ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Autonomous Systems Lab
Prof. Roland Siegwart

Master Thesis

Development of a planar low cost Inertial Measurement Unit for UAVs and MAVs

Spring Term 2008

Supervised by:
André Noth
Samir Bouabdallah
Janosch Nikolic

Author:
Samuel Fux

Contents

Abstract

This thesis deals with the problem of developing an attitude determination algorithm for a planar inertial measurement unit with low cost sensors and a microprocessor with limited processing power. Four different complementary filters and two different extended Kalman filters are presented. Their performance is evaluated by experiments.
A quaternion based complementary filter and a gravity direction cosine and earth magnetic field direction cosine estimators complementary filter are selected to be implemented on the microprocessor. For both filters, a computational time decrease by approximating trigonometric functions with lookup tables is proposed. The computational time of the gravity direction cosine and earth magnetic field direction cosine estimators complementary filter is further reduced by simplifying the integration step.
Final tests show that the quaternion based complementary filter is most suitable for an inertial measurement unit with low cost sensors and a microprocessor with limited processing power.

List of Figures

List of Tables

Symbols

Symbols

A, B, H	continuous time system matrices
$A^k_{(.)}$	exact integration by Rodrigue's formula
A_m	3-axis accelerometer measurement
A_{mx}, A_{my}, A_{mz}	accelerometer measurement components
C^n_b	direction cosine matrix
$\hat{C}^n_b$	estimated direction cosine matrix
C^b_n	rotation matrix from navigation frame to body frame
F_k, G_k, H_k	discrete time system matrices
H	earth magnetic field vector
H_m	3-axis magnetometer measurement
H_{mx}, H_{my}, H_{mz}	magnetometer measurement components
$I_{x\times x}$	$x \times x$ identity matrix
K_k	Kalman gain
P_k	a posteriori estimate error covariance matrix
P^-_k	a priori estimate error covariance matrix
Q_k	process noise covariance matrix
$Q\left(q\right)$	quaternion multiplication matrix
R_k	measurement noise covariance matrix
T	sampling period
T_ω	angular rate transformation matrix
$\hat{b}$	gyroscope's bias estimation
$b\left(t\right)$	gyroscope bias
c_{ij}	direction cosine matrix entries
g	gravitational acceleration
$h\left(t\right)$	continuous measurement equation
$h\left(x_k\right)$	discrete time measurement equation
i, j, k	imaginary parts of quaternion
$k_{(.)}$	filter parameters
m	earth magnetic field direct cosine
q	quaternion
$\hat{q}$	estimated unit quaternion representing attitude
q_0, q_1, q_2, q_3	quaternion vector components

$u\left(t\right)$	system input vector
$\bar{u}$	nominal systen input vector
v	system noise
w	measurement noise
x	gravity direction cosine vector in body frame
$\hat{x}$	estimated gravity direction cosine vector in body frame
$\bar{x}$	nominal system state vector
x_b, y_b, z_b	orthogonal vector basis of body frame
x_k	discrete time system state vector
$\hat{x}_k$	discrete time estimated system state vector
x_k^-	ahead projected system state vector
$x\left(t\right)$	continuous time system state vector
x_0, y_0, z_0	orthogonal vector basis of navigation frame
x_1, x_2, $\ldots$	components of the state vector
y_k	discrete time measurement vector
z	earth magnetic field direction cosine vector in body frame
$\hat{z}$	estimated earth magnetic field direction cosine vector in body frame
z_k	measurement
Ω	true body angular rates
Ω_m	3-axis gyroscope measurement
Γ	filter parameter
$\delta u\left(t\right)$	variation of the system input vector around the operating point
$\delta x\left(t\right)$	variation of the system state vector around the operating point
$\delta y\left(t\right)$	measurement error vector
δu_k	discrete time variation of the system input vector around the operating point
δx_k	discrete time variation of the system state vector around the operating point
δy_k	discrete time measurement error vector
δw, δv	system and measurement noise vectors, including linearization errors
ϕ, θ, ψ	roll, pitch and yaw angle
ν, μ	white noise vectors
$\omega_{(.)}$	innovation term
ω_x, ω_y, ω_z	angulare rate components
$0_{x\times x}$	$x \times x$ zero matrix

Indices

k	time step
r	rotated
n	navigation frame
b	body frame

Acronyms and Abbreviations

ASL	Autonomous Systems Lab
ETH	Eidgenössische Technische Hochschule
EKF	Extended Kalman Filter
GUI	Graphical User Interphase
IMU	Inertial Measurement Unit
KF	Kalman Filter
LUT	Lookup Table
MEMS	Micro-Electro-Mechanical Systems
MIPS	Mega instructions per second
RMS	Root Mean Square
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter

Chapter 1

Introduction

1.1 Motivation

Unmanned aerial vehicles (UAVs) and micro aerial vehicles (MAVs) will play a more and more important role in the future daily live, e.g., for inspection purposes or rescue missions. An prerequisite for successfull autonomous flight is thereby the determination and the control of the UAV's or MAV's attitude. Since UAVs and MAV's have limited size and payload, attitude determination systems should be small and lightweight. Another requirement is of course the demand for low cost.
Former successfull attitude determination systems required sensors wich have exceptional long-term bias stability and are of large size and heavy weight [43]. Such systems are widely used in military applications, such as in submarins, and their prizes easily range up to several 100'000 Dollars. Of course, such systems are not at all well suited for UAVs and MAVs.
An alternative to this expensive sensors is provided by Micro-Electro-Mechanical Systems (MEMS) sensors, but these sensors have shortcomings, such as noise and drift, which has to be corrected by software.
There already exist several manufacturers providing attitude determination systems based on MEMS. Table 1.1 gives an overview of systems available on the market.
At the *Eidgenössische Technische Hochschule* (ETH) in Zurich currently both an UAV and a MAV are developed. The UAV project, the *Sky-Sailor*, is a fully autonomous airplane equipped with solar cells to provide the energy for the propulsion system and onboard electronics. The superfluous energy from the solar cells is used to charge a battery, which delivers the energy during night until the next morning when a new cycle starts [27].
The MAV project, the *muFly*, deals with the development and implementation of a fully autonomous micro helicopter comparable in size and weight to a small bird [3].
Of course, attitude determination systems are not only used in UAVs and MAVs. Also in other domains, attitude determination systems are needed, such as in inspection robots.
As in the UAV and the MAV project the size and the weight of the employed components play a dominant role, it would be preferable to develop still smaller and more lightweight attitude determination systems than the ones available on the market.
Attitude determination systems normally consist of gyroscopes, accelerometers and magnetometers, whereat a device, containing gyroscopes and accelerometers, is commonly called Inertial Measurement Unit (IMU).
If the gyroscopes would provide perfect measurements of the UAV's turn motions, then simple integration of the gyroscope's signal would give the attitude. But since MEMS gyroscopes suffer substantially from noise and drift, other sensors like ac-

celerometers and magnetometers are needed to correct this imperfectness.
So to get the best estimate of the UAV's or MAV's attitude, a sensor fusion algorithm as shown in figure 1.1 is needed to combine the measurements from the different sensors.

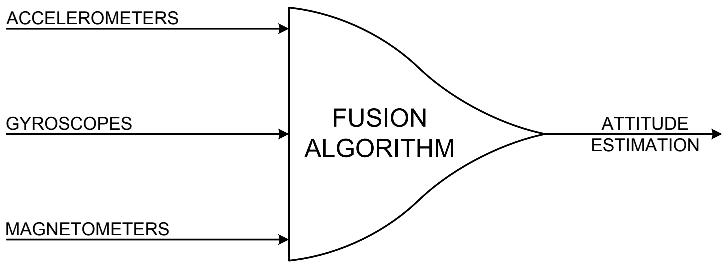

Figure 1.1: Fusion of the different sensor readings in order to get the best attitude estimation.

1.2 Goals

The objective of the ASL IMU project is to develop a planar low cost, small and lightweight IMU, which can be employed in very small UAVs and MAVs. Thereby, the focus is on the size and weight of the IMU, i.e., that compromises are consciously made on the cost of the accuracy of the ASL IMU compared with the accuracy of IMUs available on the market.
The goal of this thesis is the development of an attitude determination algorithm for this ASL IMU, i.e., the hardware design is not part of this thesis, but it is described in [4].
During the algorithm development two different hardware as shown in figure 1.2 are used:

- sensor board
- final IMU

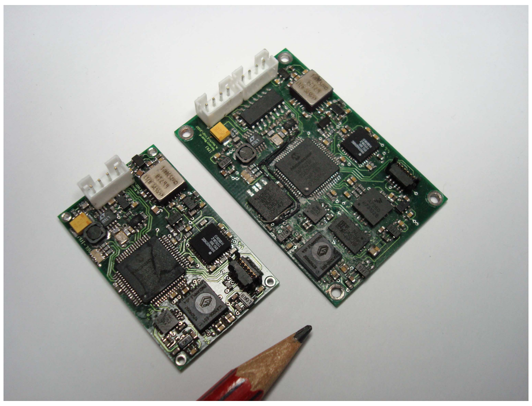

Figure 1.2: Left: final IMU, right: sensor board.

Thereby, the sensor board is a development tool wich contains redundant sensors from different manufacturers. In a first phase, the sensor board has to be interfaced, so that different attitude determination algorithms and different sensors can

Typ	Manu-facturer	Mass	Power Consumption	Size	Range	Accuracy
3DM-GX2	MicroStrain	16g[a]	810mW[c]	32 × 36 × 24mm[a]	360° all axes	±0.5° typical for static test conditions ± 2.0° typical for dynamic (cyclic) test conditions & for arbitrary orientation angles
3DM	MicroStrain	26.3g[a]	600mW[c]	43.18 × 58.42 × 7.62mm[a] or 37.08 × 139.7 × 16mm[a]	Roll: ±180° Pitch: ±180° Yaw: ±70°	Roll: ±0.7° typical (yaw from 0° - 360° & pitch = 0°) Pitch: ±0.7° typical (yaw from 0° - 360° & roll = 0°) Yaw: 1.5° typical (pitch & roll = 0°)
3DM-DH	MicroStrain	10.8g[a]	600mW[c]	28 × 67 × 8mm[a]	Roll: ±180° Pitch: ±180° Yaw: ±70°	Roll: ±0.93° typical (yaw from 0° - 360° & pitch = 0°) Pitch: ±0.33° typical (yaw from 0° - 360° & roll = 0°) Yaw: 1.0° typical (pitch & roll = 0°)
InertiaCube2	InterSense	25g[b]	600mW	28.89 × 24.38 × 33.31mm[b]	360° all axes	1° RMS at 25 °C
InertiaCube3	InterSense	17g[b]	240mW	26.2 × 39.2 × 14.8mm[b]	360° all axes	Roll/Pitch: 0.25° RMS Yaw: 1° RMS
MTi	Xsens	50g[b]	360mW	58 × 58 × 22mm[b]	360° all axes	Static accuracy Roll/Pitch <0.5° Yaw <1° Dynamic accuracy 2° RMS
MTx	Xsens	30g[b]	360mW	38 × 53 × 21mm[b]	360° all axes	Static accuracy Roll/Pitch <0.5° Yaw <1° Dynamic accuracy 2° RMS

Table 1.1: Attitude tracking systems available on the market

[a]Without housing
[b]With housing
[c]Maximum value

be tested in order to select the most appropriate sensors and the best algorithm.
In the second phase, based on the selected sensors, the final IMU has to be designed and tested together with the selected attitude determination algorithm.
The thesis mainly deals with the algorithm to be implemented on a low cost microprocessor with limited processing power.

1.3 Structure of the Thesis

The thesis is organized as follows: Chapter 2 presents a literature analysis of how other authors solved the problem of attitude determination. A theoretical background on attitude representation is given in chapter 3. In chapter 4, the sensor board used for testing different attitude determination algorithms is introduced, whereat the different sensors on the sensor board are calibrated with the procedure explained in chapter 5. Chapter 6 focuses on different attitude determination algorithms to be considered for implementation on the final IMU. The quality of the different attitude determination algorithms is checked in chapter 7. Chapter 8 shows how the computational time of two selected filters can be decreased.
The final IMU is then presented and tested in chapters 9 and 10 respectively.

Chapter 2

Literature Analysis

First, a literature analysis was accomplished to see how other authors solved the problem of attitude determination using inertial sensors. By doing so it filters out that two different approaches for sensor fusion are widely used in practice:

- Kalman filter
- Complementary filter

Following the result of the literature anylsis on these two approaches, including their advantages and disadvantages.

2.1 Kalman Filter

This kind of filter was firstly introduced by R. E. Kalman in 1960. Taking into account the knowledge of the system and the measuring device, the description of the system noise and measurement errors and the uncertainty in the dynamics models the Kalman filter (KF) produces an optimal estimate of the system state. Thus, the sensor signals and the system knowledge are fused by the Kalman filter in an optimal way [37].
Solving the attitude determination problem by applying Kalman filtering there is considerable freedom in the choice of the state vector, the measurement vector and the matrices used to describe the system [9]. This is uncovered by studying the literature, as there are many different approaches with Kalman filter.
As mentioned in [9], one gets most accurate estimation when the equations of motion of the body being tracked, are included in the system dynamics model, which was done for example by Koifman and Bar-Itzhack [17].
Niculescu [26] also uses a dynamic model of the air vehicle in order to implement different types of Kalman filters (a linear Kalman filter, a Kalman filter with nonlinear state propagation and one with gain scheduling, and an Unscented Kalman filter are implemented), whose performance is analyzed. The reported performance of the Kalman filter with nonlinear state propagation is good, but it must be emphasized that in the simulations the measurement data is obtained from the same dynamic model as used in the Kalman filter, instead from the real system.
Other authors however consciously avoid dynamic modelling, because 1) possibly it is difficult to obtain a good model [36], 2) it increases the computational burden, 3) it do not always yield the expected results and 4) the dynamic model must be adapted to any modification made to the system [15].
Marins et al [21] introduce an approach where a Gauss-Newton iteration algorithm is used to find the best quaternion that relates the gravitational acceleration and earth magnetic field measurements in the body coordinate frame to the values in

the earth coordinate frame. This best quaternion is then used as a part of the measurements for the Kalman filter, which results in linear measurement equations. Sensor drifts are not treated.
Rhebinder and Hu [32] present a linear Kalman filter whose state is the third column of the rotation matrix, hence only pitch and roll angle are observed. But they face the problem that the accelerometers are bad attitude sensors when the body is accelerated with a switching architecture consisting of two modes, one for low accelerations and one for high.
A similar approach is shown in the paper of Wang et al [43]. As the attitude is fully determined by the first and the third column of the transformation matrix from navigation frame to body frame, the state vector of their Kalman filter is composed of the entries of this columns. Since the norm of the columns of the direction cosine matrix must be equal to one, they derive from these constraints so-called pseudo-measurements and include them in the measurement equation of the Kalman filter.
A special kind of using Kalman filter for attitude determination is presented by Zhu et al [49]. Instead of the commonly used quaternion components or Euler angles, the Kalman state vector is a 6×1 vector containing sensing components of the earth gravity and magnetic field in the body frame. The measurements of the earth magnetic field and gravity are then regarded as the output of the Kalman filter, so that the measurement equation is a linear function.
A very simple Kalman filter is proposed by Vaganay et al [41] as even the kinematics of Euler angle integration are not modeled [9].
Concluding one can say that there is a huge volume of literature considering the solution of the attitude determination problem using Kalman filter. Most of the papers present approaches with more or less the same core idea: integrate the gyro measurements either using the Euler angels or quaternions and then utilize the accelerometer and magnetometer measurements to correct.

2.2 Complementary Filter

Complementary filtering is used to combine state and state derivative information, taking the varying validity of sensor information with respect to frequency into account. So the long term accuracy of the accelerometers and magnetometers can be fused with the short term precision provided by gyroscopes to get an attitude estimation [44, 2, 20].
The authors in [2] and [33] describe easy to implement complementary filters, which must be implemented for each rotational axis. The gyroscope bias is not treated explicitly.
In [20] the authors state, that it is difficult to apply robustly both the traditional linear Kalman filter techniques and the Extended Kalman filter (EKF) techniques to applications with low quality sensor systems. Additionally it is claimed that the non-Gaussian noise encountered in practice and the inherent non-linearity of the system can lead to very poor behavior of such filters.
So, a direct complementary filter and a passive complementary filter are presented whose output is the orientation matrix and not only the rotation angle of one single axis. Thereby the authors claim the passive filter to have several practical advantages over the direct filter associated with implementation and low-sensitivity to noise. In addition, the passive complementary filter is extended with gyroscope bias estimation.
In [13] an alteration to the passive filter in [20] is provided which avoids the possible introduction of poor numerical conditioning and amplification of noise into the filter. It is also proved that a complimentary filter is assymptotically stable, if at least two inertial measurements (here the gravitational and magnetic vector fields

respectively) are available in addition to full measurement of the angular rates of the system.
Metni et al. [24] introduce in their paper two observers which estimate the gravity direction cosine and the magnetic field direction cosine in the body fixed frame. The complete rotation matrix estimation can then be derived from this two estimated vectors. Gyroscope bias estimation is also provided in order to ensure the convergence of the observers. The authors state their proposed estimator to be very appropriate in the special case of autonomous hovering systems, cabable of vertical take-off, landing and quasi-stationary (hover and near hover) flight conditions.
The paper by Pflimlin et al. [29] belongs to the same series of papers as the three mentioned before, as it describes a variaton of the already introduced complementary filters. The proposed observer is claimed to be as efficient as the usual Extended Kalman filter but easier to implement in real time.
In [1] a complementary quaternion-based attitude determination filter is presented. Gyroscope bias is also taken into account by tracking the time-varying bias with a low pass filter with a very long time constant.

2.3 Conclusion

In [35] a low-cost IMU is described for which first a complementary filter was used in order to determine the attitude of a small-size helicopter. It is reported that this filter gave reasonable results, but it does not deal with the gyroscope bias well, and so an Extended Kalman filter was implemented.
Since this is the only found paper including a comparison of complementary filter with Kalman filter, and since from the other papers it is hard to decide which algorithm is the best for implementation on an IMU with low cost sensors, it would be best to implement different filters and to compare.
So in chapter 6 different attitude determination algorithms, both complementary filters and Kalman filters, are presented and compared in chapter 7.

Chapter 3

Theoretical Background

3.1 Coordinate Frames

The goal of an attitude determination algorithm is to determine the orientation of an UAV with respect to a reference frame. So two orthogonal coordinate frames are considered:

Navigation Frame The navigation frame N is represented by the orthogonal vector basis $\{x_0, y_0, z_0\}$ and is attached to the earth. Thereby x_0 points north, y_0 points east and z_0 points toward the center of earth as shown in figure 3.1.

Body Frame The body frame B is fixed to the UAV and is represented by the vector basis $\{x_b, y_b, z_b\}$. Thereby the roll axis x_b points forward, the pitch axis y_b points right and the yaw axis z_b points down as shown in figure 3.2 [29].

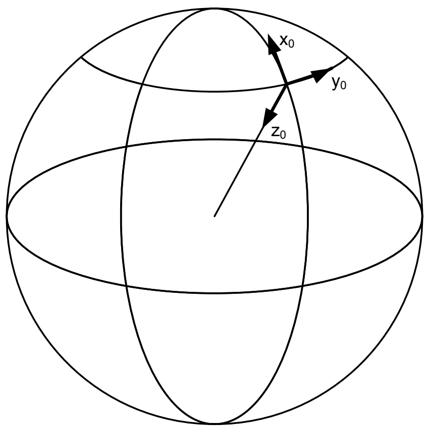

Figure 3.1: Navigation frame.

3.2 Attitude Representation

In the literature there exist different ways of representing the attitude between two coordinate frames and to transform vectors and coordinates from one coordinate frame to another one.
The following subsections give an overview of the approaches used in this thesis.

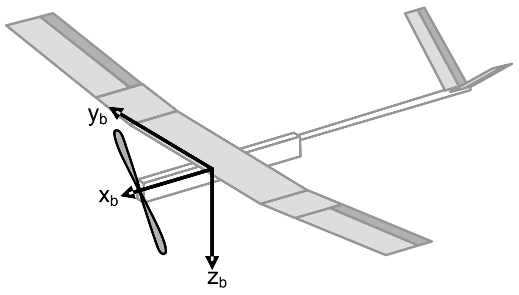

Figure 3.2: Body frame [28].

3.2.1 Rotation Matrix

The rotation matrix is used to transform vectors and coordinates from one coordinate frame to another and vice versa.
The transformation from navigation frame to body frame is based on three consecutive rotations:

1. Rotation Rotation around the gravity vector by ψ (yaw angle)

2. Rotation Rotation around the new (rotated) y-axis by θ (pitch angle)

3. Rotation Rotation around the new x-axis by ϕ (roll angle)

ϕ, θ and ψ are called *Euler angles* or *Tait-Bryan angles.* Of course other sequencies would be also possible.
The rotation matrix from navigation frame to body frame for the above mentioned rotation sequence is determined by

$$C_n^b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

The direction cosine matrix is defined as

$$\begin{aligned} C_b^n &= \left(C_n^b\right)^T = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \\ &= \begin{bmatrix} c_\theta c_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & -s_\phi c_\psi + c_\phi s_\theta s_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix} \end{aligned} \tag{3.2}$$

where the cosine and the sine of the Euler angles are abbreviated because of space reasonings by $c_{(.)}$ and $s_{(.)}$, respectively.
The Euler angles in terms of the direction cosine matrix components are

$$\begin{aligned} \phi &= \operatorname{arctan2}(c_{32},\ c_{33}) \\ \theta &= -\arcsin(c_{31}) \\ \psi &= \operatorname{arctan2}(c_{21},\ c_{11}) \end{aligned} \tag{3.3}$$

The relation between the time derivatives of the Euler angles and the body turn rates $\Omega = (\omega_x \ \omega_y \ \omega_z)^T$ is

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{bmatrix}}_{T_\omega^{-1}} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{3.4}$$

Hence, the propagation of the Euler angles with time is given by

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & \frac{\sin\phi\sin\theta}{\cos\theta} & \frac{\cos\phi\sin\theta}{\cos\theta} \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix}}_{T_\omega} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{3.5}$$

A disadvantage of this representation is the occurrence of a singularity at $\theta = \pm 90\,°$, where the roll and the yaw axis coincide. The determination of the Euler angles is then not unambiguous any more. This is also referred to as *Gimbal lock* [6, 7, 40].

3.2.2 Quaternions

Another way of representing attitude are quaternions. Quaternions have been already devised by William Rowan Hamilton in 1843.
A quaternion is defined as the sum

$$q = q_0 + q_1 i + q_2 j + q_3 k \tag{3.6}$$

and can also be represented as a vector

$$q = [q_0,\ q_1\ q_2,\ q_3]^T = \begin{bmatrix} q_0 \\ q_{1:3} \end{bmatrix} \tag{3.7}$$

The adjoint, norm and inverse of a quaternion are defined as

$$\bar{q} = \begin{bmatrix} q_0 \\ -q_{1:3} \end{bmatrix} \tag{3.8}$$

$$|q| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} \tag{3.9}$$

$$q^{-1} = \frac{\bar{q}}{|q|} \tag{3.10}$$

The multiplication of two quaternions q and p is computed element-wise using the relations in table 3.1.

·	i	j	k
i	-1	k	-j
j	-k	-1	i
k	j	-i	-1

Table 3.1: Mixed terms of the imaginary parts

But it can be also done by pre-multiplication of the second quaternion p with a matrix composed of the components of the first quaternion q, i.e.,

$$q \cdot p = Q(q)\ p \tag{3.11}$$

where

$$Q(q) = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \tag{3.12}$$

A quaternion with unity norm is referred to as unit quaternion and can be used to represent the attitude of a rigid body. Thereby, the unit quaternion

$$q = \cos(\alpha/2) + v_1 \cdot \sin(\alpha/2)\, i + v_2 \cdot \sin(\alpha/2)\, j + v_1 \cdot \sin(\alpha/2)\, k \tag{3.13}$$

represents a rotation around the vector $v = (v_1\ v_2\ v_3)^T$ by the angle α.
The rotation of a point with the position vector r around the rotation axis v is computed by

$$\mathbf{p}(r_r) = q \cdot \mathbf{p}(r) \cdot \bar{q} \tag{3.14}$$

where r_r is the position vector of the rotated point and $\mathbf{p}(r)$ is the quaternion representation of a vector, i.e.,

$$\mathbf{p}(r) = \begin{bmatrix} 0 \\ r \end{bmatrix} \tag{3.15}$$

The direction cosine matrix in terms of the components of a unit quaternion is

$$\begin{aligned} C_b^n &= \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_0q_2 + q_1q_3) \\ 2(q_0q_3 + q_1q_2) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_0q_1 + q_2q_3) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \\ &= \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1q_2 - q_0q_3) & 2(q_0q_2 + q_1q_3) \\ 2(q_0q_3 + q_1q_2) & 1 - 2(q_1^2 + q_3^2) & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_0q_1 + q_2q_3) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix} \end{aligned} \tag{3.16}$$

The mapping from Euler angles to the corresponding unit quaternion is

$$\begin{aligned} q_0 &= c_{\phi/2}\, c_{\theta/2}\, c_{\psi/2} + s_{\phi/2}\, s_{\theta/2}\, s_{\psi/2} \\ q_1 &= -c_{\phi/2}\, s_{\theta/2}\, s_{\psi/2} + c_{\theta/2}\, c_{\psi/2}\, s_{\phi/2} \\ q_2 &= c_{\phi/2}\, c_{\psi/2}\, s_{\theta/2} + s_{\phi/2}\, c_{\theta/2}\, s_{\psi/2} \\ q_3 &= c_{\phi/2}\, c_{\theta/2}\, s_{\psi/2} - s_{\phi/2}\, c_{\psi/2}\, s_{\theta/2} \end{aligned} \tag{3.17}$$

and vice versa

$$\begin{aligned} \phi &= \operatorname{arctan2}(c_{32},\, c_{33}) = \operatorname{arctan2}\left(2q_2q_3 + 2q_0q_1,\ 1 - 2\left(q_1^2 + q_2^2\right)\right) \\ \theta &= -\arcsin(c_{31}) = -\arcsin(2q_1q_3 - 2q_0q_2) \\ \psi &= \operatorname{arctan2}(c_{21},\, c_{11}) = \operatorname{arctan2}\left(2q_1q_2 + 2q_0q_3,\ 1 - 2\left(q_2^2 + q_3^2\right)\right) \end{aligned} \tag{3.18}$$

Finally, the relation between the progation of the unit quaternion with time and the body turn rates is given by [6, 7, 18, 40]

$$\begin{aligned} \dot{q} &= \frac{1}{2}\, q \cdot \mathbf{p}(\Omega) \\ &= \frac{1}{2} \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \\ &= \frac{1}{2} \begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \end{aligned} \tag{3.19}$$

3.3 Extended Kalman Filter: A Review

The Kalman filter has been published for the first time by Rudolph E. Kalman in 1960 in his paper *A New Approach to Linear Filtering and Prediction Problems.* This filter is developed for linear systems where the process noise and the measurement noise can be modeled by white Gaussian noise. If this constraints are met, the Kalman filter is optimal in the sense that it minimizes the estimated error covariance P given by

$$P = E\left[e\, e^T\right] \tag{3.20}$$

where e is the error between the true state x and the state estimation $\hat{x}$.
But since not all systems are linear, an extension, the Extended Kalman Filter (EKF), has been introduced, where the nonlinear models are linearized, so that the linear Kalman filter can be applied [14, 39, 45]. This section shortly summarizes the set of mathematical equations used in the EKF [7, 22, 45].

For a nonlinear system, the process model is given by a set of nonlinear differential equations

$$\dot{x}\left(t\right) = f\left(x\left(t\right),\, u\left(t\right)\right) \tag{3.21}$$

where

x	system state vector,
u	system input vector.

The nonlinear measurement equation is

$$y\left(t\right) = h\left(x\left(t\right)\right) + v\left(t\right) \tag{3.22}$$

with the measurement noise $v\left(t\right)$.
The linearization around an operating point yields the following linearized system dynamics

$$\begin{aligned} \dot{\delta x}\left(t\right) &= \left.\frac{\delta f}{\delta x}\right|_{\bar{x},\bar{u}} \cdot \delta x\left(t\right) + \left.\frac{\delta f}{\delta u}\right|_{\bar{x},\bar{u}} \cdot \delta u\left(t\right) + \delta w\left(t\right) \\ \delta y\left(t\right) &= \left.\frac{\delta h}{\delta x}\right|_{\bar{x},\bar{u}} \cdot \delta x\left(t\right) + \delta v\left(t\right) \end{aligned} \tag{3.23}$$

where

$\bar{x}$, $\bar{u}$	actual operating point,
δx	variation of the system state vector around the operating point,
δu	variation of the system input vector around the operating point,
δy	measurement error vector,
$\delta w, \delta v$	system and measurement noise vectors, including linearization errors.

The linearized system dynamics are commonly written as

$$\begin{aligned} \dot{\delta x}\left(t\right) &= A\left(\bar{x},\bar{u}\right)\delta x\left(t\right) + B\left(\bar{x},\bar{u}\right)\delta u\left(t\right) + \delta w\left(t\right) \\ \delta y\left(t\right) &= H\left(\bar{x}\right)\delta x\left(t\right) + \delta v\left(t\right) \end{aligned} \tag{3.24}$$

Discretizing the linearized system dynamics results in

$$\begin{aligned} \delta x_{k+1} &= F_k \delta x_k + G_k \delta u_k + \delta w_k \\ \delta y_k &= H_k \delta x_k + \delta v_k \end{aligned} \tag{3.25}$$

with

$$\begin{aligned} F_k &= e^{A_k T} \\ H_k &= H \end{aligned} \tag{3.26}$$

where T is the sampling period.
The EKF equations for the time and the measurement update are:

1. EKF time update equations ("Predict")
   The state and the error covariance matrix are projected ahead by

$$\hat{x}_k^- = \hat{x}_{k-1} + \int_{t_{k-1}}^{t_k} f\left(x\left(t\right),\, u\left(t\right)\right) \cdot dt \tag{3.27}$$

$$P_k^- = F_k P_{k-1} F_k^T + Q_{k-1} \tag{3.28}$$

   where Q_k is the discrete-time system noise covariance matrix given by $Q_k = cov\left(\delta w\; \delta w^T\right)$.

2. EKF measurement update equations ("Correct")
   The Kalman gain is computed by

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k\right)^{-1} \tag{3.29}$$

   where R_k is the discrete-time measurement covariance matrix defined as $R_k = cov\left(\delta v\; \delta v^T\right)$.
   The state estimation is updated based on the new measurement z_k by

$$\hat{x}_k = \hat{x}_k^- + K_k\left(z_k - h\left(\hat{x}_k^-\right)\right) \tag{3.30}$$

   and the update of the error covariance matrix is

$$P_k = \left(I - K_k H_k\right) P_k^- \tag{3.31}$$

For practical implementation, the ahead projection of the state $\hat{x}_k^-$ and the discretization of the system matrix F_k is simplified as

$$\hat{x}_k^- \approx \hat{x}_{k-1} + f\left(\hat{x}_{k-1},\, u_{k-1}\right) \cdot T \tag{3.32}$$

$$F_k \approx I + A_k T \tag{3.33}$$

3.3.1 Kalman Filter Tuning

The measurement noise covariance matrix R can be measured in advance of the filter operation by determining the variance of the measurement noise of some off-line sample measurements. Determining the process noise covariance matrix Q is generally more difficult as there is usually no possibilty to observe the process which is estimated [46].
In order to avoid trial-and-error for tuning Q, in [16] the tuning problem is restated as a optimization problem of a cost function. As cost function is chosen the root mean square (RMS) of the state estimation error.
Prerequisites for the method are, that the measurement covariance matrix R and the true states are available.

3.4 Complementary Filter: An Intuitive Review

As a simple example consider the first order integrator

$$\dot{x} = u \tag{3.34}$$

Assume that mesurements of both x and u with the following characteristics are available

$$y_x = L(s)\,x + \mu_x, \quad y_u = u + \mu_u + b(t) \tag{3.35}$$

where $L(s)$ is a low pass filter, y_x and y_u are noise and $b(t)$ is a deterministic perturbation dominated by low-frequency content.
Both measurements y_x and y_u can be fused to get an estimate $\hat{x}$ of the true state x by

$$\hat{x} = F_1(s)\,y_x + F_2(s)\,\frac{y_u}{s} \tag{3.36}$$

whereat the transfer function F_1 is a low-pass filter and F_2 is a high-pass filter as shown in figure 3.3. If $F_1+F_2 = 1$, then the filter form (3.36) is called *complementary filter* [20].

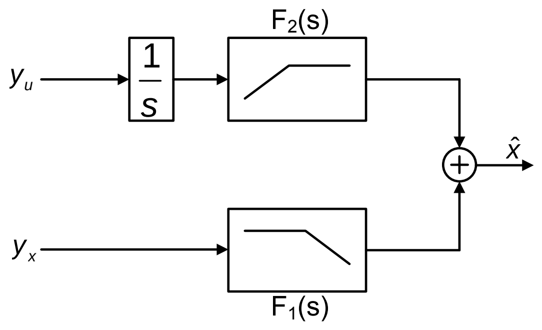

Figure 3.3: The complementary filter.

Chapter 4

Sensor Board

Before designing the final IMU, in a first step a so-called sensor board is developed. The reason for this is because the selection of the appropriate components is not an easy task. Thus, for some sensor types there are several possible sensors employed on the sensor board. So the most appropriate sensors for a low cost IMU can be determined by direct testing.
These in the first step selected sensors are then used to design in a second step the final IMU.

4.1 Components

When selecting adequate components for building up an IMU with the desired specifications, the following characteristics of the components must be taken into account:

- size
- weight
- cost
- power consumption
- sensor noise
- accuracy

The following subsections give an overview of the main components on the sensor board. By assembling these components on the sensor board, the sensor board has a weight of 8.37g and a size of 50×33mm.

4.1.1 Sensors

As there are several manufacturers of accelerometers and magnetometers fullfilling the above-mentioned requirements, the sensor board contains three 3-axis accelerometers and two 2-axis magnetometers. The reason why only 2-axis magnetometers are chosen, is that at the time of this thesis no appropriate 3-axis magnetometers are available, i.e., the available 3-axis magnetometers need too much place for external components.
Additionally the sensor board contains a 2-axis gyroscope for the roll and pitch axes and a 1-axis gyroscope for the yaw axis.
Tables 4.1, 4.2 and 4.3 give an overview of the assembled sensors and figure 4.1

shows the different sensors on the sensor board.
For more information on the selection procedure of the different sensors and the hardware development see [4].

Typ	Manufacturer	Size [mm]	Range	Interface
SCA3000-E01	VTI Technologies	7×7×1.8	±3g	digital[1]
MMA7260QT	Freescale	6×6×1.45	±1.5/±2.0/ ±4.0/±6.0g	analog
LIS3LV02DL	ST Microelectronic	7.5×4.4×0.92	±2/±6g	analog

Table 4.1: 3-axis accelerometers on the sensor board

Typ	Manufacturer	Size [mm]	Range	Axes	Inteface
ADXRS300	Analog Devices	7×7×3	±300 °/s	z axis	analog
IDG-300	InvenSense	6×6×1.5	±500 °/s	x & y axes	analog

Table 4.2: Gyroscopes on the sensor board

Typ	Manufacturer	Size [mm]	Range	Interface
HMC6052	Honeywell	3.5×3.5×0.8	±3 gauss	analog
HMC6352	Honeywell	6.5×6.5×1.5	0.10 - 0.75 gauss	digital[2]

Table 4.3: 2-axis magnetometers on the sensor board

4.1.2 Microprocessor

For processing the measurements from the different sensors, the sensor board contains the dsPIC33FJ256GP506 microprocessor from Microchip Technology Inc. Table 4.4 shows the microprocessor's specifications.
The microprocessor's speed is set to 20MIPS.

4.2 Sensor Board Specifications

Table 4.5 gives an overview of the specifications of the sensor board.

4.3 Coordinate Frames

Figure 4.2 shows the sensor board body frame and the coordinate axes of the different sensors employed on the sensor board. Since the coordinate axes of the sensors do not coincide with the body frame axes, the sensor coordinate axes must be transformed (see chapter 5).

[1]SPI
[2]I^2C

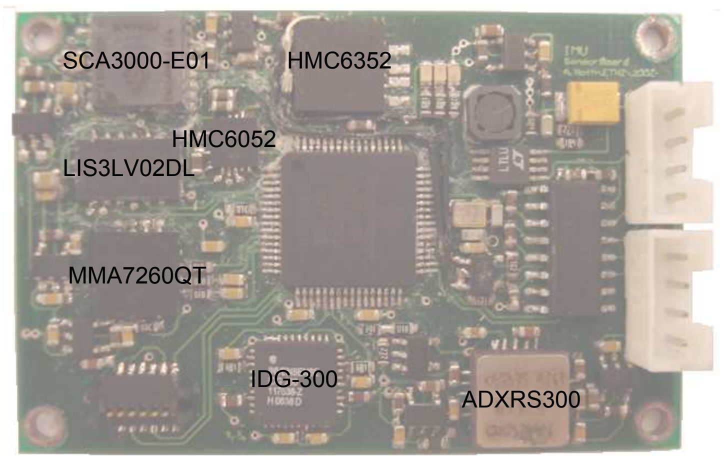

Figure 4.1: Sensor board with the different sensors.

Parameter Name	Value
V_{dd} Range	3 - 3.6V
UART	2
SPI	2
SRAM Bytes	16384
Program (FLASH) kBytes	256
Pin Count	64
ICSP	Yes

Table 4.4: Specifications of the dsPIC33FJ256GP506 microprocessor from Microchip Technology Inc.

Parameter Name	Value
Size	50×33mm
Weight	8.37g
V_{supply}	5.2 - 11V
Interface	RS-232
Baudrate	115200
Output data	Raw sensor data

Table 4.5: Sensor board specifications.

4.4 Communication Protocol

For the communication with the sensor board a standard message format is used. Figure 4.3 shows the fields contained in a message.

Preamble (1 Byte)

The preamble is at the beginning of every send message and has always the value 250 (0xFA).

Data (64 Bytes)

This field contains the data from the different sensors. Figure 4.4 shows the sequence of the different sensor values within the data field, whereas all values are 32 bit floats.

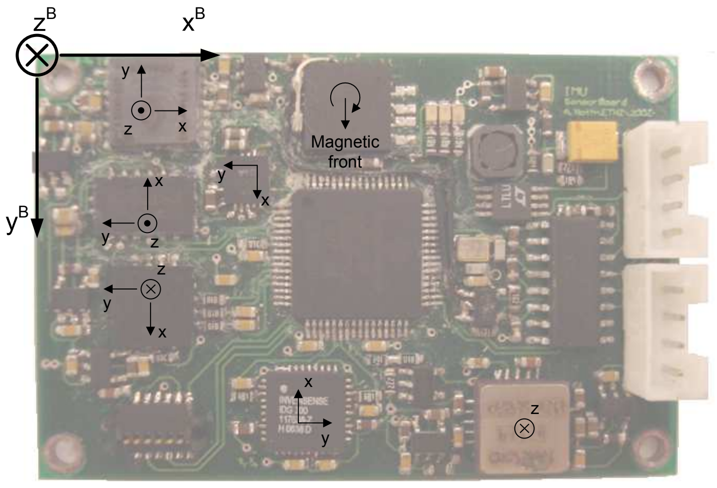

Figure 4.2: Sensor board body frame and coordinate axes of the different sensors.

Checksum (1 Byte)

The checksum is used to detect if the message arrived correctly. It is calculated by summing up all bytes of the message excluding the checksum byte, and then negating the sum. For the validation of a message the checksum must be recalculated and compared with the arrived checksum.

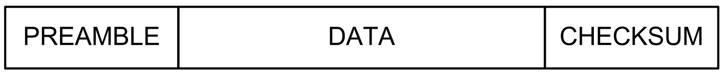

Figure 4.3: Message format of sensor board communication protocol.

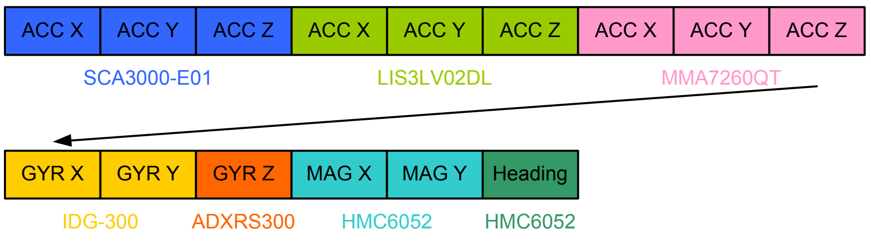

Figure 4.4: Sensor values in the data field.

Chapter 5

Sensor Calibration

The calibration of the different sensors employed on the sensor board is necessary because the sensitivity axes of the sensors do not coincide with the axes of the body frame and the sensitivity and the bias of each sensor may vary within the type series. The misalignment is due to misaligned soldering of the sensors on the sensor board and non-orthogonality of the sensitivity axes of each sensor.
Calibration is the process of adapating the parameters of the sensor error model presented in section 5.1 such that the ouput of the sensor error model agrees with the reference output [5].

5.1 Sensor Error Model

The transformation of the sensor output from the sensor sensitivity axes to the body frame can be divided into two steps as shown in figure 5.1. In a first step the sensor ouput must be transformed to a coordinate system which is orthogonal and parallel to the body frame.

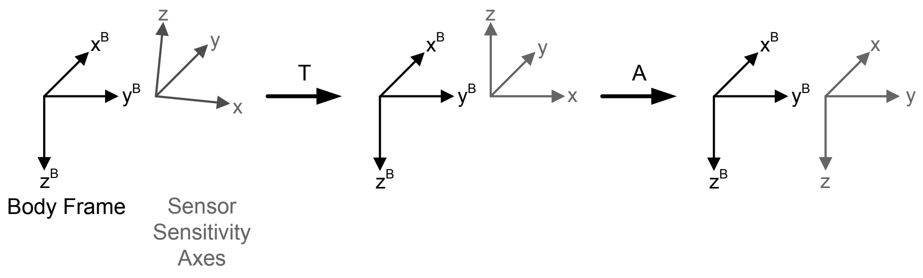

Figure 5.1: Transformation of the sensor output to body frame coordinates.

If the non-orthogonality and the misalignment of the sensor sensitivity axes are only "small", this transformation can be done by [38]

$$T = \begin{bmatrix} 1 & -\alpha_{yz} & \alpha_{zy} \\ \alpha_{xz} & 1 & -\alpha_{zx} \\ -\alpha_{xy} & \alpha_{yx} & 1 \end{bmatrix} \tag{5.1}$$

In a second step the matrix A is used to align the orthogonalized sensor ouput with the body frame. Thereby A is a simple matrix whose entries are only ± 1 or 0 and can easily be determined in advance of the calibration by using the sensor coordinate frames in section 4.3.

So the transformation of the sensor output to the body frame is

$$s^B = A\,T\,s^S \tag{5.2}$$

where s^B and s^S denote the measurement in body frame coordinates and sensor coordinates, respectively.
As the output of the sensor is usually a voltage proportional to the sensed physical quantity, the output must be scaled. Additionaly the output is often biased. So let K be the scale factor matrix and b the bias vector defined as

$$K = diag\,(k_x,\, k_y,\, k_z)\,, \qquad b = [b_x\; b_y\; b_z] \tag{5.3}$$

Consequently the sensor output model is

$$\begin{aligned} y &= K\,s^S + b + \nu \\ &= K\,T^{-1}\,A^{-1}\,s^B + b + \nu \end{aligned} \tag{5.4}$$

where $s^S = T^{-1}\,A^{-1}\,s^B$ from equation (5.2) is inserted. y is the sensor voltage output and ν is measurement noise. This model is applicable to accelerometers, gyroscopes and magnetometers.
The measurement in body frame coordinates as a function of the sensor output voltage is then

$$s^B = h\,(y,\,\theta) = A\,T\,K^{-1}\cdot(y - b) \tag{5.5}$$

where θ is the parameter vector which must be determined [38]

$$\theta = [k_x\; k_y\; k_z\; b_x\; b_y\; b_z\; \alpha_{xz}\; \alpha_{xy}\; \alpha_{yx}\; \alpha_{yz}\; \alpha_{zx}\; \alpha_{zy}] \tag{5.6}$$

5.2 Calibration

The calibration procedure is accomplished using a mechanical platform placing the IMU into different positions and rotating the IMU around different axes. As the cost of such a mechanical platform should not exceed the cost of a low cost IMU, instead of a high sophisticated platform as for example described in [12], the mechanical platform shown in figure 5.2 is used. It consists of a motor turning the sensor board around one axis. At the same time, an encoder provides the reference angle.
Comparing the compensated sensor output with the corresponding reference value allows then the estimation of the 12 unknowns in the sensor output model (5.4). As the error between the compensated sensor output and the reference should be zero, the different sensors are calibrated by minimizing the error over all measurements. Thus, the calibration procedure for finding the parameter vector $\hat{\theta}$ can be stated as a minimization problem

$$\hat{\theta} = \text{argmin}_\theta\,\{L\,(\theta)\} \tag{5.7}$$

with the cost function

$$L\,(\theta) = \sum_{m=1}^{M} \left(\|h_{ref} - h\,(y,\,\theta)\|_m\right) \tag{5.8}$$

where M is the number of measurements.
The minimization of $L\,(\theta)$ is done using the `fminsearch` function provided by MATLAB®. After the calibration, another set of measurements is used to validate the result of the calibration.
The idea for the sensor calibration is taken from [38], but here another cost function is utilized. The following subsections describe the calibration of the different sensors on the sensor board.

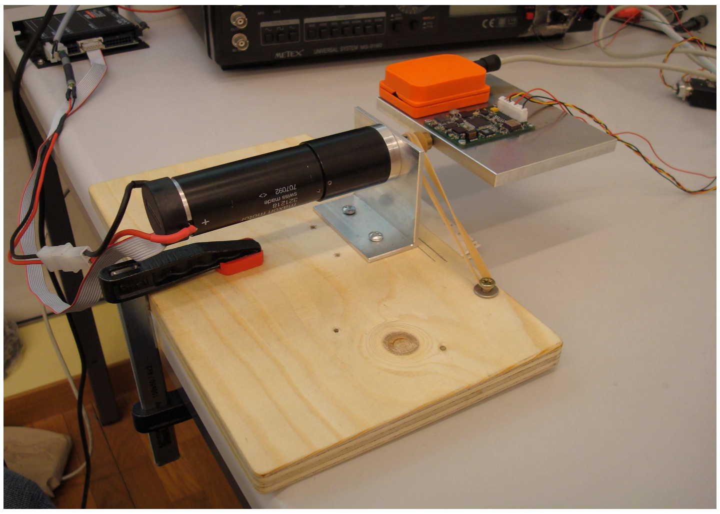

Figure 5.2: The calibration platform.

5.2.1 Accelerometers

For the calibration of the accelerometers the sensor board was placed into 15 different positions where per position 20 measurements were taken.

SCA300-E01

For this accelerometer the A and the T matrices and the bias must be determined as the sensor ouput is already scaled.
It must be pointed out that some problems encountered with the output of this sensor. Whereas the sensor z-axis outputs the gravitational acceleration as specified in the datasheet if it is vertical, the x- and the y-axes output the gravitational acceleration 180 ° inverted compared to the specification in the datasheet if they are vertical. Therefor a new coordinate system with by 180 ° inverted sensor x- and y-axes as specified in figure 5.3 is introduced. Thus, the matrix A is given by

$$A = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

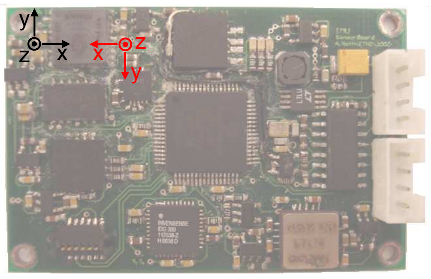

Figure 5.3: New SCA300-E01 sensor coordinate system. Black: original coordinate system, red: new coordinate system.

Figure 5.4 shows the comparison between the reference output and the corrected sensor output in body frame coordinates.

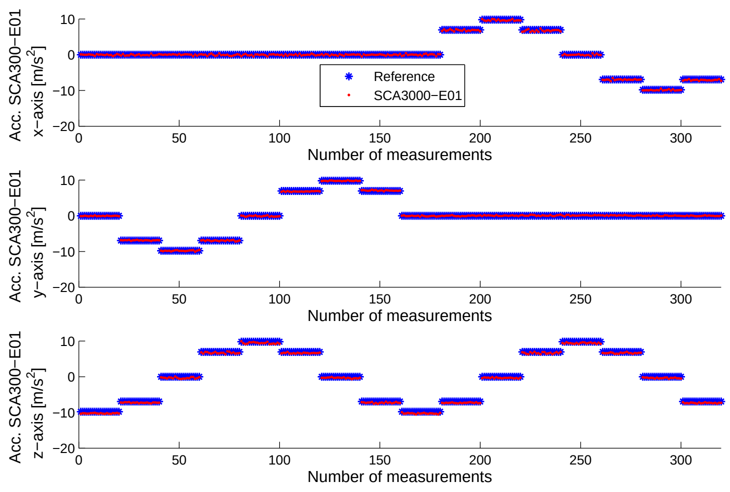

Figure 5.4: Calibration of accelerometer SCA300-E01 (in body frame coordinates).

LIS3LV02DL

This accelerometer provides the factory calibration values of the biases and scaling gains, so only the A and the T matrices must be determined. The matrix A can be specified as

$$A = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Figure 5.5 shows the result of the calibration for this accelerometer. It is not possible to calibrate the sensor in order to get a satisfactory result. Thus, this accelerometer is already rejected from the selection process after the calibration.

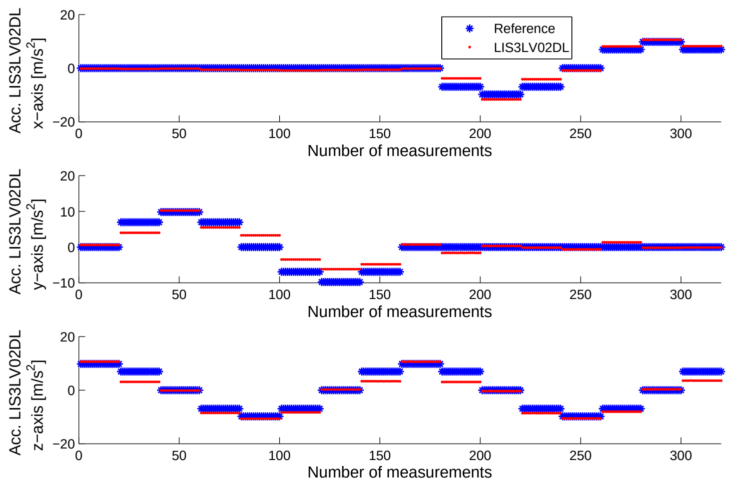

Figure 5.5: Calibration of accelerometer LIS3LV02DL (in body frame coordinates).

MMA7260QT

For this accelerometer all 12 parameters of the parameter vector θ must be identified. The matrix A is

$$A = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The calibration result is shown in figure 5.6.

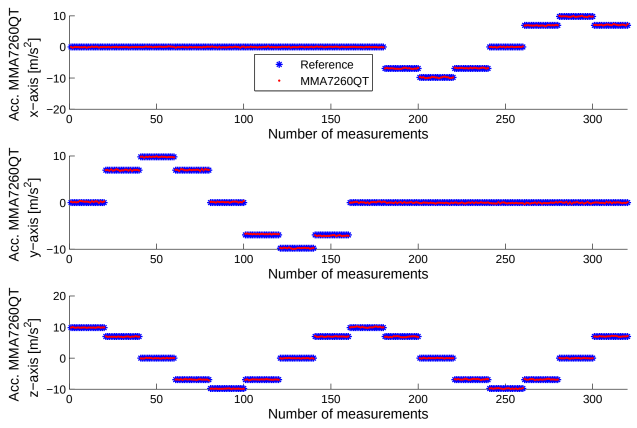

Figure 5.6: Calibration of accelerometer MMA7260QT (in body frame coordinates).

5.2.2 Gyroscopes

For the calibration of the gyroscopes it must be taken into account that gyroscopes have a drifting bias. Because of that the measurements for the calibration of the gyroscopes are taken during a short time period, so that the bias can be considered to be constant.
The measurements are taken by rotating the sensor board around the three body frame axes. Figure 5.7 shows the result of the calibration.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

5.2.3 Magnetometers

HMC6052

The magnetometer HMC6052 can not be calibrated like the sensors before because of a missing reference. Instead an indirect calibration procedure is used.
When constantly turning the magnetometer horizontaly around the body frame z-axis by a motor, the yaw angle can be computed by

$$\psi = \arctan2\left(H_{my}, H_{mx}\right) \tag{5.9}$$

The parameters of the sensor error model can then be calibrated by comparing the output of equation (5.9) with the yaw angle. Since only the rate-of-change of the yaw angle is known by the rotational speed of the motor, only the relative yaw angle

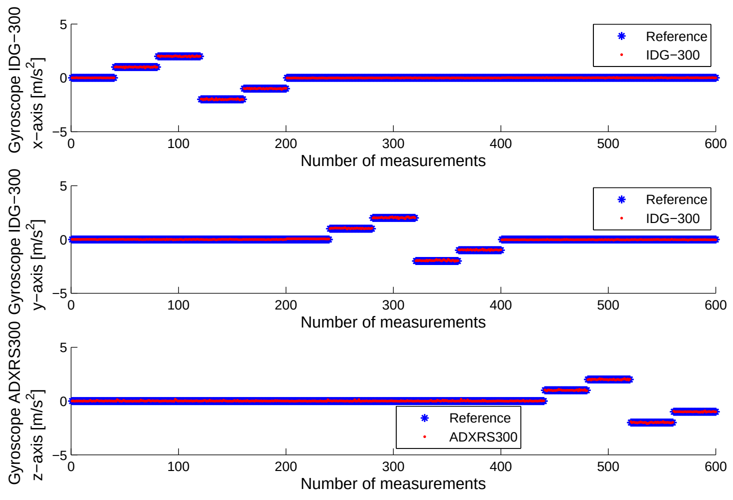

Figure 5.7: Calibration of the gyroscopes IDG-300 and ADXRS300 (in body frame coordinates).

is determined by the integration of the rotational speed. So the initial yaw angle must be also identified by the calibration procedure.
For the magnetometer HMC6052 only the bias of the sensor error model is calibrated.

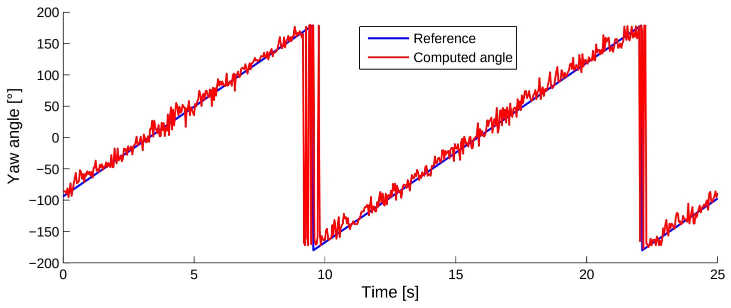

Figure 5.8: Calibration of the magnetometer HMC6052 (in body frame coordinates).

HMC6352

The magnetometer HMC6352 does not have been calibrated.

5.3 Alternative Calibration Procedure for Accelerometers

The *Department of Microsystems Engineering* at the university of Freiburg has a different procedure for the calibration of the accelerometers [34]. A prerequisite for this method is that the accelerometer has a selectable full scale, i.e., the sensor sensitivity can be changed.
The procedure includes the following steps:

1. Place the accelerometer horizontally
2. Measure the digital value from the AD converter for the different sensitivities

3. Make a graph where you plot the sensitivity on the x axis and the digital value from the AD converter on the y axis

4. Then compute the regression line. The axis intercept of the regression line is the bias (corresponds to a measurement value with a sensitivity of zero)

5. Now you can take the average of about 100 measurements and store this as statical acceleration bias

It is assumed that the sensor sensitivity is precisely known. But according to the scientists in Freiburg the procedure works quite well, despite this assumption.

Chapter 6

Attitude Determination Algorithms

This chapter presents the attitude determination algorithms implemented. As discussed in chapter 2, Kalman filters and complementary filters are investigated. But first, in section 6.1 the modeling of the sensor readings is elucidated.
Afterwards, two different Kalman filters are introduced which differ in the measurement equations, and in the last section 6.3 four complementary filters are presented.

6.1 Sensor Modeling

6.1.1 3-Axis Gyroscope

The gyroscopes measure the three components of the angular velocity Ω of the body frame relative to the inertial frame, expressed in the body frame. Since the gyrocopes suffer from drift, the gyroscope measurement model is given by

$$\Omega_m = \Omega + b\,(t) + \mu \tag{6.1}$$

where Ω are the true angular rates. The bias $b\,(t)$ is slowly time-varying and μ is Gaussian white noise. It is assumed that the dynamics of the bias b is much slower compared to the rotational dynamics of the vehicle [20, 29]. Because of the gyroscope drift and the noise, simple integeration of the angular velocities is insufficient to get the attitude.

6.1.2 3-Axis Accelerometer

The 3-axis accelerometer measures the gravitational acceleration and the linear acceleration of the vehicle. If the linear acceleration is small compared to the gravitational acceleration, then the accelerometer reading A_m provides the components of the gravity vector in the body frame [29]:

$$A_m = [A_{mx},\, A_{my},\, A_{mz}]^T = \pm g\, C_n^b\, z_0 + \nu \tag{6.2}$$

where $g = 9.81 m/s^2$ and ν is white noise. Depending on the manufacturer of the accelerometer, a positiv or negative gravitational acceleration is measured.

From the gravity vector measurement the roll and the pitch angles are given by:

$$\begin{aligned}\phi &= \begin{cases}\arctan 2\left(-A_{my}, -A_{mz}\right) & \text{for neg. meas. gravitational acc.;}\\ \arctan 2\left(A_{my}, A_{mz}\right) & \text{for pos. meas. gravitational acc.}\end{cases}\\ \theta &= \begin{cases}\arcsin\left(\frac{A_{mx}}{|A_m|}\right) & \text{for neg. meas. gravitational acc.;}\\ \arcsin\left(\frac{-A_{mx}}{|A_m|}\right) & \text{for pos. meas. gravitational acc.}\end{cases}\end{aligned} \tag{6.3}$$

But this is only reliable for slow dynamics, just as the yaw angle determination in the next subsection.

6.1.3 2-Axis Magnetometer

The magnetometers measure the earth magnetic field, which has in Zurich an inclination of about $i \approx 63\,^\circ$ [25], i.e., the earth magnetic field vector H is [29]

$$H = \begin{bmatrix} H_x \\ H_y \\ H_z \end{bmatrix} = |H| \cdot \begin{bmatrix} \cos(i) \\ 0 \\ \sin(i) \end{bmatrix} \tag{6.4}$$

where $|H| \approx 0.47G$ for Zurich. The measurement of m in the body frame is then given by

$$H_m = \left[H_{mx},\, H_{my},\, H_{mz}\right]^T = C_n^b\, H + \nu \tag{6.5}$$

where ν is white noise.
If measurements of all three axes are available and the roll and the pitch angles are already known, the yaw angle can be computed by [49]

$$\begin{aligned}\psi \;=\; & \operatorname{atan2}\left(H_{mz} \cdot \sin\phi - H_{my} \cdot \cos\phi,\; \ldots\right.\\ & \left.H_{mx} \cdot \cos\theta + \left(H_{my} \cdot \sin\phi + H_{mz} \cdot \cos\phi\right) \cdot \sin\theta\right)\end{aligned} \tag{6.6}$$

Since the sensor board contains only a 2-axis magnetometer, only the H_{mx} and the H_{my} components are available. In quasi-stationary flight conditions (roll and pitch angles near to zero), the yaw angle is approximated by equation (5.9).
But this equation is only reliable if the roll and the pitch angles are really near to zero. So another method is needed to determine the yaw angle if the roll and the pitch angles are not near to zero.
Considering only the first two components of equation (6.5) yields

$$\begin{aligned} c_\theta c_\psi H_x - s_\theta H_z &= H_{mx}\\ \left(s_\phi s_\theta c_\psi - c_\phi s_\psi\right) H_x + c_\theta s_\phi H_z &= H_{my}\end{aligned} \tag{6.7}$$

This can be rearranged as

$$\begin{bmatrix} c_\theta H_x & 0 \\ s_\phi s_\theta H_x & -c_\phi H_x \end{bmatrix} \cdot \begin{bmatrix} c_\psi \\ s_\psi \end{bmatrix} = \begin{bmatrix} H_{mx} + s_\theta H_z \\ H_{my} - c_\theta s_\phi H_z \end{bmatrix} \tag{6.8}$$

If the roll and the pitch angles are already given, the solution of this equation is

$$\begin{bmatrix} c_\psi \\ s_\psi \end{bmatrix} = \begin{bmatrix} c_\theta H_x & 0 \\ s_\phi s_\theta H_x & -c_\phi H_x \end{bmatrix}^{-1} \cdot \begin{bmatrix} H_{mx} + s_\theta H_z \\ H_{my} - c_\theta s_\phi H_z \end{bmatrix} \tag{6.9}$$

Finally, the yaw angle is computed by

$$\psi = \operatorname{arctan2}\left(s_\psi,\, c_\psi\right) \tag{6.10}$$

6.2 Extended Kalman Filters

This section presents the compared EKFs.
A dynamic model of the UAV whose attitude has to be estimated is not included into the process models of the EKFs. This is first of all because the IMU is used for different types of UAVs (fixed wing or rotating wing UAVs). Secondly, the inclusion of the UAV's dynamic model would result in an increased computational burden of the EKF [36].
To further decrease the computational burden, the attitude is represented by a unit quaternion in the state vector, which avoids the calculation of transcendental functions in the state transition matrix A_k. So the filters can be easier implemented on a low cost microprocessor with limited processing power. Additionally, the problem of the singularity at $\theta = \pm 90\,^\circ$ in the Euler angle representation is avoided [10].
Two EKFs are investigated. Both of them are using the same process model, but differ in the measurement model. For the first presented EKF the accelerometer and magnetometer measurements are used to precalculate the attitude represented by the Euler angles. The Euler angles are then delivered to the EKF for the measurement update as shown in figure 6.1.

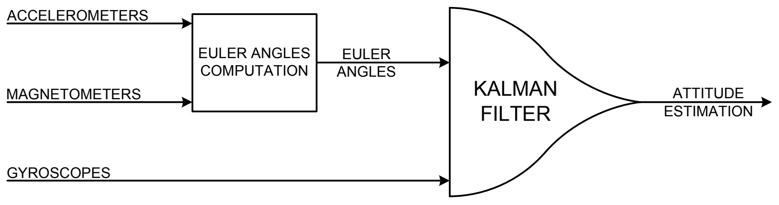

Figure 6.1: Schematic of the first Kalman filter.

The second presented EKF directly uses the accelerometer and magnetometer measurements for the measurement update.
It has to be mentioned that Julier and Uhlman present in [14] the Unscented Kalman filter (UKF) which is a new extension of the Kalman Filter to nonlinear systems. They claim their new filter to be much less difficult to implement and to have superior performance compared to that of the EKF. A comparative conclusion is made by the authors in [42].
However, LaViola [19] and St.-Pierre and Gingras [39], who make a comparison between the UKF and EKF for the problem of quaternion motion estimation and position estimation respectively, state that the accuracy of the EKF and UKF are roughly the same. But what is more important, they show experimentally that the computational cost of the UKF is much greater than the computational cost of the EKF. This is why only the EKF is considered.

6.2.1 EKF with Euler Angles Measurement

Process Model

Instead of the body dynamics differential equation, the differential equation describing the propagation of the quaternion with time is included into the filter system dynamics (equation (3.19)) [22]:

$$\begin{aligned}\dot{q}_k &= \frac{1}{2} q_k \cdot \mathbf{p}\left(\Omega_k\right) \\ &= \frac{1}{2} \Omega_k \cdot q_k \end{aligned} \tag{6.11}$$

where

$$\Omega_k = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}_k \tag{6.12}$$

Since the measurements of the angular velocities are biased, the bias must be substracted from the gyro measurements

$$\Omega_k = \begin{bmatrix} 0 & -(\omega_x - w_{x\_bias}) & -(\omega_y - w_{y\_bias}) & -(\omega_z - w_{z\_bias}) \\ (\omega_x - w_{x\_bias}) & 0 & (\omega_z - w_{z\_bias}) & -(\omega_y - w_{y\_bias}) \\ (\omega_y - w_{y\_bias}) & -(\omega_z - w_{z\_bias}) & 0 & (\omega_x - w_{x\_bias}) \\ (\omega_z - w_{z\_bias}) & (\omega_y - w_{y\_bias}) & -(\omega_x - w_{x\_bias}) & 0 \end{bmatrix}_k \tag{6.13}$$

The expansion of equation (6.11) with included Ω_k from (6.13) yields then

$$\begin{bmatrix} \dot{q_0} \\ \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix}_k = \frac{1}{2} \begin{bmatrix} -(\omega_x - w_{x\_bias}) \cdot q_1 - (\omega_y - w_{y\_bias}) \cdot q_2 - (\omega_z - w_{z\_bias}) \cdot q_3 \\ +(\omega_x - w_{x\_bias}) \cdot q_0 + (\omega_z - w_{z\_bias}) \cdot q_2 - (\omega_y - w_{y\_bias}) \cdot q_3 \\ +(\omega_y - w_{y\_bias}) \cdot q_0 - (\omega_z - w_{z\_bias}) \cdot q_1 + (\omega_x - w_{x\_bias}) \cdot q_3 \\ +(\omega_z - w_{z\_bias}) \cdot q_0 + (\omega_y - w_{y\_bias}) \cdot q_1 - (\omega_x - w_{x\_bias}) \cdot q_2 \end{bmatrix}_k \tag{6.14}$$

This shows that additionally to the quaternion components the gyroscope bias must be estimated, too. So, the system state vector is composed of

$$x_k = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix}_k = \begin{bmatrix} w_{x\_bias} \\ w_{y\_bias} \\ w_{z\_bias} \\ q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}_k \tag{6.15}$$

The time derivative of the gyroscope bias is modeled as

$$\begin{bmatrix} \dot{w}_{x\_bias} \\ \dot{w}_{y\_bias} \\ \dot{w}_{z\_bias} \end{bmatrix}_k = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}_k \tag{6.16}$$

So the whole process model in terms of the state vector is

$$\dot{x}_k = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \end{bmatrix}_k = \frac{1}{2} \begin{bmatrix} 0 \\ 0 \\ 0 \\ -(\omega_x - x_1)x_5 - (\omega_y - x_2)x_6 - (\omega_z - x_3)x_7 \\ +(\omega_x - x_1)x_4 + (\omega_z - x_3)x_6 - (\omega_y - x_2)x_7 \\ +(\omega_y - x_2)x_4 - (\omega_z - x_3)x_5 + (\omega_x - x_1)x_7 \\ +(\omega_z - x_3)x_4 + (\omega_y - x_2)x_5 - (\omega_x - x_1)x_6 \end{bmatrix}_k \tag{6.17}$$

The linearization of the system equations yields the following state transition matrix A_k

$$\begin{aligned} A_k &= \left[\frac{\partial f}{\partial x}\right]_k \\ &= \frac{1}{2}\left[\begin{array}{ccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ +x_5 & +x_6 & +x_7 & 0 & -(\omega_x - x_1) & -(\omega_y - x_2) & -(\omega_z - x_3) \\ -x_4 & +x_7 & -x_6 & (\omega_x - x_1) & 0 & (\omega_z - x_3) & -(\omega_y - x_2) \\ -x_7 & -x_4 & +x_5 & (\omega_y - x_2) & -(\omega_z - x_3) & 0 & (\omega_x - x_1) \\ +x_6 & -x_5 & -x_4 & (\omega_z - x_3) & (\omega_y - x_2) & -(\omega_x - x_1) & 0 \end{array}\right]_k \end{aligned} \tag{6.18}$$

Measurement Model

The measurement used for the measurement update (equation (3.29)-(3.31)) is the attitude represented by the roll, pitch and yaw angles

$$z_k = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{6.19}$$

So the Euler angles must be precalculated by equations (6.3) and (6.5) using the accelerometer and the magnetometer measurements.
From equations (3.18) the nonlinear measurement model is

$$h(x_k) = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \arctan 2\left(2x_6x_7 + 2x_4x_5,\ 1 - 2\left(x_5^2 + x_6^2\right)\right) \\ -\arcsin\left(2x_5x_7 - 2x_4x_6\right) \\ \arctan 2\left(2x_5x_6 + 2x_4x_7,\ 1 - 2\left(x_6^2 + x_7^2\right)\right) \end{bmatrix}_k \tag{6.20}$$

So the linear measurement matrix H_k is

$$h(x_k) = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \approx H_k \cdot x_k = \begin{bmatrix} 0_{3\times3} & \tilde{H}_k \end{bmatrix} \cdot x_k \tag{6.21}$$

with

$$\tilde{H}_k = \begin{bmatrix} \frac{\partial\phi}{\partial x_4} & \frac{\partial\phi}{\partial x_5} & \frac{\partial\phi}{\partial x_6} & \frac{\partial\phi}{\partial x_7} \\ \frac{\partial\theta}{\partial x_4} & \frac{\partial\theta}{\partial x_5} & \frac{\partial\theta}{\partial x_6} & \frac{\partial\theta}{\partial x_7} \\ \frac{\partial\psi}{\partial x_4} & \frac{\partial\psi}{\partial x_5} & \frac{\partial\psi}{\partial x_6} & \frac{\partial\psi}{\partial x_7} \end{bmatrix}_k \tag{6.22}$$

which is

$$\tilde{H}_k = \begin{bmatrix} \frac{2c_{33}x_5}{c_{33}^2+c_{32}^2} & \frac{2(c_{33}x_4+2c_{32}x_5)}{c_{33}^2+c_{32}^2} & \frac{2(c_{33}x_7+2c_{32}x_6)}{c_{33}^2+c_{32}^2} & \frac{2c_{33}x_6}{c_{33}^2+c_{32}^2} \\ \frac{2x_6}{\sqrt{1-c_{31}^2}} & \frac{-2x_7}{\sqrt{1-c_{31}^2}} & \frac{2x_4}{\sqrt{1-c_{31}^2}} & \frac{-2x_5}{\sqrt{1-c_{31}^2}} \\ \frac{2c_{11}x_7}{c_{11}^2+c_{21}^2} & \frac{2c_{11}x_6}{c_{11}^2+c_{21}^2} & \frac{2(c_{11}x_5+2c_{21}x_6)}{c_{11}^2+c_{21}^2} & \frac{2(c_{11}x_4+2c_{21}x_7)}{c_{11}^2+c_{21}^2} \end{bmatrix}_k \tag{6.23}$$

where c_{ij} are the entries of the direction cosine matrix, i.e.,

$$\begin{aligned}
c_{11} &= 1 - 2\left(x_6^2 + x_7^2\right) \\
c_{33} &= 1 - 2\left(x_5^2 + x_6^2\right) \\
c_{21} &= 2\left(x_5 x_6 + x_4 x_7\right) \\
c_{31} &= 2\left(x_5 x_7 - x_4 x_6\right) \\
c_{32} &= 2\left(x_6 x_7 + x_4 x_5\right)
\end{aligned} \tag{6.24}$$

6.2.2 EKF with Accelerometer and Magnetometer Measurements

Process Model

The process model and the state vector are the same as for the EKF in subsection 6.2.1.

Measurement Model

This EKF directly uses the measurements from the 3-axis accelerometer and the 2-axis magnetometer for the measurement update, i.e.,

$$z_k = \begin{bmatrix} A_{mx} \\ A_{my} \\ A_{mz} \\ H_{mx} \\ H_{my} \end{bmatrix}_k \tag{6.25}$$

Hence, the first three componens of the measurement model are the components of the gravity vector transformed to body frame, i.e., $\pm g \cdot C_n^b \cdot z_0$, and the 4th and the 5th component are the x and the y component of the earth magnetic field vector also transformed to body frame, i.e., $C_n^b \cdot H$.
The measurement model for accelerometers which measure a negativ gravitational acceleration in body frame coordinates consequently is

$$h\left(x_k\right) = \begin{bmatrix} -2\left(x_5 x_7 - x_4 x_6\right) \cdot g \\ -2\left(x_6 x_7 + x_4 x_5\right) \cdot g \\ -\left(1 - 2\left(x_5^2 + x_6^2\right)\right) \cdot g \\ \left(1 - 2\left(x_6^2 + x_7^2\right)\right) \cdot H_x + 2\left(x_5 x_7 - x_4 x_6\right) \cdot H_z \\ 2\left(x_5 x_6 - x_4 x_7\right) \cdot H_x + 2\left(x_6 x_7 + x_4 x_5\right) \cdot H_z \end{bmatrix}_k \tag{6.26}$$

and for accelerometers which measure a positive gravitational acceleration respectively

$$h\left(x_k\right) = \begin{bmatrix} 2\left(x_5 x_7 - x_4 x_6\right) \cdot g \\ 2\left(x_6 x_7 + x_4 x_5\right) \cdot g \\ \left(1 - 2\left(x_5^2 + x_6^2\right)\right) \cdot g \\ \left(1 - 2\left(x_6^2 + x_7^2\right)\right) \cdot H_x + 2\left(x_5 x_7 - x_4 x_6\right) \cdot H_z \\ 2\left(x_5 x_6 - x_4 x_7\right) \cdot H_x + 2\left(x_6 x_7 + x_4 x_5\right) \cdot H_z \end{bmatrix}_k \tag{6.27}$$

The linear measurement matrix H_k is then

$$H_k = \begin{bmatrix} 0_{3\times 3} & \tilde{H}_k \end{bmatrix} \tag{6.28}$$

where $\tilde{H}_k$ for the measurement model (6.26) is

$$\tilde{H}_k = \begin{bmatrix} 2x_6 \cdot g & -2x_7 \cdot g & 2x_4 \cdot g & -2x_5 \cdot g \\ -2x_5 \cdot g & -2x_4 \cdot g & -2x_7 \cdot g & -2x_6 \cdot g \\ 0 & 4x_5 \cdot g & 4x_6 \cdot g & 0 \\ -2x_6 \cdot H_z & 2x_7 \cdot H_z & -4x_6 \cdot H_x - 2x_4 \cdot H_z & -4x_7 \cdot H_x + 2x_5 \cdot H_z \\ -2x_7 \cdot H_x + 2x_5 \cdot H_z & 2x_6 \cdot H_x + 2x_4 \cdot H_z & 2x_5 \cdot H_x + 2x_7 \cdot H_z & -2x_4 \cdot H_x + 2x_6 \cdot H_z \end{bmatrix}_k \tag{6.29}$$

and for the measurement model (6.27) respectively

$$\tilde{H}_k = \begin{bmatrix} -2x_6 \cdot g & 2x_7 \cdot g & -2x_4 \cdot g & 2x_5 \cdot g \\ 2x_5 \cdot g & 2x_4 \cdot g & 2x_7 \cdot g & 2x_6 \cdot g \\ 0 & -4x_5 \cdot g & -4x_6 \cdot g & 0 \\ -2x_6 \cdot H_z & 2x_7 \cdot H_z & -4x_6 \cdot H_x - 2x_4 \cdot H_z & -4x_7 \cdot H_x + 2x_5 \cdot H_z \\ -2x_7 \cdot H_x + 2x_5 \cdot H_z & 2x_6 \cdot H_x + 2x_4 \cdot H_z & 2x_5 \cdot H_x + 2x_7 \cdot H_z & -2x_4 \cdot H_x + 2x_6 \cdot H_z \end{bmatrix}_k \tag{6.30}$$

6.3 Complementary Filters

Complementary filters try to fuse the low bandwidth accelerometer and magnetometer measurements with the high bandwidth gyroscope measurements [20]. The studied complementary filters also include the estimation of the gyroscope's bias, whereas the bias should only vary slowly.
Four different complementary filters are presented. The first one is based on the propagation with time of a quaternion representing the attitude. Then two complementary filters using the whole direction cosine matrix are shown. Finally a filter estimating the direction of the gravity and the earth magnetic field in the body frame is presented.
As the filters in the original papers use 3-axis magnetometers, the filters are adopted to be used with a 2-axis magnetometer. It must be pointed out that the subsections below concentrate only on the implementation of the different filters, for a detailed analysis, e.g., the convergence properties of the filters refer to the indicated papers.

6.3.1 Quaternion Based Complementary Filter

The filter presented in this subsection is based on the propagation of a unit quaternion with time [20].
Let $\hat{q}$ be the estimated and q the actual attitude expressed in quaternions. Then the error $\tilde{q}$ between the estimated and the actual quaternions is defined as

$$\widetilde{q} = \hat{q}^{-1} \cdot q = \begin{bmatrix} \widetilde{s} \\ \widetilde{v} \end{bmatrix} \tag{6.31}$$

The actual quaternion is calculated by using the measurements from the accelerometers and magnetometers. As the quaternion can not be derived directly from this

measurements, first, the Euler angles are calculated using equations (6.3), (6.9) and (6.10). The Euler angles are then transformed to a unit quaternion by equation (3.17).
The filter dynamics is given by

$$\dot{\hat{q}} = \frac{1}{2}\hat{q} \cdot \mathbf{p}\left(\Omega_m - \hat{b} + \omega\right) \tag{6.32}$$

where $\hat{b}$ is the gyroscope's bias estimation and ω is a correction term that is zero when $q = \hat{q}$ and is defined by

$$\omega = k_{est}\, \widetilde{s}\, \widetilde{v} \tag{6.33}$$

The proposed dynamics of the bias estimation $\hat{b}$ is

$$\dot{\hat{b}} = -k_b\, \widetilde{s}\, \widetilde{v} \tag{6.34}$$

The filter and the bias dynamics must then be integrated by

$$\hat{q}^*_{k+1} = \hat{q}_k + T \cdot \dot{\hat{q}}_{k+1} \tag{6.35}$$

$$\hat{b}_{k+1} = \hat{b}_k + T \cdot \dot{\hat{b}} \tag{6.36}$$

where T is the sampling period.
To preserve unity length of the quaternion, the integrated quaternion is divided by its norm.

$$\hat{q}_{k+1} = \frac{\hat{q}^*_{k+1}}{|\hat{q}^*_{k+1}|} \tag{6.37}$$

6.3.2 Rotation Matrix Based Complementary Filter - Type 1

This complementary filter propagetes the direct cosine matrix C^n_b [13]. The dynamics of the estimation of C^n_b is given by

$$\dot{\hat{C}^n_b} = \hat{C}^n_b \left(\Omega_m - \hat{b} + \omega\right)_\times \tag{6.38}$$

where $(\ )_\times$ is a skew-symmetric matrix, i.e.,

$$\Omega_\times = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \tag{6.39}$$

where $\Omega \in \mathbb{R}^3$.
Then let $\hat{a}$ and $\hat{m}$ be the estimates of the gravitational and the magnetic vector field directions respectively, given by

$$\hat{a} = \hat{C^b_n}\, a = \left(\hat{C^n_b}\right)^T a \qquad \hat{m} = \hat{C^b_n}\, m = \left(\hat{C^n_b}\right)^T m \tag{6.40}$$

where $a = \pm z_0$ according to the employed accelerometer and $m = H/\,|H|$.
The correction term ω and the gyroscope's bias estimation dynamics $\dot{\hat{b}}$ are

$$\begin{aligned} \omega &= -k_p \cdot \left(k_a \cdot \frac{A_m}{|A_m|} \times \hat{a} + k_m \cdot \frac{H_m}{|H_m|} \times \hat{m}\right), \qquad k_p > 0 && (6.41) \\ \dot{\hat{b}} &= -\frac{k_I}{k_P} \cdot \omega, \qquad k_I > 0 && (6.42) \end{aligned}$$

Since only the H_{mx} and H_{my} components of H_m are available, the third component H_{mz} must be calculated by (see third component of equation (6.5))

$$H_{mz} = (c_\phi\, s_\theta\, c_\psi + s_\phi\, s_\psi) \cdot H_x + c_\theta\, c_\phi \cdot H_z \tag{6.43}$$

using roll angle calculated by equation (6.3). Also from equation (6.3) s_θ and c_θ are given by (according to the employed accelerometer)

$$s_\theta = \frac{\pm A_{mx}}{|A_m|} \quad \text{and thus} \quad c_\theta = \sqrt{1 - s_\theta^2} \tag{6.44}$$

c_ψ and s_ψ are directly determined by equation (6.9).
The filter and the bias estimation dynamics must then be integrated. The integration of the bias dynamics is approximated as in equation (6.36).
For the integration of the filter dynamics (6.38), rewritten as $\dot{\hat{C}}_b^n = \hat{C}_b^n \hat{\Omega}_\times$, exact integration is used in order to make sure that $\hat{C}_b^n$ remains a rotation matrix

$$\left(\hat{C}_b^n\right)^{k+1} = \left(\hat{C}_b^n\right)^k A^k \tag{6.45}$$

where $A^k = \exp\left(\left(\Omega_m - \hat{b} + \omega\right)^k_\times\right) = \exp\left(\hat{\Omega}^k_\times\right)$ has the closed form solution given by Rodrigue's formula

$$A^k = I - \hat{\Omega}^k_\times \frac{\sin\left(\left|\hat{\Omega}^k\right| T\right)}{\left|\hat{\Omega}^k\right|} + \left(\hat{\Omega}^k_\times\right)^2 \frac{1 - \cos\left(\left|\hat{\Omega}^k\right| T\right)}{\left|\hat{\Omega}^k\right|^2} \tag{6.46}$$

6.3.3 Rotation Matrix Based Complementary Filter - Type 2

The complementary filter presented in this subsection also propagates the whole cosine direction matrix C_b^n [29].
Let $\{x,\, y,\, z\}$ and $\{\hat{x},\, \hat{y},\, \hat{z}\}$ be direct cosine vectors representing the components of the basis vectors of the navigation frame in the body frame and in the estimated body frame, respectively:

$$C_b^n = [x\, y\, z]^T \quad \text{and} \quad \hat{C}_b^n = [\hat{x}\, \hat{y}\, \hat{z}]^T \tag{6.47}$$

Assume that the measurement of C_b^n is available, i.e., it can be determined from the accelerometer and the magnetometer measurements by equation (3.2) using equations (6.3), (6.9) and (6.44).
Then define the estimation error as $\tilde{C}_b^n = \left(\hat{C}_b^n\right)^T C_b^n$.
The dynamics of the estimator filter of the direction cosine matrix is defined by

$$\dot{\hat{C}}_b^n = \hat{C}_b^n \hat{\Omega}_\times \tag{6.48}$$

with $\hat{\Omega}$ given by

$$\hat{\Omega} = \tilde{C}_b^n \left(\Omega_m - \hat{b}\right) - k\,(X + Y + Z)\,, \qquad k > 0 \tag{6.49}$$

where

$$X = \hat{x} \times x, \qquad Y = \hat{y} \times y, \qquad Z = \hat{z} \times z \tag{6.50}$$

The dynamics of the gyroscope's bias estimation is proposed as

$$\dot{\hat{b}} = \Gamma \tilde{C}_b^n\, [X + Y + Z]\,, \qquad \Gamma > 0 \tag{6.51}$$

Updating the direction cosine matrix estimation is done by using once again exact integration, i.e.,

$$\left(\hat{C}_b^n\right)^{k+1} = \left(\hat{C}_b^n\right)^k \cdot \exp\left(\hat{\Omega}_\times^k\, T\right) \tag{6.52}$$

where

$$\exp\left(\hat{\Omega}_\times^k\, T\right) = I + \frac{\sin\left(\left|\hat{\Omega}^k\right| T\right)}{\left|\hat{\Omega}^k\right|}\hat{\Omega}_\times^k + \frac{1-\cos\left(\left|\hat{\Omega}^k\right| T\right)}{\left|\hat{\Omega}^k\right|^2}\left(\hat{\Omega}_\times^k\right)^2 \tag{6.53}$$

(be conscious that equation (6.53) is not the same as equation (6.46))
For the bias dynamics integration again Euler integration (6.36) is used.

6.3.4 Gravity Direction Cosine and Earth Magnetic Field Direction Cosine Estimators

In this subsection a totally different method of using complementary filtering for the attitude determination is presented than in the previous subsections [24]. Not a quaternion or a direction cosine matrix directly representing the attitude are estimated, instead two nonlinear complementary filters are designed, one estimating the direction of gravity in the body frame and another estimating the earth magnetic field direction cosine also in the body frame. Then the attitude is determined by these two estimated vectors, which are known and constant in the navigation frame. Let $\hat{x}$ be the gravity direction estimation in the body frame from the filter and x the normalized measurement from the 3-axis accelerometer, respectively. Then the gravity direction cosine estimator has the following kinematics:

$$\dot{\hat{x}} = -\left(\Omega_m + \omega_x\right)_\times \hat{x} \tag{6.54}$$

where

$$\omega_x = -\hat{b} - k_x\left(\hat{x} \times x\right) \tag{6.55}$$

The dynamics of the earth magnetic field direction cosine estimation is

$$\dot{\hat{z}} = -\left(\Omega_m + \omega_z\right)_\times \hat{z} \tag{6.56}$$

where $z = C_n^b \cdot m$ is the earth magnetic field direction cosine in the body frame measured by the magnetometer and $\hat{z}$ is its estimation. As only a 2-axis magnetic field measurement is available, the third component of H_m must be computed by equation (6.43).
The correction term ω_z is defined as

$$\omega_z = -\hat{b} - k_z\left(\hat{z} \times z\right) \tag{6.57}$$

The dynamics corresponding to the gyroscope's bias estimator have the form:

$$\dot{\hat{b}} = k_b\left[\hat{x} \times x + \hat{z} \times z\right] \tag{6.58}$$

The update of the estimates $\hat{x}$ and $\hat{z}$ are computed by exact integration

$$\begin{aligned} \hat{x}^{k+1} &= A_x^k\, \hat{x}^k \\ \hat{z}^{k+1} &= A_z^k\, \hat{z}^k \end{aligned} \tag{6.59}$$

where $A_{(.)}^k$ is given by Rodrigue's formula

$$A_{(.)}^k = I - \left(u_{(.)}^k\right)_\times \frac{\sin\left(\left|u_{(.)}^k\right| T\right)}{\left|u_{(.)}^k\right|} + \left(u_{(.)}^k\right)_\times^2 \frac{1-\cos\left(\left|u_{(.)}^k\right| T\right)}{\left|u_{(.)}^k\right|^2} \tag{6.60}$$

with $u_{(.)}^k = \Omega_m^k + \omega_{(.)}^k$ given by equations (6.55) and (6.57), respectively.
Again equation (6.36) is used for the integration of the bias dynamics.

6.4 Implementation

The above presented attitude determination algorithms are firstly implemented on MATLAB® in order to compare the quality of each algorithm.
As synthetical created measurements never exactly represent the reality, sets of measurements from the sensors on the sensor board are used to run the algorithms offline.
Since the finally selected algorithm must be implemented on a microprocessor, all built-in functions from MATLAB® are avoided and instead own function are used, e.g., the matrix inversion used for the EKFs in section 6.2 [31]. This allows a better assessment of the time taken by the algorithms to compute a new attitude estimation.

Chapter 7

Comparison of different Attitude Estimation Algorithms

Different sensors and attitude determination algorithms were presented in chapter 4 and 6 respectively. Now the sensors and the algorithms have to be compared in order to select the best sensors for designing the final IMU and to find the best algorithm which must be implemented on the microprocessor.
For doing so, different test benches simulating the different environments, in which the IMU will be used, must be constructed. Possible test benches are presented in section 7.1.
The test benches allow the recording of some reference measurement sets with which the different attitude determination algorithms can be compared. For this comparison some performance indexes as introduced in subsection 7.2.2 are needed.

7.1 Test Bench

The final IMU is planned to be used in the following application areas:

- autonomous micro helicopters
- autonomous airplanes
- inspection robots.

In the following, for each application area the disturbances to be mainly expected are itemized:

Autonomous micro helicopter

- Vibrations caused by the motors have to be considered as the main source of disturbances
- Wind gusts can be excluded because the micro helicopters fly only indoor at the moment

Autonomous airplanes

- The main problem faced by airplanes is the non-negligible centrifugal acceleration during a big turn which distorts the measurement of the gravity vector

- Additionally, vibrations caused by the motor must be considered

Inspection robots

- Robots are moving slowly, so linear accelerations can be neglected
- When robots go down a step, knocks have to be expected
- When robots move around, there are vibrations caused by the wheels

For a good evaluation of each filter's performance, measurements from test benches emulating as effectively as possible the different application areas are needed. The following subsection describes possible test benches.

7.1.1 Test Bench Constructions

For testing the different algorithms, several test benches, every simulating a different environment, have been discussed. Thereby, the goal of every test bench is to turn the sensor board around one or several axes by simultaneous generation of the corresponding disturbance. In the following there is an overview of the realized and non-realized test benches:

- **Tests without disturbances (Test A)**
  The attitude determination algorithms can be tuned by taking measurements from a testbench without disturbances. This measurements are also suitable for a first analysis of the algorithms.
  For these tests the same test bench as for the sensor calibration can be used.

- **Simulation of centrifugal force during a big turn**
  Assuming an airplane makes a turn with a radius of $50m$ and has a velocity of $8.4m/s$ (typical values for *Sky Sailor*), the same centrifugal acceleration can be emulated by turning the sensor board on a circle with radius $0.5m$ and a velocity of $0.84m/s$ (corresponds to a rotational speed of $1.68rad/s$).
  But unlike a big turn, the change of the magnetic earth field does not correspond to reality, which also has an effect on the algorithms.

- **Airplane vibrations test (Test B)**
  The vibrations occuring on an airplane can be simulated by mounting a motor with propeller on the same plate as the sensor board. Again, the calibration test bench can be used.

- **Helicopter vibrations test (Test C)**
  The vibrations on an helicopter are simulated simplest by directly mounting the sensor board on an helicopter. The helicopter can then be mounted on the calibration test bench to turn it around one axis, but it must be fixed to guarantee that it does not fly away.

- **Robot arm (Test D)**
  Using a robot arm with 6 degrees of freedom, emulating linear accelerations by simultaneous turning around one or several arbitrary axes is possible.

- **Toy Train**
  Vibrations together with centrifugal acceleration can be produced with a toy train. The problems here are the not controlled velocity and the missing reference angles.

7.2 Comparison

7.2.1 Filter Tuning

As described in subsection 3.3.1, the measurement noise covariance matrix R of the Kalman fiters is tuned by determining the variance of the measurement noise of a measurement set. The process noise covariance matrix Q is tuned by using a position profile applied on the roll and the pitch axes respectively. The yaw axis is not considered for the tuning as the magnetic disturbances within a building do not allow a proper determination of the yaw angle. The tuning is carried out by minimizing the error root mean square (RMS) sum of the roll and pitch angles. Again `fminsearch` from MATLAB® is used.
Tests show that this tuning procedure is not suitable for the tuning of the complementary filters. Thus, the values for the tuning parameters proposed by the authors of the corresponding papers are used.

7.2.2 Performance Indexes

In order to find the best filter, the performance of each individual filter has to be evaluated, which is done by using some performance indexes. Appropriate performance indexes are [39, 19]:

- Mean computational time for determining an attitude
- Root mean square of the error
- Mean error
- Error variance

7.2.3 Analysis

This subsection analyses the perfomance of the different attitude determination algorithms in cooperation with the different sensors on the sensor board.
The accelerometer LIS3LV02DL has already shown during the calibration procedure to be unsuitable (see subsection 5.2.1), and thus, is not considered any more. Although the magnetometer HMC6352 is a 2-axis magnetometer, it can output only one of the axes (see appendix A), and so the magnetometer HMC6052 is the better choice.
Thus, only the interaction of the two accelerometers SCA3000-E01 and MMA7260QT with the algorithms must be analysed, since the gyroscopes and the magnetometer to be employed on the final IMU are already determined.
For the selection of the best suitable attitude determination algorithm and accelerometer, the performance indexes presented above are evaluated for three measurement sets, two without disturbances and one with airplane vibrations. As for the tuning only the roll and the pitch axes are taken into account.
First of all, the mean computational time for determining an attitude is analysed for all filters. Table 7.1 shows the results, whereas the best value is highlighted with blue and the worst value with red.
Since it is hard to say how much time a filter will take on a microprocessor with limited processing power, the sampling rate of the following measurements is set to 25Hz, which is rather a slow sampling rate. But this is a conservative choice and it also shows which algorithm can already produce acceptable results for slow sampling rates.

	Kalman Filters		Complementary Filters			
	Filter 6.2.1[1]	Filter 6.2.2	Filter 6.3.1	Filter 6.3.3	Filter 6.3.3	Filter 6.3.4
Mean computational time [ms][2]	0.68	0.80	0.35	0.32	0.39	0.42

Table 7.1: Analysis of the mean computational time of the attitude determination algorithms.

In the following, the RMS error, the mean error and the error variance computed for several measurement sets are presented.

Test A

The attitude determination algorithms are first tested with measurements without disturbances, called *Test A*. The test setup is the same as shown in figure 5.2.
Tables 7.2 and 7.3 show the performance indexes for all filters run with the two different accelerometers for static test conditions, i.e., the roll and the pitch axes are kept constant to 0 °. This allows the evaluation of each filters' precision.
In tables 7.4 and 7.5 the performance indexes for a position profile applied seperately on the pitch and roll axes are presented.

Performance index	Axis	Kalman Filters		Complementary Filters			
		Filter 6.2.1	Filter 6.2.2	Filter 6.3.1	Filter 6.3.3	Filter 6.3.3	Filter 6.3.4
RMS error [deg]	roll	0.83	0.79	0.6	1.22	0.63	0.59
	pitch	0.69	0.96	0.45	1.25	0.55	0.57
Mean error [deg]	roll	0.63	0.6	0.46	0.99	0.48	0.45
	pitch	0.56	0.85	0.35	1.03	0.44	0.46
Error variance [deg^2]	roll	0.29	0.26	0.14	0.51	0.18	0.14
	pitch	0.16	0.2	0.08	0.52	0.11	0.11

Table 7.2: Performance indexes of the attitude determination algorithms with accelerometer SCA300-E01 for test A and static test conditions.

Performance index	Axis	Kalman Filters		Complementary Filters			
		Filter 6.2.1	Filter 6.2.2	Filter 6.3.1	Filter 6.3.3	Filter 6.3.3	Filter 6.3.4
RMS error [deg]	roll	1.41	1.31	1.83	2.65	4.12	1.84
	pitch	0.94	1.51	0.77	2.63	1.45	1.31
Mean error [deg]	roll	1.11	1	1.49	2.14	3.78	1.48
	pitch	0.77	1.31	0.63	2.14	1.14	1.06
Error variance [deg^2]	roll	0.75	0.71	1.13	2.43	2.65	1.2
	pitch	0.3	0.56	0.2	2.35	0.78	0.59

Table 7.3: Performance indexes of the attitude determination algorithms with accelerometer MMA7260QT for test A and static test conditions.

[1] *Filter 6.2.1* means the filter from subsection 6.2.1
[2] on a laptop with Intel® Pentium® processor 1500MHz, 512 MB RAM

Performance index	Axis	Kalman Filters		Complementary Filters			
		Filter 6.2.1	Filter 6.2.2	Filter 6.3.1	Filter 6.3.3	Filter 6.3.3	Filter 6.3.4
RMS error [deg]	roll	0.28	0.38	0.39	1.08	0.55	0.73
	pitch	0.79	1.15	0.95	1.43	1.07	0.86
Mean error [deg]	roll	0.22	0.3	0.31	0.93	0.4	0.52
	pitch	0.69	1.14	0.77	1.25	0.83	0.76
Error variance [deg^2]	roll	0.03	0.05	0.06	0.29	0.15	0.27
	pitch	0.14	0.26	0.3	0.47	0.46	0.17

Table 7.4: Performance indexes of the attitude determination algorithms with accelerometer SCA300-E01 for test A and dynamic test conditions.

Performance index	Axis	Kalman Filters		Complementary Filters			
		Filter 6.2.1	Filter 6.2.2	Filter 6.3.1	Filter 6.3.3	Filter 6.3.3	Filter 6.3.4
RMS error [deg]	roll	1.48	1.23	1.59	1.59	2.24	1.8
	pitch	1.25	1.55	1.31	2.3	1.45	1.39
Mean error [deg]	roll	1.38	1.05	1.5	1.3	2.02	1.45
	pitch	1.13	1.41	1.14	2.14	1.17	1.31
Error variance [deg^2]	roll	0.3	0.42	0.29	0.83	0.95	1.12
	pitch	0.28	0.42	0.42	0.71	0.75	0.22

Table 7.5: Performance indexes of the attitude determination algorithms with accelerometer MMA7260QT for test A and dynamic test conditions.

Test B

Test B is a test, where the effect of vibrations, produced with an airplane propulsion system, is analysed. Tables 7.6 and 7.7 show the effect of these vibrations on the attitude determination algorithms.
For a description of the test bench shown in figure 7.1 and of the effect of the vibrations on the sensors see appendix B.

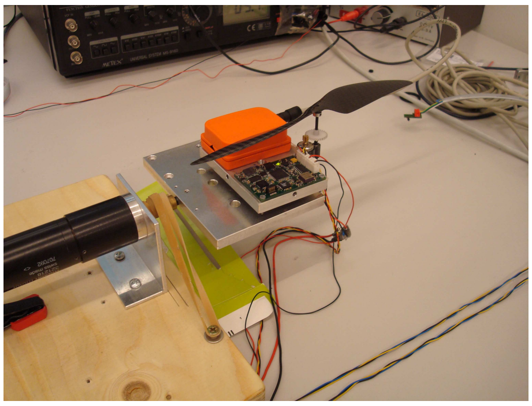

Figure 7.1: Test bench for simulating the vibrations occuring on an airplane.

Performance index	Axis	Kalman Filters		Complementary Filters			
		Filter 6.2.1	Filter 6.2.2	Filter 6.3.1	Filter 6.3.3	Filter 6.3.3	Filter 6.3.4
RMS error [deg]	roll	0.51	0.88	0.59	2.42	1.31	0.48
	pitch	0.71	2.53	0.58	2.24	1.63	0.62
Mean error [deg]	roll	0.41	0.72	0.49	2.1	1.01	0.37
	pitch	0.57	2.22	0.47	1.84	1.26	0.5
Error variance [deg^2]	roll	0.09	0.26	0.1	1.44	0.7	0.09
	pitch	0.18	1.49	0.11	1.66	1.06	0.14

Table 7.6: Performance indexes of the attitude determination algorithms with accelerometer SCA300-E01 for test B and dynamic test conditions.

Performance index	Axis	Kalman Filters		Complementary Filters			
		Filter 6.2.1	Filter 6.2.2	Filter 6.3.1	Filter 6.3.3	Filter 6.3.3	Filter 6.3.4
RMS error [deg]	roll	3.19	2.27	3.38	1.6	3.88	3.12
	pitch	2.23	1.95	2.31	3.24	4.12	3.64
Mean error [deg]	roll	3.13	2.05	3.3	1.28	3.6	3.03
	pitch	1.97	1.61	2.04	2.64	3.42	2.14
Error variance [deg^2]	roll	0.38	0.96	0.49	0.9	2.05	0.55
	pitch	1.08	1.19	1.17	3.57	5.34	2.4

Table 7.7: Performance indexes of the attitude determination algorithms with accelerometer MMA7260QT for test B and dynamic test conditions.

7.2.4 Results

Comparing the mean computational time in table 7.1 and the three performance indexes RMS error, mean error and error variance in the tables 7.2-7.7, the following characteristics stand out:

- Comparing the RMS error, the mean error and the error variance between the filters with accelerometer SCA3000-E01 and those with accelerometer MMA7260QT for the different tests, it clearly stands out that the values for the filters with accelerometer SCA3000-E01 are smaller.
- The analysis of the different sensors with and without vibrations in table B.1 in the appendix B shows, that the noise of accelerometer MMA7260QT is larger than the noise of accelerometer SCA3000-E01.
- The mean computational time for determining an attitude is smaller for the complementary filters than for the EKFs.
- Within the complementary filters, the *quaternion based complementary filter* (filter 6.3.1) and the *gravity direction cosine and earth magnetic field direction cosine estimators complementary filter* (filter 6.3.4) are less influenced by the vibrations (Test B) than the other complementary filters.
- The complementary filters are easier to tune than the EKFs.

As a consequence of the better performance indexes of the filters with accelerometer SCA3000-E01 and the better noise behaviour of this accelerometer, it is selected to be assembled on the final IMU.
And because of their good vibration rejection and their smaller mean computational

time than the *EKF with Euler angles measurements*, the *quaternion based complementary filter* (in the following called *filter 6.3.1*) and the *gravity direction cosine and earth magnetic field direction cosine estimators complementary filter* (in the following called *filter 6.3.4*) are selected to be implemented on the microprocessor. With further tests it must be analysed, which of these two filters is the best.

Chapter 8

Implementation in C30

The *quaternion based complementary filter* and the *gravity direction cosine and earth magnetic field direction cosine estimators complementary* filter are implemented in C30 to be run directly on the microprocessor. Since the microprocessor has limited processing power, simplifications of the filters are preferable to further decrease the mean computational time, so that a higher sampling rate can be achieved.
Thus, section 8.1 presents how the computational time of trigonometric functions, whose computation takes relatively a lot of time, can be reduced on the cost of the accuracy with the help of lookup tables.
And in section 8.2 a simplification of the integration in filter 6.3.4 is introduced.

8.1 Approximation of Trigonometric Functions

According to experience, the computation of a trigonometric function takes a lot of time. This time can be reduced by using lookup tables.
In advance, function values at equally distributed points over the definition domain of the function to be approximated are saved in a lookup table. During run time, only the index of the interval, in which the input value lies, must be determined.
There are three possibilities to increase the accuracy of lookup tables:

- linear interpolation between the lower and upper point [48]
- mean value between the lower and upper point [30]
- taking the nearest point

The following subsections analyse the saved time and the accuracy of lookup tables for the approximation of the trigonometric functions *sine*, *cosine*, *arcus sine* and *arcus tangent2* with the three above mentioned methods.

8.1.1 Sine ($\sin(\alpha)$)

Although the sine function has a definition domain $[-\infty, \infty]$, it can be ensured, that the two implemented filters use the sine function only in the interval $[-\pi, \pi]$ (if for filter 6.3.4 the integration approximation in the next section is used).
Instead of producing a lookup table over this whole intevall, the symmetries of the sine function can be exploited to increase the accuracy of the approximation by unchanging length of the lookup table. So, also algorithms using lookup tables only from 0 to π and from 0 to $\pi/2$ are analysed. The different algorithms are listed in appendix E.1.
To analyse the computation time, the time for computing 1000 sines in a `for`-loop

is measured and shown in table 8.1 for the different approximation algorithms. For this measurement and the measurements in the following subsections, the processor speed is set to 20MIPS.

Algorithm	**Elapsed Time [ms]**	**Reduction**	**Cycles per Computation**
sin	126	-	~2520
sinf[1]	126	-	~2520
Lookup table $[-\pi, \pi]$, linear interpolation	60	52.4%	~1200
Lookup table $[-\pi, \pi]$, mean value of two nearest points	42	66.7%	~840
Lookup table $[-\pi, \pi]$, nearest point	21	83.3%	~420
Lookup table $[0, \pi]$, linear interpolation	64	49.2%	~1280
Lookup table $[0, \pi]$, mean value of two nearest points	47	62.7%	~940
Lookup table $[0, \pi]$, nearest point	34	73.0%	~680
Lookup table $[0, \pi/2]$, linear interpolation	75	40.5%	~1500
Lookup table $[0, \pi/2]$, mean value of two nearest points	58	54.0%	~1160
Lookup table $[0, \pi/2]$, nearest point	44	65.1%	~880

Table 8.1: Time consumed by the computation of 1000 sines (Processor speed: 20MIPS).

The effect of the lookup table length on the accuracy of the different approximation algorithms is shown in figure 8.1. This figure shows the maximum error made, if the sine is computed with one of the algorithms instead of the original sine function in the interval $[-\pi, \pi]$.
Because of its fastest computation time, the nearest point algorithm with lookup table from $-\pi$ to π is selected to be implemented. As appriopriate length of the lookup table is chosen 1500 data points.

8.1.2 Cosine $(\cos(\alpha))$

Also for the cosine function it can be verified, that an approximation defined for the interval $[-\pi, \pi]$ is sufficient for the two filters (if for filter 6.3.4 the integration approximation in the next section is used).
Because of the symmetry of the cosine function, algorithms using lookup tables from 0 to π are investigated, too. The algorithms are listed in appendix E.2.
Table 8.2 presents the time consumed by each of this approximations for the computation of 1000 cosines.
The maximum error made in the interval $[-\pi, \pi]$ depending on the lookup table length is shown in figure 8.2.
For the cosine approximation also the simple nearest point algorithm with lookup table from $-\pi$ to π is selected with the same table length as for the sine lookup table.

[1] *sinf* is the sine function especially suited for float numbers

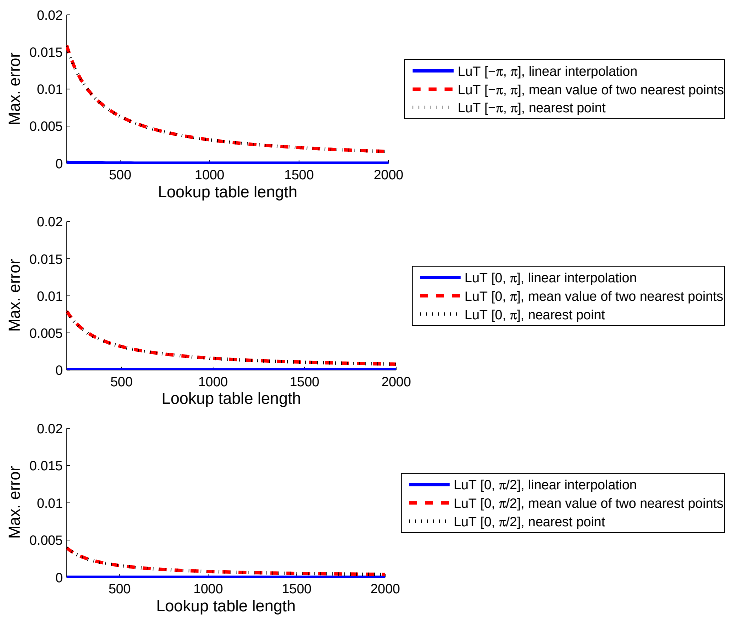

Figure 8.1: Maximum errors for the computation of sine with lookup tables.

8.1.3 Arcus Sine ($\arcsin(x)$)

The arcus sine function is defined in the domain $[-1,\ 1]$. But again, a symmetry can be exploited to increase the accuracy of the approximation with a lookup table. All the algorithms analysed for the approximation of the arcus sine are shown in appendix E.3.
The elapsed times for computing 1000 arcus sines in a `for`-loop are listed in table 8.3.
The dependence between the maximum error and the lookup table length is presented in figure 8.3. Since the arcus sine function is higly nonlinear on both sides of the definition domain, the largest errors have to be expected near to 1 and -1. Figure 8.4 shows the error distribution over the definition domain for the algorithm with lookup table over the whole definition domain [-1, 1] (1500 data points) and linear interpolation. It can be clearly seen that the errors near 1 and -1 are much larger than the errors in the rest of the definition domain.
So, the mean error over the definition domain, shown in figure 8.5 is a better indicator for an appriopriate choice of the length of the lookup table.
As in table 8.3 can be seen, the measured times for the approximations with lookup table from 0 to π are smaller than for the approximations with lookup table over the whole definition domain. This effect is illogical since the approximation with the smaller lookup table has to check a *if-else* condition, and accordingly can not be explained.
Thus, anyway the nearest point algorithm with lookup table over the whole definition domain is selected. As table length, the maximum length of 5100 data points allowed by the C30 compiler is used.

8.1.4 Arcus Tangent2 (arctan2(y, x))

The arcus tangent2 function is an enhancement of the arcus tangent function, i.e., while the result of arcus tangent is limited to the inteval $[-\pi/2,\ \pi/2]$, arcus tangent2 uses the signs of y and x to determine the quadrant. Thus, the result lies in

Algorithm	**Elapsed Time [ms]**	**Reduction**	**Cycles per Computation**
cos	147	-	~2940
cosf	147	-	~2940
Lookup table $[-\pi, \pi]$, linear interpolation	58	60.5%	~1160
Lookup table $[-\pi, \pi]$, mean value of two nearest points	41	72.1%	~820
Lookup table $[-\pi, \pi]$, nearest point	21	85.7%	~420
Lookup table $[0, \pi]$, linear interpolation	61	58.5%	~1220
Lookup table $[0, \pi]$, mean value of two nearest points	46	68.7%	~920
Lookup table $[0, \pi]$, nearest point	30	79.6%	~600

Table 8.2: Time consumed by the computation of 1000 cosines (Processor speed: 20 Mips).

Algorithm	**Elapsed Time [ms]**	**Reduction**	**Cycles per Computation**
asin	119	-	~2380
asinf	119	-	~2380
Lookup table $[-\pi, \pi]$, linear interpolation	59	50.4%	~1180
Lookup table $[-\pi, \pi]$, mean value of two nearest points	41	65.5%	~820
Lookup table $[-\pi, \pi]$, nearest point	21	82.4%	~420
Lookup table $[0, \pi]$, linear interpolation	39	67.2%	~780
Lookup table $[0, \pi]$, mean value of two nearest points	22	81.5%	~440
Lookup table $[0, \pi]$, nearest point	14	88.2%	~280

Table 8.3: Time consumed by the computation of 1000 arcus sines (Processor speed: 20 Mips).

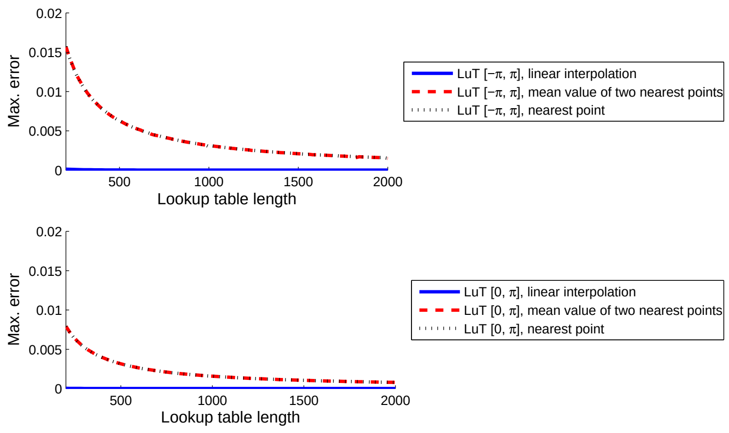

Figure 8.2: Maximum errors for the computation of cosine with lookup tables.

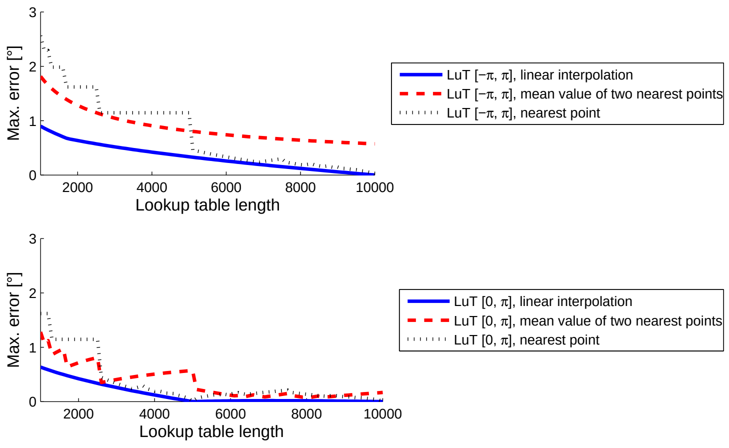

Figure 8.3: Maximum errors for the computation of arcus sine with lookup tables.

the interval $[-\pi, \pi]$ [23].
Hence, an indirect method of approximating the arcus tangent2 function is to approximate firstly the arcus tangent function, and then use *if-else* conditions to determine the quadrant. Since the definition domain of the arcus tangent function is $[-\infty, \infty]$, a lookup table is not suited to directly approximate this function. But this problem can be overcome by using the relation [8]

$$\arctan(x) = \arcsin\left(\frac{x}{\sqrt{1+x^2}}\right) \tag{8.1}$$

So, only the term in the brackets on the right side of equation (8.1) must be computed and then put into the same lookup table as used for the arcus sine function. Consequently, the accuracy of this arcustangent approximation equals the accuracy of the arcus sine approximation.
There also exists an approximation of the arcus tangent function not using a lookup table. The largest error made by this approximation is $\approx 0.27\,°$ [47] for input values near to zero. Since the arcus tangent2 function is mainly used for such input values (see for example equation (6.3)), this approximation is not well suited.
Also an direct approximation not making the indirection over the computation of the arcus tangent is found [11], but the largest error made by this algorithm is $\approx 4.07\,°$.

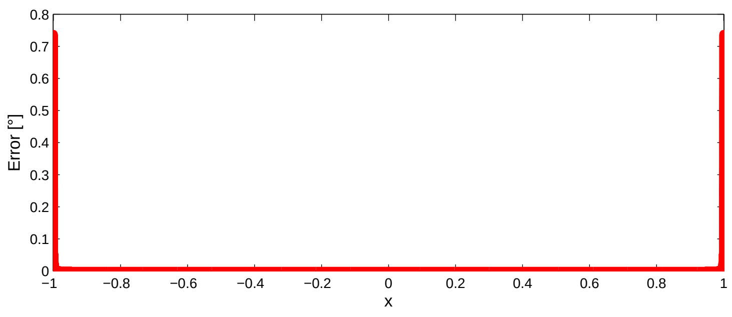

Figure 8.4: Error distribution over the definition domain for computing arcus sine with lookup table.

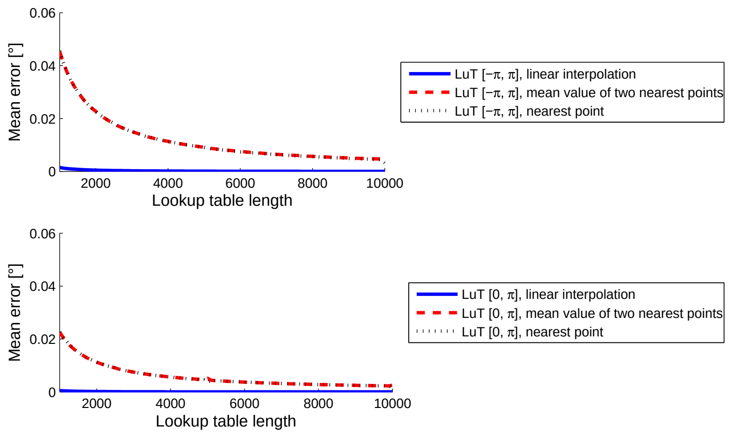

Figure 8.5: Mean errors for the computation of arcus sine with lookup tables.

The above presented algorithms are described in appendix E.4 and their computation times for 1000 arcus tangent2 are listed in table 8.4.

8.2 Approximation of the Exact Integration in Filter 6.3.4

In the *gravity direction cosine and earth magnetic field direction cosine estimators complementary filter* in subsection 6.3.4, exact integration by Rodrique's formula is used to preserve unity length of $\hat{x}$ and $\hat{z}$. Instead Euler integration can be used

$$\hat{x}^*_{k+1} = \hat{x}_k + T \cdot \dot{\hat{x}} \tag{8.2}$$

$$\hat{z}^*_{k+1} = \hat{z}_k + T \cdot \dot{\hat{z}} \tag{8.3}$$

The derivatives $\dot{\hat{x}}$ and $\dot{\hat{z}}$ are computed with equations (6.54) and (6.56) respectively. The unity length of $\hat{x}$ and $\hat{z}$ is then preserved by deviding $\hat{x}^*$ and $\hat{z}^*$ by their norms

$$\hat{x}_{k+1} = \frac{\hat{x}^*_{k+1}}{|\hat{x}^*_{k+1}|} \tag{8.4}$$

$$\hat{z}_{k+1} = \frac{\hat{z}^*_{k+1}}{|\hat{z}^*_{k+1}|} \tag{8.5}$$

Algorithm	Elapsed Time [ms]	Reduction	Cycles per Computation
atan2f	170	-	~3400
atanf	142	-	~2840
Indirect atan2 with asin lookup table	105	38.2%	~2100
Indirect atan2 without lookup table	73	57.1%	~1460
Direct atan2	80	52.9%	~1600

Table 8.4: Time consumed by the computation of 1000 arcus tangent2 (Processor speed: 20 Mips).

8.3 Effects of the Function Approximations and Integration Approximation

For the analysis of the effect of the trigonometric function approximations and of the approximation of the integration in filter 6.3.4 on the filters' performance, the performance indexes for the same data sets as in subsection 7.2.3 are evaluated for the new filters. Compared to the original values of the performance indexes no significant change can be recognized, so that the new filters can be stated to be equaly performant.

8.4 Time Consumptions

For making a decision concerning the selection of the filter, it is interesting to know the time consumption of both filters, and the effect of the trigonometric function approximations and of the approximation of the integration step in filter 6.3.4 on these times.

8.4.1 Taking Measurements

For taking the measurements from accelerometer SCA3000-E01, from gyroscopes DXRS300 and IDG-300 and from magnetometer HMC6052 the microprocessor takes 1.24ms. Of course, this time consumption is the same for both filters and can not be reduced.

8.4.2 Quaternion Based Complementary Filter

Filter

The time consumption of the *quaternion based complementary filter* with and without trigonometric function approximations is presented in table 8.5. With the approximations, 1.4ms can be saved, which corresponds to 37.84% of the total execution time of this filter.

	Time Consumption
Original Filter	3.7ms
Filter with lookup tables	2.3ms

Table 8.5: Time consumptions of the original filter 6.3.1 and of the filter with approximations.

Conversion to Euler Angles

Since this filter represents the attitude with quaternions and the IMU should output Euler angles, a conversion from quaternion to Euler angles according equation (3.18) is applied. The time consumption of the conversion with and without trigonometric function approximations is shown in table 8.6

	Time Consumption
Original conversion	0.57ms
Conversion with lookup tables	0.39ms

Table 8.6: Time consumptions of the original conversion of filter 6.3.1 and of the conversion with approximations.

8.4.3 Gravity Direction Cosine and Earth Magnetic Field Direction Cosine Estimators

Filter

Table 8.7 presents the time consumption of the *gravity direction cosine and earth magnetic field direction cosine estimators complementary filter* with and without approximations. For a better understanding, the effects of the trigonometric function approximations and the integration approximation on the time consumption are analysed both, separately and together.
Finally, the computational time of this filter is reduced by 44.18%.

	Time Consumption
Original Filter	4.3ms
Filter with integration approximation	2.8ms
Filter with lookup tables	3.4ms
Filter with lookup tables and integration approximation	2.4ms

Table 8.7: Time consumptions of the original filter 6.3.4 and of the filter with approximations.

Conversion to Euler Angles

This filter needs also an Euler angles calculation since the attitude is only indirectly represented by the estimation of the gravity and the magnetic field direction in the body frame. This conversion is made using equations (6.3) and (6.6).
Table 8.8 lists the times consumed by the conversion with and without trigonometric function approximations.

	Time Consumption
Original conversion	1.1ms
Conversion with lookup tables	0.45ms

Table 8.8: Time consumptions of the original conversion of filter 6.3.4 and of the conversion with approximations.

	Filter 6.2.1	**Filter 6.2.4**
Reading measurements	1.24ms	1.24ms
Filter	2.3ms	2.4ms
Conversion to Euler angles	0.39ms	0.45ms
Total	3.93ms	4.09ms

Table 8.9: Analysis of the mean computational time of the attitude determination algorithms on the micro processor.

8.4.4 Summary

Table 8.9 compares the time consumption of the different parts of the attitude calculation for both filters. It follows that both filters would allow a maximum sampling rate of 200Hz with a microprocessor speed of 20MIPS. But since the targeted sampling rate of the filters is 100Hz, the microprocessor speed can be reduced to 10MIPS, which also results in a reduced power consumption.

Chapter 9

Final ASL IMU

Based on the inquiries made on the sensor board and described in the previous chapters, the final ASL IMU is designed.

9.1 Coordinate Frames

The final IMU is designed using the in the previous chapters selected sensors:

- SCA3000-E01
- HMC6052
- ADXRS300 & IDG-300

So, a size of 40 × 25mm and a weight of 5.24g is finally achieved (see [4]).
Figure 9.1 shows the IMU body frame and the coordinate axes of each sensor, whereat in figure 9.2 the size of the sensor board and of the IMU are compared.

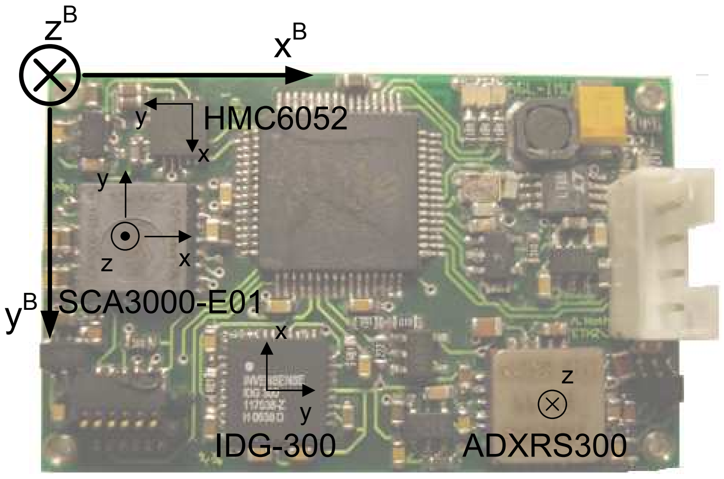

Figure 9.1: The ASL IMU.

9.2 Calibration

Each sensors' error model is calibrated as described in chapter 5.

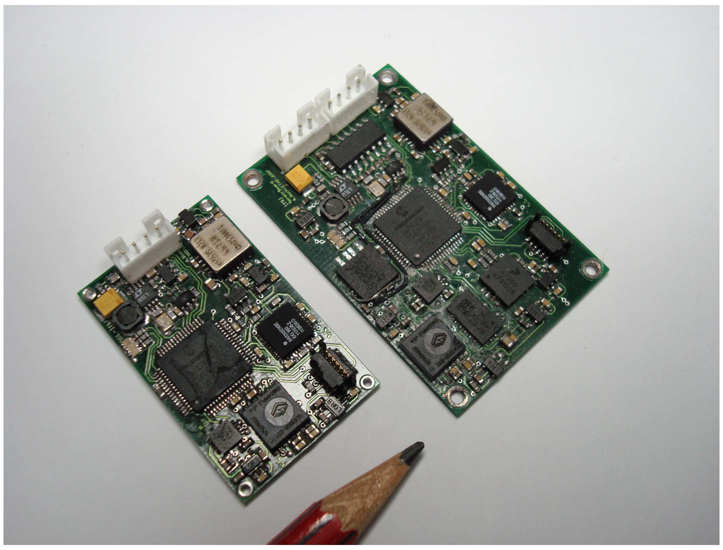

Figure 9.2: The ASL IMU and the sensor board.

9.3 IMU Specifications

Table 9.1 gives an overview of the specifications of the ASL IMU.

Parameter Name	Value
Size	40×25mm
Weight	5.24g
V_{supply}	5.2 - 11V (max. 2 Li-Ion or Li-Po cells)
Interface	RS-232
Baudrate	115200
Output data	Euler angles (Tait-Bryan angles) [rad]

Table 9.1: ASL IMU specifications.

9.4 Communication Protocol

For the final IMU the same communication protocol as for the Xsens MTx [50] IMU is used. This choice is made, because at ETH Xsens products are employed in several projects, and thus, they can be easily substituted by the developed IMU if using the same communication protocol.
The standard MTx message contains the fields shown in figure 9.3.

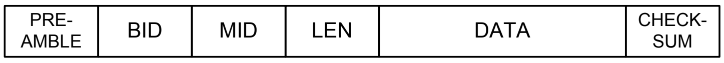

Figure 9.3: The IMU message format.

Preamble (1 Byte)

The preamble indicates the start of a message and has the value 250 (0xFA).

Bus identifier or address BID (1 Byte)

This message part has the value 255 (0xFF) and is used to indicat a ”master device”.

Message identifier MID (1 Byte)

The MID identifies the kind of message. For a message sending attitude data this field has the value 50 (0x32)

Message length LEN (1 Byte)

The message length specifies the number of bytes in the data field and has in this case the value 12.

Data (12 Bytes)

This field contains the attitude data in the form of the pitch, roll and yaw angle.

Checksum (1 Byte)

The checksum byte is computed by summing up all bytes of the message excluding the preamble and the checksum, and then negating the sum.
The message can be validated by summing up all bytes of the message excluding the preamble. If the lower byte value of the sum equals zero, the message is valid.

9.5 Power consumption

Table 9.2 gives an overview of the power consumption of the sensor board and the IMU for the different microprocessor speeds.

	Current [mA]	Voltage [V]	Power [mW]
Sensor Board dsPIC off	44	5.15	226.6
Sensor Board 20MIPS	74	5.12	378.88
Sensor Board 10MIPS	61	5.13	312.93
IMU dsPIC off	34	5.16	175.44
IMU 20MIPS	64	5.12	327.68
IMU 10MIPS	51	5.14	262.14

Table 9.2: Sensor board and IMU power consumption.

Chapter 10

Final Tests

To specify the performance of the two remaining and implemented filters, and to make a final decision regarding the filter, several tests were conducted.
First, the performance of the two filters is tested with two different position profiles on the test bench without disturbances (test A). In order to test the reactivity of the filters, the first position profile contains fast movements. The second position profile contains then both, small and large position steps. This position profile is chosen such that it can also be applied on the test bench with airplane vibrations (test B) and the test bench with helicopter vibrations (test C). So, the effect of the different vibrations can be analyzed.
In the last test (test D), a robot arm is used to emulate the small movements of a UAV helicopter when hovering. This allows the evaluation of the filters' performance in the presence of linear accelerations.

10.1 Test A

First, the performance of the two filters is tested with two different position profiles on the test bench from test A. Again, both position profiles are applied on the roll and pitch axis separately.
The first position profile is used to test the reactivity. This position profile applied on the roll axis is shown in figure 10.1. For comparison purposes the values from the Xsens MTx IMU are also shown.
As can be seen, the fast movements of the roll axis have an effect on the estimation of the pitch angle. Vice versa, if the position profile is applied on the pitch axis, the same effect on the roll axis can be observed. Of course, this can be to a certain part because of the not perfect horizontal alignment of the test bench. But this can not be the only reason for such a big deflexion. At the moment there does not exist an explanation for this effect, so further research should be done to find the cause of this deflexion.
In table 10.1 the performance indexes of the two filters for this position profile, applied on the roll and the pitch axis separately, are presented. From these values it can be concluded, that the filter 6.3.1 better determines the angle of the axis that position profile is applied on, than the filter 6.3.4. But on the other hand, the side effect on the other axis is smaller for filter 6.3.4.
The second position profile is chosen such that it can be also used for test B and test C. This allows the analysis of the effect of the different vibrations on the attitude determination of the two filters. The profile is shown in figure 10.2. In table 10.2 the performance indexes are shown, such that they can be compared later on with the performance indexes from test B and test C. Already, it can be seen that the

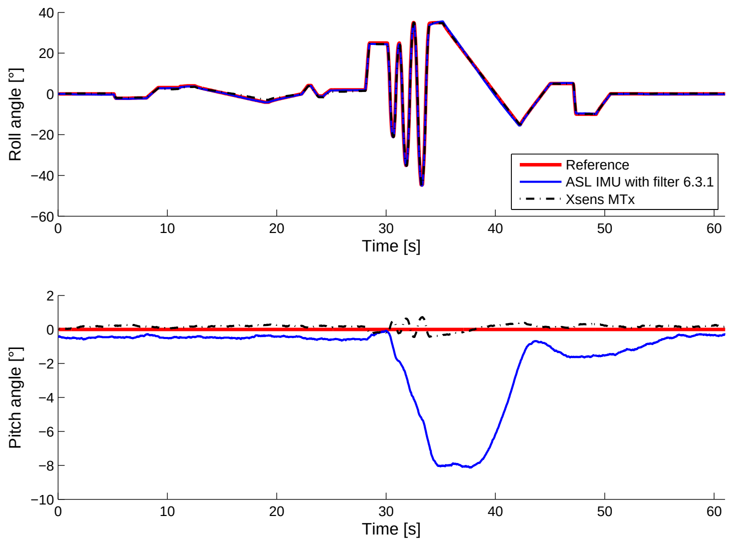

Figure 10.1: Position profile with fast movements. The fast movements of the roll axis have a cross effect on the attitude estimation of the pitch axis as described in the text.

		Position profile on roll axis		Position profile on pitch axis	
Performance index	Axis	Filter 6.3.1	Filter 6.3.4	Filter 6.3.1	Filter 6.3.4
RMS error [deg]	roll	0.38	1.3	1.54	0.49
	pitch	2.84	0.73	0.52	0.89
Mean error [deg]	roll	0.29	0.83	0.96	0.46
	pitch	1.67	0.63	0.49	0.69
Error variance [deg^2]	roll	0.06	1	1.44	0.03
	pitch	5.31	0.14	0.03	0.32

Table 10.1: Performance indexes of the two filters for test A and the first position profile.

filter 6.3.1 still better determines the angle of the axis the position profile is applied on. But it can be also seen that the side effect observed with the first position profile is not so distinctive with the second profile any more.

		Position profile on roll axis		Position profile on pitch axis	
Performance index	Axis	Filter 6.3.1	Filter 6.3.4	Filter 6.3.1	Filter 6.3.4
RMS error [deg]	roll	0.29	0.68	0.39	0.41
	pitch	0.73	0.48	0.44	0.68
Mean error [deg]	roll	0.24	0.52	0.36	0.36
	pitch	0.60	0.43	0.43	0.58
Error variance [deg^2]	roll	0.03	0.19	0.02	0.03
	pitch	0.17	0.05	0.01	0.13

Table 10.2: Performance indexes of the two filters for test A and the second position profile.

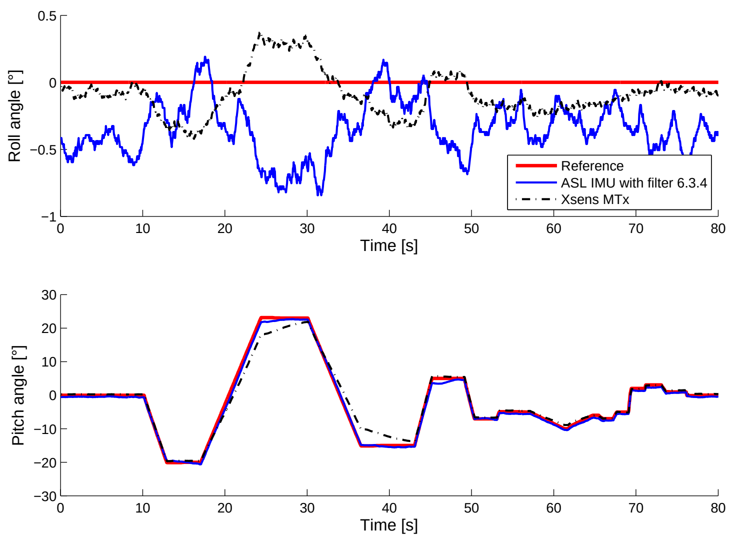

Figure 10.2: Second position profile.

10.2 Test B

The second position profile from test A is also applied in test B. Table 10.3 presents the performance indexes.
Comparing with the performance indexes from test A in table 10.2, it follows that the performance indexes are worse with the vibrations, but the influence of the vibrations is not significant.

		Position profile on roll axis		Position profile on pitch axis	
Performance index	Axis	Filter 6.3.1	Filter 6.3.4	Filter 6.3.1	Filter 6.3.4
RMS error [deg]	roll	0.40	0.59	0.67	0.68
	pitch	0.66	0.67	0.85	0.79
Mean error [deg]	roll	0.34	0.51	0.62	0.66
	pitch	0.64	0.66	0.83	0.73
Error variance [deg^2]	roll	0.04	0.09	0.06	0.03
	pitch	0.03	0.02	0.03	0.10

Table 10.3: Performance indexes of the two filters for test B and the second position profile.

10.3 Test C

As the ASL IMU is also planned to be employed on MAV helicopters, the effect of vibrations occuring on MAV helicopters on the attitude determination of the IMU is also tested. For a describtion of the test bench with MAV helicopter vibrations see appendix C.
Again, the performance indexes for the second position profile are shown in table 10.4. It must be pointed out that this test bench is not very sophisticated, e.g., the alignment of the IMU is not easy (see appendix C). So, the performance indexes are only calculated for the axes the position profile is applied on and should not

be seen as reference values. This test just shows, that the algorithms are able to roughly determine the attitude in the presence of hard vibrations as they occure on an helicopter. Figure 10.3 shows for example the result of filter 6.3.4.

		Position profile on roll axis		Position profile on pitch axis	
Performance index	Axis	Filter 6.3.1	Filter 6.3.4	Filter 6.3.1	Filter 6.3.4
RMS error [deg]	roll	3	1.58	x	x
	pitch	x	x	1.89	2.88
Mean error [deg]	roll	2.38	1.24	x	x
	pitch	x	x	1.48	2.21
Error variance [deg^2]	roll	3.33	0.95	x	x
	pitch	x	x	1.38	3.44

Table 10.4: Performance indexes of the two filters for test C and the second position profile.

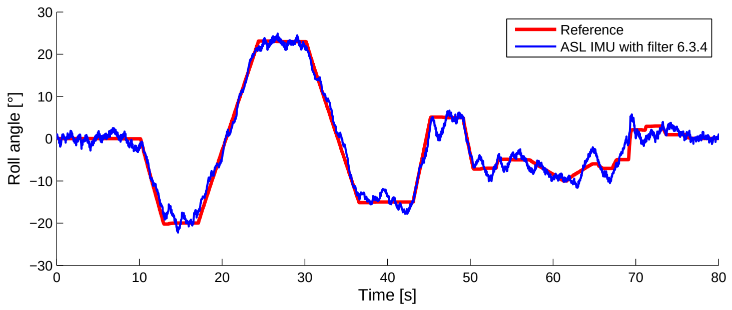

Figure 10.3: Second position profile with helicopter disturbances.

10.4 Test D

So far, only the effect of two different kinds of vibrations have been tested. Thus, tests with a robot arm are made, intending to introduce linear accelerations. Linear accelerations distort the measurement of the gravity vector, and hence, can have a big effect on the estimated attitude according to their amplitude.
The robot is programmed, such that during the tests the motion of an MAV helicopter when hovering is simulated. For analyzing how large accelerations are tolerated by the attitude determination algorithms, four different tests with increasing speeds are made, whereas the speed from one test to the next faster test is always doubled.
For a test description and a classification of the four tests, see appendix D.
The measurements from the IMU and the robot's arm position are recorded by two different computers and consequently, the two measurement sets do not have the same starting time. So, one measurement set has to be shifted temporally, which is done by hand.
Since only two to three measurements per second of the robot's arm attitude are made by the robot itself, these measurements are linearly interpolated for the computation of the performance indexes. Of course, this method implies that errors are made, since the real robot's arm attitude is not known between the measurements.
The performance indexes for the four tests with increasing speed are presented in

tables 10.5 and 10.6.
It must be pointed out, that for test D the values from the sensors on the IMU are recorded and that the attitude is computed afterwards off-line in MATLAB®.

Performance index	Axis	1 × speed		2 × speed	
		Filter 6.3.1	Filter 6.3.4	Filter 6.3.1	Filter 6.3.4
RMS error [deg]	roll	0.85	0.74	1.06	1.32
	pitch	1.23	1.17	1.42	1.51
Mean error [deg]	roll	0.57	0.55	0.78	1.05
	pitch	1.03	0.95	1.24	1.19
Error variance [deg^2]	roll	0.41	0.24	0.52	0.63
	pitch	0.45	0.47	0.47	0.86

Table 10.5: Performance indexes of the two filters for test D with normal speed and doubled speed.

Performance index	Axis	4 × speed		8 × speed	
		Filter 6.3.1	Filter 6.3.4	Filter 6.3.1	Filter 6.3.4
RMS error [deg]	roll	1.95	2.16	2.10	2.49
	pitch	2.44	2.77	3.91	2.92
Mean error [deg]	roll	1.58	1.71	1.70	1.92
	pitch	2.01	2.19	3.50	2.32
Error variance [deg^2]	roll	1.32	1.74	1.51	2.53
	pitch	1.88	2.90	3.02	3.16

Table 10.6: Performance indexes of the two filters for test D with fourfold and eightfold speed.

From the performance indexes in table 10.5 it can be seen, that both filters are able to track the attitude for normal and doubled speed. But for fourfold and eightfold speed, both filters start to show pain to estimate the angles of the roll and pitch axes.
So far, only numbers have been presented, which do not vividly display the performance of the filters. Thus, figures 10.4 - 10.7 show some details of all four tests of the robot arm test. For the reference, i.e., the robot's arm attitude measured by the robot itself, only the points where measurements are taken, are plotted, as between these measurements the robot's arm attitude is not known. It is worth to be mentioned that the alignment of the MTx IMU has not been calibrated.

10.5 Conclusion

From the final tests it is hard to say which one of the two complementary filters is the best. E.g., one filter shows some advantages in one test, whereas the other filter shows advantages in another test. I.e., from the values of the performance indexes presented above it is not possible to make a final decision, especially since the values of the performance indexes of the two filters do not differ significantly.
But by the author's impression, the *quaternion based complementary filter* is slightly better.

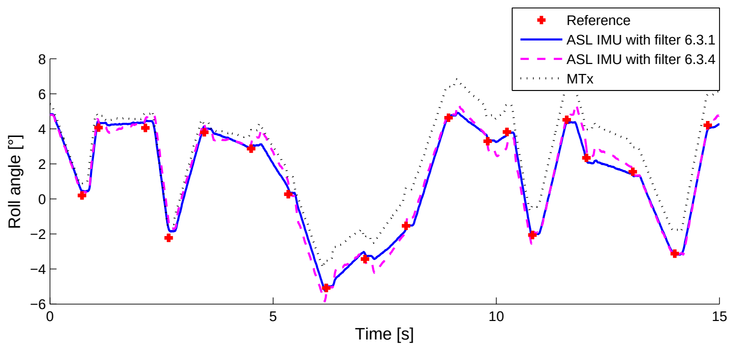

Figure 10.4: Test D with original speed.

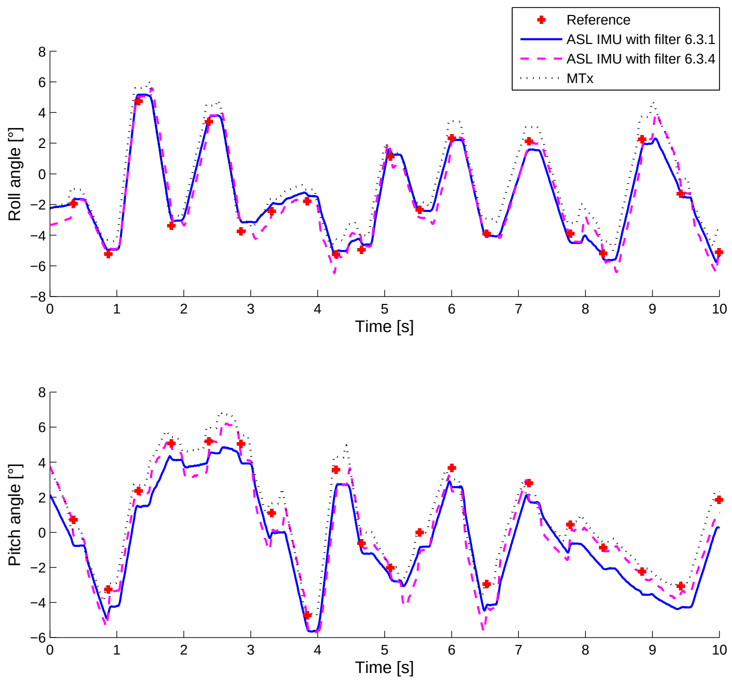

Figure 10.5: Test D with doubled speed.

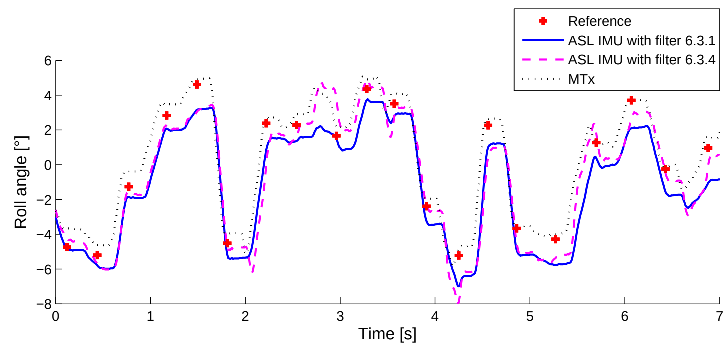

Figure 10.6: Test D with fourfold speed.

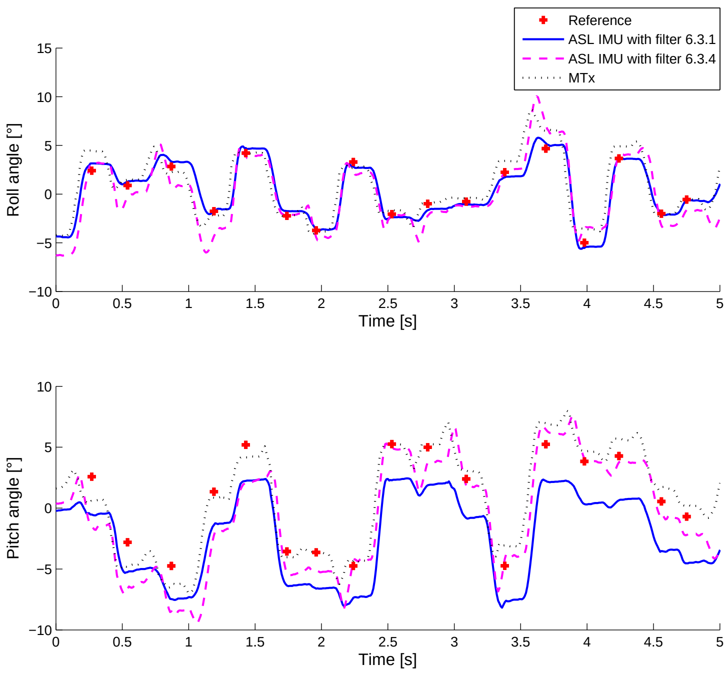

Figure 10.7: Test D with eightfold speed.

Chapter 11

Conclusion

In this thesis, two Extended Kalman filters and four complementary filters for the attitude determination of UAVs and MAVs with an IMU consisting of low cost MEMS sensors are presented. As the filters do not depend on the application, they can also be used in other application area, e.g. walking robots.
By experiments it is concluded that a *quaternion based complementary filter* and a *gravity direction cosine and earth magnetic field direction cosine estimators complementary filter* are best suited to be implemented on a microprocessor with limited processing power.
Approximating trigonometric functions with lookup tables and approximating the exact integration step of the *gravity direction cosine and earth magnetic field direction cosine estimators complementary filter* results in a significant reduction of the filters' computation time. Tests show that the filters' perfomance is not decreased by this approximations.
This time saving allows to reduce the microprocessor speed, which results in a reduced power consumption.
Final tests show, that both filters are able to track the attitude despite the low cost sensors and different disturbances.
Further works could be:

Sensor calibration If larger quantities of the ASL IMU should be produced, an easy to make sensor calibration procedure should be developed.

3-axis magnetometer It is assumed that in the future an appropriate 3-axis magnetometer will be available. Then, the 2-axis magnetometer should be replaced to increase the yaw angle accuracy and to reduce the filters' computation time.

Test bench If the development of an IMU will be continued in the future, a new test bench for simulating different disturbances and testing the attitude determination algorithms should be developed. Especially a magnetometer test bench should be considered.

Algorithm improvements The reason for the cross effect described in section 10.1 should be found. Second, the filters do not provide accurate values for a few seconds at start-up, since the gyroscope bias estimation must settle first.

Measurements synchronisation As long as the measurement synchronisation problem is not solved, an exact comparison is not possible. Particularly the determination of a possible time delay of the attitude determination algorithms is not possible at the moment.

Communication protocol A communication protocol, symplifying the setting of IMU parameters such as the sending rate of attiude data, could be developed.

Appendix A

Remarks

A.1 Sensors

This section contains remarks on the properties of the sensors, which were uncovered during the thesis.
Table A.1 shows for each sensor the power consumption and the processing time for the microprocessor for taking a measurement.

Typ	Processing Time	Power consumption
SCA3000-E01	$450\mu s$	0.495 mW[1]
MMA7260QT	$385\mu s$	1.65 mW[1]
LIS3LV02DL	$445\mu s$	2.64mW[2]
ADXRS300	$150\mu s$	30mW[3]
IDG-300	$295\mu s$	31.35mW[2]
HMC6052	$295\mu s$	29.7mW[2]
HMC6352	$593\mu s$[5]	3mW[4]

Table A.1: Microprocessor's processing time for taking measurements from a sensor (processor speed: 20MIPS) and power consumption of each sensor.

A.1.1 Accelerometer SCA300-E01

The z axis of the accelerometer SCA300-E01 assembled on the sensor board outputs the gravitational acceleration as specified in the datasheet if it is vertical. However, the x and the y axes output the gravitational acceleration 180 ° inverted with respect to the data sheet if they are vertical. This problem is handeled by inverting the x and the y axes by 180 °.

A.1.2 Accelerometer LIS3LV02DL

The first accelerometer LIS3LV02DL on the sensor board was exchanged, because it outputed non-acceptable measurements as the result of the calibration in figure

[1]Typical value for $V_{dd} = 3.3V$
[2]Maximum value for $V_{dd} = 3.3V$
[3]Typical value for $V_{dd} = 5.0V$
[4]Typical value for $V_{dd} = 3.0V$
[5]Query Mode

A.1 shows.

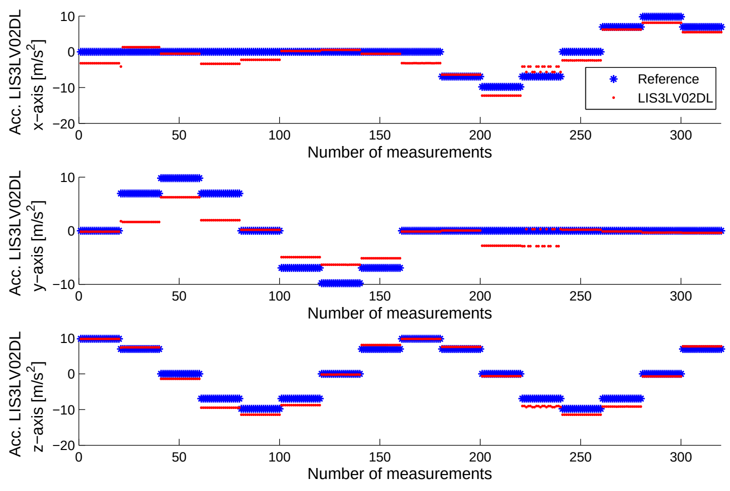

Figure A.1: Calibration of the first accelerometer LIS3LV02DL on the sensor board (in body frame coordinates).

A.1.3 Magnetometer HMC6352

The magnetometer HMC6352 has five ouput data modes:

- Heading Mode
- Raw Magnetometer X Mode
- Raw Magnetometer Y Mode
- (Calibrated) Magnetometer X Mode
- (Calibrated) Magnetometer Y Mode

There does not exist a mode which outputs both, the calibrated x and y axis magnetometer measurements.
An idea to take both measurements during one sampling period is to acquire first the measurement of one axis and then to switch the output data mode to take the measurement of the other axis. But the problem then is, that it must be included a delay of about 43ms between the two measurements.

A.2 Taking Measurements with Computer

When taking the measurement sets from the testbench for the tests A, B and C for offline processing on the computer, first the problem encoutered, that the filters detected a movement before the reference from the encoder detected it.
This is due to converter hardware and buffers between the sensors and the computer, which insert an unknown and probably varying time delay.
Listing A.1 shows the MATLAB® code providing the best results when taking the measurements from the sensor board with 25Hz.
For the final tests described in section 10.1-10.3 the lines 15-25 in listing A.1 are replaced by the lines in listing A.2
If simultaneously position commands for the motor are send, this method works

```matlab
first_execution = 1;

% start motor
EPOS = StartEpos(8,1,1);
% open port for MTx
h.MT_SetCOMPort(5);
% open port for sensor board / IMU
s = serial('COM7','BaudRate',115200, 'DataBits', 8, ...
 'FlowControl', 'none', 'StopBits', 1, 'Parity', 'none');

% start processing data MTx
h.MT_StartProcess;
tic;
while toc < recording_time
        if first_execution
                % start sending data from sensor board / IMU
                fwrite(s, 's', 'uchar');
        end
        % get data from sensor board
        port_read = fread(s, 14, 'uchar');

        if first_execution
                tic
                first_execution = 0;
        end

        % check if preamble and checksum are correct
        if ~check_messageport_read)
          SB_temp = [SB_temp; port_read'];
        else
                disp('Error␣occured␣Sensor␣Board');
                break
        end

        % get reference (EPOS) data
        time = [time; toc];
        a = double(GetPosition(EPOS));
        reference = [reference; a*360/(4*256*51) 0 0];

        % get data from MTx
        [d,status,N]=MT_get_data(h, Channels);
        if status ~= 0
                time2 = [time2; toc];
                MT = [MT; d(end,:)];
        end
end

MT_StopProcess(h) % close MTx
fwrite(s, 't', 'uchar'); % close Sensor Board
StopEpos(EPOS); % stop motor
```

Listing A.1: Code for taking measurements.

```
    if first_execution
        fwrite(s1, 's', 'uchar');
        fwrite(s2, 's', 'uchar');
        tic
        first_execution = 0;
    end
    % Get data from Sensor Board
    port_read1 = fread(s1, 14, 'uchar');
    % Get data from IMU
    port_read2 = fread(s2, 14, 'uchar');
```

Listing A.2: Code for taking measurements in the final tests.

without problems for a sampling rate of $25Hz$. But for a sampling rate of $100Hz$ the sending of position commands takes to much time. Then the program must be split on two computers. One computer sends the position commands and records the encoder values, whereat the other computer records the measurements from the sensor board or IMU. For comparing the measurements and the reference, one measurement set must then be shifted temporally.

A.3 Calibration Tool

For the calibration of each sensor's error model and for the Kalman filter tuning, an adapted version of a calibration tool developed during the semester thesis "Mean-Value Model of a Modern Diesel Engine" is used. This tool uses a graphical user interphase (GUI) to lead the user through the process of minimizing a cost function with the `fminsearch` function provided by MATLAB®.

Appendix B

Test B: Airplane Vibrations Test Bench

This appendix presents the test bench used to simulate the vibrations occuring on an airplane.
The vibrations are produced by a motor with propeller mounted on the same plate as the sensor board. To make sure that the vibrations are not totally damped by the test bench, the sensor board and the motor are not directly mounted on the plate which is on the motor's shaft. Instead they are mounted on a second plate which is on the first plate, but the fixation between the second plate and the first plate is not inflexible (see figure B.1).

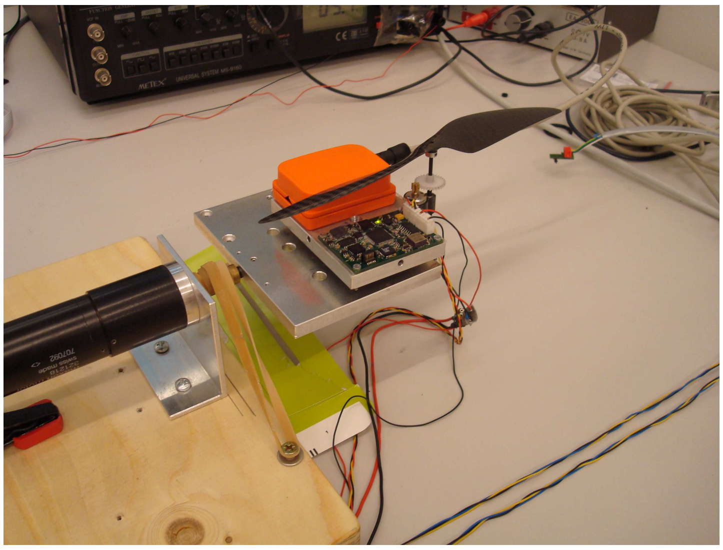

Figure B.1: Test bench for simulating the vibrations occuring on an airplane.

The propulsion system from *WES Technik* consists of a Micro DC 5-2.4 motor with an additional 1:8 gear and a Carbon 20/10cm propeller. During the experiments the motor is powered with 2.7V.
Figures B.2, B.3 and B.4 show the effect of the vibrations on the measurement readings of the accelerometers SCA3000-E01 and MMA7260QT, the gyroscopes ADXRS300 and IDG-300 and the magnetometer HMC6052 respectively. For comparison purposes the calibrated sensor values from the Xsens MTx IMU are also

shown.
As can be seen, the amplitude of the vibrations is not the same in all directions. Keep in mind that the sensitivity axes of the different sensors do not point in the same direction and that they are transformed to the body frame coordinate system (see chapter 5). This is the reason why the two accelerometers do not show the biggest effect of the vibrations on the same axis.

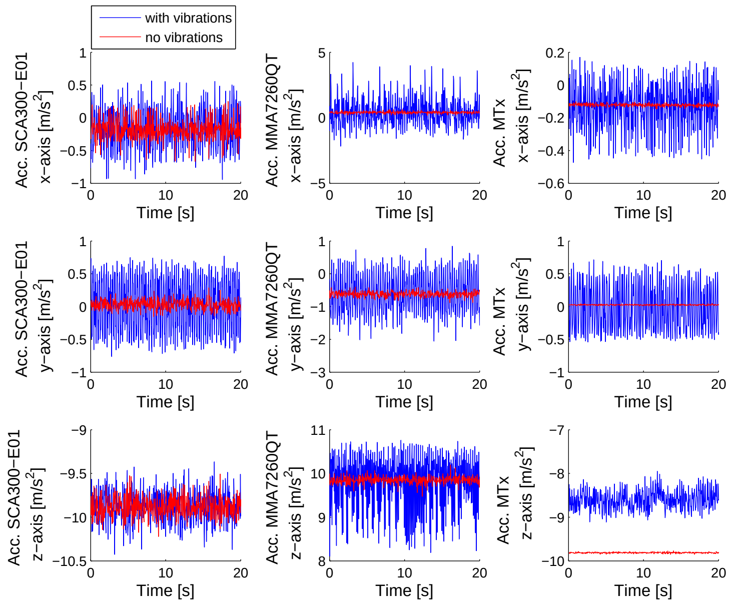

Figure B.2: Effect of the vibrations on the accelerometers.

To emphasise the effect of the vibrations on the measurement readings of the sensors, table B.1 presents the measurement variance of the different sensors and axes with and without vibrations. Additionaly the relative increase is listed.

Sensor	Axis	No vibrations	With vibrations	Unit	Increase
SCA3000-E01	x-axis	0.0245	0.1043	$(m/s^2)^2$	326.42%
	y-axis	0.0057	0.1760	$(m/s^2)^2$	2979.7%
	z-axis	0.0127	0.0338	$(m/s^2)^2$	166.82%
MMA7260QT	x-axis	0.0064	1.1470	$(m/s^2)^2$	17962%
	y-axis	0.0060	0.3716	$(m/s^2)^2$	6067.8%
	z-axis	0.0064	0.3812	$(m/s^2)^2$	5851.4%
IDG-300	x-axis	2.22^{-4}	0.0038	rad^2	1620.4%
	y-axis	2.75^{-4}	0.0014	rad^2	405.95%
DXRS300	z-axis	8.45^{-4}	0.0055	rad^2	545.21%
HMC6052	x-axis	0.0052^{-4}	0.0052^{-4}	gauss^2	0%
	y-axis	1.23^{-4}	8.01^{-4}	gauss^2	550.19%

Table B.1: Variance of the sensor measurements with and without vibrations.

To test if the magnetic field of the electric motor disturbs the sensor measurements, the running motor was hold by hand closely to the sensors. The analysis of the variance of these measurements shows, that if there is an influence, it can be neglected

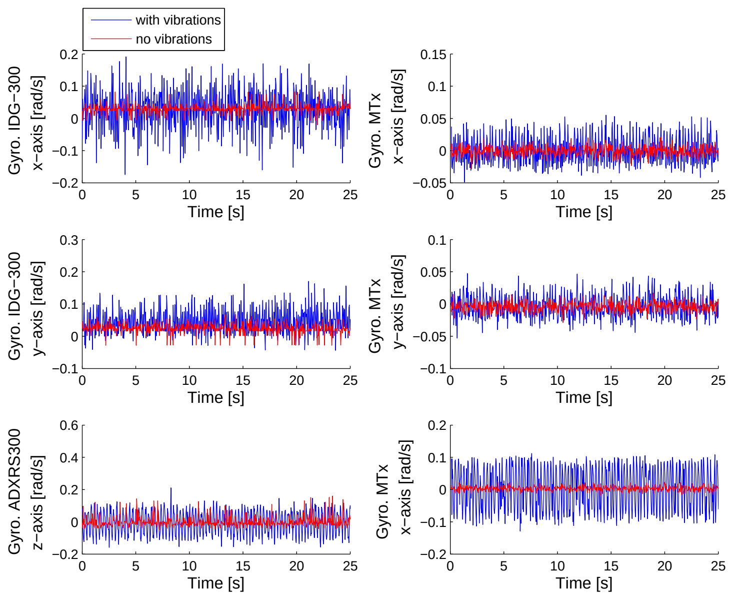

Figure B.3: Effect of the vibrations on the gyroscopes.

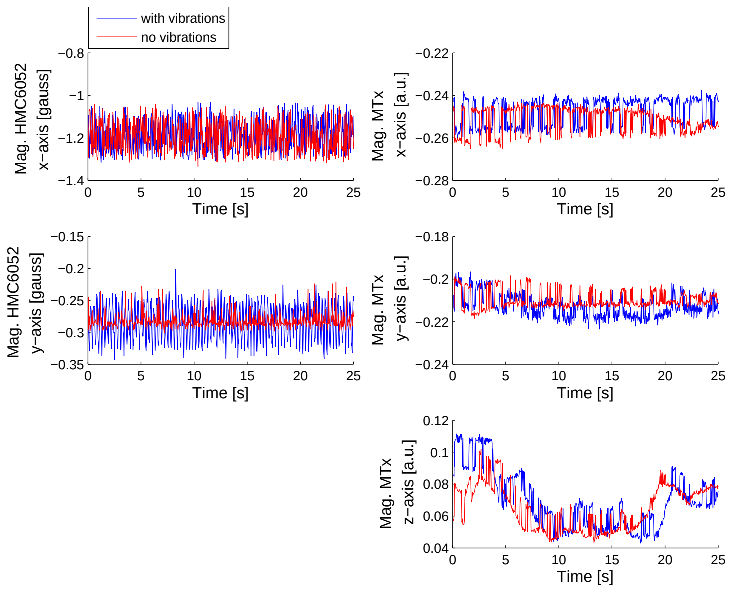

Figure B.4: Effect of the vibrations on the magnetometers.

Appendix C

Test C: Helicopter Vibrations Test Bench

In order to test the effect of the vibrations occuring on a MAV helicopter on the attitude determination algorithms, an helicopter test bench is developed as shown in figure C.1.

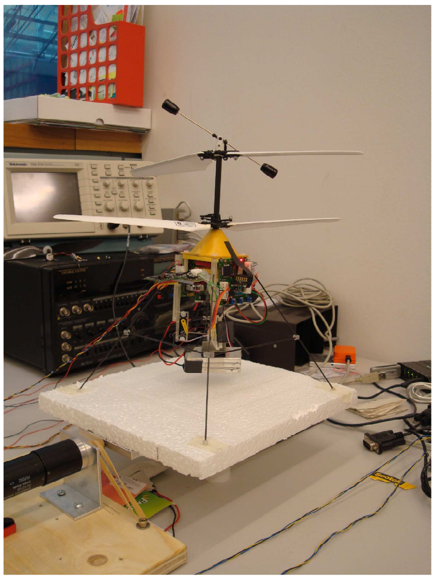

Figure C.1: The helicopter vibrations test bench.

Thereby, the ASL IMU is glued with double-faced scotch tape on the helicopter. Since this way does not allow a horizontal mounting of the IMU, the offset angle is substracted afterwards for making comparisons. The offset angle can be determined by keeping firstly the helicopter horizontal for a few seconds.
Another problem is to mount the helicopter on the styrofoam plate such that one of the roll or pitch axes coincide with the turn axis of the test bench.
These are the reasons why the performance indexes computed from measurement sets from this test bench should not be seen as reference values. They only show if the filters are able to deal with the large vibrations occuring on an helicopter. The influence of these vibrations on the sensors is shown in figure C.2, and table C.1 presents the variance of the measurement readings with and without the vibrations.

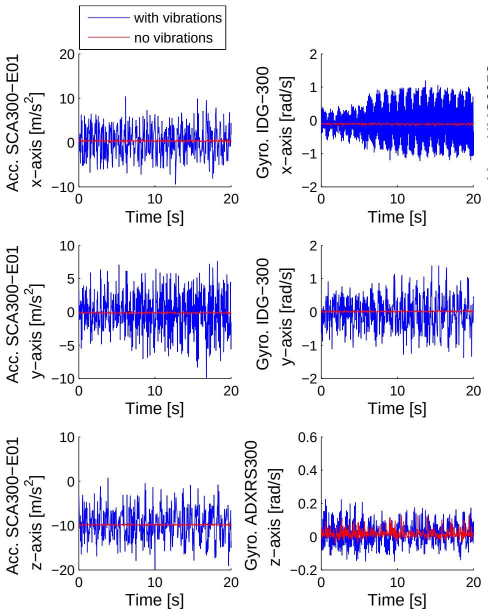

Figure C.2: Effect of the vibrations on the sensors.

Sensor	Axis	No vibrations	With vibrations	Unit	Increase
SCA3000-E01	x-axis	0.0228	11.9029	$(m/s^2)^2$	52069%
	y-axis	0.0061	9.2699	$(m/s^2)^2$	1514.7%
	z-axis	0.0116	11.1526	$(m/s^2)^2$	96016%
IDG-300	x-axis	2.90^{-4}	0.3424	rad^2	118010%
	y-axis	2.91^{-4}	0.2165	rad^2	74172%
DXRS300	z-axis	5.95^{-4}	0.0064	rad^2	972.82%
HMC6052	x-axis	4.19^{-5}	0.0018	gauss^2	4183.1%
	y-axis	4.53^{-5}	0.0012	gauss^2	2512.5%

Table C.1: Variance of the sensor measurements with and without vibrations.

Appendix D

Test D: Robot Arm

For simulating linear accelerations, the ABB 6 degree of freedom robot *IRB140* with the controler *S4* as shown in figure D.1 is used. This robot is allocated by the *Zentrum für Produkt- und Prozessentwicklung ZPP* in Winterthur[1].

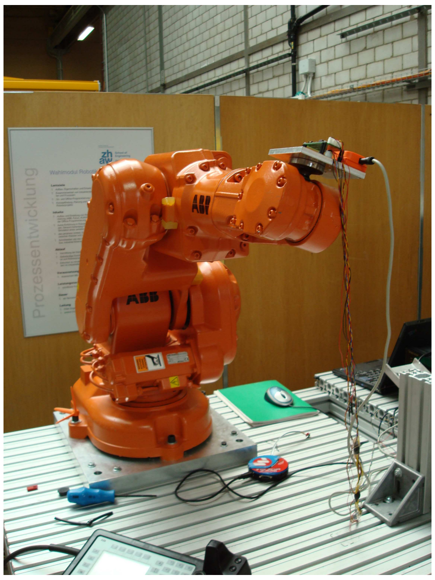
Figure D.1: The ABB IRB140 robot.

For the experiments the robot is programmed such that the motion of the robot arm is similar to the nervous behaviour of a MAV helicopter during hovering. Unfortunately, it is not possible to program smooth motions since then the accuracy of the robot's attitude measurement would suffer. Four different experiments with increasing maximum translational speed and maximum joint speed are conducted. For the slowest experiment, the maximum speeds are

$$[\text{max. translational speed, max. joint speed}] = [125mm/s,\ 25\,^\circ/s]$$

For the three other experiments, these speeds are doubled, quadruplicated and octuplicated respectively.
For specifying the experiments, figure D.2 shows the norm of the measurements of

[1]Contact person: Engel Peter

the 3-axis accelerometer.
Since the robot is programmed to take measurements of the robot's arm attitude only at the beginning and the end of a commanded movement, only two to three measurements per second are available.
The measurements from the different IMUs on the robot's arm and the measurements of the robot's arm attitude are not recorded by the same computer. So, the measurement set from the robot's arm attitude must be shifted temporally to compare the reference and the determined attitude from the IMUs.

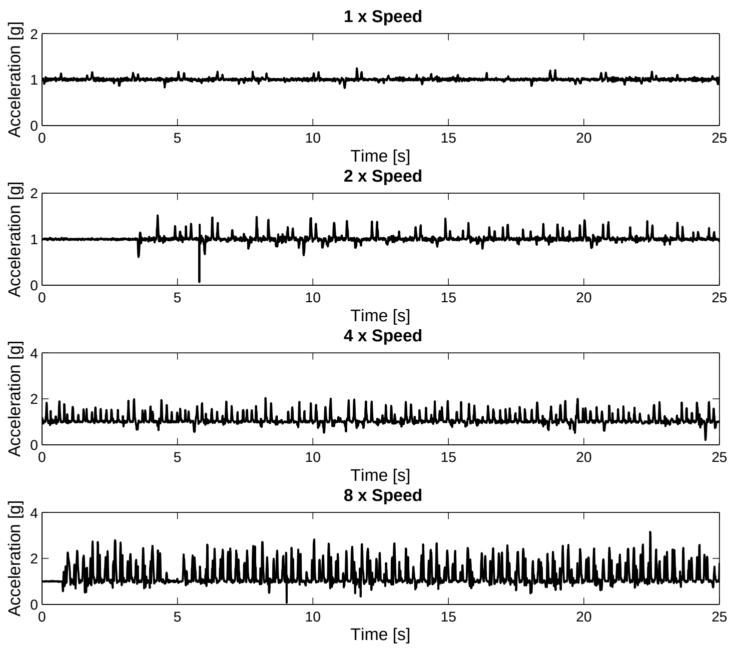

Figure D.2: Measured accelerations during the four tests with the robot.

Appendix E

Function Approximation with Lookup Tables

In chapter 8 three methods improving the accuracy of lookup tables are presented. The algorithms E.1 - E.3 introduce the general algorithms for the approximation of functions with lookup tables and linear interpolation, mean value of the two nearest points and nearest point.

Algorithm E.1 Function approximation with lookup table $[-x_{min}, x_{max}]$ and linear interpolation

```
Require: x ∈ $[-x_{min}, x_{max}]$
  float pos = $(x - x_{min}) / (x_{max} - x_{min}) * (table\_length - 2)$;
  int fpos = (int)pos;
  return table(fpos) + (table(fpos+1)-table(fpos))*(pos-fpos);
```

Algorithm E.2 Function approximation with lookup table $[-x_{min}, x_{max}]$ and mean value of two nearest points

```
Require: x ∈ $[-x_{min}, x_{max}]$
  int fpos = (int)($(x - x_{min}) / (x_{max} - x_{min}) * (table\_length - 2)$);
  return table(fpos) + (table(fpos+1)-table(fpos))/2;
```

Algorithm E.3 Function approximation with lookup table $[-x_{min}, x_{max}]$ and nearest point

```
Require: x ∈ $[-x_{min}, x_{max}]$
  int fpos = (int)($(x - x_{min}) / (x_{max} - x_{min}) * (table\_length - 1) + 0.5$);
  return table(fpos);
```

Listing E.1 shows how the lookup table for the algorithms with linear interpolation and with mean value of the two nearest points is produced. The lookup table for the nearest point algorithm is produced as presented in listing E.2.
The above presented algorithms are general and can be used for the approximation of any functions.
Since most trigonometric functions have symmetries, this can be exploited to further increase the accuracy. The following sections give an overview of such symmetry exploiting algorithms for trigonometric functions.

```
for (i=0; i< table_length; i++)
{
 table[i] = function_to_be_approximated((x_max−x_min)/...
            (table_length −2)*i + x_min)
}
```

Listing E.1: Creation of the lookup table for the lookup table approximations with linear interpolation and mean value of the two nearest points.

```
for (i=0; i< table_length; i++)
{
 table[i] = function_to_be_approximated((x_max−x_min)/...
            (table_length −1)*i + x_min)
}
```

Listing E.2: Creation of the lookup table for the lookup table approximation with nearest point.

E.1 Sine

E.1.1 Lookup table from 0 to π

As mentioned in chapter 8, the sine function must be defined for input values in the intervall $[-\pi, \pi]$. The easiest way is to use lookup tables for this whole intervall. But as the sine function has a point symmetry, this can be exploited by using a lookup table only from 0 to π.
Algorithm E.4 exemplifies how the general algorithm wiht linear interpolation must be adopted to exploit the point symmetry of the sine function..

Algorithm E.4 Sine approximation with lookup table [0, π] and linear interpolation

```
Require: x ∈ $[-x_{min}, x_{max}]$
   float pos = $abs(x)/(2 * PI) * (table\_length - 2)$;
   int fpos = (int)pos;
   if x >= 0 then
      return table(fpos) + (table(fpos+1)-table(fpos))*(pos-fpos);
   else
      return -table(fpos) - (table(fpos+1)-table(fpos))*(pos-fpos);
   end if
```

E.1.2 Lookup table from 0 to $\pi/2$

There is still another symmetry which can be exploited to increase the accuracy by unchanging table length. Again, in algorithm E.5 the adaptation is exemplified on the algorithm with linear interpolation.

E.2 Cosine

E.2.1 Lookup table from 0 to π

Since the cosine function is axissymmetric, this can be easily exploited by just taking the absolute value of the input value x as shown in algorithm E.6.

Algorithm E.5 Sine approximation with lookup table [0, $\pi/2$] and linear interpolation

```
Require: x ∈ $[-x_{min}, x_{max}]$
  if abs(x) <= PI/2 then
    if x >= 0 then
      float pos = $x/PI * (table\_length - 2)$
      int fpos = (int)pos;
      return table(fpos) + (table(fpos+1)-table(fpos))*(pos-fpos);
    else
      float pos = $-x/PI * (table\_length - 2)$
      int fpos = (int)pos;
      return -table(fpos) - (table(fpos+1)-table(fpos))*(pos-fpos);
    end if
  else
    if x > 0 then
      float pos = $(PI - x)/PI * (table\_length - 2)$
      int fpos = (int)pos;
      return table(fpos) + (table(fpos+1)-table(fpos))*(pos-fpos);
    else
      float pos = $(PI + x)/PI * (table\_length - 2)$
      int fpos = (int)pos;
      return -table(fpos) - (table(fpos+1)-table(fpos))*(pos-fpos);
    end if
  end if
```

Algorithm E.6 Cosine approximation with lookup table [0, π] and linear interpolation

```
Require: x ∈ $[-x_{min}, x_{max}]$
  float pos = $abs(x)/(2 * PI) * (table\_length - 2)$;
  int fpos = (int)pos;
  return table(fpos) + (table(fpos+1)-table(fpos))*(pos-fpos);
```

E.3 Arcus Sine

E.3.1 Lookup table from 0 to 1

Like the sine function, the arcus sine function is a point symmetric function, too. So, for using a lookup table from 0 to 1, instead from -1 to 1, the same adaptation as shown in algorithm E.4 must be made.
It must be pointed out, that for the arcus sine function the lookup tables for the algorithms with linear interpolation and with mean value of the two nearest points are produced as shown in listening E.3.

```
for (i=0; i< table_length-1; i++)
{
 table[i] = asin((1-x_min)/...
            (table_length-2)*i + x_min)
}
table[table_length-1] = asin(1);
```

Listing E.3: Creation of lookup table for arcus sine approximation with the algorithms with linear interpolation and with mean value of the two nearest points.

E.4 Arcus Tangent2

There are two different ways of approximating the arcus tangent2 function. One method is the indirect approximation, i.e., first, the arcus tangent function is approximated, and then, with *if-else* conditions, the quadrant is determined.
Algorithm E.7 shows an indirect approximation, where the arcus tangent function is approximated using relation (8.1) and the lookup table from the arcus sine function. There also exists an arcus tangent approximation not using lookup tables [47]. Al-

Algorithm E.7 Indirect atan2 approximation using arcus sine lookup table

```
temp = y/x;
angle = LookupTable_asin(temp/sqrt(1 + $temp^2$));

if x >= 0 then
  return angle;
else
  if y >= 0 then
    return angle + PI;
  else
    return angle - PI;
  end if
end if
```

gorithm E.8 introduces an indirect arcus tangent2 method using this approximation. A direct arcus tangent2 approximation is found in [11] and presented in algorithm E.9.

Algorithm E.8 Indirect atan2 approximation with approximated atan

```
if x == 0 then
  return y/abs(y)*PI/2;
else
  temp = y/x
  if abs(temp) <= 1 then
    angle = temp/(1+0.28*temp^2);
  else
    if temp > 1 then
      angle = PI/2 - temp/(temp^2 + 0.28);
    else
      angle = -PI/2 - temp/(temp^2 + 0.28);
    end if
  end if
end if

if x >= 0 then
  return angle;
else
  if y >= 0 then
    return angle + PI;
  else
    return angle - PI;
  end if
end if
```

Algorithm E.9 Direct atan2 approximation

```
coeff_1 = PI/4;
coeff_2 = 3*coeff_1;
abs_y = abs(y);
if x >= 0 then
  r = (x - abs_y) / (x + abs_y);
  angle = coeff_1 - coeff_1 * r;
else
  r = (x + abs_y) / (abs_y - x);
  angle = coeff_2 - coeff_1 * r;
end if
return y < 0.0 ? -angle : angle;
```

Bibliography

[1] E. R. Bachman, I. Duman, U. Y. Usta, R. B. McGhee, X. P. Yun, M. J. Zyda: *Orientation Tracking for Humans and Robots Using Inertial Sensors*. Proceedings of International Symposium on Computational Intelligence in Robotics and Automation, Monterey, CA, USA, pp.187-194, 1999.

[2] A.-J. Baerveldt, R. Klang: *A Low-cost and Low-weight Attitude Estimation System for an Autonomous Helicopter*. Proceedings of IEEE International Conference on Intelligent Engineering Systems, Budapest, Hungary, 1997.

[3] S. Bouabdallah, C. Bermes, D. Schafroth: *muFly*. `http://www.asl.ethz.ch/research/asl/mufly`, 30.11.07.

[4] A. Noth, S. Bouabdallah, J. A. Nikolic: *Design of the ASL Inertial Measurement Unit*. ASL internal paper, Version 2.03, 03.04.08.

[5] A. B. Chatfield: *Fundamentals of High Accuracy Inertial Navigation*. Progress in Astronautics and Aeronautics, Volume 174, 1997.

[6] J. Diebel: *Representing Attitude: Euler Angles, Unit Quaternions, and Rotation Vectors*. Stanford University, Standford, California 94301-9010, 20 October 2006.

[7] C. Eck: *Navigation Algorithms with Application to Unmanned Helicopters*. Diss. ETH No. 14402, Zürich , 2001.

[8] EXPERTS EXCHANGE *Trig lookup table*. `http://www.experts-exchange.com/Other/Math_Science/Q_21806568.html`, 17.03.2008.

[9] E. Foxlin: *Inertial Head-Tracker Sensor Fusion by a Complementary Separate-Bias Kalman Filter*. Proceedings of IEEE Virtual Reality Annual International Symposium, Los Alamitos, CA, pp.184-94, 1996.

[10] D. Gebre-Egziabher, R. C. Hayward, J. D. Powell: *Design of Multi-Sensor Attitude Determination Systems*. IEEE Transactions on Aerospace and Electronic Systems, Vol. 40, No. 2, April 2004.

[11] gamedev.net Forum: *Manually implementing atan2 (or atan)*. `http://www.gamedev.net/community/forums/topic.asp?topic_id=441464`, 17.03.2008.

[12] J. J. Hall, R. L. Williams II: *Inertial Measurement Unit Calibration Platform*. Journal of Robotic Systems, Volume 17, Issue 11, Pages 623 - 632, 2000.

[13] T. Hamel, R. Mahony: *Attitude estimation on SO(3) based on direct inertial measurements*. Proceedings of the 2006 IEEE International Conference on Robotics and Automation, Orlando, Florida, 2006.

[14] S. J. Julier, J. K. Uhlmann: *A New Extension of the Kalman Filter to Nonlinear Systems.* SPIE AeroSense Symposium, April 21 - 24, 1997.

[15] M. Jun, S. I. Roumeliotis, G. S. Sukhatme: *State Estimation of an Autonomous Helicopter Using Kalman Filtering.* Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 1999. IROS '99, 1999.

[16] S. K. Kashyap: *Automated Process Noise Covariance Computation.* National Aerospace Laboratories, Project Document FC 0309, 2003.

[17] M. Koifman, I. Y. Bar-Itzhack: *Inertial Navigation System Aided by Aircraft Dynamics.* IEEE Transactions on Control Systems Technology, Vol. 7, No. 4, July 1999.

[18] J. B. Kuipers: *Quaternions and Rotation Sequences.* Princeton University Press, Princeton, New Jersey, 1998.

[19] J. J. LaViola Jr.: *A Comparison of Unscented and Extended Kalman Filtering for Estimating Quaternion Motion.* Proceedings of the 2003 American Control Conference, 2003.

[20] R. Mahony, T. Hamel, J.-M. Pflimlin: *Complementary filter design on the special orthogonal group SO(3).* Proceedings of the IEEE Conference on Decision and Control, CDC05, Seville, 2005.

[21] J. L. Marins, X. Yun, E. R. Bachmann,R. B. McGhee,M. J. Zyda: *An Extended Kalman Filter for Quaternion-Based Orientation Estimation Using MARG Sensors.* Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, Oct. 29 - Nov. 03, 2001.

[22] M. Marmion: *Airborn Attitude Estimation using a Kalman Filter.* Master Thesis, 15. February, 2006.

[23] The MathWorks: *MATLAB Help.* Matlab 7.1.

[24] N. Metni, J.-M. Pflimlin, T. Hamel, P. Souères: *Attitude and Gyro Bias Estimation for a Flying UAV.* IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.

[25] National Geophysical Data Center: *Estimated Values of Magnetic Field Properties.* `http://www.ngdc.noaa.gov/seg/geomag/jsp/struts/calcPointIGRF`, 25.10.2007.

[26] M. Niculescu: *Sensor Fusion Algorithms for Unmanned Air Vehicles.* `http://www.u-dynamics.com/sensor_fusion/idc2002revised.pdf`, 15.10.07.

[27] A. Noth: *Sky Sailor.* `http://sky-sailor.epfl.ch/`, 30.11.07.

[28] A. Noth, R. Siegwart, W. Engel: *Autonomous Solar UAV for Sustainable Flight.* Advances in Unmanned Aerial Vehicles, State of the Art and the Road to Autonomy (K. P. Valavanis, ed.), vol. 33 of Intelligent Systems, Control and Automation: Science and Engineering, pp. 377 – 405, Springer Verlag, 2007.

[29] J. M. Pflimlin, T. Hamel, P. Souères: *Nonlinear attitude and gyroscope's bias estimation for a VTOL UAV.* International Journal of Systems Science Vol. 38, No. 3, March 2007.

[30] Polygonal Laboratory *Fast and accurate sine/cosine approximation.* `http://lab.polygonal.de/2007/07/18/fast-and-accurate-sinecosine-approximation/`, 17.03.2008.

[31] E. S. Quintana, G. Quintana, X. Sun, R. van de Geijn: *Efficient Matrix Inversion via Gauss-Jordan Elimination and its Parallelization.* Technical Report TR-98-19, Dept. of Computer Sciences, The University of Texas at Austin, 1998.

[32] H. Rehbinder, X. Hu: *Drift-free attitude estimation for accelerated rigid bodies.* Proceedings of the 2001 IEEE International Conference on Robotics & Automation Seoul, Korea. May 21-26, 2001.

[33] J. M. Roberts, P. I. Corke, G. Buskey: *Low-Cost Flight Control System for a Small Autonomous Helicopter.* Proceedings of IEEE International Conference on Robotics and Automation, 2003.

[34] A. Rottmann, M. Sippel, T. Zitterell, W. Burgard, L. Reindl, C. Scholl: *Towards an Experimental Autonomous Blimp Platform.* European Conference on Mobile Robots (ECMR), 2007.

[35] S. Saripalli, J.M. Roberts, P.I. Corke, G. Buskey: *A Tale of Two Helicopters.* Prodeedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 805-810, 2003.

[36] P. Setoodeh, A. Khayatian, E. Farjah: *Attitude Estimation By Separate-Bias Kalman Filter-Based Data Fusion.* Journal of Navigation, Vol. 57, Part 2, pages 261-274, 2004.

[37] R. Siegwart, I. Nourbakhsh: *Introduction to Autonomous Mobile Robots.* MIT Press, 2004.

[38] I. Skog, P. Händel: *Calibration of a MEMS Inertial Measurement Unit.* XVII IMEKO World Congress on Metrology for a Sustainable Development, September, 17−22, Rio de Janeiro, Brazil, 2006.

[39] M. St-Pierre, D. Gingras: *Comparison between the Unscented Kalman Filter and the Extended Kalman Filter for the Position Estimation Module of an Integrated Navigation Information System.* 2004 IEEE Intelligent Vehicles Symposium, 2004.

[40] R. Strasser: *Untersuchung von Beobacheterverfahren für eine inertiale Messeinheit.* Fortschritt-Bericht VDI Reihe 8 Nr. 1072, VDI Verlag, Düsseldorf, 2005.

[41] J. Vaganay, M. J. Aldon, A. Fournier: *Mobile Robot Attitude Estimation by Fusion of Inertial Data.* Proceedings of IEEE International Conference on Robotics and Automation, 1993..

[42] E. A. Wan, R. van der Merve: *The Unscented Kalman Filter for Nonlinear Estimation.* Proceedings of 2000 IEEE Symposium on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC), Lake Louise, Alberta, Canada, 2000.

[43] L. Wang, S. Xiong,Z. Zhou, Q. Wei, J. Lan: *Constrained Filtering Method for MAV Attitude Determination.* Proceedings of the IEEE Instrumentation and Measurement Technology Conference, 2005.

[44] S. Waslander: *STARMAC State Estimation.* 2004.

[45] G. Welch, G. Bishop: *An Introduction to the Kalman Filter.* `http://www.cs.unc.edu/~tracker/media/pdf/SIGGRAPH2001_CoursePack_08.pdf`, 2001.

[46] G. Welch, G. Bishop: *An Introduction to the Kalman Filter.* `http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf`, 2006.

[47] Wikipedia: *Arkustangens und Arkuskotangens.* `http://de.wikipedia.org/wiki/Arkustangens_und_Arkuskotangens`.

[48] Wikipedia: *Lookup Table.* `http://en.wikipedia.org/wiki/Lookup_table`, 17.03.2008.

[49] R. Zhu, D. Sun, Z. Zhou, D. Wang: *A linear fusion algorithm for attitude determination using low cost MEMS-based sensors.* MeasurementVolume 40, Issue 3, April 2007, Pages 322-328, 2007.

[50] Xsens Technologies B.V.: *MTi and MTx Low-Level Communication Documentation.* Document MT0101P, Revision F, September 20, 2006.