAV navigation. Green, yellow, and blue paths were obtained by predicting zero wind everywhere, replicating inflow conditions, and linearly interpolating between the vertical edges of the volume, respectively.

estimation techniques to solve for the wind field given any terrain model and inflow condition. Our generated wind flow data, along with the terrain and inflow data, are then used to train a deep convolutional neural network (CNN), which can then be queried online by an on-board flight controller to generate flight plans that avoid dangerous wind speeds and exploit the flow to improve flight efficiency.

Validation studies on the accuracy of our CNN wind estimates demonstrate a mean absolute prediction error of $0.44\,\mathrm{m\,s^{-1}}$ on unobserved terrains and inflows. Furthermore, this error drops to $0.33\,\mathrm{m\,s^{-1}}$ when only considering winds within our targeted flight altitudes of 30 m to 500 m. We also tested the efficacy of our wind prediction in a flight planning pipeline. We use a sampling-based planner with a cost function based on air relative distance (equivalent to travel time) to plan paths using the predicted wind field (Fig. 4.1). Our results show that the planner generates feasible (safe) paths with a higher likelihood and achieves a more accurate cost estimate when using our wind predictions as compared to using existing wind prediction techniques.

2 Related Work

2.1 Computationally-efficient fluid flow estimation

A range of applications require timely or low computational-cost estimates of fluid flows. One such application is computer graphics [38, 164]. The goal here is typically to calculate the temporal variation of a fluid (including liquids and gases) in response to disturbances in a way that results in stable and visually-realistic solutions. This can be achieved with particle-based methods that simulate larger scale flow by modeling individual particles and solving pressure and flow based on the Navier–Stokes equations across finite volumes in the region of interest. Such methods can solve and render flows with thousands of particles in real time on a standard desktop computer [91]. Recently, traditional particle-based fluid simulation approaches are being combined with data-driven methods to further speed up the simulation. In this line of work, both random forests [68] and deep networks [60, 157] have been used to increase the speed of fluid solutions by multiple orders of magnitude. However, the particle-based methods used in graphics are typically optimized for visual appearance, not physical accuracy, and therefore cannot readily be used for UAV planning applications. Moreover, most particle-based methods assume a closed volume and often a constant number of particles, rather than the flow-through type cases we expect in terrain wind flow.

Steady (time-averaged) flow is commonly used for design and analysis, such as estimating drag for automobile and aircraft design. Umetani and Bickel [148] demonstrated an approach to estimate drag, flow velocities, and pressure around 3D bodies using a Gaussian process based on a novel parameterized shape representation. Baqué et al. [12] use a similar shape parameterization approach combined with a geodesic CNN for shape optimization in aerodynamics applications. Of most relevance to the current paper is the work by Guo et al. [42], who use a CNN to predict time-averaged laminar fluid flow around shapes to speed up industrial aerodynamics applications. They use Lattice Boltzmann Methods to generate training data on two-dimensional shapes of medium complexity such as cars, as well as low-resolution three-dimensional geometric primitives. In contrast, we use RANS CFD solutions on complex and diverse 3D terrain geometries to generate training data, as RANS solutions have demonstrated high-quality predictive performance compared to other methods for complex terrain flow models [15]. We validate our CFD setup against a microscale flow model benchmark [15, 17] to ensure that the generated training data is realistic. Additionally, we propose a CNN architecture that is conditioned on inflow conditions and is capable of predicting high resolution wind fields (64^3) that are required for accurate planning.

2.2 Wind estimation

A number of research and industrial applications require wind estimates at lateral resolutions higher than 1.1 km, which is the resolution typically provided by numerical weather prediction [11]. One such application is wind turbine ‘micro-siting’ –

identifying and evaluating installation sites based on power potential and safety. CFD remains one of the most popular techniques for turbine siting applications, including both steady (time-averaged) and transient methods such as large eddy simulation (LES) [146]. Modelers also use wind tunnels and simpler numerical methods to analyze energy potential and characterize turbulence [103]. We drew on publications from turbine siting to inform the CFD methods used in this paper, and we view this as a potential application area for the work presented here.

Industrial aerodynamics, particularly for analysis of wind around built environments, is also relevant in terms of desired quantities and scale. Traditional methods used databases and parametric models [156], but CFD has increased in popularity with increased computational capacity [144]. We selected RANS CFD methods for this project due to their maturity in commercial and open source CFD solvers, proven capabilities and low computational cost relative to LES methods.

2.3 UAV planning in wind

There is also a body of work that deals with gathering in-situ measurements of the wind using UAVs. Applications include monitoring tornado genesis [35, 36, 40, 118] and cloud evolution [115, 116], as well as autonomous soaring for long endurance flight [41, 71, 74]. These works tend to focus on the information gathering aspect of the problem since precise wind sampling is constrained along the UAV trajectory. However, in many cases, data from external sensors such as radar [36] or weather balloons [72] are also incorporated into the wind field estimate to enable informative planning.

In our work, we develop a wind prediction pipeline that does not rely on external sensor data and can be executed entirely on-board the UAV. Our system only requires knowledge of the terrain beneath the flight operation region and any coarse weather (wind speed) predictions provided for example by NWP.

3 CFD Wind Data

In this work, we are interested in predicting the flow over natural terrain. In particular, we approach this as a machine learning problem of training a model offline using representative data and performing online inference during flight. To obtain suitable training data, we used a CFD solver to generate a large dataset of fluid flow solutions over a set of real terrain samples.

3.1 Terrain models

The terrains used to generate our training data were collected from the Swiss geodata service, which provides access for Swiss researchers to high-resolution (2 m lateral, 0.5 m vertical) elevation data across all of Switzerland[1]. We manually sampled 370 patches of terrain, each measuring approximately 1.2 km square. We

---

[1] geodata4edu.ch, Geodata © swisstopo

favoured selecting terrain patches that contained one side with near-constant elevation, as this allows us to simulate a formed boundary layer flowing into the region from that edge. Each terrain sample was collected as a geographically aligned GEOTIFF elevation file, which was then converted to an STL representation. An example of a terrain patch along with the corresponding CFD flow solution (represented by the streamlines colored by velocity magnitude) is illustrated in Fig. 4.1.

3.2 CFD solutions

We are primarily interested in time-averaged estimates as predicting the dynamic wind needs good knowledge of the initial conditions, which is not available onboard on the plane. We elected to use a RANS solver, namely the popular $k-\epsilon$ two-equation turbulence closure [73]. To solve the flow, we used the open source solver OpenFOAM [55]. We created an automated pipeline that ingests terrain patches as STL files and outputs flow solutions over the terrain.

For each terrain, we use the OpenFOAM SnappyHexMesh utility to generate a mesh around the terrain and solve using the steady simpleFoam solver. Wind enters through one face of the domain, perpendicular to the face. The input wind speed U, turbulent kinetic energy k and turbulence dissipation rate ϵ across the inflow face vary with height z, defined using a standard logarithmic boundary layer profile:

$$U = \frac{U^*}{\kappa} \log\left(\frac{z - z_0}{z_0}\right), \quad k = \frac{(U^*)^2}{C_\mu^{1/2}}, \quad \epsilon = \frac{(U^*)^3}{\kappa\,(z - z_0)},$$

where the friction velocity U^* is

$$U^* = \kappa U_{ref} \left[\log\left(\frac{(Z_{ref} + z_0)}{z_0}\right)\right]^{-1}. \tag{4.1}$$

We use standard values for flow constants ($\kappa = 0.4$, $C_\mu = 0.09$), and a fixed reference height of $Z_{ref} = 10$ m. All simulations use a constant surface roughness height z_0 of 0.01 m, corresponding to grass with few trees [44]. Future work will explore developing more complex estimates of surface roughness by inferring from geolocated aerial imagery.

For each terrain, we solved with reference speeds U_{ref} from $1\,\mathrm{m\,s^{-1}}$ to $15\,\mathrm{m\,s^{-1}}$ in $1\,\mathrm{m\,s^{-1}}$ increments. Although we simulated 15 wind speeds for each terrain, not all simulations converged and we discarded solutions that did not reach our specified solution tolerance. On average, each solution took approximately 4 hours to converge on a single processor. Solutions were resampled from the irregular CFD mesh onto a regular 64^3 grid after convergence. In total, we generated 3318 converged CFD solutions from 370 terrain patches.

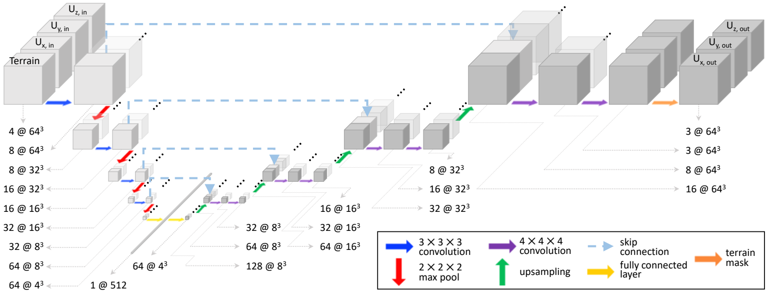


Figure 4.2: CNN architecture. We use a 3D encoder-decoder with skip-connections.

4 Wind Prediction

Given the training data provided by CFD simulation, we now develop a model that can approximate these in a computationally efficient manner. We chose to use a CNN as a function approximator, since CNNs are highly expressive while allowing for efficient inference even on mobile platforms.

For our model, the input consists of four volumetric channels, each with a spatial resolution of 64^3. The input channels are the inflow conditions ($U_{x,in}$, $U_{y,in}$, $U_{z,in}$) and a terrain model T represented as a binary occupancy grid. The inflow conditions are only specified at the input face, but to maintain consistent volumes we replicate the same wind across the entire volumetric input domain before feeding it to the network. The network outputs three channels of dimension 64^3 representing the predicted velocity ($U_{x,out}$, $U_{y,out}$, $U_{z,out}$).

To increase the size of the training dataset, we augmented the data by rotating each CFD solution around the vertical axis in the cardinal directions, and flipping the lateral dimensions. This resulted in an eight-fold increase in the number of training samples to 26,544. We partitioned the resulting dataset into three parts: 19,728 samples from 290 terrains for training, 5,120 samples (64 terrains) for validation, and 1,704 samples (16 terrains) for testing. The data was split across terrains, meaning no terrain that was seen during training is in the validation or test set.

4.1 Network architecture

We use an encoder-decoder CNN based on U-net [119]. Our final network architecture is illustrated in Fig. 4.2, and contains 5.14×10^6 trainable parameters. We store all floating points numbers in single precision (32-bit), and a single forward-pass inference requires 1.07×10^{10} floating point operations. The encoder consists of a sequence of 3D convolutional and max-pooling layers, followed by a fully connected layer. The decoder applies upsampling and 3D convolutional layers to generate the three output channels at the original resolution. We found that upsampling with nearest-neighbor interpolation combined with stride-1 convolution resulted in smoother outputs with lower error than standard transpose convolution layers, which tended to introduce artifacts in the output channels [96]. The network utilizes skip connections to preserve high-resolution feature information from the encoder to aid in the decoding to higher resolution. On a single CPU core one forward-pass inference requires 2.5 GB RAM and is finished on average in 1.6 s. We experimented with training the network both with L^1 and mean squared error (MSE) loss functions, and found that extrema values were better predicted when training with MSE loss than with L^1 loss. Such extrema often occur close to ridges or steep slopes and can be critical for safe navigation.

Removing the skip connections or the fully connected layers resulted in increases in MSE loss on the test set of 37 % and 25 % respectively. We also explored alternative minimum code sizes in the fully connected layer, and found that 512 resulted in the best performance compared to 256 (4.6 % MSE test loss), 1024 (2.4 %), 2048 (10.0 %) or 4096 (12.6 %). Using trilinear interpolation instead of nearest neighbor reduced visual artifacts in the output, but increased error slightly (6.9 %).

5 Experiments

We now evaluate the proposed approach experimentally. We start by validating the CFD pipeline that was used to generate our training data. We then evaluate the accuracy of the network predictions. Finally, we demonstrate the value of the predicted wind fields for planning safe UAV flight paths.

5.1 Verifying the CFD solution

To verify that our CFD pipeline generates realistic flow estimates, we compared predictions from our CFD pipeline against a microscale flow model verification benchmark. The verification tests described in [15, 17] provide benchmarks of prediction quality from a range of microscale prediction models, including multiple CFD solver schemes, against in-situ measurements.

To validate our CFD setup, we used our CFD pipeline to generate a flow solution on the Bolund hill case and then evaluated the result against the published data. We attempted to keep the solver setup as similar as possible to that used to generate the training dataset, however, we had to make minor adjustments to meet the specifications of the benchmark: we changed the domain shape to be 700 × 500 m, rather than the 1.2 km^2 patches used in our pipeline, and we applied the specified surface roughness on the hill and surrounding areas.

We ran one inflow case (case 2 out of 4 in [17]) with incoming wind at bearing 270°, as this is the same setup as our pipeline (single incoming wind face). Using the wind speed-up error defined in Eq. 15 of [15], our method showed a mean absolute error of 10.3 %. The mean error across all models (and all cases) in [15] was 15.8 %

and the mean for all RANS 2-equation models was 13.6 %. The best performing model (a RANS model) had a mean absolute speed-up error of 10.2 %. Thus, our approach is competitive with current methods, validating our CFD setup.

5.2 Wind flow prediction

We compare the accuracy of the network predictions against three baselines. The first is the trivial approach of predicting zero wind everywhere. The second baseline assumes known inflow conditions and replicates them across the entire domain. This is essentially the same as the input to our wind-predicting CNN. The final baseline simulates using an accurate meteorological-scale wind model. In Switzerland, the high-resolution NWP model has a lateral resolution of 1.2 km [11]. We simulate having perfect (no-noise) observations of each vertical edge of a cube enclosing a terrain patch (with 1.2 km side length) by collecting the ground truth wind values from the CFD solution, and then using a trilinear interpolation to estimate the wind inside the region

Table 4.1 shows prediction error metrics for each method, averaged across the whole domain (excluding points inside terrain) and across all samples in the test set. We report MSE and mean absolute error for the U_x and U_z components of the flow, as well as the complete flow vector $\mathbf{U}$. The U_x component represents lateral flow accuracy since the error in the U_y component is almost identical to U_x due to the terrain rotation used in data augmentation.

The quantitative evaluation shows that the network is able to predict the wind with an average absolute error of less than $0.5\,\mathrm{m\,s^{-1}}$ for the full wind vector. Both the interpolation and the inflow replication baselines also perform surprisingly well. However, their absolute errors exceed the error of the network prediction by a factor of 1.7 and 2.2, respectively. The relatively good performance of these simple baselines can be attributed to the fact that the flow far away from the terrain is generally smooth and easy to predict, with low amounts of vertical or cross-wind flow.

For our application, the UAV would generally maintain an altitude of at least 30 m above the terrain, thus the prediction accuracy of regions very close to the terrain is not as critical. To highlight the predictive performance of the models in the flight region, we analyze the estimation error for altitudes between 30 m to

Table 4.1: Wind prediction error

	MSE loss $(m/s)^2$			Mean abs. error (m/s)		
Method	U_x	U_z	$\mathbf{U}$	U_x	U_z	$\mathbf{U}$
Zero wind	2.392	0.070	1.617	6.008	0.312	11.872
Inflow copy	0.115	0.070	0.100	0.490	0.312	0.962
Interpolation	0.059	0.040	0.061	0.372	0.281	0.743
Network (ours)	0.012	0.017	0.014	0.238	0.152	0.436

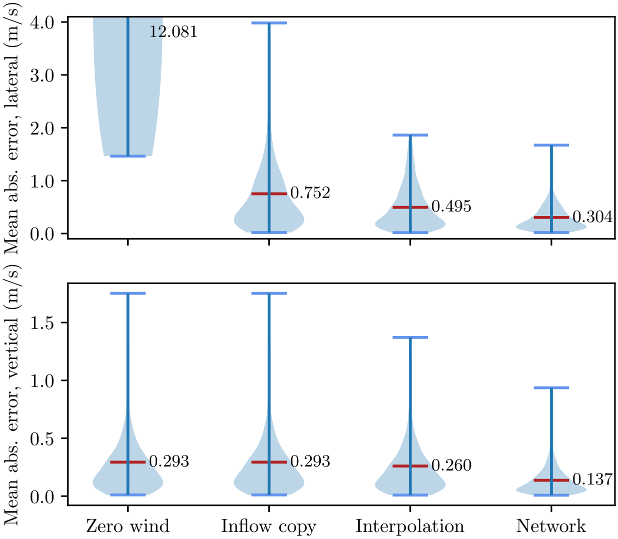


Figure 4.3: Distribution of mean absolute prediction error for target altitudes (between 30 and 500m above terrain). We show error for the lateral flow (top) and vertical flow (bottom). Labeled red bars indicate mean values.

500 m above the terrain. Figure 4.3 illustrates the distributions of mean absolute error for lateral and vertical flow in this target altitude range. The network performs well on lateral predictions, with a mean error of $0.3\,\mathrm{m\,s^{-1}}$, outperforming the closest baseline by a factor of 1.6. The advantage of learning is highlighted even more significantly in the vertical errors, where interpolation does not produce significantly better results than either zero wind or copying the inflow (which both estimate no vertical flow). Here the network outperforms the best baseline by a factor of 1.9. As shown by the distribution plots, the improvement is not only in the average error, but also in the maximal error, which can be particularly important for safe path planning.

To understand where the network has the poorest prediction performance, we visualize the magnitude of the estimation error as a volumetric plot for a terrain sample from the test set in Figure 4.4. This example illustrates one of the poorer performing predictions, and shows that the network prediction is worst in the regions behind prominent terrain features. These areas also contain the most complex flow, where the flow detaches and can form recirculation zones. These regions are also difficult to properly resolve for CFD solvers, as they usually contain

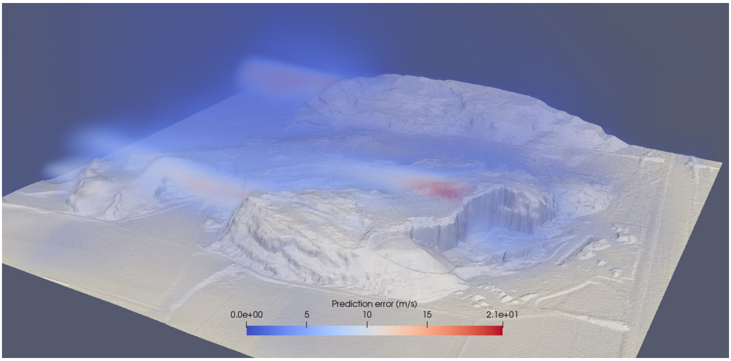


Figure 4.4: Prediction error between CFD solution and network prediction for a member of the test set. Note that flow enters from the right. Areas behind complex geometry have the highest magnitude errors.

a high amount of turbulent flow. We think that increasing the number and variety of training terrain samples may help with improving generalization across terrain variations. In future work we will be exploring how to improve the prediction performance in these regions and developing uncertainty measures, potentially based on learning from the turbulence estimate outputs of the CFD solver.

Figure 4.5 compares the qualitative performance of different methods, visualized as two-dimensional slices of the predicted three-dimensional volumes. While lateral flow is qualitatively similar for the network prediction and the interpolated result, the vertical flow is predicted much better by the network, even in the challenging case shown at the bottom.

5.3 Results: UAV path planning

We demonstrate that wind estimates from our network allow a basic planner to more reliably generate feasible plans. We use RRT* [57], a probabilistically complete and asymptotically optimal sampling-based planner. We selected RRT* because it provides anytime behaviour (monotonically improving path cost with execution time) after finding an initial solution. The planner searches for a valid path between specified start and goal positions that minimizes total cost along the path and avoids collision with the terrain.

We define the cost function as the air-relative distance based on the wind estimate which is also a proxy for travel time with a constant airspeed:

$$C = \int \frac{1}{\|\vec{a} + \vec{w}\|_2} ds, \tag{4.2}$$

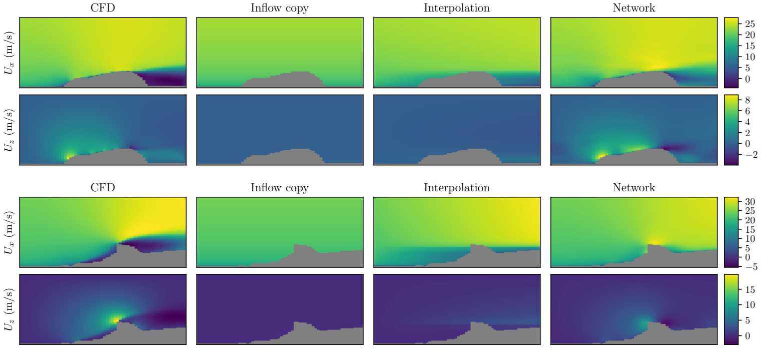


Figure 4.5: Qualitative results of wind prediction. All predictions are three-dimensional, but we show two-dimensional vertical ($x - z$) slices for visualization purposes. Flow enters from the left. We show predictions of the network, two baselines, and the CFD ground truth for both lateral and vertical flow. **Top:** A typical case. Both the lateral and the vertical flow are predicted well by the network. **Bottom:** A challenging case. Geographic features of this type were not represented well in the training set, and the region behind the sheer bluff is not as well modelled.

where $\vec{a}$ and $\vec{w}$ are the airspeed and wind vectors respectively.

In our model, the aircraft airspeed is set at $15\,\mathrm{m\,s^{-1}}$, and we impose vertical rate limits of $3.8\,\mathrm{m\,s^{-1}}$. We provide the planner with the wind estimates from each prediction method, the terrain model, and start and goal locations. The planner then returns the lowest-cost path found within a fixed time budget (20 seconds). We evaluate the true cost of each path using the same planner with the wind field generated by the CFD. We do not quantify path risk (the likelihood of collision with the terrain), but rather classify a path as invalid if any segment cannot be completed, either because the headwind is higher than the airspeed, or because the vertical speed is higher than the vertical rate limit.

We evaluate the performance for five different cases with varying terrains and wind speeds. For each of these cases a feasible path exists. Table 4.2 summarizes the results over ten runs of each scenario and setting. Planned cost is the cost estimated by the planner using the predicted wind, averaged over feasible plans. True cost is the average cost of the feasible planned paths evaluated with the true (CFD) wind field. Over two out of five cases, using the network prediction for planning significantly increased the chance to find a feasible path compared to the best baseline (0 % to 80 % and 10 % to 50 %). However, in some configurations, such as in Case 3, the network prediction is not good enough to result in feasible paths. When finding a feasible path, the network wind prediction typically provided a better estimate for the true cost of the planned path and a lower overall true cost than the baselines.

Table 4.2: Wind-aware path planning performance

Case	Prediction method	Valid	Planned cost [s]	True cost [s]
1	Network	8/10	329.6	386.0
	Interpolation	0/10	-	-
	Inflow	0/10	-	-
	Zero Wind	0/10	-	-
2	Network	5/10	299.2	478.6
	Interpolation	1/10	336.5	488.2
	Inflow	1/10	356.8	495.9
	Zero Wind	0/10	-	-
3	Network	0/10	-	-
	Interpolation	0/10	-	-
	Inflow	0/10	-	-
	Zero Wind	0/10	-	-
4	Network	10/10	38.2	38.5
	Interpolation	10/10	39.3	38.2
	Inflow	10/10	39.2	38.0
	Zero Wind	10/10	68.3	37.7
5	Network	10/10	213.9	213.4
	Interpolation	10/10	196.2	247.0
	Inflow	10/10	197.5	234.3
	Zero Wind	10/10	64.2	242.2

6 Conclusions

We have proposed a deep-learning-based approach for meter-scale wind prediction from a terrain profile. The approach can operate on a single CPU core, and therefore could be deployed on-board a UAV. To showcase the use of the predicted wind fields for UAV flight control, we integrated the wind predictions with a trajectory planner and demonstrated that safety of the planned trajectories can be significantly increased by taking the predicted wind into account. This work could be extended in multiple ways. In reality, wind fields vary over time, and future work will consider temporal variation (transient flow) and local turbulence in the wind field. Online wind measurements that are performed on-board the aircraft could be incorporated in the wind prediction pipeline to further increase its accuracy. Furthermore, a terrain model, incorporating both geometry and terrain roughness, could be extracted online from the visual stream recorded by the UAV. Finally, alternative network structures may improve prediction performance and will be explored in future work.

Paper II

WindSeer: Low-altitude real-time volumetric wind prediction over complex terrain aboard a small UAV

Florian Achermann, Thomas Stastny, Bogdan Danciu, Andrey Kolobov,
Jen Jen Chung, Roland Siegwart, and Nicholas Lawrance

Abstract

Birds and human pilots of unpowered aircraft are keenly aware of air movement patterns near terrain. This allows them to predict wind in real-time at locations hundreds of meters away and rely on these predictions to travel much farther than they otherwise could. Modern small uncrewed aerial vehicles (sUAVs) could similarly benefit from such wind information to increase flight duration and safety, but have so far lacked the necessary predictive capabilities. Indeed, existing weather models are valid only for much higher spatial, temporal and altitude scales than those at which sUAVs operate, and are far too computationally expensive to run aboard such small craft. Our work for the first time equips sUAVs with the ability to predict low-altitude wind at remote locations in real-time directly onboard the aircraft. We train our convolutional neural network (CNN), *WindSeer*, using only *synthetic* data from computational fluid dynamics (CFD) simulations and show that it can successfully predict *real* wind fields over terrain with known topography from just a few noisy spatially clustered wind measurements along an sUAV's flight path that cover less than 0.19 % of the predicted wind field volume. The CNN architecture allows for wind prediction at different resolutions and domain sizes without the need for retraining. We demonstrate that the model successfully recovers historical wind data collected by weather stations at several hills across Europe. Last but not least, we fly a series of multi-sUAV missions that validate the model's ability to make accurate predictions in real-time with limited onboard compute.

Published in:
Science Robotics, under review, 2022

1 Introduction

Birds and human pilots of light unpowered aircraft such as sailplanes and paragliders rely on their extensive familiarity with atmospheric phenomena near terrain to accurately *guess* invisible air movement – wind – taking place hundreds of meters away. They successfully use these predictions to extend their flight duration [7, 19, 158]. AI-controlled small uncrewed aerial vehicles (sUAVs) can also exploit vertical updrafts [41, 70, 99, 114]. However, since onboard sensors are able to detect the wind only at the aircraft's location, sUAVs still rely on chance to stumble upon these moving air regions in the first place, limiting their ability to plan energy-efficient routes. While remote 3D wind sensing systems such as multi-Doppler LiDAR [150] do exist, they require extensive setup and are limited to ground-based application due to their large weight of around 180 kg per sensor.

Apart from extending flight duration, wind and turbulence have a significant impact on the safety of sUAVs due to their low airspeeds, with their cruise speeds as low as $8.3\,\mathrm{m\,s^{-1}}$[98]). Turbulence characterizes rapid and chaotic changes in the wind velocity field and can be a result of sharp terrain changes. Regions with high turbulence levels tend to feature strong wind gusts and large wind changes over time, thus posing a safety risk for any sUAV. Complex terrain can even cause wind speeds exceeding the performance limits of sUAVs, which, together with turbulence, results in poor tracking of the planned flight path [137]. Climb rates of fixed-wing aircraft are even more limited, at around $2.5\,\mathrm{m\,s^{-1}}$ for a typical sUAV [77, 136]. Therefore, having accurate predictions of the vertical up- and downdrafts is often more safety critical compared to knowing the horizontal wind. If the dense wind field is known in advance, wind effects can be properly incorporated into flight planning to avoid flying through unfavorable winds or regions of high turbulence [23].

Numerical weather prediction (NWP) can accurately model relatively large-scale wind patterns with grid resolutions on the order of kilometers [154]. However, chaotic fluid-dynamic effects due to local steep terrain [22] cause wind at smaller spatial scales and low altitudes, where sUAVs typically operate, to differ significantly from these predictions. Computational fluid dynamics (CFD) simulations can generate high-resolution wind flows around terrain at smaller scales, on the order of meters or less, but are much too expensive to compute in real-time aboard an sUAV and require well-defined boundary conditions reflecting the overall weather situation [15, 17].

The representational power afforded by neural networks combined with their fast querying capabilities has elevated them to the function approximator of choice for modeling complex data in fields ranging from computer vision [88, 135] to weather prediction [155]. Indeed, neural networks are able to render stable and visually realistic solutions of fluid flows and interactions between different fluids over time [38, 61, 164]. Aerodynamic shape optimizations for cars or airfoils have been sped up by replacing the CFD simulation to compute the steady time-averaged flow with a neural network [12, 18, 117, 148]. Prior work [1] demonstrated the possibilities of using a known wind profile as inputs to a neural network flow model. Of course, in each of these examples the network receives as input either a complete inflow profile or initial flow state. For our task of modeling the wind, this would be equivalent to knowing the wind along the boundary of the region for which the predictions will be made. Such dense wind data is simply not available in real-time, nor at meter-scale. Data from NWP is an accessible source of wind information that could serve as input to the neural networks. However, flight data highlights a distinct mismatch between the kilometer-scale NWP and the measurements obtained during flight, suggesting that these inputs are unsuitable for predicting higher resolution flow behavior (Section 5.2), ultimately motivating the need for a prediction model reliant only on sparse wind measurements that can be feasibly obtained online.

In this work, we enable sUAVs to model the surrounding dense wind field and turbulence onboard and in real-time by replacing the computationally expensive CFD simulation with a compact, highly efficient convolutional neural network (CNN). Our complete wind prediction pipeline is illustrated in Fig. 5.1. *WindSeer*, the encoder-decoder CNN we propose, is, in essence, a neural simulator. It is pre-trained *offline* using 3D terrain maps, which are readily available from online services [140, 149], and *synthetic* data generated by CFD simulations. The CFD simulations are queried along simulated trajectories and their measurements are augmented with Gaussian noise and random biases, emulating the observations made by sUAVs during flight. In addition to the time-averaged wind speeds, the Reynolds-averaged Navier–Stokes (RANS) CFD simulations also compute the turbulence kinetic energy (TKE) — a metric for the strength of the turbulent velocity fluctuations in the wind field that is proportional to the sum of the variances in each dimension. Regions of high turbulence present a safety risk for sUAVs; accurate prediction of TKE would allow the aircraft to avoid them.

WindSeer is used *online* (via zero-shot transfer) aboard an sUAV to make accurate dense predictions of the *real* wind field based on local noisy wind measurements along the sUAV flight path obtained with standard sUAV onboard sensors. *WindSeer* accurately predicts important wind properties (vertical wind and TKE) hundreds of meters away from an sUAV, where they are not observable with a standard fixed-wing sUAV sensor set — airspeed, inertial measurement unit (IMU), and global navigation satellite system (GNSS). Moreover, the fully convolutional design of *WindSeer* can accept any input dimension (above the minimum size of 64^3 cells), and its predictions are location equivariant. Our results show that this allows us to use the invariance of wind properties around terrain across a relatively wide range of scales to make accurate wind and turbulence predictions at different spatial resolutions without retraining the model, thereby increasing the safety and efficiency of sUAVs.

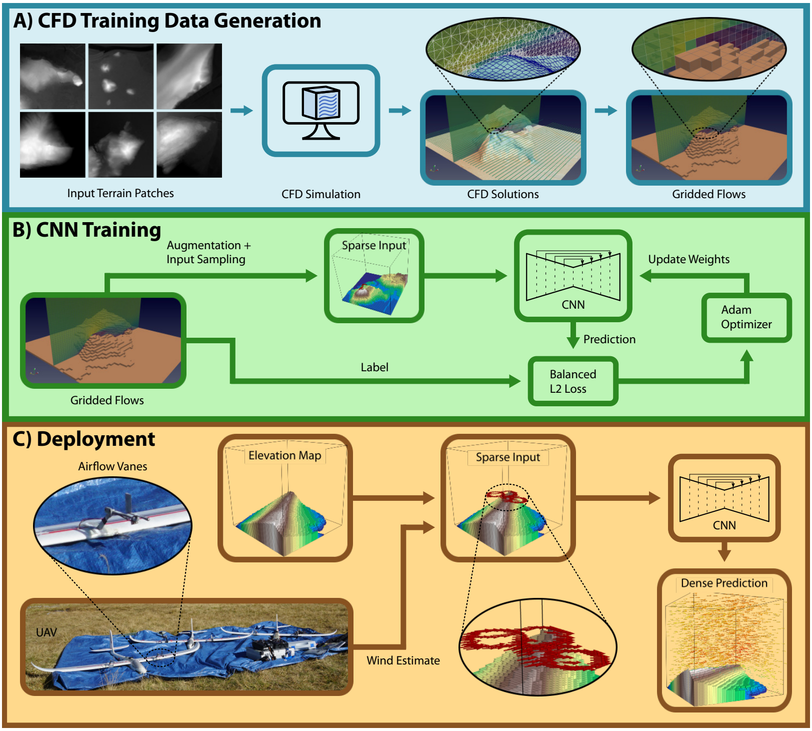


Figure 5.1: Overview of the wind prediction pipeline. (**A**) First we generate labelled flows utilizing a CFD simulation. (**B**) Then *WindSeer* is trained with measurements along randomly sampled piecewise linear trajectories to predict the dense flow. (**C**) During deployment the wind estimates from the UAV together with the known topography serve as the input to *WindSeer*.

2 Results

We evaluated *WindSeer* in a sequence of increasingly dynamic experiments. First, we compared *WindSeer*'s predictions to CFD-simulated flows over previously unobserved terrains, which allowed for a controlled evaluation of the overall *WindSeer* architecture, training scheme and prediction accuracy with dense labeled data. Second, we validated *WindSeer* against wind data from static wind masts gathered as part of large scale and long duration measurement campaigns [15, 17, 39, 142, 143]. These datasets serve as real-world test cases across a variety of terrains with increasing complexity. They also exhibit low noise levels, since the measurement locations are precisely known and the measurements are averaged over a time period of multiple hours, reducing the influence of instantaneous wind gusts on the prevailing wind estimates. *WindSeer* successfully predicted the wind and turbulence with especially low error on the vertical wind and turbulence. Finally, we evaluated our approach over several real multi-sUAV flights over the mountainous terrain of the Swiss Alps. Each sUAV used its own wind measurements to predict the wind at the other drones' locations, and we cross-validated these predictions between the aircraft. The measured wind was subject to high noise levels due to the uncertainty of the estimated sUAVs' pose and measurement errors in the airflow sensors. Nevertheless, *WindSeer* showed promising results on this data, especially predicting the vertical wind. Altogether, the experiments demonstrate multiple facets of *WindSeer*'s ability to successfully infer synthetic and real-world wind fields based on only a few noisy measurements.

2.1 *WindSeer* model

WindSeer, a convolutional neural network (CNN), takes as input a $4 \times n_x \times n_y \times n_z$ tensor (with $n_{x,y,z}$ being the respective spatial resolutions of the output prediction region). The four channels include a binary mask indicating cells containing input measurements, the terrain model (stored as a distance field) and the sparse horizontal wind speed measurements (two channels). *Note that vertical wind speed measurements are not an input to the model*, since vertical wind is not observable with a standard fixed-wing sUAV sensor set without either adding airflow angle sensors or identifying the glide polars of the sUAV. In Section S4 we show empirically that adding vertical wind as an input, if it were available, would not change prediction quality anyway. The percentage of observed cells in the input data tensors varies across experiments but is always low, ranging from $3.5 \times 10^{-6}\,\%$ to $0.19\,\%$. Thus, in our experiments *WindSeer* always operates on sparse observations.

The four-channel output of *WindSeer* has the same spatial dimension and resolution as the input tensor. The first three channels contain the three-dimensional wind prediction (W_x, W_y, W_z) and the fourth channel contains the TKE prediction. *WindSeer* leverages terrain knowledge by explicitly setting output wind and TKE predictions in cells corresponding to locations inside the terrain to zero.

In our evaluation we considered four variations of *WindSeer* [ZD4, ZD6, AD4, AD6] by varying the fill value and network depth. The fill indicator (Z or A) indicates how the wind speed input channels for the cells with no measurements are filled. We tested fill values of zero (Z) or the average of all measurements per channel (A). The network depth indicates the number of pooling/upsampling layers in the encoder/decoder of *WindSeer*. We consider depths of four (D4) and six (D6) resulting in receptive field sizes of 175 and 703 respectively.

2.2 Experiment group 1: Predicting CFD simulation flows

In this experiment, we evaluated the different *WindSeer* variations on a test set containing flows over previously unobserved terrain that were produced by the same CFD processing pipeline as the training data. The sparse input measurements include noise drawn from the same distribution as during training (10 % Gaussian noise and biases). We computed the average normalized error over all non-terrain cells for each test case. This allowed us to compare flows of different wind speeds across the test cases. Three terrain and input pairs together with the prediction error cloud are shown in Fig. 5.2 A) and B).

Prediction performance All *WindSeer* variants predicted the flow well, with median errors below 16.2 %. While the highest prediction errors occurred either close to the ground or on the lee side of the terrain, this trend is mitigated by the fact that, due to practical considerations such as payload configuration and safety, the operating altitude for sUAVs is typically over 50 m above ground level [34, 98]. Fig. 5.2 C) shows the error distribution for the four model variants over the full flow domain on the left side (blue) and excluding the lowest four cells above the terrain on the right side (green). These latter results (equivalent to only scoring the network output above an altitude of 46 m) illustrate the predictive performance for realistic sUAV flight regimes. There, all *WindSeer* variants produced more accurate wind predictions (error reduction AD4: 14.5 % to 11.5 %, AD6: 16.2 % to 13.2 %, ZD4: 15.3 % to 11.9 %, ZD6: 14.1 % to 10.8 %). All *WindSeer* variants predicted the TKE well with median relative errors between 11.2 % to 11.9 %. More detailed results are available in Section 5.5.

Error analysis The higher prediction errors close to the ground are possibly due to the differences in resolution available to *WindSeer* versus the original CFD simulation near the terrain where the effect of the surface boundary layer on the wind velocity is significant. *WindSeer* only has access to the 64^3-grid version of the terrain with a lateral and vertical resolution of 16.5 m and 11.5 m, in contrast with the CFD simulation which has variable spatial resolution (≈0.5 m, higher density near the surface). Interpolating the CFD results on the higher resolution grid possibly leads to a loss of information in the boundary layer region which in turn may result in wind patterns close to the ground which are more difficult to predict.

We evaluated three individual terrains in more detail (Fig. 5.2 A)) to assess the sensitivity of the prediction quality to the sampled input data locations. For each

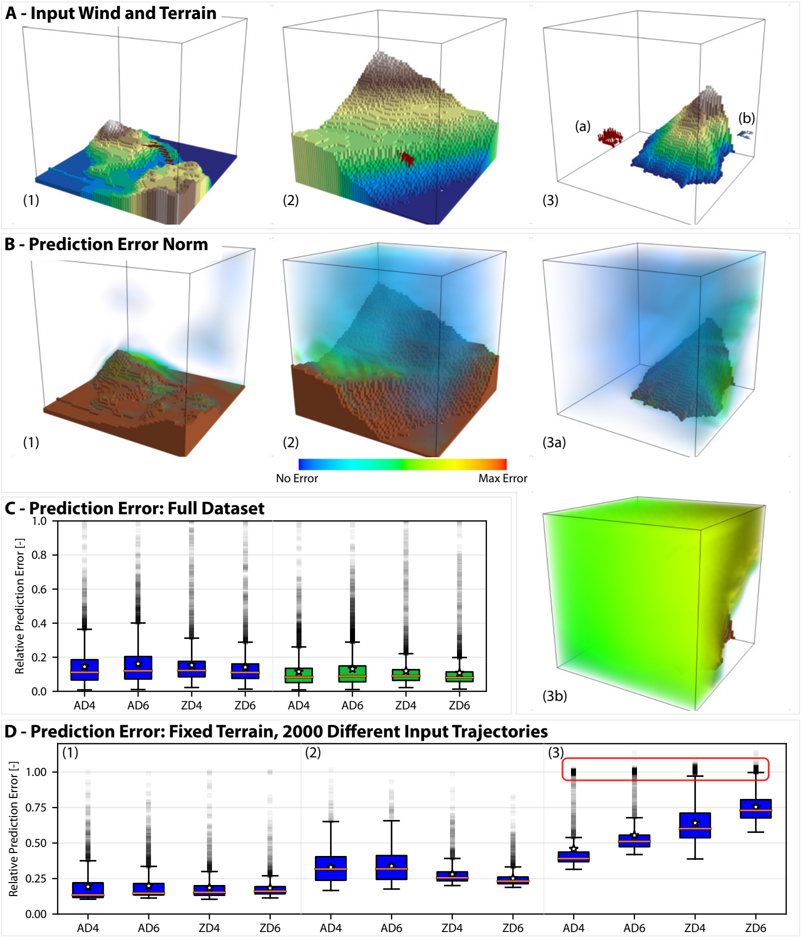


Figure 5.2: (**A**) Terrain and input wind pairs with their respective prediction error (**B**). High prediction errors can be observed close to the ground or on the lee side of the terrain. (**C**) Wind prediction performance on the CFD dataset over the full domain (blue). If the closest cells to the terrain are excluded, the error drops by 18.8 % up to 23.6 % (green). The prediction errors with 2000 random trajectories for three different terrains indicate that either AD4 or ZD6 perform best depending on the case (**D**). While most of the terrains result in uni-modal error distributions (1,2), more complex ones can have a second mode for samples from a complex flow region, indicated by the red box in (3).

terrain we randomly sampled 2000 trajectories and evaluated the prediction error (Fig. 5.2 D)). No noise or bias was added to the input data in this experiment to focus solely on the impact of the trajectory location. While for most terrains the models performed comparably (Fig. 5.2 A) (1)), there are examples showing clear trends favoring either ZD6 (Fig. 5.2 A) (2)) or AD4 (Fig. 5.2 A) (3)). Whenever the input data was sampled in regions where the model could not predict the prevailing flow well, e.g. the lee side of the hill (Fig. 5.2 A) (3b)), all models perform poorly. If such a region is large enough, multi-modal error distributions can be observed. This was the case for the terrain in (Fig. 5.2 A) (3)) where prediction failed for input wind samples on the right (lee) side of the hill.

Summary These experiments show *WindSeer*'s ability to replicate CFD predictions based on only sparse and noisy wind observations. *WindSeer* has higher prediction errors close to the ground and on the lee side of hills. In addition, not all measurement input regions result in the same predictive quality with measurements from the upwind side or in the free-flow regions further from terrain leading to more accurate predictions than those from the lee side of terrain features.

2.3 Experiment group 2: Evaluation on wind measurement campaign datasets

We used available in-situ wind and TKE measurements from three published measurement campaigns to evaluate the *WindSeer* wind prediction performance on real wind data. The Bolund hill in Denmark is a small island hill with a sharp 11 m high cliff on one side (Fig. 5.3 A)) [15, 17]. The Askervein hill located in Scotland is a gentle hill with a 116 m high peak (Fig. 5.3 C)) [142, 143]. The Perdigao region in Portugal represents the most complex test case with two roughly 300 m high parallel ridges (Fig. 5.3 E)). For each terrain, varying wind flow directions and magnitudes are available from different measurement periods (Fig. 5.3 B), D), and F)). The measurements at these sites were collected via wind velocity sensor suites (sonic or cup anemometers) mounted on masts providing data from 2 m to 100 m altitude above ground level [39].

Experiment setup Since *WindSeer* is fully convolutional, it is able to handle any input/output grid size above the minimum size of 64^3 cells. The varying geometric extents of the sites as well as the mast locations and heights required a larger grid size ($384 \times 384 \times 192$) with a higher resolution to obtain meaningful predictions. Accordingly, the grid resolution for Bolund hill was increased 30× from the training resolution to a cell size of 0.55 m horizontally and 0.38 m vertically. Similarly, the Askervein and Perdigao terrains increased resolution by 4× and 2×, respectively (Askervein cell size: $4.13\,\text{m} \times 4.13\,\text{m} \times 2.88\,\text{m}$, Perdigao cell size: $8.25\,\text{m} \times 8.25\,\text{m} \times 5.75\,\text{m}$).

The larger grid size resulted in much sparser input data in the range of $3.5 \times 10^{-6}\,\%$ to $3.2 \times 10^{-5}\,\%$, in contrast to the training density of $1.1 \times 10^{-3}\,\%$ to

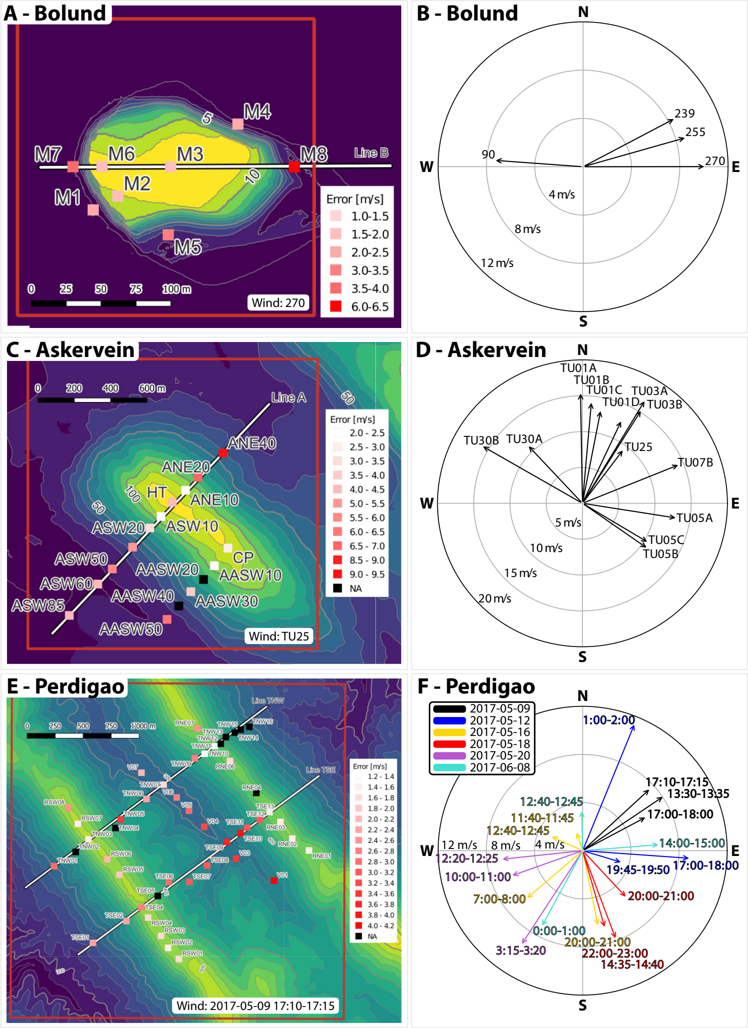


Figure 5.3: The mast locations and elevation maps for the Bolund (**A**), Askervein (**C**), and Perdigao (**E**) campaigns. The tower positions are colored by the average prediction error when using that specific mast as the input to predict the wind. In the Askervein and Perdigao case some masts did not provide a valid measurement for that experiment. (**B**, **D**, **F**) show the wind directions for the different experiments for each terrain.

0.19 %. While the models using average-filling (AD4, AD6) handled these sparsity levels well, this is not the case for the zero-fill models (ZD4, ZD6), which did not generalize to sparsity levels outside the training distribution. We observed that when fewer measurements were available (compared to training time) the zero-fill models predicted much lower wind speeds than the measured values, while when more measurements were available, the wind magnitudes were overpredicted.

We compared the two performant *WindSeer* variants against an averaging baseline (AVG) that assumes the wind and TKE is constant and always predicts the average of all measurements over the full domain. The methods predicted the wind based on the measurements from a single mast, yielding an ensemble of predictions for each wind case. We queried each prediction at all mast locations and compared it to the measured wind. We then averaged the results across all predictions and masts to get a performance metric for each wind case.

Prediction performance The averaged absolute prediction errors for the different cases are provided in Table 5.1 and Table 5.2. We computed three metrics: absolute errors in the prediction of wind magnitude S, vertical wind W, and and TKE. In most cases, both *WindSeer* variants clearly outperformed the averaging baseline (AVG) at all of these: 35/36 cases for wind magnitude, 33/36 for vertical wind, and 17/17 for TKE prediction. In the cases where AVG performed comparably to *WindSeer*, the wind direction aligns with the ridge/terrain, causing only small variations across the measurements of the different masts.

Wind magnitude S was best predicted by AD4 in 27 out of 36 cases, yielding a 17 % lower error than the baseline (AD6: 14 %). In particular, both variants predicted vertical wind and TKE, which are paramount for planning safe and efficient flight trajectories, significantly more accurately than the baseline (AD4: 43 % and 38 %, AD6: 47 % and 45 % less error than baseline, respectively). In most cases both *WindSeer* variants performed similarly to each other, explaining the small difference in the averaged error over all cases for the three metrics.

We qualitatively evaluated the influence of the measurement mast location on wind prediction quality. In Fig. 5.3 A), C), E) the masts are colored by the prediction error over all measurements of the AD4 model when using the measurements from that respective mast, with lighter color indicating input measurement mast locations that yielded more accurate predictions. Measurements from the top of the ridge/hill or from the upwind side generally resulted in low-error predictions. Measurements from the lee side of the terrains caused lower quality predictions, consistent with our findings from the CFD-simulated flows.

Prediction correlation We also report the correlation between the prediction and input measurements to assess if the models can predict the flow trends well (Table 5.3). As the AVG baseline produced a constant, location-invariant wind prediction, its correlation is undefined and not reported. Averaged over all cases, both *WindSeer* variants yielded strong positive correlations for all metrics indicating that the models were able to predict the observed trends well. For most cases the

Terrain	Case	Error S $[m/s]$ AVG	AD4	AD6	Error W $[m/s]$ AVG	AD4	AD6	Error TKE $[m^2/s^2]$ AVG	AD4	AD6
Bolund	90	1.80	**1.58**	1.61	0.85	**0.58**	0.62	1.91	1.33	**1.19**
	239	2.80	2.50	**2.49**	0.66	**0.34**	0.37	2.67	1.68	**1.66**
	255	3.24	**2.47**	2.61	0.85	**0.44**	0.46	3.43	2.12	**2.06**
	270	3.77	**2.79**	2.94	0.82	0.51	**0.48**	5.14	3.43	**3.35**
Askervein	TU25	2.58	**2.39**	2.44	1.10	0.37	**0.31**	0.61	0.41	**0.34**
	TU30A	1.14	**0.98**	1.20	0.41	0.26	**0.25**	1.38	0.72	**0.54**
	TU30B	1.80	**1.46**	2.12	0.51	0.41	**0.39**	2.82	1.40	**1.18**
	TU01A	3.26	**2.83**	3.21	1.41	0.52	**0.46**	1.89	1.06	**0.96**
	TU01B	3.24	**2.74**	3.12	1.37	0.48	**0.42**	1.64	0.98	**0.86**
	TU01C	3.55	**3.08**	3.42	1.23	0.45	**0.41**	1.17	0.72	**0.68**
	TU01D	4.21	**3.71**	4.04	1.26	0.47	**0.42**	1.62	1.17	**0.99**
	TU03A	5.29	**4.70**	5.00	1.74	0.64	**0.55**	2.04	1.31	**1.15**
	TU03B	4.90	**4.41**	4.70	1.54	0.54	**0.46**	1.82	1.21	**1.04**
	TU05A	4.71	**4.69**	4.72	0.76	0.31	**0.27**	1.73	0.99	**0.76**
	TU05B	1.18	**1.00**	1.12	0.31	**0.26**	0.28	1.40	0.63	**0.51**
	TU05C	**0.93**	0.93	1.01	0.34	0.25	**0.24**	1.09	0.43	**0.39**
	TU07B	3.42	3.27	**3.18**	1.59	0.49	**0.41**	2.44	1.85	**1.43**

Table 5.1: Absolute prediction errors for the velocity magnitude (S), vertical wind component (W), and turbulence kinetic energy (TKE) on the Bolund and Askervein datasets of the AD4 and AD6 models compared to the averaging baseline (AVG). The average error over all cases (Bolund, Askervein, Perdigao is presented in Table 5.2

Terrain	Case	Error S [m/s]			Error W [m/s]			Error TKE [m^2/s^2]		
		AVG	AD4	AD6	AVG	AD4	AD6	AVG	AD4	AD6
Perdigao 2017-05-09	13:30-13:35	2.91	2.27	**2.17**	0.85	**0.57**	0.58	-	-	
	17:10:17:15	4.41	3.33	**3.12**	1.22	0.88	**0.86**	-	-	
	17:00-18:00	3.06	2.37	**2.24**	0.80	**0.56**	0.58	-	-	
Perdigao 2017-05-12	01:00-02:00	2.82	2.31	**2.23**	0.52	0.42	**0.40**	-	-	
	17:00-18:00	2.76	2.23	**2.11**	0.70	0.57	**0.47**	-	-	
	19:45-19:50	1.15	**0.90**	0.91	0.21	**0.16**	0.18	-	-	
Perdigao 2017-05-16	07:00-08:00	1.58	1.16	**1.15**	0.27	0.27	**0.22**	-	-	
	11:40-11:45	0.85	0.77	**0.76**	0.33	0.27	**0.27**	-	-	
	12:40-12:45	0.86	0.75	**0.74**	0.29	0.22	**0.22**	-	-	
	20:00-21:00	1.35	**0.97**	1.04	**0.22**	0.31	0.29	-	-	
Perdigao 2017-05-18	14:35-14:40	2.14	1.82	**1.75**	0.41	0.36	**0.33**	-	-	
	20:00-21:00	1.54	**1.08**	1.11	**0.17**	0.19	0.20	-	-	
	22:00-23:00	1.52	**1.13**	1.18	0.17	0.17	**0.16**	-	-	
Perdigao 2017-05-20	03:15-03:20	3.59	**2.63**	2.77	**0.41**	0.55	0.46	-	-	
	10:00-11:00	2.20	1.84	**1.75**	0.51	0.42	**0.38**	-	-	
	12:20-12:25	1.93	1.71	**1.61**	0.58	0.43	**0.41**	-	-	
Perdigao 2017-06-08	00:00-01:00	2.68	**1.87**	2.03	0.35	0.38	**0.33**	-	-	
	12:40-12:45	1.20	0.90	**0.87**	0.31	**0.22**	0.24	-	-	
	14:00-15:00	2.49	1.77	**1.64**	0.74	**0.42**	0.43	-	-	
Total average		2.58	**2.15**	2.23	0.72	0.41	**0.38**	2.05	1.26	**1.12**

Table 5.2: Absolute prediction errors for the velocity magnitude (S), vertical wind component (W), and turbulence kinetic energy (TKE) on the Perdigao dataset of the AD4 and AD6 models compared to the averaging baseline (AVG). The average error over all cases (Bolund, Askervein, Perdigao is presented in the last row.

AD4 and AD6 correlations were similar. As in the results for the absolute errors, the AD4 model tended to score better for wind magnitude prediction (22 out of 36 cases) while the AD6 model performed better for the vertical wind (27 out of 36 cases). However, in contrast to the absolute error, the AD4 predicted the TKE better in 10 out 17 cases. These high correlation values show that the model predictions are useful to distinguish updraft from downdraft regions and characterize zones with potentially dangerous high turbulence levels, thus providing a valuable contribution to planning safer and more efficient sUAV trajectories.

Wind field trends along straight lines The masts in each campaign were arranged along two or more straight lines. This allowed us to qualitatively assess whether the models could predict flow trends along these lines, such as speedups or vertical up-/downdrafts, well. We expect trends to be more noticeable when the wind and line direction are parallel. We selected four cases where this is the case in the ground-truth data (wind directions and mast locations are shown in Fig. 5.3) and show the *WindSeer* and baseline predictions in Fig. 5.4. For each method and case we show three predictions using the measurements from different masts as the input. The error bars for wind speed measurements report the 1-σ uncertainty, calculated using error propagation from the standard deviations of each axis [63].

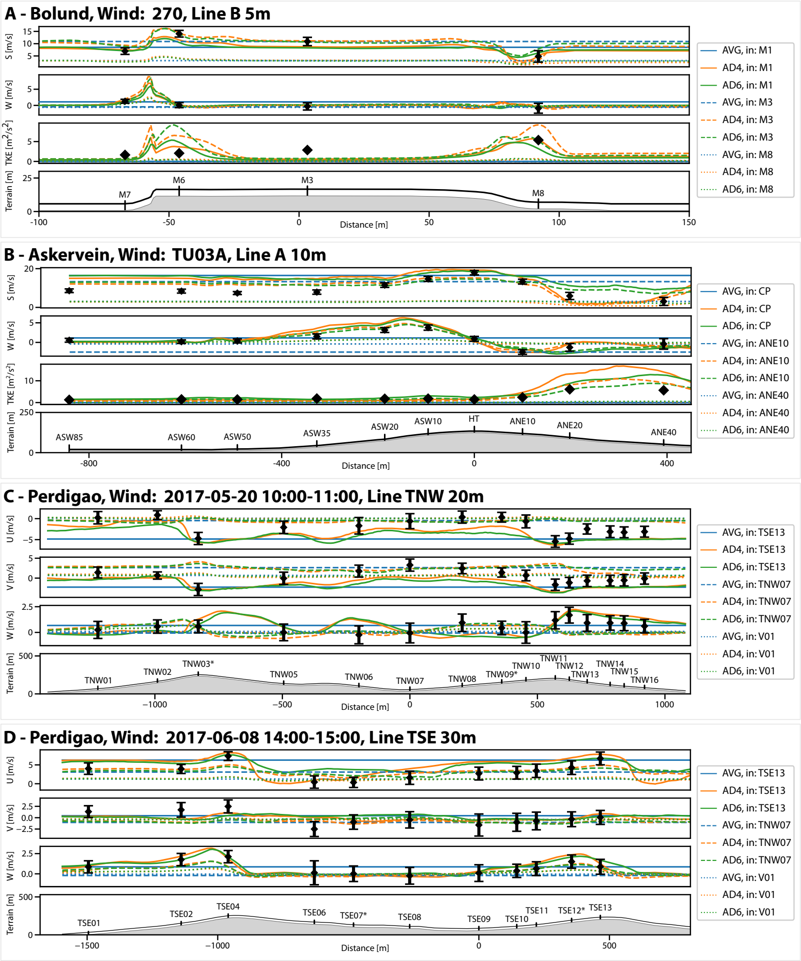
The *WindSeer* variants successfully predicted the speed changes and vertical up-/downdrafts for the cases A), B), and D) if the input mast was not on the lee side of a hill. Predictions using only measurements from the lee side (Bolund: M8, Askervein: ANE40, Perdigao: V01/TNW07) severely under-predicted the wind magnitude. Wherever TKE measurements were available, *WindSeer* predicted TKE trends well (cases A) and B)).

Fig. 5.4 C) shows an interesting case where the models predicted most regions of the flow well, but failed to predict the complex flow between the two ridges. Measurements indicate that a lee side rotor was present in the valley which occurs when flow detaches on the downwind side and causes a recirculating pattern (Section S1). This causes a prominent switch in the vertical wind direction as seen by the measurements of the towers TNW05 to TNW10. Our network training dataset does not contain samples with lee side rotors and therefore the models did not learn to predict such flow patterns.

Summary The average-fill *WindSeer* variants (AD4, AD6) were able to generalize to much sparser input data and higher grid resolutions compared to the training input and predicted the wind measured over the Askervein, Bolund, and Perdigao terrains well. The wind magnitude, the vertical wind, and TKE were significantly better predicted compared to the average constant-wind.

		Correlation S		Correlation W		Correlation TKE	
Terrain	Case	AD4	AD6	AD4	AD6	AD4	AD6
Bolund	90	**0.72**	0.67	0.50	**0.64**	0.58	**0.60**
	239	0.68	**0.73**	**0.79**	0.75	0.86	**0.93**
	255	0.82	**0.85**	0.72	**0.76**	0.82	**0.89**
	270	0.84	**0.91**	0.78	**0.84**	0.73	**0.78**
Askervein	TU25	**0.65**	0.60	0.90	**0.96**	**0.89**	0.88
	TU30A	0.61	**0.67**	0.58	**0.73**	0.42	**0.54**
	TU30B	**0.73**	0.71	**0.64**	0.63	0.23	**0.50**
	TU01A	**0.76**	0.52	0.91	**0.92**	**0.85**	0.46
	TU01B	**0.79**	0.53	0.92	**0.93**	**0.87**	0.53
	TU01C	**0.78**	0.46	0.92	**0.93**	**0.90**	0.46
	TU01D	**0.79**	0.54	0.92	**0.94**	**0.93**	0.81
	TU03A	**0.78**	0.65	0.93	**0.95**	**0.98**	0.94
	TU03B	**0.77**	0.61	0.92	**0.95**	**0.90**	0.87
	TU05A	**0.18**	0.16	0.89	**0.91**	**0.40**	0.31
	TU05B	**0.79**	0.69	0.48	**0.53**	**0.04**	-0.09
	TU05C	**0.66**	0.53	0.58	**0.60**	**0.14**	-0.05
	TU07B	**0.70**	0.66	0.90	**0.97**	0.40	**0.41**
Perdigao 2017-05-09	13:32:30	**0.82**	0.82	0.53	**0.57**	-	-
	17:12:30	0.48	**0.80**	0.35	**0.48**	-	-
	17:00-18:00	0.77	**0.77**	0.50	**0.54**	-	-
Perdigao 2017-05-12	01:00-02:00	**0.76**	0.74	0.45	**0.51**	-	-
	17:00-18:00	**0.81**	0.81	0.57	**0.61**	-	-
	19:45-19:50	0.71	**0.72**	**0.57**	0.53	-	-
Perdigao 2017-05-16	07:00-08:00	0.65	**0.68**	**0.67**	0.63	-	-
	11:40-11:45	0.51	**0.55**	**0.22**	0.18	-	-
	12:40-12:45	**0.48**	0.45	0.30	**0.31**	-	-
	20:00-21:00	**0.68**	0.64	0.21	**0.25**	-	-
Perdigao 2017-05-18	14:35-14:40	0.70	**0.72**	0.33	**0.39**	-	-
	20:00-21:00	**0.84**	0.80	0.12	**0.19**	-	-
	22:00-23:00	**0.74**	0.69	0.42	**0.50**	-	-
Perdigao 2017-05-20	03:15-03:20	0.61	**0.64**	**0.30**	0.30	-	-
	10:00-11:00	0.71	**0.76**	**0.46**	0.44	-	-
	12:20-12:25	0.66	**0.71**	**0.45**	0.42	-	-
Perdigao 2017-06-08	00:00-01:00	**0.69**	0.67	**0.44**	0.42	-	-
	12:40-12:45	0.77	**0.77**	0.46	**0.46**	-	-
	14:00-15:00	**0.82**	0.82	0.58	**0.62**	-	-
Total average		**0.70**	0.67	0.59	**0.62**	**0.64**	0.57

Table 5.3: Correlation between the measurements and predictions for the velocity magnitude (S), vertical wind component (W), and turbulence kinetic energy (TKE) on the measurement campaign datasets of the AD4 and AD6 models.



A - Bolund, Wind: 270, Line B 5m

B - Askervein, Wind: TU03A, Line A 10m

C - Perdigao, Wind: 2017-05-20 10:00-11:00, Line TNW 20m

D - Perdigao, Wind: 2017-06-08 14:00-15:00, Line TSE 30m

Figure 5.4: Predictions and measurements along characteristic lines with a constant height for each experiment. Three predictions using different input masts are shown for each model and experiment. The asterisk * indicates that no measurement was available for that respective mast at the queried height and the closest one was picked. The uncertainty of the measurements is displayed by the standard deviation of the raw high-rate data.

2.4 Experiment group 3: Predicting the wind along sUAV trajectories

We flew multiple fixed-wing sUAVs (EasyGlider (EZG) 4) simultaneously in the Swiss Jura (Chasseral) and the Swiss Alps (Oberalppass and Gotthardpass). Video 1 contains flight test footage, the flight paths together with *WindSeer* predictions. While the Chasseral mountain is an isolated ridge in the Jura, the Oberalppass and Gotthardpass represent more challenging and complex cases as they are saddles surrounded by steep mountains. The sites were selected based on accessibility by car and the wind forecast on the respective flight days. At Chasseral we flew three sUAVs simultaneously collecting wind data. The spatial constraints at the Oberalppass and Gotthardpass only allowed for two sUAVs. The flight plans for each sUAV consisted of multiple circular loiter patterns with 100 m radius. The flight paths for the Chasseral and Oberalppass flights are shown in Fig. 5.5 A) and D). The sUAVs were equipped with a modified version of the PX4 autopilot and airflow sensing to obtain a pose estimate and local 3D wind estimate. While the wind estimate could have been calculated online on the sUAV, we opted for an offline flight path reconstruction (FPR) approach. This allowed us to investigate the wind estimation performance with various hyperparameter and calibration settings.

In this experiment, *WindSeer* predicted the wind field based on a short trajectory segment-worth of noisy wind observations from one sUAV. We generated the sparse input by binning the observations based on the estimated global position into a discretized prediction grid. The grid size was set to 64^3 at the native training resolution, with the grid center at the first observed sUAV position estimate. In case of multiple measurements in one grid cell we averaged all measurements made in that cell. A high resolution elevation map was used to construct the terrain for the *WindSeer* input.

Loiter-to-loiter evaluations In these experiments, we obtained one observation per loiter by averaging all measurements along the path. This reduced the noise of individual observations, as well as the effect of possibly miscalibrated airflow sensors that lead to an oscillation of the wind estimate within a loiter (Section 5.7). *WindSeer* predicted the wind field based on the data of one loiter from one sUAV (approximately 50 to 100 measurement cells). The predicted wind field was then queried at the other loiter positions of all sUAVs and an average value for the prediction along the loiter path was calculated. We show the aggregated metrics (average absolute error and correlation) over all loiters for each flight in Table 5.4.

The safety-critical vertical wind was well represented, especially by the zero-fill *WindSeer* variants with 29 % (ZD4) and 33 % (ZD6) lower prediction errors than the baseline. The variation of the measured horizontal wind in our flight data relative to the noise level (sensor noise, wind gusts, changing weather conditions) is much smaller compared to the data from the static measurement masts, resulting in the AVG baseline approximating the *horizontal* wind well (magnitude W_{hor} and bearing Ψ_{hor}) (Fig. 5.5 C) and F)). The *WindSeer* variants provided comparable

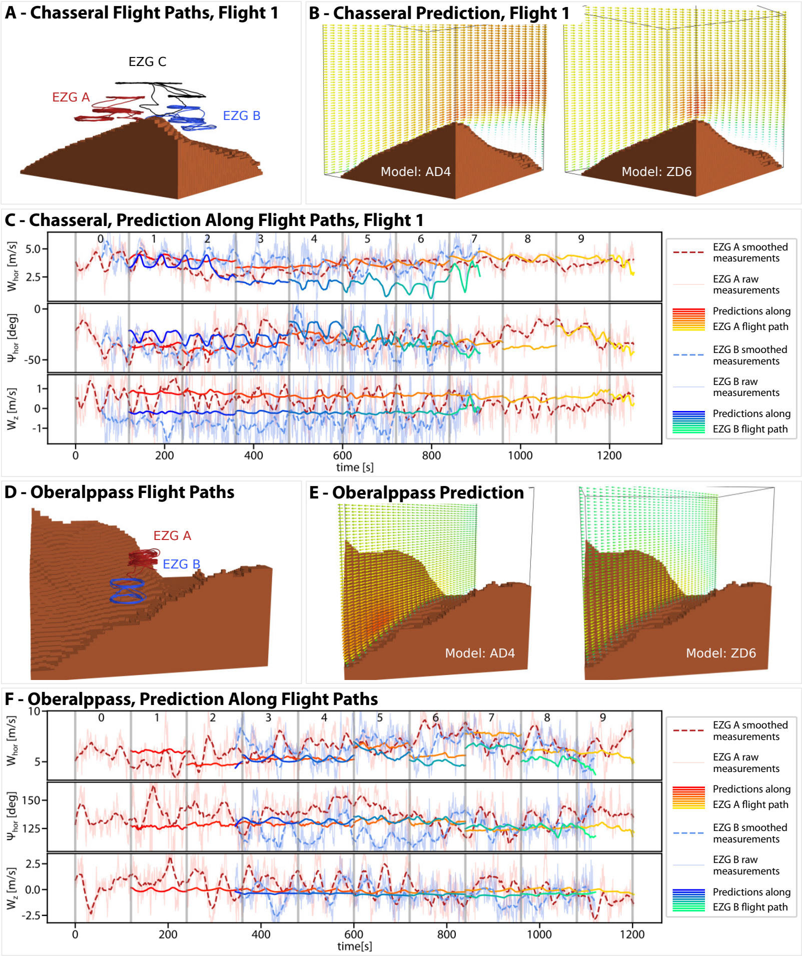


Figure 5.5: Prediction results and flight paths for two flight tests: Chasseral (**A**-**C**) and Oberalppass (**D**-**F**). The predictions along a slice are shown for the AD4 and ZD6 models (**B**, **E**). (**C**) and (**F**) show the sliding window predictions of ZD6 along the flight paths using the data from EZG A as input. Every 120 s a prediction is made using the wind data from the previous window.

Flight	Model	Mean Absolute Error			Correlation		
		W_{hor} [m/s]	Ψ_{hor} [deg]	W_z [m/s]	W_{hor}	Ψ_{hor}	W_z
Chasseral 1	AVG	0.62	**7.22**	0.53	-	-	-
	AD4	0.77	7.84	0.66	0.54	**-0.09**	0.94
	AD6	**0.61**	7.49	0.55	**0.77**	-0.17	**0.95**
	ZD4	0.87	9.18	0.41	0.26	-0.13	0.95
	ZD6	0.84	9.19	**0.38**	0.33	-0.37	0.95
Chasseral 2	AVG	0.57	**13.6**	0.50	-	-	-
	AD4	0.66	14.7	0.48	**0.50**	**-0.35**	**0.90**
	AD6	**0.48**	14.1	0.47	0.48	-0.46	0.87
	ZD4	0.73	17.2	0.35	0.49	-0.42	0.78
	ZD6	0.74	16.8	**0.31**	0.49	-0.60	0.80
Chasseral 3	AVG	0.55	**9.1**	0.49	-	-	-
	AD4	0.56	9.8	0.39	0.43	**-0.21**	0.82
	AD6	**0.55**	9.5	0.40	**0.51**	-0.33	0.84
	ZD4	0.67	14.0	0.32	0.32	-0.43	0.80
	ZD6	0.65	12.4	**0.30**	0.37	-0.38	**0.84**
Oberalppass	AVG	**0.55**	7.0	0.55	-	-	-
	AD4	1.05	19.7	0.30	0.28	-0.86	**0.81**
	AD6	3.12	7.5	0.35	**0.43**	**0.85**	0.76
	ZD4	0.65	**5.8**	**0.33**	-0.24	0.62	0.78
	ZD6	0.77	7.9	0.34	-0.46	-0.91	0.77
Gotthardpass	AVG	**1.00**	**7.5**	0.21	-	-	-
	AD4	2.55	64.3	0.57	0.26	0.75	0.21
	AD6	1.41	58.3	1.06	**0.71**	**0.98**	0.48
	ZD4	1.40	7.9	0.20	-0.06	0.97	0.48
	ZD6	1.19	7.8	**0.17**	0.33	0.63	**0.88**
All Chasseral Flights	AVG	0.58	**9.98**	0.51	-	-	-
	AD4	0.66	10.78	0.51	0.49	-0.22	**0.89**
	AD6	**0.55**	10.36	0.47	**0.55**	-0.32	**0.89**
	ZD4	0.76	13.46	0.36	0.36	-0.33	0.84
	ZD6	0.74	10.60	**0.34**	0.19	-0.11	0.86
All Flights	AVG	**0.66**	**8.88**	0.46	-	-	-
	AD4	1.12	23.27	0.48	0.40	-0.15	0.74
	AD6	1.23	19.38	0.57	**0.58**	**0.17**	0.78
	ZD4	0.86	10.82	0.32	0.15	0.12	0.76
	ZD6	0.84	9.5	**0.30**	0.09	-0.13	**0.85**

Table 5.4: The mean absolute error and the correlation for the horizontal wind magnitude W_{hor}, wind direction Ψ_{hor}, and vertical wind W_z on the loiter-averaged data. The results are the average over all loiters for all planes for the respective flight.

predictions for the Chasseral flights but struggled in the more complex Oberalppass and Gotthardpass flight data. Nevertheless, all models predicted the horizontal wind magnitude trends well, as indicated by the positive correlations, particularly evident in the Chasseral flights.

In general, the zero-fill variant predictions exhibited a lower absolute error while the average-fill models resulted in higher correlations for the horizontal wind. This indicates that the averaging models relied heavily on the average measurement value and less on the location of the measurements and therefore were prone to predicting flows at different scales, but still with the correct flow trends.

For all cases, all *WindSeer* models predicted the trends of the vertical wind well with high positive average correlations ranging from 0.74 (AD4) to 0.85 (ZD6). This indicates that *WindSeer* can predict the locations of dangerous downdrafts, as well as favourable updraft regions just based on wind measurements taken from an sUAV. For a qualitative assessment we show slices through the prediction for the models AD4 and ZD6 for one Chasseral flight (Fig. 5.5 B)) and the Oberalppass flight (Fig. 5.5 E)). The zero-filling model predictions (ZD4, ZD6) resulted in a lower wind magnitude error than the averaging models (AD4, AD6), as they predicted the average wind magnitude better. However, the averaging models predicted the flow trends with higher precision, resulting in higher correlations, albeit with an offset in the absolute flow magnitude. This same analysis holds for the vertical wind in the Chasseral flights.

Sliding-window evaluations In the second set of experiments, we qualitatively evaluated the performance of the ZD6 model for two flights. We chose ZD6 because it performed best at predicting the crucial vertical wind direction in the loiter experiment. We evaluated the model in a sequential time-windowed manner, where *WindSeer* took input data from a 120 s window to predict the wind along the flight path within the next 120 s window for the input and validation sUAVs.

Fig. 5.5 C) shows the prediction for the first Chasseral flight for the EZG A and EZG B sUAVs (using EZG A as the input data source). The difference in the horizontal wind between the two sUAVs is very small compared to the noise level. Despite this, the model was able to accurately predict the wind for certain parts of the validation flight such as windows 1 and 2. There, the model fit the horizontal wind magnitude well and even predicted the trends well for the change in wind direction, albeit with a directional offset. If the wind direction changed, such as from window 7 to 8, the predictions initially did not match the measurements well but could adjust in the subsequent windows to the changed condition as new input measurements were used for the network prediction (window 9). The model was able to accurately predict the magnitude difference in the vertical wind between the two sUAVs. It slightly under-predicted the downwind on the lee side for the validation sUAV, which can be explained by the generally worse performance of the models on the lee-side wind predictions, as shown in the previous experiments.

The sliding window predictions for the Oberalppass flights prove more challenging for *WindSeer*. Predictions using the EZG A data did not vary much between the locations of the two sUAVs despite there being some distinct horizontal wind heading changes (see windows 3 to 6). The Oberalppass and Gotthardpass are especially challenging prediction terrains, as they exhibit large altitude changes (1500 m) due to valleys and peaks within 4 km of our flight locations. High surrounding peaks can cause high gust levels, explaining the high variation in the measured wind. Furthermore, terrain outside the prediction area can significantly influence the wind features observed in the valley. In contrast, our CFD simulation setup for generating the *WindSeer* training data was limited to well-defined inflow conditions and a domain size of 1.5 km × 1.5 km, which did not allow simulating the flow over multiple large scale mountains or ridges. Thus, during training *WindSeer* did not observe wind flow fields similar to those over Oberalppass, explaining the performance difference between the Chasseral and Oberalppass/Gotthardpass flights. Nevertheless, when comparing the vertical wind predictions to the AVG baseline, *WindSeer* still produces more accurate estimates in these challenging domains.

Summary *WindSeer* accurately predicted the vertical wind, critical for planning safe and efficient flight trajectories, with a clear improvement over the baseline method. Given noisy onboard wind measurements from an sUAV, *WindSeer* was also able to predict magnitude and directional changes in the 3D wind field. Finally, since wind along the horizontal plane mostly had a predominant direction and magnitude on this small scale (roughly 500 m × 500 m), the baseline averaging method provided on par performance with *WindSeer* for predictions in this regime.

2.5 *WindSeer* inference time on small uncrewed aerial vehicle (sUAV) compute

We evaluated prediction times of *WindSeer* on flight grade hardware to show real-time performance. We ran the experiments on an Xavier NX, a single-board computer that can be carried by a typical sUAV. The average inference time over 100 runs for the models with depth 4 (AD4, ZD4) on a 64^3 prediction domain was 0.127 ± 0.003 s. Mixed-precision inference lowered the inference time to 0.058 ± 0.004 s. The models with depth 6 (AD6, ZD6) resulted in a slightly longer inference time of 0.174 ± 0.009 s, with mixed-precision dropping this further to 0.127 ± 0.005 s. These inference times show that *WindSeer* is capable of low-latency wind predictions to quickly adjust to new measurements.

3 Discussion

In this article we propose an approach to train a special CNN, *WindSeer*, for predicting the low-altitude wind and TKE around complex terrain in real-time based on sparse and noisy wind measurements and known topography. We trained *WindSeer* solely on simulated RANS CFD flows over terrain patches from Switzerland. The resulting network is able to replicate previously unobserved CFD solutions with high accuracy (median relative error below 10 %) based on sparse measurements with up to 10 % added Gaussian noise and bias.

We demonstrated zero-shot sim-to-real transfer by evaluating *WindSeer* on real wind measurements without retraining. On the historic measurement campaign datasets we showed that *WindSeer* was able to reconstruct real wind flows of different scales, at up to 30 times higher resolution than the training data. This corresponds to larger prediction domains containing up to 108 times more cells than those used for network training. On both the measurement campaign datasets and on data collected during our flight tests, *WindSeer* was able to predict the vertical wind, key to safe and efficient sUAV flight, with higher accuracy than the baseline. *WindSeer* also performed competitively against the baseline for predicting wind along the horizontal plane. When evaluated on the measurement campaign datasets, *WindSeer* produced consistently better predictions, while on the sUAV flight data, where the variation of the observed wind was small compared to noise in the data, prediction performance was more mixed. Finally, the TKE predictions were validated with CFD and measurement campaign data, allowing *WindSeer* to identify areas with high gust levels and thus enabling sUAVs to plan safer paths by avoiding these regions.

***WindSeer* variants** We evaluated a total of four *WindSeer* variants with different depth and input composition methods. The models using the average for all measurements as the fill value (AD4, AD6) generalize better to sparser input data and thus to larger prediction domains. However, they are prone to relying too much on the measured average and thus perform worse if the measured average is not a good representation of the average velocity in the flow field. On the other hand, the zero-fill models (ZD4, ZD6) are complementary, meaning that they do not generalize well to changes in the sparsity and domain size but can better encode the input measurement locations.

Future research will be needed to investigate if the drawbacks for each variant can be eliminated by adjusting the training pipeline. Increasing the depth of the *WindSeer* architecture seems to slightly improve the prediction quality on the CFD data as well as on real wind data. Deeper networks also offer the advantage of larger receptive fields, which helps to propagate the information of the measurements through the network layers.

Dense wind predictions from sparse localized measurements Previous work has shown the ability of neural networks to predict fluid flows for well-defined geometries paired with well-known inflow conditions [18, 59, 117, 148, 169]. We extended the capability of neural networks to work with sparse and noisy input data on realistic complex terrain. This enables real-time wind prediction using data that is feasible to obtain aboard an sUAV. The sparsity of the input data (0.19 % down to 3.5×10^{-6} %) exceeds previous research of sparse-to-dense neural networks that usually assume denser data around 0.75 % [54, 82], 0.2 % [79], or 6.5×10^{-3} % [50]. In the previous examples the sparse input data was distributed over the whole prediction domain while in our case we showed the prediction still performs well even if the samples are located within a spatially constrained sub-region.

3.1 Limitations and Future Work

Training domain We restricted our CFD simulation domain to 1.5 km × 1.5 km based on the initial assumption that the large scale NWP would be used in the network input. This domain size restricted the terrain to contain mostly one single major geographical feature such as a mountain or a ridge. Therefore the current CFD training data does not contain samples that include wind phenomena such as lee-side rotors, which arise in the presence of multiple mountains/ridges and may have existed during our Oberalppass and Gotthardpass sUAV flights. A larger simulation domain in the order of 10 km × 10 km could allow a better representation of such complex flows and possibly increase the wind prediction performance inside mountain ranges. A non-uniform sampling strategy when composing the input could also help to solve the sample imbalance problem by exposing the network to more examples from the lee-side flow regime during training.

RANS vs. LES Changing the CFD simulation from a time-averaged RANS solution to a time-varying model such as large eddy simulation (LES) could have multiple advantages: First, a model could be trained using the time-varying wind data to construct the input but still predict the time-averaged solution. This could result in the model learning wind gust characteristics and thus increase robustness to the noisy wind estimates from the sUAV. Secondly, a model could be trained to predict the time-varying flow representing wind gusts in the predicted wind. However, whether the information from the noisy measurements are sufficient to uniquely determine the correct flow still needs to be carefully analyzed. Depending on the sparsity of the data there are likely to be multiple flow solutions matching the observations.

Fluid flow assumptions Currently we model the air as an incompressible fluid with uniform temperature. By including temperature differences in the compressible fluid and terrain the, CFD simulation could model complex flow phenomena such as thermals [7], updrafts caused by temperature differences on the ground, or mountain waves [24, 33] that are large scale oscillations of the wind direction and magnitude behind large ridges resulting in periodically strong up- and downdrafts. However, to simulate these phenomena using CFD will require far more input data and ultimately we would still need to verify whether these simulations provide realistic flows that reflect the true airflow characteristics.

Scaling up flight experiments Our current wind sensing setup onboard the sUAV is prone to calibration errors resulting in relatively large wind estimation errors. Different sensors, such as a five hole probe [122] paired with an improved calibration procedure for the mounting offset will result in better wind estimates, making smaller variations in the wind observable. Finally, we think larger scale flight experiments in the order of multiple kilometers, ensuring the sUAVs fly in different flow regions, together with the improved wind estimation will enable us to assess if *WindSeer* is able to predict the wind at a large scale with the noisy sUAV data with similar accuracy as the data from the static measurement masts.

4 Materials and Methods

4.1 Overview

We developed a pipeline (Fig. 5.1) to train and deploy a CNN, *WindSeer*, that predicts the dense wind around complex terrain. The network training consists of two steps: First we generated a dataset of dense labelled flows over terrain patches from Switzerland using a RANS CFD solver (Fig. 5.1 A)). We then trained *WindSeer* using the label flows to simulate local wind measurements along randomly generated piecewise-linear trajectories, robustifying the predictions by adding noise to the measurements along the trajectories (Fig. 5.1 B)). The trained *WindSeer* was evaluated on (i) held back CFD-simulated flows on previously unobserved terrains, (ii) real wind data gathered in measurement campaigns [15, 17, 39, 142, 143], and (iii) wind measurements by sUAVs (Fig. 5.1 C)).

4.2 CFD wind data

We used a similar pipeline to Achermann et al. [1], to generate labelled air flows over natural terrain. In this work we extracted 563 terrain patches each with an extent of 1.5 km×1.5 km from the GeoVite service, which provides access to the swissALTI3D digital elevation map (DEM) for Swiss researchers, with a lateral resolution of 0.5 m[1]. The terrain patches exhibit at least one side with near-constant elevation allowing us to simulate a formed boundary layer flow (logarithmic profile) entering into the domain from that face. Some terrains allowed for multiple flow directions leading to 866 terrain/flow direction pairs. The vertical extent of the simulation domain was three times the height difference of the terrain with a lower bound of 1100 m minimizing the boundary effects on the flow. Each case was simulated with up to 15 different wind speeds if the automatic meshing succeeded, resulting in 7361 executed CFD runs. Only solutions that met a required optimization tolerance were accepted as fully converged solutions, which was the case in 92.9 % of the runs. We enhanced our dataset with one zero-velocity flow for each terrain that had at least one converged CFD simulation, resulting in a total of 7285 flows.

The CFD solutions are computed on an automatically generated irregular grid. We resampled each case up to a height of 1100 m to a regular $91 \times 91 \times 96$ grid resulting in a resolution of 16.5 m horizontally and 11.5 m vertically (Fig. 5.1 A)).

---

[1] `https://geovite.ethz.ch/`, recently also available on ArcGIS: `https://elevation.arcgis.com/arcgis/rest/services/WorldElevation/Terrain/ImageServer`

4.3 Data augmentation

Generating CFD flows is a computationally and labor-intensive task [1]. For reference, our 7361 CFD runs required 9168 h CPU compute time (782 h creating the meshes and 8386 h solving the flow). Unfortunately, deep networks are notoriously data-hungry and, for a complex modeling problem such as wind prediction, would typically require orders of magnitude more training data to achieve good performance. In computer vision, image augmentation methods are widely used when training deep CNNs [132]. These methods aim to improve the quality and size of the datasets when only limited data is available to prevent the networks from overfitting. In this work, we showed, for the first time, that geometric transformations can be applied to CFD flows to augment the *WindSeer* training data.

We randomize the locations of terrain features and flow directions by generating 64^3 subdomains sampled from each full $91 \times 91 \times 96$ grid. The subdomains are constructed by sampling from a range of rotations and origin translation offsets inside the full domain. We use a uniform distribution to sample the rotation and the horizontal shift with bounds ensuring that the subdomain is fully contained within the full grid. The vertical shift is sampled from a triangle distribution with lower limit and mode of 0 and an upper limit of 32. Smaller vertical offsets are favoured to focus on the complex flow regions closer to the terrain. The flow data is linearly interpolated to the coordinates of the subdomain grid, which is the same spatial resolution as the full grid.

4.4 Input and label composition

The input to *WindSeer* consists of four volumetric channels, one of which corresponds to the terrain encoding T created as a Euclidean distance transform with zeros inside the terrain. The remaining three channels include the sparse and noisy horizontal wind measurements $(U_{x,in}, U_{y,in})$ and a binary mask B indicating cells containing measurements. Previous work has shown the value of providing binary input masks to CNNs handling sparse input data [66, 147].

Measurements from realistic flight scenarios only cover a small percentage of the prediction volume along a connected path, e.g. a 30 s flight segment with our sUAV covers approximately 20 cells at the default grid resolution. Consequently, for a practical onboard wind prediction scenario, we expect the available input wind data to be very sparse and thus construct our network input to reflect this sparsity. We create the input based on the augmented dense flow by creating a mask and then selecting the measurements based on the mask. We emulate the characteristics of an sUAV flight path by filling the mask along a random piecewise linear segments with a length of 3 to 500 cells.

Noise is added to the sampled wind data in order to account for the fluctuations in the wind and sensor errors that are not captured by the RANS CFD simulations. Two types of disturbance are added, white Gaussian noise (sampled i.i.d. at each measurement from $\mathcal{N}(0, \sigma_g)$) and measurement bias (sampled from $\mathcal{U}(-0.1, 0.1)$ and applied to all measurements). The first has the purpose of simulating noise due to sensor measurements [94], while the latter simulates the effects of sensor miscalibration. The standard deviation for the Gaussian noise σ_g itself is drawn from a uniform distribution: $\sigma_g \sim \mathcal{U}\left(0, 0.1\right)$, simulating different noise levels. All the noise values are scaled with the mean wind velocity for each sample in the training set to have coherent noise levels from low to high velocity samples. Note that noise is only added to the training inputs and not to the CFD ground truth labels used to compute the network training losses.

The sparse input implies that for most cells in the input wind velocity channels $(U_{x,in}, U_{y,in})$ the values are undefined since they do not contain a measurement. We test and evaluate two approaches to filling the missing information. The first naïve approach simply zeroes all pixels without a measurement. This results in large gradients of the input for high magnitude wind. The second approach uses the per-channel-average of all measurements as the fill value resulting in a smoother input and propagating the information over the whole domain.

The labels are constructed by stacking the four volumetric channels corresponding to the three-dimensional predicted velocity $(U_{x,out}, U_{y,out}, U_{z,out})$ as well as the TKE at each cell from the CFD ground truth flows.

4.5 Model training

The wind prediction model is an encoder-decoder CNN with skip connections based on the U-Net architecture [119]. The encoder consists of a series of 3D double convolutions followed by max-pooling layers. 3D convolutions and nearest upsampling are applied in the decoder to recover predictions at the same resolution as the input data. Skip connections are utilized to pass information at multiple resolutions from the encoder to the decoder. The output is masked by the terrain, such that all predictions for cells inside the terrain are set to zero. A detailed description of the model is given in Section 5.3.

A scaled version of the mean squared error (MSE) loss is applied to train the model balancing the loss $L(\cdot)$ between the samples and channels:

$$L\left(X, Y, N\right) = \frac{1}{N} \sum_{c} \left(\frac{X_c - Y_c}{\hat{Y}_c} \right)^2 , \tag{5.1}$$

where X is the network prediction, Y the label flow, $\hat{Y}_c$ the label average per channel of the non-terrain cells, and N the number of non-terrain cells. Normalizing the error by the average label value balances the loss for flows of different magnitudes. Without accounting for the number of terrain cells in the loss, a sample with a high ratio of terrain cells would not contribute much to the overall loss. Thus, scaling according to N prevents these cases from being underrepresented in the training.

The model is trained using the Adam optimizer [62] for 3000 epochs except for AD6, where the model after 1000 epochs was chosen as further training showed increasing validation loss, suggesting over-fitting. The initial learning rate of 1.0×10^{-5} is quartered every 700th epoch.

4.6 Measurement campaign datasets

Each of the three measurement campaign datasets that we used for evaluation are publicly available but require some preprocessing to enable direct comparison with our wind prediction outputs. We convert the data from the different file formats for each measurement campaign to the same gridded format that we use to store the CFD solutions. Each experiment provides terrain data as well as wind measurements collected using static masts equipped with airflow sensors at various heights. The terrain is discretized by querying the raw data using bilinear interpolation in the center of the respective cell. The location of each measurement is converted into the cell coordinates.

WindSeer predicts the wind using the measurements from one mast. The measurements are filled into the nearest cell and averaged in case of multiple measurements in one cell. The predictions, which are obtained with trilinear interpolation at the sensing locations, are then compared to the measured wind. CNNs allow for variable input sizes, a trait we exploit to predict at a higher spatial resolution for domains with smaller length scales (see Bolund Hill below) at an increased domain size of $384 \times 384 \times 192$ cells. Since the terrain is represented as a Euclidean distance field, this gives *WindSeer* a sense of the grid resolution and thus the scale of the flow, enabling us to predict the wind at different scales.

Bolund hill The data for the Bolund hill experiment containing time-averaged wind velocities and TKE measurements is publicly available[2]. As Bolund hill exhibits only a small elevation change of 11 m, the default prediction resolution is not sufficient to account for its near-ground measurement locations. As mentioned above, we exploit the multi-scale property of *WindSeer* and increase the resolution of the prediction grid thirty-fold resulting in a domain ∼211 m × ∼211 m wide and ∼73 m tall, giving a corresponding horizontal resolution of 0.55 m and vertical resolution of 0.38 m.

Askervein hill While a digitized version of the Askervein hill topography is available[3] the wind and TKE measurements had to be manually extracted from the field report [142]. We selected 13 runs measuring the turbulent wind, where the data from most towers is provided (in certain runs data is not reported for all towers). The measurements are averaged over one to four hour intervals with varying flow magnitudes and directions. The domain size of 1584 m × 1584 m wide and 552 m tall results in a four-fold resolution increase (horizontal: 4.13 m, vertical: 2.88 m).

Perdigao The Perdigao dataset consists of multiple measurement posts of different heights ranging from 10 m up to 100 m across the valley or along the ridges[4]. We used the five minute averages and tilt corrected measurements that were recorded

---

[2] `https://www.bolund.vindenergi.dtu.dk/blind_comparison`

[3] `https://zenodo.org/record/4095052`

[4] `https://perdigao.fe.up.pt/`

throughout the measurement campaign and we consider data from six different days in our evaluation. The tower positions were not stored with sufficient precision in the dataset requiring us to manually correct the positions. We extracted the topography of the hills from the World Elevation Terrain layer provided by Esri using ArcGIS [5]. Perdigao required the largest prediction domain size, 3168 m × 3168 m wide and 1104 m tall, showcasing the wind prediction performance at double the original resolution (horizontal: 8.25 m, vertical: 5.75 m).

4.7 sUAV flight tests

We used three Multiplex EasyGlider4 airframes equipped with the Pixhawk 4 autopilot [86] using the high quality ADIS16448 IMU and the u-blox M9N GPS module for autonomous navigation. We configured the main height source of our modified PX4 autopilot [87] to the GPS height and use the barometric pressure as a fallback. An extension to the guidance law adjusting the airspeed ensured safety during strong wind conditions [137]. We used a custom designed pitot tube with the SDP31 differential pressure sensor and airflow vanes to enable measuring the 3D wind. Refer to Sections S5 and S6 for more details about the airflow sensing setup and calibration procedure. We used a ground station with QGroundControl to control and navigate the sUAVs. While the default PX4 estimator could be extended to estimate the 3D wind we opted for an offline FPR pipeline using an iterated extended Kalman filter (see a similar problem definition in [90]). The offline FPR pipeline allowed us to get high quality estimates for validating our approach and to adjust the estimation pipeline post flight.

We gathered wind data at three test sites in Switzerland. The first test site at Chasseral is one of the most topographically isolated mountains in Switzerland and is located in the Jura mountains (47° 07’ 38” N, 7° 02’ 47” E, 1548 m above mean sea level (AMSL)). The other test sites are located on the ridges of the Oberalppass (46° 39’ 24” N, 8° 40’ 21” E, 2069 m AMSL) and Gotthardpass (46° 34’ 17” N, 8° 33’ 33” E, 1960 m AMSL) in the Central Swiss Alps. These were chosen to evaluate the prediction performance for domains surrounded by complex terrain. The sUAVs were flown simultaneously in loiter patterns with a radius of 100 m leading to lateral separation between the planes of up to 800 m and measurements in different flow regimes. We planned the flights based on NWP forecasts ensuring good flight (no precipitation, fog or clouds) and stable wind conditions (wind magnitude below the cruise speed of $10\,\mathrm{m\,s^{-1}}$, direction and magnitude constant over multiple hours).

---

[5] `https://www.arcgis.com/home/item.html?id=58a541efc59545e6b7137f961d7de883`

5 Supplementary Materials

5.1 Wind characteristics terminology

The complex wind around terrain has some typical flow regions and we want to establish common terms for some of these regions as shown in Fig. 5.6. The side of the terrain where the wind direction points toward the hill is called the *upwind* side since it typically exhibits mostly rising winds. At the highest point of the terrain (*hill top*) the wind is sped up and higher wind magnitudes can be measured. The *lee side* of the terrain/hill describes the region where the wind direction typically points away from the hill top. This region is highly turbulent and can form multiple modes with completely different characteristics. Under certain conditions the flow can follow the terrain and result in prevailing downwinds. In other conditions a *lee side rotor* can form, where the wind follows a circular motion close to the terrain behind the hill. At a larger scale of multiple kilometers and under very specific conditions, mountain waves with multiple rotors can form [33].

5.2 NWP data as *WindSeer* input

Our initial hypothesis was to train *WindSeer* based on the known high-resolution terrain and predictions from large scale NWP. The Swiss COSMO 1 model provides predictions with a horizontal resolution of 1.1 km [154]. The elevation data used in the NWP models, such as GLOBE [45], is an aggregation of available high-resolution terrain sources (usually the mean or median of the high-resolution data within one cell). This smoothed topography representation neglects smaller scale terrain features and therefore provides meaningful results at a scale of multiple cells/kilometer.

We conducted a test flight to evaluate how well the NWP of one cell matches the wind measured by an sUAV. We measured the wind at one grid point of

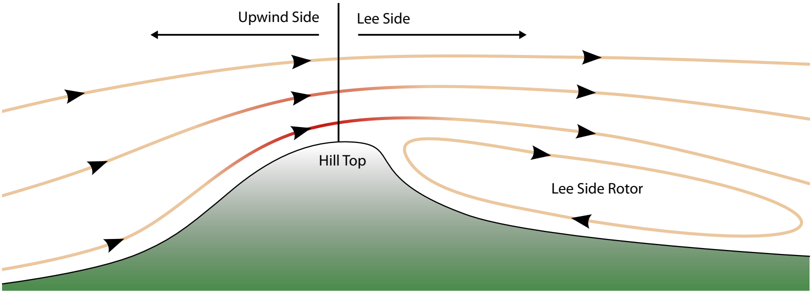


Figure 5.6: Example of wind over a hill resulting in a lee side rotor.

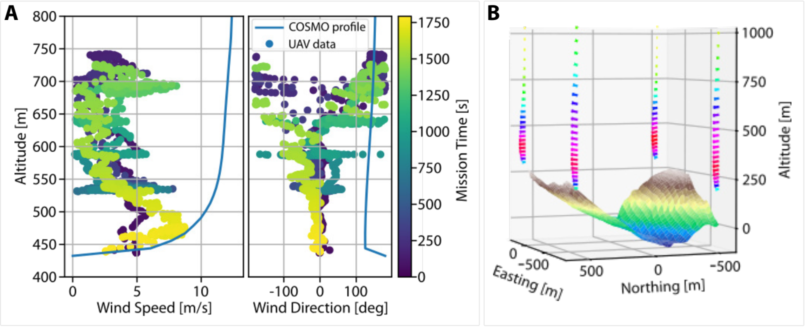


Figure 5.7: (**A**) sUAV wind measurements compared to the NWP along different flight altitudes showing the mismatch in direction and wind speed. (**B**) The coarse terrain representation in the NWP causes offsets in the prediction altitudes to the terrain.

the Swiss COSMO-1 model[6] close to Flüelen (46° 53' 33" N, 8° 36' 45" E, 436 m AMSL). While Flüelen is located within the Swiss Alps, this particular test site is surrounded by flat and smooth terrain within a 1 km radius resulting in a good match between the NWP terrain model and the high-resolution terrain.

As visible in Fig. 5.7 A), the COSMO-1 NWP poorly represents the sUAV data for both the magnitude and wind direction. Obviously in different conditions the NWP might better fit the measurements. However, this implies that, depending on the case, the NWP may or may not be accurate. Thus *WindSeer* needs another, more reliable wind data source for its training data. In addition, the coarse representation of the terrain can result in large altitude offsets of the NWP compared to the actual terrain in the presence of large elevation changes, see Fig. 5.7 B).

The NWP data may provide supplemental information to *WindSeer* if used together with the sparse measurements. However, first the mapping between the NWP data to the actual flow needs to be established. This would be a highly data-driven task, and if that connection is too noisy, *WindSeer* might learn to ignore the NWP input data altogether.

5.3 Model architecture

The *WindSeer* encoder is composed of single 3D convolutions with kernel size 3 and reflection padding to preserve the size. Using skip connections, the information at each depth is relayed to the decoder before utilizing the a max-pooling layer with kernel size of 2 to down-sample the feature map. The original domain size is restored by pairing the information from the skip connection with a nearest-

---

[6] `https://www.meteoschweiz.admin.ch/home/mess-und-prognosesysteme/warn-und-prognosesysteme/cosmo-prognosesysteme.html`

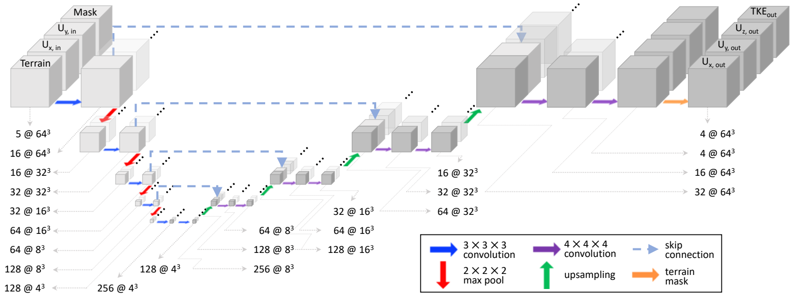


Figure 5.8: *WindSeer* architecture. We utilize a fully convolutional encoder-decoder network with skip connections.

neighbor up-sampled feature map followed by two 3D convolutions with kernel size 4. This decoder structure removes checkerboard artifacts sometimes experienced when using a decoder-encoder CNN [96]. Each convolution, except the final one, is followed by the nonlinear ReLu layer with negative slope 0.1 [83]. Finally, a terrain mask is applied forcing all predictions inside the terrain to zero. The architecture for *WindSeer* with depth 4 is shown in Fig. 5.8.

5.4 Model ablation study

We evaluated the effect of varying certain hyperparameters in the training pipeline on the model performance on a test set of previously unobserved CFD samples. The baseline model parameters are shown in Fig. 5.5, note that these parameters are different from the finalized *WindSeer* version. We used the average error norm over all non-terrain cells averaged over all samples in the test set as our metric to compare the models.

Models trained with different pooling methods (max-pooling (MP), average-pooling (AP), convolution with strides) perform comparably with a slight edge for the pooling methods over the convolution with strides (1.1 % error reduction). The model using only the horizontal wind measurement (NUZ) outperforms the baseline (BL) model, which uses the vertical measurements as well, by 2.6 %. We also varied the input trajectory lengths up to a length of 500 cells (LT). Networks trained on longer trajectories perform 13.6 % better even if they are evaluated exclusively on short trajectories with lengths of up to 50 cells. The results are shown in Fig. 5.9.

Realistic sUAV measurements are subject to noise. We model the sensor noise with a zero-mean Gaussian distribution and the sensor miscalibration with a constant bias. We evaluated the robustness of the model to different levels of such noisy input. Doing so we trained multiple models (BL architecture) with varying

Table 5.5: Baseline hyperparameter set used in the ablation study.

Hyperparameter	**Value**
learning rate	1.0×10^{-5}
learning rate decay	0.25 every 700th epoch
learning epochs	1500
learning batch size	35
max Gaussian noise std	0 %
max bias magnitude	0 %
trajectory min length	3 cells
trajectory max length	50 cells
model depth	4
pooling method	strided convolution
input no measurement value	mean
input use u_z	true

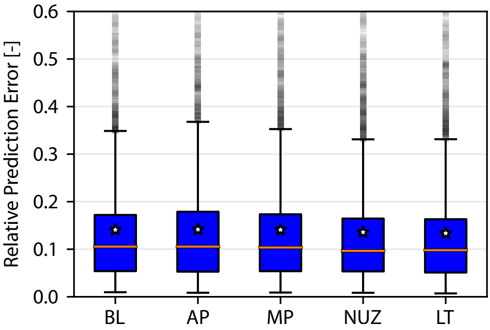


Figure 5.9: Prediction errors of model variations on the test set. The box-plot indicates the 25th, 50th and 75th percentiles and the white star the mean.

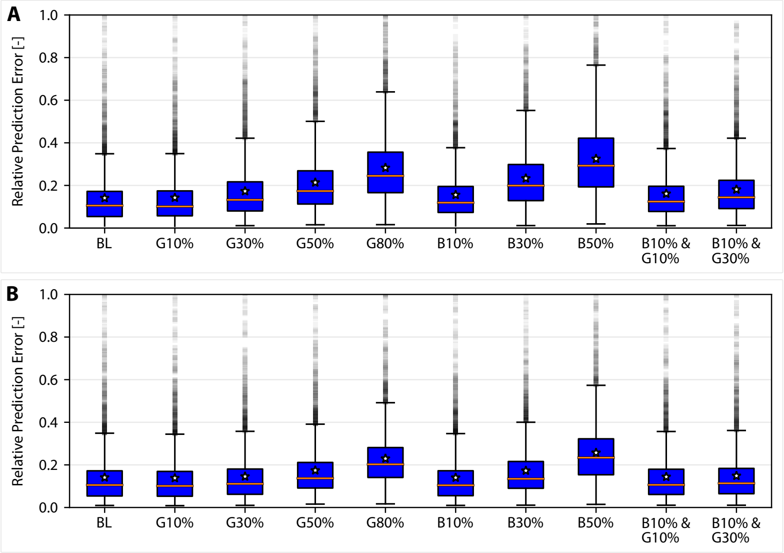


Figure 5.10: (**A**) Models trained with different levels of Gaussian noise and biases and evaluated with the same noise distribution used during training. (**B**) The same models as in (**A**) but evaluated without noise on the input data.

levels of input noise. We varied the standard deviation of the Gaussian noise between 0 % and 80 % of the average flow magnitude of the respective sample; we varied the bias between 0 % and 50 % of the flow magnitude. We then evaluated the models in two ways: First we evaluated them on the test set with the same noise distribution they observed during training. Since this is not a fair comparison, as predicting with high-noise levels is more difficult than low-noise data, we also evaluated all models against perfect data (no noise added). The results of the experiment are displayed in Fig. 5.10 A). In general, higher input noise indicates higher prediction errors, but up to a level of 10 % Gaussian noise and bias we observed similar errors. When evaluating the models on the perfect input we can see that the low-level noise models (up to 10 % bias and 30 % Gaussian noise) perform comparably to the baseline model trained without noise (Fig. 5.10 B)). Thus training models with a too high noise level will also impact the performance when they are provided with perfect input data.

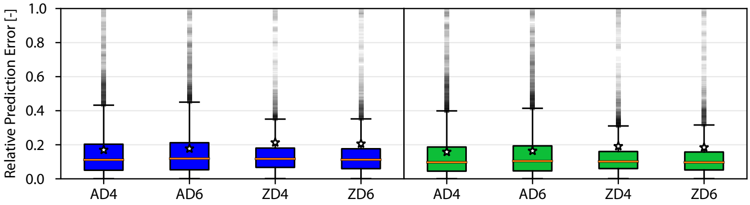


Figure 5.11: TKE relative prediction errors of the *WindSeer* variants on the CFD testset on the full domain (left, blue). Excluding the closest cells to the terrain does not change the prediction error (right, green) in contrast to the velocity errors.

5.5 *WindSeer* TKE prediction results on CFD simulation

We evaluated the TKE prediction performance of the *WindSeer* variants on previously unobserved CFD flow samples. We used the same input noise distribution as observed during training (Gaussian noise and random bias). Fig. 5.11 shows the distribution of the relative prediction errors over the full flow domain on the left side (blue) and excluding the lowest four cells above the terrain on the right side (green). In contrast to the velocity errors the TKE predictions do not significantly change on the reduced prediction volume as the computed TKE values close to the terrain tend to be smoother than the velocity values, thus suffering less from different resolutions from *WindSeer* and the CFD simulations. All the *WindSeer* result in a similar median between 11.2 % to 11.9 %. While the median is lower for the averaging models (AD4: 17.0 %, AD6: 17.7 %) compared to the zero-fill models (ZD4: 21.3 %, ZD6: 20.6 % the latter seem to have fewer outliers as the 75 percentiles are lower. Overall all *WindSeer* variants perform comparable predicting the TKE.

5.6 Design and calibration of sUAV airflow vanes

Two custom-designed wind vanes together with a pitot tube measure the 3D airflow. One vane measures the angle of attack (AoA) and the other one the angle of sideslip (AoS) of the airspeed vector relative to the sUAV body reference frame. During the flight the wings flex due to maneuvers or wind gusts causing measurement error on the vanes that are larger if the vanes are mounted further towards the wingtips. However, the prop wash makes any placement too close to the fuselage invalid since the vanes need to measure the undisturbed free flow. Therefore we place the vanes approximately one quarter of the wing length away from the fuselage (Fig. 5.12 A)).

The vane is 3D printed and balanced using metal weights at the front and back (Fig. 5.12 B)). Small ball bearings at the connection axis ensure little friction in the setup and fast response time to changing wind. A diametric radial magnet is mounted at the end of the connection axis resulting in a changing magnetic field

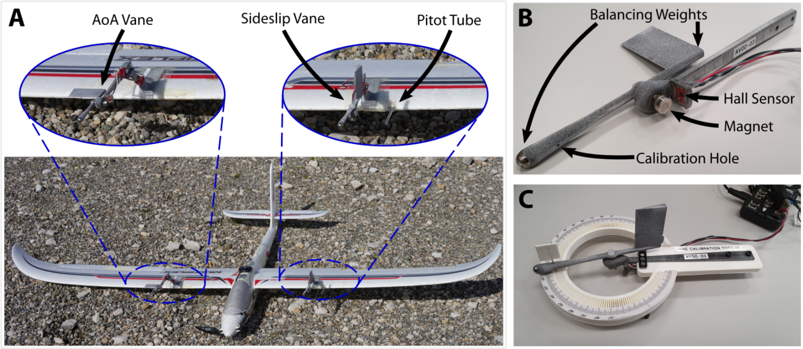


Figure 5.12: (**A**) The arrangement of the airflow sensors on the sUAV. (**B**) Components of the airflow vanes. (**C**) Calibration tool used to determine the mapping from the magnetic flux density to the angle.

(for different angles) that the hall sensor measures.

The calibration tool, shown in Fig. 5.12 C), allowed us to accurately set the vanes to angles with 2° increments, thus gathering accurate data to determine the mapping from the magnetic flux density B. We calibrated each sensor for angles ranging from $-24°$ to $24°$ using a third-order polynomial function. Fig. 5.13 shows the measurements and the resulting fit for one wind vane.

5.7 sUAV airflow sensing calibration

Raw AoA and AoS measurements are subject to mounting errors as well as aerodynamic effects from the fuselage and the wing. We defined calibration functions based on wind tunnel data provided by Heinrich et al. [47], to estimate the true airflow angles based on the sensor measurements $(\alpha_{raw}, \beta_{raw})$ in the relevant range (AoA between $0°$ to $15°$, AoS between $-10°$ to $10°$):

$$\alpha_{off} = p_{\alpha,0} + p_{\alpha,1} \cdot \alpha_{raw} + p_{\alpha,2} \left(\alpha_{raw} + p_{\alpha,3}\right) \left(v_{Aspd} + p_{\alpha,4}\right), \tag{5.2}$$

$$\begin{aligned} \beta_{off} = {} & p_{\beta,0} + p_{\beta,1} \left(v_{Aspd} + p_{\beta,2}\right) \left(1 + \tanh\left(p_{\beta,3}\left(\alpha_{raw} + p_{\beta,4}\right)\right)\right) + \\ & p_{\beta,5} \cdot \beta_{raw} + p_{\beta,6} \cdot \tanh\left(p_{\beta,7}\left(\Phi + p_{\beta,8}\right)\right), \end{aligned} \tag{5.3}$$

where the p variables are free parameters. For the wind tunnel validation, the parameters were estimated by minimising the MSE between the sensor measurements and ground truth airflow angles (orientation of the aircraft using a tunnel-mounted sting, assumed to have very low angular position error). Fitting the wind tunnel data the base functions result in an MSE for the AoA of $0.45°$ and $0.83°$ for the AoS.

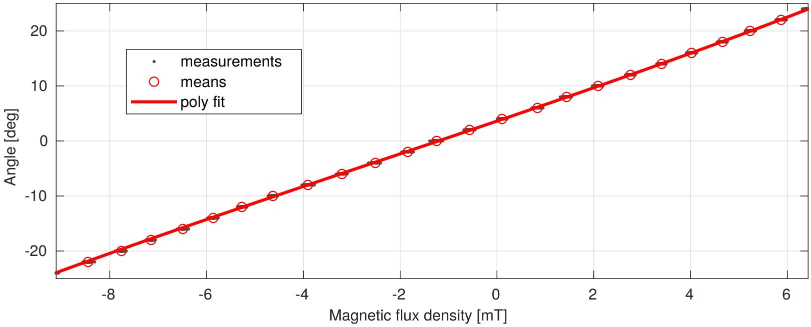


Figure 5.13: Magnetic flux density to angle mapping for one wind vane with the measured data during the calibration procedure.

However, due to variations in mounts and aircraft, this calibration could not be performed for every sensor installation. Thus, we further defined a calibration routine to estimate the parameters of Eq. 5.2, 5.3 based on data gathered during a calibration flight, removing the need to calibrate every sUAV with wind tunnel data. The underlying assumptions that ensure the parameters are observable are that the horizontal wind is piecewise constant and that there is no vertical wind during the calibration flight. We also assume the estimated attitude and global position/velocity are accurate. To cover the different flight regimes our calibration flight consisted of counter-clockwise and clockwise loiter circles of different radii ranging from 30 m to 100 m flown at different airspeeds ($10\,\mathrm{m\,s^{-1}}$ to $16\,\mathrm{m\,s^{-1}}$). We then solved for the calibration parameters and the wind (W_x, W_y) by minimizing the error using a nonlinear least-squares solver in the wind triangle over the full flight:

$$\boldsymbol{e} = R\left(\Phi, \Theta, \Psi\right) \begin{bmatrix} v_{Aspd} \\ v_{Aspd} \cdot \tanh\left(\beta - \beta_{off}\right) + l_{x,\beta} \cdot \omega_z - l_{z,\beta} \cdot \omega_x \\ v_{Aspd} \cdot \tanh\left(\alpha - \alpha_{off}\right) - l_{x,\alpha} \cdot \omega_y + l_{y,\alpha} \cdot \omega_x \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ 0 \end{bmatrix} - \boldsymbol{v_{Gnd}}, \tag{5.4}$$

where $R\left(\Theta, \Phi, \Psi\right)$ is the rotation matrix based on the current attitude, $\boldsymbol{v_{Gnd}}$ the estimated ground speed vector, and $\omega_{(.)}$ the rotational speed around the respective axis. The offset from the vanes to the autopilot origin is denoted by $l_{x,\beta}$, $l_{z,\beta}$, $l_{x,\alpha}$, and $l_{y,\alpha}$.

Using the uncalibrated measurements from the airflow sensors results in strong oscillations of the estimated horizontal wind (strongly correlated to the loiter frequency) and a vertical estimate with a non-zero mean as visible in Fig. 5.14 A). The fit piecewise linear horizontal and zero-mean vertical wind as a result of the airflow calibration pipeline are shown Fig. 5.14 B). Although there is no constraint

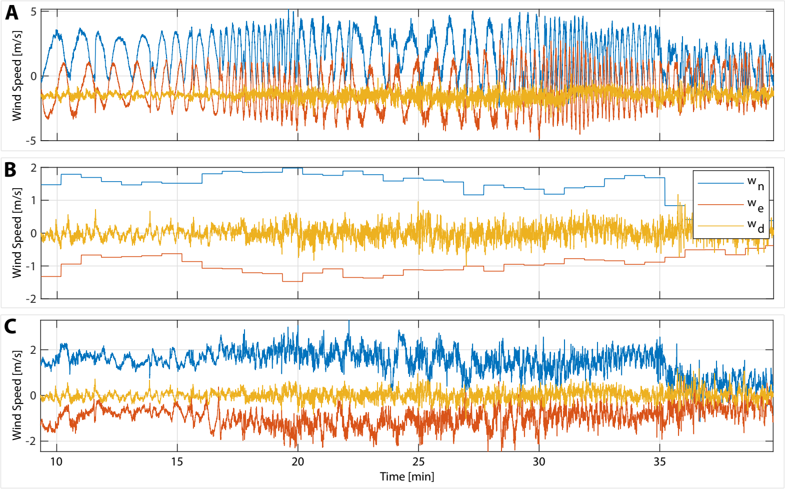


Figure 5.14: (**A**) The wind estimates based on the raw uncalibrated airflow sensor data. (**B**) The piecewise horizontal wind and zero-mean vertical wind as optimized during the calibration procedure. (**C**) The wind estimates after calibrating the airflow sensors.

on the difference between the segments in the horizontal wind, the changes are relatively small. This stable, near-constant wind (magnitude and direction) reflects the forecast and observations made from the ground during the flight. The calibration reduces the estimated oscillations in the wind significantly and results in accurately measuring the zero-mean vertical wind (Fig. 5.14 C)). However, some correlation between the wind estimates and the loiter frequency remain, indicating that the calibration function could still be improved.

5.8 Additional flight test results

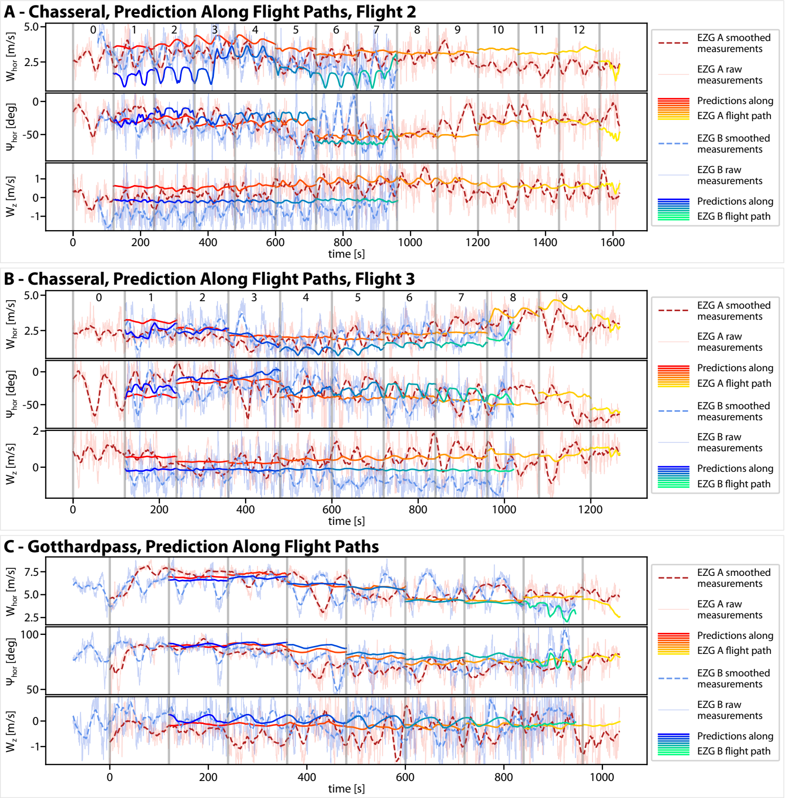


Figure 5.15: Additional sliding window prediction results for the second (**A**) and third (**B**) Chasseral flight and the Gotthardpass flight (**C**).

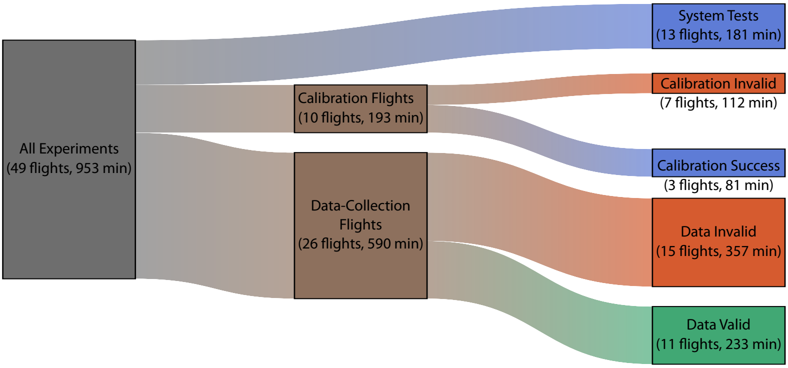


Figure 5.16: Classification of the flight tests conducted with the sUAVs.

5.9 Flight test statistics

In an effort to be transparent about the time required to conduct experiments under real-world conditions and the set-backs that may occur, we also provide statistics for the 'return rate' on experimental flights for this work. Classifying all flight tests we conducted as part of this project (49 flights over 13 test days resulting in 953 min flight time) reveals that only a limited fraction could eventually be used to evaluate the wind prediction approach (22 % of the flights and 24 % of the flight minutes) as shown in Fig. 5.16. The other flights either had the aim of testing and verifying the system setup or the data was unusable due to hardware/software issues. The reported 953 min also only counts the airborne time and not the system setup or travel time. We want to encourage other researchers to conduct experiments in the real world, even though it may require more time and effort.

5.10 Inference time experiments setup

We ran the inference time experiments on an Xavier NX[7], a low power (20 W), light weight (172 g) and small scale (103 mm × 90.5 mm × 34.66 mm) single-board computer that can be carried by a small scale sUAV. We set up the Xavier NX with the Jetpack 4.6 software kit that includes Cuda 10.2 and cuDNN 8.2.1 and installed PyTorch 1.9.0. During the evaluation we ran the Xavier in the maximum power mode using all 6 CPU cores.

---

[7] https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit

5.11 Nomenclature

α	angle of attack
β	sideslip angle
Φ	estimated roll angle
Θ	estimated pitch angle
Ψ	estimated yaw angle
v_{Aspd}	airspeed
$\boldsymbol{v_{Gnd}}$	ground speed vector
$\omega_{(.)}$	measured rotational speed

Acknowledgments We would like to thank our drone pilots David Rohr, Jaeyoung Lim, Jonas Langenegger, Tizian Steiger, and Yves Allenspach for ensuring the safety of our drones during the experiments. The tedious task of setting up the drones and the respective sensing devices was supported by Himmet Kaplan, Jonas Langenegger, Michael Riner, Thomas Mantel, Tizian Steiger, and Yves Allenspach. Last we would like to thank Alexey Dosovitskiy, Debadeepta Dey, and René Ranftl for their valuable inputs throughout the project.

The model training and evaluations were performed on the ETH Zürich Euler cluster.

Funding This project was funded, in part, by Microsoft Swiss Joint Research Center under Contract No. 2019-038 “Project Altair: Infrared Vision and AI Decision-Making for Longer Drone Flights” and the Intel Network on Intelligent Systems.

Data and materials availability The synthetic CFD training data, as well as the sUAV flight measurements will be made available through our dataset server: `https://projects.asl.ethz.ch/datasets/doku.php?id=home`. The code will be available on github: `https://github.com/ethz-asl/WindSeer`

Part B

THERMAL DETECTION

Paper III

MultiPoint: Cross-spectral registration of thermal and optical aerial imagery

Florian Achermann, Andrey Kolobov, Debadeepta Dey, Timo Hinzmann,
Jen Jen Chung, Roland Siegwart, and Nicholas Lawrance

Abstract

While optical cameras are ubiquitous in robotics, some robots can sense the world in several sections of the electromagnetic spectrum simultaneously, which can extend their capabilities in fundamental ways. For instance, many fixed-wing UAVs carry both optical and thermal imaging cameras, potentially allowing them to detect temperature difference-induced atmospheric updrafts, map their locations, and adjust their flight path accordingly to increase their time aloft. A key step for unlocking the potential offered by multi-spectral data is generating consistent, multi-spectral maps of the environment. In this work, we introduce *MultiPoint*, a novel data-driven method for generating interest points and associated descriptors for registering optical and thermal image pairs without knowledge of the relative camera viewpoints. Existing pixel-based alignment methods are accurate but too slow to work in near-real time, while feature-based methods such as SuperPoint are fast but produce poor-quality cross-spectral matches due to interest point instability in thermal images. *MultiPoint* capitalizes on the strengths of both approaches. An *offline* mutual information-based procedure is used to align cross-spectral image pairs from a training set, which are then processed by our generalized multi-spectral homographic adaptation stage to generate highly repeatable interest points that are invariant across viewpoint changes in both spectra. These are used to train a *MultiPoint* deep neural network by exposing this model to both same-spectrum and cross-spectral image pairs. This model is then deployed for fast and accurate *online* interest point detection. We show that *MultiPoint* outperforms existing techniques for feature-based image alignment using a dataset of real-world thermal-optical imagery captured by a UAV during flights in different conditions and release this dataset, the first of its kind.

Published in:
Proceedings of the 2020 Conference on Robot Learning, 2020

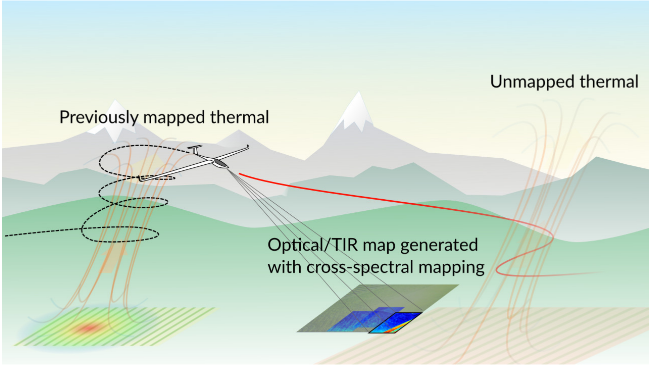

Unmapped thermal
Previously mapped thermal
Optical/TIR map generated with cross-spectral mapping

Figure 6.1: Remote thermal detection and mapping with thermal infrared (TIR) and visible-spectrum cameras.

1 Introduction

Robots have long had access to multi-spectral sensing capabilities that allow them to observe the world well beyond the limitations of human vision. Demonstrated examples include the use of sensing in visible and near-infrared spectra for monitoring crop health [121], ground penetrating radar for landmine detection [124], and others. Since many fixed-wing UAVs carry both optical and thermal infrared (TIR, a.k.a. long-wave infrared) cameras, this potentially allows them to detect regions of warm rising air called *thermals* around them, as illustrated in Fig. 6.1. Thermals arise due to parts of the ground surface absorbing more solar radiation than surrounding areas and heating up the air immediately above. Birds [7] and small UAVs [8, 41, 98] can gain potential energy and extend flight duration thanks to a thermal *if they happen to* fly through one. Detecting and mapping thermals *remotely* would enable UAVs to deliberatively plan their flight paths so as to maximize their time aloft and extend their range. While thermals themselves are invisible in both optical and TIR spectra, their generating regions on the ground can be sensed by a thermal camera on a small UAV. Crucially, generating a TIR map in-flight using standard mapping pipelines is thwarted by the poor performance of existing feature matching techniques [14, 78] on cross-spectral data [28]. To enable use cases such as this, we need a *fast, robust, and accurate* cross-spectral image registration method that would match features in thermal images with their optical counterparts and thereby allow us to leverage existing tools for optical image-based mapping to align and stitch together the corresponding TIR data.

Proposed approach. We present *MultiPoint*, a method that enables cross-spectral image registration by detecting and describing interest points common to both visible- and thermal-spectrum images. *MultiPoint* is fast enough to run on standard UAV hardware during flight, and makes only mild assumptions about the optical and thermal camera installation. The cameras can have different fields of view, may introduce different amounts of warping to the images, and their shutters need not be perfectly synchronized; all *MultiPoint* requires is that the thermal and optical images have a reasonable overlap.

MultiPoint operates via a three-stage pipeline by (1) using a base detector to label interest points, (2) robustifying the detector via homographic adaptation, i.e., generating multiple warped versions of an image to recognize only those interest points that can be consistently identified from different viewpoints, and (3) training a network that jointly identifies interest points and generates associated descriptors. Compared to SuperPoint [32], a similarly structured method for visible-spectrum features, *MultiPoint* differs in three critical aspects that make it successful at cross-spectral registration. In stage (1), for identifying interest points in both optical and thermal images, *MultiPoint* uses SURF features rather than SuperPoint's MagicPoint base detector, as the latter performs poorly at identifying consistent features in unstructured images like those of agricultural and forested terrain. In stage (2), *MultiPoint* generalizes homographic adaptation [32] to the multi-spectral setting by warping optical-thermal image pairs to produce an interest point detector that is consistent not only across viewpoint variations but also across spectra. In stage (3), *MultiPoint* trains a joint interest point detector and descriptor model using all three types of image pairs (optical-optical, thermal-thermal, and optical-thermal), with the pairs matched offline using a mutual information (MI)-maximization approach. In empirical evaluation on real-world data, we show that, thanks to these techniques, *MultiPoint* significantly outperforms existing approaches at cross-spectral feature detection, description, and image registration.

Related work. Image alignment techniques can be roughly split into pixel-based and feature-based approaches (*a more detailed treatment of related work is in the Supplement*). Pixel-based approaches perform pixel-to-pixel comparisons between images and solve for the optimal alignment, e.g., by maximizing MI. These methods work well in medical applications for aligning CT, PET and MRI scans [84], and recent improvements have further increased their accuracy in multi-spectral settings [131]. However, their computational cost is on the order of seconds for pairwise image alignments, while we are targeting real-time applications that need to operate at approximately 5 Hz. Phase correlation methods are faster, but perform poorly when attempting to align images from multiple spectra [29, 64, 141].

Feature-based methods align images by first identifying distinct matching regions (features) and then performing alignment based only on those features. Unfortunately, classic visual features such as SIFT [78] and SURF [14] struggle when faced with multi-spectral image alignment [28] due to the non-linear pixel intensity variations that exist between images from different spectra. To address this, several methods including log-Gabor histogram descriptor (LGHD) use feature descriptors based on region information rather than pixel information [4–6]. This improves performance over standard descriptors, but the average precision values are still only around 0.24. Furthermore, evaluating over regions loses many of the computational benefits of feature-based methods, with LGHD requiring on the order of seconds to perform alignment. Even with efficiency improvements such as multi-spectral feature descriptor (MFD) [95], these methods still do not run at real-time speeds.

Contributions. In summary, our contributions are as follows:

- We propose *MultiPoint* a method for training a deep neural network capable of performing both interest point identification and descriptor generation for cross-spectral image registration that is fast and accurate enough for real-time image alignment and mapping.
- We present a dataset collected from a UAV with a pair of downward-facing RGB and thermal cameras over 10 flights in different conditions. This is the first dataset of this kind, to our knowledge, and includes a set of aligned cross-spectral image pairs that can be used to train a model such as the one we evaluate in the experiments.

2 *MultiPoint* – Cross-spectral interest point detection and description

MultiPoint is a learning framework for generating labeled interest points with descriptors that are consistent between images coming from visible and thermal spectra as well as from different viewpoints. A trained *MultiPoint* network takes as input a pair of images from the same or different spectra and overlapping fields of view (but unknown relative camera poses), and returns a list of interest points and corresponding feature descriptors from both images. The resulting interest points can be matched between images and used either to estimate the relative homography between them (image alignment), or to create a sparse map of optimized multi-spectral feature locations and camera poses in 3D space with the help of a visual mapping framework.

As previously mentioned, training this network consists of three stages and requires a dataset of aligned optical and thermal image pairs. In the rest of the paper, we refer to each pair as a *multi-spectral image pair*. Details on how we collected and processed this dataset from flight data are provided in Section 3. In this section, we outline the intuition behind each of the three stages of *MultiPoint* and describe in detail how each of them works. Our open-source implementation of *MultiPoint* and the training and testing datasets are available at `https://github.com/ethz-asl/multipoint`.

2.1 Stage 1: Interest point label generation

Interest points are salient features in images that can be repeatably identified in multiple images under changes in viewpoint, lighting, and, in our case, across multiple spectra. The goal of *MultiPoint's* first stage is to produce a crude interest point identification mechanism that subsequent stages will bootstrap from and improve upon. *In particular, we would like the sets of optical and thermal interest points produced by this step to have a significant overlap, so that we can use them to match images from different spectra.*

A state-of-the-art approach for aligning *visible-spectrum* images, SuperPoint [32], trains a base detector called MagicPoint for a similar purpose. It is trained using a synthetic dataset constructed by projecting synthetic 3D scenes containing cuboids, checkerboards, and line segments to 2D. Since the corners of such shapes are naturally stable features, they are used as ground truth targets to train an interest point detector. After training MagicPoint on the same synthetic data and refining on MS COCO 14 [76] and multi-spectral data, we observed that in the latter case the interest points produced by the resulting detector cluster around edges in an image. We suspect that the synthetic dataset does not represent the multi-spectral data well, causing the transfer from the synthetic to the real domain to fail.

Our key insight for addressing this challenge is that, counterintuitively, a *non-data-driven* detector can be more consistent at generating interest points across multimodal data distributions covering conventional (optical) and unstructured and otherwise unusual images, e.g., thermal ones. Accordingly, we propose using classical detectors like SURF or SIFT instead of MagicPoint for preliminary cross-spectral interest point identification. A caveat, however, is that, unlike MagicPoint, SURF and SIFT return the locations of individual candidate interest points instead of a heatmap. This makes them very prone to small pixel-wise errors in the detected interest points. To circumvent this problem, we first construct a binary heatmap using interest points predicted by SURF and SIFT and then smooth it using a Gaussian filter.

As shown in Fig. 6.2, the label sets generated by the SURF/SIFT base detectors after multi-spectral homographic adaptation do not exhibit clustering around strong edges, as opposed to those generated by the MagicPoint base detector.

2.2 Stage 2: Multi-spectral homographic adaptation

Although Stage 1 is designed to generate reasonable interest point candidates, beyond the heuristic use of SIFT/SURF to identify these candidates it takes no

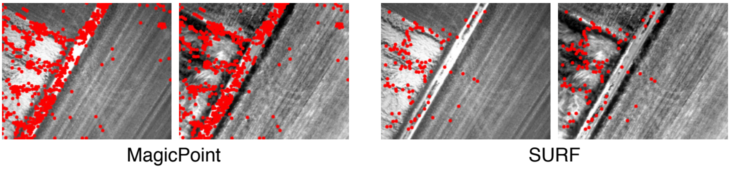


Figure 6.2: A comparison of different exported labels as a result of multi-spectral homographic adaption. The learned detector results in clustered interest points while the pipeline based on the SURF detector spreads interest points more uniformly.

explicit measures to ensure that the identified points are consistent across spectra. Stage 2 improves on Stage 1's output by zeroing in on the cross-spectrally consistent subset of candidate points by employing a generalized version of a technique called *homographic adaptation* [32].

Homographic adaptation is a form of data augmentation that effectively simulates viewpoint changes by sampling and applying multiple homographies to a single image to generate a set of warped views. Interest points (identified with the detector from the previous step) can be mapped between the warped images using the known homographies, allowing the network to learn to recognize points that can be consistently identified after warping. In existing work, homographic adaptation is used to ensure feature consistency across images from a single, visible spectrum [32].

In this work, we generalize homographic adaptation to the multi-spectral domain, modifying it to boost the cross-spectral consistency of interest point detections. To do so, we apply homographic adaptation to *multi-spectral image pairs* instead of individual frames in a process depicted in Fig. 6.3. For a pre-aligned multi-spectral image pair, an ideal interest point detector would return the same set of locations for both images. Accordingly, we sample homographies like the original homographic adaptation does, but use each random homography to warp both images, which yields a new multi-spectral pair of aligned images with effectively a new viewpoint. Next, the detector from Stage 1 is applied to both images to obtain a heatmap pair. The intersection of the heatmap pair is determined by pixel/element-wise multiplication, represented by the $\odot$ operator. Finally, the resulting heatmap is warped back into the original frame and aggregated across the N_h randomly sampled homographies. To generate "good" homographies, we follow the original homographic adaptation's approach of decomposing the transformation into a random scale, translation, in-plane rotation, and perspective distortion and sample those values from uniform distributions with pre-determined ranges.

The result is a multi-spectrum-aware scoring function $\hat{F}$ that takes as input an aligned cross-spectral image pair and returns an aggregated heatmap for generating consistent cross-spectral interest point labels for use in Stage 3:

$$\hat{F}\left(I_o; I_t; f\right) = \frac{1}{N_h} \sum_{i=1}^{N_h} H_i^{-1}\left(f\left(H_i\left(I_o\right)\right) \odot f\left(H_i\left(I_t\right)\right)\right), \tag{6.1}$$

where I_o and I_t represent the optical and thermal images respectively, f is the base detector, and H is a randomly sampled homography[1].

2.3 Stage 3: Joint detector and descriptor training

While Stages 1 and 2 could by themselves serve as an interest point detector, running them is far too slow for real-time use. Stage 3 of *MultiPoint* distills them

---

[1] We use the notation introduced in DeTone et al. [32], where $H(I)$ denotes warping the entire image I with H and Hx represents applying the homography H to the interest points x.

into a deep neural network (DNN) that, given a pair of *unaligned* images, can accurately identify cross-spectrally consistent interest points in them along with their descriptors in a fraction of a second.

We train this model using image pairs related with a known homography labeled with interest point locations generated by the multi-spectral homographic adaptation. To get such a pair, we randomly sample either a multi- or same-spectrum pair of aligned images and warp one of the images using a random but known homography. This results in the network seeing 50 % multi-spectrum pairs and 50 % same-spectrum (thermal-thermal or visible-visible) pairs during training.

The loss function is composed of a detector loss – a fully-convolutional cross entropy loss separately evaluated on each image, and a descriptor loss – a hinge loss on point correspondences. The two losses are balanced using a manually determined weighting parameter. Overall, the loss function is identical to the one used to train SuperPoint[32], but we lowered the threshold parameter for the homography-induced correspondence between the (h, w) cell and the (h', w') cell:

$$s_{hwh'w'} = \begin{cases} 1, \text{ if } \left\| \widehat{H} p_{hw} - p_{h'w'} \right\| \leq 4 \\ 0, \text{ otherwise.} \end{cases} \tag{6.2}$$

The center of a cell in the warped frame is represented by $p_{h'w'}$, and $\widehat{H} p_{hw}$ equals warping the cell centers from the unwarped into the warped frame. We observed a more distinctive descriptor response by decreasing this threshold from the original 8 to 4. We not only saw this effect when training *MultiPoint* with multi-spectral image pairs but also when retraining SuperPoint with optical-only pairs on the MS COCO 14 dataset.

3 Multi-spectral aerial image dataset

3.1 Data collection

To train the network on representative multi-spectral data, we collected data from an uncrewed aerial vehicle (UAV) and compiled a dataset of aligned multi-spectral aerial images. The images were taken using a fixed-wing UAV equipped with two downward-facing cameras, one visual (UI-5261SE Rev. 4 with a 16 mm focal length lens) and one TIR (FLIR TAU2 19 mm, spectral band 7.5–13.5 µm). The thermal camera captures images at 5 Hz and triggers the optical camera at the same rate, resulting in an average time offset of 63 ms determined by Kalibr [85], which was used to calibrate the cameras using a pinhole radial-tangential camera model. The aircraft was flown at altitudes from 80 to 150 m AGL, above a mix of farmed and lightly forested terrain in Switzerland (see Fig. 6.4). In total, 25647 image pairs were captured in ten different flights over two days with take-off times ranging from 9 a.m. to 3 p.m., resulting in varying thermal landscapes. We observed temperature changes up to 30 °C between the early morning and afternoon flights.

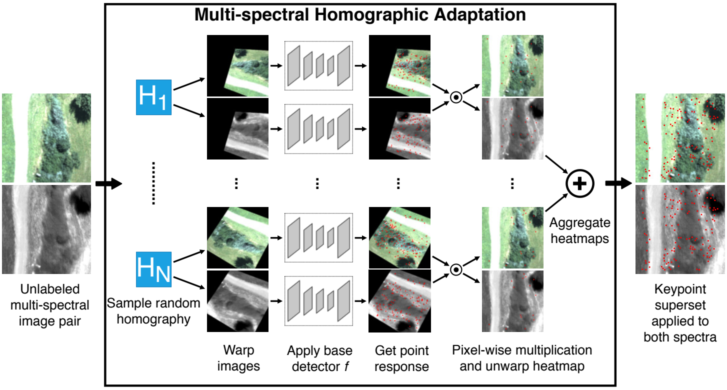

Multi-spectral Homographic Adaptation
Unlabeled multi-spectral image pair
H_1
H_N
Sample random homography
Warp images
Apply base detector *f*
Get point response
Pixel-wise multiplication and unwarp heatmap
Aggregate heatmaps
Keypoint superset applied to both spectra

Figure 6.3: Multi-spectral homographic adaptation procedure for boosting the cross-spectral consistency of the interest point detector.

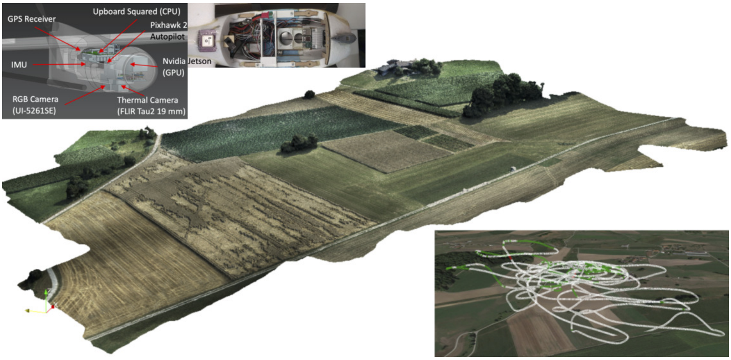


Figure 6.4: Mixed agricultural region where data were collected, reconstructed offline from visible-spectrum images using Pix4D (`https://www.pix4d.com/`). Top insets show the flight hardware. Bottom inset shows the path taken during one flight overlaid on the Google Earth (`https://www.google.com/earth/`) image. Green indicates regions of thermal updrafts and red indicates regions of downdrafts.

3.2 Offline multi-spectral feature matching

To provide labeled data for training our network, we require multi-spectral image pairs with known transformations (planar homographies) between the images. However, raw images from the aerial dataset can have varying relative transformations between the images due to the trigger time offset, exposure times, and different motions within that time frame (see Fig. 6.5). During the data collection flights we regularly observed roll rates around 20 deg/s which leads, together with the average time offset of 63 ms, to a pixel error of 25.5 px in the thermal image frame (20 px/deg). To calculate the 'ground-truth' homography between multi-spectral image pairs, we developed a pipeline based on the MI score. We also tested other multi-spectral matching approaches including LGHD but found that MI produced the most consistent results. At this stage, we were not concerned with running time since this dataset was generated to provide training data for the proposed *MultiPoint* approach.

First, we performed limited pre-processing on the raw images. RGB images were converted to greyscale and normalized. Raw thermal images are single-channel, 14-bit images where each pixel is the absolute temperature, representing a temperature range of $-40\,^{\circ}\mathrm{C}$ to $160\,^{\circ}\mathrm{C}$. We normalized each thermal image to the range between the 1st and 99th percentile of the raw temperature data to remove large temperature outliers and increase contrast.

Next, we used an optimizer to solve for the relative homography between images using the MI score as the cost function [152]. In our setup, the thermal field of view (FoV) is a subset of the optical FoV, so we solved for the homography from the optical to the thermal frame maximising the overlap. This yielded a resolution of 512×640 for each image in each aligned image pair. A (fixed) initial estimate of the relative homography was calculated from the known camera calibration and used as a starting point for the optimizer. Our method used the Nelder-Mead solver [93] from the Python SciPy package [153]. Computing the MI score requires binning the data and then computing a 2D-histogram. We chose twice the number of bins for the thermal compared to the visible spectrum to match the resolution of the pixel values. We performed alignment with three different visible (b_v) and thermal (b_t) bin number pairs in parallel $(b_v, b_t) \in \{(32, 64), (100, 200), (256, 512)\}$. The best match was selected using the MI score. We found using these settings produced the most consistent results as image quality varied with lighting, exposure, and camera orientation. Each optimization took approximately 20 s on a single CPU thread.

The image pairs from the MI method were filtered for poor matches in a two-step process. First, bad matches were automatically rejected using hand-tuned thresholds for the changes in the MI-score and homography. Second, all remaining image pairs were manually checked for visually acceptable alignment with the optimized homography estimate. Finally, 13731 out of 25647 image pairs were accepted (53.54 %). See Fig. 6.5 for an example of an accepted image pair pre- and post-alignment.

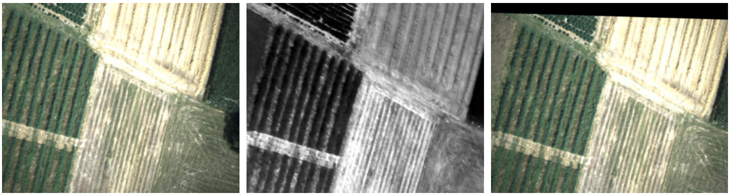


Figure 6.5: *Left*: Optical image warped with the homography determined from the camera calibration. *Middle*: TIR image. *Right*: cropped and aligned optical image using the optimized homography found with MI.

4 Results

MultiPoint is intended to provide interest points and corresponding descriptors for multi-spectral image pairs with different unknown viewpoints. Although the ultimate goal is to incorporate MutiPoint into a feature-based mapping framework for reconstructing dense aligned optical and thermal maps from aerial imagery, here we focus on traditional interest point metrics in order to demonstrate the core capabilities of *MultiPoint* versus existing detection and description methods. We show the following properties of *MultiPoint* on multi-spectral image pairs with different viewpoints:

1. Interest points are detected in the same locations (repeatable).
2. Descriptors provide distinct and correct matches (descriptor precision and matching scores).
3. Using standard feature-based image alignment with *MultiPoint* interest points results in accurate estimates of the relative homography between images (homography estimation).

4.1 Experimental setup

Interest point label generation. We used the implementations of the SURF and SIFT detectors provided by OpenCV. We evaluated different label sets, where we varied the Hessian threshold, C_1, for SURF and the number of retained features for SIFT. Our multi-spectral dataset has a large variety of textures where SURF often struggled to detect any interest point locations for a given detection threshold. In cases where the detector would return less than 50 interest points, we reran the detector with a lower threshold C_2 to mitigate this issue and still generate some interest point labels. The SIFT detector did not require this two-step process and was directly used as the base detector, as it would always return the n best interest points according to the local contrast. The final set of labels to train *MultiPoint* were generated using the SURF base detector ($C_1 = 1500$, $C_2 = 300$). We set the filter size of the Gaussian filter K_{gb} to 3 and used $N_h = 100$ random homographies for the multi-spectral homographic adaptation.

***MultiPoint* training.** The training of *MultiPoint* was done using PyTorch [104]. We used the Adam solver with default parameters to train the model for 3000 epochs with a batch size of 32 and a learning rate of 0.001. We used photometric augmentation, specifically applying motion blur, illumination changes, contrast changes, and additive shades, plus random Gaussian and speckle noise. Additionally, as an augmentation method, as well as allowing for a batch size of 32, we randomly sampled 240x320 patches out of the full resolution images (512x640).

We partitioned the multi-spectral dataset from Section 3 into training and test sets across flights. We used a total of seven flights (9340 image pairs) for training and three flights (4391 image pairs) as test data to assess the performance of the models. All flights are over similar and potentially overlapping regions, but on different days and under different lighting conditions.

4.2 *MultiPoint* detector and descriptor performance

We assessed the detector and descriptor performance of all models under comparison on the test set that contains previously unseen multi-spectral image pairs. We compared *MultiPoint*, SURF, SIFT, LGHD, and SuperPoint, for which we used the weights released by MagicLeap[2]. We used the default OpenCV implementations for SURF and SIFT and a custom Python implementation of LGHD. For each image pair, at the resolution of 512×640, we computed descriptors and interest points and then evaluated with the same set of metrics as in DeTone et al. [32].

Repeatability (Rep.) – the ratio of interest points detected in both images (where interest points are detected at the same reprojected location in both images within a pixel threshold $\delta = 4$) to the total number of detections [126] – is a measure of interest point stability. We also report the average number of interest point detections (N_{kp}). Detecting too many points slows down the subsequent matching steps, and N_{kp} tends to be positively correlated with repeatability since more total points are matched within the threshold but these may not be correct matches. The nearest-neighbour mean average precision (NN mAP) is the area under the precision-recall curve, and is a measure of the discriminating power of the descriptors. Matching score (M. Score) is the ratio of true positive matches over the total number of matches (match precision) and thus measures the combined performance of the interest point detections and descriptors.

Homography estimation is used to determine if the correct matches would be sufficient for accurate image alignment. An alignment is considered ‘correct’ if all four reprojected corners of the warped image using the estimated homography lie within a pixel threshold (ϵ) of the corners of the true homography projection. We evaluated homography estimation using three different thresholds: $\epsilon = 2, 5$, and 10 pixels. We used OpenCV implementations for brute-force (ℓ_2-norm) matching (`cv2.BFMatcher`) and to estimate the homography between matched interest points

---

[2] https://github.com/magicleap/SuperPointPretrainedNetwork

	Detector Met.		Descriptor Met.		Homography Estimation		
	Rep.	N_{kp}	NN mAP	M. Score	$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$
SURF	0.195	523	0.002	0.012	0.003	0.013	0.022
SIFT	0.264	624	0.036	0.041	0.027	0.139	0.203
LGHD	0.162	599	0.002	0.005	0.005	0.035	0.062
SuperPoint	0.163	255	0.071	0.032	0.015	0.089	0.148
MultiPoint	**0.281**	424	**0.271**	**0.134**	**0.125**	**0.507**	**0.667**

Table 6.1: Detector and descriptor metrics on the test set with random homographies applied. *MultiPoint* outperforms all baseline methods due to a significantly increased descriptor performance.

that minimizes back-projection error (`cv2.findHomography`).

The performance metrics of the different models can be seen in Table 6.1. *MultiPoint* clearly outperforms all baseline methods, boosting the descriptor metrics by a factor of 4. This results in superior homography estimation abilities. Since LGHD is not viewpoint-invariant, it performs considerably worse on estimating the homography the larger the warp between the images. A similar trend can be noted for MagicLeap's SuperPoint model. This may be related to the magnitude of warps sampled during their training process versus the ones used for *MultiPoint*. This reveals a general drawback of learned descriptors, which tend to be less effective if test data is not represented well by training data. However, a preliminary evaluation of *MultiPoint* on the MS COCO 14 test set shows similar performance to SuperPoint, indicating that *MultiPoint* generalises well across different images and landscapes. A qualitative comparison of the different methods is shown in Fig. 6.6.

4.3 Timing

We evaluated the running time of the pipeline on flight-grade hardware such as NVIDIA Xavier NX. On the test hardware, a mixed-precision forward pass of *MultiPoint* for a single image pair requires, on average, 112 ms. Non-maximum suppression requires 26 ms and interpolation for interest points takes 10 ms. The total running time of the detector and descriptor pipeline is 148 ms (6.75 Hz), meeting our target performance of 5 Hz.

5 Conclusion

We presented *MultiPoint*, a DNN and a method of training it capable of predicting interest point locations and descriptors for cross-spectral image registration. Additionally, we introduced a novel dataset for training *MultiPoint* that consists of aligned multi-spectral image pairs collected from a UAV, as well as a pipeline for creating consistent cross-spectral interest point labels.

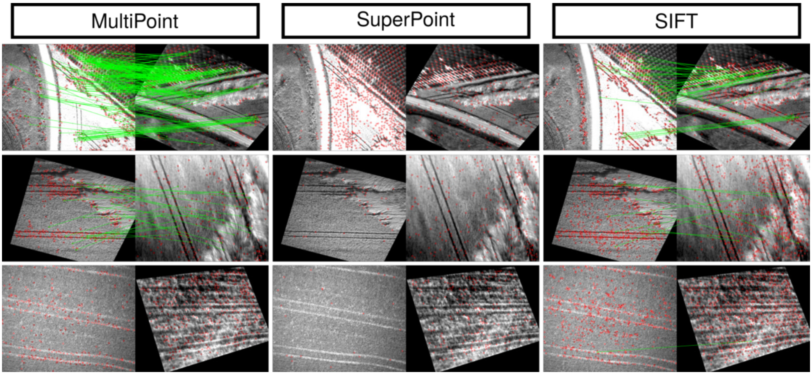


Figure 6.6: Qualitative results on the multi-spectral dataset. Correct matches, with a threshold of 4 pixels, are highlighted in green. Both SIFT and *MultiPoint* can generate correct matches if there is enough texture present in the image pair, but *MultiPoint* does so at a higher rate. SuperPoint struggles in all three examples because of the large warp between the images, and all three methods break down when cross-spectral differences are too dramatic *(bottom row)*.

In future work we plan to extend *MultiPoint* by: 1) exploring alternative model structures, especially smaller networks, to further improve prediction performance or inference times, and 2) incorporating *MultiPoint* into an online mapping framework to create consistent multi-spectral maps [48, 127].

Acknowledgments This research was funded by the Microsoft Swiss Joint Research Center. We would also like to thank Thomas Mantel for his assistance in the assembly of the sensor pod used for data collection and Felix Graule for his inputs on alternative matching techniques.

6 Appendix

6.1 Related Work

The goal of cross-spectral image alignment is to find the spatial transformation that relates two images of the same subject (overlapping fields of view) taken in different spectra usually with different cameras. This can be particularly challenging because images can appear quite different in different spectra, with some features having inverted responses, or not being visible in both images. Image alignment techniques can be roughly split into pixel-based approaches and feature-based approaches. Pixel-based methods use a pixel-to-pixel metric to evaluate the difference between images combined with an optimization scheme to solve for the optimal alignment. Such methods for performing multi-spectral image alignment include the maximization of MI, which has been shown to work well in medical applications for cross-spectral alignment, in examples such as CT, PET and MRI scans [84]. More recently, Shen et al. [131] proposed a new matching cost to better handle the dramatic structure inconsistencies and gradient variations observed in multi-spectral and multi-modal images, showing alignment accuracy improvements over MI and other state-of-the-art methods. However, the difficulty with these approaches is their computational cost, which achieves pairwise image alignments in the order of seconds, while we are targeting real-time applications that need to operate in the range of 5 Hz. On the other hand, phase correlation methods using the frequency domain representation of images provide the computational speedups needed for our targeted application. However, despite working well for visual image alignment, these methods perform poorly when attempting to align multi-spectral images [29, 64, 141].

Feature-based methods offer efficient and rapid matching by first detecting distinct regions (features) in images and then performing alignment based only on those features. Unfortunately, classic visual features such as SIFT [78] and SURF [14] struggle when faced with multi-spectral image alignment [28]. These feature descriptors operate over pixel gradients, thus the observed performance degradation is attributed to the non-linear pixel intensity variations that exist between multi-spectral images. To address this, Aguilera et al. proposed several feature descriptors based on region information rather than pixel information [4–6]. For example, the LGHD method is based on the distribution of high frequency components in a region around a point of interest. Results for matching TIR to visual images show definite improvement over other standard descriptors; however, the overall performance is still far from satisfactory, with average precision values around 0.24. Furthermore, evaluating over regions loses many of the computational benefits of feature-based methods with LGHD still taking on the order of seconds to perform alignment. Although the similar MFD proposed by Nunes and Pádua [95] was able to halve the LGHD computation time, these methods still do not provide the real-time image alignment solutions we seek that would enable online multi-spectral aerial mapping.

6.2 *MultiPoint* Hyperparameter Study

We evaluated how hyperparameter and model choices in the *MultiPoint* training pipeline affect the performance of the trained model. First, we show the performance of alternative interest point label methods and their parameters compared to the MagicPoint detector trained with synthetic shape data. Second, we investigate how showing the network different combinations of cross-spectral or single-spectrum image pairs during training effects final performance. Finally, we show variations in the *MultiPoint* architecture. The results for these variations are summarised in Table 6.4, and each model change is described below.

Interest Point Labels. We generated 10 different interest point label sets using the SURF or SIFT base detector, varying the detector parameters C_1 and

Label	C_1	C_2	K_{gb}	$N_{kp,Min}$	$N_{kp,Mean}$	$N_{kp,Max}$
SURF1	400	100	5	16	623	3852
SURF2	1500	300	3	0	133	2548
SURF3	800	200	3	0	215	2715
SURF4	100	50	0	0	75	749
SURF5	50	50	3	31	916	3092
SIFT1	1000	-	3	0	160	482
SIFT2	2000	-	3	0	359	1021
SIFT3	4000	-	3	0	695	2094
SIFT4	4000	-	5	0	947	4109
SIFT5	4000	-	0	0	122	1027

Table 6.2: Detector settings and interest point statistics for the different labels used to train *MultiPoint*. $N_{kp,Mean}$ denotes the average number of interest points per image on the dataset and $N_{kp,Min}/N_{kp,Max}$ represent the minimum/maximum respectively.

C_2, as well as the filter kernel size K_{gb}. An overview of the parameter choices and the resulting label statistics, where N_{kp} represents the number of interest points per image, can be found in Table 6.2.

Image Pair Composition. We studied the effect of always showing *MultiPoint* cross-spectral image pairs or randomly sampling cross-spectral and same-spectrum image pairs ('Randomized Pairs') during Stage 3 (joint detector and descriptor training). The latter resulted in the model seeing 50 % cross-spectral pairs and 50 % same-spectrum (thermal-thermal or visible-visible) pairs during training.

Descriptor size. We explored a range of descriptor sizes to understand the trade-off between complexity and performance. Allowing larger descriptor sizes has the potential to provide a richer descriptor space at the cost of requiring additional model parameters.

Multiple encoder heads. We evaluated the effect of modifying the network structure proposed by SuperPoint by using multiple encoding heads, one per spectrum, instead of a single encoder. The goal of the multi-headed architecture was to determine if having independent encoder heads for each spectrum would improve performance, by allowing the network to specialize by spectrum. The network was structured to have two encoder heads that share a common structure but independent parameters, and the relevant encoder head would be selected by the input image spectrum. The interest point and descriptor decoders were still shared (see Fig. 6.7).

Discussion. An overview of the evaluated model configurations is shown in Table 6.3. The MultiPoint1_X variants are a special case where we only varied the interest point label method. Performance metrics of the different *MultiPoint* variants are presented in Table 6.4.

When comparing the different MultiPoint1 models we observed that the interest point labels used for training have a large influence on the overall performance of the model. Choosing the right label leads to 23.9 % more accepted homographies

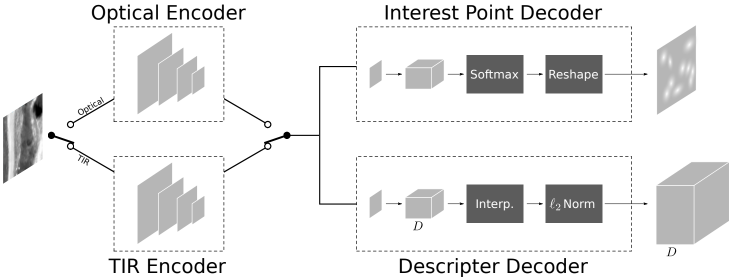


Figure 6.7: *MultiPoint* multiple encoder head architecture. Two different encoders are learned, and the image passes through either the optical or TIR encoder head depending on the image spectrum.

with a threshold $\epsilon = 5$ from the worst, MultiPoint1_SIFT4, to the best model, MultiPoint1_SURF1. However, even MultiPoint1_SIFT4 significantly outperforms the best baseline method in the homography estimation task with similar descriptor metrics. For subsequent experiments (and the final *MultiPoint* model used in the paper) we decided to use the SURF2 label set because it resulted in the highest descriptor metrics while still performing well on the homography estimation task.

We observed that lowering the descriptor size to 64 (MultiPoint2) slightly boosted performance versus the standard SuperPoint descriptor size of 256. Further reduction to 32 (MultiPoint3) led to a minor performance drop leading us to keep descriptor size 64. The model with two encoder heads (MultiPoint5) performed on par with using only a single encoder for both spectra (MultiPoint2). We conclude that a single encoder is expressive enough to detect and describe cross-spectral features.

The model trained with randomized image pairs (MultiPoint2) outperforms the model trained with only cross-spectral pairs (MultiPoint4). We suspect that seeing same-spectrum pairs during training improves the intra-spectral performance which subsequently influences the cross-spectral performance. Training the model with randomized pairs might also help in the case where we still have minor alignment errors in the training image pairs.

Similar to the findings of SuperPoint on the COCO dataset[32] we observed that the repeatability is not a strong indicator of the overall performance of a model. In fact, the two models with the highest repeatability score, MultiPoint1_SIFT4 and MultiPoint1_SIFT5, have the lowest scores in the descriptor metrics and homography estimation. When comparing the repeatability score and the number of interest points (N_{kp}) for every model for every image pair in the test set, we observed a strong positive correlation ($r = 0.791$). This leads us to the conclusion that the models still struggle to find repeatable interest points in both spectra for the image pairs in the cross-spectral dataset (aerial images of a mixed agricultural region).

Model	Labels	Multiple Heads	Randomized Pairs	Descriptor Size
MP1_X	X	No	Yes	256
MP2	SURF2	No	Yes	64
MP3	SURF2	No	Yes	32
MP4	SURF2	No	No	64
MP5	SURF2	Yes	Yes	64

Table 6.3: Configuration for the different *MultiPoint* (MP) models. The MP1_X have the same model parameter but are trained with different interest point labels.

	Detector Met.		Descriptor Met.		Homography Estimation		
	Rep.	N_{kp}	NN mAP	M. Score	$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$
MP1_SURF1	0.412	1254	0.152	0.105	**0.149**	**0.525**	**0.671**
MP1_SURF2	0.286	440	0.245	0.128	0.136	0.510	0.668
MP1_SURF3	0.334	713	0.204	0.116	0.147	0.524	0.670
MP1_SURF4	0.159	275	0.208	0.108	0.126	0.399	0.522
MP1_SURF5	0.515	2197	0.083	0.081	0.132	0.467	0.610
MP1_SIFT1	0.340	750	0.137	0.083	0.144	0.477	0.625
MP1_SIFT2	0.421	1346	0.062	0.058	0.102	0.391	0.537
MP1_SIFT3	0.563	2823	0.027	0.045	0.072	0.322	0.475
MP1_SIFT4	**0.571**	2974	0.026	0.041	0.061	0.286	0.442
MP1_SIFT5	0.311	1281	0.073	0.059	0.083	0.326	0.482
MP2	0.281	424	0.271	0.134	0.125	0.507	0.667
MP3	0.293	451	0.187	0.111	0.140	0.487	0.629
MP4	0.271	473	0.132	0.088	0.114	0.427	0.565
MP5	0.286	391	**0.280**	**0.136**	0.138	0.518	0.670

Table 6.4: Detector and descriptor metrics on the test set with view-point changes for the different *MultiPoint* (MP) variants. MP2 was selected as the best model variant for high performance across all metrics with relatively few interest points.

The final model selected as *MultiPoint* in the paper was the MultiPoint2 variant, with SURF2 interest point labeling, single encoder head, descriptor size 64 and trained with randomized input pairs.

6.3 Additional Experiment without Viewpoint Changes

We conducted a set of experiments with test data consisting only of pairs of pre-aligned images (with no additional homographic warping applied) to assess the performance of generating cross-spectral matches without viewpoint change. This more closely simulates the performance of more traditional alignment approaches such as LGHD, where cross-spectral images are likely to have only mild translations. We compared the MultiPoint2 model to the baseline methods. The results, shown in Table 6.5, show that SuperPoint and LGHD perform significantly better compared to the experiment with viewpoint changes (Table 6.1) but are still

	Detector Met.		Descriptor Met.		Homography Estimation		
	Rep.	N_{kp}	NN mAP	M. Score	$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$
SURF	0.248	630	0.006	0.022	0.028	0.088	0.124
SIFT	0.275	740	0.060	0.049	0.058	0.220	0.286
LGHD	0.151	735	0.088	0.088	0.122	0.576	0.752
SuperPoint	0.190	280	0.327	0.106	0.083	0.371	0.608
MultiPoint2	**0.303**	446	**0.462**	**0.187**	**0.257**	**0.673**	**0.793**

Table 6.5: Detector and descriptor metrics on the test set without viewpoint changes. In this set of experiments LGHD and SuperPoint produce better results but are still outperformed by MultiPoint2.

outperformed by *MultiPoint*.

6.4 Additional Experiment on MS COCO 14

We wanted to evaluate how well the *MultiPoint* model generalises to previously unseen data. To do so we did compare the performane of *MultiPoint*, trained only using the multispectral dataset, to the SuperPoint on the MS COCO 14 dataset. We conducted two sets of experiments where we varied the magnitude of the viewpoint changes. The results are shown in Table 6.6. While for smaller viewpoint changes SuperPoint outperforms *MultiPoint* we observe the opposite for the second experiment. We conclude that *MultiPoint* still performs reasonably well on this set of previously unobserved images. However, both models do not generalise that well to a change in distribution of the viewpoint changes. We infer that it is more important during training to match the correct distribution in the viewpoint changes than having as similar images as possible.

6.5 Additional Qualitative Examples

In Fig. 6.8 we show additional qualitative examples for the cross-spectral matching of *MultiPoint*, SuperPoint, and SIFT.

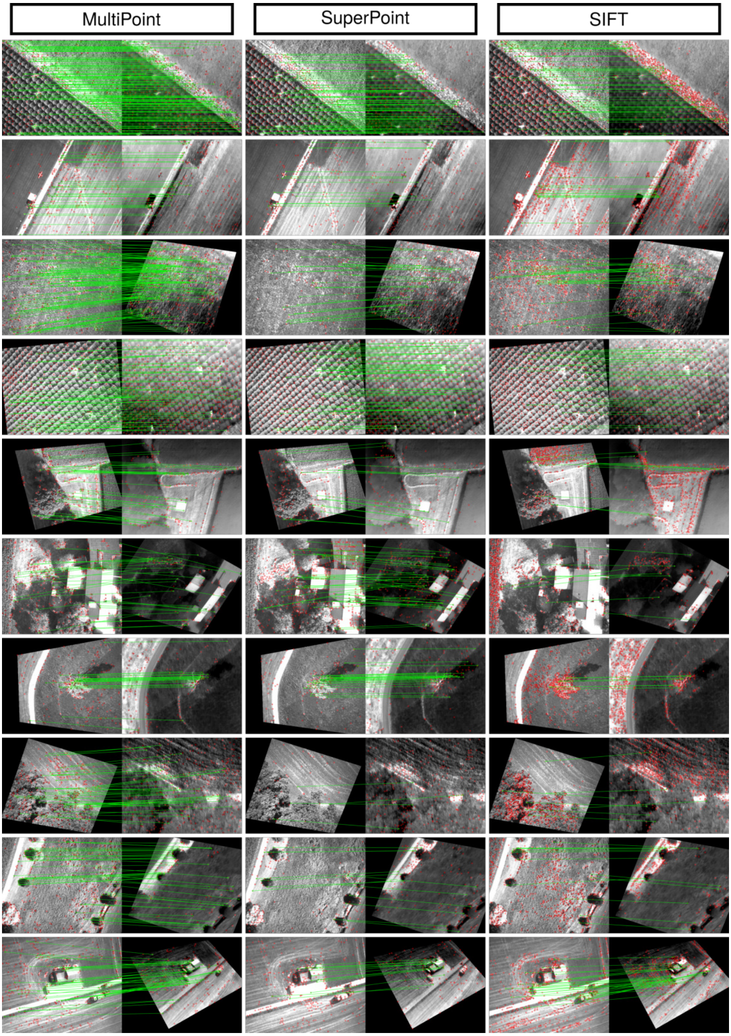


Figure 6.8: Additional qualitative feature matching results on the multi-spectral dataset. As in Fig. 6.6, correct matches with a threshold of four pixels are shown highlighted in green. The first two rows show examples with no viewpoint change (as in Section 6.3).

Small Viewpoint Changes					
	Detector Metrics		Homography Estimation		
	NN mAP	M. Score	$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$
SuperPoint	**0.941**	**0.801**	**0.972**	**0.993**	**0.997**
MultiPoint	0.818	0.465	0.832	0.948	0.974

Large Viewpoint Changes					
	Detector Metrics		Homography Estimation		
	NN mAP	M. Score	$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$
SuperPoint	0.410	0.343	0.506	0.597	0.626
MultiPoint	**0.587**	**0.465**	**0.612**	**0.870**	**0.939**

Table 6.6: Descriptor metrics on the MS COCO 14 test set with small and large viewpoint changes. In this set of experiments it depends on the magnitude of the viewpoint changes which model performs best.

Paper IV

An Outlook on Single Frame Thermal Column Detection using a CNN

Florian Achermann, Julian Haug, Tobias Zumsteg, Andrey Kolobov, Jen Jen Chung, Roland Siegwart, and Nicholas Lawrance

Abstract

Birds and human glider pilots learnt to use thermals, pockets of hot rising air, to extend their flight duration and range. So far, small uncrewed aerial vehicles (sUAVs) are able to exploit thermals if they happen to fly through one but lack the ability to remotely predict such updrafts. Different indirect clues can reveal potential updraft locations such as clouds, temperature differences on the ground, or the color and texture of the ground. However, in this work we focus on directly identifying the updraft column by detecting schlieren, which are small scale optical inhomogeneities due to varying refractive indices in air. Background oriented schlieren (BOS) methods allow visualizing the schlieren with in a static setting, assuming a known background with optical flow. However, these methods are unsuitable to deploy on an sUAV as the background is generally unknown. In this work we demonstrate as a proof of concept that training a convolutional neural network (CNN) to predict the optical flow caused due to schlieren is feasible. We first recorded a set of labelled optical flows in an indoor setup using BOS techniques. We then trained the CNN with a mixture of real and synthetically generated images predicting the two-dimensional optical flow using a single greyscale image. We evaluate our approach on previously unseen flow patterns and background images to show that predicting the optical flow due to schlieren based on a single image is possible.

Published in:
Unpublished, 2022

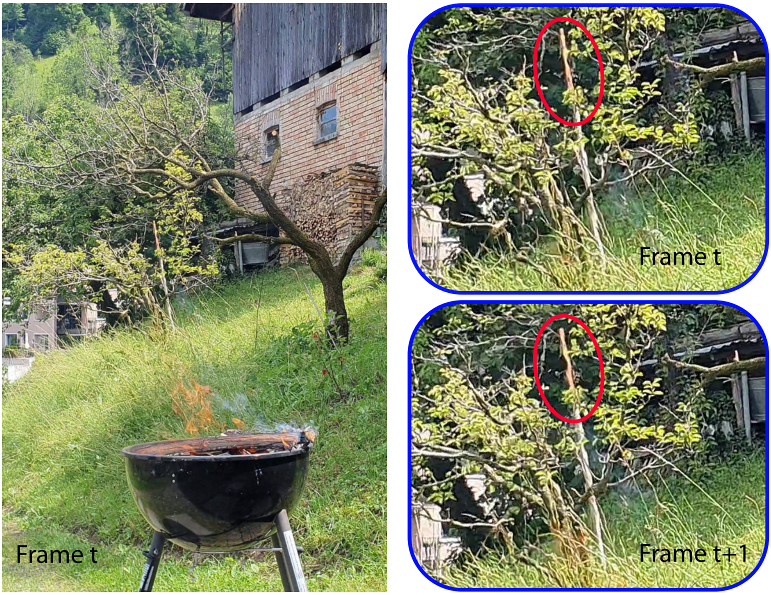


Figure 7.1: The hot air over a fire causes strong refractions (schlieren) visible to the human eye, especially well visible on the wooden pole between two consecutive frames.

1 Introduction

Birds [7, 19, 27, 158] and human glider pilots use thermal updrafts, columns of hot and moist rising air, to extend flight time by gaining potential energy. Temperature gradients are the principal cause for thermal updrafts as the buoyant hot air is less dense than the surrounding colder air. Human gliders guide their search using indirect clues that hint at possible updraft locations[1]. These clues include observing the texture and color of the ground or the landscape in general and its aspect to the sun. For example darker colored areas surrounded with a lighter colored surrounding such as a parking lot surrounded by fields is a likely thermal spot on a sunny day. Other clues include certain types of clouds [139] that form above thermal updrafts, or thermalling birds or other gliders that might reveal currently active thermal updrafts.

Small uncrewed aerial vehicles (sUAVs) have been able to exploit thermal updrafts to prolong flight time if they happen to stumble upon an updraft [8, 41, 98, 114]. Indirect clues, such as a map of previously observed thermals have been

---

[1] `https://xcmag.com/news/zen-and-the-art-of-circles-part-1/`, accessed 29.06.2022.

used to guide sUAVs for more consistent autonomous soaring [30, 31]. A consistent thermal infrared (TIR)-optical map using *MultiPoint*, presented in Paper II, could be used to predict updraft locations based on the ground temperature and color similar to human pilots. However, currently no method exists to directly observe the thermal column of hot rising air remotely from onboard sensors.

The refractive index of the warm and humid rising air in a thermal updraft differs slightly from the ambient air [26]. This difference in refractive index bends the light passing through the updraft column according to Snell's Law [20]. The resulting brightness and color changes are called schlieren, from the German word for 'streaks' [130], as shown in Fig. 7.1. Usually invisible to the human eye for lower temperature differences, these schlieren can be visualized in a controlled lab setup using multiple lenses and a point light source as a shadowgraph measuring the second derivative of the density [129]. Background oriented schlieren (BOS) methods visualize the schlieren with a more generic setup, only requiring one camera but assuming a known, high-texture background [109, 134].

During an sUAV flight the background is generally not known, thus BOS methods may not be directly applicable. BOS methods have been applied in the real-world but multiple passes over the recording area are needed to capture the background [46]. Reference-free BOS techniques, not requiring a reference background, use a stereo setup of high-quality cameras to detect the schlieren but are computationally complex and require high-texture backgrounds [110]. Since we are interested in simply identifying areas with schlieren and do not specifically require the true flow, computing the optical flow between two consecutive frames could serve as a viable approximation. However, movement of the sUAV causes viewpoint changes between the consecutive frames. Perfect alignment of the two frames is impossible in a three-dimensional environment causing artifacts in the optical flow overshadowing the sub-pixel flow of the schlieren.

In this thesis we attempt to detect the optical flow of the schlieren in a **single** optical greyscale image using a convolutional neural network (CNN). We first collect a dataset of observed schlieren in an ideal indoor lab setting using a static camera and high-texture background with different thermal generators. We extract the optical flow due to the schlieren with traditional BOS techniques using the known undistorted background. Then we generate a dataset of imagery containing schlieren with the respective label optical flows. The dataset is composed of real imagery from the indoor setting and synthetically generated images. In the latter case we treat the optical flow of the schlieren as a distortion map that is applied to the images of the Places dataset [171] to simulate the appearance of schlieren on different backgrounds and textures. We train the CNN with a mixture of the real and synthetic images and finally evaluate it on held back data and real-world imagery.

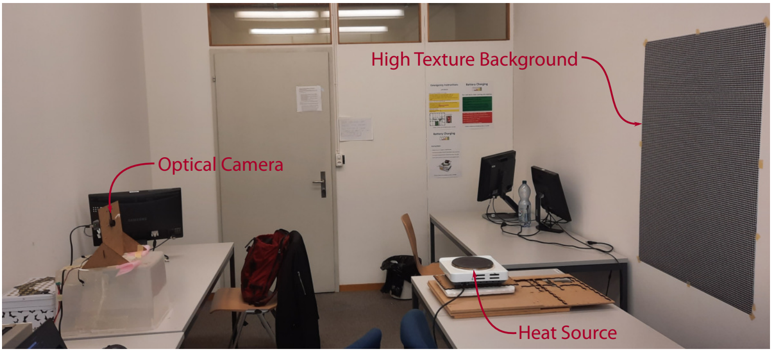


Figure 7.2: The controlled indoor setup to record thermally induced flows. The known high-texture background provides optimal conditions for computing the optical flows caused by hot air above the heat source. The background is a greyscale image with sinusoidal brightness changes in both directions with a wavelength of 8.7 mm, equivalent to 8 px in the captured image.

2 Label Flow Generation

We optimized our indoor setting to generate high-quality labelled data. We used a global shutter optical camera (UI-5261SE Rev. 4 with a 16 mm focal length lens) to capture images at 25 Hz. We used two different heat sources, a larger and a smaller electric heat plate to generate different types of thermal updrafts. The high-texture background allows for accurate computation of the sub-pixel optical flow. The setup is shown in Fig. 7.2.

We evaluated different background patterns and optical flow algorithms to optimally capture the schlieren patterns and reduce the measurement noise. Since even in our controlled setup we don't have access to ground truth schlieren we subjectively evaluated the different approaches by their signal to noise ratio and whether they could generate the sub-pixel optical flow of the schlieren. The greyscale backgrounds we considered were checkerboard patterns of different sizes, Perlin noise [105], and sinusoidal patterns. In our tests, all selected background patterns were sufficient for computing the small scale optical flows. The sinusoidal pattern with a wavelength of 8.7 mm (equivalent to 8 px in the captured image) resulted in the smallest noise levels while still offering the necessary sensitivity. To compute the optical flow of the schlieren we tested the Farnebäck [37], Horn-Schunck [49], Lucas-Kanade [80], PCAFlow [162], and SPyNet [112] algorithms. SPyNet, PCAFlow and Lucas-Kanade failed to compute the small-scale optical flow, and the magnitude of the noise was higher than the flow due to the schlieren. Farnebäck, DeepFlow, and Horn-Schunck were all able to capture the schlieren. However, DeepFlow produced an overly smooth image and failed to pick up some of the small scale details. Farnebäck exhibited the highest noise levels of these three algorithms. Finally we selected the Horn-Schunck algorithm to compute the label flow data.

By varying the distance between the camera and the heat source while keeping the distance to the background constant and using different heat sources we could record varying shapes of schlieren. Small vibrations or movements of the camera relative to the background during the recordings introduced small optical flow biases in each direction in the order of 0.0 px to 0.2 px. Since only a small portion of the recorded image contained schlieren we could determine the bias by computing the median flow value in each direction and consequently correct the flow values by subtracting the bias.

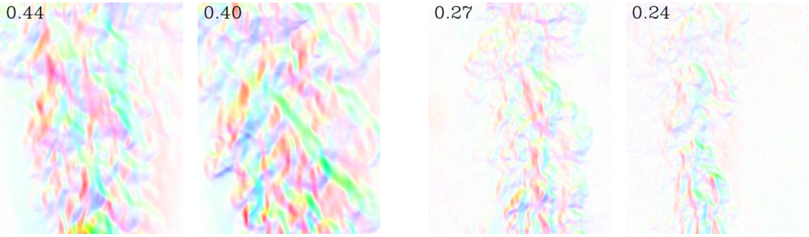
Overall we recorded and processed 8231 frames with a resolution of 1040 px × 820 px. A few examples for the different flow shapes are shown in Fig. 7.3 and color encoding is explained in Fig. 7.4. We can observe that for the same heat source the optical flow magnitudes increase when decreasing the ratio between the distance to the camera and the distance to the background. The flow generated by the large heat plate is almost immediately turbulent while the flow above the small heat plate is laminar until approximately the middle of the captured image. With these four configurations we can train the CNN with a variety of flow magnitudes and patterns.

3 CNN Training for Single Frame Schlieren Detection

We developed the pipeline to train the CNN to predict the optical flow due to schlieren with a single image. The full pipeline is shown in Fig. 7.5 and explained in detail in this chapter.

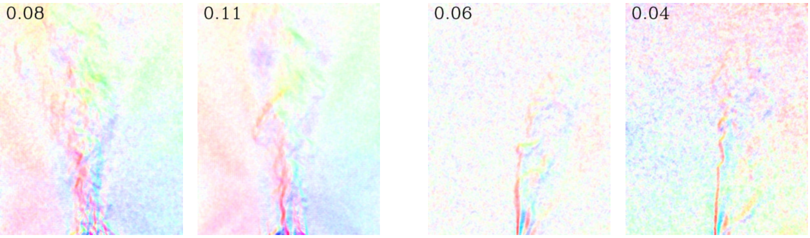
CNN Architecture The CNN architecture used in this work is based on the UNet with minor changes [119]. We have the same depth as the original UNet (four pooling and upsampling layers) and utilize the skip connections but replace the fully connected layers at the bottleneck with convolutions. This change results in a fully convolutional network that can handle inputs of varying sizes above the minimum of 16 px × 16 px. We extend the nonlinearity after each convolution with a BatchNorm layer to stabilize the training by reducing the internal covariance shift [52]. The input to the CNN is a single greyscale image. The pixel values are mapped to $[0, 1]$. The model predicts the two-dimensional optical flow.

Dataset The dataset used to train the CNN contains both real images collected to generate the label flows and synthetically generated samples. The real samples are associated pairs of the recorded high-texture background with schlieren and the corresponding label flow. Training only with these samples would cause the CNN to overfit to this specific background with sinusoidal patterns. This is the reason we introduced the synthetic samples to randomize the observed image textures and features. The high-resolution



(a) Large heat plate close to camera and far from the background

(b) Large heat plate in the middle between the camera and background



(c) Large heat plate close to background and far from the camera

(d) Small heat plate in the middle between the camera and background

Figure 7.3: Examples of different types of label flows recorded in the indoor setting. The maximum flow norm is shown in the top left corner for each image.

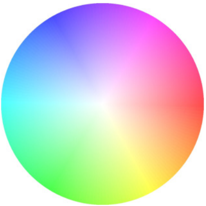


Figure 7.4: Color encoding of the optical flow in the HSV space. The hue is given by the flow direction and the saturation by the flow magnitude.

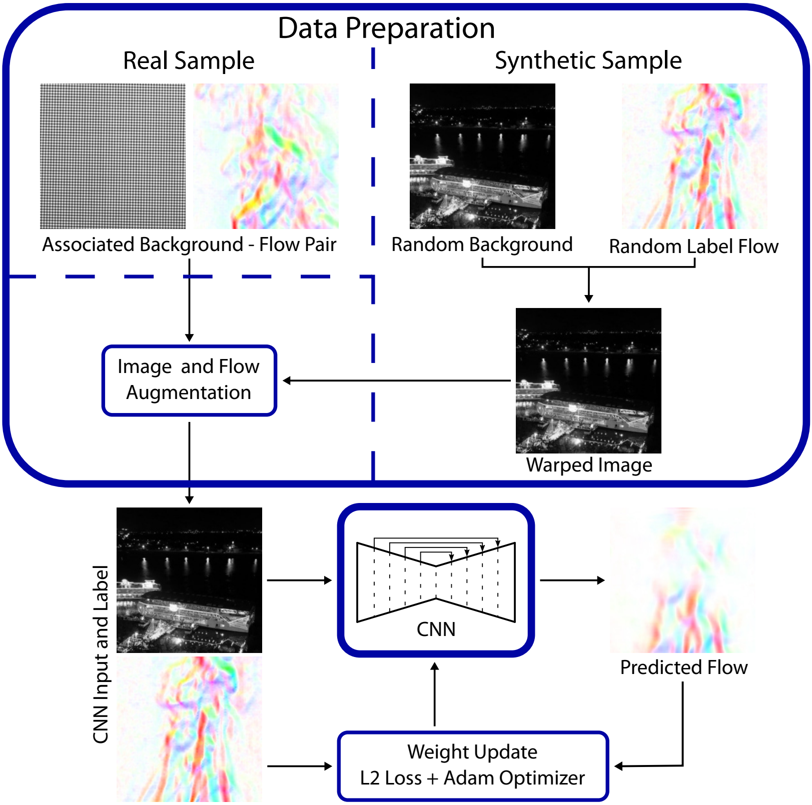


Figure 7.5: Pipeline overview to train the CNN with synthetic and real images. The input data to the CNN are either real samples from the indoor setting or synthetically generated samples with a random background distorted with a label flow.

images from the Places Standard dataset [171] above the minimum size of 480 px × 480 px are used as the background images for the synthetic samples. The places dataset with 1.8 million images contains widely varying textures and scenes.

A synthetic sample is constructed in the following way: Since the background images are smaller than the label flow we randomly crop the flows to the same size as the background image. To randomize the flow directions we flip the cropped label flow in the x- and y-directions with a probability of 0.5. In the final step to generate the synthetic sample we warp the background image according to the flow label. The real sample already contains the schlieren and does not require preprocessing. Finally we randomize the orientation of the image-flow pair that is either a synthetic or real sample by flipping the flow and the image randomly along each axis with a probability of 0.5.

During training, we randomly select the image type. A real image is sampled with probability p_R and a synthetic sample with probability $1 - p_R$. This allows us to balance the training between the real samples with the fixed background texture and the synthetic images with variable background. Figure 7.5 gives an overview of the data preparation and network training pipeline.

Loss We use a regular mean squared error (MSE) loss between the optical flow predicted by the network and the label flow to train the network. In future work we could explore if adding an additional term like the endpoint error (EPE) [51] could improve and guide the training.

Learning Framework Setup The training pipeline is implemented with the PyTorch framework [104] using the Adam optimizer [62] with a batch size of 20 samples to optimize the model weights. We trained the models for 5 million update steps with a learning rate of 1×10^{-4} (500 epochs) and another 5 million steps with a decreased learning rate of 2.5×10^{-5}. We set p_R to 0.5 resulting in the CNN observing an equal amount of synthetic and real samples during training.

Training Curve The training and validation MSE loss over the 1000 training epochs are shown in Fig. 7.6. The training loss is averaged over one epoch composed of 10'000 batches. We computed the validation loss every fifth epoch. After an initial decline in the loss the values stagnate at around 0.01 until roughly epoch 330. Up to this point the model learnt to predict the real flows well but still fails on the synthetic samples. The second decline in MSE loss shows the model learning to predict the synthetic samples as well. This phenomenon was consistent across different training runs, though the epoch of the second decrease was slightly different in each run.

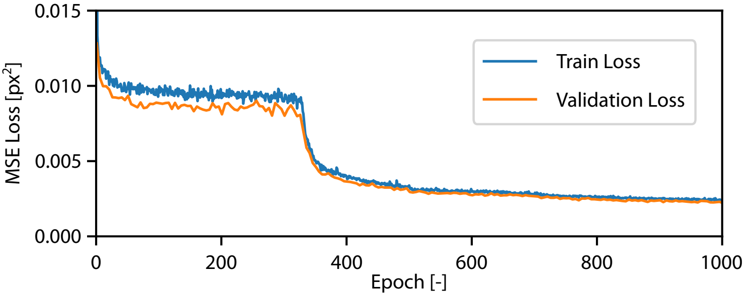


Figure 7.6: The training and validation MSE loss during the CNN training. The training losses are averaged over 10'000 update steps equalling one epoch. The validation loss is computed every fifth epoch.

4 Experiments/Results

We evaluated the trained CNN in a series of experiments with increasing difficulty. In the first set of experiments we used held back data recorded with our indoor setup together with images from the Places Validation Standard dataset to compare the CNN predictions to the optical flow from the BOS algorithm. In the second experiment we tested if the CNN can generalize to different flow patterns. We generated different flow patterns by deflecting the flow with wind or directly with a plate over the heat source. In the last set of experiments we use different real recordings to test if the CNN can detect the schlieren.

Experiment 1: Test Dataset We evaluated the model on the synthetic and real samples separately. On the test dataset with synthetic samples the model predicted the optical flows with an average prediction error of 0.060 px (median: 0.054 px) at an mean label flow magnitude of 0.105 px. The CNN performs better on the real samples with an average error of 0.035 px (median: 0.024 px, mean label magnitude: 0.086 px). In Fig. 7.7 and Fig. 7.8 we show synthetic and real samples with the CNN prediction and the error.

The flows on the real samples are predicted with high quality capturing small scale features and low magnitude flows. The model can accurately separate areas with prevailing schlieren from areas without any optical flow. All these real samples have the same high-texture background which is easier to learn as indicated by the training curve. The synthetic samples with highly varying background images pose a harder challenge for the model. The predicted flows are usually smoother and lack some of the fine details of the label flows but still match the labels well. Especially in the cases where the background lacks texture, the CNN struggles to detect the flow, e.g. image with the wind turbines in Fig. 7.8.

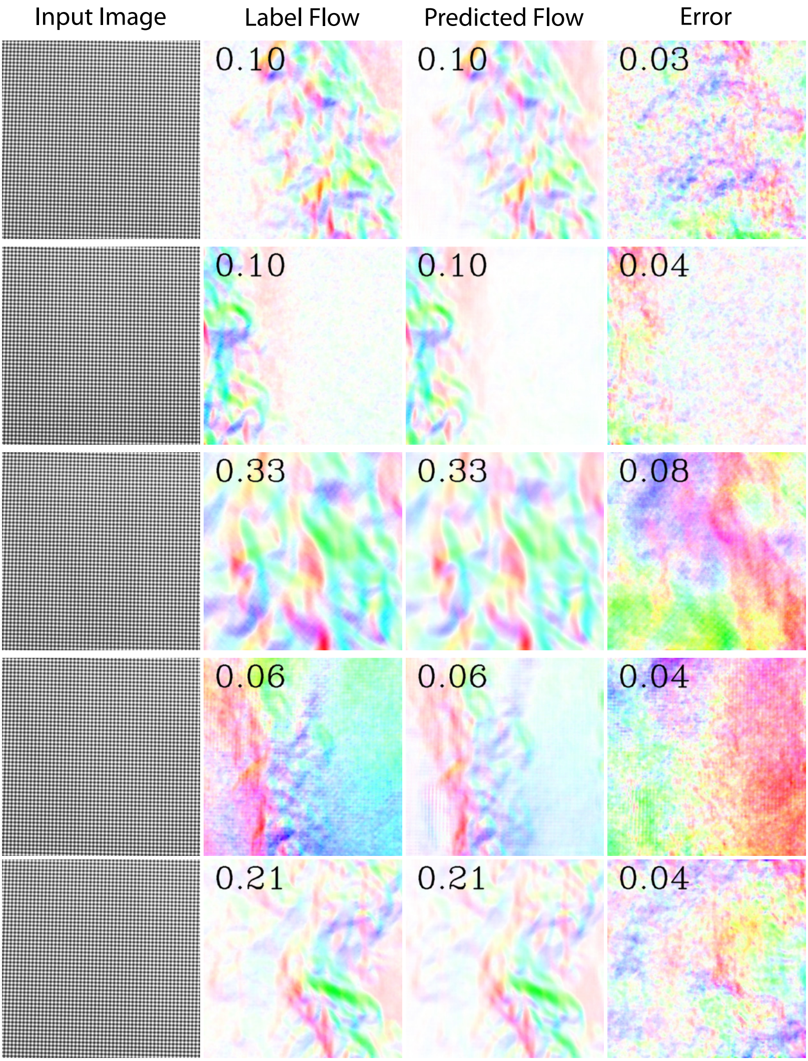


Figure 7.7: Qualitative prediction results of the CNN on previously unobserved real samples. The maximum flow magnitude in pixels for each picture is indicated in the top left corner.

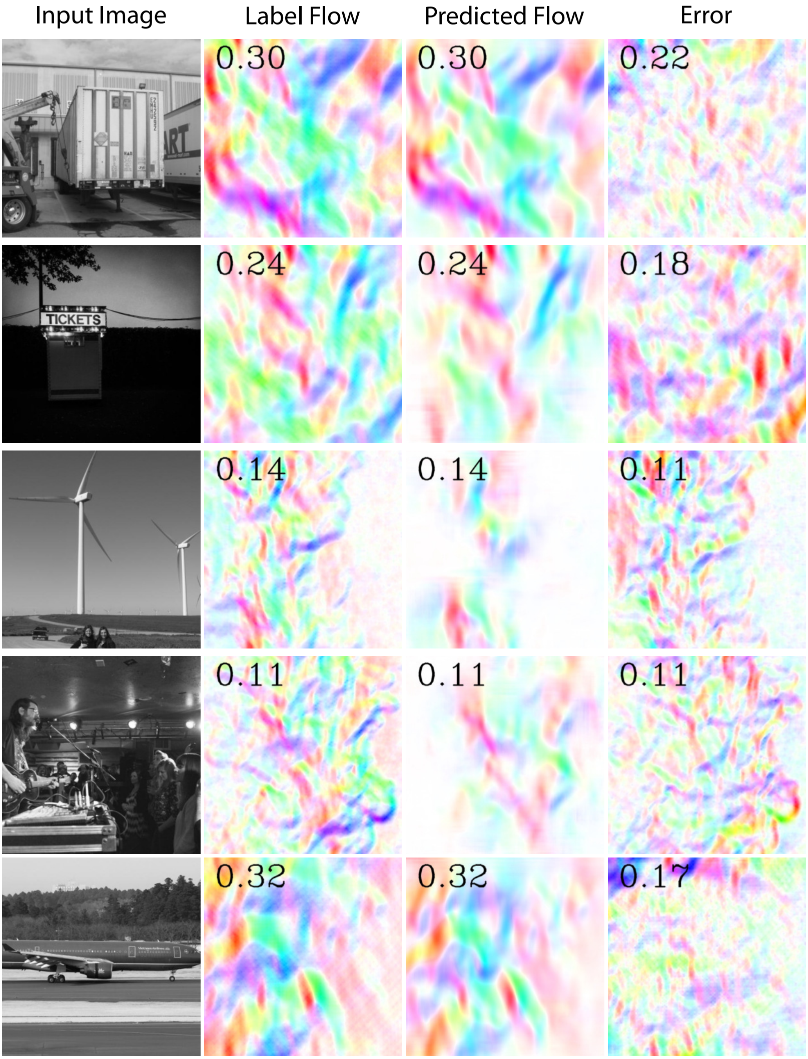


Figure 7.8: Qualitative prediction results of the CNN on previously unobserved synthetic samples. The maximum flow magnitude in pixels for each picture is indicated in the top left corner.

Experiment 2: Flow Shape Variation In our static indoor setup with the high-texture background, as shown in Fig. 7.2, we recorded flows that were influenced by wind or objects above the heat source and compare it to the BOS optical flow. The input images together with the predicted flow and BOS computed flow are presented in Fig. 7.9. It is important to note that this data contains objects and flow patterns that are completely novel to the prediction network and do not appear in any of the training data. Nevertheless, the CNN manages to predict the new optical flow patterns well, such as the vortexes at the plate tip, with minor artifacts at the border of the high-texture background and the objects. The flow quality from the CNN using a single image frame is arguably better than the flows from the two-frame BOS algorithm, which exhibits artifacts at the object borders multiple times larger than the optical flow of the schlieren. Nevertheless, we can still see that the BOS and CNN flow patterns have similar shapes, demonstrating that the CNN generalizes to different flow patterns than those observed during training.

Experiment 3: Natural Backgrounds We recorded images containing schlieren with a natural background caused by a heat plate with a moving camera in indoor and outdoor environments. The moving camera rules out the BOS algorithm providing a reference flow. As shown in Fig. 7.10 the CNN predictions contain flow artifacts with magnitudes in the order of the expected optical flow caused by the schlieren. Therefore, the CNN does not generalize yet to arbitrary environments with natural backgrounds.

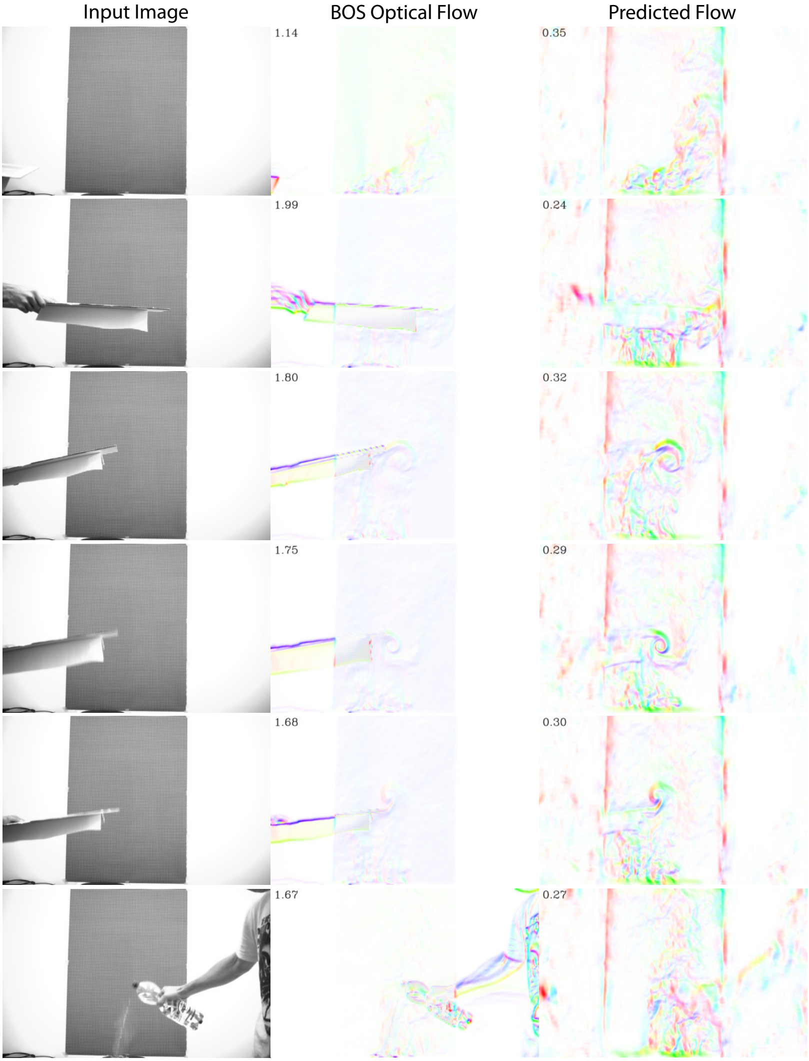


Figure 7.9: Images with schlieren recorded in the indoor setup with different flow patterns than those observed during CNN training. The flows are deflected by objects above the heat plate or sideways wind. For a sequence of frames refer to the Videos 1 and 2 attached to this report.

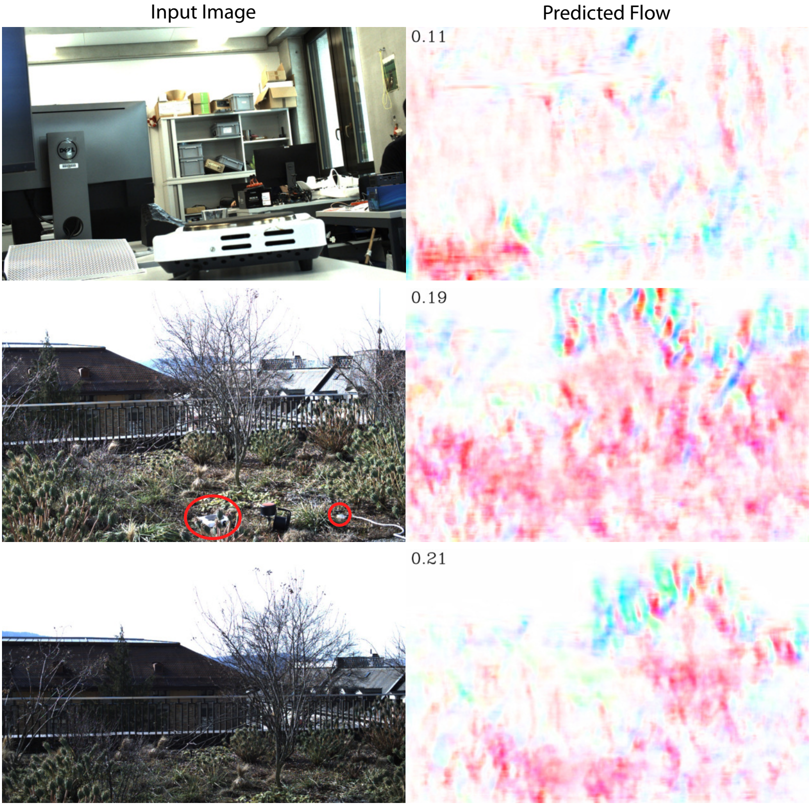


Figure 7.10: Images with schlieren recorded with natural indoor and outdoor backgrounds. The predicted flows contain artifacts due to the backgrounds with magnitudes in the order of the expected optical flow. The locations of the heat plates in the outdoor environment are highlighted in the middle frame.

5 Conclusion and Outlook

In this technical brief we investigated if a CNN can predict the sub-pixel optical flow due to schlieren using only a single greyscale image. We recorded schlieren flow patterns in a controlled static indoor environment with a traditional BOS method. We then trained the CNN with a mixture of real and synthetic samples. The real samples are images captured with the indoor setup and the synthetic samples are composed of random backgrounds distorted with the recorded optical flows. The resulting network learnt to predict the sub-pixel flow patterns well with the random backgrounds from the Places Standard dataset. With the high-texture background from the indoor setup it even predicts previously unseen flow patterns caused by objects or wind deflecting the flows. However, currently it does not generalize to any arbitrary environment with natural backgrounds. The setup in these environments differs from the training images in terms of background texture and different distances to the background and heat source. Future analysis will have to investigate the impact of these changes to the CNN prediction performance.

5.1 Future Work

To extend the promising CNN-based detector from this work to a system capable of remotely detecting thermal columns with natural backgrounds in flight we want to explore the following work packages in the future:

Label Flow Generation Small vibrations in our recording setup cause small optical flow biases. In our next setup we would ensure a more rigid placement of the camera with respect to the background to hopefully reduce the observed biases due to vibrations. Recording the flows over multiple different heat sources of different sizes, temperatures and positions relative to the camera and background could further help to improve generalization to a wider array of schlieren patterns and natural backgrounds.

The noise with respect to the optical flow decreased if the heat source was closer to the camera as seen in Fig. 7.3. So recording the flows in an ideal setup and then simulating different distances with data augmentation could improve the signal-to-noise ratio on the flow labels and enable detecting the low magnitude flows of natural thermals.

Data Augmentation We believe that with improved data handling we can increase the robustness of the optical flow predictions. By randomly rotating the flows we can further diversify the observed flow directions. Thermal sources further away from the camera close to the background essentially result in a smaller optical flow pattern of lower magnitudes. We should be able to replicate this effect by downscaling flows that were recorded closer to the camera.

By adding photogrammetric noise, such as Gaussian noise, salt-and-pepper noise, brightness changes, or contrast changes, and noise on the optical flow we might be able to increase the robustness of the CNN predictions on real images of natural backgrounds that exhibit higher noise levels.

Data Collection Our experiments with the natural background showed that the model currently struggles to generalize to generic environments. We could support the training by collecting in the indoor setting more varying data with respect to the distances to the background and the thermal source but also using different high-texture backgrounds. Together with the data augmentation this will result in the CNN observing more varying data, eventually leading to better generalization.

Alternative Network Architectures We could explore network architectures used in previous work estimating the optical flow between two images to compare the performance to the current UNet-based CNN. SPyNet [112], Flownet [51], DeepFlow [89], or CNN transformer [163] could be viable alternatives to the UNet structure. Sinusoidal activation functions have shown promising results in various tasks such as image representation or solving partial differential equations and could be an interesting direction to explore [133]. So far we have not focused on the run-time and memory requirements of the CNN. The current structure still allows for reasonable inference times on a Xavier NX for smaller images (520 px × 520 px) with 0.3 s and even larger images (1040 px × 1920 px) with 1.3 s. When exploring alternative architectures, attention should be given to the run-time of the networks on sUAV grade hardware such as the Jetson Xavier NX or Jetson Orin NX.

Sample Balancing/Loss Definition Currently we calculate the MSE loss independently in each direction to train the CNN. Introducing additional terms to the loss, such as the EPE or an MSE loss on the magnitude, could help to guide the training. We have flows of different magnitudes in our dataset as shown in Fig. 7.3. Models trained with a standard MSE loss tend to learn higher magnitude flows better and ignore lower magnitudes as such flows do not contribute much to the loss. Normalizing the flow magnitudes of each sample, e.g. with the maximum or mean flow magnitude for each sample, could help to balance the loss between the different samples.

Color Input Image We used a single greyscale image as the input and extracted the label flows from greyscale images. The refraction indices are slightly different with varying wavelengths [26]. A standard optical camera captures the visible light in the range of 400 nm to 700 nm wavelength, thus for each channel the schlieren shapes should be unique. The CNN might be able to use this additional information to predict the optical flow more precisely if we extend the input to the three channel optical image.

Predicting Representation Currently the network predicts the two-dimensional optical flow vector for each image pixel. To locate a thermal updraft column we essentially only need a classifier highlighting pixels inside the thermal column. Therefore we could reduce the output of the CNN to a single channel either predicting only the flow magnitude or even directly a thermal column probability. In both cases the CNN does not have to learn the direction of the flow, possibly leading to faster convergence during training. In certain cases, training a deep neural network (DNN) for multiple tasks simultaneously increases the performance [128]. Therefore, letting the CNN predict the magnitude or two-dimensional flow and the thermal updraft probability could improve the performance.

Updraft Column Triangulation Once we can reliably detect the thermal updraft in a single frame we can triangulate the global position of the updraft column with a moving camera. This would allow us to incorporate these predictions into a planning framework to predict remote thermal locations in flight and enable more consistent thermal soaring.

Bibliography

[1] F. Achermann, N. R. J. Lawrance, R. Ranftl, A. Dosovitskiy, J. J. Chung, and R. Siegwart. Learning to predict the wind for safe aerial vehicle planning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2311–2317, 2019.

[2] F. Achermann, A. Kolobov, D. Dey, T. Hinzmann, J. J. Chung, R. Siegwart, and N. Lawrance. Multipoint: Cross-spectral registration of thermal and optical aerial imagery. In J. Kober, F. Ramos, and C. Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 1746–1760. PMLR, 16–18 Nov 2021.

[3] F. Achermann, T. Stastny, B. Danciu, A. Kolobov, J. J. Chung, R. Siegwart, and N. Lawrance. Windseer: Low-altitude real-time volumetric wind prediction over complex terrain aboard a small uav. *Science Robotics*, under review.

[4] C. Aguilera, F. Barrera, F. Lumbreras, A. D. Sappa, and R. Toledo. Multispectral image feature points. *Sensors*, 12(9):12661–12672, 2012.

[5] C. Aguilera, F. Barrera, A. D. Sappa, and R. Toledo. A novel SIFT-like-based approach for FIR-VS images registration. In *11th International Conference on Quantitative InfraRed Thermography*, pages 1–9, 2012.

[6] C. A. Aguilera, A. D. Sappa, and R. Toledo. LGHD: A feature descriptor for matching across non-linear intensity variations. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 178–181. IEEE, 2015.

[7] Z. Akos, M. Nagy, S. Leven, and T. Vicsek. Thermal soaring flight of birds and unmanned aerial vehicles. *Bioinspiration & Biomimetics*, 5(4), 2010.

[8] M. Allen. Autonomous soaring for improved endurance of a small uninhabitated air vehicle. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*, page 1025, 2005.

[9] M. Allen. Updraft model for development of autonomous soaring uninhabited air vehicles. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, page 1510, 2006.

[10] P. Auf der Maur, B. Djambazi, Y. Haberthür, P. Hörmann, A. Kübler, M. Lustenberger, S. Sigrist, O. Vigen, J. Förster, F. Achermann, E. Hampp, R. K. Katzschmann, and R. Siegwart. Roboa: Construction and evaluation of a steerable vine robot for search and rescue applications. In *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*, pages 15–20. IEEE, 2021.

[11] M. Baldauf, A. Seifert, J. Förstner, D. Majewski, M. Raschendorfer, and T. Reinhardt. Operational convective-scale numerical weather prediction with the COSMO model: Description and sensitivities. *Monthly Weather Review*, 139(12):3887–3905, 2011. doi: 10.1175/MWR-D-10-05013.1.

[12] P. Baqué, E. Remelli, F. Fleuret, and P. Fua. Geodesic convolutional shape optimization. *arXiv preprint arXiv:1802.04016*, 2018.

[13] A. Bauknecht, B. Ewers, C. Wolf, F. Leopold, J. Yin, and M. Raffel. Three-dimensional reconstruction of helicopter blade–tip vortices using a multi-camera bos system. *Experiments in fluids*, 56(1):1–13, 2015.

[14] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[15] A. Bechmann, N. N. Sørensen, J. Berg, J. Mann, and P.-E. Réthoré. The Bolund experiment, part II: blind comparison of microscale flow models. *Boundary-Layer Meteorology*, 141(2):245, 2011.

[16] R. Bencatel, J. Tasso de Sousa, and A. Girard. Atmospheric flow field models applicable for aircraft endurance extension. *Progress in Aerospace Sciences*, 61:1–25, 2013. ISSN 0376-0421. doi: https://doi.org/10.1016/j.paerosci.2013.03.001.

[17] J. Berg, J. Mann, A. Bechmann, M. Courtney, and H. E. Jørgensen. The Bolund experiment, part I: flow over a steep, three-dimensional hill. *Boundary-layer meteorology*, 141(2):219, 2011.

[18] S. Bhatnagar, Y. Afshar, S. Pan, K. Duraisamy, and S. Kaushik. Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64(2):525–545, 2019.

[19] C. M. Bishop, R. J. Spivey, L. A. Hawkes, N. Batbayar, B. Chua, P. B. Frappell, W. K. Milsom, T. Natsagdorj, S. H. Newman, G. R. Scott, J. Y. Takekawa, M. Wikelski, and P. J. Butler. The roller coaster flight strategy of bar-headed geese conserves energy during himalayan migrations. *Science*, 347(6219):250–254, 2015. doi: 10.1126/science.1258732.

[20] M. Born and E. Wolf. *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light.* Elsevier, 2013.

[21] Y. Bühler, M. S. Adams, R. Bösch, and A. Stoffel. Mapping snow depth in alpine terrain with unmanned aerial systems (uass): potential and limitations. *The Cryosphere*, 10(3):1075–1088, 2016.

[22] R. Buizza. *Chaos and Weather Prediction.* ECMWF, 2002.

[23] A. Chakrabarty and J. Langelaan. UAV flight path planning in time varying complex wind-fields. In *2013 American Control Conference*, pages 2568–2574, June 2013. doi: 10.1109/ACC.2013.6580221.

[24] C.-Y. Chang, J. Schmidt, M. Dörenkämper, and B. Stoevesandt. A consistent steady state cfd simulation method for stratified atmospheric boundary layer flows. *Journal of Wind Engineering and Industrial Aerodynamics*, 172:55–67, 2018. ISSN 0167-6105. doi: https://doi.org/10.1016/j.jweia.2017.10.003.

[25] J. J. Chung, N. R. Lawrance, and S. Sukkarieh. Learning to soar: Resource-constrained exploration in reinforcement learning. *The International Journal of Robotics Research*, 34(2):158–172, 2015.

[26] P. E. Ciddor. Refractive index of air: new equations for the visible and near infrared. *Applied Optics*, 35(9):1566–1573, Mar 1996. doi: 10.1364/AO.35.001566.

[27] C. D. Cone. Thermal soaring of birds. *American Scientist*, 50(1):180–209, 1962.

[28] J. Cronje and J. De Villiers. A comparison of image features for registering LWIR and visual images. In *23rd Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, pages 1–8, 2012.

[29] E. De Castro and C. Morandi. Registration of translated and rotated images using finite Fourier transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):700–703, 1987.

[30] N. T. Depenbusch, J. J. Bird, and J. W. Langelaan. The autosoar autonomous soaring aircraft, part 1: Autonomy algorithms. *Journal of Field Robotics*, 35(6):868–889, 2018. doi: https://doi.org/10.1002/rob.21782.

[31] N. T. Depenbusch, J. J. Bird, and J. W. Langelaan. The autosoar autonomous soaring aircraft part 2: Hardware implementation and flight results. *Journal of Field Robotics*, 35(4):435–458, 2018. doi: https://doi.org/10.1002/rob.21747.

[32] D. DeTone, T. Malisiewicz, and A. Rabinovich. Superpoint: Self-supervised interest point detection and description. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 337–349, 2018.

[33] D. R. Durran. *Mountain Waves*, pages 472–492. American Meteorological Society, Boston, MA, 1986. ISBN 978-1-935704-20-1.

[34] S. L. Ellis, M. L. Taylor, M. Schiele, and T. B. Letessier. Influence of altitude on tropical marine habitat classification using imagery from fixed-wing, water-landing uavs. *Remote Sensing in Ecology and Conservation*, 7(1): 50–63, 2021. doi: https://doi.org/10.1002/rse2.160.

[35] J. Elston and E. W. Frew. Unmanned aircraft guidance for penetration of pretornadic storms. *Journal of guidance, control, and dynamics*, 33(1):99–107, 2010.

[36] J. S. Elston, J. Roadman, M. Stachura, B. Argrow, A. Houston, and E. Frew. The tempest unmanned aircraft system for in situ observations of tornadic supercells: design and VORTEX2 flight results. *Journal of Field Robotics*, 28(4):461–483, 2011.

[37] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis*, pages 363–370. Springer, 2003.

[38] Y. R. Fei, C. Batty, E. Grinspun, and C. Zheng. A multi-scale model for simulating liquid-fabric interactions. *ACM Transactions on Graphics (TOG)*, 37(4):51, 2018.

[39] H. J. S. Fernando, J. Mann, J. M. L. M. Palma, J. K. Lundquist, R. J. Barthelmie, M. Belo-Pereira, W. O. J. Brown, F. K. Chow, T. Gerz, C. M. Hocut, P. M. Klein, L. S. Leo, J. C. Matos, S. P. Oncley, S. C. Pryor, L. Bariteau, T. M. Bell, N. Bodini, M. B. Carney, M. S. Courtney, E. D. Creegan, R. Dimitrova, S. Gomes, M. Hagen, J. O. Hyde, S. Kigle, R. Krishnamurthy, J. C. Lopes, L. Mazzaro, J. M. T. Neher, R. Menke, P. Murphy, L. Oswald, S. Otarola-Bustos, A. K. Pattantyus, C. V. Rodrigues, A. Schady, N. Sirin, S. Spuler, E. Svensson, J. Tomaszewski, D. D. Turner, L. van Veen, N. Vasiljević, D. Vassallo, S. Voss, N. Wildmann, and Y. Wang. The perdigão: Peering into microscale details of mountain winds. *Bulletin of the American Meteorological Society*, 100(5):799 – 819, 2019. doi: 10.1175/BAMS-D-17-0227.1.

[40] E. W. Frew, J. Elston, B. Argrow, A. Houston, and E. Rasmussen. Sampling severe local storms and related phenomena: Using unmanned aircraft systems. *IEEE Robotics & Automation Magazine*, 19(1):85–95, 2012.

[41] I. Guilliard, R. J. Rogahn, J. Piavis, and A. Kolobov. Autonomous thermalling as a partially observable markov decision process. In *Robotics: Science and Systems*, 2018.

[42] X. Guo, W. Li, and F. Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 481–490, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939738.

[43] H. Hambly and R. Rajabiun. Rural broadband: Gaps, maps and challenges. *Telematics and Informatics*, 60:101565, 2021. ISSN 0736-5853. doi: https://doi.org/10.1016/j.tele.2021.101565.

[44] F. V. Hansen. Surface roughness lengths. Technical Report ARL-TR-61, Army Research Lab White Sands Missile Range NM, 1993.

[45] D. A. Hastings, P. K. Dunbar, G. M. Elphingstone, M. Bootz, H. Murakami, H. Maruyama, H. Masaharu, P. Holland, J. Payne, N. A. Bryant, et al. The global land one-kilometer base elevation (globe) digital elevation model, version 1.0. *National Oceanic and Atmospheric Administration, National Geophysical Data Center*, 325:80305–3328, 1999.

[46] J. T. Heineck, D. W. Banks, N. T. Smith, E. T. Schairer, P. S. Bean, and T. Robillos. Background-oriented schlieren imaging of supersonic aircraft in flight. *AIAA Journal*, 59(1):11–21, 2021.

[47] G.-A. Heinrich, S. Vogt, N. R. J. Lawrance, T. J. Stastny, and R. Y. Siegwart. In-wing pressure measurements for airspeed and airflow angle estimation and high angle-of-attack flight. *Journal of Guidance, Control, and Dynamics*, 0 (0):1–13, 2021. doi: 10.2514/1.G006412.

[48] T. Hinzmann, J. L. Schönberger, M. Pollefeys, and R. Siegwart. Mapping on the fly: real-time 3d dense reconstruction, digital surface map and incremental orthomosaic generation for unmanned aerial vehicles. In *Field and Service Robotics*, pages 383–396. Springer, 2018.

[49] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[50] Z. Huang, J. Fan, S. Cheng, S. Yi, X. Wang, and H. Li. HMS-Net: Hierarchical Multi-scale Sparsity-invariant Network for Sparse Depth Completion. *arXiv e-prints*, Aug. 2018.

[51] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017.

[52] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[53] G. V. Iungo, F. Viola, U. Ciri, M. A. Rotea, and S. Leonardi. Data-driven RANS for simulations of large wind farms. *Journal of Physics: Conference Series*, 625:012025, jun 2015. doi: 10.1088/1742-6596/625/1/012025.

[54] M. Jaritz, R. de Charette, E. Wirbel, X. Perrotton, and F. Nashashibi. Sparse and Dense Data with CNNs: Depth Completion and Semantic Segmentation. *arXiv e-prints*, Aug. 2018.

[55] H. Jasak, A. Jemcov, v. Tuković, et al. OpenFOAM: A C++ library for complex physics simulations. In *International workshop on coupled methods in numerical dynamics*, pages 1–20. IUC Dubrovnik, Croatia, 2007.

[56] G. Jouvet, Y. Weidmann, E. Van Dongen, M. P. Lüthi, A. Vieli, and J. C. Ryan. High-endurance uav for monitoring calving glaciers: Application to the inglefield bredning and eqip sermia, greenland. *Frontiers in Earth Science*, 7:206, 2019.

[57] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011. doi: 10.1177/0278364911406761.

[58] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[59] A. Kashefi, D. Rempe, and L. J. Guibas. A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries. *Physics of Fluids*, 33(2):027104, 2021.

[60] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, and B. Solenthaler. Deep fluids: A generative network for parameterized fluid simulations. *arXiv preprint arXiv:1806.02071*, 2018.

[61] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, and B. Solenthaler. Deep fluids: A generative network for parameterized fluid simulations. *Computer Graphics Forum*, 38(2):59–70, 2019. doi: https://doi.org/10.1111/cgf.13619.

[62] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference for Learning Representations*, 2015.

[63] H. Ku. Notes on the use of propagation of error formulas. *Journal of Research of the National Bureau of Standards. Section C: Engineering and Instrumentation*, 70C(4):263–273, 1966.

[64] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. In *Proceedings of the 1975 International Conference on Cybernetics and Society*, pages 163–165, 1975.

[65] M. Kulbacki, J. Segen, W. Knieć, R. Klempous, K. Kluwak, J. Nikodem, J. Kulbacka, and A. Serester. Survey of drones for agriculture automation from planting to harvest. In *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*, pages 353–358, 2018.

[66] R. Köhler, C. Schuler, B. Schölkopf, and S. Harmeling. Mask-specific inpainting with deep neural networks. In *German conference on pattern recognition*, pages 523–534, 09 2014. doi: 10.1007/978-3-319-11752-2_43.

[67] J. Kümmerle, T. Hinzmann, A. S. Vempati, and R. Siegwart. Real-time detection and tracking of multiple humans from high bird's-eye views in the visual and infrared spectrum. In *International Symposium on Visual Computing*, volume 10072, pages 545–556, 12 2016. ISBN 978-3-319-50834-4.

[68] L. Ladický, S. Jeong, B. Solenthaler, M. Pollefeys, and M. Gross. Data-driven fluid simulations using regression forests. *ACM Trans. Graph.*, 34(6): 199:1–199:9, Oct. 2015. ISSN 0730-0301. doi: 10.1145/2816795.2818129.

[69] E. Lamptey and D. Serwaa. The use of zipline drones technology for covid-19 samples transportation in ghana. *HighTech and Innovation Journal*, 1(2): 67–71, 2020.

[70] J. W. Langelaan. Gust energy extraction for mini- and micro- uninhabited aerial vehicles. In *Proc. of the 46th AIAA Aerospace Sciences Meeting and Exhibit*, 2008.

[71] J. W. Langelaan, N. Alley, and J. Neidhoefer. Wind field estimation for small unmanned aerial vehicles. *Journal of Guidance, Control, and Dynamics*, 34 (4):1016–1030, 2011.

[72] J. W. Langelaan, J. Spletzer, C. Montella, and J. Grenestedt. Wind field estimation for autonomous dynamic soaring. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 16–22. IEEE, 2012.

[73] B. E. Launder and B. Sharma. Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc. *Letters in heat and mass transfer*, 1(2):131–137, 1974.

[74] N. R. J. Lawrance and S. Sukkarieh. Path planning for autonomous soaring flight in dynamic wind fields. In *2011 IEEE International Conference on Robotics and Automation*, pages 2499–2505, May 2011. doi: 10.1109/ICRA.2011.5979966.

[75] P. Lemme, S. Glenister, and A. Miller. Iridium(r) aeronautical satellite communications. *IEEE Aerospace and Electronic Systems Magazine*, 14(11): 11–16, 1999. doi: 10.1109/62.809197.

[76] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[77] R.-l. Liu, Z.-j. Zhang, Y.-f. Jiao, C.-h. Yang, and W.-j. Zhang. Study on flight performance of propeller-driven uav. *International Journal of Aerospace Engineering*, 2019. doi: 10.1109/ICUAS.2018.8453377.

[78] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157. IEEE, 1999.

[79] K. Lu, N. Barnes, S. Anwar, and L. Zheng. From depth what can you see? depth completion via auxiliary image reconstruction. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11303–11312, 2020.

[80] B. D. Lucas, T. Kanade, et al. *An iterative image registration technique with an application to stereo vision*, volume 81. Vancouver, 1981.

[81] F. Ma and S. Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4796–4803, 2018.

[82] F. Ma, G. V. Cavalheiro, and S. Karaman. Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3288–3295. IEEE, 2019.

[83] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

[84] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. Multimodality image registration by maximization of mutual information. *IEEE Transactions on Medical Imaging*, 16(2):187–198, 1997.

[85] J. Maye, P. Furgale, and R. Siegwart. Self-supervised calibration for robotic systems. *IEEE Intelligent Vehicles Symposium, Proceedings*, 06 2013. doi: 10.1109/IVS.2013.6629513.

[86] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys. Pixhawk: A system for autonomous flight using onboard computer vision. In *2011 IEEE International Conference on Robotics and Automation*, pages 2992–2997. IEEE, 2011.

[87] 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)*, pages 6235–6240. IEEE, 2015.

[88] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[89] L. Mosser, O. Dubrule, and M. J. Blunt. Deepflow: history matching in the space of deep generative models. *arXiv preprint arXiv:1905.05749*, 2019.

[90] J. A. Mulder, Q. P. Chu, J. K. Sridhar, J. H. Breeman, and M. Laban. Non-linear aircraft flight path reconstruction review and new advances. *Progress in Aerospace Sciences*, 35(7):673–726, Oct. 1999. doi: 10.1016/S0376-0421(99)00005-6.

[91] M. Müller, D. Charypar, and M. Gross. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 154–159, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. ISBN 1-58113-659-5.

[92] I. M. Navon. *Data Assimilation for Numerical Weather Prediction: A Review*, pages 21–65. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[93] J. A. Nelder and R. Mead. A Simplex Method for Function Minimization. *The Computer Journal*, 7(4):308–313, 01 1965. ISSN 0010-4620. doi: 10.1093/comjnl/7.4.308.

[94] K. Nirmal, A. G. Sreejith, J. Mathew, M. Sarpotdar, A. Suresh, A. Prakash, M. Safonova, and J. Murthy. Noise modeling and analysis of an IMU-based attitude sensor: improvement of performance by filtering and sensor fusion. *arXiv e-prints*, Aug. 2016.

[95] C. F. Nunes and F. L. Pádua. A local feature descriptor based on log-gabor filters for keypoint matching in multispectral images. *IEEE Geoscience and Remote Sensing Letters*, 14(10):1850–1854, 2017.

[96] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003.

[97] P. Oettershagen, A. Melzer, T. Mantel, K. Rudin, R. Lotz, D. Siebenmann, S. Leutenegger, K. Alexis, and R. Siegwart. A solar-powered hand-launchable uav for low-altitude multi-day continuous flight. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3986–3993, 2015. doi: 10.1109/ICRA.2015.7139756.

[98] P. Oettershagen, A. Melzer, T. Mantel, K. Rudin, T. Stastny, B. Wawrzacz, T. Hinzmann, S. Leutenegger, K. Alexis, and R. Siegwart. Design of small hand-launched solar-powered UAVs: From concept study to a multi-day world endurance record flight. *Journal of Field Robotics*, 34(7):1352–1377, 2017. doi: 10.1002/rob.21717.

[99] P. Oettershagen, T. Stastny, T. Hinzmann, K. Rudin, T. Mantel, A. Melzer, B. Wawrzacz, G. Hitz, and R. Siegwart. Robotic technologies for solar-powered uavs: Fully autonomous updraft-aware aerial sensing for multiday search-and-rescue missions. *Journal of Field Robotics*, 35(4):612–640, 2018. doi: https://doi.org/10.1002/rob.21765.

[100] P. Oettershagen, J. Förster, L. Wirth, G. Hitz, R. Siegwart, and J. Ambühl. Meteorology-aware multi-goal path planning for large-scale inspection missions with solar-powered aircraft. *Journal of Aerospace Information Systems*, 16(10):390–408, 2019.

[101] P. Oettershagen, B. Müller, F. Achermann, and R. Siegwart. Real-time 3d wind field prediction onboard uavs for safe flight in complex terrain. In *2019 IEEE Aerospace Conference*, pages 1–10, 2019. doi: 10.1109/AERO.2019.8742160.

[102] Y. Ouyang, L. A. Vandewalle, L. Chen, P. P. Plehiers, M. R. Dobbelaere, G. J. Heynderickx, G. B. Marin, and K. M. Van Geem. Speeding up turbulent reactive flow simulation via a deep artificial neural network: A methodology study. *Chemical Engineering Journal*, 429:132442, 2022. ISSN 1385-8947. doi: https://doi.org/10.1016/j.cej.2021.132442.

[103] J. Palma, F. Castro, L. Ribeiro, A. Rodrigues, and A. Pinto. Linear and nonlinear models in wind resource assessment and wind turbine micro-siting in complex terrain. *Journal of Wind Engineering and Industrial Aerodynamics*, 96(12):2308 – 2326, 2008. ISSN 0167-6105. doi: https://doi.org/10.1016/j.jweia.2008.03.012.

[104] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc., 2019.

[105] K. Perlin. An image synthesizer. *ACM Siggraph Computer Graphics*, 19(3): 287–296, 1985.

[106] T. Phillips, M. Stölzle, E. Turricelli, F. Achermann, N. Lawrance, R. Siegwart, and J. J. Chung. Learn to path: Using neural networks to predict dubins path characteristics for aerial vehicles in wind. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1073–1079. IEEE, 2021.

[107] Plehiers, Pieter and Vandewalle, Laurien and Marin, Guy and Stevens, Christian and Van Geem, Kevin. Accelerating reactive CFD simulations with detailed pyrolysis chemistry using artificial neural networks. In *2019 AIChE annual meeting proceedings*, page 3. American Institute of Chemical Engineers (AIChE), 2019. ISBN 9780816911127.

[108] J. Qiu, Z. Cui, Y. Zhang, X. Zhang, S. Liu, B. Zeng, and M. Pollefeys. Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[109] M. Raffel. Background-oriented schlieren (bos) techniques. *Experiments in Fluids*, 56(3):1–17, 2015.

[110] M. Raffel, J. T. Heineck, E. Schairer, F. Leopold, and K. Kindler. Background-oriented schlieren imaging for full-scale and in-flight testing. *Journal of the American Helicopter Society*, 59(1):1–9, 2014.

[111] P. Ramesh and J. M. L. Jeyan. Comparative analysis of fixed-wing, rotary-wing and hybrid mini unmanned aircraft systems (uas) from the applications perspective. *INCAS Bulletin*, 14(1):137–151, 2022.

[112] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[113] A. Rautenberg, M. Graf, N. Wildmann, A. Platis, and J. Bange. Reviewing wind measurement approaches for fixed-wing unmanned aircraft. *Atmosphere*, 9, 10 2018.

[114] G. Reddy, J. Wong-Ng, A. Celani, T. J. Sejnowski, and M. Vergassola. Glider soaring via reinforcement learning in the field. *Nature Letters*, 562:236–239, 2018.

[115] A. Renzaglia, C. Reymann, and S. Lacroix. Monitoring the evolution of clouds with UAVs. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 278–283. IEEE, 2016.

[116] C. Reymann, A. Renzaglia, F. Lamraoui, M. Bronz, and S. Lacroix. Adaptive sampling of cumulus clouds with UAVs. *Autonomous robots*, 42(2):491–512, 2018.

[117] M. D. Ribeiro, A. Rehman, S. Ahmed, and A. Dengel. Deepcfd: Efficient steady-state laminar flow approximation with deep convolutional neural networks. *arXiv preprint arXiv:2004.08826*, 2020.

[118] J. Roadman, J. Elston, B. Argrow, and E. Frew. Mission performance of the tempest unmanned aircraft system in supercell storms. *Journal of Aircraft*, 49(6):1821–1830, 2012.

[119] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[120] A. Ryan and J. Hedrick. A mode-switching path planner for uav-assisted search and rescue. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 1471–1476, 2005.

[121] I. Sa, Z. Chen, M. Popović, R. Khanna, F. Liebisch, J. Nieto, and R. Siegwart. weednet: Dense semantic weed classification using multispectral images and MAV for smart farming. *IEEE Robotics and Automation Letters*, 3(1):588–595, 2017.

[122] L. Sankaralingam and C. Ramprasadh. Angle of attack measurement using low-cost 3d printed five hole probe for uav applications. *Measurement*, 168: 108379, 2021. ISSN 0263-2241. doi: https://doi.org/10.1016/j.measurement.2020.108379.

[123] M. Scacco, A. Flack, O. Duriez, M. Wikelski, and K. Safi. Static landscape features predict uplift locations for soaring birds across europe. *Royal Society open science*, 6(1):181440, 2019.

[124] M. Schartel, R. Burr, W. Mayer, N. Docci, and C. Waldschmidt. UAV-based ground penetrating synthetic aperture radar. In *2018 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, pages 1–4. IEEE, 2018.

[125] T. Schlegel, M. Geissmann, M. Hertach, and D. Kröpfli. Windatlas Schweiz: Jahresmittel der modellierten windgeschwindigkeit und windrichtung. Technical Report COO.2207.110.2.1073455, Federal Department of Environment, Transport, Energy and Communications (UVEK), 2016.

[126] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *International Journal of computer vision*, 37(2):151–172, 2000.

[127] T. Schneider, M. T. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart. maplab: An open framework for research in visual-inertial mapping and localization. *IEEE Robotics and Automation Letters*, 2018. doi: 10.1109/LRA.2018.2800113.

[128] M. L. Seltzer and J. Droppo. Multi-task learning in deep neural networks for improved phoneme recognition. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6965–6969, 2013.

[129] G. Settles. *Toepler's Schlieren Technique*, pages 39–75. Springer Berlin Heidelberg, Berlin, Heidelberg, 01 2001. ISBN 978-3-642-63034-7. doi: 10.1007/978-3-642-56640-0_3.

[130] G. S. Settles. *Schlieren and shadowgraph techniques: visualizing phenomena in transparent media*. Springer Science & Business Media, 2001.

[131] X. Shen, L. Xu, Q. Zhang, and J. Jia. Multi-modal and multi-spectral registration for natural images. In *European Conference on Computer Vision*, pages 309–324. Springer, 2014.

[132] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6, 2019.

[133] V. Sitzmann, J. N. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020.

[134] N. T. Smith, J. T. Heineck, and E. T. Schairer. Optical flow for flight and wind tunnel background oriented schlieren imaging. In *55th AIAA aerospace sciences meeting*, page 0472, 2017.

[135] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021.

[136] T. Stastny and R. Siegwart. Nonlinear model predictive guidance for fixed-wing uavs using identified control augmented dynamics. In *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 432–442, 2018. doi: 10.1109/ICUAS.2018.8453377.

[137] T. Stastny and R. Siegwart. On flying backwards: Preventing run-away of small, low-speed, fixed-wing uavs in strong winds. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5198–5205, 2019.

[138] T. J. Stastny, A. Dash, and R. Siegwart. Nonlinear MPC for fixed-wing UAV trajectory tracking: Implementation and flight experiments. In *AIAA Guidance, Navigation, and Control Conference*, page 1512, 2017.

[139] M. Stolle. *Towards Vision-Based Autonomous Cross-Country Soaring for UAVs*. PhD thesis, UNIVERSITE DE TOULOUSE, 2017.

[140] SwissTopo. 3d elevation program. https://www.geo.admin.ch/en/geo-information-switzerland/geodata-index-inspire/surface-representation/elevation.html, Dec 2021.

[141] R. Szeliski. Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2(1):1–104, 2007.

[142] P. Taylor and H. Teunissen. *Askervein'82: Report on the September/October 1982 Experiment to Study Boundary Layer Flow over Askervein, South Uist.* Meteorological Services Research Branch, Atmospheric Environment Service, 1983.

[143] P. A. Taylor and H. W. Teunissen. The askervein hill project: Overview and background data. *Boundary-Layer Meteorology*, 39(15), 1987.

[144] Y. Tominaga, A. Mochida, R. Yoshie, H. Kataoka, T. Nozu, M. Yoshikawa, and T. Shirasawa. AIJ guidelines for practical applications of CFD to pedestrian wind environment around buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, 96(10):1749 – 1761, 2008. ISSN 0167-6105. 4th International Symposium on Computational Wind Engineering (CWE2006).

[145] J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin. Accelerating Eulerian fluid simulation with convolutional networks. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3424–3433. PMLR, 06–11 Aug 2017.

[146] T. Uchida and Y. Ohya. Micro-siting technique for wind turbine generators by using large-eddy simulation. *Journal of Wind Engineering and Industrial Aerodynamics*, 96(10):2121 – 2138, 2008. ISSN 0167-6105. doi: https://doi.org/10.1016/j.jweia.2008.02.047. 4th International Symposium on Computational Wind Engineering (CWE2006).

[147] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger. Sparsity invariant cnns. In *2017 International Conference on 3D Vision (3DV)*, pages 11–20, 2017.

[148] N. Umetani and B. Bickel. Learning three-dimensional flow for interactive aerodynamic design. *ACM Trans. Graph.*, 37(4):89:1–89:10, July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201325.

[149] USGS. 3d elevation program. https://www.usgs.gov/3d-elevation-program, Dec 2021.

[150] N. Vasiljević, G. Lea, M. Courtney, J.-P. Cariou, J. Mann, and T. Mikkelsen. Long-range windscanner system. *Remote Sensing*, 8(11), 2016. ISSN 2072-4292.

[151] S. Verling, T. Stastny, G. Bättig, K. Alexis, and R. Siegwart. Model-based transition optimization for a vtol tailsitter. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3939–3944. IEEE, 2017.

[152] P. Viola and W. M. Wells III. Alignment by maximization of mutual information. *International journal of computer vision*, 24(2):137–154, 1997.

[153] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

[154] A. Voudouri, P. Khain, I. Carmona, E. Avgoustoglou, P. Kaufmann, F. Grazzini, and J. Bettems. Optimization of high resolution cosmo model performance over switzerland and northern italy. *Atmospheric Research*, 213: 70–85, 2018. ISSN 0169-8095. doi: https://doi.org/10.1016/j.atmosres.2018.05.026.

[155] N. Wandel, M. Weinmann, and R. Klein. Unsupervised deep learning of incompressible fluid dynamics. *CoRR*, abs/2006.08762, 2020.

[156] T. Whalen, E. Simiu, G. Harris, J. Lin, and D. Surry. The use of aerodynamic databases for the effective estimation of wind effects in main wind-force resisting systems:: application to low buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, 77-78:685 – 693, 1998. ISSN 0167-6105. doi: https://doi.org/10.1016/S0167-6105(98)00183-4.

[157] S. Wiewel, M. Becher, and N. Thuerey. Latent-space physics: Towards learning the temporal evolution of fluid flow. *arXiv preprint arXiv:1802.10123*, 2018.

[158] H. J. Williams, E. L. C. Shepard, M. D. Holton, P. A. E. Alarcón, R. P. Wilson, and S. A. Lambertucci. Physical limits of flight performance in the heaviest soaring bird. *Proceedings of the National Academy of Sciences*, 117 (30):17884–17890, 2020. doi: 10.1073/pnas.1907360117.

[159] B. Wilson, S. Wakes, and M. Mayo. Surrogate modeling a computational fluid dynamics-based wind turbine wake simulation using machine learning. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8, 2017. doi: 10.1109/SSCI.2017.8280844.

[160] L. Wirth, P. Oettershagen, J. Ambühl, and R. Siegwart. Meteorological path planning using dynamic programming for a solar-powered UAV. In *2015 IEEE Aerospace Conference*, pages 1–11, March 2015. doi: 10.1109/AERO.2015.7119284.

[161] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics.

[162] J. Wulff and M. J. Black. Efficient sparse-to-dense optical flow estimation using a learned basis and layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[163] X. Xiang, R. Abdein, and N. Lv. Unsupervised optical flow estimation method based on transformer and occlusion compensation. *Neural Computing and Applications*, pages 1–13, 2022.

[164] Y. Xie, E. Franz, M. Chu, and N. Thuerey. tempoGAN: A temporally coherent, volumetric GAN for super-resolution fluid flow. *ACM Transactions on Graphics (TOG)*, 37(4):95, 2018.

[165] T. Xue, M. Rubinstein, N. Wadhwa, A. Levin, F. Durand, and W. T. Freeman. Refraction wiggles for measuring fluid depth and velocity from video. In *European Conference on Computer Vision*, pages 767–782. Springer, 2014.

[166] W. Yi, C. Liming, K. Lingyu, Z. Jie, and W. Miao. Research on application mode of large fixed-wing uav system on overhead transmission line. In *2017 IEEE International Conference on Unmanned Systems (ICUS)*, pages 88–91, 2017.

[167] S. Yu, J. Heo, S. Jeong, and Y. Kwon. Technical analysis of vtol uav. *Journal of Computer and Communications*, 4(15):92–97, 2016.

[168] N. Zehtabiyan-Rezaie, A. Iosifidis, and M. Abkar. Data-driven fluid mechanics of wind farms: A review. *Journal of Renewable and Sustainable Energy*, 14(3):032703, 2022. doi: 10.1063/5.0091980.

[169] J. Zhang and X. Zhao. Machine-learning-based surrogate modeling of aerodynamic flow around distributed structures. *AIAA Journal*, 59(3):868–879, 2021.

[170] Z. Zhang, C. Santoni, T. Herges, F. Sotiropoulos, and A. Khosronejad. Time-averaged wind turbine wake flow field prediction using autoencoder convolutional neural networks. *Energies*, 15(1), 2022. ISSN 1996-1073.

[171] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

Curriculum Vitae

Florian Achermann
born August 02, 1992
citizen of Sisikon UR, Switzerland

2018–2022	*ETH Zurich, Switzerland* Doctoral studies at the Autonomous Systems Lab; Supervised by Prof. Roland Siegwart
2017–2018	*ETH Zurich, Switzerland* Research Assistant at the Autonomous Systems Lab
2014–2017	*ETH Zurich, Switzerland* Master of Science in Robotics, Systems, and Control (With Honors)
2011–2014	*ETH Zurich, Switzerland* Bachelor of Science in Mechanical Engineering
2005–2011	*Gymnasium, Altdorf, Switzerland*