Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Autonomous Systems Lab
Prof. Roland Siegwart

Semester-Thesis

Generic IMU-Camera Calibration Algorithm

Influence of IMU-axis on each other

Autumn Term 2012

Supervised by:
Jörn Rehder
Janosch Nikolic

Author:
Christian Krebs

Declaration of Originality

I hereby declare that the written work I have submitted entitled

Generic Camera-IMU Calibration Algorithm

is original work which I alone have authored and which is written in my own words.

Author(s)

Christian Krebs

Supervising lecturer

Prof. Roland Siegwart

With the signature I declare that I have been informed regarding normal academic citation rules and that I have read and understood the information on 'Citation etiquette' (`http://www.ethz.ch/students/exams/plagiarism_s_en.pdf`). The citation conventions usual to the discipline in question here have been respected.

The above written work may be tested electronically for plagiarism.

Place and date	Signature

Contents

Abstract

This Semester Thesis is about the extension of a batch optimizing camera-IMU calibration algorithm. This algorithm is based on the minimization of a cost function consisting of the difference between an estimated sensor model output and the real measurement. Therefore, a sensor model is needed. The extended sensor model for the IMU (inertial measurement unit) used in this calibrator considers the following influences: misalignment of the sensor axes, scale factors of the output, orientation and position transformations between the camera, the accelerometer and the gyroscope as well as a time offset between the camera and the IMU, a random-walk bias and a linear acceleration influence to the gyroscope bias. The calibration algorithm is implemented in C++. To test the algorithm, a function was written in MATLAB to generate simulated sensor data. The calibrator was run with this data and the resulting parameters were compared with the real reference parameter used in the MATLAB model. The result shows, that all calibration parameters can be detected separately by the calibrator and they do not influence each other. Additionally, the influence of the sensor noise to the calibration output variance is analyzed. Comparing the results of a calibration, considering the scale factor and the misalignment, with a calibration, which does not consider it, shows that neglecting the misalignment- and scale-influences induce a bias on the calibration values. The calibration algorithm was also applied to specific calibration datasets. They were recorded by moving the CIMU system in front of a checkerboard over a period of 90 seconds. The values of the misalignment and the scale factor can be determined well. On the other hand, the resulting values for the acceleration influence to the gyroscope and the rotation between the gyroscope and the accelerometer are smaller than the standard deviation. Their influence is so small, that it can be neglected compared to the other influences. Nevertheless, the values of the acceleration influence matrix can be determined, but with a separate static experiment. And the results show, that this influence can be modeled linearly.

Symbols

Symbols

$\boldsymbol{p}_{w,b}$	translation between world frame to body frame (described in world frame)
$\bar{\boldsymbol{q}}_{w,b}$	rotation quaternion between world and body frame (described in world frame)
$\boldsymbol{T}_{w,b}$	homogeneous transformation from body frame into world frame
$\boldsymbol{C}_{w,b}$	rotation matrix (body frame described in world frame)
ϕ, θ, ψ	roll, pitch and yaw angle
$\boldsymbol{b}_{fg}$	gyroscope random walk bias
$\boldsymbol{b}_{gg}$	gyroscope bias induced by linear acceleration
$\tilde{\boldsymbol{\omega}}$	gyroscope output prediction (by sensor model)
$\boldsymbol{\omega}^*$	gyroscope output measurement
$\boldsymbol{\omega}$	angular velocity
$\tilde{\boldsymbol{a}}$	acceleration output prediction (by sensor model)
$\boldsymbol{a}^*$	acceleration output measurement
$\boldsymbol{a}$	linear acceleration
$\boldsymbol{g}$	gravity vector
$\boldsymbol{M}$	misalignment matrix (lower triangular matrix)
$\boldsymbol{S}$	scale matrix (diagonal matrix)
$\boldsymbol{B}_g$	linear acceleration influence to gyroscope matrix (3x3)
$\boldsymbol{n}$	gaussian distributed white noise
σ_{cal}	standard deviation of the estimated calibration values

Indices

w	world reference frame (attached to checkerboard)
b	body frame (attached to accelerometer's x-axis)
c	camera frame (aligned with camera)
g	gyroscope frame (aligned with gyroscope's x-axis)
x	x-axis
y	y-axis
z	z-axis
$\tilde{}$	sensor output prediction
*	sensor measurement

Acronyms and Abbreviations

CIMU	Camera-IMU System
CT	Continuous Time
DV	Design Variables
EKF	Extended Kalman Filter
ETH	Eidgenössische Technische Hochschule
GPS	Global Positioning System
IMU	Inertial Measurement Unit
lhs	Left hand side
MEMS	Micromachined Electro-Mechanical Systems
rhs	Right hand side
SLAM	Simultaneous Localization and Mapping

Chapter 1

Introduction

The starting point of this semester thesis is the wish to be able to calibrate a tracking system without any mechanical calibration table. Mechanical calibration tables are high precision devices, where the sensors, like an Inertial Measurement Unit (IMU), can be mounted and moved on a well defined trajectory. But these devices are heavy, expensive and usually bounded to one location. If it would be possible to calibrate a Camera-IMU (CIMU) system software based, only with the information from the camera and the IMU, by moving it in front of a reference frame, this would reduce costs and would be more flexible in application. But why is it so essential to calibrate such a system? First of all, no sensor is perfect. There are always influences that affect the sensor output. Some are deterministic and others are not. The deterministic ones can be compensated, if they are known. For IMUs, a calibration is doubly important, because not the sensor output is used to determine the position but the double integral. If there exists a small error in the measurement, it accumulates over time and can leads to a large error in the final result. Therefore, IMUs are often used together with some absolute position measurements as GPS or camera triangulations. But the more precise the IMU is calibrated the more accurate the local position estimations are.

Therefore, the goals of this semester thesis are:

Table 1.1: Semester Thesis goals:

Extend an existing calibration toolbox to determine the internal IMU parameters.
Determine the following influences - IMU axis scale factor - Cross-axis sensitivity of the IMU - Linear acceleration effect to the gyroscope - Orientation between acceleration and gyroscope
The calibration should be based on the batch optimization without needing a calibration table
The goal is to: - reduce time and cost for calibration - be flexible in application - be location independent

This report is structured as follows. First, a short overview of the already existing calibration methods and the functionality of IMUs is given in chapter 2. Followed by the explanation of the used extended sensor model (chapter 3) and the implementation into the calibrator (chapter 4). In chapter 5, the results are presented and discussed. Finally, the conclusion is done in chapter 6.

Chapter 2

State of the Art

2.1 IMU Sensors

IMUs (Inertial Measurement Units) can measure angular velocities as well as linear accelerations and are often used to localize moving objects. As described by Titterton in [1], the IMU can be split up in two parts. One part is responsible to measure the specific force, called accelerometer. The second part, measuring the angular velocity, is called gyroscope. In the early beginning of using IMUs, they were big mechanical systems with many high-precision parts and the cost was accordingly high. Over time they became smaller and smaller and today, it is possible to match everything together to a size, which can be placed on a micro chip. This so called micro-machined electromechanical systems (MEMS) are fabricated in very small mechanical structures, using quartz or silicon. But with the smaller fabrication, the accuracy has decreased, the scale and noise factors have raised and other influences have arised. This seems to be bad, but it is possible to eliminate a lot of these systematic errors by compensation. For this, it is important to calibrate the IMU as well as possible and having an appropriate model of the system.
In order to understand how the IMU, especially the MEMS, works and to be able to derive later (in chapter 3) an appropriate model, a short summary of the detailed description in [1] is given in the next section.

2.1.1 MEMS Gyroscope

The MEMS gyroscope contains a mass vibrating in a direction perpendicular to the measured rotation axis. There is no rotational part in the device as in big gyroscopes. The functionality is based on the Coriolis force, which acts in the third direction perpendicular to the rotation axis as well as to the vibration direction. A scheme is shown in figure 2.1. This results in a small displacement, which can be measured by capacity. But the values are really small. A typical device produce a Coriolis force of about 10^{-7} N for an angular velocity of 1 rad/s, a displacement of 10^{-9} m and a capacitive charge of approximately 10^{-18} F [1] p.196. Therefore, the MEMS are very sensitive to shocks, which should be avoided to get reasonable measurement. Besides the balanced oscillator shown in figure 2.1, there exist also other principles like simple beam oscillators, cylindrical shell oscillators and a lot of other specializations, also such, which are more shock resistant. In most cases, three separately measurement units are needed, which are more or even less perpendicular to each other, depending of the fabrication[1].

[1]The IMU (MPU-6000A) used in the experiment (see chapter 5) contains three independent vibratory MEMS. Details can be found in the appendix A and in the datasheet p.12ff, 25

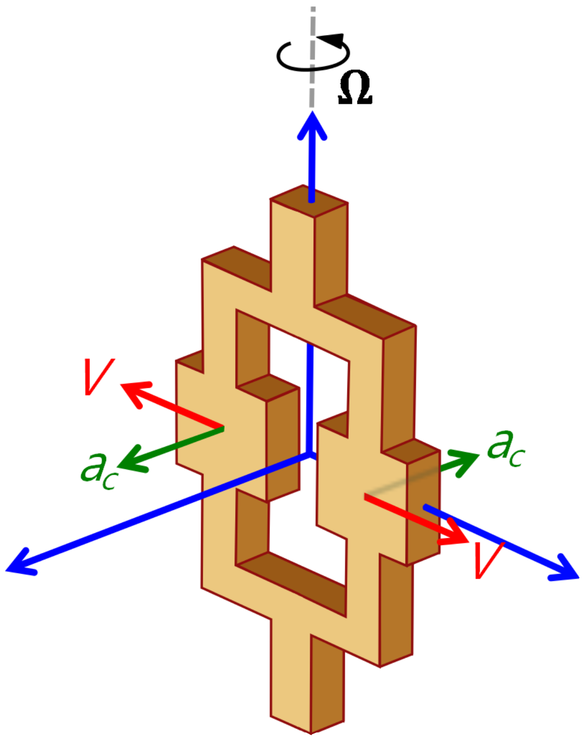

Figure 2.1: Principle of a MEMS gyroscope: The gyro oscillates in direction V (red). If the sensor rotates around the measurement axis (gray), the Coriolis force induce a displacement a_c perpendicular to the two other axes.

2.1.2 MEMS Accelerometer

Also for the accelerometer, there exist a lot of different designs. They can be separated into two main classes. The principle of the first type is based on the displacement of a proof mass by an applied acceleration. The mass is mounted with an elastical part, which can be deformated. The second type bases on the change of frequency of a vibration part, caused by tension, which change depending on the applied acceleration. This sensor type has a potentially higher accuracy in measuring micro g. The used IMU, the MPU-6000 contains an accelerometer of the first type. A schematic view is shown in figure 2.2

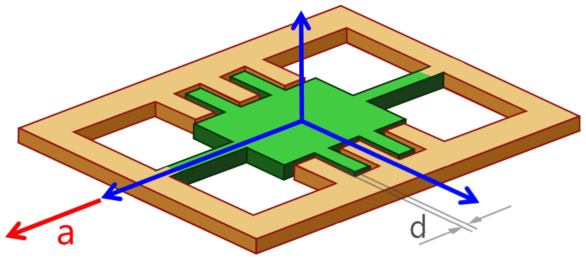

Figure 2.2: Principle of an in-plane MEMS accelerometer: An acceleration a in measurement direction leads to a displacement of the proof mass (green), which changes the distance d and therefore the capacity between the fingers.

2.1.3 Used Models

There exists many papers, in which models of the IMU are created to describe the behavior mathematically. The first step is to define, which influences should be modeled and which neglected. A comprehensive list of effects are presented in [1] p.223ff which include even effects, which are really small. Based on this information and on the data-sheets of the IMUs (MPU-6000A and ADIS16488), here a list of influences:

- initial (turn-on) bias
- in-run (random-walk) bias
- acceleration-depending bias to gyroscope (anisoelatic bias)
- static scale-factor
- temperature depending scale factor
- hysteresis in gyro/accelerometer response
- misalignment of the axis
- non-linearity
- oscillation frequency response
- magnetic field influence
- shock and vibration influences
- earth rotation (15.041 °/h)
- earth gravity (9.80665 m/s^2)
- nondeterministic noise (including all other effects)

Some influences are in normal use very small, as the magnetic field influence. Depending on the used IMU, other influences, as the earth rotation, lie in the order of the noise rate (e.g MPU-6000) and other influences are hard to model like the non-linearity. Also the temperature is often neglected, although it has a significant influence especially on MEMS. But if the initial phase is skipped and the sensor has heated up, the temperature can normally be modeled as constant and therefore the influence is neglected. But this is a little bit delicate.
The most used influences are misalignment (including cross-axis sensitivity), scale factor, random walk bias and sometimes acceleration influence to the gyro bias. Skog and Haendel propose in [6] a model with misalignment, scale factor and bias. In general, the scale factor and the bias are also temperature varying but for the calibration part, they assume them as constant. Fong, Ong and Nee model the IMU in [9], including misalignment and a scale factor and Nieminen et al. applied in [7] beyond these two influences also the acceleration influence to gyroscope bias as well as a constant bias. Based on this papers, I included in the sensor model, described in chapter 3, the following influences: misalignment (including cross-axis sensitivity), static scale factor, linear acceleration influence to gyroscope bias, transformation between accelerometer and gyroscope and a random walk bias, based on the work of P. Furgale and J. Rehder. [3], [4]
In the following, a short description to these specific influences is given.

Misalignment

The non-orthogonality of three sensor axis is described detailed in the book 'Fundamentals of High Accuracy Inertial Navigation' [2] p.25ff. It is a pure geometrical transformation between the three axis and it can be described with three angles. By assuming small misalignment angles, the hole system can be described linearly as it was also done in [6] by describing the transformation with a matrix T_a^p, which corresponds to the matrix M in [9]. In this work, I call the matrix M and a detailed description can be found in section 3.2.1.

Scale factor

Each of the three gyroscope respectively accelerometer axis has its own scale factor. All the papers above assume to model a linear behavior of the IMU. Therefore the scale factors are constants, which can be written in a diagonal matrix, like the K_{sf}^a matrix in [2] or the S matrix in [9] and [7]. If the temperature is taken into account, then the matrix S is temperature varying and not constant any more. But because the temperature normally converges to a work temperature, in most cases the temperature and also the S matrix is assumed to be constant.

Linear Acceleration Influence to Gyroscope

This influence is neglected sometimes and sometimes not. The so called anisoelastic bias describes the influence of a linear acceleration to the output of the gyroscope. As described in section 2.1.1 the MEMS gyros contain vibrating parts and they are affected by external acceleration. The effect is described and modeled linearly by Nieminen in [7]. Also in [1] p.226 the effect is described linearly. The anisoelastic bias is described in these two references, as well as in this work, with a matrix B.

Other Influences to Sensor Output

The other influences are all neglected in the mentioned references and this seems to be reasonable, because their influence is small and the non-deterministic noise is normally higher.

2.2 Calibration Methods

For a precise IMU calibration, rotation tables [1] are usually used. The sensor is mounted to the moving table, which follows, depending on the parameter to calibrate, a well defined trajectory. The position and orientation of the IMU is known exactly by means of encoders. In this experiment, it is very important, that the sensor is mounted with high precision, often done by cubes with very accurately machined faces. Also due to the requirement of high stiffness, the whole test bench is in general heavy and not flexible in application, to use it in different places.
Therefore other, more flexible and cheaper techniques, without a calibration table, were developed to calibrate IMU-Systems. They are application-optimized and can be used, if there exists a Camera-IMU system (CIMU). They use the information of both systems to calculate the calibration parameters software based.

Kalman Filter Based Algorithm

For example Mirzaei and Roumeliotis have developed and described in [12] a Kalman Filter based algorithm. They track with a camera features and use this information to have an additional position information to the IMU data. By having a sensor model (without misalignment and scale factor) they apply an iterated EKF to estimate the transformation between the camera and the IMU. Internal IMU calibration is not done here. A further work, based on this but using a continuous-time batch estimation instead of the Kalman Filter has done by Furgale et al. described in the following section.

2.2.1 Batch Optimization Algorithm

As besides the static camera to IMU transformation also the position and orientation of the CIMU system has to be estimated, it is suitable to describe this motion by continuous basis functions as done by Furgale et al. in [3]. This has the advantage, that the continuous function can be derived analytically to get the velocities and accelerations. With this technique, an algorithm was developed described in [4] to estimate the camera to IMU transformation as well as the time offset between the camera and the IMU. Based on this principle, my thesis expands the previous work of J. Rehder and P. Furgale, by applying the misalignment, scale factor, acceleration influence to gyroscope and transformation between accelerometer and gyroscope, to be able to determine the calibration parameters of the IMU by a continuous time batch estimation.

Chapter 3

Sensor Model

Based on that knowledge, I derived a sensor model for the CIMU System. As the assumption of a linear behavior of the cross-axis sensitivity seems to be reasonable (see section 2.1.3), I build a model containing transformation matrices, which map the different axis on each other with different weights. The results in section 5.2.3 confirm the assumption of linearity.

3.1 Basic Model

First we look on a simple model, which I used as basic framework for the extended model[1]. This model contains no interaction between the sensor axis. The output of the gyroscope $\tilde{\boldsymbol{\omega}}$ is modeled as a combination between the real angular velocity $\boldsymbol{\omega}_{(b)}$ represented in the body frame, a random walk bias $\boldsymbol{b}_{fg}$ and a white noise $\boldsymbol{n}_g$. The random walk bias is be initialized by the start offset $\boldsymbol{b}_{0g}$ and changes over time by integration over the zero mean noise $\boldsymbol{n}_{bg}$. The accelerometer model looks equivalent.
A symbolic representation is shown in figure 3.1

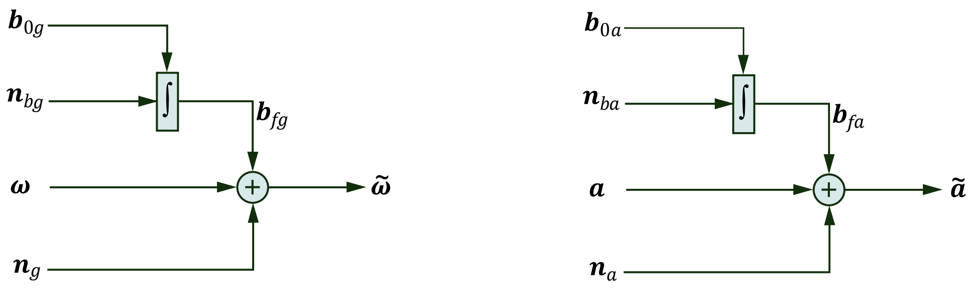

Figure 3.1: Basic sensor model for the gyroscope (left) and the accelerometer (right): The output is only affected by a random walk bias $\boldsymbol{b}_{fi}$ and white noise $\boldsymbol{n}_i$.

3.2 Extended Model

3.2.1 Axes Misalignment

I expanded the basic model by some additional assumptions. One assumption is, that the three axes of the gyroscope and the accelerometer are not exactly perpen-

[1]The basic model corresponds to the described model in [4] and is used as reference without consideration of cross-axis sensitivity

dicular on each other. To describe a misaligned coordinate frame, we need three additional angles (see section 2.1.3 or [2] p.25/26). Assuming the misalignment angles are small, we can make the simplification

$$sin(\alpha) \approx \alpha, \quad cos(\alpha) \approx 1 \;\; |\alpha \ll 1$$

With that, it is possible to write the projection of the perpendicular coordinate frame to the misaligned frame by a matrix, which looks like

$$M_{mi} = \begin{bmatrix} 1 & 0 & 0 \\ M_{yz} & 1 & 0 \\ M_{zy} & M_{zx} & 1 \end{bmatrix}$$

The matrix can be multiplied later with the corresponding vector we want to transform. The entries of the matrix are defined as

$$M_{yz} = -\alpha_{yz} \approx sin(-\alpha_{yz})$$

$$M_{zy} = \alpha_{zy} \approx sin(\alpha_{zy})$$

$$M_{zx} = -\alpha_{zx} \approx sin(-\alpha_{zx})$$

The diagonal elements would be exactly

$$1, \; cos(-\alpha_{yz}), \; cos(\alpha_{zy})cos(-\alpha_{zx})$$

but I simplified them to 1 as the misalignment angles are small.

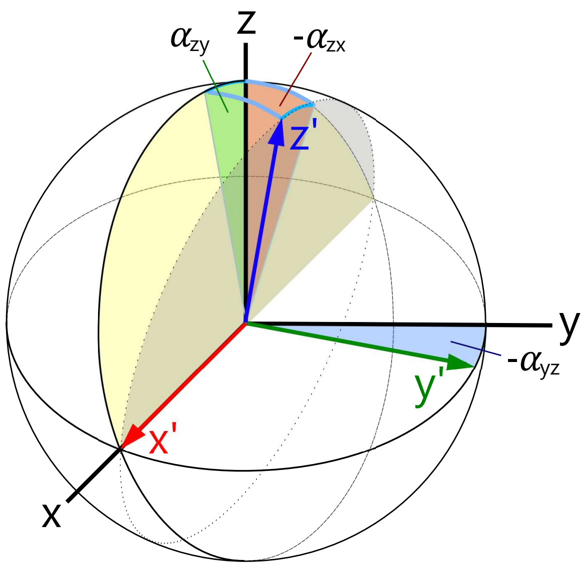

Figure 3.2: Model of the sensor axes misalignment. The x'-axis is aligned with the x-axis of the orthogonal system by definition.

3.2.2 Cross-Axis Sensitivity

The next step is to model the cross axis sensitivity. Assuming the main term of the cross axis sensitivity is linear (see section 2.1.3), we can model the influence from one axis to another with vectors, which are not exactly perpendicular and project them on each other. As this is exactly the same as a misalignment of the real physical axes, we get an additional misalignment matrix M_{cr}, which can be added to M_{mi} above. With this linear model, I defined a new overall misalignment matrix M, which contains the physical misalignment as well as the cross-axis sensitivity[2].

3.2.3 Axis Scale

The scale factors of the axes are easy to model, if we neglect as above the small non-linearity part of the sensors. I defined a diagonal matrix S containing the three scale factors of the x-, y- and z-axis.

$$S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix}$$

The values S_i are composed as $1+dS$ and define the ratio between the sensor output and the effective value.

3.2.4 Linear acceleration influence to gyroscope

The gyroscope's output is affected by linear acceleration of the hole CIMU system. Also here, I assume a linear connection between the linear acceleration and the additional bias, that is caused by it. To be able to map the influence of each acceleration axis to the three gyroscope axis, we need nine values. I allocated them in the matrix B_g:

$$B_g = \begin{bmatrix} B_{xx} & B_{xy} & B_{xz} \\ B_{yx} & B_{yy} & B_{yz} \\ B_{zx} & B_{zy} & B_{zz} \end{bmatrix}$$

The value B_{ij} describes the proportionate share of the acceleration axis j to the gyroscope axis i.

3.2.5 Complete Sensor Model

Combining all these factors, we can derive the extended sensor model shown in figure 3.3.

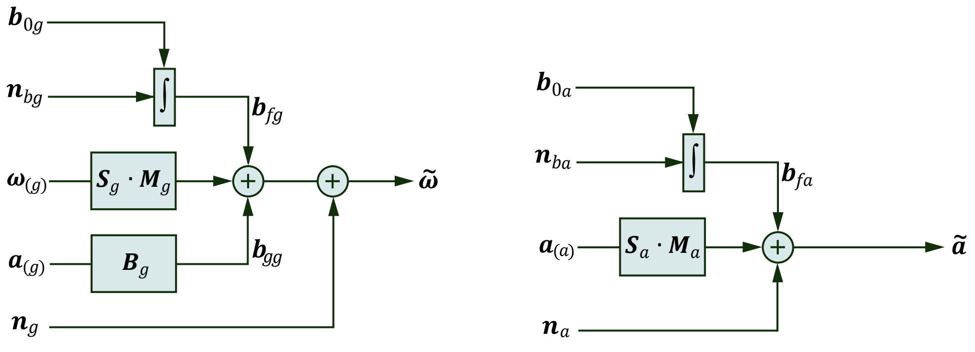

Figure 3.3: Extended sensor model: In contrast to the basic model, the input $\omega_{(g)}$ and $a_{(b)}$ are multiplied by the matrices S and M. In the gyro model, the acceleration influence B_g is also added.

[2]Hint: The cross axis sensitivity and the misalignment, as modeled here, can not be detected separately by calibration. But as they behave exactly the same, it is not necessary and we look on it as one influence parameter M

Written as algebraic formula, the sensor model of the gyroscope and the accelerometer looks like:

$$\tilde{\boldsymbol{\omega}} = S_g \cdot M_g \cdot \boldsymbol{\omega}_{(g)} + \boldsymbol{B}_g \cdot \boldsymbol{a}_{(g)} + \boldsymbol{b}_{fg} + \boldsymbol{n}_g$$

$$\tilde{\boldsymbol{a}} = S_a \cdot M_a \cdot \boldsymbol{a}_{(b)} + \boldsymbol{b}_{fa} + \boldsymbol{n}_a$$

The inputs $\boldsymbol{\omega}_{(g)}$ and $\boldsymbol{a}_{(b)}$ are the angular velocity and the acceleration written in the gyroscope frame (g) and in the body frame (b) respectively. The used coordinate systems are presented in figure 3.4 and table 3.1.

Table 3.1: coordinate systems

Body frame	**{B}**:	aligned with the accelerometer's x-axis
Camera frame	**{C}**:	aligned with the camera (x, y: image plane; z: in projection direction)
Gyroscope frame	**{G}**:	aligned with the gyroscope's x-axis
World frame	**{W}**:	reference frame, attached to a checkerboard (z-axis normal to checkerboard plane)

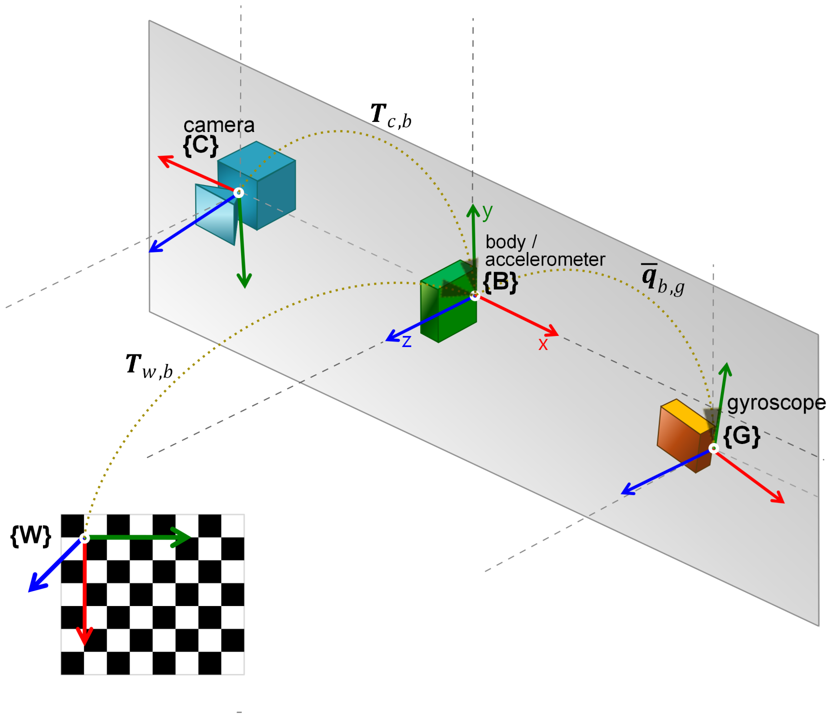

Figure 3.4: Scheme of the CIMU Sensor: Transformation between the reference (world) frame $\{W\}$, accelerometer (body) frame $\{B\}$, camera frame $\{C\}$ and gyroscope frame $\{G\}$

The transformations between the different coordinate systems are described with homogeneous transformation matrices[3] T. $T_{w,b}(t)$ is a time variant matrix describing the position $\boldsymbol{p}_{w,b}(t)$ and orientation $C_{w,b}(t)$ of the body frame relative to the

[3]Theory of homogeneous transformations described in [10] and the used notation derives from [3] with minor deviations.

world frame. It is a 4x4 matrix composed as:

$$T_{w,b}(t) = \begin{bmatrix} C_{w,b}(t) & \boldsymbol{p}_{w,b}(t) \\ 0 & 1 \end{bmatrix}$$

where $C_{w,b}(t)$ is a 3x3 rotation matrix and $\boldsymbol{p}_{w,b}(t)$ is a 3x1 translation vector from the world frame origin to the body frame origin.
The second transformation $T_{c,b}$ is time invariant and describes the constant transformation from the camera to the body frame, equivalent to $T_{w,b}(t)$ described above. The third transformation consists only of a rotation. $\bar{\boldsymbol{q}}_{b,g}$ is the quaternion, that describes the rotation between the body frame and the gyroscope frame. The quaternion can be written as rotation matrix [4] $C_{b,g}$. There is no need to know the translation between the body frame and the gyroscope frame, because the translation has no influence to the angular velocity measured in the gyroscope.
Together with this transformations, we can write $\boldsymbol{\omega}_{(g)}(t)$ and $\boldsymbol{a}_{(b)}(t)$ as a function of the angular velocity $\boldsymbol{\omega}_{(w)}(t)$ and the linear acceleration $\boldsymbol{a}_{(w)}(t)$ in the world frame

$$\boldsymbol{\omega}_{(g)}(t) = C_{b,g}^T \cdot (C_{w,b}(t))^T \cdot \boldsymbol{\omega}_{(w)}(t)$$

$$\boldsymbol{a}_{(b)}(t) = (C_{w,b}(t))^T \cdot \boldsymbol{a}_{(w)}(t)$$

Until now, I ignored the gravity $\boldsymbol{g}$ as well as the earth rotation $\boldsymbol{\Omega}_{earth}$. In contrast to the earth rotation, which is with $7.292e{-}5$ rad/s (15.041 °/h [1]) really small, the gravity ($g = 9.80665\ m/s^2$) can not be neglected. As the accelerometer perceives the gravity as an negative acceleration (from the earth away), it has to be subtracted from $\boldsymbol{a}_{(w)}$. Putting everything together the extended model can finally be described as

$$\tilde{\boldsymbol{\omega}} = S_g \cdot M_g \cdot C_{b,g}^T \cdot (C_{w,b}(t))^T \cdot \boldsymbol{\omega}_{(w)}(t) + B_g \cdot \left[C_{b,g}^T \cdot (C_{w,b}(t))^T \cdot \left(\boldsymbol{a}_{(w)}(t) - \boldsymbol{g} \right) \right] + \boldsymbol{b}_{fg} + \boldsymbol{n}_g$$

$$\tilde{\boldsymbol{a}} = S_a \cdot M_a \cdot (C_{w,b}(t))^T \cdot \left(\boldsymbol{a}_{(w)}(t) - \boldsymbol{g} \right) + \boldsymbol{b}_{fa} + \boldsymbol{n}_a$$

[4] explained in [11] formula (92)

Chapter 4

Implementation

4.1 Basic Algorithm

The hole algorithm is based on the existing batch optimization 'aslam' algorithm to determine the transformation between camera and IMU from P. Furgale and J. Rehder. It is an extension to be able estimating the IMU internal calibration data as misalignment of the axis, the scale factor, the orientation transformation between gyroscope and accelerometer and the acceleration influence to the gyroscope bias. The algorithm uses continuous time functions to describe the estimated position and orientation of the CIMU system. For detailed information refer to [4], here a short overview.

4.1.1 Basis Functions

All time varying unknowns are modeled as a combination of known basis functions, for example B-splines. As the set of this function is described by[1]

$$\boldsymbol{\Phi}(t) = [\phi_1(t), \phi_2(t), ..., \phi_B(t)]$$

where $\phi_i(t)$ are the defined basis functions, a time varying state $\boldsymbol{x}(t)$ can be approximated by

$$\boldsymbol{x}(t) = \boldsymbol{\Phi}(t) \cdot \boldsymbol{c}$$

The state can then be described by the $B \times 1$ - dimensional coefficient vector $\boldsymbol{c}$. If we now want to have the derivative of this motion, e.g. the velocity or the acceleration of the CIMU system, it is sufficient to derive the basis functions, what we can do, because they are known analytical functions

$$\boldsymbol{v}(t) = \dot{\boldsymbol{p}}(t) = \dot{\boldsymbol{\Phi}}(t) \cdot \boldsymbol{c}$$

$$\boldsymbol{a}(t) = \ddot{\boldsymbol{p}}(t) = \ddot{\boldsymbol{\Phi}}(t) \cdot \boldsymbol{c}$$

4.1.2 Error Term and Minimization

Now, the goal is to optimize all parameters and to get as close as possible to the real motion. To do this, we define a cost function, which should be minimized. It consists of the error terms e_i and the covariance matrices Q_i

$$J = 1/2 \sum_{i=1}^{n} \boldsymbol{e}_i^T Q_i^{-1} \boldsymbol{e}_i$$

[1]analogue to [4]

for all e's. The error terms are the differences between the estimated output of the sensor model (see chapter 3.2) and the real measurement. Therefore we get
for the gyroscope

$$\boldsymbol{e}_{g_k} = \tilde{\boldsymbol{\omega}}(t_k) - \boldsymbol{\omega}^*(t_k)$$

for the accelerometer

$$\boldsymbol{e}_{a_k} = \tilde{\boldsymbol{a}}(t_k) - \boldsymbol{a}^*(t_k)$$

and for the camera

$$\boldsymbol{e}_{c_{mj}} = \tilde{\boldsymbol{y}}_m(t_j) - \boldsymbol{y}^*_m(t_j)$$

where $\boldsymbol{y}_m(t_j)$ is the projection of a detected landmark point to the camera's image plane at time t_j. Note: the camera model is the same as in [4] and therefore the equation has not changed.
With the new sensor model for the gyroscope an the accelerometer, the three equations looks like[2]

$$\boldsymbol{e}_{g_k} = \left[S_g M_g C^T_{b,g} \left(C_{w,b}(t_k)\right)^T \boldsymbol{\omega}(t_k) + B_g \left[\left(C_{w,b}(t_k)\right)^T \left(\boldsymbol{a}(t_k) - \boldsymbol{g}\right)\right] + \boldsymbol{b}_{fg}\right] - \boldsymbol{\omega}^*(t_k)$$

$$\boldsymbol{e}_{a_k} = \left[S_a M_a \left(C_{w,b}(t_k)\right)^T \left(\boldsymbol{a}(t_k) - \boldsymbol{g}\right) + \boldsymbol{b}_{fa}\right] - \boldsymbol{a}^*(t_k)$$

$$\boldsymbol{e}_{c_{mj}} = h\left(T_{c,b} T_{w,b}(t_j + d)^{-1} \boldsymbol{p}^m_w\right) - \boldsymbol{y}^*_m(t_j)$$

The function h is the camera measurement model, that transforms the landmark points $\boldsymbol{p}^m_w$ to the camera's image plane and d is the time offset between the camera frame and the IMU measurement. The minimizing of the overall cost function J is done with the Levenberg-Marquardt (LM) algorithm[3]. The algorithm needs the derivatives of the error functions and to solve this analytically, the jacobians are needed. How to determine this for the new sensor model is described in the following section 4.3. But first we look on all values the calibrator has to estimate, the so called 'Design Variables'.

4.2 Design Variables

There exists a lot of unknowns in the sensor model, which can be varied by the calibrator to find the minimum of the cost function J. This adaptable values, called design variables (DV), can be distinguished into two types:

- time variant DVs
- time constant DVs

The time varying ons are modeled with a batch of B-splines as described above. Each batch consists of its own coefficient vector $\boldsymbol{c}$, whose elements are adjusted separately like time constant DVs. Table 4.1 lists all DVs used in the implemented calibrator. As all these values are unknowns, the jacobians of the error terms have to be calculated for each of this DV analytically. The following chapter outlines this for the new design variables comparing with the existing algorithm, I have extended on. The new DVs of the extended model are marked in table 4.1 with 'add'.

[2]There is one change between this implementation and the sensor model developed in chapter 3: The transformation $C_{b,g}$ in the term with B_g is neglected to increase the observability because B_g is really small and $C_{b,g}$ is almost the identity matrix.

[3]see [4]

Table 4.1: Used design variables in the IMU calibrator

Description	Symbol	Dimension	Added
Time Variant			
IMU pose described in world frame	$T_{w,b}$	6	
Gyroscope bias	$\boldsymbol{b}_{fg}$	3	
Accelerometer bias	$\boldsymbol{b}_{fa}$	3	
Time Constant			
Transformation camera frame to IMU (body frame)	$T_{c,b}$	6	
Gravity direction	$\boldsymbol{g}_w$	3	
Time offset between camera and IMU	d	1	
Misalignment gyroscope	M_g	3	add
Misalignment accelerometer	M_a	3	add
Scale factor gyroscope	S_g	3	add
Scale factor accelerometer	S_a	3	add
Acceleration influence to gyroscope	B_g	9	add
Orientation accelerometer to gyroscope	$\bar{\boldsymbol{q}}_{b,g}$	3	add

4.3 Jacobians

The Levenberg-Marquardt algorithm needs the derivative of the cost function for each design variable. This is done by the jacobian matrix. The jacobian matrix in general is defined as

$$Jac = \begin{bmatrix} \frac{\delta e_1}{\delta DV_1} & \frac{\delta e_1}{\delta DV_2} & \cdots & \frac{\delta e_1}{\delta DV_n} \\ \frac{\delta e_2}{\delta DV_1} & \frac{\delta e_2}{\delta DV_2} & \cdots & \frac{\delta e_2}{\delta DV_n} \\ \vdots & & \ddots & \vdots \\ \frac{\delta e_o}{\delta DV_1} & & & \frac{\delta e_o}{\delta DV_n} \end{bmatrix}$$

It is a $o \times n$ matrix, where o is the dimension of the error function $\boldsymbol{e}$ and n is the number of design variables.
Looking now on the IMU model. As all influences are assumed to be linear, it is a composition of matrix/vector multiplications and additions. An addition can be derived term by term. They do not influence each other. This is not the case for a multiplication. Here we have to apply the chain rule, by multiplying the derivative with the other side. Knowing this, we can concentrate to the elements separately. The three types of influences, I have considered in the sensor model, can all be described by 3×3 matrices as mentioned in chapter 3. To determine now the jacobian of such a matrix with a 3×1 vector, it is important, which elements of the matrix are design variables and which one are constants. Looking on the general case, where all elements of the matrix are design variables (e.g. case for the matrix B_g). The multiplication defined as[4]

$$B_g \cdot \boldsymbol{v} = \begin{bmatrix} B_1 & B_4 & B_7 \\ B_2 & B_5 & B_8 \\ B_3 & B_6 & B_9 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

[4]Note: the numeration of the matrix is column-based to be consistent with the implementation, I have expanded on

results in the following jacobian (derived to the nine elements of the B_g matrix)

$$Jac_{(B_g \cdot \boldsymbol{v})}(lhs) = \begin{bmatrix} v_1 & 0 & 0 & v_1 & 0 & 0 & v_1 & 0 & 0 \\ 0 & v_2 & 0 & 0 & v_2 & 0 & 0 & v_2 & 0 \\ 0 & 0 & v_3 & 0 & 0 & v_3 & 0 & 0 & v_3 \end{bmatrix}$$

and the jacobian derived to the three elements of the vector v corresponds to the left hand side (here the B_g matrix)

$$Jac_{(B_g \cdot \boldsymbol{v})}(rhs) = \begin{bmatrix} B_1 & B_4 & B_7 \\ B_2 & B_5 & B_8 \\ B_3 & B_6 & B_9 \end{bmatrix}$$

The two other influences, the misalignment matrix M and the scale factor S consist only of three design variables, all other elements are constant. The matrix M is a lower triangular matrix and S is a diagonal matrix. The jacobian can be derived from the general case with nine design variables. All columns corresponding to a constant have to be erased. Therefore I implemented in the calibration algorithm a new "Expression"- class, which is able to get several matrix patterns, defining which elements are design variables and which one are constants. Figure 4.1 shows the three different matrix patterns used.

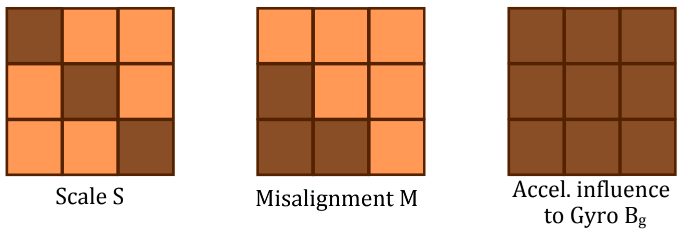

Figure 4.1: The three matrix pattern used in the calibration algorithm. The dark elements are the unknown design variables, the bright ones constant entries in the matrix, that are not estimated by the algorithm.

The jacobian for the misalignment matrix is therefore

$$Jac_{(M_i \cdot \boldsymbol{v})}(lhs) = \begin{bmatrix} 0 & 0 & 0 \\ v_2 & 0 & 0 \\ 0 & v_3 & v_3 \end{bmatrix}$$

and for the scale factor matrix S_i

$$Jac_{(S_i \cdot \boldsymbol{v})}(lhs) = \begin{bmatrix} v_1 & 0 & 0 \\ 0 & v_2 & 0 \\ 0 & 0 & v_3 \end{bmatrix}$$

This are the three patterns, used in the calibration algorithm. The implemented class is also able to handle each linear matrix transformation by giving different matrix patterns.
The different calculated jacobians can now be matched together, respectively pre-multiplied with the other side, when using the chain rule. Also here, we look on a short example and illustrate this with the following multiplication

$$S_i \cdot M_i \cdot \boldsymbol{v}$$

The jacobian can now be determined by the separate jacobians, applying the chain rule. We define the vector $\boldsymbol{w}$ as the result of the multiplication of $M_i \cdot \boldsymbol{v}$ and its jacobian $Jac_{(M_i \cdot \boldsymbol{v})}(lhs)$ is used to be pre-multiplied with the jacobian out of $S_i \cdot \boldsymbol{w}$

$$Jac_{(S_i \cdot M_i \cdot \boldsymbol{v})}(lhs) = Jac_{(M_i \cdot \boldsymbol{v})}(lhs) \cdot Jac_{(S_i \cdot \boldsymbol{w}_{M_i \cdot v})}(lhs)$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ v_2 & 0 & 0 \\ 0 & v_3 & v_3 \end{bmatrix} \begin{bmatrix} v_1 & 0 & 0 \\ 0 & M_2 \cdot v_1 + v_2 & 0 \\ 0 & 0 & M_3 \cdot v_1 + M_6 \cdot v_2 + v_3 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ v_1 \cdot v_2 & 0 & 0 \\ 0 & v_3 \cdot (M_2 \cdot v_1 + v_2) & v_3 \cdot (M_3 \cdot v_1 + M_6 \cdot v_2 + v_3) \end{bmatrix}$$

This principle is now applied to the hole error function. With that, the cost function can be minimized by the Levenberg-Marquardt algorithm, which optimizes all design variables including the IMU calibration parameters. The results are shown in the following chapter.

Chapter 5

Results

5.1 Model as Reference

5.1.1 MATLAB Generated Sensor Data

To be able to test the implemented calibration algorithm, I used sensor data generated with MATLAB. In the MATLAB function I have implemented also all the sensor influences as axis misalignment, scale factor, acceleration influence to gyroscope, random walking bias, initial bias, misalignment between the accelerometer and the gyroscope and zero mean white noise on the camera and the IMU measurements as well as earth gravity and earth rotation. The movement of the CIMU system is modeled by trigonometric functions with different frequencies to simulate a reasonable movement done in real calibration by hand (see figure 5.1). The simulation has a mean acceleration of 3.12 m/s^2 and a mean rotation velocity of 0.366 rad/s. With this generated sensor data, I was able to check the functionality of the calibration algorithm. The 'real' IMU calibration parameters (which were set in the MATLAB code) are known and serve as reference values. I tested the algorithm for all calibration parameter with separate generated sensor data, where I changed always one value. The resulting calibration parameters converged to the exact value by reducing the noise and they do not influence each other. For example increasing one value of the matrix M_g changes the corresponding entry in the output but all other entries stay at the same values (apart from noise). Examples for calibration output data are found in the Appendix B.

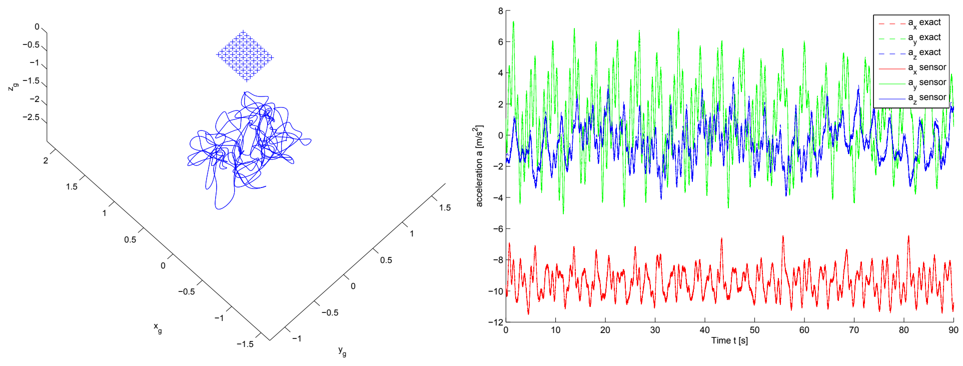

Figure 5.1: MATLAB generated calibration movement (left) to generate simulated sensor data, like the accelerometer output (right).

5.1.2 Noise Influence

Looking now on the influence of the sensor noise. The MATLAB sensor model used to generate sensor data is exactly the same as above with estimated IMU parameters corresponding to the data-sheet of the MPU-6000A sensor. The rate noise density of the MPU is shown in table 5.1.

Table 5.1: Rate noise density of the reference IMU (MPU-6000A)

Description	Value datasheet		Value SI-units
gyroscope rate noise density	$0.005\ (°/s)/\sqrt{Hz}$	=	$8.73e-5\ (rad/s)/\sqrt{Hz}$
accelerometer noise density	$400\ \mu g/\sqrt{Hz}$	=	$3.92e-3\ (m/s^2)/\sqrt{Hz}$

With this data, I generated 100 sensor data sets over 90 seconds and 200 Hz. The parameters for this datasets are all the same except the aplied random white noise. Running the calibration algorithm over this data and plotting the results as histogram, shows that the mean of the detected calibration parameters lies exactly on the true reference value, but the individual measurements differ with some variance σ^2_{cal} from the exact value. Figure 5.2 illustrate this for the estimated translation $\boldsymbol{p}_{c,b}$ between camera and body (accelerometer).

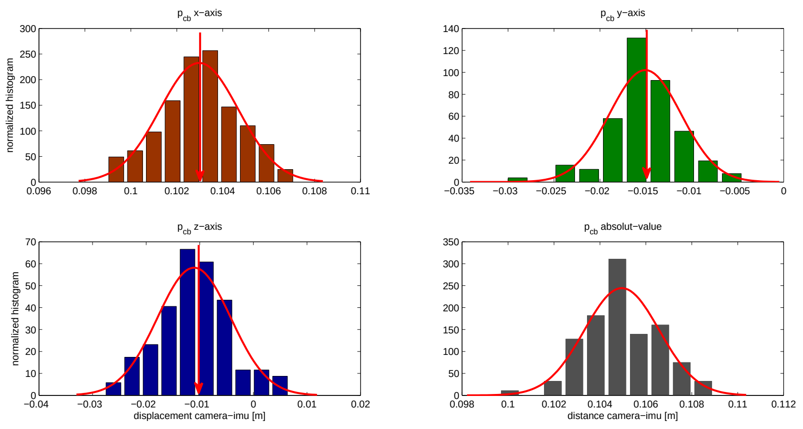

Figure 5.2: Histogram of the calibrator result for the translation between camera and body (accelerometer) of 100 datasets with noise level of a MPU-6000A. The red arrow shows the correct reference value.

To examine the influence of the noise, I reduced it two times by a factor of ten and generated 100 datasets for each value to compare it with the results above. I did the same additionally for the case without camera projection noise. The results after running the calibration algorithm are shown in figure 5.3. Only the fitted normal distributions are drawn, to make the plot more clearly.
The influence of the camera projection noise compared with the IMU output noise is very small, but affects in a small bias of the mean! If we reduce the IMU noise, the precision increases. The mean of the distribution however does not change, what means, that also with a high noise level the value can be estimated more accurately. But therefore a lot of measurements are required. Comparing the variances of the different results show that not all axis are equal sensitive to noise level change. It depends on the occurring acceleration and rotation. But by reducing the deviation of the noise level by a factor of 10 the resulting standard deviation of the translation

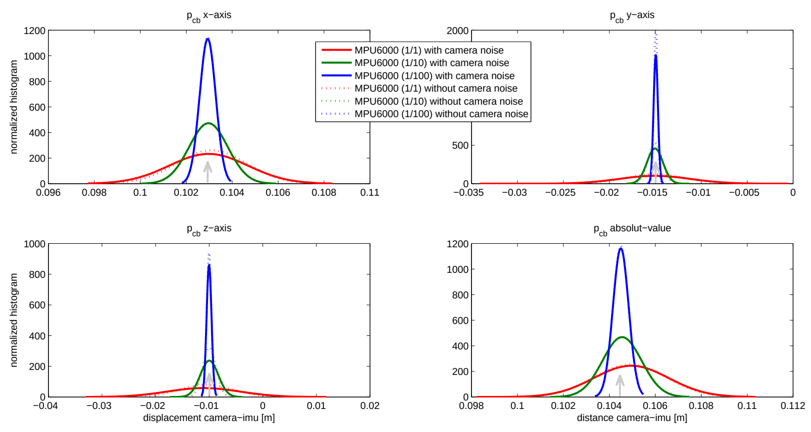

Figure 5.3: Distribution of the resulting estimation for the camera to body translation $\boldsymbol{p}_{c,b}$ for different sensor noise levels.

$\boldsymbol{p_{c,b}}$ is only reduced by a factor of 2-4. The exact values are listed in table 5.2. By this simulation it was possible to estimate approximately the variance respectively the standard deviation of all DVs used in the calibrator. The values are calculated with the noise level given by the MPU-6000A data sheet[1]. The calculated values are listed in table 5.3. The given value is the mean of the different entries of the vectors or matrices, if we took only one measurement over 90 s. It can be improved by taking longer sessions or more measurement sets. A detailed list is printed in Appendix B.

Table 5.2: Resulting standard deviation $\sigma_{cal,p_{c,b}}$ of 3 × 100 datasets with different noise levels (in [m])

Axis	$\sigma_{p_{c,b}}$ MPU(1/1)	$\sigma_{p_{c,b}}$ MPU(1/10)	$\sigma_{p_{c,b}}$ MPU(1/100)
x-axis	0.0017147	0.0008439	0.0003524
y-axis	0.0039093	0.0008685	0.0002380
z-axis	0.0068550	0.0016813	0.0004640

Table 5.3: Design Variable's standard deviations σ_{cal} for a single measurement over 90s determined with 100 datasets

Description	Symbol	σ_{cal}	
Rotation camera frame to IMU	$\bar{\boldsymbol{q}}_{c,b}$	0.001586	[-]
Translation camera frame to IMU	$\boldsymbol{p}_{c,b}$	0.004160	[m]
Gravity direction	$\boldsymbol{g}_w$	0.014348	$[m/s^2]$
Time offset between camera and IMU	d	0.000144	[s]
Misalignment gyroscope	M_g	0.002456	[rad]
Misalignment accelerometer	M_a	0.001745	[rad]
Scale factor gyroscope	S_g	0.003081	[-]
Scale factor accelerometer	S_a	0.001593	[-]
Acceleration influence to gyro	B_g	0.000211	$[\frac{rad}{s}/\frac{m}{s^2}]$
Orientation accelerometer to gyroscope	$\bar{\boldsymbol{q}}_{b,g}$	0.000658	[-]

[1]The used values are listed in Appendix A.

5.1.3 Neglecting Influences between Sensor Axes

Very interesting is the result, if we estimate that the real sensor depends on some influences between his axis. But the applied calibration assumes that all axis are exactly perpendicular and neglects influences between each other. Then the resulting calibration values deviate by a bias from the exact value. I have simulated this with the generated MATLAB sensor data, using the basic sensor model (see section 3.1) in the calibrator without the matrices M, S, $C_{b,g}$ and B. The result is illustrated in figure 5.4.

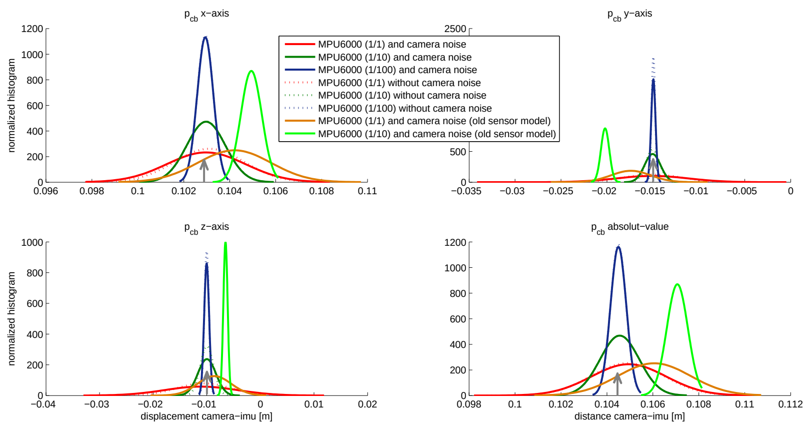

Figure 5.4: Difference between a calibration considering influences between the sensor axis (blue) and without (light green). Neglecting it results in an estimation bias.

With the used characteristic IMU values[2] the offset can be up to 5 mm for one axis. Of course, this are values calculated with the generated sensor data with some assumed misalignment parameter. Therefore the real offset can vary from this values. But the simulation shows well, that if there is some cross-axis sensitivity between the sensor axis and it is not modeled in the sensor model, we get a not negligible bias in our calibration estimation.

5.2 Calibration with Checkerboard

Until here, all calibrations are done with MATLAB generated sensor data. The tests have shown, that the calibration algorithm works well and converts to the exact value, if several measurements are taken. Now we are looking on real measurements recorded with the CIMU system developed at the Autonomous System Lab (ASL) at ETH Zurich. The CIMU system shown in figure 5.5 consists of two cameras, a high cost IMU (ADIS 16488), two separate accelerometers, a gyroscope consisting of two separate devices and the lower cost IMU (MPU-6000). The aim is to calibrate this lower cost MPU-6000 with the developed algorithm to reach accuracies coming toward the high cost ADIS.

[2]Refering to Appendix A

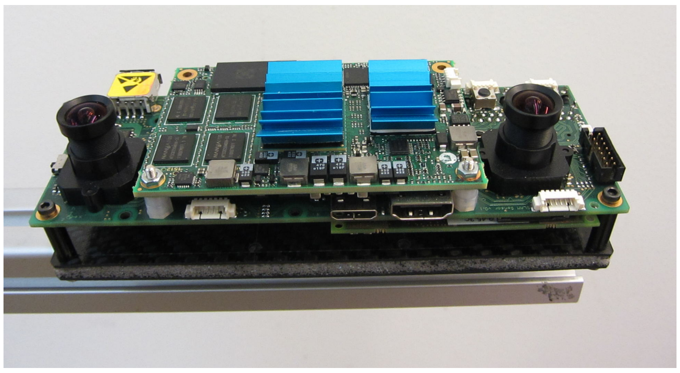

Figure 5.5: CIMU system used together with the calibration algorithm.

5.2.1 Difference between Calibration with and without Consideration of Cross-Axis Sensitivity

For the test case, I recorded 18 datasets with the CIMU system over each 90 seconds. As ground truth and reference for the camera, a checkerboard was used, standing more or less vertically. It is important to cover as much as possible of the feasible positions and orientations with the CIMU system without loosing the checkerboard out of view. The more rotation is in one calibration dataset the better and more accurate the determined calibration parameters are.
The calibration algorithm uses the camera information as well as the IMU measurement data to determine the exact trajectory of the CIMU system moved by hand. The result of the minimizing function described in chapter 4 provides a time variant transformation $T_{w,b}(t)$ from the world frame to the body frame and the two biases of the gyroscope and the accelerometer described as continuous B-splines as well as all design variables. If we now compare the results, we get a curve for the basic model[3] and one curve for the extended model, that considers the misalignment M, the scale factor S, the orientation between accelerometer and gyroscope $\bar{\boldsymbol{q}}_{b,g}$ and the anisoelatic bias B_g which defines the linear acceleration influence to the gyroscope output. Some bias can be detected similar to section 5.1.3. Of course, there is now no reference value we can compare with as in the tests with the generated sensor data. But if we look on the estimated signal (computed out of the estimated $T_{w,b}(t)$ trajectory) and compare this with the real measurement, the extended sensor data match more precisely than the other. In figure 5.6 and 5.7 the two signals are plotted comparing with the measurement.

[3]Basic model without any influences between the axis, see section 4.1.

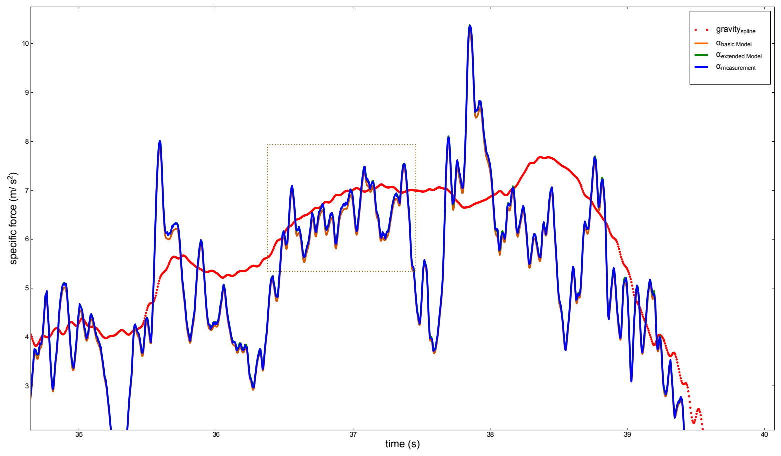

Figure 5.6: Comparing the accelerometer's z-axis measurement (blue) with the estimated measurements calculated after calibration (green and orange).

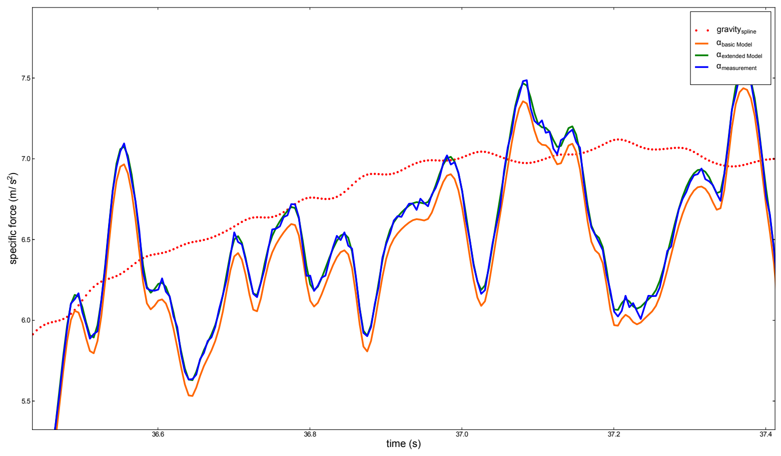

Figure 5.7: Comparing the accelerometer's z-axis measurement (blue) with the estimated measurements calculated after calibration with basic model (orange) and extended model (green).

In the shown sector the estimated measurement, using the basic model to project the calibration back to the sensor output data (considering no influences between the IMU axis), differs from the real measurement by some amount up to 0.15 m/s^2. With the extended model, the estimated value can match the real measurement without noticeable difference. It is important to mention, that this difference is not for all axis as big as in this figure. On some axis there is no difference between the basic model and the extended model. This depends on the size of the influence between the axis. If it is almost zero for some axis, as we will see in the following section, there is also no big difference between the basic model algorithm and the extended model algorithm. The influence, we see here in figure 5.7 caused mainly by the scale factor s_z of the matrix S_a.

5.2.2 Misalignment and Scale factor

As I have mentioned in the section above, the influence between the different axis are not noticeable on each axis. If we look on the output of the calibration for the misalignment matrices (M_g and M_a) and for the scale factor matrices (S_g and S_a), it can be mentioned, that for this specific calibrated CIMU system the two entries α_{zx} of the matrix M_g and the s_z of the matrix S_a are dominant. The determined calibration data[4] looks like

$$M_g = \begin{bmatrix} 1 & 0 & 0 \\ 0.00109 & 1 & 0 \\ 0.00360 & 0.01633 & 1 \end{bmatrix} \tag{5.1}$$

$$M_a = \begin{bmatrix} 1 & 0 & 0 \\ -0.00115 & 1 & 0 \\ 0.00137 & 0.00092 & 1 \end{bmatrix} \tag{5.2}$$

$$S_g = \begin{bmatrix} 1.00377 & 0 & 0 \\ 0 & 0.99688 & 0 \\ 0 & 0 & 1.00473 \end{bmatrix} \tag{5.3}$$

$$S_a = \begin{bmatrix} 1.00218 & 0 & 0 \\ 0 & 1.00008 & 0 \\ 0 & 0 & 1.01080 \end{bmatrix} \tag{5.4}$$

The values are also plotted as histogram in figure 5.8 together with the fitted Gaussian distribution.

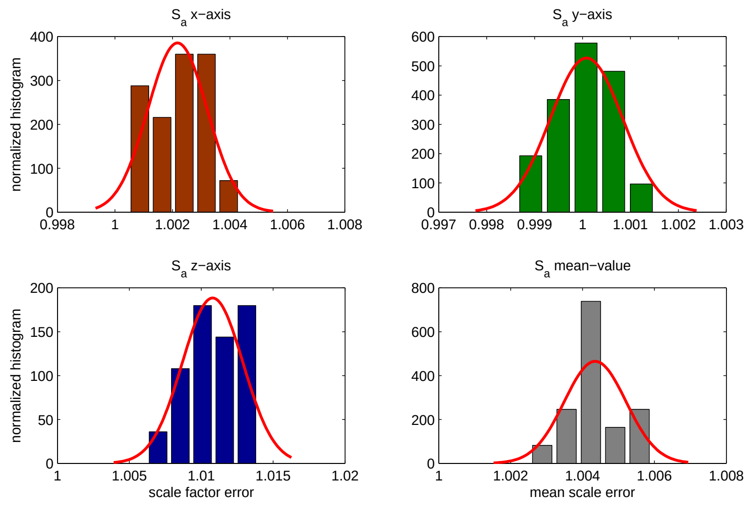

Figure 5.8: Result of the calibration algorithm for the scale factor matrix S_a of the accelerometer.

The misalignment of the gyroscope z-axis rotated about the x-axis is therefore roughly 0.94 degree and the scale factor of the accelerometer's z-axis 1.08 %. This is the effect, we have seen in the figure above. Beside this two dominant values the approximate size of the entries α_{zy} of M_g and s_x as well s_z of S_g can be determined as

$$\alpha_{zy,g} = 0.0036[rad] \approx 0.2^\circ$$

$$s_{x,g} = 1.0038$$

$$s_{z,g} = 1.0047$$

[4]Determined as the mean of the measurements number 5 to 22 with camera 0

Only roughly, because the standard deviation[5] for this three values are 0.00147 rad, 0.00077 and 0.00043 (in the same order as listed above). All other values are in the same order or smaller than their corresponding deviation and can therefore set to 0, because they are too small to be detectable and have no significant influence.

5.2.3 Linear acceleration influence to gyroscope

Looking on the results for the matrix[6] B_g , we see that all detected values lie below the corresponding standard deviation, because the values are too small. Therefore they can not be detected by this calibration method for an IMU with the noise-to-B_g ratio of the MPU-6000A. The standard deviation for the matrix entries calculated from the sensor data are in order of $1 \cdot 10^{-4}\ \frac{rad}{s} / \frac{m}{s^2}$. This is a little bit smaller than the theoretically calculated variation in section 5.1.2. But it is still too high compared with the B_g values to find. Of corse, if the B_g values are so small, we could say, that the influence is as small that we could neglect the acceleration influence to the gyroscope output. But I was interested in detecting the value nevertheless, also to see, if the assumption of linear influence is feasible. Therefore I made an other experiment. It is a static experiment to examine the anisoelatic bias of the gyroscope in case of earth gravity.

Static Anisoelatic Bias Experiment

With this experiment the influence of a linear acceleration to the gyroscope bias is observed. As the linear acceleration, the earth gravity is used, which affects in one of the three gyroscope directions. The CIMU system is now mounted statically with one gyroscope axis aligned with the gravity (see figure 5.9). If there would exist no influences, all three gyroscope outputs should be zero. Of course, they are not. The static bias obtain some shifting. If the CIMU system is now rotated by 90° to align another gyroscope axis with gravity, the gyroscope outputs are not the same as before. I have done this for all six orientations. The raw signal is shown in figure 5.10. The interesting part is only the static one, so I cut out the noisy parts of repositioning (see figure 5.11).

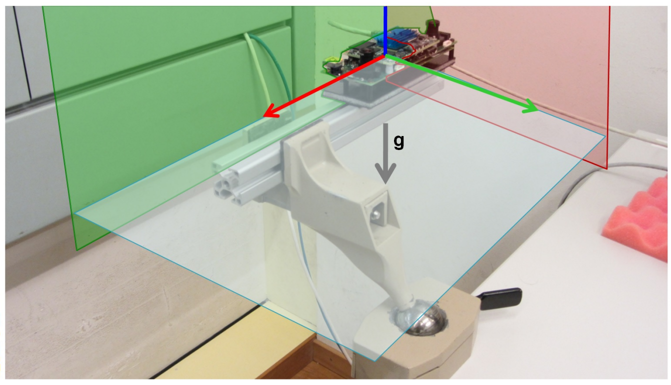

Figure 5.9: Experiment set-up to measure the influence of the earth gravity to the gyroscope outputs. In the shown configuration the gyroscope's z-axis (blue) is aligned with the gravity (gray).

[5]A detailed list of the experimentally determined variances can be found in Appendix B
[6]The values can be found in Appendix B

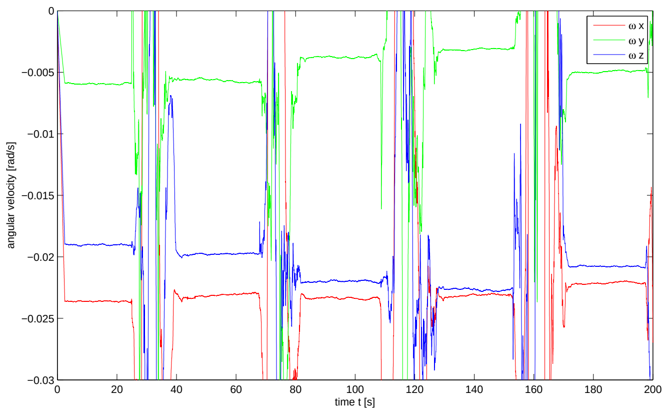

Figure 5.10: Unfiltered gyroscope output of the static experiment to determine the matrix B_g

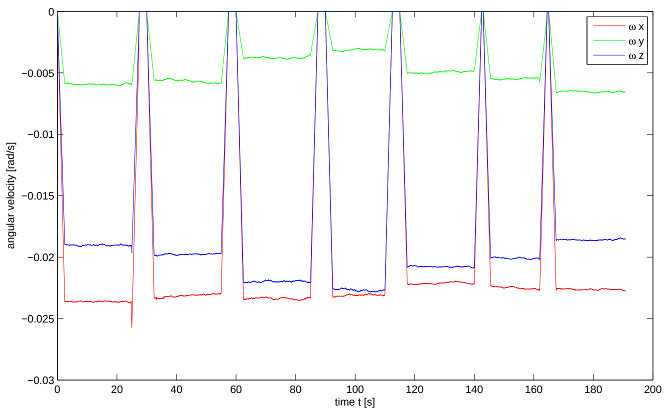

Figure 5.11: The same data as above but the rotational part is cut out. The influence of the gravity to the gyroscope is now visible, by different high levels after rotating the sensor by 90°.

The seven parts in the measurement correspond to the following orientations:

$$-z, -y,\ z,\ y,\ x, -x, -z$$

The difference depending of the orientation is not high but it can clearly be seen and is up to 0.003 rad/s per axis. Assuming the same sensor model as in the calibration algorithm and described in section 3.2, we can now compensate this influence by subtracting the $B_g \cdot \boldsymbol{a}$ term from the sensor output.

$$\boldsymbol{\omega} = \boldsymbol{\omega}^* - B_g \cdot \boldsymbol{a}$$

With reverse engineering, I was able to determine an approximation of the real B_g matrix, that the positive and negative orientation of one direction always had the same gyroscope output. The determined B_g matrix is (in $[\frac{rad}{s}/\frac{m}{s^2}]$)

$$B_g = \begin{bmatrix} -1.4e-5 & -0.1e-5 & 1.8e-5 \\ -2.6e-5 & 1.3e-4 & 1.2e-4 \\ 3.5e-5 & -1.5e-4 & -1.5e-4 \end{bmatrix}$$

And the result by applying this to the static experiment measurement is shown in figure 5.12. The bold lines are the calibrated outputs. We see, that all sections of one gyroscope axis approximately lie now on the same height. That confirms, that the anisoelatic bias B_g can be modeled linearly.

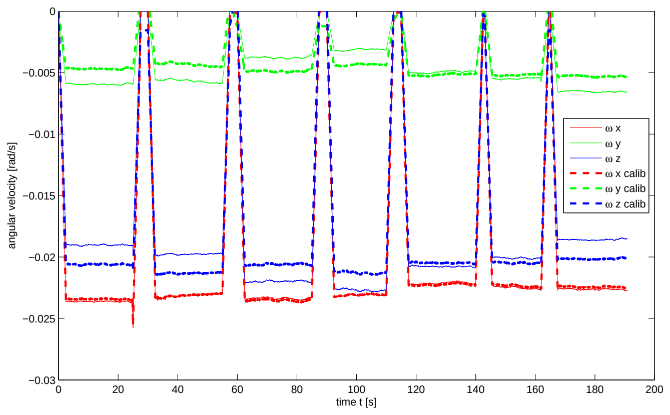

Figure 5.12: With reverse-calibration (bold lines) the sensor values approximately lie on the same height, independently of the orientation.

We can now combine these results with the detected values of the moving calibration algorithm. In figure 5.13 the resulting histograms of the matrix B_g from the calibration algorithm are drawn (blue) together with the values detected by the static experiment (red circles). As we have seen above, the values are smaller than the standard deviations. Therefore it is better, to determine the values of the matrix B_g by the static experiment and set them as constants in the calibration algorithm.

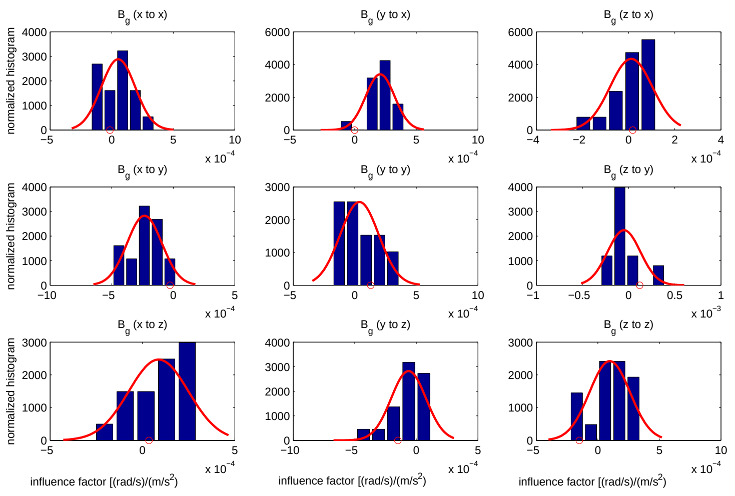

Figure 5.13: Histogram of the nine entries of the B_g matrix determined by the moving calibration algorithm. The result of the static experiment is drawn by red circles.

Chapter 6

Conclusion

The results have shown, that the misalignment and the scale factor can have an essential influence to the sensor output. It is possible to detect them with the developed calibration algorithm with some uncertainty, depending on the sensor noise level. Increasing the noise rate variance by a factor of 10 increases the calibration uncertainty by a factor of 2 - 4. All entries of the six different calibration parameters (misalignment, scale factor, rotation between accelerometer and gyroscope, transformation between camera and accelerometer, time offset between camera and IMU, acceleration influence to gyroscope) are detectable separately, without influencing each other. However, in the CIMU system used to test the calibration algorithm, the acceleration influence to gyroscope (B_g) is smaller than the corresponding standard deviation. Therefore, it is not detectable with one single calibration run over 90 seconds. But there exist another way to detect the matrix B_g. The influence can be determined by measuring the non moving gyroscope output for different orientations of the CIMU system. Hence, the gravity (a linear acceleration) affects parallel to one axis. By adjusting the matrix B_g, it is possible to align all gyroscope outputs to the same level independent of the gyroscope's orientation. The influence of linear acceleration to the gyroscope output is therefore approximately linear.
Thus, the Camera-IMU calibration algorithm allows mainly calibrating the misalignment of the sensor axis, the axis scale factor and the transformation between the different coordinate frames. For the calibration, it is important to capture as much as possible rotation and translation movements. To improve the calibration algorithm, a future work could be to expand the camera landmark detection to real scene features, in order to be not bounded to the checkerboard view. This would allow the CIMU system to rotate more than 180°, which increases the accuracy of the calibration. It would be interesting also to examine how longer datasets decrease the variance. Finally, it would be interesting to implement the new sensor model compensation in a SLAM system and compare the ground truth of walking a circuit with and without calibrated CIMU system.

Finally, I can mention, that the developed camera-IMU calibration works to detect IMU internal parameters. It can be used for systems containing cameras and IMUs without needing any mechanical calibration table. Only a checkerboard is needed. Since the algorithm is still sensitive to noise, it is important to avoid shocks by recording calibration datasets and the camera has to be intrinsic calibrated well.

Appendix A

Sensor Characteristic Values

The used IMU parameters for the MATLAB simulation, to generate the sensor data, are listed below:

Table A.1: Used IMU characteristic values for the simulation (related to the MPU-6000 datasheet)

Description	Value	SI-Unit	Reference
accelerometer noise density	0.0039	$(\mathrm{m/s^2})/\sqrt{\mathrm{Hz}}$	(D)
accelerometer random walk	0.0012	$(\mathrm{m/s})/\sqrt{s}$	(*)
accelerometer initial bias error	0.491	$(\mathrm{m/s^2})$	(D)
accelerometer misalignment -α_{yz}	0.0052	rad	(0.3 °) (D*)
accelerometer misalignment α_{zy}	-0.0017	rad	(-0.1 °) (D*)
accelerometer misalignment -α_{zx}	0.0026	rad	(0.15 °) (D*)
accelerometer scale factor s_x	1.001	[-]	(*)
accelerometer scale factor s_y	1.0008	[-]	(*)
accelerometer scale factor s_z	0.9986	[-]	(*)
gyroscope noise density	8.73E$-$5	$(\mathrm{rad/s})/\sqrt{\mathrm{Hz}}$	(D)
gyroscope random walk	0.0087	$\mathrm{rad}/\sqrt{\mathrm{s}}$	(*)
gyroscope initial bias error	0.02	rad/s	(D)
gyroscope misalignment -α_{yz}	0.0035	rad	(0.2 °) (D*)
gyroscope misalignment α_{zy}	-0.0026	rad	(-0.15 °) (D*)
gyroscope misalignment -α_{zx}	-0.0044	rad	(-0.25 °) (D*)
gyroscope scale factor s_x	1.002	[-]	(*)
gyroscope scale factor s_y	0.999	[-]	(*)
gyroscope scale factor s_z	1.0011	[-]	(*)
gyroscope acceleration influence B_g	0.00018	$(\mathrm{rad/s})/(\mathrm{m/s^2})$	(Dm)
gyroscope rotation $C_{b,g}$	±0.0175	rad	($\pm1°$) (Dm)

- Legend:
  - (D) The value corresponds to the datasheet (MPU-6000)
  - (*) The value is not available in the datasheet - an estimated value is used, based on other datasheets as ADXL320 and ADIS16488
  - (D*) The value is in the order of the datasheet (but different for all axis and usually smaller than the datasheet value)

– (Dm) The value corresponds to datasheet (MPU-6000). The used values are shown in the matrices bellow (the matrix B_g has some deviations)

$$B_g = \begin{bmatrix} -0.0012 & 0.0008 & 0.0015 \\ -0.0013 & 0.0001 & -0.0014 \\ 0.0016 & 0.0013 & -0.0009 \end{bmatrix} \left[\frac{rad}{s} / \frac{m}{s^2}\right]$$

$$C_{b,g} = \begin{bmatrix} 0.999847695 & -0.017452406 & 0 \\ 0.017452406 & 0.999847695 & 0 \\ 0 & 0 & 1 \end{bmatrix} [-]$$

$$C_{c,b} = \begin{bmatrix} -1.0000 & 0.0000 & 0.0000 \\ 0.0000 & -1.0000 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix} [-]$$

$$\boldsymbol{p}_{c,b} = \begin{bmatrix} 0.103 \\ -0.015 \\ -0.010 \end{bmatrix} [m]$$

Appendix B

Detail Result Values

B.1 Calibration of the Simulated Sensor Data

Mean and standard deviation of each 100 datasets generated in MATLAB with the same parameters but with white noise applied. Series 01 (S1) was generated with the sensor parameters described in Appendix A. Series 2 and 3 ((S2),(S3)) have the same parameters but the IMU noise is reduced by a factor of 10 respectively by a factor of 100.

Table B.1: Mean and variance of the difference (error) between the calibrator output and the 'real' reference value, used in the MATLAB file. The calibrator output itself is listed in table B.2.

DV	Mean S1	Mean S2	Mean S3	σ_{cal} S1	σ_{cal} S2	σ_{cal} S3
$\boldsymbol{p}_{c,b_{err}}$	-2.640E-05	-3.203E-05	-6.373E-05	1.715E-03	8.439E-04	3.524E-04
	-7.096E-05	-3.667E-05	5.165E-05	3.909E-03	8.685E-04	2.380E-04
	-2.106E-02	-1.998E-02	-2.002E-02	6.855E-03	1.681E-03	4.640E-04
$M_{a_{err}}$	4.933E-05	-4.392E-05	-2.916E-06	2.220E-03	2.983E-04	6.955E-05
	-1.794E-04	-1.424E-05	-2.224E-05	1.999E-03	4.456E-04	2.142E-04
	-8.920E-05	4.878E-06	-1.853E-05	1.016E-03	2.899E-04	8.108E-05
$M_{g_{err}}$	1.406E-04	4.549E-05	8.030E-06	4.352E-03	4.973E-04	8.878E-05
	-5.781E-04	-6.838E-07	-1.330E-06	4.557E-03	6.707E-04	2.081E-04
	1.800E-04	7.626E-05	5.325E-05	2.307E-03	6.393E-04	1.910E-04
$S_{a_{err}}$	3.082E-05	5.818E-06	1.239E-05	2.175E-03	2.880E-04	5.264E-05
	1.759E-04	3.818E-05	9.961E-06	1.282E-03	1.891E-04	4.801E-05
	1.656E-04	-1.712E-05	2.068E-06	1.323E-03	2.825E-04	1.011E-04
$S_{g_{err}}$	-2.435E-03	2.377E-05	1.640E-05	5.061E-03	8.903E-04	2.335E-04
	-5.218E-05	-2.862E-05	1.089E-06	3.003E-03	6.824E-04	1.842E-04
	-6.492E-05	-5.151E-06	2.678E-06	1.180E-03	2.260E-04	7.708E-05
$B_{g_{err}}$	4.356E-05	-1.708E-07	3.288E-06	2.357E-04	2.839E-05	1.532E-05
	-1.078E-05	-1.004E-06	5.578E-06	1.048E-04	1.684E-05	7.556E-06
	3.741E-06	8.672E-07	2.679E-06	2.225E-04	2.931E-05	1.354E-05
	1.183E-05	4.224E-08	-3.217E-06	2.368E-04	3.308E-05	9.953E-06
	6.443E-07	4.529E-07	-3.465E-07	8.755E-05	1.224E-05	4.148E-06
	-5.317E-06	-4.546E-06	-1.635E-06	2.015E-04	2.747E-05	6.748E-06
	3.158E-05	5.624E-06	9.451E-06	2.992E-04	7.070E-05	2.636E-05
	-2.035E-06	-1.810E-07	3.289E-06	1.530E-04	3.131E-05	6.608E-06
	-6.410E-06	2.192E-06	3.214E-06	3.577E-04	6.558E-05	1.627E-05

Table B.2: Mean and variance of the determined calibration parameters.

DV	Mean S1	Mean S2	Mean S3	σ_{cal} S1	σ_{cal} S2	σ_{cal} S3
$\bar{\boldsymbol{q}}_{c,b}$	4.199E-04	7.038E-05	2.167E-05	3.506E-04	5.241E-05	1.612E-05
[-]	-5.782E-05	-1.084E-05	2.444E-06	6.904E-04	1.118E-04	1.532E-05
	7.107E-05	3.168E-06	2.690E-06	4.429E-04	9.359E-05	1.724E-05
	1.000E+00	1.000E+00	1.000E+00	3.737E-07	1.082E-08	6.182E-10
$\boldsymbol{p}_{c,b}$	1.030E-01	1.030E-01	1.029E-01	1.715E-03	8.439E-04	3.524E-04
[m]	-1.507E-02	-1.504E-02	-1.495E-02	3.909E-03	8.685E-04	2.380E-04
	-1.106E-02	-9.979E-03	-1.002E-02	6.855E-03	1.681E-03	4.640E-04
$\bar{\boldsymbol{q}}_{b,g}$	1.000E+00	1.000E+00	1.000E+00	8.165E-06	1.076E-06	3.112E-07
[-]	-3.752E-05	4.772E-06	-1.057E-05	9.092E-04	1.869E-04	3.880E-05
	4.581E-05	-1.514E-06	-8.798E-06	8.173E-04	2.088E-04	7.000E-05
	8.853E-03	8.717E-03	8.726E-03	8.993E-04	1.228E-04	3.561E-05
$\boldsymbol{M_a}$	5.151E-03	5.244E-03	5.203E-03	2.220E-03	2.983E-04	6.955E-05
[rad]	-1.521E-03	-1.686E-03	-1.678E-03	1.999E-03	4.456E-04	2.142E-04
	2.689E-03	2.595E-03	2.619E-03	1.016E-03	2.899E-04	8.108E-05
$\boldsymbol{M_g}$	3.359E-03	3.455E-03	3.492E-03	4.352E-03	4.973E-04	8.878E-05
[rad]	-2.022E-03	-2.599E-03	-2.599E-03	4.557E-03	6.707E-04	2.081E-04
	-4.580E-03	-4.476E-03	-4.453E-03	2.307E-03	6.393E-04	1.910E-04
$\boldsymbol{S_a}$	1.001E+00	1.001E+00	1.001E+00	2.175E-03	2.880E-04	5.264E-05
[-]	1.001E+00	1.001E+00	1.001E+00	1.282E-03	1.891E-04	4.801E-05
	9.998E-01	9.998E-01	9.998E-01	1.323E-03	2.825E-04	1.011E-04
$\boldsymbol{S_g}$	1.004E+00	1.002E+00	1.002E+00	5.061E-03	8.903E-04	2.335E-04
[-]	9.991E-01	9.990E-01	9.990E-01	3.003E-03	6.824E-04	1.842E-04
	1.001E+00	1.001E+00	1.001E+00	1.180E-03	2.260E-04	7.708E-05
$\boldsymbol{B_g}$	-1.244E-03	-1.200E-03	-1.203E-03	2.357E-04	2.839E-05	1.532E-05
	8.108E-04	8.010E-04	7.944E-04	1.048E-04	1.684E-05	7.556E-06
	1.496E-03	1.499E-03	1.497E-03	2.225E-04	2.931E-05	1.354E-05
	-1.312E-03	-1.300E-03	-1.297E-03	2.368E-04	3.308E-05	9.953E-06
$\left[\frac{\frac{rad}{s}}{\frac{m}{s^2}}\right]$	9.936E-05	9.955E-05	1.003E-04	8.755E-05	1.224E-05	4.148E-06
	-1.395E-03	-1.395E-03	-1.398E-03	2.015E-04	2.747E-05	6.748E-06
	1.568E-03	1.594E-03	1.591E-03	2.992E-04	7.070E-05	2.636E-05
	1.302E-03	1.300E-03	1.297E-03	1.530E-04	3.131E-05	6.608E-06
	-8.936E-04	-9.022E-04	-9.032E-04	3.577E-04	6.558E-05	1.627E-05

B.1.1 Histograms and Plots

Distribution of the calibration output and of the error: The 'real' reference value is plotted in figure B.1 with a red arrow. In the errer plot (figure B.2), the reference is 0.

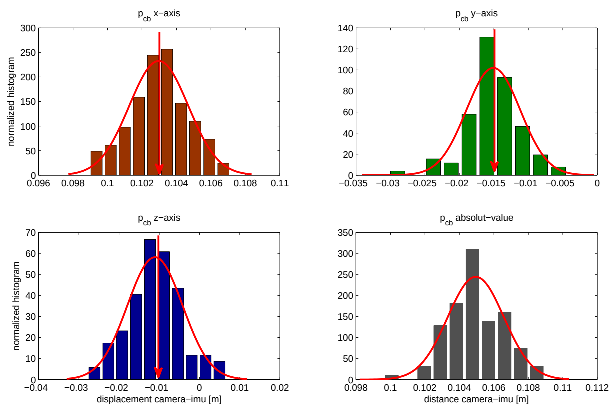

Figure B.1: Histogram of the 100 calibrated datasets. As the sensor data were generated by MATLAB the real reference value is known (red arrow). As example, the translation $\boldsymbol{p}_{c,b}$ from the camera to the body frame is shown in this figure.

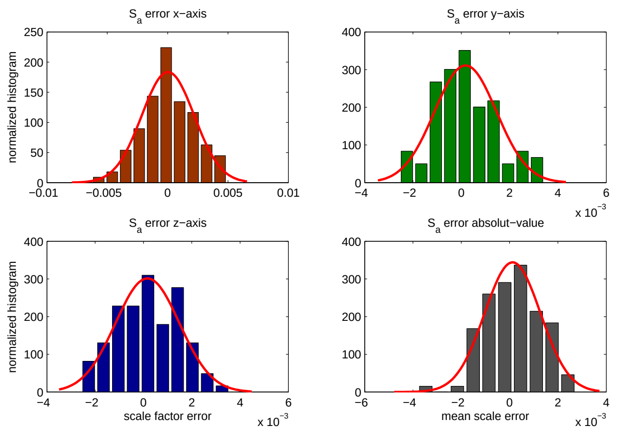

Figure B.2: Histogram of the differences between the calibration output and the reference value, used in the MATLAB function, to generate the sensor data. As example, the values of the scale matrix S_a of the accelerometer are used in this figure.

B.2 Calibration of a Real CIMU System

Total recorded datasets:	22
Total datasets used for calibration[1]:	18

The image data were used from camera 0.
Hint: The intrinsic calibration of camera 1 was not precise enough, what has resulted in inaccurate data. Therefore, this image data were not used.

[1]Hint: The datasets #1 to #4 are not used of the fact, that they were recorded by other conditions with the checkerboard rotated by 90°.

Table B.3: Calibrator output (Rotation quaternion between camera and body) [-]

Nr	$\bar{\boldsymbol{q}}_{c,b}$	i	j	k
05	0.0014254	-0.9999964	0.0015014	-0.0017139
06	0.0004735	-0.9999981	-0.0003447	-0.0018613
07	0.0004857	-0.9999981	0.0007010	-0.0017690
08	0.0008430	-0.9999979	0.0001976	-0.0018648
09	0.0002803	-0.9999988	0.0002326	-0.0014819
10	0.0012226	-0.9999973	-0.0001154	-0.0019492
11	0.0005185	-0.9999987	0.0004426	-0.0014885
12	0.0007282	-0.9999982	-0.0000287	-0.0017295
13	0.0012448	-0.9999970	0.0000465	-0.0021043
14	0.0007203	-0.9999970	0.0002407	-0.0023425
15	0.0009750	-0.9999968	0.0002058	-0.0023427
16	0.0003148	-0.9999979	0.0001928	-0.0020103
17	0.0001450	-0.9999989	0.0002518	-0.0014693
18	0.0013407	-0.9999964	0.0002305	-0.0023158
19	0.0009793	-0.9999972	-0.0001486	-0.0021637
20	0.0010089	-0.9999973	-0.0003587	-0.0020710
21	0.0011918	-0.9999976	0.0003805	-0.0017984
22	0.0009756	-0.9999973	0.0005172	-0.0020352
Mean	0.0008263	-0.9999976	0.0002303	-0.0019173
Var	1.51226E-07	5.97016E-13	1.78369E-07	7.97808E-08
σ_{cal}	0.0003889	0.0000008	0.0004223	0.0002825

Table B.4: Calibrator output (Translation between camera and body) [m]

Nr	$\boldsymbol{p}_{c,b}$ x	y	z
05	-0.0814675	-0.0107283	-0.0221052
06	-0.0796690	-0.0121275	-0.0227065
07	-0.0817452	-0.0106617	-0.0233265
08	-0.0841572	-0.0124307	-0.0232560
09	-0.0842336	-0.0113979	-0.0220847
10	-0.0824905	-0.0116150	-0.0233321
11	-0.0834794	-0.0103483	-0.0232185
12	-0.0831370	-0.0131885	-0.0223356
13	-0.0832175	-0.0106924	-0.0216024
14	-0.0851357	-0.0109839	-0.0208408
15	-0.0823265	-0.0119045	-0.0218685
16	-0.0838004	-0.0112025	-0.0216063
17	-0.0832232	-0.0115068	-0.0227816
18	-0.0826915	-0.0117618	-0.0222614
19	-0.0834032	-0.0124469	-0.0234505
20	-0.0830507	-0.0132863	-0.0222781
21	-0.0845442	-0.0109724	-0.0225447
22	-0.0840182	-0.0101441	-0.0218156
Mean	-0.0830995	-0.0115222	-0.0224119
Var	1.62612E-06	8.34936E-07	5.31580E-07
σ_{cal}	0.0012752	0.0009137	0.0007291

Table B.5: Calibrator output (Rotation quaternion between accelerometer and gyroscope) [-]

Nr	$\bar{\boldsymbol{q}}_{b,g}$	i	j	k	rot-angle[°]
05	0.9999917	0.0036003	-0.0018889	-0.0000851	0.466001
06	0.9999907	0.0035072	-0.0023752	0.0008416	0.494877
07	0.9999907	0.0039168	-0.0016294	0.0007829	0.494331
08	0.9999936	0.0035651	-0.0001582	0.0003244	0.410619
09	0.9999910	0.0041508	0.0007445	0.0004832	0.4864
10	0.9999943	0.0031938	-0.0008774	0.0005902	0.385521
11	0.9999911	0.0041903	-0.0005241	0.0002260	0.484608
12	0.9999939	0.0033470	-0.0003061	0.0009255	0.399475
13	0.9999917	0.0038492	-0.0012330	0.0005769	0.467858
14	0.9999902	0.0043592	-0.0007157	0.0002007	0.506739
15	0.9999936	0.0032703	-0.0013627	0.0005803	0.411392
16	0.9999893	0.0045955	-0.0003251	0.0002783	0.528886
17	0.9999916	0.0040343	-0.0006696	0.0001075	0.468784
18	0.9999942	0.0032802	-0.0008635	0.0003831	0.391161
19	0.9999950	0.0030235	-0.0006847	0.0006805	0.363699
20	0.9999947	0.0030550	-0.0006312	0.0009331	0.373121
21	0.9999926	0.0038453	-0.0001949	0.0001625	0.441598
22	0.9999893	0.0046035	-0.0004515	-0.0002200	0.530654
Mean	0.9999922	0.0037437	-0.0007859	0.0004317	0.450318
Var	3.40940E-12	2.55576E-07	5.15401E-07	1.13059E-07	2.94222E-3
σ_{cal}	0.0000018	0.0005055	0.0007179	0.0003362	0.054242

Table B.6: Misalignment matrix for accelerometer (left) and gyroscope (right) [rad]

Nr	$\boldsymbol{M_a}$ (m_{yz})	(m_{zy})	(m_{zx})	$\boldsymbol{M_g}$ (m_{yz})	(m_{zy})	(m_{zx})
05	0.0011379	-0.0026489	0.0043349	0.0012658	0.0063978	0.0169799
06	-0.0022313	-0.0001358	0.0018826	0.0012050	0.0070666	0.0150924
07	-0.0016830	0.0006303	-0.0000705	0.0024825	0.0062270	0.0156844
08	-0.0011915	0.0033081	0.0009306	0.0007084	0.0027507	0.0157472
09	-0.0018711	0.0063627	-0.0031930	0.0020242	0.0014798	0.0160220
10	-0.0016257	0.0007016	-0.0020333	0.0013074	0.0032202	0.0161520
11	-0.0000502	0.0022021	-0.0006508	0.0010151	0.0030258	0.0168013
12	-0.0018897	0.0020636	0.0003383	0.0020530	0.0032834	0.0148868
13	-0.0017184	0.0009591	0.0040355	0.0004394	0.0039338	0.0175789
14	-0.0001875	0.0014457	0.0015593	0.0001917	0.0031287	0.0174659
15	0.0000419	-0.0005931	0.0013508	0.0009553	0.0040070	0.0158705
16	-0.0006409	0.0023493	-0.0015320	0.0009998	0.0029195	0.0171606
17	-0.0003727	0.0001355	-0.0008174	0.0004546	0.0032902	0.0162917
18	-0.0019641	0.0006359	0.0030820	0.0008978	0.0028400	0.0162886
19	-0.0025815	-0.0004793	0.0003531	0.0010446	0.0023977	0.0148602
20	-0.0026264	-0.0000660	0.0042180	0.0014329	0.0029595	0.0152608
21	-0.0004065	0.0036200	0.0012821	0.0011487	0.0029724	0.0175789
22	-0.0008757	0.0041765	0.0014230	0.0000221	0.0029615	0.0182283
Mean	-0.0011520	0.0013704	0.0009163	0.0010916	0.0036034	0.0163306
Var	1.0733E-06	4.3965E-06	4.5648E-06	4.0837E-07	2.1664E-06	1.0214E-06
σ_{cal}	0.0010360	0.0020968	0.0021365	0.0006390	0.0014719	0.0010106

Table B.7: Scale factor matrix entries for accelerometer (left) and gyroscope (right)

Nr	$\boldsymbol{S_a}$ (s_x)	(s_y)	(s_z)	$\boldsymbol{S_g}$ (s_x)	(s_y)	(s_z)
05	1.0027612	0.9986304	1.0062298	1.0025823	0.9974070	1.0054031
06	1.0043414	0.9992792	1.0093589	1.0042454	0.9976496	1.0046972
07	1.0031097	0.9987360	1.0079992	1.0040931	0.9974098	1.0045415
08	1.0018780	0.9992454	1.0094814	1.0044117	0.9976504	1.0047849
09	1.0032742	1.0003065	1.0090443	1.0042010	0.9966733	1.0048935
10	1.0021418	1.0006684	1.0095405	1.0049157	0.9977059	1.0045361
11	1.0004808	1.0007987	1.0130320	1.0049358	0.9959325	1.0049690
12	1.0012811	1.0002110	1.0117446	1.0041234	0.9963138	1.0047671
13	1.0010886	1.0009130	1.0129002	1.0031473	0.9958562	1.0045545
14	1.0021604	0.9995972	1.0116586	1.0025070	0.9959216	1.0047270
15	1.0028810	1.0001337	1.0102772	1.0030916	0.9968580	1.0047188
16	1.0031269	1.0003004	1.0134755	1.0036973	0.9960863	1.0040788
17	1.0021462	0.9996706	1.0100027	1.0042043	0.9962371	1.0036410
18	1.0008640	1.0003236	1.0115004	1.0031423	0.9972290	1.0051345
19	1.0012814	1.0004662	1.0115395	1.0045132	0.9979315	1.0052403
20	1.0009238	1.0002328	1.0092339	1.0034577	0.9968443	1.0053799
21	1.0029187	1.0015154	1.0133440	1.0039494	0.9978997	1.0045469
22	1.0025101	1.0004577	1.0139522	1.0026965	0.9962358	1.0044969
Mean	1.0021761	1.0000826	1.0107953	1.0037730	0.9968801	1.0047284
Var	1.0713E-06	5.7399E-07	4.4826E-06	5.8826E-07	5.5184E-07	1.8768E-07
σ_{cal}	0.0010350	0.0007576	0.0021172	0.0007670	0.0007429	0.0004332

Table B.8: Acceleration influence to gyroscope bias [(rad/s)/(m/s^2)]

Nr	$\boldsymbol{B_g}$ (b_{xx})	(b_{yx})	(b_{zx})	(b_{xy})	(b_{yy})	(b_{zy})
05	0.0001057	0.0003342	-0.0000664	-0.0002955	0.0001735	0.0003939
06	0.0000599	0.0002050	0.0000333	-0.0004878	0.0000150	-0.0000560
07	-0.0000221	0.0003776	0.0000579	-0.0004916	0.0003680	0.0003472
08	-0.0000788	0.0004039	0.0000585	-0.0001312	0.0001997	-0.0000993
09	0.0001178	0.0002054	0.0000281	-0.0002645	0.0002246	-0.0000277
10	-0.0000957	0.0002157	0.0000366	-0.0001638	0.0003000	0.0000806
11	-0.0001695	0.0001939	0.0000312	-0.0004204	-0.0001310	-0.0001620
12	-0.0001653	0.0002125	-0.0000253	-0.0001029	0.0000380	-0.0001409
13	-0.0000005	0.0000990	0.0000016	-0.0002733	-0.0001285	-0.0003024
14	0.0000935	0.0001624	0.0001026	-0.0002141	-0.0000362	-0.0000993
15	0.0000262	0.0001442	-0.0000694	-0.0002732	0.0000122	-0.0000146
16	0.0001092	0.0001635	0.0000686	-0.0001397	-0.0001264	-0.0002139
17	-0.0000684	0.0002832	0.0000156	-0.0000433	-0.0000502	-0.0001619
18	0.0002342	0.0001040	-0.0001532	-0.0002057	-0.0001001	-0.0001704
19	0.0001239	-0.0001219	-0.0002306	0.0000241	0.0000713	-0.0001535
20	0.0003477	0.0002217	0.0000556	-0.0001341	-0.0001806	-0.0000802
21	0.0001950	0.0002796	0.0001146	-0.0003194	-0.0000066	0.0000366
22	0.0001424	0.0002322	0.0001186	-0.0002658	0.0000443	-0.0000653
Mean	0.0000531	0.0002065	0.0000099	-0.0002335	0.0000382	-0.0000494
Var	1.9135E-08	1.3797E-08	8.3257E-09	1.9841E-08	2.4984E-08	3.1543E-08
σ_{cal}	0.0001383	0.0001175	0.0000912	0.0001409	0.0001581	0.0001776

Table B.9: ... B_g continued (values are sorted by column major

Nr	$\boldsymbol{B_g}$ (b_{xz})	(b_{yz})	(b_{zz})
05	0.0001540	-0.0000745	-0.0002282
06	0.0002286	-0.0000158	0.0000016
07	0.0000823	0.0000956	-0.0001522
08	-0.0000161	-0.0001062	0.0001183
09	0.0000072	-0.0002486	-0.0000511
10	-0.0001300	-0.0004898	-0.0001495
11	0.0002389	-0.0000503	0.0002096
12	0.0002466	0.0001241	0.0003463
13	0.0001291	0.0000400	0.0003385
14	0.0002297	0.0000544	0.0001433
15	0.0001095	-0.0000192	0.0001818
16	0.0002754	-0.0000085	0.0002442
17	-0.0000147	-0.0001561	0.0001109
18	-0.0001281	-0.0001272	0.0001735
19	-0.0002605	0.0000039	0.0002557
20	-0.0000540	0.0000208	0.0000578
21	0.0001745	-0.0000219	0.0000437
22	0.0002984	-0.0001382	0.0001095
Mean	0.0000873	-0.0000621	0.0000974
Var	2.61707E-08	2.01164E-08	2.70591E-08
σ_{cal}	0.0001618	0.0001418	0.0001645

B.2.1 Histogram of the Results

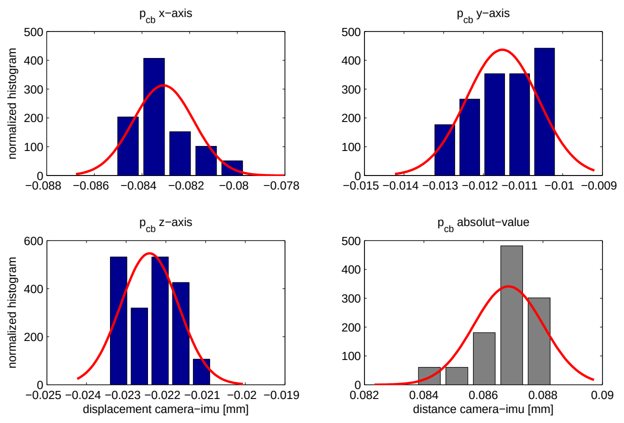

Figure B.3: Histogram of the translation vector $\boldsymbol{p}_{c,b}$ from camera to body frame.

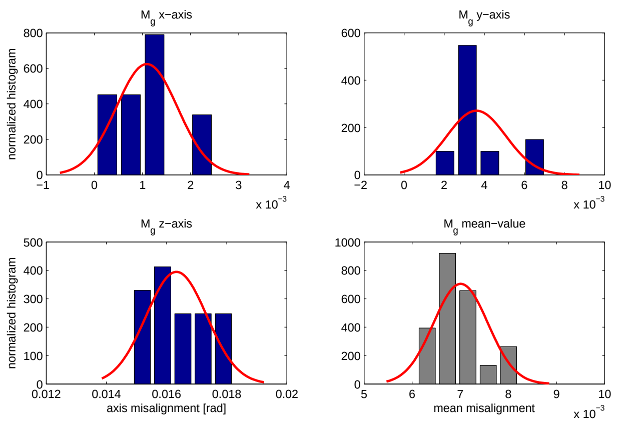

Figure B.4: Histogram of the misalignment matrix M_g of the gyroscope.

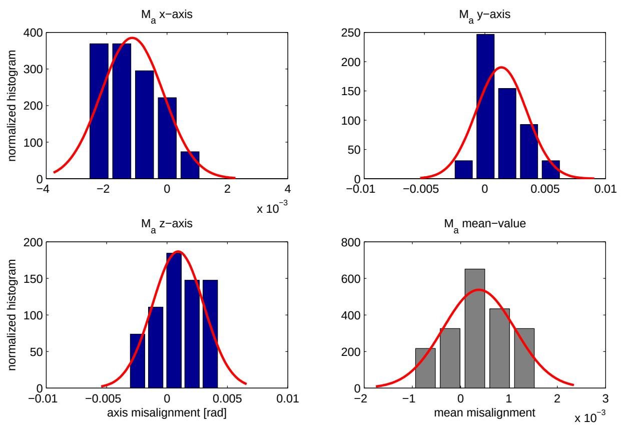

Figure B.5: Histogram of the misalignment matrix M_a of the accelerometer.

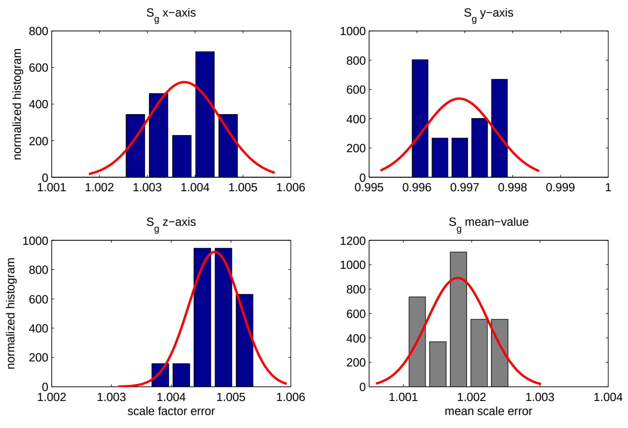

Figure B.6: Histogram of the scale factor matrix S_g of the gyroscope axes.

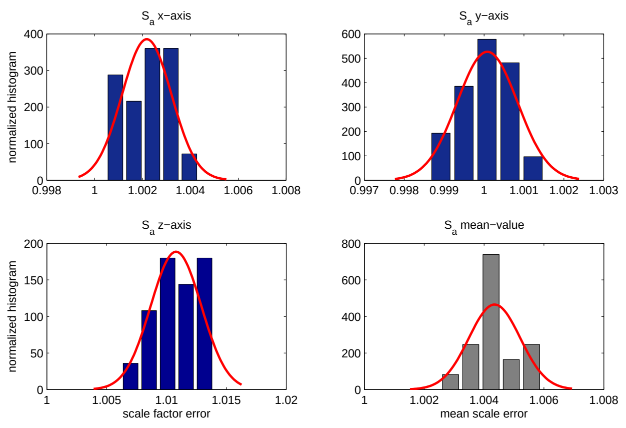

Figure B.7: Histogram of the scale factor matrix S_a of the accelerometer axes.

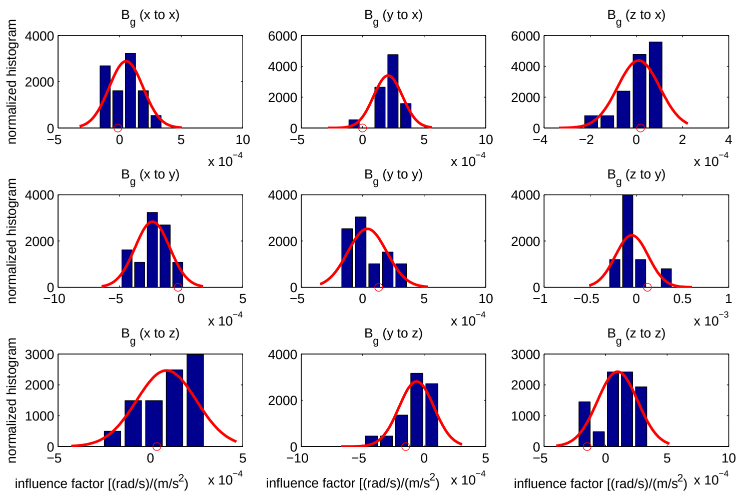

Figure B.8: Histogram of the nine entries of the matrix B_g determined by the moving calibration algorithm. The result of the static experiment is drawn by red circles.

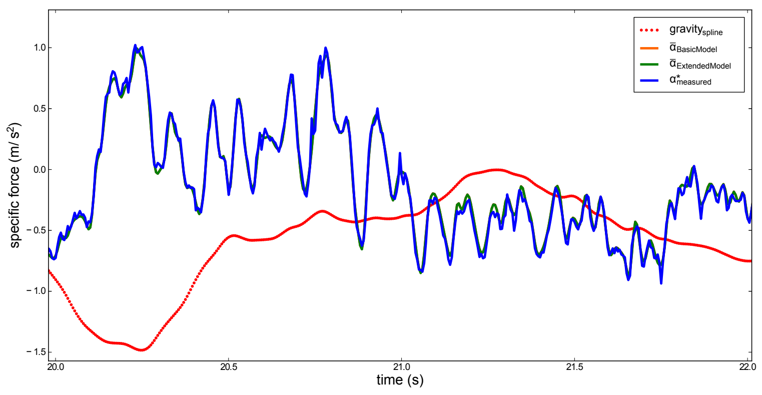

Figure B.9: Section of the accelerometer's x-axis measurement (blue). The estimated sensor output with the extended model is drawn green, the corresponding estimation with the basic model is drawn orange.

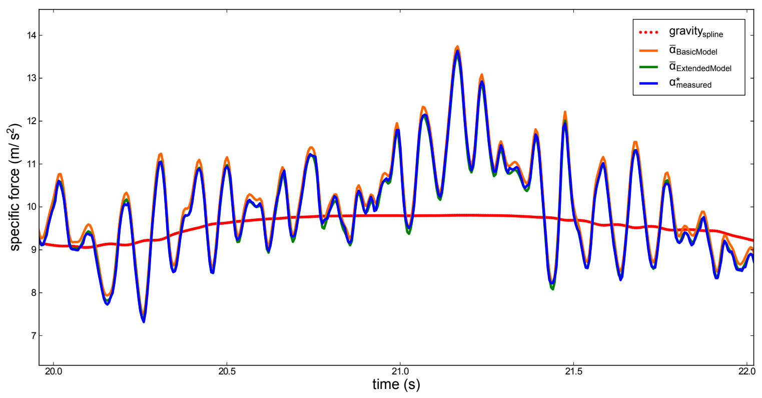

Figure B.10: Section of the accelerometer's y-axis measurement (blue). The estimated sensor output with the extended model is drawn green, the corresponding estimation with the basic model is drawn orange. (Main influence: M_g)

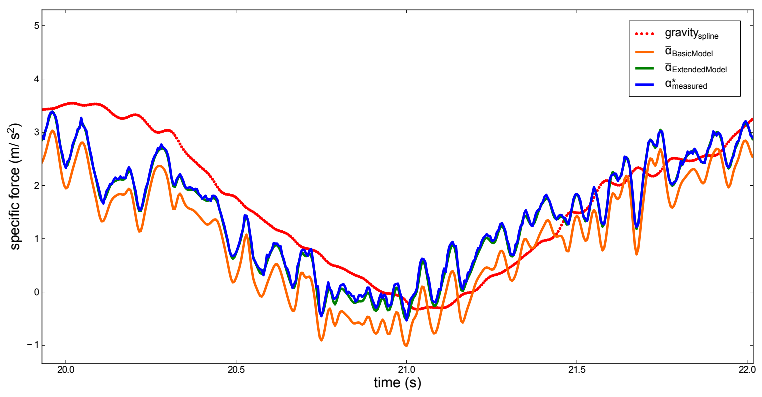

Figure B.11: Section of the accelerometer's z-axis measurement (blue). The estimated sensor output with the extended model is drawn green, the corresponding estimation with the basic model is drawn orange. (Main influence: S_{a_z})

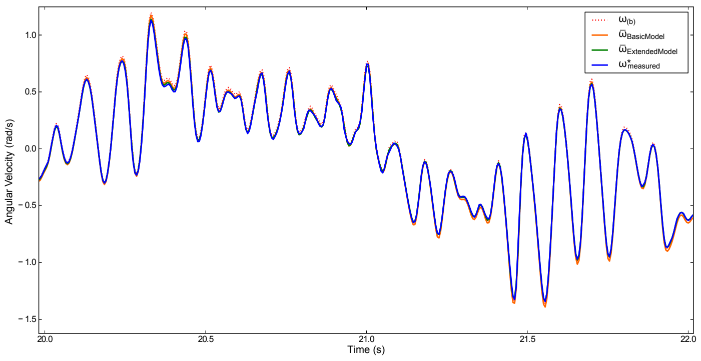

Figure B.12: Section of the gyroscope's x-axis measurement (blue). The determined angular velocity in the body frame $\omega_{(B)}$ (red doted), the estimated sensor output by the extended model (green) and the corresponding sensor output of the basic model (orange) are plotted.

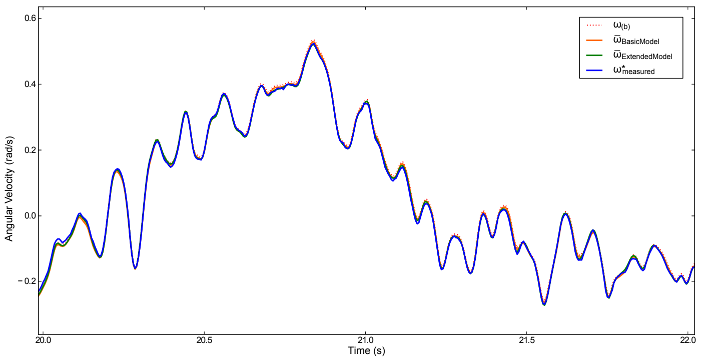

Figure B.13: Section of the gyroscope's y-axis measurement (blue). The determined angular velocity in the body frame $\omega_{(B)}$ (red doted), the estimated sensor output by the extended model (green) and the corresponding sensor output of the basic model (orange) are plotted.

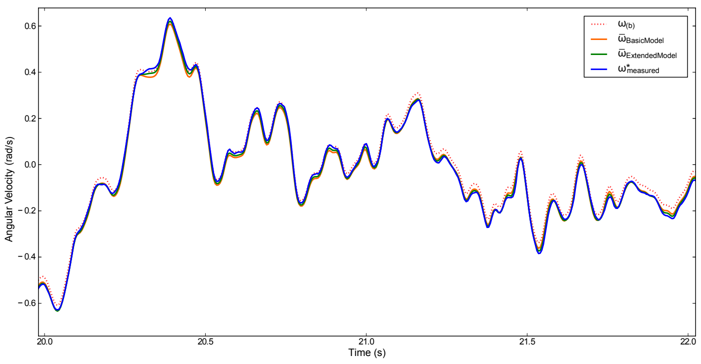

Figure B.14: Section of the gyroscope's z-axis measurement (blue). The determined angular velocity in the body frame $\omega_{(B)}$ (red doted), the estimated sensor output by the extended model (green) and the corresponding sensor output of the basic model (orange) are plotted. (Main influence: M_g)

Bibliography

[1] J. Titterton, D. Weston: *Strapdown Inertial Navigation Technology.* Institution of Electrical Engineers, 2004.

[2] A.-B. Chatfield: *Fundamentals of High Accuracy Inertial Navigation.* AIAA (American Institute of Aeronautics & Astronautics), Volume 174, 1997.

[3] P. Furgale, T.-D. Barfoot, G. Sibley: *Continuous-Time Batch Estimation using Temporal Basis Function.* IEEE International Conference on Robotics and Automation, Minnesota USA, 2012.

[4] P. Furgale, J. Rehder, R. Siegwart: *Unified Termporal and Spatial Calibration for Multi-Sensor Systems.* ASL ETH, Zurich, 2012.

[5] P.-C. Hughes: *Spacecraft attitude dynamics.* Dover Publications, 1986.

[6] I. Skog, P. Händel: *Calibration of a MEMS Inertial Measurement Unit.* XVII IMEKO World Congress, Rio de Janeiro, Brazil, 2006.

[7] T. Nieminen, J. Kangas, S. Suuriniemi, L. Kettunen: *An enhanced multi-position calibration method for consumer-grade inertial measurement units applied and tested.* Measurement Science and Technology, 2010.

[8] W.-T. Fong, S.-K. Ong, A.-Y. Nee: *Methods for in-field user calibration of an inertial measurement unit without external equipment.* Measurement Science and Technology, 2008.

[9] P.-T. Furgale: *Extensions to the Visual Odometry Pipeline for the Exploration of Planetary Surfaces.* University of Toronto, 2011.

[10] B. Nelson, S. Stoeter, P. Korba: *Theory of Robotics and Mechatronics (151-0601-00).* Lecture course, ETH Zurich, 2012.

[11] N. Trawny, S.-I. Roumeliotis: *Indirect Kalman Filter for 3D Attitude Estimation, A Tutorial for Quaternion Algebra.* Department of Compputer Science and Engineering, University of Minnesota, 2005.

[12] F.-M. Mirzaei, S.-I. Roumeliotis: *A Kalman Filter-Based Algorithm for IMU-Camera Calibration: Observability Analysis and Performance Evaluation.* Robotics, IEEE Transactions on Vol.24 No.5 , 2008.