Lehrstuhl für Steuerungs- und Regelungstechnik
Technische Universität München
Univ.-Prof. Dr.-Ing./Univ. Tokio Martin Buss

Control and Bio-Inspired Motion Estimation of a Vision-Guided Quadrotor

Tianguang Zhang

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. sc. S. Chakraborty

Prüfer der Dissertation:

1. TUM Junior Fellow Dr.-Ing. K. Kühnlenz
2. Univ.-Prof. Dr.-Ing. D. Burschka

Die Dissertation wurde am 16.06.2010 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 15.11.2010 angenommen.

Foreword

This thesis summarizes my three-year research work carried out at the Institute of Automatic Control Engineering (LSR) of the Technische Universität München from 2007 to 2010.

First of all, I would like to express my profound gratitude towards my doctoral advisor, my "Doktorvater", Dr.-Ing. Kolja Kühnlenz, who always supported me with his immense experience and invaluable advice, took the time for all my questions, and made me believe in myself. I sincerely thank Prof. Dr.-Ing./Univ. Tokio Martin Buss, who gave me the opportunity to conduct research in an inspiring working environment, for his encouragement and trust.

I would like to thank Prof. Alexander Borst from the Max-Planck-Institute of Neurobiology in Martinsried for his guidance in neurobiology and valuable discussions. I would also like to thank all the students who contributed to this thesis: Markus Achtelik, Wolfgang Bremer, Martin Gradzki, Ye Kang, Wei Li, Xiaodong Liu, Andereas Plafka, Haiyan Wu, Lei Ying, Lei Zhang, and Ke Zou for their extraordinary assistance and efforts. Special thanks go to my office colleague Micheal Scheint for his friendship and encouragement, his patience with my difficultly understandable German, and helping me know more about Germany and Munich. Great thanks also go to the other ACE team members – Andrea Bauer, Klaas Klasing, Georgios Lidoris, Quirin Mühlbauer, Florian Rohrmüller, Stefan Sosnowski, Dirk Wollherr, and Tingting Xu – for their fruitful discussions and immense assistance in all the phases of my work. I greatly appreciate the technical support from my colleagues Robert Geng, Josef Gradl, Jens Hölldampf, Wolfgang Jaschik, Horst Kubick, Thomas Lowitz, Thomas Schauß and Tobias Stoeber.

Finally, I would like to thank my wife Tingting and my parents for their unconditional love and encouragement.

Munich, June 2010. Tianguang Zhang

to Tingting

...

Abstract

The development of unmanned aerial vehicles has in recent years become a focus of active research, since they can extend the operating capability in a variety of areas such as military, industrial, and civilian domains. This work aims to establish a heterogeneous air-ground multi-robot system consisting of a vision-guided quadrotor and a mobile ground robot in order to study various aspects of a flying system.

Quadrotors have advantages such as a simple and robust mechanism as well as holonomic dynamics, which, however, exhibit challenging motion estimation and flight control problems. Technological approaches for accurate and high-frequency pose/motion estimation as well as stable and effective control for the quadrotor and bio-inspired vision strategies for flight control are studied in this thesis.

To achieve highly accurate and high-frequency pose/motion estimation which serves as a precondition for high-performance control, a continuous-discrete extended Kalman filter is applied for multi-sensor multi-rate fusion of visual and inertial information, considering thorough data synchronization. Thereby, switching between marker-based pose estimation and optical-flow-based motion estimation is conducted for different quadrotor positions. Moreover, based on the careful adaptation of various controllers, an integrated nonlinear control design combining integral backstepping and sliding mode controllers is accomplished for a complete flight scenario consisting of take-off, hovering, tracking, and landing on the moving ground robot. Sensor data processing and control algorithms are completely integrated on-board. Furthermore, insect-inspired qualitative motion detection is extended and implemented on a high-performance FPGA platform, while the quantitative motion estimation is proposed and extensively investigated taking flight control performance into account. The substantial advantages of this work are a novel system configuration, significant improvements in quadrotor pose/motion estimation in terms of high accuracy and high sample rate, enhanced control performance in relation to effectiveness and integrity, as well as advanced exploration in insect-inspired flight control. One of the first autonomous quadrotors only using on-board sensors comprising a monocular camera and IMUs is developed and its performance is extensively evaluated in simulations and real-time experiments.

The contributions significantly advance the state of the art in motion estimation and control of a vision-guided autonomous flying system and serve as a signpost for future research.

Zusammenfassung

Die Entwicklung von unbemannten Luftfahrzeugen ist ein wichtiger Schwerpunkt der aktuellen Forschung geworden, weil sie die Fähigkeiten der Menschen in vielen Bereichen wie dem Militär, der Industrie und dem Zivilleben erweitern können. Diese Arbeit befasst sich mit der Entwicklung eines heterogenen Luft-Boden Multi-Roboter-Systems bestehend aus einem sichtgeführten Quadrocopter und einem mobilen radbasierten Roboter.

Ein Quadrocopter besitzt eine einfache und robuste Mechanik sowie eine holonome Dynamik, welche dennoch eine anspruchsvolle Bewegungsschätzung und Flugregelung erfordert. In dieser Arbeit werden technische Ansätze für eine genaue und hochfrequente Lage- und Bewegungsschätzung, eine stabile und effektive Regelung sowie biologisch inspirierte bildbasierte Strategien zur Flugregelung entwickelt.

Um eine hochgenaue und hochfrequente Lage- und Bewegungsschätzung des Quadrocopters zu erzielen, welche eine Voraussetzung für hohe Regelgüte darstellt, wird ein kontinuierlich-diskreter erweiterter Kalman-Filter verwendet, der visuelle und inertiale Daten mit verschiedenen Abtastraten fusioniert. Dazu wird eine sorgfältige Datensynchronisierung durchgeführt. Die auf Markern basierende Lageschätzung und die auf optischem Fluss basierende Bewegungsschätzung werden abhängig von der Position des Quadrocopter schaltend kombiniert. Auf der Grundlage einer sorgfältigen Anpassung unterschiedlicher Regler wird ein integrierter nichtlinearer Reglerentwurf, der einen Integral-Backstepping-Regler und einen Sliding-Mode-Regler kombiniert, für ein vollständiges Szenario vorgeschlagen, das Abheben, Schweben, Verfolgen und Landen des Quadrocopters beinhaltet. Sensordatenverarbeitung und Reglungsalgorithmen sind komplett on-board integriert. Darüber hinaus wird eine von Insekten inspirierte qualitative Bewegungsdetektion erweitert und auf einer leistungsfähigen FPGA Plattform umgesetzt, während die quantitative Bewegungsschätzung präsentiert und unter der Berücksichtigung der Regelgüte ausführlich untersucht wird. Die wesentlichen Beiträge dieser Arbeit sind Methoden zur multisensorischen Bewegungsschätzung, Lageregelung, biologisch inspirierte Ansätze zur bildbasierten Geschwindigkeitsmessung und Kollisionsvermeidung auf Basis von insektenartigem Sehen sowie die Systemintegration. Hierdurch wird eine signifikante Verbesserung der Schätz- und Regelgüte des Quadrocopters in Bezug auf Genauigkeit und Frequenz sowie eine Verringerung des Rechenaufwands durch biologisch motivierte Modelle erreicht. Die Leistungsfähigkeit des autonomen Quadrocopters wird in zahlreichen Simulationen und Echtzeit-Experimenten evaluiert.

Die Beiträge erweitern den Stand der Technik in der Bewegungsschätzung und der Regelung von bildbasierten autonomen Flugsystemen und dienen als Wegweiser für die zukünftige Forschung.

Contents

Notations

Abbreviations

2D	two dimensional
3D	three dimensional
CEA	Centre d'Energie Atomique
CL	camera link
COG	center of gravity
DC	direct current
DOF	degree of freedom
EKF	extended Kalman-filter
EMD	elementary motion detector
FPAA	field programmable analog array
FPGA	field programmable gate array
GPS	global positioning system
HP	high-pass
IBC	integral backstepping controller
IDL	interaction data language
IMU	inertial measurement unit
KF	Kalman filter
LAN	local area network
LED	light emitting diode
LP	low-pass
LQ	linear quadratic
LQR	linear quadratic regulator
LUT	look-up table
LUTI	look-up table based on interpolation
LUTS	look-up table based on step-wise averaging
MAV	micro air vehicle
PD	proportional-derivative
PI	proportional-integral
PID	proportional-integral-derivative
PLK	pyramidal implementation of the lucas-kanade algorithm
RANSAC	random sample consensus
RF	receptive field
RGB	color space consisting of red, green, and blue channels
RMS	root mean square

ROI	region of interest
SMC	sliding mode control
UART	universal asynchronous receiver transmitter
UAV	unmanned aerial vehicle
UDP	user datagram protocol
UGV	unmanned ground vehicle
USB	universal serial bus
VHDL	very high speed integratd circuit hardware description language
VLSI	very-large-scale-integrated
VTOL	vertical take-off and landing
YUV	color space consisting of luminance and two chrominance channels

Conventions

Scalars, Vectors, and Matrices

x or X	scalar
$\boldsymbol{x}$	vector
$\boldsymbol{X}$	matrix

Subscripts and Superscripts

x_{d}	desired value of x
$x_{\max}$	maximum value of x
$x_{\min}$	minimum value of x
$\hat{\boldsymbol{x}}$	a posteriori estimate
$\hat{(\cdot)}^{-}$	a priori estimate
$(\cdot)^{-1}$	inverse
$(\cdot)^{T}$	transposed
$(\cdot)^{*}$	optimal value or idle state
$(\cdot)'$	homogeneous coordinate
$\tilde{(\cdot)}$	ground truth
${}_{x}(\cdot)$	in the frame x

Symbols

General

a_r	linear acceleration of the ground robot in the X_o-direction
$\boldsymbol{a}_r$	linear acceleration of the ground robot in the object frame
a_x, a_y, a_z	linear acceleration measured by the IMUs
e	error
g	gravitational constant
h	quadrotor relative altitude
${}_I x$, ${}_I y$, ${}_I z$, ${}_I \Psi$	quadrotor position and orientation in the inertial frame
K_d	derivative term
K_i	integral term
K_p	proportional term
p, q, r	quadrotor rotational velocity measured by the IMUs
$\boldsymbol{S}_b$	the body frame
$\boldsymbol{S}_c$	the camera frame
$\boldsymbol{S}_i$	the image frame
$\boldsymbol{S}_I$	the inertial frame
$\boldsymbol{S}_o$	the object frame
u, v, w	quadrotor linear velocity in the body frame
v_r	linear velocity of the ground robot in the X_o-direction
$\boldsymbol{v}_r$	linear velocity of the ground robot in the object frame
x, y, z	quadrotor position in the body frame
$\ddot{x}_{\text{trans}}$, $\ddot{x}_{\text{rot}}$	intermediate terms
$\boldsymbol{x}$	system state
$X_{(\cdot)}, Y_{(\cdot)}, Z_{(\cdot)}$	X-, Y-, and Z-axis in a specified frame
$\boldsymbol{z}$	system measurement
Ω_r	orientation of the ground robot in the inertial frame
Φ, Θ, Ψ	roll, pitch, and yaw angles
Φ_{imu}, Θ_{imu}, Ψ_{imu}	angle estimation of IMUs

Multi-Sensory Pose/Motion Estimation

a_1-a_{13}	expression terms
${}^a\boldsymbol{R}_b$	rotational matrix from frame b to frame a
${}^a\boldsymbol{T}_b$	homogeneous transformation matrix from frame b to frame a
$\boldsymbol{a}$	compensated acceleration of the gravity
e_{RMS}	RMS error
e_z	sensor error
$\boldsymbol{E}$	unit matrix
f	focal length
$\boldsymbol{f}$	system equation
$\boldsymbol{F}$	system matrix after linearization
$\boldsymbol{G}$	weigting matrix
$\boldsymbol{H}$	measurement matrix
j	counter
$\boldsymbol{J}$	image Jacobian matrix
k	time step
K	measurement dimension
l	counter
L	number of the last estimations involved in data synchronization
$\boldsymbol{L}$	Kalman-gain
m	total number of pixels on the edge
M	total number of hits
n	dimension of the system state
N	number of time intervalls in T_{imu}
$\boldsymbol{p}_q$	3D position of the quadrotor
$\boldsymbol{p}_r$	3D position of the ground robot
$\boldsymbol{P}$	state error covariance matrix
$\boldsymbol{Q}$	process noise covariance matrix
r	circle radius
$\boldsymbol{r}$	translational vector
$\boldsymbol{R}$	measurement noise covariance matrix
$s(\cdot)$, $c(\cdot)$, $t(\cdot)$	$\sin(\cdot)$, $\cos(\cdot)$, $\tan(\cdot)$
$\boldsymbol{S}_{b^1}$, $\boldsymbol{S}_{b^2}$, $\boldsymbol{S}_{b^3}$	intermediate frames
t	time
t_k	the kth time point
Δt	time intervall
T_{d1}-T_{d8}	time delays
T_e	time label of the IMU angle estimation
T_{imu}	IMU update period
T_m	time label of the current measurement
T_{oe}	time label of the a posteriori estimation
$\mathrm{Trans}(\cdot)$, $\mathrm{Rot}(\cdot)$	translation or rotation transformation
T_v	time label of the arriving vision data
T_x, T_y, T_z	relative translation velocity between camera and object

$\boldsymbol{u}$	motion information input
x_0, y_0	circle center
$\boldsymbol{x}_s$	selected system state
$\ddot{y}_{\text{trans}}$, $\ddot{y}_{\text{rot}}$	intermediate terms
z	acceleration sensor measurement
$\tilde{z}$	the ground truth of acceleration measurement
$\tilde{\boldsymbol{z}}$	the ground truth of measurement
α, β, γ	scalars
χ	point on the landing surface
δ	distance between markers
ϵ	threshold
$\boldsymbol{\nu}$	measurement noise
ρ	scale factor
σ	sensor measurement bias
ω_r	random noise
ω_x, ω_y, ω_z	relative rotation velocity between camera and object
$\boldsymbol{\omega}$	process noise
ζ	point projection in the image plane

Control Design and Implementation

A, B, C	system dynamics matrices
b	thrust factor
$\boldsymbol{e}$	control error
$e(t)$	control error
f_1-f_n	system functions
f_f, f_b, f_l, f_r	forces produced by the forward, backward, left, and right rotors
$\boldsymbol{f}$	force
$\boldsymbol{f}_{\text{unm}}$	unmodeled force
F	thrust
F_{cmd}	thrust command
$F_{\text{unm},z}$	unmodeled force in the Z_I-direction
F_z	thrust in the Z_I-direction
g_1-g_n	system functions
h_0	quadrotor desired hovering altitude
h_l	desired switching height in landing
h_t	desired switching height in take-off
i	counter
J	cost functional
J_x, J_y, J_z	moment of inertia around the x-, y-, and z-axis
$\boldsymbol{J}$	inertial matrix
K	negative scalar
$\boldsymbol{K}$	feedback matrix
l	quadrotor axis length
$\boldsymbol{L}$	angular momentum
m	quadrotor mass
M, M_1, M_2	positive scalars
n	dimension of system state
p	dimension of control input
$\boldsymbol{P}$	positive definite matrix
q	dimension of system measurement
$\boldsymbol{Q}$	semi-positive definite matrix
$\boldsymbol{R}$	positive definite matrix
s	switching line for SMC
$\boldsymbol{S}$	semi-positive definite matrix
t	time
t_d, t_{d1}, t_{d2}	time delays
T_t	take-off time
u_x, u_y, u_z, u_Ψ	intermediate controller outputs
$\boldsymbol{u}$	controller output
$\boldsymbol{v}$	quadrotor translational motion
V	Lyapunov function
$\tilde{\boldsymbol{x}}_{i-1}$	subsystem state
α	positive scalar

α_1-α_7	positive scalars
β	pseudo input
χ_1-χ_7	integrals of tracking errors
κ	time window for prediction
κ_0-κ_2	scalars
Φ_{cmd}, Θ_{cmd}, $\dot{\Psi}_{\text{cmd}}$	control commands
σ	constant
τ_f, τ_b, τ_l, τ_r	torques of the front, back, left, or right motor
τ_Φ	rolling torque
τ_Θ	pitching torque
τ_Ψ	yawing torque
$\boldsymbol{\tau}$	torque
$\boldsymbol{\omega}$	angular velocity in the body frame
Ω	rotor speed
ς	x, y, z, Ψ
ζ_1-ζ_7	new system states

Insect-Inspired Motion Detection and Estimation for Flight Control

$A1(t)$, $A2(t)$	input signals of photoreceptors
$B1(t)$, $B2(t)$	filtered signals of photoreceptors
C	amplitude of a sine-grating signal
f	LP filter function
f_s	spatial frequency
f_t	temporal frequency
$iDATA$	8-bit image data flow
$iFVAL$	frame signal
$iLVAL$	line signal
$iDVAL$	data signal
I	image signal
log	logarithmic transformation
M	multiplication
P_C, P_H, P_V	photoreceptors
P	power spectral density
R_{EMD}	EMD response
R	RF response
R_{th}	threshold for RF response
R_V, R_H	vertical or horizontal response
t	time
v	velocity
x	filter input
y	filter output
z	intermediate filter output
$\Delta\phi$	phase distance of two receptors
ω	angular velocity
θ	motion direction
τ_H	time constant of high-pass filter
τ_L	time constant of low-pass filter

List of Figures

List of Tables

1 Introduction

The development of *Unmanned Aerial Vehicles* (UAVs) has in recent years become a focus of active research, since they can extend the operating capability in a variety of areas such as military, industrial, and civilian domains. Above all, *Micro Air Vehicles* (MAVs) have gained a great interest in the robotics domain because of their small size and possible applications in indoor, complex, everyday environments. In order to study various aspects of a flying system, a quadrotor is chosen as the platform used in this work due to its robust mechanism and holonomic dynamics, which exhibit challenging motion estimation and control problems for autonomous flight.

Since vision is one of the most powerful tools for information acquisition and is widely used in most biological organisms such as humans and animals for autonomous navigation, vision-guided flight and navigation are considered able to overcome the aforementioned challenges. Moreover, insect-vision-inspired neurobiological models in particular are expected in the further development of MAVs.

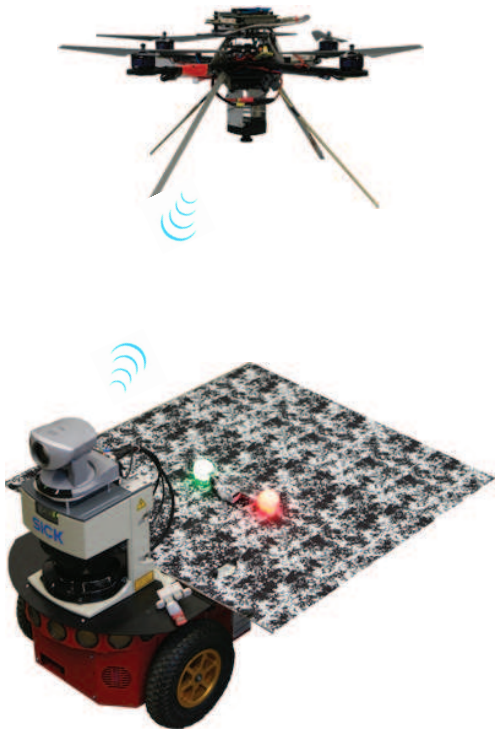

Fig. 1.1: An air-ground multi-robot system containing a mini-quadrotor and a mobile ground robot with two active markers.

A heterogeneous air-ground multi-robot system is developed as a test-bed, containing a quadrotor and a wheeled ground robot (see Fig. 1.1). This thesis aims to establish a

vision-guided flying system containing accurate pose/motion estimation as well as stable and effective control, such that this vision-guided MAV can conduct a complete flight scenario including take-off, hovering, tracking, and landing stably and safely with the help of the ground robot. Furthermore, bio-inspired vision strategies for flight control are investigated as an attractive alternative to traditional paradigms. The main challenges faced in developing the envisioned flying system are summarized below.

1.1 Challenges

The important issues in developing an autonomous vision-guided flying system consist of pose/motion estimation based on on-board sensors, control design of the non-linear dynamic system, and performance improvement in terms of efficiency considering the limitations of MAVs. The former two issues remain in the traditional design and control of a flying system, while a biologically inspired technology is considered for the latter one. The main challenges for an advanced exploration at those points considered in this thesis are summarized in this section.

Accurate Pose/Motion Estimation without External Sensors

A significant and fundamental challenge in developing UAVs is to extract and fuse useful information in a robust manner in order to achieve accurate and real-time pose/motion estimation, which serves as a basis for a stable control design and effective navigation performance. The control of UAVs relies on the knowledge of position, velocity, and orientation. However, the drift of inertial sensors commonly equipped on UAVs leads to errors during time-discrete integration, making steadily accurate estimation of the absolute pose nearly impossible. Aiming at applications without using external sensors, such as *Global Positioning System* (GPS) or external tracking systems, an on-board vision sensor is selected to cooperate with the on-board *Inertial Measurement Units* (IMUs).

Compared to the IMUs, vision sensors have advantages such as accurate pose estimation without propagating errors. However, due to the limited field of view and relatively low sampling rate as well as relatively complex data processing, visual data are not sufficient for pose/motion estimation of a highly dynamic flying system in which high-frequency noise, vibrations, and disturbance occur. A high-frequency and accurate fusion of the IMU data and vision information from an on-board monocular camera without any other additional sensors is still missing. Moreover, to achieve autonomous behavior with less time delay in an unlimited workspace, information processing – which is normally accomplished in a ground station – should be conducted on-board.

Control Issues for an Under-Actuated System with Fast, Non-Linear Dynamic Behavior

Although quadrotors have major advantages in robust mechanical design and holonomic dynamics, stable and effective control is very demanding. First, a quadrotor is an under-actuated system, which means that the number of its actuators is lower than its *Degrees*

Of Freedom (DOFs). Theoretically, the control of such systems is more complex than that of fully actuated systems. Moreover, the non-linear dynamic behavior makes stable control and guidance of a quadrotor challenging due to highly unstable dynamics, high-degree axes coupling, and nonlinear disturbance caused by linearization of the nonlinear dynamics. Furthermore, the time delay strongly influences the control performance due to the very fast dynamics of the quadrotor. In addition, a simplified control structure is desired due to hardware limitations such as small payloads and limited on-board processing power.

Moreover, for a complete flying behavior including take-off, hovering, tracking, and landing on a mobile ground robot, the individual prerequisites and requirements of each flight phase should be considered. An integrated control design is envisioned which has only been partially investigated up to now in the literature.

Limited Payload and On-Board Computational Capacity During Highly Dynamic Self-Motion

For all the MAVs, limitations such as small payloads and limited on-board computational capacity are major problems. Moreover, the highly dynamic self-motion of MAVs also needs a real-time response to the environment. Those challenges require efficient computation and implementation of motion estimation and control algorithms. One possibility is turning to biological models such as flying insects, e.g. Drosophila melanogaster and Calliphora vicina, which possess photoreceptors with high temporal resolution which they use for dynamic visuomotor pose and gaze stabilization, as well as navigation in 6 DOFs.

However, the transfer of neurobiological results to technical systems such as the quadrotor requires the adaptation of models to typical dominant and preferred system motion and suitable hardware and implementation to ensure the effectiveness of the algorithms. Moreover, the biologically inspired motion detection only provides a qualitative description of motion, while 3D motion reconstruction and quantitative estimation are desired to serve as motion feedbacks for technical systems to realize closed-loop control. How the qualitative characteristics of biological modeling can be transferred into normally quantitatively controlled technical systems remains an intriguing open question.

1.2 Main Contributions and Outline of the Thesis

In this thesis, various aspects of the development of an autonomous flying quadrotor facilitated by a mobile ground robot according to the aforementioned challenges are explored. Fig. 1.2 illustrates the main contributions and the outline of this thesis. After the state of the art is surveyed in Chapter 2, traditional quadrotor control problems are investigated first. Chapters 3 and 4 solve the multi-sensory pose/motion estimation and control design problems, respectively. In contrast, Chapter 5 explores the quadrotor motion estimation and control problem from an insect-inspired perspective and brings new insights into the development of flying systems.

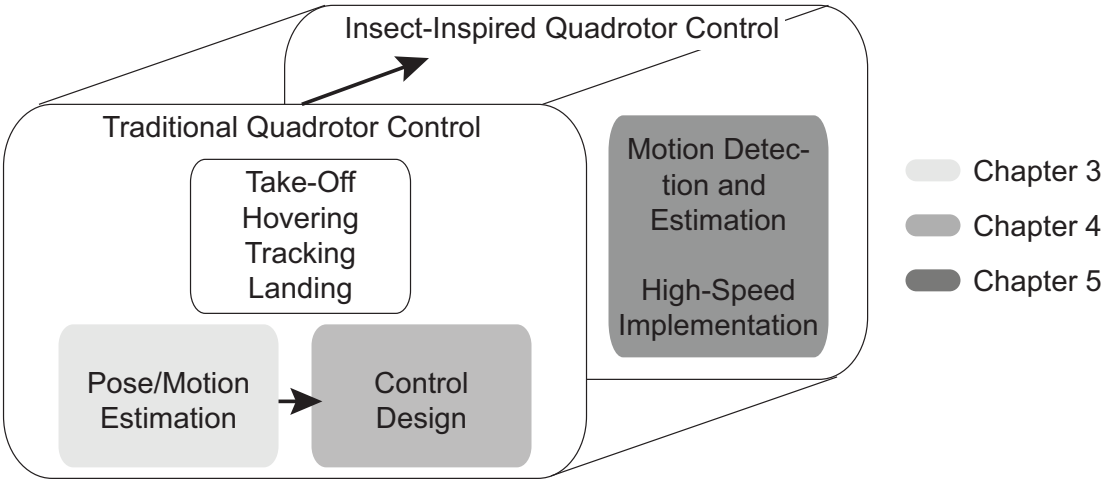

Fig. 1.2: Outline of the thesis.

Multi-Sensory Pose/Motion Estimation

One of the fundamental but challenging problems of flight control is accurate pose and motion estimation of the aerial vehicle. To deal with the drift problem of on-board IMUs and be able to acquire more information about the other agent – the mobile ground robot – in the environment, a monocular camera facing downwards is installed on the quadrotor. Vision sensors have, however, the disadvantages of limited field of view, low sampling rates, and complex image processing, all of which are unfavorable for pose and motion estimation of a highly dynamic system. In Chapter 3, which seeks to obtain accurate pose/motion estimation of a highly dynamic quadrotor relying only on on-board sensors, a thoroughly designed high-frequency fusion of the inertial data and the vision data is accomplished. Based on multi-modal sensor information, a continuous-discrete *Extended Kalman Filter* (EKF) is applied for the multi-sensory multi-rate data fusion, where the high-frequency IMU data drive the process model and the low-rate vision data correct the estimation. Data synchronization is completely conducted based on the accurately measured time delay. Moreover, switching between marker-based pose estimation and optical-flow-based motion estimation is implemented for different situations. Various real-time experiments considering the quadrotor tracking the mobile ground robot show that 1) high accuracy and high frequency of the pose/motion estimation in dynamic behaviors are obtained through the multi-sensory multi-rate data fusion; 2) one of the first autonomous quadrotors based on minimal on-board sensors and the complete integration of sensor data processing is achieved.

Effective and Integrated Control Design for a Complete Flight Scenario

Based on the accurate and high-frequency pose and motion estimation in Chapter 3, a stable and effective control design is investigated in Chapter 4. Quadrotors themselves exhibit a challenging control problem due to non-linear, under-actuated, highly unstable dynamics, time delay due to data processing, as well as the limited payload and computational

capacity requiring a simple control structure. Up to now, most state-of-the art works have only considered the control design in simulations, while the other standard works that consider real system implementation use a much simplified system model in limited scenarios. Chapter 4 aims at designing, implementing, and discussing adequate control structures for the quadrotor, to overcome the aforementioned challenges and to improve the quadrotor flying behavior in an integrated manner. The system model used here is more complex, in order to preserve the original interdependency of system states. A *Proportional-Integral-Derivative* (PID) controller, an optimal *Linear Quadratic* (LQ) controller, and non-linear controllers such as backstepping-based controllers and sliding mode controllers are carefully adapted to the quadrotor system and discussed. Based on the simulation results, an integrated control design is accomplished for quadrotor take-off, hovering, tracking, and landing on the mobile ground robot and evaluated in real-time experiments. The main contributions presented in Chapter 4 are 1) the adaptation, implementation, and real-time evaluation of various control strategies based on a complex system model; 2) an integrated control design considering controller combinations proposed and evaluated in the complete flying scenario for the first time.

Insect-Inspired Motion Detection and Estimation for Flight Control

After Chapters 3 and 4 solve the flight control problems in a traditional manner, Chapter 5 focuses on insect-inspired motion detection and estimation as an efficient alternative extension. The fundamental findings in biology and neuroscience show that insects have a small but efficient and sensitive visual system for real-time flight stabilization and control. This would provide an excellent solution for the development of MAV on-board vision, which requires simple and fast computation due to limited payload, restricted computational capacity, and fast self-motion. In Chapter 5, the novel vision and visuomotor behavior models inspired by biological paradigms are extended first, in order to adapt to the typical dominant and preferred motion patterns of the quadrotor. Two new *Receptive Fields* (RFs) for rotation detection are proposed. High-speed implementation using compatible hardware – a *Field Programmable Gate Array* (FPGA) platform – is accomplished to obtain the effectiveness of the neurobiological algorithms. The performance of the implementation is sufficient to deal with a video frame rate of 350 fps for a frame size of 256×256 pixels. The respective motion estimation is established using *Look-Up Tables* (LUTs) and extensively explored considering the influences of specific parameters such as perception difference between flies and cameras, lighting conditions, as well as the spatial frequency and spectrum of input images. Closed-loop control and obstacle avoidance are exploratively studied and show a promising possibility of fully controlling MAVs based on insect-like vision in future work. The twofold contributions in Chapter 5 are: 1) extension of efficient qualitative motion detection and its high-speed implementation on an FPGA platform; 2) explorative investigation of quantitative motion estimation for flight control.

The novel system configuration, significant improvements in quadrotor pose/motion estimation in terms of high accuracy and high frequency, enhanced control performance in relation to effectiveness and integrity, as well as advanced exploration in insect-inspired

flight control contribute to the establishment and further development of an efficiently controlled vision-guided flying system. Evaluations and demonstrations are carried out in various simulations and real-time experiments.

2 State of the Art

To achieve effectively and efficiently controlled flight of a quadrotor, different aspects such as accurate pose/motion estimation as well as a stable and effective control design should be established. Both of them play an essential role in the whole system. Furthermore, considering biological models for quadrotors with a small size and flexible dynamics, bio-inspired vision guidance strategies are also envisioned to improve the system performance or lower computational cost.

In this chapter, various flying platforms, above all, quadrotor helicopters, are introduced in Section 2.1 first. Then, sensor configurations and algorithms for quadrotor pose/motion estimation are reviewed in Section 2.2. After that, control strategies are surveyed in Section 2.3. In Section 2.4, biologically inspired further development of flying systems is described. Finally, a summary discussing the current state of the art is given in Section 2.5.

2.1 Standard Quadrotor Platforms

Currently, *Unmanned Aerial Vehicles* (UAVs) are a major focus of active research, since they can extend the operating capability in a variety of areas. Due to small sizes and flexible dynamics, *Micro Air Vehicles* (MAVs) have also become a popular research field in the robotics domain. Fig. 2.1 shows three different kinds of MAV platforms: the fixed-wing, the rotary-wing, and the flapping-wing platforms. In this thesis, subjects considered are restricted to one kind of MAV: quadrotor helicopters. Quadrotor platforms have been developed both in the commercial domain and in the research domain, introduced in this section.

Fig. 2.1: Various MAV platforms. Left: a fixed-wing MAV – the microflyer from EPFL [130]; middle: a rotary-wing MAV from UMD [59]; right: a flapping-wing MAV – the *Microbat* from AeroVironment [77].

2.1.1 Commercial Platforms

Some well-known commercial quadrotor platforms with different characteristics are summarized in Fig. 2.2.

a) b) c) d) e)

Fig. 2.2: Commercial quadrotor platforms (from left to right): a) Draganflyer X4 from Draganfly Innovations Inc. [5]; b) Hummingbird from Ascending Technologies GmbH [3]; c) md4-200 from Microdrones GmbH [8]; d) Parrot AR Drones from Parrot [2]; e) Mikrokopter *MikroKopter* [6].

The *Draganflyer X4* from Draganfly Innovations Inc. is one of the most commonly used quadrotor platforms in industry, government, and research [42, 94, 123, 128] and has been designed to carry wireless cameras via remote control [5]. The *Hummingbird* quadrotor from Ascending Technologies GmbH is small, lightweight, and developed to work safely in indoor environments [3], and used as the platform in this thesis and other research works [25, 131]. Quadrotors from Microdrones GmbH [8] are made completely of carbon fiber reinforced plastics, for low weight and high robustness. At 2010 International CES Exhibit, the *Parrot AR. Drone* was exhibited. This platform is controlled via iPhone™ or iPod touch® and can detect the user's movements and provide the user with the camera view on the screen in real time [2]. In addition, a self-construction project called *MikroKopter* serves as a quadrotor fan community [6].

2.1.2 Research Projects

Various platforms have also been developed or modified at universities and institutes for research and education. Some recent ones are given in Fig. 2.3.

A cm-scale rotorcraft called *Mesicopter* was developed at Stanford University which flies on its own power and carries sensors for atmospheric research or planetary exploration [85]. In contrast, the *X4-flyer*, developed at the Australian National University [104, 105], is much heavier than most existing experimental quadrotor platforms, weighing 4 kg with a payload of 1 kg. In [20, 21], a commercially available *Roswell Flyer* or *HMX-4 flyer* is used which is now discontinued in the market. A small quadrotor vehicle developed at the *Centre d'Energie Atomique* (CEA) in France is used in [39, 61, 62] for visual servo systems and is capable of stationary and quasi-stationary flight. An outdoor test bed called *STARMAC* was also developed at Stanford University comprising a set of autonomous quadrotor helicopters for testing multi-agent algorithms and control design [72, 73]. In [35], a quadrotor platform named *OS4* was designed. Various control strategies are implemented for *Vertical Take-off and Landing* (VTOL) behavior. Moreover, a quadrotor using minimal sensing for autonomous indoor flight is introduced in [109]. In [118], a quadrotor platform

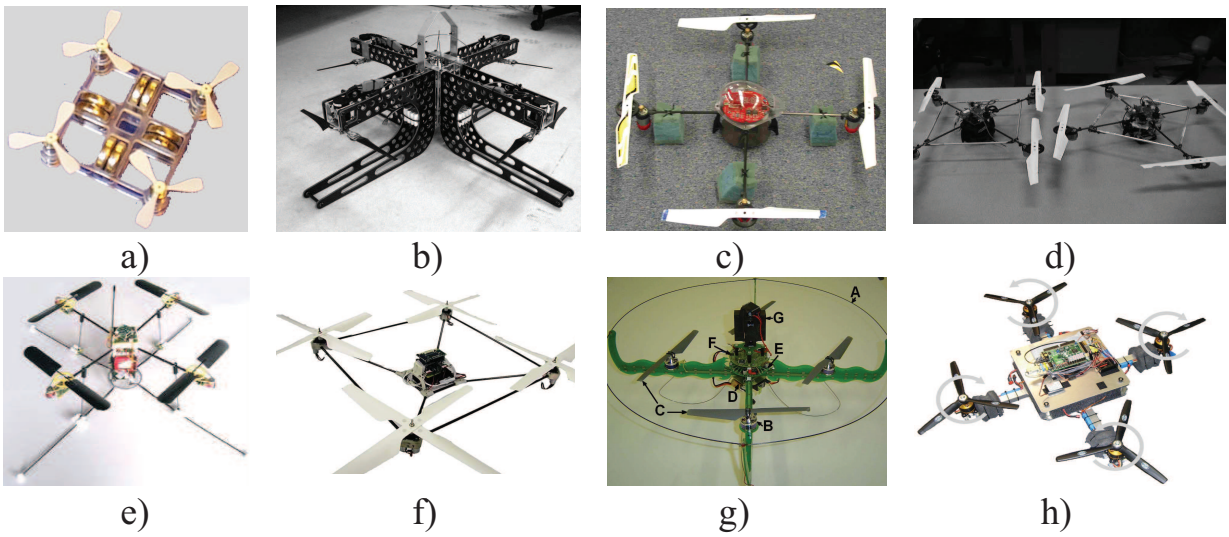

Fig. 2.3: Quadrotor projects for research and education. a) Mesicopter, Kroo et al., Stanford [85]; b) X4-flyer, Pounds et al., ANU [104, 105]; c) HMX-4 flyer, Altug et al., Penn [20, 21]; d) STARMAC, Hoffmann et al., Stanford [72, 73]; e) OS4, S. Bouabdallah, ETH [35]; f) Guenard et al., CEA [39, 61, 62]; g) Roberts et al., EPFL [109]; h) M. J. Stepaniak, OHIO [118].

is capable of lifting a payload of 10 lb and is the largest unmanned quadrotor to fly without tethers.

2.2 Multi-Sensory Pose/Motion Estimation

The fundamental and essential prerequisite for a successful autonomous flying behavior is accurate pose/motion estimation in real time based on sensor information. Aiming at this, various sensor configurations have been taken into account on the one hand; on the other hand, data fusion of multiple sensor modalities has become a promising research focus in MAVs. These two issues are surveyed in this section.

2.2.1 Multiple Sensor Modalities

Up to now, a variety of sensors have been applied to quadrotors for pose and motion estimation.

Inertial Measurement Units (IMUs)

One of the most widely used sensors on aircrafts are on-board IMUs, such as gyroscopes and accelerometers. The control of MAVs during autonomous flight relies on knowledge of variables like position, velocity, and orientation, which can be partly calculated using information provided by on-board inertial sensors. They measure the linear acceleration and angular velocity of the platforms. The pose estimation is then conducted through dead reckoning of the sensor signals.

However, the drift of inertial sensors leads to errors during time-discrete integration, making a steadily accurate estimation of the absolute pose nearly impossible [131]. Therefore, IMUs are commonly not used alone but always cooperate with additional sensors.

Global Positioning System (GPS)

Most works deploy GPS as an external reference to obtain accurate position information of the platforms. It is practical for outdoor applications [72, 74, 92]. However, in the environments where no reliable GPS signal is available, such as indoors, in cluttered urban environments, in battlefields, or on other planets, other sensors should be applied instead.

Vision Systems

Vision is one of the strongest information origins. Despite more complex information processing and transfer, vision has become a focus of the research of MAVs.

- External Vision Systems External vision systems have been applied to estimate platform position/orientation and can achieve a more accurate estimation in indoor or urban environments than GPS. Platforms are commonly equipped with artificial markers [20, 25, 115, 131]. However, autonomous flying without external sensors is the focus of this work. Therefore, external vision systems are not investigated further in this thesis.

- On-Board Vision Systems To achieve a fully autonomous flying behavior, on-board vision systems have been applied.

A novel two-camera method is proposed for estimating the full 6 *Degrees Of Freedom* (DOFs) of a quadrotor in [21]. One camera is mounted on the quadrotor, while the other pan-tilt one is located on the ground. Colored blobs are attached to both the ground camera and the bottom of the quadrotor for tracking. Moreover, these two cameras are set to see each other. A purely vision-based pose estimation is achieved.

In [79], an edge-based visual tracking system using a quadrotor equipped with an on-board forward-looking camera is realized.

In [123], a monocular on-board camera is set to face specified moire patterns to get the relative position and orientation of the quadrotor. However, this application is too specific due to the complex target patterns.

The CEA quadrotor is deployed as a platform for visual servoing in [39]. A camera is mounted on the quadrotor, observing four black planar marks on the ground as visual markers. The image data are transmitted by a wireless analog link to the ground station, which executes the control law.

A vision-based attitude and position estimation for a quadrotor helicopter is implemented in [96], where three dark colored targets are used and the altitude is controlled manually using a joystick. The flying behavior is therefore semi-autonomous.

An optical-flow-based terrain-following behavior of a VTOL UAV is considered in [70], in which an on-board camera provides the optical flow (using the Lucas-Kanade method [89]) when facing the ground assumed planar, on-board IMUs facilitate the extraction of the translational component of the optical flow, and the forward speed of the quadrotor is

assumed to be ensured by an additional sensor. However, the low sampling rate of 15 Hz and large time latencies hold back the system performance to some extent.

In [48], a visual memory containing ordered key images based on natural landmarks (Harris corners) is constructed for visual navigation of a quadrotor equipped with an on-board fisheye camera. A vision-based control law enables the quadrotor to successively reach visual waypoints during its navigation.

One of the first autonomous flying systems is established in [41], in which a vision-guided quadrotor using a single on-board camera is capable of take-off, hovering, and tracking a planar artificial landmark. The information processing is completely on-board.

Some works also use stereo cameras to provide the position of the UAVs based on stereo vision [19, 23, 78].

Other Commonly Used Sensors

Some other sensors have also been applied as additional sensors: laser range finders for indoor localization and obstacle avoidance [60, 69], infrared range sensors for wall following [91] or collision avoidance [109], ultrasonic sensors for measuring vehicle altitude [35, 109] and obstacle avoidance [35], compasses for measuring the yaw angle of the platform [128], and pressure sensors for altitude estimation [3].

However, due to the limited payload, only lightweight micro sensors can be applied. Furthermore, due to the working principles, the measurement ranges of lasers, infrared sensors, and sonars are limited. As a result, they can only be applied in a specified space with structure.

2.2.2 Multi-Sensory Data Processing

Strong evidence from biological systems indicates that multiple sensory modalities, e.g. vision, touch, and balance, are commonly used to guide the movement. Taking insects as an example, it is critically important to fuse the vision and inertial sensing for localization and motion estimation. These sensors are able to overcome the limitations and deficiencies of each other. Digital camera chips and IMUs, such as micro-machined gyroscopes and accelerometers, are now commodities, and can provide robust estimates of self-motion as well as 3D scene structure without any external infrastructure [47]. Based on these, various techniques have been developed for UAV control using multi-sensory information [35, 46].

Without Data Fusion

Some works use multiple sensors to obtain independent, complementary information. Data fusion is not further conducted. In [25], three kinds of sensors are used: an on-board sonar pointed at the ceiling for quadrotor height estimation, an on-board laser for position and yaw angle estimation, and an off-board monocular camera with a fisheye lens, which captures the quadrotor with two on-board colored markers, for quadrotor position and yaw angle estimation as a redundancy for the laser. In [35], the altitude control is only based on a downward-looking sonar, while another four sonars are used for obstacle avoidance.

With Data Fusion

If the sensor information overlaps or is dependent on each other, an elaborate data fusion is necessary. Data fusion algorithms such as the *Kalman filter* (KF) and its extensions, particle filter, as well as multi-sensor multi-rate data fusion are introduced in [95]. Various fusion techniques and applications in the literature are surveyed below.

- Complementary Filter Complementary filters have been developed to deal with the problem of data fusion for aerial vehicles. In [128], a complementary filter is used to fuse the data from accelerometers, the compass sensor, and the gyro sensors, while a nonlinear complementary filter uses IMU and dynamic pressure measurements to achieve high-frequency attitude estimates for a fixed-wing UAV [57].

- KF For linear systems, KF can be used to combine multi-sensory data for better estimation of system state, for example fusion of visual and wheel odometry data [146] or fusion of visual and inertial data [110].

- Extended Kalman Filter (EKF) Since quadrotors perform a non-linear dynamic behavior, an EKF has been used for data fusion. In [19], an EKF combines the quadrotor's pitch and roll angles with odometry data from a stereo camera or a laser at 50 Hz. The vehicle position, velocity, acceleration, and biases of the IMUs are estimated. For a helicopter with different process models as a quadrotor, an EKF functions as a mixed continuous-discrete time filter and uses visual data to correct the position and attitude estimation of the IMUs [129]. For motion estimation, optical flow information is fused with IMU data by an EKF in [80]. Quadrotor motion control is achieved based on optical flow in [80] using a downward-looking camera and IMUs. A KF is used to integrate angular velocity and optical flow as well as estimate translational components of the optical flow and vehicle angular velocity. Then, an EKF uses the translational optical flow to recover translational motion and structure parameters. As stated in the paper, the self-motion and height of the quadrotor are still recovered modulo some unknown scale factors, which can be solved by, for example, a static pressure sensor for relative height measurements.

- Particle Filter A particle filter is used to fuse the visual data and the on-board IMU data, in order to estimate 3D quadrotor position with respect to the target in [39].

If the data to be fused are from sensors with different sampling rates, multi-rate fusion should be accomplished. In [107], a theoretically sound multi-rate fusion of visual data at a lower frequency and inertial data at a higher frequency is proposed in order to estimate the orientation of the camera. A system observer and prediction architecture considering different and complementary sampling frequencies are designed to use vision data to compensate for the errors of the integrated gyro signals. As stated in the paper, this algorithm can be extended to other DOFs. However, it has not yet been evaluated in real-time experiments.

2.3 Quadrotor Control Design

A quadrotor helicopter is an unstable system with non-linear dynamics. Based on accurate pose estimation, controllers should be elaborated to enable a stable flying behavior. Due to the under-actuated characteristic, control design is very interesting and challenging. In this section, various control strategies in the literature are summarized first. Afterwards, control design for cooperative air-ground multi-robot systems is taken into account.

2.3.1 Overview of Control Strategies

In most existing works, the nonlinear dynamic model of a quadrotor is analyzed first. Then, various control techniques are applied considering characteristics of their own platforms. Above all, various control strategies such as control using the Lyapunov theory, *Proportional-Integral-Derivative* (PID) controllers, *Linear Quadratic* (LQ) controllers, the backstepping technique, and the sliding mode technique have been surveyed and evaluated in simulations and on the test-bench in [35]. A combination of PID and backstepping-based controllers – integral backstepping [36] – is successfully applied to the quadrotor for taking-off, hovering, landing, and collision avoidance.

PID Controller

Generic PID controllers are commonly used as benchmarks to evaluate quadrotor flying performance. Many previous works, whose focus is not the control design, have applied independent PID controllers for simplicity [60, 109]. Moreover, considering the zero dynamics of the quadrotor, the *Proportional-Derivative* (PD) control law can be derived from the dynamic model [21]. *Proportional* (P) controllers have been applied for altitude and yaw angle control [22]. An adaptive control extension based on PD controllers for tracking maneuvers is proposed in [106].

LQ Controller

The LQ controller is a kind of optimal control strategy for linear systems. For the non-linear dynamics of quadrotors, systems should be linearized. In [72], an LQ controller is used for inner loop attitude control of the quadrotor STARMAC and implemented on-board. Another LQ controller has also been simulated in [35]. For spatial trajectory tracking, an LQ servo controller is applied in [115] augmented by a KF. However, due to system reduction, orientation control is not possible. In [19], an LQ controller has been used to enable a stable hovering of the platform in a small, local environment.

Backstepping-Based Controller

The backstepping technique is commonly used for system processes with recursive dependency, in which some system states are controlled by the other states [83]. Analyzing the dynamic model of the quadrotor, it can be seen that some states fulfill this prerequisite. Therefore, backstepping-based controllers are widely applied for quadrotor control [21, 35, 90, 93, 115]. One of the advantages of the backstepping technique is that the

stability is guaranteed through Lyapunov functions. However, most evaluations have only been conducted in simulations.

Sliding Mode Technique

Sliding mode control (SMC) is another Lyapunov-based method for nonlinear systems and is a form of variable structure control. Considering its advantages such as robustness and ensuring Lyapunov stability, SMC has been applied in quadrotor control [35, 37, 53, 125]. But as stated in [35], the sliding mode technique does not provide excellent results, partly due to the switching nature of the controller, which seems to be ill-adapted to the dynamics of the quadrotor. To eliminate chattering when applying SMCs, saturation nonlinearity is also considered in [72]. As for backstepping-based controllers, most works have only considered them in simulations.

Modifications based on the aforementioned controllers are also performed. Besides a PD controller for the yaw angle, nonlinear controllers for roll and pitch angle control based on nested saturations are designed in [42] which stabilize integrators in cascade (angle, angular velocity, linear displacement, and linear velocity in x- or y-direction). Similar research can also be found in [56].

In most of the aforementioned works, the quadrotors' positions and yaw angles are controlled in a position-based manner. In the works which deploy on-board cameras to observe landmarks, the control problem has also been formulated as an image-based visual servoing problem [39, 61].

2.3.2 Air-Ground Multi-Robot Control

In recent years more and more autonomous multi-robot systems have been applied in military, industry, and civilian domains [101]. Especially for search-and-rescue or inspection tasks, cooperation between UAVs and *Unmanned Ground Vehicles* (UGVs) is desired and can provide 3D sensing of the environment and more flexibility than other homogeneous robot systems. Examples include the development of an air-ground robotic ensemble for environmental monitoring applications [55], the heterogeneous robot group consisting of a robotic helicopter and multiple wheeled ground robots [120], a semi-autonomous UGV aided by a UAV which flies ahead of the UGV to detect holes and other obstacles [117], coordination of multiple UAVs and multiple UGVs [43, 44, 74], landing of a helicopter on a moving target [112], multiple quadrotors equipped with on-board cameras capable of tracking a ground vehicle cooperatively [29], the teleoperated vision-based unmanned air-ground vehicles consisting of a quadrotor and a UGV called VolksBot [121], and the multi-robot platform consisting of a helicopter, a ground vehicle, and a quadrotor helicopter designed in [119] aiming at facilitating the development of multi-vehicle control algorithms. Some recent works also consider indoor cooperation of quadrotors and mobile ground robots for autonomous navigation and mapping [27, 81] and tracking multiple ground robots [68].

In most works, flying systems serve as a central/global control unit and provide information for the ground robots. In this thesis, however, feedforward control of the quadrotor by tracking the mobile ground robot is considered. The ground robot sends its motion information to the quadrotor, in order to facilitate the quadrotor performance.

2.4 Insect-Like Vision on MAVs

In recent years, applying bio-inspired algorithms or techniques to technical systems has become a tendency of robotics research. For the development of MAVs, flying insects with small sizes and flexible dynamics have become the most appropriate biological model, since they are extremely capable of efficient information processing and robust navigation in complex environments. They use information derived from optical flows for navigation and the motor control is based on this kind of relative data rather than absolute quantity of motion.

2.4.1 Findings and Modeling in Biology

The sensing systems of insects have been extensively explored in biology. For example, a fly's panoramic vision system comprises at its front end several thousand photoreceptors feeding into a 2D array of motion detecting neurons which the animal uses for dynamic visuomotor pose and gaze stabilization and navigation in 6 DOFs. A survey of visual motor computations in insects is given in [116].

The Reichardt detector [32, 54, 67, 71, 99, 108], also called the *Elementary Motion Detector* (EMD), is a well-known model which describes, at an algorithmic level, the process of local motion detection in the fly, leading from non-directional input to a direction selective output. In a structure of the fly brain called "lobula plate", large neurons are found which integrate these local motion signals and additionally form extensive connections amongst themselves [33, 64]. These neurons have large RFs and respond best to particular flow-fields such as those occurring during certain maneuvers of the fly in free flight [49, 84]. To overcome the sensitivity of the previously proposed correlation-based algorithms to certain undesired image properties such as contrast, a model with multiple layers of non-linear dynamic adaptive components is proposed in [40], based on the known neuron responses of fly brains during ego-motion. More robust performance is investigated for natural images.

2.4.2 Technical Realizations

In engineering applications such as robotics, driver assistance systems, or surveillance systems, a camera system is usually used as a sensor to gather information about the environment. Motion perception based on the fly's vision system is computationally cheap and, thus, particularly suited for those real-time applications.

In [76], a bee-inspired navigation mechanism for goal-directed aerial navigation is proposed. The combination of course stabilization and an EMD-based visual odometer is applied to a blimp-type robotic platform equipped with a panoramic camera, which is however much more easily controlled than other platforms, such as quadrotors.

Insect-inspired flying robots have been studied with the focus on biological principles such as the motion commands generated by low-resolution vision and gyroscopic data, the course stabilization, and the obstacle avoidance in [130]. A fixed-wing airplane equipped with several miniature cameras and a gyroscope is used as a testbed.

In [45], a quadrotor is enabled to autonomously navigate in a corridor-like environment with texture-rich walls and floor. On-board accelerometers and a sonar provide the altitude information, while an on-board camera with a 360° field of view provides wide-field optical flow. Additionally, an optical flow sensor is ventrally mounted on-board the quadrotor. The vehicle heading, lateral position, and forward velocity are controlled based on estimates from the decomposed patterns of the optical flow field.

A coaxial helicopter is used in [65] to test insect-inspired control algorithms. The vehicle is captured by an off-board camera. Then, the respective sensory inputs are simulated in a virtual 3D environment and used to control the yaw angle of the aerial vehicle.

2.4.3 High-Speed Implementations

In addition to optimizing the motion estimation algorithms based on insect-like vision, it is also necessary to select suitable hardware.

In [111], a new EMD circuit implemented on MAVs has been designed by using *Field Programmable Analog Array* (FPAA). In addition, several EMD-based models have been developed based on *Very-Large-Scale-Integrated* (VLSI) circuits.

In [88], a low-power VLSI chip is described which consists of a one-dimensional array of EMDs to perform motion computation. In [71], a biologically inspired VLSI system for the measurement of self-motion is introduced.

Later, a single-chip analog VLSI sensor that can detect imminent collisions was designed and tested [66].

Recently, *Field Programmable Gate Arrays* (FPGAs) are favored by engineers in implementing EMDs. In [97], a real-time algorithm for estimating motion vectors is implemented. In [26], an FPGA implementation of a bio-inspired visual sensor is introduced. However, all these implementations perform motion detection with relatively low resolutions and low frame rates.

Working with a relatively high resolution (127×100 pixels) at a relatively high frame rate (200 Hz), an FPGA-based smart camera module with a small dimension is presented in [82].

2.5 Summary

Vision guidance may be the most elegant solution for a flying system. However, vision-based pose estimation for a quadrotor platform without external GPS or a global sensor is a very challenging problem. Up to now, only a few works have applied a monocular on-board camera with IMUs to achieve accurate pose estimation without additional sensors such as lasers or sonars. The reasons are, on the one hand, that quadrotors have specific flying characteristics: A displacement in a horizontal direction is accomplished by the pitch or roll of the body, which results in a large variation of the camera view field. On the other hand, common scenarios using on-board monocular cameras in the literature are VTOL and hovering, in which the camera's limited field of view is not as critical as the tracking and landing scenarios considered in this thesis.

Controlling an under-actuated system with fast, non-linear dynamic behavior is also a demanding task. Most state-of-the-art works have shown simulated behaviors or performance on test-benches, which are, however, quite different to real world experiments. Real-time experiments have only been restrictedly conducted and only VTOL or hovering has been realized. Few works have explored a complete, integrated scenario, especially feedforward control scenarios aided by a ground robot as a mobile reference.

Although the advantages of insect flying behavior have been extensively explored in biology, realizations in technical systems are still limited. Closed-loop control based on quantitative motion estimation and sufficient implementation are missing as yet. Due to their low computational cost, performing high sensibility and robust motor control, bio-inspired algorithms are definitively going to provide a promising future for MAV development.

3 Multi-Sensory Pose/Motion Estimation

A significant challenge in developing *Unmanned Aerial Vehicles* (UAVs) is accurate estimation of their pose and motion by means of extracting and fusing useful sensor information in a robust manner. Most flying platforms including the quadrotor used in this thesis are equipped with *Inertial Measurement Units* (IMUs). However, the drift of inertial sensors leads to errors during time-discrete integration, making steadily accurate estimation of the absolute pose nearly impossible.

Vision, as one of the strongest information origins, can provide a large amount of information about position, velocity, and orientation, and is commonly used in most biological organisms such as in humans and animals for self-localization and navigation. Compared to IMUs, vision sensors have advantages such as accurate pose estimation without propagating errors.

However, due to the limited field of view and relatively low sampling rate as well as relatively complex data processing, visual data are not sufficient for pose/motion estimation and control of a highly dynamic flying system in which high-frequency noise and vibrations occur. Therefore, high-frequency and accurate multi-sensory pose/motion estimation for a quadrotor which serves as a basis for a controlled behavior is investigated.

In this chapter, aiming at obtaining accurate pose/motion estimation of a highly dynamic quadrotor relying only on on-board sensors, a thoroughly designed high-frequency fusion of the inertial sensors and the vision sensor is accomplished. Based on multi-modal sensor information, a continuous-discrete *Extended Kalman Filter* (EKF) is applied for the multi-sensory multi-rate data fusion, where the high-frequency IMU data drive the process model and the low-rate vision data correct the estimation. Complete data synchronization is conducted based on the accurately measured time delay. Various real-time experiments considering the quadrotor tracking the mobile ground robot show that 1) high accuracy and high frequency of the pose/motion estimation in dynamic behaviors are obtained through the multi-sensory multi-rate data fusion; 2) one of the first autonomous flying quadrotors based on minimal on-board sensors and the complete on-board integration of sensor data processing is achieved.

The remainder of this chapter is organized as follows. First, the system hardware configuration is briefly introduced in Section 3.1. Then, the problem addressed in this chapter and the derivation of the quadrotor dynamic model are described in Section 3.2. After that, IMU data processing, visual data processing including marker-based pose estimation and optical-flow-based motion estimation, and the multi-sensory data fusion are described in Section 3.3. In Section 3.4, experimental evaluation is conducted and presented. A discussion and a summary are given in Sections 3.5 and 3.6, respectively.

3.1 System Overview

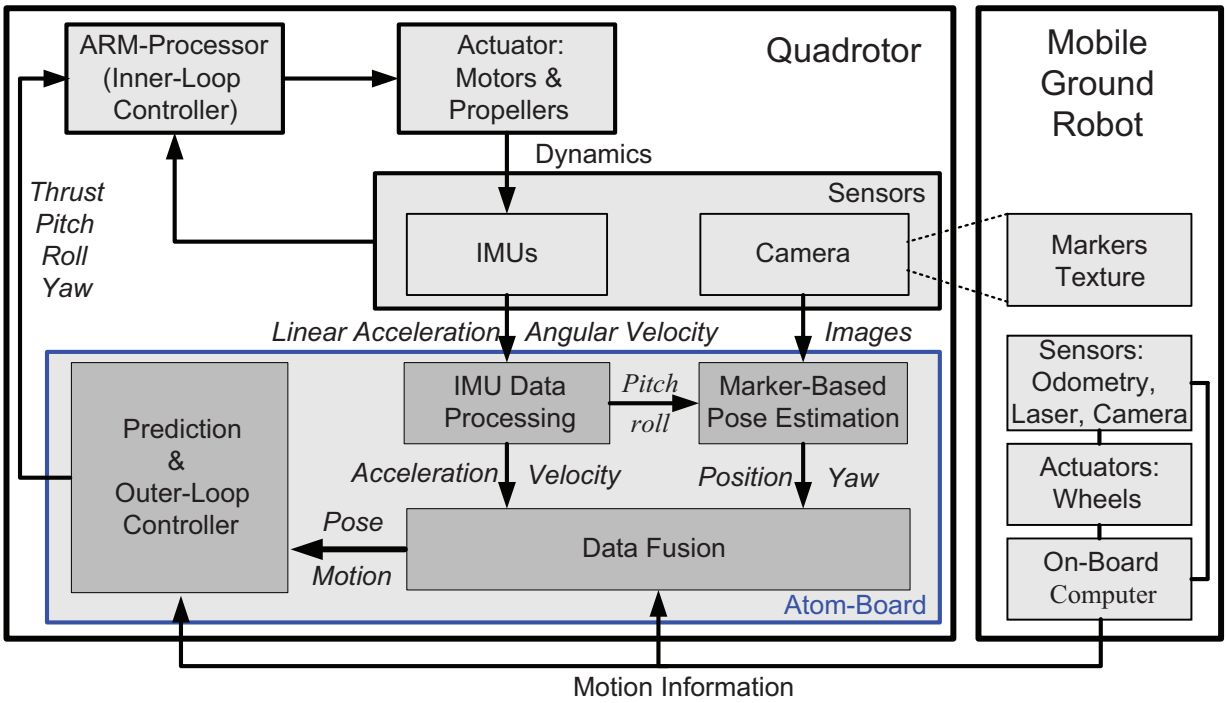

Fig. 3.1: System overview. Left: a quadrotor equipped with an ARM-processor, actuators, sensors, and an ATOM-board; right: a mobile ground robot equipped with markers/texture, sensors, actuators, and an embedded computer.

The overall air-ground multi-robot system designed in this thesis is illustrated in Fig. 3.1. A quadrotor is equipped with minimal sensors (IMUs and a monocular camera), efficient actuators (motors and propellers) as well as on-board processing and control units (an ARM-processor and a board with an ATOM CPU referred to as "ATOM-board" in this thesis). Aiming at cooperative task accomplishment, a ground robot is applied to be a mobile external reference and facilitate the flying agent – the quadrotor. Note that in this system, no ground station is used. The quadrotor and the mobile robot are autonomous.

3.1.1 Quadrotor

The Hummingbird quadrotor from Ascending Technologies GmbH [3] was chosen as the hardware platform for this work (see Fig. A.1 in Appendix A). The off-the-shelf controller offers a 1 kHz control frequency and motor update rate, which enables fast responses to changes in the environment. The configuration of the quadrotor is basically the same as described in [63].

Actuators

Two pairs of rotors spin clockwise and anticlockwise respectively. They are directly driven by high-torque DC brushless motors that are electronically commutated by optimized controllers to achieve fast responses.

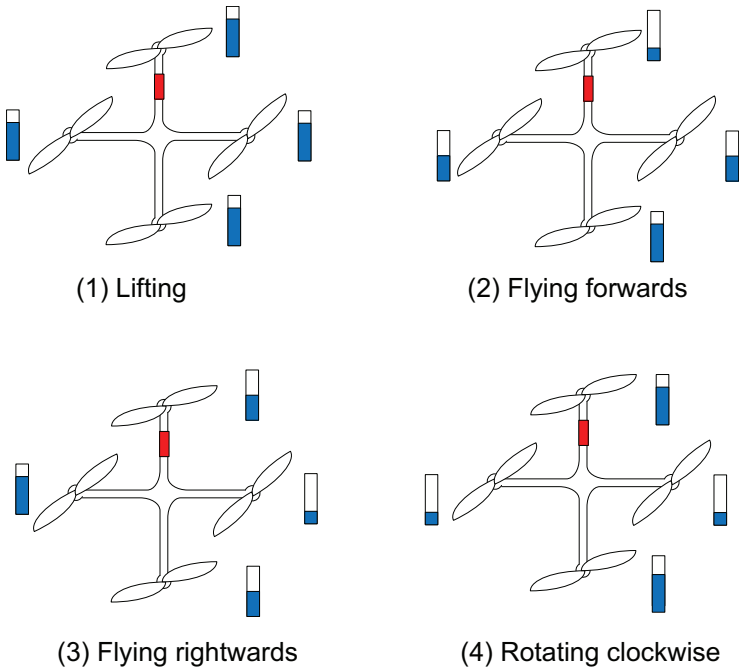

Fig. 3.2: Flying upwards (1), forwards (2), rightwards (3), and clockwise (4) by changing the rotating velocities of the respective rotors. The red bars on the quadrotor frame indicates the front rotors. The half filled bars near the rotors indicate the individual rotor rotating speed.

The flying motion of a quadrotor is determined by the rotational speeds of the four motors (see Fig. 3.2). If the rotating velocities of all four motors are increased by the same amount, the quadrotor flies upwards. When the left motor is faster than the right one, the quadrotor rolls around the front and back wings, and flies rightwards. Analogously, the quadrotor can pitch around the left and right wings due to the speed difference between the front and the back motors. The motion in the horizontal plane is realized by the pitch and roll angles. The yaw rotation around the vertical axis is caused by the difference between the angular momentum generated by these two pairs of rotors. Therefore, the quadrotor is an under-actuated system which has, through controlling the four rotor speeds, 6 *Degrees of Freedom* (DOFs) but only four control variables, namely the pitch angle, the roll angle, the yaw angle velocity, and the thrust.

On-Board Sensors

The main sensors on the quadrotor platform used in this thesis are on-board IMUs and a monocular camera.

– IMUs The inertial sensors on the quadrotor consist of three orthogonal gyroscopes measuring the angular velocity for each rotation axis and three orthogonal accelerometers measuring the acceleration for each translation axis. The absolute roll and pitch angles are estimated on-board by fusing the acceleration vector and angular speeds. A relative yaw angle is estimated by integrating the angular speed measured by the yaw gyroscope.

– Camera With respect to the limited payload of the quadrotor, a light "Firefly MV" from Point Grey Research Inc. [103] is selected as the on-board camera (see Fig. A.4 in Appendix A). The key parameters of the camera are listed in Tab. A.1. This camera faces downwards to acquire position information of the mobile ground robot.

On-Board Computational Units

The quadrotor comprises an ARM-processor to operate an off-the-shelf controller which stabilizes all three axes of rotation (roll, pitch, and yaw) with three independent *Proportional-Derivative* (PD) controllers for each angle at a frequency of 1 kHz, as the quadrotor itself is unstable during flight [63]. This controller exclusively uses IMU data as feedback and cannot be modified. In this work, this controller is called the inner-loop controller.

Although the errors due to time-discrete integration are small at the rate of 1 kHz, an outer control loop is needed to compensate for the drift and to control all 6 DOFs of the quadrotor. A small board with an ATOM CPU of Intel Corp. (1.6 GHz Dual Core) called *Flying Netbook* developed by Ascending Technologies GmbH is installed on the quadrotor. It is capable of outer-loop control computation, sensor data processing, and data fusion, which makes autonomous flight possible. The on-board camera is connected via a *Universal Serial Bus* (USB) interface with the ATOM-board, while the data transfer between the ARM-processor and the ATOM-board is via *Universal Asynchronous Receiver Transmitter* (UART). The control design is the focus of Chapter 4 and not further investigated in this chapter.

3.1.2 Mobile Ground Robot

In this air- and ground-based multi-robot system, a Pioneer P3-DX mobile robot from Mobile Robots Inc. [10] is used to operate on the floor (see Fig. B.1 in Appendix B). This robot platform is 44×38×22 cm in dimensions with two drive wheels with a diameter of 16.5 cm each. The maximum speed is 1.6 m/s. It is equipped with different sensors such as laser, sonars, and a camera. The control of the robot is achieved by programming an embedded computer. To complete the data fusion and improve the control performance, motion information of the ground robot is transmitted to the quadrotor via wireless *Local Area Network* LAN using *User Datagram Protocol* (UDP). The details of this robot can be found in Appendix B. The pose/motion estimation and control of this mobile ground robot is not the focus of this thesis and is not investigated further.

In order to robustly detect the ground robot from the quadrotor's perspective, two active markers are mounted on the robot. The center point of these markers is referred to as the reference position of the ground robot. Moreover, since the deck of the pioneer robot is too small for the quadrotor to stand on it, a 80×80 cm planar board with texture is mounted on the robot as the platform for take-off and landing.

3.2 Problem Definition and Quadrotor Dynamic Model

3.2.1 Problem Definition

In the literature, the quadrotor pose/motion estimation using on-board vision systems has been proposed in the following manners:

- Some works additionally use other sensors such as ranging sensors [35, 73, 131] to acquire complementary or redundant information and facilitate vision-based estimation, which causes an increasing payload and higher computation complexity.

- Some other works consider stereo cameras [19, 23, 78], multi-camera configuration [21], or fish-eye cameras [48]. This results in high cost and critical calibration problems.

- Quadrotors in most works are not exactly autonomous, which means the input images are normally transmitted to a ground station for image processing and computation of the control law which should be sent back to the quadrotors, resulting in time delay and a limited workspace [39, 79, 123].

- Most works only present *Vertical Take-Off and Landing* (VTOL) without references [35, 70] or use static external references [39, 96]. Few works use mobile external references [81, 123], which brings difficulty to the system due to the quadrotor's flying characteristics and the limited field of view of the camera.

To overcome the difficulties raised in the existing works, this chapter focuses on the problem of how to use a minimal sensor combination – on-board IMUs and a monocular camera at different rates – to obtain accurate pose/motion estimation. Here, a high frequency of the estimation is envisioned, as it is essential for tracking a mobile reference and the following control performance. Moreover, various aspects facilitating the estimation accuracy, such as complete on-board computation, noise variance investigation, as well as thorough time delay analysis and sensor data synchronization, should be conducted.

3.2.2 Quadrotor Dynamic Model

To obtain the quadrotor position and orientation at any one instant, a detailed derivation of its kinematic and dynamic formulations is necessary. In this section, the quadrotor dynamic model is derived.

Reference Frames

Since the parameters needed for the analysis of quadrotor dynamics are specified in different coordinate frames, various frames of reference, illustrated in Fig. 3.3, are defined first. The definitions and transformations are described below.

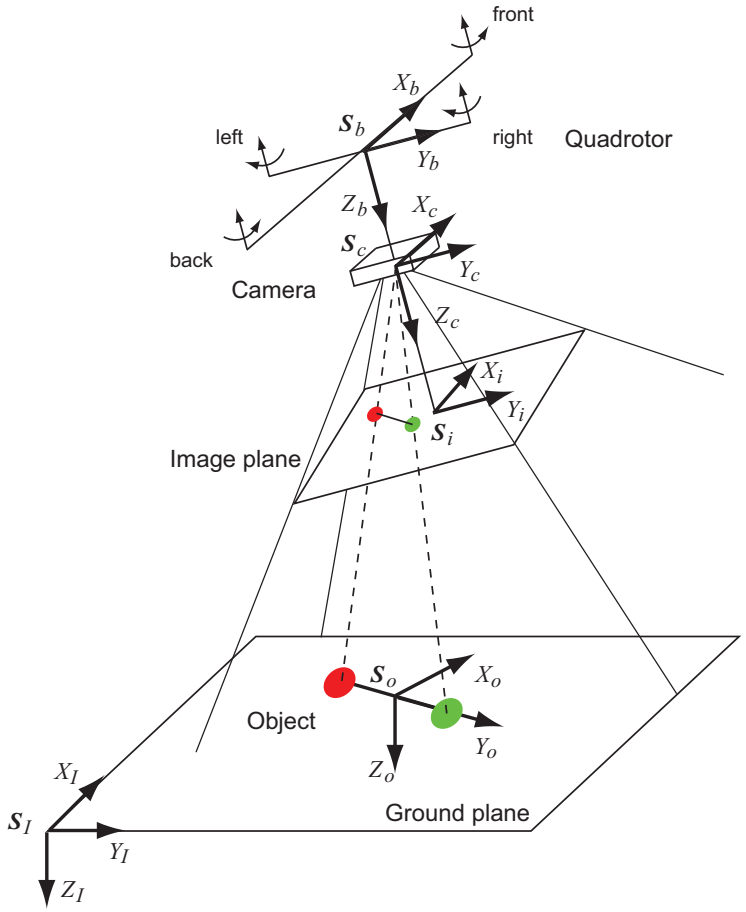

Fig. 3.3: Frames of reference. The inertial frame $\mathbf{S}_I$, the object frame $\mathbf{S}_o$, the quadrotor body frame $\mathbf{S}_b$, the camera frame $\mathbf{S}_c$, and the image plane frame $\mathbf{S}_i$.

Note: For simplification, the following expressions are used:

$$s(\cdot) \Leftrightarrow \sin(\cdot), \quad c(\cdot) \Leftrightarrow \cos(\cdot), \quad \text{and} \quad t(\cdot) \Leftrightarrow \tan(\cdot).$$

– The Inertial Frame of Reference $\mathbf{S}_I$ For the sake of convenience, the origin and the X_I/Y_I-plane of the initial reference frame is fixed on the ground, which is assumed to be plane, while the Z_I-axis perpendicularly points down to the ground plane. A 3D point in the inertial frame is denoted by ${}_I\boldsymbol{p} \in \mathbb{R}^3$. An object located on the X_I/Y_I-plane has only three DOFs: the 2D position and the orientation.

– The Object Frame $\mathbf{S}_o$ The object frame is defined with respect to the ground robot, which is equipped with two markers. The origin is the center of the markers with the coordinate ${}_I\boldsymbol{p}_r = [{}_Ix_r \;\; {}_Iy_r \;\; -24\,\text{cm}]^T$ (24 cm refers to the height of the ground robot). The subscribe r denotes the ground robot. The ground robot is a non-holonomic system and consists of a forward linear motion and an angular motion around its vertical axis. The X_o-axis points out the forward direction of the ground robot, which has a relative angle

Ω_r around the Z_I-axis with respect to X_I according to the right hand rule. The Z_o-axis is the vertical axis and coincident with the Z_I-axis. The linear velocity and acceleration of the ground robot in the current object frame are denoted by $\boldsymbol{v}_r = [v_r \quad 0 \quad 0]^T$ and $\boldsymbol{a}_r = [a_r \quad 0 \quad 0]^T$, respectively, while the angular velocity and acceleration in the current object frame are denoted by $\dot{\Omega}_r$ and $\ddot{\Omega}_r$, respectively.

The homogeneous transformation matrix between $\mathbf{S}_I$ and $\mathbf{S}_o$ can be formulated as follows:

$$ {}^I\boldsymbol{T}_o = \begin{bmatrix} {}^I\boldsymbol{R}_o & {}^I\boldsymbol{r}_o \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{3.1} $$

with the rotation matrix:

$$ {}^I\boldsymbol{R}_o = \mathrm{Rot}(Z_I, \Omega_r) = \begin{bmatrix} c\Omega_r & -s\Omega_r & 0 \\ s\Omega_r & c\Omega_r & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.2} $$

and the translation vector is the same as the position vector of the ground robot in the inertial frame:

$$ {}^I\boldsymbol{r}_o = {}_I\boldsymbol{p}_r = [{}_Ix_r \quad {}_Iy_r \quad -24\,\mathrm{cm}]^T. \tag{3.3} $$

The markers on the top of the ground robot are mounted with a distance of δ with each other. Each marker has a radius of 2 cm. The marker center positions ${}_o\boldsymbol{p}_{1/2}$ in the object frame are as follows:

$$ {}_o\boldsymbol{p}_1 = [0 \quad -\frac{\delta}{2} \quad -2\,\mathrm{cm}]^T \quad \text{and} \quad {}_o\boldsymbol{p}_2 = [0 \quad \frac{\delta}{2} \quad -2\,\mathrm{cm}]^T. \tag{3.4} $$

– The Quadrotor Body Frame $\mathbf{S}_b$ The quadrotor body frame has its origin at the center of the gravity of the quadrotor. The X_b-axis points out the nose of the airframe, the Y_b-axis points out the right wing, and the Z_b-axis points out the belly.

As the mobile ground robot is used as an external reference for the quadrotor, the relationship between $\mathbf{S}_o$ and $\mathbf{S}_b$ can be formulated as follows:

$$ {}^o\boldsymbol{T}_b = \begin{bmatrix} {}^o\boldsymbol{R}_b & {}^o\boldsymbol{r}_b \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{3.5} $$

The rotation matrix ${}^o\boldsymbol{R}_b$ is calculated through chained rotations using the *Tait-Bryan* angles, namely the yaw, pitch, and roll angles, illustrated in Fig. 3.4. After a translational displacement of ${}^o\boldsymbol{r}_b = {}_o\boldsymbol{p}_q = [{}_ox_q \quad {}_oy_q \quad {}_oz_q]^T$, the body frame is coincident with the object frame first and labeled as body frame $\boldsymbol{S}_{b^1}$. Note that the subscribe q denotes the quadrotor. The body frame $\boldsymbol{S}_{b^1}$ is rotated with the yaw angle $\Psi \in [-\pi, \pi)$ around the current Z_{b^1}-axis into the body frame $\boldsymbol{S}_{b^2}$. After that, $\boldsymbol{S}_{b^2}$ is rotated with the pitch angle $\Theta \in [-\frac{1}{2}\pi, \frac{1}{2}\pi]$ around the current Y_{b^2}-axis into the body frame $\boldsymbol{S}_{b^3}$. Then, $\boldsymbol{S}_{b^3}$ is rotated with the roll angle $\Phi \in [-\frac{1}{2}\pi, \frac{1}{2}\pi]$ around the current X_{b^3}-axis into the actual body frame $\boldsymbol{S}_b$. Finally, the rotation matrix ${}^b\boldsymbol{R}_o$ of the transformation between the object frame and the quadrotor

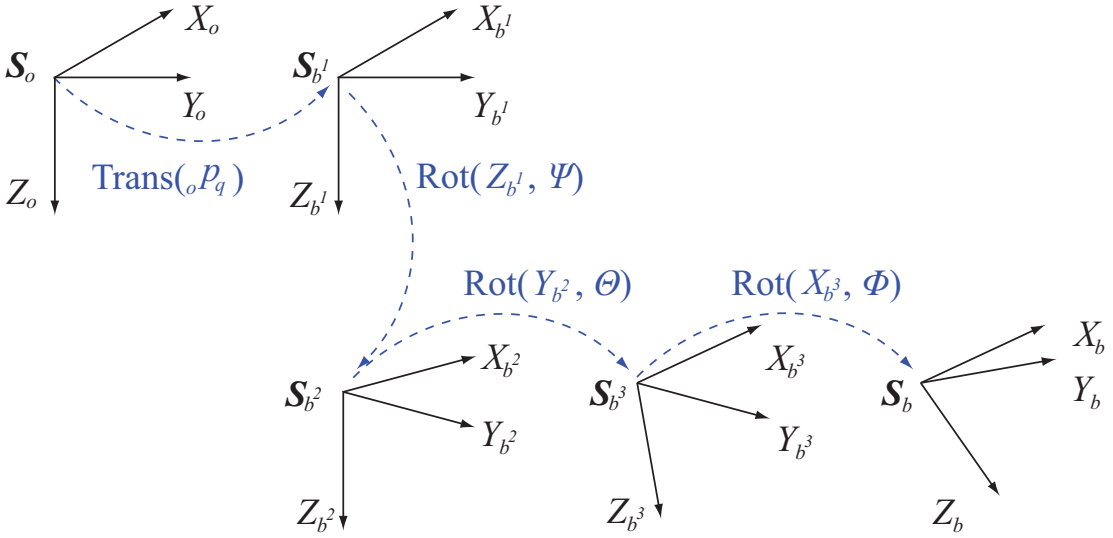

Fig. 3.4: Transformation between the object frame $\mathbf{S}_o$ and the body frame $\mathbf{S}_b$ facilitated by intermediate frames $\mathbf{S}_{b^1}$, $\mathbf{S}_{b^2}$, and $\mathbf{S}_{b^3}$.

body frame is given by

$$\begin{aligned} {}^{b}\boldsymbol{R}_{o} &= {}^{b}\boldsymbol{R}_{b^3} \cdot {}^{b^3}\boldsymbol{R}_{b^2} \cdot {}^{b^2}\boldsymbol{R}_{b^1} \cdot {}^{b^1}\boldsymbol{R}_{o} \\ &= \mathrm{Rot}(X_{b^3}, \Phi) \cdot \mathrm{Rot}(Y_{b^2}, \Theta) \cdot \mathrm{Rot}(Z_{b^1}, \Psi) \cdot \mathrm{Rot}(X_o, 0^\circ) \\ &= \begin{bmatrix} c\Theta c\Psi & c\Theta s\Psi & -s\Theta \\ s\Phi s\Theta c\Psi - c\Phi s\Psi & s\Phi s\Theta s\Psi + c\Phi c\Psi & s\Phi c\Theta \\ c\Phi s\Theta c\Psi + s\Phi s\Psi & c\Phi s\Theta s\Psi - s\Phi c\Psi & c\Phi c\Theta \end{bmatrix}. \end{aligned} \tag{3.6}$$

Then,

$${}^{o}\boldsymbol{R}_{b} = ({}^{b}\boldsymbol{R}_{o})^{T}. \tag{3.7}$$

– The Camera Frame $\mathbf{S}_c$ Since the camera looking downwards is firmly mounted on the quadrotor, the camera frame $\mathbf{S}_c$ is coincident with the body frame of the quadrotor $\mathbf{S}_b$ based on a translation vector. The origin is located at the lens center of the camera.

– The Image Frame $\mathbf{S}_i$ The image frame has its origin located at the principle point of the input image. The unit vector Y_i along the image horizontal direction directs the right rotor of the quadrotor, and the unit vector X_i points out the front rotor.

Quadrotor Dynamic Modeling

In this work, the relative pose of the quadrotor with respect to the mobile ground robot in the multi-robot system is to be estimated and controlled. To derive the kinematic and dynamic expressions, the following state variables of the quadrotor are defined:

– Position $x \quad y \quad z$ The quadrotor position with respect to the mobile ground robot ${}_o\boldsymbol{p}_q = [x \quad y \quad z]^T$ in the X_o-, Y_o-, and Z_o-directions are denoted by $x = {}_ox_q$, $y = {}_oy_q$, and $z = {}_oz_q$ for simplicity. A positive height of the quadrotor with respect to the ground robot is denoted by $h = -z$.

Since the 3D trajectory of the quadrotor in the inertial frame is also of particular interest, in the experimental results presented in the thesis, the quadrotor position $[{}_Ix \quad {}_Iy]^T$ and heading ${}_I\Psi$ are also illustrated.

– Linear Velocity $\dot{x} \quad \dot{y} \quad \dot{z}$ The linear velocity ${}_o\dot{\boldsymbol{p}}_q = [\dot{x} \quad \dot{y} \quad \dot{z}]^T$ is defined in the object frame.

The absolute position of the quadrotor in the inertial frame ${}_I\boldsymbol{p}_q$ can be calculated via coordinate transformation:

$$ {}_I\boldsymbol{p}_q = {}_I\boldsymbol{p}_r + {}^I\boldsymbol{R}_o \cdot {}_o\boldsymbol{p}_q, \tag{3.8} $$

where ${}_I\boldsymbol{p}_r$ represents the absolute position of the ground robot in the inertial frame. After differentiating Eq. 3.8, the velocity of the quadrotor in the inertial frame can be expressed by

$$ {}_I\dot{\boldsymbol{p}}_q = {}_I\dot{\boldsymbol{p}}_r + {}^I\dot{\boldsymbol{R}}_o \cdot {}_o\boldsymbol{p}_q + {}^I\boldsymbol{R}_o \cdot {}_o\dot{\boldsymbol{p}}_q, \tag{3.9} $$

and its extended form is as follows:

$$ \begin{bmatrix} {}_I\dot{x} \\ {}_I\dot{y} \\ {}_I\dot{z} \end{bmatrix} = \begin{bmatrix} v_r \\ 0 \\ 0 \end{bmatrix} + {}^I\dot{\boldsymbol{R}}_o \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} + {}^I\boldsymbol{R}_o \cdot \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}, \tag{3.10} $$

where from Eq. 3.2

$$ {}^I\dot{\boldsymbol{R}}_o = \begin{bmatrix} -s\Omega_r \cdot \dot{\Omega}_r & -c\Omega_r \cdot \dot{\Omega}_r & 0 \\ c\Omega_r \cdot \dot{\Omega}_r & -s\Omega_r \cdot \dot{\Omega}_r & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{3.11} $$

– Linear Acceleration $\ddot{x} \quad \ddot{y} \quad \ddot{z}$ The linear acceleration of the quadrotor in the object frame is denoted by ${}_o\ddot{\boldsymbol{p}}_q = [\ddot{x} \quad \ddot{y} \quad \ddot{z}]^T$.

According to Eq. 3.9, the acceleration of the quadrotor in the inertial frame can be expressed by

$$ {}_I\ddot{\boldsymbol{p}}_q = {}_I\ddot{\boldsymbol{p}}_r + {}^I\ddot{\boldsymbol{R}}_o \cdot {}_o\boldsymbol{p}_q + 2 \cdot {}^I\dot{\boldsymbol{R}}_o \cdot {}_o\dot{\boldsymbol{p}}_q + {}^I\boldsymbol{R}_o \cdot {}_o\ddot{\boldsymbol{p}}_q. \tag{3.12} $$

Then, the linear acceleration of the quadrotor in the object frame ${}_o\ddot{\boldsymbol{p}}_q = [\ddot{x} \quad \ddot{y} \quad \ddot{z}]^T$ can be calculated as follows:

$$ {}_o\ddot{\boldsymbol{p}}_q = {}^I\boldsymbol{R}_o^{-1} \cdot ({}_I\ddot{\boldsymbol{p}}_q - {}_I\ddot{\boldsymbol{p}}_r - {}^I\ddot{\boldsymbol{R}}_o \cdot {}_o\boldsymbol{p}_q - 2 \cdot {}^I\dot{\boldsymbol{R}}_o \cdot {}_o\dot{\boldsymbol{p}}_q), \tag{3.13} $$

where from Eq. 3.2

$$ {}^I\boldsymbol{R}_o^{-1} = \begin{bmatrix} c\Omega_r & s\Omega_r & 0 \\ -s\Omega_r & c\Omega_r & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.14} $$

and

$$ {}^{I}\ddot{\boldsymbol{R}}_o = \begin{bmatrix} -c\Omega_r \cdot \dot{\Omega}_r^2 - s\Omega_r \cdot \ddot{\Omega}_r & s\Omega_r \cdot \dot{\Omega}_r^2 - c\Omega_r \cdot \ddot{\Omega}_r & 0 \\ -s\Omega_r \cdot \dot{\Omega}_r^2 + c\Omega_r \cdot \ddot{\Omega}_r & -c\Omega_r \cdot \dot{\Omega}_r^2 - s\Omega_r \cdot \ddot{\Omega}_r & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{3.15} $$

Defining linear velocity of the quadrotor in the body frame as $[u \quad v \quad w]^T$, the linear accelerations of the quadrotor in the inertial frame and in the body frame have the following relationships:

$$ {}_I\dot{\boldsymbol{p}}_q = {}^{I}\boldsymbol{R}_b \cdot \begin{bmatrix} u \\ v \\ w \end{bmatrix}. \tag{3.16} $$

Neglecting the time derivative of the rotation matrix due to quadrotor symmetry and small angle velocity, the linear acceleration of the quadrotor in the inertial frame ${}_I\ddot{\boldsymbol{p}}_q$ can be written as

$$ {}_I\ddot{\boldsymbol{p}}_q \;=\; {}^{I}\boldsymbol{R}_b \cdot \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix}, \tag{3.17} $$

where ${}^{I}\boldsymbol{R}_b = {}^{I}\boldsymbol{R}_o \cdot {}^{o}\boldsymbol{R}_b$.

However, the measurements $[a_x \quad a_y \quad a_z]^T$ from the IMUs are the linear accelerations of the quadrotor in the body frame without consideration of gravitational acceleration. Note that

$$ \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + {}^{b}\boldsymbol{R}_I \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}. \tag{3.18} $$

Therefore, the linear acceleration of the quadrotor in the inertial frame ${}_I\ddot{\boldsymbol{p}}_q$ can be written as

$$ {}_I\ddot{\boldsymbol{p}}_q \;=\; {}^{I}\boldsymbol{R}_b \cdot \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}. \tag{3.19} $$

The linear velocity of the ground robot in the inertial frame is written as

$$ {}_I\dot{\boldsymbol{p}}_r = {}^{I}\boldsymbol{R}_o \cdot \begin{bmatrix} v_r \\ 0 \\ 0 \end{bmatrix}, \tag{3.20} $$

while the linear acceleration of the ground robot in the inertial frame ${}_I\ddot{\boldsymbol{p}}_r$ can be written as

$$ {}_I\ddot{\boldsymbol{p}}_r = {}^{I}\dot{\boldsymbol{R}}_o \cdot \begin{bmatrix} v_r \\ 0 \\ 0 \end{bmatrix} + {}^{I}\boldsymbol{R}_o \cdot \begin{bmatrix} a_r \\ 0 \\ 0 \end{bmatrix}. \tag{3.21} $$

Substituting Eqs. 3.14–3.21 into Eq. 3.13, ${}_o\ddot{\boldsymbol{p}}_q$ can be written as follows:

$$ {}_o\ddot{\boldsymbol{p}}_q = \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} \ddot{x}_{\text{trans}} + \ddot{x}_{\text{rot}} \\ \ddot{y}_{\text{trans}} + \ddot{y}_{\text{rot}} \\ -s\Theta \cdot a_x + s\Phi c\Theta \cdot a_y + c\Phi c\Theta \cdot a_z + g \end{bmatrix} \tag{3.22} $$

with

$$ \begin{aligned} \ddot{x}_{\text{trans}} &= c\Theta c\Psi \cdot a_x + (s\Phi s\Theta c\Psi - c\Phi s\Psi) \cdot a_y + (c\Phi s\Theta c\Psi + s\Phi s\Psi) \cdot a_z - a_r, \\ \ddot{y}_{\text{trans}} &= c\Theta s\Psi \cdot a_x + (s\Phi s\Theta s\Psi + c\Phi c\Psi) \cdot a_y + (c\Phi s\Theta s\Psi - s\Phi c\Psi) \cdot a_z, \\ \ddot{x}_{\text{rot}} &= (\dot{\Omega}_r)^2 \cdot x + \ddot{\Omega}_r \cdot y + 2\dot{\Omega}_r \cdot \dot{y}, \\ \ddot{y}_{\text{rot}} &= -\ddot{\Omega}_r \cdot x + \dot{\Omega}_r^2 \cdot y - 2\dot{\Omega}_r \cdot \dot{x} - \dot{\Omega}_r \cdot v_r. \end{aligned} \tag{3.23} $$

– Orientation $\Phi\ \Theta\ \Psi$ The quadrotor orientation (attitude) is represented using the roll angle Φ, the pitch angle Θ, and the yaw angle Ψ, which are described in the definition of the quadrotor body frame with respect to different reference frames.

– Angular Velocity $\dot{\Phi}\ \dot{\Theta}\ \dot{\Psi}$ The angular velocity measured by the on-board gyroscopes p, q, and r are defined around the X_b-, Y_b-, and Z_b-axis in the body frame. Note that p, q, and r are the velocities of standard Euler angles, while Φ, Θ, and Ψ are Tait-Bryan angles. They are defined in different frames. Therefore,

$$ \begin{aligned} p &\neq \dot{\Phi}, \\ q &\neq \dot{\Theta}, \\ r &\neq \dot{\Psi}. \end{aligned} \tag{3.24} $$

Moreover, the yaw angle Ψ_{imu} derived from the integration of the angular velocity measured by the yaw gyroscope is aligned with the inertial frame, while Ψ is defined with respect to the ground robot. Therefore,

$$ \Psi = \Psi_{\text{imu}} - \Omega_r. \tag{3.25} $$

Then, the $\dot{\Psi}$ defined as the angular velocity around the Z_o-axis is the difference between the $\dot{\Psi}_{\text{imu}}$ and the robot angular velocity $\dot{\Omega}_r$ given by:

$$ \dot{\Psi} = \dot{\Psi}_{\text{imu}} - \dot{\Omega}_r. \tag{3.26} $$

The relationship between p, q, r and $\dot{\Phi}$, $\dot{\Theta}$, $\dot{\Psi}_{\text{imu}}$ can be formulated as follows:

$$ \begin{aligned} \begin{bmatrix} p \\ q \\ r \end{bmatrix} &= \text{Rot}(X_{b^3}, \dot{\Phi}) \cdot \begin{bmatrix} \dot{\Phi} \\ 0 \\ 0 \end{bmatrix} + \text{Rot}(X_{b^3}, \Phi) \cdot \text{Rot}(Y_{b^2}, \dot{\Theta}) \cdot \begin{bmatrix} 0 \\ \dot{\Theta} \\ 0 \end{bmatrix} \\ &\quad + \text{Rot}(X_{b^3}, \Phi) \cdot \text{Rot}(Y_{b^2}, \Theta) \cdot \text{Rot}(Z_{b^1}, \dot{\Psi}_{\text{imu}}) \cdot \begin{bmatrix} 0 \\ 0 \\ \dot{\Psi}_{\text{imu}} \end{bmatrix}. \end{aligned} \tag{3.27} $$

Eq. 3.27 can be simplified by assuming that $\dot{\Phi}$, $\dot{\Theta}$, and $\dot{\Psi}_{\text{imu}}$ are very small. Then,

$$\text{Rot}(X_{b^3}, \dot{\Phi}) = \text{Rot}(Y_{b^2}, \dot{\Theta}) = \text{Rot}(Z_{b^1}, \dot{\Psi}_{\text{imu}}) = \boldsymbol{E}, \tag{3.28}$$

where $\boldsymbol{E}$ denotes a unit matrix. Then, similar to [28], the following equations are obtained:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -s\Theta \\ 0 & c\Phi & s\Phi c\Theta \\ 0 & -s\Phi & c\Phi c\Theta \end{bmatrix} \cdot \begin{bmatrix} \dot{\Phi} \\ \dot{\Theta} \\ \dot{\Psi}_{\text{imu}} \end{bmatrix}, \tag{3.29}$$

and based on Eq. 3.26

$$\begin{bmatrix} \dot{\Phi} \\ \dot{\Theta} \\ \dot{\Psi} + \dot{\Omega}_r \end{bmatrix} = \begin{bmatrix} 1 & s\Phi t\Theta & c\Phi t\Theta \\ 0 & c\Phi & -s\Phi \\ 0 & \frac{s\Phi}{c\Theta} & \frac{c\Phi}{c\Theta} \end{bmatrix} \cdot \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{3.30}$$

The derived dynamic expressions will be used to formulate the system equation and measurement equation required by EKF-aided data fusion.

3.3 Multi-Sensory Multi-Rate Data Fusion

Vision sensors and IMUs have their own advantages and drawbacks, which are nevertheless complementary to each other. For quick motion, IMUs are accurate and sensitive in a short time, whereas vision sensors are not adequate due to the relatively low frame rate and the limited bandwidth. Motion blur in input images also causes large measurement uncertainty. In contrast, for slow motion, IMUs have an unavoidable measurement uncertainty due to high-frequency noise and drifts, while vision sensors are more accurate. Therefore, multi-sensory data fusion is raised into the agenda.

Moreover, the quadrotor has its specific flying characteristic that motion in the horizontal plane is realized by pitch and roll angles. With small pitch and roll, the camera's field of view changes a lot. If the pose estimation result is poor or not fast enough, the ground robot may be lost from the camera's field of view. Therefore, a high frequency of data fusion is also a critical issue.

Fig. 3.5 illustrates the sensor data processing procedure. The linear acceleration and velocity as well as the attitude in the inertial frame (also partly in the object frame) can be computed from IMU data, while the position, heading, and their variation rates in the object frame can be obtained from vision data. They partly overlap and should be fused together in order to achieve more accurate pose/motion estimation. An EKF [127] is desired to be the dynamic optimization filter for the multi-sensory data fusion. As discussed in [24, 30], the unmodeled dynamics and uncertainties are more significant than the linearization errors. Therefore, the unscented *Kalman Filter* (KF) introduced in [124] is not selected in this case.

In this section, IMU data processing and image processing are introduced, respectively. Then, the details of data fusion using a continuous-discrete EKF is described. An essential aspect in the fusion process – the data synchronization – is discussed later.

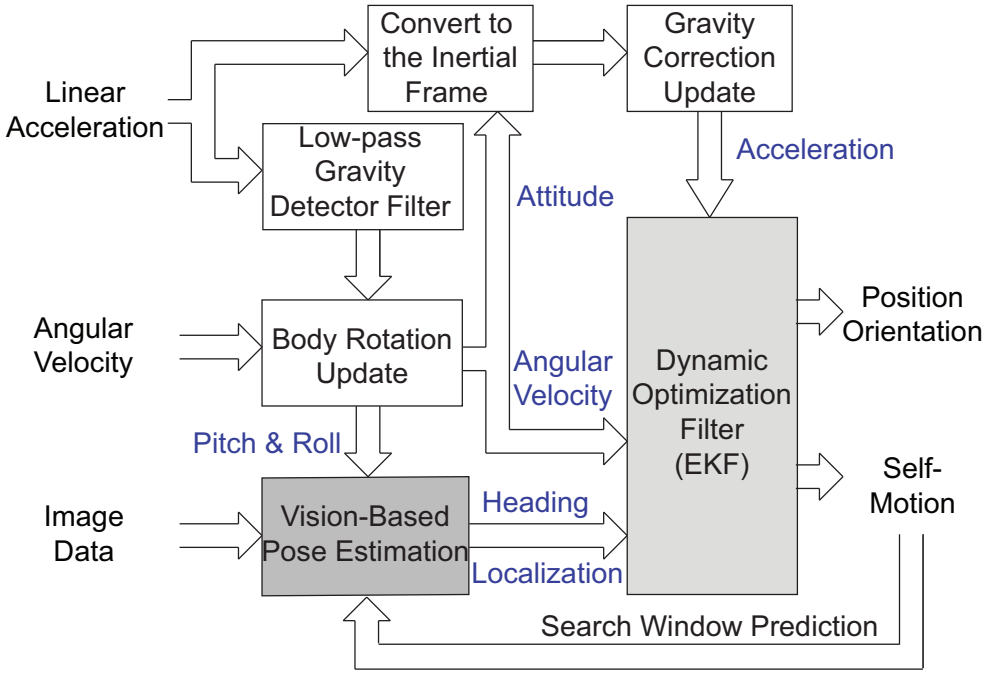

Fig. 3.5: Multi-sensory data processing and fusion.

3.3.1 IMU Information Processing

IMUs are one of the most commonly used on-board sensors for aerial vehicles. On the quadrotor used in this thesis, the on-board IMUs consist of three orthogonal accelerometers and three orthogonal gyroscopes. The former measure the linear acceleration of the quadrotor in the body frame, while the latter measure the angular velocity of the quadrotor in the body frame.

The framework of the strap-down inertial navigation system introduced in [47] is adopted. As shown in Fig. 3.5, the linear acceleration in the body frame is forwarded into a *Low-Pass* (LP) gravity detector filter. Using this filter, the sudden accelerations are eliminated and the gravity vector is detected. Together with angular velocity $[p \quad q \quad r]^T$ measured by the gyroscopes, the body rotation in the inertial frame is updated. Then, the pitch angle Θ_{imu} and the roll angle Φ_{imu} are obtained.

The on-board estimated angles Θ_{imu} and Φ_{imu} are not the same angles as the pitch and roll angles Θ and Φ defined in the previous section. After the LP filter, the force for compensating for the gravity in Z_I-/Z_o-direction is detected, which is denoted by its resulting acceleration ${}_I\boldsymbol{a}$:

$$ {}_I\boldsymbol{a} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}. \tag{3.31} $$

Furthermore, considering the coordinate transformation in Eq. 3.6, the projection of ${}_I\boldsymbol{a}$

into the quadrotor body frame can be formulated as follows:

$$
\begin{aligned}
{}_b\boldsymbol{a} = {}^b\boldsymbol{R}_I \cdot {}_I\boldsymbol{a} &= \begin{bmatrix} c\Theta c\Psi_{\text{imu}} & c\Theta s\Psi_{\text{imu}} & -s\Theta \\ s\Phi s\Theta c\Psi_{\text{imu}} - c\Phi s\Psi_{\text{imu}} & s\Phi s\Theta s\Psi_{\text{imu}} + c\Phi c\Psi_{\text{imu}} & s\Phi c\Theta \\ c\Phi s\Theta c\Psi + s\Phi s\Psi_{\text{imu}} & c\Phi s\Theta s\Psi_{\text{imu}} - s\Phi c\Psi_{\text{imu}} & c\Phi c\Theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \\
&= \begin{bmatrix} gs\Theta \\ -gs\Phi c\Theta \\ -gc\Phi c\Theta \end{bmatrix}. \qquad (3.32)
\end{aligned}
$$

Then, Ψ_{imu}, Θ_{imu}, and Φ_{imu} can be expressed by:

$$
\begin{aligned}
\Psi_{\text{imu}} &= \Psi + {}_I\Omega_r, \\
\Theta_{\text{imu}} &= \arcsin(\frac{[1\ 0\ 0] \cdot {}_b\boldsymbol{a}}{|{}_b\boldsymbol{a}|}) = \Theta, \\
\Phi_{\text{imu}} &= \arcsin(\frac{[0\ 1\ 0] \cdot {}_b\boldsymbol{a}}{|{}_b\boldsymbol{a}|}) = \arcsin(-s\Phi c\Theta). \qquad (3.33)
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
\Psi &= \Psi_{\text{imu}} - \Omega_r, \\
\Theta &= \Theta_{\text{imu}}, \\
\Phi &= \arcsin(-\frac{s\Phi_{\text{imu}}}{c\Theta_{\text{imu}}}). \qquad (3.34)
\end{aligned}
$$

The resulting roll angle Φ and pitch angle Θ are further forwarded as inputs for vision-based quadrotor pose estimation in Section 3.3.2. Ω_r can be initially ignored in the IMU measurements and will be corrected by vision data.

Having the attitude $[\Phi \quad \Theta \quad \Psi]^T$ and the linear acceleration $[a_x \quad a_y \quad a_z]^T$ measured by accelerometers in the body frame, the linear acceleration in the body frame can be computed according to Eq. 3.18. After a correction of gravity, the pure linear acceleration in the inertial frame is obtained. Linear velocity in the inertial frame is computed through integration of the acceleration. This high-frequency IMU-based estimation is used to drive the system process model in the EKF-aided data fusion.

To calculate the linear acceleration and angular velocity of the quadrotor in the object frame, the feedforward information from the ground robot is needed, which is considered in Chapter 4. In this chapter, it is regarded as noise. A correction based on vision information is applied and introduced in the next section.

3.3.2 Marker-Based Pose Estimation

To achieve a robust detection of the ground robot from the quadrotor's view, conventional feature detection is abandoned. Artificial markers are designed and mounted on the ground robot. The criteria for an appropriate marker are 1) simple and robust segmentation from its background; 2) independence from lighting conditions; 3) omni-directional detection; 4) low image processing cost.

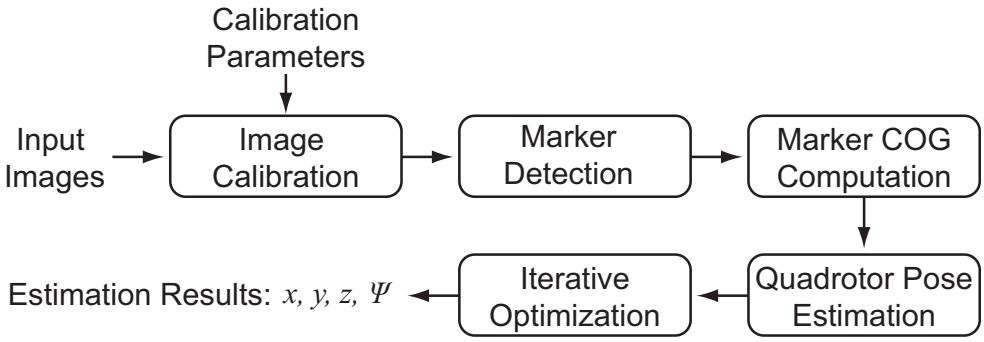

Fig. 3.6: Processing overview of the marker-based pose estimation.

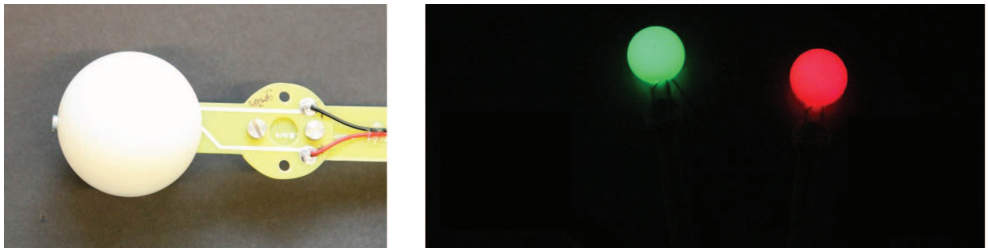

Fig. 3.7: Design of active markers (left) and a lightened view (right).

An overview of the marker-based pose estimation is illustrated in Fig. 3.6, consisting of image calibration using parameters based on a previous calibration process using *Camera Calibration Toolbox for Matlab*, marker detection, marker *Center Of Gravity* (COG) computation, quadrotor pose estimation, and iterative optimization.

Design of the Active Markers

Two active markers are mounted on a rigid frame (see Fig. 3.7). To be more independent from light conditions, the basic idea is to illuminate the markers as brightly as possible, while reducing the exposure time of the camera. This method reduces the influence of ambient light as well as distractors which might have the same color as one of the markers. During the initial experiments using high-power *Light Emitting Diodes* (LEDs), some problems occurred. When far from the camera, the LEDs appeared as only a few pixels, which made the recognition prone to error. When close to the camera, the LEDs were too bright, so the color information was lost. The small angles of beam of the LEDs also caused problems when the camera viewed the LEDs from the side. These problems are solved by using table-tennis balls which disperse the light of the LEDs diffusely. Two high-power surface mounted LEDs are placed on each side of a printed circuit board which is then inserted into the center of the table-tennis balls. As a result, a homogeneous "light bubble" can be recognized by the camera as a circle from all directions of view. Fig. 3.7 shows the design (left) and a lightened view of the markers (right).

Marker Detection

For each marker, its 2D image region and the COG are calculated for estimating the 3D reconstruction afterwards. The following steps are performed for each marker in each image.

First of all, the input image is transformed from RGB into YUV color space, which yields one luminance (Y) channel and two chrominance (U, V) channels. The advantage of the YUV color space is that the search for colors can be performed independently of lighting conditions. For each color, a minimum luminance $Y_{\min}$ and valid ranges for chrominance values $U_{\min}, U_{\max}, V_{\min}$, and $V_{\max}$ are determined. Each pixel that falls within these ranges during the search is called a *hit*.

To avoid errors caused by false positives, a robust filter is applied: Only the hits whose image coordinate ${}_i\boldsymbol{x} = ({}_ix, {}_iy)$ is close to the median of the coordinates will be considered in the calculation of the COG.

$$ {}_i\boldsymbol{x}_{\mathrm{COG}} = \begin{bmatrix} {}_ix_{\mathrm{COG}} \\ {}_iy_{\mathrm{COG}} \end{bmatrix} = \frac{1}{M} \sum_{j=1}^{M} {}_i\boldsymbol{x}_j \tag{3.35} $$

with

$$ |\mathrm{median}({}_ix) - {}_ix_j| < \epsilon \wedge |\mathrm{median}({}_iy) - {}_iy_j| < \epsilon, \tag{3.36} $$

where M denotes the number of hits that match the conditions and ${}_i\boldsymbol{x}_{\mathrm{COG}}$ the resulting COG. Once a valid COG is found, a *Region Of Interest* (ROI) is determined based on this position and the current motion estimate (see Fig. 3.5). The search in the next time step then only takes place in this region. This reduces computation time enormously and disturbances outside the ROI are automatically filtered out.

For the case that a marker is not detected completely, the COG of a marker shifts although the marker itself does not. Thus, the accuracy of recognizing the markers can be further improved by assuming a simple model for each marker: The markers are bullets, so a circle in the binary image gained from the thresholds above is expected. To reconstruct the real center of a marker using this model, three steps have to be performed (see Fig. 3.8): 1) *Correcting the foreground*: A combination of the morphological operations dilation and erosion (the morphological opening) with a 5×5 circle-structuring element is applied to the image. Gaps in the foreground will be closed, but the main structure will be kept approximately. 2) *Edge filtering*: To get only the border limiting the marker which a circle can be fitted into, edge filtering is done using a Laplace filter. 3) *Fitting the circle*: Now, the image is ready to fit a circle into the edges found in the step before. The equation of a circle in 2D can be written as: $({}_ix - {}_ix_0)^2 + ({}_iy - {}_iy_0)^2 = r^2$. The parameters to optimize are the circle center coordinate ${}_ix_0$, ${}_iy_0$ and the radius r. This nonlinear problem can be rewritten into a linear problem by a substitution:

$$ {}_ix^2 - 2_ix_i x_0 + {}_ix_0{}^2 + {}_iy^2 - 2_iy_i y_0 + {}_iy_0{}^2 = r^2 \tag{3.37} $$

$$ \Leftrightarrow \quad 2_ix\alpha + 2_iy\beta + \gamma = {}_ix^2 + {}_iy^2 \tag{3.38} $$

with

$$ {}_ix_0 = \alpha,\ {}_iy_0 = \beta,\ r^2 - {}_ix_0{}^2 - {}_iy_0{}^2 = \gamma. \tag{3.39} $$

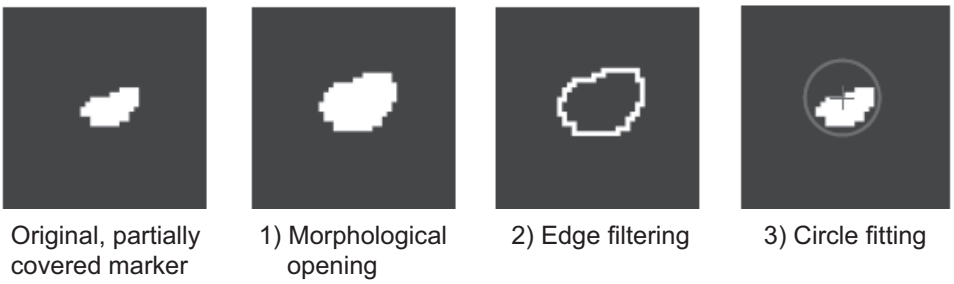

Fig. 3.8: Fitting the circle step-by-step.

The coordinates gained from the edge filtering above can now be inserted into Eq. 3.37, which leads to the following system of equations that can be solved by the method of linear least squares using the pseudo-inverse:

$$\begin{bmatrix} 2_i x_1 & 2_i y_1 & 1 \\ \vdots & \vdots & \vdots \\ 2_i x_m & 2_i y_m & 1 \end{bmatrix} \cdot \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} {}_i x_1^2 + {}_i y_1^2 \\ \vdots \\ {}_i x_m^2 + {}_i y_m^2 \end{bmatrix}, \tag{3.40}$$

where m denotes the number of the pixels on the edge. The center of the circle ${}_i x_0$ and ${}_i y_0$ as well as its radius r can now be derived from the substitution Eq. 3.39. The deviations of the center of the markers to the real centers are reduced significantly using this method, especially on partially covered markers.

3D Quadrotor Pose Estimation

As mentioned in Section 3.2, the position coordinates of the markers ${}_o\boldsymbol{p}_{1/2}$ in the object frame $\mathbf{S}_o$ have the following values:

$${}_o\boldsymbol{p}_1 = \begin{bmatrix} 0 \\ -\delta/2 \\ -2\,\mathrm{cm} \end{bmatrix} \quad \text{and} \quad {}_o\boldsymbol{p}_2 = \begin{bmatrix} 0 \\ \delta/2 \\ -2\,\mathrm{cm} \end{bmatrix}. \tag{3.41}$$

The transformation of marker positions ${}_o\boldsymbol{p}_{1/2}$ from the object frame $\mathbf{S}_o$ via the quadrotor body frame $\mathbf{S}_b$ to the camera frame $\mathbf{S}_c$ can be formalized as follows:

$$\begin{aligned} {}_c\boldsymbol{p'}_{1/2} &= {}^c\boldsymbol{T}_b \cdot {}_b\boldsymbol{p'}_{1/2} \\ &= {}^c\boldsymbol{T}_b \cdot {}^b T_o \cdot {}_o\boldsymbol{p'}_{1/2}, \end{aligned} \tag{3.42}$$

with $(\cdot)'$ indicating a homogeneous coordinate, where

$${}^b\boldsymbol{T}_o = \begin{bmatrix} {}^b\boldsymbol{R}_o & {}_o\boldsymbol{p}_q \\ 0 & 1 \end{bmatrix}, \tag{3.43}$$

The positions of the markers in the camera frame are denoted by ${}_c\boldsymbol{p}_1 = [{}_c x_1 \;\; {}_c y_1 \;\; {}_c z_1]^T$ and ${}_c\boldsymbol{p}_2 = [{}_c x_2 \;\; {}_c y_2 \;\; {}_c z_2]^T$. The matrix ${}^b\boldsymbol{R}_o$ indicates the rotation matrix from the object frame $\mathbf{S}_o$ to the quadrotor body frame $\mathbf{S}_b$, and can be computed using the yaw, pitch, and

roll angles Ψ, Θ, and Φ based on Eq. 3.7. The yaw angle is computed using an iterative estimation algorithm introduced later. The other two angles are obtained from IMU data mentioned in Eq. 3.33 in Section 3.3.1. The matrix ${}^{c}\boldsymbol{T}_{b}$ is the homogeneous transformation matrix from the quadrotor body frame $\mathbf{S}_b$ to the camera frame $\mathbf{S}_c$ and is constant. The position of the quadrotor ${}_{o}\boldsymbol{p}_{q} = [{}_{o}x_{q} \;\; {}_{o}y_{q} \;\; {}_{o}z_{q}]^T$ with respect to the ground robot is to be determined.

From the visual information, the 2D positions of the markers in the image frame ${}_{i}\boldsymbol{p}_{1} = [{}_{i}x_{1} \;\; {}_{i}y_{1}]^T$ and ${}_{i}\boldsymbol{p}_{2} = [{}_{i}x_{2} \;\; {}_{i}y_{2}]^T$ are computed using the algorithms introduced in the previous section. According to the pinhole camera model with a focal length f, the following relation can be easily derived:

$$ {}_{i}\boldsymbol{p}_{1/2} = [\frac{{}_{c}x_{1/2}}{{}_{c}z_{1/2}}f \;\; \frac{{}_{c}y_{1/2}}{{}_{c}z_{1/2}}f]^T. \tag{3.44} $$

Substituting ${}_{c}\boldsymbol{p}_{1/2} = [{}_{c}x_{1/2} \;\; {}_{c}y_{1/2} \;\; {}_{c}z_{1/2}]^T$ in (3.42) and (3.44), four equations for three unknown variables of ${}_{o}\boldsymbol{p}_{q}$ are obtained. Using the least squares method, ${}_{o}\boldsymbol{p}_{q}$ is found and used in the iterative optimization of the pose estimation result later.

For the experimental evaluation, the absolute quadrotor position ${}_{I}\boldsymbol{p}_{q}$ is computed as follows:

$$ {}_{I}\boldsymbol{p'}_{q} = {}^{I}\boldsymbol{T}_{o} \cdot {}_{o}\boldsymbol{p'}_{q}, \tag{3.45} $$

where ${}^{I}\boldsymbol{T}_{o}$ is obtained using the ground robot pose feedback with respect to the inertial frame $\boldsymbol{S}_I$ based on Eq. 3.1.

Iterative Optimization

To obtain an accurate quadrotor yaw angle Ψ and position $[x \;\; y \;\; z]^T$ based on the 2D marker positions, the gradient descent algorithm is used to iteratively estimate and optimize Ψ, x, y, and z.

Firstly, a prior yaw angle Ψ^{-} is computed using image data containing the 2D marker positions ${}_{i}\boldsymbol{p}_{1/2} = [{}_{i}x_{1/2} \;\; {}_{i}y_{1/2}]^T$ as follows:

$$ \Psi^{-} = \operatorname{atan2}\big(({}_{i}x_{2} - {}_{i}x_{1}), ({}_{i}y_{2} - {}_{i}y_{1})\big), \tag{3.46} $$

Then, using the pitch and roll angle Θ and Φ as well as Ψ^{-}, a prior rotation matrix ${}^{b}\boldsymbol{R}_{o}^{-}$ and a prior quadrotor body position ${}_{o}\boldsymbol{p}_{q}^{-}$ are calculated as described above.

After that, ${}^{b}\boldsymbol{R}_{o}^{-}$ and ${}_{o}\boldsymbol{p}_{q}^{-}$ are used to compute 2D marker positions ${}_{i}\boldsymbol{p}_{1/2}^{-}$. Then, ${}_{i}\boldsymbol{p}_{1/2}^{-}$ is compared with ${}_{i}\boldsymbol{p}_{1/2}$. The gradient descent algorithm is used here to find an optimal Ψ^{*} iteratively, until ${}_{i}\boldsymbol{p}_{1/2}^{-} \approx {}_{i}\boldsymbol{p}_{1/2}$. Concurrently, the quadrotor position ${}_{o}\boldsymbol{p}_{q}$ is optimized as well.

It is worth mentioning that the iterative optimization method used here is very computationally efficient and, therefore, does not impair the total computational performance.

3.3.3 Optical-Flow-Based Motion Estimation

During quadrotor take-off and landing, there are phases in which the quadrotor is near the surface of the mobile ground robot. For the case that the markers cannot be completely

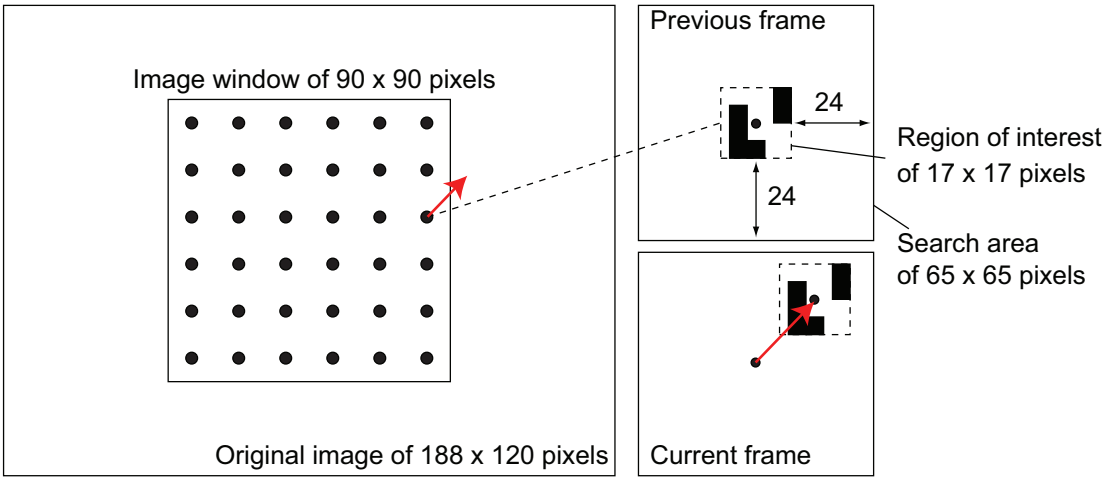

Fig. 3.9: Sketch of points of interest selection.

and robustly detected in the camera field of view, optical-flow-based motion estimation is applied. Then, the quadrotor pose is estimated by integration of the motion estimation. Two steps are contained: 1) computation of optical flow; 2) motion reconstruction based on the optical flow. In this section, *Pyramidal implementation of the Lucas-Kanade algorithm* (PLK) [38] in combination with the image Jacobian matrix and the least squares optimization method are applied to set up a visual odometry system and estimate the camera as well as the quadrotor ego-motion.

Optical Flow Algorithms

The Lucas-Kanade algorithm is a classic differential optical flow technique [89]. An improved version PLK can work robustly and efficiently to compute optical flow. One advantage of the PLK algorithm is its ability to handle a large pixel motion exactly with local sub-pixel accuracy.

First, image points of interest are selected as follows. The size of images is 188×120 pixels and an image window of 90×90 pixels around the principal point is selected. In this image window, 36 image points of interest are homogeneously selected. Around each image point of interest, an image ROI of 17×17 pixels is constructed. The optical flow for this image point of interest is calculated by matching the image ROI of the previous frame in an extended search window containing 24 more pixels in both directions in the current frame. It can be considered in a way that optical flows of 36 points of interest have been computed in total with a better accuracy. The search window is set relatively large, since if the quadrotor is near the surface of the ground robot, the field of view is limited and the image contrast is not sufficient. Outlier cancellation based on the *RANdom SAmple Consensus* (RANSAC) algorithm is applied to select 24 final points of interest.

The camera frame rate is 60 Hz. However, due to the computational time of PLK algorithm, a total rate of 15 Hz is obtained. The last two consecutive images at 60 Hz of an image sequence of 4 images are used to compute the optical flow. The maximum inter-frame motion is 24 pixels/frame, and the maximum detectable quadrotor motion in

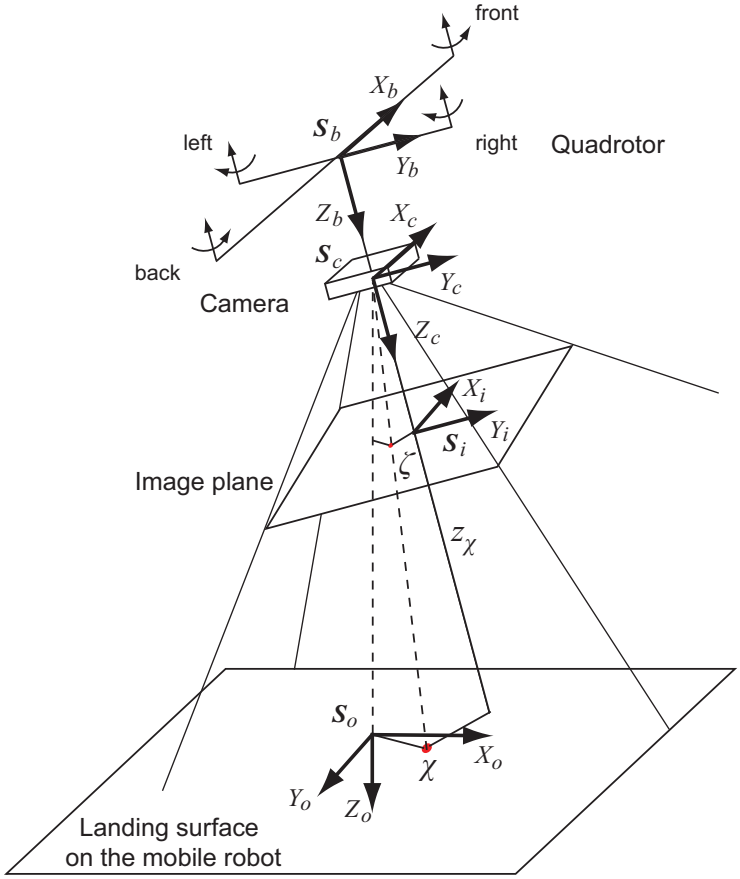

Fig. 3.10: Feature point projection.

X_I-/Y_I-directions at the lowest altitude is approximately 1 m/s.

Motion Estimation

The PLK optical flow algorithm determines the optical flow for predefined image points of interest and the camera ego-motion can be computed by using the image Jacobian matrix J, which combines the motion of a 3D point χ on the landing surface of the mobile robot in the camera frame and its projection point ζ in the image plane (illustrated in Fig. 3.10) as follows [75]:

$$\begin{bmatrix} {}_i\dot{x}_\zeta \\ {}_i\dot{y}_\zeta \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{f}{z_\chi} & 0 & -\frac{{}_ix_\zeta}{z_\chi} & -\frac{{}_ix_\zeta \cdot {}_iy_\zeta}{f} & \frac{f^2+{}_ix_\zeta^2}{f} & -{}_iy_\zeta \\ 0 & \frac{f}{z_\chi} & -\frac{{}_iy_\zeta}{z_\chi} & -\frac{f^2+{}_iy_\zeta^2}{f} & \frac{{}_ix_\zeta \cdot {}_iy_\zeta}{f} & {}_ix_\zeta \end{bmatrix}}_{\boldsymbol{J}} \cdot \begin{bmatrix} T_x \\ T_y \\ T_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}, \tag{3.47}$$

where $[{}_i\dot{x}_\zeta \;\; {}_i\dot{y}_\zeta]^T$ denotes the motion of the point ζ in the image plane and f is the focal length. z_χ is the perpendicular distance between the 3D point χ and the camera center

along the optical axis as follows:

$$z_\chi = [0 \;\; 0 \;\; 1] \cdot {}_c\boldsymbol{p}_\chi \tag{3.48}$$

with the 3D coordinate of χ in the camera frame ${}_c\boldsymbol{p}_\chi$. The relative motion of the image point χ with respect to the camera is represented by $[T_x \;\; T_y \;\; T_z \;\; \omega_x \;\; \omega_y \;\; \omega_z]^T$.

Since the image points of interest are predefined and f is constant, in order to compute the relative motion between the camera and the landing surface, z_χ should be determined for every image point of interest. The computation of z_χ is based on the rules of similar triangles as follows.

The position of the point ζ with respect to the camera frame can be written

$${}_c\boldsymbol{p}_\zeta = [{}_ix_\zeta \;\; {}_iy_\zeta \;\; f]^T. \tag{3.49}$$

According to the similar triangle rules, the following equations are built:

$$\frac{[0 \;\; 0 \;\; 1] \cdot {}^o\boldsymbol{R}_c \cdot {}_c\boldsymbol{p}_\zeta}{[0 \;\; 0 \;\; 1] \cdot {}_o\boldsymbol{p}_c} = \frac{\sqrt{{}_c\boldsymbol{p}_\zeta{}^T \cdot {}_c\boldsymbol{p}_\zeta}}{\sqrt{{}_c\boldsymbol{p}_\chi{}^T \cdot {}_c\boldsymbol{p}_\chi}} \tag{3.50}$$

and

$$\frac{f}{z_\chi} = \frac{\sqrt{{}_c\boldsymbol{p}_\zeta{}^T \cdot {}_c\boldsymbol{p}_\zeta}}{\sqrt{{}_c\boldsymbol{p}_\chi{}^T \cdot {}_c\boldsymbol{p}_\chi}}, \tag{3.51}$$

where ${}_c\boldsymbol{p}_\chi$ is the position of χ in the camera frame and the multiplication with $[0 \;\; 0 \;\; 1]$ selects the z-component of a position vector.

By combining Eq. 3.50 and 3.51, z_χ can be solved as

$$z_\chi = f \cdot \frac{\sqrt{{}_c\boldsymbol{p}_\chi{}^T \cdot {}_c\boldsymbol{p}_\chi}}{\sqrt{{}_c\boldsymbol{p}_\zeta{}^T \cdot {}_c\boldsymbol{p}_\zeta}} = f \cdot \frac{[0 \;\; 0 \;\; 1] \cdot {}_o\boldsymbol{p}_c}{[0 \;\; 0 \;\; 1] \cdot {}^o\boldsymbol{R}_c \cdot {}_c\boldsymbol{p}_\zeta}. \tag{3.52}$$

The relationship ${}^c\boldsymbol{T}_q$ between the camera frame and the quadrotor frame can be derived from system calibration. Here based on reasonable assumption, ${}^o\boldsymbol{R}_c$ and ${}^o\boldsymbol{R}_b$ are assumed to be identical. Then, Eq. 3.52 is rewritten as

$$z_\chi = \frac{f \cdot ([0 \;\; 0 \;\; 1] \cdot {}_o\boldsymbol{p}_c)}{-{}_ix_\zeta \cdot s\Theta + {}_iy_\zeta \cdot s\Phi \cdot c\Theta + f \cdot c\Phi \cdot c\Theta}. \tag{3.53}$$

The optical flow on 24 image points of interest χ_1 to χ_{24} and their projections ζ_1 to ζ_{24}

will be calculated and an overdetermined system of linear equations is obtained :

$$\begin{bmatrix} {}_i\dot{x}_{\zeta_1} \\ {}_i\dot{y}_{\zeta_1} \\ \vdots \\ {}_i\dot{x}_{\zeta_{24}} \\ {}_i\dot{x}_{\zeta_{24}} \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{f}{z_{\chi_1}} & 0 & -\frac{{}_ix_{\zeta_1}}{z_{\chi_1}} & -\frac{{}_ix_{\zeta_1}\cdot{}_iy_{\zeta_1}}{f} & \frac{f^2+{}_ix_{\zeta_1}^2}{f} & -{}_iy_{\zeta_1} \\ 0 & \frac{f}{z_{\chi_1}} & -\frac{{}_iy_{\zeta_1}}{z_{\chi_1}} & -\frac{f^2+{}_iy_{\zeta_1}^2}{f} & \frac{{}_ix_{\zeta_1}\cdot{}_iy_{\zeta_1}}{f} & {}_ix_{\zeta_1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{f}{z_{\chi_{24}}} & 0 & -\frac{{}_ix_{\zeta_{24}}}{z_{\chi_{24}}} & -\frac{{}_ix_{\zeta_{24}}\cdot{}_iy_{\zeta_{24}}}{f} & \frac{f^2+{}_ix_{\zeta_{24}}^2}{f} & -{}_iy_{\zeta_{24}} \\ 0 & \frac{f}{z_{\chi_{24}}} & -\frac{{}_iy_{\zeta_{24}}}{z_{\chi_{24}}} & -\frac{f^2+{}_iy_{\zeta_{24}}^2}{f} & \frac{{}_ix_{\zeta_{24}}\cdot{}_iy_{\zeta_{24}}}{f} & {}_ix_{\zeta 24} \end{bmatrix}}_{\boldsymbol{J}_{24}} \cdot \begin{bmatrix} T_x \\ T_y \\ T_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}. \tag{3.54}$$

where an image Jacobian matrix $\boldsymbol{J}_{24}$ of 48×6 for 24 image points of interest is built. Then the relative motion of the camera with respect to the camera frame can be solved by

$$-\begin{bmatrix} T_x \\ T_y \\ T_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = -(\boldsymbol{J_{24}}^T \cdot \boldsymbol{J_{24}})^{-1} \cdot \boldsymbol{J_{24}}^T \cdot \begin{bmatrix} {}_i\dot{x}_{\zeta_1} \\ {}_i\dot{y}_{\zeta_1} \\ \vdots \\ {}_i\dot{x}_{\zeta_{24}} \\ {}_i\dot{x}_{\zeta_{24}} \end{bmatrix}. \tag{3.55}$$

Since ${}^o\boldsymbol{R}_c$ and ${}^o\boldsymbol{R}_b$ are assumed to be identical, the angular motion of the camera and quadrotor are also identical. The linear motion of the quadrotor ${}_o\dot{\boldsymbol{p}}_q = [\dot{x} \quad \dot{y} \quad \dot{z}]^T$ is expressed by

$${}_o\dot{\boldsymbol{p}}_q = {}^o\boldsymbol{R}_b \cdot {}_b\dot{\boldsymbol{p}}_q = {}^o\boldsymbol{R}_b \cdot \left(-\begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times {}_b\boldsymbol{p}_c \right), \tag{3.56}$$

where ${}_b\boldsymbol{p}_c$ denotes the camera position in the body frame.

Then, the position of the quadrotor in the object frame at time t from time t_0 is computed by

$${}_o\boldsymbol{p}_q(t) = {}_o\boldsymbol{p}_q(t_0) + \int_{t_0}^{t} {}_o\dot{\boldsymbol{p}}_q(t) \cdot dt. \tag{3.57}$$

3.3.4 Data Fusion Using a Continuous-Discrete EKF

To achieve high-frequency and accurate pose/motion estimations, an EKF is applied for multi-sensory data fusion. The most common form of KF is the discrete-time KF. But the IMU data and images have different data rates and, therefore, arrive asynchronously. Another unusual form, namely continuous-discrete KF, mentioned in [114], is more convenient than the former one. The prediction of the current system state seems to be continuous, using the IMU data arriving at a higher rate than images to drive the system model. The marker-based pose estimation is used in the stage of correction and in the discrete form at a lower frame rate, as it is synchronized with the latest IMU data.

Note that optical-flow-based motion estimation is used for the case that the quadrotor

is near the ground robot and cannot robustly and completely detect the markers. Since the quadrotor pose is computed based on motion integration, drift also occurs, which is however smaller than IMU drift. Therefore, only the marker-based pose information is fused with IMU data, providing a reasonable correction.

In the data fusion using a continuous-discrete EKF, the predicted, a priori estimates are denoted by "$\hat{(\cdot)}^-$", while the updated, a posteriori estimates are denoted by "$\hat{(\cdot)}$".

Process Model

The system state $\boldsymbol{x} \in \mathbb{R}^n$, where $n = 9$, contains the quadrotor position and linear velocity in the object frame as well as the quadrotor orientation represented by the roll, pitch, and yaw angles:

$$\boldsymbol{x} = [x \;\; y \;\; z \;\; \dot{x} \;\; \dot{y} \;\; \dot{z} \;\; \Phi \;\; \Theta \;\; \Psi]^T. \tag{3.58}$$

The system is a non-linear process. According to Eqs. 3.22, 3.23, and 3.30, the system process equation can be formulated as follows:

$$\begin{aligned}
\dot{\boldsymbol{x}} &= [\dot{x} \;\; \dot{y} \;\; \dot{z} \;\; \ddot{x} \;\; \ddot{y} \;\; \ddot{z} \;\; \dot{\Phi} \;\; \dot{\Theta} \;\; \dot{\Psi}]^T \\
&= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\omega}) \\
&= \begin{bmatrix}
\dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x}_{\text{trans}} + \ddot{x}_{\text{rot}} \\ \ddot{y}_{\text{trans}} + \ddot{y}_{\text{rot}} \\ -s\Theta \cdot a_x + s\Phi c\Theta \cdot a_y + c\Phi c\Theta \cdot a_z + g \\ p + s\Phi t\Theta \cdot q + c\Phi t\Theta \cdot r \\ c\Phi \cdot q - s\Phi \cdot r \\ \frac{s\Phi}{c\Theta} \cdot q + \frac{c\Phi}{c\Theta} \cdot r - \dot{\Omega}_r
\end{bmatrix} + \boldsymbol{\omega},
\end{aligned} \tag{3.59}$$

where $\ddot{x}_{\text{trans}}$, $\ddot{x}_{\text{rot}}$, $\ddot{y}_{\text{trans}}$, and $\ddot{y}_{\text{rot}}$ are defined in Eq. 3.23, $\boldsymbol{u}$ denotes the measurement input from the high-frequency IMU data and the feed-forward information from the ground robot given by

$$\boldsymbol{u} = \left[a_x \;\; a_y \;\; a_z \;\; p \;\; q \;\; r \;\; a_r \;\; \dot{\Omega}_r \;\; \ddot{\Omega}_r\right]^T, \tag{3.60}$$

and $\boldsymbol{\omega}$ denotes the process noise, which is assumed to be a zero mean multi-variant normal distribution with covariance $\boldsymbol{Q}$.

The system equation above can be linearized as follows:

$$\boldsymbol{x}(t) \approx \hat{\boldsymbol{x}}(t)^- + \boldsymbol{F} \cdot (\boldsymbol{x}(t - \Delta t) - \hat{\boldsymbol{x}}(t - \Delta t)) \tag{3.61}$$

where $\boldsymbol{x}(t)$ denotes the actual state vector at time t, $\hat{\boldsymbol{x}}(t)^-$ denotes the approximate state computed using Eq. 3.59, and $\hat{\boldsymbol{x}}(t - \Delta t)$ is an a posteriori state estimate after the EKF processing at time $t - \Delta t$. The system matrix $\boldsymbol{F}$ is the Jacobian matrix of partial derivatives

of $\boldsymbol{f}$ with respect to $\boldsymbol{x}$ and can be formulated as follows:

$$\begin{aligned}\boldsymbol{F}_{[j,l,t]} &= \frac{\partial \boldsymbol{f}_{[j]}}{\partial \boldsymbol{x}_{[l]}}(\hat{\boldsymbol{x}}(t-\Delta t), \boldsymbol{u}(t-\Delta t), 0) \\ &= \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \dot{\Omega}_r^2 & \ddot{\Omega}_r & 0 & 0 & 2\dot{\Omega}_r & 0 & a_1 & a_2 & a_3 \\ -\ddot{\Omega}_r & \dot{\Omega}_r^2 & 0 & -2\dot{\Omega}_r & 0 & 0 & a_4 & a_5 & a_6 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_7 & a_8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_9 & a_{10} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{11} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{12} & a_{13} & 0 \end{bmatrix},\end{aligned} \tag{3.62}$$

where $j, l \in \{1, \cdots, 9\}$ and a_1 to a_{13} are calculated as follows:

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{bmatrix} = \begin{bmatrix} 0 & c\Phi s\Theta c\Psi + s\Phi s\Psi & -s\Phi s\Theta c\Psi + c\Phi s\Psi \\ -s\Theta c\Psi & s\Phi c\Theta c\Psi & c\Phi c\Theta c\Psi \\ -c\Theta s\Psi & -(s\Phi s\Theta s\Psi + c\Phi c\Psi) & -c\Phi s\Theta s\Psi + s\Phi c\Psi \\ 0 & c\Phi s\Theta s\Psi - s\Phi c\Psi & -(s\Phi s\Theta s\Psi + c\Phi c\Psi) \\ -s\Theta s\Psi & s\Phi c\Theta s\Psi & c\Phi c\Theta s\Psi \\ c\Theta c\Psi & s\Phi s\Theta c\Psi - c\Phi s\Psi & c\Phi s\Theta c\Psi + s\Phi s\Psi \\ 0 & c\Phi c\Theta & -s\Phi c\Theta \\ -c\Theta & -s\Phi s\Theta & -c\Phi s\Theta \end{bmatrix} \cdot \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \tag{3.63}$$

and

$$\begin{bmatrix} a_9 \\ a_{10} \\ a_{11} \\ a_{12} \\ a_{13} \end{bmatrix} = \begin{bmatrix} 0 & c\Phi t\Theta & -s\Phi t\Theta \\ 0 & \frac{s\Phi}{(c\Theta)^2} & \frac{c\Phi}{(c\Theta)^2} \\ 0 & -s\Phi & -c\Phi \\ 0 & \frac{c\Phi}{c\Theta} & -\frac{s\Phi}{c\Theta} \\ 0 & \frac{s\Phi}{c\Theta}t\Theta & \frac{c\Phi}{c\Theta}t\Theta \end{bmatrix} \cdot \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{3.64}$$

Measurement Model

The marker-based pose estimation provides the measurements of quadrotor position and its yaw angle without drifts in the object frame $\boldsymbol{S}_o$ at a frame rate of 15 Hz. Then, the system measurement is formulated as follows:

$$\boldsymbol{z} = \begin{bmatrix} {}_o\boldsymbol{p}_q \\ \Psi \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \Psi \end{bmatrix}, \tag{3.65}$$

and the system observation equation is formulated as a discrete system given by

$$\boldsymbol{z}_k = \boldsymbol{H} \cdot \boldsymbol{x}(t_k) + \boldsymbol{\nu}_k \tag{3.66}$$

with k denoting the kth measurement, $\boldsymbol{\nu}$ denoting the measurement noise, which is assumed to be a zero mean multivariate normal distribution with covariance $\boldsymbol{R}$, and the measurement transition matrix

$$\boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.67}$$

Continuous-Discrete Data Fusion

The system state is continuously predicted first:

$$\hat{\boldsymbol{x}}(t + \triangle t)^- \approx \hat{\boldsymbol{x}}(t)^- + \dot{\hat{\boldsymbol{x}}}(t)^- \cdot \triangle t, \tag{3.68}$$

where

$$\dot{\hat{\boldsymbol{x}}}(t)^- = \boldsymbol{f}(\hat{\boldsymbol{x}}(t)^-, \boldsymbol{u}(t), \boldsymbol{0}) \tag{3.69}$$

and

$$\triangle t \to 0, \tag{3.70}$$

The new state error covariance matrix $\boldsymbol{P}$ [1] is given by

$$\boldsymbol{P}(t + \triangle t)^- = \boldsymbol{P}(t)^- + (\boldsymbol{F}(t)\boldsymbol{P}(t)^- + \boldsymbol{P}(t)^-\boldsymbol{F}^T(t) + \boldsymbol{Q}) \cdot \triangle t. \tag{3.71}$$

To reduce the error caused by linearization of the non-linear system model, the update period of the IMU data T_{imu} [2] is divided into N identical time intervals. Let

$$\triangle t = \frac{T_{\text{imu}}}{N}. \tag{3.72}$$

For newly arriving IMU data, the prediction process in time $\triangle t$ will be iteratively carried out N times. For instance, if the refreshing period of the IMU data T_{imu} is about 10 ms and the iteration number N is set to 100, the prediction period is about 0.1 ms. By using this technique, remarkable errors caused by linearization can be avoided and the remaining inaccuracy remains negligible.

For the correction stage, the predicted system state is updated with the time-discrete measurements $\boldsymbol{z}_k$:

$$\boldsymbol{L}_k = \boldsymbol{P}(t_k)^-\boldsymbol{H}^T(\boldsymbol{R} + \boldsymbol{H}\boldsymbol{P}(t_k)^-\boldsymbol{H}^T)^{-1}, \tag{3.73}$$

$$\boldsymbol{P}(t_k) = (\boldsymbol{E} - \boldsymbol{L}_k\boldsymbol{H})\boldsymbol{P}(t_k)^-, \tag{3.74}$$

$$\hat{\boldsymbol{x}}(t_k) = \hat{\boldsymbol{x}}(t_k)^- + \boldsymbol{L}_k(\boldsymbol{z}_k - \boldsymbol{H}\hat{\boldsymbol{x}}(t_k)^-), \tag{3.75}$$

[1] $\boldsymbol{P}$ is symmetrical and positive definite

[2] T must not be constant

where $\mathbf{L}$ is the Kalman gain, k is the index of the kth measurement, t_k indicates the corresponding system time for the kth iteration, and $\boldsymbol{E}$ denotes a unit matrix.

Noise Investigation

The derivation of the EKF is based on white Gaussian system and measurement disturbance, but the main error source in this work is the sensor error of the IMUs, which does not fulfill this precondition. In [100] the sensor model of an accelerometer[3] is given by[4]:

$$z = \tilde{z} + \rho \cdot \tilde{z} + \sigma + \omega_r, \tag{3.76}$$

where z is the sensor output, $\tilde{z}$ is the actual acceleration (the ground truth), ρ is the scale factor [5], σ is the measurement bias or zero offset, and ω_r is the random noise. The sensor error e_z can then be expressed by

$$e_z = z - \tilde{z} = \rho \cdot \tilde{z} + \sigma + \omega_r. \tag{3.77}$$

This sensor error is colored non-Gaussian measurement noise and cannot be observed by a noise shaping filter [86].

Another non-negligible error source is the estimation error of the roll and pitch angles, which is also non-Gaussian.

Fortunately, as mentioned in [95] and [31], the EKF can still be applied in such cases with non-white non-Gaussian disturbance and give sensible results, but the covariance matrices Q and R should be tuned carefully.

Some criteria for the parameter tuning of the EKF are introduced in [18]. Since the state of the quadrotor cannot be fully observed by the available equipment, the generative approach, maximizing the joint likelihood of all the data, is not suited. To get a more accurate estimation of the states, the criterion minimizing the residual prediction error for the external ground truth is deployed.

This technique searches for the covariance matrices $\mathbf{Q}$ and $\mathbf{R}$, which minimize the quadratic error between the ground truth and the estimates:

$$\langle \boldsymbol{Q}, \boldsymbol{R} \rangle = \underset{\boldsymbol{Q},\boldsymbol{R}}{\operatorname{argmin}} \sum_{j=0}^{K} (\tilde{\boldsymbol{z}}_j - \boldsymbol{H}\hat{\boldsymbol{x}}(t_j))^T \boldsymbol{G} (\tilde{\boldsymbol{z}}_j - \boldsymbol{H}\hat{\boldsymbol{x}}(t_j)), \tag{3.78}$$

where $\tilde{\boldsymbol{z}} = [\tilde{x} \;\; \tilde{y} \;\; \tilde{z} \;\; \tilde{\Psi}]^T$ denotes the ground truth, K indicates the number of the involved measurements, and $\boldsymbol{G}$ is the weighting matrix[6] and determines the importance of every single element in $\tilde{\boldsymbol{z}}$.

Assuming that all external measurements have about the same accuracy, giving the

[3] Gyroscopes have the similar model.
[4] Only the significant terms are considered.
[5] ρ is usually expressed in polynomial form to include non-linear effects.
[6] Or inverse of the noise covariance matrix of the external measurements, which are taken as ground truth.

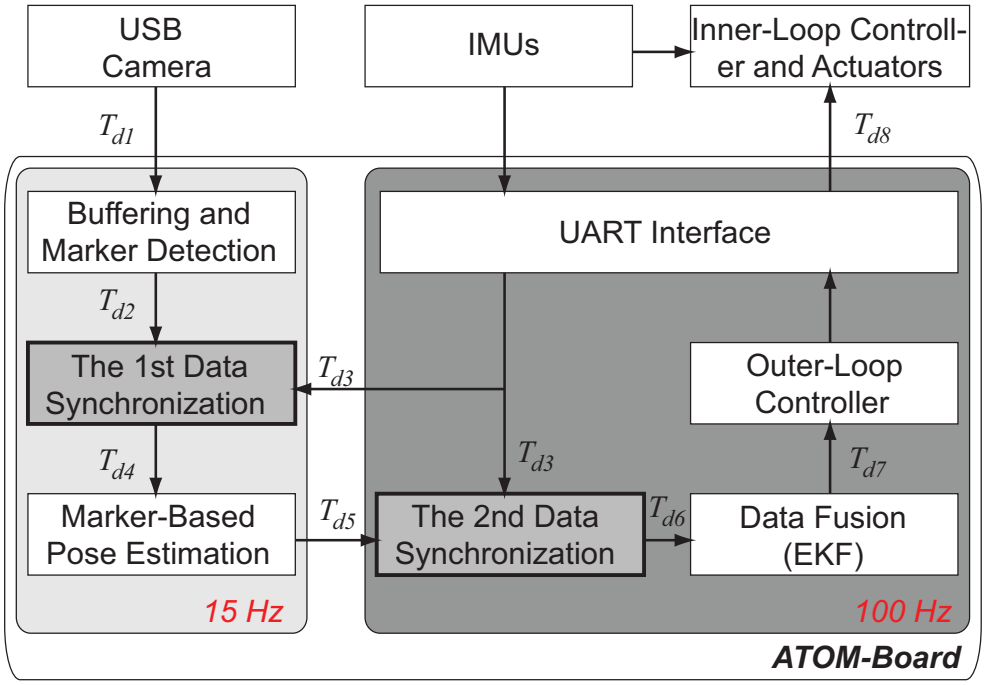

Fig. 3.11: The processing frequency in each system block and the overall system time delay, listed in Tab. 3.1. The measurement details of the time delay are shown in Appendix A.

identical weighting factor, Eq. 3.78 is simplified to:

$$\langle \boldsymbol{Q}, \boldsymbol{R} \rangle = \underset{\boldsymbol{Q},\boldsymbol{R}}{\operatorname{argmin}} \sum_{j=0}^{K} (\tilde{\boldsymbol{z}}_j - \boldsymbol{H}\hat{\boldsymbol{x}}(t_j))^T (\tilde{\boldsymbol{z}}_j - \boldsymbol{H}\hat{\boldsymbol{x}}(t_j)). \tag{3.79}$$

In Section 3.6, Q and R are experimentally determined to minimize the error between the ground truth and the estimated values.

3.3.5 Synchronization Aspect

The IMU and vision data are provided by different sensors. They are properly asynchronous, making accurate pose and motion estimation impossible. To deal with this problem, the IMU and vision data should be synchronized with each other. The time delays of sensor data and data transfer are shown in Fig. 3.11 and Tab. 3.1. The measuring procedure is described in Appendix A.

As shown in Fig. 3.11, the multi-sensor data should be synchronized at two points: synchronization before marker-based pose estimation and synchronization before EKF-aided data fusion.

The 1st Data Synchronization

To carry out marker-based pose estimation, marker positions in images from the camera and on-board estimated Φ_{imu} and Θ_{imu} angles are required. Here, the IMU data are

Symbol	Data	Time Delay
T_{d1}	Vision data	≈20 ms
T_{d2}	2D marker position	≈80 ms
T_{d3}	IMU data, Φ_{imu}, Θ_{imu}	≈30 ms
T_{d4}	Synchronized data	≈80 ms
T_{d5}	Pose estimates	≈90 ms
T_{d6}	Synchronized data	≈40 ms
T_{d7}	Fused data	≈65 ms
T_{d8}	Control commands	≈90 ms

Tab. 3.1: Detailed data transfer and time delay at each step according to Fig. 3.11.

predicted or interpolated to synchronize with the vision data, in order to obtain a smoother marker-based motion estimation.

The camera samples current marker positions every 16.67 ms[7] and the shutter is then opened for the desired exposure time. After preparation for transfer, the captured image is sent to the ATOM-board. To avoid unexpected expired images, the buffer on the ATOM-board is set to get data on demand – until the system requires new images, the buffer would not receive data. Data in the buffer are fetched by the image processing program a little later after arriving. For detection of the markers in the image, complex computation has to be executed, which takes a relatively long time, resulting in a total time delay T_{d2} of approximately 80 ms.

The IMUs on the quadrotor measure the accelerations and rotation velocities. These data are used to estimate the Φ_{imu} and Θ_{imu} angles. The IMU data at an appropriate moment are picked up as soon as the 2D marker position is achieved. The data are synchronized according to the image data. Therefore, $T_{d4} \approx T_{d2}$.

The actual time labels[8] T_v of the vision data and T_e of the estimated Φ_{imu} and Θ_{imu} can be calculated by considering the processes described above and the current time label. In fact, the angle estimations arrive mostly with less time delay, so it is more reasonable to synchronize the angles with the vision data. Then, more interpolations are carried out than predictions, which are less accurate.

A buffer is used to save the last L angle estimations based on IMU data and the time label of the estimation is $T_{e,l}$, $l \in \mathbb{N}$ and $1 \leq l \leq L$. L is chosen large enough, so that $T_{e,1} \leq T_v$.

- If $T_v \geq T_{e,k}$ and $T_v < T_{e,k+1}$, where $1 \leq k \leq L-1$, then interpolation is conducted as follows:

$$\Phi(T_v) = \Phi(T_{e,k}) + \frac{\Phi(T_{e,k+1}) - \Phi(T_{e,k})}{T_{e,k+1} - T_{e,k}} \cdot (T_v - T_{e,k}) \tag{3.80}$$

$$\Theta(T_v) = \Theta(T_{e,k}) + \frac{\Theta(T_{e,k+1}) - \Theta(T_{e,k})}{T_{e,k+1} - T_{e,k}} \cdot (T_v - T_{e,k}) \tag{3.81}$$

[7] It runs at 60 Hz.
[8] Time labels of data sampling.

- If $T_v \geq T_{e,L}$, then prediction is conducted as follows:

$$\Phi(T_v) = \Phi(T_{e,L}) + \frac{\Phi(T_{e,L}) - \Phi(T_{e,L-1})}{T_{e,L} - T_{e,L-1}} \cdot (T_v - T_{e,L}) \tag{3.82}$$

$$\Theta(T_v) = \Theta(T_{e,L}) + \frac{\Theta(T_{e,L}) - \Theta(T_{e,L-1})}{T_{e,L} - T_{e,L-1}} \cdot (T_v - T_{e,L}) \tag{3.83}$$

The 2nd Data Synchronization

For the continuous-discrete EKF, the measurement with the time delay T_{d5} used in the correction stage should be synchronized with the last prediction with the time delay T_{d3}. Since the IMU measurements are more recent than the marker-based pose estimation, the latter one has to be predicted. [52] gives an advisable approach to solving this problem: The state estimation from the EKF can be used to drive the system measurement forward. The synchronized data are based on IMU data: $T_{d6} \approx T_{d3}$

$T_{m,k}$ is the time label of the current system measurement; $T_{oe,k-1}$ and $T_{oe,k}$ are for the last and current optimal (a posteriori) estimations. Then, the measurement can be predicted by

$$\boldsymbol{z}(T_{oe,k}) = \boldsymbol{z}(T_{m,k}) + \frac{\hat{\boldsymbol{x}}_s(T_{oe,k}) - \hat{\boldsymbol{x}}_s(T_{oe,k-1})}{T_{oe,k} - T_{oe,k-1}} \cdot (T_{oe,k} - T_{m,k}), \tag{3.84}$$

where

$$\hat{\boldsymbol{x}}_s = \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{\Psi} \end{bmatrix}. \tag{3.85}$$

represents the selected a posteriori system state estimation.

After this multi-sensory fusion, more accurate position, orientation, and velocity information is achieved. With this recursive filter, the pose estimation is enabled at a higher frequency, which results in smoother control performance. The system also gives better responses to the sudden disturbance after coupling the IMU data to the closed control loop.

3.4 Performance Evaluation

Real-time experiments were conducted for the evaluation of the multi-sensory pose estimation. The experimental environment is shown in Fig. 3.12. In this section, first the ground truth and system calibration setup for the real-time experiments are described. Then, the synchronization of IMU data and vision data is conducted, through which a smoother motion estimation is obtained. After that, the marker-based pose/motion estimation without an EKF is investigated, which is followed by the EKF-aided pose/motion estimation. A hovering behavior with a fixed desired pose and a tracking behavior with time-variant

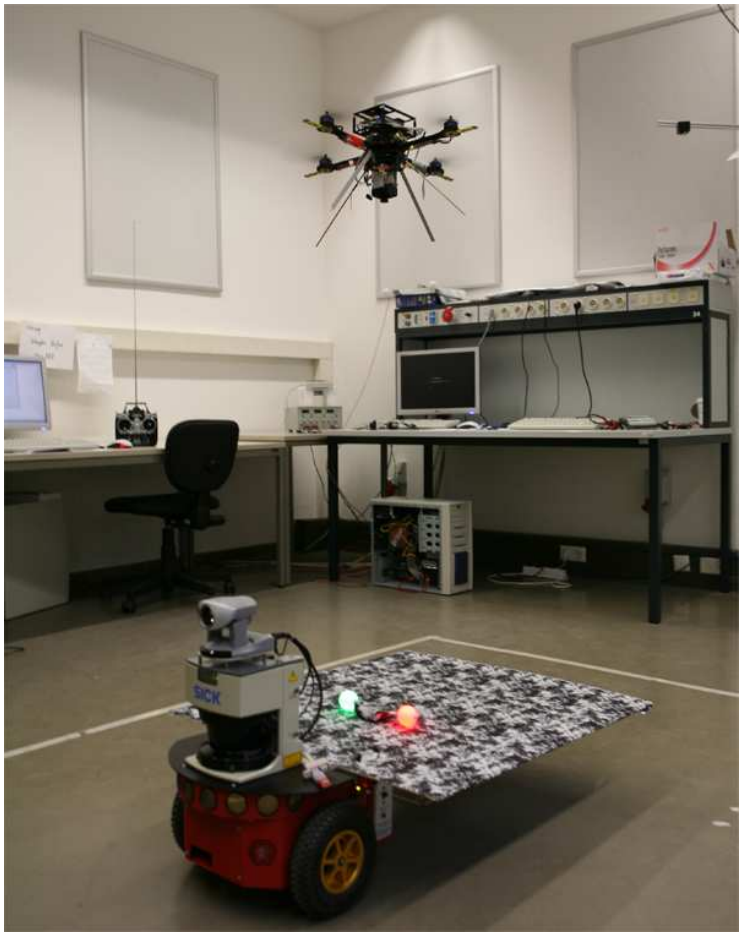

Fig. 3.12: Experimental environment.

desired pose are presented and discussed. Finally, the motion estimation results based on optical flow are shown.

3.4.1 Ground Truth and System Calibration

To evaluate the pose/motion estimation and following control performance in Chapter 4, a position/motion sensing system called *Visualeyez*$^{\text{TM}}$ *II VZ4000* from Phoenix Technologies Incorporated [11] was used, which recorded the 3D positions of four markers mounted on the quadrotor frame. A similar quadrotor pose calculation used in the previous work [131] was conducted and regarded as the ground truth of the actual flying trajectory and flying direction. Based on the ground truth from the tracking system and quadrotor pose estimation using the algorithm described previously, a system calibration of coordination transformation between the quadrotor, the ground robot, and the tracking system was accomplished. More information about this tracking system can be found in Appendix C.

3.4.2 Data Synchronization

In Section 3.3, the necessity of and the algorithm for multi-sensory data synchronization were discussed. The details of the time delay measurement is described in Appendix A,

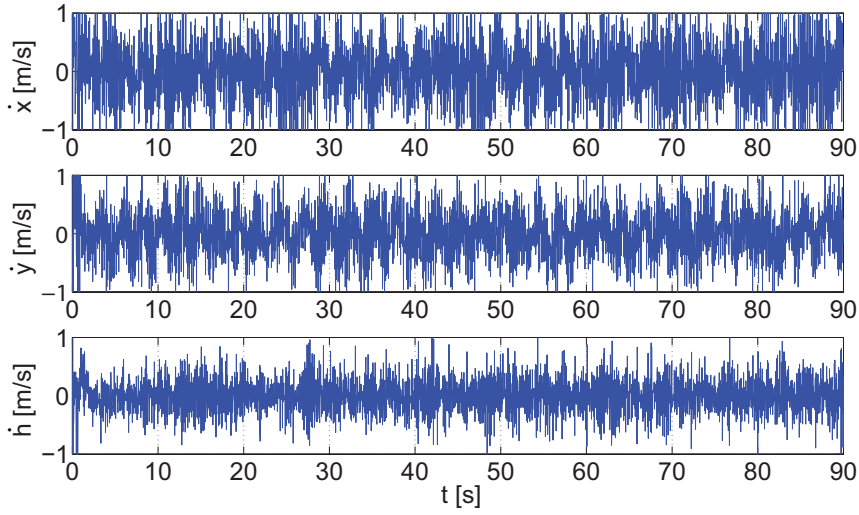

Fig. 3.13: Marker-based motion estimation using unsynchronized data.

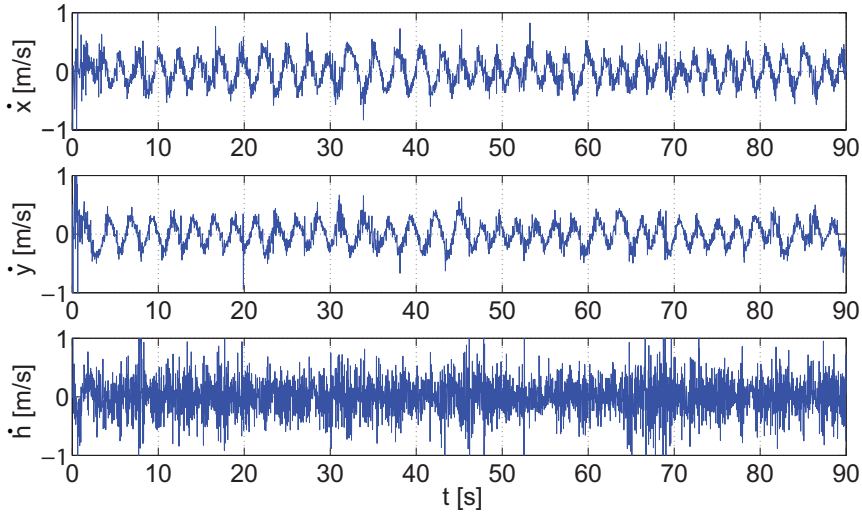

Fig. 3.14: Marker-based motion estimation using synchronized data.

while the data synchronization results are shown here.

To evaluate data synchronization at different points, suitable criteria should be considered. For the evaluation of the first data synchronization between 2D marker positions and estimated Φ_{imu} and Θ_{imu} angles, marker-based motion estimation using the derivative of obtained position is used. The motion is expected to be smooth and limited. With absolutely synchronized data, the location of the quadrotor should be well-estimated. In other words, cluttered motion indicates unsynchronized data.

The motion estimation using unsynchronized data is illustrated in Fig. 3.13. Due to the time difference between estimated Φ_{imu} and Θ_{imu} angles and marker positions, no reasonable motion can be observed.

In Fig. 3.14 the estimated Φ_{imu} and Θ_{imu} angles and marker positions are synchronized and the motions $\dot{x}$ and $\dot{y}$ are represented. For the motion estimation along Z_I-/Z_o-axis, other error sources, e.g. inaccurate IMU measurement in Z_I-/Z_o-direction causing great

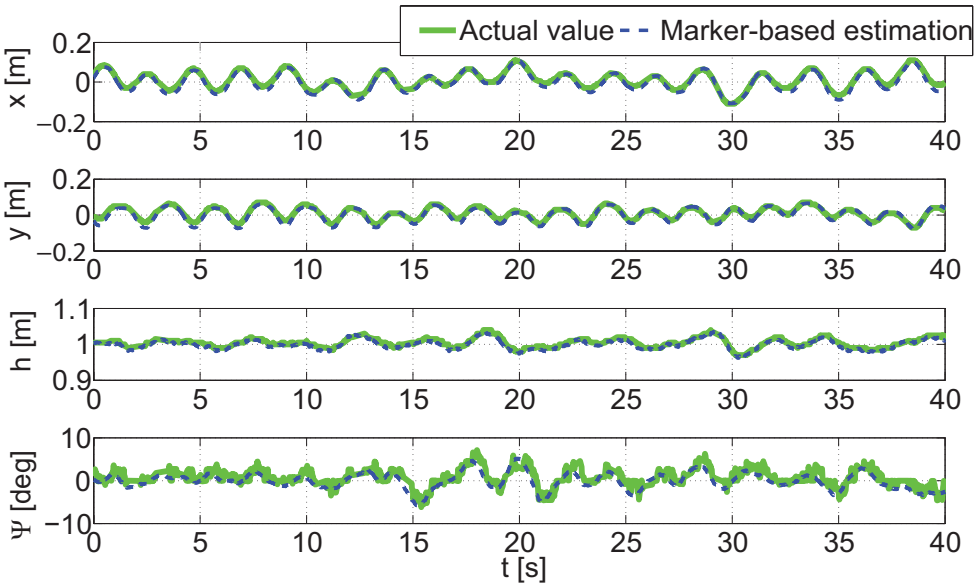

Fig. 3.15: Marker-based pose estimation.

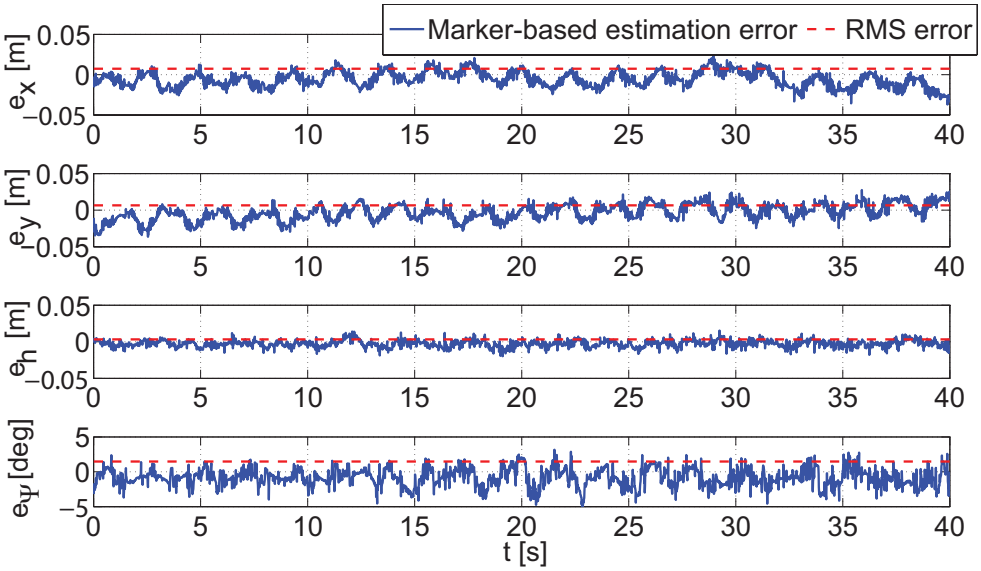

Fig. 3.16: Marker-based pose estimation error.

oscillation, angle estimation error, and marker position errors, are more dominant than unsynchronized data. Further techniques have to be considered, such as an EKF.

Analogously, the second data synchronization before EKF-aided data fusion was also conducted for improved pose/motion estimation.

3.4.3 Marker-Based Pose/Motion Estimation

In this section, the marker based pose estimation without using an EKF is investigated first. A hovering experiment at desired position $[x \;\; y \;\; h]^T = [0 \;\; 0 \;\; 1]^T$ m and with desired yaw angle $\Psi = 0$ deg was carried out.

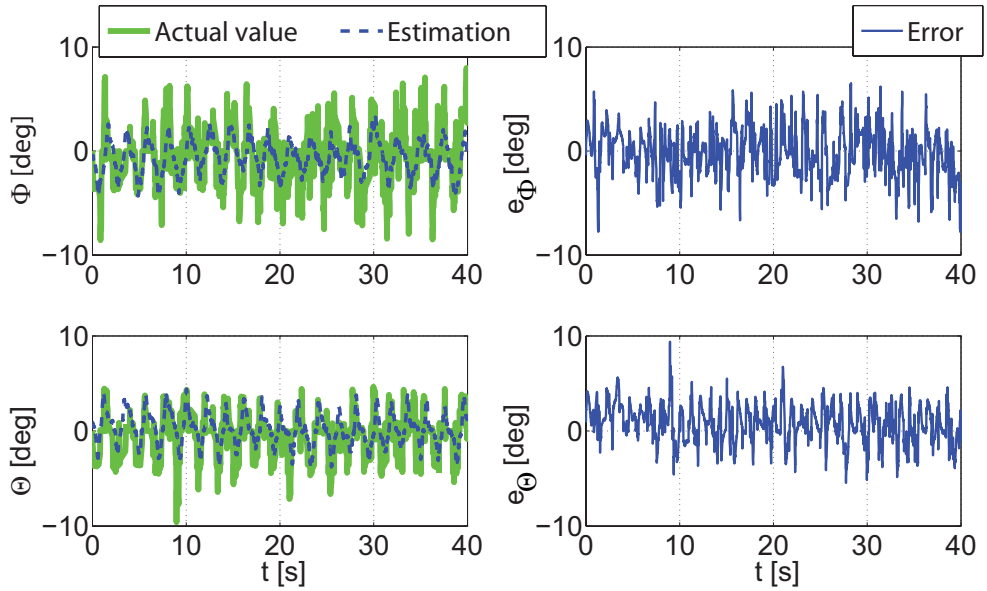

Fig. 3.17: Estimation error of the roll-angle Φ and the pitch-angle Θ.

In Fig. 3.15, the estimation results and the actual values (the ground truth) are illustrated by dashed lines and wide solid lines, respectively, which almost overlap with each other. Since the position of four markers mounted on the quadrotor were transferred to the tracking system sequentially, small errors in position estimation of one marker may cause a relatively large error in the yaw angle determination by using the tracking system. Therefore, obvious disturbance of the actual values of the yaw angle is noticed. In this case, the marker-based yaw angle estimation is more reliable than the ground truth.

The estimation errors are illustrated in solid lines in Fig. 3.16. The absolute estimation errors e_x and e_y in the X_o-/Y_o-directions are less than 0.04 m. The altitude is more accurately estimated with a maximum error e_h of 0.02 m. In spite of the measurement disturbance of the ground truth, the absolute Ψ estimation error e_Φ is less than 5 deg.

Furthermore, *Root Mean Square* (RMS) error is used here to evaluate the quality of the pose estimation:

$$e_{\mathrm{RMS}} = \sqrt{\frac{e_1^2 + e_2^2 + \cdots + e_k^2}{k}}, \tag{3.86}$$

where e_{RMS} is the RMS error, $e_1 \cdots e_k$ are estimation errors of the kth measurement. The RMS errors $e_{\mathrm{RMS},x}$, $e_{\mathrm{RMS},y}$, $e_{\mathrm{RMS},h}$, and $e_{\mathrm{RMS},\Psi}$ are 0.015 m, 0.014 m, 0.01 m, and 1.5 deg, respectively, illustrated by the dashed lines in Fig. 3.16. It is concluded that the marker based pose estimation is very accurate.

In Fig. 3.17 the estimated Φ and Θ angles are compared with the ground truth. The relatively large oscillations of the ground truth are due to the same problem caused by the tracking system as that in the yaw angle. These angle estimation errors are the main error sources of the pose estimation inaccuracy.

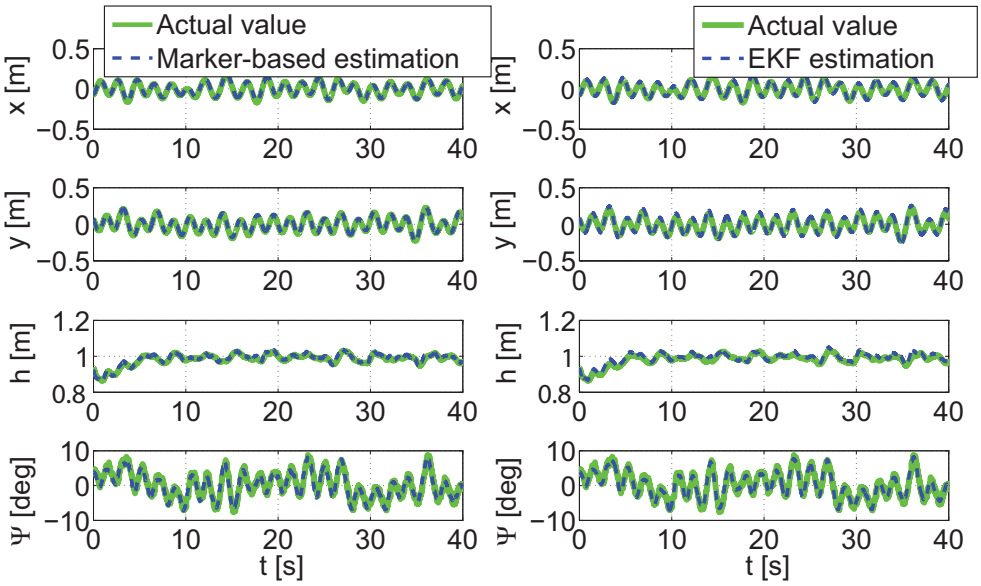

Fig. 3.18: Pose estimation result using the first set of EKF parameters.

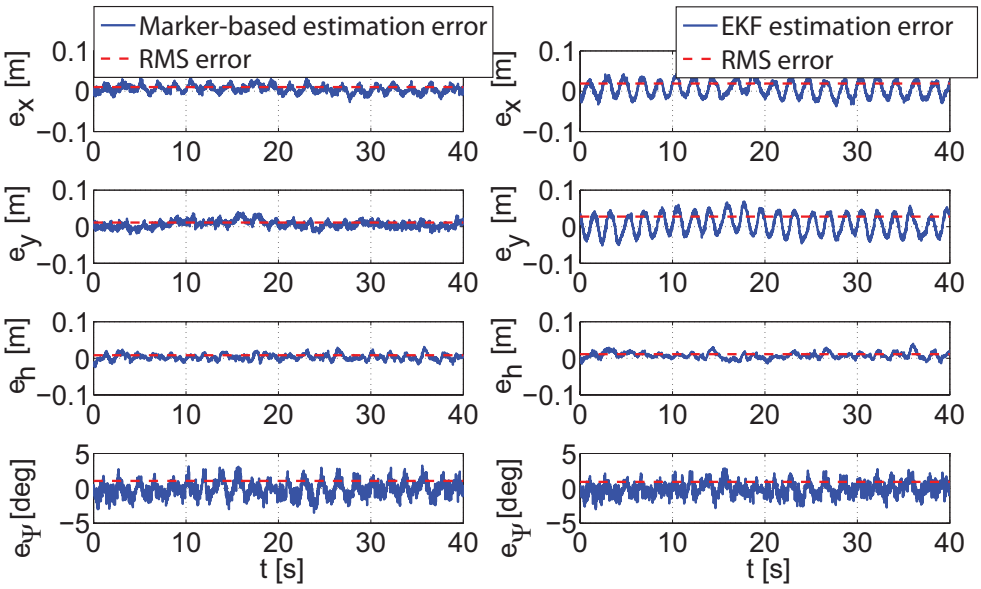

Fig. 3.19: Pose estimation errors using the first set of EKF parameters.

3.4.4 EKF-Aided Pose/Motion Estimation

The EKF-aided pose/motion estimation is investigated in this section. Using the tuned EKF, hovering and tracking behaviors of the quadrotor were exhibited where *Proportional-Integral-Derivative* (PID) controllers were used. Detailed control design will be introduced in Chapter 4.

EKF Parameter Tuning

Before using the EKF to fuse multi-sensory data, the noise covariances of the EKF have to be tuned first. The covariance matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$, which indicate the reliability of the

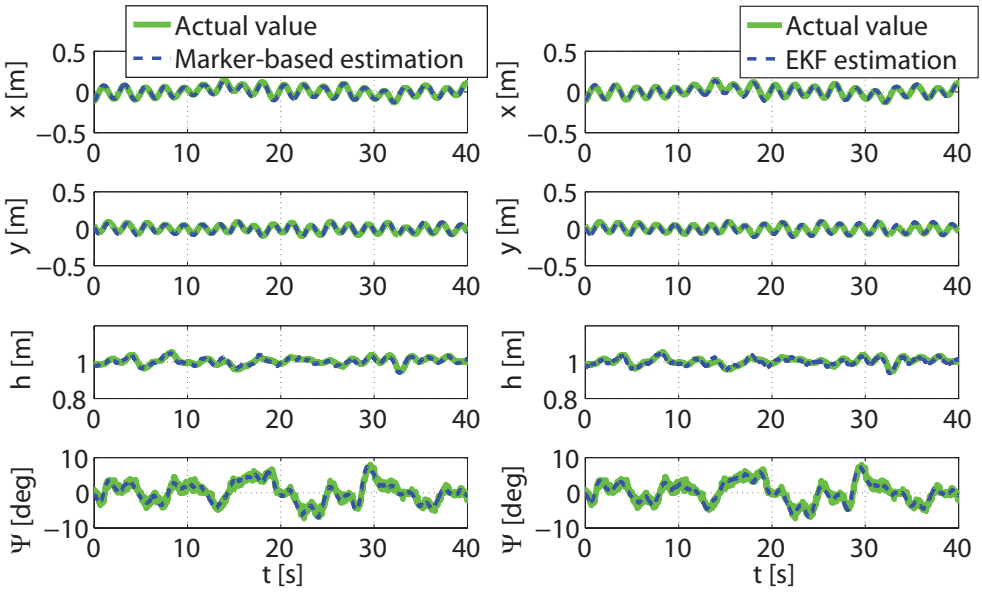

Fig. 3.20: Pose estimation result using the second set of EKF parameters.

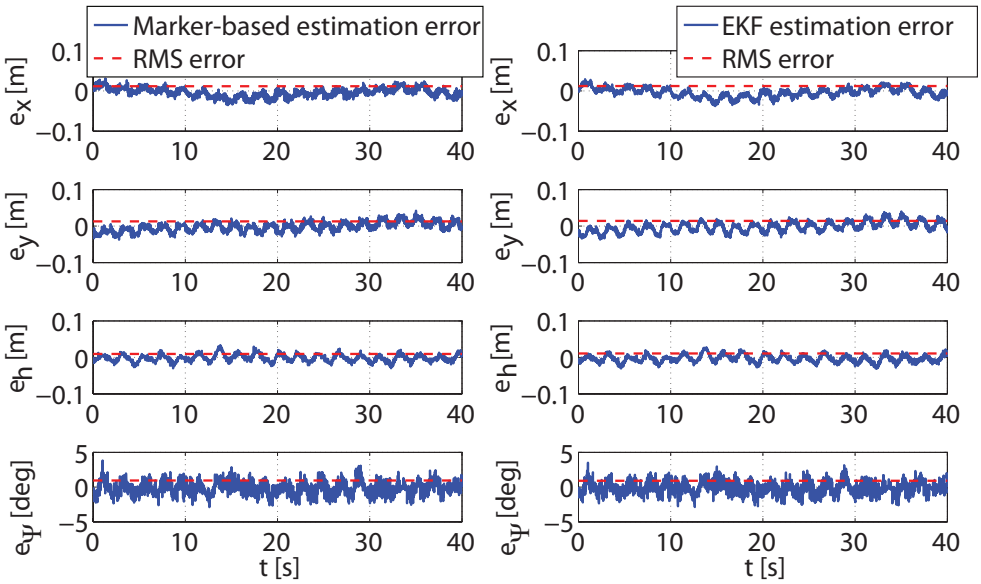

Fig. 3.21: Pose estimation errors using the second set of EKF parameters.

system prediction and system measurements, respectively, determine the performance of this iterative filter. They are assumed to be diagonal in this thesis.

Experiments were conducted for the parameter tuning. To better reflect the true state of a flying quadrotor, the quadrotor was set to hover over the ground robot. As discussed in Section 3.3, the EKF parameters should be tuned such that the residual prediction error is minimized. Several different series of parameters were applied in experiments. Then, they were tuned with respect to stochastic descent of the estimation error. RMS error was used here to evaluate the quality of the pose estimation.

To simplify this process, the $\boldsymbol{R}$ matrix can be roughly predetermined. Considering the error analysis of the marker-based pose estimation in Section 3.4.3, the terms corresponding

to x, y, and z estimations have an order of magnitude from $10^{-4}\,\mathrm{m}^2$, and Ψ estimation has a covariance about $1\,\mathrm{deg}^2$.

Fig. 3.18 and 3.20 illustrate the pose estimation of the quadrotor during hovering using marker-based estimation (left) and using the EKF (right) with two different parameter sets, while the respective estimation errors and RMS errors are illustrated in Fig. 3.19 and 3.21. For the first parameter set, the estimation error of the EKF is more significant than that of the marker-based estimation. By using the second parameter set, both estimation errors are approximately identical to each other, characterizing a better performance of the EKF pose estimation than the former one. Note that if the tracking system gives more continuous pose measurements, not stepwise, the EKF estimation should perform better than the marker-based estimation.

Since the tracking system does not provide smooth pose ground truth, no sensible motion information can be obtained as the ground truth. In order to find the parameters, which give more accurate motion estimation, the hovering quality of the quadrotor was deployed for the parameter tuning, instead of the motion measurements. Using the second parameter set, the quadrotor was able to hover at the desired pose with fewer control errors. This also indicates a better performance of the EKF motion estimation.

The hovering experiment was carried out many times, and the parameters were evaluated and tuned as described above. Finally, the following values were set in the EKF:

$$\boldsymbol{R} = \mathrm{diag}\left(0.0004\ \mathrm{m}^2,\ 0.0004\ \mathrm{m}^2,\ 0.0002\ \mathrm{m}^2,\ 0.5000\ \mathrm{deg}^2\right), \tag{3.87}$$

and

$$\begin{aligned}\boldsymbol{Q} \;=\; \mathrm{diag}(&0.0002\ \mathrm{m}^2,\ 0.0002\ \mathrm{m}^2,\ 0.1000\ \mathrm{m}^2,\ 0.0010\ (\mathrm{m/s})^2,\ 0.0010\ (\mathrm{m/s})^2,\\ &0.2000\ (\mathrm{m/s})^2,\ 0.3000\ \mathrm{deg}^2,\ 0.3000\ \mathrm{deg}^2,\ 0.6000\ \mathrm{deg}^2).\end{aligned} \tag{3.88}$$

Performance Comparison with and without EKF

Now, the pose estimation and motion estimation results without using an EKF and the results with an EKF are compared. The EKF and marker-based pose estimation run at 100 Hz and 15 Hz, respectively. To illustrate the smoothing effect more obviously, the pose estimations using both methods are compared with each other for 1 s in Fig. 3.22. The pose is more smoothly estimated by using the EKF due to the more frequently updated estimation data.

As discussed, the motion estimation is improved by considering synchronization of the multi-sensory data. In Fig. 3.23, motion estimation results with and without using the EKF are illustrated. By using this recursive filter, the high-frequency noise was filtered out in all 3 directions and the motions are sensibly represented.

RMS Error Analysis During Hovering

To evaluate quality of the EKF-aided pose estimation, hovering experiments with different desired poses were carried out. The pose estimation errors are computed and compared with each other. The desired pose varies only one term every time and for x, y, z, Ψ in

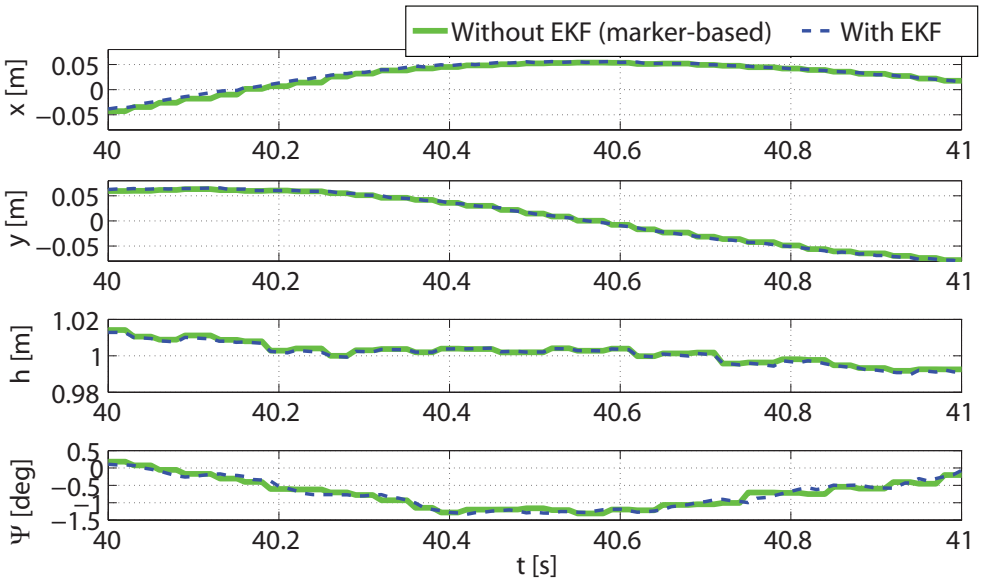

Fig. 3.22: Pose estimation results without (marker-based) and with EKF.

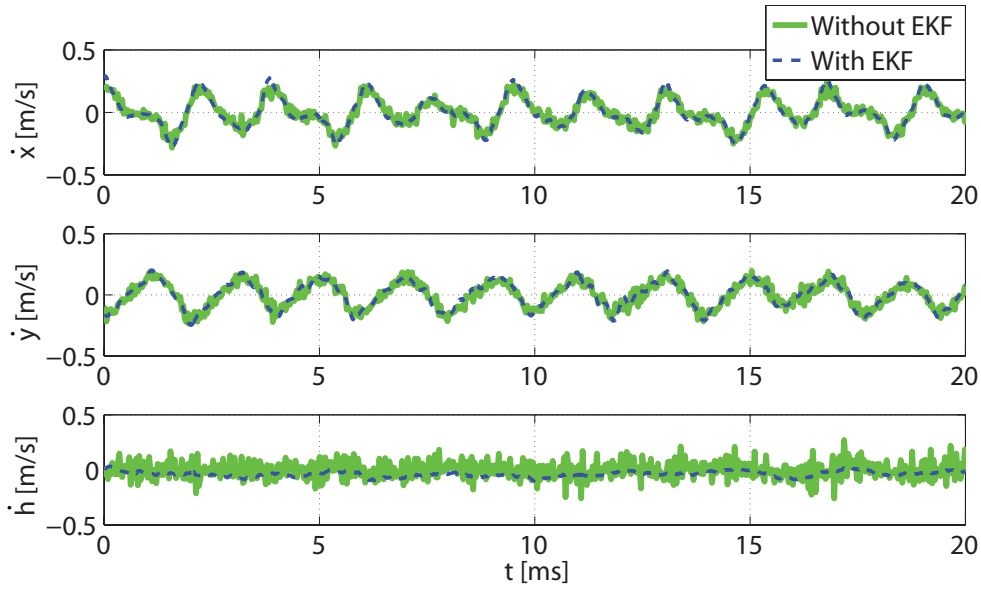

Fig. 3.23: Motion estimation results without (marker-based) and with EKF.

sequence. Detailed variations are listed in Tab. 3.2–3.5. The corresponding estimation errors are illustrated in Fig. 3.24.

The RMS errors $e_{\mathrm{RMS},x}$, $e_{\mathrm{RMS},y}$, and $e_{\mathrm{RMS},h}$ become larger if x_d and y_d move away from the zero point. There are two explanations of this phenomenon: 1) the descending of the distance between two markers in the image plane, which results in increasing measurement noise; 2) the distortion caused by the lens. Objects near to the edges of the CCD are more distorted than those near to the center point.

The RMS errors $e_{\mathrm{RMS},x}$ and $e_{\mathrm{RMS},y}$ rise approximately linearly with the height h_d. h is also better estimated at lower height, due to the larger distance between markers in the image plane. Rotation Ψ_d about the Z_o-axis has negligible effect on the pose estimation,

Nr.	x_d [m]	y_d [m]	h_d [m]	Ψ_d [deg]
1	0.0	0.0	1.0	0
2	-0.1	0.0	1.0	0
3	-0.2	0.0	1.0	0
4	0.1	0.0	1.0	0
5	0.2	0.0	1.0	0

Tab. 3.2: Variation of x_d.

Nr.	x_d [m]	y_d [m]	h_d [m]	Ψ_d [deg]
1	0.0	0.0	1.0	0
2	0.0	-0.1	1.0	0
3	0.0	-0.2	1.0	0
4	0.0	0.1	1.0	0
5	0.0	0.2	1.0	0

Tab. 3.3: Variation of y_d.

Nr.	x_d [m]	y_d [m]	h_d [m]	Ψ_d [deg]
1	0.0	0.0	1.0	0
2	0.0	0.0	0.9	0
3	0.0	0.0	0.8	0
4	0.0	0.0	1.1	0
5	0.0	0.0	1.2	0

Tab. 3.4: Variation of h_d.

Nr.	x_d [m]	y_d [m]	h_d [m]	Ψ_d [deg]
1	0.0	0.0	1.0	0
2	0.0	0.0	1.0	-45
3	0.0	0.0	1.0	-90
4	0.0	0.0	1.0	45
5	0.0	0.0	1.0	90

Tab. 3.5: Variation of Ψ_d.

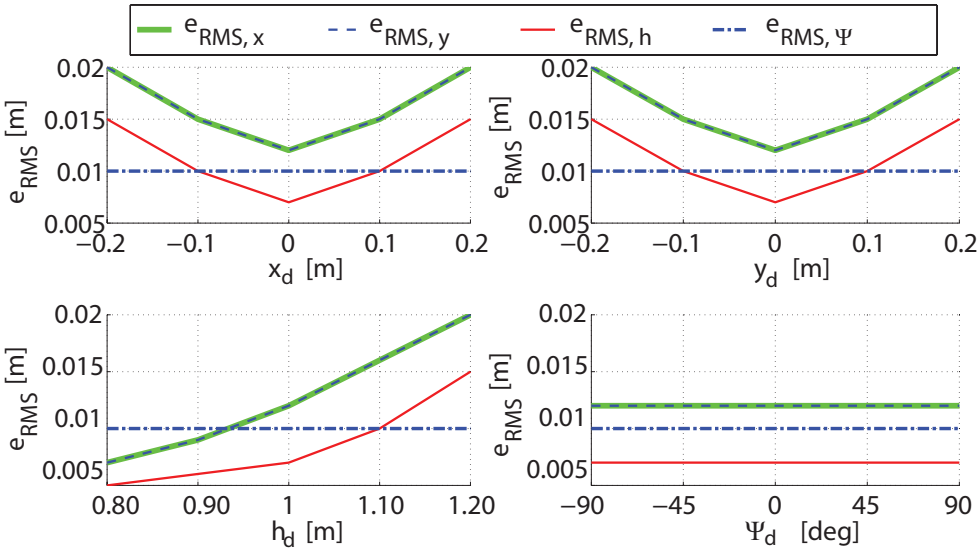

Fig. 3.24: Pose estimation errors for varying x_d, y_d, h_d, and Ψ_d.

as it results in only a rotation of the markers in the image plane.

Since the algorithm used for the marker-based pose estimation is optimized for solving the angle Ψ, Ψ is accurately estimated for all poses.

EKF-Aided Pose Estimation During Tracking

The quadrotor is planned later to cooperate with the ground robot together. Therefore, the pose estimation performance in a dynamic behavior such as tracking the movement of the ground robot should be studied.

In Fig. 3.25, the ground robot ran along an arbitrary trajectory in 30 s. The displacements in X_I-/Y_I-directions were about 0.6 m and 0.2 m. The yaw rotation was about 160 deg. The trajectory of the ground robot was recorded by the tracking system and

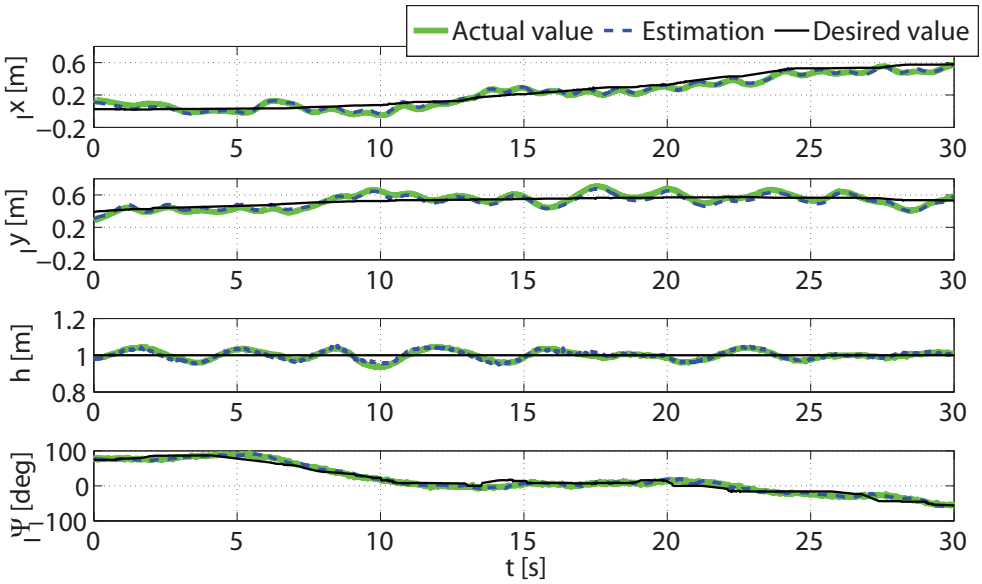

Fig. 3.25: EKF-aided pose estimation result in a quadrotor tracking experiment.

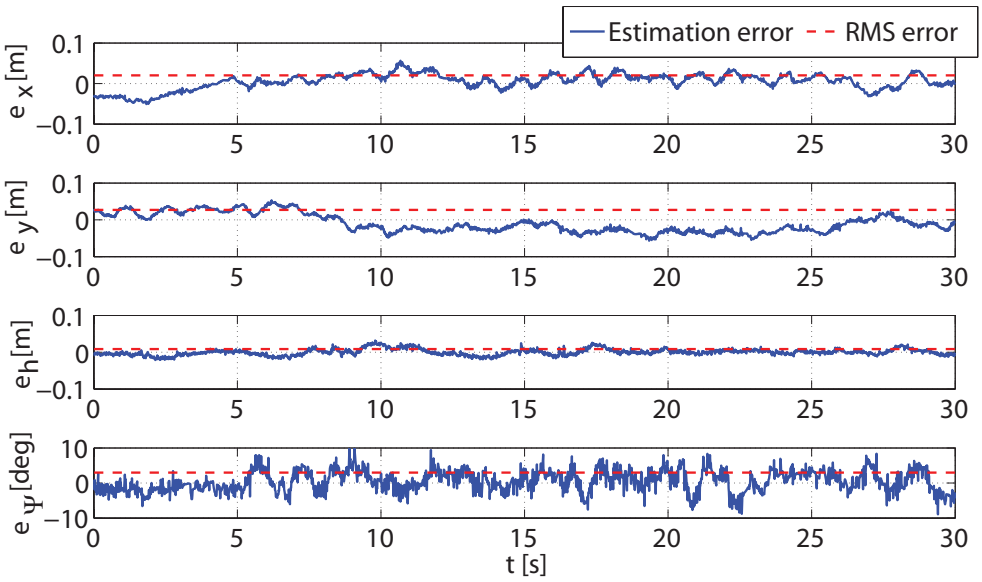

Fig. 3.26: EKF-aided pose estimation error during tracking.

fused with the movement of the quadrotor off-line. The quadrotor successfully tracked the markers on the ground robot. As shown in Fig. 3.26, the RMS errors of x, y, h, and Ψ estimations are 0.020 m, 0.027 m, 0.009 m, and 3 deg, respectively.

3.4.5 Optical-Flow-Based Pose/Motion Estimation

The motion estimation based on optical flow computation is also shown here. Fig. 3.27 shows the optical flow of 36 image points and that of 24 image points after outlier cancellation. The inaccurate optical flow vectors are removed.

The pose estimation result using optical flow motion estimation is shown in Fig. 3.28 and

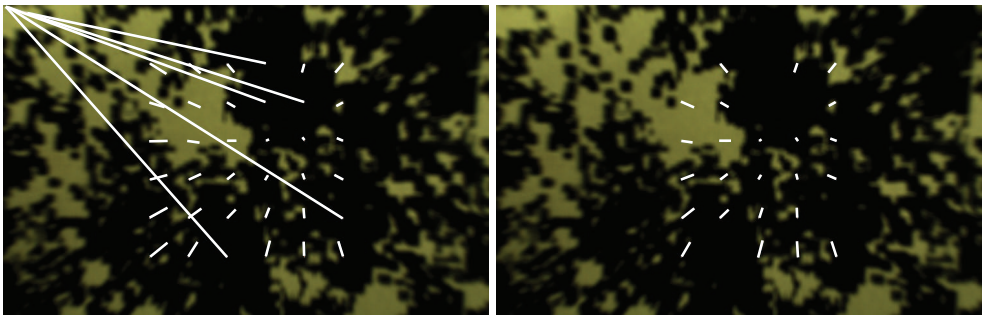

Fig. 3.27: Optical flow before and after outlier cancellation.

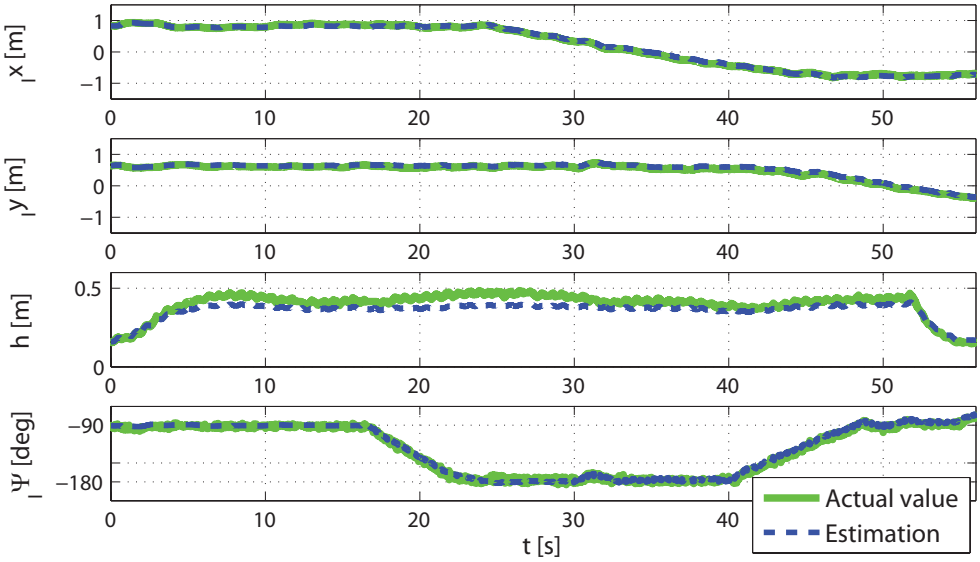

Fig. 3.28: Pose estimation result based on optical flow in a quadrotor take-off, hovering, tracking, and landing experiment.

3.29 in an experiment consisting of quadrotor take-off, hovering, tracking, and landing. The RMS errors in X_I-/Y_I-directions are very small, namely 0.034 m and 0.048 m, respectively. The yaw angle is also very accurately estimated with an RMS error of 3.4 deg. The RMS error in altitude is about 0.048 m, which is mainly due to the integration error of the motion estimation without correction. As optical-flow-based motion estimation is only applied in a short time interval in take-off and landing, the error in this order is acceptable.

3.5 Discussion

Markers, Features, and Image Points of Interest

For a vision-based tracking task, markers, features, and image points of interest can be used to describe the target. Artificial markers have the advantage of robust detection and the disadvantage of costly setup. A flexible and natural alternative is to use feature points, which can be previously trained and then applied online. However, they are sensitive to lighting conditions and require a higher computational cost. Compared to the former ones,

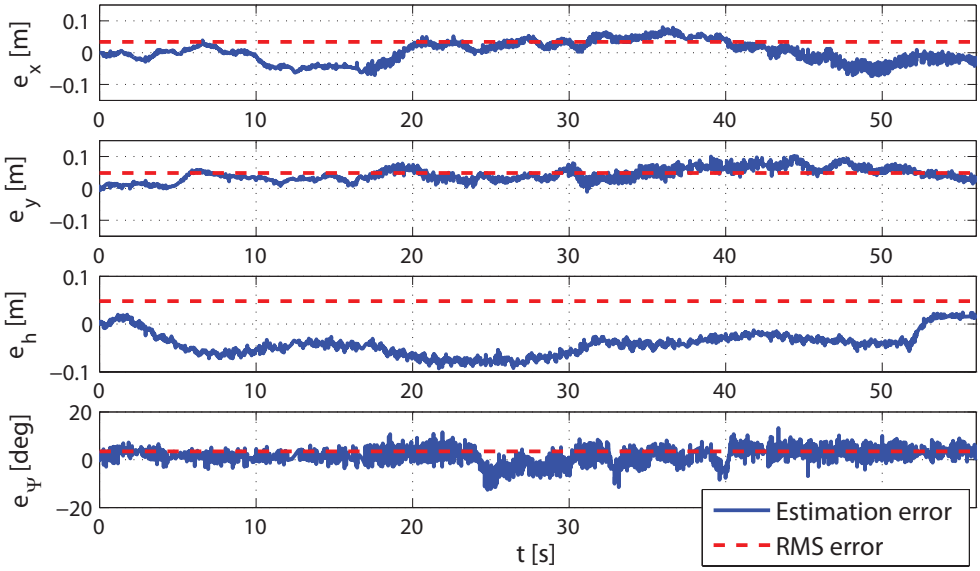

Fig. 3.29: Pose estimation error based on optical flow in a quadrotor take-off, hovering, tracking, and landing experiment.

optical flow computed using image points of interest can be used for motion estimation in a more flexible and efficient manner and is envisioned to totally replace artificial markers. A challenging problem arising in this context is that a segmentation of the target from the background should be conducted first. As optical-flow-based motion estimation is applied when the quadrotor has a lower height and the input images only contain the points of the ground robot, this problem does not need to be considered here.

Feedforward Information of the Ground Robot Motion

In this chapter, no accurate pose and motion information of the ground robot is fed forward to the quadrotor in real time. The movement of the ground robot is assumed to be noise in the system model. Since the system measurements are much more reliable than the IMU data, the estimated pose is dominated by the marker-based pose estimation. At the same time, sudden motion variations of the quadrotor come into the motion estimation through the system model driven by IMU data. Thus, the estimation results are still applicable to the tracking experiments.

If all the states of the ground robot are available, more accurate pose/motion estimation and a better tracking performance are expected to be achieved, which will be discussed in Chapter 4.

Improved Pose Estimation Performance

It is not easy to make a direct comparison with the other state-of-the-art works, since the hardware platforms, sensor modalities, sensor numbers, fusion techniques in details, or even marker designs in different works may vary. Therefore, the two most comparable works are selected here to provide an idea of the accuracy of the results presented in this thesis.

A continuous-discrete EKF-aided data fusion of vision and inertial sensors for pose estimation has been applied for a helicopter called GTmax in [129], in which the on-board camera looked forwards and detected a dark square against a light background. Using this large helicopter (11.9 ft long with a rotor diameter of 10.2 ft), a very large marker (36 ft^2), and much higher computational power (two embedded PCs), the maximum error along the optical axis was about 8% of the distance between the helicopter and the dark square, namely 9.2 ft of 110–120 ft, while the maximum error of the estimation here is only approximately 2.5%, namely 0.03 m of 1.2 m.

In [19], a quadrotor is equipped with IMUs, a stereo camera looking forwards, and laser range finders. The EKF-based fusion of IMU data and stereo vision or laser data ran at 50 Hz to estimate quadrotor position, velocity, acceleration, and the IMU biases. The variance parameters were obtained using an external motion-capture system. The authors did not mention the estimation errors explicitly. From their plots, an average displacement of 0.3 m in X_I-/Y_I-directions between the ground truth and the visual odometry with bundle adjustment can be derived.

Compared to the works mentioned above, the pose estimation presented in this thesis achieves higher accuracy and a higher frequency, which are both critical for control performance.

3.6 Summary

The fundamental and challenging problem for a stable flight of UAVs is accurate pose/motion estimation. Due to sensor limitations, such as drifts of inertial sensors, limited field of view, and low sampling rate of vision sensors, an elaborated fusion of multi-sensory data becomes an indispensable means.

In this chapter, a quadrotor in an autonomous heterogeneous air-ground system is established only using minimal on-board sensors: IMUs and a monocular camera. First, efficient marker-based pose and motion estimation is proposed and optimized. Then, a continuous-discrete EKF is applied, in which the high-frequency IMU data drive the prediction, while the estimates are corrected by the accurate and steady vision data. A high-frequency fusion at 100 Hz is achieved. Moreover, time delay analysis and data synchronizations are thoroughly conducted to further improve the pose/motion estimation. In addition, optical-flow-based motion estimation is applied for the case if the markers cannot be robustly detected, for example, during take-off or landing. The complete on-board implementation of sensor data processing and fusion without using a ground station reduces the influence of data transfer time delay, enables autonomous task accomplishment, and extends the work space. Compared to the state-of-the-art works, a higher frequency and higher accuracy of the quadrotor pose/motion estimation is obtained, exhibited in experimental evaluation of real-time hovering and tracking a moving ground robot.

Based on the accurate estimation of the quadrotor pose and motion, an effective control design is required in order to complete this autonomous flying system, which is investigated in the next chapter.

4 Control Design and Implementation

Based on high-frequency and accurate pose estimation of the quadrotor, a stable and effective control design is desired in order to realize autonomous flying. Although quadrotors have major advantages in holonomic motion and robust mechanical design without swashplates, it is very demanding to elaborate a controller mainly due to the non-linear and highly unstable dynamics, time delay, and the need for a simplified control structure caused by hardware limitations. Sophisticated control designs have normally been evaluated in simulations or limitedly evaluated in a non-autonomous manner.

This chapter is aiming at designing, implementing, and discussing adequate control structures for the quadrotor, to overcome the aforementioned challenges and to improve the quadrotor flying behavior. Standard controllers such as *Proportional-Integral-Derivative* (PID) controllers, optimal controllers such as *Linear Quadratic* (LQ) controllers, and non-linear controllers such as backstepping-based controllers and sliding model controllers are carefully adapted to this quadrotor system and discussed. Compared to most works which have implemented controllers on a real quadrotor, the system model used here is more complex, in order to preserve the original dependency of system states. Moreover, the overall time delay is derived and compensated for, while feedforward information is taken into account.

An elaborated, integrated control approach combining the advantages of these controllers is proposed and evaluated in a complete flight experiment consisting of take-off, hovering, tracking, and landing. The control performance is proved to be effective in terms of small *Root Mean Square* (RMS) control errors of 0.06 m in position control, 0.03 m in altitude control, and 3 deg in yaw control, which are much smaller than those in comparable works in the literature.

The remainder of this chapter is organized as follows: First, the problem addressed in this chapter and the system modeling for control are described in Section 4.1. In Section 4.2, various controllers are adapted to the system model used in this thesis. In Section 4.3, simulations presenting the controlled flying behaviors are conducted. Based on the simulation results, the integrated control design considering quadrotor behavior from take-off to landing is proposed. After that, real-time pose control performance is experimentally evaluated. The results are presented in Section 4.4. Finally, a discussion and a summary are given in Sections 4.5 and 4.6.

4.1 Problem Definition and System Modeling for Control

4.1.1 Problem Definition

Up to now, most existing works about controlling a quadrotor are limited in a compromise between sophisticated control design and control simplicity restricted by hardware limitation and real-time implementation:

- Since the control of a quadrotor is a challenging problem, various optimal or nonlinear control designs have been proposed but only evaluated in simulations [37, 53, 87, 93, 106]. Most real-time experiments only apply simple, standard PID or *Proportional-Derivative* (PD) controllers on a much simplified system model [21, 60, 79, 106, 109, 123, 128].

- Most works which aim at the evaluation of control performance commonly use external sensors such as *Global Positioning System* (GPS) or tracking systems as pose/-motion feedback [42, 115], such that the error sources are restricted to control design. Autonomous flight is barely considered.

- Moreover, in most air-ground multi-robot systems, flying systems serve as a central/-global control unit and provides information for the ground robots [29, 117, 119, 120]. Motion feedforward of a ground robot to a flying system has not yet been implemented and evaluated.

- Few works have considered a complete flight scenario consisting of take-off, hovering, tracking, and landing on a moving ground robot, each of which requires a unique control design.

To overcome the above limitations, this chapter aims at design, adaptation, implementation, and performance evaluation of controllers on a fully autonomous system in real-world environments. PID controllers, LQ controllers, backstepping controllers, and sliding mode controllers are all thoroughly adapted to the quadrotor dynamic system, evaluated not only in simulations but also in real-time experiments, to achieve an entire flight behavior. The on-board *Inertial Measurement Unit* (IMU) and camera data are exclusively used as sensor feedback. Furthermore, feed-forward extension of the control structure improves the control performance significantly.

4.1.2 System Modeling for Control

Ignoring the unstable and untouchable aerodynamic forces and moments, the flying motion of a quadrotor is determined by the rotational speed of the four propellers and the gravity. Fig. 4.1 illustrates the quadrotor's propellers, producing the force f_f, f_b, f_l, f_r and the torque τ_f, τ_b, τ_l, τ_r for the front, back, left, and right rotors. If the rotating velocities of all the four motors are increased by the same amount, the quadrotor will fly upwards, and vice versa. The total thrust F is the sum of the force provided by all the propellers:

$$F = f_f + f_r + f_b + f_l = b \cdot (\Omega_f^2 + \Omega_b^2 + \Omega_l^2 + \Omega_r^2), \tag{4.1}$$

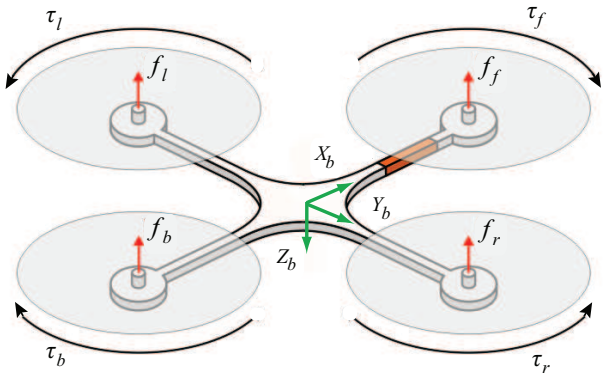

Fig. 4.1: Quadrotor schema.

where b denotes the thrust factor and Ω denotes the rotor speed.

The roll and pitch angles are under-actuated control variables. The force difference of a pair of propellers causes rolling around the X_b-axis or pitching around the Y_b-axis. The rolling torque τ_Φ and the pitching torque τ_Θ can be computed using the following equations:

$$\begin{aligned} \tau_\Phi &= l \cdot (f_l - f_r) = lb \cdot (\Omega_l^2 - \Omega_r^2); \\ \tau_\Theta &= l \cdot (f_f - f_b) = lb \cdot (\Omega_f^2 - \Omega_b^2), \end{aligned} \tag{4.2}$$

where l denotes the quadrotor axis length. The yaw rotation is caused by the difference between the angular momentum generated by these two pairs of rotors:

$$\tau_\Psi = (\tau_f + \tau_b) - (\tau_l + \tau_r). \tag{4.3}$$

Moreover, based on experimental measurement, the thrust F generated by motors can be approximately calculated by using

$$F \approx U \cdot F_{\text{cmd}} \cdot K, \tag{4.4}$$

where U is the current battery voltage, F_{cmd} is the thrust command signal between 0 and 255, and K is a negative constant due to the Z_b-direction of the quadrotor body frame. K should be experimentally determined.

Similar to [28, 35], translational motion and rotational motion of the quadrotor are derived below.

Translational Motion

Generally, according to Newton's laws, the force ${}_b\boldsymbol{f}$ is related to the linear velocity ${}_b\boldsymbol{v} = [u \;\; v \;\; w]^T$ in the body frame and the angular velocity ${}_b\boldsymbol{\omega} = [p \;\; q \;\; r]^T$ in the body frame as follows:

$${}_b\boldsymbol{f} = m \cdot (\frac{\mathrm{d}({}_b\boldsymbol{v})}{\mathrm{d}t} + {}_b\boldsymbol{\omega} \times {}_b\boldsymbol{v}) \tag{4.5}$$

Moreover, the force ${}_b\boldsymbol{f}$ comprises three components as follows:

$$ {}_b\boldsymbol{f} = \begin{bmatrix} 0 \\ 0 \\ F \end{bmatrix} + m \cdot {}^b\boldsymbol{R}_I \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \boldsymbol{f}_{\text{unm}}, \tag{4.6} $$

where F is the thrust, g is the gravitational constant, and $\boldsymbol{f}_{\text{unm}}$ denotes unmodeled force, such as hub force and ground effect [35]. As $\boldsymbol{f}_{\text{unm}}$ is not controllable, it is regarded here as noise first and is targetedly considered in the controller design, in which controllers robust to unmodeled force are applied (see Section 4.2.4).

From Eq. 4.5, the following extended equation can be derived:

$$ \frac{{}_b\boldsymbol{f}}{m} = \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix}. \tag{4.7} $$

Neglecting the Coriolis term in Eq. 4.7 and the unmodeled force in Eq. 4.6 and substituting Eq. 3.18 into Eq. 4.5 and 4.6, the following relationship is derived:

$$ \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{F}{m} \end{bmatrix}, \tag{4.8} $$

where a_x, a_y, and a_z denote the linear accelerations measured by the IMUs. Substituting Eq. 4.8 into Eq. 3.22, the following equation is obtained:

$$ \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} c\Phi s\Theta c\Psi + s\Phi s\Psi \\ c\Phi s\Theta s\Psi - s\Phi c\Psi \\ c\Phi c\Theta \end{bmatrix} \cdot \frac{F}{m} - \begin{bmatrix} a_r \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \begin{bmatrix} \ddot{x}_{\text{rot}} \\ \ddot{y}_{\text{rot}} \\ 0 \end{bmatrix}, \tag{4.9} $$

where $\ddot{x}_{\text{rot}}$ and $\ddot{y}_{\text{rot}}$ are defined in Eq. 3.23.

Rotational Motion

Similar to Eq. 4.5, the equation of Coriolis is derived using the angular momentum ${}_b\boldsymbol{L}$ and the torque ${}_b\boldsymbol{\tau}$ as follows:

$$ {}_b\boldsymbol{\tau} = \frac{\mathrm{d}({}_b\boldsymbol{L})}{\mathrm{d}t} + {}_b\boldsymbol{\omega} \times {}_b\boldsymbol{L}. \tag{4.10} $$

Moreover, the relationship between the angular momentum ${}_b\boldsymbol{L}$ and the angular velocity ${}_b\boldsymbol{\omega}$ in the body frame is described using the inertial matrix $\boldsymbol{J}$:

$$ {}_b\boldsymbol{L} = \boldsymbol{J} \cdot {}_b\boldsymbol{\omega} \tag{4.11} $$

Since the quadrotor is symmetric, the inertial matrix $\boldsymbol{J}$ is a diagonal matrix:

$$ \boldsymbol{J} = \operatorname{diag}(J_x, J_y, J_z) \quad \text{with} \quad J_x = J_y. \tag{4.12} $$

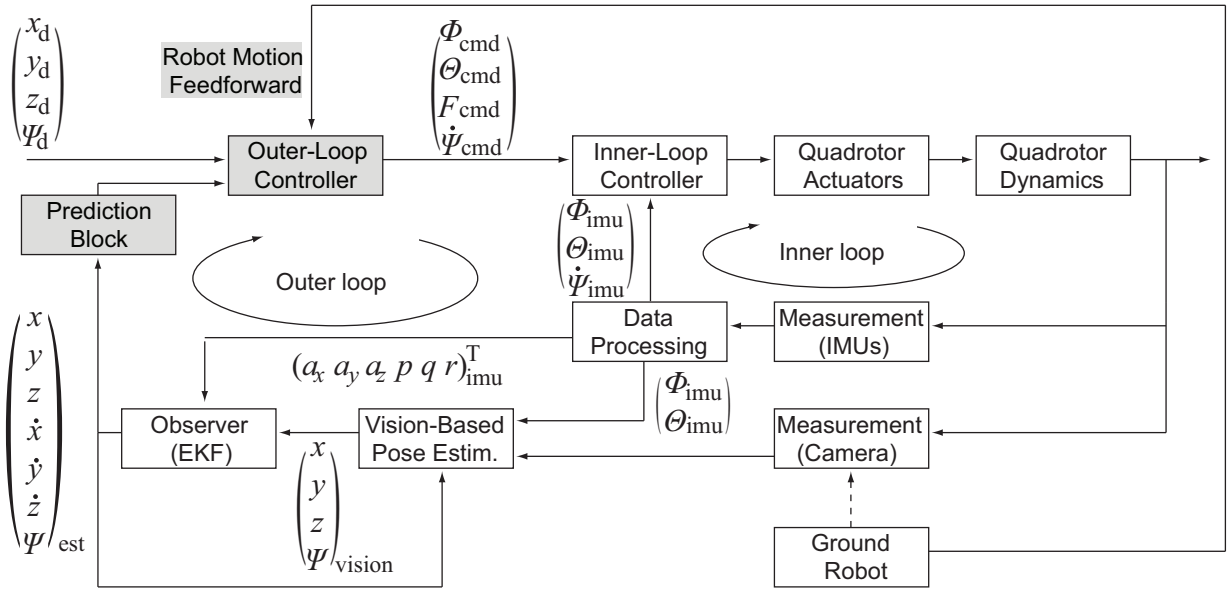

Fig. 4.2: Control structure consisting of an inner-loop controller for roll, pitch angles and yaw angle velocity stabilization and an outer-loop controller for position and yaw control. The gray boxes represent the main focuses of this section.

Then,

$$ {}_b\boldsymbol{\tau} = \frac{\mathrm{d}(\boldsymbol{J}_b\boldsymbol{\omega})}{\mathrm{d}t} + {}_b\boldsymbol{\omega} \times \boldsymbol{J}_b\boldsymbol{\omega}. \tag{4.13} $$

Similar to [28], the following equation is obtained:

$$ \begin{bmatrix} \ddot{\Phi} \\ \ddot{\Theta} \\ \ddot{\Psi} \end{bmatrix} = \begin{bmatrix} \dot{\Theta}\dot{\Psi}(\frac{J_y - J_z}{J_x}) + \frac{1}{J_x}\tau_\Phi \\ \dot{\Phi}\dot{\Psi}(\frac{J_z - J_x}{J_y}) + \frac{1}{J_y}\tau_\Theta \\ \frac{1}{J_z}\tau_\Psi \end{bmatrix}. \tag{4.14} $$

As the quadrotor rotation is already controlled using the off-the-shelf controller (the inner-loop controller mentioned in Section 3.1), the control of Φ, Θ, and Ψ is not described in detail in the following sections. The problem remaining is how to calculate the command signals Φ_{cmd}, Θ_{cmd}, and $\dot{\Psi}_{\mathrm{cmd}}$ as the inputs of the inner-loop controller, which is described in the next section.

4.2 Individual Control Design and Adaptation

Based on the quadrotor dynamics derived above, an effective control design should be developed, dealing with the non-linear, unstable, under-actuated quadrotor system. The overall control structure is illustrated in Fig. 4.2, consisting of an inner-loop controller for stabilization of the roll and pitch angles and an outer-loop controller for quadrotor position and yaw angle control.

The quadrotor is a second-order non-linear system (the quadrotor dynamics block). It performs the translational and rotational motion in the body frame, which is observed and measured by the IMUs. The IMU data are processed and forwarded into an off-the-

shelf inner-loop controller. As mentioned in Section 3.1, this controller is implemented on the ARM-processor and contains three independent PD controllers. The roll angle, the pitch angle, and the yaw angle velocity are controlled and stabilized by these controllers at a frequency of 1 kHz, totally depending on the on-board IMU data. As control input interface, four desired variables for this inner-loop controller are needed, namely the roll angle command Φ_{cmd}, the pitch angle command Θ_{cmd}, the thrust command F_{cmd}, and the yaw angle velocity command $\dot{\Psi}_{\text{cmd}}$, which are all between 0 and 255.

Moreover, the quadrotor pose measured by the camera is fused with the high-frequency IMU data in the *Extended Kalman Filter* (EKF). The relevant system states x, y, z, $\dot{x}$, $\dot{y}$, $\dot{z}$, and Ψ are estimated. These states are predicted to compensate for the time delay in a prediction block (described in Section 4.4) and then fed into an outer-loop controller for quadrotor position and yaw angle control. With respect to the desired pose x_{d}, y_{d}, z_{d}, and Ψ_{d}, the outer-loop controller produces corresponding command output Φ_{cmd}, Θ_{cmd}, $\dot{\Psi}_{\text{cmd}}$, and the thrust F_{cmd}. In addition, the motion feedforward of the ground robot is transmitted into the outer-loop controller as well (described in Section 4.3).

To find the desired pitch/roll angles, yaw angle velocity, and thrust for the inner-loop controller, most state-of-the-art works studying VTOL have applied approximations and simplifications using small angles [35]. However, this simplification is not valid here, since the quadrotor in this work should also accomplish a tracking task with a large yaw-angle variation. To deal with this problem, a system partitioning is conducted for the system dynamics defined in Eq. 4.9. A substitution is made as follows:

$$\ddot{x} = u_x = (c\Phi s\Theta c\Psi + s\Phi s\Psi)\frac{1}{m}F + \ddot{x}_{\text{rot}} - a_r, \tag{4.15}$$

$$\ddot{y} = u_y = (c\Phi s\Theta s\Psi - s\Phi c\Psi)\frac{1}{m}F + \ddot{y}_{\text{rot}}, \tag{4.16}$$

$$\ddot{z} = u_z = c\Phi c\Theta\frac{1}{m}F + g, \tag{4.17}$$

$$\dot{\Psi} = u_\Psi - \dot{\Omega}_r. \tag{4.18}$$

Controllers are designed to find u_x, u_y, u_z, and u_Ψ in the following subsections. Then, using the current pitch and roll angles estimated by the EKF described in Chapter 3, Ψ_{d} and F_{d} can be calculated in Eq. 4.18 and Eq. 4.17, respectively. After that, F_{d} is deployed in Eq. 4.15 and Eq. 4.16, resulting in

$$c\Phi s\Theta c\Psi + s\Phi s\Psi = (u_x - \ddot{x}_{\text{rot}} + a_r)\cdot\frac{m}{F_{\text{d}}} \tag{4.19}$$

$$c\Phi s\Theta s\Psi - s\Phi c\Psi = (u_y - \ddot{y}_{\text{rot}})\cdot\frac{m}{F_{\text{d}}}. \tag{4.20}$$

where the terms right to the equal marks are available either from EKF-aided system state estimation or feedforward information of the ground robot. To determine the desired values for Φ and Θ, namely Φ_{d} and Θ_{d}, a subtraction between Eq. 4.19 multiplying with $s\Psi$ and Eq. 4.20 multiplying $c\Psi$ is conducted, resulting in

$$s\Phi = \frac{m}{F_d}(u_x\cdot s\Psi - u_y\cdot c\Psi). \tag{4.21}$$

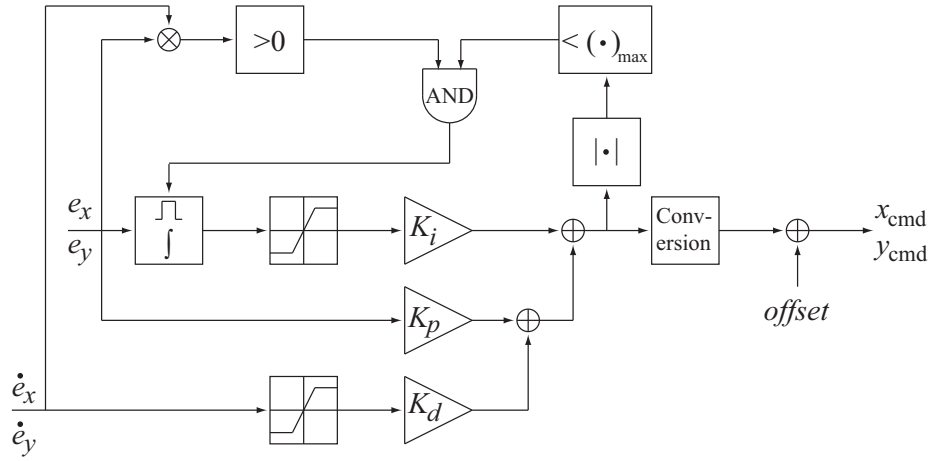

Fig. 4.3: x/y controller.

Using the yaw angle estimation from the EKF, the desired Φ_d is solved. Analogously, an addition between Eq. 4.19 multiplying with $c\Psi$ and Eq. 4.20 multiplying $s\Psi$ is conducted, resulting in

$$c\Phi s\Theta = \frac{m}{F_\mathrm{d}}(u_x \cdot c\Psi + u_y \cdot s\Psi). \tag{4.22}$$

Using the known Ψ and Φ_d, the desired Θ_d is also solved. After that, the desired values are converted into command signals F_cmd, Φ_cmd, Θ_cmd, and $\dot{\Psi}_\mathrm{cmd}$, which are forwarded into the inner-loop controller.

Now, various control strategies such as generic controllers, optimal controllers, and nonlinear controllers are applied and thoroughly adapted. In each subsection, the theoretical principle of each control type is briefly introduced. Then, the adaptation of each controller to the quadrotor system derived in this thesis is described in detail. The overall control design considering a complete flying scenario consisting of quadrotor take-off, hovering, tracking, and landing is explored in Section 4.3.

4.2.1 PID Controller

A PID controller is a generic controller widely used in the industry domain. In most state-of-the-art, real-time experiments, PID and their modifications are applied for a computationally efficient quadrotor control. In this thesis, couplings between the four channels are ignored first, which is valid for slow motion. In x- and y-directions, a linear block for small angle approximation of Φ and Θ is yielded.

Theory

The control law describing the relationship between the control error $e(t)$ and the control output $u(t)$ has the following form:

$$u(t) = K_p \cdot e(t) + K_i \int_0^t e(\tau)\mathrm{d}\tau + K_d \frac{\mathrm{d}}{\mathrm{d}t} e(t) \tag{4.23}$$

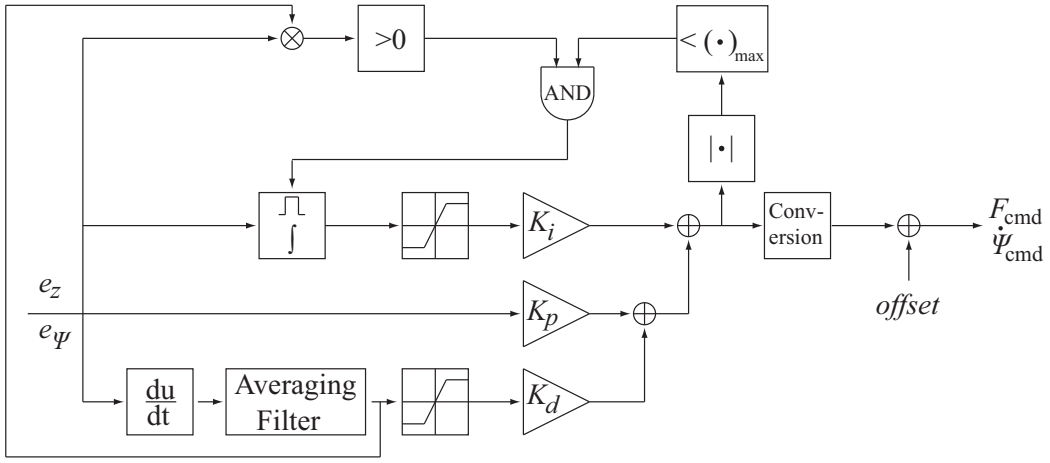

Fig. 4.4: z/Ψ controller.

where K_p, K_i, and K_d represent the proportional gain, the integral gain, and the derivative gain [16].

Adaptation

Four outer-loop controllers are implemented independently: four PID controllers for position x, y, altitude z, as well as the yaw angle velocity Ψ. The actual position, altitude, and yaw angle are provided by the EKF-aided data fusion.

For the horizontal position x and y, two identical controllers are applied, which are illustrated in Fig. 4.3. e_x and e_y are the respective control errors. The I- and D-terms have bounded outputs. To avoid the wind-up effect, the integrator is disabled for extremely large summed output. Another restriction is set to the integrator for a better performance: The integrator is enabled when the absolute value of the control error is increasing, which means $\dot{e}_x \cdot e_x > 0$ and $\dot{e}_y \cdot e_y > 0$. A conversion block and an offset convert the control outputs into the command signals.

A variation of the on-board scale factor for the accelerometer along the Z_b-axis can lead to inaccurate $\dot{z}$ estimation, and this does happen occasionally. Therefore, the z controller does not take the motion estimation $\dot{z}$ from the EKF. A differentiator and an averaging filter, which are not included in the x and y controllers, are used to gain the motion information from the control error (see Fig. 4.4). For Ψ control, the same structure as the z controller is applied.

Due to computational efficiency and relatively simple parameter adjustment, PID controllers are regarded as the reference for performance evaluation in this thesis.

4.2.2 LQ Controller

As one of the commonly used optimal controllers, LQ controllers have also been applied for quadrotor inner-loop control to stabilize the pitch, roll, and yaw angles in [35, 72]. Moreover, in [115], an LQ controller is used on a reduced system deleting yaw angle to follow a trajectory in simulations, such that the system is controllable and observable. In addition, the Matlab *Linear Quadratic Regulator* (LQR) toolbox is used in [19] to compute

the feedback controller gains, in which a stable hovering with an RMS error of 0.06 m is achieved. Therefore, LQ controllers are also investigated in this thesis, although it is assumed to be not appropriate for dynamic systems with coupled states.

Theory

An LQ controller is used in optimal control theory, if the dynamics of a controllable system can be described by a set of linear differential equations as follows:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}(t)\boldsymbol{x} + \boldsymbol{B}(t)\boldsymbol{u}; \quad \boldsymbol{x}(0) = \boldsymbol{x}_0; \\ \boldsymbol{z} &= \boldsymbol{C}(t)\boldsymbol{x}, \end{aligned} \tag{4.24}$$

while the cost functional is represented by a quadratic functional for a time horizon [16]

$$J(\boldsymbol{u}(t)) = \int_0^\infty \left[\|\boldsymbol{x}(t)\|^2_{\boldsymbol{Q}(t)} + \|\boldsymbol{u}(t)\|^2_{\boldsymbol{R}(t)} \right] \mathrm{d}t \rightarrow \min_{\boldsymbol{u}(t)}, \tag{4.25}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ denotes the system state of n dimension, $\boldsymbol{u} \in \mathbb{R}^p$ denotes the control input of p dimension, $\boldsymbol{z} \in \mathbb{R}^q$ denotes the system measurement of q dimension, and $q \leq n$. The weighting matrix $\boldsymbol{Q}(t) \geq 0$ is a semi-positive definite matrix, while $\boldsymbol{R}(t) > 0$ is positive definite.

The feedback control law that minimizes the cost functional can be formulated as follows:

$$\boldsymbol{u} = -\boldsymbol{K}\boldsymbol{x}, \tag{4.26}$$

where

$$\boldsymbol{K} = \boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P}(t) \tag{4.27}$$

and $\boldsymbol{P}(t)$ is the positive definite solution of the algebraic Matrix-Riccati-Differential equation

$$\dot{\boldsymbol{P}} = -\boldsymbol{P}\boldsymbol{A} - \boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{R} - \boldsymbol{Q}. \tag{4.28}$$

Adaptation

To apply an LQ controller in the quadrotor position and yaw control, the following system state is defined:

$$\begin{aligned} \boldsymbol{x} &= [x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5 \;\; x_6 \;\; x_7]^T \\ &= [x \;\; \dot{x} \;\; y \;\; \dot{y} \;\; z \;\; \dot{z} \;\; \Psi]^T. \end{aligned} \tag{4.29}$$

Based on the system partitioning in Eq. 4.15-4.18, the system model is first given by

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}, \tag{4.30}$$

where

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \text{ and } \boldsymbol{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.31}$$

with $\boldsymbol{u} = [u_x \;\; u_y \;\; u_z \;\; u_\Psi]$. The control laws for u_x, u_y, u_z, and u_Ψ should be calculated. Then, the actual control input for the inner-loop controller Φ_{cmd}, Θ_{cmd}, F_{cmd}, and $\dot{\Psi}_{\text{cmd}}$ are derived from the decoupling laws.

For a quadrotor tracking or landing behavior, the tracking error is to be minimized. Therefore, a substitution of the system state is conducted as follows:

$$\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}_{\text{d}}, \tag{4.32}$$

so that the following cost functional can be formulated:

$$J = \int_0^\infty \left[\|\boldsymbol{e}(t)\|^2_{\boldsymbol{Q}(t)} + \|\boldsymbol{u}(t)\|^2_{\boldsymbol{R}(t)} \right] \mathrm{d}t. \tag{4.33}$$

with the system equation:

$$\dot{\boldsymbol{e}} = \boldsymbol{A}\boldsymbol{e} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{A}\boldsymbol{x}_{\text{d}} - \dot{\boldsymbol{x}}_{\text{d}}. \tag{4.34}$$

For hovering or tracking a moving ground robot at a constant altitude, the desired system state is the relative position and velocity, namely

$$\boldsymbol{x}_d = [0 \;\; 0 \;\; 0 \;\; 0 \;\; z_{\text{d}} \;\; \dot{z}_{\text{d}} \;\; 0]^T, \tag{4.35}$$

with constant z_{d} and $\dot{z}_{\text{d}}$ and

$$\dot{\boldsymbol{x}}_{\text{d}} = \boldsymbol{0}. \tag{4.36}$$

Therefore,

$$\boldsymbol{A}\boldsymbol{x}_{\text{d}} - \dot{\boldsymbol{x}}_{\text{d}} = \boldsymbol{0}, \tag{4.37}$$

and

$$\dot{\boldsymbol{e}} = \boldsymbol{A}\boldsymbol{e} + \boldsymbol{B}\boldsymbol{u}. \tag{4.38}$$

To test the system controllability, the function ctrb$(\cdot)$ from the MATLAB control system toolbox is used. The controllability matrix has a full rank of 7 and the system is fully controllable. For experiments, the control law Eq. 4.26 is first computed off-line using the function lqr$(\cdot)$ from the MATLAB control system toolbox and applied online.

Note that for take-off or landing behavior, the desired quadrotor altitude z_{d} and vertical velocity $\dot{z}_{\text{d}}$ vary all the time, so that Eq. 4.37 and 4.38 are not valid. Therefore, LQ controllers can only be applied for hovering and tracking behavior.

4.2.3 Backstepping-Based Controller

Since the quadrotor possesses non-linear dynamics, one of the non-linear controllers – the backstepping technique – is considered [83] in order to ensure the Lyapunov stability.

Theory

Backstepping is a recursive control design technique for non-linear systems in a strictly feedback form [17] given by

$$\begin{aligned}
\dot{x}_1 &= f_1(x_1) + g_1(x_1)x_2; && (4.39)\\
\dot{x}_2 &= f_2(x_1, x_2) + g_2(x_1, x_2)x_3; && (4.40)\\
&\vdots \\
\dot{x}_i &= f_i(x_1, \cdots, x_i) + g_i(x_1, \cdots, x_i)x_{i+1};\\
&\vdots \\
\dot{x}_n &= f_n(x_1, \cdots, x_n) + g_n(x_1, \cdots, x_n)u && (4.41)
\end{aligned}$$

with system state $\boldsymbol{x} \in \mathbb{R}^n$, $f_1(0) = \cdots = f_n(0) = 0$, and $g_i(x_1, \cdots, x_i) \neq 0$ for $1 \leq i \leq n$.

For the systems in this recursive structure, each sub-system with the system state $\tilde{\boldsymbol{x}}_{i-1} = [x_1, \cdots, x_{i-1}]^T$ can be represented as a system plant with a system input x_i. For instance, $x_2 = \beta_1$ can be regarded as a "pseudo control input" for the sub-system 4.39 and be so defined that this sub-system is stabilized at the idle state $x_1^* = 0$. Then, for the sub-system 4.40, $x_3 = \beta_2$ is interpreted as the "pseudo input", which stabilizes the sub-system 4.40 at the idle state $x_2^* = 0$. Finally, the system input u is selected such that the overall system is stabilized. The "pseudo control input" is defined by means of Lyapunov function, which ensures the Lyapunov stability at each step.

The application of the backstepping technique on the quadrotor flight control is described below in detail. Two different backstepping-based controllers, the integrator backstepping and the integral backstepping, are introduced.

System Dynamic Model

Similar to the previous section on LQ controllers, the system state is given by Eq. 4.29. Using integrator backstepping and integral backstepping, the control laws for u_x, u_y, u_z, and u_Ψ are calculated.

Adaptation of Integrator Backstepping

Integrator Backstepping is the simplest form of a backstepping-based controller. As mentioned above, a stability of the subsystem at the idle state $x_i^* = 0$ is desired. In this thesis, the system state should be controlled to follow the desired trajectory and minimize the tracking error. Therefore, a new system state $\boldsymbol{\zeta} = \boldsymbol{x}_\mathrm{d} - \boldsymbol{x}$ is introduced such that each sub-system is stabilized at $\zeta_i^* = 0$ using the "pseudo control input".

Step 1: Then,

$$\zeta_1 = x_{1\mathrm{d}} - x_1, \tag{4.42}$$

where $x_{1\mathrm{d}}$ is the desired value of system state x_1.

The backstepping-based controller uses Lyapunov theorem to ensure the stability at the idle state, a positive definite Lyapunov function V should be defined for ζ_1:

$$V(\zeta_1) = \frac{1}{2}\zeta_1^2, \tag{4.43}$$

and the time derivatives of ζ_1 and $V(\zeta_1)$ are given by

$$\dot{\zeta}_1 = \dot{x}_{1\mathrm{d}} - \dot{x}_1, \tag{4.44}$$

$$\dot{V}(\zeta_1) = \zeta_1\dot{\zeta}_1. \tag{4.45}$$

From $\dot{x}_1 = x_2$ in Eq. 4.30 and 4.31, x_2 is defined here as a pseudo control input β_1. Then,

$$\dot{V}(\zeta_1) = \zeta_1(\dot{x}_{1\mathrm{d}} - \beta_1). \tag{4.46}$$

To let $\dot{V}(\zeta_1)$ be negative definite and achieve the Lyapunov stability through that, $\dot{V}(\zeta_1)$ can be represented by

$$\dot{V}(\zeta_1) = -\alpha_1\zeta_1^2 \quad \text{with} \quad \alpha_1 > 0. \tag{4.47}$$

Then, the pseudo control input β_1 should be chosen as follows:

$$\beta_1 = \dot{x}_{1\mathrm{d}} + \alpha_1\zeta_1. \tag{4.48}$$

α_1 is a control parameter which will be experimentally determined.

Step 2: For the system state x_2, ζ_2 is introduced:

$$\zeta_2 = x_{2\mathrm{d}} - x_2, \tag{4.49}$$

where $x_{2\mathrm{d}} = \beta_1$ is the desired value for x_2. Therefore, based on Eq. 4.48

$$\zeta_2 = \beta_1 - x_2 = \dot{x}_{1\mathrm{d}} + \alpha_1\zeta_1 - x_2. \tag{4.50}$$

A control term for velocity control is considered here. From Eq. 4.44 and 4.50,

$$\dot{\zeta}_1 = \dot{x}_{1\mathrm{d}} - \dot{x}_1 = \dot{x}_{1\mathrm{d}} - x_2 = \zeta_2 - \alpha_1\zeta_1. \tag{4.51}$$

The Lyapunov function is now formulated as

$$V(\zeta_1, \zeta_2) = \frac{1}{2}\zeta_1^2 + \frac{1}{2}\zeta_2^2, \tag{4.52}$$

and its time derivative

$$\dot{V}(\zeta_1, \zeta_2) = \zeta_1\dot{\zeta}_1 + \zeta_2\dot{\zeta}_2. \tag{4.53}$$

The time derivative of ζ_2 is given by Eq. 4.50:

$$\dot{\zeta}_2 = -\dot{x}_2 + \ddot{x}_{1\mathrm{d}} + \alpha_1 \dot{\zeta}_1 \tag{4.54}$$

Based on the system model Eq. 4.15 and Eq. 4.51, $\dot{\zeta}_2$ is written in

$$\dot{\zeta}_2 = -u_x + \ddot{x}_{1\mathrm{d}} + \alpha_1(\zeta_2 - \alpha_1\zeta_1) \tag{4.55}$$

Based on Eq. 4.55 and 4.51, the time derivative of the Lyapunov function $V(\zeta_1, \zeta_2)$ can be formulated as follows:

$$\begin{aligned} \dot{V}(\zeta_1, \zeta_2) &= \zeta_1(\zeta_2 - \alpha_1\zeta_1) + \zeta_2(\ddot{x}_{1\mathrm{d}} - u_x + \alpha_1(\zeta_2 - \alpha_1\zeta_1)) \\ &= -\alpha_1\zeta_1^2 + \zeta_1\zeta_2 + \zeta_2(\ddot{x}_{1\mathrm{d}} - u_x + \alpha_1(\zeta_2 - \alpha_1\zeta_1)) \\ &= -\alpha_1\zeta_1^2 + \zeta_2(\zeta_1 + \ddot{x}_{1\mathrm{d}} - u_x + \alpha_1(\zeta_2 - \alpha_1\zeta_1)) \end{aligned} \tag{4.56}$$

To let $\dot{V}(\zeta_1, \zeta_2)$ be negative definite in the following representation,

$$\dot{V}(\zeta_1, \zeta_2) = -\alpha_1\zeta_1^2 - \alpha_2\zeta_2^2, \quad \text{with} \quad \alpha_2 > 0, \tag{4.57}$$

the real control input u_x should be formulated as follows:

$$u_x = (1 - \alpha_1^2) \cdot \zeta_1 + (\alpha_1 + \alpha_2) \cdot \zeta_2 + \ddot{x}_{\mathrm{d}}, \tag{4.58}$$

where $\ddot{x}_{1\mathrm{d}} = \ddot{x}_{\mathrm{d}}$.

Step 3: Analogously, the other control input u_y and u_z can be calculated as follows:

$$\begin{aligned} u_y &= (1 - \alpha_3^2)\zeta_3 + (\alpha_3 + \alpha_4)\zeta_4 + \ddot{y}_{\mathrm{d}}, \\ u_z &= (1 - \alpha_5^2)\zeta_5 + (\alpha_5 + \alpha_6)\zeta_6 + \ddot{z}_{\mathrm{d}}, \end{aligned} \tag{4.59}$$

where

$$\begin{aligned} &\alpha_3, \alpha_4, \alpha_5, \alpha_6 > 0, \\ &\zeta_3 = x_{3\mathrm{d}} - x_3, \\ &\zeta_4 = \dot{x}_{3\mathrm{d}} + \alpha_3\zeta_3 - x_4, \\ &\zeta_5 = x_{5\mathrm{d}} - x_5, \\ &\zeta_6 = \dot{x}_{5\mathrm{d}} + \alpha_5\zeta_5 - x_6. \end{aligned} \tag{4.60}$$

Step 4: For the yaw angle control, a new system state z_7 is defined similar to step 1:

$$\begin{aligned} \zeta_7 &= x_{7\mathrm{d}} - x_7, \\ \dot{\zeta}_7 &= \dot{x}_{7\mathrm{d}} - \dot{x}_7 = \dot{x}_{7\mathrm{d}} - u_\Psi. \end{aligned} \tag{4.61}$$

Then,

$$u_\Psi = \alpha_7\zeta_7 + \dot{\Psi}_{\mathrm{d}}. \tag{4.62}$$

Adaptation of Integral Backstepping

The *Integral Backstepping Controller* (IBC) is an improved version of the integrator backstepping. The difference is that in the former, an integral-term is comprised in the definition of the Lyapunov functions. Its applications showed control improvement in the quadrotor flight performance [36, 122]. Therefore, IBC is also investigated in this work and considered for the control of the quadrotor position x, y, the altitude z, and the yaw angle Ψ.

Step 1: Similar to the integrator backstepping, the new system state is defined as the control error:

$$\begin{aligned} \zeta_1 &= x_{1\mathrm{d}} - x_1, \\ \dot{\zeta}_1 &= \dot{x}_{1\mathrm{d}} - \dot{x}_1, \end{aligned} \tag{4.63}$$

and the Lyapunov function $V(\zeta_1)$ is defined as follows:

$$\begin{aligned} V(\zeta_1) &= \tfrac{1}{2}\zeta_1^2 + \tfrac{1}{2}\lambda_1\chi_1^2, \\ \dot{V}(\zeta_1) &= \zeta_1\dot{\zeta}_1 + \lambda_1\chi_1\zeta_1, \end{aligned} \tag{4.64}$$

where

$$\chi_1 = \int_0^t \zeta_1(\tau)\mathrm{d}\tau \tag{4.65}$$

representing the integral term. A pseudo control input $\beta_1 = x_2$ is defined for $\dot{x}_1 = x_2$. Then,

$$\dot{\zeta}_1 = \dot{x}_{1\mathrm{d}} - \beta_1, \tag{4.66}$$

and therefore

$$\dot{V}(\zeta_1) = \zeta_1(\dot{x}_{1\mathrm{d}} - \beta_1) + \lambda_1\chi_1\zeta_1. \tag{4.67}$$

To let $\dot{V}(\zeta_1)$ be negative semi-definite and ensure the stability by applying the La Salle theorem,

$$\dot{V}(\zeta_1) = -\alpha_1\zeta_1^2, \quad \text{with} \quad \alpha_1 > 0, \tag{4.68}$$

the pseudo control input β_1 should be formulated as follows:

$$\beta_1 = \dot{x}_{1\mathrm{d}} + \alpha_1\zeta_1 + \lambda_1\chi_1. \tag{4.69}$$

Step 2: Similarly, another new state ζ_2 is defined as follows:

$$\zeta_2 = x_{2\mathrm{d}} - x_2 = \beta_1 - x_2. \tag{4.70}$$

In contrast to [35], the new Lyapunov function is defined with a further integral term $\frac{1}{2}\lambda_2 p_2^2$ and the stability is analyzed:

$$V(\zeta_1, \zeta_2) = \frac{1}{2}\zeta_1^2 + \frac{1}{2}\zeta_2^2 + \frac{1}{2}\lambda_1\chi_1^2 + \frac{1}{2}\lambda_2\chi_2^2. \tag{4.71}$$

Its time derivative is written as follows:

$$\dot{V}(\zeta_1, \zeta_2) = \zeta_1\dot{\zeta}_1 + \zeta_2\dot{\zeta}_2 + \lambda_1\chi_1\zeta_1 + \lambda_2\chi_2\zeta_2, \tag{4.72}$$

where

$$\chi_2 = \int_0^t \zeta_2(\tau)\mathrm{d}\tau \tag{4.73}$$

From Eq. 4.70 and 4.69, the following equation can be derived:

$$\zeta_2 = \dot{x}_{1\mathrm{d}} + \alpha_1\zeta_1 + \lambda_1\chi_1 - x_2. \tag{4.74}$$

Since $x_2 = \dot{x}_1$ and $\dot{\zeta}_1 = \dot{x}_{1\mathrm{d}} - \dot{x}_1$,

$$\dot{\zeta}_1 = \zeta_2 - \alpha_1\zeta_1 - \lambda_1\chi_1, \tag{4.75}$$

and then

$$\begin{aligned} \dot{\zeta}_2 &= \ddot{x}_{1\mathrm{d}} + \alpha_1\dot{\zeta}_1 + \lambda_1\zeta_1 - \dot{x}_2 \\ &= \ddot{x}_{1\mathrm{d}} + \alpha_1(\zeta_2 - \alpha_1\zeta_1 - \lambda_1\chi_1) + \lambda_1\zeta_1 - \dot{x}_2 \\ &= \ddot{x}_{1\mathrm{d}} - u_x + \alpha_1(\zeta_2 - \alpha_1\zeta_1 - \lambda_1\chi_1) + \lambda_1\zeta_1. \end{aligned} \tag{4.76}$$

Substituting $\dot{\zeta}_1$ and $\dot{\zeta}_2$ in Eq. 4.72, the following equation is obtained:

$$\begin{aligned} \dot{V}(\zeta_1, \zeta_2) &= \zeta_1\dot{\zeta}_1 + \zeta_2\dot{\zeta}_2 + \lambda_1\chi_1\zeta_1 + \lambda_2\chi_2\zeta_2 \\ &= \zeta_1(\zeta_2 - \alpha_1\zeta_1 - \lambda_1\chi_1) + \zeta_2(\ddot{x}_{1\mathrm{d}} - u_x + \alpha_1(\zeta_2 - \alpha_1\zeta_1 - \lambda_1\chi_1) + \lambda_1\zeta_1) \\ &\quad +\lambda_1\chi_1\zeta_1 + \lambda_2\chi_2\zeta_2 \\ &= -\alpha_1\zeta_1^2 + \zeta_1\zeta_2 + \lambda_2\chi_2\zeta_2 + \zeta_2(\ddot{x}_{1\mathrm{d}} - u_x + \alpha_1(\zeta_2 - \alpha_1\zeta_1 - \lambda_1\chi_1) + \lambda_1\zeta_1) \\ &= -\alpha_1\zeta_1^2 + \zeta_2(\zeta_1 + \ddot{x}_{1\mathrm{d}} - u_x + \alpha_1(\zeta_2 - \alpha_1\zeta_1 - \lambda_1\chi_1) + \lambda_1\zeta_1 + \lambda_2\chi_2). \end{aligned} \tag{4.77}$$

For the stability by applying the La Salle theorem,

$$\dot{V}(\zeta_1, \zeta_2) = -\alpha_1\zeta_1^2 - \alpha_2\zeta_2^2, \quad \text{with} \quad \alpha_2 > 0. \tag{4.78}$$

Then, the control input u_x should be formulated as follows:

$$\begin{aligned} u_x &= \alpha_2\zeta_2 + \zeta_1 + \alpha_1(\zeta_2 - \alpha_1\zeta_1 - \lambda_1\chi_1) + \lambda_1\zeta_1 + \lambda_2\chi_2 + \ddot{x}_{1\mathrm{d}} \\ &= \zeta_1(-\alpha_1^2 + 1 + \lambda_1) + \zeta_2(\alpha_1 + \alpha_2) - \alpha_1\lambda_1\chi_1 + \lambda_2\chi_2 + \ddot{x}_{\mathrm{d}}, \end{aligned} \tag{4.79}$$

where $\ddot{x}_{1\mathrm{d}} = \ddot{x}_{\mathrm{d}}$.

Step 3: Analogously, u_y and u_z are computed as follows:

$$u_y = \zeta_3(-\alpha_3^2 + 1 + \lambda_3) + \zeta_4(\alpha_3 + \alpha_4) - \alpha_3\lambda_3\chi_3 + \lambda_4\chi_4 + \ddot{y}_{\mathrm{d}}, \tag{4.80}$$

$$u_z = \zeta_5(-\alpha_5^2 + 1 + \lambda_5) + \zeta_6(\alpha_5 + \alpha_6) - \alpha_5\lambda_5\chi_5 + \lambda_6\chi_6 + \ddot{z}_{\mathrm{d}}, \tag{4.81}$$

where

$$\begin{aligned}
\zeta_3 &= x_{3d} - x_3, & \zeta_4 &= \dot{x}_{3\mathrm{d}} + \alpha_3\zeta_3 + \lambda_3\chi_3 - x_4,\\
\chi_3 &= \int_0^t \zeta_3(\tau)\mathrm{d}\tau, & \chi_4 &= \int_0^t \zeta_4(\tau)\mathrm{d}\tau,\\
\zeta_5 &= x_{5\mathrm{d}} - x_5, & \zeta_6 &= \dot{x}_{5\mathrm{d}} + \alpha_5\zeta_5 + \lambda_5\chi_5 - x_6,\\
\chi_5 &= \int_0^t \zeta_5(\tau)\mathrm{d}\tau, & \chi_6 &= \int_0^t \zeta_6(\tau)\mathrm{d}\tau.
\end{aligned} \tag{4.82}$$

Step 4: Analogously, the control input for the yaw angle control u_Ψ is computed as follows:

$$u_\Psi = \alpha_7\zeta_7 + \lambda_7\chi_7 + \dot{\Psi}_\mathrm{d} \tag{4.83}$$

where

$$\begin{aligned}
\zeta_7 &= x_{7\mathrm{d}} - x_7\\
\chi_7 &= \textstyle\int_0^t \zeta_7(\tau)\mathrm{d}\tau
\end{aligned} \tag{4.84}$$

Finally, the real control input for the inner-loop controller Φ_cmd, Θ_cmd, F_cmd, and $\dot{\Psi}_\mathrm{cmd}$ are computed as described in Section 4.2.

4.2.4 Sliding Mode Control

Sliding Mode Control (SMC) is a nonlinear control method with the main advantage of robustness to parameter variations as well as unmodeled dynamics and variational external noise. During the quadrotor landing and take-off, especially when the quadrotor flies near the surface of the mobile robot or the ground, unmodeled force in the system, such as ground effect, drag, and inaccuracy of the thrust equation, is not negligible. Therefore, sliding mode control is chosen to control the vertical motion of the quadrotor in the landing and take-off processes, while the horizontal motion and orientation are controlled using other controllers.

Before the control design based on sliding mode control is described, path planning during the landing and take-off is presented, considering smooth motion and position/velocity constraints at the start and end points. Here, the quadrotor altitude $h = -z$ is assigned a desired value $h_\mathrm{d}(t)$. The quadrotor starts to take off from 0 m, while desired altitude of hovering and tracking is denoted by h_0. The end altitude after landing is also 0 m.

SMC in Landing

To achieve a safe landing, a suitable landing path planning should be considered. Both the altitude h and the descending speed $\dot{h}$ along the negative Z_I/Z_o-axis are to be controlled. An exponential trajectory is chosen for the landing process and can be written as

$$h_\mathrm{d}(t) = h_0 \cdot e^{-\sigma\cdot t}, \tag{4.85}$$

where h_0 is the start altitude of landing, σ is the damping factor. Then the desired velocity and acceleration are

$$\dot{h}_\mathrm{d}(t) = -\sigma \cdot h_0 \cdot e^{-\sigma\cdot t} \tag{4.86}$$

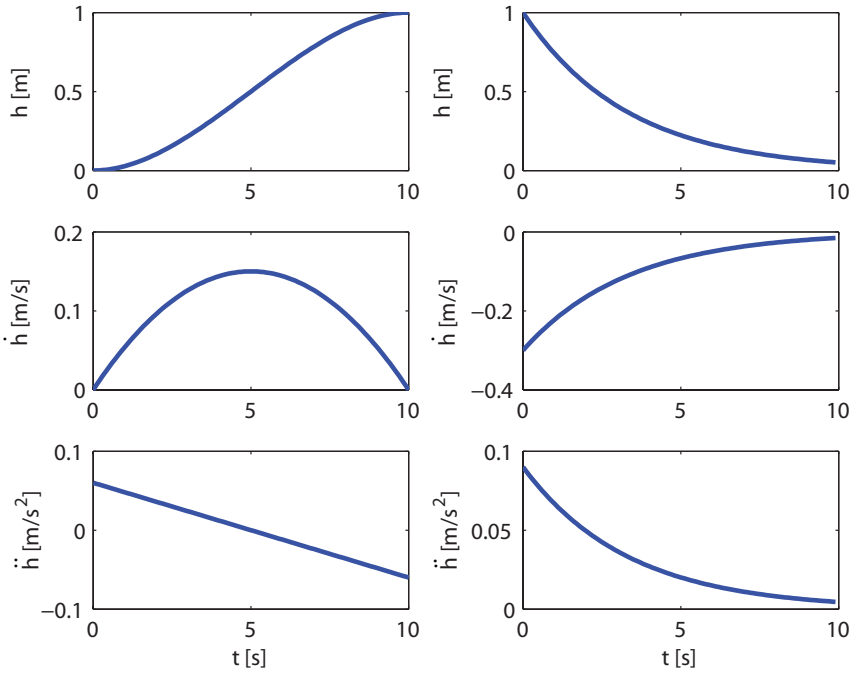

Fig. 4.5: Desired altitude, vertical velocity, and vertical acceleration in take-off (left) and landing (right).

and

$$\ddot{h}_{\mathrm{d}}(t) = \sigma^2 \cdot h_0 \cdot e^{-\sigma \cdot t}. \tag{4.87}$$

The right column in Fig. 4.5 illustrates the desired altitude, velocity, and acceleration in the vertical direction during landing for a start altitude $h_0 = 1\,\mathrm{m}$ with $\sigma = 0.3$. This path provides a maximum vertical velocity at the beginning and a zero vertical velocity at the end point, which are very essential and practical for landing.

The thrust F projected along the Z_I-/Z_o-axis is denoted as F_z. Based on Eq. 4.4, the vertical acceleration $\ddot{h}$ can be expressed by

$$\ddot{h} = -\frac{F_z + m \cdot g + F_{\mathrm{unm},z}}{m} = -\frac{F_z}{m} - g - \frac{F_{\mathrm{unm},z}}{m}, \tag{4.88}$$

where m is the mass of the quadrotor, $F_{\mathrm{unm},z}$ is unmodeled force and noise in the system projected along the Z_I-/Z_o-axis, such as drag, inaccuracy of the thrust equation, or the reflected air flow from the ground.

To ensure the quadrotor motion along the desired trajectory through compensating for the unmodeled force, the control input $F_{\mathrm{cmd},z}$ can be chosen as follows :

$$F_z = U \cdot F_{\mathrm{cmd},z} \cdot K = -m \cdot c \cdot (\dot{h}_{\mathrm{d}} - \dot{h}) - m \cdot \ddot{h}_{\mathrm{d}} - m \cdot g, \tag{4.89}$$

where c is a proportional gain.

By substituting Eq. 4.85, 4.86, and 4.87 into 4.89, the input $F_{\mathrm{cmd},z}$ is expressed by

$$F_{\mathrm{cmd},z} = \frac{F_z}{U \cdot K} = \frac{m \cdot c \cdot (\sigma \cdot h + \dot{h}) - m \cdot \sigma^2 \cdot h - m \cdot g}{U \cdot K}. \tag{4.90}$$

Then Eq. 4.88 can be rewritten as

$$\ddot{h} = (-c \cdot \sigma + \sigma^2) \cdot h - c \cdot \dot{h} - \frac{F_{\mathrm{unm},z}}{m}. \tag{4.91}$$

Because of the unmodeled acceleration $\frac{F_{\mathrm{unm},z}}{m}$, no proof can be given that h and $\dot{h}$ would converge exponentially to zero. Therefore an extra term based on the sliding mode control is added to Eq. 4.89 and 4.90 as follows:

$$F_z = -m \cdot c \cdot (\dot{h}_{\mathrm{d}} - \dot{h}) - m \cdot \ddot{h}_{\mathrm{d}} + M \cdot \mathrm{sign}(s) - m \cdot g, \tag{4.92}$$

$$F_{\mathrm{cmd},z} = \frac{F_z}{U \cdot K} = \frac{m \cdot c \cdot (\sigma \cdot h + \dot{h}) - m \cdot \sigma^2 \cdot h + M \cdot \mathrm{sign}(s) - m \cdot g}{U \cdot K}, \tag{4.93}$$

where M' is a positive constant and s is the switching line with

$$s = \sigma \cdot h + \dot{h}. \tag{4.94}$$

Then $\ddot{h}$ is modified to be

$$\ddot{h} = (-c \cdot \sigma + \sigma^2) \cdot h - c \cdot \dot{h} - \frac{M}{m} \cdot sign(s) + \frac{F_{\mathrm{unm},z}}{m}. \tag{4.95}$$

The domain of a sliding mode can be determined by

$$s \cdot \dot{s} < 0, \text{ for } s \neq 0. \tag{4.96}$$

Based on Eq. 4.94 and 4.95

$$\dot{s} = \sigma \cdot \dot{h} + \ddot{h} = -(c - \sigma) \cdot (\sigma \cdot h + \dot{h}) - \frac{M}{m} \cdot sign(s) - \frac{F_{\mathrm{unm},z}}{m}. \tag{4.97}$$

By the examination of the sliding mode domain, Eq. 4.96 is rewritten as

$$s \cdot \dot{s} = -(c - \sigma) \cdot (\sigma \cdot h + \dot{h})^2 - \frac{M}{m} \cdot s \cdot sign(s) - s \cdot \frac{F_{\mathrm{unm},z}}{m} < 0, \text{ for } s \neq 0. \tag{4.98}$$

which is held, if the conditions

$$c > \sigma \tag{4.99}$$

and

$$M > |\frac{F_{\mathrm{unm},z}}{m}|_{\max} \tag{4.100}$$

are held, where $|\frac{F_{\mathrm{unm},z}}{m}|_{max}$ is the maximal absolute value of $\frac{F_{\mathrm{unm},z}}{m}$, the trajectory in h-$\dot{h}$ space will reach the switching line s.

If the sling mode exists for the whole switching line, h and $\dot{h}$ are governed only by s.

They can be expressed by

$$h(t) = h(t_{\text{reach}}) \cdot e^{-\sigma \cdot (t - t_{\text{reach}})} \tag{4.101}$$

and

$$\dot{h}(t) = -\sigma \cdot h(t_{\text{reach}}) \cdot e^{-\sigma \cdot (t - t_{\text{reach}})}, \tag{4.102}$$

where t_{reach} is the time of reaching the switching line.

To prove the existence of the sliding model on the switching line s, the following conditions have to be fulfilled:

$$\lim_{s \to 0^+} \dot{s} < 0 \text{ and } \lim_{s \to 0^-} \dot{s} > 0. \tag{4.103}$$

while for the switching line, there is

$$s = \sigma \cdot h + \dot{h} = 0. \tag{4.104}$$

By substituting Eq. 4.104 into 4.97, the following equation is obtained:

$$\dot{s} = -(c - \sigma) \cdot s - \frac{M}{m} \cdot \text{sign}(s) - \delta_z' = -\frac{M}{m} \cdot \text{sign}(s) - \frac{F_{\text{unm},z}}{m}. \tag{4.105}$$

To fulfill Eq. 4.103, there must be

$$M > |\frac{F_{\text{unm},z}}{m}|_{\max}.$$

Summarized, Eq. 4.99 and 4.100 should be held, so that the switching line can be reached from an arbitrary start point and the states h, $\dot{h}$ are only governed by s after reaching the switching line. The exponential convergence is then achieved.

SMC in Take-Off

Less critical than landing is the quadrotor take-off phase. For take-off, a cubic spline trajectory for the vertical direction is planned. A cubic spline brings the advantage that both the start and end velocities are zero and is formulated as

$$h_{\text{d}}(t) = 3 \cdot \frac{h_0}{T_t^2} \cdot t^2 - 2 \cdot \frac{h_0}{T_t^3} \cdot t^3, \tag{4.107}$$

where h_0 is the desired altitude after the take-off and T_t is the take-off time interval. Then the desired velocity is

$$\dot{h}_{\text{d}}(t) = 6 \cdot \frac{h_0}{T_t^2} \cdot t - 6 \cdot \frac{h_0}{T_t^3} \cdot t^2 \tag{4.108}$$

with the start vertical velocity $\dot{h}_d(t = 0) = 0$ and the vertical velocity $\dot{h}_d(t = T_t) = 0$ after the take-off phase. The desired acceleration is

$$\ddot{h}_{\text{d}}(t) = 6 \cdot \frac{h_0}{T_t} - 12 \cdot \frac{h_0}{T_t^3} \cdot t. \tag{4.109}$$

Aiming at a desired altitude of 1 m directly over the mobile ground robot, the evolution of the height, the vertical velocity, and the vertical acceleration during the take-off time

$T_t = 10\,\mathrm{s}$ are shown in Fig. 4.5 (left).

The SMC design for take-off is described below, similarly to the one for landing. The control error e_h during the take-off phase can be defined as

$$e_h = h_\mathrm{d} - h. \tag{4.110}$$

Therefore, the Lyapunov function is written in

$$V(e_h) = \frac{1}{2} e_h^2, \tag{4.111}$$

and its time derivative $\dot{V}(e_h)$ should be smaller than zero. Then, by defining

$$\dot{V}(e_h) = e_h \cdot \dot{e}_h = -\alpha \cdot e_h^2 < 0, \text{ with } \alpha > 0, \tag{4.112}$$

the following equation is obtained:

$$\dot{e}_h = -\alpha \cdot e_h. \tag{4.113}$$

Based on Eq. 4.110,

$$\dot{e}_h = \dot{h}_\mathrm{d} - \dot{h}, \tag{4.114}$$

and then,

$$\dot{h} = \dot{h}_\mathrm{d} - \dot{e}_h = \dot{h}_\mathrm{d} + \alpha \cdot e_h \tag{4.115}$$

Based on Eq. 4.115, the switching surface s is defined as follows:

$$\begin{aligned} s &= \dot{h}_\mathrm{d} + \alpha \cdot e_h - \dot{h} \\ &= \dot{e}_h + \alpha \cdot e_h \end{aligned} \tag{4.116}$$

and

$$\dot{s} = \ddot{e}_h + \alpha \cdot \dot{e}_h \tag{4.117}$$

The domain of a sliding mode can be determined by

$$s \cdot \dot{s} < 0, \quad \text{for} \quad s \neq 0, \tag{4.118}$$

where

$$\begin{aligned} s \cdot \dot{s} &= (\dot{e}_h + \alpha \cdot e_h) \cdot (\ddot{e}_h + \alpha \cdot \dot{e}_h) \\ &= \dot{e}_h \cdot \ddot{e}_h + \alpha \cdot \dot{e}_h^2 + \alpha \cdot e_h \cdot \ddot{e}_h + \alpha^2 \cdot e_h \cdot \dot{e}_h \\ &= \ddot{e}_h \cdot (\dot{e}_h + \alpha \cdot e_h) + \alpha \cdot \dot{e}_h^2 + \alpha^2 \cdot e_h \cdot \dot{e}_h. \end{aligned} \tag{4.119}$$

To let $s \cdot \dot{s} < 0$, $\ddot{e}_h$ should be formulated as follows:

$$\ddot{e}_h = -\alpha \cdot \dot{e}_h - M_1 \cdot \mathrm{sign}(s) - M_2 \cdot s, \tag{4.120}$$

where $M_1 > 0$ and $M_2 > 0$. Then

$$\ddot{h} = \ddot{h}_{\mathrm{d}} + \alpha \cdot \dot{e}_h + M_1 \cdot \mathrm{sign}(s) + M_2 \cdot s, \tag{4.121}$$

where

$$\ddot{h} = -(\frac{F_z}{m} + g + \frac{F_{\mathrm{unm},z}}{m}). \tag{4.122}$$

To examine the existence of the sliding mode, Eq. 4.103 should be fulfilled. Thereby, based on Eq. 4.117 and 4.120,

$$\dot{s} = -M_1 \cdot \mathrm{sign}(s) - M_2 \cdot s < 0 \quad \text{for} \quad s \to 0^+, \tag{4.123}$$

and

$$\dot{s} = -M_1 \cdot \mathrm{sign}(s) - M_2 \cdot s > 0 \quad \text{for} \quad s \to 0^-. \tag{4.124}$$

Therefore, the sliding mode exists for $M_1 > 0$ and $M_2 > 0$.

The individual control design and adaptation to the system model presented in this thesis serve as a basis for an integrated control design combining the advantages, which is described in the next section.

4.3 Integrated Control Design Based on Simulation Results

The objective of this chapter is to develop an integrated control design for a complete flying of the quadrotor consisting of take-off from the top of a stationary mobile ground robot, hovering over it, tracking the robot movement, and landing on the moving robot. Each phase possesses its own distinct characteristics and requires a proper control design. The properties of each flying phase are described below:

- In the take-off scenario, the quadrotor is supposed to take off from the upper surface of the ground robot along a cubic spline trajectory in the vertical direction, such that the quadrotor starts with a zero and ascending velocity and ends in hovering at a desired altitude directly over the ground robot with a zero vertical velocity. In the forepart of take-off, the markers cannot be robustly and completely detected. Therefore, optical flow based motion estimation is needed for the quadrotor control.

- In the hovering scenario, the quadrotor is supposed to hover over the ground robot, which has no motion on the ground plane. Since the robot is not moving, the desired values for the quadrotor position, altitude, and orientation in the inertial world are fixed.

- In the tracking scenario, the quadrotor is supposed to track the ground robot, which has motion on the ground plane. The desired value for the quadrotor's altitude is fixed, while the desired values for the quadrotor position and yaw angle in the inertial

frame vary with the position and the orientation of the ground robot. A stable controller is desired here to enable the quadrotor to respond quickly and robustly to robot motion, considering the quadrotor non-linear dynamics and disturbance, e.g., caused by wind.

- In the landing scenario, the quadrotor is supposed to track the moving ground robot and land on its top. The desired quadrotor position and yaw angle vary with the position and the orientation of the ground robot, while the desired altitude also varies. Overall, the landing phase is more critical than the take-off phase. First, in the landing phase, the quadrotor also has to track the ground robot movement, while the ground robot is stationary in the take-off phase. Moreover, the requirement of an accurate position estimation and control is higher in landing, as the quadrotor has to safely and smoothly land on the moving robot. In addition, the unmodeled force gets larger and larger in landing, while it gets smaller and smaller in take-off.

The performance of PID, LQ, IBC, and SMC using various desired trajectories of the ground robot is simulated in Matlab and compared using a similar simulation model to that in [28], which is modified here based on the system model derived in this thesis, to validate the control strategies and the derivative model of the quadrotor. Some representative results are shown in this section. Based on the simulation results, an integrated control design is developed and applied to the real quadrotor system.

4.3.1 Comparison of Different Controllers

In the first simulation, PID, LQ, IBC, and SMC are applied to a quadrotor, which performs take-off in the first 10 s along a cubic spline trajectory in the vertical direction, hovering and tracking a mobile ground robot from 10 s to 50 s, and landing in the time interval between 50 s and 60 s. The initial position of the quadrotor relative to the ground robot is $[0 \;\; 0 \;\; 0]^T$ m. The desired height during the hovering and tracking is 1 m over the ground robot and the quadrotor should have the same orientation as the ground robot. In the landing part, unmodeled force is simulated and applied, in which the horizontal noise is a zero-mean Gaussian noise with a variance of $0.01\,\mathrm{N}^2$, while the vertical noise is a Gaussian noise with a mean value of 0.2 N and a variance of $0.01\,\mathrm{N}^2$. The mobile robot is static in the first 15 s and moves straight forward at a linear velocity of 0.1 m/s. After moving 2.6 m, the robot turns 90 deg to its left and moves straight forward. The rotational velocity of the ground robot is 0.3 rad/s ($\approx$17.19 deg/s).

The control performance is illustrated in Fig. 4.6. Controllers such as IBC (solid lines), LQ (dotted lines), and PID (dot-dash lines) are applied in different phases, while the SMC controller is only applied in the landing phase (dashed lines). The control errors e_x and e_y, the quadrotor height h, the vertical velocity $\dot{h}$, and the control error of the yaw angle e_Ψ are shown. In the take-off, PID and IBC perform very well and converge quickly to the desired hovering height, while LQ, which is assumed to be inappropriate for the take-off and landing processes, needs longer to stabilize at 1 m. In the hovering and tracking, IBC performs the best in terms of smaller control errors compared to PID and LQ, especially after the robot rotation at approximately 33 s. The maximum control errors e_x and e_y using IBC are approximately 0.06 m and the maximum e_Ψ is 6 deg.

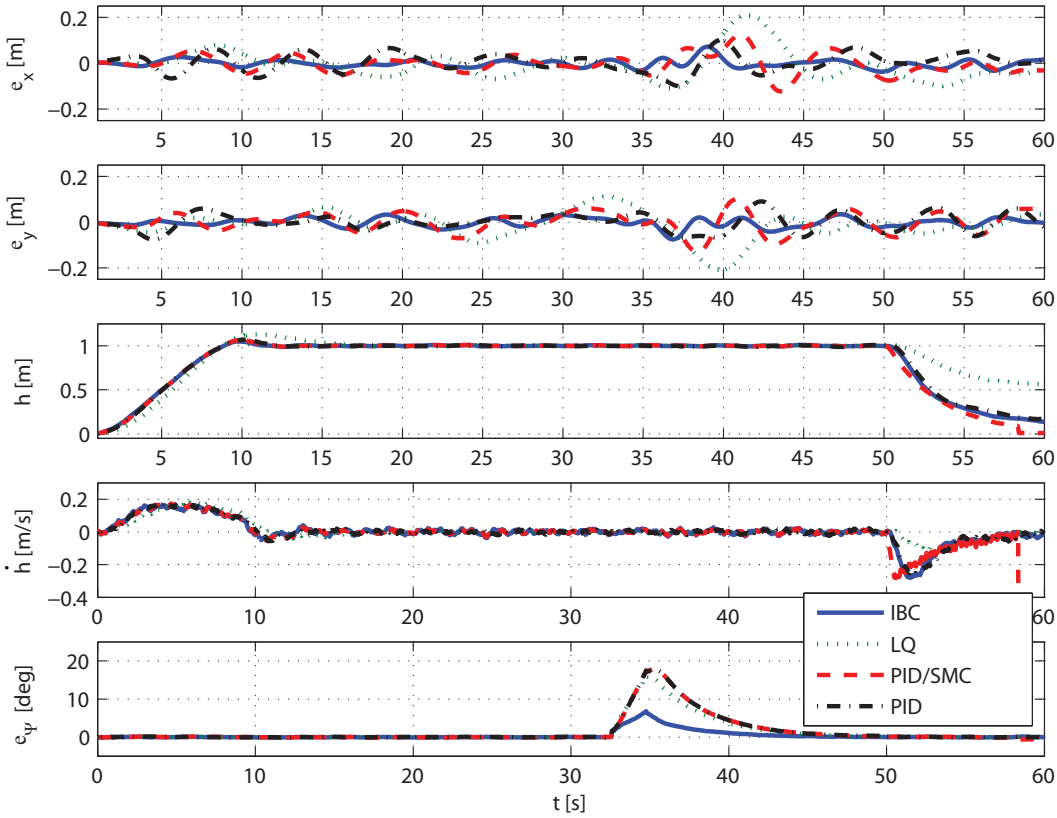

Fig. 4.6: Results comparison using various control designs for take-off, tracking, and landing with simulated horizontal and vertical drag during landing.

Fig. 4.7 illustrates the control results of the take-off phase for path planning using a cubic spline (solid lines) and a linear interpolation (dashed lines). Here a PID controller is applied. Using the cubic spline planning, the vertical velocity is zero after take-off, resulting in a quick convergence of the quadrotor altitude to the desired altitude.

The landing phase is enlarged and shown in Fig. 4.8. In simulations, the landing phase is conducted using marker-based pose estimation. The thrust of the quadrotor is set to be zero, if the altitude of the quadrotor is lower than 0.1 m. Using the exponential path with $\sigma = 0.3$, it takes about 8 s to land from 1 m to 0.1 m. The SMC succeeds in controlling the quadrotor to land, shown in the dashed lines. Due to the unmodeled forces such as drag or reflected air flow from the upper surface of the ground robot, the quadrotor fails to land using IBC and PID. LQ can definitely not be applied in the landing phase.

4.3.2 Performance against Disturbance

Moreover, control performance against disturbance is also considered. The trajectory and motion of the ground robot is the same as the previous one. In this simulation, a north wind with an amplitude of 0.1 m/s, a zero bias, a frequency of 0.5 rad/s, and a zero phase, as well as an east wind with an amplitude of 0.1 m/s, a zero bias, a frequency of 0.6 rad/s,

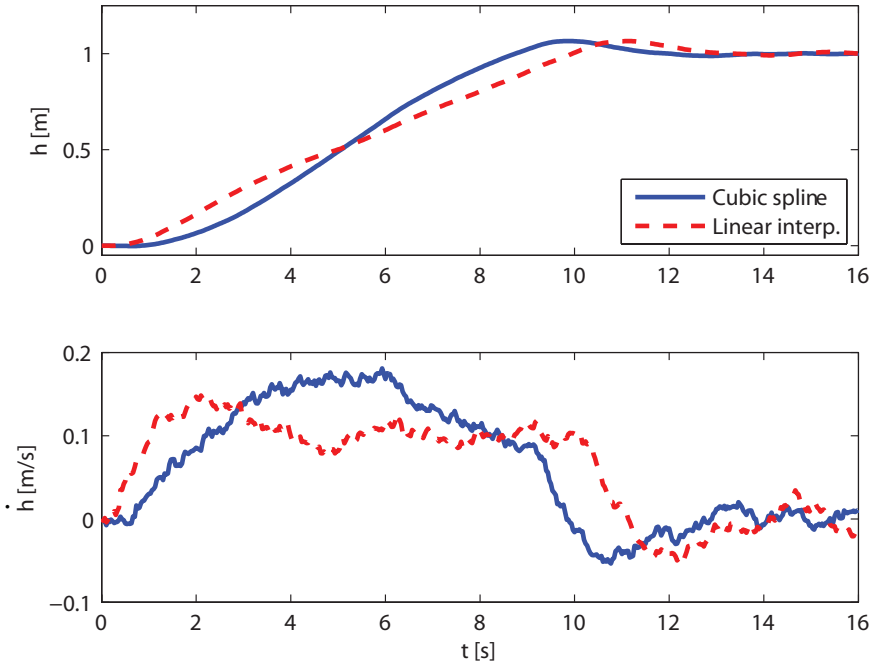

Fig. 4.7: Results comparison using cubic spline or linear interpolation for the path planning in take-off. A PID controller is applied.

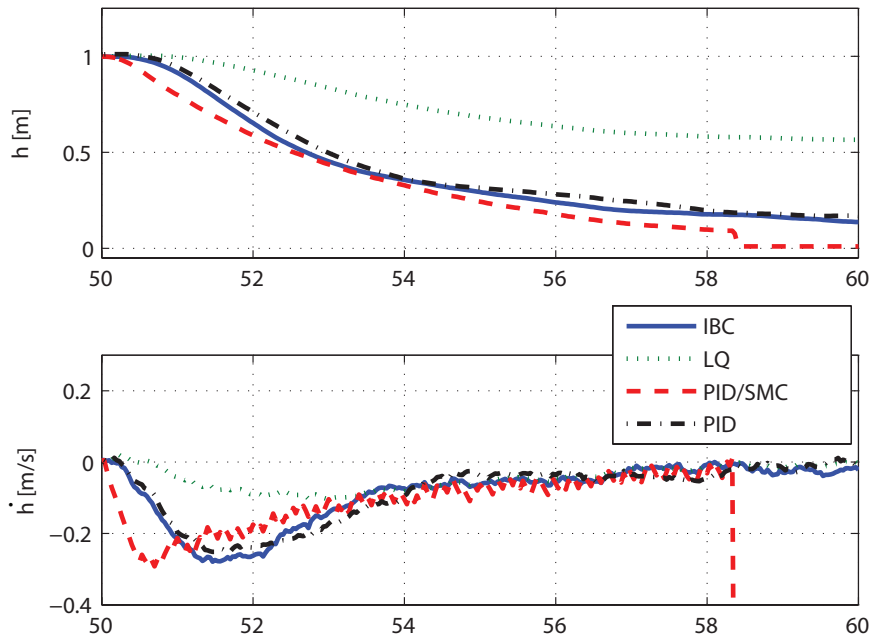

Fig. 4.8: Results comparison using various control designs for landing with simulated horizontal and vertical drag during landing.

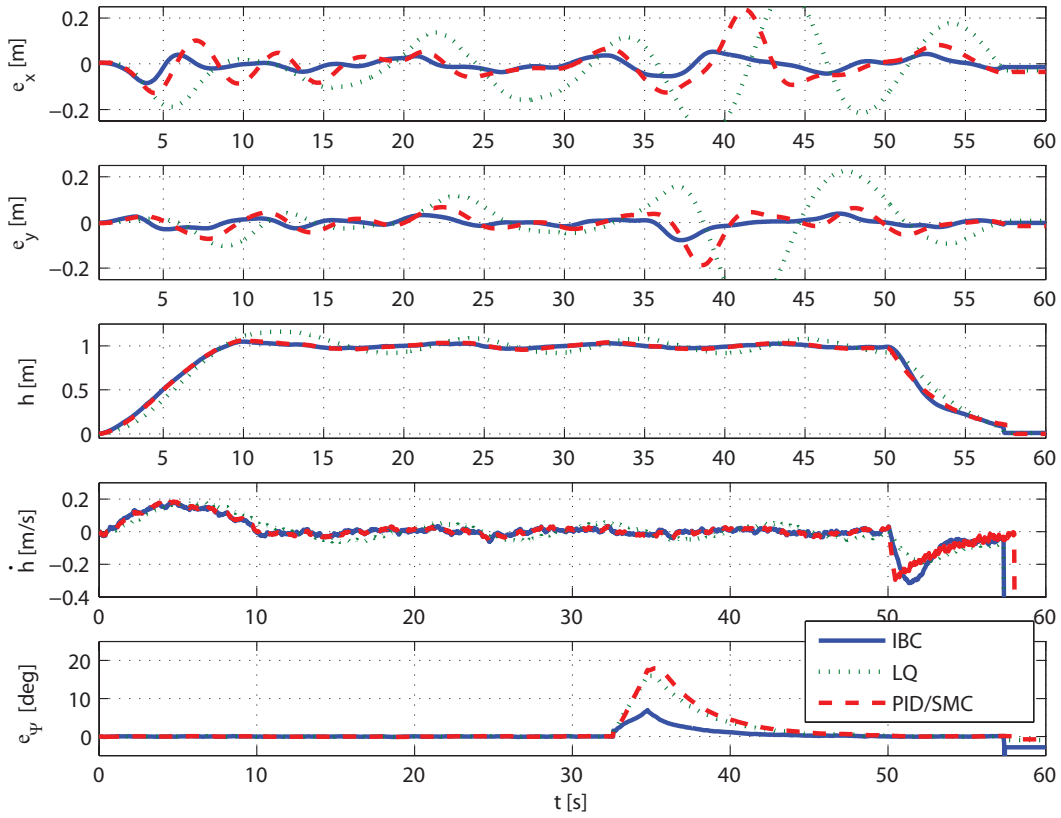

Fig. 4.9: Results comparison using various control designs for take-off, tracking, and landing with simulated north and east winds after take-off.

and a phase of 1.4 rad are concurrently applied after take-off. Here no unmodeled force is applied in the landing part.

Fig. 4.9 illustrates the control errors e_x and e_y in x-/y-directions, the quadrotor altitude h and vertical velocity $\dot{h}$, as well as the control error e_Ψ of the yaw angle, using IBC (solid lines), LQ (dotted lines), and PID with SMC for landing (dashed line). In spite of the larger control errors due to the disturbances than those in Fig. 4.6, IBC exhibits a more stable and robust control behavior than LQ and PID.

4.3.3 Influence of Feedforward Extension

Aiming at cooperation tasks with a stable and robust flying not using GPS, a cooperating partner – the mobile ground robot – is supposed to facilitate the quadrotor flying and tracking. The influence of the feedforward extension of the control structure is investigated here.

Fig. 4.10 shows the quadrotor tracking errors e_x, e_y, and e_Ψ without motion feedforward from the ground robot (solid lines) and with feedforward (dashed lines) using PID. As the desired altitude does not change during tracking, e_z is not shown here. The ground robot moves along a polygon trajectory with three rotations. With the help of motion feedforward

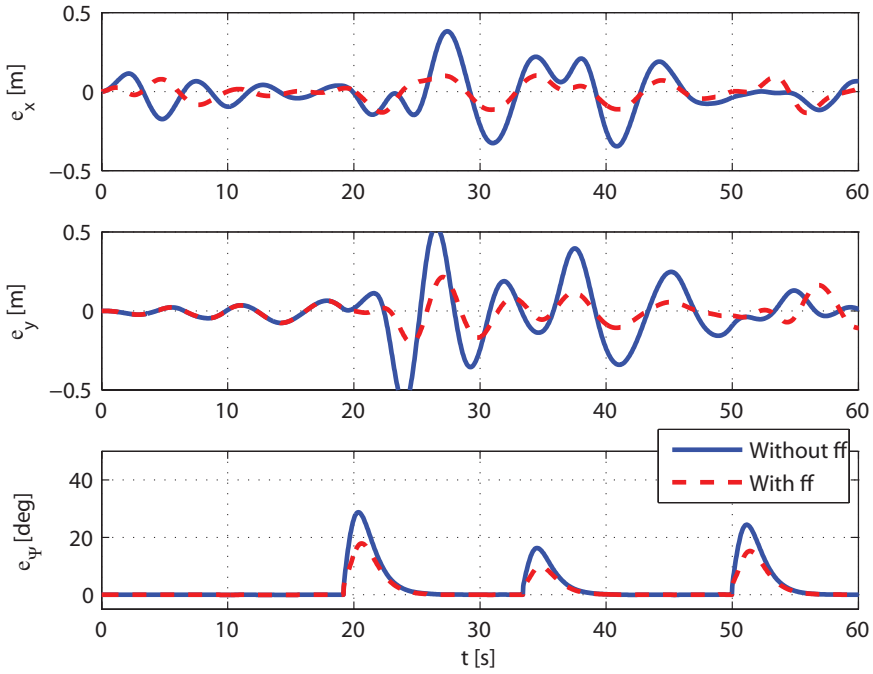

Fig. 4.10: Influence of feedforward (ff) extension using a PID controller. The mobile robot moves along a polygon trajectory.

of the ground robot, the control errors are significantly reduced and a larger maximum velocity of the mobile robot can be set. For example, along this polygon trajectory, the velocity of the ground robot cannot exceed 0.11 m/s without motion feedforward, while the maximum velocity of the robot is 1.55 m/s if motion feedforward is applied. Similar results are also obtained using IBC and LQ.

4.3.4 Integrated Control Design

Based on the simulation results, an integrated control design for take-off, hovering, tracking, and landing phases is proposed, shown in Tab. 4.1, considering different focuses and requirements in different flight phases.

At the beginning, the quadrotor is located on the top of the mobile ground robot. The quadrotor increases its altitude using SMC to deal with the unmodeled forces while keeping its x-/y-position using IBC. After the altitude reaches h_t, optical-flow-based motion estimation is replaced by marker-based pose estimation for better accuracy. Cubic spline path planning is applied to achieve a zero start/end vertical velocity.

In the hovering and tracking phases, IBC is used for small tracking errors, better stability, and robustness against disturbance.

Similar to take-off, the landing phase also consists of a marker-based pose estimation and an optical-flow-based motion estimation. The path is planned as an exponential trajectory to maximize the start velocity and obtain a zero end velocity, such that the quadrotor can smoothly land on the moving ground robot. A cooperation of IBC and SMC (vertical

Flight Phase	Height	Vision Processing	Path Planning	Controller
Take-off	$< h_t$	optical flow	cubic spline	IBC/SMC
Take-off	$> h_t$	marker	cubic spline	IBC/SMC
Hovering	$\approx h_0$	marker	–	IBC
Tracking	$\approx h_0$	marker	–	IBC
Landing	$> h_l$	marker	exponential	IBC/SMC
Landing	$< h_l$	optical flow	exponential	IBC/SMC

Tab. 4.1: Integrated control design for the autonomous flight. h_t: switching height in take-off; h_0: desired height during hovering and tracking; h_l: switching height in landing.

direction) is applied.

The LQ controller is not used in the proposed scenario, since it only provides a subordinate performance. Besides the negative influences of disturbances and unmodeled forces, the main reason is that the LQ controller is set up without considering quadrotor dynamics.

Moreover, feedforward extension is realized through the communication between the quadrotor and the ground robot, to improve the control performance.

4.4 Real-Time Experimental Evaluation

The integrated controller including PID, IBC, and SMC is implemented on the hardware to demonstrate the control performance. First, the system time delay is compensated for. Then, experiments comparing PID and IBC in hovering and tracking are conducted. Finally, an autonomous flight consisting of take-off, hovering, tracking, and landing is enabled in real-time experiments.

4.4.1 Compensation of System Time Delay

Time delay is considered an important issue in Chapter 3 for pose/motion estimation due to the data synchronization problem. In the control loop, it can lead to unexpected oscillation, especially for highly dynamic systems. To stabilize the flight of the quadrotor at the desired position, the time delay of the control input – the actual value – should be determined first. As mentioned in Section 3.3.5, the vision data is synchronized with the IMU data twice. So the system time delay is the running time of the signal from IMUs to the actuators on the quadrotor again. With well-designed experiments, the average transport time t_{d1} of the communication between the ARM-processor and the ATOM-board is measured (see Appendix A). The running time t_{d2} of the software on the ATOM-board is variable. This is measured by using the timer integrated in the software. The total time delay t_d can then be expressed as

$$t_d = t_{d1} + t_{d2}. \tag{4.125}$$

According to real-time experiments, the system time delay t_d is known as greater than 80 ms and less than 110 ms in most cases. A control output delayed by more than 80 ms is no more suitable for a stable flight while tracking a quickly moving target. To compensate for the undesired time delay, a state predictor is deployed here. It is reasonable to assume

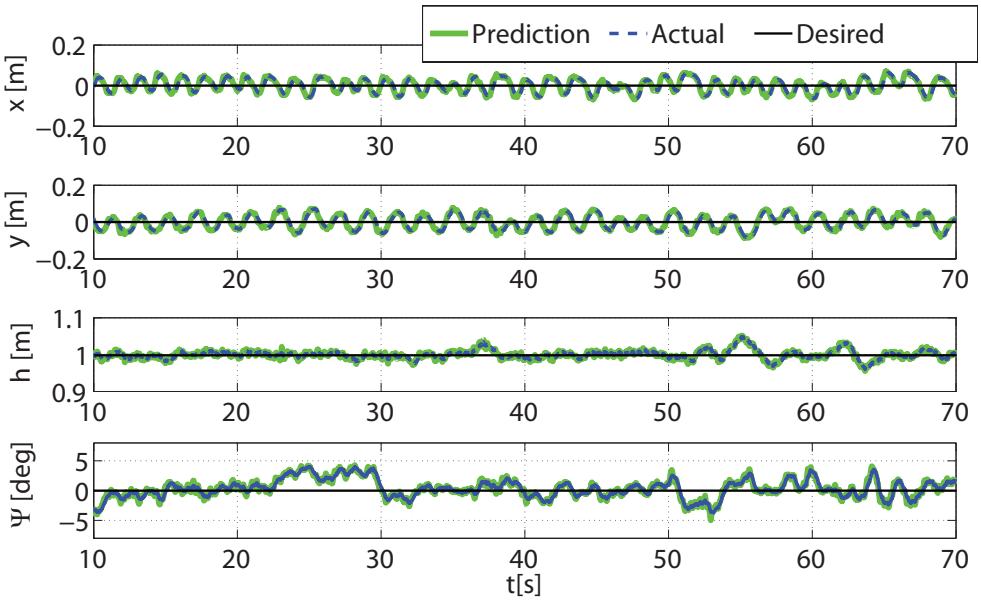

Fig. 4.11: Pose prediction result based on the last 40 estimations.

that the 3D position and the yaw angle of the quadrotor is a second-order function of the time as follows:

$$\varsigma = \kappa_0 + \kappa_1 \cdot t + \kappa_2 \cdot t^2, \tag{4.126}$$

where ς is the ${}_I x$, ${}_I y$, ${}_I z$, and ${}_I \Psi$ in the inertial frame. The last κ estimated states are used to approximate the function of the time. Parameters κ_0–κ_2 are expected to be constant in the current time window corresponding to the concerned κ estimations. The linear equations are relatively well-conditioned and can be directly solved by using the pseudo-inverse. This structure of the predictor also has another advantage – filtering of the high frequency shaking of the estimation. By carefully selected κ, the states can be predicted reliably. In the prediction block illustrated in Fig. 4.2, the predicted states are forwarded into the outer-loop controller.

In Fig. 4.11, the pose is properly predicted concerning the last 40 estimations for solving κ_0–κ_2. There are some slight oscillations observed in the z and Ψ predictions, which have however only a negligible negative effect.

4.4.2 Tracking Performance Using PID and IBC

In this subsection, control performance is constrainedly studied by using the position measurement provided by the tracking system (see Appendix C), while the on-board pose/motion estimation is not applied here. Temporarily ignoring the take-off and landing phases as well as unmodeled forces in those two phases, PID and IBC are compared in hovering and tracking scenarios.

IBC vs. PID in Hovering

The control performances using IBC and PID are compared for quadrotor hovering in Fig. 4.12. The desired position of the quadrotor is $[0 \;\; 0 \;\; 0.5]^T$ m. Both PID and IBC provide

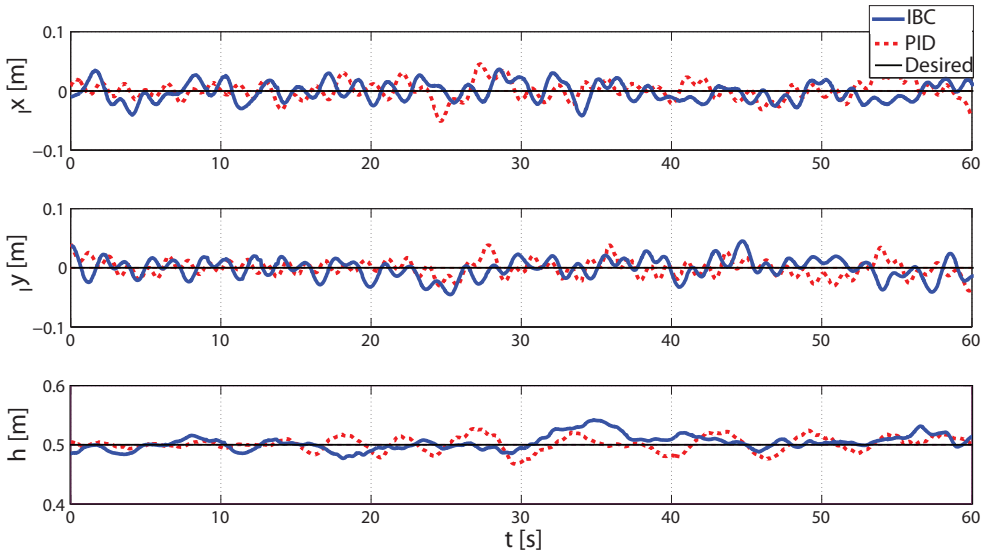

Fig. 4.12: Control performance using PID and IBC in quadrotor hovering scenario.

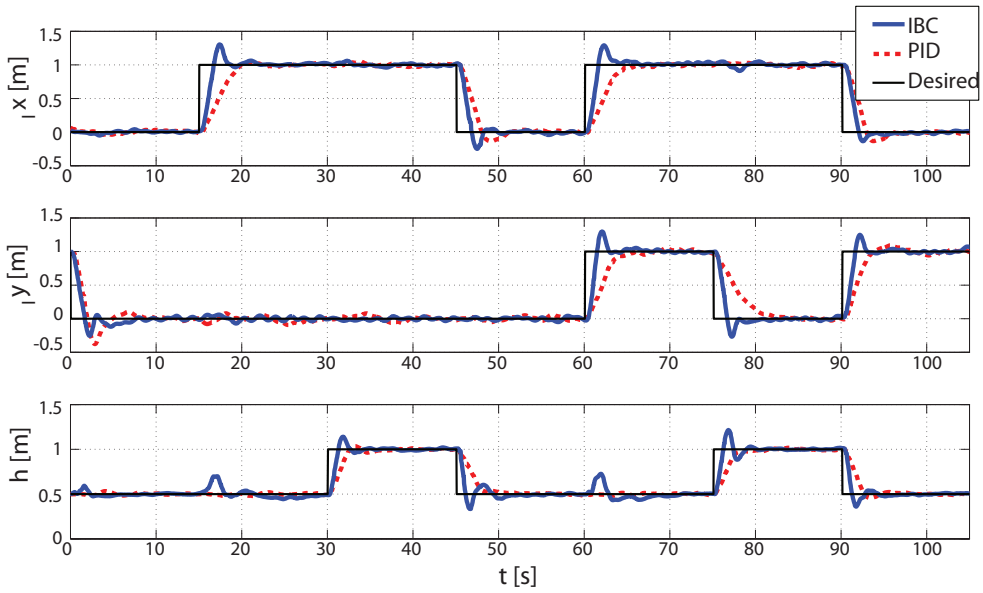

Fig. 4.13: Control performance using PID and IBC in quadrotor tracking scenario.

control errors smaller than 0.05 m in each direction. There is no significant difference between them.

IBC vs. PID in Tracking

The control performances using IBC and PID are compared for quadrotor tracking in Fig. 4.13. The desired position varies in X_I-, Y_I-, and Z_I-directions. Using IBC, the quadrotor responds quickly to the control errors, while PID needs a longer transient time but provides a small overshoot. Some other parameters for PID were also tested, which provided a quick response like IBC. However, they led to instable control performance.

Considering the robustness to disturbance and overall evaluations, IBC is chosen to be

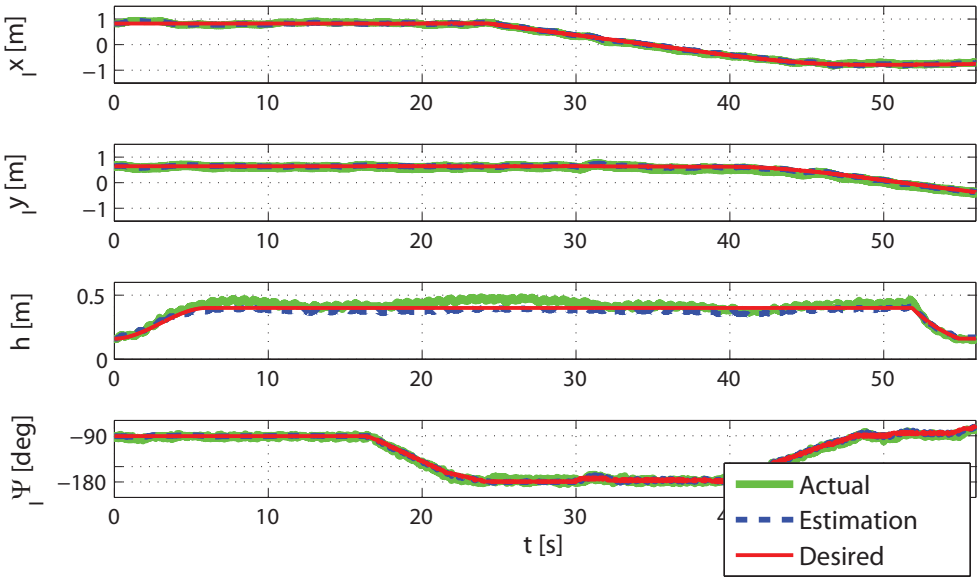

Fig. 4.14: Control performance based on optical flow.

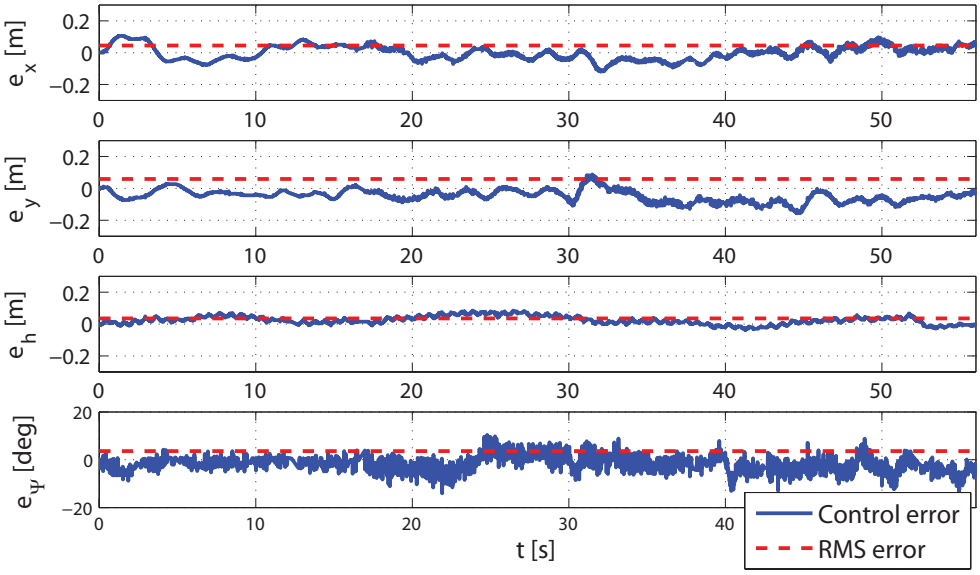

Fig. 4.15: Control error based on optical flow.

the controller for the tracking behavior in which the desired position of the quadrotor in the inertial frame consistently changes, in order to achieve a quick response.

4.4.3 Autonomous Flight Using Optical Flow Feedback

As an alternative to marker-based pose estimation, optical-flow-based motion estimation can also be applied in the whole scenario. Fig. 4.14 and 4.15 show the quadrotor position, altitude, and yaw angle evolution with the respective control errors of an experiment, in which the quadrotor took off, tracked a moving robot, and landed on the moving robot exclusively using optical-flow-based motion estimation information. The desired height

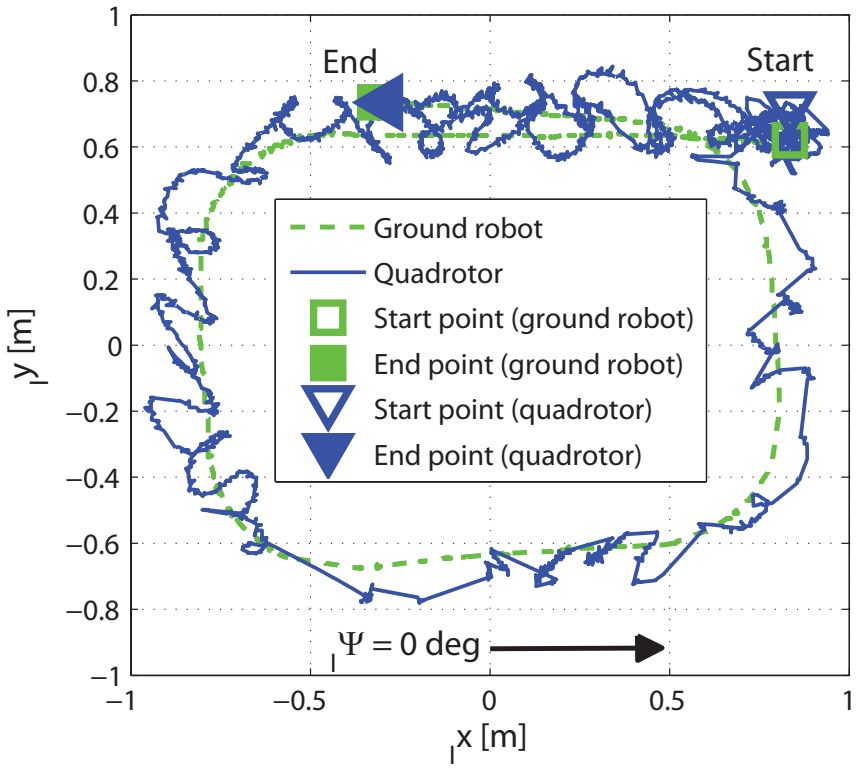

Fig. 4.16: Trajectories of the ground robot (dashed line) and the quadrotor (solid line) in the x-/y-plane of the inertial frame. Squares: start point (hollow) and end point (filled) of the ground robot. Circles: start point (hollow) and end point (filled) of the quadrotor.

during tracking is 0.5 m.

The positions in X_I-/Y_I-directions, the altitude, and the yaw angle were well controlled. The RMS control errors are 0.045 m, 0.06 m, 0.035 m, and 3.6 deg, respectively. An integrated drift would be more obvious if the desired height during tracking is higher, as there is no correction using the absolute pose during a relatively long time interval. Therefore, optical-flow-based motion estimation is only applied briefly in the take-off and landing phases, because the markers are not completely visible at this moment.

4.4.4 Autonomous Flight Using Integrated Control Design

Finally, an overall test of the autonomous flight using the proposed integrated control design was conducted. Fig. 4.16 illustrates the trajectories of the ground robot (the dashed line) and the quadrotor (the solid line). The start point and the end point of the quadrotor are $[0.835 \;\; 0.72]^T$ m and $[-0.29 \;\; 0.735]^T$ m in the inertial frame, shown in the hollow and filled circles. The hollow and filled squares denote the start and end point of the ground robot, namely $[0.835 \;\; 0.62]^T$ m and $[-0.32 \;\; 0.735]^T$ m. A quadrotor or robot orientation of 0 deg points out the positive X_I-direction.

The respective quadrotor position ${}_Ix$, ${}_Iy$, height h, and yaw angle ${}_I\Psi$ in the inertial frame are illustrated in Fig. 4.17. The representative time points are also shown along

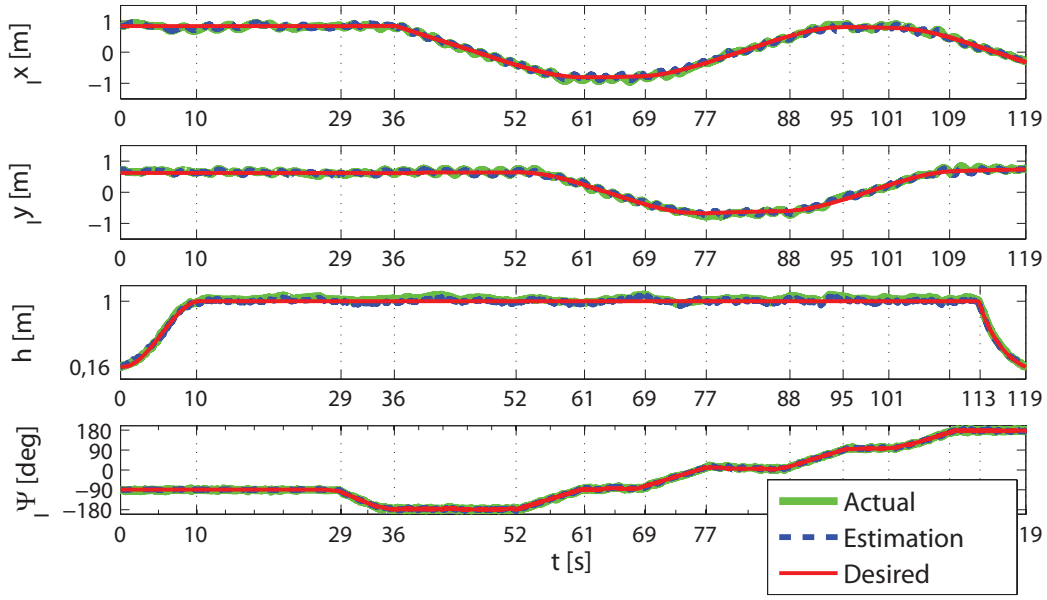

Fig. 4.17: Control performance consisting of take-off, hovering, tracking, and landing.

the time scale. Since the quadrotor's behavior is not triggered by time but by the state estimation, the time points are approximate. At the start point, the quadrotor was placed on the top of the ground robot with a height of $h = 0.16\,\text{m}$ due to the landing skids mounted on the quadrotor with a height of 0.16 m, and an absolute orientation of ${}_I\Psi = -90\,\text{deg}$. The ground robot had the same start orientation as the quadrotor. The height parameters are $h_0 = 1\,\text{m}$, $h_t = 0.66\,\text{m}$, and $h_l = 0.56\,\text{m}$. The flying process is described in detail below.

- From 0 s to approximately 10 s, the quadrotor was in the take-off phase. In the first 6 s, optical-flow-based motion estimation was applied to provide the motion information. After the quadrotor had reached a height of 0.66 m, marker-based pose estimation was applied between 6 s and 10 s. At 10 s, the quadrotor reached its desired height, namely 1 m over the ground robot. IBC was used to control the quadrotor position and yaw angle, while SMC was used to control the altitude.

- From 10 s to 29 s, the quadrotor was hovering over the ground robot at the desired altitude using IBC.

- From 29 s to 36 s, the ground robot was rotating. The orientation was changed from -90 deg to -180 deg. The quadrotor continued hovering over the ground robot and rotated with the ground robot. Here, a pure orientation was exhibited.

- From 36 s to 52 s, the ground robot was moving to the negative X_I-direction with a linear velocity of 0.109 m/s. The quadrotor started to track the ground robot using IBC. At 52 s, 69 s, 88 s, and 101 s, four rotations of 90 deg each were performed. At 109 s, the quadrotor yaw angle reached 180 deg.

- From 113 s to 119 s, the quadrotor performed landing behavior using IBC/SMC to land on the ground robot, which was moving along the negative X_I-direction.

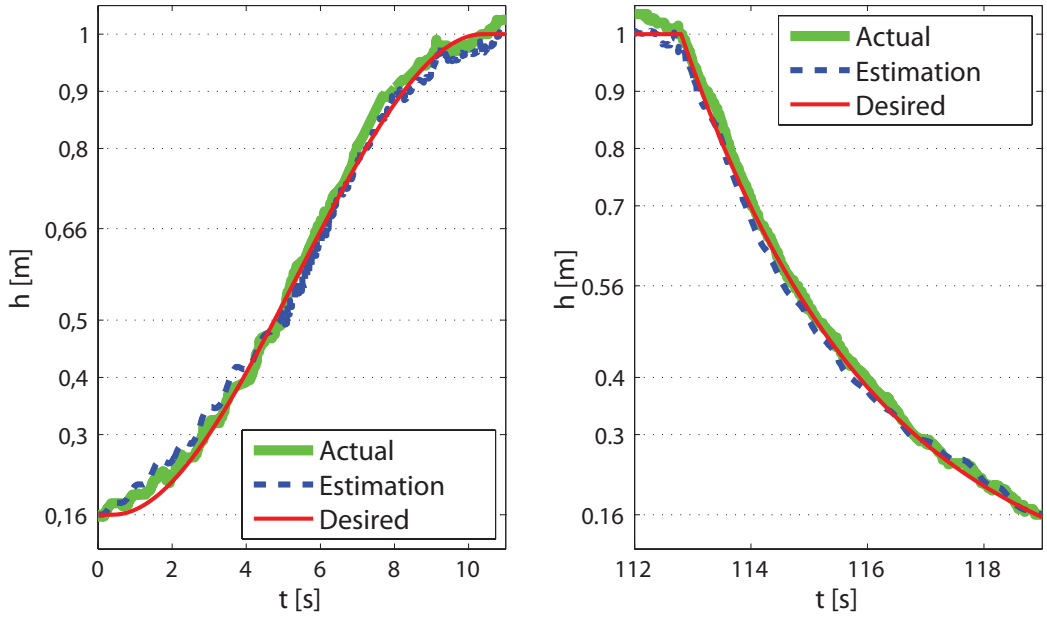

Fig. 4.18: Left: quadrotor height variation in the take-off phase. Right: quadrotor height variation in the landing phase.

Item	$e_{\mathrm{RMS},x}$ [m]	$e_{\mathrm{RMS},y}$ [m]	$e_{\mathrm{RMS},h}$ [m]	$e_{\mathrm{RMS},\Psi}$ [deg]
Pose estimation	0.04	0.036	0.029	2.575
Control result	0.057	0.054	0.034	2.92

Tab. 4.2: RMS errors in x, y, h, and Ψ.

The quadrotor vertical trajectories in take-off and landing are shown in Fig. 4.18. The switching altitude from optical-flow-based motion estimation to marker-based position estimation in take-off was at $h_t = 0.66\,\mathrm{m}$ approximately, while the switching point in landing was at $h_l = 0.56\,\mathrm{m}$ approximately. The reason for a larger h_t is that the quadrotor was not directly over the markers at the beginning of take-off. In the take-off phase, the quadrotor had to reach a higher altitude to be able to detect the markers robustly. In contrast, in the landing phase, the quadrotor had the markers in the field of view at the beginning of the landing phase and could land using marker-based pose estimation for as long as possible.

The pose estimation results and the control errors are illustrated in Fig. 4.19 and 4.20. The solid lines show the estimation errors and control errors in position, altitude, and yaw angle, respectively, while the RMS errors are shown in dashed lines. The RMS errors are also compared in Tab. 4.2. Both estimation and control performances are very successful. The control RMS errors are approximately 0.05 m in X_I-/Y_I-directions, 0.034 m in altitude, and 2.92 deg in yaw angle. The straight line segments of the control errors such as e_Ψ between the 23 s and the 25 s are due to missing tracking data (the ground truth) from the tracking system, since there were fewer than three markers detected by the tracking system in those time intervals.

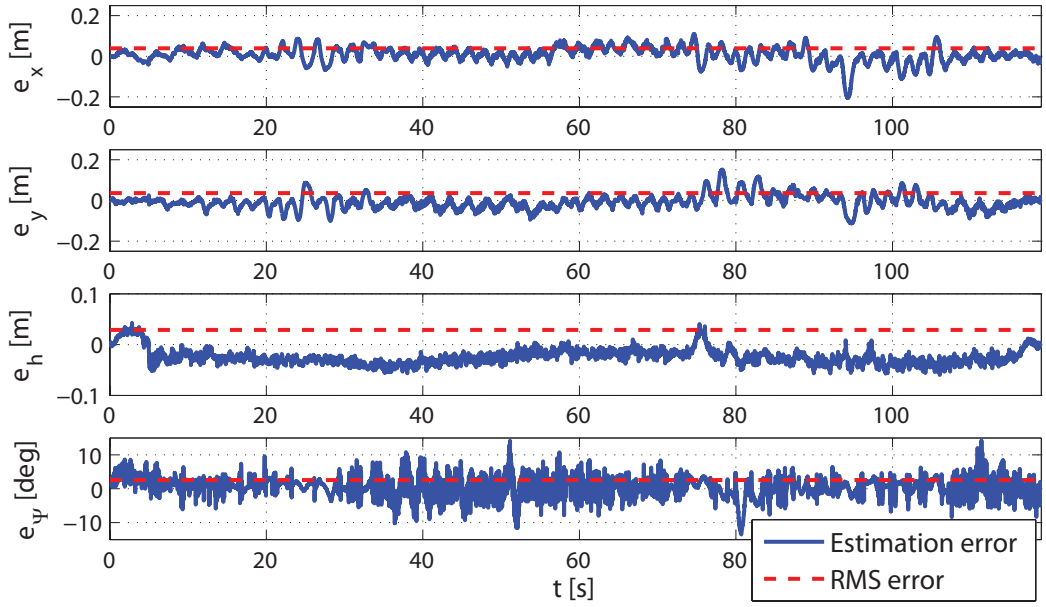

Fig. 4.19: Estimation error in the complete flight scenario.

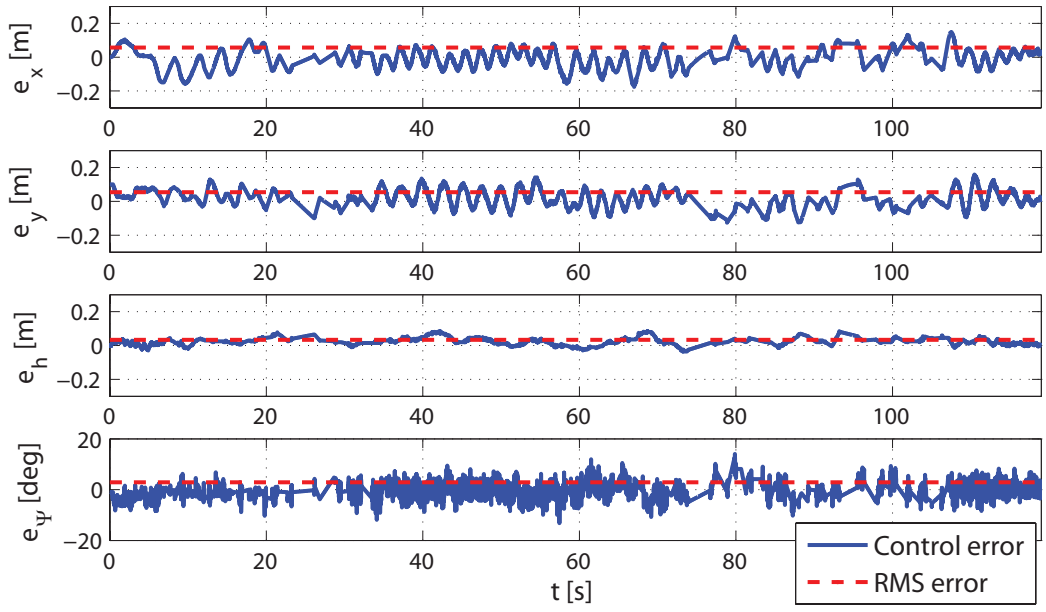

Fig. 4.20: Control error in the complete flight scenario.

4.5 Discussion

Complex System Modeling

Compared to the most comparable work in [35], system simplifications such as small angle approximations are not applied in this work. The complexity and non-linearity of the quadrotor system are preserved. Therefore, the control design here is more advantageous for dynamic scenarios.

In addition, while the IMU measurements a_x and a_y are ignored in most existing works [28, 35], they are considered through quadrotor pose estimation in this work. In the control design, the unmodeled force is also targetedly considered, which is commonly neglected in other works.

Improved Stability Analysis

Moreover, this work provides a more complete stability analysis for IBC considering the integral term for the quadrotor velocity control, which has not yet been considered in the literature.

Furthermore, the design, stability analysis, and implementation of SMC particularly considering path planning using cubic spline interpolation for take-off and exponential trajectory for landing are proposed for the first time in this work.

Integrated Flight Scenario

While the state-of-the-art works have only partially studied various aspects of quadrotor behavior and most of them are only evaluated in simulations, the first integrated approach for quadrotor flying from take-off, hovering, tracking, to landing using minimal on-board sensors is proposed and evaluated in real-time experiments using elaborated control design. In the whole process, switching mechanisms between marker-based pose estimation and optical-flow-based motion estimation as well as between IBC and SMC are proposed and accomplished. In addition, the whole data processing and control algorithms are conducted totally on-board for the complete scenario.

Improved Control Performance

The control performance presented here is much better than those in the comparable works in terms of smaller RMS errors, although the scenario considered in this work is more dynamic, complex, and challenging. The results of two related works are shown below. In [35], a VTOL flight consisting of take-off, hovering (altitude control), and landing was conducted using an integral backstepping controller similar to the IBC used here, without consideration of a second integral term for velocity control. The maximum altitude control error during hovering was 0.03 m at a height of 0.5 m, while the total RMS error of altitude control during take-off, hovering, tracking, and landing in this work is only 0.03 cm at a hovering height of 1 m. Another work [62] considered visual servoing problems and applied backstepping technique to control a *Centre d'Energie Atomique* (CEA) quadrotor observing four markers on the ground. The desired position was $[0 \;\; 0 \;\; 1.4]^T$ cm. The authors stated that the closed-loop performance of the system maintained an error of approximately 10 cm around the desired position. Therefore, improved control performance is obtained through the integrated and non-linear control design.

4.6 Summary

Based on accurate pose estimation of the quadrotor, a stable and effective control design is desired in order to realize autonomous flying. Mainly due to non-linear dynamics and time delay, it is demanding to elaborate a controller for this under-actuated system. Sophisticated control designs have normally been evaluated in simulations or limitedly evaluated in a non-autonomous manner.

This chapter aims at designing, implementing, and discussing adequate control structures for the quadrotor to overcome the aforementioned challenges and to improve the quadrotor flying behavior. Standard controllers such as PID controllers, optimal controllers such as LQ controllers, and non-linear controllers such as backstepping-based controllers and sliding model controllers are carefully adapted to this quadrotor system and discussed. Compared to most works which have implemented controllers on a real quadrotor, the system model used here is less simplified, in order to preserve the original dependency of system states. Moreover, the overall time delay is derived and compensated for, while feedforward information is also taken into account.

An elaborated, integrated control approach combining the advantages of these controllers is proposed and evaluated in a complete flight experiment consisting of take-off, hovering, tracking, and landing. The control performance is proved to be effective in terms of small RMS control errors of 0.06 m in position control, 0.03 m in altitude control, and 3 deg in yaw control, which are much smaller than those in comparable works in the literature.

The effective control performance completes the development of the vision-guided autonomous quadrotor. However, a quadrotor with a limited payload and fast, dynamic self-motion requires a simple, fast sensorimotor control, which can conduct motor activities using limited sensory perceptions. In contrast to the traditional manner of developing a flying system, bio-inspired technology is considered in the next chapter for further improvement.

5 Insect-Inspired Motion Detection and Estimation for Flight Control

In recent years, the robotics community has shown a great interest in applying biologically inspired technology to technical systems. One of the essential reasons is that biological models such as humans and animals exhibit high efficiency in visuomotor control. The neurobiological findings show that a number of flying insects, e.g. Drosophila melanogaster and Calliphora vicina, possess photoreceptors with high temporal resolution which they use for dynamic visuomotor pose and gaze stabilization, and navigation in 6 *Degrees Of Freedom* (DOFs). This could provide an alternative extended solution for the development of *Micro Air Vehicle* (MAV) on-board vision, which requires simple and fast computation due to limited payload, restricted computational capacity, and fast, dynamic self-motion. While Chapters 3 and 4 propose traditional quadrotor control based on accurate pose/-motion estimation and effective control design, Chapter 5 focuses on insect-inspired flight control and implementation (see Fig. 5.1).

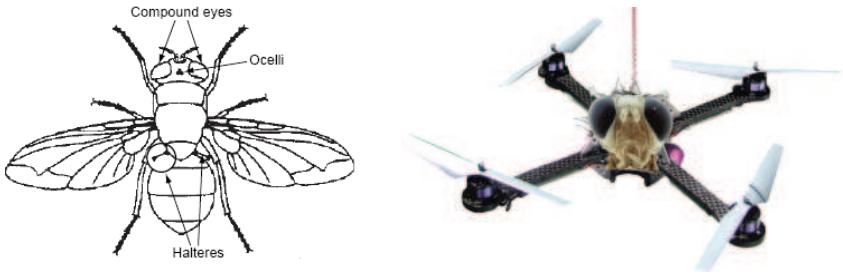

Fig. 5.1: From biological model to MAV development. Left: a fly with the most important perceptive organs related to flight control: the compound eyes, the ocelli, and the halteres, reprinted from [130]; right: illustration of an envisioned bio-inspired quadrotor with flies' vision.

Considering the aforementioned advantages of flying insects, an array of biologically inspired *Elementary Motion Detectors* (EMDs) for local motion detection (resulting in optical flow field) and the *Receptive Field* (RF) concept for global motion detection are chosen as the starting point for the development of a bio-inspired quadrotor, as an alternative to traditional paradigms. The challenge lies in the extraction and analysis of the flow field, for the ego-motion estimation of a system, for instance, the quadrotor. Moreover, technical realization and applications from the viewpoints of control engineering and computer vision based on insect vision are envisaged.

In this chapter, in order to adapt to the typical dominant and preferred motion patterns of the quadrotor, the novel vision and visuomotor behavior models inspired by biological

paradigms are extended first. Two new RFs for rotation detection are proposed. High-speed implementation using compatible hardware – a *Field Programmable Gate Array* (FPGA) platform – is accomplished to obtain the effectiveness of the neurobiological algorithms. The performance of the implementation is sufficient to deal with video frame rates of 350 fps or above for a frame size of 256×256 pixels.

As EMDs and RFs suggested in fundamental studies can only provide a solution to qualitative motion detection [147], the respective motion estimation is established with the help of *Look-Up Tables* (LUTs) and extensively explored considering the influences of specific parameters such as perception differences between flies and cameras, lighting conditions, as well as the spatial frequency and spectrum of input images. Closed-loop control and obstacle avoidance are exploratively investigated and show a promising possibility of fully controlling MAVs based on insect-like vision in future work.

The remainder of this chapter is organized as follows. First, in Section 5.1, the biological principles of flies' vision systems and the problem addressed in this chapter are briefly introduced. Qualitative motion detection consisting of the basic and elaborated EMD models, the extended RFs, and the high-performance implementation on FPGAs is described in Section 5.2. In Section 5.3, quantitative motion estimation and control are proposed. In Section 5.4, obstacle detection and collision avoidance based on the qualitative and quantitative motion information are investigated. A discussion and a summary are given in Sections 5.5 and 5.6.

5.1 Theoretical Foundations and Problem Definition

5.1.1 Biological Principles of the Fly Vision

A fly's panoramic vision system comprises at its front end several thousand photoreceptors feeding into a 2D array of motion detecting neurons which the animal uses for dynamic visuomotor pose and gaze stabilization and navigation in 6 DOFs.

As one of the most successful animals in the evolution, flies possess a very efficient flight control system, such that they can respond very quickly based on low-resolution vision. Therefore, innovative researchers in the robotics domain take flies as their biological models for MAV control problems [45, 65, 76, 130]. Since the topic considered in this chapter is an interdisciplinary one, the fly vision system is briefly introduced here.

In most cases, flies have two large compound eyes on the head with ocelli on the top and halteres, illustrated in Fig. 5.1. The ocelli are simple photoreceptors without lenses and used to measure the brightness. The halteres work as a gyroscope and provide information for course stabilization. The ocelli, the halteres, and the compound eyes play an essential role in the flight control of flies [130]. Since the flying behavior is mainly controlled by vision, the flies vision system is the focus of this section.

Fig. 5.2 illustrates a fly's visual and central nervous system. Each compound eye contains 3000–4000 tiny optical units, called ommatidia [88], while each ommatidium possesses its own lens and photoreceptor. The resolution of an ommatidium is approximately 1.5°. The spatial resolution of fly eyes is not very high, but their field of view is wide (nearly 360 deg). Moreover, ommatidia are highly temporally sensitive and respond to temporal

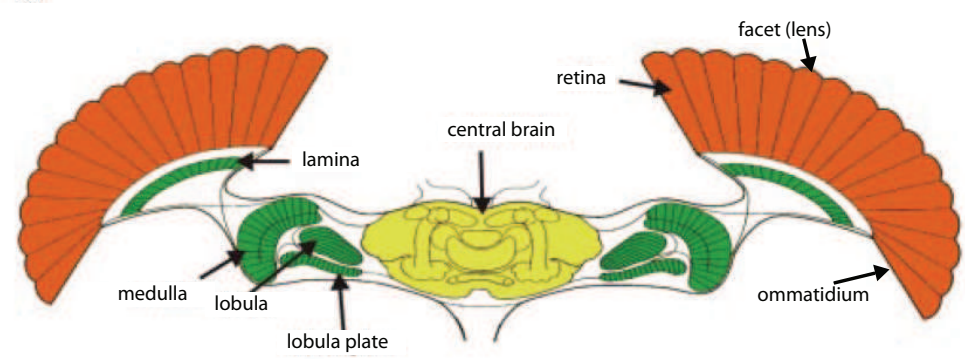

Fig. 5.2: A fly's visual and central nervous system, adapted from [33, 130].

frequencies up to 200–300 Hz [51], which is much more sensitive than with humans (less than 100 Hz).

The neurons in the brain responsible for image processing comprise three layers [33]: lamina, medulla, and lobula complex consisting of the lobula and the lobula plate. The neurons in the lamina receive the input from the photoreceptors and amplify temporal changes. The medulla then detects the local motion and sends the information to the lobula complex. The lobula complex receives input from medulla elements in parallel [33] and converges the preprocessed photoreceptor information into the lobula plate. In the lobula plate, large neurons are found which integrate these local motion signals and additionally form extensive connections amongst themselves [33, 64]. These neurons have large RFs and respond best to particular flow-fields which occur during certain maneuvers of the fly in free flight [49, 84].

5.1.2 Motion Detector and Receptive Field

Based on the biological principle of flies vision, computational models for local motion and global motion detection are developed. The state-of-the-art motion detector and RFs are introduced here.

Motion Detector

Motion detection is one of the most basic tasks that a visual system has to perform. Motion information is not explicitly represented at the output level of retinal photoreceptors. Instead, it has to be computed from the changing retinal images by the nervous system [34]. Using an elaborated motion detector model and some RFs, some properties of the biological visual system have been converted into computational models to estimate ego-motion for engineering applications.

– The Reichardt Model The earliest and probably the most famous model of motion detection inspired by biological systems was developed by Reichardt and Hassenstein in 1956 [67] and called EMD. The Reichardt detector describes, at an algorithmic level, the process of local motion detection in the fly, leading from non-directional input to a direction

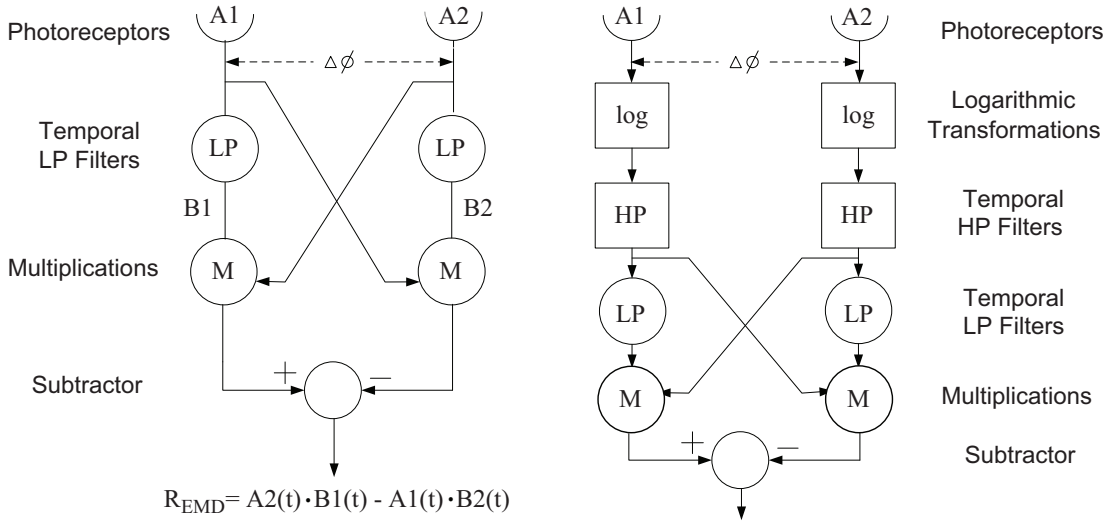

Fig. 5.3: Left: the simple Reichardt detector [33]; right: the elaborated EMD [58]. Two extensions are made: logarithmic transformations and the temporal HP filters.

selective output [32, 54, 67, 71, 99, 108]. Fig. 5.3 (left) presents a simplified version of this correlator model.

$A1$ and $A2$ are two photoreceptors. The input signals of $A1$ and $A2$ are temporally delayed by *Low-Pass* (LP) filters, resulting in $B1$ and $B2$. With $A1(t)$ and $A2(t)$ representing the input signals at the left and right inputs, and $B1(t)$ and $B2(t)$ representing the corresponding filtered signals, one obtains the output $R(t)$ of a motion detector:

$$R_{\mathrm{EMD}}(t) = A2(t) \cdot B1(t) - A1(t) \cdot B2(t). \tag{5.1}$$

The detector generates a direction-sensitive response because of the subtraction between the two symmetric detector halves. Some examples of signal inputs and their EMD responses are shown in Fig. 5.4. In a) a positive peak-like signal moves to the right. Its respective output is shown in d), namely a positive response. If this peak-like signal moves to the left side, as shown in b), its output is a negative response, as shown in e). The response is zero when no motion exists. However, the response of the Reichardt detector is not always as simple as a peak. For example, for a pulse-like input signal in c), the sign of its response in f) does not directly indicate its motion direction.

– The Elaborated EMDs The simple Reichardt detector has two major drawbacks: 1) The response is sensitive to edge contrast, reducing the robustness to lighting conditions; 2) The response to a step edge is complicated, making scene interpretation difficult. Therefore, the simple detector has been improved and two preprocessors are added in [58]:

- Logarithmic transformation is applied in order to reduce the sensitivity to lighting conditions directly after the receptors;

- A temporal *High-Pass* (HP) filter is added after the logarithmic transformation, in order to obtain a simple response to the most common edge type as step edge in

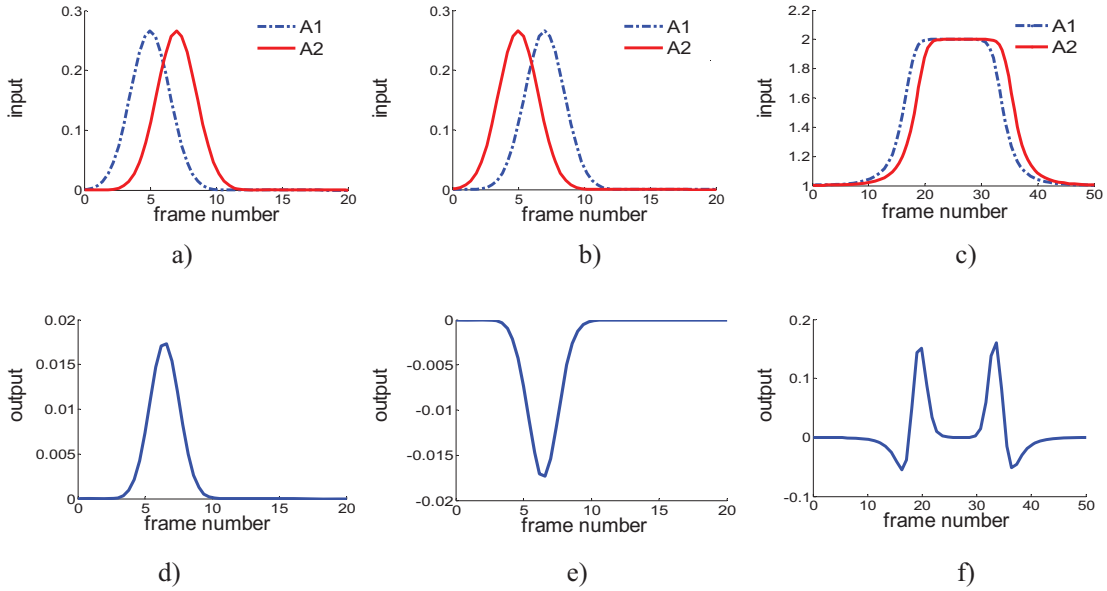

Fig. 5.4: Response of the Reichardt detector to a moving peak and to a moving pulse [58]; a) a peak moving to right; b) a peak moving to left; c) a pulse moving to right; d), e), and f) are the respective EMD responses to a), b), and c).

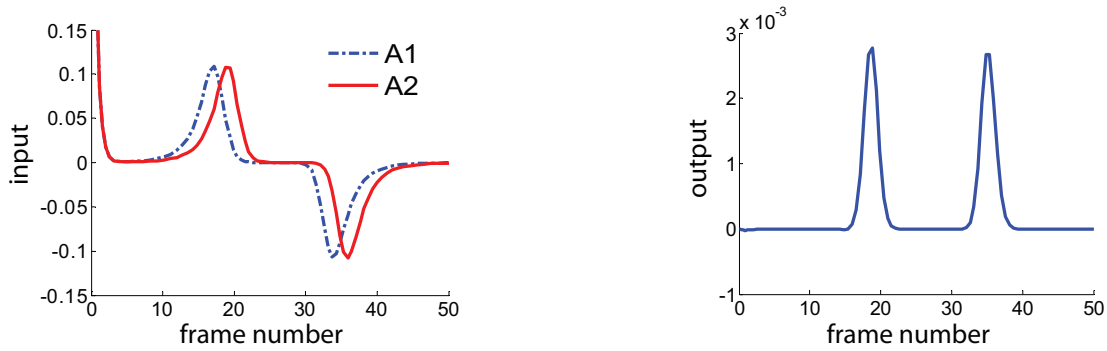

Fig. 5.5: Left: a moving peak signal transformed from a moving pulse signal illustrated in Fig. 5.4 c) by the temporal HP filter added in the elaborated EMD model. Right: the respective response of the elaborated motion detector to a pulse moving to the right [58].

natural images.

The elaborated EMD is shown in Fig. 5.3 (right). The moving pulse signal shown in Fig. 5.4 c) (or equivalently a moving step edge) is transformed by the temporal HP filters into to a moving peak as shown in Fig. 5.5 (left), leading to a simple response as shown in Fig. 5.5 (right). The positive or negative output indicates the motion direction.

– Two-Dimensional Motion Detection As an EMD can only detect one-dimensional motion along the line connecting its receptors, one simple algorithm is implemented in [97] to realize two-dimensional motion detection. A pair of EMDs are combined as shown in Fig. 5.6. The vertical motion vector component is observed by receptors P_V and P_C, while

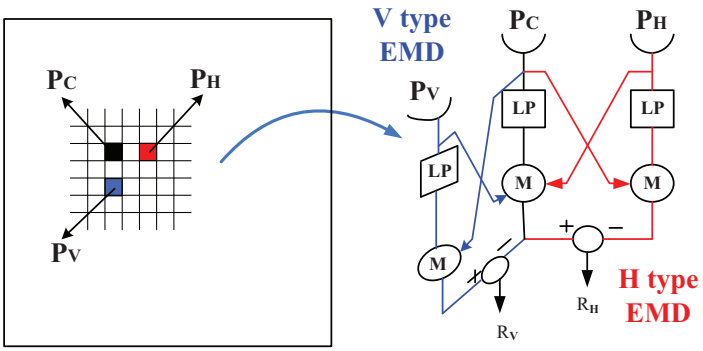

Fig. 5.6: Two-dimensionally placed receptors. Left: three photoreceptors P_C, P_H, and P_V. Right: V-type EMD and H-type EMD.

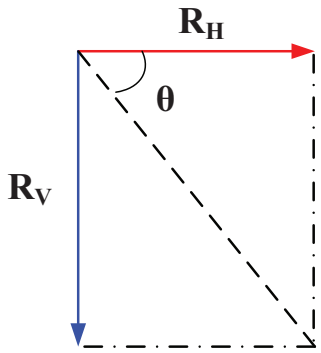

Fig. 5.7: Motion direction

the horizontal motion vector component is observed by receptors P_H and P_C. The outputs of receptor P_C are labeled as R_V and R_H in Fig. 5.6 (right). Following [98], this EMD pair is a combination of two types of EMDs:

- H-type EMD responding to local horizontal motion.
- V-type EMD responding to local vertical motion.

Motion direction can be approximately estimated by computing the ratio of R_V and R_H as illustrated in Fig. 5.7 and formulated as:

$$\theta = \arctan(\frac{R_V}{R_H}). \tag{5.2}$$

Then, an optical flow vector on the receptor P_C is obtained.

However, strictly speaking, the motion direction θ should be computed as

$$\theta = \arctan(\frac{v_V}{v_H}), \tag{5.3}$$

where the horizontal velocity component v_H and vertical velocity component v_V are not exactly proportional to the responses R_V and R_H. The quantitative relationship between the velocity and EMD response will be further explored in Section 5.3.

Receptive Fields

In neurobiology, the cells in a fly's visual system which have been found to be involved in optical flow (obtained by the proposed EMDs) processing are called motion sensitive wide-field neurons. These neurons can be classified by their sensitivities to different kinds of motions. For example, some of them are more sensitive to horizontal motion, while some others are more sensitive to vertical motion. Then, these neurons integrate the signals of EMDs spatially in their RFs, where each single EMD only analyzes the local motion along its sensitive direction. After the local motion of a photoreceptor is detected by a respective EMD pair, the global motion or the ego-motion comes into consideration, which is solved

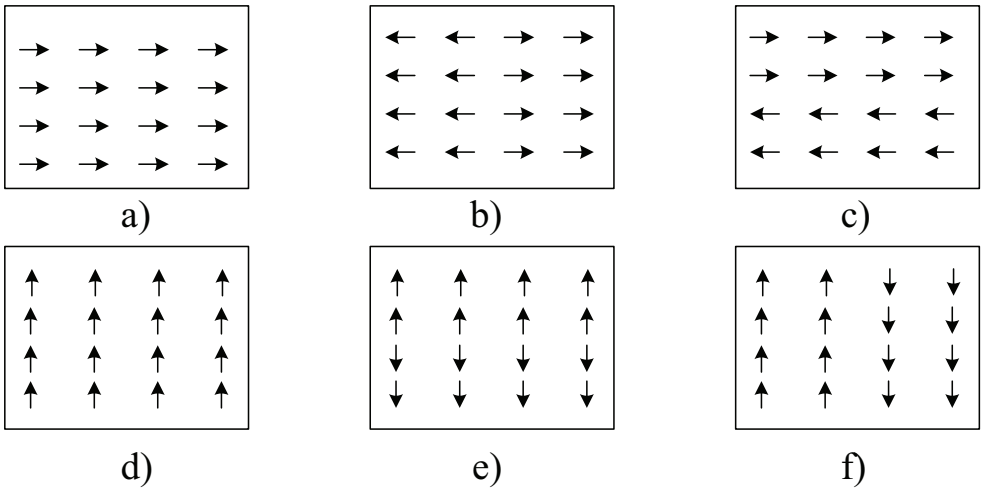

Fig. 5.8: Six RFs. a) RF for horizontal component of left-right motion; b) RF for horizontal component of forward-backward motion; c) RF for horizontal component of clockwise-anticlockwise rotation; d) RF for vertical component of up-down motion; e) RF for vertical component of forward-backward motion; f) RF for vertical component of clockwise-anticlockwise rotation.

by the RFs. The RF response R is a sum of EMD responses R_{EMD} in each RF as follows:

$$R = \sum_{i=1}^{n} R_{\mathrm{EMD},i}, \tag{5.4}$$

where n denotes the total number of EMDs in the respective RF.

In order to detect global motion, four RFs (a), b), d), and e) in Fig. 5.8) have been proposed in [58], which are sensitive to certain motion types. The RF a) is used to detect left-right motion based on vertical edges in the input images, while d) is used to detect up-down motion based on horizontal edges. Forward and backward motion can be detected using RFs b) and e), while b) is sensitive to vertical contrast and e) is sensitive to horizontal contrast. A horizontal motion, for instance, cannot be detected if only horizontal edges are obtained in the input image, which is called the aperture problem in the conventional optical flow detection.

The RF c) and f) are extensions made in this thesis and will be introduced in the next section.

5.1.3 Problem Definition

The RFs proposed in the literature only deal with translational motion. However, rotation information plays a key role in the control of flying systems. Therefore, RFs dealing with rotational motion should also be considered.

Moreover, the major advantage of a fly's visuomotor system is that flies can respond to environments robustly and quickly based on their low-resolution vision. One of the reasons is that the visual information perceived by the compound eyes is processed in a parallel structure. To ensure the high-speed characteristics of a fly-inspired approach, effective,

parallel implementation on a suitable hardware is one of the prerequisites for controlling a technical system. In the literature, hardware implementation of biological motion detection models has been realized on *Field Programmable Analog Array* (FPAA) [111], *Very-Large-Scale-Integrated* (VLSI) circuits [66, 71, 88], and FPGAs [26, 82, 97]. However, these implementations perform motion detection with low frame rates for relatively small image sizes.

Furthermore, EMDs and RFs suggested in fundamental studies can only provide a solution to qualitative motion detection. The EMD response and the actual motion do not have a unique relationship. However, the quadrotor control depends on accurate quantitative motion estimation. Up to now, the related technical realizations have not yet considered much at this point. Some works apply the insect-inspired motion detection on blimp-type robotic platform [76], which has stable dynamics during flying and is simple to be controlled during very slow movement. Others use panoramic cameras [45] or off-board cameras in simulated environments [65]. The need for a quantitative motion representation to control flying systems is not so critical as here in this thesis, as they mainly focus on the relative motion. Therefore, how the qualitative characteristics of biological modeling can be transferred into normally quantitatively controlled technical systems remains an intriguing question.

In this chapter, qualitative motion detection with a high-performance hardware implementation and quantitative motion estimation considering flight control in different environments are two focuses, which are addressed in the next sections.

5.2 Qualitative Motion Detection

Most flying robots including quadrotors have 6 DOFs. However, only four RFs for translation in three directions are proposed [58]. As mentioned in the previous chapters, the horizontal motion of a quadrotor is realized by roll and pitch angles, while the heading direction is one of the most important quadrotor states determined by the yaw angle. In order to achieve an insect-vision-based control of quadrotor flight, detection of rotational motion is a very important aspect.

In this section, two new RFs for rotation detection are proposed first, in order to cover all types of global motion based on the local motion detection. Then, the hardware implementation of insect-inspired motion detection based on the elaborated EMDs and the extended RFs, utilizing the parallel processing character of FPGA to achieve very high-speed motion detection, is described in detail. Simulations and real-time experiments are conducted for performance evaluation.

5.2.1 Extension of Receptive Fields

Based on [58], two additional RFs for rotation detection are proposed, illustrated in Fig. 5.8: RF c) is sensitive to the rotation for vertical edges and RF f) for horizontal edges.

By adding these two RFs, the system is supposed to cover all types of movements. Each RF demonstrates various sensitivities to a certain motion type. Now, the RFs can be used to indicate camera ego-motion direction.

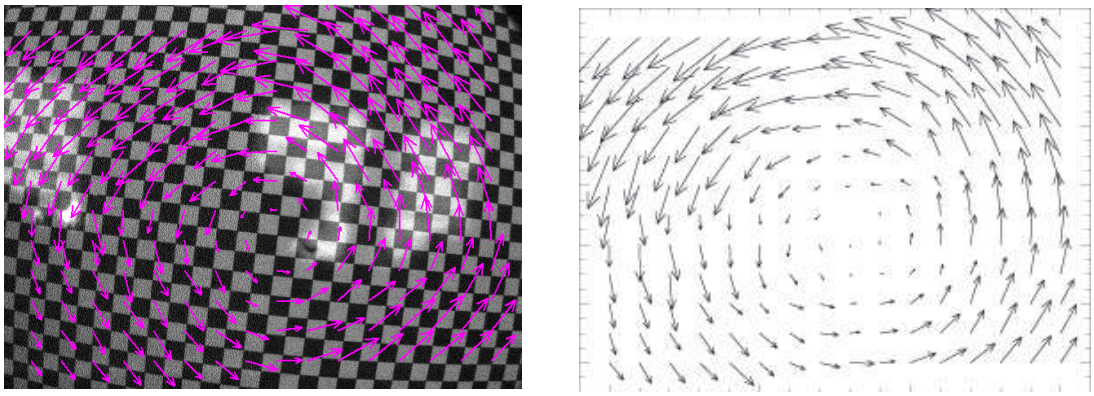

Fig. 5.9: Optical flow of rotation. Left: the original image of the background pattern with local optical flow field detected by the elaborated EMDs; right: optimized result without background.

Rotation Detection in Simulations

To validate the designed RFs for rotation detection, simulations using *Interactive Data Language* (IDL) from CREASO [4] were conducted. The inputs of this simulation are image sequences captured by a camera which was fixed on the end effector of a 6-DOF robot arm (Stäubli RX90B [13]) and moved in front of a static background composed of black and white squares (see Fig. 5.9 left). Through the operation of the robot arm, various movements of the camera can be executed, such as translation and rotation. Fig. 5.9 (right) shows all the local motion vectors detected by the elaborated EMD for a pure anticlockwise rotation. As the rotation axis was not exactly identical with the camera optical axis, the optical flow vectors in the image upper part in Fig. 5.9 (right) are not exactly the same as those in the lower part.

Fig. 5.10 shows the outputs of the six RFs. At the very beginning, the robot arm performed an acceleration process. Then, the rotation tended to be approximately constant. The outputs of the RFs c) and f) for the clockwise-anticlockwise rotation are more obvious than those from the other four RFs. The negative values indicate that the camera was moving in an anticlockwise direction. Due to the symmetry of the background, the responses of both RFs for the rotation detection are almost equal. But in the natural world, they are sensitive to horizontal and vertical edges, respectively. This simulation shows that the rotation can be successfully detected by the improved RFs.

5.2.2 Implementation on FPGA with High-Speed Performance

To preserve the advantage of insect-like vision such as high-speed motion detection, an appropriate implementation can make a significant contribution. As a test platform, FPGAs are chosen as the hardware platform due to their parallel computation property. The implementation of the aforementioned algorithm consisting of the elaborated EMDs and the RFs on an FPGA platform is described in this section.

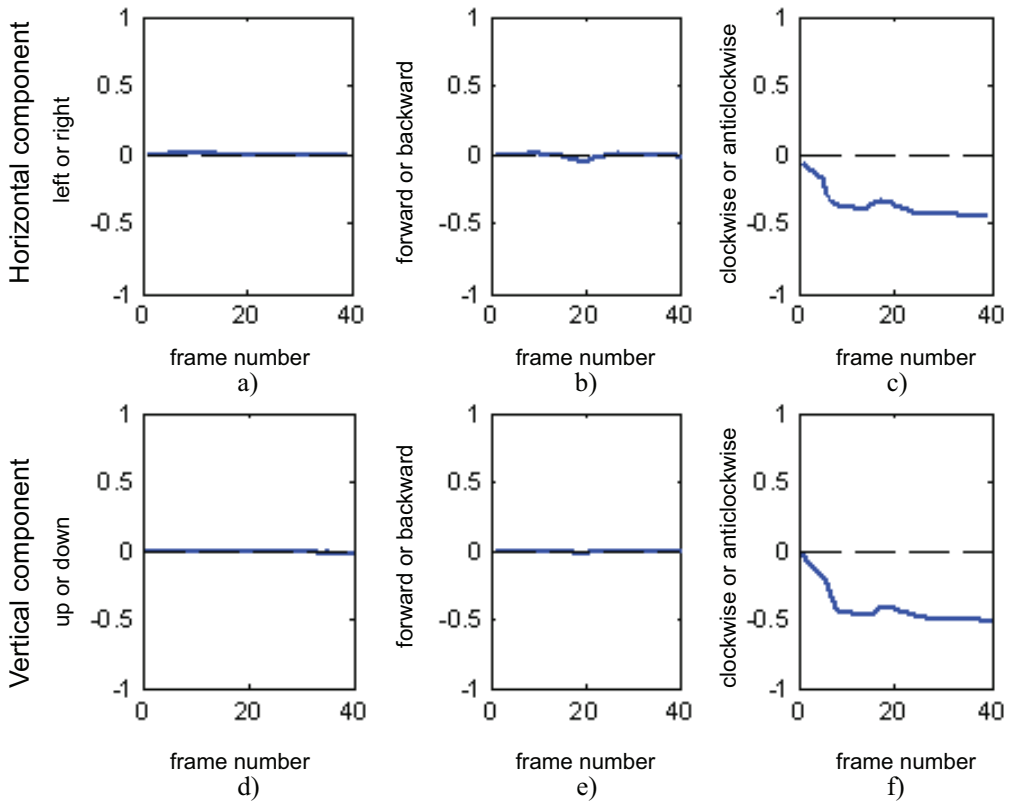

Fig. 5.10: Responses of the six RFs to the anticlockwise rotation of a camera around the optical axis. The indexes a)–f) are the RF indexes in Fig. 5.8.

Development Environment

The hardware components are shown in Fig. 5.11. Firstly, the image is continuously captured by a high-speed camera (MC1311, Microtron [9]). Then, the image is transmitted to the Tsunami FPGA platform (SBS technologies) with Altera® Stratix® EP1S40 FPGA processors [1] via the novel communication protocol camera link (CL) which is specially designed for computer vision communication. The elaborated motion detector and the algorithm of the RFs are implemented on the FPGA. After the processing of the image, the results of the EMDs and related outputs of the RFs are sent to a host PC (AMD Opteron242 with 2GB RAM).

Fig. 5.12 shows the overall physical system hierarchy illustrating the communication between different modules. The main focus of this section, namely FPGA implementation of the model and algorithms in *Very High Speed Integrated Circuit Hardware Description Language* (VHDL), is highlighted by the dashed rectangle in Fig. 5.12.

Architecture of the VHDL Program

The VHDL program architecture is shown in Fig. 5.13. The image processing is performed along with the data flow. First, the image data (8-bit iDATA) provided by the camera are fed into the *EMD module*, which performs the optical flow calculation in both horizontal

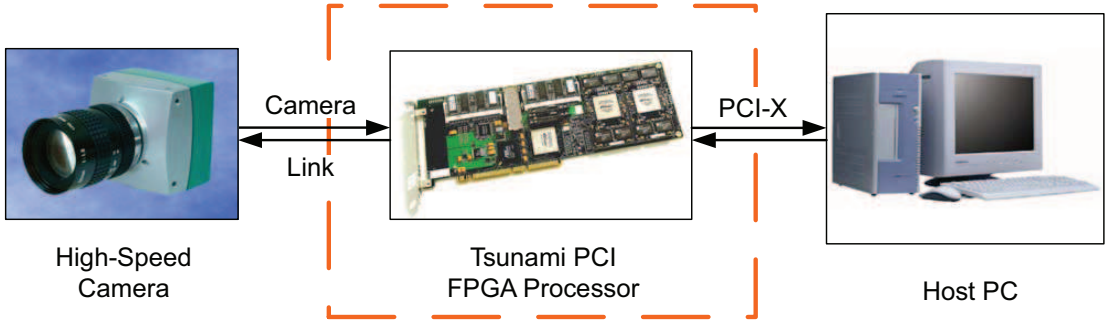

Fig. 5.11: Hardware platform consisting of a high-speed camera, an FPGA-board, and a host PC.

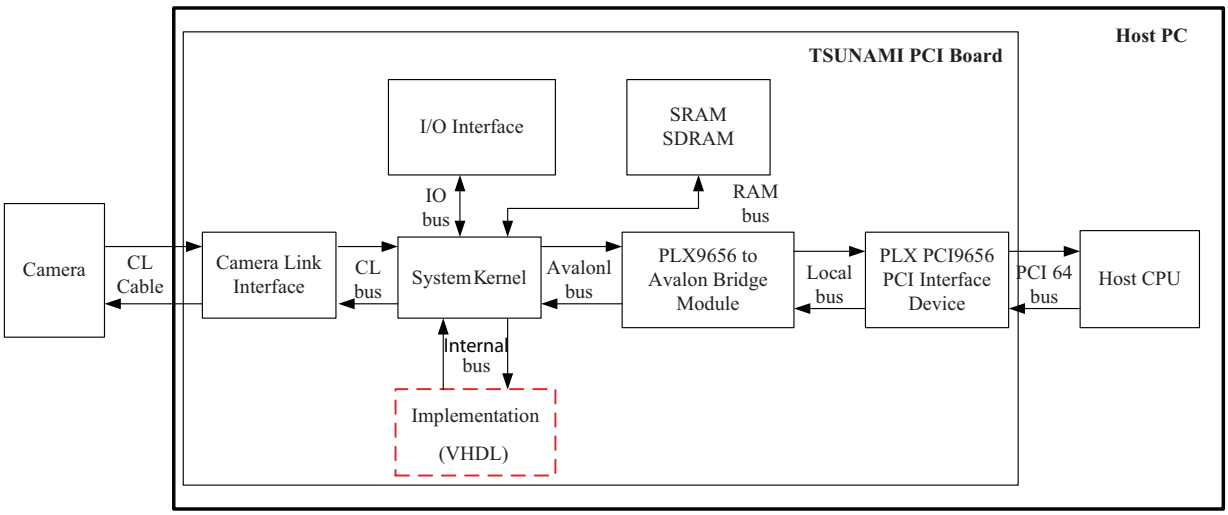

Fig. 5.12: Hardware system hierarchy [113].

and vertical directions. The other inputs of the EMD module are the previous outputs of the filters that are saved in the on-board *M-RAM*. Then, the outputs of the EMDs are connected to two modules: the *data insert module* and the *RF module*. Finally, all the results are written back to the original data flow, which is transferred to the host PC.

Moreover, there are three signals relevant to images: the frame signal (iFVAL), the line signal (iLVAL), and the data signal (iDVAL). These signals going with the image data flow will also be sent to all control modules in order to synchronize the image processing.

The structures of these modules are briefly introduced below.

- EMD Module The elaborated motion detector introduced in Section 5.1 has five main components that should be programmed in VHDL: logarithmic transformation, HP filter, LP filter, multiplication, and subtractor. The designs of multiplication and subtractor are relatively simple and neglected in this section.

The logarithmic transformation is realized through an LUT. The output of each photoreceptor is an 8-bit value and has a range between 0 and 255. For more precision, the values are extended to 16 bits in the following form: the highest bit indicates the value being either positive or negative, bits 14 to 6 belong to the integer part, and the last 6

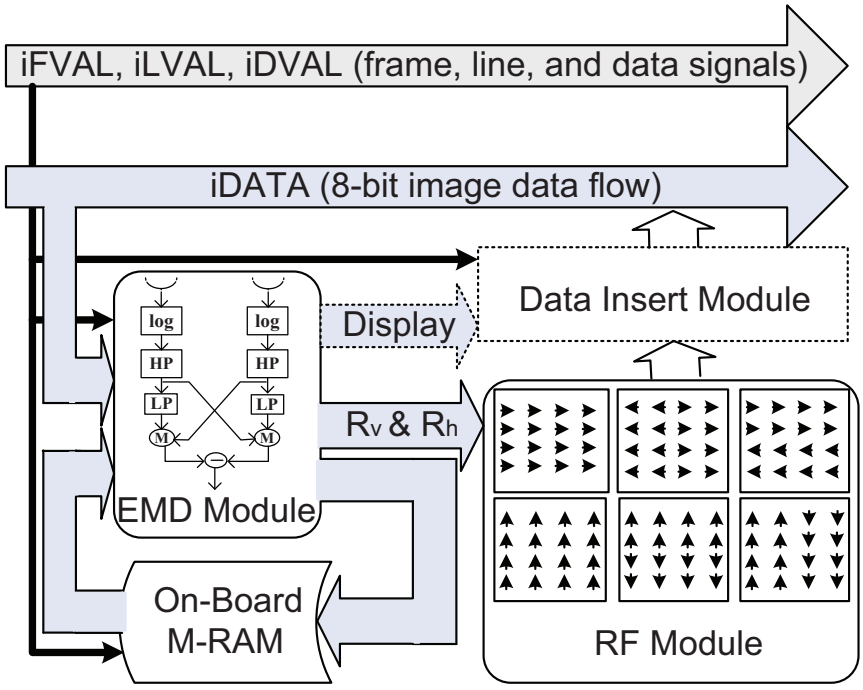

Fig. 5.13: VHDL program architecture consisting of the EMD module, RF module, M-RAM controller, and data insert module.

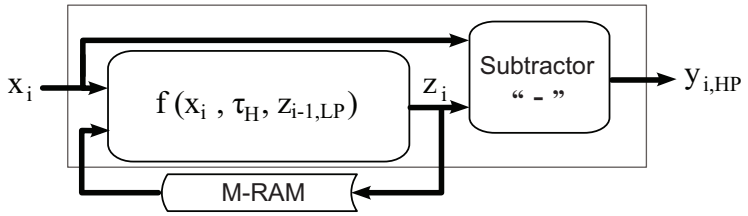

Fig. 5.14: The architecture of the HP filter based on the LP filter function f.

bits are the fraction part. After logarithmic transformation ($ln(1+I)$) of an image signal I, the output value is between 0 and 5.45.

The HP filter selected in this thesis is a recursive single-pole temporal HP filter. The design of the HP filter is based on a single-pole temporal LP filter, which is designed as follows:

$$y_{i,LP} = f(x_i, \tau_L, y_{i-1,LP}) = x_i/\tau_L + (1 - 1/\tau_L) \cdot y_{i-1,LP}, \tag{5.5}$$

where x_i denotes the input data, $y_{i,LP}$ denotes the output data of the LP filter, $y_{i-1,LP}$ denotes the previous output data of the LP filter, and τ_L the time constant of the LP filter.

Respectively, the HP filter is designed as follows:

$$y_{i,HP} = x_i - z_i, \tag{5.6}$$

where $y_{i,HP}$ denotes the output data of the HP filter, and z_i denotes the intermediate output of a LP filter with a time constant τ_H. Fig. 5.14 shows the structure of the HP filter. The intermediate output z_i of the LP filter f with the time constant τ_H is saved in the on-board memory M-RAM.

- M-RAM Controller In Fig. 5.13, an on-board M-RAM is required to save the output values of the filters. An M-RAM controller is designed to distribute the read and write addresses as well as write enable signals.

Note that the input frame from the camera is set up to a resolution of 1280×256 pixels, as the camera frame rate is only related to the number of pixel rows. However, only an image region of 256×256 pixels is processed, as the other image regions in the data flow are used to save the final and intermediate results. The write enable signal is used to perform this selection and a pixel counter which calculates the pixel number in a line is needed. When the pixel count is between 512 and 767, the output values of the filters are stored with the M-RAM controller. More information can be found in the data insert module section.

- RF Module As illustrated in Fig. 5.8, six RFs are applied for global motion detection. The output of each RF for each image is designed to be a signed 40-bit value (5 bytes). The highest bit indicates that the value is positive or negative. The total output values of the six RFs have 30 bytes.

- Data Insert Module In order to analyze and display the results on the host PC, a data insert module is designed to write the results of both H-type EMDs and V-type EMDs in the selected 256×256 range back to the original data flow. The image data which are not in the *Region Of Interest* (ROI) are replaced by the results of the EMDs as well as the outputs of the six RFs. When the data flow is sent to the host PC, the results can be read from it and thus the local motion vectors are drawn over the original frame. The results of the RFs are also read out and plotted in real time.

The main frequency of the FPGA system is 160 Mhz. For each image (1280×256 pixels), the transmission from the camera to the FPGA platform takes approximately 2 ms. Only 40 system clocks (about 0.25 μs) are totally charged additionally for the whole implementation on the FPGA platform. Thus, this system can work ideally at more than 450 fps.

5.2.3 Experimental Evaluation

The system was successfully tested in several experiments. Different motion manners were detected under various backgrounds. In the experiments, the backgrounds were always fixed in front of the camera while the camera was mounted on the end-effector of the 6-DOF Stäubli robot arm moving along some simple trajectories.

- Left-Right Motion

In the first experiment, the camera was performing a horizontally left-right motion. Fig. 5.15 (left) shows the optical flow on the background. The arrows represent the local optical flow detected by the elaborated EMDs. Almost all of them are pointing right. Since the outputs of the EMDs are influenced by several parameters, such as the velocity, the luminance, and the contrast, the lengths of the arrows are not identical throughout

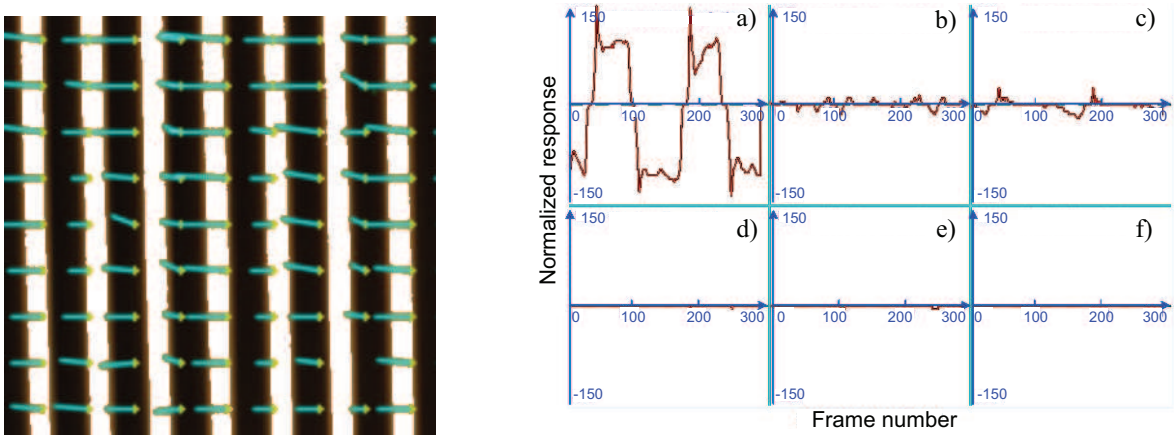

Fig. 5.15: Local motion and global motion detection using FPGA implementation of EMDs and RFs. A camera was performing a horizontally left-right motion in front of a pattern with vertical stripes. Left: optical flow detected by the EMDs; right: the global motion detected by the six RFs. a) horizontal component of left-right motion; b) horizontal component of forward-backward motion; c) horizontal component of clockwise-anticlockwise rotation; d) vertical component of up-down motion; e) vertical component of forward-backward motion; f) vertical component of clockwise-anticlockwise rotation.

the whole image. In addition, the spherical effect of the wide-angle camera also influences the result to some extent.

Fig. 5.15 (right) shows the responses of the six RFs indicating the global motion. They are normalized to the range between -150 and 150. Ideally, when the camera moves horizontally, only RF a) for the horizontal component of a left-right motion will have an output. And the outputs of the other five RFs should be zero. The result in this experiment is almost as expected. The positive/negative value denotes that the camera moves horizontally to the right/left. The RFs b) and c) for horizontal components have very small outputs as well, which are probably caused by the asymmetry between the left and right half-images as well as the top and bottom half-images. The outputs of the three RFs for vertical motion components d)–f) are almost zero. Combining the results of these six RFs, a conclusion can be drawn: The camera was moving horizontally right or left during this experiment.

- Forward-Backward Motion

The elaborated motion detector can also be used to detect the optical flow field induced by the camera moving backwards or forwards. A diffused or contracted optical flow field can be observed. In this experiment, the background of some concentric rings was applied. The circle center was set to be aligned with the optical axis of the camera.

By moving the camera backwards away from the background, a contracted optical flow field can be observed, as shown in Fig. 5.16 (left). In Fig. 5.16 (right), the outputs of RFs b) and e) for forward-backward motion are obvious. Moreover, the RFs a) and d) for left-right/up-down motion have relatively small outputs. The main reason is that the

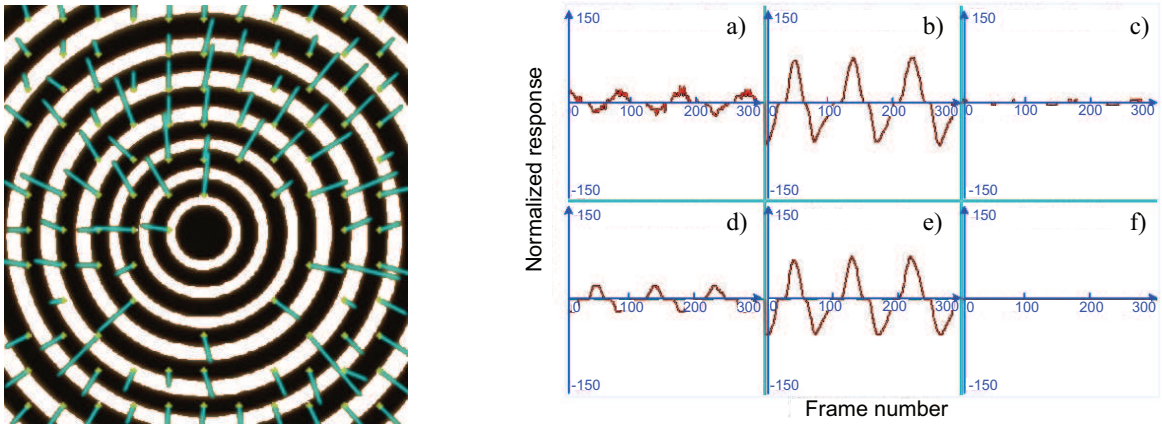

Fig. 5.16: Local motion and global motion detection using FPGA implementation of EMDs and RFs. A camera was performing a forward-backward motion in front of a pattern with concentric rings. Left: optical flow detected by the EMDs; right: the global motion detected by the six RFs.

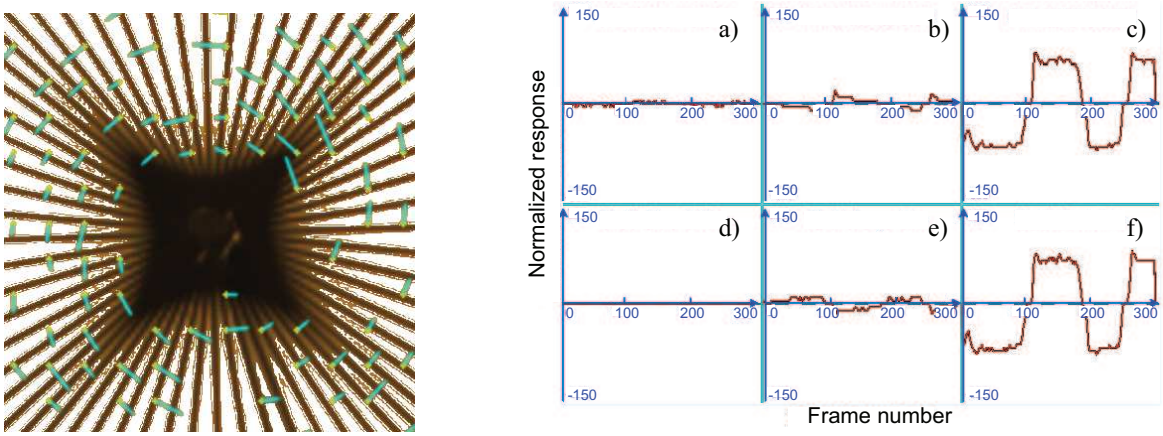

Fig. 5.17: Local motion and global motion detection using FPGA implementation of EMDs and RFs. The camera was performing a clockwise-anticlockwise rotation in front of a pattern with radial lines. Left: optical flow detected by the EMDs; right: the global motion detected by the six RFs.

center of the RFs was not exactly coincident with the optical axis of the camera.

- Clockwise and Anticlockwise Rotation

Fig. 5.17 (left) shows the optical flow of clockwise camera rotation with a pattern of radial lines and Fig. 5.17 (right) is the result of the RFs for rotation. In the experiment, the camera rotated alternately clockwise (positive values) and anticlockwise (negative values). Note that due to the symmetric pattern used here, the rotation responses to the horizontal and vertical directions/edges are almost the same.

According to the experimental results and discussion above, the system is demonstrated to be efficient at detecting local optical flow induced by different motion manners of the camera. The elaborated motion detectors and the RFs implemented on the FPGA platform can detect and assort motion behaviors quickly and correctly. The system is successfully tested at a frame rate of 350 fps with the resolution of 256×256 pixels. Apart from the time delays due to image capturing and data transfer, the system delay purely charged for the computation on the FPGA is approximately only 0.25 μs. In this respect, this system is more efficient than some existing solutions, such as the VLSI sensor in [66] (132-pixel image, 30 fps) and the FPGA implementation in [97] (88×88 pixels, 29 fps). This high-performance system can be applied for motion detection for humanoids and ground and aerial vehicles.

5.3 Quantitative Motion Estimation and Control

While the previous section focuses on qualitative motion detection based on fly's vision, this section deals with the quantitative motion estimation, aiming at closed-loop control of flying systems using insect-like vision. As the visual responses of a fly's compound eye and a camera with a planar sensor surface to the same environment are different, in the following part, motion estimation and control based on the fly's vision and the camera's vision are investigated first. Then, motion estimation based on natural images is further studied aiming at flight control. Finally, the estimation of complicated motion is discussed.

5.3.1 Fly-Vision-Based Motion Estimation and Control

First, motion estimation and control issues are investigated based on fly vision. The quantitative relationship between ego-velocity and the respective RF response is explored using mathematical solutions, simulations, and real-time experiments.

Mathematical Derivation

Theoretically, the response of an EMD $R_{\mathrm{EMD}}(t)$ of a sine-grating image with a single spatial frequency f_s can be formulated as follows [50]:

$$R_{\mathrm{EMD}}(t) = \frac{C^2 \cdot 2\pi\tau_L \cdot f_t}{1 + (2\pi\tau_L \cdot f_t)^2} \sin(2\pi f_s \Delta\phi), \tag{5.7}$$

where

$$f_t = f_s \cdot v, \tag{5.8}$$

with f_t denoting the temporal frequency of the input signal, v the velocity, C the amplitude, τ_L the time constant of the LP filters, and $\Delta\phi$ the angular distance of two photoreceptors in an EMD (see Fig. 5.3 and Eq. 5.1).

Considering images with a quasi full spectrum of spatial frequency such as natural

images, the response changes into an integral of f_s given by

$$R_{\mathrm{EMD,LP}}(t) = \int_0^\infty P(f_s) \frac{2\pi\tau_L \cdot f_s \cdot v}{1 + (2\pi\tau_L \cdot f_s \cdot v)^2} \sin(2\pi f_s \Delta\phi) \mathrm{d}f_s, \tag{5.9}$$

where $P(f_s)$ indicates the power spectral density, which can be strongly influenced by the image under consideration [50]. More information on this point can be found in Section 5.3.3. Through spectrum analysis of natural images, it is reasonable to assume $P(f_s) = \frac{1}{f_s}$. Fig. 5.18 (left) illustrates the EMD response $R_{\mathrm{EMD,LP}}$ to velocity at $\tau_L = 35\,\mathrm{ms}$. Note that the subscript "LP" is used here only to distinguish Eq. 5.7 and 5.9. Both response computation contains LP filters.

After inserting first-order HP filters into the EMD, the response of the elaborated EMD is

$$R_{\mathrm{EMD,HP}}(t) = \int_0^\infty P(f_s) \frac{2\pi\tau_L \cdot f_s \cdot v}{1 + (\frac{\tau_L}{\tau_H})^2 + (\frac{1}{2\pi f_s v \tau_H})^2 + (2\pi\tau_L f_s v)^2} \sin(2\pi f_s \Delta\phi) \mathrm{d}f_s, \tag{5.10}$$

where τ_H is the time constant of the HP filter. Suppose that

$$\frac{\tau_L}{\tau_H} = 2, \tag{5.11}$$

different responses at different τ_L values are obtained, illustrated in Fig. 5.18 (right). Using the HP filters, the EMD responses at low frequencies become smooth. Moreover, the relationship between the EMD response and the velocity is nonlinear and non-monotonic. To find a unique correspondence between the response and the velocity, a certain range of the velocity is constrained, in which the correspondence is monotonic. As shown in Fig. 5.18 (right), the lower the τ_L is, the larger the monotonic range is. Using the HP filters, the monotonic range for the same τ_L is also larger than that without using the HP filters (see Fig. 5.18 left). For example, the maximum response using $\tau_L = 65\,\mathrm{ms}$ is at the velocity of $v = 22\,\mathrm{deg/s}$, while the maximum response using $\tau_L = 35\,\mathrm{ms}$ is at the velocity of $v = 45\,\mathrm{deg/s}$. Then, an extension of the monotonic range is obtained through applying a small τ_L.

Simulational Determination of an LUT

After the relationship between the velocity and the EMD response is studied in theory, simulations are designed and conducted to determine an LUT, which represents a monotonic relationship between the velocity and the RF responses of input images.

Fig. 5.19 illustrates the input image of a fly when it is facing a homogeneous vertical stripe pattern. Since flies possess round complex eyes, the projected stripes are approximately homogeneously distributed.

A black-white stripe pattern with black stripes of 50 pixels wide and a distance of 50 pixels between two neighbored vertical stripes is chosen as the environment pattern around a fly. The fly is rotating around its vertical axis, resulting in a changing yaw angle and a horizontally moving stripe pattern in its field of view. The pattern could be filtered by a horizontal Gaussian filter in order to simulate the low resolution of fly vision. An

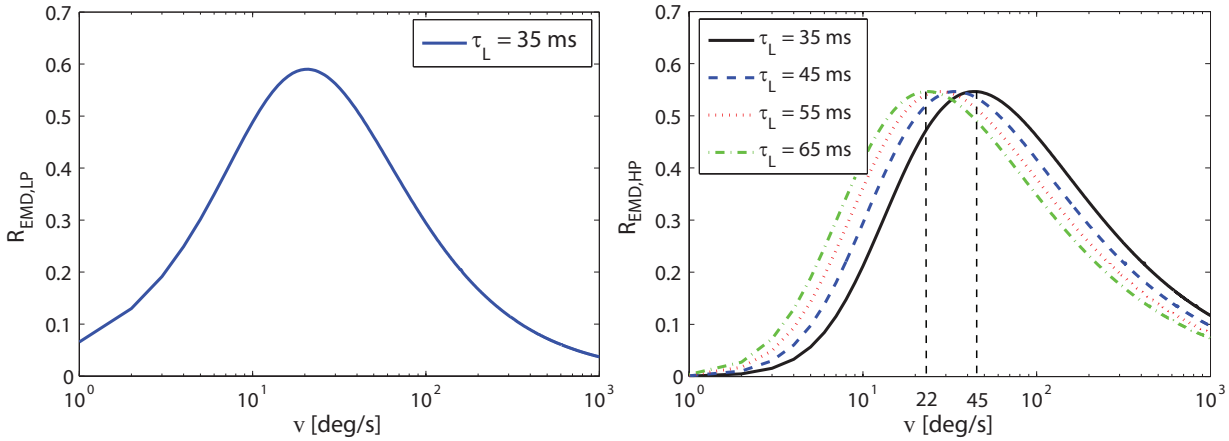

Fig. 5.18: Left: simulated normalized response of EMDs $R_{\mathrm{EMD,LP}}$ with respect to velocity at $\tau_L = 35\,\mathrm{ms}$; right: simulated normalized responses of the elaborated EMDs $R_{\mathrm{EMD,HP}}$ with respect to velocity at $\tau_L = 65\,\mathrm{ms}$, 55 ms, 45 ms, and 35 ms, where $\tau_H = 2\tau_L$.

Fig. 5.19: A fly observing a stripe pattern (left) and its projection on fly vision (right).

example without Gaussian filtering is shown in Fig. 5.19 (right).

To simulate the rotation, the pattern is moved rightwards at a speed of 1 pixel/frame in the first 100 frames, 2 pixels/frame in the second 100 frames, and 3 pixels/frame in the third 100 frames and so on, through which a constant velocity in every 100 frames and an overall ascending velocity along the time scale are achieved.

Fig. 5.20 (top-left) shows the RF response with respect to the frame number. Overall, the response increases with the increasing velocity first, then decreases. For the constant velocity in every 100 frames, a quasi constant response is achieved. An enlarged view of the responses between the 600th and the 700th image frames is also shown, in which the variation of of the responses for the same velocity is illustrated. The peaks which occur at the moment when a new velocity is applied are due to the operations of LP filters and HP filters. These kinds of responses can also be investigated in humans and animals when their velocity abruptly increases [126]. Using an average value of the responses in every 100 frames, an approximation of the relationship between the response and the velocity in pixels/frame is derived in Fig. 5.20 (top-right), which is very similar to the theoretically simulated one in Fig. 5.18.

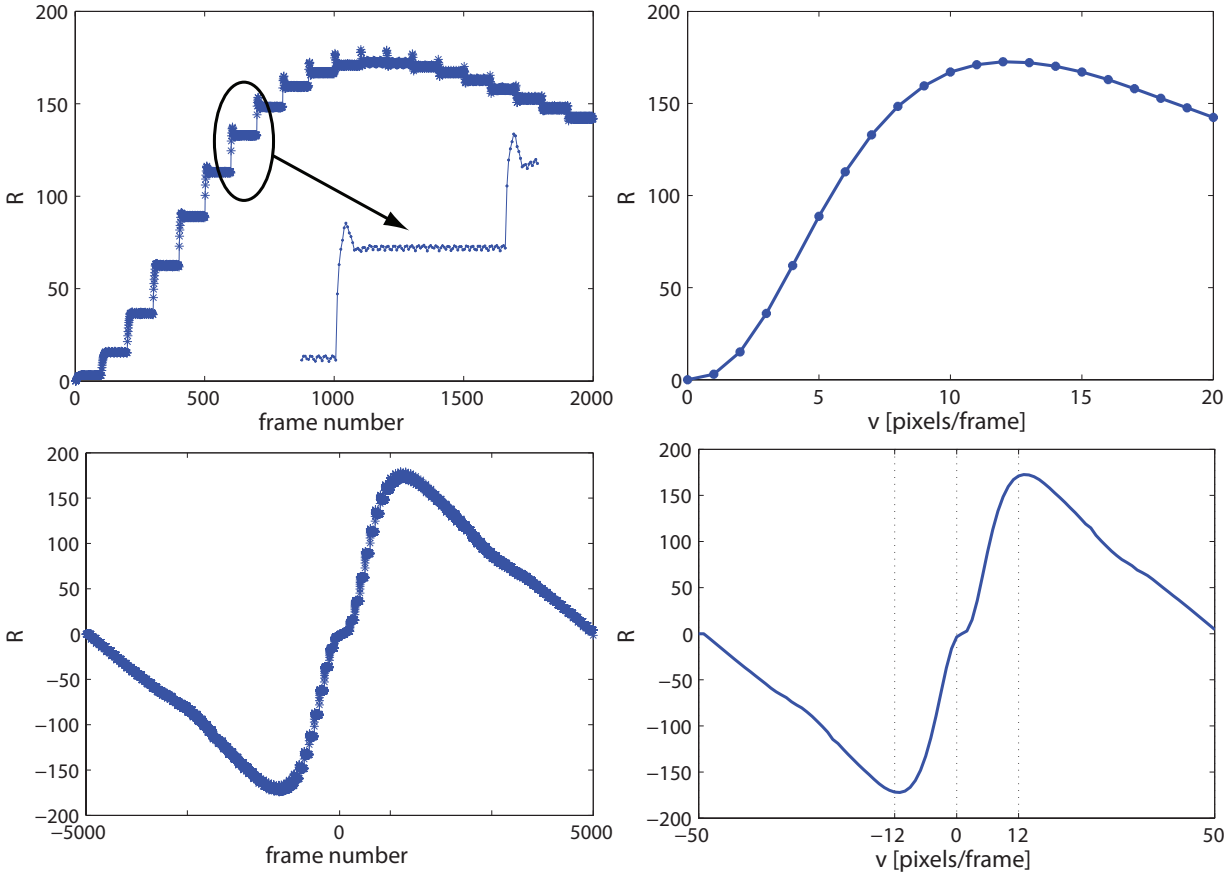

Fig. 5.20: Simulated results of the RF responses with respect to motion. Left column: the original RF responses; right column: the respective average responses. The negative frame number indicates an image variation in the opposite direction.

A similar simulation using the same pattern is conducted, in which the velocity is increased from 1 pixel/frame in the first 100 frames to 50 pixels/frame between the 4900th and the 5000th frames and also in the negative direction indicated by the negative frame number. The original RF responses are illustrated in Fig. 5.20 (bottom-left). An interesting symmetric response pattern is obtained here: The response at 50 pixels/frame (at the 5000th image frame) is the same at -50 pixels/frame (at the -5000th image frame). The reason is that at a velocity of 50 pixels/frame, the input image is the same as the one at a velocity of -50 pixels/frame. The image variation in the negative direction is the exact opposite of the image variation in the positive direction. The RF response possesses a periodic property. An RF response period of 100 pixels/frame is yielded here. Therefore, the monotonic range is also extended in the negative direction, for example, from -12 pixels/frame to 12 pixels/frame as shown in Fig. 5.20 (bottom-right).

Moreover, different responses can be obtained for the same velocities due to different spatial frequencies of the input images. The spatial frequency can be regarded as an indicator of the distance between the texture and the fly vision. The denser the stripes in the images are, the farther the stripes are from the fly, and the smaller the averaged RF responses are. This implication can be used to determine the distance between the vision sensor and the object at a known ego-velocity, which is further considered for obstacle

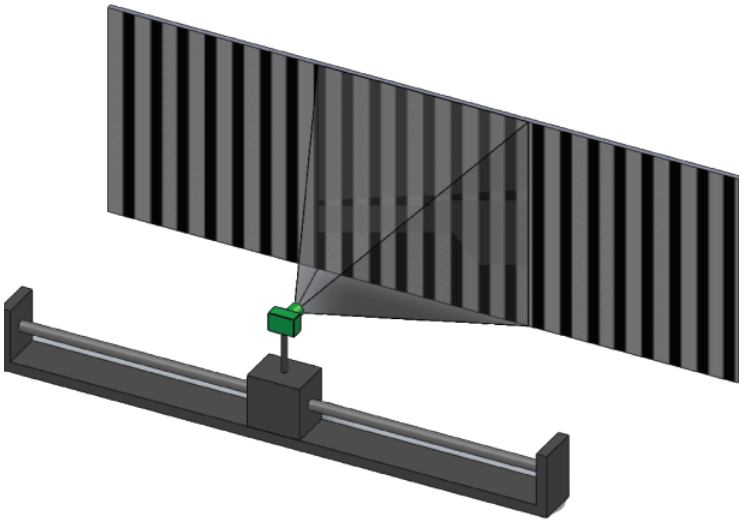

Fig. 5.21: Illustration of the experiment for determination of an LUT for the correspondence between RF response and the velocity.

avoidance in the next section.

Based on the simulation results, it can be concluded that the RF response is non-monotonic with respect to the motion. However, the monotony can be achieved in a constrained velocity interval. Based on the monotonic interval, a unique relationship between the motion and the RF response can be achieved.

Experimental Determination

An LUT between the motion and the RF response is experimentally determined in this section.

As mentioned, if a fly is rotating around its vertical axis facing a homogeneous stripe pattern, its visual projection of the stripe pattern is homogeneously distributed. Then, the varying of the yaw angle results in a translation in the horizontal axis in the projected input image. Therefore, the input images of a camera with translational movement facing a homogeneous stripe pattern can be used to imitate the yaw angle changing of a fly.

– Experimental Setup An extended setup of Fig. 5.11 is used, illustrated in Fig. 5.21 and 5.22. The camera is installed on a linear motor, facing a vertical stripe pattern. The camera data is transferred through the CL and processed on the FPGA platform. The results are transferred from a Windows PC to a Linux PC, which controls the linear motor via a control box. Moreover, motor sensor feedback is also measured and processed in this control box.

– LUT To obtain the relationship between the RF response and the camera motion, a large number of tests using different camera velocities aided by the linear motor were conducted. The linear motor was controlled by a *Proportional-Integral* (PI) controller with the parameters $K_p = 200$ and $K_i = 6$ to produce velocities from $0\,\mathrm{m/s}$ to $1\,\mathrm{m/s}$. A friction

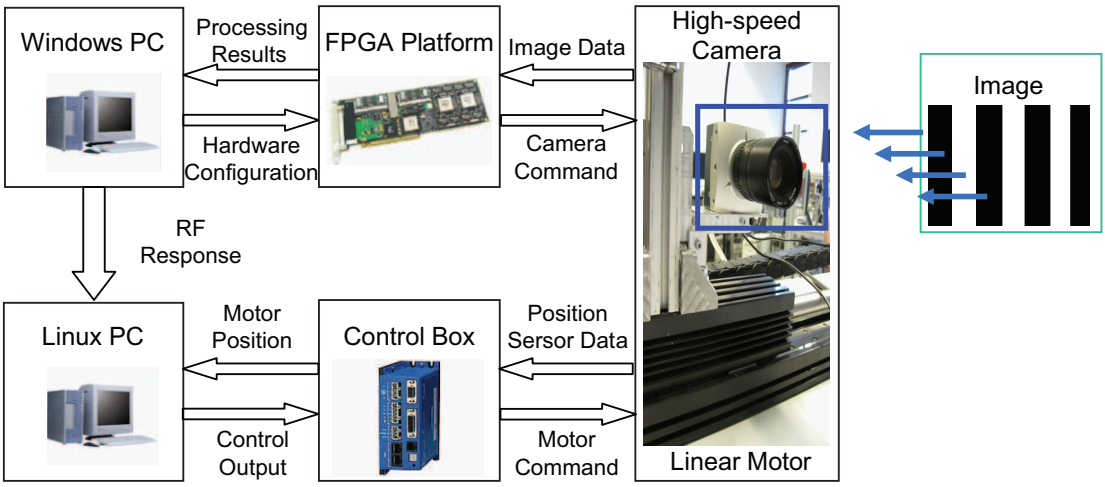

Fig. 5.22: Hardware components of the experimental setup.

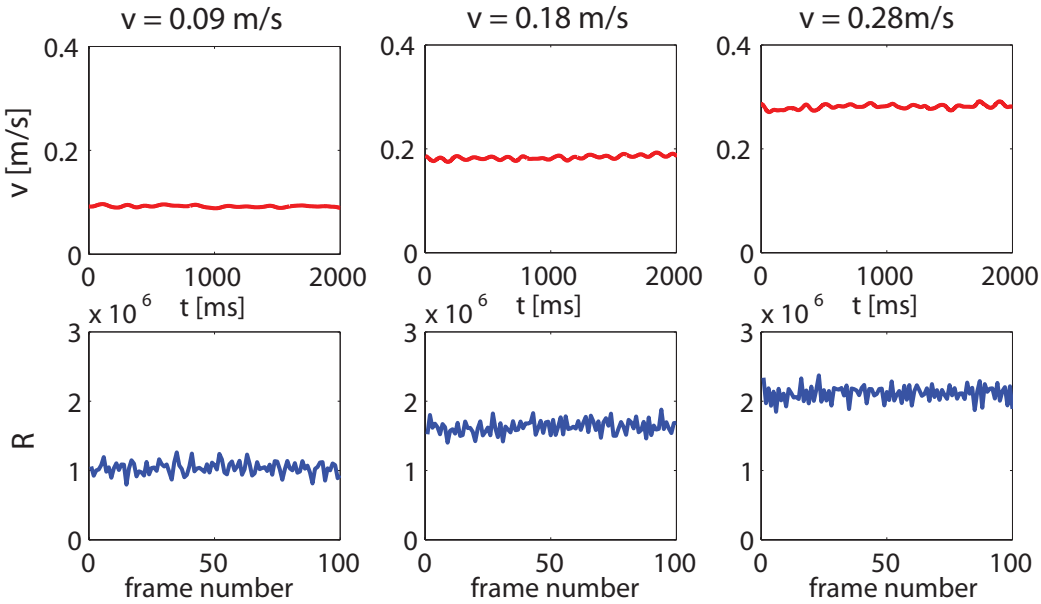

Fig. 5.23: Camera velocities with their corresponding RF responses.

compensator was also added to the control loop.

Fig. 5.23 shows three different camera velocities and their corresponding RF responses exemplarily. For each velocity, the oscillation of RF responses similar to the simulation results occurs. Using the averaged RF responses at various camera velocities, a diagram showing the relationship between the RF response and the camera velocity is illustrated in Fig. 5.24. At lower velocities, the RF response increases with an increasing velocity. After reaching a maximum at 0.42 m/s, the RF response decreases as expected.

Fig. 5.25 shows the RF responses to a continuously varying velocity between -1 m/s and 1 m/s. In both tests, a correspondence is noticed that the RF response reaches its maximum for the velocity of 0.42 m/s. Then, for velocities from 0 m/s to 0.42 m/s, an LUT representing the relationship between the RF response and the camera velocity is found, shown in Fig. 5.26, which is however biased by the current lighting conditions, shown

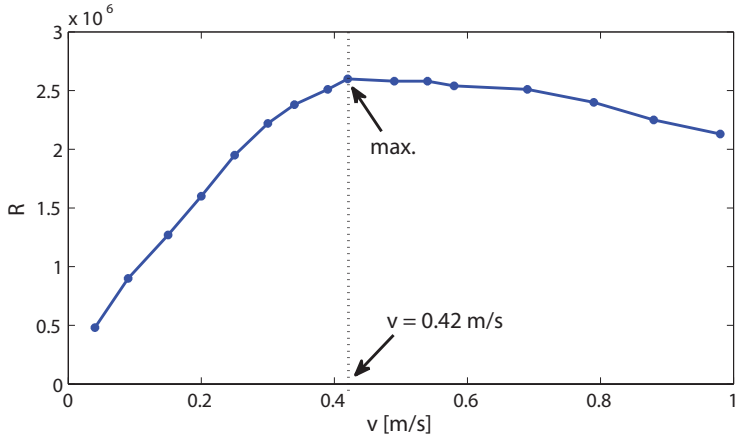

Fig. 5.24: The averaged RF responses to different camera velocities.

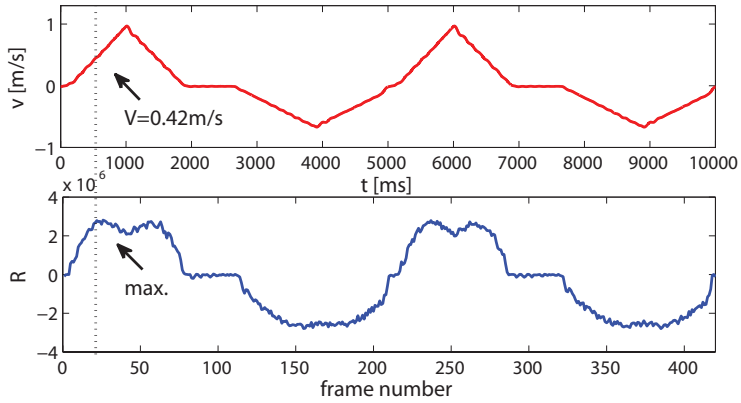

Fig. 5.25: The RF responses to the varying camera velocity.

below.

– Influence of Lighting Conditions The influence of different lighting conditions on the LUT is illustrated in Fig. 5.26. The RF response is proportional to the camera aperture. The brighter the current lighting is, the steeper the relationship curve is. Since the lighting intensity has a great influence on the RF response, the LUT is not unique in different lighting conditions.

As it is known, flies possess an organism called ocelli (see Fig. 5.1, left), which senses the lighting conditions. In this thesis, the LUT used in the following experiments was manually adjusted in advance according to the lighting conditions in the experimental environment. An LUT capable of the adaptation to the lighting conditions is subject to future work.

Closed-Loop Control

To verify the quantitative motion estimation based on the LUT, closed-loop control of camera ego-motion was conducted. The reference velocity along the linear axis increased from 0 m/s to 0.42 m/s and decreased to -0.42 m/s, illustrated by the dashed line in Fig. 5.27. The control performance and control errors using the visual feedback and motor feedback

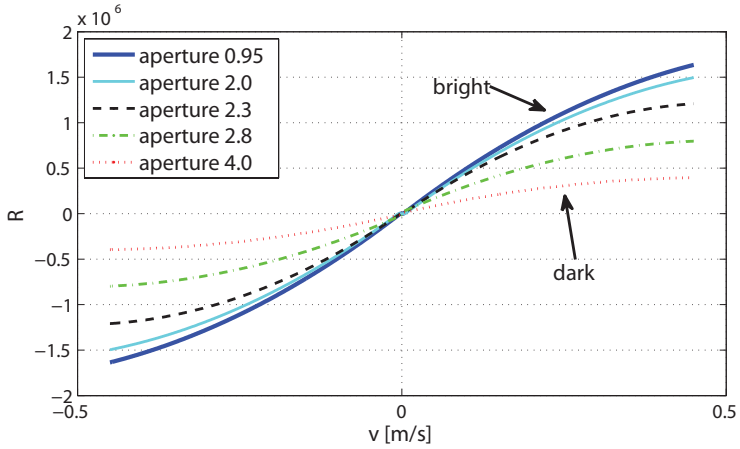

Fig. 5.26: Influence of lighting conditions on the smoothed relationship between the RF response and the camera velocity.

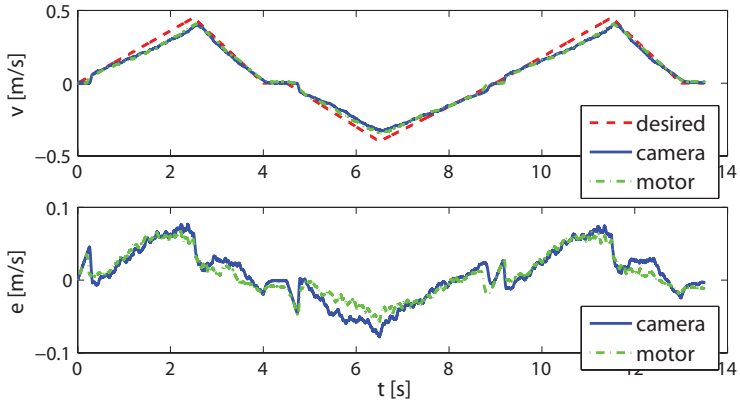

Fig. 5.27: Closed-loop control using motor encoder and visual feedback. Top: the camera velocity controlled using camera feedback based on the LUT and motor encoder feedback; bottom: the respective control errors.

are shown in solid lines and dot-dash lines. The controlled velocities using both feedbacks almost overlap with each other. The maximum error of visual feedback with respect to the reference velocity is approximately 0.025 m/s, lying in the same order of the control errors using the motor feedback, which is assumed to be very accurate. It is concluded that an LUT can solve the nonlinearity problem of the motion estimation based on the EMDs and the RFs and provide suitable feedback for closed-loop control.

5.3.2 Camera-Vision-Based Motion Estimation and Control

Considering camera's vision, a forward-looking camera on a quadrotor is simulated. In contrast to fly's vision, a camera has a planar sensor surface, resulting in an input image, in which the pattern is expanded at the sensor edges (see Fig. 5.28). Considering the planar sensor surface of the camera, a similar procedure to Section 5.3.1 is also conducted for the determination of an LUT of the RF response caused by yaw angle variation.

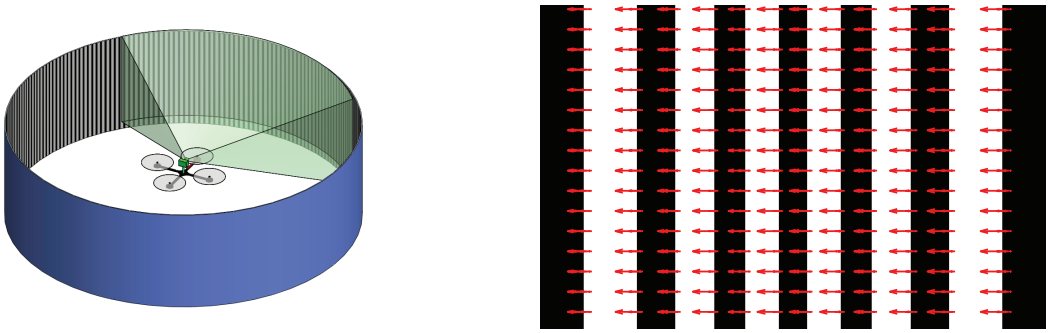

Fig. 5.28: A camera mounted on a quadrotor observing a stripe pattern (left) and the optical flow field detected by the elaborated EMD on a camera image (right).

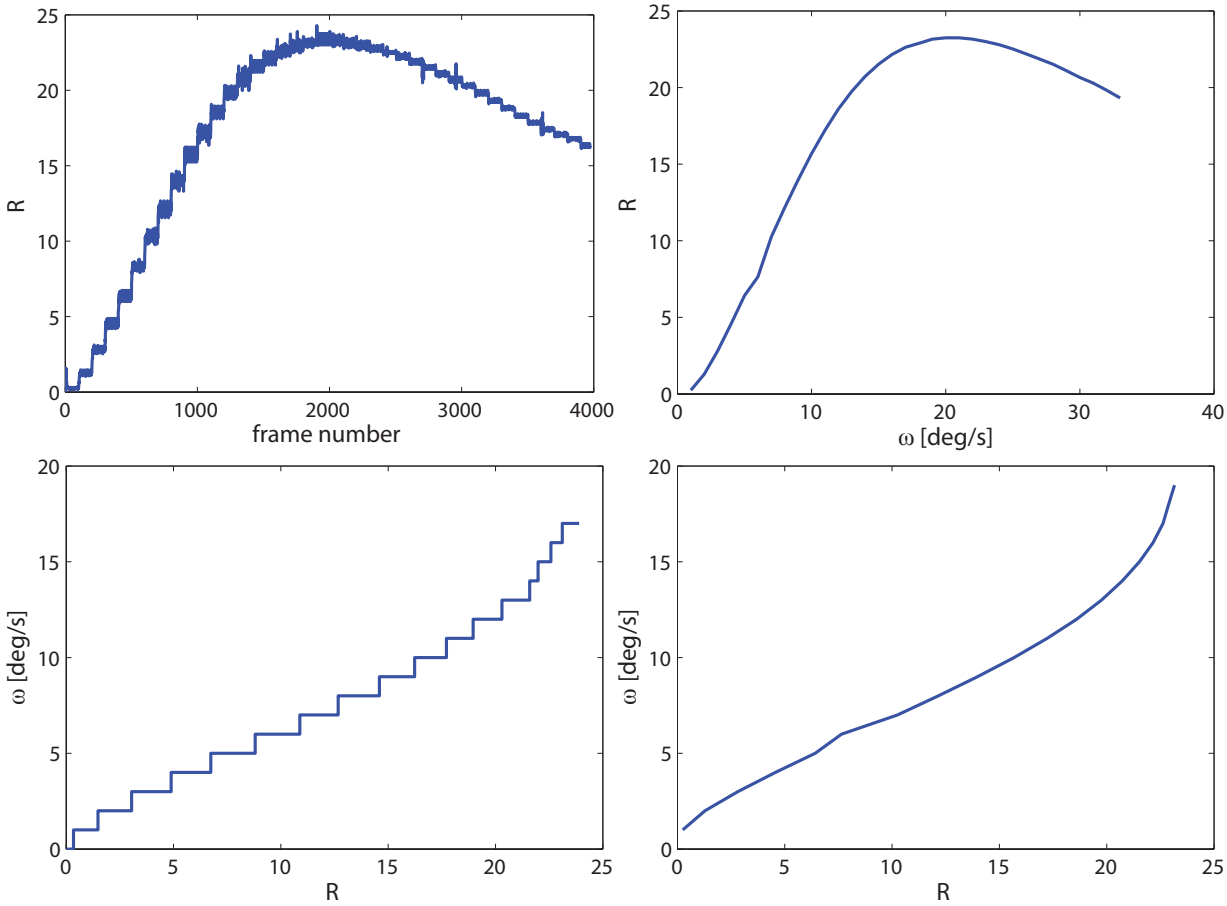

Fig. 5.29: LUT establishment for a forwards-looking camera with rotation around its vertical axis (yaw angle). Top: the original (left) and averaged (right) simulated RF responses; bottom: LUTs based on stepwise averaging (left) and interpolation (right).

LUT Establishment

Fig. 5.28 (right) shows the EMD-based optical flow (the arrows) of a simulated input image by rotating the yaw angle of the quadrotor. Note that the stripes in the input image of

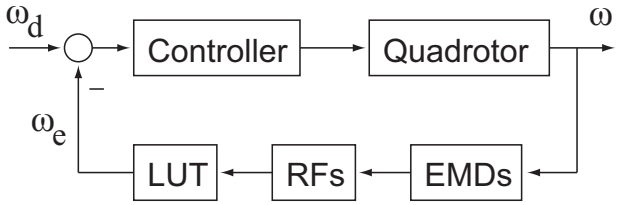

Fig. 5.30: Closed-loop control structure. ω_d: desired angular velocity; ω: actual angular velocity; ω_e: estimated angular velocity.

a homogeneous stripe pattern are not homogeneous and have a wider spectrum of spatial frequency than the one in Fig. 5.19 (right).

Fig. 5.29 shows the simulation results consisting of the original RF responses with respect to image frames (top-left), the averaged responses (top-right), and LUTs between camera velocity and the RF response using a stepwise averaging (abbr. LUTS, bottom-left) and using linear interpolation (abbr. LUTI, bottom-right).

Based on the interpolation method, an angular velocity under 19 deg/s using 100 frames/s approximately for the simulation can be estimated uniquely using LUTI. The LUTS based on a stepwise averaging of the original RF responses shows a similarity to the fly, human, or the other animals' vision, namely a small motion estimation resolution, while at certain responses the velocity cannot be uniquely estimated [126].

Closed-Loop Control

Now, both LUTI and LUTS are applied in closed-loop control of quadrotor yaw angle. Fig. 5.30 illustrates the control structure based on EMDs, RFs, and the LUT. Three trajectories of different desired velocities are given, namely constant velocity, stepwise-constant velocity, and constantly accelerated velocity.

The results are illustrated in Fig. 5.31. The left column shows the results using LUTI, while the right column shows the results using LUTS. For a constant desired velocity of 10 deg/s, illustrated in the first row, LUTI shows a smoother control trajectory than the LUTS, while the estimation of LUTI oscillates around the actual value. For a stepwise-constant velocity, both LUTI and LUTS perform well at lower velocities. At a higher desired velocity, the controlled actual velocities have very small oscillations around the desired value, while both estimations have relatively large oscillations around the desired value. For a constantly accelerated velocity, LUTI performs well in terms of small control errors and small estimation errors. In contrast, LUTS does not perform so well in terms of stepwise converging control errors and large oscillations of the estimation due to ambiguous relationship in the LUT. Overall, both LUTI and LUTS exhibit a good control performance. According to specified control conditions and requirements, LUTI or LUTS can be respectively applied. For example, LUTS is more beneficial for limited hardware requiring low computational cost or for the reduction of chattering in position control such as quadrotor altitude control in hovering. Further exploration at this point is subject to future work.

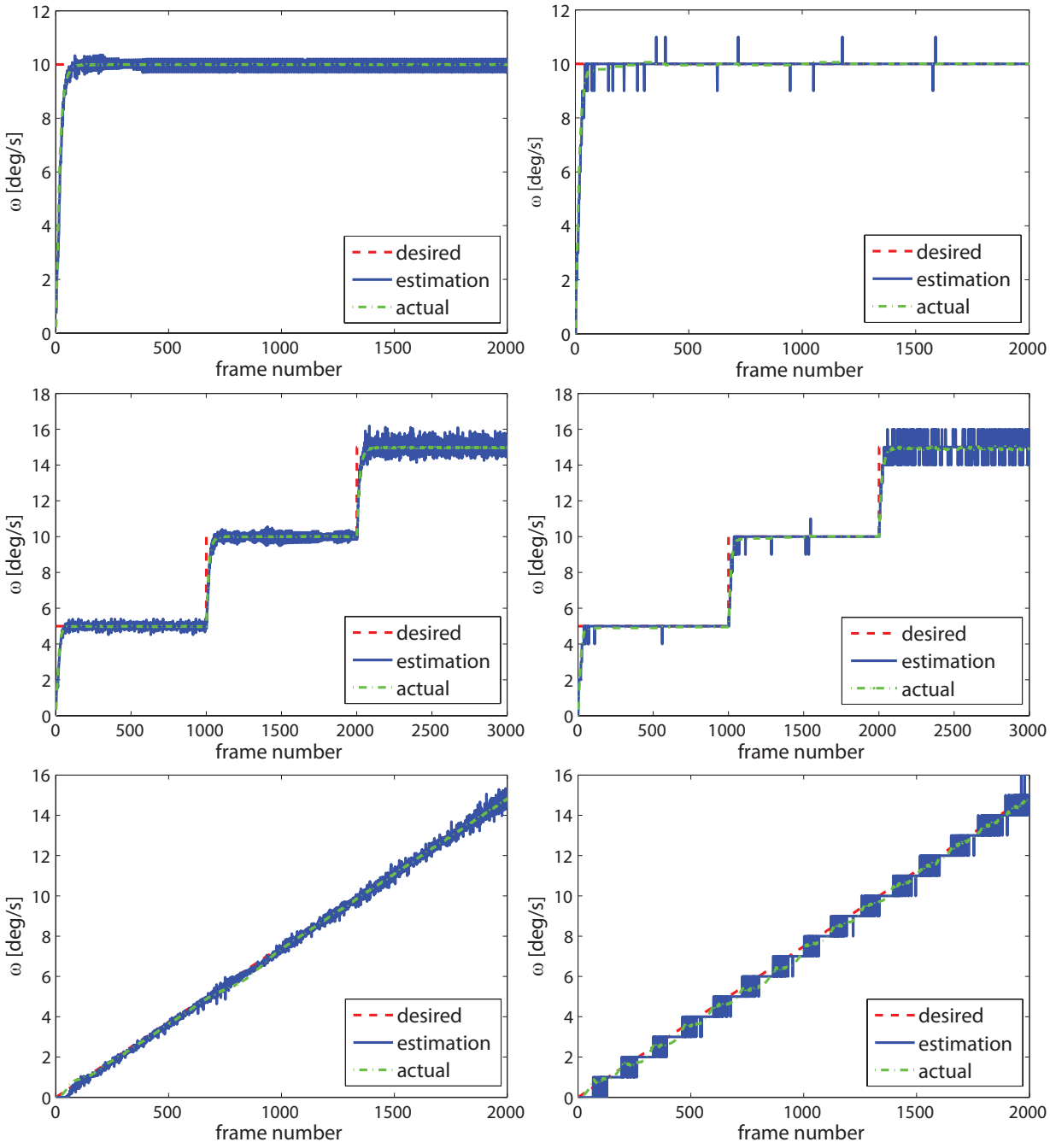

Fig. 5.31: Closed-loop control of yaw angle of the forward-looking camera using different desired velocities. Top: constant velocity; middle: stepwise-constant velocity; bottom: constantly accelerated velocity. Left: control results using LUTI; right: control results using LUTS. Dashed lines: the desired velocity ω_{d}; solid lines: the velocity estimation ω_e; dot-dash lines: the actual velocity after control ω.

5.3.3 Explorative Analysis Based on Natural Images

As mentioned previously, the spatial frequency of an input image has a major influence on the relationship between the RF response and the ego-motion. The sine-gating image for the mathematical derivation in Section 5.3.1 has a single spatial frequency f_s, while

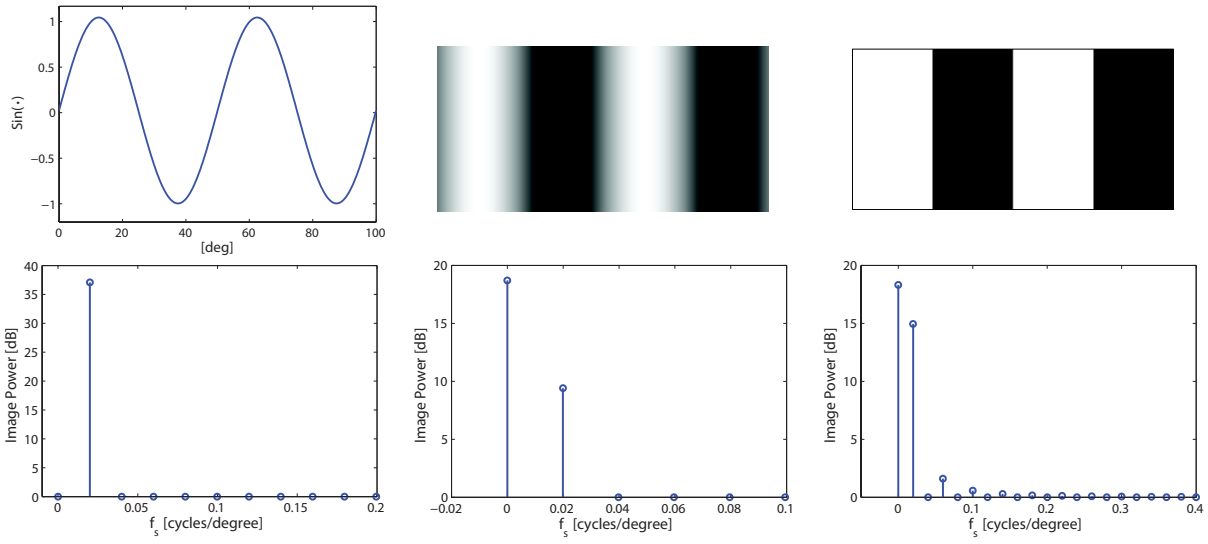

Fig. 5.32: Input signals (top row) and their spectra of spatial frequency (bottom row). From left to right: a sine-gating signal, a sine-gating image, and a black-white stripe pattern.

the Gaussian filtered stripe patterns investigated for fly vision and cameras only possess a narrow spatial frequency spectrum, as shown in Fig. 5.32. The applications of the LUT based on those images are limited, such as visual servoing in artificial environments. In contrast, natural images possess a wide spectrum of spatial frequency. In this section, the relationship between RF response and motion based on natural images is explored for further implications and inspirations.

The work in [40] proposed a robust neurobiological model for optical flow coding in natural scenes, consisting of five processing stages: phototransduction, spatial-temporal redundancy reduction, local motion estimation, local motion adaptation, and large-field integration. The authors tested their model using a variety of natural images and proved that the RF response computed using their model for the same angular velocity is almost the same. Moreover, the accelerations are also well estimated. In order to cover a wide spatial frequency spectrum, the prerequisite of high-resolution panoramic images is of particular importance using their model. However, as stated in the paper, the complexity of the model may be too much to realize in a real-time application.

Compared to [40], this work aims at closed-loop control using natural images with a low computational cost. Therefore, an explorative attempt is made here. Fig. 5.33 shows three of the natural images used in this thesis. Their spectra of spatial frequency are illustrated in Fig. 5.34 (left). A much wider spectrum of each input image is obtained in contrast to the artificial images shown in Fig. 5.32. Using these three images, the RF responses with respect to various angular velocities are simulated and illustrated in Fig. 5.34 (right). The RF responses and the relative errors are very similar among different images. The form and the trend are also similar to those using artificial stripe patterns. Therefore, it is concluded that it is possible to establish an LUT and closed-loop control using natural images, as shown in Section 5.3.2.

Fig. 5.33: Natural images 1, 2, and 3 (from left to right) used in the simulation.

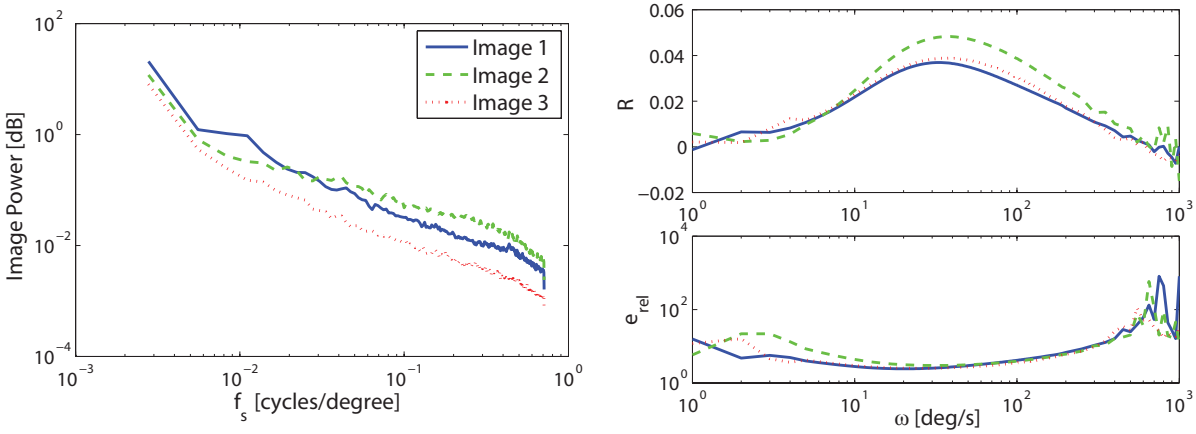

Fig. 5.34: Left: spatial frequency spectra of images 1, 2, and 3; right: the RF responses of images 1, 2, and 3 with respect to velocity and the relative error e_{rel}.

5.3.4 Explorative Analysis of Complicated Motion

After motion in one direction is extensively studied in the previous sections, complicated motion is considered here. An example is shown in Fig. 5.35. A camera has concurrent rotational and translational motion. Then, its input image provides a complicated flow field like c) in Fig. 5.35. The central problem is how to extract motion components from the available complicated RF response – flow field c), which means the rotation and translation responses a) and b) are searched for.

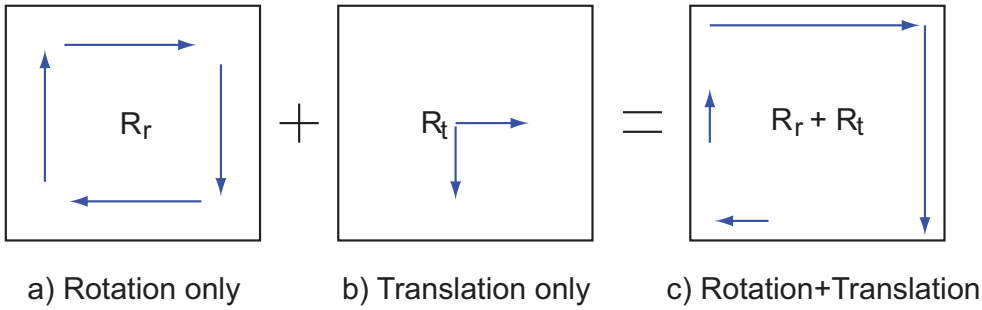

Fig. 5.35: Illustration of the composing of complicated motion – merging of concurrent camera rotation and translation.

Taking the quadrotor motion estimation in Chapter 3 as an example, for a downward-

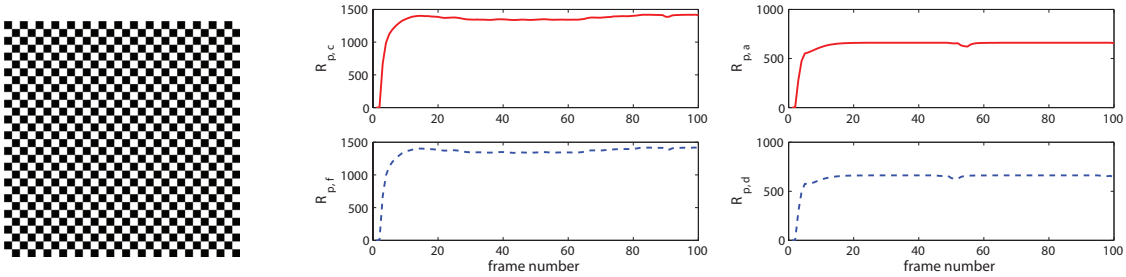

Fig. 5.36: The downward-looking camera is facing a chessboard pattern. Left: the chessboard pattern; middle: RF responses $R_{p,c}$ and $R_{p,f}$ of pure clockwise rotation on the RFs c) and f); right: RF responses $R_{p,a}$ and $R_{p,d}$ of pure translation on the RFs a) and d).

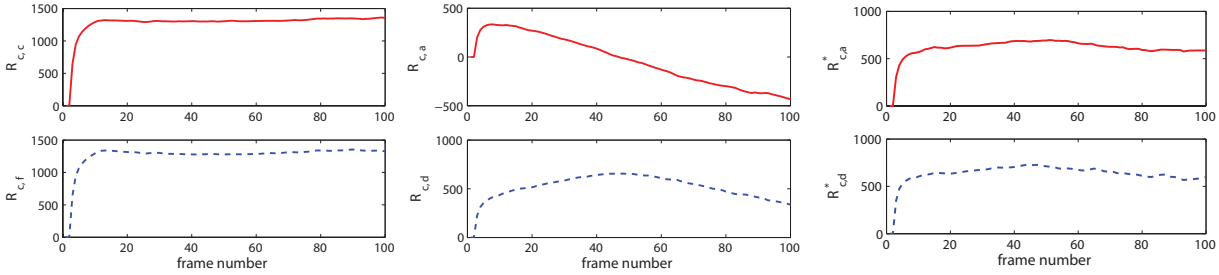

Fig. 5.37: The downward-looking camera is facing a chessboard pattern. Left: RF responses $R_{c,c}$ and $R_{c,f}$ of the complicated motion on RFs c) and f); middle: RF responses $R_{c,a}$ and $R_{c,d}$ of the complicated motion on RFs a) and d); right: rotated RF responses $R^*_{c,a}$ and $R^*_{c,d}$ of the complicated motion on RFs a) and d).

looking camera of a hovering quadrotor, assuming a constant altitude, the optical flow in the field of view contains both translation in left-right/forward-backward directions and yaw angle rotation. Therefore, simulations are conducted to investigate this motion decomposing problem.

Two simulations are conducted to compare the RF responses of the pure rotation and pure translation with the decomposed RF responses resulting from the overlapping complicated camera motion. The clockwise rotation velocity is set to 1 degree/frame in the camera frame, while the translation velocities in both directions are set to 1 pixel/frame in the inertial frame. The translation is defined in the inertial frame, as it is the conventional way for designing a desired quadrotor trajectory.

In the first simulation, an artificial chessboard pattern is used, as shown in Fig. 5.36 (left). The chessboard pattern possesses both horizontal edges and vertical edges. The RF responses of a pure rotation $R_{p,c}$ and $R_{p,f}$ on RFs c) and f) are shown in Fig. 5.36 middle, while the RF responses of a pure translation $R_{p,a}$ and $R_{p,d}$ on RFs a) and d) are shown in Fig. 5.36 (right).

If the quadrotor performs a complicated motion consisting of yaw angle rotation and translation in X_I-/Y_I-directions, the RF responses $R_{c,c}$, $R_{c,f}$, $R_{c,a}$, and $R_{c,d}$ of the resulting complicated motion based on RFs c), f), a), and d) are shown in Fig. 5.37 (left and middle columns), which indicate the rotation and translation component of the overall motion. Compared to the responses $R_{p,c}$ and $R_{p,f}$ for a pure clockwise rotation illustrated

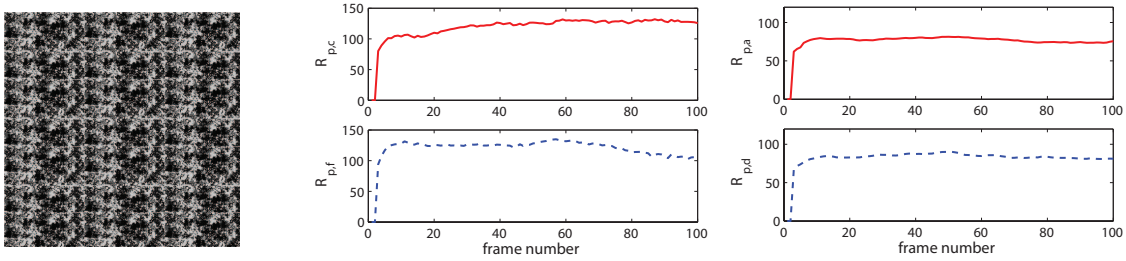

Fig. 5.38: The downward-looking camera is facing the irregular pattern used in Fig. 3.27 in Chapter 3. Left: the irregular pattern; middle: RF responses $R_{p,c}$ and $R_{p,f}$ of pure clockwise rotation on the RFs c) and f); right: RF responses $R_{p,a}$ and $R_{p,d}$ of pure translation on the RFs a) and d).

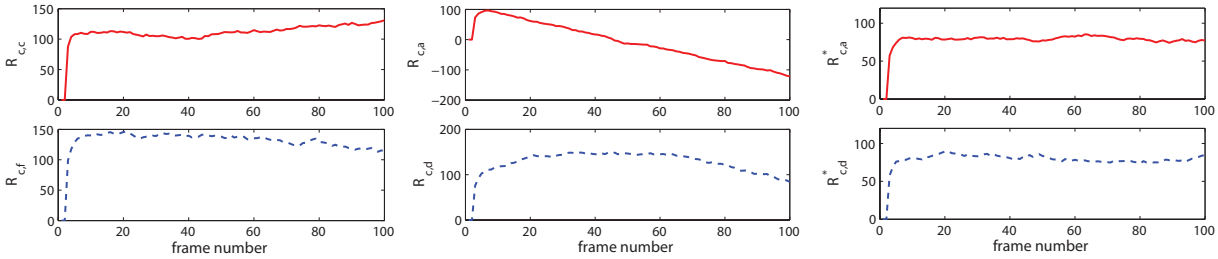

Fig. 5.39: The downward-looking camera is facing the irregular pattern used in Fig. 3.27 in Chapter 3. Left: RF responses $R_{c,c}$ and $R_{c,f}$ of the complicated motion on RFs c) and f); middle: RF responses $R_{c,a}$ and $R_{c,d}$ of the complicated motion on RFs a) and d); right: rotated RF responses $R^*_{c,a}$ and $R^*_{c,d}$ of the complicated motion on RFs a) and d).

in Fig. 5.36 (middle), it can be concluded that the rotation component is accurately decomposed from the complicated motion.

Note that the camera translation in X_I-/Y_I-directions is defined in the inertial frame and the RF responses $R_{c,a}$ and $R_{c,d}$ are detected in the camera frame. There is a rotational relationship ${}^I\boldsymbol{R}_c$ between them, which can be calculated using the decomposed rotation components $R_{c,c}$ and $R_{c,f}$. After the decomposed responses of translation motion $R_{c,a}$ and $R_{c,d}$ using RFs a) and d) are multiplied with the instantaneous rotation matrix, the following rotated responses $R^*_{c,a}$ and $R^*_{c,d}$ are obtained:

$$\begin{bmatrix} R^*_{c,a} \\ R^*_{c,d} \\ 1 \end{bmatrix} = {}^I\boldsymbol{R}_c \cdot \begin{bmatrix} R_{t,a} \\ R_{t,d} \\ 1 \end{bmatrix}, \tag{5.12}$$

as illustrated in Fig. 5.37 (right), which indicate the camera translation component of the overall motion in the inertial frame. Compared to the responses of a pure translation motion in the inertial frame as shown in Fig. 5.36 (right), it can be concluded that the translation component is also successfully decomposed from the complicated motion, facilitated by the rotation component decomposed from the complicated motion.

The same simulation is conducted using the same irregular texture as the one used in Chapter 3 for optical flow computation, illustrated in Fig. 5.38 (left). The RF responses

$R_{p,c}$ and $R_{p,f}$ of a pure rotation as well as $R_{p,a}$ and $R_{p,d}$ of a pure translation are shown in Fig. 5.38 (middle and right, respectively). The RF responses $R_{c,c}$, $R_{c,f}$, $R_{c,a}$, $R_{c,d}$ of the complicated motion and the rotated RF responses $R^*_{c,a}$, $R^*_{c,d}$ are shown in Fig. 5.39 (left, middle, and right, respectively). Similar results to those using the chessboard pattern are obtained.

Overall, the conclusion can be drawn that the RF response can be applied in complicated motion estimation. Based on the decomposed global motion detection, a quantitative LUT can be applied, to provide a complete feedback of quadrotor motion for control tasks.

5.4 Towards Insect-Inspired Collision Avoidance

As it is known, flies perform collision avoidance all the time when flying. Respectively, a fast collision avoidance capability using the visual information is also a critical issue for the safety of the flying system. In order to realize stable and safe flying of the quadrotor, two basic flying behaviors are considered: a stable tracking behavior and a high-speed obstacle avoidance behavior. Fig. 5.40 (left) illustrates the envisioned quadrotor flying behavior in a finite state machine. In tracking behavior, the quadrotor tracks the moving ground robot using the on-board downward-looking camera. Moreover, the insect-inspired motion detection and motion estimation with high temporal sensitivity is envisioned to alert the quadrotor and trigger the obstacle avoidance behavior using, for instance, an additional forward-looking camera. Since this bio-inspired obstacle detection algorithm is very computationally efficient and can be implemented on-board, the quadrotor deploys a very fast response to obstacles and then enters into the obstacle avoidance behavior.

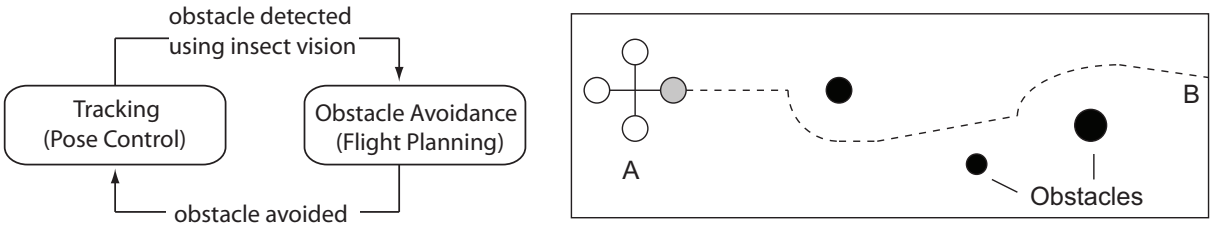

Fig. 5.40: Left: Finite state machine of envisioned quadrotor flying behavior. Right: example scenario of pillar avoidance while flying from A to B.

Fig. 5.40 (right) illustrates a possible scenario: During the flight, it is essential to avoid positive obstacles such as vertical pillars in a hall while flying from A to B, where estimation of the obstacle position to the quadrotor is critical information.

As mentioned in the previous section, the RF response is also related to the spatial frequency of the stripes which indicates the distance of the stripes to the vision sensor. Therefore, assuming that the quadrotor flying velocity and the dimension of obstacles are known, the relative distance between the quadrotor and the obstacle can be derived based on the RF response.

Simulations are conducted to investigate the relationship between the RF response and the relative distance at a constant quadrotor flying velocity. The results are shown in Fig. 5.41. The quadrotor starts to fly in the direction of an obstacle with a distance of

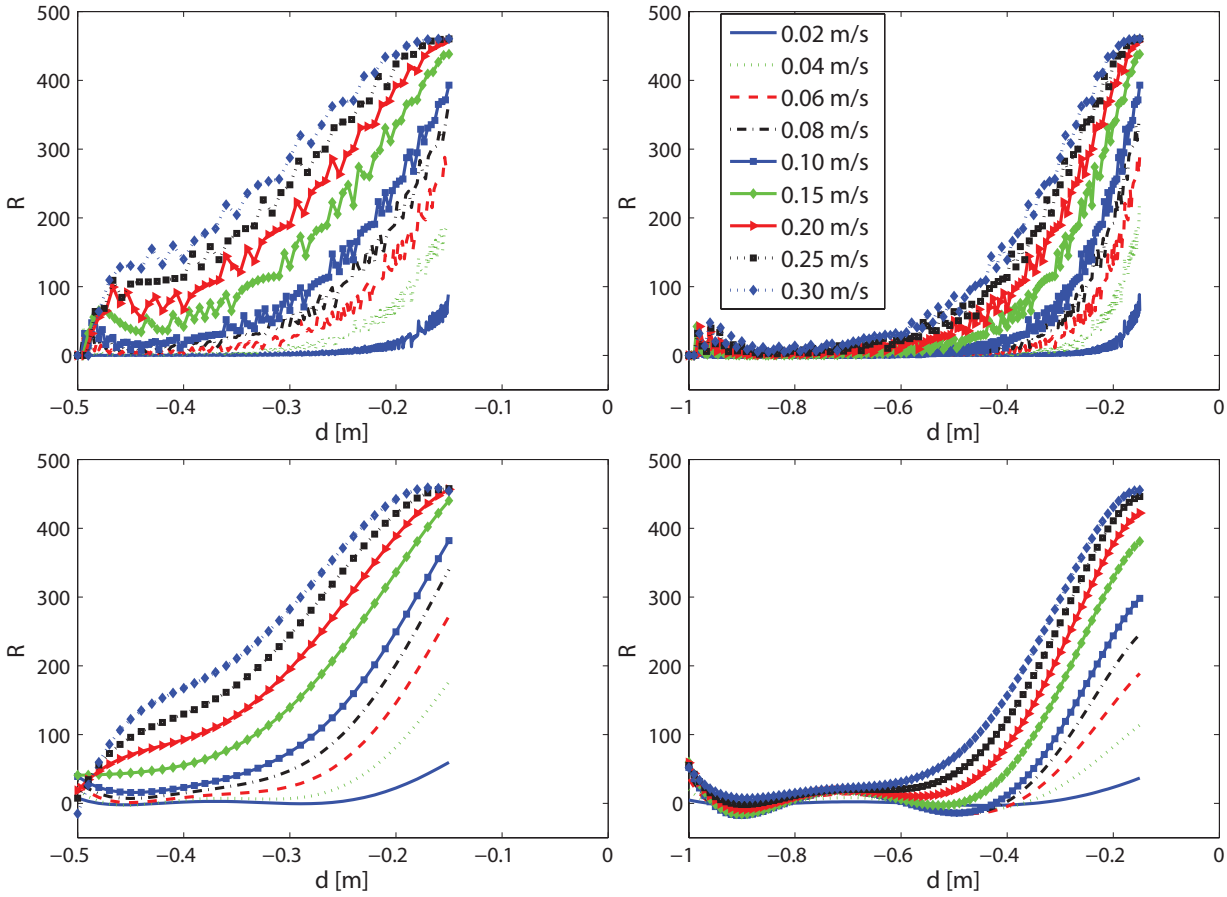

Fig. 5.41: Relationship between the RF response and the relative distance between the quadrotor and the obstacle at known velocities. Left column: the quadrotor flies from 0.5 m towards the obstacle; right column: the quadrotor flies from 1 m towards the obstacle. Top row: the original RF responses; Bottom row: the smoothed responses.

0.5 m (Fig. 5.41 left) and 1 m (Fig. 5.41 right) with various constant velocities. The top figures are the original responses based on the input images during flight, while the bottom figures are the respective smoothed responses.

Overall, the RF responses increase with a decreasing relative distance. At the same distance, the RF responses increase with an increasing velocity. Those phenomena also correspond to humans' intuitive perception. Interestingly, it is shown that the RF responses are not the same if the quadrotor flies from a different start point. But still the nearer to the obstacle the quadrotor is, the more similar the RF responses are (note that the significant difference between the two smoothed responses is mainly due to the smoothing functions). Therefore, it is reasonable to set a threshold R_{th} for the response to trigger the quadrotor to detect obstacles and re-plan its flight trajectory, in order to avoid obstacles.

A flight planning algorithm is shown in Algo. 1 as an example. A quadrotor flies straight forward in a bounded space. If the quadrotor reaches the boundary, it turns 90 deg back to the flying space. If optical flow pairs in columns computed by EMDs are detected in its image ROI (see Fig. 5.42 right), an obstacle between the optical flow columns is detected in its desired flight trajectory. Then, the RF response R are computed. If $R > R_{th}$, the quadrotor turns 20 deg to avoid the obstacle. The simulation result is shown in Fig. 5.42

Algorithm 1 Behavior: *obstacle avoidance*

```
set image ROI
while 1 do
  capture an image
  calculate EMD optical flow in ROI
  if optical flow pairs in columns detected then
    define obstacle
    calculate RF response R
    if R > R_th then
      quadrotor turns 20 degree
      set HP/LP to zero
    end if
  end if
  if reached boundary then
    quadrotor turns 90 degree
  end if
end while
```

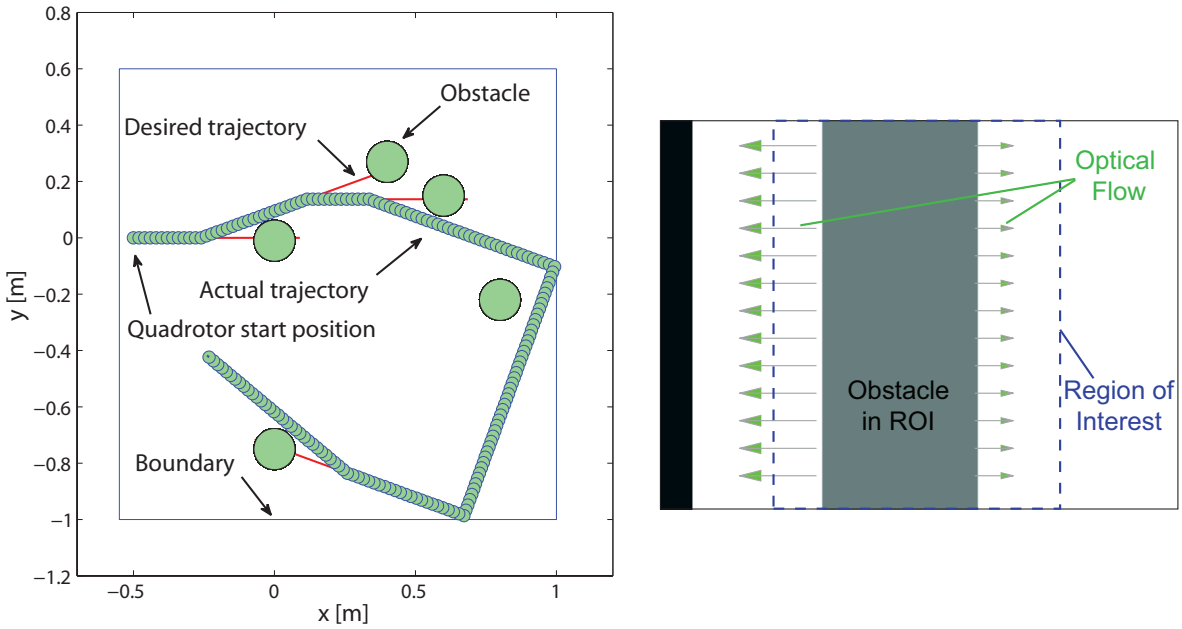

Fig. 5.42: Left: simulated quadrotor flying trajectory in a space consisting of several obstacles; right: an input image with an obstacle detected in the image ROI.

(left). The quadrotor succeeds in avoiding all the obstacles.

Another possible extension similar to the obstacle avoidance considered here is that flying through an unblocked area, e.g. a door or a window, based on the visual responses. These aspects open up a variety of research directions in flight planning and are subject to future research.

5.5 Discussion

Limited Inter-Frame Motion

Due to the requirement of a monotonic relationship between the RF response and camera velocity, the inter-frame velocity is limited in a bounded extent. Therefore, in order to detect and estimate motion with a higher velocity, a higher frame rate is desired. A high-speed implementation such as FPGA-implementation described in the previous section should also be equipped to fulfill the requirement. However, there is always a trade-off between highly accurate algorithms and high-speed implementation.

Importance of Vision-Based Gyroscopes

The RFs for rotation detection and the rotation estimation proposed in this thesis are very essential when considering quadrotor applications, as the horizontal motion of a quadrotor is realized by roll and pitch angles, and the heading direction is one key system state for quadrotor flight such as for tracking and landing. One of the possible extensions and applications is to establish vision-based gyroscopes based on RF responses, which can be an important complement to the standard gyroscopes in terms of improved accuracy and robustness to noise.

Hardware Limitation

Due to the relatively large dimension of the FPGA-board used here, this implementation was not able to be applied to the quadrotor. The algorithms and implementation are applied to a smart camera SC-MVC01 from VIDEOR eneo [15] of a cooperating partner [7]. This drawback can also be overcome by using a small FPGA module [82].

Multi-Sensory Motion Estimation

The insect-inspired motion estimation considered in this work has only covered vision-based aspects. However, biological models commonly use multi-modal sensors to accomplish this task. For instance, humans apply vision, inner ear, and even tactile perception to navigate. As mentioned in Section 5.1, flies use compounded eyes to perceive visual flow, ocelli to measure the brightness in the environment, halteres as a gyroscope to control flight, and also the hair over the body wings to perceive wind or pressure.

For a more accurate and faster response, multi-modal sensor cooperation could be a reasonable solution as stated in Chapter 3. The bio-inspired vision system, the on-board IMUs, or even tactile sensor skin could be combined to realize a high-performance insect-like flying robot, which is subject to future work due to currently limited hardware.

Computational Intelligence

Considering the disadvantageous characteristics of insect-like vision such as relatively low resolution and low accuracy, controllers based on fuzzy logic or neural networks might be compatible for insect-like flight control.

5.6 Summary

As an alternative to traditional development of vision-guided flying systems, biologically inspired techniques have the advantage in terms of efficiency. Transfer and implementation of neurobiological results of vision-based motion estimation are of particular importance and necessity for the current development of cognitive systems. Moreover, how the qualitative characteristics of biological modeling can be transferred into normally quantitatively controlled technical systems is a very interesting and challenging question.

In this chapter, the novel vision and visuomotor behavior models inspired by biological paradigms are extended first, in order to adapt to the typical dominant and preferred motion behaviors of the quadrotor. Two new RFs for rotation detection are proposed. High-speed implementation using compatible hardware – an FPGA platform – is accomplished to obtain the effectiveness of the neurobiological algorithms. The performance of the implementation is sufficient to deal with video frame rates of 350 fps or above for a frame size of 256×256 pixels.

As EMDs and the RFs suggested in fundamental studies can only provide a solution to qualitative motion detection, the respective motion estimation is established with the help of LUTs and extensively explored considering the influences of specific parameters such as the perception difference between flies and cameras, lighting conditions, the spatial frequency and frequency spectrum of input images. Closed-loop control and obstacle avoidance are exploratively investigated, which show a promising possibility of fully controlling MAVs in future work.

Limitations of this bio-inspired model include the dependency on backgrounds with contrast as a common problem in computer vision. Multi-rate sensor data fusion with, for example, IMUs would be necessary for applications in the scenarios with irregular backgrounds, which is subject to future work. Moreover, the online computation of scene motion fields while the quadrotor is in motion is another challenge of dynamic vision in dynamic environments. In addition, control issues based on insect-like vision should be further studied, e.g. visual servo control [137] or applications of fuzzy logic controllers and neural networks.

6 Conclusions and Future Directions

6.1 Concluding Remarks

The development of *Unmanned Aerial Vehicles* (UAVs) has in recent years become one of the focuses of active research, since they can extend the operating capability in a variety of areas such as military, industrial, and civilian domains. Above all, *Micro Air Vehicles* (MAVs) have gained a great interest in the robotics domain because of their small size and possible applications in indoor, complex, everyday environments. To study various aspects of a flying system, a quadrotor is chosen as the platform used in this thesis due to the robust mechanism and holonomic dynamics, which exhibits challenging motion estimation and control problems for autonomous flight.

Since vision is one of the most powerful tools for information acquisition and is widely used in most biological organisms such as humans and animals for autonomous navigation, vision-guided flight and navigation are considered able to overcome the aforementioned challenges. Moreover, insect-vision-inspired neurobiological models in particular are expected for further development of MAVs.

In this thesis, a heterogeneous air-ground multi-robot system is developed as a test-bed, containing a mini-quadrotor with complete on-board integration of sensor data processing and control algorithms as well as a wheeled ground robot. This work aims to establish a vision-guided flying system containing accurate pose/motion estimation as well as stable and effective control, such that this vision-guided MAV can conduct a complete performance including take-off, hovering, tracking, and landing stably and safely with the help of the ground robot. Furthermore, bio-inspired vision strategies for flight control are investigated as an attractive alternative to traditional paradigms. Applications and examples are presented for demonstration and evaluation. The main approaches along with the main contributions are highlighted below.

One of the fundamental but challenging problems of flight control is accurate pose and motion estimation of the aerial vehicle. To deal with the drift problem of on-board *Inertial Measurement Units* (IMUs) and to be able to acquire more information about the other agent – the mobile ground robot – in the environment, a monocular camera facing downwards is equipped on the quadrotor. Vision sensors have, however, the disadvantages of limited field of view, relatively low sampling rates, and complex image processing, all of which are critical for pose and motion estimations of a highly dynamic system. In Chapter 3, aiming at obtaining accurate pose/motion estimation of a highly dynamic quadrotor relying only on on-board sensors, a thoroughly designed high-frequency fusion of the inertial data and the vision data is accomplished. Based on multi-modal sensor information, a continuous-discrete *Extended Kalman Filter* (EKF) is applied for the multi-sensory multi-

rate data fusion, where the high-frequency IMU data drive the process model and the low-rate vision data correct the estimation. Data synchronization is completely conducted based on the accurately measured time delay. Moreover, switching between marker-based pose estimation and optical-flow-based motion estimation is conducted for different quadrotor positions. Various real-time experiments considering the quadrotor tracking the mobile ground robot show that 1) high accuracy and high frequency of the pose/motion estimation in dynamic behaviors are obtained through the multi-sensory multi-rate data fusion; 2) one of the first autonomous quadrotors based on minimal on-board sensors and the complete integration of sensor data processing is achieved.

Based on the accurate and high-frequency pose and motion estimation in Chapter 3, a stable and effective control design is investigated in Chapter 4. Quadrotors themselves exhibit a challenging control problem due to non-linear, under-actuated, highly unstable dynamics, time delay, as well as the limited payload and computational capacity requiring a simple control structure. Up to now, most state-of-the art works have only considered the control design in simulations, while the other standard works that consider real system implementation use a much simplified system model in limited scenarios. Chapter 4 aims at designing, implementing, and discussing adequate control structures for the quadrotor, to overcome the aforementioned challenges and to improve the quadrotor flying behavior in an integrated manner. The system model used here is not much simplified, in order to preserve the original dependency of system states. A PID-controller, an optimal LQ controller, and non-linear controllers such as backstepping-based controllers and sliding mode controllers are carefully adapted to the quadrotor system and discussed. Based on simulation results, an integrated control design is accomplished for quadrotor take-off, hovering, tracking, and landing on the mobile ground robot and evaluated in real-time experiments. The main contributions presented in Chapter 4 are: 1) the adaptation, implementation, and real-time evaluation of various control strategies based on a relatively complex system model; 2) an integrated control design considering controller combinations proposed and evaluated in a complete flying scenario for the first time.

After Chapters 3 and 4 solve the flight control problems in a traditional manner, Chapter 5 focuses on insect-inspired motion detection and estimation as an efficient alternative extension. The fundamental findings in biology and neuroscience show that insects have a small but efficient and sensitive visual system for real-time flight stabilization and control. This would provide an excellent solution for MAVs on-board vision development, which requires simple and fast computation due to limited payload, restricted computational capacity, and fast, dynamic self-motion. In Chapter 5, the novel vision and visuomotor behavior models inspired by biological paradigms are extended first, in order to adapt to the typical dominant and preferred motion patterns of the quadrotor. Two new *Receptive Fields* (RFs) for rotation detection are proposed. High-speed implementation using compatible hardware – a *Field Programmable Gate Array* (FPGA) platform – is accomplished to obtain the effectiveness of the neurobiological algorithms. The performance of the implementation is sufficient to deal with a video frame rate of 350 fps for a frame size of 256×256 pixels. The respective motion estimation is established with the help of *Look-Up Tables* (LUTs) and extensively explored considering the influences of specific parameters such as perception difference between flies and cameras, lighting conditions,

as well as the spatial frequency and spectrum of input images. Closed-loop control and obstacle avoidance are exploratively studied and show a promising possibility of fully controlling MAVs based on insect-like vision in future work. The twofold contributions in Chapter 5 are: 1) extension of efficient qualitative motion detection and its high-speed implementation on an FPGA platform; 2) explorative investigation of quantitative motion estimation quadrotor flight control.

Summarizing, the overall contributions of quadrotor motion estimation and control proposed in this thesis are the novel system configuration, significant improvements in quadrotor pose/motion estimation in terms of high accuracy and high frequency, enhanced control performance in relation to effectiveness and integrity, as well as advanced exploration in insect-inspired flight control. One of the first autonomous quadrotors only using on-board sensors (a monocular camera and IMUs) is developed and its performance is extensively evaluated in simulations and real-time experiments. The contributions significantly advance the state of the art in motion estimation and control of a vision-guided autonomous flying system and serve as a signpost for future research.

6.2 Outlook

Quadrotors, as one of the most attractive MAV platforms, have been intensively studied due to their wide application domain. Research on quadrotors provides the possibility of enabling autonomous flight and intelligent deployment. However, the research up to now mainly focuses on low-level hardware design and the development of control algorithms. There is still a large number of open questions and interesting future directions remaining, some of which are suggested below.

- *Improved vision guidance of autonomous flight* – The main advantage of vision guidance of flying systems is that vision can provide a large amount of information and direct measurement without integration, which is the most natural method used in autonomous biological systems for localization, planning, and reaction to the environment. Therefore, feature-based or scene-based visual processing without the use of artificial markers can provide more flexibility for MAV applications in complicated natural environments. However, due to the limited computational capacity, the state-of-the-art works considering a complex vision guidance commonly use an additional ground station for information processing, which brings problems of time delay and also limits the exertion of autonomy and the work space. An improved vision guidance which is not only simple and effective but also natural and flexible is subject to future work.

- *High-level coordination and cooperation of the air-ground multi-robot system* – In order to improve task performance and capability of flying systems, cooperation of one or more MAVs and *Unmanned Ground Vehicles* (UGVs) in multi-robot systems can be taken into consideration. In this work, the ground robot serves as a mobile reference and provides motion information to facilitate quadrotor flying. Thereby, high-level coordination and cooperation of the air-ground system are some of the

most interesting and challenging problems, such as 3D simultaneous localization and mapping as well as search-and-rescue.

– *Insect-inspired quadrotor control in full Degrees Of Freedom (DOFs)* – Due to the small size, efficient low-resolution information processing, and their high-speed, parallel sensorimotor structure, insects are regarded as an intelligent biological model for MAV flight control. In this thesis, a preliminary exploration of insect-inspired yaw control for quadrotors is conducted and should be extended to full DOFs first. Moreover, scenarios such as obstacle detection and collision avoidance are interesting topics remaining in this area. Insect-like multi-sensory data fusion and combination with probabilistic models such as the Kalman filter are also of particular interest. In addition, controllers based on fuzzy logic or neural networks might be compatible for insect-like flight control and should be considered.

Research on vision guidance, cooperation with ground agents, and bio-inspired strategies will have a large impact on the development of vision-guided autonomous flying systems. Applications of MAVs are expected to be an inevitable component of life in the future.

A Quadrotor Platform

A.1 Hardware Description

The quadrotor platform used in this thesis is a Hummingbird quadrotor from Ascending Technologies GmbH [3]. It is a small and lightweight platform which is designed for lab experiments. Based on the original setup of the quadrotor, a monocular camera and a small board with an Atom CPU (referred to as "ATOM-board") are installed in this thesis. A top view of the platform is shown in Fig. A.1.

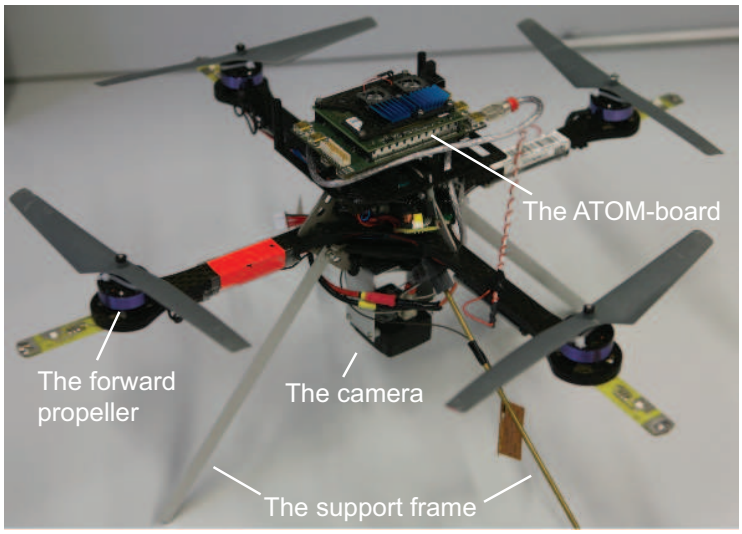

Fig. A.1: Top view of the quadrotor platform.

The diameter of the quadrotor is 36.5 cm. It possesses four flexible, harmless propellers of 19.8 cm each, which are driven by brushless DC motors. The weight without battery and other additional hardware is 219 g. The maximum thrust of four motors is 1320 g. The maximum payload is 350 g.

A control board on the platform called "X3d" contains an ARM processor which operates an inner-loop controller to stabilize rotations around three axes (roll, pitch, and yaw) (see Fig. A.2). Originally, three gyroscopes (Murata ENC-03R) and three axial accelerometers are integrated on this control board, providing sensor data to the inner-loop controller at 1 kHz. More information can be found in [3].

The on-board camera and an extra computation unit are installed in this thesis. The on-board camera is a Firefly MV camera (FMVU-03MTM/C) from Point Grey Research Inc. [12], shown in Fig. A.4. The key parameters of the camera are listed in Tab. A.1. More information can be found at [103].

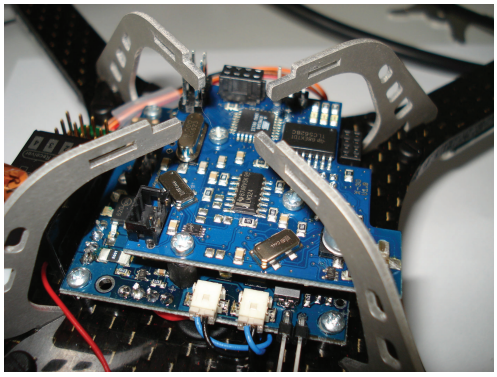

Fig. A.2: The X3d board [102].

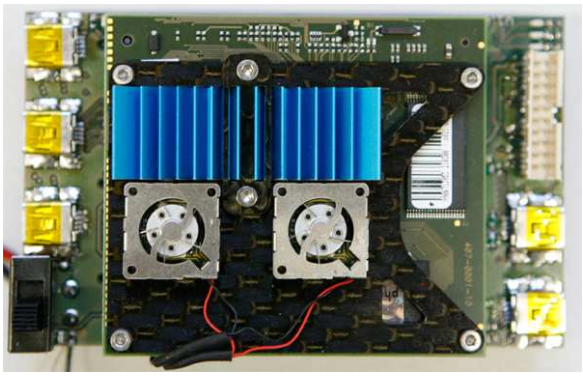

Fig. A.3: The flying netbook [3].

Fig. A.4: The Firefly MV camera [103].

Specification	Description
Mass	37 gram (with microlens)
Dimensions	24.4 × 44 × 34 mm
Image	Color
Resolution	640 × 480 pixels
Frame Rate	60 FPS
Focal distance	2.2 mm
Field of view	96° × 68°
Interfaces	USB 2.0

Tab. A.1: Key parameters of the camera [103].

The ATOM-board with an Atom CPU of Intel Corp. (1.6 GHz Dual Core) called "Flying Netbook" is developed by Ascending Technologies GmbH (see Fig. A.3). Its dimension is 100×60×25 mm and its weight is 90 g. The maximum power consumption is 5 W. It is capable of image data processing, sensor data fusion, and computation of the outer-loop control law, which makes autonomous flight possible. The on-board camera is connected via USB interface with the ATOM-board, while the data transfer between the ARM-Processor and the ATOM-board is via Universal Asynchronous Receiver Transmitter (UART). More information can be found in [3].

In addition, a support frame consisting of four legs are mounted to uphold the quadrotor. Two of them are made of flexible copper and can damp the landing.

The total weight of the fully equipped quadrotor is approximately 700 g, which is beyond the maximum payload plus the quadrotor self-weight. In spite of this, the quadrotor flied stably and safely in the experiments which indicates that the control design in this work is stable and effective.

A.2 Experimental Measurement of Time Delay

The procedure of the measurement of time delay for the IMU data and vision data is presented here in detail.

A.2.1 Time Delay of the IMU Data

The time delay of the IMU data consists of three parts: 1) IMU sampling time; 2)IMU data transfer from IMUs to the ARM-processor (referred to as "arm" below); 3) IMU data transfer from the ARM-processor to the ATOM-board, illustrated by Fig. A.5. The linear accelerations a_x, a_y, a_z and rotation velocities p, q, r are measured by the IMUs at 1 kHz, which means a sampling time less than 1 ms. Using the IMU measurements, the roll angle Φ_{imu} and the pitch angle Θ_{imu} are estimated at the same rate on the ARM-processor. Then, the IMU data and angle estimations are transmitted together to the ATOM-board via UART. Varying of the total transfer time is mainly caused by this transmission.

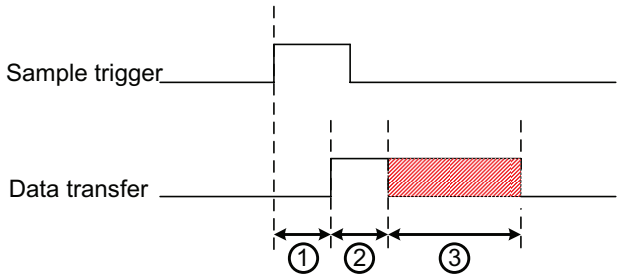

Fig. A.5: Time delay analysis of IMU data. 1) IMU sampling time of less than 1 ms; 2) IMU data transfer time from IMUs to the ARM-processor on the quadrotor; 3) IMU data transfer time from the ARM-processor to the ATOM-board.

IMUs and ATOM-Board Time Synchronization

There is a time stamp in each data package sent by the ARM-processor to the ATOM-board. This time stamp indicates when the IMU data are sampled. It runs from 0 to 65535, which is generated by a 16-bit counter, and gives more advantage for the data synchronization later. But note that the time unit used on the ARM-processor is not identical as that on the ATOM-board. As shown in Fig. A.6, the time stamps are different at the same measurement points. Therefore, the time on the ARM-processor and the time on the ATOM-board should be firstly synchronized locally.

The ratio between ARM-processor time t_{arm} and the ATOM-board time t_{atom} is denoted by P_t:

$$P_t = \frac{t_{\text{atom}}}{t_{\text{arm}}}. \tag{A.1}$$

To obtain P_t accurately, the communication between the ARM-processor and the ATOM-board runs a couple of minutes and repeats several times. The value of P_t is evaluated to be 1.9841175. In Fig. A.7, the synchronized ARM-processor time t_{arm} and ATOM-board time t_{atom} overlap each other.

IMU Transfer Time

The IMU data are transmitted to the ATOM-board for further processing via UART from the ARM-processor. The data refreshing periods of the transmission on the ARM-processor and on the ATOM-board are illustrated in Fig. A.8. The actual IMU data refreshing period

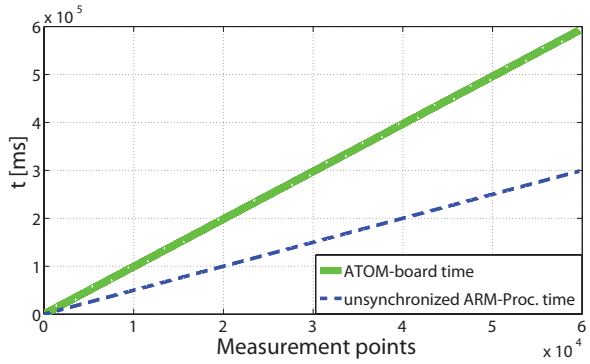

Fig. A.6: Unsynchronized ARM-processor time and the ATOM-board time.

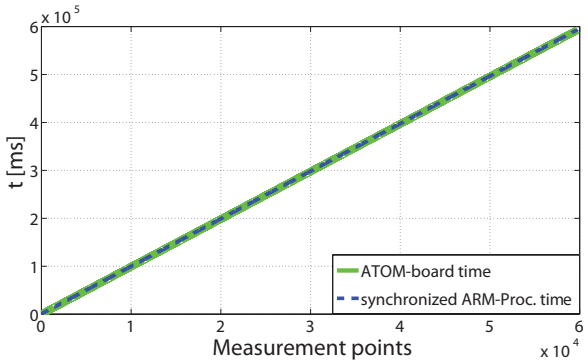

Fig. A.7: Synchronized ARM-processor time and the ATOM-board time.

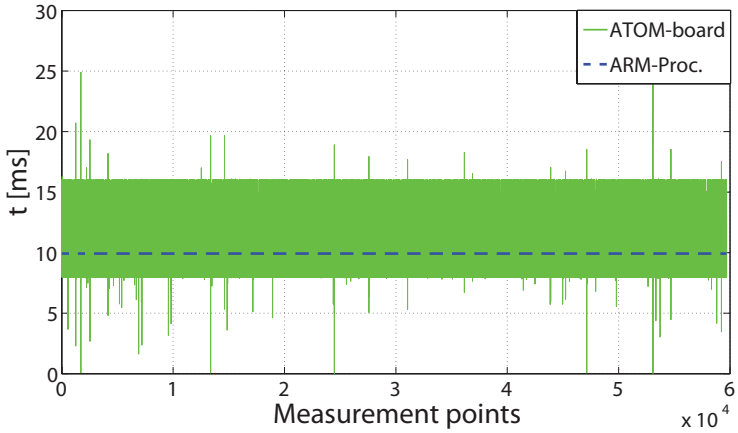

Fig. A.8: IMU data refreshing period.

for this transmission on the ARM-processor side illustrated by the dashed line is about 9.96 ms, while the period for the receiving on the ATOM-board illustrated by the solid line is not constant.

The time label of the kth data package sent by the ARM-processor is denoted by $T_{\mathrm{arm},k}$, while $T_{\mathrm{atom},k}$ is the corresponding time label for receiving of this package on the ATOM-board. A low-pass filter is used to determine the time base $T_{\mathrm{atom},0}$:

$$T_{\mathrm{atom},0}{}^{k} = T_{\mathrm{atom},0}{}^{k-1} \cdot \frac{T_{\mathrm{lp}} - 1}{T_{\mathrm{lp}}} + [T_{\mathrm{atom},k} - (T_{\mathrm{arm},k} - T_{\mathrm{arm},0})] \cdot \frac{1}{T_{\mathrm{lp}}}, \tag{A.2}$$

where T_{lp} is time constant for the low-pass filter and set to 500. The $T_{\mathrm{atom},0}$ will be calculated iteratively for every new data package and tends to be constant after several seconds. This time base is used to measure the average transfer time of the data package from the ARM-processor, since real-time measurement of the single transfer time is impossible.

The IMU data and the estimated Φ_{imu}, Θ_{imu} angles are almost synchronous. Therefore, only the transfer time of the IMU data has to be determined. An experiment is designed to measure this average value.

As shown in Fig. A.9, the quadrotor was fixed on a linear axis. The motor moves only along the axis. The X_b-axis of the quadrotor was also set parallel to this track. A high-

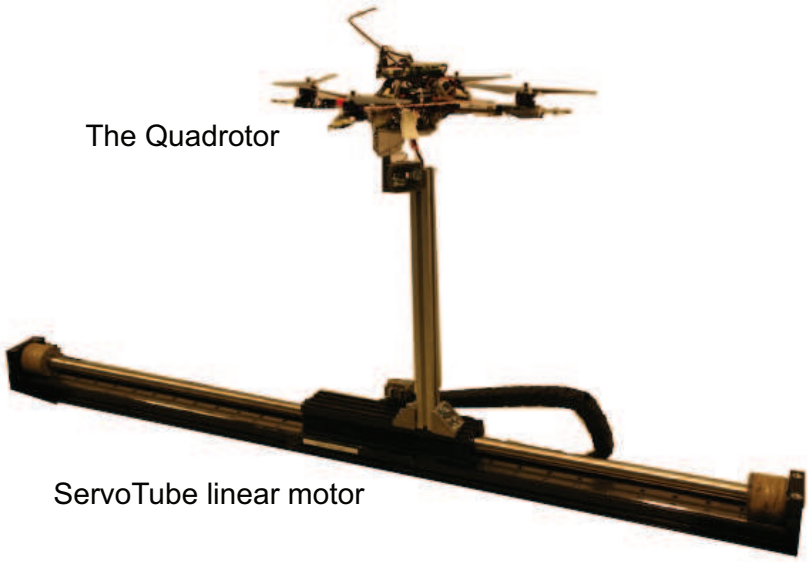

Fig. A.9: Experiment for measuring the IMU data transfer time.

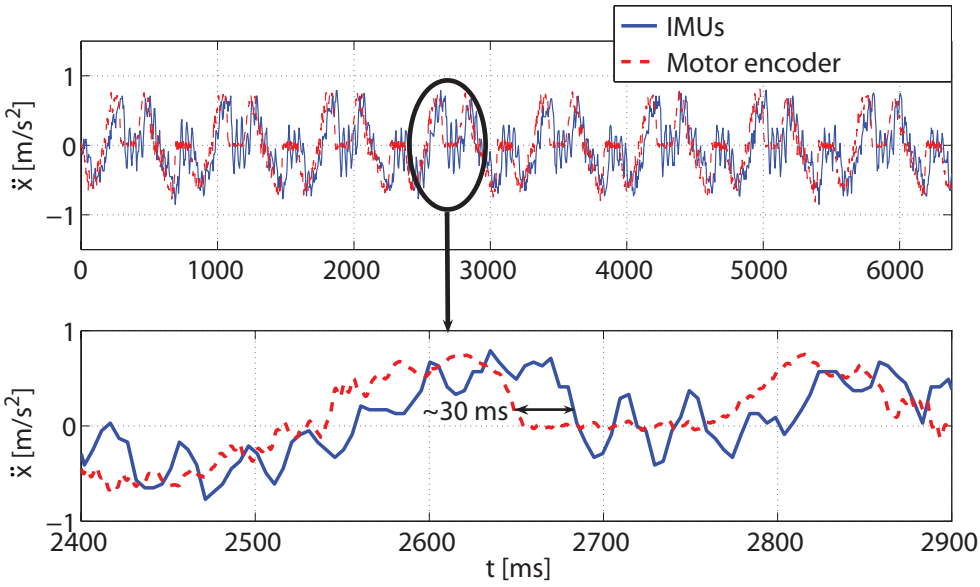

Fig. A.10: Accelerations measured by IMUs on the quadrotor and by the motor encoder of the linear axis.

speed data acquisition card was used for position measurement, in which the data transfer time is less than 1 ms and negligible. The acceleration of the quadrotor along the X_b-axis can be obtained by considering the second derivative of the position. The accelerations measured by the IMUs on the quadrotor and by the motor encoder of the linear axis are saved to the hard disk and analyzed off-line.

In Fig. A.10, the IMUs and the motor encoder of the linear axis give almost the same measurements. They are merely shifted differently on the time axis and the IMU measurement bears some oscillations, which are caused by vibrations of the quadrotor with respect to the stand bar. By zoomed in a time interval, the time difference between the

two measurements can be read directly. The average value of the data transfer time from IMUs to the ATOM-board is approximately 30 ms.

A.2.2 Time Delay of the Vision Data

A time stamp, which indicates the arriving time of the current image, is available in the data structure of the camera. For data synchronization, only the transfer time of the vision data delivered to the ATOM-board should be determined.

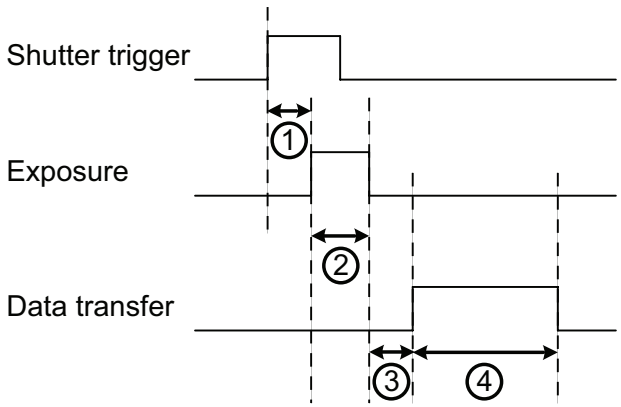

Fig. A.11: Transfer time analysis of vision data. 1) shutter open time of less than 10 μs; 2) the exposure time of 2 ms; 3) a preparation time for vision data transfer of 1 ms; 4) time for data transfer from the camera to the ATOM-board of 16.67 ms.

Crucial information about the vision data transfer time is provided in [103]. In Fig. A.11 it is illustrated, how the time delay of the vision data can be calculated. The shutter open time is determined by the hardware used in the camera and less than 10 μs. The physical exposure time is between 0.06 ms and 33.19 ms. With the driver for USB cameras, it can be set to an integer value on the interval $[1, 621]$. For a robust detection of the markers, this value is set to 30 and means an exposure time of 2 ms. A series of operations, e.g. A/D conversion, pixel correction, white balancing, buffering, is carried out before data transfer and takes about 1 ms. Note that the time of image data transfer is influenced by the frame rate of the camera. The 16.67 ms here refers to a frame rate at 60 Hz. The total time delay is approximately 20 ms.

A.2.3 Total Time Delay

The program for communication takes 5 ms to deliver IMU data and the estimated Φ_{imu}, Θ_{imu} angles to other running processes. Then, data from the ARM-processor are available for these processes after 30-35 ms.

The image processing of marker detection takes about 60 ms, and runs for every 60 ms. Although the frame rate of the camera is 60 Hz, the buffering time of the image on PC varies. The average transfer time of the vision data from shutter opening to synchronization with estimated Φ_{imu}, Θ_{imu} angles is approximately 80 ms and given by repeated real-time experiments. The synchronized data are all delayed by about 80 ms.

The process for the marker-based pose estimation takes less then 10 ms. However, because of the synchronization between multi-threads, the preliminarily estimated pose data for data fusion have a time delay of 90 ms. The most recently received IMU data are synchronized with this estimation and fed into the EKF. The input data for the EKF are all delayed by about 40 ms due to the time delay of IMU data.

B Pioneer P3-DX Robot

In this air-ground multi-robot system, a Pioneer P3-DX mobile robot from Mobile Robots Inc. [10] is used to operate on the floor (see Fig. B.1). This robot platform is 44×38×22 cm in dimensions with two drive wheels with a diameter of 16.5 cm each. The maximum speed is 1.6 m/s. It is equipped with different sensors such as Laser, sonars, and a camera etc. The control of the robot is achieved by programming an embedded computer using *Player/Stage* developed by an international team of robotics researchers [14]. More information can be found in [10].

Fig. B.1: The Pioneer P3-DX mobile robot with two active markers.

In order to realize a robust detection of the ground robot from the quadrotor's perspective, two active markers are mounted on the robot. The center point of these markers is referred as the reference position of the ground robot. Moreover, since the deck of the pioneer robot is too small for the quadrotor to stand on it, a 80×80 cm planar board with texture is mounted on the robot as the platform for take-off and landing.

The operating commands are transmitted to this mobile ground robot via wireless LAN using the *Transmission Control Protocol* (TCP) and *Internet Protocol* (IP) to start/stop the movement of the ground robot. To complete the data fusion and improve the control performance of the flying robot, motion information of the ground robot is uninterruptedly sent to the quadrotor employing the *User Datagram Protocol* (UDP) due to the real-time character.

C Tracking System

In this thesis, a tracking system *Visualeyez*™ *II VZ4000* from Phoenix Technologies Incorporated [11] is used, providing the ground truth of the quadrotor position and orientation.

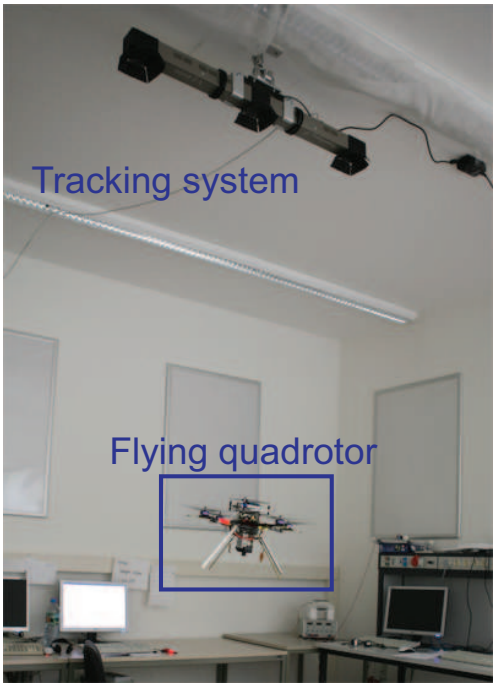

Fig. C.1: Experimental environment.

C.1 System Overview

The experimental environment is shown in Fig. C.1: The quadrotor with is flying in the field of view of the tracking system. Fig. C.2 shows the communication and data transfer of the total setup, which consists of the VZ4000 tracker, four *Light Emitting Diode* (LED) markers mounted on the quadrotor, a Windows PC with corresponding software *VZSoft 2.80*, and a Linux PC optionally, if the quadrotor should be controlled based on the position/orientation ground truth (see Section 4.4.2).

The VZ4000 tracker consists of three cameras with parallel optical axes and a baseline of 0.5 m between each two neighboring cameras. It can measure position of 1-24 markers in the 3D workspace at a resolution of 0.015 mm from a distance of 1.2 m with an accuracy of 0.5-0.7 mm RMS. In this thesis, the VZ4000 is mounted on the ceiling of the laboratory as shown in Fig. C.1 and illustrated in Fig. C.3. The cameras are directed facing downwards. The distance between the cameras and the ground is approximately 2.3 m. Since the

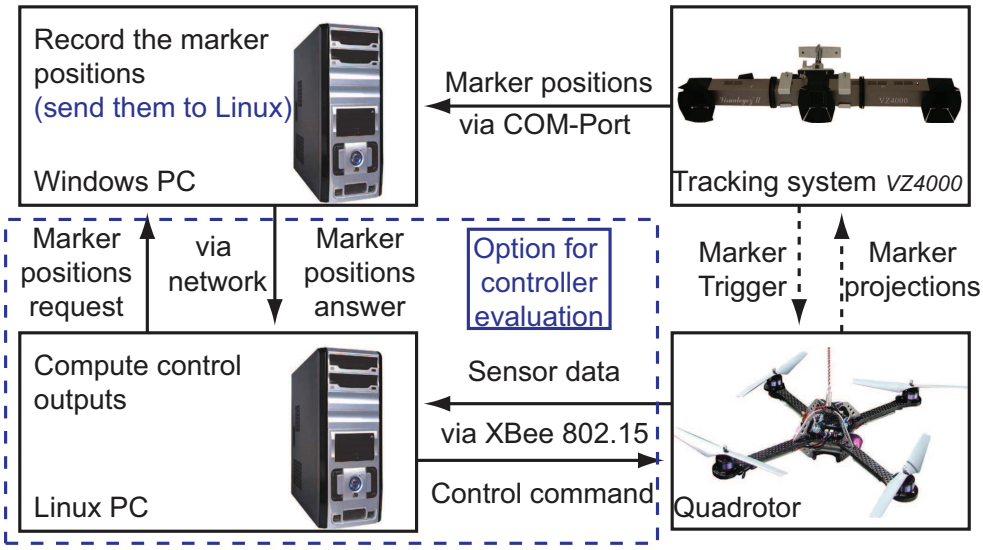

Fig. C.2: System overview.

cameras have a field of view of 90 deg each, the markers should not be higher than 1.8 m, such that the markers can be observed by at least two cameras. Otherwise, the markers can not be reliably tracked by the system.

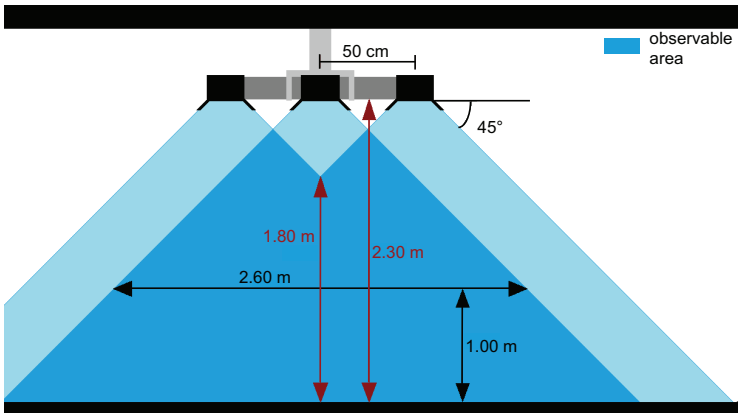

Fig. C.3: The observable area of the VZ4000.

The markers are mounted on each axis of the quadrotor and placed as far from the rotors as possible such that the markers cannot be totally covered by the rotating blades (see Fig. A.1). The markers are numbered in a clockwise direction, while the first marker is placed on the forward axis marked by a red label. Moreover, an additional circuit board is installed on the quadrotor for power supply as well as data transmitting and receiving with the tracking system. More information can be found at [11].

Since the software *VZSoft 2.80* of the VZ4000 tracker is only compatible with Microsoft Windows system, a Windows PC (AMD Athlon XP 3000+, 2.1 GHz, 1 GB RAM) is applied. During flight, the tracking system triggers the markers to lighten one after another at a very high frequency. The current lightened marker is projected into the cameras of the tracker. The marker positions are then transferred subsequently to the windows PC

via COM-interface. The windows PC records the marker positions for off-line analysis.

If the quadrotor should be controlled based on the position information provided by the tracking system, a Linux PC (AMD Athlon 64×2 5200+, 2 GB RAM) is also applied. It sends request of marker positions to the window PC, which transmits the marker positions via network to it. The linux PC also receives the quadrotor sensor data and computes control command signals based on different control designs. The control commands are transmitted from the linux PC to the quadrotor via XBee 802.15. The computation time of control laws is under 1 ms on the linux PC. The total time delay of the control commands is approximately 70 ms.

C.2 Data Analysis

In this section, the marker detection and sensor noise investigation using this tracking system are briefly introduced.

C.2.1 Marker Detection

Since the markers are fixed on the quadrotor axes, one or more markers may be occluded by the propeller blades during flight. An experiment was conducted to analyze the detection rate of the markers, in which the quadrotor was fixed on the ground and the duration of validity of each marker in 30 s was recorded in a file. Different thrust forces were applied.

The percentage detection rate of each marker and the percentage detection rate of different number of markers with respect to various thrust commands are shown in Tab. C.1.

Thrust command [0, 255]	**100**	**120**	**160**
Marker 1 observed (in %)	97.398543	97.810219	96.982310
Marker 2 observed (in %)	95.109261	95.307612	95.733611
Marker 3 observed (in %)	96.982310	95.828989	97,606660
Marker 4 observed (in %)	94.901145	94.056309	95.005203
4 markers detected (in %)	85.119667	83.941606	85.848075
3 markers detected (in %)	14.151925	15.119916	13.631634
2 markers detected (in %)	0.728408	0.938478	0.520291
1 marker detected (in %)	0.000000	0.000000	0.000000
0 marker detected (in %)	0.000000	0.000000	0.000000

Tab. C.1: Marker detection rate at different thrust forces.

The position of the quadrotor is calculated as the center of the four markers. If at least three markers are detected by the VZ4000 tracker, the position of the other marker and then the pose of the quadrotor can be computed from the positions of the three observed markers. A detection rate of more than 99% for at least three markers is obtained. If fewer than three markers are detected, the current quadrotor pose is assumed to be the same as that at the last time point. More information about the position estimation of the center of four markers can be found in [131].

C.2.2 Noise Analysis

The noise of marker position measurement is experimentally evaluated. A marker was fixed on the ground with a Z_I-coordinate of 0 m. Its position was measured by the VZ4000 tracker at a frequency of 100 Hz. Fig. C.4 illustrates the position measurements in the Z_I-direction and their mean value, namely 1.2 mm. This systematic error is corrected in the system calibration procedure.

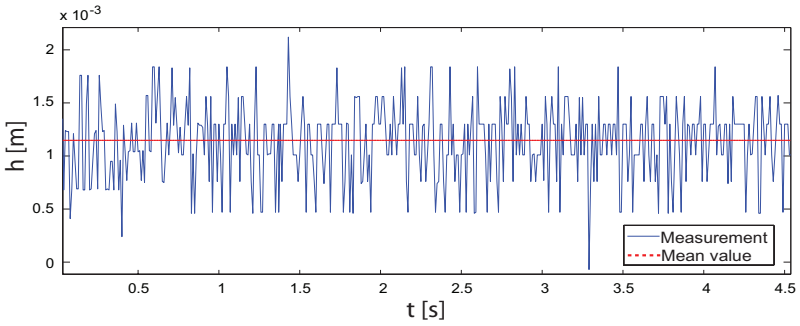

Fig. C.4: Position measurement of a static marker using the VZ4000 tracker.

C.3 Graphical User Interface

A *Graphical User Interface* (GUI) is developed for controller evaluation as shown in Fig. C.5, containing user inputs, a 3D display, a debug window, controller selection, and some outputs such as quadrotor pose and battery voltage.

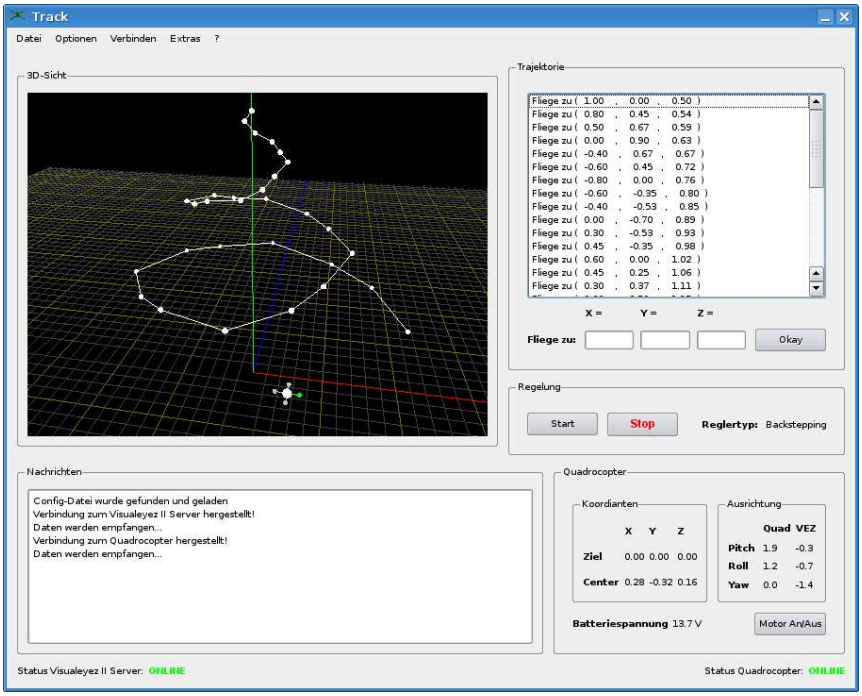

Fig. C.5: The graphical user interface.

Bibliography

[1] Altera Corporation. http://www.altera.com.

[2] AR. Drone, PARROT SA. http://ardrone.parrot.com.

[3] Ascending Technologies GmbH. http://www.asctec.de.

[4] CREASO GmbH. http://www.creaso.com.

[5] Draganfly Innovations Incorporated. http://www.draganfly.com.

[6] HiSystems GmbH. http://www.mikrokopter.de.

[7] Max Planck Institute of Neurobiology. http://www.neuro.mpg.de.

[8] Microdrones GmbH. http://www.microdrones.com.

[9] Mikrotron GmbH. http://www.mikrotron.de.

[10] Mobile Robots Incorporated. http://www.mobilerobots.com.

[11] PhoeniX Technologies Incorporated. http://www.ptiphoenix.com.

[12] Point Grey Research Incorporated. http://www.ptgrey.com.

[13] Stäubli Group. http://www.staubli.com.

[14] The Player Project. http://playerstage.sourceforge.net.

[15] VIDEOR E. Hartig GmbH. http://www.videor.com.

[16] *Lecture Notes: Regelungs- und Steuerungstechnik 1.* Lehrstuhl für Steuerungs- und Regelungstechnik, Technische Universität München, 2009.

[17] *Lecture Notes: Regelungs- und Steuerungstechnik 2.* Lehrstuhl für Steuerungs- und Regelungstechnik, Technische Universität München, 2009.

[18] P. Abbeel, A. Coates, M. Montemerlo, A. Y. Ng, and S. Thrun. Discrimitive training of kalman filters. In *Proceedings of Robotics: Science and System*, page 38, Cambridge, USA, 2005.

[19] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy. Autonomous navigation and exploration of a quadrotor helicopter in gps-denied indoor environments. In *Proceedings of First Symposium on Indoor Flight Issues*, 2009.

[20] E. Altug, J. P. Ostrowski, and R. Mahony. Control of a quadrotor helicopter using visual feedback. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2002)*, volume 1, pages 72–77, 2002.

[21] E. Altug, Ostrowski J. P., and Taylor C. J. Control of a quadrotor helicopter using dual camera visual feedback. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2003)*, volume 3, pages 4294–4299, 2005.

[22] E. Altug and C. Taylor. Vision-based pose estimation and control of a model helicopter. In *Proceedings of the IEEE International Conference on Mechatronics (ICM 2004)*, pages 316–321, 2004.

[23] O. Amidi, T. Kanade, and K. Fujita. A visual odometer for autonomous helicopter flight. *Journal of Robotics and Autonomous Systems*, 28:185–193, 1999.

[24] A. Anderson, D. Bittle, D. Dean, R. Flowers, G. Hester, and J. Hodel. Ekf and ukf state estimation comparison for rotating rockets. In *Proceedings of IEEE Southeastcon*, pages 373–378, 2009.

[25] G. Angeletti, J. R. Pereira Valente, L. Iocchi, and D. Nardi. Autonomous indoor hovering with a quadrotor. In *Workshop Proceedings of International Conference on Simulation, Modeling and Programming for Autonomous Robots*, 2008.

[26] F. Aubepart, M. El Farji, and N. Franceschini. FPGA implementation of elementary motion detectors for the visual guidance of micro-air-vehicles. In *Proceedings of the IEEE International Symposium on Industrial Electronics*, volume 1, pages 71–76, 2004.

[27] A. Bachrach, A. Winter, R. He, G. Hermann, S. Prentice, and N. Roy. Range – robust autonomous navigation in gps-denied environment. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2010), Anchorage, Alaska, USA*, pages 1096–1097, 2010.

[28] R. W. Beard. Quadrotor dynamics and control. Technical report, Brigham Young University, 2008.

[29] B. Bethke, M. Valenti, and J. How. Cooperative vision based estimation and tracking using multiple UAVs. In *Proceedings of the 7th International Conference on Cooperative Control and Optimization*, volume 369, pages 179–189, 2007.

[30] M. Bisgaard, A. la Cour-Harbo, and J. D. Bentsen. Full state estimation for helicopter slung load system. In *Proceedings of AIAA Guidance Navigation and Control Conference*, 2007.

[31] L. Bodizs, B. Srinivasan, and D. Bonvin. Preferential estimation via tuning of the kalman filter. In *Proceedings of the 8th IFAC International Symposium on Dynamics and Control of Process Systems (DYCOPS 2007)*, 2004.

[32] A. Borst and M. Egelhaaf. Principles of visual motion detection. *Trends in Neurosciences*, 12:297–306, 1989.

[33] A. Borst and J. Haag. Neural networks in the cockpit of the fly. *Journal of Comparative Physiology A*, 188:419–437, 2002.

[34] A. Borst, C. Reisenman, and J. Haag. Adaptation of response transients in fly motion vision. ii. model studies. *Vision Research*, 43:1309–1322, 2003.

[35] S. Bouabdallah. *Design and control of quadrotors with application to autonomous flying*. PhD thesis, Lausanne, EPFL, 2007.

[36] S. Bouabdallah and R. Siegwart. Full control of a quadrotor. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, pages 153–158, San Diego, CA, 2007.

[37] H. Bouadi, M. Bouchoucha, and M. Tadjine. Sliding mode control based on backstepping approach for an UAV type-quadrotor. In *World Academy of Science, Engineering and Technology 26*, pages 22–27, 2007.

[38] J. Y. Bouguet. Pyramidal implementation of the lucas-kanade feature tracker. Technical report, Intel Corporation, 1999.

[39] O. Bourquardez, R. Mahony, N. Guenard, F. Chaumette, T. Hamel, and L. Eck. Image-based visual servo control of the translation kinematics of a quadrotor aerial vehicle. *IEEE Transactions on Robotics*, 25(3):743–749, 2009.

[40] R. S. Brinkworth and D. C. O'Carroll. Robust model for optic flow coding in natural scenes inspired by insect biology. *PLoS Computational Biology*, 5(11):1–14, 2009.

[41] D. Burschka et al. DLR/TUM Flying Robot Project. http://flyingrobots.visual-navigation.com.

[42] P. Castillo, A. Dzul, and R. Lozano. Real-time stabilization and tracking of a four-rotor mini rotorcraft. *IEEE Transactions on Control Systems Technology*, 12(4):510–516, 2004.

[43] L. Chaimowicz, B. Grocholsky, J. F. Keller, R. V. Kumar, and C. J. Taylor. Experiments in multirobot air-ground coordination. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2004)*, volume 4, pages 4053–4058, 2004.

[44] L. Chaimowicz and V. Kumar. *Distributed Autonomous Robotics Systems 6*, chapter Aerial Shepherds: Coordination among UAVs and Swarms of Robots, pages 243–252. Springer Japan, 2007.

[45] J. Conroy, G. Gremillion, B. Ranganathan, and J. S. Humbert. Implementation of wide-field integration of optic flow for autonomous quadrotor navigation. *Autonomous Robot*, 27:189–198, 2009.

[46] P. Corke. An inertial and visual sensing system for a small autonomous helicopter. *Journal of Robotic Systems*, 21:43–51, 2004.

[47] P. Corke, J. Lobo, and J. Dias. An introduction to inertial and visual sensing. *The International Journal of Robotics Research*, 26:219–535, 2007.

[48] J. Courbon, Y. Mezouar, N. Guenard, and P. Martinet. Visual navigation of a quadrotor aerial vehicle. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009)*, pages 5315–5320, 2009.

[49] H. Cuntz, J. Haag, F. Foerstner, I. Segev, and A. Borst. Robust coding of flow-field parameters by axo-axonal gap junctions between fly visual interneurons. *Proceedings of the National Academy of Science of the United States of America*, 104(24):10229–10233, 2007.

[50] R. O. Dror, D. C. O'Carroll, and S. B. Laughlin. Accuracy of velocity estimation by reichardt correlators. *Journal of the Optical Society of America A (Optics, Image Science and Vision)*, 18:231–252, 2001.

[51] R. Dudley. *The biomechanics of insect flight*. Princeton University Press, 2000.

[52] M. G. Earl, Dapos, and R. Andrea. Real-time attitude estimation techniques applied to a four rotor helicopter. In *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC 2004)*, volume 4, pages 3956–3961, 2004.

[53] M. Ö. Efe. Robust low altitude behavior control of a quadrotor rotorcraft through sliding modes. In *Proceedings of the 15th Mediterranean Conference on Control and Automation*, pages 1–6, 2007.

[54] M. Egelhaaf and A. Borst. A look into the cockpit of the fly: Visual orientation, algorithms, and identified neurons. *The Journal of Neuroscience*, 13:4563–4574, 1993.

[55] A. Elfes, M. Bergerman, J. R. H. Carvalho, E. C. Paiva, J. J. G. Ramos, and S. S. Bueno. Air-ground robotic ensembles for cooperative applications: Concepts and preliminary results. In *Proceedings of International Conference on Field and Service Robotics*, pages 75–80, 1999.

[56] J. Escareno, S. Salazar-Cruz, and R. Lozano. Embedded control of a four-rotor UAV. In *Proceedings of the 2006 American Control Conference (ACC 2006)*, 2006.

[57] M. Euston, P. Coote, R. Mahony, J. Kim, and T. Hamel. A complementary filter for attitude estimation of a fixed-wing UAV. In *Proceedings of International Conference on Robots and Systems (IROS 2008)*, pages 340–345, 2008.

[58] F. Faille, A. Borst, and G. Faerber. Biologically inspired motion detector for video interpretation. received by Biological Cybernetics on April 12, 2007.

[59] J. Grauer, J. Conroy, J. Hubbard Jr., J. Humbert, and D. Pines. System identification of a miniature helicopter. *Journal of Aircraft*, 46(4):1260–1269, 2009.

[60] S. Grzonka, G. Grisetti, and W. Burgard. Autonomous indoors navigation using a small-size quadrotor. In *Workshop Proceedings of International Conference on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR 2008)*, pages 455–463, 2008.

[61] N. Guenard, T. Hamel, and R. Mahony. A practical visual servo control for a unmanned aerial vehicle. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA 2007)*, pages 1342–1348, 2007.

[62] N. Guenard, T. Hamel, and R. Mahony. A practical visual servo control for an unmanned aerial vehicle. *IEEE Transactions on Robotics*, 24(2):331–340, 2008.

[63] D. Gurdan, J. Stumpf, M. Achtelik, K.-M. Doth, G. Hirzinger, and D. Rus. Energy-efficient autonomous four-rotor flying robot controlled at 1 khz. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2007)*, pages 361–366, Roma, Italy, 2007.

[64] J. Haag and A. Borst. Neural mechanism underlying complex receptive field properties of motion-sensitive interneurons. *Nature Neuroscience*, 7:628–634, 2004.

[65] S. Han, A. D. Straw, M. H. Dickinson, and R. M. Murray. A real-time helicopter testbed for insect-inspired visual flight control. In *Proceedings of International Conference on Robotics and Automation (ICRA 2009)*, pages 3055–3060, 2009.

[66] R. R. Harrison. A biologically inspired analog ic for visual collision detection. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 52:2308–2318, 2005.

[67] B. Hassenstein and W. Reichardt. Systemtheoretische analyse der zeit-, reihenfolgen- und vorzeichenauswertung bei der bewegungsperzeption des rüsselkäfers chlorophanus. *Naturforsch*, 11b:513–524, 1956.

[68] R. He, A. Bachrach, and N. Roy. Efficient planning under uncertainty for a target-tracking micro-aerial vehicle. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2010), Anchorage, Alaska, USA*, 2010.

[69] R. He, S. Prentice, and N. Roy. Planning in information space for a quadrotor helicopter in a gps-denied environment. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2008)*, pages 1814–1820, 2008.

[70] B. Herisse, T. Hamel, R. Mahony, and F. X. Russotto. A nonlinear terrain-following controller for a vtol unmanned aerial vehicle using translational optical flow. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2009)*, pages 3251–3257, 2009.

[71] C. M. Higgins and S. A. Shams. A biologically inspired modular vlsi system for visual measurement of self-motion. *IEEE Sensors Journal*, 2:508–528, 2002.

[72] G. Hoffmann, D. G. Rajnarayan, S. L. Waslander, D. Dostal, J. S. Jang, and C. J. Tomlin. The stanford testbed of autonomous rotorcraft for multi agent control (starmac). In *Proceedings of the 23rd Digital Avionics Systems Conference*, volume 2, pages 12.E.4–121–10, 2004.

[73] G. M. Hoffmann, H. Huang, S. L. Wasl, and E. C. J. Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2007.

[74] M. A. Hsieh, A. Cowley, J. F. Keller, L. Chaimowicz, B. Grocholsky, V. Kumar, C. J. Taylor, Y. Endo, R. C. Arkin, B. Jung, D. F. Wolf, G. S. Sukhatme, and D. C. MacKenzie. Adaptive teams of autonomous aerial and ground robots for situational awareness. *International Journal of Field Robotics*, 24(11-12):991–1014, 2007.

[75] S. Hutchinson, G. Hager, and P. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, 1996.

[76] F. Iida. Biologically inspired visual odometer for navigation of a flying robot. *Robotics and Autonomous Systems*, 44:201–208, 2003.

[77] M. T. Keennon and J. M. Grasmeyer. Development of the black widow and microbat MAVs and a vision of the future of MAV design. In *Proceedings of AIAA/ICAS International Air and Space Symposium and Exposition: The Next 100 Years*, 2003.

[78] J. Kelly, S. Saripalli, and G. S. Sukhatme. *Field and Service Robotics*, chapter Combined Visual and Inertial Navigation for an Unmanned Aerial Vehicle, pages 255–264. Springer Berlin/Heidelberg, 2008.

[79] C. Kemp. *Visual Control of a Miniature Quad-Rotor Helicopter*. PhD thesis, Churchill College, University of Cambridge, 2006.

[80] F. Kendoul, I. Fantoni, and K. Nonami. Optic flow-based vision system for autonomous 3d localization and control of small aerial vehicles. *Robotics and Autonomous Systems*, 57(6-7):591–602, 2009.

[81] B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller. Multiple relative pose graphs for robust cooperative mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2010), Anchorage, Alaska, USA*, pages 3185–3192, 2010.

[82] T. Köhler, F. Röchter, J. P. Lindemann, and R. Möller. Bio-inspired motion detection in an fpga-based smart camera module. *Bioinspiration and Biomimetics*, 4(1), 2009.

[83] P. V. Kokotovic. The joy of feedback: Nonlinear and adaptive. *Control Systems Magazine*, 12:7–17, 1992.

[84] H. G. Krapp and R. Hengstenberg. Estimation of self-motion by optic flow processing in single visual interneurons. *Nature*, 384:463–466, 1996.

[85] I. Kroo, F. Prinz, M. Shantz, P. Kunz, G. Fay, S. Cheng, T. Fabian, and C. Partridge. The mesicopter: A miniature rotorcraft concept, phase ii final report. Technical report, Stanford University, 2001.

[86] H. Kuhlmann. Kalman-filtering with coloured measurement noise for deformation analysis. In *Proceedings of the 11th FIG Symposium on Deformation Measurements*, 2003.

[87] L. Lai, C. Yang, and C. Wu. Time-optimal control of a hovering quad-rotor helicopter. *Journal of Intelligent and Robotics Systems*, 45:115–135, 2006.

[88] S. Liu. A neuromorphic a VLSI model of global motion processing in the fly. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 47:1458–1467, 2000.

[89] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, 1981.

[90] T. Madani and A. Benallegue. Control of a quadrotor mini-helicopter via full state backstepping technique. In *Proceedings of the 45th IEEE Conference on Decision and Control (CDC 2006)*, pages 1515–1520, 2006.

[91] A. Matsue, W. Hirosue, H. Tokutake, S. Sundada, and A. Ohkura. Navigation of small and lightweight helicopter. *Transactions on Japan Society Aeronautical and Space Sciences*, 48:177–179, 2005.

[92] L. Mejias, S. Saripalli, P. Campoy, and G. S. Sukhatme. Visual servoing of an autonomous helicopter in urban areas using feature tracking. *Journal of Field Robotics*, 23(3-4):185–199, 2006.

[93] A. A. Mian and D. Wang. Nonlinear flight control strategy for an underactuated quadrotor aerial robot. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control (ICNSC 2008)*, pages 938–942, 2008.

[94] L. D. Minh and C. Ha. Station-keeping of quadrotor MAV using vision based measurement in hover. In *Proceedings of the 2009 International Forum on Strategic Technologies (IFOST 2009)*, pages 112–117, 2009.

[95] H. B. Mitchell. *Multi-sensor data fusion: An Introduction.* Springer, 2007.

[96] A. A. Mkrtchyan, R. R. Schultz, and W. H. Semke. Vision-based autopilot implementation using a quadrotor helicopter. In *Proceedings of the Aerospace Conference (AIAA 2009)*, 2009.

[97] E. Nakamura, S. Asami, T. Takahashi, and K. Sawada. Real time parameter optimization for elementary motion detectors. In *Proceedings of the IEEE International Conference on Image Processing*, pages 1065–1068, 2006.

[98] E. Nakamura, M. Ichimura, and K. Sawada. Fast global motion estimation algorithm based on elementary motion detectors. *IEEE International Conference on Image Processing*, 2:297–300, 2002.

[99] T. R. Neumann and H. Bulthoff. Insect inspired visual control of translatory flight. *The Advances in Artificial Life.*, 2159:627–636, 2001.

[100] M. Park and Y. Gao. Error analysis and stochastic modeling of low-cost mems accelerometer. *Journal of Intelligent and Robotic Systems*, 46:27–41, 2006.

[101] L. E. Parker. Distributed intelligence: Overview of the field and its application in multi-robot systems. *Journal of Physical Agents, special issue on multi-robot systems*, 2(1):5–14, 2008.

[102] PC Quadrat GmbH, www.x3d-shop.de. *X-3D-BL: Assembly Manual.*

[103] Point Grey Research Incorporated. *Point grey firefly mv technical reference.*

[104] P. Pounds, R. Mahony, and P. Corke. Modelling and control of a quad-rotor robot. In *Proceedings of the Australasian Conference on Robotics and Automation (ACRA 2006)*, 2006.

[105] P. Pounds, R. Mahony, P. Hynes, and J. Roberts. Design of a four-rotor aerial robot. In *Proceedings of the Australasian Conference on Robotics and Automation (ACRA 2002)*, 2002.

[106] J. C. Raimundez and A. F. Villaverde. Adaptive tracking control for a quad-rotor. In *Proceedings of the 6th EUROMECH Nonlinear Dynamics Conference (ENOC 2008)*, 2008.

[107] H. Rehbinder and B. K. Ghosh. Multi-rate fusion of visual and inertial data. In *Proceedings of International Conference on Multisensor Fusio and Integration for Intelligent Systems (MFI)*, pages 97–102, 2001.

[108] W. Reichardt and M. Egelhaaf. Properties of individual movement detectors as derived from behavioural experiments on the visual system of the fly. *Biological Cybernetics*, 58:287–294, 1988.

[109] J. F. Roberts, T. S. Stirling, J. C. Zufferey, and D. Floreano. Quadrotor using minimal sensing for autonomous indoor flight. In *Proceedings of European Micro Air Vehicle Conference and Flight Competition (EMAV 2007)*, 2007.

[110] S. I. Roumeliotis, A. E. Johnson, and J. F. Montgomery. Augmenting inertial navigation with image-based motion estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2002)*, volume 4, pages 4326–4333, 2002.

[111] F. Ruffier, S. Viollet, S. Amic, and N. Franceschini. Bio-inspired optical flow circuits for the visual guidance of micro-air vehicles. In *Proceeding of the IEEE International Symposium on Circuits and Systems*, volume 3, pages 846–849, 2003.

[112] S. Saripalli and G. S. Sukhatme. Landing a helicopter on a moving target. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2007)*, pages 2030–2035, 2007.

[113] SBS Technologies Incorporated. *Wave FPGA Tool Kit Documentation, Tsunami PCI Technical Data Sheet.*

[114] V. Shin, D. Y. Kim, G. Shevlyakov, and K. Kim. A low-complexity suboptimal filter for continuous-discrete linear systems with parametric uncertainties. *Signal Processing*, 87(10):2392–2406, 2007.

[115] A. Soumelidis, P. Gaspar, G. Regula, and B. Lantos. Control of an experimental mini quad-rotor UAV. In *Proceedings of the16th Mediterranean Conference on Control and Automation*, 2008.

[116] M. V. Srinivasan and S. Zhang. Visual motor computation in insects. *Annual Review of Neuroscience*, 27:679–696, 2004.

[117] T. Stentz, A. Kelly, H. Herman, P. Rander, O. Amidi, and R. Mandelbaum. Integrated air/ground vehicles system for semi-autonomous off-road navigation. In *Proceedings of AUVSI Unmanned Systems Symposium*, 2002.

[118] M. J. Stepaniak. *A quadrotor sensor platform.* PhD thesis, Russ College of Engineering and Technology of Ohio University, 2008.

[119] E. Stingu and F. L. Lewis. A hardware platform for research in helicopter UAV control. *Journal of Intelligent and Robotic Systems*, 54 (1-3):387–406, 2009.

[120] G. S. Sukhatme, J. F. Montgomery, and R. T. Vaughan. *Robot teams: From diversity to polymorphism*, chapter Experiments with cooperative aerial-ground robots, pages 345–367. AK Peters, Ltd., 2002.

[121] H. Surmann, D. Holz, S. Blumenthal, T. Linder, P. Molitor, and V. Tretyakov. Teleoperated visual inspection and surveillance with unmanned ground and aerial vehicles. *International Journal of Online Engineering*, 4(4):26–38, 2008.

[122] Y. Tan, J. Chang, J. He, and H. Tan. Advanced nonlinear control strategy for motion control systems. In *Proceedings of the 3rd International Conference on Power Electronics and Motion Control (IPEMC 2000)*, 2000.

[123] G. P. Tournier, M. Valenti, and J. P. How. Estimation and control of a quadrotor vehicle using monocular vision and moire patterns. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2006.

[124] E. A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communication, and Control Symposium*, 2000.

[125] S. Waslander, G. M. Hoffmann, J. S. Jang, and C. J. Tomlin. Multi-agent quadrotor testbed control design: Integral sliding mode vs. reinforcement learning. In *Proceedings of International Conference on Intelligent Robots and Systems (IROS 2005)*, 2005.

[126] A. B. Watson and A. J. Ahumada. Model of human visual-motion sensing. *Journal of the Optical Society of America A*, 2(2):322–342, 1985.

[127] G. Welch and G. Bishop. An introduction to the kalman filter. Technical report, Department of Computer Science, University of North Carolina at Chapel Hill, 2006.

[128] K. W. Weng and M. S. Z. Abidin. Design and control of a quad-rotor flying robot for aerial surveillance. In *Proceedings of the 4th Student Conference on Research and Development (SCOReD 2006)*, 2006.

[129] A. D. Wu, E. N. Johnson, and A. A. Proctor. Vision-aided inertial navigation for flight control. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005.

[130] J. C. Zufferey. *Bio-inspired Vision-based Flying Robots*. PhD thesis, Ecole Polytechnique Federale de Lausanne (EPFL), 2005.

Own Publications

[131] M. Achtelik, **T. Zhang**, K. Kühnlenz, and M. Buss. Visual tracking and control of a quadcopter using a stereo camera system and inertial sensors. In *Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA 2009)*, pages 2863–2869, 2009. **Toshio Fukuda Best Award in Mechatronics**.

[132] A. Bauer, K. Klasing, T. Xu, S. Sosnowski, G. Lidoris, Q. Mühlbauer, **T. Zhang**, F. Rohrmüller, D. Wollherr, K. Kühnlenz, and M. Buss. Video: The autonomous city explorer project. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2009)*, pages 1595–1596, Kobe, Japan, 2009. **Finalist for Best Video Award**.

[133] K. Kühnlenz, H. Wu, **T. Zhang**, A. Borst, and M. Buss. FPGA design and implementation of insect-inspired reichardt motion detector and receptive field. *Chinese Journal of Image and Graphics*, 14 (12):2489–2496, 2009.

[134] Q. Mühlbauer, S. Sosnowski, T. Xu, **T. Zhang**, K. Kühnlenz, and M. Buss. The autonomous city explorer project: Towards navigation by interaction and visual perception. In *Proceedings of the 1st International Workshop on Cognition for Technical Systems*, number 49, Munich, Germany, 2008.

[135] Q. Mühlbauer, S. Sosnowski, T. Xu, **T. Zhang**, K. Kühnlenz, and M. Buss. Navigation through urban environments by visual perception and interaction. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2009)*, pages 3558–3564, Kobe, Japan, 2009.

[136] H. Wu, **T. Zhang**, A. Borst, K. Kühnlenz, and M. Buss. An explorative study of applying reichardt-model in visual servo control. In *Proceedings of the 1st International Workshop on Cognition for Technical Systems*, number 41, Munich, Germany, 2008.

[137] H. Wu, **T. Zhang**, A. Borst, K. Kühnlenz, and M. Buss. An explorative study of visual servo control with insect-inspired reichardt-model. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2009)*, pages 345–350, Kobe, Japan, 2009.

[138] T. Xu, H. Wu, **T. Zhang**, K. Kühnlenz, and M. Buss. Environment adapted active multi-focal vision system for object detection. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2009)*, pages 2418–2423, Kobe, Japan, 2009.

[139] T. Xu, **T. Zhang**, K. Kühnlenz, and M. Buss. *Computer Vision*, chapter Towards High-Speed Vision for Attention and Navigation of Autonomous City Explorer, pages 189–214. IN-TECH, 2008.

[140] T. Xu, **T. Zhang**, K. Kühnlenz, and M. Buss. Attentional object detection with an active multi-focal vision system. *International Journal of Humanoid Robotics*, 7(2):223–243, 2010.

[141] **T. Zhang**. Insect-inspired visuomotor control for an autonomous mini-quadrotor flying system. Technical report, Institute of Automatic Control Engineering (LSR), Technische Universität München, 2007.

[142] **T. Zhang**. Multi-sensory ego-motion estimation for an autonomous mini-quadrotor in 3d-space. Technical report, Institute of Automatic Control Engineering (LSR), Technische Universität München, 2008.

[143] **T. Zhang**, Y. Kang, M. Achtelik, K. Kühnlenz, and M. Buss. Autonomous hovering of a vision/IMU guided quadrotor. In *Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA 2009)*, pages 2870–2875, 2009.

[144] **T. Zhang**, W. Li, M. Achtelik, K. Kühnlenz, and M. Buss. Multi-sensory motion-estimation and control of a mini-quadrotor in an air-ground multi-robot system. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO 2009)*, pages 45–50, 2009. **Finalist for Best Student Paper Award**.

[145] **T. Zhang**, W. Li, K. Kühnlenz, and M. Buss. Multi-sensory motion estimation and control of an autonomous quadrotor. *IEEE/ASME Transactions on Mechatronics*, 2010, submitted.

[146] **T. Zhang**, X. Liu, K. Kühnlenz, and M. Buss. Visual odometry for the autonomous city explorer. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009)*, pages 3513–3518, St. Louis, MO, USA, 2009.

[147] **T. Zhang**, H. Wu, A. Borst, K. Kühnlenz, and M. Buss. An FPGA implementation of insect-inspired motion detector for high-speed vision systems. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2008)*, pages 335–340, Pasadena, CA, USA, 2008.