Delft University of Technology

Autonomous landing of Micro Air Vehicles through bio-inspired monocular vision

Ho, Hann Woei

DOI
10.4233/uuid:7efd562f-82fa-468f-b074-bfa7d640a9ee

Publication date
2017

Document Version
Final published version

Citation (APA)
Ho, H. W. (2017). *Autonomous landing of Micro Air Vehicles through bio-inspired monocular vision*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:7efd562f-82fa-468f-b074-bfa7d640a9ee

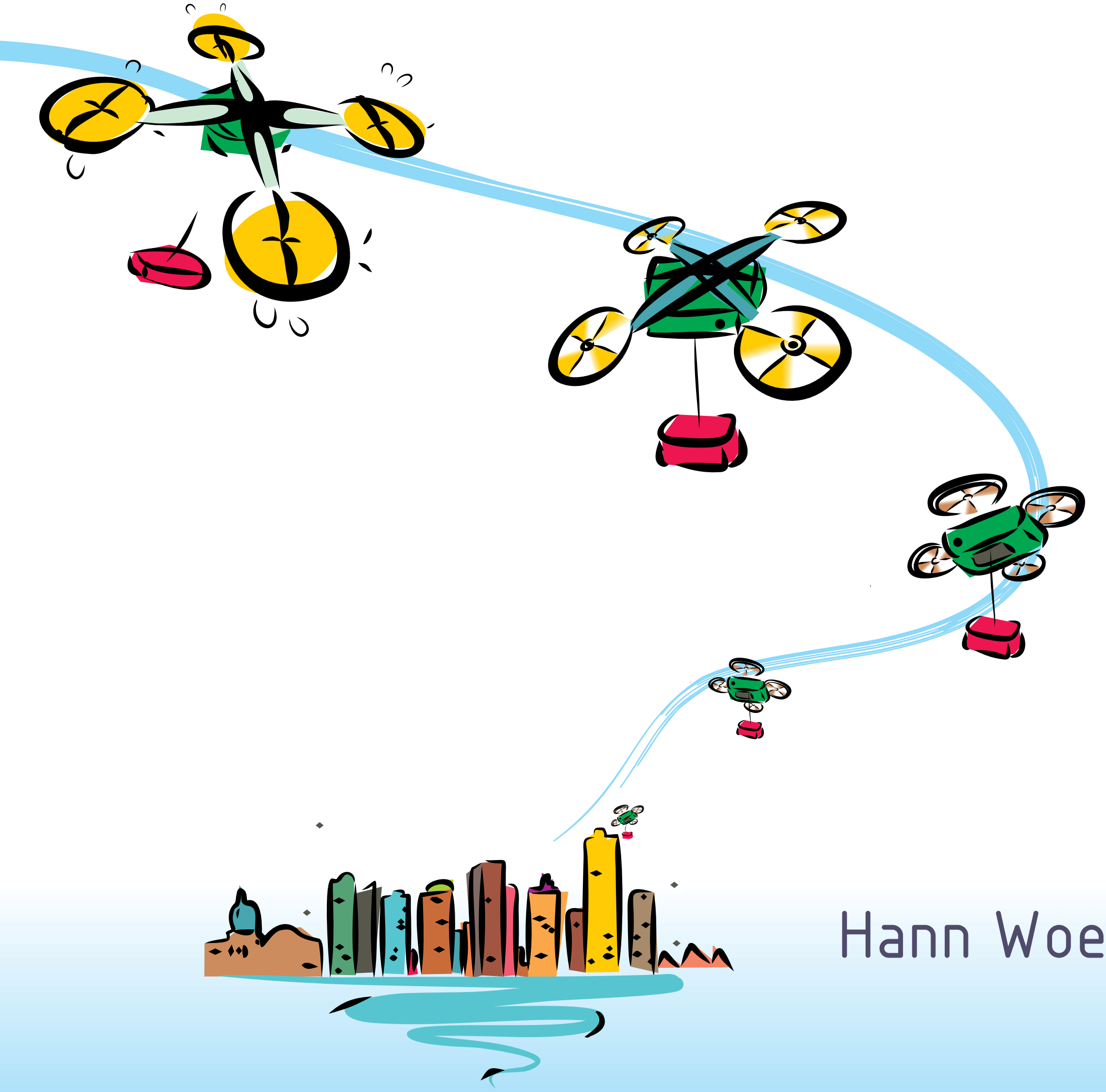
Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Autonomous Landing of Micro Air Vehicles through Bio-inspired Monocular Vision



Hann Woei Ho

AUTONOMOUS LANDING OF MICRO AIR VEHICLES THROUGH BIO-INSPIRED MONOCULAR VISION

AUTONOMOUS LANDING OF MICRO AIR VEHICLES THROUGH BIO-INSPIRED MONOCULAR VISION

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op woensdag 21 juni 2017 om 15:00 uur

door

Hann Woei HO

ingenieur luchtvaart en ruimtevaart
geboren te Batu Pahat, Johor, Malaysia

Dit proefschrift is goedgekeurd door de

promotor: prof. dr. ir. M. Mulder
copromotor: dr. G.C.H.E. de Croon

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof. dr. ir. M. Mulder,	Technische Universiteit Delft, promotor
Dr. G.C.H.E. de Croon,	Technische Universiteit Delft, copromotor
Onafhankelijke leden:	
Prof. dr. ir. J. A. Mulder,	Technische Universiteit Delft
Prof. dr. ir. M. J. T. Reinders,	Technische Universiteit Delft
Prof. dr. E. O. Postma,	Universiteit Tilburg
Dr. T. Seidl,	Westfälische Hochschule
Dr. F. Ruffier,	Aix-Marseille Université

Keywords: Monocular Vision; Bioinspiration; Micro Air Vehicle; Autonomous Landing; Optical Flow; Efference Copy; Appearance Cues; Distance Estimation; Adaptive Control; Visual Servoing; Obstacle Detection; Self-supervised Learning.

Printed by: Ipskamp Drukkers, Enschede, The Netherlands.

Front & Back: Designed by Yazdi Ibrahim Jenie.

ISBN 978-94-6186-818-3

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

Copyright © 2017 by Hann Woei Ho. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior permission in writing from the proprietor.

To my beloved family and friends . . .

SUMMARY

AUTONOMOUS LANDING OF MICRO AIR VEHICLES THROUGH BIO-INSPIRED MONOCULAR VISION

Hann Woei HO

Autonomous flying vehicles – especially the small and light-weight Micro Air Vehicles (MAVs) – are currently not capable of finding a safe landing site by themselves and performing smooth landings. The main reason is that most of the state-of-the-art solutions are computationally expensive and thus restrict their application on MAVs which have severely limited processing power and sensors available. Hence, this kind of vehicles requires an energy-efficient landing solution using a light-weight sensor.

A promising option for MAV landing is to draw inspiration from tiny flying insects. The motivation comes from the fact that these insects have very limited neural and sensory resources, yet they can perform complex tasks, such as navigating in a dense, obstacle-rich environment and carrying out smooth landings. This means that they possess efficient and robust approaches to perception and control problems, and these approaches could be ideal for designing the control strategies for MAV landing. Research shows that, to realize many of these complex tasks, flying insects use control strategies which heavily rely on optical flow.

However, there are many challenges when using solely optical flow for MAV landing. Optical flow is ‘scaleless’ and does not provide the distance to an object or the observer’s velocity, but only the ratio of them. The scaleless property of optical flow can lead to various stability problems of MAV landings. For instance, for landing on a flat surface, optical flow control causes instabilities at some points in the phase of the landing when the control gains are not adapted to the height. Thus, height information is important in optical flow landings. Besides, for a fully autonomous MAV, being able to land on a flat surface is not sufficient. The MAV must also be able to find a suitable landing site by itself. In fact, optical flow information obtained from a single camera is sparse and needs to be further interpreted to understand what the flow pattern actually represents. Therefore, MAVs need efficient algorithms which can analyze the optical flow to search for a ground surface that is relatively flat and also – perhaps even more important – free of obstacles.

This thesis aims to overcome the optical flow related control problems in MAV landings, and has the following main problem statement:

How can Micro Air Vehicles utilize bio-inspired monocular vision to land fully autonomously?

The problem statement can be divided into a low-level and a high-level landing control part, both using bio-inspired monocular vision methods. The low-level landing control part ensures a smooth landing trajectory by solving the scaleless problems when using only optical flow for landings, whereas the high-level landing control part searches for a suitable landing site for the MAV using monocular vision. This leads to the following research questions:

1. How can MAVs cope with the scaleless property of optical flow from a single camera for optical flow landing?
2. How can MAVs autonomously find a suitable landing site with monocular vision?

To address the first research question, two approaches are proposed. The first approach directly tackles the problem by introducing an extended Kalman filter (EKF) algorithm that estimates the height and vertical velocity of an MAV during landings. The algorithm utilizes the knowledge of the control input, the so-called efference copy, and the observed flow divergence resulting from this input. For the prediction step in the EKF, the efference copy is used to predict the states, i.e., the height and vertical velocity, while for the correction step in the EKF the observed flow divergence is used to correct the predicted states. By doing that, the height and vertical velocity of an MAV can be estimated.

To check the observability of the system, a nonlinear observability analysis was carried out. The analysis shows that with the knowledge of the control input and flow divergence the system is observable. The algorithm was tested in computer simulation as well as in multiple flight tests in indoor and windy outdoor environments. Results show that the algorithm manages to estimate the height and velocity of the MAV during landing accurately, compared to the ground truth provided by a motion tracking system for the indoor flights or a sonar for the outdoor flights. After obtaining the height estimate, several flight tests also demonstrate that either the control gain can be adapted to the height during constant flow divergence landings, or a linear control with the height estimate feedback can be used to land the MAV smoothly.

The second approach solves the scaleless problem of optical flow landing by proposing a control strategy that automatically adjusts the control gain during landing. This strategy has two phases which intend to achieve a smooth landing trajectory using constant flow divergence. In Phase *I*, the MAV detects its height in hover using an oscillating movement and sets the gains accordingly. In Phase *II*, the constant flow divergence landing is activated, and the gains are reduced exponentially during descent and adjusted properly when the actual trajectory deviates too much from the expected constant flow divergence landing. In this strategy, Phase *I* ensures that proper initial gains are selected to maximize the control performance and thus avoid the MAV to descend too fast, while Phase *II* prevents self-induced oscillations when descending.

To realize Phase *I*, a novel way to detect self-induced oscillations in real-time was developed based on the observation of the flow divergence. This method detects oscillations experienced by the MAV by examining the covariance function of a windowed flow divergence and a time-shifted windowed flow divergence. This function is computationally efficient and can capture both the relative phase and the magnitude of deviations in the signal. Both simulation examples and flight tests show larger covariances of the signal at the instance when the oscillations occur. By detecting oscillations, the MAV can select proper initial gains. To examine the feasibility of Phase *II*, a stability analysis of the adaptive control strategy was performed. This analysis proves that the system is not subjected to self-induced oscillations when the adaptive controller is used during the constant flow divergence landing. In addition, multiple flight tests demonstrate smooth landings in both indoor and windy outdoor environments using the adaptive control strategy.

To answer the second research question, this thesis discusses an efficient algorithm that uses the optical flow information to search for safe landing sites. These are defined as relatively flat surfaces that do not have a too large inclination and, most importantly, are free of obstacles. To find a safe landing site, the proposed algorithm fits the optical flow field using RANdom SAmple Consensus (RANSAC). This algorithm assumes that the landing surface is planar. The results of the fitting provide two important estimates, i.e., the surface slope, which shows the inclination of the ground surface, and the surface roughness, which indicates the presence of obstacles. When the fit is bad, the surface roughness is high, and thus the plane is likely to include some obstacles. In contrast, when the fit is good, the surface roughness is low, and thus there are no obstacles of any size.

To first evaluate the accuracy of the slope estimate, artificial images, hand-held videos, and images from an MAV were tested. The results show that the algorithm is able to estimate the slope angles of a landing surface accurately when the camera is calibrated, and the MAV rotation is compensated. Additionally, a landing experiment was performed to show that the algorithm can distinguish between a stairway and a flat surface and land on a flat surface. To evaluate an obstacle detection method based on surface roughness, multiple on-board image sets taken in nine different indoor and outdoor scenes were tested. In fact, the success of finding a safe landing site depends on the obstacle detection performance with a reasonable false alarm. The obstacle detection performance is measured by the true positive rate, i.e., the percentage of obstacles which are correctly identified, while the false alarm is given by the false positive rate, i.e., the percentage of empty landing areas wrongly classified as containing an obstacle. The classification tests show that for the performance measure the true positive rates are higher than 90% for all scenes, while for the false alarm measure the false positive rates are less than 25% for all scenes. These results show high proportions of correct identification of obstacles and non-obstacles images. Most of the undetected cases occurred because only a small part of the obstacles appears at the edge of the images. This makes them difficult to be detected as the surface roughness measures the presence of obstacles in the global image. Multiple flight tests demonstrate the success of identifying the safe landing sites.

In addition, this research attempts to ‘go beyond’ optical flow approaches for the safe

landing site selection. A novel setup of self-supervised learning is introduced, in which optical flow serves as a ‘scaffold’ to learn the visual appearance of obstacles. This setup is to deal with the main problem of using monocular motion cues, i.e., their requirement of significant camera movement. By first using the surface roughness to detect obstacles, a function is learned that maps the appearance features represented by texton distributions to the surface roughness. This approach allows the MAV to select a suitable landing site while hovering. In addition, despite the fact that the surface roughness represents the global value for the presence of obstacles, this appearance learning allows for the pixel-wise segmentation of obstacles when analyzing local image patches.

To evaluate the classification performance of this approach, similar image sets as mentioned above were used. Results show that the true positive rates are slightly better than the classification using the surface roughness from optical flow. Similar undetected cases were found using this approach. However, the false positive rates are higher for some complex indoor scenes. This can also be seen from the larger normalized root mean square errors in more challenging indoor scenes where the appearance variation is larger. Based on the analysis, this approach is able to detect most of the obstacles and find the possible safe landing sites. Additionally, using this approach also resulted in successful landings in both indoor and outdoor experiments.

In conclusion, the bio-inspired monocular vision is a promising approach to achieve autonomous landings of an MAV. It is shown that the bio-inspired control strategies as practiced by flying insects, such as constant flow divergence landing, can only be successfully implemented on MAVs by solving some fundamental problems of optical flow. In addition, learning the obstacle appearance from optical flow can further improve the landing capabilities of MAVs. This thesis proposes several solutions which lead to novel insights that are not only important to a tiny drone but may also prove their value to biology.

Future work on adapting the bio-inspired approach to novel sensors that mimic insects’ eyes is recommended. These novel sensors are small and can directly measure optical flow. Hence, the sensing efficiency and the performance of the methodology can be further improved. For height estimation, it is recommended to augment the state vector in the extended Kalman filter with the wind disturbance as an additional state. Although the outdoor flight test showed that the wind factor might not be a problem, compensating for the wind disturbance might further improve the accuracy of the height estimate. Additionally, for the self-supervised learning of obstacle appearance, it is also interesting to try out advanced learning methods, such as deep learning. The proposed future work would be of great help in enhancing the flying capabilities of MAVs, and ideally moving towards the remarkable capabilities of flying insects.

CONTENTS

Summary **vii**

1 Introduction **1**
1.1 Autonomous flight of Micro Air Vehicles . . . 1
1.2 Landing problems of Micro Air Vehicles . . . 2
1.3 Visual landing solutions for Micro Air Vehicles . . . 3
1.4 Bio-inspired solutions for autonomous landings . . . 4
1.5 Problem statement and research questions . . . 7
1.6 Scope of the thesis . . . 8
1.7 Research approach . . . 9
1.8 Outline of the thesis . . . 11

I Low-level landing control **13**

2 Monocular Distance Estimation for MAV Landing **15**
2.1 Introduction . . . 16
2.2 Background . . . 17
2.2.1 Flow Divergence . . . 17
2.2.2 Constant Flow Divergence Guidance and Control . . . 17
2.3 EKF-based Height and Velocity Estimation Using Flow Divergence and Control Input . . . 18
2.3.1 Extended Kalman Filter . . . 19
2.3.2 Nonlinear Observability of the Estimation . . . 21
2.3.3 Features-based Flow Divergence Estimation . . . 21
2.4 Computer Simulation . . . 21
2.4.1 Landing Simulation with Simulated Control Inputs . . . 22
2.5 Experiments Results and Discussion . . . 23
2.5.1 Landing Using Constant Flow Divergence with Fixed-gain Control . . . 23
2.5.2 Landing Using Constant Flow Divergence with Height-adapted Control . . . 25
2.5.3 Landing Using Height and Velocity from EKF . . . 26
2.5.4 Discussion on Wind Effect and Rejection . . . 28
2.6 Conclusion . . . 30

3 Adaptive Control Strategy for Constant Flow Divergence Landing **31**
3.1 Introduction . . . 32
3.2 Background . . . 33
3.2.1 Flow Divergence . . . 33
3.2.2 Constant Flow Divergence Guidance Strategy . . . 34
3.2.3 Conventional Control Scheme . . . 35

3.3 Characterization of Flow Divergence 35
3.3.1 Flow Field Divergence Estimator 36
3.3.2 Size Divergence Estimator 37
3.3.3 Testing Platform and Data Logging. 38
3.3.4 Delay Estimation and Noise Model. 39
3.4 Influence of Delay and Noise on Fixed-Gain Constant Flow Divergence Landing 43
3.4.1 Simulation Results 43
3.4.2 Flight Test Results 45
3.5 Adaptive Gain Strategy for Constant Divergence Landing 46
3.5.1 Adaptive Control Strategy 46
3.5.2 Stability Analysis of the Adaptive Controller 47
3.5.3 Real-Time Oscillations Detection 50
3.6 Flight Tests 51
3.6.1 Indoor Flight Tests 53
3.6.2 Outdoor Flight Tests 55
3.7 Conclusions. 55

II High-level landing control **57**

4 Surface Slope Estimation Using Optical Flow **59**
4.1 Introduction 60
4.2 Vision Algorithm for Slope Estimation 61
4.2.1 Optical Flow Equations for a Sloped Planar Surface 61
4.2.2 Optical Flow Field Parameter Estimation. 62
4.2.3 Extraction of Variables Relevant for Landing 62
4.2.4 Implementation 64
4.3 Experiments on Image Sequences. 65
4.4 Landing Experiment 69
4.5 Conclusion 70

5 Self-supervised Learning of Obstacle Appearance **73**
5.1 Introduction 74
5.2 Related Work 75
5.3 Method 76
5.3.1 Surface Roughness from Optical Flow 76
5.3.2 Appearance from Texton Distribution 78
5.3.3 Regression Learning of Surface Roughness. 78
5.4 Obstacle Detection 79
5.4.1 Obstacle Detection with Surface Roughness 80
5.4.2 Obstacle Detection with Visual Appearance 81
5.5 Generalization 86
5.5.1 Generalization of SSL Method to Different Environments 86

5.6 Flight Tests 92
  5.6.1 Experiment Platform 92
  5.6.2 Processing Time of Computer Vision Algorithms 93
  5.6.3 Autonomous Landing Strategy Using ϵ^* from Optical Flow 94
  5.6.4 Autonomous Landing Strategy Using $\hat{\epsilon}$ from Appearance 95
5.7 Discussion 98
  5.7.1 Influence of Height on Roughness ϵ^* and $\hat{\epsilon}$ 98
  5.7.2 Applications 99
5.8 Conclusion 99

6 Conclusion and Recommendations **101**
6.1 Discussion 101
  6.1.1 Low level landing control 101
  6.1.2 High level landing control 103
6.2 Final conclusion 105
6.3 Future work 106

A Discretization of linear state space models **109**

References **111**

Nomenclature **120**

Samenvatting **127**

Acknowledgements **133**

Curriculum Vitæ **135**

List of Publications **137**

1

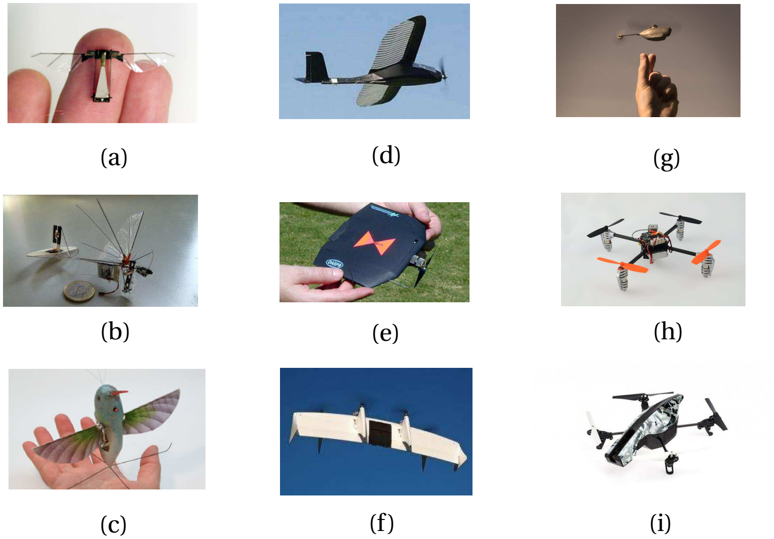
INTRODUCTION

1.1. AUTONOMOUS FLIGHT OF MICRO AIR VEHICLES

UNMANNED Aerial Vehicles (UAVs) have become very popular since many years ago. At that time no one could imagine that one day these vehicles would be operated so close to our daily life. Recently, the use of UAVs in the civilian field has rapidly grown as they provide inexpensive and more effective solutions to our day-to-day problems compared to manned aircraft. These vehicles, without a pilot on-board, can be controlled remotely by ground crews or fly autonomously.

Remotely-controlled civilian UAVs are mostly used by hobbyists or aerial photographers. Autonomous UAVs have wider applications, ranging from cost-effective mail and package delivery [1, 2] to life-saving ambulance UAVs [3]. Autonomous UAVs do not only significantly reduce the workload of the ground crews, but also save time, money, and effort to recruit and train human pilots. Most importantly, autonomous UAVs have tremendous potential capabilities to perform tasks in dangerous areas where the risk to humans would be unacceptable, such as wildlife monitoring, civil security, fire scene inspections, and search and rescue missions after a nuclear plant disaster.

In many of these applications, the UAVs will have to fly in narrow environments and even close to people. This has led to a huge interest in smaller UAVs, the so-called Micro Air Vehicles (MAVs). Due to their small size and light weight, they are often safer and more practical to be operated in urban and indoor environments. The US Defense Advanced Research Projects Agency (DARPA) defined the physical size of an MAV to be smaller than 15 centimeters, and maximum take-off weight to be less than 100 grams in an MAV program in 1997 [4]. Recently, due to the exponential growth in drone industry, this term has generally referred to smaller and man-portable UAVs. Some examples of MAVs are presented in Fig. 1.1. They range from a 700 g commercially available, tail-sitter quadrotor [5] to a 60 mg research-based, insect-like MAV [6]. Since they are so small, these vehicles are barely visible in the sky and thus hard to be operated by human pilots in a line-of-sight operation. Furthermore, for some missions, more than one MAV is often required to operate together, i.e., in a swarm of MAVs. Here, it is absolutely necessary to enable them to fly completely by themselves.



(a) (d) (g)

(b) (e) (h)

(c) (f) (i)

Figure 1.1: Examples of different types of Micro Air Vehicles (MAVs). Flapping wing: (a) RoboBee [6] , (b) Delfly Micro [7], (c) Nano Hummingbird [8]; Fixed wing: (d) BATCAM [9] (e) Black Widow [10]; Hybrid: (f) Quadshot [5]; Rotary wing: (g) Black Hornet PD-100 [11], (h) Ladybird [12], (i) Parrot AR.Drone 2.0 [13]

1.2. LANDING PROBLEMS OF MICRO AIR VEHICLES

Any flying vehicle will need to land eventually, so will the MAVs. Since these vehicles need to fly autonomously, the autonomous landing capability is indispensable. For example, in emergency situations where the vehicle detects a serious problem, the vehicle could be forced to land by itself as quickly as possible by overriding all other commands. Moreover, when these vehicles operate outside the line-of-sight, they need to have the autonomous landing capability, without the help of ground pilots or other external systems.

A major concern of the current landing systems is that they do not have reliable methods that would allow them to decide *by themselves* which areas are safe to land and how to approach the desired landing target. For instance, the recent demonstration of package delivery using an MAV by Amazon shows the feasibility of an MAV sending a package to its consumer [14]. A marker which is placed by the consumer on the ground is used to allow the MAV to land safely. In some circumstances in which the marker is unavailable, these vehicles need to be able to sense and avoid the surrounding objects in the environment and perform a smooth descent, all by themselves. Without this capability, the safety of the surrounding animals, humans, and property cannot be ensured. As a result, the use of these vehicles will definitely be prohibited by the regulations.

Compared to larger UAVs, MAVs face greater challenges in performing autonomous landing. The main problem for MAVs lies in the restriction of their size and weight. In other words, the sensors they can carry on-board and the processing power available are severely limited. Unlike larger UAVs, many intelligent algorithms which require large sensor(s) and heavy processing are not available to MAVs to perform autonomous landing. Hence, they need a computationally much more efficient landing solution, using much smaller and lighter sensors which are also much less accurate than their larger counterparts.

1.3. VISUAL LANDING SOLUTIONS FOR MICRO AIR VEHICLES

MAVs are typically equipped with some conventional, yet miniaturized, sensors on-board, such as Inertial Measurement Unit (IMU), barometer, magnetometer, and Global Positioning System (GPS). These sensors are mainly used in basic functionalities, such as self-stabilization, waypoint guidance and navigation. For navigation, GPS is the most commonly used sensor in MAVs. Recent developments of GPS navigation techniques (e.g., real-time kinematic, RTK, and Differential GPS, DGPS) have improved GPS accuracy and allow for more precise positioning of MAVs. However, GPS is only reliable for high altitude navigation. When flying at low altitudes, it can only be used when a very detailed map of the ground, where the locations of all obstacles are available, is known in advance. In addition, due to its weak signal, GPS is not applicable to indoor environments and also urban areas with surrounding tall buildings.

Modern auxiliary sensors play a vital role to provide additional information to MAVs. Active sensors, such as laser rangefinders, measure the distance to a target using a method called Light Detection and Ranging (Lidar) which emits a laser light and measures its deflection time. This method is often used to build a 3D map [15, 16]. Although this sensor can provide accurate measurements, it is not suitable for MAVs due to the size and weight which is typically more than 100 g. In contrast, visual sensors are passive sensors which can provide rich information about the surroundings and the MAV's self-motion. Multiple cameras used in stereo vision can also measure the distances to many points in an image [17–19]. However, stereo vision is limited in range. Furthermore, having two cameras is less energy or weight efficient than a single camera. Thus, for MAVs, single cameras are preferable to be installed and used on-board.

Intelligent algorithms which utilize this visual sensory information from a single camera are needed to perform the autonomous landing task. If a visual landing target is known, object detection and tracking is often used with a downward-looking camera for locating a static or dynamic landing platform over time [20, 21]. The image position of an MAV with respect to the target can be determined and used to control the MAV. Moreover, with the known target size, the distance to the target and its velocity can be estimated. Some common designated markers used as landing targets are shown in Fig. 1.2. Although one of the real world applications [14] mentioned in the previous section relies on a designated marker for landing, using this approach restricts the autonomy of MAVs. For example, when the marker is wrongly placed or unavailable, the MAV might be misled, or it cannot decide where to land. Additionally, in some emergency situations, the MAV needs to land as quickly as possible. Therefore, it is essential that these vehicles know how to select a safe landing spot all by themselves, without depending on external systems.

In an unknown environment, where no known landing target is present, a standard approach in robotics is visual Simultaneous Localization and Mapping (SLAM). It locates and tracks all features in the field of view of the camera and constructs a 3D map of the surrounding environment. Visual SLAM is currently the most popular method

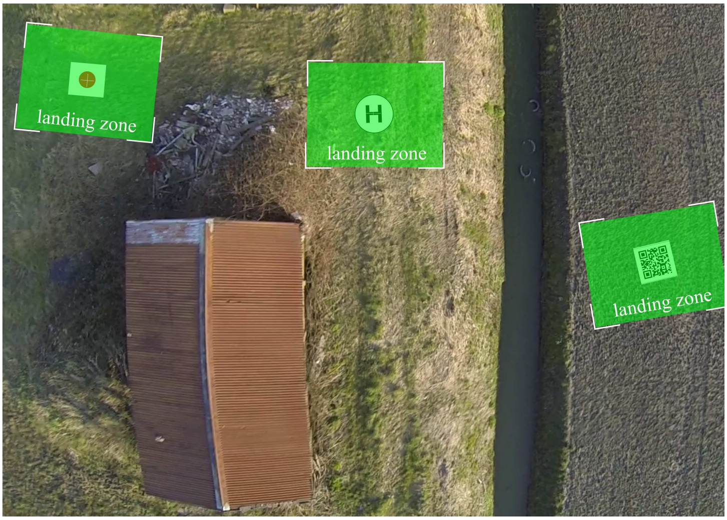


Figure 1.2: Examples of designated markers used as landing targets. In the current applications, Amazon Prime Air [14], for example, needs one of these specific markers to know where to land. It would be more practical if the autonomous UAVs could find a suitable landing spot all by themselves, without any markers.

used in robotics and can be categorized into feature-based and direct methods. In the feature-based method, the SLAM problem consists of features extraction, and computation of camera pose and scene geometry using these features [22–24]. The reliability of this method depends on the detected features and it is insufficiently robust to some scenes in which the features are hardly visible. The direct method solves this limitation by directly using image intensities to generate (semi-) dense maps [25–27]. Although the computational efficiency and accuracy of this method have been improved over the years [28–32], it still uses much more computational resources than strictly required and is not available on smaller platforms.

1.4. BIO-INSPIRED SOLUTIONS FOR AUTONOMOUS LANDINGS

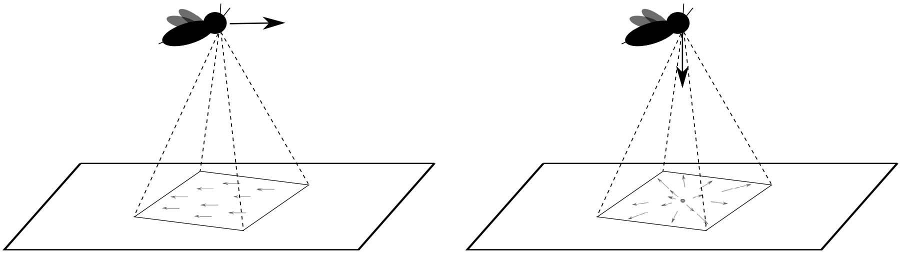
MAVs are limited in their size and weight. These constraints make MAVs challenging for achieving fully autonomous landing. For instance, they are restricted in the amount of payload and the processing capability. MAVs need very efficient algorithms that use a small and light-weight sensor, such as a single camera.

A promising option for MAV landing is to draw inspiration from biological systems. Flying insects can perform complex tasks using very limited neural and sensory resources [33]. Honeybees, for example, weigh only about one tenth of a gram on average and have a tiny brain of 1 mm^3, which contains only around 960,000 neurons [34] (for comparison: the human brain has about 100 billion neurons [35]). They primarily rely on their eyes to perform safe and smooth landings [36, 37]. Research shows that these insects have incredibly efficient and robust solutions to undertake these perception and control problems. These solutions can serve as the bio-inspired design principles for MAV control strategies [38, 39].

For knowing their movements and sensing objects in the surroundings while flying, the main visual cue used by flying insects is optical flow. Optical flow refers to the apparent motion of objects in a scene perceived by an observer during his/her movement through the world [40, 41]. Flying insects perform amazing maneuvers by heavily relying on optical flow. For instance, honeybees can use this information to center their positions when they are flying through a narrow space and to avoid obstacles [42].

Flying insects also perform approach and landing using optical flow. For example, honeybees keep the ventral flow constant to carry out a grazing landing, in which the direction of motion is nearly parallel to the landing surface [36, 37, 43]. Ventral flow as shown in Fig. 1.3a can be defined as a measure of the lateral velocity divided the height. Many studies on autonomous landings of MAVs and spacecraft have focused on using this strategy [33, 38, 44–46]. In fact, the success of these landings is governed by a pitching law profile [45, 46]. This profile is designed in such a way that the vehicles can reduce their lateral velocity while maintaining the ventral flow at a desired set point and thus automatically decrease the height. However, to achieving a certain descent profile, for instance, an exponential decay of the height, the design of the pitching profile requires the estimates of the height and vertical velocity of these vehicles. Additionally, for vertical landings, ventral flow is not dominant, and its value is (close to) zero.

On the other hand, flow divergence as shown in Fig. 1.3b or time-to-contact (i.e., the reciprocal of flow divergence) can provide the vertical motion of an observer relative to the ground [47–49]. A series of experiments shows evidence that bees land with constant flow divergence [49]. By keeping flow divergence constant, honeybees reduce their vertical velocities to almost zero at touchdown. Since flow divergence is a measure of the vertical velocity divided by the height, it is straightforward to show that when the flow divergence is held constant, the vertical velocity will decay exponentially to zero and so does the height.



(a) Ventral flow.

(b) Flow divergence.

Figure 1.3: Different patterns of optical flow resulted from different movements.

Optical flow strategies followed by honeybees are typically praised, as they seem to guarantee a smooth landing by only using optical flow. This is remarkable because the optical flow is 'scaleless' and provides the ratio of velocity to distance but not velocity and distance separately. This means that honeybees land without any knowledge of velocity and distance but some observables of their ratio. The scaleless property of optical flow can be seen from Fig. 1.4. In this figure, an MAV which hovers at a fixed position uses a downward-looking camera to measure the optical flow from a moving object on the

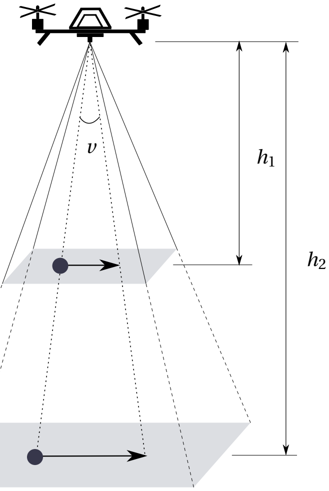


Figure 1.4: ‘Scale-less’ property of optical flow. This property can be clearly seen from the case that the same amount of optical flow v can be perceived by a stationary observer from a near, slow-moving object and a distant, fast-moving object. For instance, if $v = 1s^{-1}$, it could be perceived from an object that moves at $1ms^{-1}$ at $h_1 = 1m$ or at $10ms^{-1}$ at $h_2 = 10m$.

ground at different heights. The same amount of the optical flow can be observed from a closer object which moves slower and a distant object which moves faster. Using the optical flow, i.e., the ratio of velocity to distance, would be contrary to typical engineering solutions that opt to separate these physical quantities. For instance, using additional information from other sensors, such as accelerometers and sonar, can help to dissect the optical flow and estimate its distance and velocity components. However, minimal sensor solutions, e.g., using a single camera, are preferable for MAVs which have limited weight and size.

Using only a single camera (the optical flow) for landing on a flat surface is still a huge challenge for MAVs. Optical flow control causes instabilities at some points in the phase of the landing when the MAV control gain is not adapted to the height [50]. Several studies, therefore, use a gain-scheduling technique based on time-to-contact or a hybrid strategy which switches between the control of time-to-contact and flow divergence to achieve successful landings [51, 52]. In fact, these solutions require the knowledge of the height. For instance, those control gains are scheduled to a range of certain heights and velocities (or time-to-contact) for a landing profile which has a specific initial height and velocity. If the initial height and velocity deviate too much from the designed values, a smooth landing cannot be achieved. Therefore, the height information plays a major role in optical flow landings.

Using bio-inspired vision solutions in landing an MAV involves the feedback control of often very noisy visual cues (i.e., optical flow estimates). Some studies have shown the feasibility of using optical flow for autonomous landing of MAVs [52–54]. However, the landings were either performed at a relatively slow pace, or only a few successful landings using visual feedback were presented [53, 54]. The only study which performed many flight tests did not use any visual sensor but used other sensors for control [52].

Solving the problem of landing on a flat surface is not sufficient for a fully autonomous MAV. Namely, the MAV also has to select a suitable, flat landing surface by itself. From the literature, many studies use optical flow observed from a forward-looking camera for avoiding obstacles while navigating in an indoor or urban environment [55, 56], but it seems none of them has a solution which uses optical flow from a downward-looking camera to select landing sites. To utilize optical flow, MAVs need an efficient algorithm which can interpret optical flow to search for a relatively flat ground surface that is also, and most importantly, free of obstacles.

1.5. PROBLEM STATEMENT AND RESEARCH QUESTIONS

Based on the challenges mentioned in the previous section, the main problem statement of this thesis can be formulated as follows:

Problem statement

How can Micro Air Vehicles utilize bio-inspired monocular vision to land fully autonomously?

The problem statement can be divided into two parts: (I) low-level landing control, and (II) high-level landing control, both using bio-inspired monocular vision methods.

Low-level landing control is responsible for ensuring a smooth landing trajectory using for instance the constant flow divergence strategy. By controlling the upward thrust of the vehicle to keep the observed flow divergence constant, both height and velocity of an MAV decay exponentially during landing. However, the scaleless property of optical flow leads to a problem of optical flow control in MAV landings. For instance, a fixed-gain control cannot guarantee a desired landing trajectory [50]. In fact, the height information plays an important role in optical flow control. For instance, the height dictates how strongly the MAV has to react for maintaining a stable constant flow divergence landing [50]. This leads to the first research question (RQ):

RQ1: How can MAVs cope with the scaleless property of optical flow from a single camera for optical flow landing?

High-level landing control relates to searching for a suitable landing site for the MAV. A desired landing site is free of obstacles and should not have a too large inclination. Flying insects are good at avoiding obstacles using optical flow. However, optical flow information obtained from a single camera is sparse and needs to be further interpreted to understand what the flow pattern actually represents in the real world. Therefore, the second research question is formulated as:

RQ2: How can MAVs autonomously find a suitable landing site with monocular vision?

These two research questions are formulated based on the challenges encountered in the two-level architecture defined above. In general, guidance, navigation, and control (GNC) is a framework of flight control. In that definition, navigation is finding the vehicle's position and velocity along the route. Guidance is comparing positions with desired positions along the route and finding the best trajectory to fly from the current waypoint to the designated waypoint. Control is the way which the vehicles try to follow the guidance trajectory. The relationship to the framework adapted in this thesis is as follows: The low-level control involves the vertical guidance and control of the MAV landing. The vertical guidance is tightly integrated with the vertical control because by just keeping flow divergence constant (i.e., the control) a desired exponential landing trajectory of the MAV (i.e., the guidance) can be achieved. The high-level control relates to a navigation task, i.e., searching for a suitable landing site for the MAV.

1.6. SCOPE OF THE THESIS

To focus on the main goal of this thesis, reflected by the problem statement stated above, the scope of this thesis is defined as follows:

- **MAV type:** This research tests with the most common type of MAV, which is a quadrotor, as shown in Fig. 1.5. This platform is chosen because it has a hovering capability, which is suitable for the flight tests in both indoor and outdoor environments. However, in principle, the algorithms are portable to other types of MAVs.

- **Visual perception:** A small downward-looking camera sensor, which is fixed to the bottom of the MAV, is used for the visual perception. The use of this camera limits the field of view of the vehicle and provides a relatively low image quality. In addition, the camera is a traditional frame-based CMOS camera. However, all findings are also relevant to neuromorphic sensors [57, 58]. This research also assumes that the environment is stationary, and thus, the optical flow is solely generated by the self-motion of the MAV.

- **Information processing:** A feature-based method is used to compute optical flow from the image sequence captured by the camera. This method detects features on images and tracks them over multiple images. It is computationally efficient and can be easily adapted to other common miniature cameras. However, it only provides sparse motion fields. A direct method, on the other hand, gives more dense motion fields by recovering image motion at all pixels with contrast. This method is more computationally expensive and sensitive to appearance variations compared to the feature-based method. Both optical flow computation methods work in textured environments. Since this research focuses on how to use optical flow for autonomous landing of the MAV, the proposed algorithm generalize to direct and other methods of determining optical flow.

- **Lateral control:** The current approach used in this research to perform lateral maneuvers is based on (1) the standard navigation techniques, i.e., using a GPS (outdoor flights) or a motion capture system (indoor flights), for knowing the vehicle position and computing the position of the selected landing site, and (2) the stabilization using optical flow, for hovering control. After the landing position is known, the MAV first navigates to the desired landing position using the standard navigation techniques and then stabilize itself laterally using optical flow during the vertical descent. The lateral control has not yet been discussed in this chapter because it is not the main focus of this research. However, the step to lateral control without a GPS or motion tracking system is not a big one.

Figure 1.5: A quadrotor is used as a testing platform in this research.

1.7. RESEARCH APPROACH

In this thesis, four methods are proposed to answer the two research questions stated above (two methods for each question). These methods can be summarized as:

1. The scaleless property of optical flow (**RQ 1**) in low-level landing is tackled with:
    (a) A monocular distance estimation using knowledge of the applied control input to 'scale' the optical flow, in order to obtain the height and the velocity of the MAV and further use these estimates for landing.
    (b) An adaptive control strategy which automatically adjusts the control gain which is implicitly adapted to the height information to achieve a smooth and high-performance landing.
2. A suitable landing site (**RQ 2**) in high-level landing can be identified using:
    (a) Surface slope and roughness estimated using optical flow field fitting, to know the inclination of a landing surface and to detect obstacles on the ground.

(b) A novel setup of self-supervised learning (SSL) of obstacle appearance using the surface roughness obtained from optical flow, to efficiently detect obstacles by just looking at still images.

These four items are further discussed in detail as follows.

First, to directly tackle the scaleless problem of optical flow control, an algorithm that uses an Extended Kalman Filter (EKF) is introduced to estimate the height and velocity of an MAV from optical flow. In fact, it is challenging to use only the flow divergence (the ratio of velocity to height) observed from a single camera to estimate the height. The proposed concept is to utilize the knowledge of the control input, the so-called efference copy, and the observed flow divergence that results from this input. In the algorithm, the efference copy predicts the effect of an action (the prediction step in EKF), while the flow divergence observes the movement of the MAV (the update step in EKF). By doing that, the height and velocity of an MAV can be estimated.

Second, an adaptive control strategy for constant flow divergence landing is proposed. This strategy allows for a smooth and high-performance landing of the MAV by automatically adjusting the control gains during landing. There are two phases involved in this strategy. The first phase is to determine near-optimal initial control gains in hover by gradually increasing the control gain until an oscillation is detected. The stable gain just before the oscillation is used as the initial gain for the second phase. Here, the constant flow divergence landing is activated with an exponential decay of the control gains, with mechanisms in place to adjust the gains when the actual trajectory deviates from the expected constant flow divergence landing.

Third, optical flow is utilized to search for safe landing sites. These are defined as relatively flat surfaces that do not have a too large inclination and, most importantly, are free of obstacles. To find a safe landing site, an algorithm that fits the optical flow field using RANdom SAmple Consensus (RANSAC) is proposed. This algorithm assumes that the landing surface is planar. The results of the fitting provide two important estimates, i.e., the surface slope which shows the inclination of the ground surface and the surface roughness which indicates the presence of obstacles. When the fit is bad (=high roughness), the plane is likely to include some obstacles. On the other hand, when the fit is good (=low roughness), there are no obstacles of any size.

Finally, an attempt is made in this thesis to 'go beyond' optical flow approaches for the safe landing site selection. Obviously, using optical flow requires the MAVs to move. When there is hardly any vehicle movement, there is not enough visual input to the computer for further interpretation. Optical flow can also be sparse and less informative, e.g., in environments which have a little texture. And also, by only relying on optical flow for obstacle detection, the MAV needs to repeat its optical flow based estimation also for objects that are the same and that have been experienced (perhaps many times) before. A novel setup of self-supervised learning (SSL) is introduced, in which optical flow provides the target for supervised learning of some function. This function maps appearance features to a surface roughness measure. The MAV first uses optical flow to determine the surface roughness, and then simultaneously extracts appearance features represented by texton distributions from each image. Then, it learns a function that maps appearance features to the surface roughness. After learning, the MAV can detect obstacles by just examining appearance features in still images. In this way, the

MAV can find a safe landing site without moving.

1.8. OUTLINE OF THE THESIS

A schematic diagram showing the main content of each chapter is presented in Figure 1.6. The body of this thesis focuses on bio-inspired landing control of an MAV, which is divided into two parts.

The first part, Chapters 2 and 3, presents the low-level landing control of the MAVs which intends to deal with scaleless property of optical flow **RQ 1**. Chapter 2 presents a monocular distance estimation of an MAV using the knowledge of control input and flow divergence. Chapter 3 describes an adaptive control strategy to solve the fundamental problem of gain selection for optical flow landing.

The second part, Chapters 4 and 5, investigates the high-level landing control of the MAVs which aims to find a suitable landing site for an MAV **RQ 2**. Chapter 4 shows an optical flow algorithm to estimate slope and roughness of the landing surface for landing site selection. Chapter 5 introduces a novel setup of self-supervised learning (SSL) of obstacle appearance using surface roughness from optical flow.

Chapter 6 summarizes and concludes the results obtained and gives recommendations for future work.

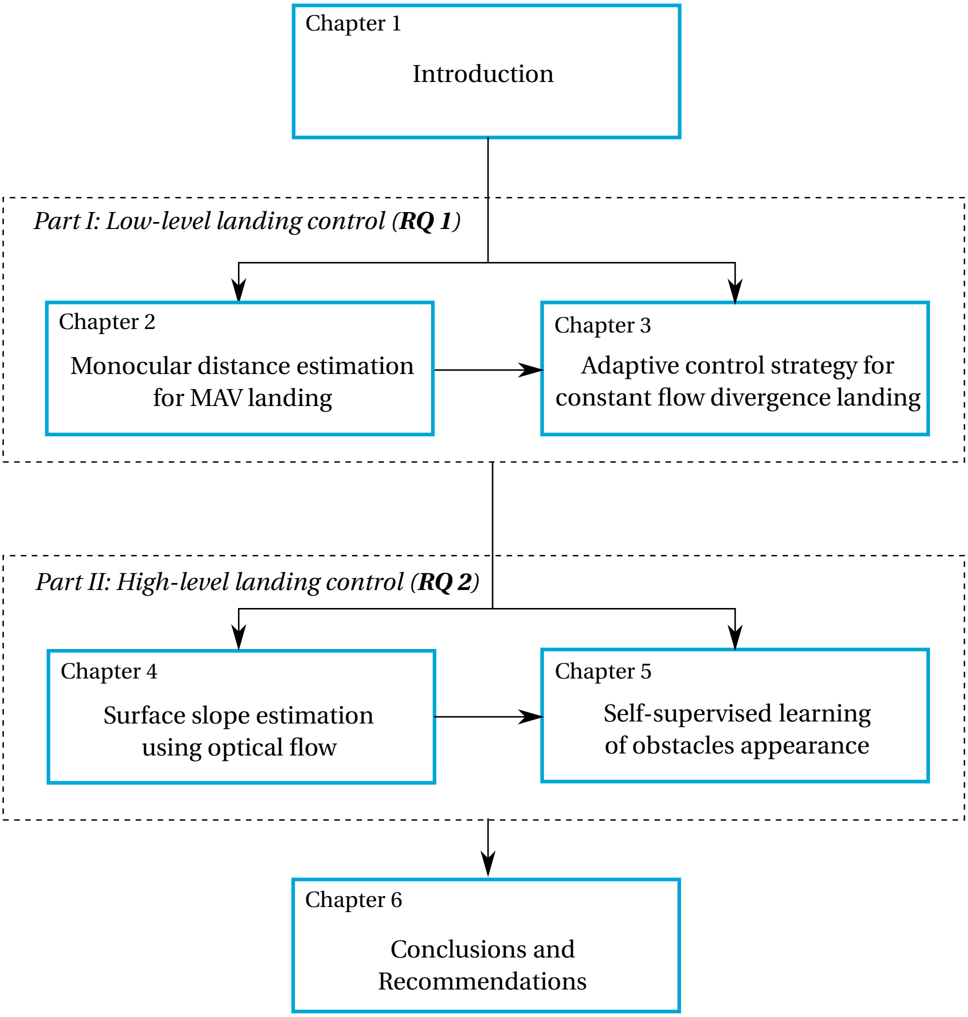

Chapter 1
Introduction
Part I: Low-level landing control (RQ 1)
Chapter 2
Monocular distance estimation for MAV landing
Chapter 3
Adaptive control strategy for constant flow divergence landing
Part II: High-level landing control (RQ 2)
Chapter 4
Surface slope estimation using optical flow
Chapter 5
Self-supervised learning of obstacles appearance
Chapter 6
Conclusions and Recommendations

Figure 1.6: Outline of the thesis.

I

LOW-LEVEL LANDING CONTROL

2

MONOCULAR DISTANCE ESTIMATION FOR MAV LANDING

Previous studies have reported that flying insects, such as honeybees and dragonflies, heavily rely on optical flow for avoiding obstacles while navigating and performing smooth landings [43, 59]. This method is particularly suitable for MAVs, which have limited processing capabilities and payloads. One of the greatest challenges of using optical flow is the fact that optical flow does not directly provide us the distance to an object or velocity, but the ratio of them. This scaleless property of optical flow can lead to various problems of optical flow control. To directly tackle this issue, this chapter introduces an algorithm that estimates the height and vertical velocity of an MAV from the optical flow and the knowledge of the control input, the so-called efference copy.

This chapter begins by providing some knowledge about the flow divergence and the constant flow divergence guidance and control strategy in Section 2.2. After that, Section 2.3 describes the proposed EKF-based height and velocity estimation using the flow divergence and the applied control inputs, the nonlinear observability analysis of the system, and the vision algorithm used to estimate flow divergence with a single camera. Then, Section 2.4 and 2.5 shows the simulation results of the estimation and the results of several landings of an MAV, respectively. Lastly, a conclusion with future work is drawn in Section 2.6.

This chapter is based on the following article.
H. W. Ho, G. C. H. E. de Croon, and Q. P. Chu, Distance and velocity estimation using optical flow from a monocular camera, International Journal of Micro Air Vehicles (accepted) [60]

2.1. INTRODUCTION

Micro Air Vehicles (MAVs) have gained in popularity in recent years. Due to their small size, they are easier and safer to use than large Unmanned Aerial Vehicles (UAVs). However, the size and weight constraints make the MAVs more challenging to be deployed for autonomous missions. To deal with the problems, it requires reducing the amount of payload or sensors and developing more intelligent and efficient algorithms.

Optical flow, which can be extracted from a monocular camera, provides a very promising solution for miniaturization. Optical flow refers to the apparent visual motion of objects in a scene relative to an observer [40]. This information tells us not only how fast the camera moves, but also how close it is relative to the things it sees. Flying insects use it as the main visual cue for sensing their own movements and surrounding objects while flying. They can perform complex tasks, such as landing, by only relying on this visual input and using limited neural resources. For instance, honeybees heavily use optical flow to perceive the environment and avoid dangerous objects while flying [43, 59].

To accomplish vertical landings, honeybees use a divergent pattern of optical flow or the so-called flow divergence. By keeping the flow divergence constant, honeybees can perform smooth landings. This strategy leads to exponential decay of both height and velocity to zero at touchdown, and it is ideal for MAVs landings [44, 53, 54, 61]. Often, a straightforward proportional or proportional integral (P or PI) feedback controller is used for MAVs to perform constant flow divergence landing.

However, in our previous studies, we found that fixed-gain controls used in constant flow divergence can lead to instability of the landing [50, 62]. This is because the critical control gain is directly proportional to the height [50]. Most of the current approaches to solving this problem are to use a gain-scheduling technique which is actually designed according to the knowledge of an initial height [51, 52]. However, if no additional sensor is installed, the height is unknown. Besides, if the initial height deviates too much from the one which is used to design the gain-scheduling structure, this method will not work properly. Thus, the knowledge of the height is important for optical flow control.

In fact, optical flow does not give us the absolute distance to a surface or velocity of the camera directly. There are four known methods to estimate distances from optical flow. First, one can add sensors that directly or indirectly measure distance, velocity, or accelerations. The right type of filtering and fusion can then give distance estimates. For instance, some studies include an Inertial Measurement Unit (IMU), a camera sensor, and a pressure sensor, and use a Kalman-based sensor fusion technique to estimate distances [63, 64].

Second, when performing a perfect constant flow divergence landing, the control input (the thrust) or so-called efference copy follows a specific monotonously decreasing function over time, which can be used to estimate distances [65]. They demonstrated this approach by moving a camera along a track toward an image scene to first estimate an initial distance. Since the constant flow divergence control will result in an exponential decay of distances, they simplified the estimation problem to an exponential propagation of distances after obtaining the initial value.

Third, a stability approach that detects self-induced oscillation caused by high controller gains and uses these gains to estimate distances was introduced [50]. It was shown analytically that there exists a directly proportional relationship between the critical control gain, i.e., the gain which causes instability, and the height. Based on this relationship, the MAV detects self-induced oscillation and uses these gains to estimate the height.

Fourth, by knowing the effect of the control inputs, one can “scale” the optical flow to obtain the distance and velocity. An early study on the dynamic effects in visual servoing showed that the control gain is a function of the distance [66]. It was used in an adaptive control scheme for visual servoing [67]. The relationship also implies that the closed-loop response depends on the distance which opens up the possibility of estimating the distance to a target from closed-loop dynamics.

In this paper, we build upon the principle presented in the fourth method to estimate the height (distance to the ground) and vertical velocity of an MAV applying it for the first time to optical flow landing. An extended Kalman filter (EKF) is used to estimate the height and vertical velocity of an MAV using the knowledge of the control input in combination with the flow divergence observed from a monocular camera. This algorithm uses the efference copy to predict the effects of an action, and the observed flow divergence to correct the prediction. The significant advantage of this method is that we simplify the nonlinear control of the constant flow divergence to a linear control that uses height and velocity estimates. Thus, this provides the possibility of using some optimization techniques with vision output from a monocular camera to improve the performance of the control.

The remainder of the paper is set up as follows: In the first section, we provide some knowledge about the flow divergence and the constant flow divergence guidance and control strategy. The following section describes the proposed EKF-based height and velocity estimation using the flow divergence and the control inputs, the nonlinear observability analysis of the system, and vision algorithms used to estimate flow divergence with a monocular camera. Then, the next section shows the results of estimation in computer simulation while in the succeeding section we also show the results of multiple landings of an MAV. Finally, a conclusion with future work is drawn.

2.2. BACKGROUND

2.2.1. FLOW DIVERGENCE

For vertical landing of an MAV, flow divergence (D) or visual looming as shown in Fig. 2.1 can be computed as follows:

$$D = \frac{V_Z}{Z}. \tag{2.1}$$

where Z is the distance to the MAV from the ground in the direction of Z^w and V_Z is the rate of change of Z. When the MAV is approaching the ground, we measure positive Z, negative V_Z, and thus $D < 0$ according to the coordinate systems shown in Fig. 2.2. The camera coordinate system is assumed to coincide with the body coordinate system.

2.2.2. CONSTANT FLOW DIVERGENCE GUIDANCE AND CONTROL

Constant flow divergence approach has been used for vertical landing of the MAVs. This approach controls the vertical dynamics of the MAVs by tracking a constant flow divergence. When $|D|$ equals a positive constant c, we can derive from Eq. 2.1 the height

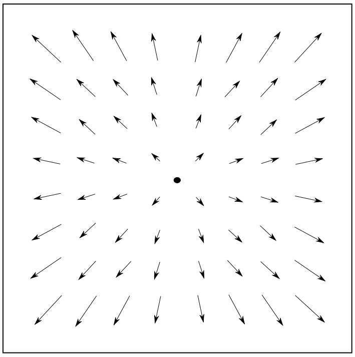


Figure 2.1: Divergence of optical flow vectors (flow divergence) when the observer is approaching a surface.

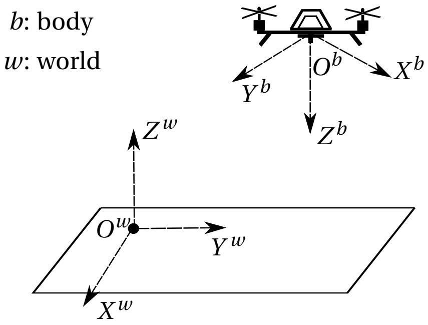


Figure 2.2: MAV body ($O^bX^bY^bZ^b$) and world ($O^wX^wY^wZ^w$) coordinate systems.

$Z = Z_0e^{-ct}$, and velocity $V_Z = -cZ_0e^{-ct}$, where Z_0 is the initial height. This shows that both height and velocity will decrease exponentially and eventually become zero when touching the ground.

A proportional feedback controller is used to track the desired flow divergence D^* as shown in Eq. (2.2):

$$\mu = K_p(D^* - D), \tag{2.2}$$

where K_p is the gain of the proportional controller.

A double integrator system is used to model the dynamics of an MAV towards the ground in one-dimensional space. The continuous state space model can be written as:

$$\begin{aligned} \dot{\mathbf{r}}(t) = f\big(\mathbf{r}(t), \mu(t)\big) &= \mathbf{A} \cdot \mathbf{r}(t) + \mathbf{B} \cdot \mu(t) \\ &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{r}(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mu(t), \end{aligned} \tag{2.3}$$

$$y(t) = h\big(\mathbf{r}(t)\big) = [r_2(t)/r_1(t)] = D, \tag{2.4}$$

where $\mathbf{r} = [r_1, r_2]^T = [Z, V_Z]^T$ and μ is the control input.

It is clear that the model dynamics in Eq. (2.3) are linear but the observation in Eq. (2.4) is nonlinear. Fig. 2.3 shows a time response of the system when tracking a constant flow divergence. In this figure, we can see that the MAV accelerates in the first 2.5 s and then decelerates to zero velocity to touch the ground. Both height and velocity of the MAV decrease exponentially to zero in the end.

2.3. EKF-BASED HEIGHT AND VELOCITY ESTIMATION USING FLOW DIVERGENCE AND CONTROL INPUT

An overview of the methodology is presented in Fig. 2.4. Images captured by a camera are served as the inputs to the software while the control input commands the actua-

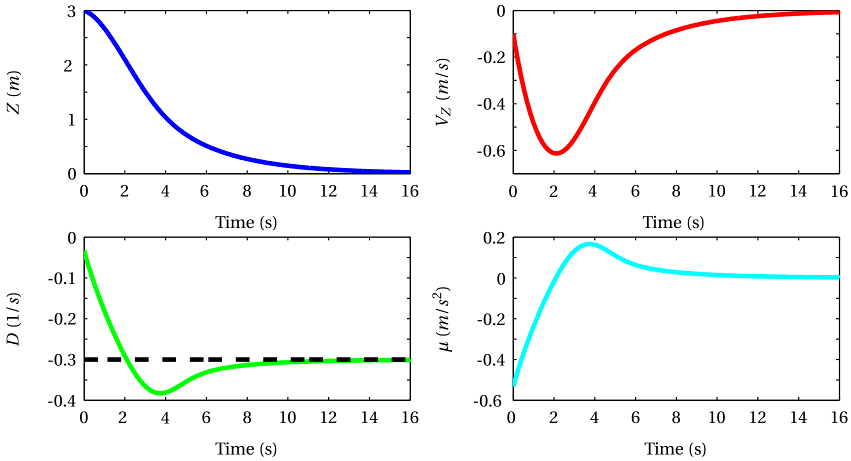


Figure 2.3: Height Z, velocity V_Z, and flow divergence D of an MAV during landing using constant flow divergence strategy ($D^* = -0.3\ s^{-1}$) with control input μ.

tor to control the MAV. In this section, we describe: (1) an extended Kalman filter (EKF) algorithm using the flow divergence and efference copies to estimate the height and velocity of an MAV during landing, (2) a nonlinear observability analysis of the system, (3) a vision method to compute the flow divergence.

2.3.1. EXTENDED KALMAN FILTER

In practice, the flow divergence is computed using an on-board processor. Therefore, we used a discrete-time extended Kalman filter to estimate the height and vertical velocity of the MAV. The system model and observation model are shown in Eq. 2.5 and 2.6, respectively.

$$\dot{\mathbf{x}}(t) = f\big(\mathbf{x}(t), \mu(t)\big) + \mathbf{w}(t). \tag{2.5}$$

$$\mathbf{z}_k = h\big(\mathbf{x}_k\big) + \mathbf{v}_k. \tag{2.6}$$

where $f(\cdot)$ and $h(\cdot)$ are the system matrix and the observation matrix which are obtained from Eq. 2.3 and 2.4, respectively. $\mathbf{w}(t)$ and $\mathbf{v}_k$ are the system noise and observation noise which are both assumed to be zero mean multivariate Gaussian process with covariance $\mathbf{Q}$ and $\mathbf{R}$, respectively.

Several computational steps taken in EKF when the update of flow divergence is obtained are shown below:

a) One-step ahead prediction:

$$\hat{\mathbf{x}}_{k|k-1} = \hat{\mathbf{x}}_{k-1|k-1} + \int_{t_{k-1}}^{t_k} f\big(\mathbf{x}(\tau), \mu(\tau)\big) \mathrm{d}\tau \tag{2.7}$$

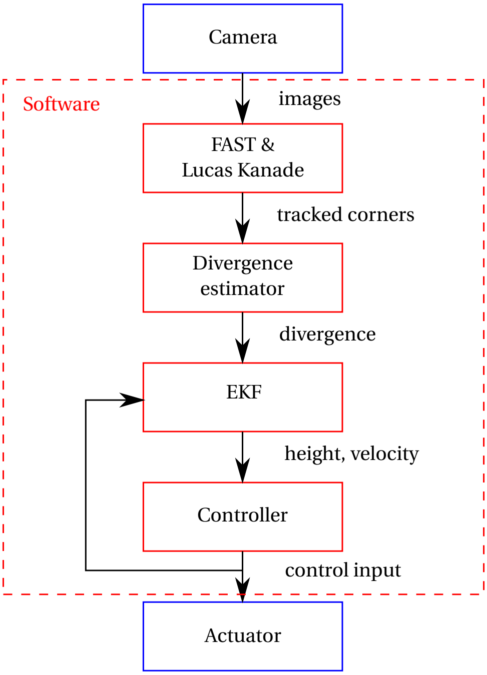


Figure 2.4: An overview of the methodology.

b) Covariance matrix of the state prediction error vector:

$$\mathbf{P}_{k|k-1} = \Phi_k \mathbf{P}_{k-1|k-1} \Phi_k^T + \Gamma_k \mathbf{Q} \Gamma_k^T \tag{2.8}$$

where Φ and Γ are the discretized matrices of **A** and **B**, and can be computed as below (their derivations can be found in the appendix A):

$$\Phi_k = \begin{bmatrix} 1 & t_k - t_{k-1} \\ 0 & 1 \end{bmatrix}, \quad \Gamma_k = \begin{bmatrix} (t_k - t_{k-1})^2/2 \\ t_k - t_{k-1} \end{bmatrix} \tag{2.9}$$

c) Kalman gain:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^\top \left(\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^\top + \mathbf{R} \right)^{-1} \tag{2.10}$$

where

$$\mathbf{H}_k = \frac{\partial h}{\partial \mathbf{x}} \bigg|_{\hat{\mathbf{x}}_{k|k-1}} = \left[-\frac{\hat{x}_{2_{k|k-1}}}{\hat{x}^2_{1_{k|k-1}}}, \frac{1}{\hat{x}_{1_{k|k-1}}} \right]^T \tag{2.11}$$

d) Measurement update:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \left(\mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1}) \right) \tag{2.12}$$

where $\mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1})$ is called the innovation of EKF.
e) Covariance matrix of state estimation error vector:

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \mathbf{R} \mathbf{K}_k^T \tag{2.13}$$

2.3.2. NONLINEAR OBSERVABILITY OF THE ESTIMATION

In this subsection, we show that with the flow divergence and the efference copy/control input the system is observable, i.e., any distinct states are distinguishable by applying a bounded measurable input (e.g., a piecewise constant input). To check the observability of the nonlinear system, the observability rank condition can be used ([68]).

We first construct the observability algebra which consists of repeated Lie derivatives of the observation model in Eq.2.4, $h\big(\mathbf{r}(t)\big) = r_2(t)/r_1(t)$, with respect to the model dynamics in Eq.2.3, $f\big(\mathbf{r}(t), \mu(t)\big) = [r_2(t), \mu]^T$. The system is observable if and only if the Jacobian of the observability algebra is full rank. In our case, the observability algebra $\mathcal{O}$ can be written as:

$$
\begin{aligned}
\mathcal{O} = \begin{bmatrix} L_f^0 h\big(\mathbf{r}(t)\big) \\ L_f^1 h\big(\mathbf{r}(t)\big) \end{bmatrix} &= \begin{bmatrix} h\big(\mathbf{r}(t)\big) \\ \frac{\partial h(\mathbf{r}(t))}{\partial \mathbf{r}(t)} \cdot f\big(\mathbf{r}(t), \mu(t)\big) \end{bmatrix} \\
&= \begin{bmatrix} r_2(t)/r_1(t) \\ -(r_2(t))^2/(r_1(t))^2 + \mu/(r_1(t)) \end{bmatrix}
\end{aligned}
\tag{2.14}
$$

Next, we analyze $\mathcal{O}$ to see whether its Jacobian is full rank. The first function of the observability algebra is $y(t) = h\big(\mathbf{r}(t)\big) = [r_2(t)/r_1(t)] = D$ from Eq.2.4 while its second function is the first Lie derivative with respect to the dynamics which equals to $-(r_2(t))^2/(r_1(t))^2 + \mu/(r_1(t)) = -(h\big(\mathbf{r}(t)\big))^2 + \mu/(r_1(t))$. From this result, we can clearly see that the system is (locally) observable (i.e., a full rank Jacobian of $\mathcal{O}$) if and only if the control input, $\mu \neq 0$, $r_1(t) \neq 0$, and $r_2(t) \neq 0$. In other words, we are able to estimate the distinct states, i.e., Z and V_Z, by using both flow divergence and control input in the algorithm.

2.3.3. FEATURES-BASED FLOW DIVERGENCE ESTIMATION

In computer simulation, we used Eq. 2.1 to compute flow divergence, while in flight tests, we estimated flow divergence based on Eq. 2.15 (Proof can be found in [62].). For each image captured by an on-board camera, corners were detected using the FAST algorithm [69, 70], and tracked in the next image using the Lucas-Kanade tracker [71]. Then, the image distances between every two corners at one image, $d_{(t-\Delta t),i}$ and at the next image, $d_{t,i}$ were computed. By further computing the ratio of $d_{(t-\Delta t),i} - d_{t,i}$ to $d_{(t-\Delta t),i}$, we can measure the expansion and contraction of the flow. We took the average of these ratios, and with a known time interval between these two images Δt, we estimated divergence using Eq. 2.15:

$$
\widehat{D} = \frac{1}{n} \cdot \frac{1}{\Delta t} \sum_{i=1}^{n} [\frac{d_{(t-\Delta t),i} - d_{t,i}}{d_{(t-\Delta t),i}}],
\tag{2.15}
$$

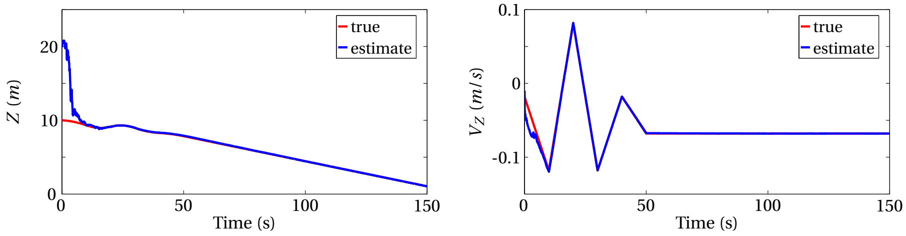
where n is the total number of tracked corners. In practice, the vision output is often noisy. Therefore, we used a low pass filter to reduce the noise in the estimation.

2.4. COMPUTER SIMULATION

Before performing flight tests, we simulated the proposed algorithm presented in the previous section in MATLAB to show the feasibility of the algorithm.

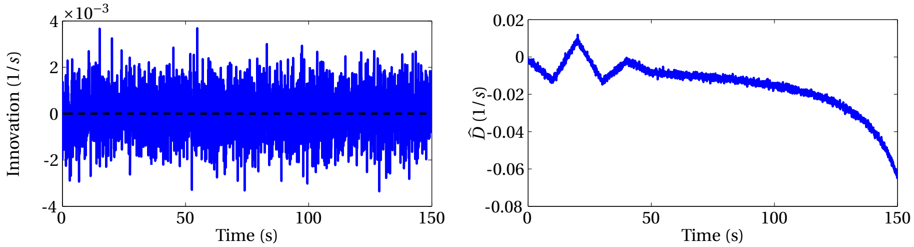
2.4.1. LANDING SIMULATION WITH SIMULATED CONTROL INPUTS

In the simulation, we generated the height and velocity with a timestamp of 0.05 s using a set of control inputs, μ as shown in Fig. 2.5e. These data served as ground truth for validation. The flow divergence measurement was generated using Eq. 2.1 with a measurement noise standard deviation of 0.001 s^{-1} as illustrated in Fig. 2.5d.



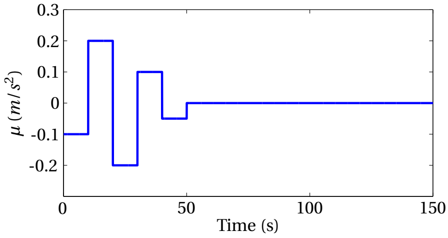
(a) Height.

(b) Velocity.



(c) Innovation.

(d) Flow divergence.



(e) Control input.

Figure 2.5: EKF-based height and vertical velocity estimation from flow divergence for landing control.

By using the control input, μ and the flow divergence measurement, D, we estimated the height, Z and the velocity, V_Z with the proposed EKF algorithm. Fig. 2.5 shows estimated states (Z and V_Z) and their ground truth, innovation of the EKF, flow divergence measurement, and the control inputs. In this simulation, we can observe that the estimated height and velocity converge and follow the ground truth after a few seconds, even with different initial conditions (Z_0 and V_{Z_0}) than the actual values. In addition, the innovation of EKF has zero mean implying that the filter is working correctly. Simulation results show that the proposed algorithm is able to estimate the distance to the ground and velocity accurately.

2.5. EXPERIMENTS RESULTS AND DISCUSSION

We implemented the EKF algorithm and vision algorithms in Paparazzi Autopilot, an open source autopilot software [72]. A Parrot AR.Drone 2.0 equipped with a downward-looking camera was used as a testing platform, and all algorithms were running on-board the MAV. We used an *OptiTrack* system to track the position of the MAV in order to provide the ground truth of its height and velocity *only for validation purposes*, and these measurements were *not* used in the estimation. In this section, we show the feasibility of the algorithm by performing three different control strategies for MAVs landings.

Before executing the algorithm in real-world experiments, we need to check the relationship between the control input μ (which is assumed to be the acceleration in the model) and the resulting acceleration a_Z imposed on our platform, e.g., $a_Z = f(\mu)$. Fig. 2.6 shows the vertical acceleration a_Z resulted from the control input μ used in performing a constant flow divergence landing. The acceleration measurement was obtained from the on-board IMU but was *not* used in the following flight tests. Knowing that the variation of the control input μ from the hovering command μ_g is small (total command to motors $\mu_T = \mu_g + \mu$), a linear function is used to map the control input to the acceleration. From the mapping, we obtained $a_z = 9.906\mu + 0.047$ plotted as a dashed line in the figure.

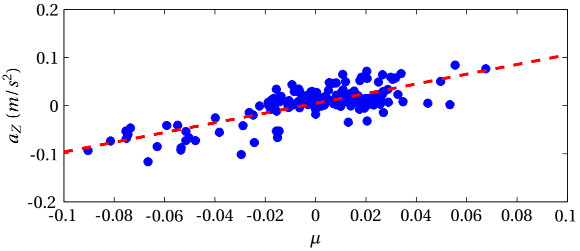


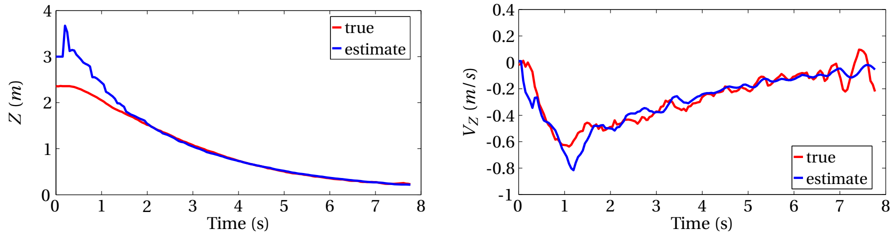
Figure 2.6: Mapping of the control input μ to the vertical acceleration a_Z using a linear function ($a_z = 9.906\mu + 0.047$).

2.5.1. LANDING USING CONSTANT FLOW DIVERGENCE WITH FIXED-GAIN CONTROL

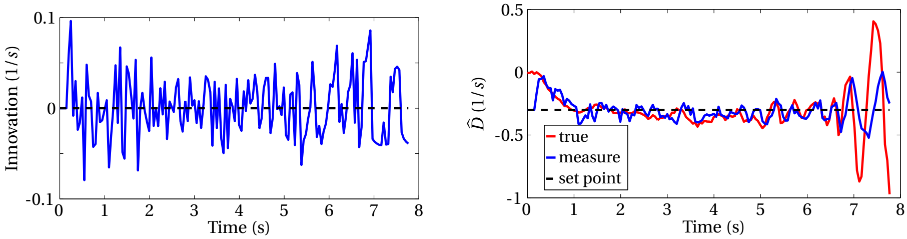
The first strategy we used in the flight tests is the constant flow divergence. A flow divergence set point was tracked for the entire landing. During the landing, the EKF algorithm was running in real-time to estimate the height and velocity of the MAV.

Fig. 2.7 shows the estimation results of the landing with a constant divergence of $-0.3\ s^{-1}$ using a fixed-gain control. The initial value of the height for EKF was set to be $\hat{Z}_0 = 3m$ which was different from the true initial height, i.e., $Z_0 \approx 2m$. In the figure, we can see that the height estimate converges to the true height, even though the initial value is different. In addition, the velocity estimate also matches well with the true velocity.

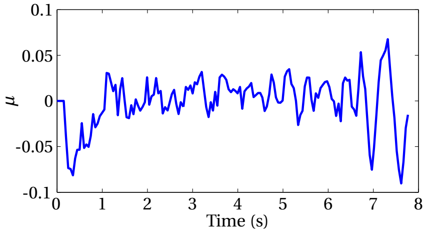
From the landing results, we can observe that this strategy exponentially decreases both height and velocity to nearly zero by only tracking a constant flow divergence. In fact, the drawback of vision algorithms is that when the image is too dark, the algorithms



(a) Height. (b) Velocity.



(c) Innovation. (d) Flow divergence.



(e) Control input.

Figure 2.7: Fixed-gain landing with constant flow divergence ($D^* = -0.3\ s^{-1}$) with EKF-based height and velocity estimation.

will not work properly. For instance, when the camera is too close to the ground, the flow divergence estimate becomes incorrect. To deal with that, we constantly decrease the trim throttle when the MAV is very close to the ground ($Z < 0.2m$) to allow the MAV for touchdown. Another fact is that a fine-tuned control gain or gains adapted with the height are needed to solve the instability issue [62]. This issue can be seen from the measured flow divergence in Fig. 2.7d when the MAV is close to the ground.

To show the reliability of the proposed algorithm, we performed multiple landings with different flow divergence set points ($D^* = -0.1, -0.2, -0.3\ s^{-1}$) at different initial heights ($Z_0 \approx 2, 3m$). The same initial guess of the height ($3m$) was used in the EKF algorithm for these flight tests. Fig. 2.8 shows the height and velocity estimates using EKF algorithm in different landings. In this figure, we can see that both height and velocity estimates match well with the true values. With a larger set point, the MAV landed faster with the exponential decrease in both height and velocity. Besides, these results also show that a good guess of the initial height has a faster convergence of the estimates.

The height and velocity estimated from EKF can then be adapted to tune the controller gain in real-time for the constant divergence landing. This will be discussed in

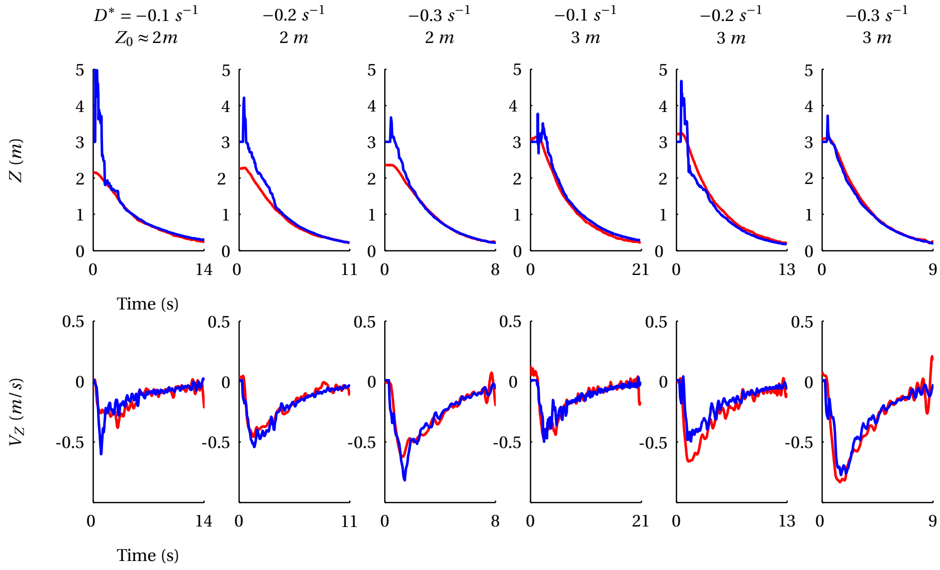


Figure 2.8: Fixed-gain multiple landings with different flow divergence set points ($D^* = -0.1\ s^{-1}, -0.2\ s^{-1}, -0.3\ s^{-1}$) at different initial heights ($Z_0 = 2m, 3m$). The estimates are shown with blue lines while their true values are represented by red lines.

the coming section. By doing that, oscillations can be reduced, and thus a smooth landing can be achieved by only using a monocular camera.

2.5.2. LANDING USING CONSTANT FLOW DIVERGENCE WITH HEIGHT-ADAPTED CONTROL

Experiments presented in the previous subsection were performed using a manually-tuned control gain in order to achieve stable landings. In fact, the critical control gain, i.e., the control gain which causes instability, is directly proportional to the height [50]. This means that if a fixed gain is used in constant flow divergence landing, oscillation will occur at a certain height. When the height is known from the proposed algorithm, it can be used to automatically adjust the control gain in such a way that the oscillation is eliminated [62]. This second strategy is used to keep the advantages of using constant flow divergence landing, i.e., achieving an exponential landing profile by just tracking a desired flow divergence set point, and avoid the instability due to a fixed-gain control.

By letting $K_p = f(Z) = k \cdot Z$ where k is a positive constant, the proportional controller in Eq. 2.2, $\mu = K_p(D^* - D)$ becomes:

$$\begin{aligned}\mu &= k \cdot Z(D^* - \frac{V_Z}{Z}) \\ &= kD^* Z - kV_Z \\ &= -\mathbf{k} \cdot \mathbf{r}(t), \qquad (2.16)\end{aligned}$$

where $\mathbf{k} = [k_1 \quad k_2] = [-kD^* \quad k]$.

This controller simplifies the problem to a linear control problem. To find k, we can first solve this linear control problem using the pole-zero analysis. From Eq. 2.3, the model dynamics becomes:

$$\begin{aligned}\dot{\mathbf{r}}(t) &= \mathbf{A}\cdot\mathbf{r}(t)+\mathbf{B}\cdot(-\mathbf{k}\cdot\mathbf{r}(t))\\ &= (\mathbf{A}-\mathbf{B}\cdot\mathbf{k})\cdot\mathbf{r}(t)\\ &= \begin{bmatrix} 0 & 1\\ -k_1 & -k_2\end{bmatrix}\cdot\mathbf{r}(t).\end{aligned} \tag{2.17}$$

To achieve stable conditions, all the poles should lie in the left half of the s-plane. The poles of the system are obtained as shown below:

$$\lambda_{1,2}=\frac{-k_2\pm\sqrt{k_2^2-4k_1}}{2}. \tag{2.18}$$

Therefore, for a stable system, $k_2^2 \geq 4k_1$. By substituting $k_1=-kD^*$ and $k_2=k$, we get $k^2\geq -4kD^*$ and thus $k(k-4c)\geq 0$, where $c=-D^*$. Since $k>0$, $k\geq 4c$.

Three flight tests were performed at different flow divergence set points (D^*=-0.1, -0.2, -0.3 s^{-1}) shown in Fig. 2.9. The initial guess of the height for EKF was set to be different from the true height in order to test the convergence of the estimates. When the landing was activated, the control gain was adapted to the height according to $K_p=k\cdot\widehat{Z}$. Note that the value of K_p used in the real flight needs to be mapped according to the result shown in Fig. 2.6. Since the initial height was selected to be much higher than the true height, it was expected that a slight oscillation occurred at the beginning of the landing. After the height estimate converged to the true height, the landing was smooth as the correct gain was used. In addition, it can be seen from the last column of Fig. 2.9 that the tracking of these three different D^* is good. This strategy ensures a stable and high-performance landing.

2.5.3. LANDING USING HEIGHT AND VELOCITY FROM EKF

The third strategy we used for landing is tracking a height profile using the estimates from EKF. To this end, the landing begins by (1) giving an initial excitation to the MAV, or (2) using constant divergence strategy, for the first few seconds. This can ensure that the EKF converges and the controller tracks the right height estimate.

This first initialization was performed in an indoor flight test by giving a block input as the excitation as shown in Fig. 2.10e to the MAV for the first 0.5 s. This allowed the MAV to go down or move in order to 'observe' flow divergence and initialize the EKF algorithm. After that, the controller used the estimates from the EKF to track a desired profile or set points for landing.

Fig. 2.10 shows the height and velocity estimates from EKF compared with their ground truth, the innovation of EKF, the flow divergence from the vision algorithms and the ground truth, and the control input. After giving the initial excitation, the MAV started to track a height profile represented by the black dash-dot line. This profile was generated using $x_{1,k}^*=x_{1,k-1}^*+x_2^*\cdot\Delta t$ with $x_{1,0}^*=\hat{Z}_0$ and $x_2^*=-0.2ms^{-1}$. The results show

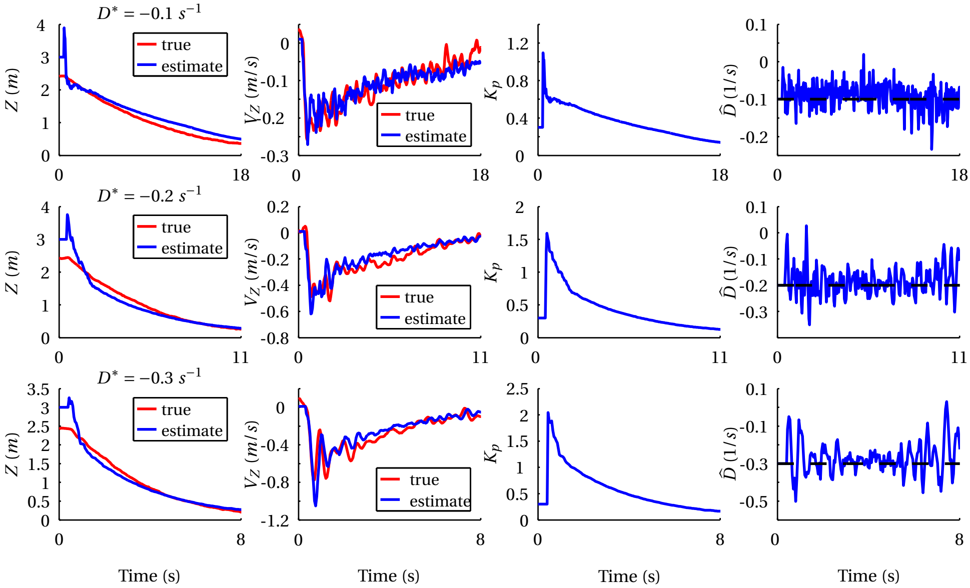


Figure 2.9: Height-adapted control for constant flow divergence landings with different flow divergence set points ($D^* = -0.1, -0.2, -0.3\ s^{-1}$).

that the height and velocity can be estimated accurately using the proposed EKF algorithm and further used in the controller for landing. The zero mean innovation of the EKF also tells us that the filter was working properly.

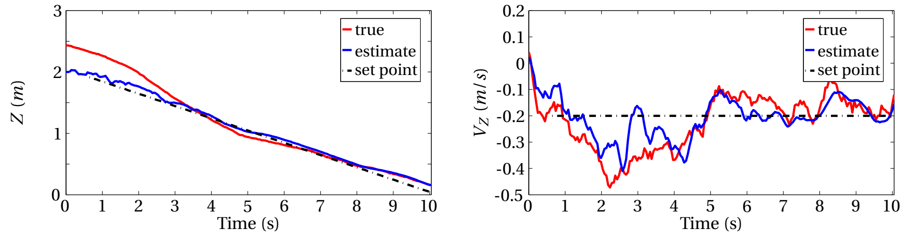
The second initialization was conducted in an outdoor flight by tracking a desired divergence for the first few seconds as shown in Fig. 2.11. Since the *OptiTrack* system is not available for outdoor flight, we can only use the onboard sensor, such as sonar, to provide the true height and velocity (dZ/dt) for accuracy comparison.

The landing started at 5 m, and the desired divergence of $-0.2\ s^{-1}$ was tracked. After 2 s, the controller was switched to track a height profile as presented in Fig. 2.11a which was generated based on the velocity estimate at the instance when the controller was switched ($\approx -0.8\ ms^{-1}$). In fact, the height profile can also be created based on the desired velocity (see Fig. 2.10). The results show that both height and velocity estimates are accurately estimated, compared to the measurements from sonar. In addition, the height estimate can also be used in landing control. It can also be observed from Fig. 2.11 that the estimates converge soon after the landing starts. Thus, the time interval for initialization can also be shortened. Note that the previous indoor flights were performed to validate the proposed algorithm using the highly accurate *OptiTrack* system while the outdoor flight was conducted to show the robustness of the algorithm to a windy outdoor environment. Some videos of the flight tests are available online [1].

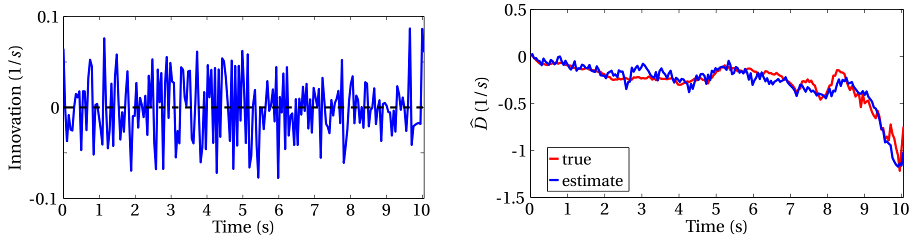
In this paper, we presented three landing strategies using: (1) constant flow diver-

---

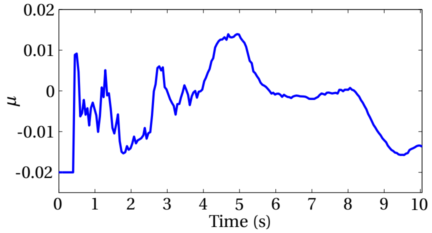
[1] Experiment videos: `https://goo.gl/KiJurr`



(a) Height. (b) Velocity.



(c) Innovation. (d) Flow divergence.



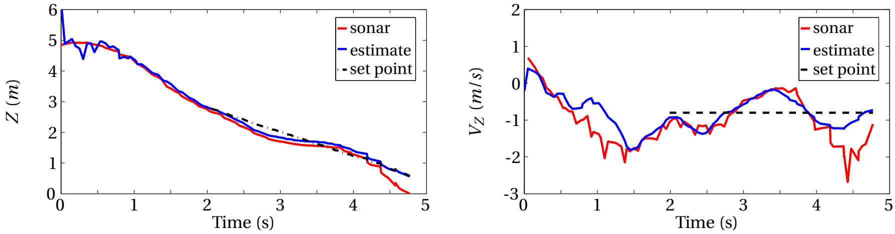
(e) Control input.

Figure 2.10: Landing with a linear controller using estimates from EKF (Indoor Flight).

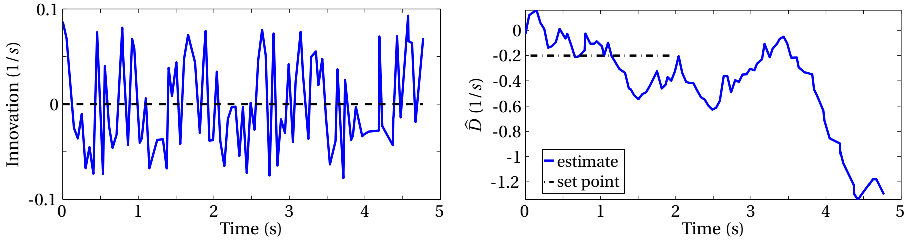
gence with fixed-gain control, (2) constant flow divergence with height-adapted control, and (3) height and velocity estimates. All these strategies can provide a smooth landing solution for MAVs. The first strategy, however, is likely to cause instabilities at some points during the landing. The second strategy solves this problem by adapting height estimate from EKF to the control gain. This allows stable landings without affecting the tracking performance. Due to fast convergence of height, we only observe a slight oscillation at the start of the landings. Lastly, the third strategy uses the height and velocity from EKF directly for landing. Although it requires an initial excitation for observing optical flow, this strategy lands the MAV by following a desired landing profile and allows further performance improvement using gain optimization techniques.

2.5.4. DISCUSSION ON WIND EFFECT AND REJECTION

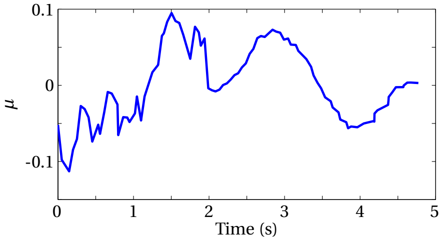
In principle, external disturbance, such as wind, has no problem in control as it can be rejected easily. For instance, when using a proportional feedback control with height estimates, the steady-state error due to wind can occur, but it can be eliminated using the analysis of final value theorem on our model with the wind. From this analysis, we



(a) Height. (b) Velocity.



(c) Innovation. (d) Flow divergence.



(e) Control input.

Figure 2.11: Landing with a linear controller using estimates from EKF (Outdoor Flight).

found that in order to reject for instance a constant wind, the proportional control gain should be $K_P = d_0/x(\infty)$, where $x(\infty)$ is the required steady state error, and d_0 is the wind disturbance estimate.

However, for state estimation, the wind as the external force is not accounted for in our model at the moment. The practical experiment performed in a windy outdoor environment shows that this factor might not be problematic. However, a more thorough investigation and a solution for this problem might be necessary. For instance, we can augment the state vector in Eq.2.3 with the wind disturbance as an additional state, i.e., $\dot{d}_{wind} = w$ (w can be modeled by the random walk [73, 74]) and modify the second state to be $\dot{V}_Z = \mu + d_{wind}$. By doing that, the wind factor is included in the model, and thus the estimation can be improved in the presence of the wind [75]. Both discussions to deal with external disturbance will be the future work of this study.

2.6. CONCLUSION

In conclusion, we proposed an algorithm to use an extended Kalman filter (EKF) to estimate the height and vertical velocity of an MAV from the flow divergence and the knowledge of the control input while approaching a surface and use these estimates for landing control. This algorithm was tested in computer simulation as well as in multiple flight tests in indoor and windy outdoor environments. The results show that the proposed EKF approach managed to estimate the height and velocity of the MAV accurately compared to the ground truth provided by the external cameras. In addition, the estimates were used in the controller to smoothly land the MAV. The proposed approach avoids the complexity of having nonlinearity in the constant flow divergence based landing by 'splitting' the flow divergence into the height and velocity which allows the use of a linear controller. For future work, an augmented state Kalman filter can be used to improve the estimation when strong external disturbances are involved. Also, the algorithm opens up the possibilities to use gain optimization techniques to improve the control performance.

3

ADAPTIVE CONTROL STRATEGY FOR CONSTANT FLOW DIVERGENCE LANDING

In the previous chapter, knowledge of the thrust is used to deal with the scaleless problem of optical flow control. In this chapter, a novel control strategy is introduced to automatically adjust the control gain during the constant flow divergence landing. This strategy has two phases. At the beginning of a landing, the MAV detects the height using an oscillating movement and set the gains accordingly (Phase *I*). After that, the gains are reduced exponentially during descent and adjusted properly when the actual trajectory deviates too much from the expected constant flow divergence landing (Phase *II*). This strategy allows for a stable and high-performance landing of the MAV with optical flow. It does not rely on knowledge of the thrust or any additional sensor.

This chapter first gives some background on the constant flow divergence guidance strategy, and show results of computer simulations with a conventional control scheme assuming a perfect flow divergence estimate in Section 3.2. After that, Section 3.3 shows a characterization of the flow divergence estimates as obtained with a single camera. Then, Section 3.4 analyzes the conventional closed-loop control with the delay and noisy estimates in both computer simulation and flight test. Next, Section 3.5 introduces the adaptive control scheme to deal with the instability problems encountered with the conventional control scheme. In Section 3.6 demonstrates the real-world experiments. Finally, conclusions and recommendations are given in Section 3.7.

This chapter is based on the following articles.

H. W. Ho and G. C. H. E. de Croon, Characterization of flow field divergence for MAVs vertical control landing, in AIAA Guidance, Navigation, and Control Conference (San Diego, California, USA, 2016) p. 0106. [76]

H. W. Ho, G. C. H. E. de Croon, E. van Kampen, Q. P. Chu, and M. Mulder, Adaptive control strategy for constant optical flow divergence landing, IEEE transactions on robotics (submitted). [62]

3.1. INTRODUCTION

Autonomous landing is challenging for Micro Air Vehicles (MAVs), which have payload constraints and limited computing capability. Many earlier studies have used traditional methods of sensing and navigating involving active sensors, such as a laser range finder [15, 77], or a stereo camera [17, 19]. Although they give accurate and redundant measurements, they are costly and heavy for MAVs. In addition, methods using stereo camera are limited in their perception range. A monocular camera would be preferred for MAVs, also due to its light weight and low power consumption [78, 79].

Visual Simultaneous Localization and Mapping (SLAM) is the most commonly used method for navigating using a monocular camera. This method locates all the detected features in the camera field of view and determines the vehicle's location and 3D-structure of the landing surface at these points [22, 80]. Although its computational efficiency and accuracy has been improved over the years [23, 24, 81], visual SLAM still requires more computational resources than strictly necessary for landing.

Besides SLAM, another method is inspired by tiny flying insects, which accomplish complex flight control tasks using limited neural and sensory resources [33]. For instance, honeybees mainly rely on their eyes to perform smooth landings [36, 37]. They possess extremely efficient and robust solutions to landing problems. These bio-inspired solutions could provide design principles for flight control strategies in MAVs [38, 39].

In flying insects, optical flow is probably the mostly used source of visual information. When approaching a ground surface, the expansion of the flow (flow divergence) provides a perception of the observer's relative motion to the ground. Honeybees reduce their speeds to almost zero at touchdown by keeping flow divergence constant [44, 49]. This strategy is typically praised, since it does not seem to rely on knowledge about the height and approaching speed. Optical flow only captures the ratio of height and velocity. Several studies have implemented a constant flow divergence strategy with a fixed-gain controller, e.g., on an MAV for landing [53, 54]. Additionally, time-to-contact (the reciprocal of flow divergence) based landing strategies have been performed on rotorcraft, such as the TauPilot [52], which implemented the guidance and control scheme as proposed by Tau theory [82, 83].

However, there is a fundamental problem when actually controlling a constant flow divergence landing, i.e., the controller gain(s) depends on the height. No true solution has been presented for this problem. A few studies, focusing on decreasing time-to-contact landings, schedule the gains according to the time-to-contact [51, 52]. However, the initial gain depends on the height and velocity. Deviating significantly from these assumed initial conditions leads to severely hampered performance or even a crash. In addition, such gain-scheduling is not possible for constant flow divergence (or constant time-to-contact) landings.

A recent theory developed by one of the authors shows the relationship between the height and the controller gain [50]. It was shown analytically that for a specific fixed gain, optical flow control becomes unstable at a specific given height. Instead of regarding this as a problem, it was proposed that the MAV can detect oncoming self-induced oscillations and use these for estimating the height. One of the uses of this theory is to detect self-induced oscillations close to the landing surface for landing triggering.

The main contribution of this paper is that we leverage this theory to propose a strategy that solves the fundamental problem of gain selection for optical flow landing. This strategy allows for a smooth and high performance landing of the MAV, by adjusting the controller gains during landing. Two other, smaller, contributions of this paper are: 1) to develop a novel way to detect oscillations in real-time based on observation of the flow divergence, and 2) to characterize the flow divergence measured from a single camera mounted on a quadrotor MAV (this part of the work is partly based on a conference paper [76]).

The remainder of the article is structured as follows. In Section 3.2 we provide some background on the constant flow divergence guidance strategy, and show results of computer simulations with a conventional control scheme assuming a perfect flow divergence estimate. Section 3.3 presents a characterization of the flow divergence estimates as obtained with a monocular camera. Section 3.4 then shows the analysis of the conventional closed-loop control with the delay and noisy estimates in both computer simulation and flight test. In Section 3.5 the adaptive control scheme is introduced to deal with the instability problems encountered with the conventional control scheme. Section 3.6 demonstrates the real-world experiments. Conclusions and recommendations are given in Section 3.7.

3.2. BACKGROUND

3.2.1. FLOW DIVERGENCE

The definition of axes used in this paper is illustrated in Fig. 3.1. In this figure, the body reference frame is denoted as $O^bX^bY^bZ^b$, where O^b is located at the center of gravity of an MAV, X^b points forward, Y^b is starboard, and Z^b points downward. World reference frame is a fixed frame on the ground and uses North-East-Up (X^w-Y^w-Z^w) system.

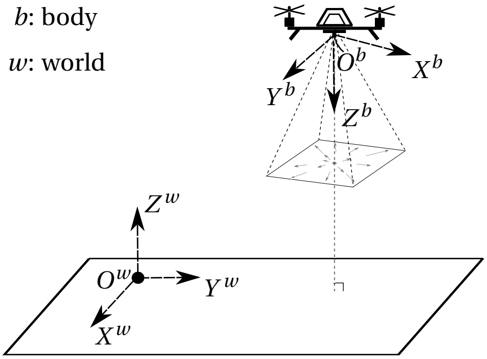


Figure 3.1: MAV body ($O^bX^bY^bZ^b$) and world ($O^wX^wY^wZ^w$) reference frames.

Since a camera is attached to the body of the MAV, the camera reference frame ($O^cX^cY^cZ^c$) can be assumed to be aligned with the body reference frame, where the camera is facing downward or in the positive Z^c direction. When the MAV is approaching a flat ground surface, the camera observes a divergent pattern of optical flow, the so-called flow divergence shown in Fig. 3.2. Flow divergence of a feature point in an im-

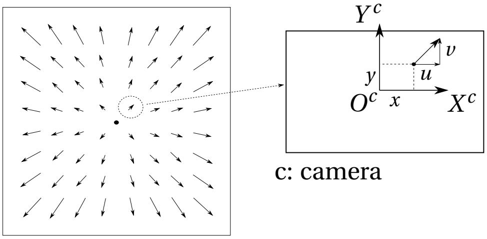


Figure 3.2: Divergence of optical flow (flow divergence) when an observer is approaching a surface.

age is defined as the partial derivatives of its velocities (u and v) at its position (x and y) in the camera image coordinates system [61]:

$$D(x,y) = \frac{\partial u(x,y)}{\partial x} + \frac{\partial v(x,y)}{\partial y}. \tag{3.1}$$

This is illustrated on the right side of Fig. 3.2 in which one of optical flow vectors is enlarged and shown in an image. By examining all $D(x,y)$ of the available features in an image, a 'global' flow divergence, D which is of particular interest in this paper can be obtained. For vertical landings, flow divergence can be defined as the ratio of the vertical velocity, V_Z to its height from the ground, Z:

$$D = \frac{V_Z}{Z}. \tag{3.2}$$

Flow divergence can be used to determine the time-to-contact, τ which is reciprocal to D. For vertical landing of an MAV, $Z > 0$, $V_Z < 0$, and thus $D < 0$.

3.2.2. CONSTANT FLOW DIVERGENCE GUIDANCE STRATEGY

The common guidance strategy using flow divergence for vertical landing is the constant flow divergence approach [53]. By keeping the flow divergence constant, $D = -c$, we can control the dynamics of the landing with a suitable c. To examine the influence of c on this strategy during a landing maneuver, the equations of motion describing the height Z, vertical velocity V_Z, and vertical acceleration A_Z of the MAV are:

$$Z = Z_0 e^{-ct},\ V_Z = -cZ_0 e^{-ct},\ A_Z = c^2 Z_0 e^{-ct}, \tag{3.3}$$

where Z_0 is the initial height above the landing surface.

Fig. 3.3 shows the effect of c on the height, velocity, and acceleration time histories with the same initial height. It is clear that only flow divergence, $c > 0$ will lead to convergence of the states to zero. With different positive values of c, we can manipulate the dynamics of the maneuver. For example, the larger the c is, the faster the states converge to zero. The practical feasibility of these maneuvers, however, also depends on the vehicle limitations, such as the maximum vertical velocity that can be achieved.

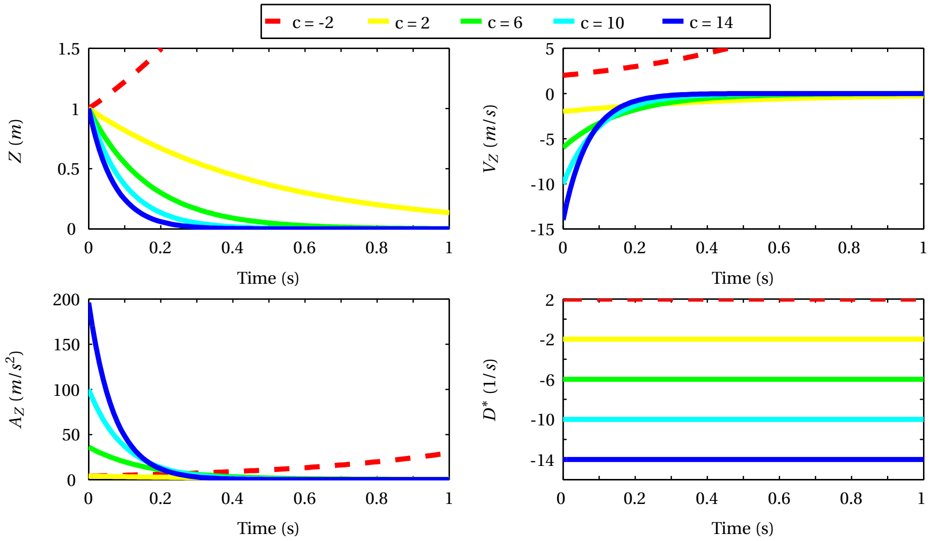


Figure 3.3: Constant flow divergence guidance with different desired flow divergence $D^* = -c$.

3.2.3. CONVENTIONAL CONTROL SCHEME

To track the desired flow divergence D^*, a relatively straightforward proportional feedback controller can be used:

$$\mu = K_p(D^* - D), \tag{3.4}$$

where K_p is the gain of the proportional controller.

To simplify the analysis, we model the dynamics of an object, moving towards a surface in one dimensional space, using a double integrator. The state space model can then be written as:

$$\dot{\mathbf{r}}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{r}(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mu(t), \tag{3.5}$$

$$y(t) = [r_2(t)/r_1(t)] = D, \tag{3.6}$$

where $\mathbf{r} = [r_1, r_2]^T = [Z, V_Z]^T$ and μ is the control input.

Eqs. (3.5) and (3.6) show that the model dynamics are linear but its observation is nonlinear. To visualize the feasibility of the proportional controller to track a constant reference (e.g., $D^* = -0.3$), a time response of the system is plotted in Fig. 3.4. In this figure, both height and velocity are approaching zero in the end. During this maneuver, the vehicle accelerated in the first $2s$ and then decelerated to zero velocity to touch the ground.

3.3. CHARACTERIZATION OF FLOW DIVERGENCE

In the previous section we showed that, with a simple proportional controller and 'perfect' estimate of flow divergence, the vehicle can be landed smoothly with zero velocity touching the ground. However, in a real scenario, we cannot avoid having delays and

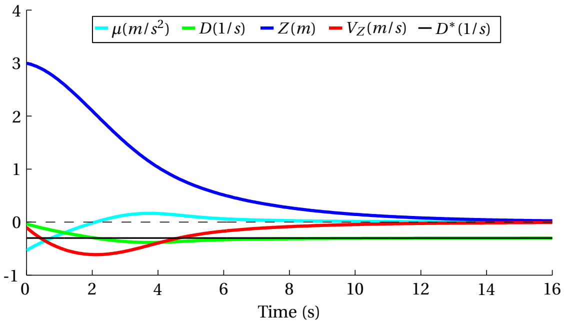


Figure 3.4: Fixed-gain closed-loop landing control using a constant flow divergence strategy.

noise in the sensor measurements. For this reason, we will need to characterize the inaccuracies induced in estimating the flow divergence, to investigate the effects of these sensor inaccuracies on the feasibility of using a simple controller in practice. In this paper, we use two different methods to estimate the flow divergence. One is based on flow field fit, the other uses a more direct method to compute the expansion and compression of the optical flow vectors.

3.3.1. FLOW FIELD DIVERGENCE ESTIMATOR

We first need to estimate the 'raw' flow divergence from a camera sensor. In this study, optical flow vectors are computed based on a sparse corner tracking method using Features from Accelerated Segment Test (FAST) [69, 70], integrated with a Lucas-Kanade tracker [71]. The first vision algorithm that estimates the flow divergence of the optic flow field [84] is based on early findings by Longuet-Higgins and Prazdny [85]. The algorithm assumes that (a) a pinhole camera model pointing downward is used, (b) the surface in sight is planar, and (c) the angular rates of the camera can be measured and used to de-rotate the optical flow. Under these assumptions, the optical flow equation can be expressed as follows:

$$\mathbf{u} = \overbrace{-\vartheta_x}^{p_{u1}} + \overbrace{(\vartheta_x a + \vartheta_z)}^{p_{u2}}\mathbf{x} + \overbrace{\vartheta_x b}^{p_{u3}}\mathbf{y} - \overbrace{a\vartheta_z}^{p_{u4}}\mathbf{x}^2 - \overbrace{b\vartheta_z}^{p_{u5}}\mathbf{xy}, \tag{3.7}$$

$$\mathbf{v} = \overbrace{-\vartheta_y}^{p_{v1}} + \overbrace{\vartheta_y a}^{p_{v2}}\mathbf{x} + \overbrace{(\vartheta_y b + \vartheta_z)}^{p_{v3}}\mathbf{y} - \overbrace{b\vartheta_z}^{p_{v4}}\mathbf{y}^2 - \overbrace{a\vartheta_z}^{p_{v5}}\mathbf{xy}, \tag{3.8}$$

where $\vartheta_x = V_X/Z$, $\vartheta_y = V_Y/Z$, and $\vartheta_z = V_Z/Z$ are the velocities in x^b, y^b, and z^b direction scaled with respect to the height Z. a and b are the gradient of the ground surface.

By re-writing Eqs. (3.7) and (3.8) into matrix form, as shown in Eq. (3.9), the parameter vectors $\mathbf{p_u} = [p_{u1}, p_{u2}, p_{u3}, p_{u4}, p_{u5}]$ and $\mathbf{p_v} = [p_{v1}, p_{v2}, p_{v3}, p_{v4}, p_{v5}]$ can be estimated using a maximum likelihood linear least squares estimate within a robust random sample consensus (RANSAC) estimation procedure [86]:

$$u = \mathbf{p_u}[1, x, y, x^2, xy]^T, \ v = \mathbf{p_v}[1, x, y, y^2, xy]^T. \tag{3.9}$$

The estimated parameters provide important information for bio-inspired navigation, such as ventral flow, surface slope [84], flow divergence, time-to-contact, etc. In this study, we are primarily interested in estimating the flow divergence:

$$\widehat{D} = p_{u2} + p_{v3}. \tag{3.10}$$

Note that Eqs. (3.7) and (3.8) can be simplified by neglecting the second-order terms, as in this study we focus on landing upon flat surfaces without any inclination ($a = 0$ and $b = 0$). Therefore, a linear fit of the optical flow field can be obtained.

3.3.2. SIZE DIVERGENCE ESTIMATOR

We propose a more straightforward way to estimate flow divergence by measuring the size of the lines connecting between features in consecutive image frames. The left side of Fig. 3.5 is a pinhole camera model showing the actual and image size of the lines connecting two features indicated by L and d, respectively. On the right side of this figure, the geometry illustrates the change of the size of the projected lines in the image plane, from $d_{t-\Delta t}$ to d_t when the MAV is moving towards the ground, from $Z_{t-\Delta t}$ to Z_t. Using similar triangles, we can write the following relationships:

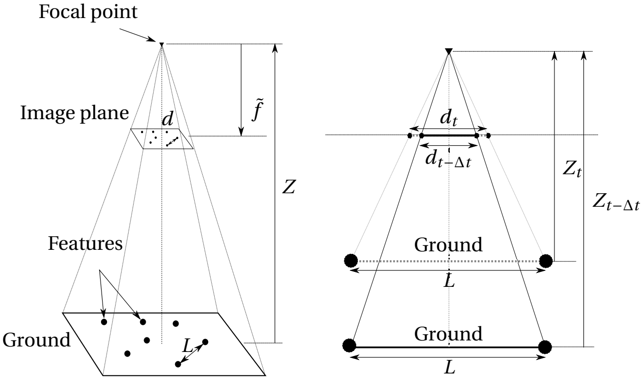


Figure 3.5: Pinhole camera model (left) and projected lines on image plane when approaching ground (right).

$$\frac{L}{Z_{t-\Delta t}} = \frac{d_{t-\Delta t}}{\tilde{f}}, \; \frac{L}{Z_t} = \frac{d_t}{\tilde{f}}, \tag{3.11}$$

where $\tilde{f}$ is the focal length of the camera while Δt is the timestamp between two consecutive images. By substituting L, we can obtain:

$$\frac{d_t}{Z_{t-\Delta t}} = \frac{d_{t-\Delta t}}{Z_t}. \tag{3.12}$$

From the geometry in Fig. 3.5 and Eq. (3.12), it is reasonable that when the MAV moves closer to the ground, i.e., $Z_t < Z_{t-\Delta t}$, the size of the line in the image plane becomes larger, i.e., $d_t > d_{t-\Delta t}$. By recalling Eq. (3.2), flow divergence can also be expressed as follows:

$$D_t = \frac{1}{\Delta t}[1 - \frac{Z_{t-\Delta t}}{Z_t}]. \tag{3.13}$$

By substituting Eq. (3.12) into Eq. (3.13), we can obtain the size divergence of one feature line:

$$D_{s_t} = \frac{1}{\Delta t}[\frac{d_{t-\Delta t} - d_t}{d_{t-\Delta t}}]. \tag{3.14}$$

To obtain a more reliable estimate of size divergence, we can use the average of all detected feature lines, $\widehat{D}_s$ in our computation:

$$\widehat{D}_s = \frac{1}{n}\sum_{i=1}^{n} D_{s_{t_i}}. \tag{3.15}$$

When the MAV moves towards the ground surface, the lines connecting features extend leading to $\widehat{D}_s < 0$, and vice versa.

3.3.3. TESTING PLATFORM AND DATA LOGGING

A Parrot AR.Drone 2.0 is used as our platform for performing flight tests. The downward-facing camera on this MAV is of particular interest here. We implemented aforementioned algorithms to estimate the flow divergence in Paparazzi Autopilot, an open-source autopilot software[1]. Fig. 3.6 shows the overview of the control architecture of Paparazzi and the integration of the computer vision module. All the computer vision and control algorithms are run on-board the MAV.

In Fig. 3.6, images are captured from the downward-looking camera in the vision module. These images are processed using the computer vision algorithms presented in the subsections above. The angular rates (p, q, r) from the IMU are used in the optical flow computations to reduce the effects of MAV rotation on the optical flow vectors. One of the flow divergence estimates, $\widehat{D}$ or $\widehat{D}_s$, is used in the vertical guidance loop to perform automatic landings.

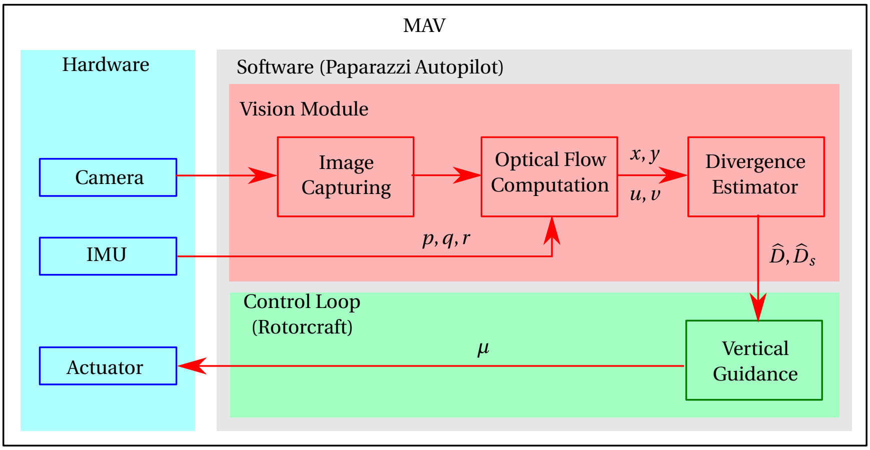


Figure 3.6: Control architecture of Paparazzi Autopilot and the integration of the computer vision module.

---

[1] Paparazzi Autopilot: http://wiki.paparazziuav.org/wiki/Main_Page

In our experiments, we start logging the data while the MAV is hovering at a height around 1.5m. By only controlling the climb rate, we measure the variation of estimated flow divergences $\widehat{D}$ and $\widehat{D}_s$, the height above the ground Z, and the vertical velocity V_Z. Note that in order to guarantee good measurements of Z and V_Z, which will serve as our ground truth for flow divergence D, we use an external motion tracking system (OpticTrack), to provide these measurements.

Fig. 3.7 shows the measurements log for estimating the delay model and noise model of the flow divergence estimates. We deliberately varied the vehicle climb rate to obtain a wide range of $\widehat{D}$, $\widehat{D}_s$, Z and V_Z measurements.

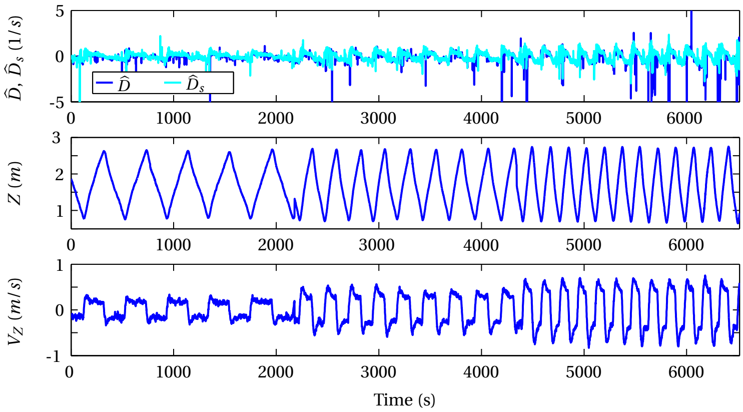


Figure 3.7: Log of flow divergence estimates, height, and vertical velocity during a vertical maneuver of an AR.Drone 2.0.

3.3.4. DELAY ESTIMATION AND NOISE MODEL

After obtaining flow divergence estimates, this subsection describes how to characterize their properties. There are two steps proposed: 1) to estimate the delays in the estimates, and 2) to model the noise using the delay-corrected data.

DELAY ESTIMATION

We estimated the delay Lag_i of every sample $i = 1, 2, ..., N$ by comparing the datasets of flow divergence estimates $\widehat{D}$ and $\widehat{D}_s$ with the ground truth D. First, W windowed samples, $f(i, i+W)$ and $g(i+m, i+W+m)$ of ground truth and flow divergence estimates, respectively, were selected, where m is the moving index. Second, the sums of square error between both samples sets were computed. Then, by repeating the aforementioned steps with different sample sets of flow divergence estimates (from $m = 0$ to $m = M$), we looked for the sample set with minimum error. The number of estimates lagging behind the ground truth can then be estimated as:

$$Lag_i = \arg\min_{m} \sum_{j=0}^{W} (f(i,j) - g(i,j,m))^2. \tag{3.16}$$

Based on observation of the estimates compared with the ground truth, the delays are expected to be consistent. Multiplying the average of Lag with the sampling time Δt, we can obtain the time delay of each data set. The estimates have, on average, a lag of 2 and 1, which are equivalent to $0.1s$ and $0.05s$ for $\widehat{D}$ and $\widehat{D}_s$, respectively ($\Delta t \approx 0.05s$). These lags are used to correct the flow divergence estimates. Fig. 3.8 illustrates that the corrected flow divergence $\widehat{D}_{cr}$, and the corrected size divergence $\widehat{D}_{s_{cr}}$, indeed better match the ground truth D. In this figure, the bottom left plots show the enlarged view of flow divergence estimates with the ground truth from $2200s$ to $2400s$ while their corresponding plots which are corrected for the delay are presented on the bottom right.

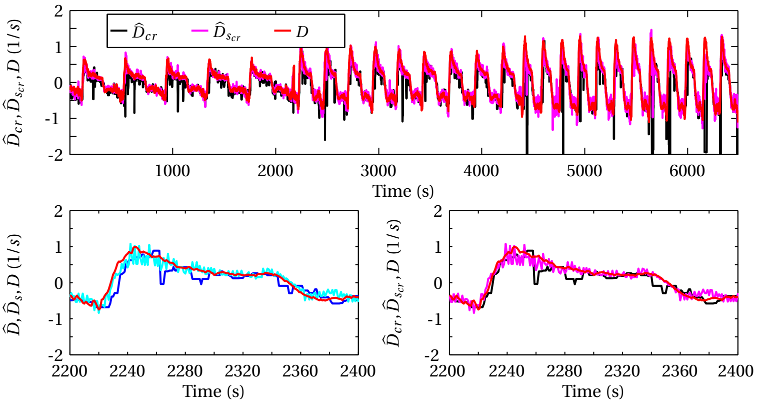


Figure 3.8: The estimated and delay-corrected flow divergences, together with their ground truth.

Note that to avoid the estimate of Lag to be driven by noise and outliers, the flow divergence estimates were pre-filtered using a moving median filter. The delay caused by this filter (i.e., $(N_{win}-1)/2$, where N_{win} is its window size) was subtracted from the estimated Lag to obtain the actual lag.

NOISE MODEL

After correcting for the delay, we can proceed to model the noise of the flow divergence estimates. Fig. 3.9 plots the flow divergence estimates $\widehat{D}$ and $\widehat{D}_s$ against the corresponding ground truth D. This figure illustrates the deviation of the estimated flow divergence from its ideal condition. There are two groups of estimated flow divergence plotted in the figure, i.e., the flow divergence without (circles) and with delay correction (asterisks), respectively. The root mean square errors (RMSEs) of $\widehat{D}_{cr}$ and $\widehat{D}_{s_{cr}}$ with delay correction (≈ 0.6059 $1/s$ and ≈ 0.1469 $1/s$) are smaller than their corresponding RMSEs without delay correction (≈ 0.6206 $1/s$ and ≈ 0.1526 $1/s$). Overall, both are noisy signals and slightly deviate from their ideal condition, especially when the flow divergence becomes more negative and more positive. This could happen when the ground features can hardly be tracked due to, e.g., an aggressive maneuver, or when the vehicle is either very close to the ground or far away from the ground.

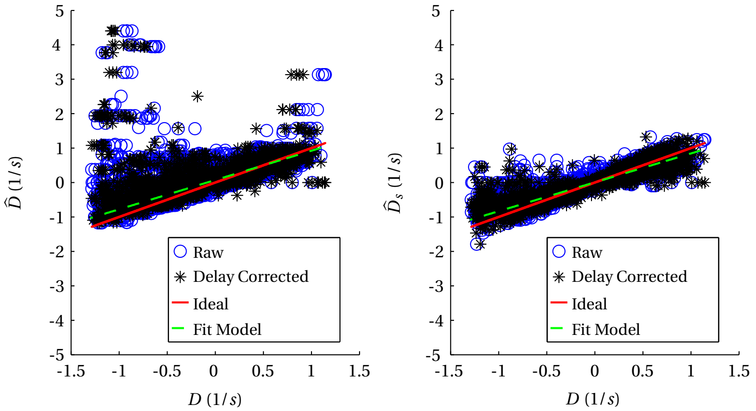


Figure 3.9: Deviation of the estimated flow divergences $\widehat{D}$ (left) and $\widehat{D}_s$ (right) from their ground truth D.

To take into account this condition in the noise model, we fitted a linear function, as shown in Eq. (3.17) to the delay-corrected estimates using a bisquare weights regression. This method is preferable as it minimizes the influence of outliers on the fit by giving less weight to the data far away from the fitted line. The fitted models are drawn in as a *dashed line* in Fig. 3.9:

$$f_1(D) = e_1 \cdot D + e_2, \tag{3.17}$$

where e_1 and e_2 are the fit coefficients. The fitted models for $\widehat{D}$ and $\widehat{D}_s$ are given as $\widehat{D}_{fit}$ and $\widehat{D}_{s_{fit}}$.

The next step is to estimate the variances of the estimates with respect to the fitted models. Fig. 3.10 shows the absolute errors of the estimates versus ground truth flow divergence. The *circles* represent the absolute errors of the estimate with respect to the fitted model. Since we can observe (also from Fig. 3.9) that the errors are higher for larger values of D, a quadratic fit function is more suitable:

$$f_2(D) = e_3 \cdot D^2 + e_4 \cdot D + e_5, \tag{3.18}$$

where e_3, e_4, and e_5 are the fit coefficients. The *solid line* in Fig. 3.10 is the fitted line of these absolute errors which represents the variance. Table 3.1 lists the fit coefficients of the noise models (e_1, e_2, e_3, e_4, and e_5) for both $\widehat{D}$ and $\widehat{D}_s$ that are used in the computer simulations and the controller design.

Fig. 3.11 presents the probability density functions of the models errors, $err_{\widehat{D}}$ and $err_{\widehat{D}_s}$. These model errors are computed by subtracting the data generated based on Eqs. (3.17) and (3.18) from the corresponding flow divergence estimates. In this figure, we fitted a Gaussian model (*solid lines*) to each distribution, we obtain $err_{\widehat{D}} = N(0.0173, 0.1292)$ and $err_{\widehat{D}_s} = N(6.1979 \times 10^{-4}, 0.0937)$. This shows that both estimated noise models are quite accurate.

To validate the noise model, we plotted generated data with estimates of $\widehat{D}$ and $\widehat{D}_s$ as shown in Fig. 3.12. In this figure, the generated data sets show that the errors are higher for larger values of D, which are similar to the observation in Fig. 3.10. However, some inevitable outliers in the estimates remain, as there are neglected in both noise models.

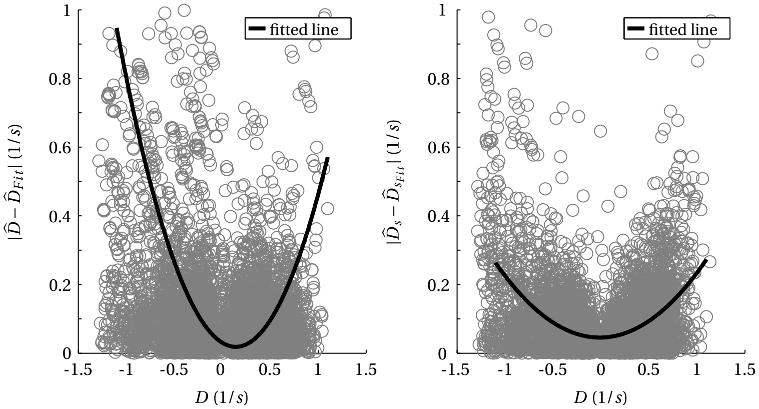


Figure 3.10: Absolute errors of the flow divergence estimates $\widehat{D}$ (left) and $\widehat{D}_s$ (right) versus their ground truth D.

Table 3.1: Fit coefficients of noise models for $\widehat{D}$ and $\widehat{D}_s$.

Coefficients	$\widehat{D}$	$\widehat{D}_s$
e_1	0.8519	0.8393
e_2	-0.0655	-0.0060
e_3	0.5766	0.1841
e_4	0.1918	-0.0043
e_5	0.0412	0.0455

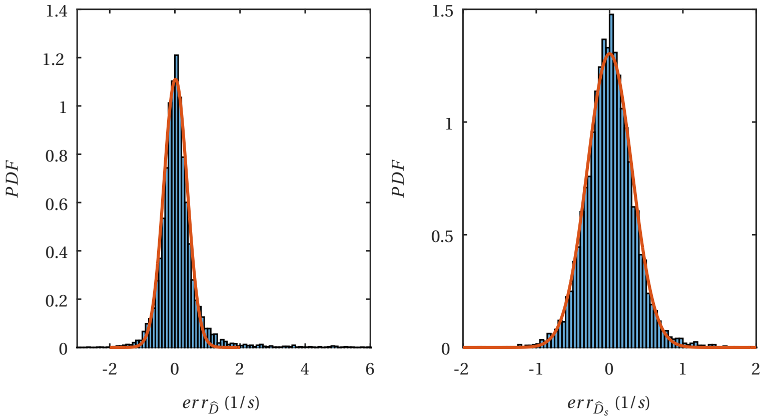


Figure 3.11: Probability density functions of the estimate errors of $\widehat{D}$ (left) and $\widehat{D}_s$ (right) with respect to the fitted model.

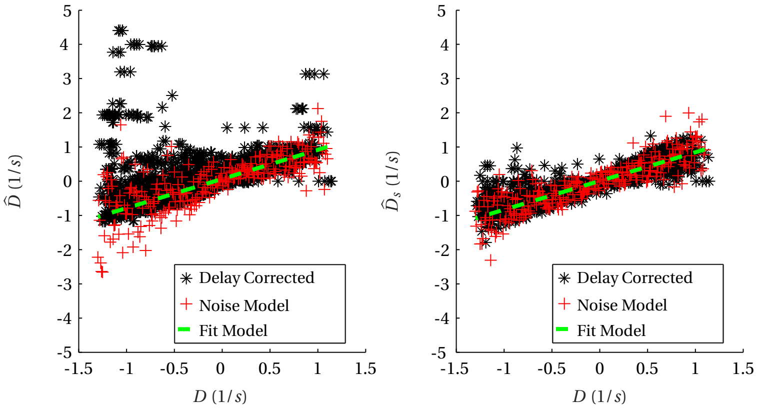


Figure 3.12: The flow divergence estimates $\widehat{D}$ (left) and $\widehat{D}_s$ (right) and the data generated using the noise models.

Note that since $\widehat{D}_s$ has less delay and noise, we use it for the following simulations and flight tests. Additionally, the term 'flow divergence' and the symbol $\widehat{D}$ will be used, instead of size divergence and its symbol $\widehat{D}_s$, in order to have a more general expression and to maintain consistency.

3.4. INFLUENCE OF DELAY AND NOISE ON FIXED-GAIN CONSTANT FLOW DIVERGENCE LANDING

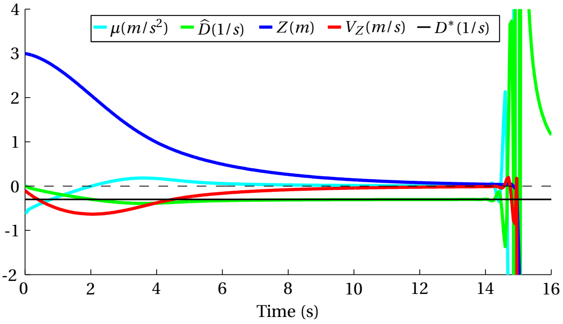
In the previous section, we estimated the delay and noise in the vision measurement obtained from an on-board camera. This section investigates the effects of the delay and noise on the constant flow divergence landing using the conventional control scheme as described in Section 3.2. This is performed in both computer simulations and real-world experiments.

3.4.1. SIMULATION RESULTS

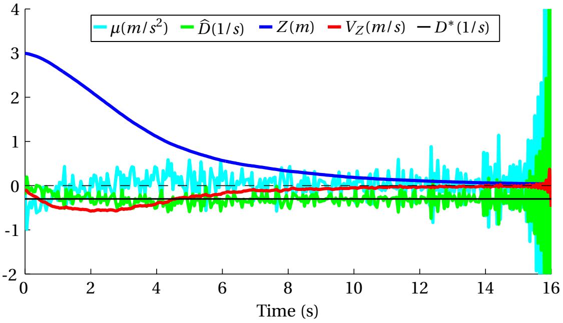
The same model and controller described in Section 3.2 are used in this simulation. For a fair comparison, the settings including D^*, K_p, and the initial conditions of Z and V_Z are set to be the same. The control analysis of the system is performed in sequence by adding: (1) a delay, (2) a noise model, and (3) both delay and noise model into the observation model in Eq. (3.6). Their results are presented in Fig. 3.13.

ADDING DELAY

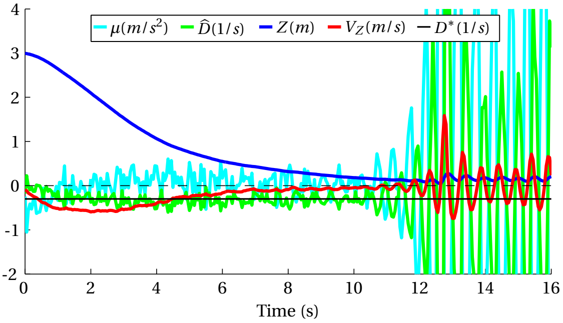
In Section 3.3, we estimated a delay of 2 samples, i.e., a time delay of $0.1s$, in the flow divergence measured from the vision system. This delay is added to the observation model in the simulation. Fig. 3.13a plots the time response of the states using the flow divergence based control. The result shows that with this delay both Z and V_Z converge to almost zero quicker than the response of the perfect observation, but the MAV becomes unstable when it is very close to the ground.



(a) Adding delay.



(b) Adding noise model.



(c) Adding delay and noise model.

Figure 3.13: Adding delay and noise to constant flow divergence landing using fixed-gain controller leads to self-induced instability at a low height.

ADDING NOISE

Next, only the noise model is added to the observation model. Fig. 3.13b shows the effects of noise to the time response of the states. Similar to adding delay, the MAV becomes unstable when both height and velocity are very small. In practice, for both cases the oscillations can be avoided by throttling down or completely switching off the engine when the MAV is very close to the landing surface; also, in a real-world scenario a sufficiently low gain can be selected so that the MAV's landing gear touches the ground before oscillations occur.

ADDING DELAY AND NOISE

In reality, we have both delay and noise in the estimate of flow divergence from the vision system. Therefore, we also examine the effects of both delay and noise on the control scheme performance. Fig. 3.13c shows that large oscillations occur sooner and are amplified further when the MAV moves close to the landing surface.

3.4.2. FLIGHT TEST RESULTS

In this subsection, we present the flight test results for vertical landing controls using the conventional control scheme (e.g., a fixed gain). For the experiments, a proportional and integral (PI) controller is used to reject external disturbances and thus minimize the steady-state errors:

$$\mu = K_p \left[(D^* - D) + \frac{1}{\kappa} \int (D^* - D)\, dt \right] \tag{3.19}$$

$\kappa = \frac{K_p}{K_i}$ is the integral time constant and K_i is the integral gain.

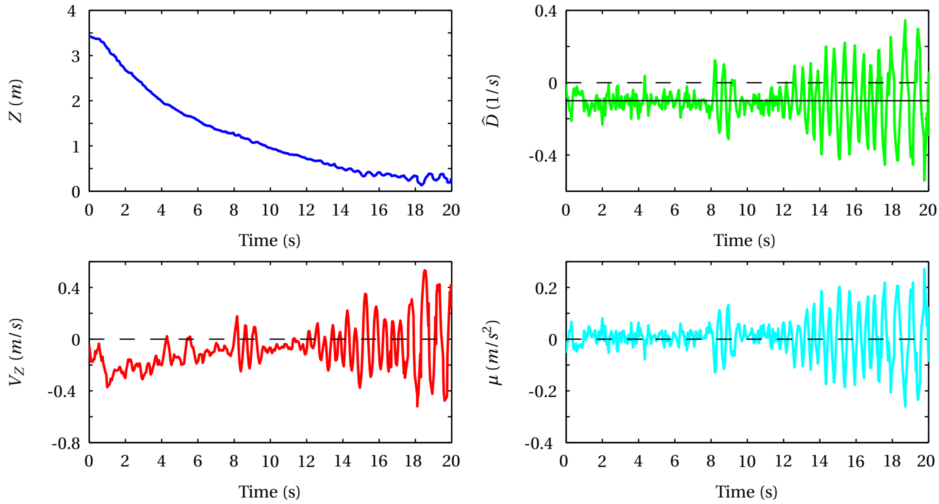


Figure 3.14: Constant flow divergence landing of the MAV using a fixed-gain controller.

The same MAV platform as described in Section 3.3.3 is used for the experiments with all vision and control algorithms running on-board. Fig. 3.14 shows the experiment results of constant flow divergence landing using the basic, fixed-gain, controller (e.g., $K_p = 0.6, K_i = 0.1$). Clearly, large oscillations occur for flow divergence, height, and velocity due to the delay and noise of the flow divergence estimate, even though the height and velocity exponentially decay to zero. We used a slightly smaller desired flow divergence than the one used in the simulations, i.e., $D^* = 0.1s^{-1}$ in order to obtain a better view of the oscillations before touching the ground surface.

3.5. ADAPTIVE GAIN STRATEGY FOR CONSTANT DIVERGENCE LANDING

In the previous section, we observed that the presence of time delay and measurement noise leads to instability of constant flow divergence landings when the basic, fixed-gain, controller is used. The oscillations can be further amplified when the vehicle is getting closer to the ground or can even happen at an earlier stage of the landing when a large gain is used. This section introduces a novel way to reject oscillations in constant flow divergence landings.

3.5.1. ADAPTIVE CONTROL STRATEGY

Fig. 3.15 shows the proposed adaptive control strategy for constant divergence landings. There are two phases in this strategy: (I) Determination of near-optimal initial controller gains, and (II) Landing with an adaptive gain.

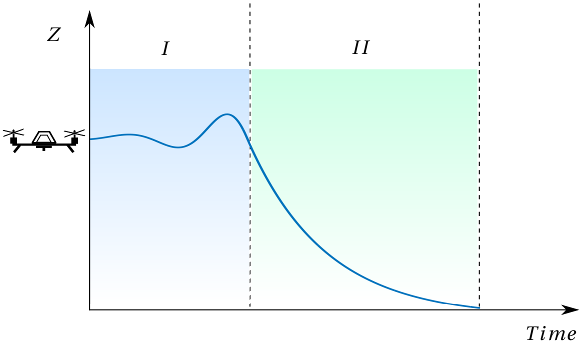


Figure 3.15: Two-phase adaptive control strategy for constant flow divergence landing. In phase I, the MAV increases the gain until it starts to oscillate. This allows the MAV to select a suitable gain for phase II, in which it lands while reducing the gain exponentially.

DETERMINATION OF NEAR-OPTIMAL INITIAL GAINS

The initial controller gains K_0 are the important parameters that we need to determine for the PI controller before starting the landing. If these initial values are too small, tracking of D^* will not be accurate. In contrast, if they are too large, self-induced oscillations can happen at the beginning of the landing. To deal with both cases, the MAV hovers by tracking $D^* = 0$ using a PI controller while small initial values of K_0, which are gradually increased until an oscillation is detected. Then, the stable gain just *before* the oscillation is used as the initial K_0 for phase II. Both gains from P and I controllers undergo this process.

LANDING

Once the stable initial gains are obtained, the constant flow divergence landing is activated by tracking $D^* = -c$. We know that the value of the gain at which instability starts to occur depends linearly on the height [50], and that during a constant divergence landing the height decays exponentially. Therefore, we have the gains decay exponentially to mitigate and, if possible, eliminate the oscillations when moving close to the ground. In this strategy, phase I ensures that proper initial gains are chosen to have a good performance of the tracking, while phase II prevents self-induced oscillations when descending. In the following subsections, the stability analysis of the adaptive controller and the real-time oscillation detection method used in this strategy are described in details.

3.5.2. STABILITY ANALYSIS OF THE ADAPTIVE CONTROLLER

In this subsection, we show that the linearized system is not subject to self-induced oscillations when the adaptive controller is used for constant flow divergence landings. In fact, we know from Eq. (3.3) that $Z = Z_0 e^{D^* t}$ when $D = D^*$. As mentioned, to cope with the instability problem, we introduce:

$$K_p = K_{p_0} e^{D^* t}, \quad K_i = K_{i_0} e^{D^* t} \tag{3.20}$$

where K_{p_0} and K_{i_0} are the initial gains of the PI controller which relates to the initial height ($K_0 = f(Z_0)$) and can be obtained using the method presented in Subsection 3.5.1. By recalling Eqs. (3.5) and (3.6), $\dot{\mathbf{r}} = [r_2, \mu]^T$ and $y(t) = [r_2/r_1]$, where $\mathbf{r} = [r_1, r_2]^T = [Z, V_Z]^T$.

To understand the dynamical behavior of this system, we first analyze the phase portrait of the system. Fig. 3.16 shows the system's trajectories with arrows and three cases with different initial states. All states of these cases converge to zero in the end. Most importantly, we can observe that positive V_Z, which could happen due to external disturbances, will eventually become negative (i.e., the MAV descends). This can also be seen from Eq. (3.19) that when V_Z becomes positive, the controller will further reduce the thrust and lead to $V_Z < 0$.

Here, we study the stability of the discrete system. We will see that even introducing just a zero-order hold form in which it has a discrete sample time and thus a small delay in the system already suffices to get self-induced instability. By linearizing and discretizing the system model, we obtain:

$$\Phi = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} -\frac{V_Z}{Z^2} & \frac{1}{Z} \end{bmatrix}, \quad \mathbf{D} = [0]. \tag{3.21}$$

The open-loop transfer function of the discrete system can be determined to be:

$$\begin{aligned} G(\sigma) &= C(\sigma I - \Phi)^{-1}\Gamma \\ &= \frac{\Delta t\left[(2Z - \Delta t V_Z)\sigma - (2Z + \Delta t V_Z)\right]}{2Z^2(\sigma - 1)^2}, \end{aligned} \tag{3.22}$$

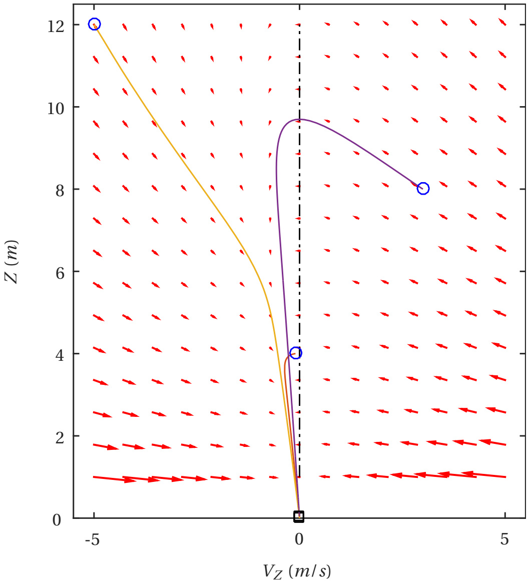


Figure 3.16: Phase portrait of the constant flow divergence landing.

where σ is the discrete frequency domain operator (variable z typically used for this term could be confused with the height Z, thus σ is used to avoid confusion).
The discrete form of the PI controller can be written as:

$$F_{PI}(\sigma) = K_p\left[1 + \frac{\Delta t}{2\kappa}\left(\frac{\sigma+1}{\sigma-1}\right)\right]. \tag{3.23}$$

The closed-loop transfer function of the discrete system is:

$$\begin{aligned} H(\sigma) &= \frac{G(\sigma)\cdot F_{PI}(\sigma)}{1+G(\sigma)\cdot F_{PI}(\sigma)} \\ &= \frac{\mathcal{N}(\sigma)}{\mathcal{D}(\sigma)}, \end{aligned} \tag{3.24}$$

where

$$\begin{aligned} \mathcal{N}(\sigma) =& K_p\Delta t\left[(\Delta t - 2\kappa) + (\Delta t + 2\kappa)\sigma\right]\left[(\Delta t V_Z + 2Z)\right. \\ &+ \left.(\Delta t V_Z - 2Z)\sigma\right], \\ \mathcal{D}(\sigma) =& K_p\Delta t\left[V_Z\Delta t^2(\sigma+1)^2 + 2\Delta t(\kappa V_Z - Z)(\sigma^2-1)\right. \\ &- \left.4Z\kappa(\sigma-1)^2\right] - 4\kappa Z^2(\sigma-1)^3. \end{aligned}$$

The zeros of the system can be obtained to be:

$$\sigma_{0_1} = \frac{2Z + \Delta t V_Z}{2Z - \Delta t V_Z} \quad \text{or} \quad \sigma_{0_2} = \frac{2\kappa - \Delta t}{2\kappa + \Delta t}. \tag{3.25}$$

We know that due to a relatively small and positive Δt, $Z > 0$ and $V_Z < 0$, thus $0 < \sigma_{0_1} < 1$. In contrast, σ_{0_2} depends on κ. For a stable discrete system, all poles should lie inside a unit circle in σ-plane. From Eq. (3.24), we know that all poles are located at $\sigma = 1$ when $K_p = 0$. As the gain increases, two poles move toward the two finite zeros presented in Eq. (3.25) and the third pole moves toward the negative infinite zero. From this observation, there are two factors which can affect the stability of the system: (1) the gain at $\sigma = -1$, the so-called critical P-gain, K_{cr}, and (2) the influence of κ on σ_{0_2}. For the reader to understand the first case, we plot a root locus of the closed-loop discrete system for $Z = 100m$ and $Z = 10m$ with $\Delta t = 0.03s$ in Fig. 3.17. In this figure, both results of the different heights lead to the same root locus plot, but with different values of the gain (see Eq. (3.26)). Let $\sigma = -1$ in Eq. (3.24), we obtain:

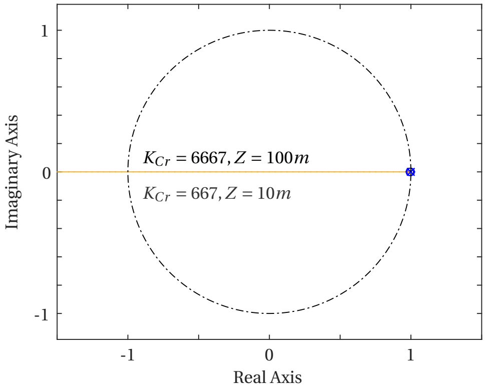


Figure 3.17: Root locus of the closed-loop discrete system for two different heights.

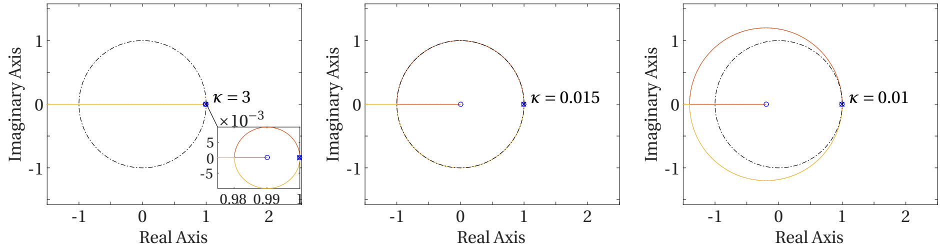
$$K_{cr} = \frac{2Z}{\Delta t}. \tag{3.26}$$

This relation is exactly the same as the one found in [50] for a pure P-controller. It shows that the critical gain depends on the sample time Δt, and - importantly - on the height Z. From this result, the controller gain should be $0 < K_p < K_{cr}$ in order to have a stable system. If K_0 is set to be $< 2Z/\Delta t$ and K_p scales with the same exponent as Z, it will stay below that threshold for the rest of the trajectory.

For the second case, we investigate the influence of κ on σ_{0_2}, and thus the stability of the system. Fig. 3.18 shows three root loci of the closed-loop discrete system for three different values of κ. As mentioned above, one pole moves to σ_{0_2}, and another pole moves to negative infinite zero. We will prove that a part of this root locus is a circle with center located at σ_{0_2}. Since we know that these two poles are 1 and the finite zero is σ_{0_2}, we can write the system characteristic equation for this specific case to be approximately [87]:

$$1 + \frac{K_p\left(\sigma - \sigma_{0_2}\right)}{(\sigma - 1)^2} = 0 \tag{3.27}$$

By substituting $\sigma = \varrho + j\chi$ where ϱ and χ are the real and imaginary part of σ into



(a) $\kappa > \frac{\Delta t}{2}$.

(b) $\kappa = \frac{\Delta t}{2}$.

(c) $\kappa < \frac{\Delta t}{2}$.

Figure 3.18: Root locus of the closed-loop discrete system for three different integral time constants.

Eq. (3.27), we obtain:

$$\frac{K_p\left(\varrho - \sigma_{0_2} + j\chi\right)}{\left(\varrho - 1 + j\chi\right)^2} = -1 \tag{3.28}$$

We know that for every point on root locus, the angle condition must be satisfied:

$$\tan^{-1}\left(\frac{\chi}{\varrho - \sigma_{0_2}}\right) - \tan^{-1}\left(\frac{\chi}{\varrho - 1}\right) - \tan^{-1}\left(\frac{\chi}{\varrho - 1}\right) = 180^o \tag{3.29}$$

By solving Eq. (3.29), we get the following:

$$\left(\varrho - \sigma_{0_2}\right)^2 + \chi^2 = \left(1 - \sigma_{0_2}\right)^2 \tag{3.30}$$

Eq. (3.30) shows that this part of the root locus is a circle centered at $(\sigma_{0_2}, 0)$ with radius $1 - \sigma_{0_2}$ in the σ-plane. Since this circle is tangent to the unit circle at 1, it will coincide with the unit circle if $\sigma_{0,2} = 0$. From this result, the system is stable if $\sigma_{0,2} \geq 0$ and hence $\kappa \geq \frac{\Delta t}{2}$ obtained from Eq. (3.25). In fact, in practice, the value of κ is usually set to be much greater than 1, meaning that $K_p >> K_i$. Still, since K_p has to decrease when going down, it means that K_i needs to decrease exponentially as well in order to prevent $K_i > K_p$ and keep κ constant.

To summarize, for the discrete system which possesses a small delay, instability of the closed-loop system will happen if $K_p > K_{cr}$ or $\kappa < \Delta t/2$. With the proposed adaptive controller shown in Eq. (3.20), K_p and K_i are kept small enough to prevent self-induced instability, while always being as high as possible to maximize control performance.

3.5.3. REAL-TIME OSCILLATIONS DETECTION

In the first phase of our landing strategy, the MAV has to increase its control gains until it starts to self-induced oscillations. In addition, in phase II it can happen that the MAV is not able to keep the flow divergence constant, and that it descends too fast. In that case, the exponentially decreasing gains K_p and K_i may cause self-induced oscillations that have to be detected and dealt with (by resetting the gains to appropriate values). For both these cases, we need a method to detect self-induced oscillations experienced by the vehicle in real-time.

There are a few methods in the literature to detect self-induced oscillations, typically relying on a fast Fourier transform (FFT) [88, 89]. The reason an FFT is used, is that self-induced oscillations are a “resonance” property of the control system and hence have a typical frequency. However, FFT methods are computationally expensive. Therefore, we detect self-induced oscillations by examining the covariance function of a windowed flow divergence $\widehat{D}$ and a time-shifted windowed flow divergence $\widehat{D}'$:

$$cov(\widehat{D},\widehat{D}') = E[(\widehat{D}-E(\widehat{D}))(\widehat{D}'-E(\widehat{D}')^T)], \tag{3.31}$$

where $E(\widehat{*})$ is the expected value of windowed flow divergence. The covariance is chosen, since it is much faster to compute than an FFT, while capturing both the relative phase and the magnitude of deviations in the signal. The “price paid” is that it will only react to a single frequency, but this is exactly what we want for the detection of self-induced oscillations.
Fig. 3.19 illustrates the oscillations detection method using the covariance function. The

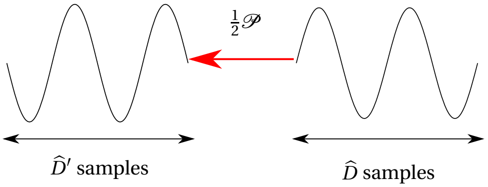


Figure 3.19: Oscillations detection method using the covariance function, $cov(\widehat{D},\widehat{D}')$.

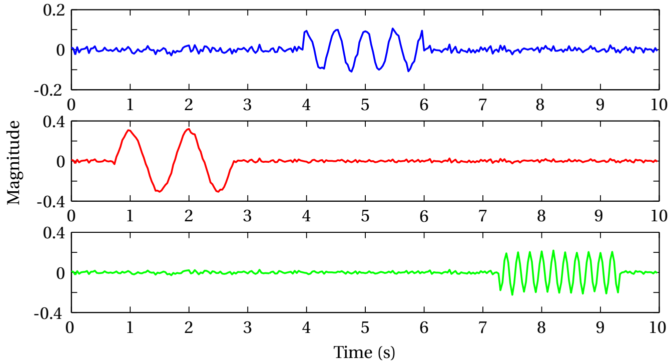
$\widehat{D}'$ samples set is the previous samples set of $\widehat{D}$, which is $\frac{1}{2}\mathscr{P}$ lagging behind the current samples set of $\widehat{D}$, where $\mathscr{P}$ is the period of one full oscillation.

To show the feasibility of this method, consider the signals which are generated with oscillations of different magnitudes (0.1, 0.3, 0.2) and frequencies (2, 1, 5 Hz), and noise is added to these signals as shown in Fig. 3.20a. By computing their discrete Fourier Transforms (DFT), the frequency of the oscillation in each signal can be found as presented in Fig. 3.20b. Then, the proposed covariance function of the generated signals can be computed, based on the period of the oscillation known from the DFT. Fig. 3.20c clearly shows that the covariances of the generated signals are large at the instance when the oscillations occur.

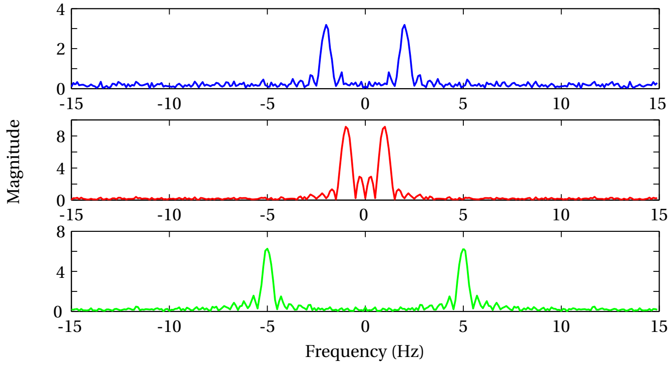
This method is implemented on the MAV and tested in the same experiment presented in Fig. 3.14. Fig. 3.21 shows the covariance of the flow divergence, indicating that we can successfully detect the oscillations in a real flight. For instance, in Fig. 3.14, we observe that strong oscillations happen from $8s$ to $10s$ and from $12s$ to $20s$, and this leads to highly negative covariance values at these time instances as shown in Fig. 3.21. Thus, this method provides a new way to detect oscillations in real-time.

3.6. FLIGHT TESTS

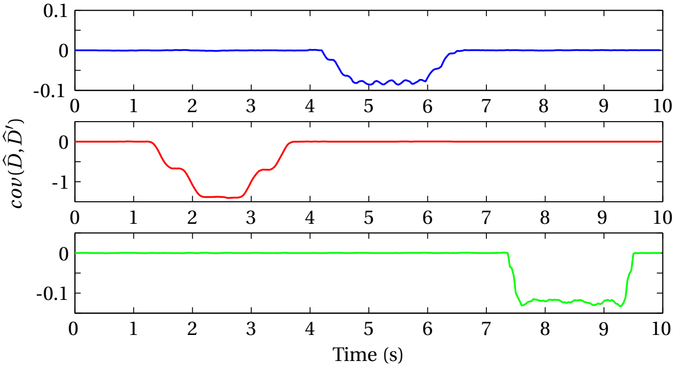
In this section, we present the flight tests results for vertical landing control using the adaptive controller. The same MAV platform as described in Section 3.3.3 is used for the experiments, with all vision and control algorithms running on-board. To show the



(a) Generated signals of different frequencies and amplitudes, with additional noise.



(b) Discrete Fourier Transform of the generated signals.



(c) Covariances of the generated signals with delays $\frac{1}{2}\mathscr{P}$ matching the half periods of the frequencies.

Figure 3.20: Generated signals with different magnitudes (0.1, 0.3, 0.2) and frequencies (2, 1, 5 Hz) of oscillations and noise, and the corresponding Discrete Fourier Transform and covariances.

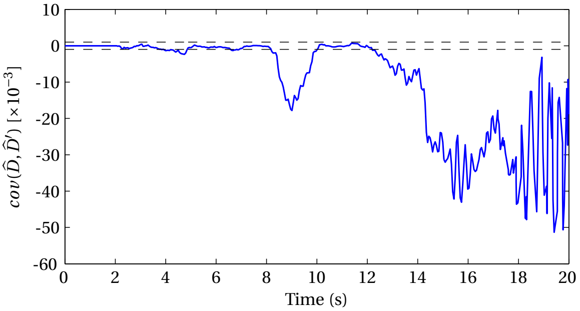


Figure 3.21: Covariance of the flow divergence obtained from the experiment presented in Fig. 3.14.

robustness of the proposed method in the face of external disturbances, flight tests are performed not only in indoor but also in outdoor environments as shown in Fig. 3.22. [2]

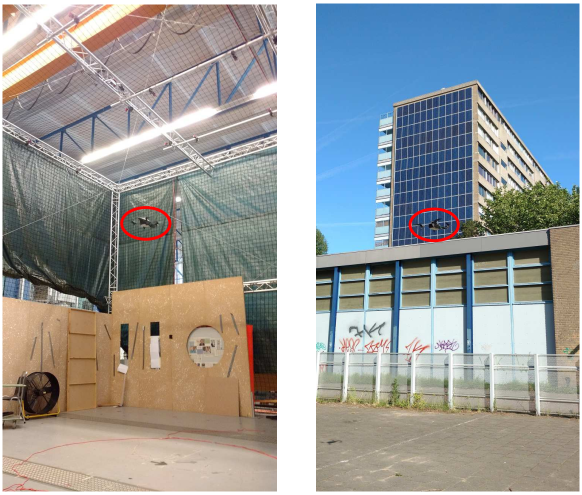


Figure 3.22: Indoor and outdoor environments for the flight tests.

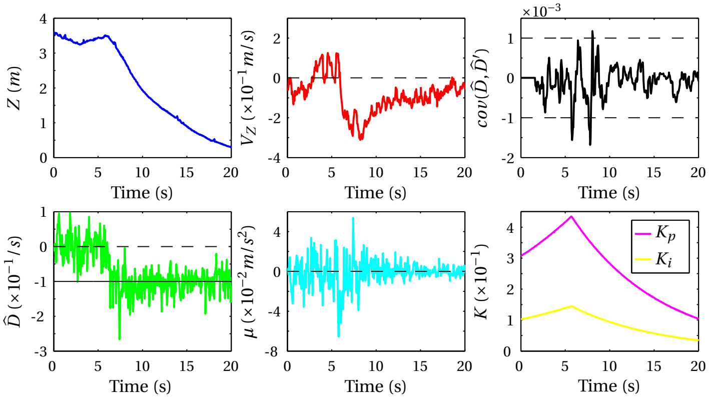
3.6.1. INDOOR FLIGHT TESTS

For indoor landing tests, the vertical control is performed using the flow divergence from an on-board camera while the horizontal control is executed using the position and velocity provided by the OptiTrack system. Fig. 3.23 shows the experiment results of the landings at different desired flow divergence values ($D^* = -0.1, -0.2, -0.3$).

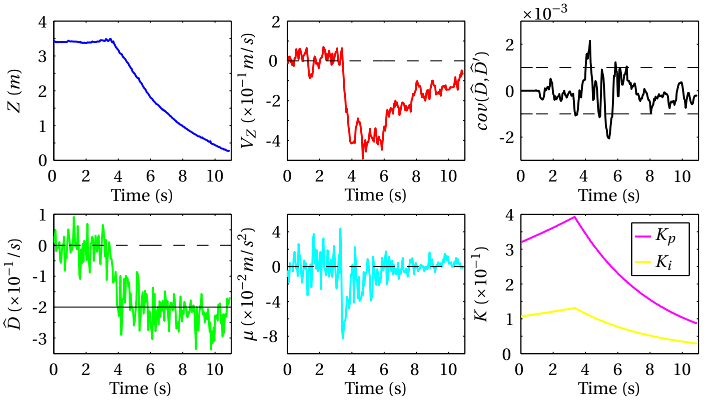
From the results shown in this figure, the controller gains are gradually increased until the first oscillation is detected while hovering, i.e., by tracking $D^* = 0$. After obtaining the initial gains, the landing starts by tracking $D^* = -0.1$, -0.2, or -0.3. There are two important observations from these results: 1) $\widehat{D} \approx D^*$, and 2) $cov(\widehat{D},\widehat{D}')$ is small

---

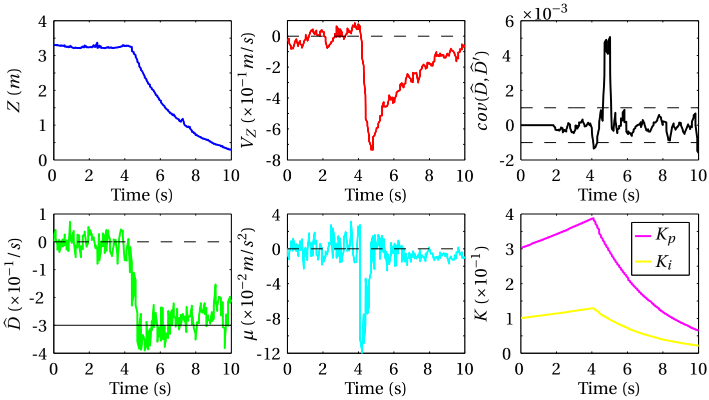
[2] Explanatory video with experiments: `https://goo.gl/ZDPP3m`



(a) $D^* = -0.1$.



(b) $D^* = -0.2$.



(c) $D^* = -0.3$.

Figure 3.23: Constant flow divergence landing using the adaptive controller (indoor environment).

and bounded ($\approx 10^{-3}$). This means that the tracking performance is good, and the self-induced oscillations are prevented (see Figs. 3.14 and 3.21 for comparison). Additionally, the adaptive control scheme performs well for the three desired flow divergences.

3.6.2. OUTDOOR FLIGHT TESTS

Outdoor flight is more challenging, due to the presence of wind. In outdoor experiments, the vertical dynamics are also controlled using flow divergence, while the horizontal dynamics are stabilized using translational optical flow estimates. The wind speed during the flights was reported to be around 8 $knots$, and the maximum gust was approximately 10 $knots$[3]. Fig. 3.24 shows the results of the outdoor landing tests at different desired flow divergences ($D^* = -0.1, -0.2, -0.3$).

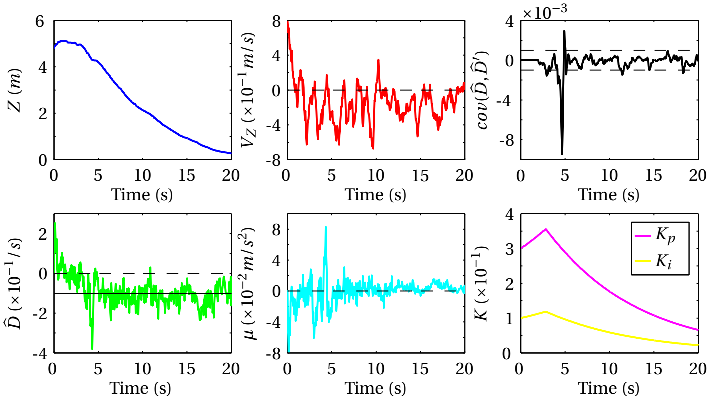
From the outdoor flight results, the adaptive controller tracks the desired flow divergences well, and the self-induced oscillations are prevented. Because of the unknown wind disturbances, it can be observed that slight perturbations exist, but the controller is sufficiently robust to deal with wind. Note that in the figure the height is obtained from an ultrasound sensor, while the vertical velocity is provided from a GPS which has an accurate of $< 3m$ and has a relatively low update rate, i.e., $5Hz$. Both these sensors have only been used for logging purposes, and not in the control.

3.7. CONCLUSIONS

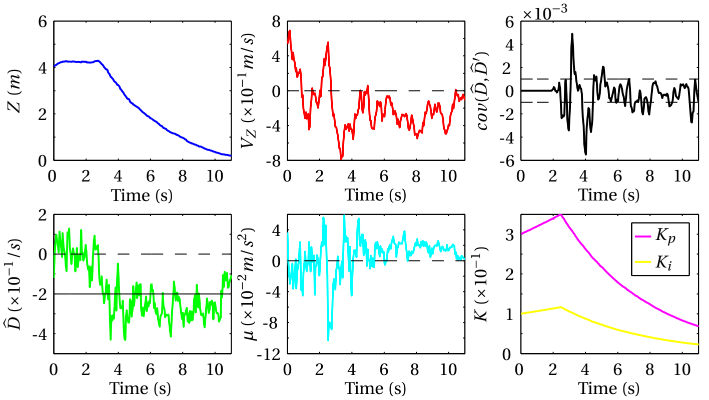
A control strategy has been developed to solve the fundamental problem of gain selection for constant flow divergence landings. The delay and noise models of the estimates were first obtained, and their effects on closed-loop control performance were investigated. In the presence of the delay and noise, computer simulations as well as real flight tests show that oscillations occur during the landings when a fixed-gain controller is used. We propose an adaptive controller which first initializes a near-optimal gain by means of an oscillating movement and then exponentially reduces this gain during descent. A stability analysis shows that the adaptive gain strategy indeed prevents self-induced oscillations and instability. This strategy was implemented on a Parrot AR.Drone 2.0 with all vision and control algorithms running on-board. Multiple successful landing flight tests, in both indoor and windy outdoor environments, were performed using the adaptive gain strategy.

---

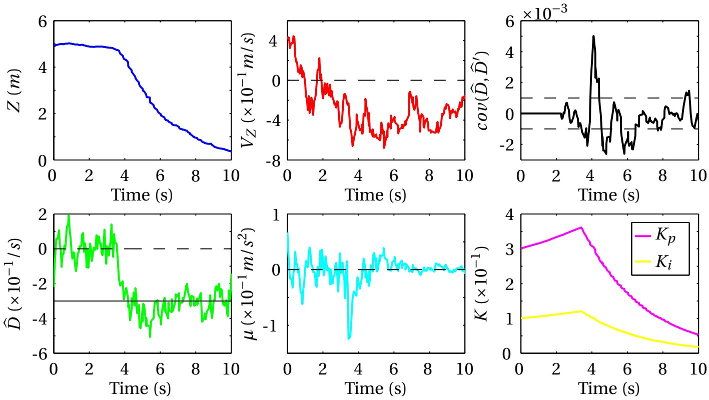
[3] Wind speed reference: https://www.windfinder.com/



(a) $D^* = -0.1$.



(b) $D^* = -0.2$.



(c) $D^* = -0.3$.

Figure 3.24: Constant flow divergence landing using the adaptive controller (outdoor environment).

II

HIGH-LEVEL LANDING CONTROL

4

SURFACE SLOPE ESTIMATION USING OPTICAL FLOW

Previous chapters show the low-level landing control of the MAV, which aims to deal with the scaleless property of optical flow. This chapter investigates the high-level landing control of the MAV which intends to search for suitable landing sites for an MAV. The safe landing sites are defined as relatively flat surfaces that do not have a too large inclination and, most importantly, are free of obstacles. To find a suitable landing site, this chapter proposes an algorithm that fits the optical flow field using RANdom SAmple Consensus (RANSAC). The results of the fitting provide two important estimates, i.e., the surface slope that represents the inclination of the ground surface and the surface roughness which indicates the presence of obstacles. This chapter presents the landing site selection using the surface slope for autonomous landing of an MAV.

This chapter begins by describing the vision algorithm that uses optical flow to estimate useful information for MAV landing in Section 4.2. After that, Section 4.3 examines the accuracy of the slope estimation by testing it on artificial images, hand-held videos, and images from an MAV. Then, Section 4.4 discusses the results of a landing experiment with an MAV. Lastly, conclusions are drawn in Section 4.5.

This chapter is based on the following article.
G. C. H. E. de Croon, H. W. Ho, C. de Wagter, E. van Kampen, B. D. W. Remes, and Q. P. Chu, Optic-flow based slope estimation for autonomous landing, International Journal of Micro Air Vehicles 5, 287 (2013). [84]

4.1. INTRODUCTION

Autonomous landing is an important capability for Micro Air Vehicles (MAVs), especially if they have to operate outside line-of-sight. While a barometer provides information on altitude, it does not provide information on the height of the MAV above the terrain, nor on the suitability of the terrain for landing. Active sensors such as a laser scanner [15] or multiple cameras as in stereo vision [77] do offer such information. They instantaneously estimate distances to many points on the landing surface. However, such sensor setups only work at lower heights and are not energy or weight efficient. Since such efficiency is important for MAVs, it would be beneficial to have a robust landing strategy based on a single downward-pointing camera.

With a monocular camera setup, two main approaches to autonomous landing can be employed. The first approach is visual Simultaneous Localization And Mapping (SLAM) [22, 23], in which the 3D locations of all features in sight are determined. Various approaches to visual SLAM have been proposed over the years, which have consistently improved with respect to accuracy and computational effort [24, 80]. Still, SLAM delivers a lot of detailed information on the environment that is not strictly required for the landing task, and hence uses more computational resources than necessary.

The second approach is bio-inspired in the sense that it directly uses the optical flow field for control. The earliest studies of this approach [33, 44] are based on the biological finding that bees keep the ventral flow constant during a grazing landing [36, 37]. The ventral flow is equal to the translational velocity divided by the height, and keeping it constant results in a soft touch down. Since the ventral flow cannot account for the vertical dynamics, recent engineering studies [47, 48, 90] complement the ventral flow with the time-to-contact, i.e., the height divided by vertical velocity. Interestingly, biologists have now confirmed that one of the engineered strategies is also used by honeybees: they keep the time-to-contact constant while landing on flat surfaces [49]. The advantage of a bio-inspired approach is that light-weight neuromorphic sensors with high sensitivity and update frequency can be used directly for controlling the landing [90, 91].

The above-mentioned bio-inspired landing studies focus on autonomous landing on a flat surface. However, many real-world missions for MAVs will involve unknown landing sites that may be cluttered or have a slope. Indeed, bees can choose their landing site and adapt their body attitude to the slope of the surface on which they are landing [92]. For an MAV the detection of the landing surface's slope would also be of interest, either for adapting the MAV's attitude or for evaluating the suitability of the surface for landing.

In this article, a computationally efficient computer vision algorithm is proposed that uses a bottom camera. First, optical flow vectors are determined and then a second order fit of the entire optical flow field is made. The fit provides information on the ventral flow, time-to-contact, roughness of the landing surface, and surface slope. The algorithm is computationally efficient and robust to noisy optical flow vectors that are likely to occur in a real MAV landing scenario. In addition, it lends itself well for translation to algorithms for novel sensors that mimick insect eyes [57, 93].

The remainder of the article is structured as follows. In Section 4.2, the vision algorithm is explained. It is tested on image sequences in Section 4.3. The results of a landing experiment with a Parrot AR drone are discussed in Section 4.4. Finally, conclusions are drawn in Section 4.5.

4.2. VISION ALGORITHM FOR SLOPE ESTIMATION

4.2.1. OPTICAL FLOW EQUATIONS FOR A SLOPED PLANAR SURFACE

The vision algorithm proposed for the slope estimation is based on the early optical flow work in [85]. In the explanation of the algorithm, a pinhole camera model is employed. The algorithm assumes (1) the camera to point downward, (2) the rotation rates to be known from the MAV's state estimation (e.g., using gyros), and (3) the landing surface in sight to be predominantly planar. Under these assumptions, the optical flow vectors in the image follow the equations:

$$u = (-V_x + xV_z)/Z, \tag{4.1}$$

$$v = (-V_y + yV_z)/Z, \tag{4.2}$$

with u and v the (derotated) optical flow in the x- and y-direction of the image, and V_x, V_y, V_z the motion in X, Y, and Z direction, respectively. x and y are image coordinates. Figure 4.1 shows the relevant coordinate axes. Please note that the Z-axis is aligned with the camera's optical axis and that the coordinate $(X,Y,Z) = (0,0,0)$ is located at the intersection of the camera's principal axis with the ground surface. The height of the camera above the ground surface is represented with h. The surface height Z can be modelled as a plane:

$$Z = h + aX + bY, \tag{4.3}$$

with h the height above the surface at $(x, y) = (0,0)$. The slope angles of the surface are:

$$\alpha = \text{atan}(a) \tag{4.4}$$

$$\beta = \text{atan}(b) \tag{4.5}$$

By a transformation of coordinates, in [85], the surface height is rewritten as:

$$z = (Z - h)/Z = ax + by, \tag{4.6}$$

, where $x = X/Z$ and $y = Y/Z$. Please remark that the normalized x-coordinate in Eq. (4.6) is related to the pixel coordinate $\tilde{x}$ by $x = \tilde{x}/\tilde{f}$, where $\tilde{f}$ is the focal length in pixels. In order to keep the equations uncluttered, normalized coordinates will be used for the remainder of the article. However, one should keep in mind that the actual slope angle in degrees can only be retrieved in the case of a calibrated camera (with $\tilde{f}$ known).

In addition to the coordinate transformation, the velocities are scaled with respect to the height h, i.e., $\vartheta_x = V_x/h$, $\vartheta_y = V_y/h$, and $\vartheta_z = V_z/h$. This leads to:

$$u = (-\vartheta_x + x\vartheta_z)(1 - z) \tag{4.7}$$

$$v = (-\vartheta_y + y\vartheta_z)(1 - z) \tag{4.8}$$

Replacing z with $ax + by$ (from Eq. (4.6)), leads to:

$$u = -\vartheta_x + (\vartheta_x a + \vartheta_z)x + \vartheta_x by - a\vartheta_z x^2 - b\vartheta_z xy \tag{4.9}$$

$$v = -\vartheta_y + \vartheta_y ax + (\vartheta_y b + \vartheta_z)y - b\vartheta_z y^2 - a\vartheta_z xy \tag{4.10}$$

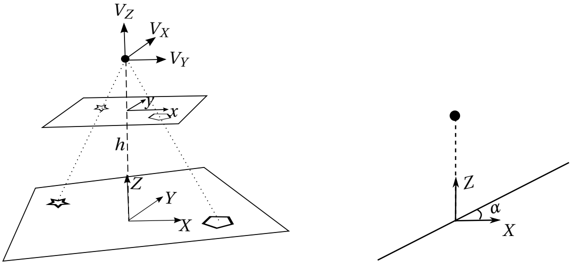


Figure 4.1: Left: the main coordinate axes involved in the optical flow calculations. Right: illustration of a ground surface with a nonzero slope α.

4.2.2. OPTICAL FLOW FIELD PARAMETER ESTIMATION

In [85], the rotational optical flow terms were left in, and the discussion hence goes toward how to find the Focus-of-Expansion (FoE) and determine the first and second order spatial flow derivatives at that point. In that way, all terms can be identified (translational, rotational, and the slopes of the surface). However, determining the FoE is a difficult task in itself, and errors in its location can have a large influence on the subsequent results. In addition, since MAVs typically have access to gyro measurements, the rotational components do not have to be determined. So instead of finding the FoE, the vision algorithm proposed in this article immediately determines the parameters of the optical flow field:

$$u = \mathbf{p_u}[1, x, y, x^2, xy]^T, \tag{4.11}$$

$$v = \mathbf{p_v}[1, x, y, y^2, xy]^T, \tag{4.12}$$

The parameter vectors $\mathbf{p_u}$ and $\mathbf{p_v}$ are estimated separately with a maximal likelihood linear least squares estimate within a robust random sample consensus (RANSAC) estimation procedure [86], with 5 points per fit and 20 iterations. Figure 4.2 illustrates the result of such a fit.

4.2.3. EXTRACTION OF VARIABLES RELEVANT FOR LANDING

The so-determined parameters can then be used to estimate the following variables of interest for an autonomous landing. The **ventral flow** is set to $(\vartheta_x, \vartheta_y) = (p_{u0}, p_{v0})$. The **time-to-contact** and **slopes** can be retrieved from the first order spatial derivatives at the center of the image $(x, y) = (0, 0)$:

$$\left.\frac{\partial u}{\partial x}\right|_{x=0,y=0} = \vartheta_x a + \vartheta_z = p_{u1}, \tag{4.13}$$

$$\left.\frac{\partial u}{\partial y}\right|_{x=0,y=0} = \vartheta_x b = p_{u2}, \tag{4.14}$$

$$\left.\frac{\partial v}{\partial x}\right|_{x=0,y=0} = \vartheta_y a = p_{v1}, \tag{4.15}$$

$$\left.\frac{\partial v}{\partial y}\right|_{x=0,y=0} = \vartheta_y b + \vartheta_z = p_{v2}, \tag{4.16}$$

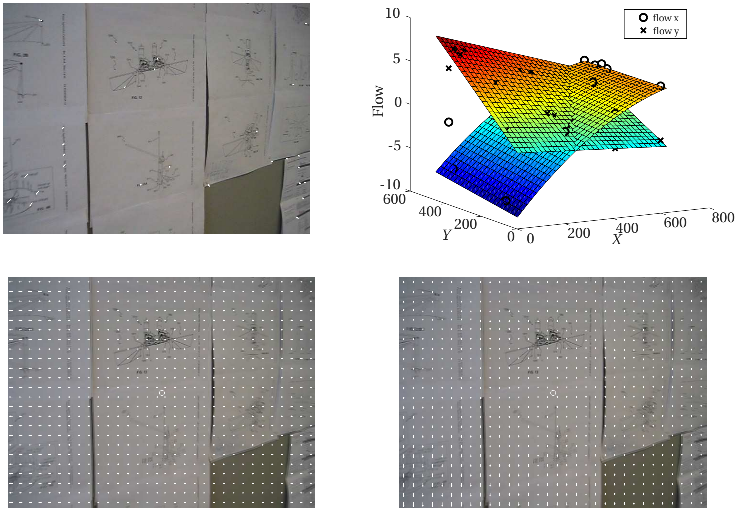


Figure 4.2: Illustration of the vision process. Top left: the determined optical flow vectors. Top right: quadratic fits for the flow in x-direction (circle markers) and y-direction (cross-markers). Bottom left: estimated flow field in x-direction. Bottom right: estimated flow field in y-direction. The motion of the camera is straight towards the wall, which has an angle of ∼ 45° to the camera axis.

The slopes can then be retrieved as $a = p_{v1}/\vartheta_y$ and $b = p_{u2}/\vartheta_x$. However, these equations become ill-conditioned and hence sensitive to even the smallest of noise if ϑ_x or ϑ_y are small. If there is insufficient motion in X and Y direction, then it may still be possible to estimate the slopes on the basis of the second order derivatives:

$$\frac{\partial u}{\partial x \partial x} = -2a\vartheta_z = 2p_{u3}, \tag{4.17}$$

$$\frac{\partial u}{\partial x \partial y} = -b\vartheta_z = p_{u4}, \tag{4.18}$$

$$\frac{\partial v}{\partial x \partial y} = -\vartheta_z a = p_{v4}, \tag{4.19}$$

$$\frac{\partial v}{\partial y \partial y} = -2b\vartheta_z = 2p_{v3}, \tag{4.20}$$

which leads to two formulas for a and two formulas for b. However, all of these formulas depend on ϑ_z, the relative vertical velocity. When looking at the formulas for the first order spatial derivatives, one will notice that ϑ_z cannot be determined without knowing a or b - seemingly introducing a chicken-and-egg problem. The solution lies in remembering that we only need these second order derivatives if the ventral flow estimate(s) are small. If, for example, $\vartheta_x \approx 0$ then:

$$\frac{\partial u}{\partial x} = \vartheta_x a + \vartheta_z \approx \vartheta_z, \tag{4.21}$$

and, similarly, for $\vartheta_y \approx 0$:

$$\frac{\partial v}{\partial y} = \vartheta_y b + \vartheta_z \approx \vartheta_z, \tag{4.22}$$

In summary, when the horizontal flow is not sufficient, the relative velocity ϑ_z can be determined. In turn, this leads to slope estimates $a = -p_{u3}/\vartheta_z$, $a = -p_{v4}/\vartheta_z$, $b = -p_{u4}/\vartheta_z$, and $b = -p_{v3}/\vartheta_z$. All these equations become ill-conditioned if ϑ_z becomes small. This is intuitive, since if there is also little motion in the Z-direction, then no estimates of slope are possible. The remaining question is then how to deal with cases in which an MAV has velocities in multiple directions. For optimal estimation, one should fuse the different a and b estimates on the basis of their certainties and possible prior probability distributions. However, in this preliminary work slopes are determined on the basis of the first-order optical flow derivatives if there is sufficient motion in the X,Y-plane and only on the basis of the second-order optical flow derivatives if there is not.

The time-to-contact τ is determined on the basis of the divergence:

$$D = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \tag{4.23}$$

, with $\tau = 2/D$. Hence, τ includes the ventral flow and the slopes (see Eqs. (4.13) and (4.16)). This makes sense, because in the case of pure horizontal motion, the MAV can still intersect the landing surface if it has a slope. However, if one is only interested in the component in the Z-direction, $\tau = 1/\vartheta_z$ could be used. In the current work the first definition is employed. Please note that in contrast to the slope angle, the time-to-contact can also be determined without knowledge of $\tilde{f}$. Moreover, the time-to-contact is in principle expressed in frames. Knowledge of the time interval between image frames is necessary to transform it to seconds.

Finally, the **roughness** of the slope is related to how good the optical flow vectors fit with the above-described quadratic model. The RANSAC procedure returns a number of inliers and an error, which can both serve as measures for roughness.

4.2.4. IMPLEMENTATION

A main decision for the implementation of the vision algorithm described above is the choice of optical flow algorithm. We employ a sparse optical flow setup with the method of Shi and Tomasi [94] to find good features for tracking and Lucas-Kanade's pyramid method [95] for optical flow determination. We have made both an implementation in MATLAB, for off-line experiments, and in C++ for online experiments on an MAV. The MATLAB code of the vision algorithm is available at `http://mavlab.lr.tudelft.nl/`, so that the reader can test it on his / her own image sequences. Please note that the open source code involves our own implementation of Lucas-Kanade's optical flow tracking, while we used the implementation of [71] for the experiments. This code gives slightly better results. The code from [71] is openly available, but copyrighted, so we chose not to include it in our code package. However, readers can download it and incorporate it easily into the code.

4.3. EXPERIMENTS ON IMAGE SEQUENCES

In this section, a preliminary test of the vision algorithm is performed by applying it to various image sequences. The algorithm is tested on three types of image sequences: (1) image zooms, (2) manually made videos, and (3) images from the Parrot AR drone. Figure 4.3 shows five example images.

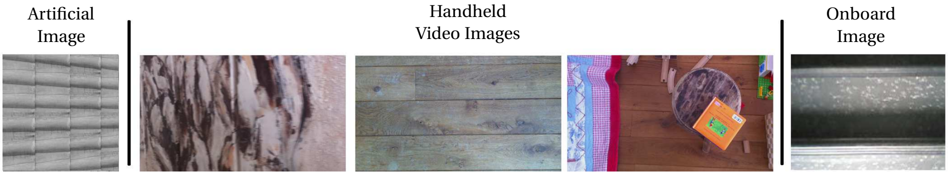


Figure 4.3: Example images. From left to right: artificial image, images from manual videos (wall, flat scene, structured scene), image from the drone's bottom camera.

The test on artificial image sequences is motivated by the fact that in this setup the ground-truth values are exactly known. The virtual camera used in these experiments has a field-of-view of 50° and generates relatively small 256 × 256 pixel images. To test the determination of time-to-contact, five image zooms were performed on a flat roof texture. The ground-truth time-to-contact decreases at a constant rate from 200 to 5. The results of the time-to-contact estimation are shown in Figure 4.4. The estimated time-to-contact is very accurate, even up to 200 frames from contact.

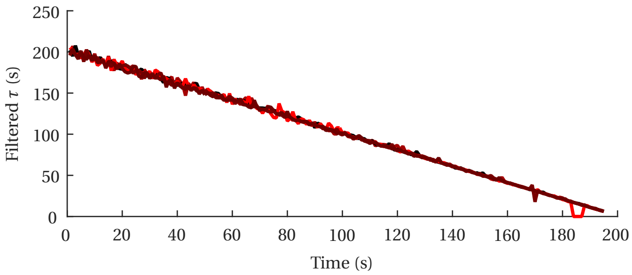


Figure 4.4: Results of time-to-contact estimation on five image zooms. The ground-truth time-to-contact goes from 200 to 5.

In order to test the slope estimation, a virtual camera moves with respect to the roof texture image, which is placed at various slope angles in the X-direction: $\alpha \in \{0, 15, 30, 45, 60\}^\circ$. Three types of movements are studied: horizontal movement in the X-direction, horizontal movement in the Y-direction, and vertical movement towards the artificial landing surface (Z-direction).

Figure 4.5 shows the results of the artificial movement sequences. The top left plot shows the slopes estimated during a constant movement in the Y-direction (with $\vartheta_y \sim 3$). Overall, the estimated slopes are rather close to the ground-truth values, with errors in the order of a few degrees. There are a few outlier cases with errors of up to 15 degrees. The top right plot shows the more difficult case of a constant movement toward the surface, in the Z-direction. Although the estimates rely on second-order derivatives, the slopes are still estimated rather accurately. During the last 10 time steps of the approach, the implemented optical flow algorithm has problems determining the correct optical flow vectors. This leads to a degradation of the slope estimation. The bottom left plot shows the slopes estimated during a constant movement in the X-direction. Although the results are close to the ground-truth values at the start of the approach, after 40 frames a few of the estimates start to deviate from the ground-truth in the order of tens of degrees. The cause of this phenomenon lies in the fact that the movement brings the virtual camera further from the landing surface, reducing the magnitude of the ventral flow ϑ_x. The bottom right plot shows ϑ_x over time during the movement. A smaller ventral flow leads to less accurate slope estimates.

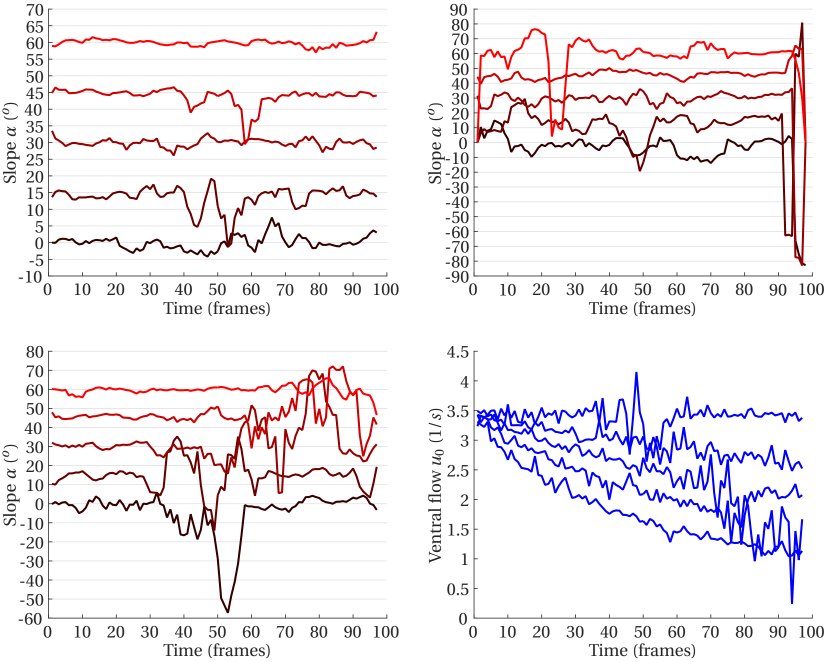


Figure 4.5: Slope estimation results on artificial movement sequences. The ground-truth values for the slopes α are $\{0, 15, 30, 45, 60\}^\circ$. Top left: slopes estimated during movement in the Y-direction. Top right: slopes estimated during movement in the Z-direction. Bottom left: slopes estimated during movement in the X-direction. Bottom right: ventral flow ϑ_x during movement in the X-direction.

For further testing, three sets of videos have been made of a textured wall. In the first set the camera moves in the Y-direction (first up and then down), in the second set in the X-direction (first left and then right), and in the third set the camera moves in the Z-direction (in this case toward the wall). Put in a landing context, the first two sets cover horizontal motion, while the third set covers vertical motion toward the landing surface. Each set contains 5 image sequences in which the angle of the wall roughly iterates over the angles $\{0, 15, 30, 45, 60\}$ degrees. Please remark that the camera is moved in-hand, so additional motions and even rotations are present in the images. Since there is no clear ground-truth and the camera is uncalibrated, this experiment mainly serves the goal of verifying that the slope a increases with an increasing angle to the wall (the slope values are not transformed to angles in this case). Figure 4.3 shows an example image from the sequence at ~ 45 degrees.

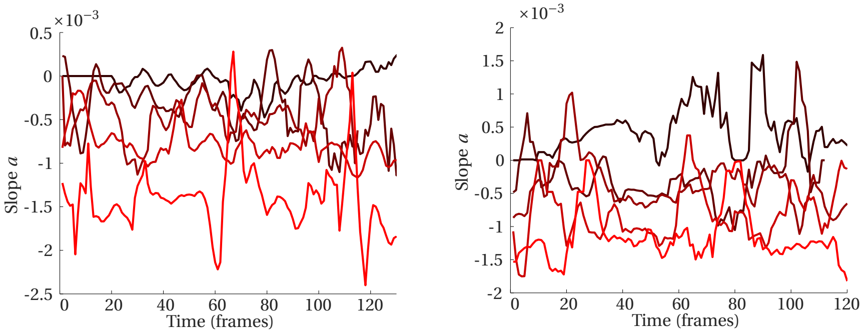


Figure 4.6: Results of slope estimation on video sequences. Left: motion in Y-direction. Right: motion in X-direction. The plots show the estimated slope a (y-axis) over time (x-axis). The brightness of the lines are determined by the angle with the wall, with the darkest color corresponding to 0 degrees and the brightest color corresponding to 60 degrees.

Figure 4.6 shows the results of this experiment for motion in the Y-direction (left plot) and X-direction (right plot). The plots show the estimated slope a (y-axis) over time (x-axis). The brightness of the lines are determined by the angle with the wall, with the darkest color corresponding to 0 degrees and the brightest color corresponding to 60 degrees. The lines are somewhat noisy, especially when the motion gets small, e.g., at reversal points of the movements. However, in both plots, a clear ordering can be seen from dark to bright slopes, as desired.

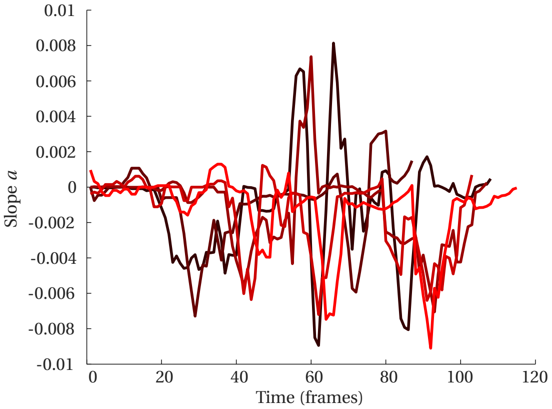


Figure 4.7: Results of slope estimation on video sequences with the camera moving in Z-direction. The plots show the estimated slope a (y-axis) over time (x-axis). The brightness of the lines are determined by the angle with the wall, with the darkest color corresponding to 0 degrees and the brightest color corresponding to 60 degrees.

Figure 4.7 contains the results for motion in the Z-direction, illustrated in the same manner. In this case, the results are much less clear. The lower degree slopes switch between positive and negative values, while the higher degree slopes are constantly negative. The worse results may be explained by the rotations in the video not accounted for by optical flow derotation. In addition, the second-order spatial derivatives of the optical flow field are more difficult to determine than the first order ones.

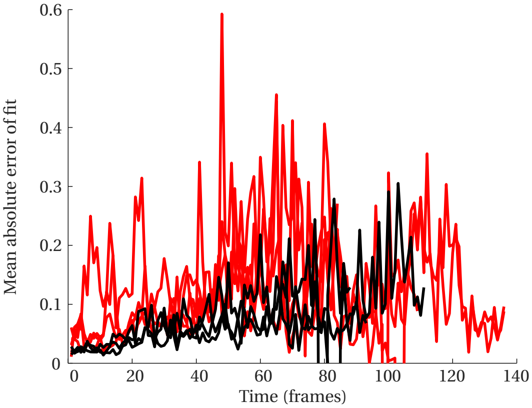


Figure 4.8: Results for estimation of landing surface roughness. Mean absolute error of the quadratic fits (y-axis) over time (x-axis). The black lines illustrate the results for the flat surfaces, while the red lines illustrate the results for the scenes with 3D structure.

Subsequently, the detection of roughness is tested with the help of two sets of each three videos. The first set contains flat surfaces, while the second set contains considerable 3D structure. The videos are made by moving toward the ground surface, i.e., in the Z-direction. The results are shown in Figure 4.8, which shows the mean absolute errors of the quadratic fits. The black lines illustrate the results for the flat surfaces, while the red lines illustrate the results for the scenes with 3D structure. The videos of flat surfaces have clearly a lower error than the videos of structured scenes. However, placing a threshold is not straightforward, as the error also seems to depend on the time-to-contact. In particular, lower time-to-contacts entail larger optical flow, which leads to larger errors.

Finally, the slope estimation algorithm is run on the 160×120 pixel images of the Parrot AR drone's bottom camera, while it moves down a stairs. The rotations of the drone are not accounted for by derotation. Figure 4.9 shows the estimated slope a in the x-direction (blue) and b in y-direction (green) during a part of the trajectory over the stairs. The y-direction is in the direction of the stairs. Indeed, while a is reasonably small and switches from positive to negative and back, b is larger and mostly negative, as it should be. Please note that the slopes are not expressed in angles.

In summary, when the camera is calibrated and rotation is accounted for, the vision algorithm is able to retrieve the angle of a landing surface rather accurately. In the case of an uncalibrated camera and in the presence of small rotations, the slope estimates vary clearly with the angle of the wall in the case of movements in the X,Y-plane. The slope

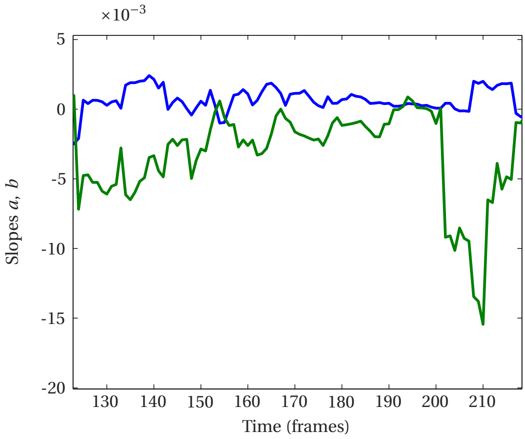


Figure 4.9: Results for estimation of slope with the Parrot AR drone's bottom camera. The plot shows a (blue) and b (green) over time (x-axis).

angle estimation then depends only on the first order spatial derivatives of the optical flow. A higher ventral flow leads to better slope estimates. The results on videos with motion in the Z-direction give worse results. Finally, the vision algorithm provides a cue for the detection of 3D structure below the MAV.

4.4. LANDING EXPERIMENT

The above-described vision algorithm is implemented in TU Delft's *SmartUAV* ground control station, and tested in real-time on a Parrot AR drone. The goal of the experiment is to show that the vision algorithm can work in real-time, real-world conditions.

In order to show that the vision algorithm can be used in a generic way, a staircase was chosen as the test environment. Stairs are challenging, since it is not a flat inclined surface. The setup for the experiment is presented in Figure 4.10. The experiment started with a forward motion of the drone (in the Y-direction) generated by a small pitch angle. This forward motion was controlled by proportional controller using the feedback of the velocity estimated from the AR drone optical flow measurement. The purpose of moving the drone in forward direction was to generate ventral flow to estimate the slope in real-time. The drone thus travels down the stairway and to a flat ground in the end. The height of the drone was under control of the Parrot firmware, implying that it remained at a fixed height above the steps of the stairs.

A simple landing strategy was used in this preliminary experiment, i.e. when the value of the estimated slope was close to zero $b \in [-0.0005, 0.0005]$, a landing command was given to land the drone immediately. The results of the estimated slope b from the onboard images captured during the flight is shown in Figure 4.11. It is clearly seen that the value of the slope was always negative when the drone was flying above the stairs. At the instant when it reached the flat surface, the value of the slope changed its sign and the autoland was activated to bring the MAV to the ground.

Please note that the drone moves forward by pitching forward. The experimental

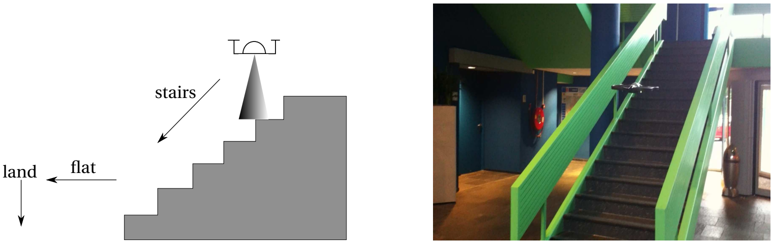


Figure 4.10: Landing experiment setup

results show that this considerably influences the estimated slope, which was not accounted for in the code. This led to slightly less negative slope values above the stairs and slightly more positive slope values above the flat floor[1]. If a camera calibration is available, the angle of the surface can be estimated in degrees. The estimated attitude of the MAV can then be subtracted from this angle.

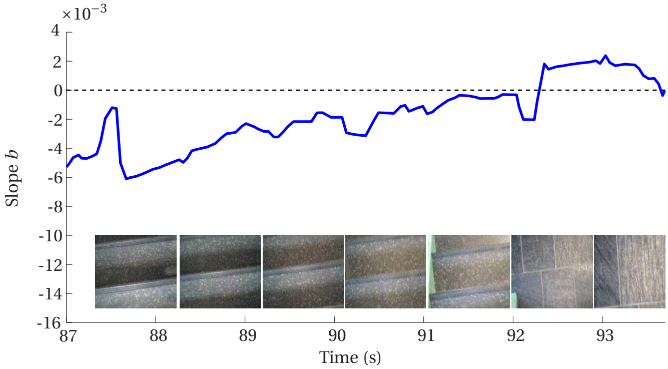


Figure 4.11: Real-time slope estimation results with the Parrot AR drone's bottom camera. The plot shows the estimated slope b and images taken from the onboard camera over time (x-axis)

4.5. CONCLUSION

The main conclusion is that the proposed vision algorithm is able to extract ventral flow, time-to-contact, roughness, and slope of the landing surface beneath an MAV. A preliminary landing experiment shows that the algorithm is computationally efficient enough to run in real-time and can discriminate between the stairs and a flat surface. The experiments show that it is important to take into account the pitch angle of the MAV while determining the slope of the surface.

Future work will focus on a further investigation of how optical flow can be used to best estimate the variables of interest. For example, the current maximal likelihood estimation has its limitations, especially if there is a lot of noise or if the assumption is violated that rotation is accounted for. A maximal a posteriori estimate probably would

---

[1] A video of the experiment can be found here: `http://www.youtube.com/watch?v=TXIPb1NvUJY`

better cope with such situations. In addition, combining slope estimates in a Bayesian fashion could considerably improve the estimates and allow for the exploitation of translation in multiple directions. In order to create such a Bayesian estimation scheme, we need to gain more insight into matters such as the relation between the magnitude of the ventral flow and slope estimation accuracy. Finally, in this article we have not studied the control-part of landing in detail. In future work, a more elaborate landing procedure utilizing the estimated variables will be devised and tested in a more complex environment.

5

SELF-SUPERVISED LEARNING OF OBSTACLE APPEARANCE

In the previous chapter, the surface slope estimated from the optical flow is used to find a suitable landing site for an MAV. However, using optical flow requires the MAV to move. When there is hardly any vehicle movement, there is not enough visual information. To solve this problem, this chapter attempts to 'go beyond' optical flow approaches for the safe landing site selection. A novel setup of self-supervised learning (SSL) is introduced, in which optical flow serves as a scaffold to learn the visual appearance of obstacles.

This chapter first discusses related works on self-supervised learning in Section 5.2. After that, Section 5.3 describes the proposed concept to learn the visual appearance of obstacles based on the surface roughness from optical flow. Then, Section 5.4 shows the results of both optical flow- and appearance-based obstacle detection. Section 5.5 explains the generalization of the SSL approach to different environments. In Section 5.6 demonstrates landing experiments where the proposed algorithms run onboard an MAV. Section 5.7 discusses the results and potential applications of the proposed approach. Finally, conclusions are drawn in Section 5.8.

This chapter is based on the following articles.
H. W. Ho, C. De Wagter, B. D. W. Remes, and G. C. H. E. de Croon, Optical flow for self-supervised learning of obstacle appearance, in Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on (IEEE, Hamburg, Germany, 2015) pp. 3098–3104 [96].
H. W. Ho, C. De Wagter, B. D. W. Remes, and G. C. H. E. de Croon, Optical-flow based self-supervised learning of obstacle appearance applied to MAV landing, The International Journal of Robotics Research (submitted). [97]

5.1. INTRODUCTION

To reduce the risk and cost of human intervention, autonomous flight of Micro Air Vehicles (MAVs) is highly desired in many circumstances. In particular, autonomous landing is a challenging, but essential task of the flight, as it needs to be done in a limited space and time [78]. Hence, quickly searching for a safe landing spot is required during landing in autonomous flights [98].

Many existing methods for finding a suitable landing spot use multiple cameras [17–19] or active sensors such as a laser range finder [15, 77] to estimate the distance to many points on the landing surface. While both methods can provide accurate measurements, their perception range is limited and they are heavy and costly for small MAVs. Therefore, use of a single camera is preferable as it is light-weight and has low power consumption [79].

State-of-the-art algorithms for autonomous landing purely rely on motion cues. There are two main approaches: (1) visual Simultaneous Localization and Mapping (SLAM) and (2) bio-inspired optical flow control.

The first approach can be categorized into feature-based and direct methods to solve the SLAM problems, i.e., to determine the vehicle's location and 3D-structure of the surrounding environment. The feature-based method [28, 80] decouples these SLAM problems into extraction of features and computation of camera pose and scene geometry based on tracking these features [22–24]. However, this approach is not sufficiently robust to challenging scenes where the features are hardly detected. The direct method tries to avoid this limitation by using image intensities directly to generate (semi-) dense maps [25–27]. Although the computational efficiency and accuracy of visual SLAM have been improved over the years [28–32], this approach still uses more computational resources than are strictly necessary.

The second approach is inspired by flying insects, which heavily rely on optical flow for navigation. Biologists first found that honeybees perform a grazing landing by keeping the ventral flow (lateral velocities divided by height) constant [33, 36, 37, 44]. This approach guarantees a soft landing but does not control its vertical dynamics. To deal with that, recent studies proposed using time-to-contact (height divided by vertical velocity) [47–49]. The optical flow field can also be used during landing to identify and avoid obstacles [84, 99].

For sensing obstacles with motion cues, either the vehicle or the obstacle obviously needs to move. This requirement is a drawback of motion cue approaches because it would be both safer and more efficient for MAVs that have hovering capability to detect the obstacles underneath the vehicle in hover. This would require MAVs to exploit currently unused *appearance* cues - the information contained in still images. Human pilots are very able to identify potential landing sites in still images, by recognizing obstacles and flat landing areas in view. This is based on years of experience, to learn what obstacles look like.

In this paper, we propose an approach that allows MAVs to learn about the appearance of obstacles and suitable landing sites completely by themselves. The approach involves a novel setup of Self-Supervised Learning (SSL), in which motion cues provide the targets for the supervised learning of a function that maps appearance features to a surface roughness measure (see Fig. 5.1 for an overview). We showed the feasibility of the proposed SSL concept in [96]. The current article significantly extends upon [96] by means of a systematic analysis with experiments. This includes experiments in which the learned appearance is used in the control loop and a study of the generalization of the learned appearance to various indoor and outdoor environments.

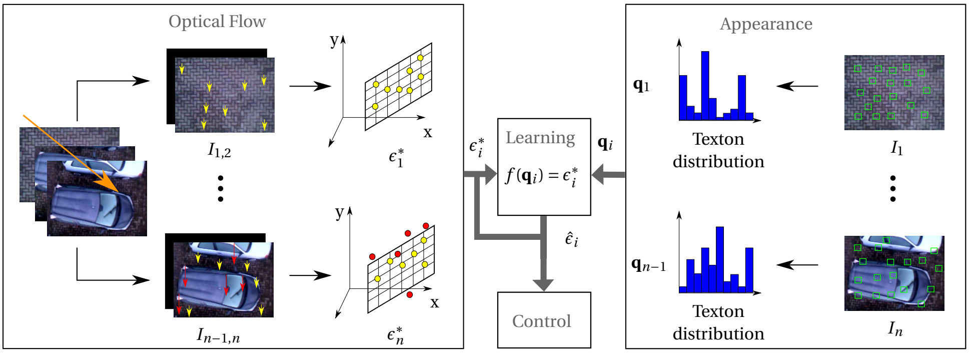


Figure 5.1: Overview of the novel self-supervised learning (SSL) setup. The MAV starts flying using a roughness measure ϵ^* extracted from optical flow algorithm (left). A function is learned that maps appearance features, $\mathbf{q}$ from a *still* image to an estimate of the roughness measure $\hat{\epsilon}$ (right). After learning, the MAV can determine whether there are obstacles below based on a still image, allowing landing site selection in hover.

The remainder of the article is set up as follows: In Section 5.2, we discuss related work on self-supervised learning in more detail. In Section 5.3, we describe our proposed concept to learn the visual appearance of obstacles based on *surface roughness*, ϵ^* from optical flow. Section 5.4 presents the results of both optical flow- and appearance-based obstacle detection, and Section 5.5 explains generalization of our SSL approach to different environments. Then, Section 5.6 demonstrates landing experiments where the proposed algorithms run onboard an MAV. In Section 5.7, we discuss the results and potential applications of the proposed approach. Finally, we draw conclusions in Section 5.8.

5.2. RELATED WORK

There are several remarkable achievements with SSL on autonomous driving cars, where stereo vision, laser scanners, or bumpers provided supervised outputs to learn the appearance of obstacles on the road [100–102]. In [100, 101], a close-range map is generated by the laser scanners to identify a nearby patch of drivable surface. This patch is used to train appearance models to extend the road detection range. In [102], terrain classification obtained from 3D information using stereo camera is used for training a convolutional neural network (ConvNet). The input image patches to the trained ConvNet need to be normalized based on the estimated distance so that the obstacles always have the same size. Then, the trained model can be used to detect obstacles beyond the range of stereo camera.

In [103], optical flow was used for tracing back the obstacles in time when they are far away. The supervised outputs were still provided by stereo vision and bumpers. A non-linear regression based depth estimation method using only a monocular camera was used for MAVs to learn a deliberate scheme for navigating through a cluttered environment [104]. However, the training set was made using a stereo vision system on a ground robot and offboard image processing was done in the flight.

A major difference of the approach we propose and previous work on SSL is that optical flow is used for generating the supervised outputs. To the best of our knowledge, there is only one other SSL study that also used optical flow to provide the supervised outputs. The study in [105] used optical flow from a camera mounted on a car to learn a ground color model, assuming knowledge of the camera position relative to the ground. The learned ground color model aided in filtering optical flow vectors in order to improve the accuracy of the optical-flow based visual odometry.

In this article, we use the optical flow from a downward-looking camera mounted on an MAV to learn the appearance of obstacles. In contrast to [105], we intend for the MAV to be able to use the learned appearance of obstacles even in the absence of supervisory cue of optical flow. This leads to a very interesting extension of the MAV's autonomous flight capabilities: while the robot initially only uses motion cues, and hence needs to move significantly in order to see obstacles, after learning it is able to see obstacles without moving.

5.3. METHOD

An overview of the proposed method is shown in Fig. 5.1. In the proposed setup of SSL, the MAV first uses optical flow to determine the landing surface roughness ϵ^*. Simultaneously, the MAV extracts appearance features from each image, resulting in a feature vector $\mathbf{q}$. During operation, it uses the ϵ^* as targets for the supervised learning of $f(\mathbf{q})$, a function that maps appearance features to the surface roughness. After learning, the MAV can map $\mathbf{q}$ directly to an estimate of surface roughness $\hat{\epsilon}$.

In this study, we focus on computationally efficient methods for the real world application to small MAVs which have limited computing capabilities. Therefore, we select a fast computer vision method and straightforward learning model which are suitable for real-time implementation. However, the proposed SSL setup does not preclude the use of other advanced computer vision and learning methods which could give even better results in terms of accuracy and generalization, at the cost of more computational effort.

We explain our method in three subsections. First, in Subsection 5.3.1 the optical flow algorithm to estimate ϵ^* is introduced. Second, in Subsection 5.3.2 the texton method to represent appearance with $\mathbf{q}$ is explained. Finally, in Subsection 5.3.3 the learning of the function $\mathbf{f}(\mathbf{q})$ is described.

5.3.1. SURFACE ROUGHNESS FROM OPTICAL FLOW

In this section, a computationally efficient method is proposed to extract information from the optical flow field, which will allow the MAV to detect obstacles and determine if a landing spot is safe. The optical flow algorithm used to determine a safe landing spot is based on early findings in [85]. The algorithm was developed in previous research [84] by assuming that (a) a pinhole camera model pointing downward is used, (b) the surface in sight is planar, and (c) the angular rates of the camera can be measured and used to de-rotate the optical flow. Under these assumptions, the equation of the optical flow vectors can be expressed as follows:

$$u = -\vartheta_x + (\vartheta_x a + \vartheta_z)x + \vartheta_x by - a\vartheta_z x^2 - b\vartheta_z xy, \tag{5.1}$$

$$v = -\vartheta_y + \vartheta_y ax + (\vartheta_y b + \vartheta_z)y - b\vartheta_z y^2 - a\vartheta_z xy, \tag{5.2}$$

where u and v are the optical flow vectors in x and y image coordinates system, respectively (see Fig. 5.2). $\vartheta_x = V_x/h$, $\vartheta_y = V_y/h$, and $\vartheta_z = V_z/h$ are the corresponding velocities in X, Y, and Z directions scaled with respect to the height h. Slope angles of the surface, α and β are the arctangent of a and b, respectively in Eqs. (5.1), (5.2). In this work, we compute optical flow using the sparse corner detection method with FAST [69, 70] and Lucas-Kanade tracker [71] to reduce computation for on-board processing. Note that since this proposed concept does not constrain the way to compute optical flow, other methods computing optical flow, e.g. dense optical flow can also be used.

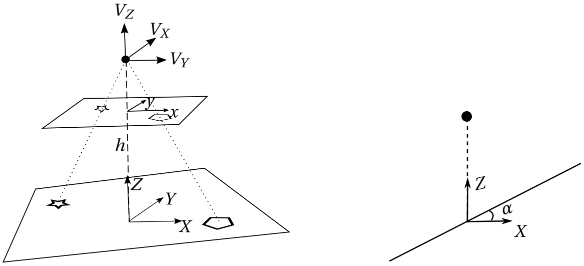


Figure 5.2: Left: A pin hole model. Right: An inclined ground surface with slope α.

By re-writing Eqs. (5.1) and (5.2) into matrix form as shown below, the parameter vectors $\mathbf{p_u} = [p_{u1}, p_{u2}, p_{u3}, p_{u4}, p_{u5}]$ and $\mathbf{p_v} = [p_{v1}, p_{v2}, p_{v3}, p_{v4}, p_{v5}]$ can be estimated using a maximal-likelihood linear least-squares estimate within a robust random sample consensus (RANSAC) estimation procedure [86]:

$$u = \mathbf{p_u}[1, x, y, x^2, xy]^T, \tag{5.3}$$

$$v = \mathbf{p_v}[1, x, y, y^2, xy]^T. \tag{5.4}$$

These fits using RANSAC returns the number of inliers and the fitting error. If there are obstacles sticking out of the landing surface, their optical flow vectors will not fit with the second assumption of a planar landing surface. This leads to a higher fitting error, ϵ^* which can thus be interpreted as a measure of average *surface roughness* of the entire area in the field of view in Eq. (5.5):

$$\epsilon^* = \epsilon_u + \epsilon_v, \tag{5.5}$$

with ϵ_u and ϵ_v sum of absolute errors of the RANSAC estimation in Eqs. (5.3) and (5.4) divided by the number of tracked corners. Thus, we can use ϵ^* to detect obstacles near the ground surface by fitting the optical flow field. Note that Eqs. (5.1) and (5.2) can be simplified by neglecting the second-order terms if the MAV only moves laterally. Therefore, a linear fit of the optical flow field can be used.

5.3.2. APPEARANCE FROM TEXTON DISTRIBUTION

In this study the visual appearance is described using the *texton* method [106], based on the extraction of small image patches (see Fig.5.3). The advantage of using texton method is that it not only encodes the color distribution of an image, but also their textures. This improves the accuracy of the appearance representation while maintaining its efficiency for real-time application.

With this method, first a *dictionary* is created consisting of *textons*, i.e., the cluster centroids of small image patches. For our implementation, we follow our previous work in [107], and learn the dictionary with Kohonen clustering [108]. After creation of the dictionary, image appearance can be represented as a texton distribution. To this end, a number of image patches are randomly extracted from an image and per patch the closest texton can be added to a corresponding bin in a histogram. By normalizing it with the number of patches, a maximum likelihood estimate of the texton probability distribution **q** is obtained. The texton method showed competitive results on texture classification tasks [106] with respect to computationally much more complex methods such as Gabor filter banks. In previous research, the texton method has been used to calculate the appearance variation cue [109] and to learn how to recognize heights and obstacles [107] but it was never applied to SSL.

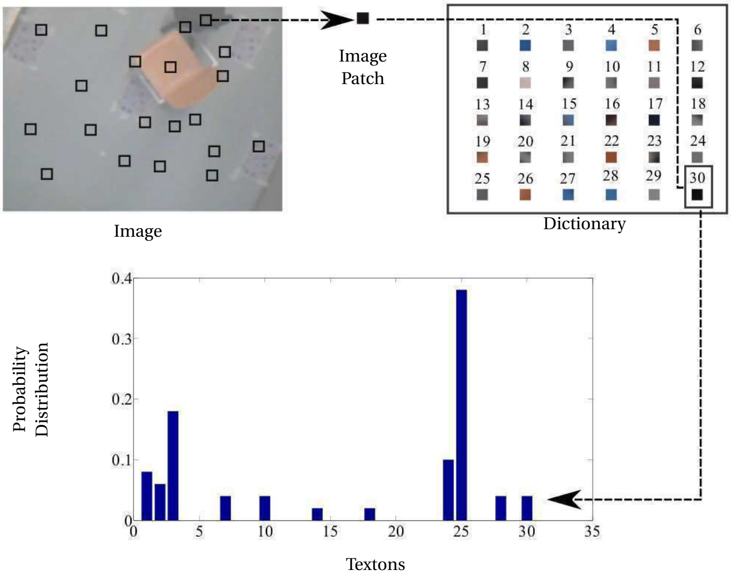


Figure 5.3: Texton Method: A number of image patches is randomly selected from an image. The patches are then compared to textons in a dictionary to form a texton probability distribution of the image.

5.3.3. REGRESSION LEARNING OF SURFACE ROUGHNESS

In order to learn the visual appearance of obstacles on the landing surface, a regression model is learned that maps a texton distribution to a surface roughness value. This function is learned by using the optical flow based roughness estimate ϵ^* as the regressand. In this subsection, we choose one of the regression methods for SSL based on the preliminary test results.

To show feasibility and reliability of the relationship, various regression methods (such as Linear, Ridge, LASSO, Kernel smoother, Pseudo-inverse, Partial least squares, k-nearest neighbor regressions) were tried out to perform the learning using *prtools* [110] in MATLAB. Preliminary tests have shown little difference between the learning methods. For instance, the normalized Root Mean Square Errors (NRMSE) computed using Eq. (5.6) on the test sets are $\sim 10\%$ for the learning methods.

$$NRMSE = \frac{1}{\epsilon^*_{max} - \epsilon^*_{min}} \sqrt{\frac{\sum_{t=1}^{n} (\hat{\epsilon}_t - \epsilon^*_t)^2}{n}}, \tag{5.6}$$

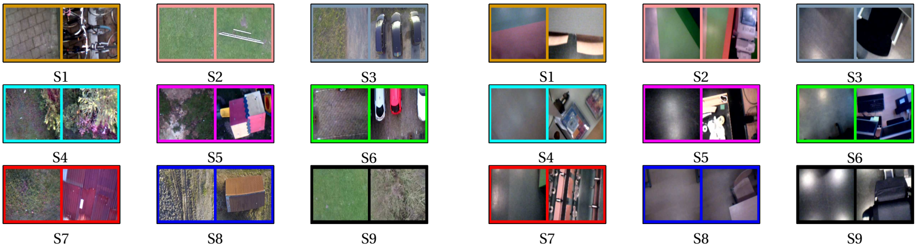
Since they all give reasonably good results, the linear regression method is used for this study, due to its simplicity and computational efficiency. After obtaining the texton distributions $\mathbf{q}$ of m number of visual words and the roughness ϵ^* for n images, a linear regression model expressed in Eq. (5.7) can be trained.

$$f(q_i) = \rho_1 q_{i1} + ... + \rho_m q_{im} + \Lambda, \quad i = 1, ..., n \tag{5.7}$$

where Λ is a bias and ρ are the regression coefficients, which are optimized so that $f(\mathbf{q}) \approx \epsilon^*$.

5.4. OBSTACLE DETECTION

In this section, we test how well the methods in Section 5.3 work to detect obstacles. To this end, we made multiple image sets of onboard MAV images from a downward-looking camera. Outdoors MAV flew at $\approx$ 15 meters high in 9 different environments, ranging from a car parking lot to a park. Indoors MAV flew at $\approx$ 2 meters high in 9 different indoor environments, ranging from the canteen to office spaces. Both the indoor and outdoor datasets show considerable variation in appearance of the landing surface and obstacles, and hence represent a significant challenge for machine learning methods. Fig. 5.4 shows the onboard images without (left) and with (right) obstacle of each environment.



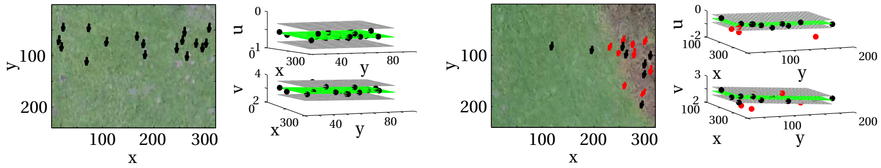
(a) Outdoor datasets

(b) Indoor datasets

Figure 5.4: Image datasets collected from outdoor and indoor environments (Scenes S1-S9). For each environment, an image is shown without obstacle (left) and with obstacle (right).

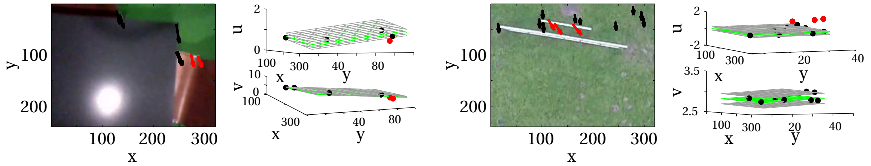
5.4.1. OBSTACLE DETECTION WITH SURFACE ROUGHNESS

To show that the proposed surface roughness, ϵ^* can be used as a metric to detect obstacles, images from the datasets were analyzed. Fig. 5.5 presents optical flow vectors measured from the consecutive images, and their fits of the optical flow field. In this figure, optical flow vectors are indicated by arrows in the image while the fit planes are presented in green and the error thresholds are shown as the gray planes. The red arrows in the images and red dots in the fits indicate which optical flow vectors contribute to larger error of the fits, the so-called outliers. When there is no obstacle or 3D structure in the images, the measured optical flow vectors are uniform, and thus most of them lie on the fitted plane within the bound of error threshold. This can be seen from the fits of the optical flow on the grass field in Fig. 5.5a. In contrast, irregular optical flow vectors can be expected from the images containing obstacles. This leads to more outliers or larger errors of the fits as shown in Fig. 5.5b (trees on the right), Fig. 5.5c (chair below the table), and Fig. 5.5d (fence on the top).



(a) Non-obstacle (grass field)

(b) Obstacle (trees)



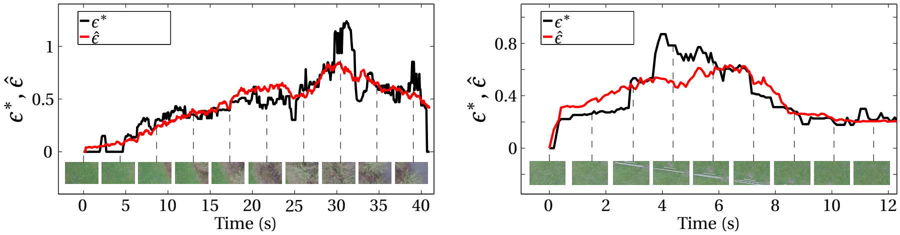
(c) Obstacle (table and chair)

(d) Obstacle (fence)

Figure 5.5: Outdoor and indoor images without and with obstacles, and their corresponding optical flow fits. Non-obstacle: Consistent optical flow vectors (black) contribute to low fitting errors. Obstacle: Irregular optical flow vectors (red) lead to high fitting errors.

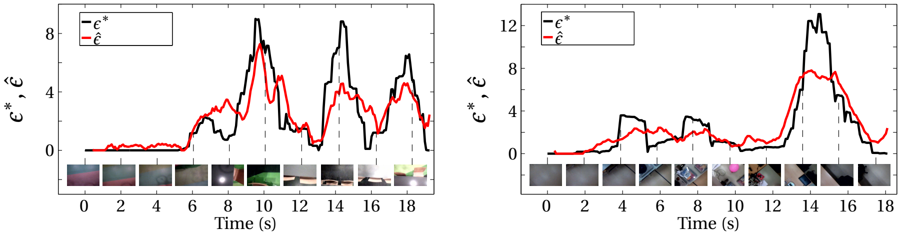
In addition, we analyzed for all 9 scenes in both outdoor and indoor environments datasets the surface roughness, ϵ^* estimated from the optical flow algorithm (black line) when the MAV flew over areas without and with obstacles. Fig. 5.6 presents the results of 2 scenes out of 9 for outdoor (S9 and S2) and indoor (S1 and S4) environments. This figure clearly shows that the roughness value is higher when there is an obstacle than when there is no obstacle on the landing surface. Note that we can see from this figure that the scale of the surface roughness varies with heights in flights (outdoor $\approx 15\ m$ and indoor $\approx 2\ m$), and this observation is discussed in Section 5.7.

To evaluate the classification capability using ϵ^*, we labeled the test set using a threshold of $\epsilon^*_{th} = 0.3$ and compared to manually labeled results using images. The true positive (TP) rate (the portion of obstacles that are detected) and false positive (FP) rate (the portion of empty landing areas wrongly classified as containing an obstacle) for all



(a) Outdoor scene I (S9)

(b) Outdoor scene II (S2)



(c) Indoor scene I (S1)

(d) Indoor scene II (S4)

Figure 5.6: Obstacle detection using roughness estimates ϵ^* from optical flow algorithm and $\hat{\epsilon}$ from appearance while navigating.

scenes shown in Fig. 5.6 are computed and shown in Table 5.1. In this table, most of the classification results show high proportions of correct identification of obstacles and non-obstacles images. However, the fence shown in Fig. 5.6b can sometimes be missed by the feature point tracker set to only track 25 points. The undetected obstacles mostly appear at the border of the images as shown in Fig. 5.7. For example, we can see that from top left to bottom right images in this figure, the undetected obstacles appear either on the top right (trees), top (fence), top (chairs), or left (chair). The result shows that despite the sparseness of feature-based optical flow tracking, this algorithm generally manages to detect the obstacles in the field of view and thus identify safe landing spots. While results can be improved with more computational power, this work will show the applicability of the method even in the case of severe computational limitations.

Table 5.1: Performance measure of a classification test using surface roughness, ϵ^*

Metrics	Fig. 5.6a	Fig. 5.6b	Fig. 5.6c	Fig. 5.6d
TP rate	0.99	0.95	0.96	0.91
FP rate	0.12	0.24	0.01	0.00

5.4.2. OBSTACLE DETECTION WITH VISUAL APPEARANCE

In this subsection, training and testing is done in the same scene, where training is done on the first 80% of a dataset and testing on the remaining 20% of the data set. First, a

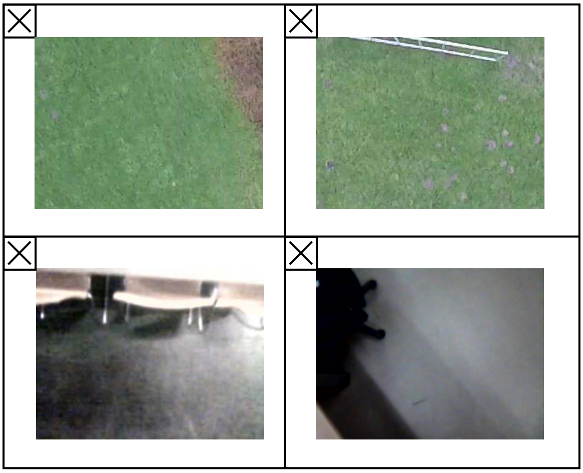
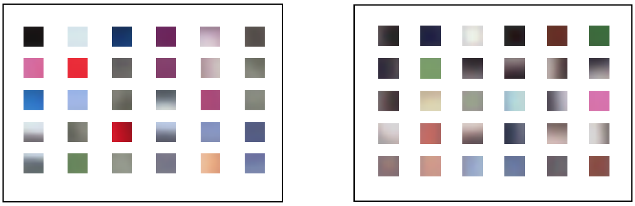


Figure 5.7: Undetected obstacles with surface roughness, ϵ^*.

dictionary was trained, and an example dictionary for scene S5 in each indoor and outdoor dataset is shown in Fig. 5.8. As can be seen from this figure, some textons represent a specific type of color while other textons represent a specific type of texture, such as horizontal or vertical gradients.



(a) Outdoor dictionary

(b) Indoor dictionary

Figure 5.8: Trained dictionaries used in texton method for scene S5 in outdoor and indoor datasets.

Then, a linear regression model was trained to map the texton distribution which represents the appearance of the image to the roughness ϵ^* estimated with the optical flow algorithm. Table 5.2 presents the performance metrics of the learning and classification shown in Fig. 5.6. The NRMSE show that it is more challenging to learn the regression model in more complex indoor environments where the appearance variation is larger. Concerning the TP rates, overall results are comparable or slightly better than the classification using ϵ^*. However, for indoor environments, the FP rates are higher due to complexity of the indoor scenes. Fig. 5.9 shows some undetected obstacles with visual appearance, $\hat{\epsilon}$. Although the TP rates are slightly higher than the TP rates of optical flow approach, obstacles appearing at the border of the image can sometimes be missed. The bottom right image is interesting, since the border of the table is the missed "obstacle". The objects on the table (like the closed laptop) are actually too flat to form a real obstacle to landing. Next, we are going to show that the learned model is able to detect obstacles when there is a movement (while translating) and no movement (while hovering), and also the results of pixel-wise obstacle segmentation.

Table 5.2: Normalized Root Mean Square Errors (NRMSE) on test sets

Metrics	Fig. 5.6a	Fig. 5.6b	Fig. 5.6c	Fig. 5.6d
NRMSE (%)	10.8	12.7	20.7	19.1
TP rate	0.96	0.96	0.97	0.96
FP rate	0.00	0.20	0.51	0.18

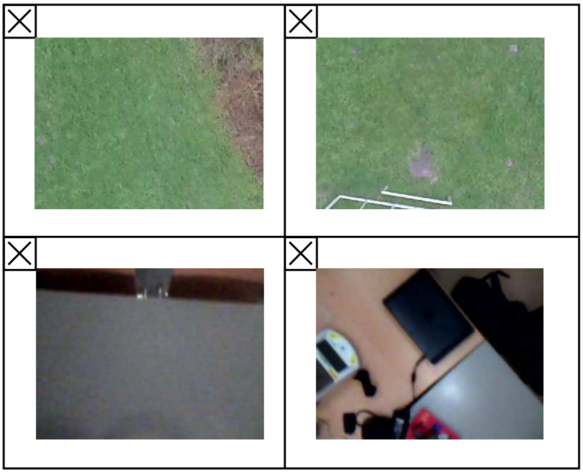


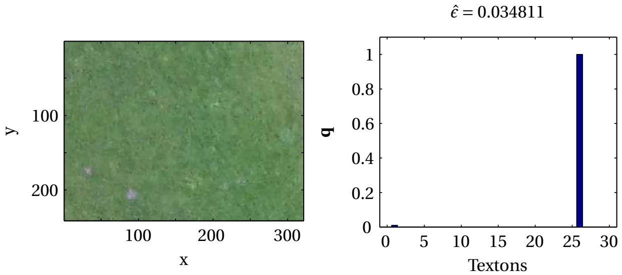
Figure 5.9: Undetected obstacles with visual appearance, $\hat{\epsilon}$.

WHILE TRANSLATING

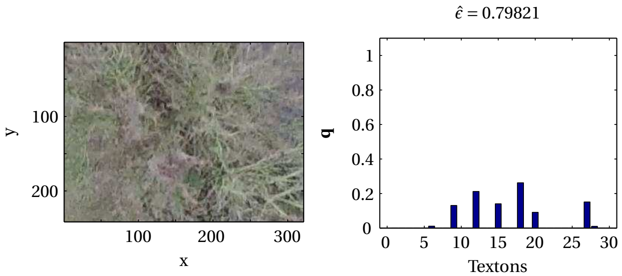
To compare $\hat{\epsilon}$ with ϵ^* from optical flow algorithm, we also plotted $\hat{\epsilon}$ (red line) on Fig. 5.6. This figure clearly shows that both roughness estimates are higher when there is an obstacle than when there is no obstacle on the landing surface. Thus, the results demonstrate that the MAV can detect obstacles using $\hat{\epsilon}$ and thus identify safe spots to land upon.

WHILE HOVERING

Having to move close to obstacles in order to detect them takes time and represents a risk. It would be more efficient and safer to detect obstacles without needing to move. Fig. 5.10 and Fig. 5.11 show texton distributions for obstacles and non-obstacle images and their corresponding $\hat{\epsilon}$ for outdoor and indoor scenes, respectively. From these figures, we can observe that the texton distribution is rather different for images with and without obstacles. After learning this results in different surface roughness values $\hat{\epsilon}$.

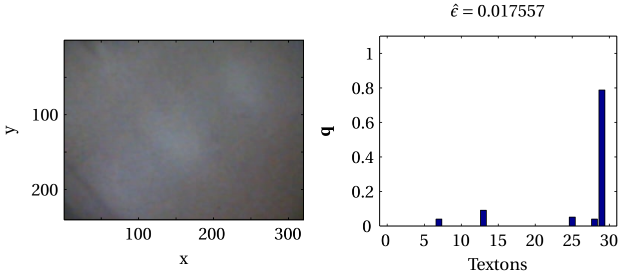


(a) Grass field

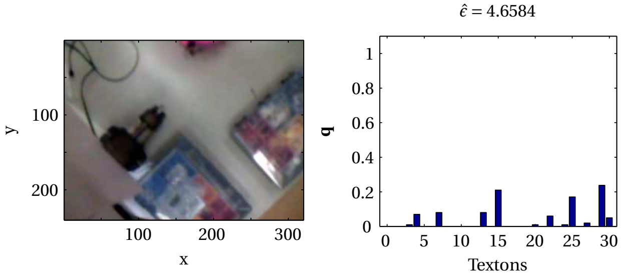


(b) Trees

Figure 5.10: Outdoor scene: Obstacle detection using roughness estimates $\hat{\epsilon}$ from appearance while hovering.



(a) Floor



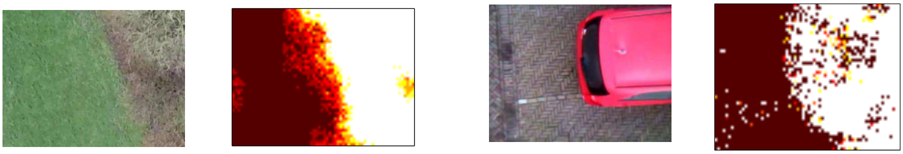
(b) Table

Figure 5.11: Indoor scene: Obstacle detection using roughness estimates $\hat{\epsilon}$ from appearance while hovering.

PIXEL-WISE OBSTACLE SEGMENTATION

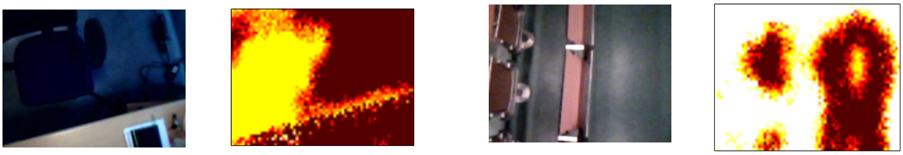
The roughness value resulting from the optical flow algorithm is a *global* value for the presence of obstacle in the entire image. This study shows, after SSL, that the MAV will not only be able to detect the presence of obstacles, but will even be capable of pixel-wise segmentation of obstacles. The basis for this capability is the *local* nature of the image patches involved in the construction of the texton distribution.

To show this, a sub-image with a window size of 50×50 pixels of an image was moved across $x-$axis of the image for each line in $y-$axis with increment of 4 pixels until it covered the whole image. For each sub-image, the texton distribution was formed using 50 image patches and mapped to a roughness value with the regression function discussed above. This creates a new image containing the roughness estimate for each area of the scene. Fig. 5.12 shows two still images from outdoor and indoor scenes and their corresponding regression maps which are color-coded using roughness values. In these figures, the obstacles clearly have a higher value (marked with light yellow color) in the regression maps while the safe landing area have lower value of roughness (marked with dark red color) in the map. In Outdoor Scene 1, it can be seen that the trees in the image on the right are seen as obstacle and marked as yellow. The same holds for the car in the image on the right in Outdoor Scene II are seen as an obstacle and marked as yellow. In Indoor Scene I, the black chair (top left in the image) and the border of the table (bottom right in the image) are detected as obstacles. The table itself is not seen as an obstacle: When having only the table in view, it is considered sufficiently flat and hereby offers an landing place for the MAV. The retractable tables and chairs (center and left in the image) in Indoor Scene II are also detected as obstacles while the floor is found as a safe place for landing. Note that this method with a moving window is used to show that our approach can also segment the obstacles in an image. However, it is computationally expensive since it processes almost every pixel in the image, and we actually do not need all the detailed information unless we need to land on a narrow place.



(a) Outdoor Scene I (S9)

(b) Outdoor Scene II (S6)



(c) Indoor Scene I (S6)

(d) Indoor Scene II (S7)

Figure 5.12: Obstacle localization using roughness $\hat{\varepsilon}$ from SSL method. The light yellow color in roughness map represents the presence of obstacles.

5.5. GENERALIZATION

5.5.1. GENERALIZATION OF SSL METHOD TO DIFFERENT ENVIRONMENTS

There is a main question remaining for the proposed method, i.e., how well the learned mapping from visual appearance to roughness will generalize to different environments. As in any learning scheme, this depends on the training and test distribution and on the learning method. In order to be successful, the training and test distribution should be sufficiently similar. In computer vision, this similarity does not only depend on the environment, but also on the visual features extracted from the images and how invariant they are to for instance rotation, scaling, and lighting changes.

In the context of SSL of obstacle appearance, we expect that features and learning methods can be found that generalize well over different environments and conditions. For instance, when we humans look at a Google maps image, we can discern obstacles such as trees and buildings rather well from areas that are more suitable for landing such as grass fields. Such a classification performance is also within reach of computer vision methods [111]. Of course, the computationally efficient texton distributions and straightforward learning methods used for onboard implementation on small MAVs are quite limited. However, even a limited generalization to a visually very different environment does not have to pose a problem in SSL method. Two strategies are available to deal with this: (1) continuously learn the mapping when the MAV is moving enough with respect to the visual scene, and (2) detecting when the learned mapping is receiving different inputs and hence producing uncertain outputs. If the estimated outputs are uncertain, the MAV can rely again on optical flow and adapt its mapping to the new environment. In this section, we show that the uncertainty of outputs in a visually different environment can be evaluated with a Naive Bayes classifier and Shannon entropy.

NAIVE BAYES CLASSIFIER

Given a distribution of n textons ($\mathbf{q} = (q_1, \ldots, q_n)$) to be classified, the Naive Bayes classifier for k possible classes is given as in Eq. (5.8). In this study, we have two classes ($k = 2$), i.e., presence and absence of an obstacle, each of which can be represented by a distribution of $n = 30$.

$$p(C_k|q_1, \ldots, q_n) \propto p(C_k) \prod_{i=1}^{n} p(q_i|C_k) \tag{5.8}$$

To create a Naive Bayes classifier, we first assign a roughness threshold, $\hat{\epsilon}_{th}$ (= 0.3) to classify the distributions based on its corresponding roughness estimate, $\hat{\epsilon}$ into two classes labeled C_1 for obstacle or C_2 for non-obstacle according to Eq. (5.9). Based on this dataset, learning of the Naive Bayes classifier was performed using *prtools* [110] in MATLAB.

$$C_k = \begin{cases} C_1 & \text{if } \hat{\epsilon} > \hat{\epsilon}_{th} \\ C_2 & \text{if } \hat{\epsilon} < \hat{\epsilon}_{th} \end{cases} \tag{5.9}$$

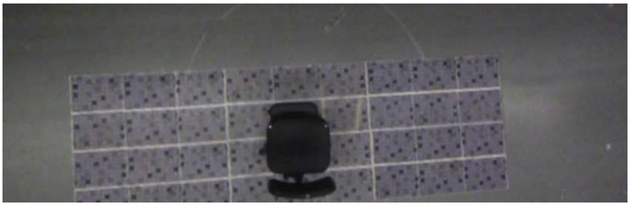
SHANNON ENTROPY

In information theory, Shannon entropy can be used to provide the amount of 'disorder'or uncertainty of a system [112]. In this study, the entropy, H can also be implemented to detect the change of the environment online based on the outputs of Naive Bayes classifier, as expressed in Eq. (5.10).

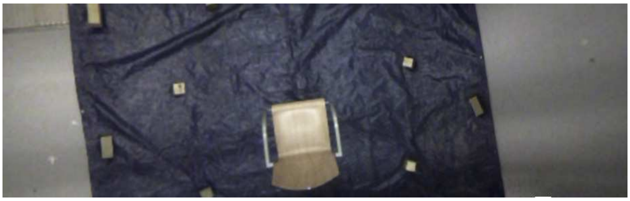
$$H = \sum_{k=1}^{2} p(C_k|\mathbf{q}) \log_2(p(C_k|\mathbf{q})) \tag{5.10}$$

ANALYSIS ON TWO DIFFERENT ENVIRONMENTS

The MAV was flown in two visually different environments (E1 and E2) in which one different obstacle was placed as shown in Fig. 5.13. A linear regression model was trained in E1 and then the texton distributions were logged for E1 and E2 by repeatedly flying the MAV over the obstacles. Here, we investigate how well the regression model trained in E1 performs in E2. Please note that despite the use of a similar object (a chair), the environments are visually very different (the chair being dark in E1 and bright in E2, the surface being grey in E1 and dark blue in E2).



(a) Environment 1 (E1)



(b) Environment 2 (E2)

Figure 5.13: Images of two different environments stitched using on-board images.

80% of the distributions in E1 were used to train the Naive Bayes classifier and the rest of the distributions were used for testing purposes. Both test sets from E1 and E2 were tested on the Naive Bayes classifier and the classification error, Err is computed using $Err = (FP + FN)/(n_{test})$, where FP and FN are the number of false positive and false negative, respectively from a total number of the test data, n_{test}. The errors on test sets for E1 and E2 are 4.67% and 11.37%, respectively. This error evaluates the dataset by the classifier without considering the class prior. The error is higher for the test set in E2, thus indicating that the generalization to E2 is indeed more difficult than the generalization to E1's test set.

In the top part of Fig. 5.14 and Fig. 5.15, the red line is the roughness estimate and square boxes show ground truth position of the obstacle where black areas indicate full visibility of the obstacle in the field of view of the camera while half visibility is shown using gray areas. In the bottom part of these figures, the blue line shows the uncertainty of the outputs from the Naive Bayes classifier with the Shannon entropy. In Fig. 5.14, the uncertainty can be observed at the edges of the obstacles. It is reasonable as only a very small part of the obstacle was captured in the image and it can actually give a good checking for the trained environment itself before making the decision. In Fig. 5.15, the results of roughness estimate demonstrate that the SSL model actually remains quite effective even when flying in a different environment. Although these results show that the generalization is achieved to some extent, there are some regions which were wrongly classified (e.g. $15-20s$ and $35-40s$). In fact, the entropy gives a more continuous uncertainty of the outputs due to the difference in visual appearance in E2. There is one part (e.g. $25-30s$) where they both agree with their outputs because the field of view of the camera consists of largely the same gray ground on the right side of test field (see Fig. 5.13). By using this information, the MAV is able to detect the change of environment and trigger the optical flow algorithm to re-train the linear regression model so that it can adapt itself to the new environment.

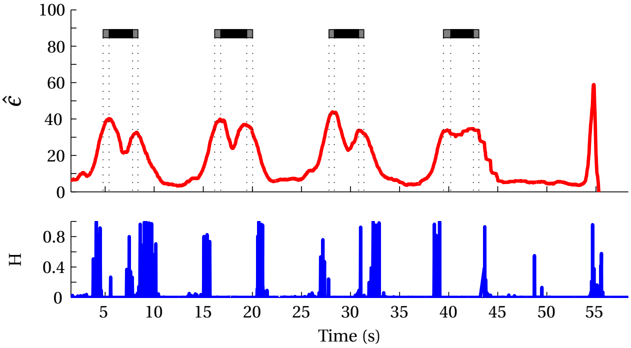


Figure 5.14: Uncertainty measure H (bottom) and roughness estimate $\hat{\varepsilon}$ (top) in E1.

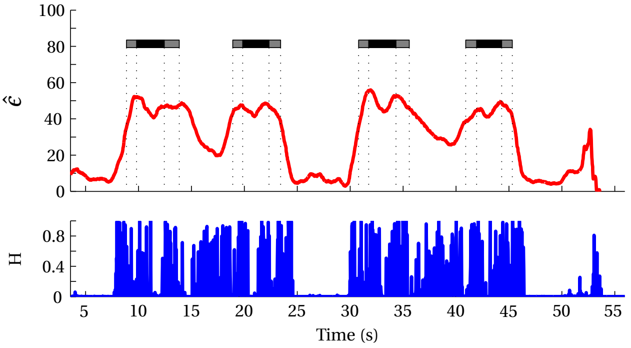


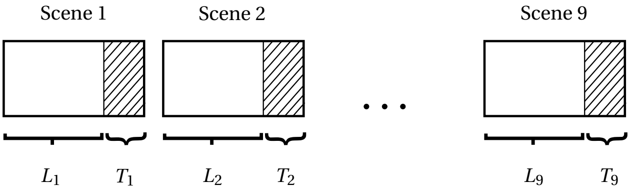
Figure 5.15: Uncertainty measure H (bottom) and roughness estimate $\hat{\varepsilon}$ (top) in E2.

K-FOLD TEST ON MULTIPLE REAL SCENES

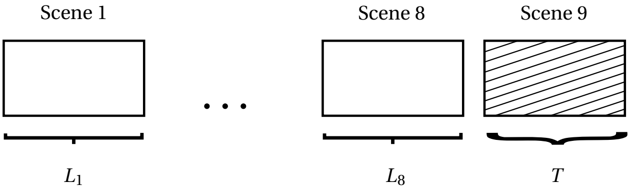
In order to evaluate the performance of our classification models in an unknown scene or for an untrained dataset, we perform two K-fold cross-validations on multiple real scenes in both outdoor and indoor environments presented in Fig. 5.4. The border color of each scene in the figure was used in plotting the results to distinguish different scenes. Fig. 5.16 shows the K-fold tests where L indicates the learning sets and T represents the test sets.

In the first K-fold validation, 8/9 ratio of samples from each scene were randomly selected and used for training $(L_1 + L_2 + \ldots + L_9)$ while the remaining 1/9 ratio of samples were tested $(T_1, T_2, \ldots, T_9)$. We iterated this process 9 times so that all data has been used as test set. We performed this test to evaluate the learning capability of the proposed SSL setup in which multiple scenes with different variation of appearance are included in the training set.

In the second K-fold validation, we tested the exploration performance of our SSL setup. This can be done by using datasets from 8 out of 9 scenes for training $(L_1 + L_2 + \ldots + L_8)$ and testing on the remaining dataset (T_9). The remaining dataset can be completely different or slightly similar to the training set. We iterated the process until all scenes were tested. Therefore, in total we performed 9 tests for each K-fold cross-validation in both outdoor and indoor environment and surface roughness ϵ^* was used as the ground truth in the validation.



(a) K-fold cross-validation 1

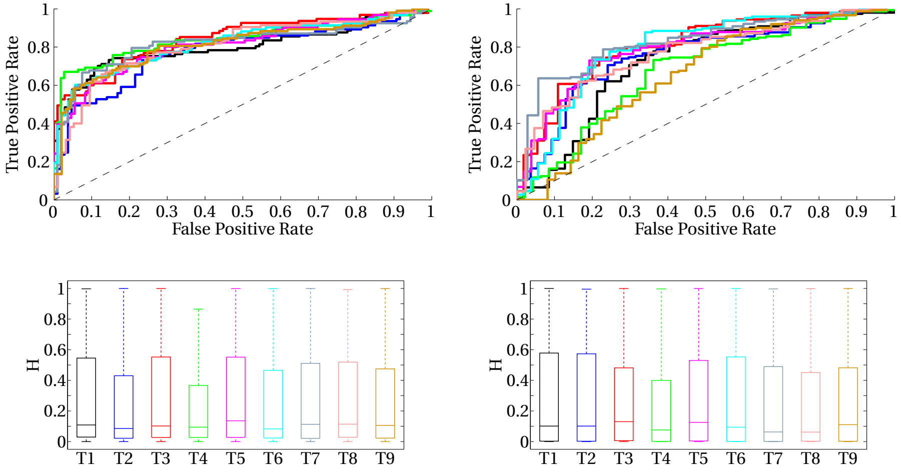


(b) K-fold cross-validation 2

Figure 5.16: Two K-fold cross-validations for evaluation of learning and exploration capabilities of the classifier.

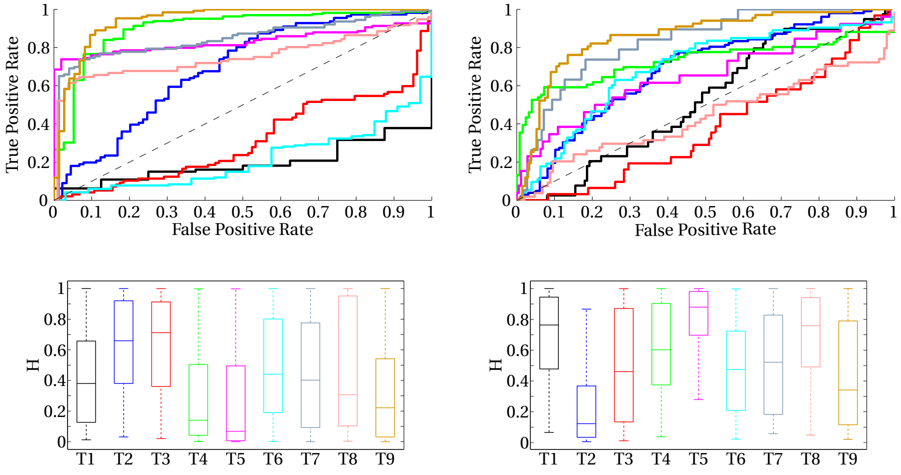
To illustrate the performance of the classifier, we plot a receiver operating characteristic (ROC) curve for each K-fold test in Fig. 5.17 and Fig. 5.18. In addition, the uncertainty measure from Shannon entropy for the corresponding test is also presented. Table 5.3 shows the Area Under ROC Curve (AUC) and NRMSE for each test which is a common metric used to evaluate the ROC curves and regression tests, respectively.

For landing, we would like to have a rather high true positive rate (≈ 0.9) in order not to miss obstacles, while a rather high false positive rate may be acceptable - this will just reduce the available landing locations but will not endanger the MAVs. In Fig. 5.17, we can observe that to achieve a true positive rate of ≈ 0.9, we need to compromise with ≈ 0.5 false positive rate. Two examples in the second validation of untrained outdoor and indoor test sets with a chosen threshold result in TP rate of ≈ 0.9 and FP rate of ≈ 0.5 are



(a) Outdoor scenes. (b) Indoor scenes.

Figure 5.17: ROC curves and uncertainty measures for K-fold cross-validations 1. The results are given the color of the test set.



(a) Outdoor scenes. (b) Indoor scenes.

Figure 5.18: ROC curves and uncertainty measures for K-fold cross-validations 2. The results are given the color of the test scene (see Fig. 5.4).

Table 5.3: Performance measure of ROC curves using Area Under the Curve (AUC - higher is better) and NRMSE (% - lower is better)

K-fold	Environment	Metrics	T1	T2	T3	T4	T5	T6	T7	T8	T9
1	Outdoor	AUC	0.79	0.77	0.83	0.84	0.80	0.81	0.81	0.80	0.80
		NRMSE	15.4	17.5	16.4	16.1	16.2	18.9	14.0	14.4	12.9
	Indoor	AUC	0.72	0.75	0.81	0.67	0.77	0.78	0.80	0.75	0.64
		NRMSE	13.0	13.6	17.7	14.8	12.8	15.2	15.9	13.9	13.4
2	Outdoor	AUC	0.23	0.68	0.31	0.90	0.83	0.22	0.85	0.75	0.93
		NRMSE	84.2	87.2	43.8	22.9	33.2	27.0	21.8	52.9	18.1
	Indoor	AUC	0.52	0.68	0.37	0.73	0.65	0.69	0.82	0.44	0.85
		NRMSE	30.0	29.1	34.8	22.2	83.3	24.1	28.9	72.1	27.6

shown in Fig. 5.19 and 5.20, respectively. In these figures, l_1 and l_2 indicate the time steps where the presence of obstacles are predicted using ϵ^* and $\hat{\epsilon}$, and H is the uncertainty measure of $\hat{\epsilon}$. The results in these figures show that the obstacles can be detected, but many of the places considered safe by the optical flow algorithm are being classified by $\hat{\epsilon}$ as containing obstacles. In Fig. 5.19, we can observe that the false positive cases happen mostly at at the beginning ($\approx 0s-3s$) and also at images in which obstacles appear partly in the images ($\approx 4s-6s$). If we look at the uncertainty measure, it shows an uncertain moment at the beginning and at the end of the image sequence. Similarly, in Fig. 5.20, false positive happens at the beginning with high uncertainty measure. If aim for a low $\hat{\epsilon}$ estimate and H, we can still find a suitable landing place though.

The average AUC for outdoor and indoor scenes in the first validation are 0.81 and 0.74, respectively. These results are rather good and indicate that the SSL method does not have difficulty of learning multiple scenes together, and it can predict scene with obstacles quite accurately if similar or part of the scene was learned. In Fig. 5.17b, it can be seen that the uncertainty measures are relatively small (≈ 0.1) indicating that the test environment is not too different from the training environment. Comparing outdoor and indoor environments, we can observe that learning of the indoor scenes is more difficult. This is due to the fact that indoor scenes are typically more complex than outdoor scenes. For instance, in an indoor scene, we can find many man-made objects and floors which look very different in terms of colors and textures, and thus the representation of these visual appearances can be more difficult.

In the second validation, for completely unknown scenes, the performance of the classifier becomes worse. If we compare the results of NRMSE in Table 5.3, we can observe that test errors are mostly larger in validation 2. Although some ROC curves are still acceptable (T4, T5, T7, and T9 outdoors for example), there are also ROC curves that are even poorer than random (e.g. T1, T3, T6 in outdoor scenes, and T1, T3, T8 in indoor scenes). However, the main observation is that the uncertainty measures for this K-fold test are much higher than for the first one, especially for those which have ROC curves worse than random. This indication allows the MAV to know when and where to re-train the regression model to adapt to new changes in its surroundings.

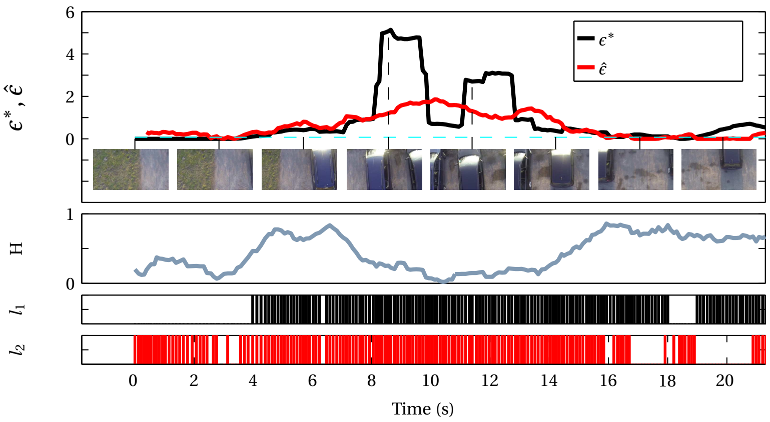


Figure 5.19: Classification of obstacles with ϵ^* (l_1) and $\hat{\epsilon}$ (l_2) in an untrained outdoor scene (S3), and the corresponding uncertainty measure, H. This example with a particular threshold of $\hat{\epsilon}_{th} = 0.076$ is chosen in the K-fold validation 2 which resulted in TP rate of 0.9 and FP rate of 0.5.

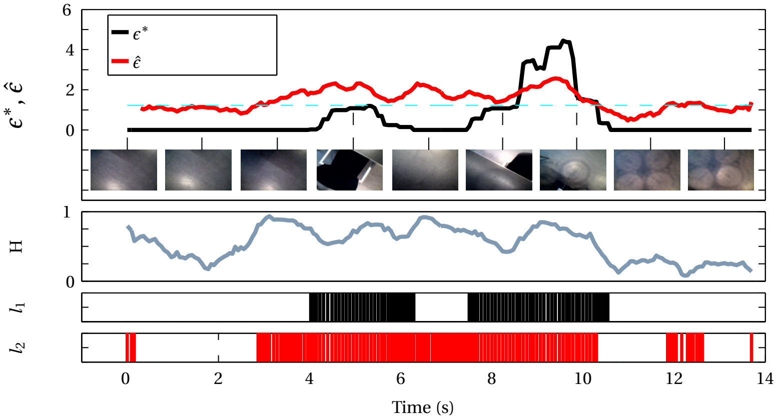


Figure 5.20: Classification of obstacles with ϵ^* (l_1) and $\hat{\epsilon}$ (l_2) in an untrained indoor scene (S3), and the corresponding uncertainty measure, H. This example with a particular threshold of $\hat{\epsilon}_{th} = 1.225$ is chosen in the K-fold validation 2 which resulted in TP rate of 0.9 and FP rate of 0.5.

5.6. FLIGHT TESTS

5.6.1. EXPERIMENT PLATFORM

A Parrot AR.Drone 2.0[1] and Bebop[2] are used as a testing platform for indoor and outdoor experiments, respectively in this study. They are equipped with a downward-looking camera which runs up to 60 *FPS* and is of particular interest to us for the landing purpose. Instead of using the original Parrot AR.Drone program, an open-source autopilot

---

[1] http://ardrone2.parrot.com

[2] http://www.parrot.com/products/bebop-drone/

software, Paparazzi Autopilot[3] is used because it allows us to have direct access to the sensors and control the MAV [72, 113]. We created a computer vision module in Paparazzi Autopilot to capture and process images from the camera and test our proposed algorithm in flight tests. Fig. 5.21 shows the overview of the control architecture of Paparazzi and how it integrates the vision module. All the computer processing tasks in this study are performed using the on-board processor of the MAV so that the MAV does not rely on the ground control station (GCS). This can avoid mission failure due to loss or delay of data transmission between MAV and GCS.

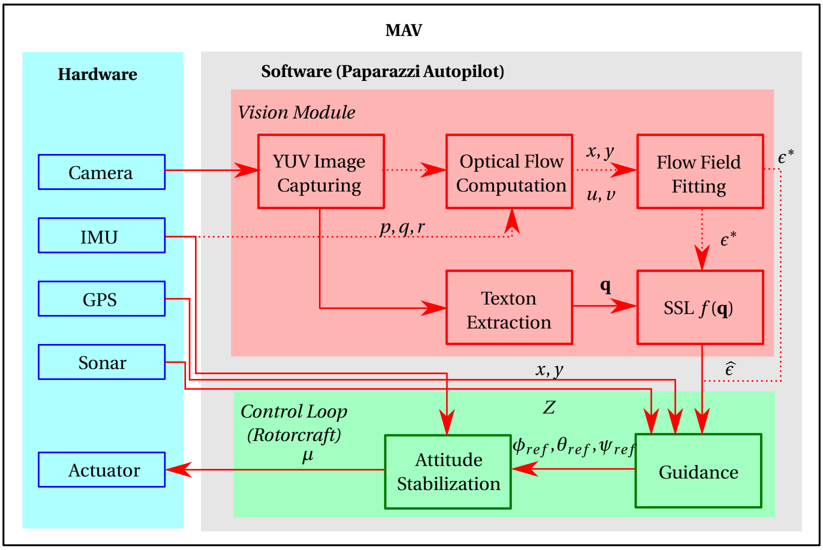


Figure 5.21: Integration of computer vision module in Paparazzi Autopilot. The left blue box shows the hardware of the MAV while the right gray box indicates the software architecture used in the MAV. The process flow guided by the dashed line can be neglected after the learning is complete.

In Fig. 5.21, images are captured from the downward-looking camera in the vision module. These images are processed using the computer vision algorithms (presented in Section 5.3), such as the optical flow and the texton methods, to detect obstacles and find a suitable landing spot. The IMU from the MAV is used in the optical flow algorithm to reduce the effect of MAV rotation on optical flow measurements. The output from the vision module (e.g. the position where it is considered safe to land upon) is fed into the control loop in Paparazzi autopilot to control the MAV. A GPS or motion tracking system provides position measurements of the MAV for outdoor or indoor navigation purposes. In this study, vision is purely used for the detection of obstacles on the landing surface.

5.6.2. PROCESSING TIME OF COMPUTER VISION ALGORITHMS

To examine the computational efficiency of the optical flow and SSL algorithms, we measured the times taken by each process of the algorithms. Table 5.4 shows the average

---

[3] `http://wiki.paparazziuav.org`

processing time required for each stage of both algorithms. In our experiments, the maximum number of corners in optical flow algorithm and the number of samples used in SSL are both set to 25. Note that this value can be tuned to include more or less information from the images, however, a higher value requires more computational time. The total processing times (see Table 5.4) show that both algorithms are computationally efficient and can be executed on-board the MAV. For instance, when both methods are running during data acquisition for learning, it costs roughly 35 ms which is approximately 28 FPS. The processing time is even faster after learning as the SSL method runs at the frame rate of $\approx 60\ Hz$. This frame rate is sufficient to capture surrounding information for fast moving MAVs.

Table 5.4: Average processing time for each stage of the vision algorithms

Optical Flow	Corner Detection	Corner Tracking	Flow Fitting	Total
Time (ms)	14.04	4.01	2.01	**20.06**

SSL	Distribution Extraction	Regression Function	Total
Time (ms)	15.13	0.01	**15.14**

5.6.3. AUTONOMOUS LANDING STRATEGY USING ϵ^* FROM OPTICAL FLOW

In this subsection, we first explain a landing strategy using ϵ^* to guide the MAV to land on a safe landing place. Then, we demonstrate the experiment results from the flight tests we performed with the landing strategy.

LANDING STRATEGY

Here, we propose a straightforward landing strategy which allows the MAV to decide where to land safely using ϵ^* from optical flow algorithm. Since this method requires movement of the MAV to detect obstacles, the MAV needs to fly over the potential landing area in order to search for a suitable landing spot/ waypoint. In this strategy, we define the safest landing spot as a waypoint in which it covers the largest area without obstacles underneath the MAV. To determine this waypoint, we first classify on-board images into safe ($SF = 1$) and unsafe ($SF = 0$) classes by thresholding ϵ^* with an empirically determined constant, ϵ_{th} as shown in Eq. (5.11).

$$SF_i = \begin{cases} 1 & \text{if } \epsilon_i^* < \epsilon_{th} \\ 0 & \text{if } \epsilon_i^* > \epsilon_{th} \end{cases} \tag{5.11}$$

Then, the largest area without obstacle can be found by choosing the largest value of $A_{SF} = \sum_{i=1}^{i=L} SF_i$ where $i = 1 : L$ are a set of continuous images which are classified as safe images. Once an obstacle is detected ($SF_{i=L+1} = 0$), A_{SF} is reset to 0 and $i = 1$. Furthermore, the middle waypoint of the safest area, P_{land} is considered as the desired landing spot and it is continuously updated when the largest A_{SF} is found.

EXPERIMENT RESULTS

The optical flow algorithm and landing strategy were implemented in a computer vision module in Paparazzi Autopilot and ran in real-time on a Parrot AR drone as described in Subsection 5.6.1. A motion tracking system, OptiTrack was used to serve as an indoor GPS to allow the MAV to fly autonomously in the arena according to a simple flight plan (see Fig. 5.22). The MAV took off from its HOME position, and followed a route starting from waypoint 1 and ending at waypoint 8. After scanning the whole landing site using its on-board camera, it navigated to a safe landing waypoint using the proposed landing strategy and landed there.

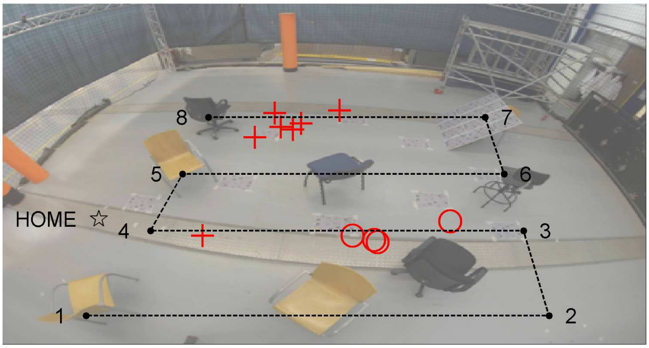


Figure 5.22: Flight plan and landing waypoints of the MAV based on roughness estimate ϵ^* from optical flow algorithm. Black dashed line represents the flight path following waypoints 1 to 8 and HOME is the position where the MAV take-off. Red markers indicate the landing spots P_{land} directed using the landing strategy.

We performed 11 landing tests with the Parrot AR drone. In each test, it first flew the entire flight plan and then returned to the point it considered safest. Fig. 5.23 shows the safety value *SF* along the flight path in one of the landing experiments. In this figure, the longest stretch A_{SF} is located between waypoints 7 and 8, and the landing spot P_{land} (indicated as black cross) is selected in this area. The landing spots for all the experiments are plotted with red markers in Fig. 5.22. All landings were successful and on flat/non-obstacle ground. Still, we have indicated with circle markers in the figure 4 landing spots that were rather close to obstacles in the environment. We expect that measurement noise is the cause of landing on these locations rather than the safer area higher up in the image.

5.6.4. AUTONOMOUS LANDING STRATEGY USING $\hat{\epsilon}$ FROM APPEARANCE

To utilize the advantage of SSL method, we switch to detecting obstacles using only the appearance while the MAV is hovering.

LANDING STRATEGY

In the previous subsections, we used the whole image to determine whether the region underneath the MAV was a safe landing place. However, it did not tell us where to go if

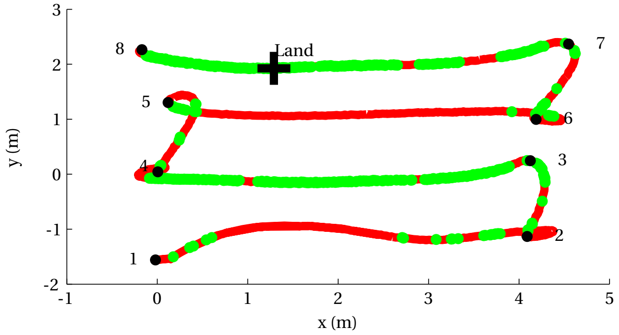


Figure 5.23: Safety value *SF* along the flight path in one of the experiments. Green markers indicate the value of *SF* equals to 1. Black cross shows where the landing spot is.

we found the obstacle(s). Therefore, we propose a straightforward method for allowing the MAV to decide where to land using $\hat{\epsilon}$. In this strategy, the image is divided into nine regions (grid of 3 × 3). Using the SSL method, we estimate $\hat{\epsilon}$ in each region and choose the region with minimum $\hat{\epsilon}$ as the place we are going to land or reject them if they are all higher than a threshold value (indicating the presence of an obstacle). To compute the movement, we take the distance d_c from the image center to the center of the region where it is considered safe and use the height h measured from the sonar sensor to project this distance in pixels to physical distance in meters d_p using Eq. (5.12).

$$d_p = \frac{d_c \times h}{\tilde{f}} \tag{5.12}$$

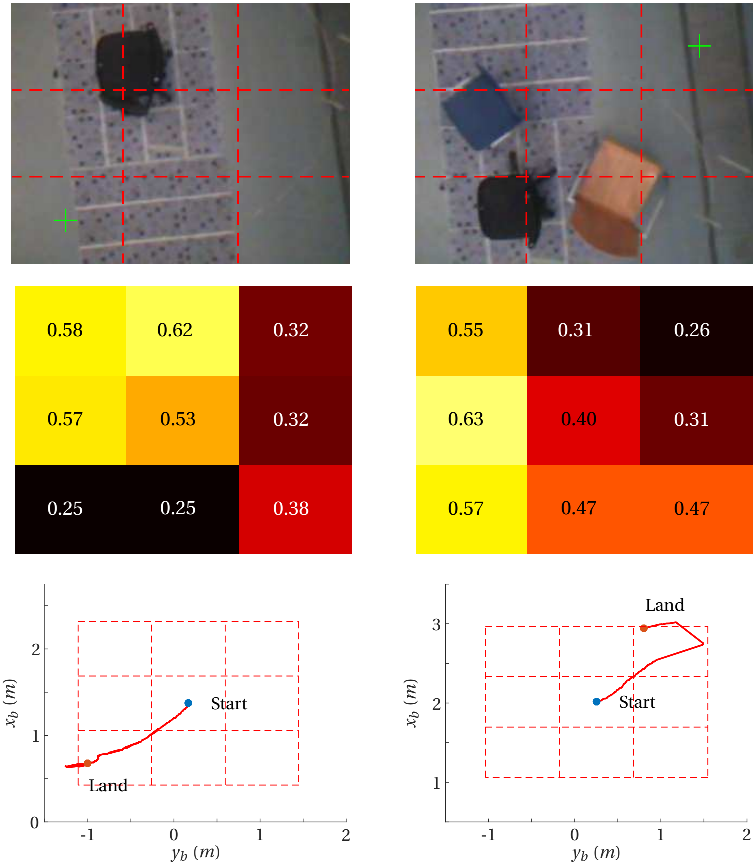
where $\tilde{f}$ is the focal length of the camera in pixels.

EXPERIMENT RESULTS

The landing experiments were conducted in indoor and outdoor environments at the heights above the ground of ≈ 3.5 m and ≈ 7 m, respectively. In the experiments, the MAV hovered at waypoints where one or more obstacles were underneath it. Once the landing strategy was activated, the MAV autonomously moved and landed at a place it considered safe. Fig. 5.24 shows the results of the flight tests in an indoor environment with the presence of single (first column) and multiple obstacles (second column). The top row of this figure presents the images taken when the landing strategy was activated. These images were divided into nine regions (separated by red lines) for $\hat{\epsilon}$ computation and the minimum $\hat{\epsilon}$ (indicated with a green cross) was chosen as the landing spot. The center row of the figure shows the $\hat{\epsilon}$ values for each region with colors representing the degree of visibility of the obstacle scaled using $\hat{\epsilon}$. The yellow color indicates the presence of an obstacle whereas the black color shows the absence of an obstacle. The bottom row of the figure illustrates the flight path of the MAV from the position where the landing strategy began to the position where the MAV landed. The results clearly demonstrate that the proposed strategy manages to lead the MAV to a safe landing place. [4]

---

[4] Experiment video: `https://goo.gl/Le5HT2`



0.58	0.62	0.32
0.57	0.53	0.32
0.25	0.25	0.38

0.55	0.31	0.26
0.63	0.40	0.31
0.57	0.47	0.47

Figure 5.24: Indoor Experiment: Cross markers in the top row images show the chosen landing spot. Center row presents $\hat{e}$ for all regions with color scaled according $\hat{e}$ (Yellow = obstacle; Black = non-obstacle). Bottom row presents the flight path of the MAV from the time when the landing strategy activated until the time when it landed.

5

Fig. 5.25 shows the results of the flight test in a windy outdoor environment (wind speed ≈ 11 knots). The obstacles are a camp (center bottom of the image) and trees surrounding it. In this figure, the top left shows the chosen landing spot according to the position of the minimum value of the 9 $\hat{e}$ presented in the bottom left using SSL landing strategy. The top right illustrates the flight path of the MAV while bottom right represents the mean value of the 9 $\hat{e}$ and the height of the MAV. Also, the instances when the landing strategy was activated and when the landing was performed are indicated as the black and blue vertical lines, respectively. Note that in this experiment, the training was not done in this scene but in another, similar outdoor environment consisting of trees and grass field. This illustrates that the learned appearance can allow MAVs to select a landing spot from hover even in a different environment than the trained one.

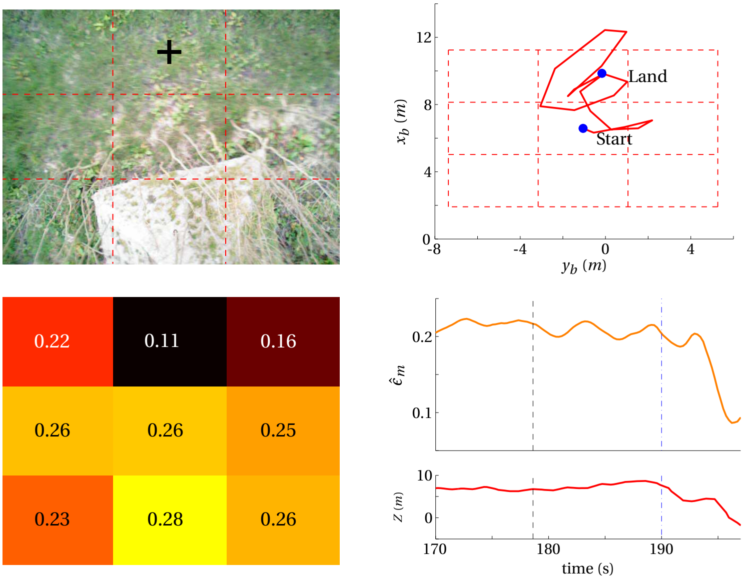


Figure 5.25: Outdoor Experiment: Cross marker in the top left images shows the chosen landing spot. Bottom left presents $\hat{\epsilon}$ for all regions with color scaled according $\hat{\epsilon}$ (Yellow = obstacle; Black = non-obstacle). Top right presents the flight path of the MAV from the time when the landing strategy activated until the time when it landed. Bottom right shows the average of the roughness in 9 regions, $\hat{\epsilon}_m$ and height, Z

5.7. Discussion

5.7.1. Influence of Height on Roughness ϵ^* and $\hat{\epsilon}$

One issue with the current implementation of the proposed SSL approach is that it does not take the height and velocity of the MAV into account. In fact, the optical flow measurements are dependent on the MAV's velocity and the distance to the features. For example, assuming the MAV is moving laterally at constant velocity, we measure smaller optical flow at larger heights than at lower heights. Similarly, concerning obstacle appearance, the size of the obstacle looks smaller when we observe it further away. As a consequence, something that seems like an obstacle at a low height may not be detected as obstacle at a large height. This effect can be seen in Fig. 5.24 and 5.25; at 3.5 m indoors, obstacles give a response of $\hat{\epsilon} > 0.3$, while at 7 m outdoors, obstacles give a response of $\hat{\epsilon} > 0.15$.

In order to investigate the influence of height on the surface roughness estimates, we conducted experiments flying the vehicle at various heights ($2m$, $3m$, and $4m$) and the same velocity ($\approx 0.6\ m/s$) in the same structured environment. Fig. 5.26 shows the effect of height on the results of the optical flow algorithm and SSL method. The roughness value at a lower altitude corresponds better to the presence or absence of obstacles than at a higher altitude, while flying at the same velocity. The results suggest that there is a height at which the optical flow algorithm and the appearance-based method can no longer detect the obstacle sizes used in the experiments. In an application scenario where the MAV will have to fly at a wide range of heights (e.g., in between 0 and 300 meters), this phenomenon has to be taken into account. One solution avenue could be to parameterize the appearance model of obstacles with the MAV's height and velocity.

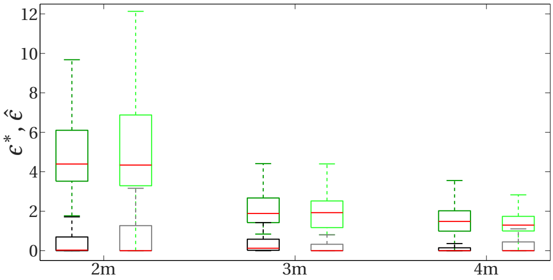


Figure 5.26: Effect of height on roughness. Dark green and light green boxplots represent the roughness ϵ^* and $\hat{\epsilon}$, respectively measured in the presence of obstacles. Black and gray boxplots represent the roughness ϵ^* and $\hat{\epsilon}$, respectively measured in the absence of obstacles.

5.7.2. APPLICATIONS

It is common for learning methods that the results are best when the test data distribution is similar to the training data distribution. For the proposed setup of SSL, this means that the results are best when the test environment is similar in appearance to the training environment (see Section 5.5). This implies that the approach will be most successful on MAVs that operate in a limited variation of environments, such as indoor MAVs flying in a warehouse for keeping track of stock or flying surveillance rounds in an industrial plant. Also, the approach will work well for MAVs that are always flying over forest areas for spotting live stock or flying over the same fields in an agricultural application. This being said, the results in Section 5.5 show that if the test environment is dissimilar from the training environment, this can be successfully detected by the MAV with uncertainty measures that can be determined by machine learning methods. In such a case, the MAV can decide to rely again on optical flow in order to adapt its mapping from appearance to obstacle detection.

5.8. CONCLUSION

We have introduced a novel setup for SSL, in which optical flow provides the supervised outputs. The surface roughness ϵ^* from the optical flow algorithm allows obstacle detection and safe landing spot selection with a straightforward landing strategy when the MAV has lateral movement. A regression function is learned that maps texton distributions to the roughness estimate ϵ^* from optical flow algorithm. We have shown that $\hat{\epsilon}$ from SSL does not only manage to detect, but even segment obstacles in the image, without having to move. A landing strategy analyzing $\hat{\epsilon}$ in nine regions in an image is able to guide the MAV to land on an area without obstacle. Both methods using roughness estimates led to successful landings in indoor experiments with a Parrot AR drone and outdoor experiments with a Parrot Bebop, both running all vision and learning algorithms on-board. In addition, we have investigated the generalization of the SSL method to different environments. Although the results show that the generalization is achieved to some extent, we recommend the use of uncertainty measures inherent to some machine learning methods to detect when the appearance of the environment is significantly different from that of the training environment. We have shown that the Shannon entropy of a Naive Bayes classifier can allow a robot to detect this and fall back on optical flow again in order to re-adapt to the new environment. We conclude that the proposed approach to SSL has the potential to significantly extend the sensory capabilities of MAVs with a single camera.

6

CONCLUSION AND RECOMMENDATIONS

The main findings of the thesis are reviewed by first answering the two research questions presented in Chapter 1. Then, a final conclusion is drawn based on the main contributions of this research. Lastly, some suggestions for future work are proposed.

6.1. DISCUSSION

This thesis has examined the use of bio-inspired monocular vision to perform autonomous landing of MAVs. The main contribution of this thesis can be divided into two parts: (I) low-level landing control ensuring a smooth landing with high performance, and (II) high-level landing control to search for a safe landing site. These contributions were considered essential in achieving the primary goal of this thesis, reflected by the problem statement in Chapter 1:

Problem statement

How can Micro Air Vehicles utilize bio-inspired monocular vision to land fully autonomously?

Two research questions (**RQs**) defined in Chapter 1 aim to answer the problem statement. Those questions are answered within the content of four chapters in this thesis, and the answers are summarized and discussed in each subsection below.

6.1.1. LOW LEVEL LANDING CONTROL

The first research question (**RQ 1**) relates to the scaleless problem of using optical flow in low-level landing control. It was formulated as follows:

RQ1: How can MAVs cope with the scaleless property of optical flow from a single camera for optical flow landing?

This research question is answered through two approaches: monocular distance estimation, and adaptive landing control strategy.

MONOCULAR DISTANCE ESTIMATION

Obtaining height estimates from optical flow alone is indeed challenging. The nonlinear observability analysis presented in Chapter 2 shows that the model is observable if and only if there exists a control input and non-zero states (i.e., height and velocity). Thus, an Extended Kalman Filter (EKF) algorithm is proposed to estimate height and vertical velocity of an MAV using the flow divergence and the control input, while approaching a landing surface. In this algorithm, the control input is used to predict the effect of an action, while the flow divergence observes the movement of the MAV. After obtaining the height estimate, either the control gain can be adapted to the height during constant flow divergence landings, or a linear control with the height feedback can be used to land the MAV with a desired predefined landing profile.

The proposed algorithm was validated by comparing the estimation results of several flight tests with the states obtained from (i) a highly accurate *OptiTrack* system for indoor flights, and (ii) a sonar sensor for outdoor flights. Three different landing strategies were presented to validate the estimation. The first strategy performed multiple landings with constant flow divergence at different setpoints ($D^* = -0,1,-0.2,-0.3s^{-1}$) and at different heights ($Z = 2$ and $3m$). These flight tests demonstrate the feasibility of the height estimation while performing a constant flow divergence landing. Since this strategy used the optical flow feedback with fixed-gain control, instabilities happen at some points during the landing. To cope with this problem, the second strategy adapted the height estimate to the control gain. This strategy allows the MAV to land with an exponential decay of both height and velocity, while maintaining stable conditions. The last strategy used both height and velocity estimates directly in landing control. Although an initial excitation is needed to measure optical flow, this strategy allows the MAV to follow a desired predefined landing profile. All these flight tests show that the proposed algorithm can estimate the height and velocity of the MAV accurately, and achieve smooth landings with these estimates.

ADAPTIVE LANDING CONTROL STRATEGY

The fundamental problem of optical flow landing is that the control gain depends on the height [50]. Chapter 3 solves this problem for a constant flow divergence landing, by introducing a control strategy that automatically adjusts the gain during landing. This strategy has two phases. At the beginning of a landing, the MAV detects height using an oscillating movement and sets the gain accordingly (Phase *I*). Then, the gains are reduced exponentially during descent and adjusted properly when the actual trajectory deviates too much from the expected constant flow divergence landing (Phase *II*). In this strategy, Phase *I* ensures that proper initial gains are chosen to have a good performance of the tracking, while Phase *II* prevents self-induced oscillations when descending.

To realize Phase *I*, a novel way to detect self-induced oscillations in real-time was developed based on observation of the flow divergence. This method detects oscillations experienced by the MAV by examining the covariance function of a windowed flow divergence and a time shifted windowed flow divergence. This function is chosen because its computation is fast and it can capture both the relative phase and the magnitude of deviations in the signal. Both simulation examples and flight tests presented in Chapter 3 show that the covariances of the signals are large at the instance when the oscillations happen. This method helps the MAV to select proper initial gains to maximize the control performance and thus avoid the MAV to descend too fast.

To examine the feasibility of Phase *II*, a stability analysis of the adaptive controller was made in Chapter 3. This analysis proves that the system is not subjected to self-induced oscillations when the adaptive controller is used for the constant flow divergence landing. This adaptive controller allows the control gains to be kept small enough to prevent self-induced oscillations, while being as high as possible to maximize control performance. Multiple flight tests were performed successfully in both indoor and windy outdoor environments using the adaptive control strategy.

The main difference between the two approaches presented in this subsection is that the first one estimates distance to the ground and uses it for landing control, whereas the latter one does not need a distance estimate. Both have their own advantages and limitations. For instance, by estimating the height and velocity, the first approach opens up the possibilities to use gain optimization techniques to improve the control performance. In addition, different landing strategies can be used with the height estimate as presented in Chapter 2. One limitation of this approach at the moment could be the simple model used in the algorithm which does not take into account external disturbances. For the second approach, height information is implicitly adapted to the control gain, which allows a smooth landing of an MAV. This method is dedicated to solve the problem of optical flow landing, but this might not be the only solution for MAV landings. Overall, both approaches lead to successful landings.

6.1.2. HIGH LEVEL LANDING CONTROL

The second research question (**RQ2**) corresponds to searching for a suitable landing site in high-level landing control. It was defined as follows:

> RQ2: How can MAVs autonomously find a suitable landing site with monocular vision?

This research question is answered through the following two methods: obstacle detection using optical flow, and self-supervised learning of obstacles appearance.

OBSTACLE DETECTION USING OPTICAL FLOW

After achieving a stable landing with optical flow, the remaining problem is to search for a safe landing site using optical flow. In this thesis, a safe landing site is defined as a relatively flat landing surface which does not have a too large inclination and is, most importantly, free of obstacles. Chapter 4 and the first part of Chapter 5 aim to find a safe landing site from the interpretation of optical flow fields. Chapter 4 estimates the slope of the landing surface by fitting the entire optical flow field using RANdom SAmple Consensus (RANSAC), while the first part of Chapter 5 uses the same approach to estimate the surface roughness to detect obstacles. Based on the slope and roughness estimates, a safe landing site can be found.

The proposed algorithm is computationally efficient to run in real-time with the on-board processor, and can also extract useful information such as surface slope, surface roughness, flow divergence, and ventral flow. To evaluate the accuracy of the slope estimate, artificial images, hand-held videos, and images from an MAV were tested in Chapter 4. From these results, we conclude that the algorithm manages to retrieve the slope angles of a landing surface accurately when the camera is calibrated, and the MAV rotation is accounted for. A landing experiment was also performed to show that the algorithm is efficient enough to run in real-time and is able to distinguish between the stairs and a flat surface.

In addition, the first part of Chapter 5 evaluates an obstacle detection method based on surface roughness ϵ^* estimated using optical flow. In fact, the success of finding a safe landing site depends on the obstacle detection performance with a reasonable number of false alarms. The obstacle detection performance is measured by true positive (TP) rate, i.e., the percentage of obstacles which are correctly identified, while the false alarm is measured by false positive (FP) rate, i.e., the percentage of empty landing areas wrongly classified as containing an obstacle. In general, the MAV is not safe if the analysis shows a low TP rate, and the MAV also cannot find a place to land if the FP rate is found to be too high. In Chapter 5, multiple on-board image sets captured in nine different indoor and outdoor scenes were tested. The performance measure of the classification tests shows that the TP rates are higher than 90% for all the scenes. For the false alarm measure, the FP rates are less than 25% for all the scenes. From these results, it can be concluded that most of the classification results show high proportions of correct identification of obstacles and non-obstacles images. Still, part of the obstacles can sometimes be missed by the limited number of tracked features. Most of the undetected cases happened because a small part of the obstacles appears at the edge of the images. This makes them difficult to be detected because the surface roughness measures the presence of obstacles in the global image. Several flight tests presented in Chapter 5 demonstrate the success of identifying the safe landing sites.

SELF-SUPERVISED LEARNING OF OBSTACLES APPEARANCE

To go beyond optical flow approaches, the surface roughness estimated using optical flow is served as a scaffold to learn the visual appearance of obstacles. The second part of Chapter 5 introduces this novel setup to deal with the main problem of using monocular motion cues, that is, their requirement of significant camera movement. By first using the surface roughness to detect obstacles, a function is learned that maps the appearance represented by texton distributions to the surface roughness. After learning, the MAV can detect obstacles based on static images. This approach allows the MAV to select a suitable landing site while hovering. Additionally, despite the fact that the surface roughness represents the global value for the presence of obstacles, this appearance learning allows for the pixel-wise segmentation of obstacles when analyzing local image patches.

Similar image sets as mentioned in the previous subsection were used to test the classification performance of appearance-based surface roughness $\widehat{\varepsilon}$. The results show that the TP rates are comparable to or slightly better than the classification using only ε^*. Similar to the optical flow method, the obstacles that appear partly on the border of the image makes them nearly "invisible" and hard to be detected when the surface roughness of the entire image is used. The FP rates are higher for complex indoor scenes. This can also be seen from the larger Normalized Root Mean Square Errors (NRMSE) in more challenging indoor scenes where the appearance variation is larger. Based on the analysis, this approach manages to detect most of the obstacles and find the possible safe places to land. In addition, using appearance-based surface roughness also led to successful landings in both indoor and outdoor experiments.

One concern of the appearance learning, which is also one of the common learning problems, is how well the learned model will generalize to different environments. Chapter 5 investigates this problem by performing K-fold tests in two different ways on the image sets to evaluate the learning capability and the exploration performance, respectively. The tests results on the learning capability show that the learning method does not have any difficulty of learning multiple scenes together, and it can accurately predict the scene with obstacles if similar or part of the scene was learned. On the other hand, for completely unknown scenes (exploration evaluation), the performance of the classifier becomes worse. To deal with that, an uncertainty measure provided by the Shannon entropy of a Naive Bayes classifier is used to detect environment change. From the observation of the tests, the uncertainty measures are, obviously, much higher for the unknown scenes than for the scenes that have been learned. This uncertainty measure can be used to direct the MAV to return to the optical flow method to adapt to the new environment.

6.2. FINAL CONCLUSION

In this thesis, bio-inspired monocular vision is utilized to achieve autonomous landings of an MAV. It is found that the bio-inspired strategies as practiced by insects, such as constant flow divergence landing, cannot be successfully used by MAVs without solving some fundamental problems of optical flow. Several solutions are proposed in this thesis. These engineering solutions lead to novel insights that are not only important to a tiny drone but may also prove their value to biology.

In the low-level landing control, a smooth landing of an MAV is achieved by either directly estimating the height or implicitly adapting the control gain to the height. Indeed, height information allows the controller to know how light or strong the feedback loop should be to perform a stable optical flow landing. In biology, it has been shown that honeybees heavily rely on optical flow for landings [49]. It is not unlikely that they know how much effort is required to flap their wings based on the object range which might be perceptually connected to optical flow in order to have a smooth landing.

In the high-level landing control, a safe landing site is determined by estimating the surface slope and roughness from the optical flow. That information is obtained from a fit of the optical flow field. Although the biological relevance is not yet known, this algorithm works well in MAVs and perhaps fits with what we know from insects. Furthermore, the learning of obstacle's appearance from optical flow allows the MAV to make a quicker decision of selecting a safe landing site. This could actually happen in insects' brains as well.

This possible biological relevance still needs to be investigated. Nevertheless, the bio-inspired strategies are extremely useful for MAVs as the efficiency of these strategies are 'practically' proven by tiny flying insects. Although the future development of advanced sensors and processors might enable the use of heavy-processing solutions and the integration of multiple sensors in the autonomous flight of MAVs, efficient algorithms are still valuable as they save computing effort without affecting other payloads performance. In addition, they are suitable for use during relatively fast maneuvers.

6.3. FUTURE WORK

The bio-inspired approach presented in this thesis uses a monocular camera from an MAV to achieve autonomous landing tasks. Nonetheless, this approach can also be adapted to different sensors, such as novel sensors that mimic insects' eyes [57, 114]. These artificial compound eyes can be seen as comprised of many tiny cameras, each with their own lens, which are all sensitive to motion. They are small and can provide an energy-efficient solution to directly measure optical flow without going through several processes needed from an ordinary camera, such as feature detection and tracking. Thus, it is expected that using the artificial compound eyes can further improve the sensing efficiency and the performance of the methodology presented in this thesis.

For height estimation, the current model used in the EKF does not take into account any external disturbances, such as wind. The flight test performed in a windy outdoor environment showed that this factor might not be a problem. However, a further investigation and a solution for compensating for wind effects might be necessary. For instance, in the presence of wind, the accuracy of the height estimation can be improved by augmenting the state vector with the wind disturbance as an additional state. The dynamics of this augmented state can be modeled as the random walk [73, 74]. Additionally, since a 3-axis accelerometer is commonly equipped in an MAV, the vertical acceleration measurement can be integrated into the model to further improve the estimation and to better deal with external disturbances.

The adaptive control strategy presented in this thesis is used to deal with the gain selection of constant flow divergence landings. The stability of this control strategy was proven using a linearized discrete model of a quadrotor moving in the vertical direction. It would be useful that a nonlinear stability analysis can be carried out to have a more comprehensive stability solution. For instance, a Lyapunov stability theorem can be used to analyze the stability conditions of the nonlinear model.

The surface slope estimated from the optical flow algorithm is currently used to evaluate the suitability of the landing surface. It is also possible to use the estimated slope to adapt the MAV attitude to the inclined surface. This will require the knowledge of the vehicle attitudes which can be known from an on-board IMU. By obtaining both surface slope and vehicle attitudes, the relative angle of the vehicle to the inclined surface can be estimated. Furthermore, a flight trajectory can be designed to safely land the MAV to the inclined surface. This approach is suitable in some real-world emergency situations where only an inclined surface is available for landing, such as on a moving ship deck.

The self-supervised learning of obstacles appearance was done using a linear regression learning method. Although using this method is straightforward and computationally efficient, it is also interesting to try out more advanced learning methods, such as deep learning. Deep learning reduces the time-consuming feature engineering problem and provides an architecture which can lend itself well to new problems. This learning method can definitely outperform the statistical learning approach. However, a deep learning approach requires a large amount of training data and expensive computation. Thus, for now, it is only applicable for larger UAVs or off-board learning.

The proposed future work could further improve the autonomous flight capabilities of MAVs, hopefully approaching more and more these capabilities of flying insects.

A

DISCRETIZATION OF LINEAR STATE SPACE MODELS

The continuous state space model is given as:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \tag{A.1}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \tag{A.2}$$

The discretized state space model can be expressed below:

$$\mathbf{x}[k+1] = \Phi\mathbf{x}[k] + \Gamma\mathbf{u}[k], \tag{A.3}$$

$$\mathbf{y}[k] = \mathbf{C}_d\mathbf{x}[k] + \mathbf{D}_d\mathbf{u}[k] \tag{A.4}$$

where

$$\Phi = e^{\mathbf{A}\Delta t}, \tag{A.5}$$

$$\Gamma = \left(\int_{\tau=0}^{\Delta t} e^{\mathbf{A}\tau}\, d\tau\right)\mathbf{B}, \tag{A.6}$$

$$\mathbf{C}_d = \mathbf{C}, \tag{A.7}$$

$$\mathbf{D}_d = \mathbf{D}, \tag{A.8}$$

Δt is the sampling time. We can find both Φ and Γ at the same time by computing the matrix exponential:

$$exp(\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \Delta t) = \begin{bmatrix} \Phi & \Gamma \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{A.9}$$

REFERENCES

[1] R. D'Andrea, *Guest editorial can drones deliver?* IEEE Transactions on Automation Science and Engineering **11**, 647 (2014).

[2] D. Borščová and K. Draganová, *Utilization possibilities of unmanned aerial systems in postal and parcel services,* Acta Avionica **16** (2014).

[3] M. Alec, *Drones for good*, Master's thesis, Delft University of Technology (2014).

[4] J. M. McMichael and M. S. Francis, *Micro air vehicles-toward a new dimension in flight,* `http://www.uadrones.net/military/research/1997/0807.htm` (1997), DARPA document, Accessed: 2017-01-03.

[5] P. Sinha, P. Esden-Tempski, C. A. Forrette, J. K. Gibboney, and G. M. Horn, *Versatile, modular, extensible VTOL aerial platform with autonomous flight mode transitions,* in *Aerospace Conference, 2012 IEEE* (IEEE, Big Sky, Montana, USA, 2012) pp. 1–17.

[6] R. J. Wood, *The first takeoff of a biologically inspired at-scale robotic insect,* IEEE transactions on robotics **24**, 341 (2008).

[7] C. de Wagter, S. Tijmons, B. D. W. Remes, and G. C. H. E. de Croon, *Autonomous flight of a 20-gram flapping wing MAV with a 4-gram onboard stereo vision system,* in *2014 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, Hong Kong, China, 2014) pp. 4982–4987.

[8] M. Keennon, K. Klingebiel, H. Won, and A. Andriukov, *Development of the nano hummingbird: A tailless flapping wing micro air vehicle,* in *AIAA aerospace sciences meeting* (AIAA Reston, VA, Nashville, Tennessee, USA, 2012) pp. 1–24.

[9] *BATCAM - Battlefield Air Targeting Camera Autonomous Micro Air Vehicle,* `http://www.designation-systems.net/dusrm/app4/batcam.html`, accessed: 2017-01-03.

[10] J. M. Grasmeyer, M. T. Keennon, *et al.*, *Development of the black widow micro air vehicle.* Progress in Astronautics and aeronautics **195**, 519 (2001).

[11] *Black hornet PD-100,* `http://www.proxdynamics.com/products/pd-100-black-hornet-prs`, accessed: 2017-01-03.

[12] B. D. W. Remes, P. Esden-Tempski, F. Van Tienen, E. Smeur, C. de Wagter, and G. C. H. E. de Croon, *Lisa-s 2.8 g autopilot for GPS-based flight of MAVs,* in *International Micro Air Vehicle Conference and Competition 2014 (IMAV 2014)* (Delft University of Technology, Delft, The Netherlands, 2014).

[13] *Parrot AR.Drone 2.0,* https://www.parrot.com/us/drones, accessed: 2017-01-03.

[14] *Amazon Prime Air,* https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011, accessed: 2017-01-04.

[15] A. Bachrach, R. He, and N. Roy, *Autonomous flight in unknown indoor environments,* International Journal of Micro Air Vehicles **1**, 217 (2009).

[16] L. Wallace, A. Lucieer, C. Watson, and D. Turner, *Development of a UAV-LiDAR system with application to forest inventory,* Remote Sensing **4**, 1519 (2012).

[17] S. B. Goldberg, M. W. Maimone, and L. Matthies, *Stereo vision and rover navigation software for planetary exploration,* in *Aerospace Conference Proceedings, 2002. IEEE*, Vol. 5 (IEEE, Big Sky, Montana, USA, 2002) pp. 5–2025.

[18] R. Brockers, Y. Kuwata, S. Weiss, and L. Matthies, *Micro air vehicle autonomous obstacle avoidance from stereo-vision,* in *SPIE Defense+ Security*, Vol. 9084 (International Society for Optics and Photonics, Baltimore, Maryland, USA, 2014) pp. 90840O–90840O.

[19] J. Park and Y. Kim, *Landing site searching algorithm of a quadrotor using depth map of stereo vision on unknown terrain,* in *AIAA Infotech@ Aerospace 2012* (Garden Grove, California, 2012) pp. 19–21.

[20] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme, *Visually guided landing of an unmanned aerial vehicle,* IEEE transactions on robotics and automation **19**, 371 (2003).

[21] D. Lee, T. Ryan, and H. J. Kim, *Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing,* in *Robotics and Automation (ICRA), 2012 IEEE International Conference on* (IEEE, St Paul, Minnesota, USA, 2012) pp. 971–976.

[22] K. Celik, S.-J. Chung, and A. K. Somani, *MVCSLAM: Mono-vision corner SLAM for autonomous micro-helicopters in GPS denied environments,* in *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (Honolulu, Hawaii, 2008) p. 6670.

[23] M. Blosch, S. Weiss, D. Scaramuzza, and R. Siegwart, *Vision based MAV navigation in unknown and unstructured environments,* in *Robotics and automation (ICRA), 2010 IEEE international conference on* (IEEE, Anchorage, Alaska, USA, 2010) pp. 21–28.

[24] S. Weiss, D. Scaramuzza, and R. Siegwart, *Monocular-SLAM–based navigation for autonomous micro helicopters in GPS-denied environments,* Journal of Field Robotics **28**, 854 (2011).

[25] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, *DTAM: Dense tracking and mapping in real-time,* in *Computer Vision (ICCV), 2011 IEEE International Conference on* (IEEE, Barcelona, Spain, 2011) pp. 2320–2327.

[26] J. Engel, J. Sturm, and D. Cremers, *Semi-dense visual odometry for a monocular camera,* in *Proceedings of the IEEE International Conference on Computer Vision* (Sydney, Australia, 2013) pp. 1449–1456.

[27] T. Schops, J. Engel, and D. Cremers, *Semi-dense visual odometry for AR on a smartphone,* in *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on* (IEEE, Munich, Germany, 2014) pp. 145–150.

[28] M. Li and A. I. Mourikis, *High-precision, consistent EKF-based visual–inertial odometry,* The International Journal of Robotics Research **32**, 690 (2013).

[29] J. Engel, T. Schöps, and D. Cremers, *LSD-SLAM: Large-scale direct monocular SLAM,* in *Computer Vision–ECCV 2014* (Springer, Zurich, Switzerland, 2014) pp. 834–849.

[30] C. Forster, M. Pizzoli, and D. Scaramuzza, *SVO: Fast semi-direct monocular visual odometry,* in *Robotics and Automation (ICRA), 2014 IEEE International Conference on* (IEEE, Hong Kong, China, 2014) pp. 15–22.

[31] V. Desaraju, N. Michael, M. Humenberger, R. Brockers, S. Weiss, and L. Matthies, *Vision-based landing site evaluation and trajectory generation toward rooftop landing,* in *Proceedings of Robotics: Science and Systems* (Berkeley, USA, 2014).

[32] R. Mur-Artal, J. Montiel, and J. D. Tardos, *ORB-SLAM: A versatile and accurate monocular SLAM system,* Robotics, IEEE Transactions on **31**, 1147 (2015).

[33] N. Franceschini, S. Viollet, F. Ruffier, and J. Serres, *Neuromimetic robots inspired by insect vision,* Advances in Science and Technology **58**, 127 (2009).

[34] R. Menzel and M. Giurfa, *Cognitive architecture of a mini-brain: The honeybee,* Trends in cognitive sciences **5**, 62 (2001).

[35] F. A. Azevedo, L. R. Carvalho, L. T. Grinberg, J. M. Farfel, R. E. Ferretti, R. E. Leite, R. Lent, S. Herculano-Houzel, *et al.*, *Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain,* Journal of Comparative Neurology **513**, 532 (2009).

[36] E. Baird, M. V. Srinivasan, S. Zhang, and A. Cowling, *Visual control of flight speed in honeybees,* Journal of Experimental Biology **208**, 3895 (2005).

[37] E. Baird, M. V. Srinivasan, S. Zhang, R. Lamont, and A. Cowling, *Visual control of flight speed and height in the honeybee,* in *From Animals to Animats 9* (Springer, Berlin, Germany, 2006) pp. 40–51.

[38] F. Ruffier and N. Franceschini, *Optic flow regulation: The key to aircraft automatic guidance,* Robotics and Autonomous Systems **50**, 177 (2005).

[39] M. Garratt, *Biologically inspired vision and control for an autonomous flying vehicle,* Ph.D. thesis, The Australian National University (2007).

[40] J. J. Gibson, *The perception of the visual world.* (Boston: Houghton Mifflin, 1950).

[41] J. J. Gibson, *The ecological approach to visual perception: classic edition* (Psychology Press, 2014).

[42] M. Srinivasan, M. Lehrer, W. Kirchner, and S. Zhang, *Range perception through apparent image speed in freely flying honeybees,* Visual neuroscience **6**, 519 (1991).

[43] M. Srinivasan, S. Zhang, M. Lehrer, and T. Collett, *Honeybee navigation en route to the goal: Visual flight control and odometry,* The Journal of Experimental Biology **199**, 237 (1996).

[44] F. Ruffier and N. Franceschini, *Aerial robot piloted in steep relief by optic flow sensors,* in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on* (IEEE, Nice, France, 2008) pp. 1266–1273.

[45] F. Valette, F. Ruffier, S. Viollet, and T. Seidl, *Biomimetic optic flow sensing applied to a lunar landing scenario,* in *Robotics and Automation (ICRA), 2010 IEEE International Conference on* (IEEE, Alaska, USA, 2010) pp. 2253–2260.

[46] D. Izzo, N. Weiss, and T. Seidl, *Constant-optic-flow lunar landing: Optimality and guidance,* Journal of Guidance, Control, and Dynamics **34**, 1383 (2011).

[47] B. Herisse, T. Hamel, R. Mahony, and F.-X. Russotto, *The landing problem of a VTOL unmanned aerial vehicle on a moving platform using optical flow,* in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on* (IEEE, Taipei, Taiwan, 2010) pp. 1600–1605.

[48] D. Izzo and G. C. H. E. de Croon, *Landing with time-to-contact and ventral optic flow estimates,* Journal of Guidance, Control, and Dynamics **35**, 1362 (2012).

[49] E. Baird, N. Boeddeker, M. R. Ibbotson, and M. V. Srinivasan, *A universal strategy for visually guided landing,* Proceedings of the National Academy of Sciences **110**, 18686 (2013).

[50] G. C. H. E. de Croon, *Monocular distance estimation with optical flow maneuvers and efference copies: A stability-based strategy,* Bioinspiration & biomimetics **11**, 016004 (2016).

[51] G. C. H. E. de Croon, D. Alazard, and D. Izzo, *Controlling spacecraft landings with constantly and exponentially decreasing time-to-contact,* IEEE Transactions on Aerospace and Electronic Systems **51**, 1241 (2015).

[52] F. Kendoul, *Four-dimensional guidance and control of movement using time-to-contact: Application to automated docking and landing of unmanned rotorcraft systems,* The International Journal of Robotics Research **33**, 237 (2014).

[53] B. Herisse, F.-X. Russotto, T. Hamel, and R. Mahony, *Hovering flight and vertical landing control of a VTOL unmanned aerial vehicle using optical flow,* in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on* (IEEE, Nice, France, 2008) pp. 801–806.

[54] M. T. Alkowatly, V. M. Becerra, and W. Holderbaum, *Bioinspired autonomous visual vertical control of a quadrotor unmanned aerial vehicle,* Journal of Guidance, Control, and Dynamics **38**, 249 (2014).

[55] L. Muratet, S. Doncieux, Y. Briere, and J.-A. Meyer, *A contribution to vision-based autonomous helicopter flight in urban environments,* Robotics and Autonomous Systems **50**, 195 (2005).

[56] J.-C. Zufferey and D. Floreano, *Fly-inspired visual steering of an ultralight indoor aircraft,* IEEE Transactions on Robotics **22**, 137 (2006).

[57] D. Floreano, R. Pericet-Camara, S. Viollet, F. Ruffier, A. Brückner, R. Leitel, W. Buss, M. Menouni, F. Expert, R. Juston, *et al.*, *Miniature curved artificial compound eyes,* Proceedings of the National Academy of Sciences **110**, 9267 (2013).

[58] J. Conradt, *On-board real-time optic-flow for miniature event-based vision sensors,* in *Robotics and Biomimetics (ROBIO), 2015 IEEE International Conference on* (IEEE, Zhuhai, China, 2015) pp. 1858–1863.

[59] L. F. Tammero and M. H. Dickinson, *The influence of visual landscape on the free flight behavior of the fruit fly Drosophila melanogaster,* Journal of Experimental Biology **205**, 327 (2002).

[60] H. W. Ho, G. C. H. E. de Croon, and Q. P. Chu, *Distance and velocity estimation using optical flow from a monocular camera,* International Journal of Micro Air Vehicles (accepted).

[61] C. McCarthy, N. Barnes, and R. Mahony, *A robust docking strategy for a mobile robot using flow field divergence,* Robotics, IEEE Transactions on **24**, 832 (2008).

[62] H. W. Ho, G. C. H. E. de Croon, E. van Kampen, Q. P. Chu, and M. Mulder, *Adaptive control strategy for constant optical flow divergence landing,* IEEE transactions on robotics (submitted).

[63] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, *A robust and modular multi-sensor fusion approach applied to MAV navigation,* in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, Tokyo, Japan, 2013) pp. 3923–3929.

[64] J. Engel, J. Sturm, and D. Cremers, *Scale-aware navigation of a low-cost quadrocopter with a monocular camera,* Robotics and Autonomous Systems **62**, 1646 (2014).

[65] F. van Breugel, K. Morgansen, and M. H. Dickinson, *Monocular distance estimation from optic flow during active landing maneuvers,* Bioinspiration & biomimetics **9**, 025002 (2014).

[66] P. I. Corke and M. C. Good, *Dynamic effects in high-performance visual servoing,* in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on* (IEEE, Nice, France, 1992) pp. 1838–1843.

[67] H. J. Asl, G. Oriolo, and H. Bolandi, *An adaptive scheme for image-based visual servoing of an underactuated UAV,* International Journal of Robotics and Automation **29**, 92 (2014).

[68] R. Hermann and A. J. Krener, *Nonlinear controllability and observability,* IEEE Transactions on automatic control **22**, 728 (1977).

[69] E. Rosten and T. Drummond, *Machine learning for high-speed corner detection,* in *Computer Vision–ECCV 2006* (Springer, Graz, Austria, 2006) pp. 430–443.

[70] E. Rosten and T. Drummond, *Fusing points and lines for high performance tracking,* in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on,* Vol. 2 (IEEE, Beijing, China, 2005) pp. 1508–1515.

[71] J. Y. Bouquet, *Pyramidal implementation of the Lucas Kanade feature tracker,* Intel Corporation, Microsoft Research Labs, Tech. Rep. (2000).

[72] B. D. W. Remes, D. Hensen, F. van Tienen, C. de Wagter, E. van der Horst, and G. C. H. E. de Croon, *Paparazzi: How to make a swarm of Parrot AR drones fly autonomously based on GPS.* in *International Micro Air Vehicle Conference and Flight Competition (IMAV2013)* (Toulouse, France, 2013) pp. 17–20.

[73] C.-S. Hsieh, *Robust two-stage Kalman filters for systems with unknown inputs,* IEEE Transactions on Automatic Control **45**, 2374 (2000).

[74] S. H. Park, P. S. Kim, O.-K. Kwon, and W. H. Kwon, *Estimation and detection of unknown inputs using optimal FIR filter,* Automatica **36**, 1481 (2000).

[75] P. Lu, E. van Kampen, C. de Visser, and Q. P. Chu, *Nonlinear aircraft sensor fault reconstruction in the presence of disturbances validated by real flight data,* Control Engineering Practice **49**, 112 (2016).

[76] H. W. Ho and G. C. H. E. de Croon, *Characterization of flow field divergence for MAVs vertical control landing,* in *AIAA Guidance, Navigation, and Control Conference* (San Diego, California, USA, 2016) p. 0106.

[77] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, *Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments,* in *SPIE Defense, Security, and Sensing,* Vol. 7332 (International Society for Optics and Photonics, Orlando, Florida, USA, 2009) pp. 733219–733219.

[78] J. L. Sanchez-Lopez, J. Pestana, S. Saripalli, and P. Campoy, *An approach toward visual autonomous ship board landing of a VTOL UAV,* Journal of Intelligent & Robotic Systems **74**, 113 (2014).

[79] P. Li, M. Garratt, and A. Lambert, *Monocular snapshot-based sensing and control of hover, takeoff, and landing for a low-cost quadrotor,* Journal of Field Robotics **32**, 984 (2015).

[80] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, *MonoSLAM: Real-time single camera SLAM,* Pattern Analysis and Machine Intelligence, IEEE Transactions on **29**, 1052 (2007).

[81] J. Artieda, J. M. Sebastian, P. Campoy, J. F. Correa, I. F. Mondragón, C. Martínez, and M. Olivares, *Visual 3-D SLAM from UAVs,* Journal of Intelligent and Robotic Systems **55**, 299 (2009).

[82] D. N. Lee, *Guiding movement by coupling taus,* Ecological psychology **10**, 221 (1998).

[83] D. N. Lee, *General tau theory: Evolution to date.* Perception **38**, 837 (2009).

[84] G. C. H. E. de Croon, H. W. Ho, C. de Wagter, E. van Kampen, B. D. W. Remes, and Q. P. Chu, *Optic-flow based slope estimation for autonomous landing,* International Journal of Micro Air Vehicles **5**, 287 (2013).

[85] H. C. Longuet-Higgins and K. Prazdny, *The interpretation of a moving retinal image,* Proceedings of the Royal Society of London. Series B. Biological Sciences **208**, 385 (1980).

[86] M. A. Fischler and R. C. Bolles, *Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,* Communications of the ACM **24**, 381 (1981).

[87] R. Ganesh, *Control engineering* (Pearson education, 2010) pp. 279–282.

[88] G. V. Chowdhary, S. Srinivasan, and E. N. Johnson, *Frequency domain method for real-time detection of oscillations,* Journal of Aerospace Computing, Information, and Communication **8**, 42 (2011).

[89] P. Rzucidło, *The detection of pilot-induced oscillations,* Aviation **11**, 15 (2007).

[90] G. Sabiron, P. Chavent, T. Raharijaona, P. Fabiani, and F. Ruffier, *Low-speed optic-flow sensor onboard an unmanned helicopter flying outside over fields,* in *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (IEEE, Karlsruhe, Germany, 2013) pp. 1742–1749.

[91] F. Expert, S. Viollet, and F. Ruffier, *Outdoor field performances of insect-based visual motion sensors,* Journal of Field Robotics **28**, 529 (2011).

[92] C. Evangelista, P. Kraft, M. Dacke, J. Reinhard, and M. V. Srinivasan, *The moment before touchdown: landing manoeuvres of the honeybee apis mellifera,* Journal of Experimental Biology **213**, 262 (2010).

[93] Y. M. Song, Y. Xie, V. Malyarchuk, J. Xiao, I. Jung, K.-J. Choi, Z. Liu, H. Park, C. Lu, R.-H. Kim, *et al.*, *Digital cameras with designs inspired by the arthropod eye,* Nature **497**, 95 (2013).

[94] J. Shi and C. Tomasi, *Good features to track,* in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on* (IEEE, Seattle, Washington, USA, 1994) pp. 593–600.

[95] B. D. Lucas, T. Kanade, *et al.*, *An iterative image registration technique with an application to stereo vision,* in *IJCAI*, Vol. 81 (Vancauver, British Columbia, Canada, 1981) pp. 674–679.

[96] H. W. Ho, C. De Wagter, B. D. W. Remes, and G. C. H. E. de Croon, *Optical flow for self-supervised learning of obstacle appearance,* in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on* (IEEE, Hamburg, Germany, 2015) pp. 3098–3104.

[97] H. W. Ho, C. D. Wagter, B. D. W. Remes, and G. C. H. E. de Croon, *Optical-flow based self-supervised learning of obstacle appearance applied to MAV landing,* The International Journal of Robotics Research (submitted).

[98] L. Mejias, P. Campoy, K. Usher, J. Roberts, and P. Corke, *Two seconds to touchdown-vision-based controlled forced landing,* in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on* (Ieee, Beijing, China, 2006) pp. 3527–3532.

[99] R. Roberts and F. Dellaert, *Optical flow templates for superpixel labeling in autonomous robot navigation,* in *5th Workshop on Planning, Perception and Navigation for Intelligent Vehicles (PPNIV13)* (Tokyo, Japan, 2013).

[100] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. R. Bradski, *Self-supervised monocular road detection in desert terrain,* in *Robotics: science and systems*, Vol. 38 (Philadelphia, Ann Arbor, Michigan, USA, 2006).

[101] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, *et al.*, *Stanley: The robot that won the darpa grand challenge,* Journal of field Robotics **23**, 661 (2006).

[102] U. A. Muller, L. D. Jackel, Y. LeCun, and B. Flepp, *Real-time adaptive off-road vehicle navigation and terrain classification,* in *SPIE Defense, Security, and Sensing* (International Society for Optics and Photonics, Baltimore, Maryland, USA, 2013) pp. 87410A–87410A.

[103] A. Lookingbill, J. Rogers, D. Lieb, J. Curry, and S. Thrun, *Reverse optical flow for self-supervised adaptive autonomous robot navigation,* International Journal of Computer Vision **74**, 287 (2007).

[104] D. Dey, K. S. Shankar, S. Zeng , R. Mehta, M. T. Agcayazi, C. Eriksen, S. Daftry, M. Hebert , and J. A. D. Bagnell, *Vision and learning for deliberative monocular cluttered flight,* in *Field and Service Robotics (FSR)*, Vol. 113 (Toronto, Canada, 2015) pp. 391–409.

[105] B. Lee, K. Daniilidis, and D. D. Lee, *Online self-supervised monocular visual odometry for ground vehicles,* in *Robotics and Automation (ICRA), 2015 IEEE International Conference on* (IEEE, Seattle, Washington, USA, 2015) pp. 5232–5238.

[106] M. Varma and A. Zisserman, *Texture classification: Are filter banks necessary?* in *Computer vision and pattern recognition, 2003. Proceedings. 2003 IEEE computer society conference on*, Vol. 2 (IEEE, Madison, Wisconsin, USA, 2003) pp. II–691.

[107] G. C. H. E. de Croon, E. de Weerdt, C. de Wagter, B. D. W. Remes, and R. Ruijsink, *The appearance variation cue for obstacle avoidance,* Robotics, IEEE Transactions on **28**, 529 (2012).

[108] T. Kohonen, *Self-organizing maps* (Springer-Verlag, New York, 2001).

[109] G. C. H. E. de Croon, K. de Clercq, R. Ruijsink, B. D. W. Remes, and C. de Wagter, *Design, aerodynamics, and vision-based control of the delfly,* International Journal of Micro Air Vehicles **1**, 71 (2009).

[110] R. Duin, P. Juszczak, P. Paclik, E. Pekalska, D. de Ridder, D. Tax, and S. Verzakov, *A matlab toolbox for pattern recognition,* PRTools version **3** (2000).

[111] A. Cesetti, E. Frontoni, A. Mancini, and P. Zingaretti, *Autonomous safe landing of a vision guided helicopter,* in *Mechatronics and Embedded Systems and Applications (MESA), 2010 IEEE/ASME International Conference on* (IEEE, Qingdao, ShanDong, China, 2010) pp. 125–130.

[112] C. E. Shannon, *A mathematical theory of communication,* ACM SIGMOBILE Mobile Computing and Communications Review **5**, 3 (2001).

[113] G. Hattenberger, M. Bronz, and M. Gorraz, *Using the paparazzi uav system for scientific research,* in *IMAV 2014, International Micro Air Vehicle Conference and Competition 2014* (Delft, The Netherlands, 2014) pp. 247–252.

[114] K.-H. Jeong, J. Kim, and L. P. Lee, *Biologically inspired artificial compound eyes,* science **312**, 557 (2006).

NOMENCLATURE

ACRONYMS

AUC	Area Under Curve
ConvNet	CONVolutional neural NETwork
DARPA	Defense Advanced Research Projects Agency
DFT	Discrete Fourier Transform
DGPS	Differential Global Positioning System
EKF	Extended Kalman Filter
FAST	Features from Accelerated Segment Test
FFT	Fast Fourier Transform
FN	False Negative
FoE	Focus of Expansion
FP	False Positive
FPS	Frame Per Second
GNC	Guidance, Navigation, and Control
GPS	Global Positioning System
IMU	Inertia Measurement Unit
LASSO	Least Absolute Shrinkage and Selection Operator
Lidar	LIght Detection And Ranging
MAV	Micro Air Vehicle
NRMSE	Normalized Root Mean Square Error
PDF	Probability Density Function
PID	Proportional-Integral-Derivative
RANSAC	RANdom SAmple Consensus
RMSE	Root Mean Square Error

ROC	Receiver Operating Characteristic
RQ	Research Question
RTK	Real-Time Kinematic
SLAM	Simultaneous Localization And Mapping
Sonar	SOund Navigation And Ranging
SSL	Self-Supervised Learning
TP	True Positive
UAV	Unmanned Aerial Vehicle

GREEK SYMBOLS

α,β	slope angles of the ground surface	[$rad,^o$]
Δt	time interval between two images	[s]
ϵ^*	surface roughness estimated from optical flow	
κ	integral time constant	
Λ	regression model bias	
λ	system poles	
Γ	discretized matrix of **B**	
Φ	discretized matrix of **A**	
$\mathscr{O}$	observability algebra	
μ	control input	
μ_g	hovering command	
μ_T	total command	
ϕ,θ,ψ	roll, pitch, yaw along the body axis	[rad]
$\rho_{(\cdot)}$	regression coefficients	
σ	discrete frequency domain operator	
σ_0	zeros of a discrete system	
τ	time-to-contact	[s]
ϱ,χ	real and imaginary part of σ	
$\vartheta_x,\vartheta_y,\vartheta_z$	velocities along the body axis scaled w.r.t. the height	[$1/s$]

$\widehat{\varepsilon}$	surface roughness estimated from visual appearance	

LATIN SYMBOLS

$\mathbf{A}$	system matrix	
$\mathbf{B}$	input matrix	
$\mathbf{C}$	output matrix	
$\mathbf{D}$	feedforward matrix	
$\mathbf{H}_k$	observation model which relates the state to the measurement	
$\mathbf{K}_k$	Kalman gain vector	
$\mathbf{p_u}, \mathbf{p_v}$	estimated parameter vectors of optical flow model	
$\mathbf{q}$	texton probability distribution	
$\mathbf{r}(t)$	true state vector	
$\mathbf{v}_k$	observation noise	
$\mathbf{w}(t)$	system noise	
$\mathbf{x}(t)$	estimated state vector	
$\mathbf{z}_k$	measurement vector	
$\mathcal{N}(\sigma), \mathcal{D}(\sigma)$	numerator and denominator of the discrete closed-loop function	
$\mathcal{P}$	period of one full oscillation	$[s]$
$\tilde{f}$	camera focal length	$[pixel]$
$\widehat{D}$	flow divergence estimate	$[1/s]$
$\widehat{D}'$	time-shifted windowed flow divergence	$[1/s]$
$\widehat{D}_s$	size divergence estimate	$[1/s]$
a, b	tangents of surface slope angles α and β	
a_Z	vertical acceleration along the body axis	$[m/s^2]$
A_{SF}	safe area	
$cov(\cdot)$	covariance function	
D	actual or ground truth flow divergence	$[1/s]$
d	image distance between two features on the ground	$[pixel]$
D^*	flow divergence set point	$[1/s]$

d_0	wind disturbance estimate	
d_c	image distance from its center to sub-image center of safe region	$[pixel]$
d_p	actual distance from the current position to a safe landing position	$[m]$
$E[\cdot]$	expected value	
$e_{(\cdot)}$	fit coefficients of noise model for flow divergence estimates	
Err	classification error	
$err_{\widehat{D}}, err_{\widehat{D}_s}$	noise model errors of $\widehat{D}$ and $\widehat{D}_s$	$[1/s]$
$F_{PI}(\sigma)$	discrete PI controller	
$G(\sigma), H(\sigma)$	open-loop and closed-loop transfer function of a discrete system	
H	uncertainty measure	
h	height of a camera above the ground surface	$[m]$
K_i	integral control gain	
K_p	proportional control gain	
K_{cr}	critical proportional control gain	
L	actual distance between two features on the ground	$[m]$
l_1, l_2	time steps where the presence of obstacles are predicted using ϵ^* and $\widehat{\epsilon}$	
$L_f h$	Lie derivatives of the observation model w.r.t the model dynamics	
Lag	number of sample delays of flow divergence estimate	
n	total number of tracked features	
n_{test}	total number of test data	
N_{win}	window size of a moving median filter	
p, q, r	angular rates along the body axis	$[rad/s]$
P_{land}	desired landing waypoint	
Q	system noise covariance	
R	measurement noise covariance	
t	time	$[s]$
u, v	horizontal and vertical image velocities	$[pixel/s]$
V_X, V_Y, V_Z	velocities along the body axis	$[m/s]$

w	random walk model	
x, y	image point	$[pixel]$
X, Y, Z	real-world coordinates	
$y(t)$	output vector	
Z	vertical position along the body axis	$[m]$

REFERENCE FRAMES AND OTHERS

$(\cdot)^T$	transpose term
$\dot{(\cdot)}$	derivative term
$\widehat{(\cdot)}$	estimated term
$O_b X_b Y_b Z_b$	body coordinate system
$O_c X_c Y_c Z_c$	camera coordinate system
$O_w X_w Y_w Z_w$	world coordinate system

SUBSCRIPTS

$(\cdot)_{cr}$	delay-corrected term
$(\cdot)_{fit}$	fitted term
$(\cdot)_m$	measured term
$(\cdot)_{ref}$	reference term
$(\cdot)_{th}$	threshold term

SAMENVATTING

AUTONOMOUS LANDING OF MICRO AIR VEHICLES THROUGH BIO-INSPIRED MONOCULAR VISION

Hann Woei HO

Zelfstandig vliegende drones – en dan vooral de kleine en lichte varianten bekend als microdrones – kunnen op het moment niet zelf een veilige landplek vinden, en ook geen vloeiende landing maken. De hoofdreden hiervoor is dat de meeste huidige oplossingen nog teveel rekenkracht vereisen en dus niet toepasbaar zijn op microdrones, die bijzonder weinig rekenkracht en sensoren aan boord hebben. Dit type drones vereist daarom een energie-efficiënte oplossing om te landen, gebruikmakend maakt van een lichte sensor.

Een veelbelovende oplossingsrichting voor het zelfstandig landen van microdrones is om inspiratie te halen uit hoe kleine vliegende insecten dit doen. De reden hiervoor is dat insecten zeer complexe taken uit kunnen voeren – zoals navigeren in omgevingen met veel obstakels, en het uitvoeren van zachte landingen – terwijl ze maar erg weinig neuronen hebben in hun brein en ook weinig en lichte sensoren. Dit betekent dat ze efficiënte en robuuste oplossingen hebben voor allerlei waarnemings- en besturingsproblemen, die wellicht ideaal geschikt kunnen zijn voor het besturen van microdrones, en dus ook voor het zelfstandig landen. Onderzoek toont aan dat vliegende insecten voor het uitvoeren van complexe navigatietaken veel gebruik maken van de manier waarop objecten door hun visuele beeld bewegen, dat wil zeggen van de *optische stroom*.

Echter, er zijn veel uitdagingen wanneer een microdrone alleen optische stroom wil gebruiken om te landen. De optische stroom is niet “geschaald”, en geeft dus noch informatie over de afstand tot een object, noch over de relatieve snelheid van de camera tot een object. Het geeft alleen informatie over de snelheid gedeeld door de afstand. De ongeschaalde optische stroom kan tot verschillende stabiliteitsproblemen leiden tijdens het landen van een microdrone. Bijvoorbeeld, als een microdrone op een plat oppervlak wil landen, dan komt er een punt waarop de reacties van de microdrone te sterk worden gegeven de hoogte: de microdrone gaat hierdoor oscilleren. Dus is hoogte-informatie belangrijk voor optische stroom landingen. Bovendien, voor een volledig zelfvliegende drone is landen op een plat vlak niet genoeg. De microdrone moet zelf ook een geschikte landingsplek kunnen bepalen. Aangezien optische stroom alleen bepaald kan worden op plaatsen met textuur, geeft het altijd slechts gedeeltelijke informatie en moet de stroom verder verwerkt worden voor interpretatie. Microdrones hebben efficiënte algoritmen nodig die, gegeven de optische stroom, kunnen bepalen of een oppervlak plat is en vrij van obstakels.

Het doel van dit proefschrift is om de besturingsproblemen gerelateerd aan het landen met optische stroom op te lossen. Dit leidt tot de volgende probleemstelling:

Hoe kunnen microdrones biologisch geïnspireerde monoculaire waarneming gebruiken om volledig zelfstandig te landen?

De probleemstelling kan opgesplitst worden in een besturing op laag niveau en een besturing op hoog niveau, beide aan te pakken met biologisch geïnspireerde methoden. Het lage niveau verzekert een vloeiende landing door het schaalprobleem van optische stroom op te lossen voor zelfstandig landen, terwijl het hoge niveau het zoeken naar een geschikte landingsplaats betreft. Dit alles met een enkele camera. Dit leidt tot de volgende onderzoeksvragen:

1. Hoe kunnen microdrones omgaan met het ontbreken van schaal van optische stroom van een enkele camera voor zelfstandig landen?
2. Hoe kunnen microdrones autonoom een geschikte landingsplaats vinden met een enkele camera?

Voor de beantwoording van de eerste onderzoeksvraag worden twee aanpakken voorgesteld. De eerste aanpak benadert het probleem door een filter te introduceren, een zogenaamd “extended Kalman filter”, of EKF. Dit filter schat de hoogte en de verticale snelheid van een microdrone tijdens een landing. Het algoritme gebruikt voorkennis van de stuursignalen, de zogenoemde “efference copy”, en de vervolgens geobserveerde expansie van de optische stroom. De voorspellingsstap in de EKF gebruikt de efference copy om de toestand te voorspellen (hoogte en snelheid), terwijl voor de correctiestap van het filter de geobserveerde beeldexpansie wordt gebruikt. Hiermee kunnen zowel hoogte als snelheid geschat worden.

Om de observeerbaarheid van het systeem te bepalen, is er een non-lineaire waarneembaarheidsanalyse uitgevoerd. De analyse toont aan dat met de kennis van de stuursignalen en de beeldexpansie het systeem waarneembaar is. Het algoritme is zowel in simulatie als op een echte vliegende robot getest, waarbij meerdere experimenten zowel binnens- als buitenshuis zijn uitgevoerd. De resultaten laten zien dat het algoritme erin slaagt om hoogte en snelheid nauwkeurig te schatten tijdens de landing. De geschatte waardes worden vergeleken met zeer nauwkeurige metingen van een bewegingsvolgsysteem binnen en van een sonar buiten. De verkregen hoogteschattingen kunnen zowel gebruikt worden voor het bepalen van de juiste stuurreacties tijdens een optische stroom landing of in een lineair besturingssysteem dat de geschatte hoogte direct gebruikt voor de besturing. Beide methodes resulteren in vloeiende landingen.

De tweede aanpak lost het schaalprobleem op door middel van een strategie die de sterkte van de stuurreacties aanpast tijdens de landing. Deze strategie heeft twee es die een constante expansie van het beeld en een vloeiende landing nastreven. In fase *I* detecteert de microdrone zijn hoogte tijdens het stilhangen in de lucht, door de stuurreacties te laten toenemen tot aan het punt waar oscillaties optreden. In fase *II* wordt de constante beeldexpansie landing ingezet, en worden de stuurreacties over de tijd exponentieel teruggebracht. Ook worden aanpassingen gedaan als de microdrone afwijkt van de gewenste beeldexpansie. In deze strategie zorgt fase *I* ervoor dat de goede initiële stuurreacties gekozen worden, opdat er snel gereageerd kan worden op afwijkingen, terwijl fase *II* ervoor zorgt dat zelf-geïnduceerde oscillaties voorkomen worden tijdens de landing.

Om fase *I* te realiseren is er een nieuwe methode ontwikkeld om te detecteren wanneer de microdrone zelf oscillaties veroorzaakt, gebaseerd op de geobserveerde beeldexpansie. Deze methode detecteert de oscillaties door het bepalen van de covariantie tussen een sequentie van observaties, waarbij de sequentie tot aan het huidige moment vergeleken wordt met een even lange sequentie van een aantal tijdstappen geleden. De methode is rekentechnisch efficiënt en kan zowel de relatieve fase als de grootte van de oscillaties van het signaal schatten. Zowel simulaties als echte experimenten laten grotere negatieve covarianties zien wanneer er oscillaties optreden. Door het detecteren van deze oscillaties kan de microdrone de juiste sterkte van de stuurreacties bepalen. Om de mogelijkheid van fase *II* te onderzoeken, is er een stabiliteitsanalyse uitgevoerd van het systeem. Deze analyse laat zien dat het systeem niet zelf oscillaties zal induceren als de voorgestelde strategie wordt gebruikt tijdens een landing met constante beeldexpansie. Bovendien tonen meerdere experimenten met de microdrone aan dat de strategie leidt tot vloeiende landingen, zowel binnen als buiten.

Om de tweede onderzoeksvraag te beantwoorden, wordt in dit proefschrift een efficiënt algoritme voorgesteld dat optische stroom gebruikt om veilige landingsplaatsen te identificeren. Zulke landingsplaatsen horen relatief vlak te zijn, niet te steil, en – veruit het belangrijkst – geen obstakels te hebben. Om een veilige landingsplaats te vinden gebruikt het algoritme eerst een RANdom SAmple Consensus (RANSAC) schatting, om een lineaire parameterisatie te vinden van het optische stroomveld. De schatting bevat een parameter die de steilheid van het oppervlak representeert en een parameter die de “ruwheid”, en dus mogelijke aanwezigheid van obstakels, van het oppervlak representeert. Als de lineaire schatting slecht past bij het optische stroomveld, dan is de ruwheid en dus de kans op obstakels hoog. Als daarentegen de schatting goed past bij het stroomveld, dan is de ruwheid laag en zijn er waarschijnlijk geen significante obstakels voor het landen.

Om de nauwkeurigheid van de steilheidsschatting te schatten zijn kunstmatige plaatjes, met de hand gemaakte video’s, en plaatjes vanaf een microdrone gebruikt. De resultaten laten zien dat het algoritme de steilheid nauwkeurig kan schatten als de camera gekalibreerd is en rotaties worden gecompenseerd. Bovendien is er een landingsexperiment uitgevoerd dat het verschil kan maken tussen een trap en een platte vloer, waarbij de drone zelf besluit te landen pas als hij de trap af is. Om de ruwheidsschatting te evalueren, zijn er testen uitgevoerd met plaatjes uit negen verschillende binnen- en buitenomgevingen. Het succes van het vinden van een landingsplaats hangt af van een obstakeldetectie mechanisme met een beperkte hoeveelheid “vals alarm”, dat wil zeggen onnodige detecties als er geen obstakels zijn. Het succes van de methode wordt bepaald door middel van de “positieve detectieratio”, met andere woorden het percentage obstakels dat ook daadwerkelijk gedetecteerd wordt, en de “vals alarmratio”, het percentage van vlakke landingsplaatsen waar de methode toch onterecht obstakels ziet. De experimenten tonen aan dat de positieve detectieratio hoger dan 90% is voor alle omgevingen, terwijl de vals alarmratio lager dan 25% is voor alle omgevingen. Deze classificatieprestatie is meestal goed genoeg voor het vinden van een landingsplaats. De meeste van de ongedetecteerde obstakels komen voor wanneer er slechts een klein stukje van het obstakel in de rand van het beeld zichtbaar is. Dit maakt het obstakel moeilijk te detecteren, aangezien het algoritme de aanwezigheid van obstakels in een heel plaatje evalueert. Meerdere experimenten waarin een microdrone het algoritme gebruikt, laten zien dat de microdrone succesvol een landingsplaats kan kiezen.

Bovendien probeert dit onderzoek “voorbij” optische stroom te gaan voor het selecteren van een geschikte landingsplaats. Een nieuwe opzet wordt geïntroduceerd voor “leren met zelf-supervisie”, waarin de schattingen van de optische stroom gebruikt worden als een leraar die de microdrone vertelt wanneer er obstakels in zicht zijn. De microdrone kan dan associaties gaan maken tussen het voorkomen (kleuren en texturen) van objecten in zicht, en de aanwezigheid van obstakels. Deze nieuwe opzet lost een belangrijk probleem van monoculaire optische stroom op, namelijk dat die vereist dat de microdrone voldoende beweegt. Door de optische stroom gebaseerde ruwheid van het oppervlak te gebruiken voor het detecteren van obstakels, wordt er een functie geleerd die het voorkomen van objecten in zicht – gerepresenteerd als een distributie van “textons” – projecteert naar de bepaalde ruwheid. Na het leren kan de microdrone dan een geschikte landingsplaats selecteren door gewoon stil te hangen in de lucht. Interessant genoeg, ondanks het feit dat de ruwheid slechts oordelen geeft over een geheel plaatje, kan de functie na het leren gebruikt worden om objecten op pixelniveau te segmenteren, door het evalueren van lokale textondistributies.

Om de classificatieprestatie te bepalen van deze aanpak zijn dezelfde datasets gebruikt als voor de optische stroom methode. De resultaten laten een licht betere positieve detectieratio zien dan met optische stroom. Echter, de vals alarmratio’s zijn wat hoger voor de complexere binnen- omgevingen, waar het visueel voorkomen van obstakels veel varieert. Dit leidt binnen ook tot een wat hogere genormaliseerde gemiddelde kwadratische fout. Gebaseerd op de analyse kan geconcludeerd worden dat de aanpak de meeste obstakels en geschikte landingsplaatsen kan vinden. Binnen- en buitenexperimenten leiden tot succesvolle landingen.

Samenvattend kan geconcludeerd worden dat biologisch geïnspireerde monoculaire waarneming veelbelovend is voor het autonome landen van een microdrone. In het proefschrift wordt aangetoond dat de op het gedrag van insecten geïnspireerde strategieën, zoals het constant houden van de beeldexpansie voor landen, alleen succesvol uitgevoerd kunnen worden door bepaalde fundamentele problemen op te lossen van optische stroom besturing. Daarnaast kan het leren van het visuele voorkomen op basis van optische stroom de landingsvaardigheden van microdrones nog verder verbeteren. Dit proefschrift stelt verschillende oplossingen voor die tot nieuwe inzichten leiden die niet alleen belangrijk zijn voor een microdrone, maar ook wellicht hun waarde kunnen hebben voor de biologie.

Het wordt aangeraden toekomstig werk te verrichten voor het aanpassen van de biologisch geïnspireerde aanpak aan nieuwe sensoren die insectenogen beter benaderen. Deze nieuwe sensoren zijn klein en kunnen direct optische stroom meten. Daardoor kan de efficiënte van waarnemen en rekenen nog verder verbeterd worden. Voor hoogteschatting wordt het aangeraden de toestandsrepresentatie van het EKF uit te breiden met een variabele voor de windverstoring. Alhoewel buitenexperimenten aantoonden dat de wind geen groot probleem hoeft te vormen, zou het toevoegen van zo'n variabele de nauwkeurigheid verder kunnen verhogen. Tenslotte, voor het leren met zelfsupervisie zou het interessant zijn om methoden van diepe neurale netwerken toe te passen. Het voorgestelde toekomstig werk zou significant kunnen bijdragen aan de vliegvaardigheden van microdrones, en ze in het ideale geval dichter bij de unieke vaardigheden van kleine vliegende insecten brengen.

ACKNOWLEDGEMENTS

This thesis summarizes my four-year research work which was carried out at Micro Air Vehicle Laboratory (MAVLab) of the Faculty of Aerospace Engineering, Delft University of Technology. This thesis would never be possible without the contributions and support from a number of individuals.

First of all, I would like to express my sincerest gratitude to my promoter, Prof. Dr. ir. Max Mulder, for his invaluable guidance and support throughout my Ph.D. studies. I really appreciate your time and efforts in reading this thesis and giving me constructive reviews on this thesis as well as on our manuscript. Working with you will definitely benefit my future career in academia.

I also would like to convey my profoundest gratitude and tremendous respect to my co-promoter and daily supervisor, Dr. Guido C.H.E. de Croon, for his great enthusiasm, divine guidance, and continuous support in these four years. Your passionate interest in science has inspired me a lot. I really enjoy the time when we had challenging scientific discussions, performed experiments together, and attended conferences and competitions. These experiences enriched my study life here. Also, many thanks for the boundless patience you have given and knowledge you have shared throughout my Ph.D. journey. Without them, this Ph.D. could never be achievable.

Many thanks to Dr. Qiping Chu for his scientific guidance and all other support during my MSc. and Ph.D. studies. You always made me feel inspired and motivated whenever having discussion and meetings with you. This gives me more confidence in my research work. Also, I would like to thank for your kindness for offering help and support for my family during the tough time I had in the Netherlands.

I would like to thank Dr. ir. Erik-Jan van Kampen for his guidance and support. I really enjoy our discussions which provide me a lot of interesting and creative ideas. Also, I would like to thank for giving me many helpful comments when reviewing our manuscripts.

I also greatly appreciate the time and effort spared by the committee members of my Ph.D. defense to assess my thesis and attend my defense.

I would like to acknowledge Bart Remes, Christophe de Wagter, Erik van der Horst for all their support throughout my MSc. and Ph.D. studies. It is my great pleasure to work with you guys, and it will be a good memory in my life. Many thanks to all other MAVLab members: Sjoerd Tijmons, Ewoud Smeur, Matěj Karásek, Kimberly Mcguire, Kevin van Hecke, Kirk Scheper, Lodewijk Sikkel, Mario Coppola, Andries Koopmans, Freek van Tienen, Elisabeth van der Sman, Roland Meertens for all interesting discussions and support.

My thanks also go out to the support I received from all my colleagues in the division of control and simulation: Bob Mulder, Jacco Hoekstra, René van Paassen, Coen de Visser, Clark Borst, Joost Ellerbroek, Olaf Stroosma, Marilena Pavel, Alexander in ’t Veld, Hans Mulder, Ferdinand Postema, Andries Muis, Harold Thung, Alwin Damman, Gijsbert den Toom, Herman Koolstra, Maarten Tielrooij, Rolf Klomp, Gustavo Mercado Velasco, Jan Comans, Deniz Yilmaz, Jia Wan, Liguo Sun, Laurens van Eykeren, Rita Valente Pais, Herman Damveld, Yazdi Jenie, Jan Smisek, Tommaso Mannucci, Jaime Junell, Sophie Armanini, Tao Lu, Ye Zhang, Yingzhi Huang, Sherry Wang, Shuo Li, Sihao Sun, Wei Fu, Henry Tol, Tim Visser, Ivan Miletovic, Emmanuel Sunil, Kasper van der El, Annemarie Landman, Jerom Maas, João Caetano, Isabel Metz, Julia Rudnyk, Dirk Van Baelen, Sarah Barendswaard. Besides, a warm thank to Daan Pool, Dyah Jatiningrum, Junzi Sun for your support for my family. A special thank to Bertine Markus for your assistance in administrative works.

I gratefully acknowledge the funding I received from Malaysian Ministry of Higher Education (MOHE) and Universiti Sains Malaysia (USM) for giving me the opportunities to pursue my MSc. and Ph.D. studies in the Netherlands. Many thanks also to the USM staff members: Prof. ir. Dr. Mohd Zulkifly Abdullah, Prof. Dr. Zaidi Mohd Ripin, Farah Hamid, Saidatul Shakira Mohd Sani, Noren Asfarina Halim, Sarina Razak, Norazreen Razak, and Hajar Zituakmar Mohd Fauzi for offering their constant support to me and my family.

My profound gratitude goes out to my parent, Fok Peng Ho and Lay Boon Gan. Without your unconditional love and support, I could never reach this stage. Also, many thanks to my three sisters, Siew Tein Ho, Siew Chin Ho, and Siew Jye Ho for giving me support and taking good care of our parent when I was away from home these years.

Last but not least, I would like to convey my deepest gratitude to my wife, Ye Zhou for her unremitting encouragement, unconditional support, and unfailing love. Not to forget to thank my little one, Zi Xuan Ho for making me happy every day. Both of you are my ‘optical flow’, guiding me to avoid obstacles, find a safe landing site, and approach it smoothly.

Curriculum Vitæ

Hann Woei HO

05-09-1984 Born in Batu Pahat, Johor, Malaysia.

Education

2005–2009	Bachelor of Engineering in Aerospace Engineering (Hons.) School of Aerospace Engineering, Universiti Sains Malaysia (USM)
2010–2012	Master of Science in Aerospace Engineering (Distinction) Faculty of Aerospace Engineering, Delft University of Technology (TU Delft)
2013–2017	Ph.D. in Aerospace Engineering Faculty of Aerospace Engineering, Delft University of Technology (TU Delft) *Thesis:* Autonomous landing of Micro Air Vehicles through bio-inspired monocular vision *Promotor:* Prof. dr. ir. M. Mulder

Awards

2009	USM Gold Medal for the best final year student in Aerospace Engineering
2012	Distinction for MSc. in Aerospace Engineering, TU Delft
2013	Best Graduate Student Paper in CEAS EuroGNC
2013	First prize in International Micro Air Vehicle competition 2013 (IMAV 2013)

Experiences

2009	Production Engineer Shimano Components (M) SDN. BHD.
2010	Design Engineer Matromatic Handling Systems (M) SDN. BHD.

ACADEMIC ACTIVITIES

Reviewer for Applied Soft Computing

Reviewer for International Conference on Intelligent Robots and Systems (IROS)

Reviewer for International Micro Air Vehicle Conference and Competition (IMAV)

Reviewer for International Conference on Mechatronic and Embedded Systems and Applications (MESA)

Reviewer for International Conference on Unmanned Aircraft Systems (ICUAS)

LIST OF PUBLICATIONS

JOURNALS

4. G. C. H. E. de Croon, **H. W. Ho**, C. De Wagter, E. van Kampen, B. D. W. Remes, and Q. P. Chu, *Optic-flow based slope estimation for autonomous landing*, International Journal of Micro Air Vehicles, 5(4):287–298, 2013.

3. **H. W. Ho**, G. C. H. E. de Croon, and Q. P. Chu, *Distance and velocity estimation using optical flow from a monocular camera*, International Journal of Micro Air Vehicles (accepted).

2. **H. W. Ho**, G. C. H. E. de Croon, E. van Kampen, Q. P. Chu, and M. Mulder, *Adaptive gain control strategy for constant optical flow divergence landing*, IEEE transactions on robotics (in revision).

1. **H. W. Ho**, C. De Wagter, B. D. W. Remes, and G. C. H. E. de Croon, *Optical-flow based self-supervised learning of obstacle appearance applied to MAV landing*, The International Journal of Robotics Research (submitted).

CONFERENCE PROCEEDINGS

5. **H. W. Ho**, G. C. H. E. de Croon, and Q. P. Chu, *Distance and velocity estimation using optical flow from a monocular camera*, in International Micro Air Vehicle Conference and Competition 2016 (IMAV 2016) (Beijing, PR of China, 2016) p. 121-128.

4. **H.W. Ho** and G. C. H. E. de Croon, *Characterization of flow field divergence for MAVs vertical control landing*, in AIAA Guidance, Navigation, and Control Conference (San Diego, California, USA, 2016) p. 0106.

3. **H. W. Ho**, C. De Wagter, B. D. W. Remes, and G. C. H. E. de Croon, *Optical flow for self-supervised learning of obstacle appearanc*, in IEEE/RSJ International Conference on Intelligent Robots and Systems, 2015 (IROS 2015) (IEEE, Hamburg, Germany, 2015).

2. G. C. H. E. de Croon, **H. W. Ho**, C. De Wagter, E. van Kampen, B. D. W. Remes, and Q. P. Chu, *Optic-flow based slope estimation for autonomous landing*, in International Micro Air Vehicle Conference and Competition 2013 (IMAV 2013) (Ecole Nationale de l'Aviation Civile, Toulouse, France, 2013).

1. **H. W. Ho**, Q. P. Chu, *Automatic Landing System of a Quadrotor UAV Using Visual Servoing*, in CEAS EuroGNC conference 2013 (Delft University of Technology, Delft, The Netherlands, 2013) (Best Graduate Student Paper).

ISBN: 978-94-6186-818-3