Towards Robust Visual-Controlled Flight of Single and Multiple UAVs in GPS-Denied Indoor Environments

Dissertation
der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Volker Grabe
aus Hamburg

Tübingen
2014

Tag der mündlichen Qualifikation:	07.03.2014
Dekan:	Prof. Dr. Wolfgang Rosenstiel, Universität Tübingen
1. Berichterstatter:	Prof. Dr. Andreas Schilling, Universität Tübingen
2. Berichterstatter:	Prof. Dr. Davide Scaramuzza, Universität Zürich

Abstract

Having had its origins in the minds of science fiction authors, mobile robot hardware has become reality many years ago. However, most envisioned applications have yet remained fictional—a fact that is likely to be caused by the lack of sufficient perception systems. In particular, mobile robots need to be aware of their own location with respect to their environment at all times to act in a reasonable manner. Nevertheless, a promising application for mobile robots in the near future could be, e.g., search and rescue tasks on disaster sites. Here, small and agile flying robots are an ideal tool to effectively create an overview of the scene since they are largely unaffected by unstructured environments and blocked passageways.

In this respect, this thesis first explores the problem of ego-motion estimation for quadrotor Unmanned Aerial Vehicles (UAVs) based entirely on onboard sensing and processing hardware. To this end, cameras are an ideal choice as the major sensory modality. They are light, cheap, and provide a dense amount of information on the environment. While the literature provides camera-based algorithms to estimate and track the pose of UAVs over time, these solutions lack the robustness required for many real-world applications due to their inability to recover a loss of tracking fast. Therefore, in the first part of this thesis, a robust algorithm to estimate the velocity of a quadrotor UAV based on optical flow is presented. Additionally, the influence of the incorporated measurements from an Inertia Measurement Unit (IMU) on the precision of the velocity estimates is discussed and experimentally validated. Finally, we introduce a novel nonlinear observation scheme to recover the metric scale factor of the state estimate through fusion with acceleration measurements. This nonlinear model allows now to predict the convergence behavior of the presented filtering approach. All findings are experimentally evaluated, including the first presented human-controlled closed-loop flights based entirely on onboard velocity estimation.

In the second part of this thesis, we address the problem of collaborative multi robot operations based on onboard visual perception. For instances of a direct line-of-sight between the robots, we propose a distributed formation control based on ego-motion detection and visually detected bearing angles between the members of the formation. To overcome the limited field of view of real cameras, we add an artificial yaw-rotation to track robots that would be invisible to static cameras. Afterwards, without the need for direct visual detections, we present a novel contribution to the mutual localization problem. In particular, we demonstrate a precise global localization of a monocular camera with respect to a dense 3D map. To this end, we propose an iterative algorithm that aims to estimate the location of the camera for which the photometric error between a synthesized view of the dense map and the real camera image is minimal.

Zusammenfassung

Mobile Roboter sind schon länger zur Realität geworden, nachdem sie ursprünglich Sciencefiction Romanen entstammen. Trotzdem sind viele der vorausgesagten Anwendungen bis heute nicht Teil unseres täglichen Lebens geworden. Ein wichtiger Grund hierfür ist die noch immer unzureichende Fähigkeit der Roboter die vielen durch Sensoren gewonnenen Information über die Umgebung sinnvoll zu verarbeiten. Dies betrifft insbesondere die Fähigkeit eines Roboters seine eigene Position im Raum zu berechnen, ohne die sinnvolle Bewegungen oder eine Interaktion mit der Umgebung nicht möglich sind. Erfolgversprechende Anwendungsgebiete für Mobile Roboter könnten in naher Zukunft jedoch Einsätze zur Suche, Bergung und Rettung nach Katastrophen sein. Für dieses Einsatzszenario eignen sich insbesondere kleine und agile Flugroboter, die nicht durch am Boden liegende Trümmerteile behindert werden können.

Diese Arbeit befasst sich daher mit dem Problem der Erkennung der Eigenbewegung von Quadrokoptern (hubschrauber-ähnliche Fluggeräte mit vier Propellern) ausschließlich auf Basis von Systemen, deren Sensoren und Recheneinheiten vollständig auf dem Flugroboter installiert sind. Als hauptsächliche Sensoren bieten sich hier insbesondere Kameras an, da sie leicht und günstig sind und dabei umfassende Informationen über die Umgebung liefern. Während bisherige auf Kameras basierende Algorithmen hauptsächlich darauf ausgelegt sind die Position des Quadrokopters zu bestimmen, sind diese Verfahren, auf Grund ihrer mangelnden Fähigkeit eine einmal verlorene Positionsschätzung schnell wieder auszugleichen, für viele Anwendungsgebiete ungeeignet. Der erste Teil dieser Arbeit befasst sich daher mit einem Verfahren um die aktuelle Geschwindigkeit des Flugroboters auf Basis von Optischem Fluss zu berechnen. Darüber hinaus wird untersucht, inwieweit die Einbeziehung von Messwerten der internen Rotations- und Beschleunigungssensoren zu einer Verbesserung der Geschwindigkeitsschätzung führt. Abschließend wird ein nicht-linearer Beobachter vorgestellt, der den metrischen Skalierfaktor mit Hilfe der kamera-basierten Messungen sowie der Beschleunigungssensoren berechnet. Dieses System erlaubt damit erstmals die Vorhersage der temporalen Eigenschaften des Filters. Die anschließende ausführliche experimentelle Evaluierung beinhaltet zudem den ersten dokumentierten Flug eines Quadrokopters basierend auf ausschließlich direkt auf dem Roboter berechneten Geschwindigkeitsinformationen.

Im zweiten Teil der Arbeit wird dann das Problem der Kollaboration von mehreren Robotern basierend auf visuellen Sensoren untersucht. Für Fälle einer direkten Sichtverbindung zwischen den Robotern wird ein Algorithmus zur Steuerung der Roboterformation vorgestellt, der auf der visuellen Erkennung der Eigengeschwindigkeit und der Winkel zwischen den einzelnen Robotern der Formation basiert. Um zudem Limitierungen realer Kameras bedingt durch ein natürlicherweise beschränktes Sichtfeld auszugleichen, schlagen wir eine zusätzliche horizontale Rotation des Quadrokopters vor um auch ansonsten außerhalb des Sichtfeldes befindliche Roboter detektieren zu können. Anschließend, für den Fall, dass kein Sichtkontakt zwischen den Robotern besteht, wird ein Verfahren zur präzisen Lokalisierung einer einzelnen Kamera in einer dreidimensionalen Karte präsentiert. Unser iterativer Algorithmus versucht dabei die Position der Kamera so zu bestimmen, dass der Unterschied zwischen dem Bild der Kamera sowie einem synthetisierten Bild der Karte von der gleichen Position möglichst minimal ist.

Acknowledgements

The submission and defense of this thesis highlights the end of three years of very intense work, many important insights, and many fruitful discussions that have influenced my work to great extents. Clearly, ambitious research projects cannot be accomplished by oneself. Therefore, I would like to thank all colleagues, collaborators, and friends who shared their time and ideas with me over the last years.

In particular, I would like to thank my supervisor Dr. Paolo Robuffo Giordano for all his support, advice, and often also patience. Similarly, I would like to express my gratitude to Prof. Davide Scaramuzza for hosting my visit in his lab at the University of Zürich that eventually extended to the full supervision of the second half of my PhD. Both, Paolo and Davide were available for clarifying discussions even late at night. I feel privileged to have been supervised by them.

On similar lines, I like to thank Prof. Heinrich Bülthoff for having invited me to become part of his group at the Max-Planck Institute for Biological Cybernetics. He created an exceptional open research environment that made the time of my PhD studies a memorable experience. Furthermore, I like to thank Prof. Andreas Schilling for co-supervising my work and for his many great ideas and suggestions. I am grateful to all members of my defense committee Prof. Andreas Schilling, Prof. Davide Scaramuzza, Prof. Andreas Zell, and Prof. Heinrich Bülthoff for their availability and their interest in my work.

I like to express particular gratitude to Dr. Antonio Franchi for many helpful discussions and for sharing his knowledge with me. Although he was never required to supervise and assist my work, he nevertheless always offered a helping hand and provided valuable suggestions when necessary.

From my co-workers in both Tübingen and Zürich, special thanks are meant to my co-authors, in particular, Christian Forster, Dr. Antonio Franchi, Carlo Masone, Martin Riedel, and Markus Ryll. I like to thank them and all the other numerous present and past colleagues for the many joint fruit- and cheerful moments in the last years.

Last, I like to thank Dr. Katharina Mülling for having supported my work and life as both my friend and partner in all of the last years.

Contents

List of Abbreviations

2D / 3D / 6D	Two / Three / Six Dimensions
CPU	Central Processing Unit
DARPA	Defense Advanced Research Projects Agency
DoF / DoFs	Degree of Freedom / Degrees of Freedom
EKF	Extended Kalman Filter
FAST	Features from Accelerated Segment Test
FOV	Field of View
GNU	GNU's Not Unix
GPS	Global Positioning System
GPU	Graphics Processing Unit
HSV	Hue, Saturation and Value
ICP	Iterative Closest Point
IMU	Inertia Measurement Unit
LED	Light Emitting Diode
MAV	Micro Aerial Vehicle
NED	North-East-Down coordinate frame convention
P3P	Perspective-3-Point
PC	Personal Computer
PE	Persistency of Excitation
PCI	Peripheral Component Interconnect
PID	Proportional-Integral-Derivative
PTAM	Parallel Tracking And Mapping
px	PiXel
RAM	Random-Access Memory
RANSAC	RANdom SAmple Consensus
RGB	Red, Green, Blue
RGBD	Red, Green, Blue, Depth
RMS / RMSE	Root Mean Square / Root Mean Square Error
ROM	Read-Only Memory
ROS	Robot Operating System
SD	Secure Digital
SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization And Mapping
UAV	Unmanned Aerial Vehicle
USB	Universal Serial Bus
WiFi	Trademark for WLAN products
WLAN	Wireless Local Area Network
W.l.o.g.	Without loss of generality

List of Variables

Conventions

a, b	Scalar
$\boldsymbol{a}, \boldsymbol{b}$	Vector
$\boldsymbol{A}, \boldsymbol{B}$	Matrix
A, B	Set or image
$\mathcal{A}, \mathcal{B}$	Coordinate frame
${}^{\mathcal{A}}\boldsymbol{b}$	Vector b expressed in frame A
${}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{AB}}$	Translation vector between frame $\mathcal{A}$ and $\mathcal{B}$ expressed in frame $\mathcal{C}$
${}^{\mathcal{A}}\boldsymbol{p}_{\mathcal{B}}$	Translation vector between frame $\mathcal{A}$ and $\mathcal{B}$ expressed in frame $\mathcal{A}$
${}^{\mathcal{A}}[\boldsymbol{R}\|\boldsymbol{t}]_{\mathcal{B}}$	Roto-Translation from frame $\mathcal{A}$ to $\mathcal{B}$
${}^{\mathcal{A}}\boldsymbol{R}_{\mathcal{B}}$	Rotation from frame $\mathcal{A}$ to $\mathcal{B}$
a_m	Quantity a as provided by a (noisy) sensor
$\hat{a}$	Estimate of quantity a
$\dot{a}$	Temporal derivative of quantity a
$[\boldsymbol{a}]_\times \in \mathbb{R}^{3\times3}$	Skew symmetric matrix associated to vector $\boldsymbol{a} \in \mathbb{R}^3$ such that $a \times b = [\boldsymbol{a}]_\times b$
$\|\boldsymbol{a}\| \in \mathbb{R}$	Norm of vector $\boldsymbol{a}$

Most important variables used globally throughout this thesis

Latin symbols

$\boldsymbol{a} \in \mathbb{R}^3$	Linear acceleration
$d \in \mathbb{R}$	Distance from the (planar) environment
$\boldsymbol{e}$	Estimation error
$\boldsymbol{g} \in \mathbb{R}^3$	Vector of the gravitational acceleration, $\|\boldsymbol{g}\| \approx 9.81$
$\boldsymbol{H}$	Continuous homography matrix
$\boldsymbol{I}_i \in \mathbb{R}^{i\times i}$	Identity matrix of dimension i
I	Image, defined as two dimensional array of monochrome pixel intensities
$\boldsymbol{K}$	Kalman gain matrix
$K_a \in \mathbb{R}$	Gain of quantity or component a
$N \in \mathbb{R}$	Number of features or agents
$\boldsymbol{n} \in \mathbb{R}^3$	Normal vector of a plane, $\|\boldsymbol{n}\| = 1$
$\boldsymbol{s}$	State vector of a filter
$\boldsymbol{u} \in \mathbb{R}^3$	Flow vector of a feature on the image plane, $\boldsymbol{u}_z = 0$
$\boldsymbol{v} \in \mathbb{R}^3$	Linear velocity
$\boldsymbol{X} \in \mathbb{R}^3$	Location of a visual feature in the environment
$\boldsymbol{x} \in \mathbb{R}^3$	Location of a pixel or visual feature on the image plane, $\boldsymbol{x}_z = 1$

Greek symols

$\Delta_t \in \mathbb{R}$	Temporal difference between two measurements
$\mathbf{\Sigma}$	Uncertainty matrix or diagonal matrix with singular values
Φ	Set of optical flow vectors
$\mathbf{\Omega}$	Matrix that encapsulates all acceleration components
Ω	Set of all pixels suitable for the comparison of two images
$\boldsymbol{\omega} \in \mathbb{R}^3$	Angular velocity

All other symbols are only defined locally for one section and might be used in other sections in a different context. This allowed the authors to adopt to common notations and symbols whenever possible.

1. Introduction

While stationary robotic arms have already become an essential part of many factories, the use of many mobile robots and in particular Unmanned Aerial Vehicles (UAVs) is still mostly limited to academic research. Opposed to stationary robots, mobile robots have to master the perception-to-action loop for a wide range of environments. Therefore, robust pose estimation systems are an essential component for all mobile robot applications.

Overcoming current limited perception systems by human-in-the-loop setups, in the last years, search and rescue robots became an application of increasing importance for civil mobile robots. In these scenarios, e.g., those staged for the DARPA Robotics Challenge Trials in December 2013[1], a robot is remotely commanded through a hazardous environment that is unsafe for direct human intervention. However, disaster sites might be challenging to navigate. A robot would have to pass through narrow indoor passages, overcome debris, and potentially climb stairs or ladders in multi story buildings. To overcome these challenges, small and agile flying vehicles capable of hovering flight make an ideal choice for initial investigations. In fact, small-size UAVs, such as quadrotors, have been used to examine disaster sites after earthquakes, such as, Christchurch (New Zealand, 2011) and Emilia Romagna (Italy, 2012), and most prominently at the Fukushima Daiichi power plant in Japan, 2011. Similarly, quadrotors have been used for the inspection of otherwise inaccessible places, e.g., industrial scale furnaces.

In particular the high agility of quadrotor UAVs as opposed to other flying vehicles has attracted the scientific community in the last five years. Several control challenges have been addressed, making the quadrotor are well studied platform for even aggressive and yet precise maneuvers, e.g., sideways open-loop flights through narrow openings [Mellinger et al., 2012] or complex closed-loop tasks [Müller et al., 2011].

However, in all these works, the perception problem was omitted by using stationary external motion tracking systems with sub-millimeter accuracy. Without the availability of pre-calibrated motion tracking systems at unstructured disaster sites, UAVs are often remotely controlled from the save distance by highly trained human operators using visual feedback streamed from the vehicle. While a human-in-the-loop design might be desirable for many applications given a humans high situational awareness and superior reasoning, the need for extensive training to operate a UAV clearly limits the range of applications for quadrotors.

Therefore, the development of onboard perception systems marks an essential requirement to overcome the limitations of current computer or human controlled quadrotor systems. These constraints have motivated the extensive use of onboard cameras as main sensory modality. They combine a low weight and low costs with a rich sensory feedback. However, in general, information obtained from optical systems lacks a metric scale and thus needs to be fused with metric observations from other sensors.

In the last years, several camera-based pose estimation systems have been proposed that make use of Simultaneous Localization And Mapping (SLAM) techniques to build and track

[1] `http://www.theroboticschallenge.org/`

a map. However, these approaches require a reliable and constant tracking of optical features over an extended amount of time. This requirement makes these systems susceptible to even short tracking failures caused by occluded views, moving objects, or blurry camera images that, if possible at all, cause a timely recovery of the last pose. During this time, the position of the unstable quadrotor system cannot be regulated and the flight behavior becomes unpredictable. On the contrary, this is not the case when relying on ego-motion estimation from optical-flow, as, in this case, data extraction and association is performed on each two consecutively acquired images. The resulting velocity estimates can be used for the implementation of a velocity based controller that can be intuitively used by human operators.

Therefore, in this thesis, we first aim for the development of ego-motion estimation systems based entirely on onboard perception. They allow even untrained humans to operate a quadrotor UAV using low level velocity commands. In the second part of this thesis, several approaches on the next step towards vision aided multi robot collaboration tasks are discussed.

1.1. Summary of Contributions

Our work mainly contributes to two aspects of academic research. In the following, we detail the main contributions in both fields.

1.1.1. Ego-Motion Estimation from Optical Flow

For many years, control of quadrotor UAVs was mostly based on external state estimation systems, such as GPS, or external processing on ground stations. In contrast, in this thesis, we developed and tested algorithms for the control of UAVs relying purely on velocity estimation from onboard sensors and processing. In fact, we were able to present the first results from a closed-loop controlled flight of a quadrotor UAV based on velocity estimates from an all-onboard vision system. For this purpose, we developed and compared three implementations to extract velocity information from optical flow. Additionally, we presented algorithms to filter outlying observations of the vision system to improve the state estimate. Finally, we also introduced a novel non-linear sensor-fusion filter that allows to fully characterize the systems convergence behavior. Thus, it is now possible to predict the error of the metric scale factor and therefore also the velocity estimate for a given flight trajectory. Our publications inspired several other research labs to engage in similar research activities.

Our contributions in this field have led to the following publications:

- Grabe, V., Scaramuzza, D., Robuffo Giordano, P. (2014). Nonlinear Ego-Motion Estimation from Optical Flow for Online Control of a Quadrotor UAV. International Journal of Robotics Research (under review).
- Grabe, V., Bülthoff, H. H., Robuffo Giordano, P. (2013). A Comparison of Scale Estimation Schemes for a Quadrotor UAV based on Optical Flow and IMU Measurements. In Proceedings of the International Conference on Intelligent Robots and Systems. Tokyo, Japan.
- Grabe, V., Bülthoff, H. H., Robuffo Giordano, P. (2012). Robust Optical-Flow Based Self-Motion Estimation for a Quadrotor UAV. In Proceedings of the International Conference on Intelligent Robots and Systems. Vilamoura, Portugal.

- Grabe, V., Bülthoff, H. H., Robuffo Giordano, P. (2012). On-board Velocity Estimation and Closed-loop Control of a Quadrotor UAV based on Optical Flow. In Proceedings of the International Conference on Robotics and Automation. St. Paul, MN, USA.

1.1.2. Mutual Robot Localization based on Visual Observations

If available, mutual localization of several robots can be achieved using direct visual observations among the robots. Otherwise, robots can localize themselves with respect to other robots in an indirect way through the exchange of coordinates in a common reference frame. To the former, we contributed a formation control that is based on the direct visual bearing observation of other flying robots. To the latter, we presented a novel algorithm to precisely localize a camera within a dense map using analysis-by-synthesis techniques. This novel approach of localization of one sensor with respect to a different other is then extended with further project descriptions in the future work section.

Our contributions in this field have led to the following publications:

- Grabe, V., Scaramuzza, D. (2014). Global Localization of a Monocular Camera in RGB 3D Maps through Photometric-Error Minimization. Technical Report.
- Franchi, A., Masone, C., Grabe, V., Ryll, M., Bulthoff, H. H., Rubuffo Giordano, P. (2012). Modeling and Control of UAV Bearing Formations with Bilateral High-level Steering. The International Journal of Robotics Research, 31(12), 1504–1525.
- Grabe, V., Masone, C., Ryll, M., Franchi, A., Bulthoff, H. H., Rubuffo Giordano, P. (2011). Implementation of a Decentralized Formation Control based on Visual Bearing Angles for Multiple UAVs. Technical Report.

1.1.3. Additional Contributions

Most of our work relies on the free TeleKyb framework that has been released to the public. This contribution is not discussed in detail as part of this thesis since several authors were strongly involved in the design and development of this software framework. It is documented in the following publication:

- Grabe, V., Riedel, M., Bülthoff, H. H., Robuffo Giordano, P., Franchi, A. (2013). The TeleKyb Framework for a Modular and Extendible ROS-based Quadrotor Control. In Proceedings of the European Conference on Mobile Robots. Barcelona, Spain.

1.2. Organization of the Thesis

This thesis is organized along the two main topics highlighted in Sec. 1.1. Additionally, Figure 1.1 provides an overview of the addressed problems.

In Chapter 2, we first discuss approaches for the control of a single UAV based on metric velocity observations by means of onboard cameras. In particular, we address the problem of velocity estimation from optical flow in Sec. 2.2, outlier rejection approaches in Sec. 2.3, and online scale estimation techniques in Sec. 2.4.

In Chapter 3, we then expand the state estimation problem to multiple UAVs and discuss two selected approaches to deal with the mutual robot localization problem. In Sec. 3.2, we

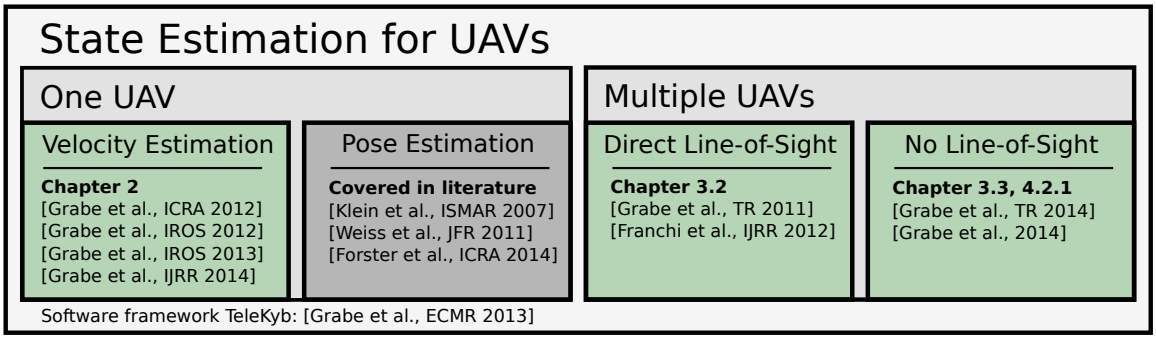


Figure 1.1.: Visualized organization of the thesis. The problem of onboard state estimation for UAVs is investigated for both individual UAVs and groups of quadrotors. For the first subproblem, we contribute to the field of velocity estimation while the problem of pose estimation is extensively covered in the literature. Approaches to the mutual localization problem are proposed for both an existing and an occluded line-of-sight between the UAVs. Publications listed for 2014 were under review or in preparation as of March 2014.

propose a formation controller based on the direct visual detection of other flying robots nearby. Conversely, in Sec. 3.3, we present an indirect way of localization without the availability of a direct line-of-sight between the robots. Here, one robot with a camera is localized within a dense map created by a second robot.

Finally, the thesis is concluded in Chapter 4 including a detailed outlook to future projects that extend our work on multi robot cooperation.

2. Ego-Motion Estimation from Optical Flow for Online Control of a Quadrotor UAV

Summary

For the control of Unmanned Aerial Vehicles (UAVs) in GPS-denied environments, cameras have been widely exploited as main sensory modality for a successful UAV state estimation. However, the use of visual information for ego-motion estimation presents several theoretical and practical difficulties, such as, data association, occlusions, and the lack of direct metric information when relying on monocular cameras. In this chapter, we address all these issues in two ways. First, we propose a robust ego-motion estimation algorithm to recover the (non-metric) linear and angular UAV velocities from optical flow by exploiting the *continuous homography constraint* in presence of planar regions. Then, we address the problem of retrieving the (unknown) metric scale by fusing the visual information with the onboard Inertia Measurement Unit (IMU). To this end, two different estimation strategies are proposed and critically compared: a first one exploiting the classical Extended Kalman Filter (EKF) formulation, and a second one based on a novel nonlinear estimation framework. The main advantage of the latter scheme lies in the possibility of imposing a desired transient response to the estimation error, which is typically not possible with an EKF due to its inherent linearization of the system dynamics. We indeed show that the nonlinear scheme yields considerably superior performance in terms of convergence rate and predictability of the estimation error behavior. The chapter is then concluded by an extensive experimental validation, including all-onboard closed-loop control of a real quadrotor UAV in real-world conditions.

The work described in this chapter has been published in:

- Grabe, V., Bülthoff, H. H., Robuffo Giordano, P. (2012). On-board Velocity Estimation and Closed-loop Control of a Quadrotor UAV based on Optical Flow. In Proceedings of the International Conference on Robotics and Automation. St. Paul, MN, USA.
- Grabe, V., Bülthoff, H. H., Robuffo Giordano, P. (2012). Robust Optical-Flow Based Self-Motion Estimation for a Quadrotor UAV. In Proceedings of the International Conference on Intelligent Robots and Systems. Vilamoura, Portugal.
- Grabe, V., Bülthoff, H. H., Robuffo Giordano, P. (2013). A Comparison of Scale Estimation Schemes for a Quadrotor UAV based on Optical Flow and IMU Measurements. In Proceedings of the International Conference on Intelligent Robots and Systems. Tokyo, Japan.
- Grabe, V., Scaramuzza, D., Robuffo Giordano, P. (2014). Nonlinear Ego-Motion Estimation from Optical Flow for Online Control of a Quadrotor UAV. International Journal of Robotics Research (under review).

2.1. Introduction

In recent years, inspection, search, and rescue tasks have become one of the most important envisaged applications for Unmanned Aerial Vehicles (UAVs). For instance, small-size UAVs, such as quadrotors, have been used to investigate disaster sites after earthquakes, e.g., the Fukushima Daiichi power plant in Japan. Along similar lines, some large-scale research projects have been recently funded on these and related topics, see, e.g., [EU Collaborative Project ICT-248669; EU Collaborative Project ICT-287617; EU Collaborative Project ICT-600958]. Indeed, thanks to their high agility, pervasiveness and customizability, quadrotors represent an ideal robotic platform for navigating in harsh and cluttered environments, either when operating in full autonomy, or under the (partial) remote control of skilled human operators.

In all cases, a widely-recognized key component for the successful deployment of such systems is a reliable state estimation module able to deal with highly unstructured and/or GPS-denied indoor environments. This typically imposes a major requirement on the system: since the target environment for the considered applications cannot be prepared before the deployment of the vehicle, the UAV is constrained to only rely on onboard sensing and processing capabilities. These constraints have motivated, over the last years, the extensive use of onboard cameras as main sensory modality for state estimation purposes, see [Weiss et al., 2013a; Scaramuzza et al., 2014] for a recent overview. Vision indeed provides a rich sensory feedback which could, in principle, yield a full understanding of the surrounding environment. However, an effective use of the visual information also presents many theoretical and practical difficulties. For instance, robust data extraction and association across multiple frames is often a major issue in real-world scenarios, especially when comparing images from non-consecutive points of view. Also, when relying on monocular onboard cameras, any position/velocity information can only be retrieved up to an arbitrary scale factor, which must then be disambiguated by fusing the visual information with independent metric measurements from other onboard sensors. This has motivated many recent works on data fusion exploiting the concurrent metric measurements from onboard accelerometers embedded in the Inertia Measurement Units (IMUs) present on most flying robotic systems [Martinelli, 2012; Li and Mourikis, 2013; Omari and Ducard, 2013].

Existing work on *metric* camera-based state estimation mostly relies on SLAM techniques to build and maintain a map of visual features while the position of the vehicle in the map is estimated in parallel. Acceleration measurements are then used to reconstruct the metric scale of the underlying map [Martinelli, 2012]. This has been achieved in [Nützi et al., 2011] for the well-known Parallel Tracking and Mapping (PTAM) algorithm [Klein and Murray, 2007], and, even more remarkably, from the observation of just a single feature over time by providing a closed-form solution for the computation of the metric scale factor [Kneip et al., 2011a; Martinelli, 2012]. However, all these approaches depend on the possibility to continuously track features over an extended period of time. Therefore, the level of robustness required for the UAV control cannot be guaranteed as the visual system could be affected by, e.g., unexpected occlusions, blurry images, or the need of heavy computations for a reliable feature matching. This is not the case, however, when relying on motion estimation from *optical flow*, as, in this case, data extraction and association is performed on consecutive (and thus spatially very near) acquired images. Motivated by these considerations, a first contribution of this work is then the development of an approach based on optical-flow decomposition for providing a reliable and robust ego-motion estimation module for safe UAV operation in unstructured environments.

Exploiting purely optical flow, a system capable of hovering and landing on a moving platform was presented in [Herissé et al., 2012]. However, this work did not consider the issue of determining the unknown scene scale factor since the proposed control approach was implemented by only relying on the non-metric linear velocity directly obtained from a monocular camera. An approach combining optical flow decomposition and sensor fusion for metric scale estimation was instead presented in [Honegger et al., 2013] by developing a small sensor for velocity estimation onboard UAVs. However, the proposed sensor relied on a ground facing sonar for metric scale estimation, thus limiting the vehicle to near ground operations within the range of the sonar. Finally, by exploiting an EKF to fuse optical flow measurements with IMU readings, in [Weiss et al., 2012b], the authors demonstrated the possibility of estimating the metric scale, sensor biases, and the IMU/camera relative pose. This system was later extended in [Weiss et al., 2013b]. However, the system was mainly designed to support the initialization of the PTAM framework in near hovering mode rather than for closed-loop UAV control over an extended period of time.

It can be noted that most of the presented camera-based state estimation methods rely on the classical EKF framework to fuse together the available sensor readings (e.g., from vision and IMU) for then extracting the metric scale. However, despite being widespread, the use of a EKF-based scheme presents two major (and often overlooked) drawbacks: (*i*) it necessarily involves a linearization of the (nonlinear) system dynamics (such as when dealing with visual-inertial estimation problems), and (*ii*) as a consequence, it does not allow for an explicit characterization of the estimation error behavior. Therefore, we propose a novel visual-inertial estimation scheme exploiting optical flow and IMU measurements based on a recently-developed nonlinear observation framework for active structure from motion [Spica and Robuffo Giordano, 2013]. Compared to a classical EKF, the use of our nonlinear filter yields an estimation error dynamics with a *fully characterized convergence behavior*, in particular equivalent to that of a reference linear second-order system with assigned poles. It is then possible, for instance, to *actively impose* a desired error transient response by suitably acting on the estimation gains and on the UAV motion. Similarly, the convergence time of the estimation error can be predicted in terms of percentages of the initial error. Finally, the reported results also show that the use of the proposed nonlinear filter yields a substantial faster (and more controlled) error convergence compared to a classical and 'fully-informed' EKF, thus making it a viable and robust alternative to other consolidated schemes.

The rest of the chapter is then structured as follows: in Sec. 2.2 we first review the proposed ego-motion estimation algorithm from optical flow which provides a (non-metric) estimation of the UAV linear/angular velocity. Subsequently, Sec. 2.4 introduces two estimation schemes meant to recover the unknown scale factor by fusing the visual information with the IMU readings: a filter based on the standard EKF machinery, and a novel deterministic nonlinear observer. The two filters are then compared by highlighting, for the latter, the possibility of characterizing and actively shaping its error transient response. Afterwards, Sec. 2.5 reports and discusses the results of several simulations and experiments aimed at validating and comparing the two ego-motion estimation approaches, and, finally, some experiments of closed-loop control on a quadrotor UAV are presented. Section 2.6 then concludes the chapter and discusses some open points and future directions.

2.2. Ego-Motion Estimation from Optical Flow

The adopted approach for ego-motion estimation from perceived optical flow is based on the decomposition of the *continuous homography matrix* [Ma et al., 2004], complemented with the angular velocity measurements obtainable from an onboard IMU.

A distinguishing feature of our method with respect to most of the previous literature is the use of a *continuous approach* for motion recovery. In fact, the typical incremental ego-motion estimation algorithms (e.g., the *visual odometry* [Scaramuzza and Fraundorfer, 2011]) assume presence of small but *finite* camera displacements over frames, and are thus based on reconstruction methods involving the *discrete* epipolar/homography constraints. However, since most cameras acquire images at high rates, we judged more appropriate the adoption of a *continuous* approach to recover, at each step, the camera instantaneous linear/angular velocity rather than a finite displacement over time.

In our experimental setup, we assume a down-facing camera in an approximately flat outdoor or indoor scenario. Features on the ground are tracked between any two consecutive frames to compute an optical flow field $\Phi = ((\boldsymbol{x}_1, \boldsymbol{u}_1), \ldots, (\boldsymbol{x}_N, \boldsymbol{u}_N))$ as function of N pairs $(\boldsymbol{x}_i, \boldsymbol{u}_i)$ of detected features $\boldsymbol{x}_i \in \mathbb{R}^3$ on the image plane and associated image velocities $\boldsymbol{u}_i \in \mathbb{R}^3$. In indoor hallways as well as in many outdoor scenarios and during flight at greater heights, one can safely assume that most tracked feature are located on a horizontal ground plane. This assumption is exploited first by reviewing an algorithm to retrieve the linear and angular velocities $(\boldsymbol{v} \in \mathbb{R}^3, \boldsymbol{\omega} \in \mathbb{R}^3)$ of a moving camera based on the continuous four-point algorithm for planar scenes [Ma et al., 2004]. We then show how to extend it in the cases of (i) known angular velocities from onboard gyroscopes, and (ii) additional known direction of the ground plane normal vector $\boldsymbol{n}$. This is motivated by the fact that, apart from a measurement of $\boldsymbol{\omega}$, typical IMUs are also able to sense the local direction of gravity $\boldsymbol{g}$ which, given our assumptions, coincides with the ground plane normal vector.

Consequently, we then present an experimental validation and thorough comparison of these three methods against a known ground truth. Finally, we show the experimental results of using our proposed solutions as a feedback term for closed loop control of a quadrotor UAV.

In a first step, we now review the classical reconstruction algorithm based on the continuous homography constraint [Ma et al., 2004].

2.2.1. Review of the Continuous Homography Constraint

Seen from a moving camera, the apparent velocity of a static point in space $\boldsymbol{X} \in \mathbb{R}^3$ as a result of the camera motion is

$$\dot{\boldsymbol{X}} = [\boldsymbol{\omega}]_\times \boldsymbol{X} + \boldsymbol{v} \tag{2.1}$$

where $\boldsymbol{v} \in \mathbb{R}^3$, $\boldsymbol{\omega} \in \mathbb{R}^3$ are the camera linear/angular velocity (all expressed in the camera frame), and $[\boldsymbol{\omega}]_\times \in so(3)$ is the skew-symmetric matrix associated to vector $\boldsymbol{\omega} \in \mathbb{R}^3$ such that $[\boldsymbol{\omega}]_\times \boldsymbol{X} = \boldsymbol{\omega} \times \boldsymbol{X}$ where $\times$ denotes the cross product in $\mathbb{R}^3$.

Consider a set of point features located on a common plane of equation $\boldsymbol{n}^T \boldsymbol{X} = d$ where $\boldsymbol{n} \in \mathbb{S}^2$ is the unit normal vector to the plane, and $d \in \mathbb{R}$ the orthogonal distance of the plane

to the camera frame. By rearranging the plane constraint as $\frac{1}{d}\boldsymbol{n}^T\boldsymbol{X} = 1$, eq. (2.1) becomes

$$\dot{\boldsymbol{X}} = [\boldsymbol{\omega}]_\times \boldsymbol{X} + \boldsymbol{v}\frac{1}{d}\boldsymbol{n}^T\boldsymbol{X} = \left([\boldsymbol{\omega}]_\times + \frac{1}{d}\boldsymbol{v}\boldsymbol{n}^T\right)\boldsymbol{X} = \boldsymbol{H}\boldsymbol{X}. \tag{2.2}$$

Matrix $\boldsymbol{H} \in \mathbb{R}^{3\times3}$ is commonly referred to as the *continuous homography matrix*: it encodes the camera linear/angular velocity ($\boldsymbol{v}$, $\boldsymbol{\omega}$), as well as the scene structure ($\boldsymbol{n}$, d).

Defining $\lambda\boldsymbol{x} = \boldsymbol{X}$ for a scalar depth factor $\lambda \in \mathbb{R}$ as the image of a point $\boldsymbol{X}$, and exploiting the fact that $\dot{\boldsymbol{X}} = \dot{\lambda}\boldsymbol{x} + \lambda\boldsymbol{u}$ and $\dot{\boldsymbol{x}} = \boldsymbol{u}$, where $\boldsymbol{u}$ is the *observed* velocity of the point $\boldsymbol{x}$ on the image plane, one obtains

$$\boldsymbol{u} = \boldsymbol{H}\boldsymbol{x} - \frac{\dot{\lambda}}{\lambda}\boldsymbol{x}. \tag{2.3}$$

The depth factor λ in (2.3) can be removed by pre-multiplication of $[\boldsymbol{x}]_\times$ (note that $[\boldsymbol{x}]_\times b\boldsymbol{x} = 0$ for any vector $\boldsymbol{x} \in \mathbb{R}^3$ and scalar $b \in \mathbb{R}$). This results in the so-called *continuous homography constraint* [Ma et al., 2004]

$$[\boldsymbol{x}]_\times \boldsymbol{H}\boldsymbol{x} = [\boldsymbol{x}]_\times \boldsymbol{u} \tag{2.4}$$

which involves the measured ($\boldsymbol{x}$, $\boldsymbol{u}$) and the (unknown) continuous homography matrix $\boldsymbol{H}$.

2.2.2. The 4-Point Algorithm

Matrix $\boldsymbol{H}$ in (2.4) can be recovered from a set of N measured pairs ($\boldsymbol{x}_i$, $\boldsymbol{u}_i$) of detected features $\boldsymbol{x}_i \in \mathbb{R}^3$ on the image plane and associated feature velocities $\boldsymbol{u}_i \in \mathbb{R}^3$. The elements of $\boldsymbol{H}$ are stacked into a vector $\boldsymbol{H}^S = [H_{11}, H_{21}, \cdots, H_{33}] \in \mathbb{R}^9$ and one can rewrite (2.4) as

$$\boldsymbol{a}_i^T \boldsymbol{H}^S = [\boldsymbol{x}_i]_\times \boldsymbol{u}_i \tag{2.5}$$

where $\boldsymbol{a}_i = \boldsymbol{x}_i \otimes [\boldsymbol{x}_i]_\times \in \mathbb{R}^{9\times3}$, and $\otimes$ stands for the Kronecker product. By then defining $\boldsymbol{A} = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_N]^T \in \mathbb{R}^{3N\times9}$ and $\boldsymbol{B} = [[\boldsymbol{x}_1]_\times \boldsymbol{u}_1, \cdots, [\boldsymbol{x}_N]_\times \boldsymbol{u}_N]^T \in \mathbb{R}^{3N}$, one obtains the linear system

$$\boldsymbol{A}\boldsymbol{H}^S = \boldsymbol{B}. \tag{2.6}$$

Assuming presence of at least $N \geq 4$ detected feature pairs ($\boldsymbol{x}_i$, $\boldsymbol{u}_i$), system (2.6) can be solved in a least-square sense as $\boldsymbol{H}^S = \boldsymbol{A}^\dagger\boldsymbol{B}$, with $\boldsymbol{A}^\dagger$ denoting the pseudo-inverse of matrix $\boldsymbol{A}$.

After having recovered $\boldsymbol{H}$, using standard techniques [Ma et al., 2004], it is further possible to algebraically decompose it into the scaled linear velocity $\boldsymbol{v}/d$, the angular velocity $\boldsymbol{\omega}$, and the plane normal $\boldsymbol{n}$. However, it can be shown that, in general, two physically-equivalent solutions are compatible with a given homography matrix $\boldsymbol{H}$ [Ma et al., 2004].

This first algorithm then allows us to estimate the quantities ($\boldsymbol{v}/d$, $\boldsymbol{\omega}$, $\boldsymbol{n}$) from two consecutive visual optical flow observations and without the use of additional sensor readings or previously acquired data structures such as, e.g., a map. This first version of the ego-motion recovery algorithm is referred to as *V1* in all of the following developments.

2.2.3. Exploiting a Known Angular Velocity

Since any typical onboard IMU can directly measure the angular velocity $\boldsymbol{\omega}_{IMU}$, we can consider $\boldsymbol{\omega}_m = \boldsymbol{\omega}_{IMU}$ as *known* from external (i.e., not vision-based) sources. Knowledge of $\boldsymbol{\omega}$ can then be used to derotate the perceived optical flow field as

$$\begin{bmatrix} u'_x \\ u'_y \end{bmatrix} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} - \begin{bmatrix} -x_x x_y & 1+x_x^2 & -x_y \\ -(1+x_y)^2 & x_x x_y & x_x \end{bmatrix} \boldsymbol{\omega}_m, \tag{2.7}$$

where the interaction matrix relating $\boldsymbol{u}$ to $(\boldsymbol{v}, \boldsymbol{\omega})$ was exploited [Chaumette and Hutchinson, 2006]. This derotation step then reduces matrix $\boldsymbol{H}$ to

$$\boldsymbol{H} = \frac{1}{d}\boldsymbol{v}\boldsymbol{n}^T. \tag{2.8}$$

Since $\boldsymbol{n}$ spans $\boldsymbol{H}^T$ and $\|\boldsymbol{n}\| = 1$, we can obtain $\boldsymbol{n}$ from the singular value decomposition $\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ as the first column of matrix $\boldsymbol{V}$. The inherent sign ambiguity can be resolved by enforcing $\boldsymbol{n}_z > 0$. Having retrieved $\boldsymbol{n}$, we then obtain $\boldsymbol{v}/d = \boldsymbol{H}\boldsymbol{n}$.

This algorithm, which is referred to as *V2* in the following, requires the observation of at least three feature pairs $(\boldsymbol{x}_i, \boldsymbol{u}_i)$ and yields a *unique* solution for $\boldsymbol{v}/d$ and $\boldsymbol{n}$.

2.2.4. Exploiting a Known Angular Velocity and Plane Orienation

In most indoor environments and when considering UAVs with down-looking cameras, the dominant plane can be safely taken as horizontal with, thus, its normal vector $\boldsymbol{n}$ parallel to the gravity vector. Therefore, one can exploit the ability of onboard IMUs to estimate (via internal filtering, see Sec. 2.5.6 for a detailed discussion and experimental validation) the local gravity vector, thus allowing to consider $\boldsymbol{n}_m \approx \boldsymbol{n}_{IMU}$ as measured independently from the visual input.

Plugging (2.8) in (2.4) yields

$$[\boldsymbol{x}]_\times \frac{1}{d}\boldsymbol{v} = \frac{[\boldsymbol{x}]_\times \boldsymbol{u}}{\boldsymbol{n}_m^T \boldsymbol{x}}. \tag{2.9}$$

where now $\boldsymbol{n}_m^T\boldsymbol{x}$ is a known quantity. Letting $\boldsymbol{b} = ([\boldsymbol{x}]_\times \boldsymbol{u})/(\boldsymbol{n}_m^T\boldsymbol{x}) \in \mathbb{R}^3$, one then obtains the following equation linear in $\boldsymbol{v}/d$ (the only unknown left)

$$[\boldsymbol{x}_i]_\times \frac{\boldsymbol{v}}{d} = \boldsymbol{b}_i. \tag{2.10}$$

A least-square approximation of $\boldsymbol{v}/d$ over all N tracked features can be obtained by stacking all $[\boldsymbol{x}_i]_\times$ into the matrix $\boldsymbol{A} = [[\boldsymbol{x}_1]_\times, \cdots, [\boldsymbol{x}_N]_\times]^T \in \mathbb{R}^{3N\times 3}$ and all $\boldsymbol{b}_i$ into the vector $\boldsymbol{B} = [\boldsymbol{b}_1, \cdots, \boldsymbol{b}_N]^T \in \mathbb{R}^{3N}$ resulting in the linear system

$$\boldsymbol{A}\frac{\boldsymbol{v}}{d} = \boldsymbol{B} \tag{2.11}$$

which can be solved as $\boldsymbol{v}/d = \boldsymbol{A}^\dagger \boldsymbol{B}$.

Note that any two distinct feature point vectors $\boldsymbol{x}_i, \boldsymbol{x}_j$ are never parallel due to the perspective projection of the camera. Thus, in principle, only two flow vectors are required to obtain a solution for $\boldsymbol{v}/d$. However, a more robust estimation is obtained by incorporating all observed flow vectors.

This third algorithm is referred to as *V3* in the following.

2.2.5. Final Discussion

We conclude this first section with some final considerations on the choice of basing the proposed ego-motion estimation algorithm upon the *homography* constraint instead of the *epipolar* constraint that is often exploited in computer and robotic vision. In general, the applicability of the homography vs. epipolar constraint depends on the structure of the environment: methods relying on the *epipolar constraint* are more appropriate for highly unstructured scenes where a collection of points in *general position* can always be detected. That is, the depth within the scene should be considerably larger than the distance between the scene and the camera. On the other hand, solutions based on the *homography constraint* should be favored when dealing with approximately planar scenes or large distances between the camera and the scene in combination with wide angle lenses. Indeed, in the ideal case of features extracted from a perfect planar scene, use of the epipolar constraint is not any longer guaranteed to yield a unique solution for the ego-motion recovery [Ma et al., 2004].

Furthermore, compared to epipolar-based algorithms, the homography constraint remains better conditioned also in case of stationary flight, e.g., with a small amount of translational motion (indeed, the epipolar constraint vanishes for a zero translation/linear velocity). Therefore, while the epipolar constraint can be more suited for persistently-translating maneuvers or vehicles (e.g., fixed-wing UAVs), the homography constraint is more robust for typical indoor (close to stationary) flight regimes of vertical take-off and landing vehicles such as quadrotor UAVs.

These considerations then motivated our choice of the *homography* constraint for dealing with the ego-motion recovery. Indeed, as explained, in many scenarios, such as indoor hallways or aerial coverage, one can safely assume presence of a dominant ground plane spanning most of the observed features. Coupled with the RANdom SAmple Consensus (RANSAC)-based outlier rejection described in the next Sec. 2.3, our proposed solution then proved to yield accurate results as discussed in the experimental evaluation in Sec. 2.5.3.

2.3. Segmentation of Features

To further improve the robustness of the velocity estimation system, we developed a quantitative measure of how well a given plane fits a set of observed features. In the following, we first propose a criterion to discriminate between features on the plane and outliers. This method is then used to obtain a real-time feature classification using onboard hardware as the quadrotor explores an unknown environment.

2.3.1. Planarity Measures

To test whether a certain group of observed features belongs to a common plane, we considered two different quantitative measures. We start noting that a pre-requisite for solving eq. (2.6) is that $\boldsymbol{B} \in \mathcal{R}(\boldsymbol{A})$ where $\mathcal{R}(\boldsymbol{A})$ denotes the range space of matrix $\boldsymbol{A}$. This can be restated as

$$\operatorname{rank}(\boldsymbol{A}) = \operatorname{rank}([\boldsymbol{A}\ \boldsymbol{B}]) = 8,$$

since $\operatorname{rank}(\boldsymbol{A}) = 8$ by construction. Let $\sigma_i \geq 0$, $i = 1 \dots 10$, be the singular values of the augmented matrix $[\boldsymbol{A}\ \boldsymbol{B}] \in \mathbb{R}^{3N \times 10}$ ordered from the largest to the smallest one. As a measure of how well the condition $\operatorname{rank}(\boldsymbol{A}) = 8$ is satisfied by the given set of measured features/optical

flow, one can monitor the value of σ_9. In fact, $\sigma_9 = \sigma_{10} = 0$ if $\text{rank}(\boldsymbol{A}) = 8$ holds, i.e., if all the observed points belong to a common plane, and $\sigma_9 > 0$ otherwise. Therefore, the value of $\sigma_9 \geq 0$ can be exploited as a measure of how well a certain set of features/optical flow meets the planarity constraint.

However, it is also possible to obtain an equivalent information by resorting to a different argument. Let $\boldsymbol{H}^S = \boldsymbol{A}^\dagger \boldsymbol{B}$ represent the least-square solution of the linear system (2.6): in order to obtain a measure of how well $\boldsymbol{H}^S$ is actually solving system (2.6), one can consider the 'reprojection' vector $\boldsymbol{E} \in \mathbb{R}^{3N}$

$$\boldsymbol{E} = \boldsymbol{B} - \boldsymbol{A}\boldsymbol{H}^S = \boldsymbol{B} - \boldsymbol{A}\boldsymbol{A}^\dagger \boldsymbol{B} = (\boldsymbol{I}_{3N} - \boldsymbol{A}\boldsymbol{A}^\dagger)\boldsymbol{B}$$

which should vanish if (2.6) admits an exact solution, i.e., if all the observed feature points belong to the same plane. Therefore, another equivalent measure of the planarity assumption besides the previously discussed σ_9 can be taken as $e = \|\boldsymbol{E}\|/N$.

Although yielding similar information, we found in practice that e was much less sensitive to noise issues compared to σ_9. Additionally, the quantity e presents two advantages. First, it does not require intense additional computations since the pseudo-inverse of matrix $\boldsymbol{A}$, i.e., $\boldsymbol{A}^\dagger$, is already computed when solving (2.6) for recovering the continuous homography matrix. Secondly, this method can be used to test effectively whether a new feature could be added to an already known plane made of other feature points. For this purpose, one can obtain $\boldsymbol{H}^S = \boldsymbol{A}^\dagger \boldsymbol{B}$ using only the features which are already known to form a plane. Then, by constructing a new matrix $\boldsymbol{A'}$ and $\boldsymbol{B'}$ out of the new feature point as described in Sec. 2.2.2, one can decide the membership to the given plane by testing $\|\boldsymbol{B'} - \boldsymbol{A'}\boldsymbol{A}^\dagger \boldsymbol{B}\|/N$ against a threshold.

2.3.2. Algorithm for Robust Velocity Estimations

Having an effective measure to test both the plane hypothesis and the assignment of new features to existing planes, we can then design an algorithm that is able to dynamically decide whether an observed feature belongs to the dominant plane. Then, exploiting this clustering, we can obtain a better estimation of both the scaled linear velocity and the plane normal when using algorithm *V2*.

In the following, we sketch the algorithm that is used to process new images as they are captured:

- *Initialization.* Before the flight or after the dominant plane was lost, a RANSAC inspired approach is used to pick an initial set of features. The validity of the potential plane is evaluated using our method described above.
- *Update phase.* As a first step, all features already known to be part of the plane are updated to the current location as reported by the flow tracker. Additionally, all those features which have not been observed in the current frame are deleted from the plane.
- *Estimation step.* The scaled linear velocity is estimated together with the plane normal vector. Thus, the estimated linear velocity is independent of the particular plane orientation, allowing for a broader range of environments (algorithm *V2*). This linear velocity is then taken as the best estimation of $\boldsymbol{v}/d$.
- *Integration of new features.* Afterwards, the algorithm tests all other observed features against the current model of the known plane that has been built and maintained during

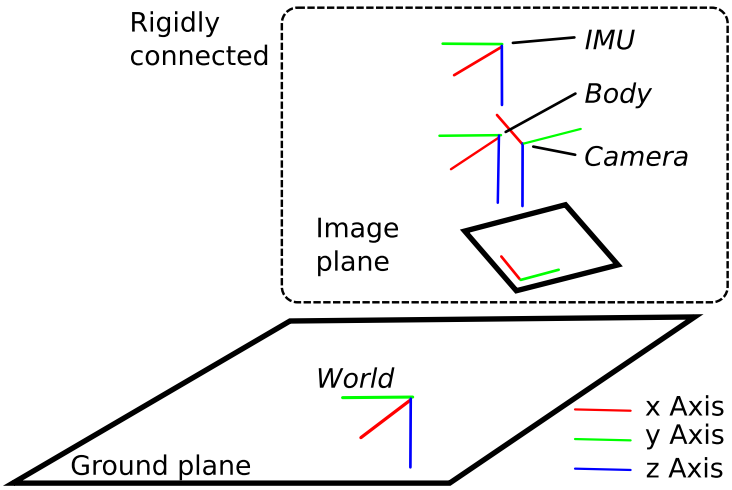


Figure 2.1.: Locations of the IMU ($\mathcal{I}$), camera ($\mathcal{C}$), body ($\mathcal{B}$) and world frame ($\mathcal{W}$) relative to each other. Frames $\mathcal{I}, \mathcal{B}$ and $\mathcal{C}$ are assumed to be rigidly linked to each other. The world frame $\mathcal{W}$ is oriented horizontally with its z-axis pointing down, following the NED (North-East-Down) convention commonly used in air/space scenarios.

the last steps. A threshold is used to decide whether a new feature is added to the feature set using the generalized method to test individual features.

- *Validation of the feature set.* The performance of each feature in the set is monitored periodically. Thus, initially trusted features that turned out to be outliers (since, e.g., the camera has approached the scene from a far distance) are excluded from the set in this step.

2.4. Online Scale Estimation

The ego-motion algorithm (in its three variants) presented in Sec. 2.2 allows us to obtain a reliable estimation of the scaled camera linear velocity $\boldsymbol{v}/d$. In this section, we discuss two estimation schemes meant to recover the plane distance d by fusing the optical flow decomposition with the onboard accelerometer measurements.

2.4.1. Equations of Motion

We start by deriving the equations of motion that are relevant to the case under consideration. In the following, we denote with $\mathcal{B}, \mathcal{C}, \mathcal{I}$ and $\mathcal{W}$ the body, camera, IMU and inertial world frame, respectively. The origin of frame $\mathcal{B}$ is assumed to be located at the quadrotor barycenter, while frames $\mathcal{C}$ and $\mathcal{I}$ are rigidly attached to $\mathcal{B}$, see Fig. 2.1. Throughout the text, left superscripts is exploited to indicate the frames where quantities are expressed in. The symbol ${}^{\mathcal{X}}\boldsymbol{R}_{\mathcal{Y}} \in SO(3)$ is used to denote the rotation matrix from frame $\mathcal{X}$ to frame $\mathcal{Y}$, and ${}^{\mathcal{Z}}\boldsymbol{p}_{\mathcal{X}\mathcal{Y}} \in \mathbb{R}^3$ to represent the vector from the origin of frame $\mathcal{X}$ to the origin of frame $\mathcal{Y}$, expressed in frame $\mathcal{Z}$. We also introduce the following quantities instrumental for the next developments: $\boldsymbol{g} \in \mathbb{R}^3$ as the gravity vector, and ${}^{\mathcal{I}}\boldsymbol{f} \in \mathbb{R}^3$, ${}^{\mathcal{I}}\boldsymbol{\omega} \in \mathbb{R}^3$ as the specific acceleration and angular velocity at the origin of $\mathcal{I}$.

We define ${}^{\mathcal{C}}\boldsymbol{v} = {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{W}}\,{}^{\mathcal{W}}\dot{\boldsymbol{p}}_{\mathcal{WC}}$ as the camera linear velocity in camera frame, and ${}^{\mathcal{B}}\boldsymbol{v} = {}^{\mathcal{B}}\boldsymbol{R}_{\mathcal{W}}\,{}^{\mathcal{W}}\dot{\boldsymbol{p}}_{\mathcal{WB}}$ as the body linear velocity in body frame. Since ${}^{\mathcal{B}}\boldsymbol{R}_{\mathcal{C}}$, ${}^{\mathcal{B}}\boldsymbol{R}_{\mathcal{I}}$, ${}^{\mathcal{B}}\boldsymbol{p}_{\mathcal{BC}}$ and ${}^{\mathcal{B}}\boldsymbol{p}_{\mathcal{BI}}$ are assumed to be constant, from standard kinematics the following relationships hold

$$
\begin{aligned}
{}^{\mathcal{B}}\boldsymbol{v} &= {}^{\mathcal{B}}\boldsymbol{R}_{\mathcal{C}}({}^{\mathcal{C}}\boldsymbol{v} + [{}^{\mathcal{C}}\boldsymbol{\omega}]_{\times}\,{}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{CB}}) = {}^{\mathcal{B}}\boldsymbol{R}_{\mathcal{C}}\,{}^{\mathcal{C}}\boldsymbol{v} + [{}^{\mathcal{B}}\boldsymbol{\omega}]_{\times}\,{}^{\mathcal{B}}\boldsymbol{p}_{\mathcal{CB}}, && (2.12)\\
{}^{\mathcal{C}}\dot{\boldsymbol{v}} &= {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}({}^{\mathcal{I}}\boldsymbol{a} + [{}^{\mathcal{I}}\dot{\boldsymbol{\omega}}]_{\times}\,{}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}} + [{}^{\mathcal{I}}\boldsymbol{\omega}]^2_{\times}\,{}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}}) - [{}^{\mathcal{C}}\boldsymbol{\omega}]_{\times}\,{}^{\mathcal{C}}\boldsymbol{v}\\
&= {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}\,{}^{\mathcal{I}}\boldsymbol{a} + [{}^{\mathcal{C}}\dot{\boldsymbol{\omega}}]_{\times}\,{}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{IC}} + [{}^{\mathcal{C}}\boldsymbol{\omega}]^2_{\times}\,{}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{IC}} - [{}^{\mathcal{C}}\boldsymbol{\omega}]_{\times}\,{}^{\mathcal{C}}\boldsymbol{v}, && (2.13)\\
{}^{\mathcal{C}}\boldsymbol{\omega} &= {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}\,{}^{\mathcal{I}}\boldsymbol{\omega} && (2.14)\\
{}^{\mathcal{C}}\dot{\boldsymbol{\omega}} &= {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}\,{}^{\mathcal{I}}\dot{\boldsymbol{\omega}} && (2.15)
\end{aligned}
$$

where ${}^{\mathcal{I}}\boldsymbol{a} = {}^{\mathcal{I}}\boldsymbol{R}_{\mathcal{W}}\,{}^{\mathcal{W}}\ddot{\boldsymbol{p}}_{\mathcal{WI}}$ is the linear acceleration experienced by the IMU. We note that ${}^{\mathcal{I}}\boldsymbol{a} = {}^{\mathcal{I}}\boldsymbol{f} + {}^{\mathcal{I}}\boldsymbol{g}$ and ${}^{\mathcal{I}}\boldsymbol{g} = {}^{\mathcal{I}}\boldsymbol{R}_{\mathcal{W}}[0,0,g]^T$ in case of a horizontal orientation of the world frame, see Fig. 2.1.

In presence of a planar scene ${}^{\mathcal{C}}\boldsymbol{n}^T\,{}^{\mathcal{C}}\boldsymbol{X} + d = 0$ holds and one also has (see, e.g., [Robuffo Giordano et al., 2008])

$$
\begin{aligned}
{}^{\mathcal{C}}\dot{\boldsymbol{n}} &= -[{}^{\mathcal{C}}\boldsymbol{\omega}]_{\times}\,{}^{\mathcal{C}}\boldsymbol{n} && (2.16)\\
\dot{d} &= {}^{\mathcal{C}}\boldsymbol{v}^T\,{}^{\mathcal{C}}\boldsymbol{n}. && (2.17)
\end{aligned}
$$

Finally, according to this notation, the decomposition of the optical flow summarized in Sec. 2.2 allows us to directly measure the scaled linear velocity ${}^{\mathcal{C}}\tilde{\boldsymbol{v}} = {}^{\mathcal{C}}\boldsymbol{v}/d$. The estimation schemes presented in the following are then meant to recover the (unmeasurable) value of the plane distance d and the metric linear velocity vector ${}^{\mathcal{C}}\boldsymbol{v}$.

2.4.2. Scale Estimation based on the Extended Kalman Filter

As a first approach to estimate the distance to the planar scene d, we develop a classical EKF upon the system equations of motion. In particular, we adopt the discrete version of the EKF, and let index $k \in \mathbb{N}$ denote the k-th iteration step. For clarity, we append a right subscript m to identify all those quantities that are directly available through one of the onboard sensors, e.g., specific force ${}^{\mathcal{I}}\boldsymbol{f}_m$ and angular velocity ${}^{\mathcal{I}}\boldsymbol{\omega}_m$ from the IMU, and the scaled linear velocity ${}^{\mathcal{C}}\tilde{\boldsymbol{v}}_m = ({}^{\mathcal{C}}\boldsymbol{v}/d)_m$ from the camera.

We define the EKF state vector $\boldsymbol{s}$ to consist of the metric camera linear velocity in camera frame ${}^{\mathcal{C}}\boldsymbol{v}$ and the camera distance to the planar scene d:

$$
\boldsymbol{s} = \begin{bmatrix} {}^{\mathcal{C}}\boldsymbol{v} \\ d \end{bmatrix}, \qquad {}^{\mathcal{C}}\boldsymbol{v} \in \mathbb{R}^3, d \in \mathbb{R}. \tag{2.18}
$$

Given the high update rate and low latency of the onboard IMU, acceleration measurements are available for all prediction steps. Therefore, we rewrite (2.13) in terms of the measurements ${}^{\mathcal{I}}\boldsymbol{f}_m$ and ${}^{\mathcal{I}}\boldsymbol{\omega}_m$ obtained from the IMU in frame $\mathcal{I}$:

$$
\begin{aligned}
{}^{\mathcal{C}}\dot{\boldsymbol{v}} &= {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}({}^{\mathcal{I}}\boldsymbol{a} + [{}^{\mathcal{I}}\dot{\boldsymbol{\omega}}]_{\times}\,{}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}} + [{}^{\mathcal{I}}\boldsymbol{\omega}_m]^2_{\times}\,{}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}}) - [{}^{\mathcal{C}}\boldsymbol{\omega}_m]_{\times}\,{}^{\mathcal{C}}\boldsymbol{v}\\
&\approx {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}({}^{\mathcal{I}}\boldsymbol{f}_m + {}^{\mathcal{I}}\boldsymbol{g} + [{}^{\mathcal{I}}\boldsymbol{\omega}_m]^2_{\times}\,{}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}}) - [{}^{\mathcal{C}}\boldsymbol{\omega}_m]_{\times}\,{}^{\mathcal{C}}\boldsymbol{v} && (2.19)
\end{aligned}
$$

Since no direct measurement of ${}^{\mathcal{I}}\dot{\boldsymbol{\omega}}$ is available on most quadrotor setups, and ${}^{\mathcal{I}}\boldsymbol{\omega}$ being typically a noisy signal, we approximate ${}^{\mathcal{I}}\dot{\boldsymbol{\omega}} \approx \mathbf{0}$ in (2.19) rather than attempting to recover ${}^{\mathcal{I}}\dot{\boldsymbol{\omega}}$ via a numerical differentiation. Consequently, using (2.17) and (2.19), the following equations govern the predicted state $\bar{\boldsymbol{s}}[k]$:

$$ {}^{\mathcal{C}}\bar{\boldsymbol{v}}[k] = {}^{\mathcal{C}}\hat{\boldsymbol{v}}[k-1] + \Delta_t \, {}^{\mathcal{C}}\dot{\boldsymbol{v}}[k] \tag{2.20} $$

$$ \bar{d}[k] = \hat{d}[k-1] + \Delta_t \, {}^{\mathcal{C}}\hat{\boldsymbol{v}}[k]^T \, {}^{\mathcal{C}}\boldsymbol{n}[k] \tag{2.21} $$

where Δ_t denotes the sampling time of the filter.

Although most quantities derived in the following steps are time varying, from now on, for the sake of exposition clarity, we omit the time dependency $[k]$ wherever possible.

To compute the predicted covariance matrix of the system uncertainty $\bar{\boldsymbol{\Sigma}}[k] \in \mathbb{R}^{4\times4}$, we first derive the Jacobian matrix $\boldsymbol{G}[k] \in \mathbb{R}^{4\times4}$

$$ \begin{aligned} \boldsymbol{G} &= \begin{bmatrix} \dfrac{\partial \, {}^{\mathcal{C}}\bar{\boldsymbol{v}}[k]}{\partial \, {}^{\mathcal{C}}\hat{\boldsymbol{v}}[k-1]} & \dfrac{\partial \, {}^{\mathcal{C}}\bar{\boldsymbol{v}}[k]}{\partial \hat{d}[k-1]} \\ \dfrac{\partial \bar{d}[k]}{\partial \, {}^{\mathcal{C}}\hat{\boldsymbol{v}}[k-1]} & \dfrac{\partial \bar{d}[k]}{\partial \hat{d}[k-1]} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{I}_3 - \Delta_t [{}^{\mathcal{I}}\boldsymbol{\omega}_m]_\times & \mathbf{0}_{3\times1} \\ \Delta_t \, {}^{\mathcal{C}}\boldsymbol{n}^T & 1 \end{bmatrix}. \end{aligned} \tag{2.22} $$

Matrix $\hat{\boldsymbol{\Sigma}}[k-1]$ from the previous step is then propagated as:

$$ \bar{\boldsymbol{\Sigma}}[k] = \boldsymbol{G}\hat{\boldsymbol{\Sigma}}[k-1]\boldsymbol{G}^T + \boldsymbol{R}. \tag{2.23} $$

Here, matrix $\boldsymbol{R} \in \mathbb{R}^{4\times4}$ is obtained from

$$ \boldsymbol{R} = \boldsymbol{V} \begin{bmatrix} \mathrm{cov}({}^{\mathcal{I}}\boldsymbol{f}_{\mathrm{m}}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathrm{cov}({}^{\mathcal{I}}\boldsymbol{\omega}_{\mathrm{m}}) \end{bmatrix} \boldsymbol{V}^T \tag{2.24} $$

where

$$ \begin{aligned} \boldsymbol{V} &= \begin{bmatrix} \dfrac{\partial \, {}^{\mathcal{C}}\bar{\boldsymbol{v}}[k]}{\partial \, {}^{\mathcal{I}}\boldsymbol{f}_m} & \dfrac{\partial \, {}^{\mathcal{C}}\bar{\boldsymbol{v}}[k]}{\partial \, {}^{\mathcal{I}}\boldsymbol{\omega}_m} \\ \dfrac{\partial \bar{d}[k]}{\partial \, {}^{\mathcal{I}}\boldsymbol{f}_m} & \dfrac{\partial \bar{d}[k]}{\partial \, {}^{\mathcal{I}}\boldsymbol{\omega}_m} \end{bmatrix} \\ &= \begin{bmatrix} \Delta_t \, {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}} & \Delta_t({}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}\boldsymbol{M} + [{}^{\mathcal{C}}\hat{\boldsymbol{v}}[k]]_\times \, {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}) \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} \end{bmatrix} \in \mathbb{R}^{4\times6} \end{aligned} \tag{2.25} $$

$$ \boldsymbol{M} = ({}^{\mathcal{I}}\boldsymbol{\omega}_m^T \, {}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}})\boldsymbol{I}_3 + {}^{\mathcal{I}}\boldsymbol{\omega}_m \, {}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}}^T - 2 \, {}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}} \, {}^{\mathcal{I}}\boldsymbol{\omega}_m^T, \tag{2.26} $$

and $\mathrm{cov}({}^{\mathcal{I}}\boldsymbol{f}_{\mathrm{m}}) \in \mathbb{R}^{3\times3}$, $\mathrm{cov}({}^{\mathcal{I}}\boldsymbol{\omega}_{\mathrm{m}}) \in \mathbb{R}^{3\times3}$ are the covariance matrixes of the accelerometers/gyroscopes sensors in the IMU, which can be experimentally determined.

The predicted state $\bar{\boldsymbol{s}}$ is then updated whenever a new scaled visual velocity estimate $\boldsymbol{z}_m = {}^{\mathcal{C}}\tilde{\boldsymbol{v}}_m = \left({}^{\mathcal{C}}\boldsymbol{v}/d\right)_m$ becomes available from the optical flow decomposition. Let $\bar{\boldsymbol{z}}$ be the predicted scaled visual velocity estimation based on the predicted state $\bar{\boldsymbol{s}}$

$$ \bar{\boldsymbol{z}} = \frac{{}^{\mathcal{C}}\bar{\boldsymbol{v}}}{\bar{d}}. \tag{2.27} $$

The kalman gain $\boldsymbol{K} \in \mathbb{R}^{4\times3}$ is obtained as

$$\boldsymbol{K} = \bar{\boldsymbol{\Sigma}}\boldsymbol{J}^T(\boldsymbol{J}\bar{\boldsymbol{\Sigma}}\boldsymbol{J}^T + \mathrm{cov}(\boldsymbol{z}_\mathrm{m}))^{-1}, \tag{2.28}$$

where $\mathrm{cov}(\boldsymbol{z}_\mathrm{m}) \in \mathbb{R}^{3\times3}$ is the covariance matrix of the scaled visual velocity measurement, and the Jacobian $\boldsymbol{J} \in \mathbb{R}^{3\times4}$ is given by

$$\begin{aligned}\boldsymbol{J} &= \begin{bmatrix} \dfrac{\partial\bar{\boldsymbol{z}}[k]}{\partial {}^{\mathcal{C}}\bar{\boldsymbol{v}}[k]} & \dfrac{\partial\bar{\boldsymbol{z}}[k]}{\partial\bar{d}[k]} \end{bmatrix} \\ &= \begin{bmatrix} \dfrac{\boldsymbol{I}_3}{\bar{d}} & -\dfrac{{}^{\mathcal{C}}\bar{\boldsymbol{v}}}{\bar{d}^2} \end{bmatrix}. \end{aligned} \tag{2.29}$$

Finally, the predicted state $\bar{\boldsymbol{s}}[k]$ is updated to the estimated state $\hat{\boldsymbol{s}}[k]$, together with uncertainty matrix $\hat{\boldsymbol{\Sigma}}[k]$, as

$$\hat{\boldsymbol{s}} = \begin{bmatrix} {}^{\mathcal{C}}\hat{\boldsymbol{v}} \\ \hat{d} \end{bmatrix} = \begin{bmatrix} {}^{\mathcal{C}}\bar{\boldsymbol{v}} \\ \bar{d} \end{bmatrix} + \boldsymbol{K}\,(\boldsymbol{z}_m - \bar{\boldsymbol{z}}) \tag{2.30}$$

$$\hat{\boldsymbol{\Sigma}} = (\boldsymbol{I}_4 - \boldsymbol{K}\boldsymbol{J})\bar{\boldsymbol{\Sigma}}. \tag{2.31}$$

Discussion

Several quantities are needed for the implementation of the proposed EKF. Apart from the estimated state $\hat{\boldsymbol{s}}[k]$, one needs knowledge of:

1 the constant IMU/camera rotation matrix ${}^{\mathcal{I}}\boldsymbol{R}_{\mathcal{C}}$ and displacement vector ${}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}}$;
2 the IMU angular velocity ${}^{\mathcal{I}}\boldsymbol{\omega}_m$;
3 the scaled camera linear velocity ${}^{\mathcal{C}}\tilde{\boldsymbol{v}}_m = ({}^{\mathcal{C}}\boldsymbol{v}/d)_m$;
4 the IMU linear acceleration ${}^{\mathcal{I}}\boldsymbol{a} = {}^{\mathcal{I}}\boldsymbol{f}_m + {}^{\mathcal{I}}\boldsymbol{g}$;
5 the plane normal ${}^{\mathcal{C}}\boldsymbol{n}$.

The quantities in item 1 are assumed to be known from a preliminary IMU/camera calibration phase, see, e.g., [Lobo and Dias, 2007], while vector ${}^{\mathcal{I}}\boldsymbol{\omega}$ in item 2 is available directly from the IMU gyroscope readings. Similarly, vector ${}^{\mathcal{C}}\tilde{\boldsymbol{v}}_m$ in item 3 is retrieved from the optical flow decomposition described in Sec. 2.2.

Measurement of the linear acceleration ${}^{\mathcal{I}}\boldsymbol{a}$ in item 3 requires the specific acceleration ${}^{\mathcal{I}}\boldsymbol{f}_m$ (directly available through the IMU accelerometer readings) and knowledge of the gravity vector ${}^{\mathcal{I}}\boldsymbol{g}$ in IMU frame. An estimation of this latter quantity is also provided by standard IMUs *in near-hovering conditions*, a fact largely exploited when recovering the UAV attitude from onboard sensing, see, e.g., [Mahony et al., 2008]. In our case, we found the employed IMU able to provide a reliable estimation of ${}^{\mathcal{I}}\boldsymbol{g}$ even when undergoing non-negligible accelerations thanks to an internal filtering scheme. A further discussion including an experimental validation of these observations is provided in Sec. 2.5.6.

Finally, the normal of the planar scene ${}^{\mathcal{C}}\boldsymbol{n}$ can be directly recovered from the optical flow decomposition as discussed in Sects. 2.2.2–2.2.3 (this step can also be complemented by the use of filtering techniques such as those discussed in [Eudes et al., 2013]). In case of a horizontal planar scene often found in indoor environments, the direction of ${}^{\mathcal{C}}\boldsymbol{n}$ can be additionally identified as the direction of ${}^{\mathcal{I}}\boldsymbol{g}$. This latter possibility was exploited in all the experiments reported in Sec. 2.5 that indeed involved a horizontal ground plane and thus a plane normal ${}^{\mathcal{C}}\boldsymbol{n}$ parallel to $\boldsymbol{g}$.

2.4.3. Scale Estimation based on a Nonlinear Estimation Scheme

In this Section, we summarize the derivations of an alternative *nonlinear observer* for retrieving the plane distance d based on a framework originally proposed in [De Luca et al., 2008; Robuffo Giordano et al., 2008] for dealing with structure from motion problems, and recently revisited in [Spica and Robuffo Giordano, 2013] in the context of *nonlinear active estimation*.

Compared to the previously discussed EKF, this estimation scheme does not require any linearization/approximation step of the system dynamics. This results in an overall cleaner design which, following the analysis reported in [Spica and Robuffo Giordano, 2013], also allows to *fully characterize* the estimation error transient response. In particular, we discuss how one can impose to the estimation error a transient response equivalent to that of a *reference linear second-order system with desired poles* by suitably acting on the estimation gains and on the UAV motion. This possibility enables the designer, for instance, to choose the needed combination of estimation gains/UAV motion yielding a desired estimation performance. Furthermore, it allows to predict the convergence time of the filter in terms of percentages of the initial error[1] in advance. However, we also note that, as opposed to the EKF, the filter design assumes deterministic dynamics. Therefore, it does not explicitly account for the noise present in the system concerning state transitions and sensor measurements. This point is thoroughly addressed in Sec. 2.5.

For the sake of exposition, the following developments are here formulated in continuous time. We first recall a classical result of the adaptive control literature, also known as the Persistency of Excitation Lemma [Marino and Tomei, 1995], upon which the next developments are based.

Lemma 1 (Persistency of Excitation)**.** *Consider the system*

$$\left\{ \begin{array}{rcl} \dot{\boldsymbol{\xi}} & = & -\boldsymbol{D}\boldsymbol{\xi} + \boldsymbol{\Omega}^T(t)\boldsymbol{z}, \\ \dot{\boldsymbol{\zeta}} & = & -\boldsymbol{\Lambda}\boldsymbol{\Omega}(t)\boldsymbol{P}\boldsymbol{\xi}, \end{array} \right. \tag{2.32}$$

where $\boldsymbol{\xi} \in \mathbb{R}^m$, $\boldsymbol{\zeta} \in \mathbb{R}^p$, $\boldsymbol{D} > 0$, $\boldsymbol{P} = \boldsymbol{P}^T > 0$ *such that*

$$\boldsymbol{D}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{D} = \boldsymbol{Q}, \qquad \text{with } \boldsymbol{Q} > 0,$$

and $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^T > 0$. *If* $\|\boldsymbol{\Omega}(t)\|$, $\|\dot{\boldsymbol{\Omega}}(t)\|$ *are uniformly bounded and the* persistency of excitation *(PE) condition is satisfied, i.e., there exist a* $\Delta_t > 0$ *and* $\gamma > 0$ *such that*

$$\int_t^{t+\Delta_t} \boldsymbol{\Omega}(\tau)\boldsymbol{\Omega}^T(\tau)\mathrm{d}\tau \geq \gamma \boldsymbol{I} > 0, \qquad \forall t \geq t_0, \tag{2.33}$$

then $(\boldsymbol{\xi}, \boldsymbol{\zeta}) = (\boldsymbol{0}, \boldsymbol{0})$ *is a globally exponentially stable equilibrium point.*

This result can be exploited as follows: assume a vector $\boldsymbol{s} = [\boldsymbol{s}_1^T \ \boldsymbol{s}_2^T]^T \in \mathbb{R}^{m+p}$ can be split into a *measurable* component $\boldsymbol{s}_1$ and an *unmeasurable* component $\boldsymbol{s}_2$. Defining an estimation vector $\hat{\boldsymbol{s}} = [\hat{\boldsymbol{s}}_1^T \ \hat{\boldsymbol{s}}_2^T]^T \in \mathbb{R}^{m+p}$, and the corresponding estimation error

$$\boldsymbol{e} = \left[\begin{array}{c} \boldsymbol{\xi} \\ \boldsymbol{\zeta} \end{array} \right] = \left[\begin{array}{c} \boldsymbol{s}_1 - \hat{\boldsymbol{s}}_1 \\ \boldsymbol{s}_2 - \hat{\boldsymbol{s}}_2 \end{array} \right],$$

[1] Therefore, given an upper bounded initial error $d(t_0) - \hat{d}(t_0)$, one can, for instance, plan in advance the duration of the UAV motion so as to yield a guaranteed accuracy in the estimated plane distance.

the goal is to design an update rule for $\hat{\boldsymbol{s}}$ such that the closed-loop error dynamics matches formulation (2.32). When this manipulation is possible, Lemma 1 ensures global exponential convergence of the estimation error $\boldsymbol{e} = [\boldsymbol{\xi}^T \ \boldsymbol{\zeta}^T]^T$ to $\mathbf{0}$, thus allowing to infer the unmeasurable value of $\boldsymbol{s}_2$ from knowledge of $\boldsymbol{s}_1$. The PE condition (2.33) plays the role of an observability constraint: estimation of $\boldsymbol{s}_2$ is possible iff matrix $\boldsymbol{\Omega}(t) \in \mathbb{R}^{p\times m}$ is *sufficiently exciting* over time in the sense of (2.33). We finally note that, being $\boldsymbol{\Omega}(t)$ a generic (but known) *time-varying* quantity, formulation (2.32) is not restricted to only span the class of linear systems, but it can easily accommodate nonlinear terms as long as they are embedded in matrix $\boldsymbol{\Omega}(t)$.

We now detail how to tailor (2.32) to the case under consideration. We start by defining

$$s_2 = \frac{1}{d}$$

and

$$\boldsymbol{s}_1 = {}^{\mathcal{C}}\tilde{\boldsymbol{v}} = \frac{{}^{\mathcal{C}}\boldsymbol{v}}{d} = {}^{\mathcal{C}}\boldsymbol{v} s_2$$

with, therefore, $m = 3$ and $p = 1$. Exploiting (2.17), the dynamics of s_2 is given by

$$\dot{s}_2 = -\frac{\dot{d}}{d^2} = -\frac{{}^{\mathcal{C}}\boldsymbol{v}^T\, {}^{\mathcal{C}}\boldsymbol{n}}{d^2} = -\frac{{}^{\mathcal{C}}\tilde{\boldsymbol{v}}^T\, {}^{\mathcal{C}}\boldsymbol{n}}{d} = -s_2 \boldsymbol{s}_1^T\, {}^{\mathcal{C}}\boldsymbol{n}. \tag{2.34}$$

As for the dynamics of $\boldsymbol{s}_1$, using (2.13) we have

$$\begin{aligned}
\dot{\boldsymbol{s}}_1 &= {}^{\mathcal{C}}\dot{\boldsymbol{v}} s_2 + {}^{\mathcal{C}}\boldsymbol{v}\dot{s}_2 = {}^{\mathcal{C}}\dot{\boldsymbol{v}} s_2 - {}^{\mathcal{C}}\boldsymbol{v} s_2 \boldsymbol{s}_1^T\, {}^{\mathcal{C}}\boldsymbol{n} \\
&= {}^{\mathcal{C}}\dot{\boldsymbol{v}} s_2 - \boldsymbol{s}_1 \boldsymbol{s}_1^T\, {}^{\mathcal{C}}\boldsymbol{n} \\
&= ({}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}\, {}^{\mathcal{I}}\boldsymbol{a} + [{}^{\mathcal{C}}\boldsymbol{\omega}]_\times^2\, {}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{IC}} + [{}^{\mathcal{C}}\dot{\boldsymbol{\omega}}]_\times\, {}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{IC}}) s_2 \\
&\quad - [{}^{\mathcal{C}}\boldsymbol{\omega}]_\times \boldsymbol{s}_1 - \boldsymbol{s}_1 \boldsymbol{s}_1^T\, {}^{\mathcal{C}}\boldsymbol{n} \\
&= \boldsymbol{\Omega}^T(t) s_2 - [{}^{\mathcal{C}}\boldsymbol{\omega}]_\times \boldsymbol{s}_1 - \boldsymbol{s}_1 \boldsymbol{s}_1^T\, {}^{\mathcal{C}}\boldsymbol{n}
\end{aligned} \tag{2.35}$$

with

$$\begin{aligned}
\boldsymbol{\Omega}^T(t) &= {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}\, {}^{\mathcal{I}}\boldsymbol{a} + [{}^{\mathcal{C}}\boldsymbol{\omega}]_\times^2\, {}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{IC}} + [{}^{\mathcal{C}}\dot{\boldsymbol{\omega}}]_\times\, {}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{IC}} \\
&\approx {}^{\mathcal{C}}\boldsymbol{R}_{\mathcal{I}}\, {}^{\mathcal{I}}\boldsymbol{a} + [{}^{\mathcal{C}}\boldsymbol{\omega}]_\times^2\, {}^{\mathcal{C}}\boldsymbol{p}_{\mathcal{IC}} \in \mathbb{R}^3.
\end{aligned} \tag{2.36}$$

As with the EKF, we approximate ${}^{\mathcal{I}}\dot{\boldsymbol{\omega}} \approx \mathbf{0}$ since no direct measurement of the UAV angular acceleration is available on board.

We can then design the update rule for the estimated state $\hat{\boldsymbol{s}}$ as

$$\begin{cases}
\dot{\hat{\boldsymbol{s}}}_1 &= \boldsymbol{\Omega}^T(t)\hat{s}_2 - [{}^{\mathcal{C}}\boldsymbol{\omega}]_\times \boldsymbol{s}_1 - \boldsymbol{s}_1 \boldsymbol{s}_1^T\, {}^{\mathcal{C}}\boldsymbol{n} + \boldsymbol{D}\boldsymbol{\xi} \\
\dot{\hat{s}}_2 &= -\hat{s}_2 \boldsymbol{s}_1^T\, {}^{\mathcal{C}}\boldsymbol{n} + \boldsymbol{\Lambda}\boldsymbol{\Omega}(t)\boldsymbol{\xi}
\end{cases} \tag{2.37}$$

with $\boldsymbol{D} > 0$ and $\boldsymbol{\Lambda} > 0$ being symmetric and positive definite gain matrixes. Note that (2.37) involves only measured quantities, including a feedback action on $\boldsymbol{\xi} = \boldsymbol{s}_1 - \hat{\boldsymbol{s}}_1$, i.e., the measurable component of the error vector. With this choice, the dynamics of the estimation error $\boldsymbol{e} = [\boldsymbol{\xi}^T \ \zeta^T]^T$ then becomes

$$\begin{cases}
\dot{\boldsymbol{\xi}} &= -\boldsymbol{D}\boldsymbol{\xi} + \boldsymbol{\Omega}^T(t)\zeta \\
\dot{\zeta} &= -\boldsymbol{\Lambda}\boldsymbol{\Omega}(t)\boldsymbol{\xi} - \zeta \boldsymbol{s}_1^T\, {}^{\mathcal{C}}\boldsymbol{n}.
\end{cases} \tag{2.38}$$

It is easy to verify that, by letting $\boldsymbol{P} = \boldsymbol{I}_3$, the formulation (2.32) is almost fully recovered apart from the spurious scalar term

$$g(\boldsymbol{e}, t) = -\zeta \boldsymbol{s}_1^T {}^{\mathcal{C}}\boldsymbol{n} \tag{2.39}$$

in (2.38). Nevertheless, exponential convergence of the estimation error $\boldsymbol{e}(t)$ to $\boldsymbol{0}$ can still be proven by resorting to Lyapunov theory and by noting that the spurious term $g(\boldsymbol{e}, t)$ is a *vanishing perturbation* of an otherwise globally exponentially stable nominal system, i.e., $g(\boldsymbol{0}, t) = \boldsymbol{0}$, $\forall t$. The reader is referred to [De Luca et al., 2008; Robuffo Giordano et al., 2008; Spica and Robuffo Giordano, 2013] for additional discussions and proofs of these facts.

We note that the design of observer (2.37) did not require any linearization step as opposed to the previously presented EKF thanks to the more general class of (nonlinear) systems spanned by formulation (2.32). Instrumental, in this sense, is the choice of considering, as state variable s_2, the *inverse* of the plane distance d: this manipulation allows us to obtain linearity of the state equations in s_2, thus ultimately making it possible to apply the PE Lemma to the case under consideration. Similar *inverse parameterizations* for the scene scale can also be found in other works dealing with the issue of range estimation from moving cameras, see, e.g., [De Luca et al., 2008; Civera et al., 2008].

It is also worth to analyze, in our specific case, the implication of the PE condition (2.33) for obtaining a converging estimation. With $\boldsymbol{\Omega}^T(t) \in \mathbb{R}^3$ being a vector, condition (2.33) requires that $\|\boldsymbol{\Omega}(t)\|$ (i.e., at least one component) does not ultimately vanish over time. Since vector $\boldsymbol{\Omega}^T(t)$ represents the camera linear acceleration through space with respect to the inertial world frame $\mathcal{W}$, we recover the well-known condition that the estimation of d is possible if and only if the camera undergoes a physical acceleration. Consequently, moving at constant velocity with respect to $\mathcal{W}$ does not allow the estimation to converge. Note, however, that the estimation of the up-to-scale velocity remains still possible.

Finally, as done in the previous Sec. 2.4.2, we list the quantities that are necessary for the implementation of the proposed observer (2.37). In addition to the estimated state $\hat{\boldsymbol{s}}$, these are:

1 the constant IMU/camera rotation matrix ${}^{\mathcal{I}}\boldsymbol{R}_{\mathcal{C}}$ and displacement vector ${}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{IC}}$;
2 the IMU angular velocity ${}^{\mathcal{I}}\boldsymbol{\omega}_m$;
3 the scaled camera linear velocity $\boldsymbol{s}_1 = {}^{\mathcal{C}}\tilde{\boldsymbol{v}}_m$;
4 the IMU linear acceleration ${}^{\mathcal{I}}\boldsymbol{a} = {}^{\mathcal{I}}\boldsymbol{f}_m + {}^{\mathcal{I}}\boldsymbol{g}$;
5 the plane normal ${}^{\mathcal{C}}\boldsymbol{n}$.

Thus, apart from the same quantities as discussed in Sec. 2.4.2 for the EKF, no additional measurements are required for the nonlinear estimation scheme (2.37).

Shaping the Estimation Transient Response

We now apply the theoretical analysis developed in Sec. 2.4.3 to the case at hand. Thereby, we aimed at characterizing the transient response of the error system (2.38) in the unperturbed case, i.e., with $g(\boldsymbol{e}, t) = \boldsymbol{0}$.

Let $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = \boldsymbol{\Omega}$ be the singular value decomposition of matrix $\boldsymbol{\Omega}$ where $\boldsymbol{\Sigma} = [\boldsymbol{S} \;\; \boldsymbol{0}] \in \mathbb{R}^{p \times m}$, $\boldsymbol{S} = \mathrm{diag}(\sigma_i) \in \mathbb{R}^{p \times p}$, and $0 \leq \sigma_1 \leq \ldots \leq \sigma_p$ are the singular values of $\boldsymbol{\Omega}$. In the case under consideration ($p = 1$, $m = 3$), it is $\boldsymbol{U} = 1$ and $\sigma_1 = \|\boldsymbol{\Omega}\|$. Let also $\boldsymbol{\Lambda} = K_\alpha \boldsymbol{I}$, $K_\alpha > 0$ (scalar

gain $\boldsymbol{\Lambda}$). By then designing the gain $\boldsymbol{D} \in \mathbb{R}^{3\times3}$ as

$$\boldsymbol{D} = \boldsymbol{V} \begin{bmatrix} D_1 & \mathbf{0}_{1\times2} \\ \mathbf{0}_{2\times1} & \boldsymbol{D}_2 \end{bmatrix} \boldsymbol{V}^T \tag{2.40}$$

with $D_1 \in \mathbb{R} > 0$ and $\boldsymbol{D}_2 \in \mathbb{R}^{2\times2} > 0$, it is possible to show that, under the change of coordinates

$$\eta = \frac{1}{\sqrt{K_\alpha}\sigma_1}\zeta \tag{2.41}$$

and in the approximation $\sigma_1(t) = \|\boldsymbol{\Omega}(t)\| \approx const$, the estimation error $\zeta(t)$ (when expressed in the coordinates $\eta(t)$) obeys the following second-order linear dynamics

$$\ddot{\eta} = -D_1\dot{\eta} - K_\alpha\sigma_1^2\eta, \tag{2.42}$$

that is, a (unit-)mass-spring-damper system with stiffness $K_\alpha\sigma_1^2$ and damping D_1.

The transient response of the estimation error $\zeta(t) = s_2(t) - \hat{s}_2(t) = 1/d(t) - 1/\hat{d}(t)$ can then be imposed by properly 'placing the poles' of system (2.42). This can be achieved by:

1. taking $D_1 = 2\sqrt{K_\alpha}\sigma_1$ so as to obtain a critically-damped state evolution for (2.42) (real and coincident poles). Note that this choice univocally determines D_1 as a function of the system state (the current value of $\sigma_1 = \|\boldsymbol{\Omega}\|$);
2. *actively* enforcing $K_\alpha\sigma_1^2(t) = K_\alpha\|\boldsymbol{\Omega}(t)\|^2 = \sigma_d^2 = const$ over time for some desired $\sigma_d^2 \neq 0$, that is, by flying with a given non-zero and constant norm of the camera linear acceleration $\boldsymbol{\Omega}$ scaled by gain K_α (a free parameter whose role is detailed in the following).

From standard linear system theory, these choices then result in the following behavior for the estimation error $\zeta(t) = 1/d(t) - 1/\hat{d}(t)$

$$\zeta(t) = (1 + \sigma_d t)e^{-\sigma_d t}\zeta(t_0). \tag{2.43}$$

We conclude noting that, in general, equation (2.43) represents a *reference evolution* for $\zeta(t)$ since (i) in real-word conditions it may be hard to maintain *exactly* a $\|\boldsymbol{\Omega}(t)\| = const$ over time (condition also needed to render exactly valid the change of coordinates (2.41)), and (ii) this analysis omits the effect of the (vanishing) disturbance $g(\boldsymbol{e}, t)$ in (2.39). Nevertheless, Sec. 2.5 shows a good match between the reported results and the reference behavior (2.43).

Finally, we comment on the role of gain K_α (the only free parameter of the nonlinear observer): being $\sigma_d = \sqrt{K_\alpha}\|\boldsymbol{\Omega}\|$, the same convergence rate for $\zeta(t)$ (dictated by the value of σ_d, see (2.43)) can be equivalently obtained by either accelerating faster or by increasing gain K_α. While increasing gain K_α may always appear more convenient in terms of reduced control effort, practical issues such as noise, discretization or quantization errors, may impose an upper limit on the possible value of K_α, thus necessarily requiring a larger $\|\boldsymbol{\Omega}\|$ (larger acceleration norm) for obtaining the desired convergence speed. More details on this aspect are presented in the following Sec. 2.5.

2.5. Experimental Evaluation

This section reports the results of several experiments meant to illustrate and validate the proposed framework for ego-motion estimation on simulated and real scenarios. After a brief

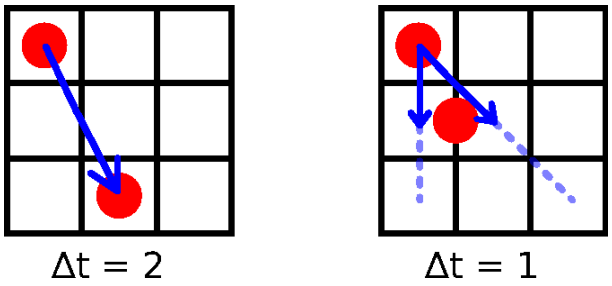


Figure 2.2.: The influence of high frame rates on the quality of the computed optical flow. An observation made in a short time frame is subject to more noise than an observation which was made over an extended period of time. The center of a feature might be located on the edge of a pixel and is thus assigned to one of the adjacent pixels. The influence of this discretization artifact increases as the vector shortens due to higher sampling rates.

discussions of some considerations when implementing the optical flow extraction in Sec. 2.5.1 and the description of the experimental setup in Sec. 2.5.2, we replicate the structure of the previous sections by first comparing in Sec. 2.5.3 the three different variants discussed in Sec. 2.2 for retrieving the scaled linear and angular velocity from optical flow decomposition. Then, we proceed to present two experiments in Sec. 2.5.4 on the outlier rejection techniques demonstrated in Sec. 2.3. Afterwards, in Sec. 2.5.5, we analyze a series of experiments involving the two scale estimation techniques illustrated in Sec. 2.4. An assessment of the ability of the employed IMU in estimating the gravity direction while the quadrotor undergoes the accelerations needed to recover the unknown scale is also presented in Sec. 2.5.6. Further considerations about the possibility of predicting (and imposing) an estimation transient behavior on the nonlinear observer of Sec. 2.4.3, and about its robustness against noise, are then discussed in Sects. 2.5.7 and 2.5.8, respectively. The experimental evaluation is finally concluded in Sec. 2.5.9 by reporting the results of a closed-loop velocity control on a real quadrotor UAV with all the computations performed on board.

2.5.1. Considerations on the Extraction of Optical Flow

We mostly made use of the established implementations provided with OpenCV[2] to process the incoming frames. In particular, we extracted an initial set of FAST (Features from Accelerated Segment Test) features [Rosten and Drummond, 2005] from the first image and tracked the features on consecutive images using the pyramidal version [Bouguet, 1999] of the Lukas-Kanade tracker [Lucas and Kanade, 1981]. We limited the number of maintained features to reduce the computational load caused by the tracking process. Whenever the size of the feature set dropped below a threshold, new FAST features were sampled and added to the set. Since both the floor and the walls of our flight arena are uniformly white, for most experiments, we placed structured carpets on the ground and posters on the walls to provide a sufficient amount of texture. No special lighting conditions were used apart from the default ceiling lights.

Interleaved Optical Flow Extraction

The comparison of velocity estimation techniques described in Sec. 2.5.3 was carried out on less powerful hardware that reached a frame rate of only 17 Hz for image acquisition and processing. In that work, we were computing the optical flow by means of subtraction of feature locations on any two consecutive frames scaled by the elapsed time. However, more powerful hardware, such as the more recent onboard computer described in Sec. 2.5 and App. A.5, supports considerably higher frame rates. Since efficient feature tracking is always carried out in the discrete pixel space, a certain rounding error to the next pixel coordinate cannot be avoided. An increased frame rate, however, results in the observation of even shorter optical flow vectors prone to an highly increased rounding error as explained in Fig. 2.2.

Software solutions or hardware improvements, such as sub-pixel refinement or an increased resolution of the sensor, can help to reduce the effect this problem. However, both solutions increase the computational load. While we do use sub-pixel refinement, we also extended the temporal baseline between the compared feature sets to allow for the observation of longer optical flow vectors. Implemented in an interleaved manner, the system can still make use of the benefits of an increased frame rate. Thus, we developed a method to extract the optical flow over a predefined baseline of length δ_t while the best possible frame rate is still maintained. Therefore, features are tracked from one frame to the next as they are acquired, but the resulting feature locations for each frame are stored together with a time stamp. Afterwards, the frame with the time stamp closest to $t_{now} - \delta_t$ is used for the computation of the optical flow relative to the most recent image.

Equally Distributed Feature Tracking over the Entire Image

In order to provide reliable state estimates, the tracked optical features should be equally distributed over the entire image independent of the camera motion. This requirement imposed the ability of the system to both actively sample features in sparse parts of the image as well as to drop features from the set of tracked points in over-represented areas. At the same time, the maximum number of all features should always be limited by a configurable bound to limit the computation time of the feature tracking algorithm.

To this end, we propose a segmentation of the image into a configurable amount of tiles. For each tile, a desired lower and upper bound on the number of features is specified. While the upper bound is implicitly defined as the maximum number of features for the entire image divided by the number of tiles, the lower bound is a free parameter. After each feature tracking step, the number of features is counted for each tile. If more features than allowed are found for one tile, the feature with the lowest Shi-Tomasi score [Shi and Tomasi, 1994] is removed until the number of features is again within the allowed range. Similarly, when the number of features on one tile drops below the chosen minimum, new FAST features [Rosten and Drummond, 2005] are sampled on that tile. However, regardless of the number of detected features, a threshold on the Shi-Tomasi score is applied to exclude features of low quality in, e.g., texture-free regions of the image. Therefore, the desired minimal amount of features is intendedly not guaranteed.

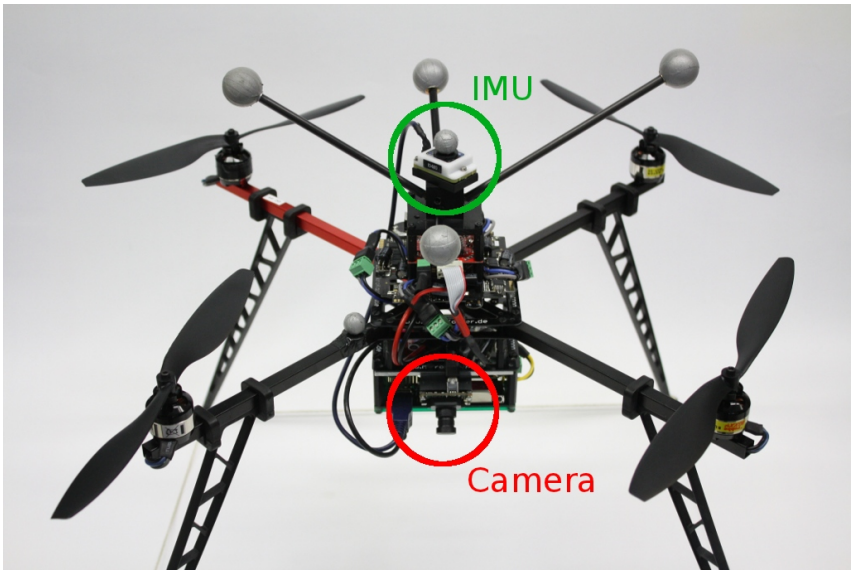


Figure 2.3.: Quadrotor UAV used for the experiments with highlighted locations of IMU and camera. The x-axis of the body frame is oriented along the red metal beam of the frame.

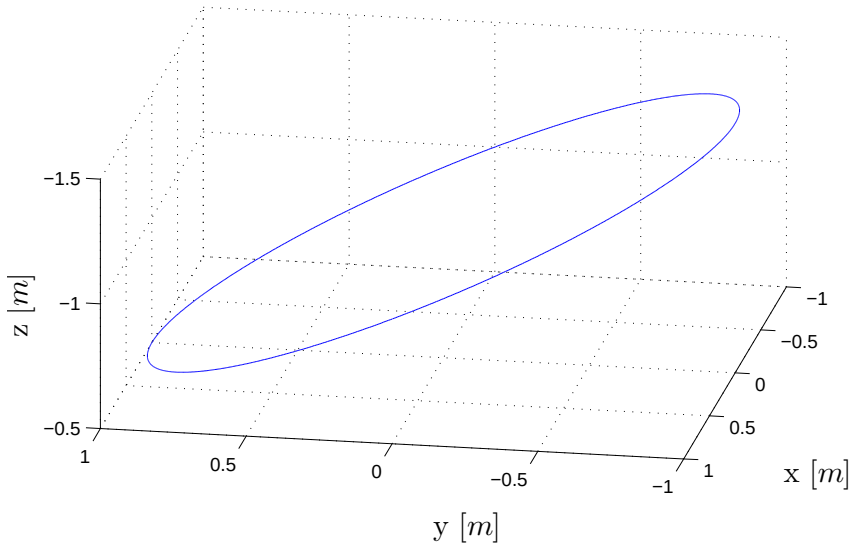


Figure 2.4.: Circular trajectory used for the experiments to compare the three velocity estimation algorithms *V1–V3*. The trajectory has a diameter of 2 m and lies on a plane tilted by 26.5° with respect to the ground. The UAV completes one revolution every 10 s. The height varies between 0.5 m and 1.5 m. A similar trajectory, but lying on a horizontal plane, has also been used for the experiments of Sec. 2.5.5

2.5.2. Experimental Setup

For our experiments, we made use of a quadrotor UAV that is described in App. A.1. The free *TeleKyb* software framework [Grabe et al., 2013b] was used as middleware and control software. The location of all relevant sensors and frames can be found in Figs. 2.1 and 2.3. The quadrotor was equipped with an additional IMU (frame $\mathcal{I}$, see App. A.3) to provide the measurements of the specific acceleration ${}^{\mathcal{I}}\boldsymbol{f}_m$, the angular velocity ${}^{\mathcal{I}}\boldsymbol{\omega}_m$, and the gravity vector ${}^{\mathcal{I}}\boldsymbol{g}_m$ at 200 Hz. The gravitational vector was estimated internally by the IMU via fusion of the measurements from the accelerometers, rate gyroscopes, temperature sensor, and a 3D magnetometer (see Sec. 2.5.6 for an analysis of this orientation estimate). The visual input was provided by a pre-calibrated MatrixVision mvBlueFox camera (frame $\mathcal{C}$, see App. A.2) mounted down-facing on the UAV. The onboard processing was carried out on a small PC with an Intel Atom 1.8 GHz dual core processing unit (see App. A.5), while velocity command inputs were transmitted to the vehicle using a wireless serial interface. Note that for the following experiments on the comparison of velocity estimation techniques, older hardware with just a single 1.66 GHz CPU core was used (see App. A.4).

2.5.3. Comparison of Algorithms V1–V3 for Ego-Motion Estimation from Optical Flow

We start by presenting a comparison of the three modalities of IMU integration into the velocity estimation process as discussed in Sects. 2.2.2–2.2.4 for recovering $\boldsymbol{v}/d$ and $\boldsymbol{\omega}$ during flight. For the reader's convenience, we recall that algorithm *V1* (Sec. 2.2.2) is only based on the perceived optical flow for obtaining $(\boldsymbol{v}/d,\, \boldsymbol{\omega})$, algorithm *V2* (Sec. 2.2.3) exploits the additional (independent) measurement of $\boldsymbol{\omega}$ from the onboard rate gyroscopes, and algorithm *V3* (Sec. 2.2.4) further exploits the independent knowledge of the scene normal $\boldsymbol{n}$ that is approximated to coincide with the gravity vector.

To allow for a controlled and direct comparison of the three algorithms *V1*–*V3*, we recorded a dataset while flying along a circular trajectory of 2 m in diameter that is shown in Fig. 2.4. The trajectory was chosen to have the UAV accelerating sinusoidally along the three Cartesian directions with a period of 10 s, and a maximum speed and acceleration of 0.6 m/s and 0.296 m/s^2, respectively. The height varied from 0.5 m to 1.5 m along the trajectory. Additionally, the vehicle was periodically rotated around its body z-axis in the range of $[-70° \dots 70°]$. The quadrotor relied on an external optical motion tracking system that also provided a ground truth regarding the position of the UAV. A ground truth velocity has then been derived from this ground truth position reading. Onboard hardware was used to record vision and IMU data during flight. Afterwards, all three algorithms of Sects. 2.2.2–2.2.4 were tested offline on this common dataset to allow for a direct comparison.

Figure 2.5 summarizes the results of this first experiment. The plots in Figs. 2.5a, 2.5c, and 2.5e show a superimposition of the three methods to estimate the linear velocity $\boldsymbol{v}$ against the ground truth provided by the derived position measurements of the external tracking system. In all three cases, for the purpose of presentation only, the plotted metric linear velocity $\boldsymbol{v}$ was recovered from the measured $(\boldsymbol{v}/d)_m$ using the ground truth measurement of d. Figure 2.5g then reports the corresponding error norms.

[2] `opencv.willowgarage.com`

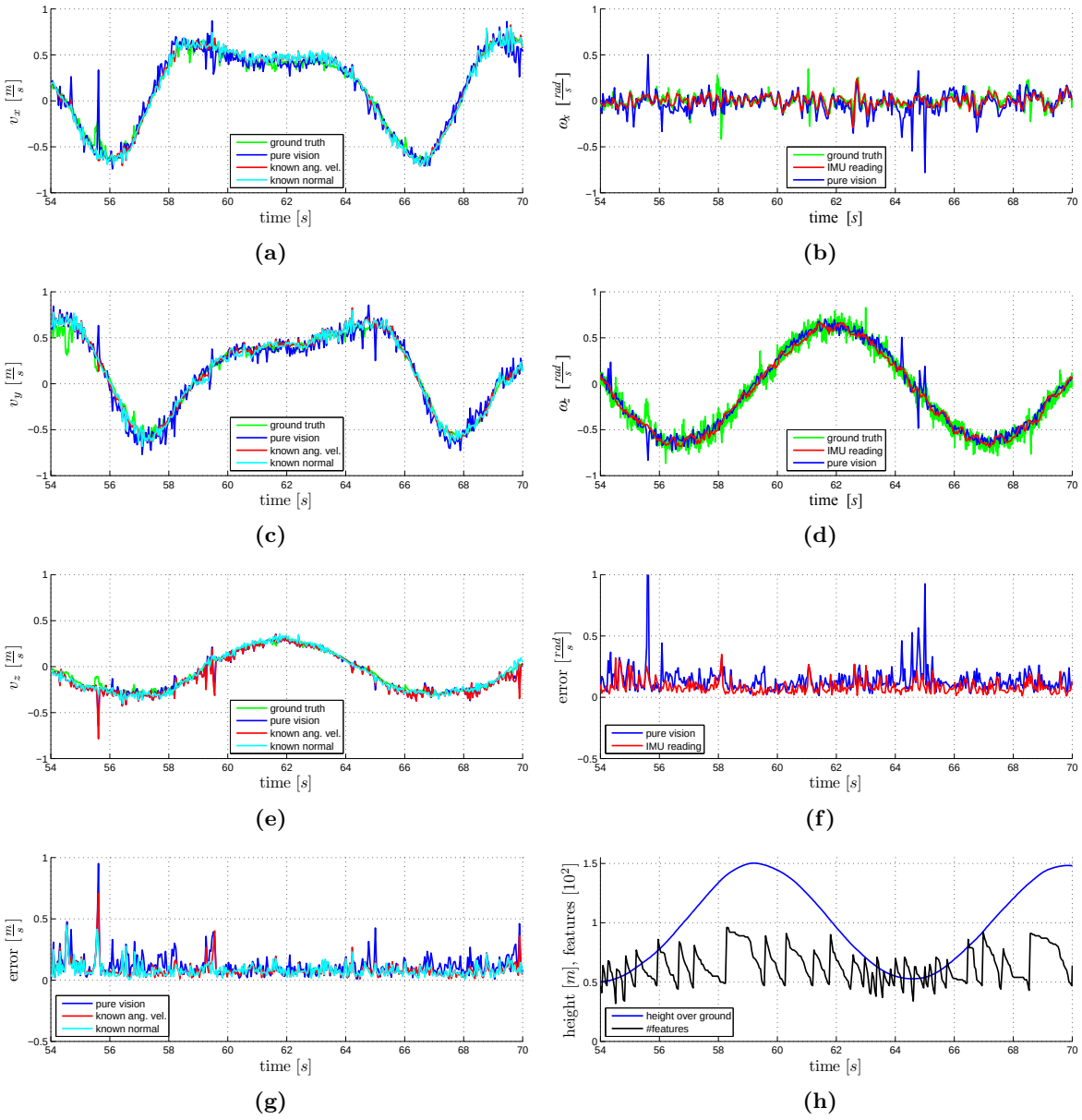


Figure 2.5.: Comparison of the three grades of IMU integration *V1* ('pure vision'), *V2* ('known ang. vel'), and *V3* ('known normal') against a ground truth for the estimation of the scaled linear and angular velocity. For the purpose of visualization only, the non-metric linear velocity estimates have been scaled using the scene depth d as obtained from the optical tracking system. (a) Linear velocity along the x, (c) y, and (e) z axis. (b) Angular velocity for the rotation around the x and (d) z axis. Results similar to the ones presented for the x axis were found for rotations around the y axis. (g) Norm of the error between ground truth and both the estimated linear and (f) angular velocities. (h) Altitude d of the vehicle above the scene and the number of tracked features: this latter quantity influences the quality of the tracking system.

Employed algorithm:	*V1* (2.2.2)	*V2* (2.2.3)	*V3* (2.2.4)
Linear velocity			
Mean error	0.134 m/s	0.117 m/s	0.113 m/s
Standard deviation	0.094 m/s	0.093 m/s	0.088 m/s
Angular veclocity			
Mean error	0.151 rad/s	IMU	IMU
Standard deviation	0.110 rad/s	IMU	IMU

Table 2.1.: Comparison of the errors in retrieving the scaled linear and angular velocity for the three levels of IMU integration as described in Sects. 2.2.2–2.2.4: the pure visual estimate (*V1*), the inclusion of angular velocity readings from the rate gyroscopes (*V2*), and the additional integration of the estimated plane normal vector (*V3*).

From these results, we can conclude that all three algorithms *V1*–*V3* yield a satisfactory performance. However, one can also note that the increasing integration of IMU measurements for algorithms *V2*–*V3* reduces the mean error of the linear velocity estimation compared to algorithm *V1*, see Table 2.1. In particular, failures of the feature detection system, e.g., at second 55.5 in the presented data, can be partly compensated using the IMU measurements.

A comparison of the angular velocity retrieved from the visual system (algorithm *V1*) against the gyroscope readings and the ground truth of the external tracking system is given in Figs. 2.5b and 2.5d while the error compared to the ground truth is plotted in Fig. 2.5f. We can note that the IMU provides a less noisy estimate almost free of spikes, while the camera-based estimation is prone to outliers in the case of high eminent optical flow during low altitude flights.

Figure 2.5h indicates the origin of the irregularities found in the error plots. Since the vehicle is moving with a constant speed in world frame, $\boldsymbol{v}/d$ increases with lower altitudes d. Thus, the feature detector is forced to sample new features more often since they vanish from the field of view faster. This both slows down the algorithm due to the frequent feature re-sampling processes and does only allow for the maintenance of a sets of 30 to 65 feature pairs instead of the, in this case, intended 50 to 100 pairs.

Table 2.1 provides a quantitative summary of all reported results. One can again notice the improvements obtained when fusing the decomposition of the perceived optical flow with measurements from the IMU with, in particular, algorithm *V3* performing slightly better than *V2*.

2.5.4. Experimental Evaluation of the Outlier Rejection Approach

To investigate the robustness of the outlier rejection algorithm proposed in Sec. 2.3, we conducted two experiments. The aim of these experiments was the evaluation of our feature filtering approach in the presence of clutter and the therefore violated assumption of a planar environment. To allow for a comparison of both the proposed outlier rejection system and the unmodified algorithm on the same sensory input, we recorded the video stream together with the IMU measurements. For this dataset, we commanded the UAV using feedback from an external optical motion tracking system along an arbitrary trajectory commanded by a human operator. The motion tracking system was additionally used to provide a reliable ground truth. The quadrotor

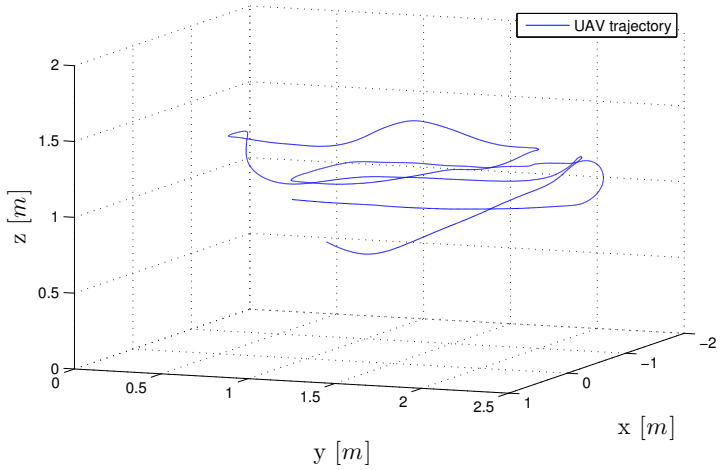


Figure 2.6.: A section of the trajectory that was used for the outlier filtering experiments. The section shown here was recorded during the 30 s period of the evaluation that corresponds to the plots presented in Fig. 2.7. The trajectory extends 2.2 m in x, 2.1 m in y, and 1.2 m in the z direction.

took off from a flat horizontal floor and moved towards a table placed on the right side of the room. The UAV was then commanded to explore the room in which two more objects were placed: a small ladder and a stool with an object on it. The resulting trajectory is partly shown in Fig. 2.6. The objects and the walls of the room were textured using the same pattern to equalize the initial feature density on the objects and the planar walls.

For the evaluation of the experiment, we consider a total flight time of 120 s. Figs. 2.7a–2.7c report the recovered scaled linear velocity for a 30 s period of this experiment. Each of the three plots shows a comparison between: (*i*) the unmodified algorithm which, for each image frame, considers all observed flow vectors for the estimation of the linear velocities, (*ii*) the suggested clustering approach of planar features, and (*iii*) the ground truth obtained from the optical motion tracking system. To allow for a comparison of the non-metric velocity estimates with the ground truth, during the generation of the plots, the obtained estimates were scaled by the distance d to the dominant plane as obtained from the motion tracking system.

The error relative to the ground truth for both system during the presented 30 s section of the dataset is shown in Fig. 2.7d. Computing the mean error over the entire 120 s of the experiment, we find that the original approach reaches an accuracy of 0.111 m/s, while the proposed improved solution yields an error of 0.089 m/s Thus, the proposed outlier rejection approach was able to improve the ego-motion estimation by 25% compared to the unfiltered system. The standard deviation improved slightly from an initial 0.087 m/s to 0.081 m/s when using the modified algorithm. Figs. 2.7a–2.7d indicate that the overall estimation error when using the suggested outlier rejection step is mostly reduced by enhanced velocity estimates in the horizontal plane. However, the vertical velocity estimate is only improved to a small extend.

Figure 2.8 demonstrates the accuracy of the proposed feature rejection process for features on non-planar regions of two different environments. In the case of Fig. 2.8c, our algorithm is able to reject the features located on a table of 0.75 m height and a board close by. Similarly, in the scene of Fig. 2.8d, the system excludes features on the back wall and two irregularly

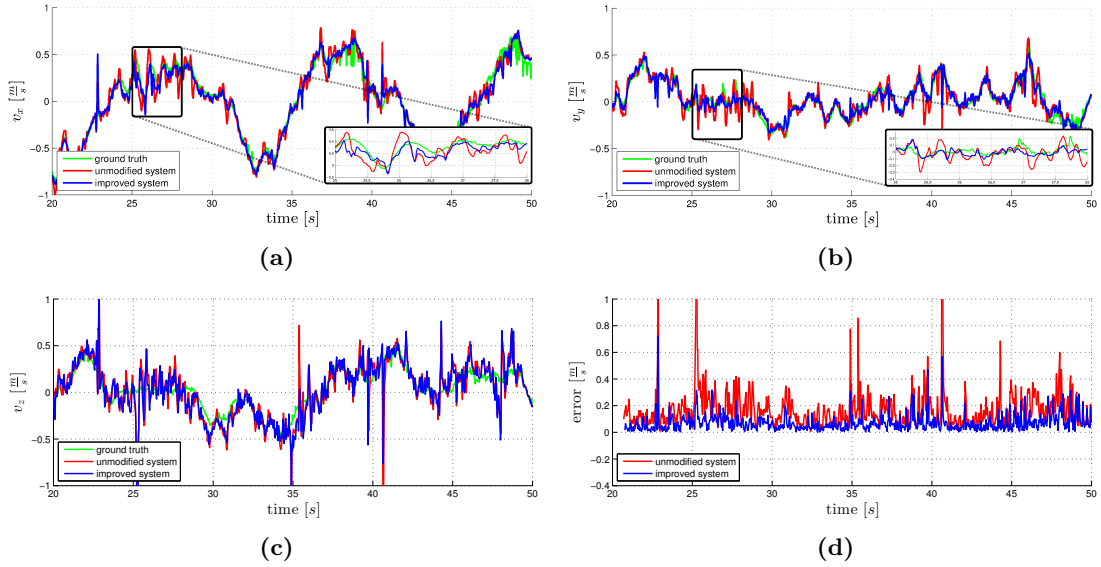


Figure 2.7.: Estimated linear velocities in camera frame along the (a) x, (b) y, and (c) z axis for both the unmodified and the improved algorithm together with the ground truth. (d) Error of the original and improved approach compared to the ground truth.

shaped objects. To quantitatively verify the ability of our system to reject outliers, we counted the number of features on 5 randomly selected images. In average, 20% of all initially detected features were actually outliers. Out of these, our system rejected 94% successfully.

The algorithm was mainly designed to avoid a false-positive classification of outliers and therefore the addition of unsuitable features to the observed dominant plane. Consequently, a relatively high false-negative classification rate was accepted.

2.5.5. Comparison of the two Scale Estimation Schemes

In the following, the two scale estimation schemes of Sects. 2.4.2 and 2.4.3 are first evaluated in simulation to compare their performance in ideal conditions. Subsequently, results obtained on recoded sensor data from real flights are presented. We also compare our solution against a recent related work by qualitatively reproducing the experiments presented in [Weiss et al., 2012b].

Before starting the discussion, we note that both filters allow for two distinct possibilities to retrieve the metric linear velocity $\boldsymbol{v}$. Indeed, a first possibility is to just set $\hat{\boldsymbol{v}} = (\boldsymbol{v}/d)_m \hat{d}$ exploiting the estimated $\hat{d}$ and the measured $(\boldsymbol{v}/d)_m$ from the optical flow decomposition. However, a second possibility is to exploit the *internal* estimation of $\boldsymbol{v}$ maintained by both filters, that is $\hat{\boldsymbol{v}}$ from (2.30) in the EKF case, and $\hat{\boldsymbol{v}} = \hat{\boldsymbol{s}}_1/\hat{s}_2$ for the nonlinear observer (see Sect. 2.4.3). Both possibilities have their advantages and disadvantages: in general, the second possibility may result in a less noisy but potentially more delayed estimation of $\boldsymbol{v}$ (caused by the 'filtering' action of both estimation schemes). Nevertheless, for the sake of obtaining a less noisy estimate, all the following simulation and experiment results rely on this second possibility.

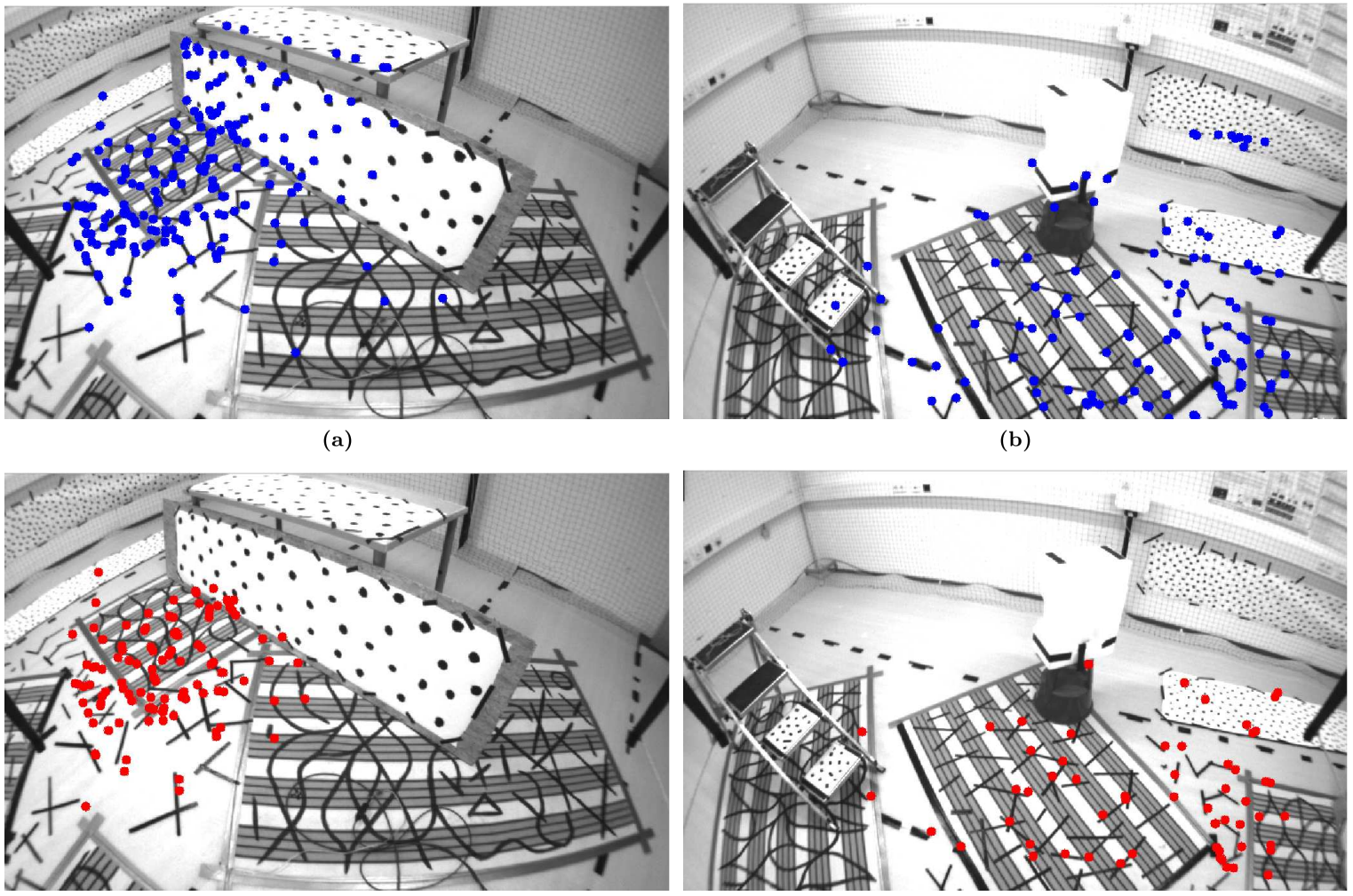


Figure 2.8.: Direct comparison of (a,b) the original and (c,d) the improved algorithm. While the unmodified system considers all observed features, the outlier filtering approach is able to exclude the features on the table and the upper region of the board that was leaned against the table. In the second environment, the features on the back wall and, apart from one false-positive classification, the two objects were removed successfully.

Simulated Data

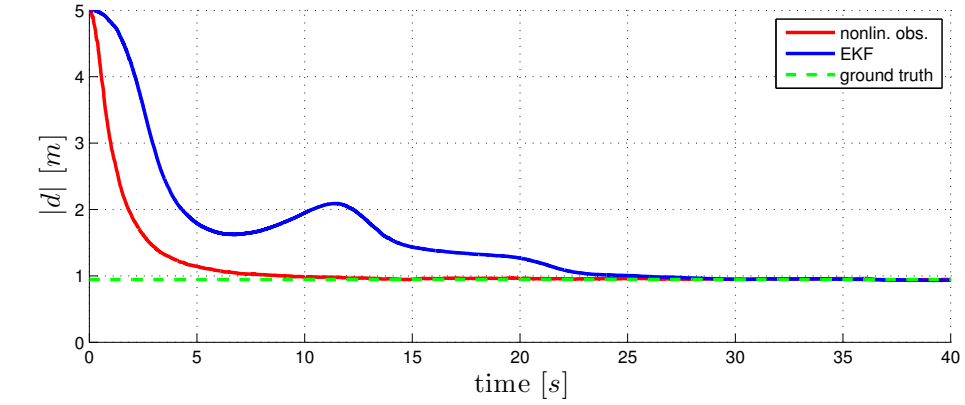
To illustrate the convergence of the proposed estimation schemes, we generated a synthetic acceleration profile together with the corresponding (simulated) sensor readings. This resulted in a camera motion similar to the horizontal circular trajectory used for the evaluation on recorded datasets in following section with an associated constant acceleration norm $\|\boldsymbol{\Omega}\| \approx 0.296\,\mathrm{m/s^2}$. All generated sensor readings were perturbed with an additive zero-mean Gaussian noise with covariance matrices taken from the real sensor characteristics: $0.00004\,\boldsymbol{I}_3\,\mathrm{m/s^2}$, $0.00002\,\boldsymbol{I}_3\,\mathrm{rad/s}$ and $0.00001\,\boldsymbol{I}_3\,1/\mathrm{s}$ for ${}^{\mathcal{I}}\boldsymbol{f}_m$, ${}^{\mathcal{I}}\boldsymbol{\omega}_m$, and for the unscaled linear velocity estimate from optical flow $({}^{\mathcal{C}}\boldsymbol{v}/d)_m$, respectively. The same covariance matrices were employed in the steps (2.24)–(2.28) of the EKF (thus fixing all the free parameters of the filter). As for the nonlinear observer, its only free parameter (gain K_α) was taken as $K_\alpha = 6$: this value was chosen so as to obtain, for both filters, the same noise level in the estimated $\hat{d}(t)$ after convergence. This ensured a 'fair' comparison among the EKF and the nonilnear observer in terms of overall performance. The choice of $K_\alpha = 6$ resulted in a value of $\sigma_d = \sqrt{K_\alpha}\|\boldsymbol{\Omega}\| \approx 0.725\,\mathrm{m/s^2}$ for the ideal estimation error convergence in (2.43).

To demonstrate the robustness of both filters to unknown initial scene depth estimates $\hat{d}$, the initial conditions were chosen as $\hat{d}(t_0) = 5\,\mathrm{m}$ and ${}^{\mathcal{C}}\hat{\boldsymbol{v}}(t_0) = [0\ 0\ 0]^T\,\mathrm{m/s}$. At the beginning of the experiment, the actual distance from the scene was $d(t_0) = 1\,\mathrm{m}$ while the velocity along the trajectory had a norm of $\|{}^{\mathcal{C}}\boldsymbol{v}(t_0)\| \approx 0.6\,\mathrm{m/s}$. To indicate the uncertainty of the initial state, we chose $\boldsymbol{\Sigma}(t_0) = \mathbf{1}_{4\times4}$.

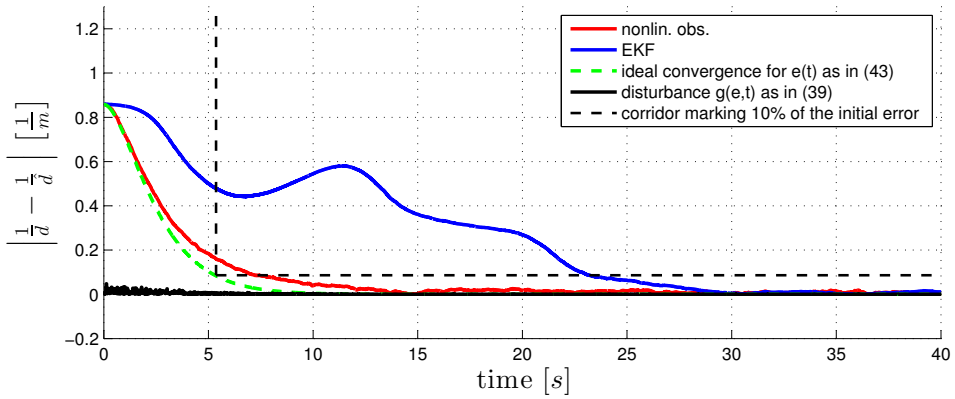
Figure 2.9a reports a comparison of the estimation performance of the scene depth d by both the EKF and the nonlinear observer on the same UAV trajectory and sensor readings, together with the ground truth. One can then appreciate the faster convergence of the estimation error for the nonlinear observer with respect to the 'fully-informed' EKF (in the sense of being aware of the system noise characteristics). This better performance can be ascribed to the lack of linearization of the system dynamics in the nonlinear observer case.

Furthermore, Fig. 2.9b shows the behavior of the estimation error $\zeta(t) = 1/d(t) - 1/\hat{d}(t)$ for both filters, with the superimposed 'ideal' transient response (2.43). We can then note the very good match of this latter ideal response with the behavior of the estimation error $\zeta(t)$ for the nonlinear observer base algorithm, thus confirming the theoretical analysis of Sec. 2.4.3. In this figure, a horizontal band at 10 % of the initial error is also shown: according to the ideal response (2.43), the estimation error of the nonlinear observer should have dropped below 10 % of the initial error $\zeta(t_0)$ after 5.36 s. The real observer is slightly delayed compared to this prediction because of the presence of the perturbation term $g(\boldsymbol{e}, t)$ in (2.39) (also shown in the plot). Indeed, presence of this term (neglected in the analysis) initially slows down the convergence rate of $\hat{d}(t)$ with respect to its predicted behavior. Note that, ideally, one should have found $g(\boldsymbol{e}, t) = 0$ in this case, since $\boldsymbol{v}$ has always been perpendicular to the plane normal $\boldsymbol{n}$ (see again (2.39)). However, the simulated noise in the measurements resulted in the presence of a small $\boldsymbol{v}_z$ which gave rise to the initial value of $g(\boldsymbol{e}, t) > 0$. Additional details on the possibility to predict the convergence time of the nonlinear observer are also given in Sec. 2.5.7.

The linear velocities estimated by both observers are shown in comparison in Fig. 2.10. Similar to the estimation of the scale factor d, the estimate for the metric velocity $\boldsymbol{v}$ converges faster in the case of the nonlinear observer. However, both scale estimation schemes provide very reliable velocity measurements after 30 s as shown in Fig. 2.11 in which the norm of the velocity estimation errors are reported.

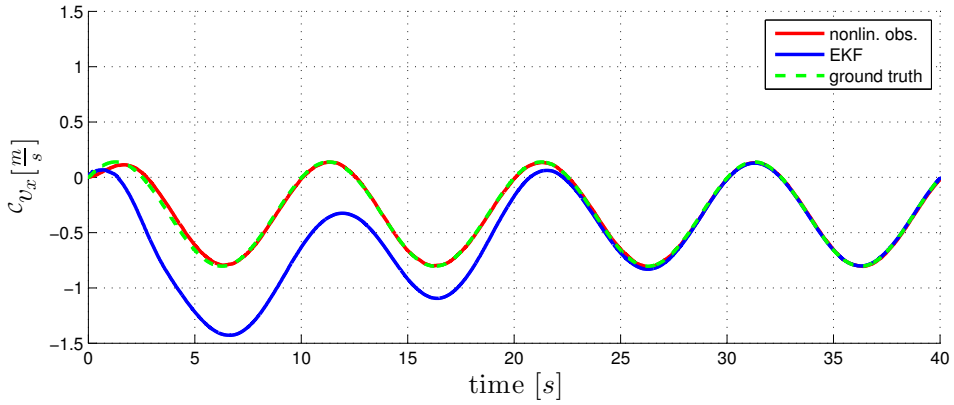


(a)

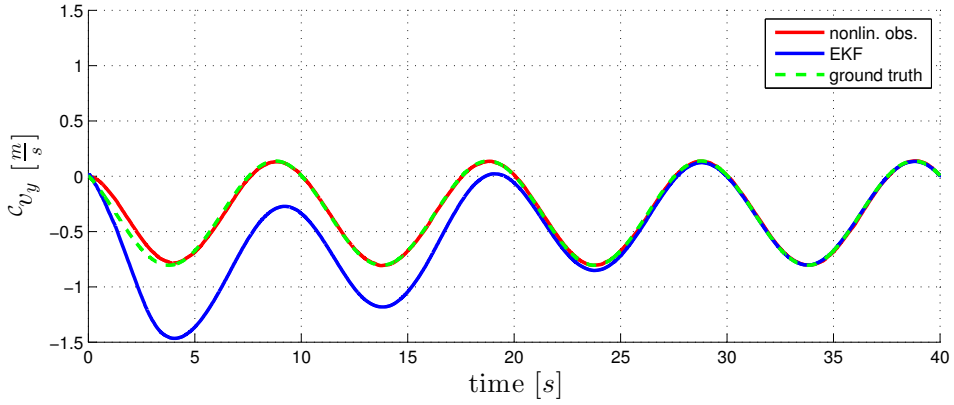


(b)

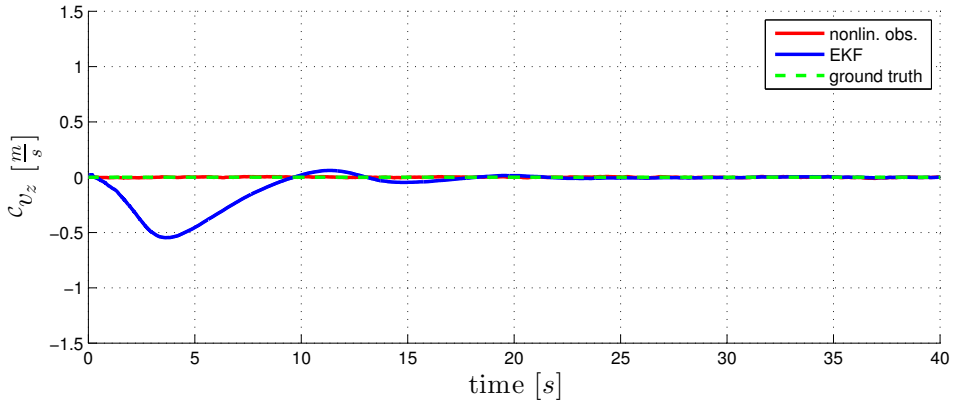
Figure 2.9.: Estimated plane distance $\hat{d}$ using a simulated dataset. (a) Estimated distance $\hat{d}$ and (b) the estimation error $|1/d - 1/\hat{d}|$ for the nonlinear observer and the EKF based approach. Additionally, the ideal convergence of the nonlinear observer as per (2.43) is plotted together with the influence of the neglected disturbance $g(\boldsymbol{e}, t)$ given in (2.39). The vertical dashed line denotes the predicted time at which the estimation error of the nonlinear observer should have dropped below the threshold of 10% of the initial error (highlighted with a horizontal dashed line).



(a)



(b)



(c)

Figure 2.10.: Estimated linear velocity from simulated noisy measurements compared to the ground truth in the (a) x, (b) y, and (c) z direction.

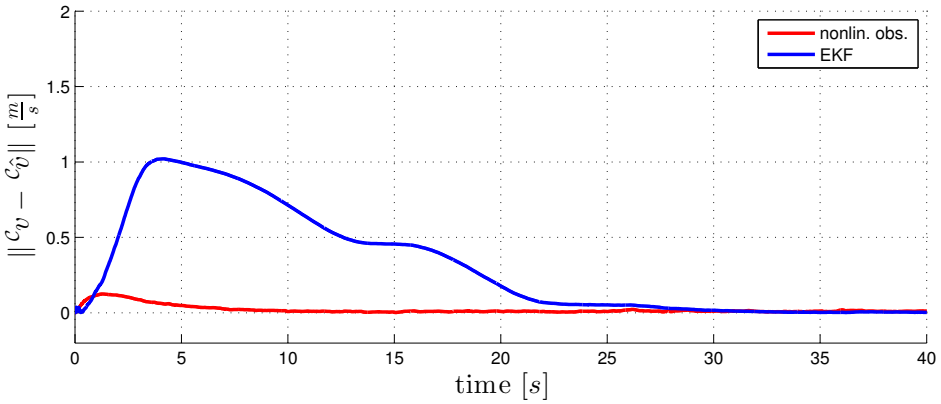


Figure 2.11.: Norm of the error for the linear velocity estimates on simulated data for the two filtering approaches with respect to the ground truth.

	EKF	nonlin. obs.
Experiments in simulation		
RMS error after convergence (scale)	0.0075 m	0.0078 m
RMS error after convergence (velocity)	0.0071 m/s	0.0111 m/s
Experiments on recorded data		
RMS error after convergence (scale)	0.0525 m	0.0357 m
RMS error after convergence (velocity)	0.0954 m/s	0.0833 m/s
Comparison with [Weiss et al., 2012b]		
RMS error x-axis (velocity)	0.0101 m/s	0.0074 m/s
RMS error y-axis (velocity)	0.0141 m/s	0.0095 m/s
RMS error z-axis (velocity)	0.0107 m/s	0.0114 m/s
Improvement over [Weiss et al., 2012b] (avg)	250 %	323 %

Table 2.2.: Quantitative comparison of the two scale estimation approaches as presented in Sec. 2.4. These results are discussed in Sec. 2.5.5.

In Tab. 2.2, the quantitative results for the comparison of both scale estimation approaches are summarized. The listed Root Mean Square (RMS) error for both systems was computed after 30 s had passed and both estimates had converged.

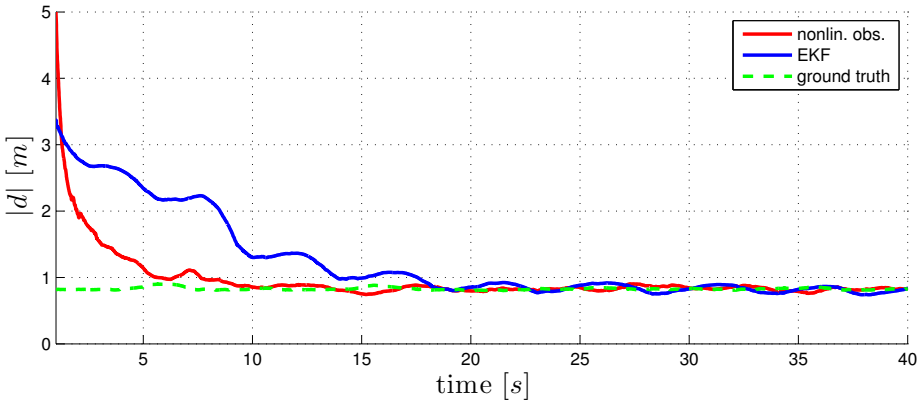
Recorded Data

After the tests in simulation, we compared both scale estimation approaches on a recorded dataset that was collected during a flight of a real quadrotor UAV. The chosen circular trajectory was similar to the one used in Sec. 2.5.3 and shown in Fig. 2.4. However, to yield a horizontal trajectory at a constant height, the sinusoidal command in the z direction was initially omitted to allow for a simplified interpretation of the results. The flight was again controlled using an external motion tracking system that was additionally used to provide ground truth information. The chosen trajectory was characterized by an acceleration of $\|\boldsymbol{\Omega}\| \approx 0.296\,\mathrm{m/s^2}$, thus resulting in a $\sigma_d = \sqrt{K_\alpha}\|\boldsymbol{\Omega}\| \approx 0.725\,\mathrm{m/s^2}$ for the ideal system (2.43). However, on the real quadrotor, the actual accelerations were found to be slightly higher than those expected from the ideal circular trajectory due to additional constant regulations necessary to overcome small real-world disturbances. All sensor offsets were calibrated before the recording of the dataset, and the covariance matrices of ${}^{\mathcal{I}}\boldsymbol{f}_m$, ${}^{\mathcal{I}}\boldsymbol{\omega}_m$ and $({}^{\mathcal{C}}\boldsymbol{v}/d)_m$ were estimated over a period of 60 s. Again, both filters were initialized with $\hat{d}(t_0) = 5\,\mathrm{m}$ and ${}^{\mathcal{C}}\hat{\boldsymbol{v}}(t_0) = [0\ 0\ 0]^T\,\mathrm{m/s}$ while the actual initial height was $d(t_0) \approx 1\,\mathrm{m}$ and the initial velocity was $\|{}^{\mathcal{C}}\boldsymbol{v}(t_0)\| \approx 0.6\,\mathrm{m/s}$. The uncertainty of the initial state was indicated with the same choice of $\boldsymbol{\Sigma}(t_0) = \mathbf{1}_{4\times4}$. The gain for the nonlinear observer K_α was selected as $K_\alpha = 6$ given the findings in simulation.

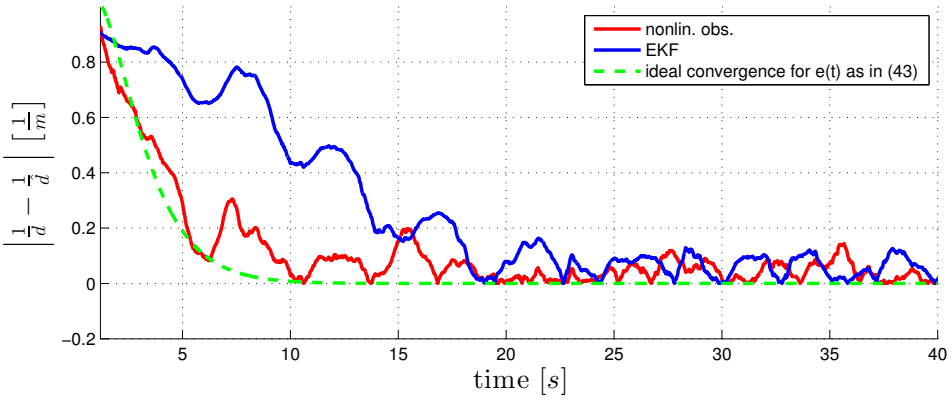
Figure 2.12a shows the behavior of the estimated scene distance $\hat{d}(t)$ for both filters with the superimposed ground truth $d(t)$. The behavior of the estimation error $\zeta(t) = 1/d(t) - 1/\hat{d}(t)$ is shown in Fig. 2.12b, and the behavior of $d(t) - \hat{d}(t)$ is also reported in Fig. 2.12c for a better appreciation of the convergence behavior. Again, from these plots one can note the good match among the actual estimation error of the nonlinear observer against the ideal response (2.43). On this dataset from real sensor measurements, the predicted convergence rate was more accurate than in the case of simulated data. Consequently, convergence is reached faster for both approaches as well. This can be explained by the fact that the real acceleration of the vehicle was slightly higher than the acceleration expected from the commanded circular trajectory alone, and this in turn compensated for the initial 'disturbing' effects of the perturbation $g(\boldsymbol{e}, t)$ in (2.39).

Similar to the convergence of $\hat{d}$, the estimated metric linear velocities are presented in Fig. 2.13. The error against the ground truth as shown in Fig. 2.14 demonstrates a fast convergence rate for the nonlinear observer. Nevertheless, the EKF yielded a reliable output after approximately 20 s as well. All quantitative results can be compared in Tab. 2.2.

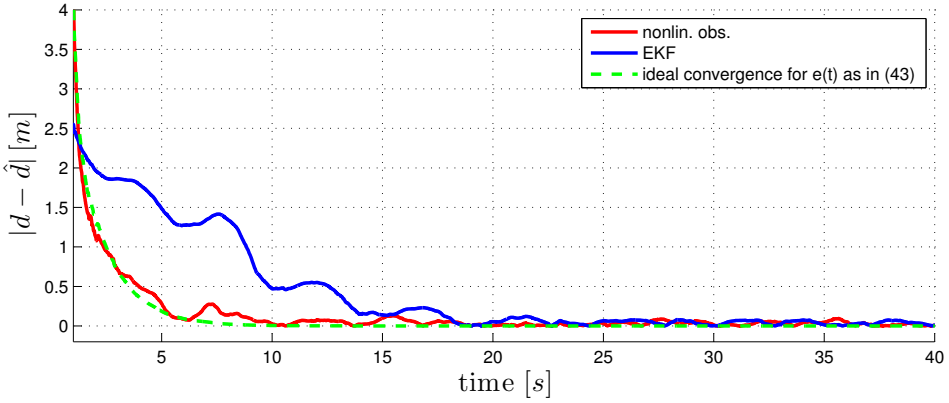
As an additional validation of our approach, we also tested both sensor fusion algorithms on an inclined circular trajectory similar to the one depicted in Fig. 2.4 in order to investigate the case of a *time-varying* plane distance $d(t)$. The plots shown in Fig. 2.15 again confirm that both the scene depth and the vertical component of the linear velocity can be reliably estimated despite the more challenging UAV motion concerning a time-varying $d(t)$. Similarly, an almost perfect match between the actual and ideal transient response of the estimation error can be observed.



(a)

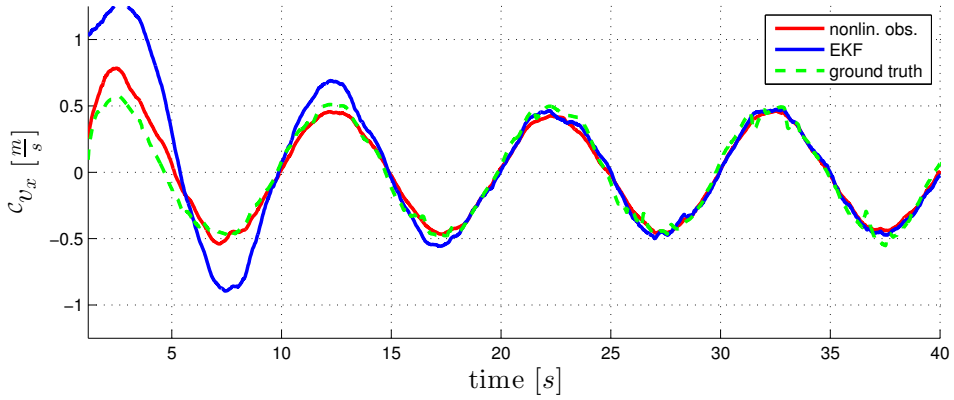


(b)

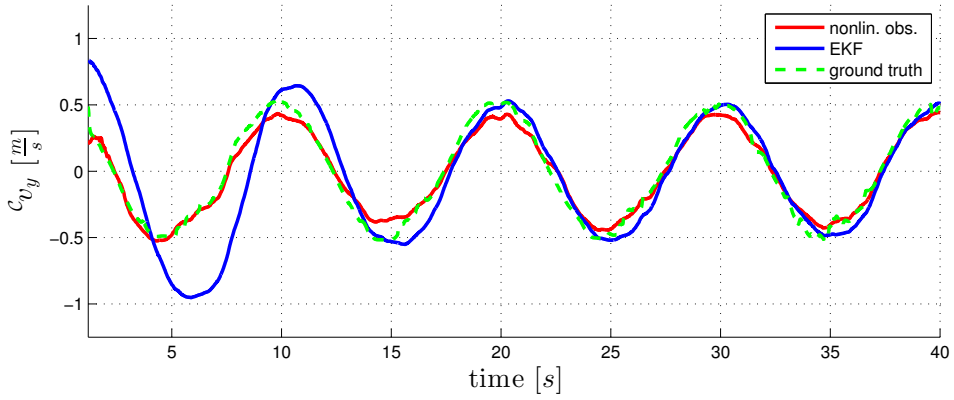


(c)

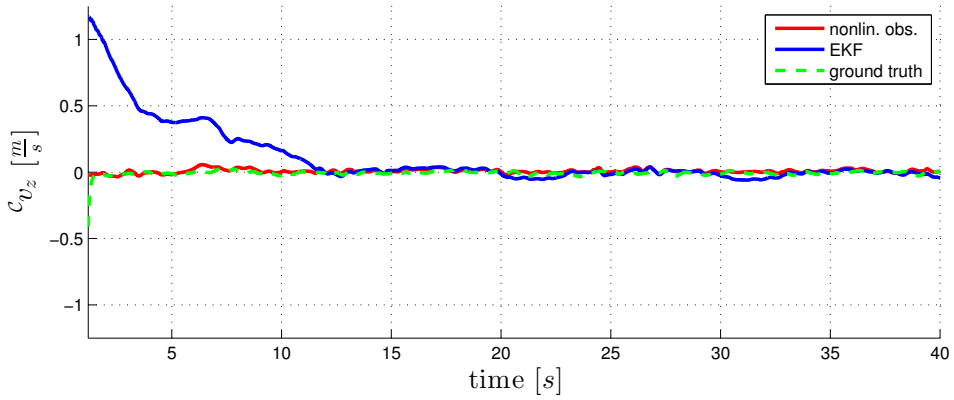
Figure 2.12.: (a) Metric scale factor d estimated from a recorded dataset, the corresponding estimation errors $1/d(t) - 1/\hat{d}(t)$ (b) and $\|d(t) - \hat{d}(t)\|$ (c) compared against the ground truth. Convergence from an initial assumption of a plane distance of 5 m to the real distance of approximately 1 m is shown, together with the ideal transient response of the nonlinear observer as per (2.43). Note the good match between the predicted and actual behavior of the estimation error and the faster overall convergence obtained with the proposed nonlinear observer compared to the EKF scheme.



(a)



(b)



(c)

Figure 2.13.: Estimated linear velocities from a recorded dataset with superimposed ground truth information in the (a) x, (b) y, and (c) z direction.

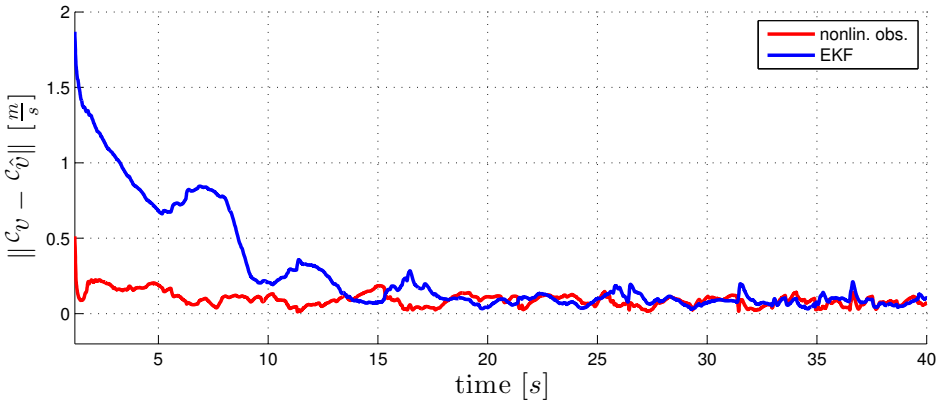


Figure 2.14.: Norm of the velocity estimation error on a recorded dataset for the two sensor fusion approaches with respect to the ground truth.

Motivated by these good results of the nonlinear observer scheme in comparison to the EKF, all following experiments exploit the nonlinear filtering approach.

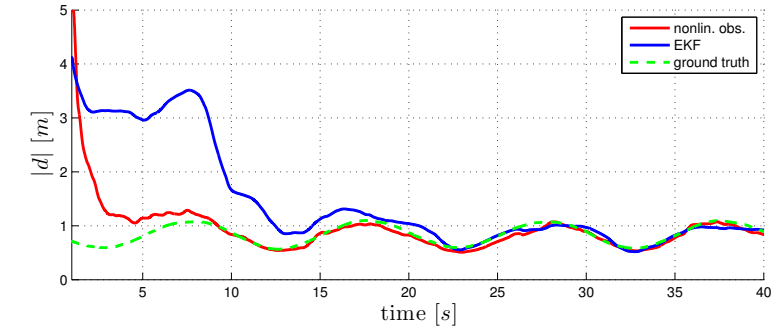
Comparison to Previous Work

For a direct comparison with the state-of-the-art velocity scale estimation approach, we repeated the experiment in [Weiss et al., 2012b] under similar conditions using the nonlinear observation scheme. Therefore, the camera-IMU system was moved on a trajectory consisting of small sinusoidal hand-held motions. This resulted in a vehicle trajectory with an amplitude of about 0.1 m/s, a frequency of approximately 1 Hz, and a height of 0.5 m. An optical motion tracking system was used to obtain a ground truth. All three Cartesian directions were tested individually.

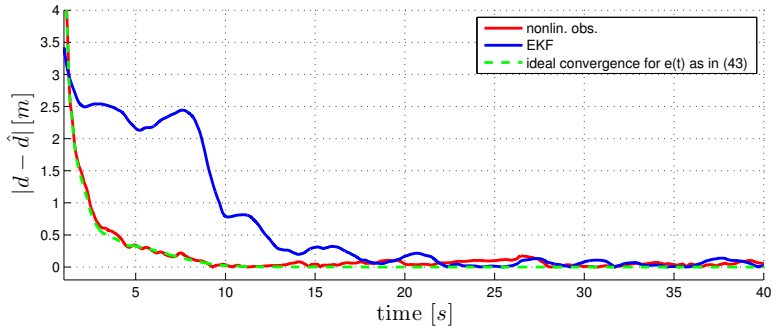
Figure 2.16 shows the metric velocity estimates compared to the ground truth for the nonlinear observer. On this trajectory, we found an RMS error of $[0.0074, 0.0095, 0.0114]$ m/s for the three Cartesian directions, respectively. This allows us to compare our results to the error of $[0.028, 0.035, 0.025]$ m/s reported in [Weiss et al., 2012b]. Therefore, using the nonlinear observer, we could obtain an average improvement of 320 % compared to the estimation accuracy of the state-of-the-art velocity estimation system. All results are summarized in Tab. 2.2. Note that in [Weiss et al., 2012b], the experiments were carried out at a height of $d \approx 1$ m as opposed to $d \approx 0.45$ m in our experiments due to technical simplifications. Since the estimation error is typically directly proportional to the scene depth d, one can expect an improvement of approximately 700 % over [Weiss et al., 2012b] under normalized conditions when transformed to a height of $d = 1$ m. Although the experimental conditions were obviously different since the dataset used in [Weiss et al., 2012b] has not been made available, we believe our results still indicate the good potential of the proposed nonlinear observer in dealing with scale estimation from vision.

2.5.6. Estimation of the Gravity Vector

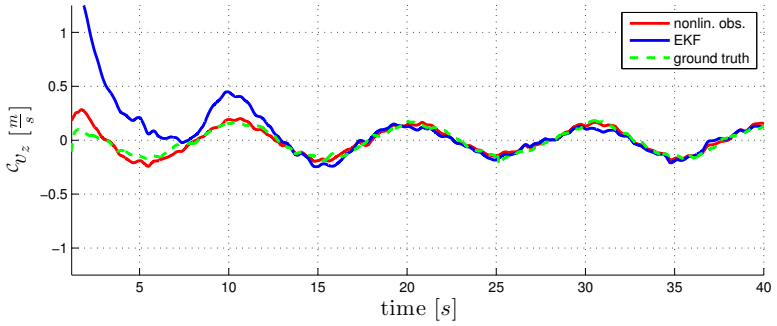
Both scale estimation schemes require the availability of the gravity vector $\boldsymbol{g}$, which, as explained in Sec. 2.4.2, we assume to be provided by the IMU itself. In order to verify the precision of this gravity estimation, we compared the IMU estimate of $\boldsymbol{g}$ against the ground truth obtained from



(a)

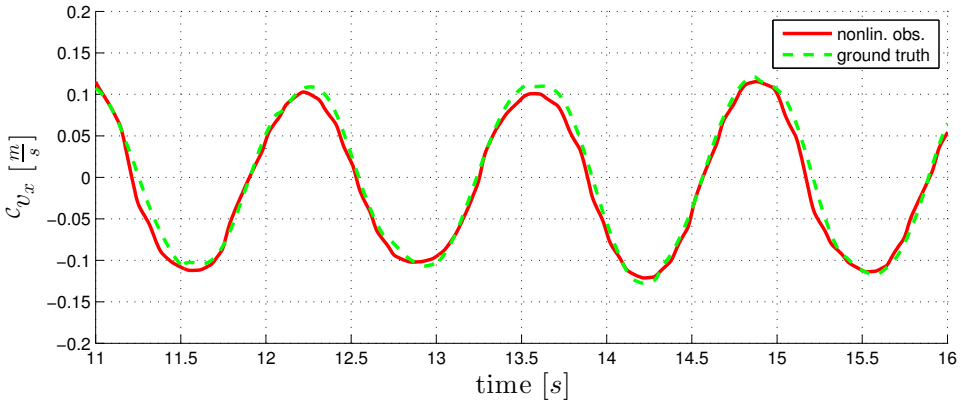


(b)

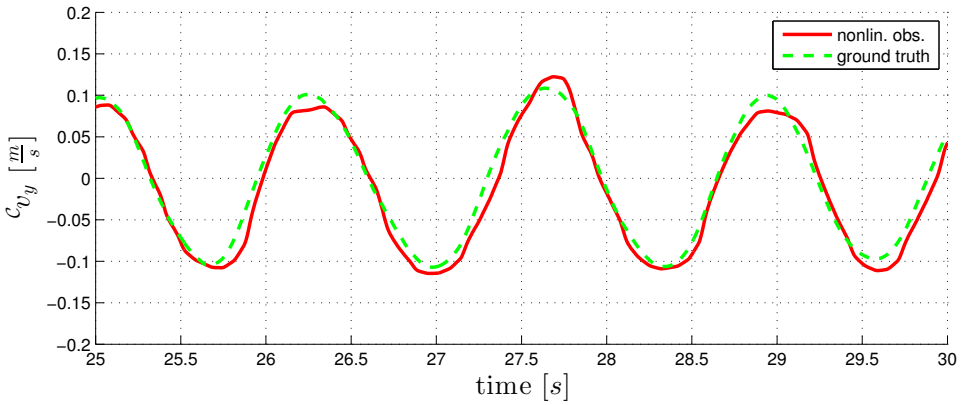


(c)

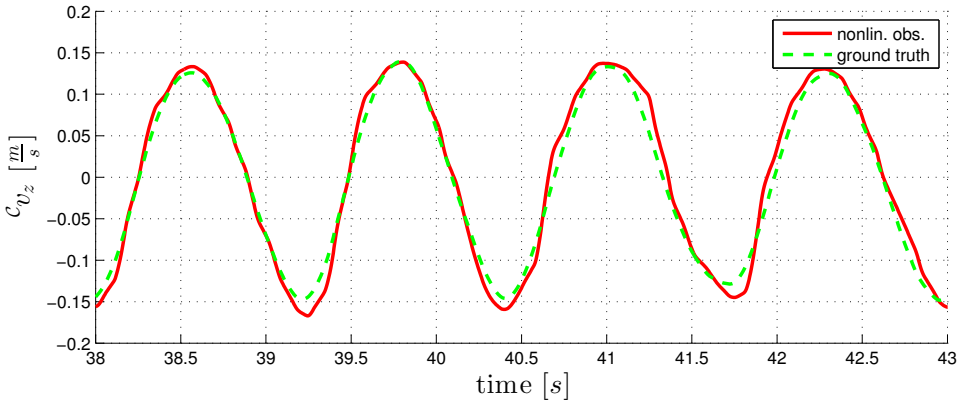
Figure 2.15.: Estimation results from experiments on a recorded flight along an inclined circular trajectory, thus characterized by a time-varying $d(t)$. (a) Estimation of the scene depth d, (b) Estimation error $|d(t) - \hat{d}(t)|$ compared to the ground truth and the ideal convergence behavior (2.43), and (c) the estimation of the vertical velocity $\boldsymbol{v}_z$. The performance of the nonlinear observer in recovering $d(t)$ and $\boldsymbol{v}_z(t)$ is not affected by the more challenging UAV motion.



(a)



(b)



(c)

Figure 2.16.: Velocity estimates for the nonlinear observer on the experimental conditions used in [Weiss et al., 2012b]. Results for the (a) x, (b) y, and (c) z direction are plotted against the ground truth.

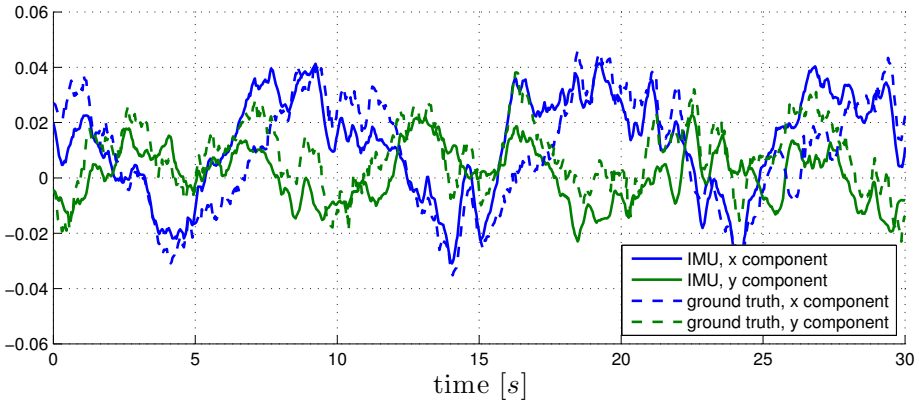


Figure 2.17.: x and y component of the normalized gravity vector estimated through sensor fusion within the IMU compared to the ground truth obtained from an optical tracking system. In average, the error between the IMU estimation and the ground truth is in the range of 1°.

the motion tracking system during the horizontal circular flights conducted for the experiments of Sec. 2.5.5.

The comparison of the x and y components of $\boldsymbol{g}$ are shown in Fig. 2.17. In average, the error of the gravity estimation from the IMU versus the ground truth is smaller than 1° despite the accelerated motion undergone by the UAV. Thus, the error is below the accuracy with which we were able to define the quadrotor object frame in the ground truth providing motion tracking system. Therefore, we conclude that the estimated $\boldsymbol{g}$ from the onboard IMU is suitable for our needs.

2.5.7. Predicting the Error Convergence Time for the Nonlinear Observer

As shown in Figs. 2.9b–2.12c and discussed in Sects. 2.5.3–2.5.5, keeping a constant acceleration norm $\|\boldsymbol{\Omega}(t)\|$ allows us to impose a behavior equivalent to the ideal reponse (2.43) on the estimation error $\zeta(t) = 1/d(t) - 1/\hat{d}(t)$ of the nonlinear observer. This possibility admits for a more general observation. Given a desired time for reaching some percentage of the initial error $\zeta(t_0)$, and choosing a gain K_α, one can exploit (2.43) to determine the needed *acceleration norm* to achieve this goal. Similarly, for a given trajectory with a known acceleration norm and a given gain K_α, one can determine the time needed to reach a desired percentage of the initial estimation error. The relation between acceleration norm and convergence time is plotted in Fig. 2.18 for the three cases of reaching 10 %, 1 %, and 0.1 % of the initial error under the (arbitrary) choice of $K_\alpha = 6$. Thus, for a bounded initial estimation error, i.e., with a maximum possible distance to the planar scene known before launch, one can uniquely predict how long the quadrotor has to fly (with constant acceleration norm) for having a guaranteed accuracy in the estimated distance. We note that this analysis can, of course, be done for any choice of gain K_α.

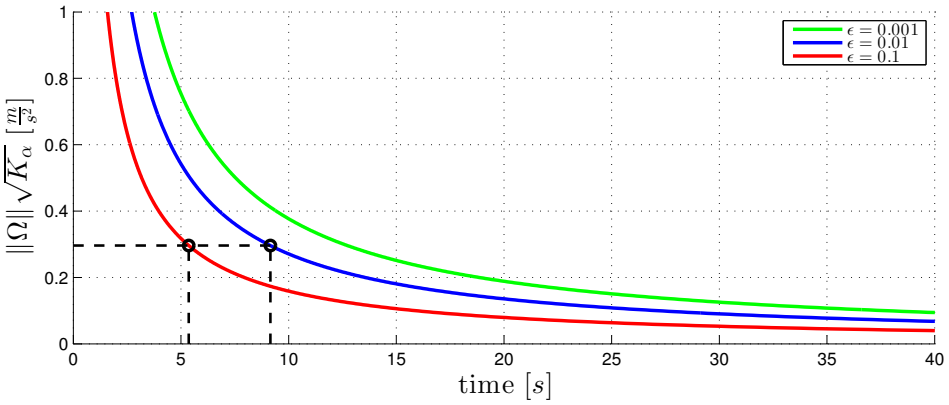


Figure 2.18.: Plot of the relation between the required 'excitement' $\sqrt{K_\alpha}\|\boldsymbol{\Omega}\|$ and the convergence time t needed to reach the same convergence criteria for the inverse scale estimation error $|1/d - 1/\hat{d}|$. The plot was generated for the choice of $K_\alpha = 6$. The three curves show the relation between the time needed to reach a percentage $\epsilon = [10\%, 1\%, 0.1\%]$ of the initial error versus $\sqrt{K_\alpha}\|\boldsymbol{\Omega}\|$. The plot can be read as follows: on a trajectory with an acceleration norm of $0.296\,\mathrm{m/s^2}$, the error $|1/d - 1/\hat{d}|$ will drop below 10% of its initial value within 5.36 s and below 1% within 9.15 s.

2.5.8. Noise Robustness of the Nonlinear Observer

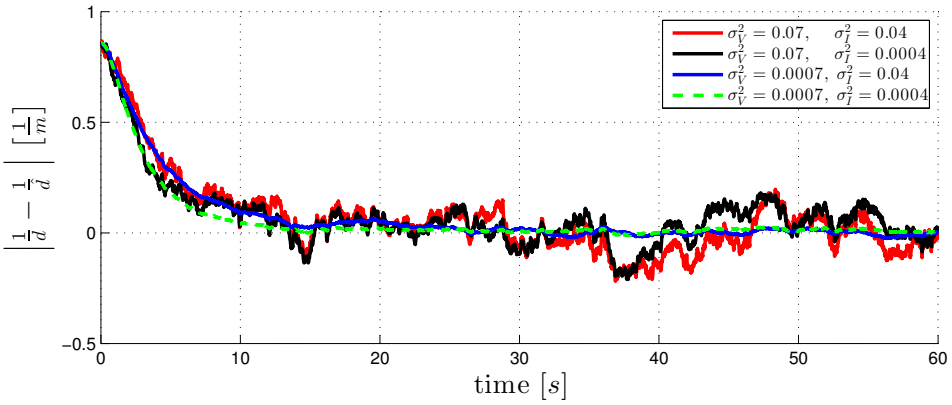
As opposed to the fully informed EKF, the nonlinear observer handles the presence of noise in sensor readings in an indirect way. In particular, the single gain K_α regulates the 'aggressiveness' of the filter, influencing both the convergence speed and its sensitivity to noise. Therefore, as explained, in all the previous experiments we tuned K_α to obtain the same noise level in the estimated $\hat{d}(t)$ with respect to the results found for the EKF.

To test the robustness of the nonlinear observer to increased sensor noise levels for the same choice of K_α, we increased the variance of the simulated noise by a factor of 100 for both the IMU acceleration readings and the scaled velocity $\boldsymbol{v}/d$ from the optical flow decomposition. Figure 2.19a shows the influence of different sensor noise levels on the estimation error $\zeta(t) = 1/d(t) - 1/\hat{d}(t)$. Despite the high noise level, the filter demonstrates a good level of robustness in generating a consistent state estimation.

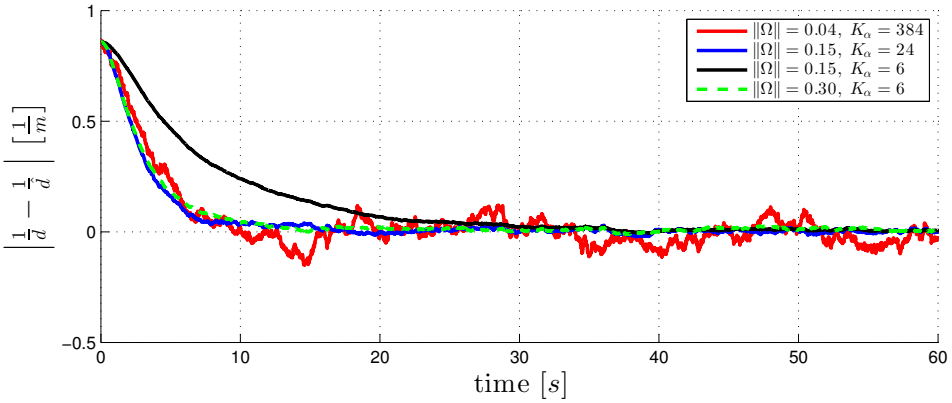
Furthermore, from the theoretical analysis of Sec. 2.4.3, the convergence rate of the nonlinear observer in (2.43) is determined by the quantity $\sigma_d = \sqrt{K_\alpha}\|\boldsymbol{\Omega}\|$. Thus, one can always trade smaller accelerations with a higher gain K_α for obtaining the same (ideal) convergence rate. In this sense, we compared a circular flight with an acceleration norm of $\|\boldsymbol{\Omega}\| \approx 0.296\,\mathrm{m/s^2}$ and a choice of $K_\alpha = 6$ to two flights with acceleration norms of $0.148\,\mathrm{m/s^2}$ and $0.037\,\mathrm{m/s^2}$, and the gain K_α chosen so as to yield the same σ_d (i.e., same error convergence). The results are reported in Fig. 2.19b. As expected, by keeping a constant σ_d, one obtains the same transient behavior for the estimation error $\zeta(t)$, although a higher noise level is induced by the larger employed K_α. Therefore, gain K_α can be used to tune the 'aggressiveness' of the filter.

2.5.9. Closed-Loop Control of a Real Quadrotor UAV

For a final validation of the overall framework, we made use of the estimated metric velocity to 'close the loop' and control the UAV motion in real time. For this experiment, we combined



(a)



(b)

Figure 2.19.: Influence of sensor noise on the nonlinear observer scheme. (a) Robustness of the nonlinear observer to different noise levels. The variance values of $\sigma_V^2 = 0.0007\,\mathrm{m}^2/\mathrm{s}^2$ and $\sigma_I^2 = 0.0004\,\mathrm{m}^2/\mathrm{s}^4$ for the visual and inertial sensor data, respectively, correspond to the actual noise found on the real sensors. The noise was then increased 100 times to test the robustness. (b) plot showing the trade-off between the 'control effort' (norm of the acceleration $\|\boldsymbol{\Omega}\|$) and the gain K_α. In case of lower accelerations, the convergence rate can be re-established using a higher gain K_α but at the expense of an increased noise level.

algorithm *V2* for the optical flow decomposition with the scale estimation obtained from the nonlinear observer. The quadrotor described in App. A.1 was commanded using a gamepad to give velocity commands sent via a wireless link while all perception and processing was carried out on board the UAV. Again, a motion tracking system was employed to record ground truth velocity estimates.

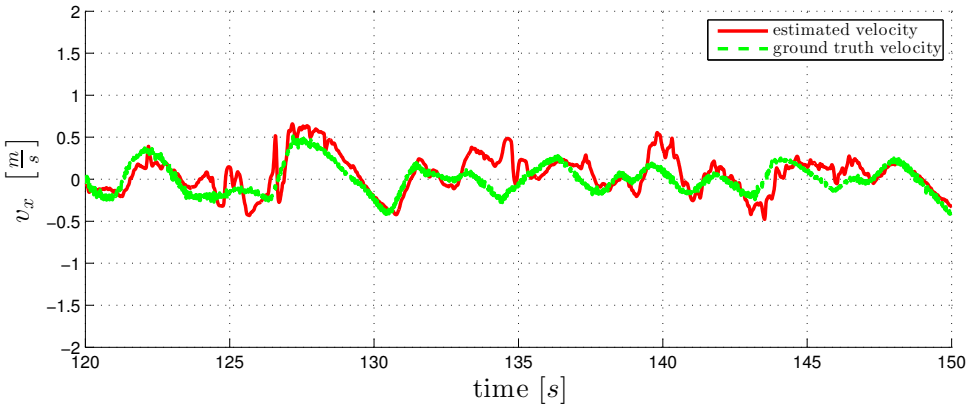
Figure 2.20 shows a section of a longer flight relying purely on onboard sensors. To test the robustness of the system, we moved an object through the field of view of the down-facing camera at time 125, causing some disturbances in the estimated velocity. Starting from time 134, the vehicle was additionally commanded to move in the vertical direction, causing the height, and therefore the distance to the plane, to change. The plots show some temporary over- and under-estimations of the linear velocity due to the abrupt commanded motions, but otherwise the proposed approach is always able to adapt to the changing height and provides a reliable ego-motion estimation.

A video from one of these experiments can be found online: `http://youtu.be/ohmLr3mCop8` In this video, external views of the vehicle and the image stream from the down-facing camera are shown while the vehicle navigates purely based on onboard sensors and processing capabilities. An example of the view from the down-facing camera is provided in Fig. 2.21.

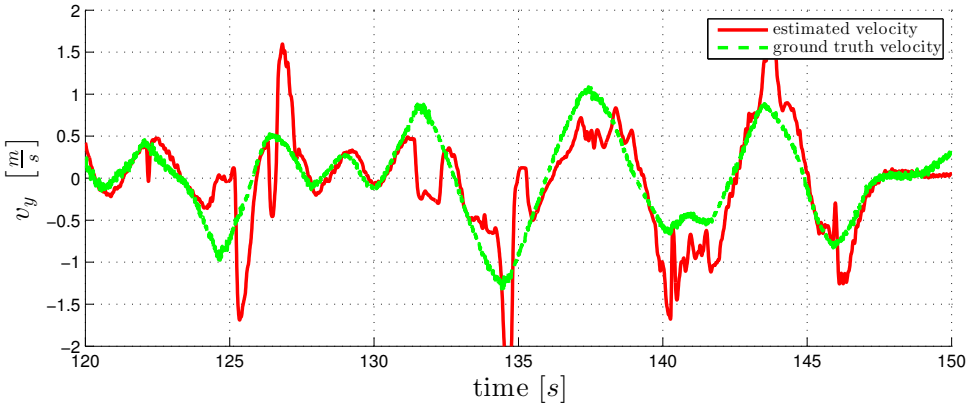
Concluding the experiment, we believe these experiments demonstrate that the presented velocity-based state estimation pipeline can be effectively used for the closed-loop control of real quadrotor UAVs using solely onboard hardware.

2.6. Conclusions and Future Work

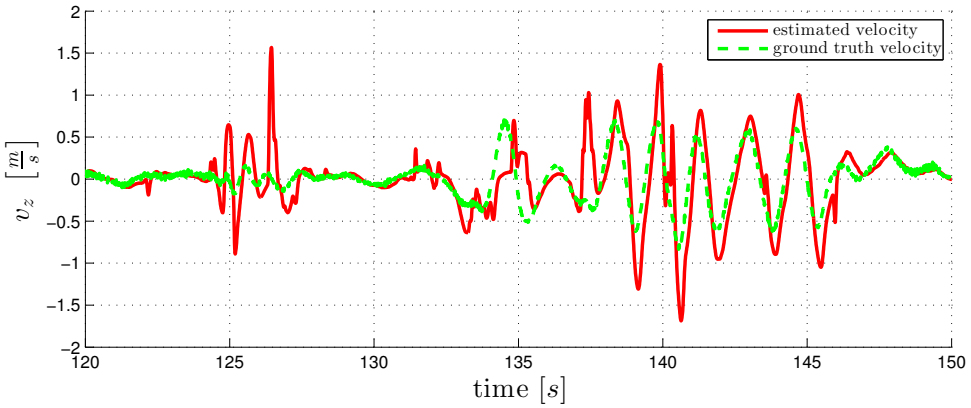
In this chapter, we addressed the need for a reliable onboard ego-motion estimation for quadrotor UAVs to overcome the boundaries of controlled lab environments. To this end, we first discussed three variants of an algorithm based on the continuous homography constraint to obtain an estimation of the UAV scaled linear velocity and angular velocity from the decomposition of the perceived optical flow. This step, indeed, allows us to retrieve ego-motion information independently of map and known landmarks. Furthermore, it eliminates the need for tracking features over an extended period of time. Subsequently, we extensively discussed the issue of estimating the (unknown) metric scale factor by fusing the scaled velocity estimates from the optical flow decomposition with the high frequency accelerometer readings of an onboard IMU. Scale estimation was achieved by proposing two estimation schemes: a first one based on a classical EKF and a second one on a novel nonlinear observation framework. Results from experiments on simulated and recorded sensor measurements were presented to assess and compare the performance of both filters under ideal and real conditions. When compared to the EKF, the nonlinear observer demonstrated a consistently better performance in terms of convergence rate of the scale estimation error. Furthermore, the proposed theoretical analysis showed the possibility to actively impose (and thus predict) the error transient response of the nonlinear observer by suitably acting on the estimation gain and the UAV motion (the norm of its acceleration). This analysis was, again, confirmed by the reported simulative/experimental results under several conditions, also involving different sensor noise levels to test the robustness of the approach. With the advantage of a fast and predictable convergence for recovering the metric UAV linear velocity, the nonlinear observer proved to be a suitable choice for the fully onboard implementation of a closed-loop velocity control of flying vehicles. In particular, we



(a)



(b)



(c)

Figure 2.20.: Estimated and ground truth velocity during a closed-loop flight of a quadrotor UAV for the (a) x, (b) y, and (c) z direction. An object was moved through the field of view at second 125 disturbing the estimated velocity.

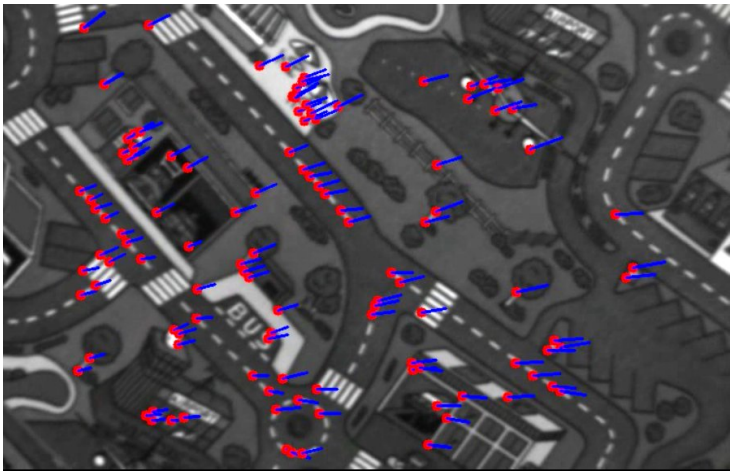


Figure 2.21.: View from the onboard camera as used for the optical flow extraction. Optical flow vectors tracked between the last two images are highlighted.

successfully demonstrated the reliability of the proposed framework in closed-loop experiments on a quadrotor UAV.

2.6.1. Future Work

Despite the convincing results, we are considering to extend our work in several ways. A first possibility is to further exploit the awareness of the error convergence behavior when using the nonlinear filter. Inspired by what has been presented in [Achtelik et al., 2013] in the context of *offline* path planning, one could devise an *online* strategy meant to adjust the input UAV trajectory (e.g., the velocity commands) in order to maintain a desired level of *excitement*. In our case, this excitement would be represented by the quantity $\sigma_d = \sqrt{K_\alpha}\|\mathbf{\Omega}\|$. This could result in, e.g., execution of small circular trajectories when the vehicle is commanded to hover in place with, instead, a more precise tracking of the desired/commanded velocities when undergoing more aggressive maneuvers.

On a similar note, we are currently investigating the possibility to dynamically adapt gain K_α of the nonlinear observer. Indeed, in the presence of noisy sensor readings, it might be desirable to start with a high value of K_α for imposing a quick initial error convergence, and to then reduce it once a sufficient convergence level is reached to obtain a smoother state estimation (see also Fig. 2.19b).

Furthermore, the quality of the initial scaled velocity estimate could be improved by dynamically selecting the temporal baseline for the computation of the optical flow vectors depending on the eminent velocity of the optical flow on the camera image (compare Sec. 2.5.1). Therefore, during high velocity motions and flight at low heights above the scene, features would be tracked over a shorter period of time. On the other hand, during slow motions, the temporal baseline could be extended to increase the quality of the state estimate.

2.7. Acknowledgments

The authors like to thank Dr. Antonio Franchi and Martin Riedel for their valuable suggestions and contribution on the development of the underlying software framework TeleKyb (see App. A.6 for more details).

3. Towards Visual Perception and Control for Multi UAV Collaboration

Summary

While we extensively addressed the problem of velocity controlled flight for a single UAV by means of onboard ego-motion estimation in Chapter 2, in this Chapter, we shift the focus towards the problem of state estimation for multiple UAVs. Key component to allow a group of robots to collaboratively achieve a common goal is the awareness of each others location within the shared environment. This mutual localization problem could be either circumvented by using direct visual observations of the other robots or actively addressed by exchanging location coordinates in a common reference frame. Thus, the latter approach requires one robot to localize itself with respect to the map of another. In this chapter, we present one approach for each of these two general techniques.

First, we present a decentralized formation control based on velocity estimation and direct vision-based perception of bearing angles to the other members of the formation. To overcome the limited field of view (FOV) of real cameras and to detect other agents surrounding the robot, we suggest to superimpose an artificial yaw-rotation. Therefore, a filter predicts the bearing angle towards a particular agent and is periodically updated whenever the other robot is visually detected. Our approach is finally evaluated using closed loop experiments with three real quadrotor UAVs and a human operator.

Afterwards, we introduce a global dense localization approach of a monocular camera with respect to a dense 3D map. To this end, we suggest to two stage algorithm, involving a coarse global localization stage using sparse state-of-the-art feature matching techniques and a novel iterative analysis-by-synthesis pose-refinement process. The second stage minimizes the photometric error between the real monocular camera image and a synthesized view of the dense map from the predicted view-point.

The work described in this chapter has been partly published in:

- Grabe, V., Masone, C., Ryll, M., Franchi, A., Bulthoff, H. H., Rubuffo Giordano, P. (2011). Implementation of a Decentralized Formation Control based on Visual Bearing Angles for Multiple UAVs. Technical Report.
- Franchi, A., Masone, C., Grabe, V., Ryll, M., Bulthoff, H. H., Rubuffo Giordano, P. (2012). Modeling and Control of UAV Bearing Formations with Bilateral High-level Steering. The International Journal of Robotics Research, 31(12), 1504–1525.
- Grabe, V., Scaramuzza, D. (2014). Global Localization of a Monocular Camera in RGB 3D Maps through Photometric-Error Minimization. Technical Report.

3.1. Overview

In this section, we introduce two approaches towards the mutual localization problem based on direct and indirect observations. First, in Sec. 3.2, we present a formation control based entirely on direct visual bearing angle observations between each two robots within the formation. Afterwards, in Sec. 3.3, for cases without the availability of a direct line of sight between the robots, we propose an indirect localization approached based on the precise localization of a camera within a dense 3D map obtained by a second robot.

3.1.1. Bearing Angle Formations based on Direct Visual Detection

In Section 3.2, we present an innovative distributed system for the control of groups of UAVs using visually established bearing angle measurements from onboard camera, i.e., without relying on distance measurements or global localization. While [Franchi et al., 2011] presents theoretical derivations behind bearing based formation control in the general case and suggests a haptic teleoperation system as external user interface, we focus on the implementation and experimental evaluation of the complete system using a group of quadrotors. The required bearing angles are retrieved purely visually by means of onboard cameras and processing. We further extend the control framework to cope with the limited field of view of real cameras by periodical horizontal scanning movements and estimations of the bearing angles whenever the other UAVs are out of sight. We demonstrate the validity of our approach with closed-loop experiments using a set of three real quadrotor UAVs.

3.1.2. Camera Localization within a Dense 3D Map

In Section 3.3, we tackle the problem of globally localizing a camera-equipped mobile agent in a dense 3D map created by another agent equipped with, e.g., an RGBD sensor or a monocular dense reconstruction system. After an initial coarse global pose estimate obtained from putative point correspondences using state-of-the-art algorithms, the camera location and orientation is refined using a analysis-by-synthesis approach that minimizes the photometric error between a synthesized view of the 3D map and the actual monocular camera image. An extensive experimental verification of the approach run on challenging indoor datasets is used to evaluate our approach.

The approach of mutual localizations from heterogeneous sensors is later extended with further ideas and concepts in the future work (see Sec. 4.2).

3.2. Implementation of a Decentralized Formation Control based on Visual Bearing Angles for Multiple UAVs

The control of a group of mobile robots became more and more an active focus of research in the recent years. In fact, swarms are able to react more flexibly to situational changes than one large robot [Howard et al., 2006]. Furthermore, their versatility accommodates a swarm to a variety of different missions, e.g., coordinated transport of loads using a formation [Fink et al., 2010], distributed surveillance tasks [Alexis et al., 2009], joint object throwing and catching [Ritz et al., 2012], and building of 3D structures [Willmann et al., 2012].

To an even greater extent than most work on individual UAVs, recent literature on the control of UAV formations relies on external tracking systems to retrieve either the absolute location of all UAVs or their mutual positions, that is, the formation. This, however, is not suitable for many usage scenarios where robots enter previously unmapped areas or are out of the range of GPS signals. Thus, we tried to establish a formation control with the help of cheap and light weight onboard sensors. Although a camera fits these needs best, bearing-plus-distance based visual formation control either requires the integration of IMU measurements or known sized objects to obtain distance perception, e.g. [Das et al., 2002; Olfati-Saber et al., 2007; Moshtagh et al., 2009]. However, when dealing with large aerial distances, low camera resolutions, or wide optical field of views, distance estimation based on known object dimensions becomes infeasible.

In [Franchi et al., 2011], an approach which uses only relative bearing angles to stabilize a formation of multiple UAVs has been derived from a theoretical perspective. In contrast to previous work, this control system allows a user to actively translate and expand the full formation, while most cited approaches from the literature focus on basic leader following strategies.

In this section, however, we discuss the challenges arising from the implementation of the proposed controller on an actual robotic system. To provide true decentralization, we designed the system to solely rely on onbaord sensors for the perception of the remaining agents within the formation. To this end, we rely on cameras and the visual detection of individually colored markers on each robot to obtain the required bearing angle measurements. However, the field of view of cameras is naturally limited and the fitting of multiple cameras for omni directional vision is not always suitable. To allow for an nevertheless unconstrained motion, we propose the robot to periodically scan its environment by means of rotation around its center axis and predict the angle to those robots which are currently outside of the field of view.

The remainder of this section is structured as follows: the theory behind [Franchi et al., 2011] is briefly reviewed in Sec. 3.2.1. Afterwards, we address the problem of a limited camera field of view in Sec. 3.2.2 and extend the proposed controller accordingly. Our experimental setup is described in Sec. 3.2.3 while the experimental results are presented and discussed in Sec. 3.2.4. Finally, in Sec. 3.2.5, we conclude our work and present an outlook to future projects in Sec. 3.2.6.

3.2.1. Vision-Based Bearing-Angle Formations

We recall the most important theoretical derivations presented in [Franchi et al., 2011] here. The reader is referred to [Franchi et al., 2011] for further information and detailed proofs.

Assuming a common knowledge of the vertical axis z through sensor fusion on the onboard IMU (see Sec. 2.5.6 for a demonstration of feasibility), we address the control of position and

rotation $(\boldsymbol{p}_i,\ \theta_i) \in \mathbb{R}^3 \times S^1$ by means of velocity and rotation $(\boldsymbol{u}_i,\ \omega_i)$ in body frame of each kinematic agent. The unit vector *relative bearing* ${}^i\hat{\boldsymbol{\beta}}_{ij} \in S^2$ is defined as ${}^i\hat{\boldsymbol{\beta}}_{ij} = {}^iR(\boldsymbol{p}_j - \boldsymbol{p}_i)/\|\boldsymbol{p}_j - \boldsymbol{p}_i\|$ where ${}^iR$ denotes the inverse rotation matrix of θ_i around the z axis. Relative bearings are obtained from the cameras in the form of the two components azimuth ${}^i\alpha_{ij} \in (-\pi\ \pi]$ and elevation ${}^i\eta_{ij} \in [-\pi/2\ \pi/2]$ (explained in detail in Sec. 3.2.3). The transformation between the two representation is given by ${}^i\hat{\boldsymbol{\beta}}_{ij} = \left(\cos{}^i\eta_{ij}\cos{}^i\alpha_{ij}\ \ \cos{}^i\eta_{ij}\sin{}^i\alpha_{ij}\ \ \sin{}^i\eta_{ij}\right)^T$.

Let us define the set $\mathcal{N} = \{(i,j) \in \{1,\dots,N\}^2 | i \neq j\}$. A *bearing-formation* is defined through the desired relative bearings $\{{}^i\hat{\boldsymbol{b}}_{ij} \equiv ({}^i\hat{\boldsymbol{a}}_{ij}, {}^i\hat{\boldsymbol{e}}_{ij})\}_{(i,j)\in\mathcal{N}}$ between all the N agents. In [Franchi et al., 2011] is was proved that a set of only $3N-4$ bearings is sufficient to specify such a formation. In addition we showed that these formations can be further translated, dilated and rotated around an axis parallel to the z axis by a user while the relative bearings are kept constant (synchronized motions). However, in order to perform any synchronized rotation, the measurements of the agent inter-distances are necessary. Unfortunately, for typical areal distances, it is not feasible to accurately estimate the distance to other agents in an unknown environment by means of a visual system only. Therefore, we do not allow the control of synchronized rotations through the user.

The control input is split in two components, namely $(\boldsymbol{\mu}_i,\ \omega_i) = (\boldsymbol{\mu}_i^f,\ \omega_i^f) + (\boldsymbol{\mu}_i^m,\ 0)$. The term $(\boldsymbol{\mu}_i^f,\ \omega_i^f)$ denotes the signal for the automatic control of the bearing formation (covered in Subsection 3.2.2). The term $(\boldsymbol{\mu}_i^m,\ 0)$ allows the user to translate and dilate the formation by setting $\boldsymbol{\mu}_i^m = {}^iR\boldsymbol{\nu}^t - r\gamma_{12i}\hat{\boldsymbol{\beta}}_{i1}$, with $\gamma_{ijk} = \frac{\|{}^j\hat{\boldsymbol{\beta}}_{ji} \times {}^j\hat{\boldsymbol{\beta}}_{jk}\|}{\|{}^k\hat{\boldsymbol{\beta}}_{ki} \times {}^k\hat{\boldsymbol{\beta}}_{kj}\|}$, where the vector $\boldsymbol{\nu}^t \in \mathbb{R}^3$ is used to control the translation and $r \in \mathbb{R}$ sets the expansion rate of the formation.

For interaction with a human operator, we propose a first force-feedback device with three degrees of freedom (DoFs) and a second with 1-DoF to control the translational speed $\boldsymbol{\nu}^t$ as well as the expansion rate r of the formation respectively. In particular, we set $\boldsymbol{\nu}^t = \lambda_t \boldsymbol{x}_t$ and $r = \lambda_r x_r$ with $\boldsymbol{x}_t \in \mathbb{R}^3$ and $x_r \in \mathbb{R}$ being the position of the two control devices and $\lambda_t > 0$, $\lambda_r > 0$ factors to map the positions to desired velocities.

The actual quadrotor follows the kinematic agent velocity as described by the tracking controller presented in the next Sec. 3.2.2). However, when using real vision based velocity perception system such as the one presented in Chapter 2, some tracking errors inevitably occur. To improve the controllability of the system by a human operator, in our experimental evaluation, the user receives two haptic force feedbacks which are proportional to this velocity and expansion rate tracking errors.

In the remainder of this Sec. 3.2, we use Greek letters for measured quantities and the homologous Latin characters for their desired values since the excessive use of subscripts to indicate measurements between different agents renders the use of the otherwise preferred subscribed m impractical.

3.2.2. Bearing Formations with Limited Field of View

Camera hardware suitable for the use on small flying robots usually suffers from a limited FOV. Furthermore, restricted onboard processing power often does not allow for a parallel use of multiple cameras. Thus, in contrast to the assumptions made in [Franchi et al., 2011], we have to deal with a limited FOV for the implementation on real robot. To compensate for a horizontally limited FOV, two strategies apply: (i) horizontal shifting of the agent or (i) rotation

of the agent around the body z axis. We opt for this latter yaw rotation strategy since it is fast and preserves mutual positions of all robots within the formation.

In particular, we focus on a limited *horizontal* FOV only, mainly due to two reasons. First, we primarily test formations with a horizontal dominant dimension, which, for collaboration tasks such as joined transportations of heavy loads, are the most relevant ones. Lastly, since quadrotors are under-actuated, it is unfeasible to change their roll/pitch orientation independently from the horizontal speed. However, the proposed approach can be easily extended to a limited vertical FOV provided that the camera is mounted on an additional tilting unit. Therefore, we assume the agent i to be measuring the azimuth angle ${}^i\alpha_{ij}$, if and only if ${}^i\alpha_{ij} \in [a_{\min}, a_{\max}]$.

We first introduce a relaxed definition of a bearing formation and then propose a controller based on this formulation, which overcomes the limitation of the horizontal FOV.

We introduce the following relaxed bearing formation definition: two bearing formations $\{({}^i\alpha_{ij}, {}^i\eta_{ij})\}_{(i,j)\in\mathcal{N}}$ and $\{({}^i\alpha'_{ij}, {}^i\eta'_{ij})\}_{(i,j)\in\mathcal{N}}$ are equivalent if the shift ${}^i\alpha'_{ij} - {}^i\alpha_{ij}$ is the same for every $(i,j) \in \mathcal{N}$, i.e., if the difference between any two azimuths is constant

$$ {}^i\alpha_{ij} - {}^i\alpha_{ik} = {}^i\alpha'_{ij} - {}^i\alpha'_{ik} \quad \forall (i,j),(i,k) \in \mathcal{N}. \tag{3.1} $$

The rotation speed ω_i of agent i does not affect the difference in (3.1). In fact, from the dynamic equation of the azimuth we have:

$$ \begin{aligned} {}^i\dot{\alpha}_{ij} - {}^i\dot{\alpha}_{ik} = \; & \frac{1}{\delta_{ij}}({}^j\hat{\boldsymbol{\alpha}}_{ji}^T\boldsymbol{\mu}_j - {}^i\hat{\boldsymbol{\alpha}}_{ij}^T\boldsymbol{\mu}_i) - \\ & \frac{1}{\delta_{ik}}({}^k\hat{\boldsymbol{\alpha}}_{ki}^T\boldsymbol{\mu}_k - {}^i\hat{\boldsymbol{\alpha}}_{ik}^T\boldsymbol{\mu}_i). \end{aligned} \tag{3.2} $$

Therefore, the rotation dynamic of an agent can be freely chosen without affecting the relaxed bearing formation.

Formation Control with a Limited Field of View

To deal with the limited FOV in the horizontal plane, we ask the agents to follow an opportunely shaped bearing trajectory, i.e., any agent $h \in (1,\dots,N)$ is forced to rotate with a given yaw rate $\omega_{\text{rot},h}(t)$ in order to execute a periodic motion (e.g., a sinusoidal or a constant-slope trajectory). By suitably choosing $\omega_{\text{rot},h}(t)$ and while the bearing formation is maintained, agent h would be able to periodically measure the relative azimuth ${}^h\alpha_{hj}$ of another agent j for a fraction of the trajectory period. For the remaining fraction, a direct measure of the azimuth is not given. Hence it must be estimated on the basis of motion proprioception.

We define the modified version of the bearing formation control problem, which accounts for the horizontally limited FOV.

Problem 1 (Relaxed bearing-formation control)**.** *Given a set of feasible desired bearings* $\{{}^i\hat{\boldsymbol{b}}_{ij} = ({}^ia_{ij}, {}^ie_{ij})\}_{(i,j)\in\mathcal{N}}$*, find a control law* $(\boldsymbol{\mu}_i^f, \omega_i^f)$ *depending on* $\{{}^i\hat{\boldsymbol{\beta}}_{ij}\}_{(i,j)\in\mathcal{N}}$ *which steers* ${}^i\hat{\boldsymbol{\beta}}_{ij}$ *to the trajectory* ${}^i\tilde{\boldsymbol{b}}_{ij}(t) = ({}^ia_{ij} + \theta_{\text{rot},i}(t), {}^ie_{ij})$ *with* $\theta_{\text{rot},i}(t) = \int_t \omega_{\text{rot},i}(t)\mathrm{d}t$ *and the distances* $\{\delta_{ij}\}_{(i,j)\in\mathcal{N}}$ *to a constant non-zero value,* $\forall (i,j) \in \mathcal{N}$*.*

First we present the control law used to solve Problem 1 assuming that the azimuth is always measured. Afterwards, we describe the estimate that we used in the real case. Let us consider

the following control law:

$$\boldsymbol{\mu}_1^f = \mathbf{0} \tag{3.3}$$

$$\omega_1^f = -\,\omega_{\mathrm{rot},1} \tag{3.4}$$

$$\begin{aligned}\boldsymbol{\mu}_2^f = &-\frac{K_p}{\cos{}^1\eta_{12}}\Big[\sin({}^1\alpha_{12} - \theta_{\mathrm{rot},1} - {}^1a_{12})\boldsymbol{M}\,{}^1\hat{\boldsymbol{\beta}}_{12} + \\ &\left(\sec{}^1\eta_{12}\,{}^1\hat{\boldsymbol{\beta}}_{12} - \sec{}^1e_{12}\,{}^1\hat{\boldsymbol{b}}_{12}\right)\cdot\hat{\boldsymbol{z}})\hat{\boldsymbol{z}}\Big]\end{aligned} \tag{3.5}$$

$$\omega_2^f = -\,\omega_{\mathrm{rot},2} + K_\omega \sin({}^2\alpha_{21} - \theta_{\mathrm{rot},2} - {}^2a_{21}) \tag{3.6}$$

$$\boldsymbol{\mu}_i^f = -\,K_p\,{}^i\boldsymbol{R}_1\,\left(\bar{\delta}_{1i}\,{}^1\hat{\boldsymbol{\beta}}_{1i} - \bar{d}_{1i}\boldsymbol{R}(\theta_{\mathrm{rot},1})\,{}^1\hat{\boldsymbol{b}}_{1i}\right) \tag{3.7}$$

$$\omega_i^f = \begin{cases} -\omega_{\mathrm{rot},i} + K_\omega \sin({}^i\alpha_{i1} - \theta_{\mathrm{rot},i} - {}^ia_{i1}) & \text{if } \cos{}^ie_{i1} \neq 0 \\ -\omega_{\mathrm{rot},i} + K_\omega \sin({}^i\alpha_{i2} - \theta_{\mathrm{rot},i} - {}^ia_{i2}) & \text{otherwise} \end{cases} \tag{3.8}$$

where $i \in [3,\ldots,N]$, $\hat{\boldsymbol{z}} = (0\,0\,1)^T$, $\bar{\delta}_{1i} = \gamma_{12i}\sec({}^1\eta_{12})$, $\bar{d}_{1i} = \frac{\left\|{}^2\hat{\boldsymbol{b}}_{21}\times{}^2\hat{\boldsymbol{b}}_{2i}\right\|}{\left\|{}^i\hat{\boldsymbol{b}}_{i1}\times{}^i\hat{\boldsymbol{b}}_{i2}\right\|}\sec({}^1e_{12})$, ${}^i\hat{\boldsymbol{b}}_{ij} \equiv ({}^ia_{ij}, {}^ie_{ij})$, ${}^i\boldsymbol{R}_1$ can be computed as $\boldsymbol{R}^T({}^i\alpha_{i1})\boldsymbol{R}(\pi)\boldsymbol{R}({}^1\alpha_{1i})$, denoting by $\boldsymbol{R}(*)$ the rotation matrix of a given angle around $\hat{\boldsymbol{z}}$ and $\boldsymbol{M} = \begin{bmatrix}\boldsymbol{M}' & \mathbf{0}_{2\times1}\\ \mathbf{0}_{1\times2} & 0\end{bmatrix}$ with $\boldsymbol{M}' = \begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}$.

Given a starting configuration described by the bearings $\{{}^i\hat{\boldsymbol{\beta}}_{ij}^0 \equiv ({}^i\alpha_{ij}^0, {}^i\eta_{ij}^0)\}_{i,j=1,\ldots,N}$ such that $\left\|{}^1\hat{\boldsymbol{\beta}}_{1i}^0 \times {}^2\hat{\boldsymbol{\beta}}_{2i}^0\right\| \neq 0$ and $\cos{}^1\eta_{12}^0 \neq 0$, and a set of feasible desired bearing trajectories $\{{}^i\tilde{\boldsymbol{b}}_{ij}(t) \equiv ({}^ia_{ij} + \theta_{\mathrm{rot},i}(t), {}^ie_{ij})\}_{i,j=1,\ldots,N}$ such that $\left\|{}^1\tilde{\boldsymbol{b}}_{1i}(t) \times {}^2\tilde{\boldsymbol{b}}_{2i}(t)\right\| \neq 0$, $\cos{}^1e_{12} \neq 0$ for all $i = 3,\ldots,N$, the control laws in (3.3-3.8) asymptotically, and almost globally, steer ${}^i\hat{\boldsymbol{\beta}}_{ij} \to {}^i\tilde{\boldsymbol{b}}_{ij}(t)$ and $\delta_{ij} \to \bar{d}_{1i}\delta_{12}^0\cos{}^1\eta_{12}^0$, for any $(i,j) \in \mathcal{N}$.

As an estimate ξ of the azimuth ${}^h\alpha_{hj}$, we use the following dynamics:

$${}^h\dot{\xi}_{hj} = K_\xi({}^h\alpha_{hj} - {}^h\xi_{hj}) - \omega_h \tag{3.9}$$

where the constant $K_\xi \in \mathbb{R}_{\geq 0}$ is positive when the measure is available (i.e., ${}^i\alpha_{ij} \in [a_{\min}, a_{\max}]$) and zero otherwise. If the bearing formation is maintained, the estimates (3.9) will converge to the actual value. On the other hand, for a sufficient large value of K_ξ, the estimate will not diverge even if the bearing formation changes.

3.2.3. Experimental Setup

Our experimental setup shown in Fig. 3.1 consists of three quadrotors (see App. A.1), each equipped with a monocular camera and a unique visible beacon, and two force feedback devices to generate motion commands. The visual bearing angle detection system and the velocity tracking controller run on a small PC onboard the quadrotor. To simplify the computational setup and to overcome software license limitations, the three high-level formation controllers were delegated to an additional GNU-Linux machine running Matlab. Both sides communicate by means of wireless ethernet. The haptic feedback devices are connected to the additional Linux PC. For a pure evaluation of the formation control system proposed in this section, we rely on an optical motion tracking system to recover the metric velocity of each robot, however, the visual ego-motion estimation system presented in Chapter 2 could be used likewise.

Figure 3.1.: Experimental setup for the experiments on a bearing angle formation: (a) triangular formation during the experiments. The translation of agent 1 (green) on the right is controlled only by the operator. Agent 2 and 3 (yellow and cyan) move on the associated lines towards agent 1 according to the commanded expansion rate. (b) Sketch of the formation as seen from above. (c) Subjective view of agent 1 with agent 2 and 3 highlighted by the visual detection system. (d) Omega.6 (left) and Omega.3 (right) haptic-feedback devices used to control the expansion and translation respectively.

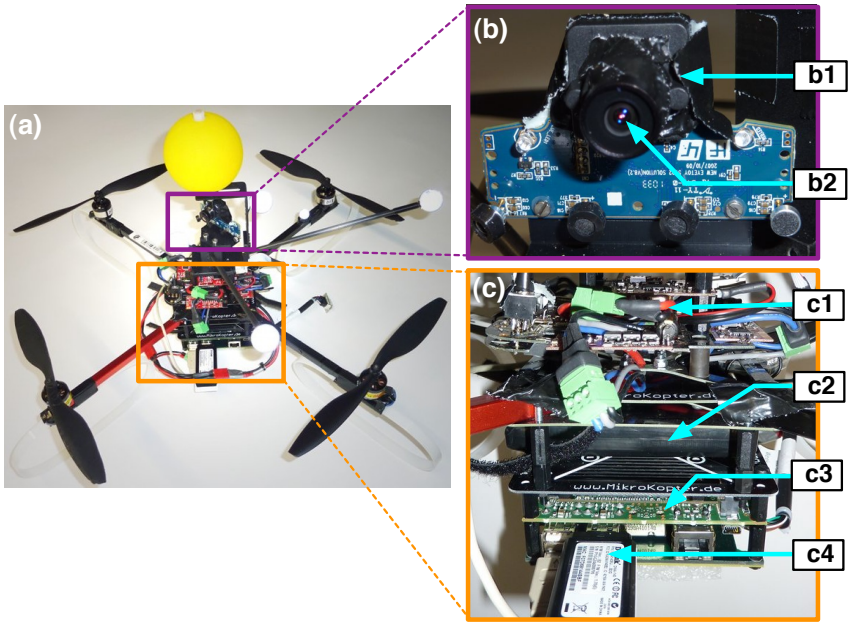


Figure 3.2.: Quadrotor setup for the experiments on the bearing angle formation. (a) Quadrotor in its flight configuration. (b) Camera setup, b1) Playstation Eye 3 camera with M12 lens holder attached, b2) 140° lens. (c) Computational setup c1) Microcontroller and IMU, c2) 2200 mAh Lithium Polymer Battery, c3) Small PC with Intel Atom CPU and extension board holding the connectors, c4) WLAN adapter

The Haptic Input Device

An Omega.3 haptic feedback device is used together with an Omega.6 device[1] to capture the translational and dilational control commands in $\mathbb{R}^3$ and $\mathbb{R}^1$ respectively. The Omega.3 device features three DoF while the Omega.6 has six DoF of which the three translational DoF are actuated. For the Omega.6, we constrained the motion to one DoF as indicated in Figure 3.1d. The forces described in Sec. 3.2.1 are presented to the human operator with a frequency of 2.5 kHz.

The Quadrotors

The three quadrotors used for the experiments are described in more detail in App. A.1. The entire optical detection system is implemented on a small Intel Atom 1,66 GHz processing board (see App. A.4) that can be seen on Fig. 3.2c3.

To allow for an independent evaluation of the presented approach including the visual bearing angle estimation, we relied on an external motion tracking system to obtain the velocities of the quadrotors. However, the relative bearings are estimated by means of a vision-based system implemented on the onboard Intel Atom hardware.

[1] `http://www.forcedimension.com`

Vison-based Bearing Measurements

We decided to use a modified low cost Sony Playstation Eye 3 color camera for our experiments which was originally designed for the consumer market. The camera and all modifications are described in more detail in App. A.2 and are shown in Fig. 3.2b. The calibration was done individually for all cameras prior to the experiment.

Figure 3.2a shows the configuration of one of our quadrotors. The camera is mounted 0.09 m above the central body frame. Small shock absorbers help to reduce vibrations carried from the motors to the camera. Reflective markers serve as tracking markers for the external ground truth providing motion tracking system. To allow for a simple and reliable visual tracking among the UAVs, we mounted a ball in an individual color 0.08 m above the camera of each quadrotor. This offset between camera and tracked ball results in an error that cannot be corrected without the availability of distance measurements. Thus, to allow for an evaluation of the visual system itself, we incorporate these offsets into the data obtained from the ground truth for the results presented in Sec. 3.2.4.

To reduce the development time, we mainly relied on implementations provided by the free OpenCV library. Our tracking algorithm first segments the input image for all of the predefined ball colors individually by exploitation of the HSV (Hue, Saturation and Value) color space. Then, noise in the resulting binary image is eroded and the remaining segmented shapes are expanded to allow a more reliable blob detection even for distant objects. In the following, ellipses are fitted for all segmented areas while non-circular ellipses up to a threshold are rejected. If there was more than one blob detected, the one closest to the last known position is chosen. Additionally, an outlier rejection is applied. Since neither the distance to the traced quadrotor nor the size of the blob is assumed to be known, the rejection is done based on imposing a maximum allowed angle change. Finally, lookup tables generated during the calibration of the cameras are used to obtain the angles corresponding to the center of the remaining blob.

A visualization of the output of the visual blob detection system is shown in Fig. 3.1c. Note that it is indeed possible to obtain an estimate of the distance to other UAVs exploiting the known diameter of the tracked colored ball. However, when using wide angle lenses or low camera resolutions, this is only reliable for short distances. For our camera setup, this was therefore not feasible for distances larger than 1 m and in particular not for the larger aerial distances targeted by our work.

The mounting of the camera introduces a small offset between the direction of the camera and the x axis of the UAV. For a compensation, we calibrated the extrinsic parameters of all camera-UAV setups prior to the experiments, that is we measured the average offset for both azimuth and elevation over the entire FOV.

Experimental Design

For the validation of our bearing angle based formation control approach using real robots, we first chose a triangular configuration of three robots with each of them being able to see all other robots at the same time (Figure 3.1a). Thus, in this first experiment, we were able to use the original controller presented in [Franchi et al., 2011]. All quadrotors were roughly oriented towards the center of gravity of the formation, inducing relative bearing angles around $-30°$ and $30°$. In the starting configuration (approximately 2.1 m distance between the quadrotors), the height was set to 0.7 m, 0.55 m, and 0.85 m for agents 1, 2, and 3 respectively.

In a second set of experiments, we altered the formation such that one angle of the triangle exceeded 90° and therefore the corresponding agent 3 was only able to retrieve the relative bearing to one of the two other agents at a time. Thus, agent 3 was forced to use the controller for a limited FOV as presented in Sec. 3.2.2 while agent 1 and 2 did not rotate. We used a sinusoidal profile with an amplitude of 26° and a frequency of 0.8 Hz for the scanning rotation $\theta_{\text{rot}}(t)$. For this second experiment, the initial height was set to 0.7 m for all quadrotors.

An optical motion tracking system was used to provide a reliable ground truth with sub-millimeter accuracy. W.l.o.g, the estimator was initialized with the help of data obtained from this motion tracking system prior to the experiment to allow for a save sequential take-off procedure of the UAVs.

3.2.4. Experimental Results and Discussion

In the following, we describe several experiments to validate our approach for visually controlled bearing formations. First, we analyze the accuracy of the visual system. We then evaluate the system with three real robots. The experiments are further documented in the videos `http://youtu.be/9RiGd7DTk34` and `http://youtu.be/YGjMXqGnvWI`.

First, to validate our visual bearing angle detection system, we compared the output of our algorithm with the ground truth obtained from our motion tracking system. For this experiment, we recorded the real bearing angles calculated from the positions and orientations given by the motion tracking system together with the relative bearing angles obtained from our visual system during an actively controlled flight with three UAVs. W.l.o.g., we picked the angles between agent 1 and 2 as well as between agent 1 and 3 for evaluation. The results are presented in Fig. 3.3.

For the presented data, we found a mean error of $-0.24°$ and $-0.44°$ with a standard deviation of 0.30° and 0.40° for the azimuth between agent 1 and 2 (Fig. 3.3a) and agent 1 and 3 (Fig. 3.3b) respectively. The mean elevation error was $-0.51°$ and $-0.53°$ with a standard deviation of 1.46° and 1.45° respectively. The higher standard deviation for the elevation is due to the less robust height control compared to the yaw control of our quadrotors. The controller was able to stabilize azimuth and elevation up to a mean error of $-1.70°$ and $-4.62°$ with a standard deviation of 2.72° and 2.36° for azimuth and elevation respectively compared to the desired values (see Figs. 3.3c and 3.3d). The high standard deviation is mainly a consequence of the constantly changing human control commands and the lag inherent to the system. In turn, the quadrotors demonstrated to follow the commanded velocities precisely. The mean velocity tracking error of agent 1 was 0.001 m/s, 0.002 m/s, and 0.006 m/s with a standard deviation of 0.110 m/s, 0.056 m/s, and 0.042 m/s for the x, y, and z axis respectively, see Figs. 3.3e and 3.3f.

We were able to prove real-time performance on the described limited onboard hardware without any particular optimizations. The algorithm runs with a frequency of 7 Hz. However, we measured a constant temporal lag of three frames compared to the data obtained from the motion tracking system. The lag of precisely three frames was observed independent of the capabilities of the underlying hardware system and thus the achieved frame rate. We assume that it was caused by the use of the default video-for-linux drivers that were manually patched to support the employed Playstation eye camera. Thus, on the mobile setup, this lag measured in average 500 ms in contrast to 120 ms on desktop hardware, where the algorithm was executed with a frequency of 30 Hz.

In the second set of experiments, agent 3 was set to perform a scanning movement to detect both agent 1 and 2 periodically. The relative azimuth angles of agent 3 are presented in Fig. 3.4. Note that the outer end of the FOV was reached at approximately $-43.5°$ and $43.5°$ for agent 1 and 2 respectively. Thus, the mean error of the measured azimuth angle between our algorithm and the ground truth was computed only when the respective other robot was visible within the FOV of agent 3. In those instances, we found the mean error of the visual estimate to be $0.32°$ with a standard deviation of $1.52°$. The tracking error between our visual estimate and the desired azimuth angle was $0.10°$ with a standard deviation of $4.25°$. For the elevation, as expected, we found results similar to the first experiment, namely $0.25°$ for the mean error of the visual angle estimate and $1.41°$ for the standard deviation.

To compensate for roll and pitch motion associated with accelerations of the quadrotor, we tested a compensator based on IMU readings in simulation. However it turned out that this compensation was not improving the performance of the system significantly. Thus, we decided to omit the compensation which would have introduced the need of an additional low-level communication pathway between the microcontroller and the onboard PC.

3.2.5. Conclusions

In this section, we were able to demonstrate the feasibility of a decentralized formation control for flying robots relying purely on visually estimated relative bearing angles between agents and the perception of ego-motion. The bearing angles were computed using an onboard visual detection system. The validity of the system has been analyzed using a group of three real quadrotor UAVs. The system proved to be stable even with a large temporal lag, introduced by the usage of low-cost camera hardware in combination with constrained processing power.

The contributions of the suggested framework are therefore two-fold:

- Presentation of an effective solution to cope with limited field of views on quadrotor UAVs
- Conduction of the first closed-loop human-controlled formation flight of a group of UAVs based entirely on perception and computation techniques that can be implemented on onboard hardware and without the need to obtain absolute or relative pose estimates

3.2.6. Future Work

With the aim to become fully independent of external tracking devices, we are planning to integrate the optical flow based velocity estimation system presented in Chapter 2 on a secondary visual system of each vehicle. The resulting formation control would be completely independent of any external ground station or motion tracking system. We are further planning to extend the system to include the avoidance of obstacle points based on the detection of bearing angles towards the obstacles. Additionally, we consider to autonomously choose the optimal magnitude of the scanning motion $\theta_{rot}(t)$ based on the formation layout itself.

Furthermore, to increase the robustness of the system to colored or poorly illuminated environments, we plane to replace the current colored ball shown in Fig. 3.2a in future projects. In Fig. 3.5, the active LED-powered beacon is shown that we developed for this purpose in 2011. The bright light facilitates very short shutter speeds and less sensitivity to false detections caused by other colored objects in the scene. In the last years, this design has been adopted by other research labs and companies for similar purposes and even art displays.

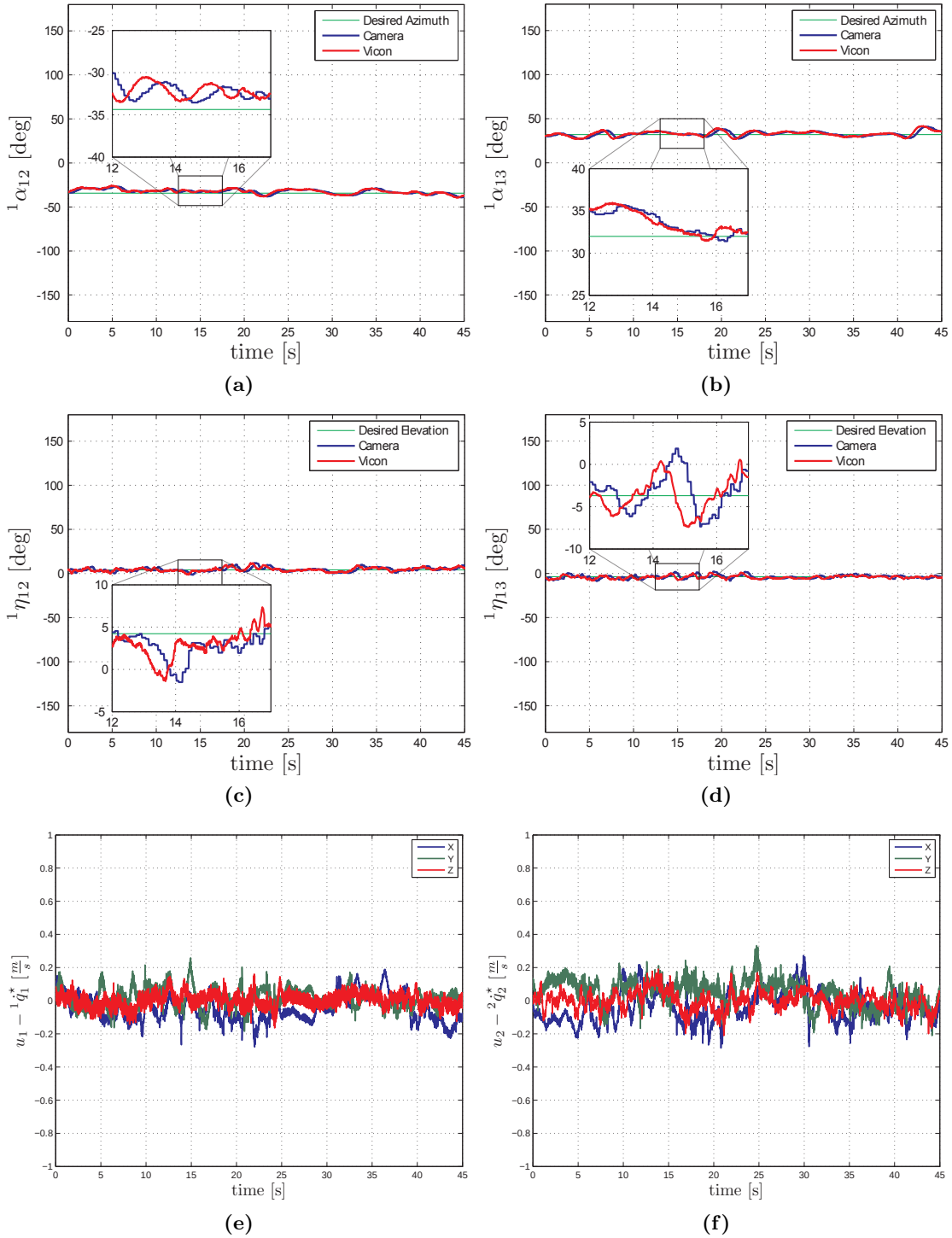


Figure 3.3.: Bearing angles as obtained from our onboard visual system in comparison with the ground truth and the desired bearing angles when all agents could directly observe the bearings to all other agents at all times. (a) Azimuth angle between agent 1 and 2 and (b) between agent 1 and 3. (c) Elevation angle between agent 1 and 2 and (d) between agent 1 and 3. (e) Velocity tracking error of agent 1 and (f) agent 2.

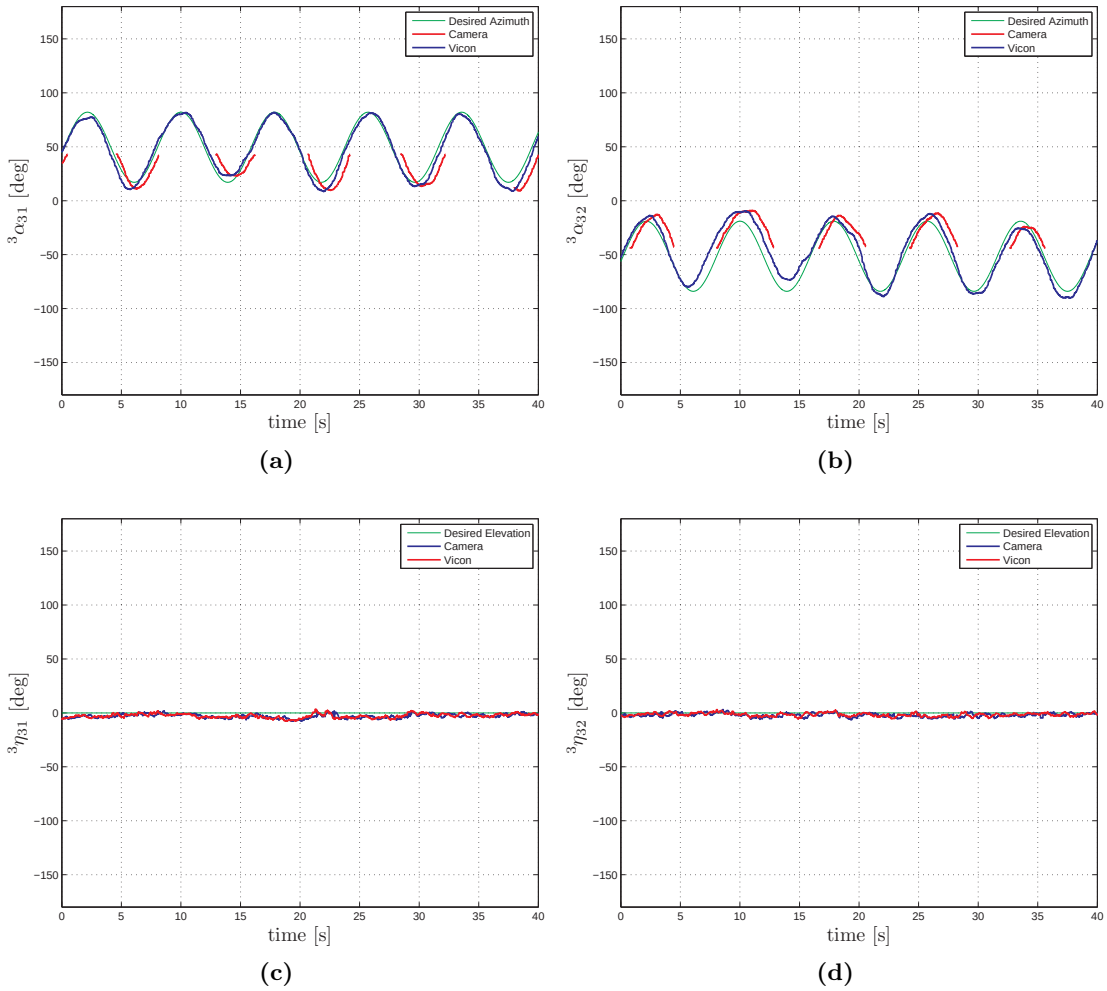


Figure 3.4.: Azimuth and elevation angles during the second experiment on bearing angle formations. (a) Azimuth angle between rotating agent 3 and agent 1 (static yaw angle) and (b) between agent 3 and 2. The output of the visual bearing estimation system (red) is only plotted when the corresponding other agent was within the FOV of agent 3. (c) Elevation angle between agent 3 and 1 and (d) between agent 3 and 2.

Figure 3.5.: Sanded acrylic glass beacon with embedded RGB LED on top of the modified Playstation eye3 camera.

3.3. Global Localization of a Monocular Camera in Dense 3D Maps through Photometric-Error Minimization

Over the last years, Simultaneous Localization and Mapping (SLAM) systems have become highly popular for the navigation of autonomous mobile robots. While navigation based on laser range finders has been used on mobile robots for many years, the availability of small and lightweight cameras has opened a new field of SLAM research even on Micro Aerial Vehicles (MAVs), such as quadrotors [Blösch et al., 2010]. Based on cameras, robots are able to continuously track their pose in all 6 degrees-of-freedom (DoF) while constructing a three-dimensional map to recover their location even after tracking was interrupted [Nützi et al., 2011; Weiss et al., 2011, 2012a].

The recent introduction of affordable combined color and distance sensors, known as RGBD sensors, has led to several innovative improvements in precision and reliability, which further increased the range of image based SLAM techniques [Shen et al., 2012].

These advances in autonomous map building naturally raise the problem of mutual localization of multi-robot teams, an ability essential to accomplish common tasks with joint efforts. The problem of mutual localization becomes fundamentally important in the growing field of rescue robotics with heterogeneous agents. Each agent (both robot or human) may carry different sensors and possesses different computational resources. Possible scenarios are ground robots which are supported by MAVs to find the best path through rough terrains after natural disasters. Additionally, MAVs of different sizes could be used to pass through small openings otherwise inaccessible. Even humans could benefit largely when working in a mixed human-robot team. After recent disasters, robots were sent to inspect and map hazardous areas [Michael et al., 2012]. Human forces could be guided more effectively when using wearable camera-based localization systems.

For many applications, it is important to localize an agent with a camera within a 3D map previously obtained by another agent. In this work, we tackle the problem of precise global localization of a camera-equipped mobile agent in a dens 3D map created by another agent. The 3D map could likewise be created using any monocular or stereo dense reconstruction system, however, in this work and w.l.o.g., we resort to RGBD sensors. One considered setup is shown in Fig. 3.6.

To estimate the pose of a camera, most state-of-the-art localization approaches are based on a minimization of the reprojection error of a sparse feature set in a least-square sense. Therefore, a large amount of information contained in the camera image is neglected completely. Furthermore, least-square optimizations are known to be highly sensitive to noise and, thus, require an efficient additional outlier-rejection step. To overcome these limitations, computationally expensive map-maintenance algorithms, such as bundle adjustment, are a common choice. However, this limits the possible map size drastically [Blösch et al., 2010; Newcombe et al., 2011b].

Related Work

To the best of our knowledge, this is the first work that takes advantage of using the whole image information to achieve a precise global localization of a camera with respect to a given dense 3D map from a different sensor.

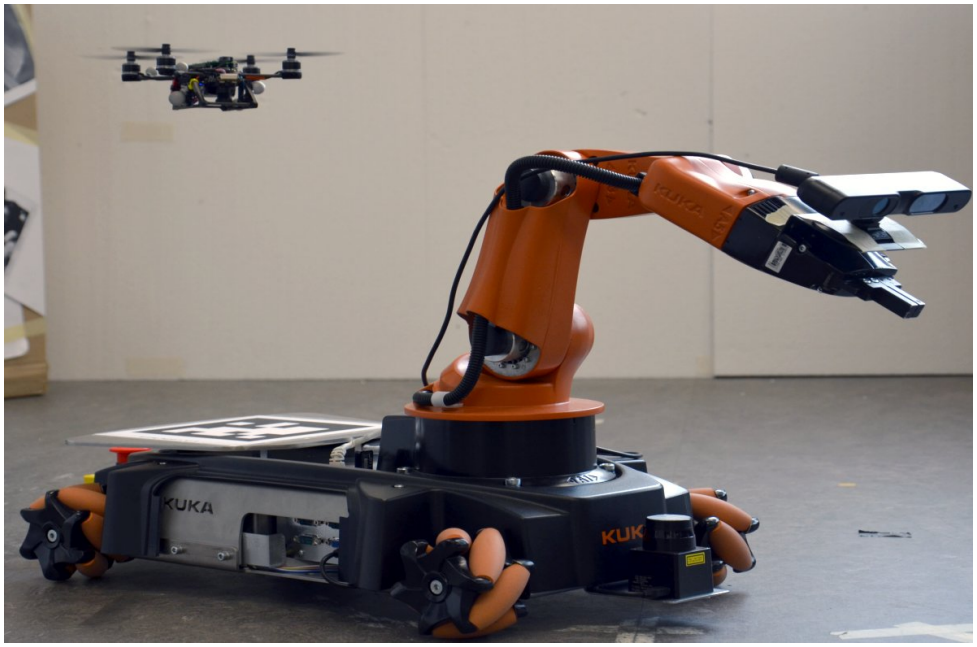


Figure 3.6.: Example setup considered in this work. One small quadrotor MAV with just a monocular camera is localized globally within a dense map created by a second robot with, e.g., an RGBD sensor.

Early work on dense real-time monocular 3D reconstruction and localization is reported in [Newcombe et al., 2011b; Lovegrove, 2011]. Similarly to our approach, they achieve robust camera tracking in small environment by minimizing the photometric error. However, their systems are not designed to provide any re-localization abilities and are, therefore, unsuitable for mutual localization tasks involving multiple robots. Furthermore, they require powerful additional hardware in the form of a fast Graphics Processing Unit (GPU) to achieve real-time performance.

Based on an RGBD sensor, *KINECT fusion* was proposed in [Newcombe et al., 2011a]. The system builds a texture-less dense 3D map on the GPU entirely based on the depth image but does not attempt loop-closure detection and the resulting pose estimate therefore drifts over time. Since it does not store any textures, the tracking fails on planar and other repetitively structured surfaces. For the same reason, KINECT fusion cannot be used to localize other cameras that do not provide depth images.

RGBDSLAM [Engelhard et al., 2011] builds a dense point-cloud and further maintains a sparse feature set for internal loop-closure detection. This feature set could be used for a global localization of a second camera as well. However, the reprojection-error–based optimization of the pose estimate relies on matching a sparse set of visual features only. In our experiments, we found the resulting pose estimate to have an error of up to 10% of the scene depth.

In our approach, we use the sparse feature set stored with an RGBDSLAM-based 3D map to obtain a prior for a novel iterative analysis-by-synthesis pose refinement process that minimizes the photometric error between a synthesized view of this dense map and the camera image.

In an effort to avoid confusion, we refer to the RGBD camera as 'RGBD sensor' or just 'sensor' unless otherwise stated. The monocular camera is consequently addressed as 'camera'.

The remainder of this section is structured as follows. In Sec. 3.3.1, we first describe the construction of the underlying 3D map and consequently state the problem formulation. In Sec. 3.3.3, we explain the calculation of the initial pose estimate of the camera. This initial position is then refined using our proposed method described in Sec. 3.3.4. In Sec. 3.3.6, relevant implementation details are discussed. The system is experimentally evaluated in Sec. 3.3.8 and the results are presented in Sec. 3.3.9. Finally, this work is concluded in Sec. 3.3.10.

3.3.1. Construction of the 3D Map

In this section, we summarize the construction of a 3D map that consists of both a dense 3D point-cloud and sparse feature descriptors.

We start from the open-source implementation RGBDSLAM [Engelhard et al., 2011] of an algorithm suggested earlier in [Henry et al., 2012]. Both the 3D points as well as visual feature locations are stored in local coordinate frames $\mathcal{K}_i$ associated with keyframe $\boldsymbol{k}_i$. All keyframes are arranged within a pose graph that is extended as the map grows. During motion, the RGBD sensor is constantly tracked and whenever the current location is sufficiently far from any previous keyframe, a new keyframe is added to the graph. Additionally, a sparse set of visual features $\boldsymbol{F}_i$ associated to each keyframe is used to match the current view against all existing keyframes to detect loop-closures. For this, we rely on matching SIFT (Scale-Invariant Feature Transform) feature descriptors [Lowe, 1999]. Pose-graph optimization is consequently used to reduce the error within the keyframe graph using all detected loop-closures.

Since the original RGBDSLAM implementation lacks several important properties, we improved the overall performance and added the storage of all original keyframe images. Furthermore, the original implementation neglects camera-distortion parameters completely and does not align the RGB sensor with the depth images.

To simplify the interaction with the 3D map for the proposed setup, we implemented an interface to RGBDSALM based on the well known Robot Operating System (ROS).

3.3.2. Problem Formulation

We address the following problem. Let M be a given 3D map composed of l keyframes $\boldsymbol{k}_i,\ i \in [1 \dots l]$. A coordinate frame $\mathcal{K}_i$ is associated to each keyframe i and each keyframe holds a dense set $\boldsymbol{P}_i$ of colored 3D points $\boldsymbol{p}_j,\ j \in [1 \dots m]$ together with a sparse set $\boldsymbol{F}_i$ of feature descriptors $\boldsymbol{f}_h,\ h \in [1 \dots N]$. Without loss of generalization, $\mathcal{K}_1$ is considered to be the origin of the map. Furthermore, let $I^{\mathcal{C}}$ be a monocular camera image as seen from camera frame $\mathcal{C}$. We attempt to find the best transformation ${}^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{C}}$ from the origin of the map $\mathcal{K}_1$ to the camera frame $\mathcal{C}$ that minimizes the photometric error.

3.3.3. Computation of an Initial Pose Estimate

In this section, we describe a method to compute an initial global estimate of the camera location that is later refined using a local analysis-by-synthesis optimization technique (see Sec. 3.3.4). For the global estimate, we rely on an efficient closed-form solution of the *Perspective-3-Point problem* (P3P) [Kneip et al., 2011b]. Given the 3D location of three points in some frame $\mathcal{K}$ together with their corresponding directional observations from a calibrated camera in frame

$\mathcal{C}$, the algorithm computes two solutions for the unknown roto-translation ${}^{\mathcal{K}}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{C}}$. A fourth feature is then used to solve the ambiguity. For the details of this algorithm, we refer the reader to [Kneip et al., 2011b].

To create the complete set of 3D map feature locations from all keyframes for the P3P algorithm, all keyframes of the map have to be rotated such that their associated feature locations are expressed in the frame of the origin, in our case frame $\mathcal{K}_1$ associated to the initial keyframe $\boldsymbol{k}_1$. The transformation ${}^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{K}_i}$ necessary to rotate all N feature locations $\boldsymbol{f}_h, h \in [1 \cdots N]$ of a keyframe $\boldsymbol{k}_i$ into $\mathcal{K}_1$ is obtained from the pose graph.

Since we rely on SIFT feature matching to obtain the correspondence between the up to $N \cdot l$ features of the map and those found in the current camera image, it is not advisable to match the image features against all features of the map at once. Using the original implementation of SIFT feature matching, a match would be considered reliable only if it differs sufficiently from the next best match [Lowe, 1999]. Since the same feature is usually visible in multiple keyframes with almost the same descriptor, matching all features of the map at once would result in many candidate matches with similar descriptors, which in fact correspond to the identical visual feature in the real world. Thus, a more time-consuming match-selection process involving comparisons with all neighboring matches would become necessary to select good features that defer sufficiently from the next best match that actually corresponds to a different feature. Instead, we match the features of one keyframe at a time against the camera image and add the resulting matches to a global set of good matches.

Having established the correspondences between the features visible in the camera image and those stored in the map, *Random Sample Consensus* (RANSAC) is used to discard outliers. The system repetitively selects four feature matches to compute a model of the camera location and orientation. It is ensured that these four features are in a non-degenerate configuration, that is, that their 3D locations are adequately spaced and that they are not collinear. This process is repeated until a sufficiently-large number of additional features supporting this model is found. Finally, the transformation ${}^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{C}}$ is computed again using this extended set of matches.

In our experiments, when using this initial pose estimate, we found an accuracy of 0.1 m to 0.2 m when the camera was located 1 m to 2 m from the scene. This finding of an error of 10% of the scene depth corresponds to the error reported by the authors of [Kneip et al., 2011b]. In the next section, we describe our novel analysis-by-synthesis driven algorithm to achieve a more precise localization based on this initial pose estimate.

3.3.4. Iterative Pose Optimization

To locally optimize the camera roto-translation estimate by minimizing the photometric error, we propose an iterative two-step algorithm. Starting from an initial global estimate through putative point correspondences, we alternate between a rendering step—in which the dense map is rendered into an image using the parameters of the calibrated monocular camera—and an image-alignment step that computes an improved camera pose by minimizing the photometric error between the synthesized view and the real camera image through image warping.

Rendering of a Synthesized Camera View

In this step, an image of the 3D map is rendered from the latest pose estimate of the camera using the known intrinsic camera and distortion parameters obtained during an initial camera calibration step.

As described before, the 3D map consists of l keyframes $\boldsymbol{k}_i$, $i \in [1 \dots l]$ where each frame has a local coordinate frame $\mathcal{K}_i$ assigned to it. These frames follow the common camera frame convention and the z-axis is aligned with the camera principal axis. Up to m 3D points ${}^{\mathcal{K}_i}\boldsymbol{p}_j$, $j \in [1 \dots m]$ per keyframe have to be rotated into the common base frame $\mathcal{K}_1$ first. Here, m corresponds to the number of pixel in the RGBD sensor image, e.g. $m = 307,200$ for a Kinect or Asus Xtion. Consequently, all points ${}^{\mathcal{K}_i}\boldsymbol{p}_j$ are rotated into $\mathcal{K}_1$ using ${}^{\mathcal{K}_1}\boldsymbol{p}_j = {}^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{K}_i}\, {}^{\mathcal{K}_i}\boldsymbol{p}_j$. Once again ${}^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{K}_i}$ is obtained from the internal pose graph of the 3D map.

In order to reduce the time needed to project all up to $m \cdot l$ map points into the virtual camera image, we do not consider keyframes for which the z-axis differs from the current principal axis of the camera orientation by more than a thresholded angle. Afterwards, other points far to the sides or behind of the camera are removed as well.

All remaining points ${}^{\mathcal{K}_1}p_j$ are projected onto the virtual camera image. Since, usually, more than one point of the map would be projected into the same pixel, only the point closest to the camera is rendered. Thus, while iterating over all points, we construct both a z-buffer depth image (containing the scene depth for each pixel) and the resulting output image. This simplified rendering procedure has linear-time complexity on the number of points in the map and is further linear in space complexity on the number of pixels in the virtual camera image.

An example of a resulting synthesized image is shown in Fig. 3.11a.

3.3.5. Minimizing the Photometric Error

To optimize the viewpoint, we propose a method that minimizes a dense photometric cost function for every pixel. This approach has been pioneered by Lucas and Kanade [Lucas and Kanade, 1981] to compute the optical flow in an image. It works by using the spatial intensity gradient of a small image patch to accelerate the search. Many variations of the original Lucas-Kanade tracking have been proposed (see, e.g., [Baker and Matthews, 2001] for a summary). While the initial method was intended to optimize a 2D alignment, it can also be used to estimate the 6 degree-of-freedom (DoF) pose of one camera with respect to a second camera image [Lovegrove, 2011].

For the photometric cost function to minimize, we chose the sum of squared differences between the pixels in the synthesized view I^S and the image from the monocular camera I^C:

$$F(\mathbf{T}) = \frac{1}{2} \sum_{\mathbf{x} \in \Omega} \left(I^C\left[f_{\mathbf{x}}(\mathbf{T})\right] - I^S[\mathbf{x}] \right)^2 \qquad \text{where}$$

$$f_{\mathbf{x}}(\mathbf{T}) = \pi \left({}^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{C}}\, \mathbf{T} \pi^{-1}(\mathbf{x})\, \xi(\mathbf{x}) \right).$$

The $\pi()$ and $\pi^{-1}()$ functions represent the pinhole camera projection and back-projection functions respectively and $\xi(\mathbf{x})$ is the depth at pixel $\mathbf{x}$. Ω denotes the set of pixel locations in I^C and I^S as both images have the same dimensions by construction. ${}^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{C}}$ is the homogeneous 4×4 matrix rigid-body transformation between the camera frame $\mathcal{C}$ and the frame of the dense

map $\mathcal{K}_1$. The cost F is minimized iteratively through finding an update transformation $\mathbf{T}$ that minimizes the current cost:

$$\hat{\mathbf{T}} = \arg\min_{\mathbf{T}} F(\mathbf{T})$$

$$^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{C}} \leftarrow {}^{\mathcal{K}_1}[\boldsymbol{R}|\boldsymbol{t}]_{\mathcal{C}}\ \hat{\mathbf{T}},$$

where the incremental transformations $\mathbf{T}$ are parametrized using the Lie algebra $\mathfrak{se}_3$. The rigid body transformation update $\mathbf{T}$ can be recovered through the exponential map $\mathbf{T} = \exp(\mathbf{T})$. This nonlinear least-squares problem can be solved by means of the Gauss-Newton or Levenberg-Marquardt algorithms [Baker and Matthews, 2001].

For the localization of the considered monocular camera, $\xi(\mathbf{x})$ is unknown. Thus, we approximate $\xi(\mathbf{x})$ with the scene depth found at the same pixel in the rendered view I^S of the dense 3D map.

3.3.6. Implementation

In this section, we cover the most important implementation details concerning the theoretical approach discussed in the previous Sec. 3.3.4.

Rendering of the Synthetic View

In an effort to speed up the rendering process of the synthesized camera view, only points from keyframes for which the z-axis is oriented similarly to the principal axis of the camera are considered. Although SIFT features are known to be viewpoint-angle invariant for up to 60° on a planar scene, empirically, we did not find any reliable matches on more complex 3D scenes once the camera was tilted by more than 40°. As suggested in the literature [Henry et al., 2012], the best result for a synthesized view is obtained when planar patches with normals pointing in the direction of the camera are used. Since the available computational power on robots is insufficient for an extraction of local patches for all points, all feature point patches are assumed to share the normal of the image plane of their respective keyframe. Thus, the choice of a normal angle of 40° from the principal axis of the camera as a threshold to exclude keyframes ensures a good compromise between selecting points with a patch facing the camera, the need to render a sufficiently large area of the scene, as well as efficiency considerations. Additionally, this filter step removes points from backsides of objects invisible to the camera efficiently. Potential holes[2] in the rendered image, caused by this filtering step or the lack of information in the 3D map at that location, are ignored in the image alignment process described in Sec. 3.3.5.

Furthermore, points too distant from the line of sight of the camera are not considered. The choice of the used threshold for the rejection of distant features depends on the field of view of the camera and is necessary to avoid projection artifacts caused by lenses with strong distortions.

Localization of the Monocular Camera

The synthesized camera view usually suffers from rough edges and sometimes small rendering artifacts. Thus, both images I^S and I^C need to be filtered with Gaussian blur prior to the alignment step.

[2] Pixels in a synthesized image which are undefined since no color was assigned to them are often referred to as 'holes'

Before aligning the rendered view I^S with the original image I^C, the mean intensity of both images is equalized for all pixel locations $\boldsymbol{x} \in \Omega$: $I(\boldsymbol{x}) = I(\boldsymbol{x}) + (c - \bar{I})$. $\bar{I}$ denotes the mean intensity of the image I and c is a constant which is additionally added to avoid dark low-contrast images otherwise caused by the mean subtraction. Furthermore, all regions of the rendered image for which there was no information available in the map are masked as they should not be considered for the alignment. That is, all pixels unsuitable for the comparison of I^S and I^C are removed from the set Ω of considered pixels in the images. This ensures the robustness of our algorithm even in the presence of holes in the rendered image or when only portions of the mapped scene are visible from the viewpoint of the camera.

Computation of the Relative Photometric Error

To allow for an evaluation of the suggested pose refinement approach with respect to the 3D map (that is, independent of errors present within the pose-graph of the RGBDSLAM map used for the implementation), we had to resort to a direct measure of the photometric error reduction. Indeed, for a globally minimal photometric error, the synthesized view of the 3D map corresponds to the camera image in the best possible way and any further movement of the camera in any of the 6 DoF would increase the error. Since all information available in the camera image is used for the computation of the photometric error, the camera pose estimate cannot be improved any further by resorting to other means of localization. Hence, since we aim for an optimal localization of the camera with respect to the 3D map, an external tracking system cannot be used to provide a ground truth since the 3D map will never be perfectly aligned to the origin of the motion tracking system. In fact, given a minimal photometric error, a ground truth from a motion tracking system will only measure the internal error of the pose-graph that relates the keyframes of the chosen 3D map system, in our case RGBDSLAM. However, for successful joint multi-robot applications, a mutual localization of the robots with respect to *each other* is essential, while minor inaccuracies with respect to some arbitrary *global* map are usually acceptable. For this reason, we resort to the photometric error for the evaluation of the complete system. However, to obtain a comparison with a ground truth measurements as well, we devised an additional experiment on a map consisting of just a single keyframe, thus, eliminating possible errors within the pose-graph of the map (see Sec. 3.3.8).

Assuming a noiseless 3D map, an ideal camera calibration, and an environment that is not self-similar or otherwise ill-structured, the iterative camera pose refinement process will eventually converge into an optimal pose for which the photometric error of the rendered camera view with respect to the real camera image reaches a global minimum. In this case, the accuracy of the refinement approach is bounded by the resolution of the camera.

Since two completely different cameras are involved, the intensity of the same patch in both images is not identical despite the mean equalization. Thus, the absolute photometric error expressed as the sum-of-square difference of pixel intensities in the synthesized image I^S and the real camera images I^C

$$r = \frac{1}{\|\Omega\|} \sum_{\boldsymbol{x} \in \Omega} [I^S(\boldsymbol{x}) - I^C(\boldsymbol{x})]^2 \tag{3.10}$$

will never reach zero. Therefore, the photometric error as such cannot be used for the comparison of the refinement iterations. Instead, for the evaluation, we employ the relative photometric

Algorithm 1: Incremental camera pose estimation

Input: 3dMap M, cameraImage I^C
Output: cameraPose $[\boldsymbol{R}|\boldsymbol{t}]$
1 cameraPose $[\boldsymbol{R}|\boldsymbol{t}] \leftarrow$ P3P(3dMap M, cameraImage I^C) *// see Sec. 3.3.3*
2 **for** 15 *iterations or until termination criterion reached* **do**
3 renderedImage $I^S \leftarrow$ renderImage(cameraPose $[\boldsymbol{R}|\boldsymbol{t}]$, 3dMap M) *// see Sec. 3.3.4*
4 cameraPose $[\boldsymbol{R}|\boldsymbol{t}] \leftarrow$ refinePose(renderedImage I^S, cameraImage I^C) *// see Sec. 3.3.5*
5 **end**

error $\hat{r}_i$ between the initial error r_0 and the error r_i after the ith iteration of the pose refinement process expressed in percentile $\hat{r}_i = 100(r_0/r_i - 1)$.

Selection of the Termination Criterion

In real world scenarios, the pose-graph of the map is not ideal and will cause small distortions within the rendered images. Thus, the iterative optimization might alternate between equally-good pose estimates in close proximity and a termination criterion becomes necessary. For the conducted experiments, we chose to run not more than 15 iterations and to interrupt the process once the suggested camera location refinement drops below 0.001 m.

Communication and System Layout

Our setup is designed to run on at least two autonomously operating robots, one equipped with, e.g., an RGBD sensor to build and maintain a dense 3D map, and the other one with just a monocular camera. We further assume that both agents provide at least a small amount of onboard processing power and share a common communication channel. ROS is used to provide inter-process communication. Our system is flexible in that the computationally expensive steps could be carried out on either of both agents or even on an independent ground station, depending on availability.

3.3.7. Summary of the Algorithm

Algorithm 1 summarizes the proposed system for the precise localization of a camera within a dense map. In line 1, an initial estimate of the camera pose is computed as described in Sec. 3.3.3. The iterative refinement is carried out in line 3 and 4. First, a synthesized image is rendered from the current camera pose estimate according to Sec. 3.3.4. The rendered image is then aligned in all 6 DoF with the real camera image to update the estimated viewpoint of the camera following the explanations in Sec. 3.3.5.

3.3.8. Experiments

For the evaluation of our algorithm, we have conducted three experiments. First, we evaluate the full system and simulate the localization of a camera-equipped robot within a dense 3D map. Here, we discuss results on a cluttered three dimensional scene constructed from multiple boxes in detail. The findings are confirmed by a second similar experiment on a desktop scene.

Additionally, we tested the proposed system in an motion capture system to compare the results against a metric ground truth.

Hardware Setup

Our experimental setup consists of two main components: the RGBD sensor carried by one robot and a monocular camera on a second robot. Figure 3.6 illustrates a potential setup of a ground robot holding an RGBD sensor and a miniature quadrotor equipped with just a camera as considered in our experiments.

A MatrixVision BlueFox monochrome camera (see App.A.2) was used to localize the second agent with respect to the 3D map. The camera was moved and rotated randomly within a volume of approximately $4 \times 3 \times 2\,\mathrm{m}^3$ in front of the mapped scene.

To ease the conduction of the experiments, we ran all parts of the distributed algorithm on a single laptop using communication through localhost. Clearly, most parts of the proposed algorithm could be parallelized on dedicated GPU hardware for real-time performance. Since this exceeds the scope of this work, all computations are currently carried out on two cores of the CPU.

Conducted Experiments

To demonstrate the precision and reliability of the presented approach, we tried to localize a camera within a given 3D map and to further refine its initial position estimate.

For the first experiment, the mapped environment consisted of a cluttered scene of approximately $3 \times 2\,\mathrm{m}^2$ with elements up to 1 m in height. The scene is shown in Fig. 3.11b. An Asus Xtion Pro RGBD sensor was used to create the 3D map exploiting the modified RGBDSLAM system described in Sec. 3.3.1. Repeatedly, we tried to localize the camera from 130 randomly-selected images acquired on a 10 m trajectory within the 3D map of the environment. The map was composed of 75 keyframes and contained around 25 million 3D points and 22,000 feature descriptors.

To increase the validity of our findings, we repeated this experiment in a second setting, a cluttered desktop environment depicted in Fig 3.7. In this setting, 200 more diverse camera locations were estimated from a map consisting of 30 keyframes.

In a third experiment, we tested our approach in a motion capture system to compare the improved pose against a metric ground truth. To avoid errors introduced by an imprecise pose-graph of the third-party RGBDSLAM system, we created a small 3D map from a textured wall consisting of just a single keyframe. The origin of this keyframe was located using an OptiTrack motion tracking system from NaturalPoint with millimeter precision. Consequently, a camera was localized from 15 tracked ground truth locations within this map. The initial pose estimate using the state-of-the-art P3P system was compared to the result of the proposed optimization algorithm using the ground truth locations of the camera.

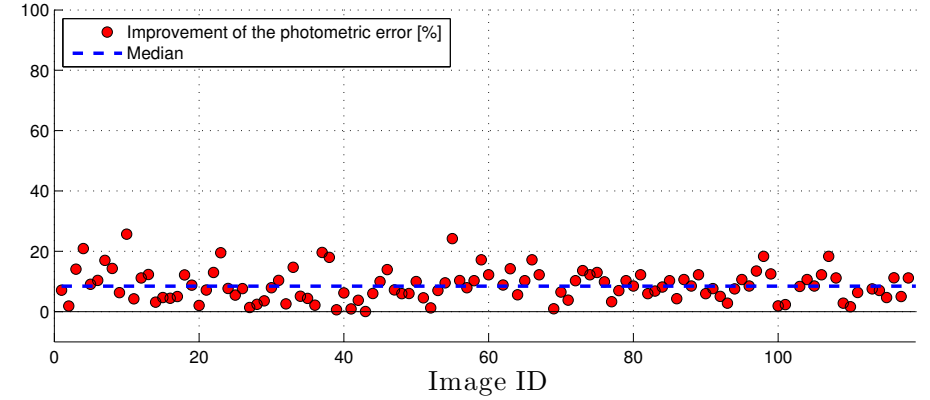
3.3.9. Results and Discussion

Overall, 93% of all tested camera poses were recovered successfully within a small range of their real location. In our experiments, the system was always able to detect unsuccessful localizations

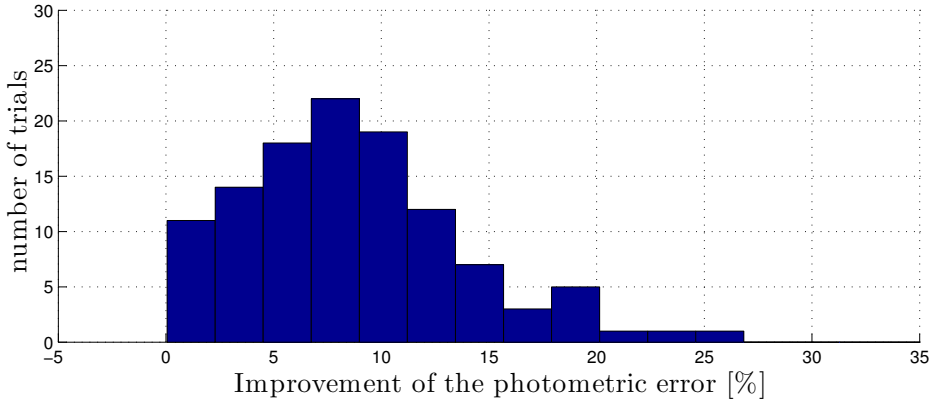
Figure 3.7.: Part of the desktop scene used in the experimental evaluation.

	Cluttered	Desktop	Wall
Successful localizations	97%	90%	87%
False-positive localizations	0	0	0
Improvement of the photometric error	19%	25%	10%
Scene depth (average)	2 m	1.5 m	0.8 m
Position update (average)	0.05 m	0.05 m	0.02 m
Position update (maximum)	0.19 m	0.19 m	0.03 m
Orientation update (average)	1.5°	0.9°	0.5°
Orientation update (maximum)	6.6°	4.2°	1.5°
Distance from closest keyframe (avg.)	0.70 m	0.95 m	0.20 m

Table 3.1.: Comparison of the results for the three environments.



(a)



(b)

Figure 3.8.: Improvement of the photometric error after the iterative refinement of 118 camera pose estimates. (a) Improvement of the photometric error for each image together with the median for all 118 camera poses. 4 trials with improvements of more than 100% are not shown. (b) Histogram of the relative photometric error.

attempts. Hence, it did not report any false-positive localizations. The major results for all three experiments are summarized in Tab. 3.1.

Experiment on a Cluttered Environment

In the first experiment, 126 out of 130 camera locations were successfully recovered in the initial global localization step. In 2 instances, the state-of-the-art P3P based global initialization step failed to predict a camera location at all, while in the remaining 2 instances, a wrong initial estimate was returned. Opposed to existing approaches, our system was able to detect these 2 false positives by construction since the predicted view and the real image did not match. In an additional 8 cases, the iterative error minimization process was unable to further improve the initial estimate. Figure 3.8a shows the improvement of the photometric error for the remaining 118 trials. Here, the initial camera pose estimate was refined by 0.004 m to 0.190 m with an average of 0.046 m. The orientation estimate was improved by 0.02° to 6.55° with an average

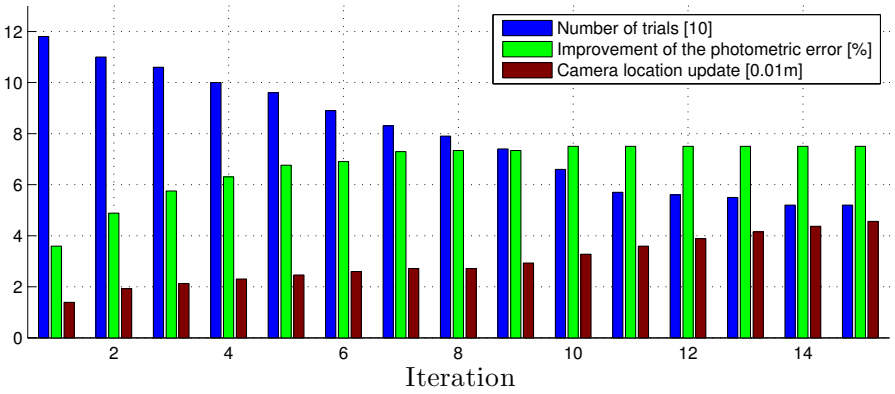


Figure 3.9.: Influence of the number of refinement iterations on the relative photometric error in percent together with the total distance the camera pose estimate was optimized with respect to the initial estimate in [cm]. In blue, the number of the 118 camera localization trials which passed an iteration without reaching the termination criterion as defined in Sec. 3.3.6 is plotted. The number of this trial instances is plotted in units of [10].

of 1.45°. The improved estimate of the camera pose in average reduced the photometric error by 19.27% with respect to the synthesized view from the initial pose estimate. The median was 8.45% (see Fig. 3.8b). In four cases of poor initial pose estimates, the relative photometric error was improved by 100% up to 334%.

In Fig. 3.9, the influence of each iteration during the camera position refinement process on the photometric error is visualized together with the distance the camera pose has been corrected from the initial estimate. The plot also illustrates the number of images that did not reach the termination criterion (as defined in Sec. 3.3.6) after each iteration. It becomes clear that, after approximately 8 iterations, the gained improvement per iteration does not justify further optimization in most cases. Depending on the required precision, the optimization phase could be shortened by choosing a more relaxed termination criterion at the expense of less accurate pose estimates.

The reduction of the photometric error is visualized in Fig. 3.10 by an exemplary camera pose estimation in each of the three environments. The actual camera image was subtracted from the synthesized view before and after the pose refinement process.

During all localization trials, we also recorded the keyframes of the 3D map with closest distance to the camera to investigate the viewpoint invariance of our system. For some images, the camera was located up to 2.01 m away or was rotated by up to 61.9° with respect to the closest keyframe. In average, the distance between the camera and the closest keyframe was 0.695 m while the average orientation difference was 13.4°. Figure 3.11b shows an image of the monocular camera next to the image of the closest keyframe in Fig. 3.11c. Clearly, the proposed system is robust to different viewpoints of the camera with respect to the trajectory of the RGBD sensor during the creation of the 3D map.

In the current implementation, all computations were carried out on a single core of a laptop CPU. Thus, the computation of the initial P3P based global localization took in average 0.25 s while the generation of each synthesized image took 0.75 s. Each image alignment required

(a) (b) (c)

Figure 3.10.: Absolute-difference images between the original monocular image and the synthesized view for three camera poses, each in one of the three tested environments. The upper row shows the real camera image of the scene. The middle row visualizes the photometric error after the state-of-the-art P3P-based pose estimate while the lower row presents the error after the proposed algorithm has been applied to improve the pose estimate. Red indicates regions with high photometric error while blue represents a small error. (a) Cluttered environment constructed from boxes. The RGBDSLAM keyframe that covers the lower left corner of the map was slightly misaligned causing a constant photometric error in that region of the image. (b) Desktop scene. Since the entire scene was covered by a comparatively small number of keyframes, the internal error of the map is relatively large and thus the final estimate still contains regions with large errors. (c) Wall scene from the ground truth experiment with a map composed of a single keyframe. The average error is relatively high due to very different lighting conditions during map creation and the camera localization.

(a) (b) (c)

Figure 3.11.: View of the scene as (a) synthesized view from 18 million points of the 3D map after pose optimization, (b) seen from the wide-angle monochrome MatrixVision Bluefox camera and (b) the closest keyframe of the 3D map captured using an Asus Xtion. Note that unique features such as the checkerboard are not specifically exploited by the algorithm and have been placed purely for visualization purposes (see Fig. 3.10a).

0.05 s. Thus, without the availability of specialized hardware, such as a GPU, the proposed precise localization framework would be a good choice for a periodical background map-alignment process. To use this CPU-based system for pure pose tracking, additional fusion with other state estimation sensors, such as odometry readings or an inertial measurement unit, would be necessary for high frequency estimates. However, when implemented on a GPU, real-time performance should be achievable since GPUs are tailored for image rendering and alignment. Depending on the design of the experimental setup, additional network transportation delays have to be considered.

Experiment on a Desktop Scene

Due to the smaller number of keyframes compared to the box environment that was used to cover a similarly sized environment, the average distance between the camera and the trajectory of the RGBD sensor during map creation was 0.95 m and therefore the rate of successful localization was slightly lower at 90%. This drop is due to more frequent failures of the initial P3P pose estimation, while in all other cases our system was able to improve the estimate above the state-of-the-art. Over the 90% of successful localizations, on this dataset, our algorithm was able to reduce the average photometric error by 25%. All other results support the findings of the first experiment.

Experiment with Ground Truth Comparison

All 15 localization attempts were successful and in 13 cases our algorithm was able to improve the camera pose estimate by up to 0.03 m with respect to the ground truth location in the given small environment of less than 1 m in depth. In two cases, the initial estimate could not be improved any further. This finding underlines that our system also allows to globally improve the pose estimate with respect to an external world frame, provided that an ideal map is given. In this small and planar environment, the state-of-the-art approach provides good results as well and thus the additional improvement of our system is smaller compared to the other experiments.

3.3.10. Conclusions

In this work, we proposed a method to precisely estimate the pose of a monocular camera with respect to a dense 3D map. For this purpose, we first calculate an initial global estimate of the camera pose by solving the P3P problem embedded in a RANSAC framework. Afterwards, this initial state-of-the-art pose estimate is further refined locally in a novel iterative analysis-by-synthesis process which aims to minimize the photometric error between the rendered view of the 3D map from the estimated camera pose and the real camera image. Thus, all information captured within the camera image is exploited. For this purpose, a rendered view of the map from the estimated camera pose is aligned in 6 DoF to the actual camera image.

The system proved to be able to precisely localize the camera in 93% of the tested cases while not producing any false-positive localizations. Furthermore, our algorithm was shown to be robust to different camera viewpoints with respect to the 3D map. Experiments with an optical tracking system demonstrated that the proposed system not only improves the camera pose estimate with respect to the 3D map but also the estimate of its real world location.

Our work can be exploited for the mutual localization of two or more robots where only one is equipped with an dense mapping system. Therefore, our analysis-by-synthesis approach could greatly improve efficiency in multi-agent joint operations, such as rescue scenarios even in mixed human-robot teams.

3.3.11. Future Work

Currently, we are exploring possibilities to achieve real-time performance through the exploitation of current small-scale GPUs which became available even on board level computers recently. We expect that especially the time consuming rendering process of the synthesized camera image will benefit largely from the use of dedicated parallel hardware. A similar increase in performance can be expected for the other stages of our algorithm such as the SIFT feature detection as well as the image alignment process.

3.3.12. Acknowledgments

The authors like to thank Christian Foster for his contribution to the implementation of the image alignment process and furthermore Matia Pizzoli for the joint efforts to create the collected datasets.

4. Conclusions and Future Work

In this final chapter, we summarize our work and discuss future directions of research.

4.1. Conclusions

Robust and reliable state estimation is one of the essential components for the control of any robot. For operations independent from designated ground stations and external tracking system, such as, GPS or motion tracking systems, all required computations have to be carried out relying solely on onboard hardware and sensors. In this thesis, we addressed the problem of ego-motion estimation for flying robots, in particular quadrotor UAVs. Additionally, we propose approaches to tackle the mutual localization problem that arises from multi robot scenarios. Our major achievements could be summarized as follows:

- Velocity estimation from the decomposition of optical flow based on the continuous homograpy constraint using increasing amounts of information from onboard gyroscopes [Grabe et al., 2012a]
- Outlier rejection techniques to detect planar patches within the scene to increase the robustness of the least-square optimization based velocity estimates [Grabe et al., 2012b]
- Presentation of a nonlinear sensor fusion approach with IMU measurements to recover the metric scene depth and therefore the metric scale of the velocity estimates [Grabe et al., 2013a, 2014]
- Characterization of the convergence behavior for the scale estimation filter [Grabe et al., 2014]
- Formation control based on the visual observation of bearing angles towards the other members of the formation [Franchi et al., 2012], [Grabe et al., 2011, unpub.]
- Precise localization of a camera with respect to a dense map using an analysis-by-synthesis approach [Grabe and Scaramuzza, 2014]
- Release of parts of the underlying code to the public by means of the freely available TeleKyb control framework for UAVs [Grabe et al., 2013b]

In Section 2.2, we presented a method to estimate the velocity of a quadrotor UAV from the decomposition of optical flow given by consecutive observations from a down-facing monocular camera. In particular, we compared three different levels of gyroscope integration to simplify and improve the computed estimates. While the resulting velocity estimates are naturally not metric, we found our systems to yield highly accurate results in various experiments. Therefore, these estimates can be used for a lightweight directional human-in-the-loop control of real quadrotor UAVs. In this scenario, the human operator compensates the lacking metric scale factor with experience.

Using advanced outlier filtering techniques as presented in Sec. 2.3, we were able to segment features on planar patches of the scene to increase the robustness of the previously presented

approach. This is particularly important when relying on optimizations in the least-square sense since their accuracy is known to be affected by outliers. Thus, using our method, we are not only able to segment both objects on a planar scene or planar patches in an otherwise non-planar environment, but also drastically improve state estimates based on homography constraints such as the presented algorithm to recover the ego-motion of a quadrotor UAV.

In Section 2.4, we introduced a novel nonlinear observer to fuse non-metric velocity estimates from the visual system with metric readings from accelerometers to obtain the metric scale factor for the visual velocity estimates. In extensive experiments, we compared the proposed filter to an EKF implementation. For both simulated and recorded data, we found a highly improved convergence rate. Additionally, the fully characterized exponential convergence behavior allows to predict the otherwise unknown error of the scale factor for the acceleration profile of the robot. This has not been possible using any of the existing approaches. A gain factor allows to regulate between a fast conference rate and a noisy estimate depending on the requirements imposed by the current task. Having integrated this ego-motion estimation system into our control framework TeleKyb [Grabe et al., 2013b], we were able to present the first results from a closed-loop flight of a quadrotor UAV relying purely on onboard sensors and computations. As opposed to other approaches based on global pose estimation, the presented velocity based approach is robust to short temporary failures of the vision system since it does not require the interruption-free localization with respect to a map.

Being able to control a UAV using only onboard resources, we addressed the problem of mutual robot localization for joint operations of multiple robots in Sec. 3. In particular, whenever a direct line of sight exists among a sufficiently large subset of robots in a formation, we presented a control algorithm based on optically detected bearing angles to the other members of the formation in Sec. 3.2. We also provide a solution to overcome limited field of views when using actual camera hardware. The approach does not require knowledge of metric distances within the formation since they are often difficult to obtain by means of onboard sensors. Again, we suggest a human-in-the-loop approach to counteract for potential drift. The system was validated using a setup of three real quadrotor UAVs that relied on onboard visual bearing angle detection systems.

For situations in cluttered environments where a direct visual detection among the robots is impossible, in Sec. 3.3, we provide a solution to precisely globally localize a camera with respect to a dense 3D map. Hereby, the map could be equally well obtained using either an RGBD sensor or, e.g., a monocular dense reconstruction approach. An experimental evaluation demonstrated that our analysis-by-synthesis approach can achieve a considerable improvement over current state-of-the-art techniques. Naturally, apart from the localization of one robot with respect to a second, our approach could also be used to fuse multiple maps more precisely.

To conclude this thesis, we were able to develop and rigorously test a quadrotor UAV that is able to autonomously execute velocity commands by tracking the current metric velocity purely by means of onboard hardware: a camera, an IMU, and a small processing unit. We then developed two approaches that allow vision guided UAVs to collaborate in a multi robot setup. Here, we consider both a formation controller based on direct visual bearing angle observations and the indirect localization of a camera with respect to a dense 3D map created by an arbitrary visual dense mapping approach.

4.2. Future Work

In this global future work section, we particularly address new directions of research in the field of multi modal mapping related problems. Future work directly linked to one of the projects described in detail in Chapter 2 and 3, such as improvements and extensions, can be found in the respective future work sections.

The localization and pose estimation problem has been widely addressed for individual robots. In particular, in Chapter 2, we extensively addressed the problem of a camera based velocity estimation and control of a quadrotor UAV. In most cases found in the literature, the robot uses a single set of sensors exclusively for this task, e.g., laser range finders or cameras. However, the problem of mutual localization among different robots using different means of localization has remained an open challenge. Nevertheless, this problem is essential for any fail-safe cooperation among multiple robots without the need for a designated supervising base station. In Sec. 3.3, we introduced a precise algorithm to localize a camera equipped robot within a dense map created by another agent carrying, e.g., an RGBD sensor. However, in the future, we plan to extend our work and address further sensor combinations. In the following Sec. 4.2.1, we discuss initial results on the localization of a ground robot equipped with a laser range finder within a dense 3D map. Afterwards, in Sec. 4.2.2, we take a more general approach to map an environment using a heterogeneous set of robots and sensors in an optimal way.

4.2.1. Localization of a Laser Range Finder in a dense 3D map

Up to today, most ground robots still rely on two dimensional laser range finders as their major source of information on their environment. While laser scanners as a sensor are neither a cheap nor particularly light source of information, they provide reliable metric distance measurements to all objects surrounding the robot. This enables simple obstacle avoidance routines but the low amount of perceived information does not allow for efficient localization or mapping algorithms. In fact, even well received literature in the field does not cover the problem of mapping the environment but only address the localization problem within an already provided map [Thrun et al., 2006]. For the full SLAM problem, the freely available *GMapping* framework uses a Rao-Blackwellized particle filter to build up a grid map as the vehicle traverses the environment [Grisetti et al., 2007].

However, current smaller autonomous robots including most areal systems rely on light and versatile cameras to map the environment. Especially for the emerging number of projects addressing air–ground collaboration, e.g. [EU Collaborative Project ICT-600958; Forster et al., 2013], a localization of the ground robot within the map created by a flying camera would be highly desirable. However, to the best of our knowledge, the problem of localizing a laser range finder within a map built by a different sensor has not been addressed so far.

Problem Description

In this project, we plan to solve the following problem: given a sufficiently large dense 3D map, localize a 2D laser scanner within this map. The required map could be obtained from either RGBD 'Kinect like' sensors, dense stereo systems, or scale aware dense monocular systems. To facilitate this six dimensional problem, we assume the availability of orientation estimates for all sensors involved. Thus, the orientation of the laser scan plane relative to the dense map is

known and reduces the problem to four dimensions: the 3D location and the yaw orientation of the laser scanner. Using orientation estimates from magnetometers, the problem could even be reduced to three Cartesian dimensions.

Proposed Methodology

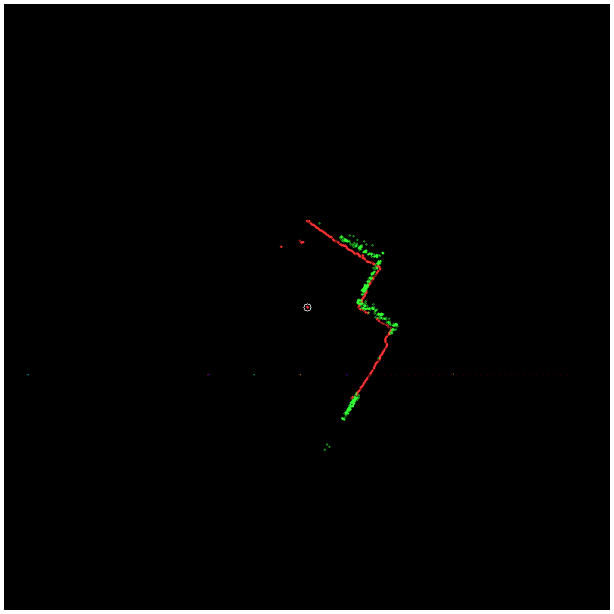
Similar to our proposed solution for a precise localization of a camera within a dense map, we propose a particle filter combined with an analysis-by-synthesis approach to address the described problem. Depending on the availability of initial location priors, particles representing potential locations of the laser range finder are distributed over the three or four dimensional search space. The location of all particles is then updated by comparing a synthesized laser scan for each particle with the actual scan obtained from the real laser range finder. After an outlier removal step, a current efficient implementation of the *Iterative Closest Point (ICP)* algorithm such as, e.g., [Pomerleau et al., 2013] could be used to update the location of the particles to minimize the difference between the observed and the simulated laser scan points. Using a sufficiently dense initial particle set, the predicted location of the laser range finder should eventually converge into the real location.

The key component of the proposed approach is the synthesis of the predicted laser scan from a given pose within the dense 3D map. Since this simulation has to be carried out multiple times per second for all > 1000 particles, this algorithm needs to be highly time efficient. To this end, we propose the following steps:

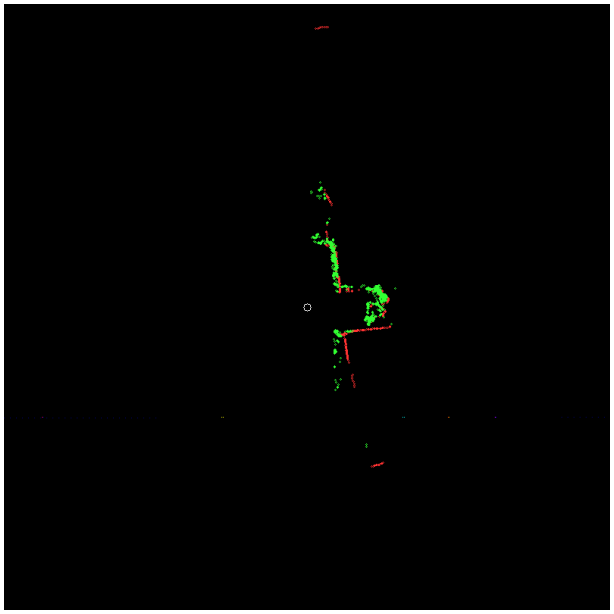
1. A virtual plane is placed in the dense map. This plane has to include the given location of the particle and have the known orientation of the laser range finder with respect to the 3D map.
2. All points of the dense map for which the distance to the plane along its normal is smaller than a threshold are projected onto the plane. This threshold could additionally depend on the distance to the particle to reflect orientation uncertainties.
3. All points on the plane that are not actually visible to the scanner are removed. To avoid a quadratic computational complexity with the number of points on the plane, all points could be first clustered into angle segments (corresponding to the angular resolution of the scanner) and only the closest point within each segment is rendered into the final scan. Correctly implemented, this procedure even achieves linear complexity with the number of points on the plane.

Initial Results

For initial tests, we rigidly connected a monocular camera and a Hokuyo URG-04LX-UG01 laser scanner and localized the camera within a previously created sparse map of visual SIFT features [Lowe, 2004]. From this initial estimate and the known relative pose of the laser scanner with respect to the camera, we predicted the laser scan within the sparse map. The resulting synthesized scans before any possible ICP based pose optimization can be seen in Fig. 4.1 together with the real scan for two different environments. Although the map has been sparse for this initial test which resulted in blind spots for uniform areas without rich textures and therefore only few SIFT features (see, e.g., the lower part of Fig. 4.1a), the quality of the



(a)



(b)

Figure 4.1.: Initial results for the simulation of a laser scan in comparison to a scan obtained from a real laser range finder. The white circle in the center denotes the location of the laser scanner. Red dots indicate intersections of laser scan rays with objects while green dots represent expected scan points as predicted based on a metric sparse 3D map. (a) Environment with wall, floor, and cardboard box. (b) Environment with a gap between two tables.

synthesized laser scans appears to be suitable for further pose optimization steps based on the ICP algorithm.

4.2.2. Multimodal Mapping

Robots are usually equipped with a minimal set of sensors that suffices the need of their application best. Furthermore, most robots are still intended for isolated use, potentially even in a constrained environment without other robots, humans, or even stationary obstacles. Therefore, a wide range of sensors is used on robots in general, but most localization or SLAM algorithms are highly specific for one particular type of sensor. Naturally, this complicates the localization problem among a set of heterogeneous robots using a variety of different sensors. Equally, since they are unable to localize themselves with respect to each other, the robots cannot contribute to one common map. From this observation, the problem of multimodal mapping arises.

Problem Description

Consider the problem of a swarm of heterogeneous robots that are supposed to collaboratively map a certain environment using different sensors. The resulting map should unite the measurements and conclusions from all sensors involved. Similarly, the data structure should allow to combine the findings from multiple robots and sensors to draw new conclusions on the environment. Therefore, observations of the same physical point in the environment by different sensors need to be clustered into one joint observation once the uncertainty has converged.

Discussion

The problem could be addressed by providing pairwise localizations techniques among all different sensors of interest. In this sense, the approach to localize a camera with respect to a dense map created from, e.g., an RGBD sensor, stereo, or monocular dense tracking in Sec. 3.3 and the considerations made in Sec. 4.2.1 for laser range finders and dense maps provide partial solutions. However, the different physical principles underlying the sensors require adequate considerations concerning the different noise and error characteristics. Providing individual solutions for all pairs of senors might not provide the necessary flexibility to dynamically integrate all collected information in one common framework.

Therefore, an optimization in the least-square sense similar to bundle-adjustment could be used to find a map that minimizes the uncertainties of all sensor observations. This process would have to be iterative since converged uncertainties could create new links between observations. Regions with high uncertainties for a particular sensor could be revisited by a robot with the appropriate sensor in a multi-robot multi-sensor active mapping approach.

Clearly, this project would be of high complexity but might start an entirely new field of robotic research with new applications that allow for a higher level of implicit autonomy.

Appendix

A. Hardware Description

A.1. Quadrotor UAV

The quadrotor UAVs from Mikrokopter[1] used in all experiments measure 0.55 m in diameter and typically have a total mass of 1.3 kg in flight with all sensors attached. For some experiments, extended legs were used to guarantee a focused camera image even when landed. Each motor produces a maximum thrust of 5 N which provides the quadrotor with a maximum vertical acceleration of 5.2 $\mathrm{m/s^2}$. All computational components are centered in the middle of the UAV. A low-level flight controller is implemented on either an Atmega 644p or 1284p microcontroller. The low-level control loop regulates the attitude (roll ϕ, pitch θ, and yaw ψ) and the thrust of the quadrotor by means of a PID controller with a frequency of 480 Hz with respect to the data provided by an integrated IMU. Please note that this IMU is otherwise not used for any sensor fusion purposes (compare Sec. A.3). The desired values for ϕ, θ, and ψ are read with a frequency of ~ 100 Hz from the serial port.

A.2. Cameras

In the following, we list all cameras that have been used throughout our experiments. For all cameras, the calibration has been done before the experiments using the camera calibration toolbox for Matlab[2]. To save computational time, for each camera, a lookup table holding the bearing angles for all pixels in the image has been numerically pre-calculated. The values were linearly interpolated between these precomputed values to gain an approximate sub-pixel accuracy.

A.2.1. BlueFox Camera

For most experiments, a MatrixVision[3] BlueFox monochrome camera was used. A 140° lens projected a field-of-view of approximately 100° onto the 752×480 pixel $1/3''$ global shutter sensor. The camera is capable to deliver up to 90 images per second.

A.2.2. Playstation Eye 3 Camera

For the experiments discussed in Sec. 3.2.3, we decided to use a modified low cost Sony Playstation Eye 3[4] color camera, originally designed for the consumer market. The camera is able to provide 60 frames of 640×480 pixel per second. With the casing removed, the camera weights

[1] http://www.mikrokopter.de
[2] http://www.vision.caltech.edu/bouguetj/calib_doc/
[3] http://www.matrix-vision.com
[4] http://www.playstation.com/

25 g including the lens. To increase the FOV, we removed the original lens and attached a default M12 lens holder together with a filter for infrared light. For most experiments, we used a 140° lens which projects an image of $90° \times 65°$ onto the $1/4''$ sensor. The camera in its final configuration can be seen in Fig. 3.2b. Black tape was used to shield the sensor from scattered light.

A.3. Inertia Measurement Unit

For most experiments, the quadrotor was equipped with an additional 3DM-GX3-25 IMU from MicroStrain[5] to provide the measurements of the specific acceleration, angular velocity, magnetic field vector, temperature, and gravitational vector at 200 Hz. The gravitational vector was estimated internally within the IMU via fusion of the measurements from the accelerometer, rate gyroscopes, temperature sensor, and a 3D magnetometer (see Sec. 2.5.6 for an analysis of this orientation estimate).

A.4. Single Core Processing Board

For all onboard computations, we have first used a small Intel Atom QSeven board which was mounted underneath the quadrotor (Figure 3.2c3). The CPU operated with 1,66 GHz and used 1 GB of RAM. The Linux operating system loaded from a 8 GB flash ROM chip and could access additional storage on an external SD-Card. A WiFi adapter as well as camera and IMU were attached via default USB plugs. A GPU was not available.

A.5. Dual Core Processing Board

For all later work, the onboard processing was carried out on a small PC with an Intel Atom 1.8 GHz dual core processing unit and 4 GB of memory. The Linux operating system was stored on a fast 32 GB Compact Flash card. Again, all sensors were connected by means of USB while a WiFi card occupied the internal PCI-Express port. This PC did not feature a GPU.

A.6. Software

For the control of the quadrotors in all experiments, we make use of our free and open TeleKyb framework [Grabe et al., 2013b]. All image processing steps rely on methods from the OpenCV[6] library. Inter process communication between the many different parts of our software is based on the Robot Operating System (ROS)[7].

[5] http://www.microstrain.com/
[6] http://opencv.org/
[7] http://www.ros.org/

B. Bibliography

M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart. Path Planning for Motion Dependent State Estimation on Micro Aerial Vehicles. In *Proceedings of the International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013.

K. Alexis, G. Nikolakopoulos, A. Tzes, and L. Dritsas. Coordination of helicopter uavs for aerial forest-fire surveillance. In *Applications of Intelligent Control to Engineering Systems*, volume 39 of *Intelligent Systems, Control and Automation: Science and Engineering*, pages 169–193. Springer Netherlands, 2009.

S. Baker and I. Matthews. Equivalence and Efficiency of Image Alignment Algorithms. In *Proceedings of the Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1090–1097, Kauai, HI, USA, 2001.

M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart. Vision Based MAV Navigation in Unknown and Unstructured Environments. In *Proceedings of the International Conference on Robotics and Automation*, pages 21–28, Anchorage, AK, USA, 2010.

J.-Y. Bouguet. Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm. *In Practice*, 1(2):1–9, 1999.

F. Chaumette and S. Hutchinson. Visual Servo Control, Part I. Basic Approaches. *Robotics & Automation Magazine*, 13(4):82–90, 2006.

J. Civera, A. J. Davison, and J. Montiel. Inverse Depth Parametrization for Monocular SLAM. *Transactions on Robotics*, 24(5):932–945, 2008.

A. K. Das, R. Fierro, V. Kumar, P. Ostrowski, J. Spletzer, and C. J. Taylor. A vision-based formation control framework. *IEEE Transactions on Robotics and Automation*, 18(5):813–825, 2002.

A. De Luca, G. Oriolo, and P. Robuffo Giordano. Feature Depth Observation for Image-based Visual Servoing: Theory and Experiments. *International Journal of Robotics Research*, 27 (10):1093–1116, 2008.

N. Engelhard, F. Endres, J. Hess, J. Sturm, and W. Burgard. Real-time 3D visual SLAM with a hand-held RGB-D camera. In *Workshop Proceedings of the European Robotics Forum*, Vasteras, Sweden, 2011.

EU Collaborative Project ICT-248669. AIRobots. `www.airobots.eu`.

EU Collaborative Project ICT-287617. ARCAS. `www.arcas-project.eu`.

EU Collaborative Project ICT-600958. Sherpa. `www.sherpa-project.eu/sherpa`.

A. Eudes, P. Morin, R. Mahony, and T. Hamel. Visuo-inertial fusion for homography-based filtering and estimation. In *Proceedings of the International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 2013.

J. Fink, N. Michael, S. Kim, and V. Kumar. Planning and control for cooperative manipulation and transportation with aerial robots. *International Journal of Robotics Research*, 30(3): 324–334, 2010.

C. Forster, M. Pizzoli, and D. Scaramuzza. Air-Ground Localization and Map Augmentation Using Monocular Dense Reconstruction. In *Proceedings of the International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 2013.

A. Franchi, C. Masone, H. H. Bülthoff, and P. Robuffo Giordano. Bilateral Teleoperation of Multiple UAVs with Decentralized Bearing-only Formation Control. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 2215–2222, San Francisco, CA, USA, 2011.

A. Franchi, C. Masone, V. Grabe, M. Ryll, H. H. Bülthoff, and P. Robuffo Giordano. Modeling and Control of UAV Bearing Formations with Bilateral High-level Steering. *The International Journal of Robotics Research*, 31(12):1504–1525, 2012.

V. Grabe and D. Scaramuzza. Global Localization of a Monocular Camera in RGB 3D Maps through Photometric-Error Minimization. Technical report, 2014.

V. Grabe, H. H. Bülthoff, and P. Robuffo Giordano. On-board Velocity Estimation and Closed-loop Control of a Quadrotor UAV based on Optical Flow. In *Proceedings of the International Conference on Robotics and Automation*, pages 491–497, St. Paul, MN, USA, 2012a.

V. Grabe, H. H. Bülthoff, and P. Robuffo Giordano. Robust Optical-Flow Based Self-Motion Estimation for a Quadrotor UAV. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 2153–2159, Vilamoura, Portugal, 2012b.

V. Grabe, H. H. Bülthoff, and P. Robuffo Giordano. A Comparison of Scale Estimation Schemes for a Quadrotor UAV based on Optical Flow and IMU Measurements. In *Proceedings of the International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 2013a.

V. Grabe, M. Riedel, H. H. Bülthoff, P. Robuffo Giordano, and A. Franchi. The TeleKyb Framework for a Modular and Extendible ROS-based Quadrotor Control. In *Proceedings of the European Conference on Mobile Robots*, Barcelona, Spain, 2013b.

V. Grabe, D. Scaramuzza, and P. Robuffo Giordano. Nonlinear Ego-Motion Estimation from Optical Flow for Online Control of a Quadrotor UAV. *International Journal of Robotics Research (under review)*, 2014.

G. Grisetti, C. Stachniss, and W. Burgard. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.

P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments. *The International Journal of Robotics Research*, 31(5):647–663, 2012.

B. Herissé, T. Hamel, R. Mahony, and F.-X. Russotto. Landing a VTOL Unmanned Aerial Vehicle on a Moving Platform Using Optical Flow. *Transactions on Robotics*, 28(1):77–89, 2012.

D. Honegger, L. Meier, P. Tanskanen, and M. Pollefeys. An Open Source and Open Hardware Embedded Metric Optical Flow CMOS Camera for Indoor and Outdoor Applications. In *Proceedings of the International Conference on Robotics and Automation*, pages 1736–1741, Karlsruhe, Germany, 2013.

A. Howard, L. E. Parker, and G. S. Sukhatme. Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. *International Journal of Robotics Research*, 25(5-6):431–447, 2006.

G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *Proceedings of the International Symposium on Mixed and Augmented Reality*, pages 225–234, Nara, Japan, 2007.

L. Kneip, A. Martinelli, S. Weiss, D. Scaramuzza, and R. Siegwart. Closed-Form Solution for Absolute Scale Velocity Determination Combining Inertial Measurements and a Single Feature Correspondence. In *Proceedings of the International Conference on Robotics and Automation*, pages 4546–4553, Shanghai, China, 2011a.

L. Kneip, D. Scaramuzza, and R. Siegwart. A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2969–2976, Colorado Springs, CO, USA, 2011b.

M. Li and A. I. Mourikis. High-precision, consistent EKF-based visual-inertial odometry. *International Journal of Robotics Research*, 32(6):690–711, 2013.

J. Lobo and J. Dias. Relative Pose Calibration Between Visual and Inertial Sensors. *The International Journal of Robotics Research*, 26(6):561–575, 2007.

S. Lovegrove. *Parametric Dense Visual SLAM*. PhD thesis, Imperial College London, 2011.

D. G. Lowe. Object Recognition from Local Scale-Invariant Features. In *Proceedings of the International Conference on Computer Vision*, pages 1150–1157 vol.2, Vancouver, BC, Canada, 1999.

D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, BC, Canada, 1981.

Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision*. Springer, 2004.

R. Mahony, T. Hamel, and J.-M. Pflimlin. Nonlinear Complementary Filters on the Special Orthogonal Group. *Transactions on Automatic Control*, 53(5):1203–1218, 2008.

R. Marino and P. Tomei. *Nonlinear Control Design: Geometric, Adaptive and Robust.* Prentice Hall, 1995.

A. Martinelli. Vision and IMU Data Fusion: Closed-Form Solutions for Attitude, Speed, Absolute Scale, and Bias Determination. *Transactions on Robotics*, 28(1):44–60, 2012.

D. Mellinger, N. Michael, and V. Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research*, 31(5):664–674, 2012.

N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, K. Ohno, E. Takeuchi, and S. Tadokoro. Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, 29(5):832–841, 2012.

N. Moshtagh, N. Michael, A. Jadbabaie, and K. Daniilidis. Vision-based, distributed control laws for motion coordination of nonholonomic robots. *IEEE Transactions on Robotics*, 25(4): 851–860, 2009.

M. Müller, S. Lupashin, and R. D'Andrea. Quadrocopter ball juggling. In *International Conference on Intelligent Robots and Systems*, pages 5113–5120, San Francisco, CA, USA, Sept. 2011.

R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *International Symposium on Mixed and Augmented Reality*, pages 127–136, Basel, Switzerland, 2011a.

R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *International Conference on Computer Vision*, pages 2320–2327, Barcelona, Spain, 2011b.

G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart. Fusion of IMU and Vision for Absolute Scale Estimation in Monocular SLAM. *Journal of Intelligent & Robotic Systems*, 61(1-4): 287–299, 2011.

R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

S. Omari and G. Ducard. Metric Visual-Inertial Navigation System Using Single Optical Flow Feature. In *Proceedings of the European Control Conference*, pages 1310–1316, Zurich, Switzerland, 2013.

F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat. Comparing ICP variants on real-world data sets. *Autonomous Robots*, 34(3):133–148, 2013.

R. Ritz, M. W. Müller, M. Hehn, and R. D'Andrea. Cooperative quadrocopter ball throwing and catching. In *International Conference on Intelligent Robots and Systems*, pages 4972–4978, Vilamoura, Portugal, 2012.

P. Robuffo Giordano, A. De Luca, and G. Oriolo. 3D Structure Identification from Image Moments. In *Proceedings of the International Conference on Robotics and Automation*, pages 93–100, Pasadena, CA, USA, 2008.

E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *Proceedings of the International Conference on Computer Vision*, pages 1508–1515, Beijing, China, 2005.

D. Scaramuzza and F. Fraundorfer. Visual Odometry: Part I - The First 30 Years and Fundamentals. *IEEE Robotics and Automation Magazine*, 18(4):80–92, 2011.

D. Scaramuzza, M. C. Achtelik, L. Doitsidis, F. Fraundorfer, E. B. Kosmatopoulos, A. Martinelli, M. W. Achtelik, M. Chli, S. A. Chatzichristofis, L. Kneip, D. Gurdan, L. Heng, G. H. Lee, S. Lynen, L. Meier, M. Pollefeys, R. Siegwart, J. C. Stumpf, P. Tanskanen, C. Troiani, and S. Weiss. Vision-Controlled Micro Flying Robots: from System Design to Autonomous Navigation and Mapping in GPS-denied Environments. *IEEE Robotics and Automation Magazine*, 2014.

S. Shen, N. Michael, and V. Kumar. A stochastic differential equation-based exploration algorithm for autonomous indoor 3d exploration with a micro-aerial vehicle. *The International Journal of Robotics Research*, 31(12):1431–1444, 2012.

J. Shi and C. Tomasi. Good features to track. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, number June, pages 593–600, Seattle, WA, USA, 1994.

R. Spica and P. Robuffo Giordano. A Framework for Active Estimation: Application to Structure from Motion. In *Proceedings of the Conference on Decision and Control*, Florence, Italy, 2013.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2006.

S. Weiss, M. Achtelik, L. Kneip, D. Scaramuzza, and R. Siegwart. Intuitive 3D Maps for MAV Terrain Exploration and Obstacle Avoidance. *Journal of Intelligent & Robotic Systems*, 61: 473–493, 2011.

S. Weiss, M. W. Achtelik, M. Chli, and R. Siegwart. Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV. In *Proceedings of the International Conference on Robotics and Automation*, pages 31–38, Saint Paul, MN, USA, 2012a.

S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart. Real-time Onboard Visual-Inertial State Estimation and Self-Calibration of MAVs in Unknown Environments. In *Proceedings of the International Conference on Robotics and Automation*, pages 957–964, Saint Paul, MN, USA, 2012b.

S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart. Monocular Vision for Long-term Micro Aerial Vehicle State Estimation: A Compendium. *Journal of Field Robotics*, 30(5):803–831, 2013a.

S. Weiss, R. Brockers, and L. Matthies. 4DoF Drift Free Navigation Using Inertial Cues and Optical Flow. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 4180–4186, Tokyo, Japan, 2013b.

J. Willmann, F. Augugliaro, T. Cadalbert, R. D'Andrea, F. Gramazio, and M. Kohler. Aerial Robotic Construction Towards a New Field of Architectural Research. *International Journal of Architectural Computing*, 10(03):439–460, 2012.

C. Selbstständigkeitserklärung

Ich, Volker Grabe, erkläre hiermit, dass ich die zur Promotion eingereichte Arbeit mit dem Titel *Towards Robust Visual-Controlled Flight of Single and Multiple UAVs in GPS-Denied Indoor Environments* selbständig verfasst (soweit nicht anderweitig durch den Verweis auf zugrunde liegende Veröffentlichungen gekennzeichnet), nur die angegebenen Quellen und Hilfsmittel benutzt und wörtlich oder inhaltlich übernommene Stellen als solche gekennzeichnet habe. Ich erkläre, dass die Richtlinien zur Sicherung guter wissenschaftlicher Praxis der Universität Tübingen (Beschluss des Senats vom 25.5.2000) beachtet wurden.

Teile dieser Arbeit wurden in der vorliegenden Form im Zuge wissenschaftlicher Publikationen veröffentlicht und enthalten zu einem gewissen Anteil Überlegungen, Formulierungen und Ergebnisse der entsprechend ausgewiesenen Ko-Autoren. Die zugrunde liegenden Publikationen mit einer Auflistung der beteiligten Autoren sind in den jeweiligen Kapitelübersichten zitiert. Kleinere Beiträge weiterer Personen sind zudem in den entsprechenden Acknowledgments aufgeführt.

Ich versichere an Eides statt, dass diese Angaben wahr sind und dass ich nichts verschwiegen habe. Mir ist bekannt, dass die falsche Abgabe einer Versicherung an Eides statt mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft wird.

Diese Promotionsarbeit wurde in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt.

Ort, Datum Unterschrift