Universität Rostock

Traditio et Innovatio

Konzepte, Methoden und Anwendungen zur Analyse und Optimierung von eingebetteten Inertialsensorsubsystemen bei menschlicher Interaktion

Dissertation

Vorgelegt von:

Nils Büscher,

geboren am 27.07.1989 in Witten

zur Erlangung des
akademischen Grades eines Doktor-Ingenieur (Dr.-Ing.)
der Fakultät für Informatik und Elektrotechnik
der Universität Rostock

https://doi.org/10.18453/rosdok_id00004355

UNIVERSITÄT ROSTOCK | FAKULTÄT FÜR INFORMATIK UND ELEKTROTECHNIK
Institut für Angewandte Mikroelektronik und Datentechnik
Albert-Einstein-Str. 26, 18059 Rostock | Tel: +49 381 498-7251

Gutachter

Prof. Dr.-Ing. habil. Christian Haubelt, Universität Rostock,
Institut für Angewandte Mikroelektronik und Datentechnik

Prof. Dr.-Ing. Michael Hübner, B-TU Cottbus-Senftenberg,
Fakultät 1 - Fachgebiet Technische Informatik

Prof. Dr. rer. nat. Matthias Kranz, Universität Passau,
Lehrstuhl für Informatik mit Schwerpunkt Eingebettete Systeme

Datum der Einreichung: 07.03.2022

Datum der Verteidigung: 31.03.2023

Danksagung

Mein Dank gilt Prof. Dr. Ing. habil. Christian Haubelt, der mich bei meiner Dissertation als Doktorvater betreut und unterstützt hat. Er hat mir die Freiheit gegeben, meine Forschungen in die Richtungen zu lenken, die ich als relevant und Wissen-bringend erachtet habe und hat dafür gesorgt, dass ich meine Dissertation durchführen konnte. Weiterhin möchte ich mich bei Prof. Dr. Michael Hübner von der B-TU Cottbus-Senftenberg für das Zweitgutachten bedanken. Auch für die hilfreichen Anmerkungen und Fragen, die diese Dissertationsschrift bereichert haben.

Besonders danken möchte ich meiner Frau Claudia Büscher und meiner Familie für die Unterstützung und vielen lieben sowie unterstützenden Worte während der Erarbeitung meiner Dissertation sowie beim Schreiben dieser Dissertationsschrift. Danken möchte ich außerdem meinen Kollegen und besonders Herrn Daniel Gis für die gute und produktive Zusammenarbeit. Zu guter Letzt gilt mein Dank auch Rainer Dorsch und Bosch Sensortec für die Zusammenarbeit und Forschungskooperation.

Eidesstattliche Versicherung

Hiermit versichere ich, dass ich die von mir vorgelegte Arbeit mit dem Titel

Konzepte, Methoden und Anwendungen zur Analyse und Optimierung von eingebetteten Inertialsensorsubsystemen bei menschlicher Interaktion

selbständig und ohne unerlaubte fremde Hilfe angefertigt, keine anderen als die angegebenen Quellen und Hilfsmittel verwendet und die den verwendeten Quellen und Hilfsmitteln wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Rostock, den 07.03.2022

Nils Büscher

Zusammenfassung

Inertialsensoren spielen eine zentrale Rolle bei der Umsetzung zahlreicher neuer technischer Anwendungen, insbesondere im Bereich der Mensch-Maschine-Interaktion (MMI). Dazu zählen Methoden wie Gestenerkennung für die Steuerung eines Gerätes, virtuelle Zeigegeräte, Headsets für Virtuelle Realität (VR) oder Indoor-Navigation. Durch immer leistungsfähigere Hardware ist eine fast unbegrenzte Anzahl unterschiedlicher Anwendungen möglich. Auch in kleineren eingebetteten Systemen werden immer häufiger die Daten der Inertialsensoren direkt verarbeitet und genutzt. Damit die Anwendungen verlässlich funktionieren, müssen die Messdaten der Inertialsensoren sowie die durch Sensordatenfusion erzeugten Ergebnisse genau, robust und zuverlässig sein. Besonders bei Smartphones haben Untersuchungen aber gezeigt, dass die Qualität der Sensordaten und der Sensordatenfusion sehr stark variieren kann. Bei der Entwicklung von Anwendungen sowohl auf Smartphones, als auch auf anderen eingebetteten Systemen stellen die Unterschiede bei der Qualität der Sensoren oder der Sensordatenfusion eine große Herausforderung dar.

Das Ziel der in dieser Dissertationsschrift beschriebenen Arbeiten ist es, Entwickler:Innen eine Hilfestellung bei der Erstellung von zuverlässigen und genauen Anwendungen zu geben. Die Arbeiten stellen neuartige Konzepte und Methoden vor, mit denen es auf einfache Weise und ohne teure externe Hilfsmittel möglich ist, die Qualität der Sensordatenfusion oder des Inertialsensorsubsystems zu beurteilen. Hierbei werden Konzepte, Methoden und Anwendungen auf zwei Abstraktionsebenen vorgestellt. Zum einen auf der Anwendungsebene, auf der die Anwendungen auf Daten von bereits vorhandenen Inertialsensorsubsystemen zugreifen. Zum anderen auf der Firmware-Ebene, auf der direkt auf dem eingebetteten System eine Verarbeitung oder Fusion der Sensordaten stattfinden soll.

Weiterhin werden Erkenntnisse aus Untersuchungen vorgestellt, die als Entscheidungshilfe dienen sollen, um bei der Entwicklung die richtigen Algorithmen und Datenformate auszuwählen. Zuletzt wird eine Anwendung vorgestellt, mit der es auf schnelle und unkomplizierte Weise möglich ist, Sensordaten zu manipulieren und zu erweitern, um Anwendungen zu entwickeln und zu testen, oder um Algorithmen zur Gestenerkennung zu trainieren und zu verifizieren.

Abstract

Inertial sensors play an important role in the implementation of numerous new technical applications, especially in the field of human–machine interfaces (HMI). These include methods such as gesture recognition for controlling a device, digital pointing devices, headsets for virtual reality (VR) or indoor navigation. Due to increasingly powerful hardware, an almost unlimited number of different applications is possible. However, also on smaller embedded systems, data from inertial sensors is processed and used directly more often. In order for the applications to function dependably, the data measured from the inertial sensors and the results generated by sensor fusion have to be accurate, robust and reliable. However, especially for smartphones, studies have shown that the quality of sensor data and sensor data fusion varies dramatically. When developing applications on both smartphones and embedded systems, the differences in the quality of sensors or sensor fusion poses a great challenge.

The goal of the works presented in this thesis is to provide guidance to developers in developing reliable and accurate applications. This thesis presents concepts and methods that make it possible to assess the quality of sensor data fusion of the inertial sensor subsystem in a simple way and without the requirement for expensive external tools. The concepts, methods, and applications are presented at two levels of abstraction. First, at the application level, where applications access the data from already existing inertial sensor subsystems. Secondly, on the firmware level, where processing or fusion of sensor data is implemented directly on the firmware of the embedded system.

Furthermore, findings from investigations are presented, which should serve as a decision support to select the right algorithms and data formats during development for a selected use case. Finally, an application is presented with which it is possible to manipulate and augment sensor data in a fast and uncomplicated way to evaluate applications, train and verify algorithms for gesture recognition or assess processing steps of the sensor data prior to being implemented in the firmware.

Inhaltsverzeichnis

Abkürzungsverzeichnis xiii

Abbildungsverzeichnis xv

Tabellenverzeichnis xix

Publikationen xxi

1 Einleitung 1
- 1.1 Motivation und Zielsetzung . . . 2
- 1.2 Publikationen . . . 3
- 1.3 Aufbau der Dissertationsschrift . . . 5
- 1.4 Hinweise . . . 6

2 Wissenschaftlicher Kontext und Stand der Technik 7
- 2.1 Wissenschaftlicher Kontext . . . 7
- 2.2 Stand der Technik . . . 9
- 2.3 Wissenschaftliche Beiträge . . . 13

3 Grundlagen 15
- 3.1 Mensch-Maschine-Interaktion . . . 15
- 3.2 Eingebettete Systeme . . . 19
- 3.3 Mathematische Begriffe . . . 24
  - 3.3.1 Transformationsmatrizen . . . 24
  - 3.3.2 Homogene Koordinaten . . . 25
  - 3.3.3 Quaternionen . . . 25
- 3.4 Inertialsensoren . . . 27
- 3.5 Sensordatenverarbeitung . . . 29
  - 3.5.1 Vorverarbeitung . . . 29
  - 3.5.2 Sensordatenfusion . . . 29
- 3.6 Sensorfehler und Fehler bei der Sensordatenfusion . . . 34
- 3.7 Statistische Bewertungsmethoden . . . 37
- 3.8 Bewertung der Sensordaten . . . 39

4 Methoden und Untersuchungen auf Anwendungsebene **47**
4.1 Bewertung der Smartphone-Sensoren . . . 48
4.1.1 Probleme der bestehenden Messmethode . . . 49
4.1.2 Nutzerführung . . . 51
4.1.3 Statistische Fehleranalyse . . . 56
4.1.4 Weitere Tests . . . 62
4.1.5 Gesammelte Ergebnisse . . . 63
4.1.6 Zusammenfassung und Bewertung . . . 67
4.2 Bewertung des Magnetometers . . . 69
4.2.1 Konzept . . . 69
4.2.2 Implementierung . . . 70
4.2.3 Ergebnisse . . . 72
4.2.4 Bewertung und Abschwächung von Nutzungsfehlern . . . 73
4.2.5 Mögliche Verbesserungen durch Bildanalyse . . . 75
4.3 Bewertung mittels Roboterarm . . . 78
4.4 Bewertung von digitalen Zeigegeräten . . . 80
4.4.1 Test für Handhabbarkeit . . . 82
4.4.2 Test für Reaktionsfähigkeit & Geschwindigkeit . . . 84
4.4.3 Test für Präzision & Stabilität . . . 85
4.4.4 Testdurchführung . . . 88
4.4.5 Ergebnisse . . . 89
4.5 Zusammenfassung . . . 95

5 Methoden und Untersuchungen auf Firmware-Ebene **97**
5.1 Sensor-in-the-Loop . . . 98
5.2 Funktionale und extrafunktionale Eigenschaften . . . 99
5.2.1 Genutzte Hardware . . . 99
5.2.2 Extrafunktionale Eigenschaften . . . 100
5.2.3 Funktionale Eigenschaften . . . 108
5.2.4 Zusammenfassung . . . 119
5.2.5 Weiterführende Untersuchungen . . . 119
5.3 Augmentation von Inertialsensor-Daten . . . 120
5.3.1 Konzept . . . 121
5.3.2 Implementierung . . . 125
5.3.3 Anwendungsbeispiele . . . 129
5.3.4 Zusammenfassung . . . 138
5.3.5 Zukünftige Entwicklung . . . 139
5.4 Zusammenfassung . . . 140

6 Zusammenfassung der Dissertationsschrift **141**
6.1 Bewertung der Ergebnisse . . . 142
6.2 Ausblick . . . 144

A Literaturverzeichnis **I**

B Liste der Veröffentlichungen und Fachvorträge auf Tagungen **XV**

C Liste der betreuten studentischen Arbeiten **XVII**

D Anhang **XIX**

D.1 Berechnung der Varianzanalysen XIX
D.2 Berechnungen der Messsystemanalyse XXI
D.3 Berechnungen des Konfidenzintervalls XXII
D.4 Berechnungen der linearen Regression XXII
D.5 Messungen für die MEMS-Score-Anwendung XXIV
D.6 Messungen für den Test des Magnetometers XXVIII
D.7 Ergebnisse des Tests für digitale Zeigegeräte XXIX
D.8 Messergebnisse der Untersuchung der Sensorfusionsalgorithmen XXX

Abkürzungsverzeichnis

µC . . . . . Mikrocontroller
AHRS . . . Haltungs- & Kursreferenzsystem (*Attitude & Heading Reference System*)
ALU . . . . . arithmetisch-logische Einheit (*Arithmetic Logic Unit*)
AR . . . . . Erweiterte Realität (*Augmented reality*)
CPU . . . . . Zentrale Recheneinheit (*Central Processing Unit*)
DOF . . . . . Freiheitsgrade (*Degrees of Freedom*)
DSP . . . . . Digitaler Signalprozessor
EIS . . . . . digitale Bildstabilisierung (*Electronic Image Stabilization*)
FB . . . . . Funktionsblock
FIR . . . . . Filter mit endlicher Impulsantwort (*Finite Impulse Response*)
FOV . . . . . Sichtfeld (*Field of View*)
FPGA . . . . . Field Programmable Gate Array
FPU . . . . . Gleitkommaeinheit (*Floating Point Unit*)
Gage R&R . . . . . Messsystemanalyse (*Gage Repeatability & Reproducibility*)
GPS . . . . . Globales Positionsbestimmungssystem (*Global Positioning System*)
GUI . . . . . Grafische Benutzeroberfläche (*Graphical User Interface*)
IIR . . . . . Filter mit unendlicher Impulsantwort (*Infinite Impulse Response*)
IMU . . . . . Inertiale Messeinheit (*Inertial Measurement Unit*)
IoT . . . . . Internet der Dinge (*Internet of Things*)
ISO . . . . . Internationale Organisation für Normung
I²C . . . . . Inter-Integrated Circuit
MEMS . . . . . Mikro-Elektro-Mechanische Systeme
MMI . . . . . Mensch-Maschine-Interaktion
MSA . . . . . Messsystemanalyse
OIS . . . . . optische Bildstabilisierung (*Optical Image Stabilization*)
PnP . . . . . Perspective-n-Point
RAM . . . . . Arbeitsspeicher (*Random-Access Memory*)
ROM . . . . . Festwertspeicher (*Read-Only Memory*)

SiL . . . . . Sensor-in-the-Loop
SPI . . . . . Serial Peripheral Interface
UAV . . . . . Unbemanntes Luftfahrzeug (*Unmanned Aerial Vehicle*)
VR . . . . . Virtuelle Realität
WMM . . . . . World Magnetic Model

Abbildungsverzeichnis

1.1 Einordnung der wissenschaftlichen Publikationen 3

3.1 Beispiel einer Anwendung mit Gestenerkennung: Ein Smartpen 16
3.2 Beispiel eines Tests für Fitt's Law mit zwei Zielen. Abgewandelt aus [A30] 17
3.3 Beispiel eines kreisförmigen Tunnels. Abgewandelt aus [A32] 18
3.4 Bekanntes Beispiel eines eingebetteten Systems: Das Smartphone 19
3.5 Aufbau einer Fließkommazahl nach dem IEEE 754 Standard 21
3.6 Aufbau einer Festkommazahl mit Vorzeichen 22
3.7 Aufbau eines intelligenten Sensors 23
3.8 Beispiel eine Transformation im zweidimensionalen Raum 24
3.9 Aufbau eines Mikro-Elektro-Mechanische Systeme (MEMS)-Sensors . . . 28
3.10 IMU mit jeweils drei orthogonalen Achsen 28
3.11 Linearer und nicht-linearer Verlauf 35
3.12 Auswirkung elektromagnetischer Störungen auf das Magnetometer 35
3.13 Veranschaulichung des Messvorgangs für die Kamera-Referenz 39
3.14 Messablauf für die Bestimmung der Genauigkeit der Fusionsalgorithmen . 40
3.15 Schritte zum Bestimmen der Referenz 41
3.16 Beziehung zwischen Weltkoordinaten und Kamerakoordinaten 41
3.17 Kreuzkorrelation zwischen Messdaten und Referenzdaten 44

4.1 Messunterschiede unter gleichen Bedingungen 49
4.2 Messunterschiede zwischen zwei Testpersonen 49
4.3 Umgestaltung von virtuellem Horizont zu Fadenkreuz mit Markierungen 51
4.4 Benutzeroberfläche für die Drehung des Smartphones beim Sensortest . . 52
4.5 Einfärbung des Fadenkreuzes bei Abweichung von der Bewegungsvorgabe 53
4.6 Wiederholbarkeit und Reproduzierbarkeit der Sensorbewertung 54
4.7 Unterschiedliche Messergebnisse bei gleichen Ausgangsbedingungen . . . 55
4.8 Visualisierung einer Kalibrierung der Sensoren während der Messung . . 57
4.9 Drift in den Sensordaten sowie Messfehler und Regressionsanalyse 58
4.10 Offset in den Sensordaten sowie Messfehler und Regressionsanalyse . . . 59
4.11 Skalierungsfehler der Sensordaten sowie Messfehler und Regressionsanalyse 60
4.12 Verteilung der Fehlerarten bei den untersuchten Geräten 61
4.13 Ergebnisse der MEMS-Score-Anwendung gruppiert nach Hersteller 65
4.14 Ergebnisse der MEMS-Score-Anwendung gruppiert nach Sensorfusion . . 66

4.15 Vergleich von *High-Fidelity* und normalen Sensoren 67
4.16 Ablauf des Tests für die absolute Orientierung 70
4.17 Phase 2 des Tests: Auswahl der Bezugspunkte 71
4.18 Phase 3 des Tests: Messung 71
4.19 Auswirkung einer Drehung um das falsche Rotationszentrum 73
4.20 Beispiel einer Fehlkalibrierung des Magnetometers 75
4.21 Bestimmung der Template-Position bei der zweiten Messung 76
4.22 Ergebnisse der Messung der Sensordatenfusion mit einem Roboterarm . . 78
4.23 GUI für den Test zur Handhabbarkeit und die Ergebnisse 82
4.24 GUI für den Test für Genauigkeit und für die Ergebnisse 85
4.25 Optimaler Pfad und gemessener Pfad eines instabilen Zeigers 87
4.26 Gesamtwertung der Handhabbarkeit der Zeigegeräte 89
4.27 Bewertung der Reaktionsfähigkeit & Geschwindigkeit der Zeigegeräte . . 90
4.28 Gesamtbewertung der Genauigkeit der Zeigegeräte 91
4.29 Verteilung der Testergebnisse der Zeigegeräte für die Genauigkeit 92
4.30 Normalisierter Mittelwert für die Gesamtbewertung 93
4.31 Normalisierte subjektive Bewertung der Testpersonen 94

5.1 Datenfluss zum Aufzeichnen und Injizieren der Daten mit der SiL-Plattform 98
5.2 Größe der Binärdateien im ROM ohne Nutzung des Magnetometers . . . 101
5.3 Größe der Binärdateien im ROM mit Nutzung des Magnetometers 102
5.4 Benötigter Arbeitsspeicher ohne Verwendung des Magnetometers 103
5.5 Messaufbau zum Ermitteln der Ausführungszeit der Fusionsalgorithmen . 104
5.6 Mittlere Ausführungszeit der Fusionsalgorithmen ohne Magnetometer . . 105
5.7 Mittlere Ausführungszeit der Fusionsalgorithmen mit Magnetometer . . . 106
5.8 Standardabweichung der Ausführungszeiten der Algorithmen 107
5.9 Verteilung der Ausführungszeiten der Algorithmen 107
5.10 Mittlerer Winkelfehler beeinflusst vom Drift auf der z-Achse 109
5.11 Beispiel einer Messung mit Drift auf der z-Achse (in Euler-Winkeln) . . . 109
5.12 Mittlerer Winkelfehler nach der Korrektur des Drifts 110
5.13 Einfluss der Messfaktoren auf das Ergebnis (ohne Magnetometer) 111
5.14 Einfluss der Faktoren gruppiert nach Datenformat (ohne Magnetometer) 112
5.15 Mittlerer Winkelfehler mit Verwendung des Magnetometers 113
5.16 Einfluss der Messfaktoren auf das Ergebnis (mit Magnetometer) 114
5.17 Einfluss der Faktoren gruppiert nach Datenformat (mit Magnetometer) . 115
5.19 Vergleich von mittlerer Winkelgeschwindigkeit zu mittlerem Fehler 117
5.20 Vergleich von Durchführungsgenauigkeit zu mittlerem Fehler 118
5.21 Mögliche Versuchsaufbauten für das Frameworks für Datenaugmentation 122
5.22 Beispiel für eine Manipulation der Daten. 126
5.23 Datenfluss des Beispiels von Abbildung 5.22 126
5.24 Kontrollfluss eines Funktionsblocks bei einer Datenabfrage 127
5.25 GUI des Node-Editors für die Manipulation der Sensordaten 128

5.26 Graph zum Anzeigen der originalen und der veränderten Sensordaten . . 129
5.27 Weißes, pinkes und rotes Rauschen 130
5.28 Funktionsblöcke zum Generieren von Bias und Skalierungsfehler 131
5.29 Funktionsblock zum Generieren von Nichtlinearität in den Sensordaten . 132
5.30 Generieren des Package-Ausrichtungsfehlers 132
5.31 Generierte Fehler auf x-Achse für Bias und Temperaturdrift 133
5.32 Verarbeitungsschritte für die Überlagerung von Sensordaten 134
5.33 Daten und Ergebnisse bei der Augmentation von Sensordaten 135
5.34 Verarbeitungsschritte für die Überlagerung von Sensordaten 136
5.35 Verarbeitungsschritte für die Überlagerung von Sensordaten 137
5.36 Elemente für Turing-Vollständigkeit des Frameworks 137
5.37 Beispiel einer minimalen Turing-Maschine 138

Tabellenverzeichnis

4.1 Gemessene Fehler bei den untersuchten Smartphones 61
4.2 Verfügbarkeit von Inertialsensoren und virtuellen Sensoren in Smartphones 63
4.3 Beispielhafte Ergebnisse des Magnetometer-Tests 72
4.4 Bewertungen der Handhabbarkeit der Zeigegeräte 90

5.1 Größe der Binärdatei in Byte im ROM ohne Nutzung des Magnetometers 101
5.2 Größe der Binärdatei in Byte im ROM mit Nutzung des Magnetometers . 102
5.3 Benötigter Arbeitsspeicher in Byte ohne Verwendung des Magnetometers 103
5.4 Benötigter Arbeitsspeicher in Byte mit Verwendung des Magnetometers . 103
5.5 Mittlere Ausführungszeit der Fusionsalgorithmen ohne Magnetometer . . 105
5.6 Mittlere Ausführungszeit der Fusionsalgorithmen mit Magnetometer . . . 106
5.7 Mittlerer Winkelfehler bei 15 Messungen mit Korrektur der z-Achse . . . 110
5.8 Mittlerer Winkelfehler bei 15 Messungen mit Magnetometer 113

D.1 Daten für die Gage R&R-Analyse ohne Nutzerführung XXIV
D.2 Daten für die Gage R&R-Analyse mit Nutzerführung XXIV
D.3 Daten über die *High-Fidelity*-Sensoren in Android-Smartphones XXV
D.4 Bewertung der Sensorsubsysteme nach Sensorhersteller mit 50 Hz XXV
D.5 Bewertung der Sensorsubsysteme nach Sensorhersteller mit 200 Hz XXV
D.6 Bewertung der Sensorsubsysteme nach Sensorfusion mit 50 Hz XXVI
D.7 Bewertung der Sensorsubsysteme nach Sensorfusion mit 200 Hz XXVI
D.8 Bewertung der Sensorsubsysteme nach Smartphone-Hersteller mit 50 Hz . XXVI
D.9 Bewertung der Sensorsubsysteme nach Smartphone-Hersteller mit 200 Hz XXVII
D.10 Ergebnisse der ersten Messreihe für den Magnetometer-Test XXVIII
D.11 Ergebnisse der zweiten Messreihe für den Magnetometer-Test XXVIII
D.12 Testergebnisse für Genauigkeit & Präzision von digitalen Zeigegeräten . . XXIX
D.13 Größe der Binärdatei im ROM ohne Magnetometer und -O0 Optimierung XXX
D.14 Größe der Binärdatei im ROM mit Magnetometer und -O0 Optimierung XXX
D.15 Benötigter Arbeitsspeicher ohne Magnetometer mit -O3 Optimierung . . XXXI
D.16 Benötigter Arbeitsspeicher mit Magnetometer mit -O3 Optimierung . . . XXXI
D.17 Mittlere Ausführungszeit ohne Magnetometer mit -O0 Optimierung . . . XXXII
D.18 Mittlere Ausführungszeit mit Magnetometer mit -O0 Optimierung XXXII
D.19 Messergebnisse Filteralgorithmen ohne Magnetometer XXXIII
D.20 Messergebnisse Filteralgorithmen mit Magnetometer XXXIV

Publikationen

Korrespondierender Autor

[1] N. Büscher, D. Gis, J. P. Wolff und C. Haubelt, „Data Augmentation Framework for Smart Sensor System Development Using the Sensor-in-the-Loop Prototyping Platform,“ in *International Workshop of Rapid System Prototyping (RSP)*, Paris, France: IEEE, 2021. DOI: 10.1109/RSP53691.2021.9806209,

Abstract: Sensor sub-systems are becoming more complex as they increasingly take on various tasks, from signal processing to pattern recognition. This off-loading of continuous processes can decrease power consumption of the entire system, as the application processor of e.g. a smartwatch can spend more time in low-power modes. The design and validation of these sensor subsystems are thus becoming more difficult and time consuming. This development process relies heavily on the availability of suitable sensor signals for testing. We propose a modular component-based framework to generate a multitude of tests using either prerecorded sensor signals or artificial signals to allow for a both faster and more thorough development process and prototyping. Using the recently proposed Sensor-in-the-Loop architecture, the work at hand demonstrates not only the ability to test and develop the software offline in a simulation, but to also use the augmented sensor signal data directly on the hardware prototype at run-time. The framework can be used in various testing- and development setups to simulate sensor characteristics, processing steps and possible errors. We show, based on three comprehensive examples, that the proposed framework is able to simulate the common errors found in inertial MEMS sensors, generate signal traces to develop, train, and evaluate gesture recognition algorithms, and simulate pre-processing steps in order to evaluate their feasibility before they are implemented in the sensor firmware.

Beitrag: Für diese Arbeit wurde die Architektur des Frameworks und der Funktionsblöcke entworfen und prototypisch implementiert, sowie die grafische Benutzeroberfläche entworfen. Zusammen mit den Mitautoren wurde der Funktionsumfang erarbeitet.

Verwendung: Diese Arbeit wird in Abschnitt 5.3 vorgestellt.

[2] N. Büscher, D. Gis, V. Kühn und C. Haubelt, „On the Functional and Extra-Functional Properties of IMU Fusion Algorithms for Body-Worn Smart Sensors,“ *MDPI Sensors*, Nr. 8, 2021. DOI: 10.3390/s21082747,

Abstract: In this work, four sensor fusion algorithms for inertial measurement unit data to determine the orientation of a device are assessed regarding their usability in a hardware restricted environment such as body-worn sensor nodes. The assessment is done for both the functional and the extra-functional properties in the context of human operated devices. The four algorithms are implemented in three data formats: 32-bit floating-point, 32-bit fixed-point and 16-bit fixed-point and compared regarding code size, computational effort, and fusion quality. Code size and computational effort are evaluated on an ARM Cortex M0+. For the assessment of the functional properties, the sensor fusion output is compared to a camera generated reference and analyzed in an extensive statistical analysis to determine how data format, algorithm, and human interaction influence the quality of the sensor fusion. Our experiments show that using fixed-point arithmetic can significantly decrease the computational complexity while still maintaining a high fusion quality and all four algorithms are applicable for applications with human interaction.

Beitrag: Bei dieser Arbeit wurde die Testdurchführung entworfen, die Sensorfusionsalgorithmen in unterschiedlichen Datenformaten implementiert und die Tests für die funktionalen Eigenschaften durchgeführt und auswertet.

Verwendung: Diese Arbeit wird in Abschnitt 5.2 vorgestellt.

[3] N. Büscher, D. Gis, S. Stieber und C. Haubelt, „Multi-Aspect Evaluation Method for Digital Pointing Devices," in *Proceedings of Pervasive and Embedded Computing and Communication Systems*, Bd. 8, Wien, Österreich: PECCS, Sep. 2019, S. 128–135. DOI: 10.5220/0008355701280135,

Abstract: For decades the computer mouse has been used as the most common input device for laptops and computersalike. However for speeches a presentation remote with a laser pointer was used because they allowed the pre-senter more freedom. With the emergence of small and lightweight inertial sensors, a new type of presentationremotes becomes popular. These remotes use inertial sensors to move a digital pointer allowing presenters toshow things on more than one screen or use enhancement methods like highlighting a region. Using inertialsensors however proves to be a difficult task and can lead to problems with the usability of such devices.When developing such systems, the designer faces the problem that no method for quantifying the usability ofpointing devices based on inertial sensors is available. In the paper at hand, we propose an evaluation methodconsisting of three different tests to assess the manageability, speed and precision of digital pointing devicesfor a measurable comparison. Additionally, we conducted an evaluation to show that our tests reflect the sub-jective assessment from the users. Our quantitative test results showed a strong correlation to the qualitativesubjective assessment from the users.

Beitrag: Für diese Veröffentlichung wurden die Testabläufe entworfen und die Testprogramme umgesetzt. Die Tests und Befragungen der Testpersonen wurden zu gleichen Teilen von allen Autoren durchgeführt.

Verwendung: Diese Arbeit wird in Abschnitt 4.4 vorgestellt.

[4] N. Büscher, M. Stieringer und C. Haubelt, „Evaluation Method for the absolute Orientation from the Rotation Vector on Mobile Devices," in *Proceedings of the*

IEEE International Symposium on Inertial Sensors and Systems (INERTIAL), Naples, Florida, USA: IEEE, Apr. 2019, S. 179–183. DOI: 10.1109/ISISS.2019.8739713,

Abstract: Smartphone applications for indoor navigation or augmented reality play an increasingly important role on mobile devices. However, the accuracy of the inertial sensors on mobile devices is a limiting factor. In this paper, we propose a novel test method to assess the accuracy of the absolute orientation from the Rotation Vector using the time proven bearing method from marine navigation. The test allows every smartphone user to test the Rotation Vector of their device without the need for additional equipment. Additionally, our work considers magnetic effects and user errors and introduces means to mitigate the influences to ensure a reliable test result.

Beitrag: Es wurde die grundlegende Idee für die Bewertung der absoluten Orientierung erarbeiten. Zudem wurden die Konzepte entworfen, wie die Methode evaluiert werden kann. Weiterhin wurden die Ergebnisse verifiziert.

Verwendung: Diese Arbeit wird in Abschnitt 4.2 vorgestellt.

[5] N. Büscher, L. Middendorf, C. Haubelt, R. Dorsch und F. Wegelin, „Statistical analysis and improvement of the repeatability and reproducibility of an evaluation method for IMUs on a smartphone," in *Proceedings of the Symposium on Engineering Interactive Computing Systems (EICS'16)*, Brüssel, Belgien: ACM, Juni 2016, S. 149–158, ISBN: 978-1-4503-4322-0. DOI: 10.1145/2933242.2933255,

Abstract: An increasing number of mobile applications, like indoornavigation or augmented reality, rely on highly accurate inertial sensor systems. However, there exists no standardized test for common inertial sensors, like accelerometers and gyroscopes to assess their accuracy. To easily determine the sensor system quality, a mobile sensor test application for smartphones was published recently. Although this test has several advantages, it lacks reproducibility and repeatability. As a remedy, we designed a special user interface that guides the operator through the test and helps judging the quality of the test conductance. While poorly conducted tests are neglected, only the meaningful ones are evaluated, so that our technology enables a more detailed analysis of the sensor systems. In this paper, we propose a statistical approach capable to distinguish between the four most important systematic errors we observed. Experiments with five state-of-the-art smartphones show that reproducibility and repeatability were significantly improved, enabling a fair comparison of the smartphone sensor systems.

Beitrag: Die Nutzerführung wurde entworfen und implementiert. Die statistische Auswertung der Messergebnisse mittels Gage R&R wurde konzipiert, durchgeführt und ausgewertet.

Verwendung: Diese Arbeit wird in Abschnitt 4.1 vorgestellt.

Mitautor

[6] D. Gis, N. Büscher und C. Haubelt, „Real-Time Power Analysis of Smart Sensors Using Advanced Debugging Methods," *MDPI Micromachines*, Nr. 12, 2021. DOI: 10.3390/mi12111276,

Abstract: To get a good estimate of the power consumption of an embedded system, including its firmware is a crucial step in the development of systems with a severely constrained power supply. This is especially true for cases where the device is powered by a small battery or through energy harvesting. The state-of-the-art approaches to measure or estimate the power consumption are formal methods, using power debugging tools with the real hardware or simulation based estimations. In the work at hand, a novel method to estimate the power consumption is proposed, it utilizes the Sensor-in-the-Loop architecture and enhancing it with a power estimation functionality. The proposed method combines the benefits of former methods, allowing for run-time analysis of the power-consumption in a reproducible way using recorded data without the need for power debugging hardware. In the experiments it is shown that, once set up, the proposed method is able to estimate the power consumption with an error of less than 1 % compared to a power debugging hardware. Thus, the proposed method provides a reliable and fast way to estimate the systems power consumption.

Beitrag: Für diese Veröffentlichung erfolgte eine Mitwirkung bei Aufbau, Auswertung und Durchführung der Messungen. Zudem wurden Teile der Veröffentlichung verfasst

Verwendung: Diese Arbeit ist Teil der SiL Architektur, die in Abschnitt 5.2 verwendet wird.

[7] D. Gis, N. Büscher und C. Haubelt, „Investigation of Timing Behavior and Jitter in a Smart Inertial Sensor Debugging Architecture," *MDPI Sensors*, Nr. 14, 2021. DOI: 10.3390/s21144675,

Abstract: Due to upcoming higher integration levels of microprocessors, the market of inertial sensors has changed in the last few years. Smart inertial sensors are becoming more and more important. This type of sensor offers the benefit of implementing sensor-processing tasks directly on the sensor hardware. The software development on such sensors is quite challenging. In this article, we propose an approach for using prerecorded sensor data during the development process to test and evaluate the functionality and timing of the sensor firmware in a repeatable and reproducible way on the actual hardware. Our proposed Sensor-in-the-Loop architecture enables the developer to inject sensor data during the debugging process directly into the sensor hardware in real time. As the timing becomes more critical in future smart sensor applications, we investigate the timing behavior of our approach with respect to timing and jitter. The implemented approach can inject data of three 3-DOF sensors at 1.6 kHz. Furthermore, the jitter shown in our proposed sampling method is at least three times lower than using real sensor data. To prove the statistical significance of our experiments, we use a Gage R&R analysis, extended by the assessment of confidence intervals of our data

Beitrag: Bei dieser Veröffentlichung wurde bei Konzeption und Durchführung der Gage R&R Analyse unterstützt sowie Teile der Veröffentlichung verfasst.
Verwendung: Diese Arbeit ist Teil der SiL Architektur, die in Abschnitt 5.2 verwendet wird.

[8] D. Gis, N. Büscher und C. Haubelt, „Advanced Debugging Architecture for Smart Inertial Sensors using Sensor-in-the-Loop," in *International Workshop of Rapid System Prototyping (RSP)*, IEEE, Sep. 2020, S. 1–7. DOI: 10.1109/RSP51120.2020.9244851,

Abstract: Smart inertial sensors have emerged in the last years enabling developers to implement sensor fusion or tasks like gesture detection directly in the sensor hardware and thus reducing the processing latency and energy consumption. The development of software for smart inertial sensors however faces difficulties from the limited hardware capabilities of the µC hardware and the lack of options to conduct reproducible tests directly on the hardware. We propose a Sensor-in-the-Loop architecture that allows a developer to record data from a smart sensor and inject the recorded data back into the sensor at a later time to evaluate the performance of the software in a reproducible way. With the proposed architecture it is possible to examine the performance and computational load on real hardware with recorded data thus allowing developers to optimize and improve the software in a more targeted way. In our implementation, the memory overhead for the code instrumentalization is just 0.63 %. The used RTT interface is at least four times faster than the regular SPI sensor interface. In experiments, we show that our proposed architecture is able to record and inject data of up to three 3-DOF sensors with 1.6 kHz sampling frequency in real time.

Beitrag: Es wurde eine GUI zur Betrachtung und Injektion der Sensorwerte prototypisch implementiert und gemeinsam mit dem korrespondierenden Autor Tests der SiL-Architektur entworfen und durchgeführt. Des Weiteren wurden Teile der Veröffentlichung mitverfasst.
Verwendung: Die SiL Architektur wird als Werkzeug in Abschnitt 5.2 verwendet.

[9] R. Dorsch, C. Haubelt, S. Stieber, D. Gis und N. Büscher, „Verfahren zum Verarbeiten der Sensorsignale einer Drehratensensoranordnung, Vorrichtung und Verfahren zum Betreiben der Vorrichtung," DE102020205943B3, Mai 2021,

Abstract: Die Erfindung betrifft ein Verfahren zum Verarbeiten der Sensorsignale einer Drehratensensoranordnung mit mindestens drei Drehratensensorkomponenten, die Sensorsignale g_x, g_y, g_z für drei linear unabhängige Raumrichtungen in einem Sensorbezugssystem liefern, bei dem ein Ausgangssignal mit zwei Signalkomponenten g'_y, g'_z für zwei linear unabhängige Raumrichtungen eines absoluten Bezugssystems erzeugt wird, indem zumindest die Sensorsignale g_y, g_z für zwei der drei linear unabhängigen Raumrichtungen des Sensorbezugssystems in das absolute Bezugssystem transformiert werden, wodurch eine Verdrehung ϕ der Drehratensensoranordnung um die dritte linear unabhängige Raumrichtung des Sensorbezugssystems zumindest teilweise kompensiert wird. Der Kompensation der Verdrehung ϕ der Drehratensensoranordnung um die dritte linear unabhängige Raumrichtung des Sensorbe-

zugssystems wird bei der Transformation der zumindest zwei Sensorsignale g_y, g_z zumindest das dritte Sensorsignal g_x für die dritte linear unabhängige Raumrichtung des Sensorbezugssystems zugrunde gelegt.

Beitrag: Für diese Veröffentlichung wurden die Algorithmen für die Kompensation der Verdrehung entworfen, implementiert und getestet sowie an Teilen der Implementierung der Firmware mitgewirkt.

Verwendung: Das Patent ging aus Arbeiten an einem digitalen Laserpointer hervor, bei welchen auch die Arbeit von Abschnitt 4.4 entstand.

[10] D. Gis, C. Haubelt und N. Büscher, „Verfahren zum Verarbeiten der Sensorsignale einer Drehratensensoranordnung, Vorrichtung und Verfahren zum Betreiben der Vorrichtung," DE102021200244A1, März 2022,

Abstract: Es wird ein Verfahren zur Überprüfung, Evaluation und/oder Fehlerdiagnose eines Sensorsystems, wobei das Sensorsystem mindestens eine Sensoreinheit und eine interne Datenverarbeitungseinheit aufweist, wobei die Sensoreinheit in Abhängigkeit eines physikalischen Stimulus Sensordaten ausgibt und die interne Datenverarbeitungseinheit dazu konfiguriert ist, die von der Sensoreinheit ausgegebenen Sensordaten zu Ausgangsdaten zu verarbeiten, wobei die von der Sensoreinheit ausgegebenen Sensordaten in einem Aufzeichnungsschritt durch eine externe Rechnereinrichtung aus dem Sensorsystem ausgelesen und aufgezeichnet werden, die aufgezeichneten Sensordaten in einem Injektionsschritt durch die externe Rechnereinrichtung in die interne Datenverarbeitungseinheit eingespeist werden, und die eingespeisten Sensordaten in einem Verarbeitungsschritt von der internen Datenverarbeitungseinheit verarbeitet werden. Weiterhin wird ein Sensorsystem und eine System aus einem Sensorsystem und einer Rechnereinrichtung vorgeschlagen.

Beitrag: Es wurden Teile der Software als Proof-of-Concept implementiert und gemeinsam mit dem korrespondierenden Autor Tests der SiL-Architektur entworfen und durchgeführt.

Verwendung: Das Patent betrifft die SiL-Architektur, die in Abschnitt Abschnitt 5.2 verwendet wird.

[11] D. Gis, C. Haubelt und N. Büscher, „Method for Checking, Evaluation, and/or Error Diagnosis of a Sensor System, Sensor System, and System," US000011589-139B2, Feb. 2023,

Abstract: A method for checking, evaluation, and/or error diagnosis of a sensor system. The sensor system includes at least one sensor unit and an internal data processing unit. The sensor unit outputs sensor data as a function of a physical stimulus and the internal data processing unit is configured to process the sensor data output by the sensor unit to form output data. The sensor data output by the sensor unit is read out from the sensor system and recorded by an external processor unit in a recording step. The recorded sensor data are fed by the external processor unit into the internal data processing unit in an injection step. The fed-in sensor data are processed by the internal data processing unit in a processing step. A sensor system, and a system made up of a sensor system and a processor unit are also described.

Beitrag: Es wurden Teile der Software als Proof-of-Concept implementiert und gemeinsam mit dem korrespondierenden Autor Tests der SiL-Architektur entworfen und durchgeführt.
Verwendung: Das Patent betrifft die SiL-Architektur, die in Abschnitt Abschnitt 5.2 verwendet wird.

[12] D. Gis, C. Haubelt und N. Büscher, „Method for Testing, Evaluation, and/or Error Diagnosis of a Sensor System, Sensor System, and System," CN0001142538-31A, 2022,

Abstract: A method for testing, evaluation, and/or error diagnosis of a sensor system. The sensor system includes at least one sensor unit and an internal data processing unit. The sensor unit outputs sensor data as a function of a physical stimulus and the internal data processing unit is configured to process the sensor data output by the sensor unit to form output data. The sensor data output by the sensor unit is read out from the sensor system and recorded by an external processor unit in a recording step. The recorded sensor data are fed by the external processor unit into the internal data processing unit in an injection step. The fed-in sensor data are processed by the internal data processing unit in a processing step. A sensor system, and a system made up of a sensor system and a processor unit are also described.

Beitrag: Es wurden Teile der Software als Proof-of-Concept implementiert und gemeinsam mit dem korrespondierenden Autor Tests der SiL-Architektur entworfen und durchgeführt.
Verwendung: Das Patent betrifft die SiL-Architektur, die in Abschnitt Abschnitt 5.2 verwendet wird.

1 Einleitung

Inertialsensoren ermöglichen es, die Beschleunigung, Drehgeschwindigkeit und/oder Orientierung des Sensors im Raum zu bestimmen [A1]. Am Körper getragen erlauben sie Rückschlüsse auf die Bewegungen, die eine Person durchführt [A2]. Sie spielen somit zunehmend eine wichtige Rolle bei der Umsetzung neuer technischer Anwendungen, insbesondere im Bereich der Mensch-Maschine-Interaktion (MMI). Dazu zählen Methoden wie z.B. Gestenerkennung für die Steuerung eines Gerätes, virtuelle Zeigegeräte, Headsets für Virtuelle Realität (VR) oder Indoor-Navigation. Auch in der (Sport-)Medizin werden körpernahe Inertialsensoren verwendet, um die Bewegungen einer Person aufzunehmen und zu analysieren [A3]. Das vermutlich am häufigsten anzutreffende System mit Inertialsensoren ist heutzutage das Smartphone. Durch immer leistungsfähigere Hardware ist auf diesem eine fast unbegrenzte Anzahl unterschiedlicher Anwendungen möglich. Auch in kleineren eingebetteten Systemen, wie etwa Smartpens oder Wearables werden immer häufiger die Daten der Inertialsensoren direkt verarbeitet. Alle diese Anwendungen setzen effektive Inertialsensoren und eingebettete Systeme voraus, um die Daten aufzunehmen, zu verarbeiten und daraus höherwertige Daten zu generieren.

Damit die Anwendungen verlässlich funktionieren, müssen die Messdaten der Inertialsensoren und die Ergebnisse der Sensordatenfusion genau, robust und zuverlässig sein [A1]. Besonders bei Smartphones haben Untersuchungen aber gezeigt, dass die Qualität der Sensordaten und der Sensordatenfusion sehr heterogen ist [A4]. Hierbei spielen die Qualität der verbauten Sensorhardware und die verwendeten Sensorfusionsalgorithmen eine Rolle. Beide Faktoren haben aber auch bei der Verarbeitung der Daten auf anderen eingebetteten Systemen einen maßgeblichen Einfluss auf das Endergebnis. Hinzu kommt bei menschlicher Interaktion, dass die Bewegungen für dieselbe Geste sehr unterschiedlich ausfallen können. Bei der Entwicklung von Anwendungen für Smartphones und eingebettete Systeme stellen die Unterschiede bei der Qualität eine große Herausforderung dar [A5]. Anwendungen für Smartphones sollten optimalerweise so entwickelt werden, dass sie mit einer möglichst großen Bandbreite an Genauigkeiten der (virtuellen) Sensoren funktionieren. Andernfalls wird die Smartphone-Anwendung auf vielen Geräten nicht zufriedenstellend funktionieren. Bei kleineren eingebetteten Systemen muss während der Entwicklung darauf geachtet werden, einen guten Mittelweg zwischen ressourcensparenden Verfahren und der Qualität der Ergebnisse zu finden. Gerade bei menschlicher Interaktion ist es unerlässlich, dass die Anwendungen je nach Anwendungsgebiet sehr zuverlässig funktionieren, um von einer großen Nutzerbasis akzeptiert zu werden.

1.1 Motivation und Zielsetzung

Die genannten Herausforderungen bei der Entwicklung von Anwendungen für menschliche Interaktion mittels Bewegungsdaten führen zu einer zeit- und kostenintensiven Entwicklung. Gerade die Verfügbarkeit von Daten, Messmethoden und Informationen über die Qualität unterschiedlicher Sensorsubsysteme ist hier häufig ein Problem. Für Anwendungen, die auf möglichst vielen Geräten laufen sollen, bedarf es eines fundierten Wissens über die Bandbreite der verschiedenen Sensorsubsysteme, um abschätzen zu können, wie die Anwendungen mit den Qualitätsunterschieden zurechtkommen. Dies erfordert eine erhebliche Anzahl an Tests der entwickelten Anwendung auf unterschiedlichsten Systemen. Aber auch für das Erkennen und die Interpretation von menschlichen Bewegungen werden Daten von vielen unterschiedlichen Menschen benötigt, die eine Geste oder Bewegung sehr häufig wiederholen [A6]. Um in möglichst vielen Situationen zuverlässig zu funktionieren, müssen zudem Messungen von vielen verschiedenen Szenarien aufgenommen werden. Bei ressourcenbeschränkten eingebetteten Systemen müssen potenziell viele Setups untersucht und verglichen werden, um das beste Ergebnis zu erhalten.

Wie man sich leicht vorstellen kann, ist es oftmals jedoch sehr schwer oder nicht möglich, alle für eine Anwendung notwendigen Informationen und Daten zu erhalten. Dies kann dazu führen, dass eine Anwendung letztendlich fehlerhaft oder ungenau ist und nur unzureichend funktioniert. Ein Beispiel hierfür ist die Indoor-Lokalisation mittels der Inertialsensoren von Smartphones. Untersuchungen haben hier gezeigt, dass es noch gravierende Probleme bei der genauen Bestimmung der Position gibt [A7], [A8]. Dies kann sehr negative Auswirkungen auf die Funktionsfähigkeit der Anwendung oder den Ruf eines Entwicklers haben.

Die Motivation für die in dieser Dissertationsschrift beschriebenen Arbeiten ist es, Entwickler:Innen eine Hilfestellung bei der Erstellung von Anwendungen und Sensorsubsystemen zu geben. Dafür werden Konzepte und Methoden vorgestellt, mit denen es auf einfache Weise und ohne teure externe Werkzeuge möglich ist, die Qualität der Sensordatenfusion oder des Inertialsensorsubsystems zu beurteilen. Ferner wurden die beschriebenen Methoden genutzt, um Untersuchungen sowohl auf der Anwendungsebene auf dem Smartphone, als auch auf Firmware-Ebene in eingebetteten Systemen durchzuführen. Die daraus gewonnenen Erkenntnisse sollen bei der Entwicklung als Informationsquelle und Entscheidungshilfe dienen. Beispielweise, um abschätzen zu können, mit welchen Fehlern man bei Smartphone-Anwendungen rechnen muss. Des Weiteren wird eine Anwendung vorgestellt, mit der es auf schnelle und unkomplizierte Weise möglich ist, die erforderlichen Sensordaten zu generieren. Sei es, um die Toleranz einer Anwendung für die Fehler eines Sensorsubsystems zu testen oder um Algorithmen für die Gestenerkennung zu trainieren und zu verifizieren. Das Ziel ist es, die Entwicklung solcher Anwendungen zu unterstützen und somit letztendlich zu zuverlässigeren und besseren Anwendungen beizutragen.

1.2 Publikationen

Die wissenschaftlichen Publikationen, die in dieser Dissertationsschrift beschrieben werden, befassen sich mit Konzepten, Methoden und Anwendungen für die Untersuchung der Qualität von Inertialsensoren und Sensorfusionsalgorithmen. Weiterhin werden auch Ergebnisse aus Untersuchungen vorgestellt, die diese Methoden und Konzepte anwenden. Der Fokus liegt hierbei auf Anwendungsbereichen in der Unterhaltungselektronik und speziell bei der Interaktion mit Menschen. Die oben genannten Themen werden hierbei auf zwei Ebenen behandelt. Die zeitliche und thematische Einordnung der Publikationen ist in Abbildung 1.1 dargestellt.

Auf der ersten Ebene werden Konzepte und Methoden vorgestellt, die sich mit der Evaluation der Qualität der Sensordaten sowie der Sensordatenfusion auf Anwendungsebene beschäftigen. Hierbei handelt es sich um Anwendungen, die Sensordaten aus einem schon bestehenden System erhalten, wie etwa von einem Smartphone oder einem Smartpen. Die Daten wurden dabei bereits vom System vorverarbeitet oder es wurden durch Sensorfusionsalgorithmen bereits (Zwischen-)Ergebnisse berechnet.

Auf der zweiten Ebene werden Konzepte, Methoden und Untersuchungen auf der Firmware-Ebene betrachtet. Auf dieser erhalten die entwickelten Anwendungen die Messdaten direkt von den Inertialsensoren und müssen Filterung, Verarbeitung und Sensordatenfusion selbst durchführen. Entscheidend bei diesen Systemen ist, dass sie oftmals mit wenig Speicher und Rechenkapazität auskommen müssen.

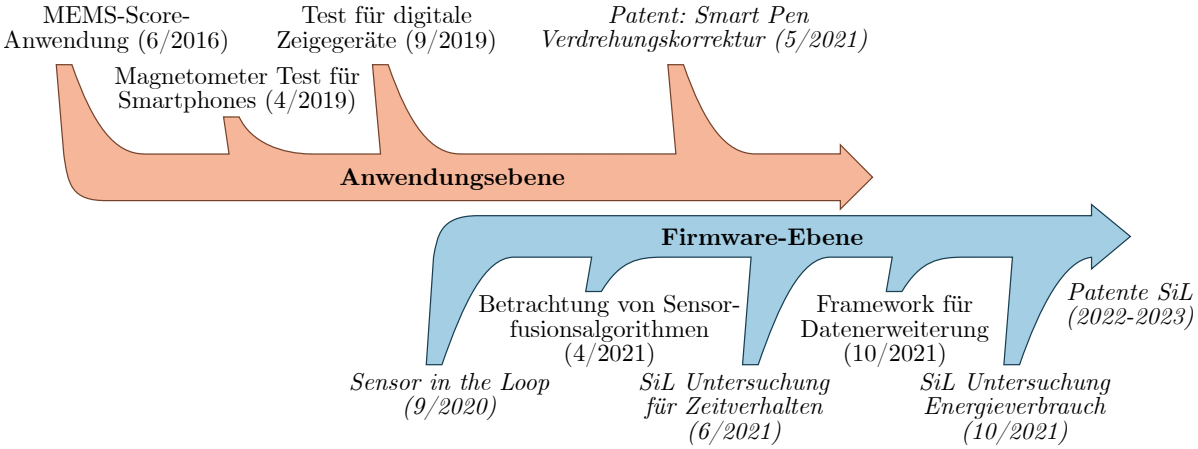


Abbildung 1.1: Eigene wissenschaftliche Publikationen, getrennt nach Anwendungsebene und Firmware-Ebene. Publikationen in kursiv sind als Mitautor

Die erste Publikation auf der Anwendungsebene befasst sich mit einer Erweiterung einer Methode zur Bewertung der Genauigkeit des virtuellen Sensors für die Orientierung (Game-Rotation Vector) eines Android Smartphones, die in [A4] vorgestellt wurde. Diese

Methode wurde um eine Nutzerführung erweitert, die gewährleisten soll, dass die Messungen wiederholbar und reproduzierbar sind. Weiter wurden Methoden hinzugefügt, die es erlauben, die Fehlerursachen innerhalb der Messungen zu ermitteln. Die Methodik an sich wurde abgeändert, um die Zuverlässigkeit der verwendeten Bildverarbeitung zu erhöhen. Die Ergebnisse dieser Arbeit wurden 2016 auf der *Engineering Interactive Computing Systems (EICS)* Konferenz vorgestellt [B1]. Eine Weiterentwicklung dieser Arbeit wurde 2019 auf der IEEE Sensors Konferenz vorgestellt [B2]. Diese Weiterentwicklung befasst sich mit einer Methode zur Erfassung und Bewertung der absoluten Orientierung mit korrekter Ausrichtung am Erdmagnetfeld, welche in der vorherigen Arbeit nicht möglich war. Das Hauptaugenmerk war hierbei die Methode zu Bestimmung der Vergleichsreferenz für die Bewertung.

Eine weitere Methode für die Bewertung von Sensorsubsystemen auf Anwendungsebene, ist die 2019 auf der *Pervasive and Embedded Computing and Communication Systems (PECCS)* Konferenz vorgestellte Methodik zur Bewertung von digitalen Zeigegeräten, insbesondere Smartpens [B3]. Wichtig bei dieser Arbeit war, eine Metrik für die Bewertung zu finden, bei der die subjektive Nutzungserfahrung mit der quantitativen Bewertung korreliert. Im Rahmen von Arbeiten in dem Bereich der Smartpens ist ein bereits erteiltes Patent für die energieeffiziente Verdrehungs-Korrektur entstanden [B4].

Auf der Firmware-Ebene wurde an der Entwicklung der Sensor-in-the-Loop-Architektur mitgewirkt, welche es ermöglicht, vorher aufgenommene Sensordaten wieder in den Inertialsensor einzuspeisen, um eine zuverlässige und reproduzierbare Methode bereitzustellen, Firmware auf Sensor zu entwickeln und zu testen. Diese Arbeit wurde von Herrn Gis 2020 auf dem *Rapid System Prototyping (RSP)* Workshop vorgestellt [B5]. Eine um eine Analyse des Zeitverhaltens erweiterte Version dieser Arbeit wurde 2021 im *MDPI Sensor Journal* veröffentlicht [B6]. Mitgewirkt wurde hierbei im Bereich der statistischen Signifikanz-Analyse der Messdaten. Weiterhin wurde an der Erweiterung der Sensor-in-the-Loop-Architektur für die Messung des Energieverbrauchs mitgewirkt. Diese Arbeit wurde 2021 im *MDPI Micromachines Journal* veröffentlicht [B7].

Auf der Firmware-Ebene wurde weiterhin eine Untersuchung über die funktionalen und extrafunktionalen Eigenschaften unterschiedlicher Sensorfusionsalgorithmen und Datentypen durchgeführt. Hierbei wurden statistische Bewertungsmethoden verwendet, um den Einfluss der unterschiedlichen Faktoren zu ermitteln. Die Ergebnisse dieser Untersuchung wurden 2021 im *MDPI Sensor Journal* veröffentlicht [B8]. Als Referenz diente hier die bereits aus [A4] bekannte und abgewandelte Methodik, um Referenzdaten aus Kamerabildern zu extrahieren. Weiterhin wurde auf der Firmware-Ebene ein Framework auf Basis von Sensor-in-the-Loop entwickelt, welches es ermöglicht, Daten von Inertialsensoren zu Laufzeit zu manipulieren oder zu augmentieren. Dies soll es ermöglichen, die Entwicklung und Evaluation von Sensor-Formware zu beschleunigen und robuster zu gestalten. Das Framework wurde 2021 auf dem *Rapid System Prototyping (RSP)* Workshop im Rahmen der *Embedded Systems Week (ESWEEK)* vorgestellt.

1.3 Aufbau der Dissertationsschrift

Der Aufbau dieser Dissertationsschrift gliedert sich wie folgt:

In Kapitel 2 wird erläutert, wie der aktuelle Stand der Technik ist und wie diese Dissertationsschrift in diesem Kontext einzuordnen ist. Hierbei wird auf vergleichbare Arbeiten eingegangen und wo die Unterschiede zu dieser Dissertationsschrift zu finden sind. Weiterhin wird ausgeführt, worin der wissenschaftliche Mehrwert der durchgeführten Arbeiten liegt.

In Kapitel 3 werden die Grundlagen beschrieben, die für das generelle Verständnis dieser Dissertationsschrift notwendig sind und für mehrere der beschriebenen Forschungsarbeiten relevant sind. Grundlagen, die nur für einen abgeschlossenen Teil dieser Dissertationsschrift relevant sind, werden am Anfang der jeweiligen Kapitel erläutert.

Der erste Schwerpunkt dieser Dissertationsschrift über Methoden und Untersuchungen von Sensorsubsystemen auf der Anwendungsebene wird in Kapitel 4 beschrieben. Hierbei wird zuerst die MEMS-Score-Anwendung für die Bewertung des Sensorsubsystems auf Smartphones sowie deren Erweiterungen beschrieben. Anschließend folgt die Beschreibung eines Bewertungsverfahrens für digitale Zeigegeräte und im Speziellen für Smartpens.

Um die Bewertung sowie Methoden für die Bewertung von Inertialsensoren sowie Sensorfusionsalgorithmen auf der Firmware-Ebene geht es im zweiten Schwerpunkt dieser Dissertationsschrift in Kapitel 5. Hierbei wird kurz auf die Sensor-in-the-Loop-Architektur als Grundlage eingegangen. Anschließend folgt eine Untersuchung der funktionalen und nichtfunktionalen Eigenschaften mehrerer Sensorfusionsalgorithmen und Datenformate. Schließlich wird ein Framework vorgestellt, das es ermöglicht, aufgenommene Sensordaten zur Laufzeit zu manipulieren oder zu augmentieren, um die Entwicklung und Evaluation von Sensor-Firmware zu unterstützen.

Abschließend werden in der Zusammenfassung in Kapitel 6 die Ergebnisse der in dieser Dissertationsschrift behandelten Arbeiten resümiert. Hierbei wird nochmal eine abschließende Bewertung vorgenommen, sowie auf Unzulänglichkeiten und weiterhin bestehende Probleme eingegangen. Zuletzt wird noch ein Ausblick gegeben, wie die Arbeiten weitergeführt werden könnten.

1.4 Hinweise

Die meisten der in dieser Dissertationsschrift beschriebenen Methoden und Dissertationsschrift wurden zuvor vom Autor auf begutachteten Konferenzen, Workshops und/oder Fachzeitschriften veröffentlicht. Diese Dissertationsschrift fasst die Ergebnisse der jeweiligen Arbeiten zusammen und stellt sie in den Kontext einer ganzheitlichen Betrachtung über die Bewertung und Verbesserung von eingebetteten Inertialsensorsubsystemen. Zu diesem Zweck wird die Formulierung vereinheitlicht und die Arbeiten durch umfangreichere Experimente und Auswertungen erweitert.

In dieser Dissertationsschrift werden drei unterschiedliche Arten an Quellen zitiert. Der Zitationsstil richtet sich hierbei nach dem IEEE Referenzstil und ist jeweils entsprechend nach Art der Quelle gekennzeichnet sind. Referenzen mit der Kennzeichnung [**A *Nummer***] beziehen sich auf Literatur auf welche in dieser Dissertationsschrift Bezug genommen wird, oder aus der Informationen stammen. Mit [**B *Nummer***] gekennzeichnete Referenzen verweisen auf Veröffentlichungen des Autors dieser Dissertationsschrift. Bei Referenzen nach dem Muster [**C *Nummer***] handelt es sich um Verweise auf studentische Abschlussarbeiten, die vom Autor betreut wurden.

Bei der Verwendung von Abkürzungen werden diese bei der ersten Verwendung ausgeschrieben. Die dazugehörige Abkürzung wird dahinter in Klammern angegeben. Stammt die genutzte Abkürzung aus dem Englischen, so ist auch die volle englische Bezeichnung in den Klammern zu finden.

Die in dieser Dissertationsschrift genutzten Grafiken wurden ausnahmslos vom Autor dieser Dissertationsschrift selbst erstellt. Bei Grafiken, die aus anderen Quellen abgeleitet sind, wird dies durch die Nennung der Quelle in der Bildbeschreibung deutlich gemacht. Es wurde darauf geachtet, die Texte in dieser Dissertationsschrift möglichst geschlechtsneutral zu formulieren. So weit wie möglich wurde versucht, auf geschlechterspezifische Nomen zu verzichten. Wenn dies nicht möglich war, wurde die gegenderte Form mit Doppelpunkt verwendet, beispielsweise „Entwickler:Innen“. In seltenen Fällen wird auf eine gegenderte Form zugunsten einer besseren Lesbarkeit verzichtet, dies ist nicht als Ausgrenzung zu verstehen.

Für die Grafiken und Abbildungen wurden Farben (und Formen) verwendet, die auch für Menschen mit Farbsehschwäche (Dyschromasie) unterscheidbar sind. Die hierfür verwendeten Farben stammen aus [A9] oder wurden daraus abgeleitet.

2 Wissenschaftlicher Kontext und Stand der Technik

Wie in der Einleitung beschrieben, beschäftigen sich die in dieser Dissertationsschrift vorgestellten Arbeiten mit Methoden für die Bewertung, Entwicklung und Optimierung von Inertialsensorsubsystemen in eingebetteten Systemen sowie Anwendungen, die Daten aus diesen nutzen. Hierbei liegt der Fokus auf Anwendungen, bei denen ein System oder Eingabegerät von einem Menschen bedient wird und dessen Bewegungen mittels der Bewegungssensoren aufgenommen, verarbeitet und in Aktionen umgesetzt werden. Dies kann Beispielsweise ein Smartpen sein, der als digitales Zeigegerät genutzt wird, oder ein Controller, der eine Steuerung mittels Gesten erlaubt. Das primäre Ziel dieser Arbeiten ist es, Methoden, Konzepte und Anwendungen zu entwickeln, mittels derer es auf einfache und unkomplizierte Weise ermöglicht ist, das Sensorsubsystemen eines eingebetteten Systems zu bewerten.

Folgend wird zunächst der wissenschaftliche Kontext der Dissertationsschrift und der Arbeiten beschrieben. Hierbei wird eine Abgrenzung zu anderen Fachgebieten, wie der Messtechnik, der MMI und der digitalen Signalverarbeitung vorgenommen. Anschließend wird der aktuelle Stand der Technik erläutert, bezogen auf die Themengebiete der einzelnen Veröffentlichungen und auf übergeordneter Ebene für diese Dissertationsschrift. Schließlich wird aufgezeigt, was die Neuheiten der in dieser Dissertationsschrift vorgestellten wissenschaftlichen Arbeiten sind.

2.1 Wissenschaftlicher Kontext

In allen in dieser Dissertationsschrift vorgestellten Arbeiten geht es um die Entwicklung, Nutzung und Bewertung von Inertialsensorsubsystemen im Kontext der menschlichen Interaktion. Die Inertialsensorsubsystemen sind hierbei integriert in ein eingebettetes System, etwa in ein Smartphone, Wearable oder in körpernaher Sensorik. Bei den Untersuchungen geht es vorrangig um die technischen Aspekte und Implikationen der Interaktion mit Menschen sowie die Restriktionen und Besonderheiten der eingebetteten Systeme. Es wird untersucht, wie sich die Daten und Ergebnisse aus Bewegungssensoren und Sensorsubsystemen bewerten und nutzen lassen, die von Geräten stammen, welche von einem Menschen genutzt werden.

Die vorgestellten Arbeiten überschneiden sich teilweise mit dem Forschungsgebiet der Mensch-Maschine-Interaktion. Dieses interdisziplinäre Forschungsgebiet beschäftigt sich im Generellen mit der Art und Weise, wie Menschen effizient mit Computern und anderen technischen Geräten interagieren können [A10], sei es mittels traditioneller Eingabegeräte, oder mittels neuer Methoden wie Sprachsteuerung oder Gestenerkennung. In dieser Dissertationsschrift geht es jedoch nicht vorrangig darum, neue Interaktionsmöglichkeiten zwischen Mensch und Maschine zu entwickeln, sondern um die Beurteilung der Genauigkeit und Zuverlässigkeit von Systemen, die Menschen für diese Interaktion verwenden. Dies ist besonders bei den Untersuchungen auf Anwendungsebene in Kapitel 4 der Fall. Zum einen gibt es hier Überschneidungen bei dem Design der entwickelten Testmethode für die Qualität des Sensorsubsystems in Abschnitt 4.1. Die Tests sollten so entworfen werden, dass sie möglichst intuitiv verstanden werden und einfach durchführbar sind. Zum anderen geht es bei der vorgestellten Methode für die Bewertung digitaler Zeigegeräte in Abschnitt 4.4 direkt darum, die subjektive Bewertung von Nutzer:Innen quantitativ ermitteln zu können. Ein Unterschied zu vielen Arbeiten auf dem Gebiet der Mensch-Maschine-Interaktion besteht darin, dass die Untersuchungen primär aus technischer Sicht betrachtet werden. Das vorrangige Ziel der Untersuchungen und vorgestellten Methoden ist es, Möglichkeiten zu schaffen, die verbaute Sensorik oder das Inertialsensorsubsystem zu bewerten und zu verbessern.

Die Themen dieser Dissertationsschrift überschneiden sich ebenfalls teilweise mit den Gebieten Messtechnik und digitalen Signalverarbeitung. Bei der Messtechnik handelt es sich vorrangig um Verfahren, Vorrichtungen und Methoden für die Messung von physikalischen Größen oder Prozessen [A11] sowie deren Bewertung. Bei den im Fokus dieser Dissertationsschrift stehenden Bewegungssensoren handelt es sich ebenfalls um Messeinrichtungen für die Bestimmung physikalischer Größen. Für die vorgestellten Arbeiten ist ein solides Grundverständnis über die physikalische Funktionsweise der untersuchten Mikro-Elektro-Mechanische Systeme (MEMS)-Sensoren notwendig, um die durch die Messungen entstehenden Messfehler deuten und verstehen zu können. Jedoch geht es nicht um die Entwicklung neuer oder präziserer Messmethoden oder um hochgenaue Messungen im industriellen oder technischen Kontext. Der Fokus liegt hier auf Methoden und Konzepte, die es gerade Entwickler:Innen erleichtern sollen, Anwendungen im Bereich der menschlichen Interaktion insbesondere in der Unterhaltungselektronik zu entwickeln.

Bei der digitalen Signalverarbeitung geht es um die Transformation digitaler Signale in einem digitalen System. [A12]. Insbesondere geht es um die Verarbeitung, Interpretation und Veränderung der Daten für einen vorgesehenen Zweck, beziehungsweise um die Extraktion von Informationen aus den erhaltenen Signalen. Beispielsweise kann eine Verarbeitung der Daten durch einen Filter mit endlicher Impulsantwort (Finite Impulse Response, FIR) erfolgen. Die Sensorfusionsalgorithmen, die in dieser Dissertationsschrift betrachtet werden, sind ebenfalls Verfahren der digitalen Signalverarbeitung. Ebenfalls häufig angewandt werden Filter, die das Rauschen in den Sensordaten reduzieren. Eine

Abgrenzung zur klassischen digitalen Signalverarbeitung kann hier darin gesehen werden, dass nicht die Entwicklung neuer Filter oder Verfahren für die Verarbeitung der Daten im Vordergrund steht, sondern die Möglichkeit, bestehende Sensorfusionsalgorithmen zu bewerten oder deren Entwicklung zu vereinfachen.

Vorrangig können die vorgestellten Arbeiten dem Forschungsgebiet der Technischen Informatik, insbesondere der Datentechnik und eingebetteten Systeme zugeordnet werden. Die Arbeiten befassen sich allesamt mit der Verarbeitung und Nutzung der Sensordaten im Kontext eingebetteter Systeme und berücksichtigen dabei deren Beschränkungen und Besonderheiten.

2.2 Stand der Technik

Bei den in dieser Dissertationsschrift betrachteten Systemen handelt es sich vorrangig um eingebettete Systeme, entweder in Form von Smartphones oder als Komponenten eines größeren Systems, wie etwa intelligente Sensoren. Neben den herkömmlichen Versionen mit Mikroprozessor [A13], digitalem Signalprozessor (DSP) oder zentraler Recheneinheit (Central Processing Unit, CPU), wird vermehrt auch spezielle Hardware für Künstliche Intelligenz/maschinelles Lernen verwendet. Diese kann beispielsweise mit TinyML programmiert werden und ermöglicht es, Gesten von Nutzern zu erkennen oder auch neue Gesten zu erlernen, mittels derer ein Nutzer mit den Geräten interagieren kann [A14], [A15]. Weiterhin kann mithilfe der KI-Unterstützung auch eine Reduktion der zu übertragenden Daten durchführen werden und es so erlauben, „Always On“ Funktionalitäten mit geringem Stromverbrauch zu realisieren. Aktuell liegt der Fokus dieser neuen Technologien vorrangig auf der Gestenerkennung und der Datenreduktion [A16]. Daher werden die Möglichkeiten einer KI-Gestützten Fusionierung von Sensordaten zur Bestimmung der Orientierung in dieser Dissertationsschrift nicht weiter betrachtet.

In dieser Dissertationsschrift geht es insbesondere um Methoden für die Bewertung in Inertialsensorsubsystemen. Üblicherweise findet eine solche Bewertung der Qualität und Zuverlässigkeit der Ergebnisse des Inertialsensorsubsystems bei der Entwicklung einer jeden neuen Anwendung oder eines jeden neuen Systems statt. Oftmals werden Bewertungsmethoden für das Sensorsubsystem im Kontext eines speziellen Anwendungsfalles präsentiert. Dabei geht es primär um die Nutzbarkeit der vorgestellten Anwendung und nicht um die Bewertungsmethode an sich. Beispielsweise wird in [A2] eine spezielle Apparatur verwendet, um die Genauigkeit der entwickelten Methode bei der Bestimmung der Bewegung eines Arms zu ermitteln.

Es wurden auch Arbeiten veröffentlicht, die sich mit einer generellen Methodik beschäftigen, die es erlaubt, die Qualität und Zuverlässigkeit eines Inertialsensorsubsystems zu bestimmen. So wird in [A17] eine einfach nutzbare Methode vorgestellt, die die Qualität von Inertialsensoren bewertet, indem mehrere Sensoreinheiten starr an einer Platte

befestigt werden. Diese Bewertungsmethode unterscheidet sich von der Methode, die in dieser Dissertationsschrift genutzt und weiterentwickelt wird [A4], darin, dass keine Referenz für die Bewertung genutzt wird. Die Bewertung der Qualität geschieht in [A17] über die Bestimmung der Unterschiede zwischen den Messwerten. Auch wenn dies eine einfache Bewertung erlaubt, so kann keine absolute Aussage über die Qualität getätigt werden, da nur die relativen Unterschiede betrachtet werden können. Im Kontext der Analyse von menschlichen Bewegungen wurde in [A18] eine Methodik vorgestellt, die wie [A4] ebenfalls ein Kamerasystem verwendet, um die Qualität und Genauigkeit von Sensorfusionsalgorithmen zu untersuchen. Diese Methodik lässt sich ebenfalls universell für die Bewertung eines Inertialsensorsubsystemen einsetzen, ist jedoch nicht darauf ausgelegt, einfach nutzbar zu sein. Eine Arbeit, die ebenfalls eine Methode vorstellt, die es erlaubt auf kostengünstige und einfache Weise die Inertialsensoren zu bewerten, wird in [A19] vorgestellt. Auch diese Methode eignet sich für die Bewertung der Inertialsensoren. Es werden für die Bestimmung der Qualität der Sensoren jedoch keine Referenzen verwendet, sondern konstante Größen wie Gravitation und Magnetfeldstärke. Dies erlaubt einen generellen Eindruck über die Qualität einiger Sensoren, funktioniert aber nicht für die oft verwendete Orientierung. Ohne eine Referenz lässt sich zudem keine absolute Aussage über die Qualität der Sensoren treffen. Eine weitere Methodik für die Bewertung der Inertialsensoren wird in [A20] vorgestellt. In dieser Methode wird ein Field Programmable Gate Array (FPGA) für die Bewertung genutzt. Die Referenzen werden durch eine genaue statische Positionierung und Nutzung einer Präzisions-Drehscheibe ermittelt.

Für die Bewertung der Inertialsensoren sowie der Sensorfusion in Smartphones gibt es mehrere wissenschaftliche Arbeiten, die sich mit diesem Thema befassen. Zuerst ist hier die Arbeit [A4] zu nennen, welche die Methodik für die Bewertung der Sensoren des Smartphones mithilfe der integrierten Kamera und einem ausgedruckten Muster beschreibt. Auf dieser Arbeit baut die in dieser Dissertationsschrift vorgestellte Methodik auf und entwickelt diese weiter. Eine sehr ähnliche Methode wird auch im Rotationvector Crosscheck des CTS-Verifier von Google verwendet [A21], der die Kompatibilität und Qualität von neuen Android-Geräten gewährleisten soll. Dieser Test zielt hauptsächlich darauf ab, dass die Messfehler für die Orientierung innerhalb festgesetzter Grenzen bleiben. Für Nutzer:Innen oder Entwickler:Innen von Android-Anwendungen ist diese Anwendung nicht geeignet, um die Qualität der verbauten Sensoren zu bewerten. In [A22] wird eine weitere Anwendung für Android-Smartphones vorgestellt, die deren Sensoren bewertet. Die Tests werden statisch durchgeführt und erlauben die Analyse der statistischen Eigenschaften der Sensorwerte, wie etwa Standardabweichung der Messwerte, Kontinuität der Zeitstempel oder die Auflösung der Sensordaten. Für eine direkte Bewertung der Genauigkeit des Sensorsubsystems ist diese Anwendung jedoch nicht nutzbar. Des Weiteren werden statistische Untersuchungen, wie sie in [B8] durchgeführt werden, beispielsweise auch für die Bewertung von Lichtsensoren genutzt [A23]. In [A24] wird eine weitere Methode vorgestellt, mit der die Inertialsensoren eines Smartphones bewertet

werden können. Für die Bewertung und Generierung der Referenz wird ein Roboterarm verwendet. Wie eine Untersuchung in Abschnitt 4.3 zeigt, kann diese Methode aber gerade bei Smartphones zu unbrauchbaren Ergebnissen führen.

Wie bereits angeklungen ist, wird die Kamera des Smartphones für die Bewertung der Daten der Inertialsensoren verwendet. Eine Kombination von Kameradaten mit den Daten von Inertialsensoren ist durchaus üblich. Sehr häufig werden die Daten von Gyroskop und/oder Beschleunigungssensor verwendet, um das Bild der Kamera zu stabilisieren [A25]. Die verwendeten Verfahren sind die optische Bildstabilisierung (Optical Image Stabilization, OIS) und die digitale Bildstabilisierung (Electronic Image Stabilization, EIS) sowie hybride Varianten. Bei der optischen Bildstabilisierung werden entweder die Linse oder der Sensor gedreht und verschoben, um das Bild zu stabilisieren, bei der digitalen Bildstabilisierung sind Bildsensor und Linse starr. Die Stabilisierung wird hier dadurch erreicht, dass das digitale Bild verschoben wird. Hierfür darf nur ein Ausschnitt in der Mitte des Bildes die eigentliche Aufnahme zeigen. Die Ränder werden benötigt, um das Bild verschieben zu können. Beide Methoden haben gemein, dass sie die Daten von Inertialsensoren benötigen, um die Korrektur durchzuführen. Oftmals ist hierfür ein Sensor direkt bei der Kamera verbaut, um die benötigten Eigenschaften wie eine möglichst geringe Latenz zu erreichen [A26]. Der Unterschied zu den Arbeiten in dieser Dissertationsschrift ist, dass hier die Daten der Kamera verwendet werden, um das Inertialsensorsubsystem zu bewerten. Bei der Bildstabilisierung werden die Daten der Inertialsensoren genutzt, um das Kamerabild zu verbessern.

In dieser Dissertationsschrift wird eine Methode für die Bewertung von digitalen Eingabegeräten vorgestellt. Obwohl es für diese Geräte die ISO-Norm 9242 gibt, die deren allgemeine Nutzbarkeit sicherstellen soll [A27], wurde in [A28] festgestellt, dass dies nicht ausreicht, um eine gute subjektive Erfahrung bei der Nutzung zu gewährleisten. Eine Arbeit, die sich mit der Genauigkeit von digitalen Zeigegeräten befasst, wird in [A29] vorgestellt. Es werden ebenfalls mehrere messbare Größen abgeleitet, anhand derer man die Performance der untersuchten Zeigegeräte bewerten kann. Hierbei wird jedoch nicht die subjektive Erfahrung bei der Benutzung untersucht, sondern wie effizient sich die untersuchten digitalen Zeigegeräte für eine typische Office-Umgebung nutzen lassen. Ziel der in dieser Dissertationsschrift beschriebenen Arbeit ist es, anhand von Messmethoden abzuschätzen, wie die subjektive Erfahrung bei der Benutzung des Gerätes sein wird. Die zwei meistverbreiteten Methoden für die Bewertung von Zeigegeräten sind Fitt's Law [A30] [A31] und Accot-Zhai Steering Law [A32]. Fitt's law wird auch in der genannten ISO-Norm für die Bewertung verwendet. Bei beiden Methoden wird als Metrik für die Bewertung der Zeigegeräte die Geschwindigkeit verwendet, mit der Ziele auf dem Bildschirm angesteuert werden können. Die Funktionalität beider Methoden wird in den Abschnitt 3.1 näher erläutert. Es ist hier anzumerken, dass beide Bewertungsmethoden in der Arbeit zur Bewertung des Smart-Pens, die in dieser Dissertationsschrift vorgestellt wird, nicht verwendet werden, da die Beschränkung auf die Geschwindigkeit nicht ausreicht, um die untersuchten technischen Aspekte zu beurteilen.

In Abschnitt 5.2 dieser Dissertationsschrift werden die Ergebnisse von Untersuchungen vorgestellt, die die funktionalen und extrafunktionalen Eigenschaften unterschiedlicher Sensorfusionsalgorithmen für die Bestimmung der Orientierung betrachten. Ähnliche Untersuchungen wurden bereits in anderen Arbeiten durchgeführt, jedoch unter anderen Bedingungen oder mit anderen Methoden. So wurde in [A33] ebenfalls vergleichende Untersuchungen von Kalman-filter, Mahony-Filter und Madgwick-Filter durchgeführt. Bezüglich der Nutzbarkeit der Fusionsalgorithmen kommt diese Arbeit auf dieselben Ergebnisse. Weitere Arbeiten, die die Nutzbarkeit von Mahony Filter und Madgwick-Filter bestätigen, werden in [A34] und [A35] vorgestellt. Hierbei wird in [A34] die einfachere Kalibrierung der Komplementärfilter hervorgehoben. Im Gegensatz hierzu zeigt sich in den Untersuchungen, die in [A36] durchgeführt wurden, dass das Kalman-Filter deutlich bessere Ergebnisse liefert. Die in [A36] gezeigten Messungen wurden jedoch mithilfe eines Roboters durchgeführt, was einem industriellen oder technischen Kontext nahekommt. Im Gegensatz zu den Untersuchungen, die in dieser Dissertationsschrift vorgestellt werden, wird in den genannten Untersuchungen nicht der Einfluss unterschiedlicher Datenformate betrachtet. Ebenfalls werden die extrafunktionalen Eigenschaften wie Rechenaufwand und Speichergröße nicht betrachtet.

Zuletzt wird in dieser Dissertationsschrift ein Framework vorgestellt, welches es erlaubt, aufgenommene Sensordaten zu augmentieren und zu verändern. Dies soll es ermöglichen, auf einfache Weise ein in Entwicklung befindliches Sensorsubsystem zu bewerten oder Datensätze für Training und Verifikation von Algorithmen für die Gestenerkennung zu generieren. Für die Bewertung von Sensorsubsystemen werden oftmals künstlich erzeugte Sensordaten verwendet, welche mit der Hilfe von Werkzeugen erstellt werden. Zwei etablierte Werkzeuge hierfür sind die *Sensor Fusion and Tracking Toolbox* von Matlab [A37] und der Open-Source Simulator von Aceinna [A38]. Das Hauptziel von diesen ist die Simulation von Szenarien im Bereich Navigation und von Fahrzeugsystemen in Kombination mit GPS-Daten. Im Vordergrund steht die Generierung von künstlichen Daten für Inertialsensoren aus vordefinierten Trajektorien und Sensormodellen und nicht die Erweiterung oder Veränderung von aufgenommenen Sensordaten. Eine Kombination der genannten Werkzeuge mit dem vorgestellten Framework ist denkbar, um auf schnelle und einfache Weise Abwandlungen und Erweiterungen der generierten Sensordaten zu erzeugen. Eine Kombination mit anderen Arbeiten, die sich mit der Generierung von künstlichen Sensordaten befassen ist ebenfalls möglich. Etwa mit den Simulatoren, die in [A39] und [A40] präsentiert werden. Das Prinzip und die Vorteile der Augmentation von Sensordaten für maschinelles Lernen werden in [A41] und [A42] vorgestellt. Hierbei werden Bilddaten einer Kamera verändert, um mehr Daten zum Training lernfähiger Algorithmen zu erzeugen. Auch im Bereich der Mensch-Maschine-Interaktion werden Sensordaten durch Veränderungen generiert, um Algorithmen darin zu trainieren, menschlichen Aktivitäten zu erkennen [A43].

2.3 Wissenschaftliche Beiträge

Im vorherigen Abschnitt wurde diskutiert, wie die in dieser Dissertationsschrift beschriebenen Arbeiten im aktuellen Stand der Technik einzuordnen sind. Auf der einen Seite im Bereich von Arbeiten, die sich mit der Bewertung und Entwicklung von Sensorsubsystemen oder Sensorfusionsalgorithmen beschäftigen und auf der anderen Seite im Bereich von Arbeiten, die sich mit der Generierung von Testfällen, künstlichen Sensordaten oder dem Training von lernfähigen Algorithmen beschäftigen. Einer Besonderheit der hier vorgestellten Arbeiten ist der Kontext der menschlichen Interaktion.

Die wissenschaftliche Neuheit der in dieser Dissertationsschrift beschriebenen Arbeiten besteht darin, dass die beschriebenen Methoden, Konzepte und Anwendungen darauf ausgelegt sind, einfach und unkompliziert von Nutzer:Innen und Entwickler:Innen genutzt zu werden. Bei der Methode zur Bewertung des Sensorsubsystems von Smartphones wird lediglich der Ausdruck eines Musters benötigt, um den Test durchzuführen. Die Verwendung derselben Methode bei der Untersuchung der Sensorfusionsalgorithmen mit unterschiedlichen Datentypen hat hier auch die Vielseitigkeit dieses Ansatzes gezeigt. Viele der anderen verwandten Arbeiten für die Bewertung eines Sensorsubsystems setzen teures Equipment, wie einen Roboterarm [A24] oder Laborbedingungen voraus. Diese Voraussetzungen sind gerade bei Endanwender:Innen oder bei kleineren Unternehmen nicht vorhanden. Die in dieser Dissertationsschrift vorgestellten Arbeiten ermöglichen diesen, Bewertungen ohne zusätzliches Equipment durchzuführen.

Die Untersuchung der Sensorfusionsalgorithmen verwendet ebenfalls die in [A4] vorgestellte Methode zur Bewertung der Ergebnisse. Es wird damit aufgezeigt, wie solche Untersuchungen ohne teures zusätzliches Equipment trotzdem zuverlässig durchgeführt werden können. Weiterhin untersuchen die meisten Arbeiten, die sich mit der Bewertung und dem Vergleich von Sensorfusionsalgorithmen beschäftigen, nur die Unterschiede zwischen den Algorithmen an sich [A33], [A34], [A35]. Welche Vor- und Nachteile die Nutzung von anderen Datentypen haben kann, wird in keiner der verwandten Arbeiten betrachtet. Auch eine statistische Untersuchung der Faktoren, die sich auf die Ergebnisse auswirken, ist eine Neuheit, die in dieser Dissertationsschrift vorgestellt wird.

Die letzte Arbeit, die das Framework für die Augmentation von Sensordaten vorstellt, zeigt eine neuartige Methode, mithilfe derer die Sensorfirmware direkt auf der Hardware bewertet und entwickelt werden kann. Die Neuheit besteht hierbei einerseits darin, dass die Daten auf einfach Weise manipuliert werden können und die Ergebnisse live auf der Hardware beobachtet werden können. Zudem befassen sich verwandte Arbeiten oftmals mit der Generierung von künstlichen Sensordaten [A37], [A38] und nicht mit der Veränderung und Augmentation bestehender Daten. Daher kann das vorgestellte Framework auch als Erweiterung für diese Methoden betrachtet werden.

3 Grundlagen

In diesem Kapitel werden die Grundlagen erläutert, auf denen diese Dissertationsschrift aufbaut. Die in diesem Kapitel beschriebenen Grundlagen sind für das Verständnis der gesamten Dissertationsschrift von Relevanz. Grundlagen, die nur für einen Teil der Arbeit wichtig sind, werden am Anfang des jeweiligen Kapitels erläutert.

Der Fokus dieser Dissertationsschrift liegt auf der Bewertung und Verbesserung von Anwendungen, die Daten aus Inertialsensoren für eine Mensch-Maschine-Interaktion nutzen. Daher wird zuerst beschrieben, worum es sich bei der Mensch-Maschine-Interaktion handelt, wozu sie notwendig ist und welche Besonderheiten es bei der Interaktion durch Bewegungen gibt. Die meisten der Untersuchungen werden im Kontext von eingebetteten Systemen durchgeführt. Daher werden anschließend die Grundlagen eingebetteter Systeme und im Speziellen intelligenter Sensoren behandelt. Nachfolgend wird eine Übersicht über Inertialsensoren, insbesondere MEMS-Sensoren gegeben. Aufbauend auf den Informationen über Inertialsensoren wird die Sensordatenfusion erläutert. Die Fehler, die bei Messungen der Inertialsensoren und bei der Sensordatenfusion entstehen und für diese Dissertationsschrift relevant sind, werden danach beschrieben. Abschließend werden die statistischen Bewertungsmethoden erläutert, die in dieser Dissertationsschrift für die Analyse verwendet werden, um Eigenschaften und Fehlereinflüsse der Sensordatenfusion zu analysieren, oder die Sensorsubsysteme zu bewerten.

3.1 Mensch-Maschine-Interaktion

Wie es der Name vermuten lässt, handelt es sich bei der Mensch-Maschine-Interaktion um die Schnittstelle, mittels derer ein Mensch mit einem Computer, einer Industrieanlage oder anderen technischen Geräten, wie etwa Internet der Dinge (Internet of Things, IoT)-Geräten interagiert. Die Interaktion ist hierbei bidirektional [A44]. Für Nutzer:Innen muss es eine Möglichkeit geben, dem Computer einen Befehl mitzuteilen. Der Computer muss über eine Möglichkeit verfügen, der nutzenden Person Informationen mitzuteilen oder die Eingaben von dieser zu quittieren [A10]. Klassisch geschieht die Eingabe über Tastatur und Maus oder über einen Touchscreen. Die Ausgabe geschieht primär über einen Bildschirm, kann aber auch auditiv erfolgen. Im letzten Jahrzehnt hat sich ein ganzes interdisziplinäres Forschungsfeld gebildet, welches sich mit MMI beschäftigt [A44]. Dieses umfasst viele unterschiedliche Teilbereiche wie Kognition, Kontext, Psychologie,

Design und Interaktion [A45]. Es werden neue und intuitive Methoden gesucht und entwickelt, mittels derer Menschen natürlicher mit Maschinen und Computern interagieren können. Hierzu zählt auch die Eingabe von Befehlen mithilfe von Sprache oder Gesten [A46][A47]. In dieser Dissertationsschrift wird ausschließlich die Schnittstelle zwischen Mensch und Maschine betrachtet. Etwa, wie mittels Gesten oder Bewegungen von den Inertialsensoren Daten erzeugt werden, die vom Computer verwendet, bewertet und interpretiert werden können.

Interaktion mittels Gesten

Bei der Interaktion zwischen Mensch und Computer mittels Gesten führt eine Person bestimmte Bewegungen aus, die vom Computer als Befehl erkannt werden. Diese Bewegungen werden entweder von einem externen Gerät mittels Kamera [A48], Infrarot [A49] oder Inertialsensoren [A50], [A51] erkannt. Der Fokus dieser Dissertationsschrift liegt auf der Interaktion mittels Gesten, die von Inertialsensoren aufgenommen werden, und im weiteren Sinne auf der generellen Aufnahme und Verarbeitung von menschlichen Bewegungsdaten mittels Inertialsensoren. Für diese Art der Interaktion wird ein Messsystem benötigt, welches von der nutzenden Person bewegt wird, um deren Bewegung zu erkennen. Hierbei kann es sich wie bei [A50] und [A51] um speziell für diesen Zweck entwickelte Eingabegeräte handeln, wie etwa um einen Smartpen, der in Abbildung 3.1 konzeptuell dargestellt ist. Ferner können jedoch auch Smartphones, Smartwatches oder an Kleidung oder Schuhen befestigte Sensoren (Wearables) für diese Aufgabe verwendet werden.

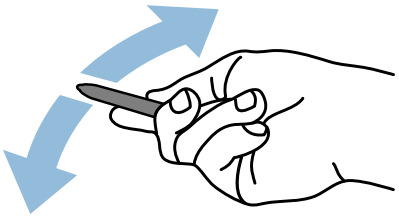


Abbildung 3.1: Beispiel einer Anwendung mit Gestenerkennung: Ein Smartpen

Digitale Zeigegeräte

Die am häufigsten verwendete Eingabemethode an Computern sind digitale Zeigegeräte wie die Computermaus und der Touchscreen bei mobilen Endgeräten [A45]. Die Computermaus ermöglicht es, einen Zeiger auf dem Bildschirm des Gerätes zu bewegen und Bedienelemente auszuwählen. Wenn neue Zeigegeräte entwickelt werden, ist es sinnvoll, diese mit bestehenden Geräten zu vergleichen, oder eine Möglichkeit zu haben, die Nutzbarkeit des neuen digitalen Zeigegerätes zu bewerten. Eine solche Untersuchung wird

auch in Abschnitt 4.4 beschrieben. Für die Bewertung von digitalen Zeigegeräten haben sich in der MMI zwei Verfahren etabliert. Diese sind Fitt's Law und die Accot-Zhai Steering Law.

Fitt's Law

Mit der ersten Bewertungsmethode, Fitt's Law, soll bewertet werden, wie schwierig es ist, Ziele auf dem Bildschirm mit dem Zeigegerät auszuwählen. In die Bewertung fließen die Distanz zwischen den Zielobjekten, deren Größe sowie die Zeit ein, die gebraucht wird, um das Zielobjekt zu erreichen. Oftmals wird Fitt's Law für die Bewertung von grafischen Oberflächen genutzt, um deren Effizienz zu bewerten. Mit einer standardisierten Testumgebung lässt sich aber auch die Nutzbarkeit von digitalen Zeigegeräten bewerten [A30]. Für die Bewertung von Zeigegeräten wird zuerst der Schwierigkeitsindex berechnet, der in Gleichung (3.1) gezeigt ist. D ist hierbei die Distanz zwischen den Objekten und W deren Breite. Der Aufbau ist in Abbildung 3.2 dargestellt.

$$ID = \log_2\left(\frac{D}{W} - 1\right) \tag{3.1}$$

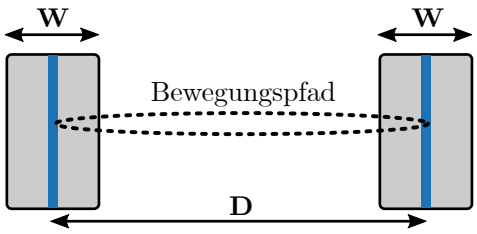


Abbildung 3.2: Beispiel eines Tests für Fitt's Law mit zwei Zielen. Abgewandelt aus [A30]

Für die endgültige Bewertung wird der Durchsatz TP errechnet, der sich aus dem Schwierigkeitsindex ID und der Ausführungszeit t ergibt. Dies ist in Gleichung (3.2) gezeigt.

$$TP = \frac{ID}{t} \tag{3.2}$$

Daraus ergibt sich, dass das Bewertungskriterium für einen einzelnen Test die Geschwindigkeit ist, mit der die Zielobjekte angesteuert werden können. Für eine umfassende Bewertung eines Zeigegerätes sind Tests mit unterschiedlichen Schwierigkeitsindices notwendig.

Accot-Zhai Steering Law

Die Accot-Zhai Steering Law kann als Erweiterung von Fitt's Law gesehen werden. Fitt's Law hat die Beschränkung, dass die Bewertung nur für schnelle zielgerichtete Bewegungen funktioniert. Kontinuierliche Bewegungen können nicht bewertet werden [A31]. Bei der Accot-Zhai Steering Law fließt der Pfad in die Bewertung mit ein. Bei dieser Bewertungsmethode muss der Zeiger entlang eines vorher definierten Pfades bewegt werden. In Abbildung 3.3 ist beispielhaft ein kreisförmiger Tunnel dargestellt [A32].

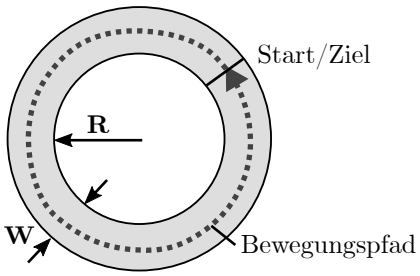


Abbildung 3.3: Beispiel eines kreisförmigen Tunnels. Abgewandelt aus [A32]

Für die Bewertung des Zeigegerätes wird wieder der Schwierigkeitsindex berechnet, der sich aus der gewählten Form des Tunnels ergibt. Für einen kreisförmigen Tunnel ist der Schwierigkeitsindex in Gleichung (3.3) gezeigt.

$$ID = \frac{2\pi R}{W} \tag{3.3}$$

Hierbei ist R der Radius des Kreises und W die Breite des Tunnels. Die Gesamtbewertung ergibt sich dann aus der Zeit t, die benötigt wird, um den Zeiger durch den Pfad zu bewegen, wie in Formel Gleichung (3.4) dargestellt.

$$t = a + b \cdot ID \tag{3.4}$$

Die Konstanten a und b sind hierbei experimentell zu ermitteln. Wie aus den Formeln zu entnehmen ist, wird die Geschwindigkeit, mit der der Zeiger bewegt werden kann, als Bewertungskriterium herangezogen. Es werden nur die Testdurchläufe zur Bewertung genutzt, bei denen der Zeiger innerhalb des vordefinierten Tunnels bleibt.

Besonderheiten bei menschlichen Bewegungen

Die Aufnahmen von Inertialsensordaten menschlicher Bewegungen unterscheiden sich oft stark von den Aufnahmen von Inertialsensordaten technischer Systeme. Beispielsweise bei Robotern sind die Bewegungen sehr exakt, geradlinig und abrupt. Die Bewegungen von einem Menschen sind fließender und es ist sehr unwahrscheinlich, dass eine Bewegung präzise wiederholt werden kann. Besonders bei der Erkennung von Gesten kann daher selten mit exakten Werten gearbeitet werden. Oftmals wird maschinelles Lernen genutzt, um die Gesten zu erkennen [A52].

Eine weitere Besonderheit, die in dieser Dissertationsschrift eine Rolle spielt, sind die Frequenzanteile, welche in den Daten von Aufnahmen menschlicher Bewegungen zu finden sind. Im Gegensatz zu Robotern oder anderen technischen Geräten lassen sich in Aufnahmen menschlicher Bewegungen selten hochfrequente Anteile finden, da die Fähigkeit eines Menschen, ruckartige Bewegungen auszuführen, begrenzt sind. Dafür lassen sich jedoch auch niederfrequente Anteile in den Bewegungen finden, welche nicht direkt zu der eigentlichen Geste oder Bewegung gehören [A3].

3.2 Eingebettete Systeme

Bei den in dieser Dissertationsschrift betrachteten Geräten handelt es sich überwiegend um eingebettete Systeme. Eingebettete Systeme sind digitale Computersysteme, die in einem technischen Kontext eingebunden sind [A53], [A54]. Sie wurden für eine bestimmte Aufgabe entwickelt und die Hardware ist auf diese Aufgabe abgestimmt, beispielsweise auf die Steuerung einer Waschmaschine. Ein eingebettetes System ist ein Teil eines größeren Systems, in dem es Mess-, Steuer- oder Kontrollfunktionen übernimmt. Dabei verfügt das eingebettete System meist über Sensorik, um Informationen über die Umgebung oder über das technische System zu erhalten [A54]. Diese Informationen werden verarbeitet und anschließend in Steuerbefehle umgesetzt. Ein typisches Beispiel eines eingebetteten Systems ist die Steuerung einer Waschmaschine oder einer Ampel.

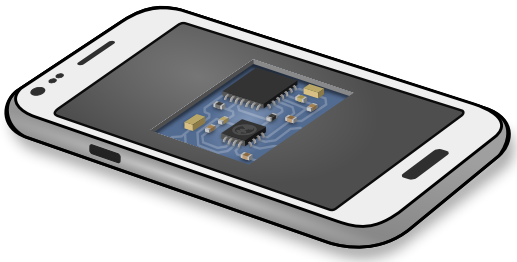


Abbildung 3.4: Bekanntes Beispiel eines eingebetteten Systems: Das Smartphone

Ein besonderes Merkmal der eingebetteten Systeme ist die meist eingeschränkte Ressourcenverfügbarkeit. Besonders im Vergleich mit Rechnersystem wie einem Laptop oder Desktop-PC verfügen eingebettete Systeme nur über wenig Speicher und Rechenleistung. Der Grund dafür ist, dass eingebettete Systeme klein, kostengünstig und energiesparend sein sollen [A53]. Beispielsweise soll in einer autonomen Messstation, die mit einer Batterie betrieben wird, die Lebensdauer der Batterie möglichst lang sein. Die Ressourcenknappheit in einem eingebetteten System spielt im Laufe dieser Dissertationsschrift, speziell in Abschnitt 5.2 eine wichtige Rolle.

Eine Sonderstellung unter den eingebetteten Systemen hat das in Abbildung 3.4 gezeigte Smartphone. Zum einen sind diese Geräte nicht auf eine spezielle Aufgabe beschränkt, sondern, ähnlich wie ein Laptop oder Desktop-PC, in der Lage eine Vielzahl an Programmen auszuführen. Weiterhin sind Smartphones deutlich weniger in ihren Ressourcen beschränkt. Gerade Speicher und Rechenleistung sind vergleichbar mit (älteren) Allzweckcomputern. Einige Gemeinsamkeiten mit den klassischen eingebetteten Systemen bleiben. Etwa die Erfüllung spezieller Aufgaben wie Telefonieren, Fotografieren oder Sensorik zum Erfassen der Umgebung. Die Beschränkung bezüglich des Energieverbrauchs bleibt ebenfalls teilweise bestehen. Man kann ein Smartphone auch als eine Mischung aus einem eingebettetem System und einem Allzweckcomputer betrachten [A55].

Datenformate

Bei Allzweckcomputern, anwendungsspezifischen Systemen und auch eingebetteten Systemen existieren zwei Basisformate, in denen Zahlen dargestellt werden können. Als Integer für ganzzahliger Werte, oder als Fließkommazahl für rationale Werte. Aufgrund der komplexeren Darstellung von Fließkommazahlen gibt es gerade in eingebetteten Systemen und kleinen Mikrocontrollern keine Hardware-Unterstützung für mathematische Operationen mit diesen. Diese Operationen benötigen im Gegensatz zu ihren Äquivalenten bei den Integer-Zahlen deutlich mehr Chipfläche und Energie. Ist keine Hardware-Unterstützung für Fließkommazahlen vorhanden, müssen die Operationen in der Software emuliert werden. Dies benötigt deutliche mehr Rechenzeit. Um trotzdem auch auf einem Mikrocontroller nicht auf ganzzahlige Werte beschränkt zu sein, kann man die Integer auch als Festkommazahlen interpretieren. Hierfür müssen lediglich einige mathematische Operationen angepasst werden. Gerade in eingebetteten Systemen mit eingeschränkter Rechenkapazität muss oft abgewägt werden, welche Darstellung der Werte verwendet wird. Folgend werden die beiden Formate für rationale Zahlen vorgestellt.

Fließkommazahlen

Standardmäßig verwenden die meisten Algorithmen für Sensordatenfusion Fließkommazahlen für die Berechnung und Speicherung der Werte. Die Fließkommazahlen werden bevorzugt verwendet da sie im Gegensatz zu Festkommazahlen einen großen Wertebereich abdecken können. Bei Fließkommazahlen mit einfacher Genauigkeit können Zahlen im Bereich von $\pm 10^{-44.85}$ bis zu $\pm 10^{38.53}$ dargestellt werden [A56]. Die Genauigkeit beträgt dabei ~ 7 Dezimalstellen [A57]. Der Aufbau einer Fließkommazahl ist in Abbildung 3.5 dargestellt.

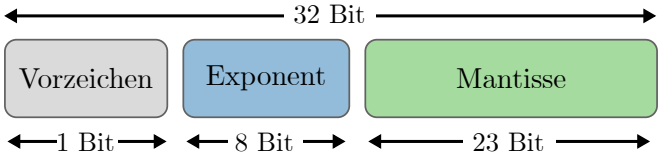


Abbildung 3.5: Aufbau einer Fließkommazahl nach dem IEEE 754 Standard, übernommen aus [A56]

Das erste Bit wird genutzt, um das Vorzeichen darzustellen, 8 Bit für den Exponenten und 23 Bit für die Mantisse. Im Gegensatz zu Integer-Zahlen, bei denen eine mathematische Operation, wie etwa eine Addition innerhalb eines Taktzyklus möglich ist, werden bei Fließkommazahlen mehrere Operationen benötigt:

1. Angleichen der Exponenten beider Zahlen. Dafür wird ein arithmetischer Shift nach links für die kleinere Zahl durchgeführt.
2. Addition der Mantissen beider Zahlen. Dies kann als normale Integer-Addition durchgeführt werden.
3. Normalisierung des Ergebnisses der Addition.

Bei der Normalisierung werden Mantisse und Exponent so angepasst, dass der ganzzahlige Anteil der Mantisse immer 1 ist, wobei die 1 nicht mitgespeichert wird [A56]. Die Darstellung muss also der Form $Zahl := 1, Mantisse \times 2^{Exponent}$ entsprechen.

Festkommazahlen

Bei den Festkommazahlen handelt es sich um Integerwerte, die so interpretiert werden, dass n Bits den ganzzahlige Anteil der Zahl darstellen und m Bits der Nachkommaanteil [A58]. Hierbei gilt, dass $n+m \stackrel{!}{=} Bitlänge-1$ ist. Ein Bit ist für das Vorzeichen reserviert. Der Aufbau einer 32 Bit-Festkommazahl ist in Abbildung 3.6 dargestellt.

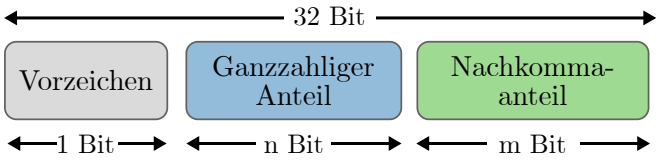


Abbildung 3.6: Aufbau einer Festkommazahl mit Vorzeichen

Aus dem Aufbau ergibt sich, dass Genauigkeit und Wertebereich durch die Position des Kommas bestimmt sind. Je mehr Bits für den Nachkommaanteil genutzt werden, desto genauer ist die Darstellung. Der Wertebereich wird aber um denselben Faktor kleiner. Die Rechenvorschriften bei Festkommazahlen sind deutlich einfacher als bei Fließkommazahlen.

Für Addition und Subtraktion können bei Festkommazahlen mit derselben Anzahl an Nachkommastellen die normalen Operationen für Integer-Zahlen verwendet werden. Bei der Multiplikation und Division können zwar auch die Operationen für Integer verwendet, es muss jedoch jeweils das nächstgrößeren Datenformat für Zwischenergebnisse der Operation verwendet werden, um einen Überlauf der Daten zu vermeiden [A59]. Bei 16 Bit Daten wird ein 32 Bit-Datentyp verwendet werden, bei 32 Bit ein 64 Bit-Datentyp. Weiterhin ist nach der Multiplikation ein Schieben des Ergebnisses um m Nachkommastellen nach rechts erforderlich (entspricht einer Division durch 2^m). Nach einer Division ist es erforderlich, dass der Dividend vor der Division um m Nachkommastellen nach links geschoben wird (entspricht einer Multiplikation mit 2^m).

Weiterhin kann es notwendig sein, die Ergebnisse mathematisch zu runden. Sowohl bei Division als auch Multiplikation würden ohne Runden die nicht mehr darstellbaren Bit abgeschnitten werden, was einem permanenten Abrunden gleichzusetzen ist und zu einem wachsenden Fehler führen würde.

Intelligente Sensoren

Ein besonderes eingebettetes System, welches in dieser Dissertationsschrift betrachtet wird und in Abschnitt 5.2 eine Rolle spielt, ist der intelligente Sensor (engl. Smart Sensor). Diese Systeme umfassen neben den Sensoren zur Aufnahme physikalischer Größen auch eine Recheneinheit für die Vorverarbeitung oder Fusion der aufgenommenen Sensordaten sowie eine Schnittstelle für die Kommunikation mit anderen Geräten [A60], [A61]. Für die Kommunikation kommen sowohl Funk- also auch kabelbasierte Methoden infrage. Man kann einen intelligenten Sensor als eine Weiterentwicklung von der Inertialen Messeinheit (Inertial Measurement Unit, IMU) ansehen.

Ein exemplarischer Aufbau eines intelligenten Sensors ist in Abbildung 3.7 abgebildet. Die drei Sensoren, Beschleunigungssensor (ACC), Gyroskop (GYR) und Magnetometer

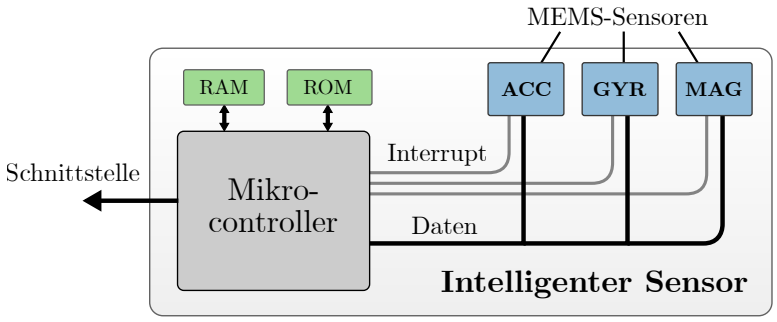


Abbildung 3.7: Intelligenter Sensor bestehend aus Mikroprozessor, drei Inertialsensoren, Speicher und Kommunikationsschnittstelle

(MAG) sind mittels Datenleitung und Interrupt für Ereignisse mit dem Mikrocontroller verbunden. Dieser hat Zugriff auf Speicher (Arbeitsspeicher (Random-Access Memory, RAM) und Festwertspeicher (Read-Only Memory, ROM)), sowie auf eine Kommunikationsschnittstelle.

3.3 Mathematische Begriffe

Viele der Arbeiten dieser Dissertationsschrift verwenden spezielle mathematische Methoden für die Beschreibung oder Manipulation von Objekten im dreidimensionalen Raum $\mathbb{R}^3$. Hierbei handelt es sich um Transformationsmatrizen, homogene Koordinaten und Quaternionen. Quaternionen dienen zur Darstellung der räumlichen Orientierung und werden von den untersuchten Sensorfusionsalgorithmen verwendet. Transformationsmatrizen und homogene Koordinaten kommen bei der Bestimmung der Referenz durch die Kamera zum Einsatz.

3.3.1 Transformationsmatrizen

Transformationsmatrizen oder auch Abbildungsmatrizen werden in der linearen Algebra verwendet. Sie beschreiben eine lineare Abbildung zwischen zwei Vektorräumen. Mittels Transformationsmatrizen können die Koordinaten eines Punktes im Raum transformiert, also geändert werden. Transformationsmatrizen werden in der Computergrafik verwendet, um Rotation, Skalierung, Scherung, Spiegelung und Translation eines Objektes im Raum zu beschreiben. Der Vorteil der Darstellung dieser Eigenschaften in Form eine Matrix ist, dass alle Informationen in einer einzigen Matrix vereint werden können und somit die Komplexität der Berechnung deutlich verringert werden kann [A62]. Ein Beispiel einer Transformation mit Drehung und Translation im zweidimensionalen Raum ist in Abbildung 3.8 dargestellt.

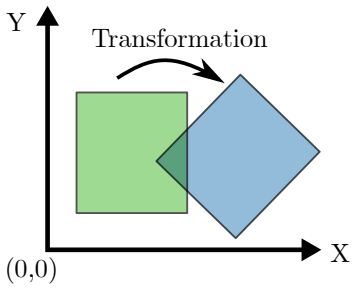


Abbildung 3.8: Beispiel eine Transformation im zweidimensionalen Raum

In der euklidischen Geometrie wird die Position eines Punktes im $\mathbb{R}^3$ über einen Vektor mit drei Achsen x, y und z beschrieben. Um eine Translation in einer Transformationsmatrix darzustellen, ist es notwendig, dass diese eine Dimension mehr hat, als der Vektorraum, in dem die Transformation durchgeführt werden soll [A62]. Im dreidimensionalen Raum wird also eine 4×4 Matrix benötigt. Durch die Erweiterung auf eine weitere Dimension w, werden die Koordinaten eines Objektes im Raum als homogene Koordinaten beschrieben.

3.3.2 Homogene Koordinaten

Homogene Koordinaten werden in der Computergrafik verwendet, um mittels 4×4 Transformationsmatrizen Translationen durchführen zu können. Wie in Unterabschnitt 3.3.1 beschrieben, können dreidimensionale Transformationsmatrizen Punkte im Raum drehen und skalieren, aber keine Translation durchführen. Hierfür wird anstatt der euklidische, die projektive Geometrie für die Beschreibung der Position eines Objektes verwendet. Die drei Koordinaten im Raum werden durch eine weitere Dimension w erweitert [A63]. Bei der Verwendung der Transformationsmatrizen für die lineare Transformation von Objekten ist der w Wert immer 1 und kann ignoriert werden. Der daraus entstehende vierdimensionale Raum wird auch Projektionsraum genannt.

Die Verwendung von homogenen Koordinaten erlaubt es auch, mithilfe der intrinsischen Matrix für die (virtuelle) Kamera eine projektive Abbildung durchzuführen. Bei einer Transformation mit einer Projektionsmatrix ändert sich auch der w Wert. Dieser kann als Skalierungsfaktor für die Darstellung des Punktes angesehen werden [A64]. Somit ist es in der Computergrafik möglich, Objekte im (virtuellen) dreidimensionalen Raum auf die zweidimensionale Bildfläche zu übertragen und darzustellen [A65].

Homogene Koordinaten werden auch beim Perspective-n-Point (PnP) Problem verwendet, um die Transformationsmatrix zu finden, die zusammen mit der bekannten Projektionsmatrix die Transformation der bekannten Punkten im dreidimensionalen Raum auf deren bekannte Punkte auf der Projektionsfläche beschreibt [A66].

3.3.3 Quaternionen

In dieser Dissertationsschrift werden in vielen der vorgestellten Arbeiten Quaternionen für die Darstellung der räumlichen Orientierung verwendet. Quaternionen stellen eine Alternative zu den bekannten Rotationsmatrizen (Transformationsmatrizen) und Euler-Winkeln dar. Verglichen mit Euler-Winkeln sind Quaternionen weniger intuitiv zu verstehen, haben jedoch deutliche Vorteile diesen gegenüber. Die beiden wichtigsten Unterschiede sind, dass es nicht zu einem Gimbal-Lock kommen kann und es bei Quaternionen keine Mehrdeutigkeiten gibt. Verglichen mit Rotationsmatrizen sind Quaternionen kompakter und benötigen weniger Rechenoperationen um beispielsweise zwei Rotationen hintereinander auszuführen.

Quaternionen bestehen aus vier Elementen, wie in Formel Gleichung (3.5) gezeigt ist. Wichtig anzumerken ist hier, dass Quaternion keine Gemeinsamkeiten mit homogene Koordinaten haben, auch wenn diese ebenfalls vier Dimensionen aufweisen. Es handelt sich im zwei unterschiedliche Konzepte.

$$q = (q_0, iq_1, jq_2, kq_3) \tag{3.5}$$

Quaternionen sind eine Erweiterung der komplexen Zahlen, bei denen zwei weitere imaginäre Zahlen hinzugefügt wurden, i, j und k sind jeweils eigenständige imaginäre Einheiten. Jede der imaginären Zahlen steht hierbei für eine Achse im dreidimensionalen Raum ($\mathbb{R}^3$). Mit Quaternionen können im $\mathbb{R}^3$ allgemein Reflexionen, Skalierung und Rotationen beschrieben werden. Vorwiegend werden jedoch Rotationen beschrieben. Hierfür müssen alle Quaternion Einheitsquaternionen sein, also die Länge eins haben:

$$|q| = \sqrt{({q_0}^2 + {q_1}^2 + {q_2}^2 + {q_3}^2)} = 1 \tag{3.6}$$

Vom Verständnis her, kann man Quaternionen als eine Abwandlung der Axis-Angle Darstellung verstehen, bei der die trigonometrischen Funktionen Sinus und Cosinus implizit enthalten sind. Dies bedeutet auch, dass bei der Beschreibung und Berechnung von Rotationen mittels Quaternionen keine trigonometrischen Funktionen notwendig sind. Bei der Axis-Angle Darstellung $(\theta, \hat{x}, \hat{y}, \hat{z})$ wird eine Rotation durch einen Vektor $(\hat{x}, \hat{y}, \hat{z})$ und einen Winkel θ beschrieben. Der Vektor gibt die Richtung an, um die gedreht wird, der Winkel die Stärke der Drehung [A67].

Eine Umwandlung von der Axis-Angle Darstellung in eine Quaternion lässt sich wie folgt durchführen:

$$q_0 = \cos\left(\frac{\theta}{2}\right) \tag{3.7}$$

$$q_1 = \hat{x} \cdot \sin\left(\frac{\theta}{2}\right) \tag{3.8}$$

$$q_2 = \hat{y} \cdot \sin\left(\frac{\theta}{2}\right) \tag{3.9}$$

$$q_3 = \hat{z} \cdot \sin\left(\frac{\theta}{2}\right) \tag{3.10}$$

Umwandlungen zwischen Rotationsmatrizen und Euler-Winkeln sind ebenso möglich.

Am häufigsten wird bei Quaternionen die Multiplikation verwendet, um mehrere Rotationen nacheinander anzuwenden. Soll beispielsweise die Rotation q um p weiter gedreht werden, dann werden q und p mithilfe des Hamilton-Produkts miteinander multipliziert: $r = q \cdot p$ [A68], [A69]. Wichtig ist hier anzumerken, dass diese Operation nicht kommutativ ist, also $p \cdot q \neq q \cdot p$ Das Hamilton Produkt für zwei Quaternionen ist in Gleichung (3.11) definiert:

$$\begin{aligned} qp = &(q_0p_0 - q_1p_1 - q_2p_2 - q_3p_3) \\ &+ (q_0p_1 + q_1p_0 + q_2p_3 - q_3p_2)\, i \\ &+ (q_0p_2 - q_1p_3 + q_2p_0 + q_3p_1)\, j \\ &+ (q_0p_3 + q_1p_2 - q_2p_1 + q_3p_0)\, k \end{aligned} \tag{3.11}$$

Weiterhin wird häufig der Unterschied zwischen zwei Quaternionen benötigt. Dieser kann über das Hamilton Produkt der Konjugierten einer Quaternion berechnet werden:

$$r = q \cdot p' \tag{3.12}$$

Die Konjugierte q' einer Quaternion q lässt sich durch die Invertierung des Vorzeichens der drei imaginären Komponenten errechnen:

$$q' = (q_0, -q_1, -q_2, -q_3) \tag{3.13}$$

Weiterhin sind auch Addition, Skalarprodukt, Kreuzprodukt und andere Operationen, wie eine Interpolation zwischen zwei Rotationen möglich. Wichtig ist hier zu beachten, dass die Quaternionen nach der Berechnung wieder auf eine Länge von ein normiert werden müssen.

3.4 Inertialsensoren

Die Daten, welche in dieser Dissertationsschrift genutzt und untersucht werden, stammen von sogenannten Inertialsensoren. Bei Inertialsensoren handelt es sich um Messeinrichtungen, die eine lineare oder radiale Bewegungen messen können und diese in digitale Signale umsetzen [A70]. Die lineare Beschleunigung wird mittels eines Beschleunigungssensors ermittelt, die Drehrate mittels eines Gyroskops. Besonders in der Unterhaltungselektronik werden oft MEMS-Sensoren verwendet, die sich durch ihre geringe Größe sowie ihren geringen Stromverbrauch und Produktionskosten auszeichnen [A71]. Bei MEMS-Sensoren werden elektrische sowie mikromechanische Strukturen zusammen auf einem Chip vereint. Die mikromechanischen Strukturen transformieren die physikalische Kraft mithilfe eines in eine analoge Spannung, welche dann durch einen Analog-Digital-Wandler in ein digitales Signal umgewandelt wird. Eine Logikeinheit bereitet die Daten auf, speichert sie in Registern und ermöglicht die Kommunikation mittels Serial Peripheral Interface (SPI) oder Inter-Integrated Circuit (I^2C). Eine Übersicht über die Struktur eines MEMS-Sensors ist in Abbildung 3.9 dargestellt.

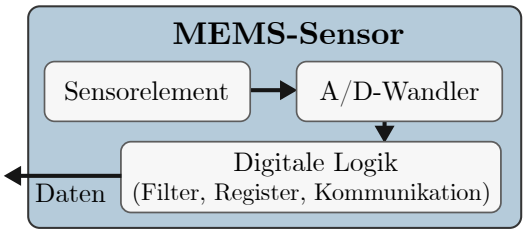


Abbildung 3.9: Aufbau eines MEMS-Sensors

Durch ihre geringe Größe haben MEMS-Sensoren ein nicht vernachlässigbares Rauschen [A72], welches sich unter anderem aus thermischen Rauschen und Funkelrauschen zusammensetzt. Weiterhin kommen noch andere Fehler, wie Bias-Fehler, Skalierungsfehler oder Ausrichtungsfehler hinzu [A73]. Es lässt sich schon anhand von Datenblättern ablesen, dass MEMS-Sensoren im Bereich der Unterhaltungselektronik oftmals deutlich ungenauer sind als Äquivalente, die in der Luft- und Raumfahrt oder der Industrie verwendet werden [A74].

Gerade beim Einsatz in eingebetteten Systemen werden Gyroskop und Beschleunigungssensors häufig zu einer sogenannten inertialen Messeinheit (IMU) zusammengefasst. Dabei bilden sie, zusammen mit einer digitalen Verarbeitungseinheit, ein geschlossenes System, welches über eine definierte Schnittstelle wie SPI oder I²C die Messwerte als digitales Signal ausgibt. Wenn Gyroskop und Beschleunigungssensor in einer IMU verbaut sind, spricht man von einer IMU mit 6 Freiheitsgraden (Degrees of Freedom, DOF).

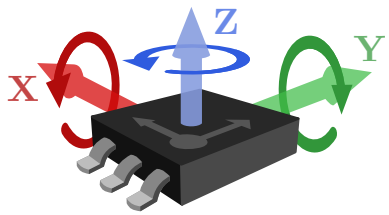


Abbildung 3.10: IMU mit jeweils drei orthogonalen Achsen zum Messen der Drehrate und Beschleunigung

Bei IMUs mit MEMS-Sensoren kommt zusätzlich zu Gyroskop und Beschleunigungssensor oftmals noch ein Magnetometer zum Einsatz. Das Magnetometer funktioniert ähnlich wie ein Kompass und misst das Magnetfeld, welches den Sensor umgibt. Mittels Magnetometer ist man in der Lage, die Himmelsrichtung anhand des gemessenen Erdmagnetfelds zu bestimmen [A75]. Obwohl das Magnetometer streng genommen keinen Bewegungssensor darstellt, wird es sehr häufig bei einer Sensordatenfusion mit Rotations- und Beschleunigungsdaten verwendet, um die räumliche Orientierung zu bestimmen. Wenn in einer IMU zusätzlich ein Magnetometer verwendet wird, spricht man von einer 9-DOF IMU.

3.5 Sensordatenverarbeitung

Bei der Verarbeitung und Verwendung von Inertialsensordaten in der Unterhaltungselektronik werden fast nie die Rohdaten der Inertialsensoren direkt verwendet, um eine Aufgabe zu erfüllen. Entweder werden die Daten eines Sensors so verarbeitet, dass sie für einen bestimmten Zweck genutzt werden können, oder es werden die Daten mehrerer Inertialsensoren miteinander kombiniert, um die Zuverlässigkeit der Daten zu erhöhen oder neue Informationen zu gewinnen.

Die Vorverarbeitung der Sensordaten sowie deren Fusion bilden einen integralen Bestandteil dieser Dissertationsschrift. Es werden sowohl Methoden vorgestellt, mittels derer man eine bereits implementierte Sensordatenfusion auf Smartphone oder Smartpens beurteilen kann, als auch Methoden, um direkt Fusionsalgorithmen in unterschiedlichen Implementierungen miteinander zu vergleichen.

3.5.1 Vorverarbeitung

Bei der Vorverarbeitung der Sensordaten werden diese so aufbereitet, dass sie von einem System genutzt werden können. Bei der Verwendung von MEMS-Sensoren sind die Signale oft verrauscht, sodass es notwendig ist, dieses Rauschen mithilfe eines Glättungsfilters zu verringern. Weiterhin kann es auch sein, dass nur bestimmte Eigenschaften der gemessenen Daten notwendig sind. Beispielsweise kann nur die Stärke einer Bewegung von einem Programm als Information benötigt werden und nicht die gemessenen Bewegungen auf allen drei Achsen. Gerade im Bereich der Gestenerkennung werden oft Features aus den Sensordaten extrahiert, um eine Geste zu erkennen [A76]. Weiterhin ist auch eine Reduktion der Datenrate durch ein Resampling ein häufig verwendeter Vorverarbeitungsschritt. Durch die Reduktion der Datenrate können der Kommunikationsaufwand und das Rauschen reduziert werden [A77], indem man die Daten mit einer hohen Frequenz aufnimmt und danach ein Downsampling durchführt [A78].

3.5.2 Sensordatenfusion

Bei der Sensordatenfusion werden die Daten mehrerer Sensoren miteinander verbunden, um entweder die Genauigkeit der Sensordaten zu erhöhen oder höherwertige Daten aus den vorhandenen Daten zu generieren. Gerade bei der Verwendung von MEMS-Sensoren sind die Sensordaten durch Messfehler oder Messrauschen überlagert. Wenn für einen bestimmten Zweck eine genauere Messung notwendig ist, können die Daten mehrerer redundanter Sensoren miteinander kombinieren werden, um eine genauere Messung zu erhalten [A79]. Es ist auch möglich, die Daten von Sensoren zu kombinieren, die zwar

dieselbe physikalische Größe messen, jedoch für die Messung unterschiedliche Prinzipien anwenden. Auf diese Weise können die Vorteile beider Sensoren vereint werden.

Weiterhin lassen sich auch die Daten von Sensoren miteinander kombinieren, die unterschiedliche physikalische Größen messen. Sehr häufig werden hierbei die Daten von Beschleunigungssensor und Gyroskop miteinander kombiniert. Durch die Kombination von Beschleunigungssensor und Gyroskop lässt sich die Lage des Sensors um Raum bestimmen. Wenn noch ein Magnetometer verwendet wird, kann man zudem noch die Richtung bestimmen. Ein solches System wird oft als Haltungs- & Kursreferenzsystem (Attitude & Heading Reference System, AHRS) bezeichnet [A80]. Für die Bestimmung der Orientierung im Raum gibt es eine Vielzahl an unterschiedlichen Fusionsfiltern, wie das Mahony-Filter [A81], das Madgwick-Filter [A82] oder das Komplementärfilter [A83]. Diese Filter werden folgend genauer beschrieben. Ein weiteres Filter, welches für die Bestimmung der Orientierung genutzt werden kann, ist das Kalman-Filter [A84]. Dieses bildet hier jedoch eine Ausnahme, da es als Filter für eine Vielzahl an Anwendungen genutzt werden kann und die Bestimmung die Orientierung nur eine Möglichkeit darstellt [A85]. Das Kalman-Filter und das Komplementärfilter nutzen die Methodik aus [A86], um die Orientierung aus Beschleunigungssensor und Magnetometer direkt als Quaternion zu bestimmen. Diese Methode hat den Vorteil, dass in keinem Berechnungsschritt rechenintensive trigonometrische Funktionen benötigt werden.

Komplementärfilter

Das Komplementärfilter ist das einfachste der in diesem Kapitel betrachteten Sensorfusionsalgorithmen. Bei diesem Filter wird ausgenutzt, dass sich die Fehler vom Gyroskop komplementär zu den Fehlern von Magnetometer und Beschleunigungssensor verhalten. Die Daten des Gyroskops zeigen ein nur geringes Rauschen. Dafür wird der Fehler bei der bestimmten Orientierung über die Zeit größer, da die gemessenen Winkelgeschwindigkeiten integriert werden müssen und sich die Sensorfehler so akkumulieren. Beim Beschleunigungssensor und Magnetometer verhält sich der Fehler gegensätzlich. Die Daten an sich sind verrauscht, auf lange Zeit aber im Mittel korrekt. Diese Eigenschaften werden beim Komplementärfilter ausgenutzt [A83]. Die Orientierung, die mithilfe des Gyroskops bestimmt wird, wird mit einem Hochpass gefiltert. Die Orientierung aus Beschleunigungssensor und Magnetometer wird mit einem Tiefpass gefiltert. Dies geschieht auf einfache Weise, indem bei einem Update der Orientierung beide Werte über eine gewichtete Addition wie in Gleichung (3.14) gezeigt kombiniert werden.

$$q_t = (1 - \alpha) \cdot q_{am} - \alpha \cdot q_{gyr} \tag{3.14}$$

Die Quaternion q_{am} für die Orientierung aus Beschleunigungssensor und Magnetometer wurde mithilfe der Methode aus [A86] bestimmt. Die Orientierung q_{gyr} wird errechnet aus der vorherigen Orientierung q_{t-1}, auf die die gemessenen Winkelgeschwindigkeiten aufaddiert werden. Dieses Filter nutzt einen einzigen Parameter $\alpha \in [01]$ für die Kalibrierung. Dieser bestimmt die Gewichtung zwischen q_{am} und q_{gyr}.

Mahony-Filter

Das Mahony-Filter ist eine Erweiterung des Komplementärfilters. Es wird eine zusätzliche Korrektur der Daten vom Beschleunigungssensor durch einen Proportional-Integral-Regler (PI-Regler) vorgenommen [A81]. Dieser Schritt wird in Gleichung (3.15) gezeigt. Für die Korrektur wird der proportionale Fehler $\vec{e}_p$ zwischen der geschätzten Orientierung und der ermittelten Orientierung aus den Daten von Beschleunigungssensor und Magnetometer bestimmt. Diese Methodik erlaubt es diesem Filter, bessere Ergebnisse zu erzielen und dennoch nicht deutlich rechenintensiver als das Komplementärfilter zu sein. Mit dem Integral-Anteil des Reglers lässt sich ein Bias der Daten des Gyroskops kompensieren.

$$\vec{\hat{\omega}} = \vec{\omega} + K_p \vec{e}_p + K_i \vec{e}_i \tag{3.15}$$

Um die korrigierte Winkelgeschwindigkeit $\vec{\hat{\omega}}$ zu erhalten, wird der proportionale Fehler $\vec{e}_p$ gewichtet mit der proportionalen Gewichtung K_p des PI-Reglers auf die gemessene Winkelgeschwindigkeit addiert. Weiterhin wird das Integral des Fehler $\vec{e}_i$ gewichtet dem integralen Gewichtungsfaktor K_i des PI-Reglers auf die Winkelgeschwindigkeit addiert Die Berechnung von $\vec{e}_i$ wird in Gleichung (3.16) beschrieben. Hierbei wird der Fehler $\vec{e}_p$ über die Zeit aufsummiert. Der Fehler wird also über die Zeit integriert. Δt ist die Zeit, die zwischen der aktuellen und der vorherigen Messung vergangen ist.

$$\vec{e}_i = \vec{e}_i + \vec{e}_p \Delta t \tag{3.16}$$

Der proportionale Fehler $\vec{e}_p$ ergibt sich aus der Summe des Kreuzproduktes aus den Gravitationsvektoren, welche jeweils aus der Orientierung von q_{acc} und q_{gyr} bestimmt werden. Die korrigierte Winkelgeschwindigkeit $\vec{\hat{\omega}}$ wird anschließend genutzt, um die neue Orientierung zu bestimmen.

Madgwick-Filter

Das Madgwick-Filter basiert, wie das Mahony-Filter, auf dem Komplementärfilter. Es wird jedoch das Gradientenverfahren genutzt, um den Fehler in der Orientierung zu bestimmen und die Daten aus Gyroskop, Beschleunigungssensor und Magnetometer miteinander zu fusionieren [A82],[A87]. Dieses Filter benutzt einen einzelnen Parameter β, welcher die Schrittweite des Gradientenverfahrens und somit die Stärke der Korrektur bestimmt. Die Korrektur mithilfe des Parameters β ist in Gleichung (3.17) gezeigt.

$$q_t = q_{t-1} + (q_{gyr} - \beta S)\Delta t \tag{3.17}$$

Der Wert q_{gyr} ist die Quaternion, welches die zuletzt gemessene Winkelgeschwindigkeit aus den Daten des Gyroskops enthält. Die Quaternion für den Korrekturschritt S wird berechnet aus der aktuellen Orientierung und den Daten von Beschleunigungssensor und Magnetometer mithilfe des Gradientenverfahrens. Das korrigierte Update wird mit dem zeitlichen Abstand zur letzten Messung dt multipliziert. Da für das Madgwick-Filter nur ein Parameter existiert, ist es nicht möglich, direkt einen Bias in den Daten des Gyroskops zu korrigieren. Für eine einfache Korrektur soll dieses Filter jedoch bessere Ergebnisse liefern als das Mahony-Filter.

Kalman-Filter

Das Kalman-Filter ist das komplexeste Filter, das in diesem Kapitel untersucht wird. Theoretisch ist das Kalman-Filter ein optimaler Fehlerschätzer, wenn es sich um ein lineares Problem mit einem Gauß-verteilten Rauschen handelt [A84], [A88]. Bei der Berechnung werden bei dem Kalman-Filter Matrizen und Vektoren verwendet.

Das Kalman-Filter ist ein rekursives Filter, das abwechselnd einen Vorhersage-Schritt und einen Korrekturschritt durchführt. Der erste Schritt ist der Vorhersage-Schritt auf Basis des aktuellen Zustands und der aktuellen Kovarianzen. Die Berechnung des Vorhersage-Schrittes ist in Gleichung (3.18) und Gleichung (3.19) dargestellt.

$$\vec{x}_t^- = \vec{x}_{t-1}\boldsymbol{A} \tag{3.18}$$

$$\boldsymbol{P}_t^- = \boldsymbol{A}\boldsymbol{P}_{t-1}\boldsymbol{A}^T + \boldsymbol{Q} \tag{3.19}$$

Der Vektor $\vec{x}$ enthält den Zustand des Systems. Im betrachteten Szenario ist dies die Quaternion für die Orientierung. Die Matrix $\boldsymbol{P}$ enthält die Kovarianzmatrix, welche die geschätzten Fehler enthält. Die Matrix $\boldsymbol{A}$ ist die Zustandsübergangsmatrix für die Vorhersage des nächsten Zustands. Im Fall eines erweiterten (nicht-linearen) Kalman-Filters wird diese Matrix bei jedem Update-Schritt neu berechnet. In der Matrix $\boldsymbol{Q}$

ist die Schätzung des Systemrauschens enthalten. Nach der Vorhersage enthält der Zustandsvektor x_t^- die Vorhersage für den nächsten Zustand, die Matrix $\boldsymbol{P}_t^-$ die Schätzung für die Fehler im nächsten Schritt.

Der zweite Schritt ist der Korrekturschritt, welcher in den Gleichungen (3.20), (3.21) und 3.22 gezeigt ist.

$$\boldsymbol{K}_t = \boldsymbol{P}_t^- \boldsymbol{H} (\boldsymbol{H} \boldsymbol{P}_t^- \boldsymbol{H}^T + \boldsymbol{R})^{-1} \tag{3.20}$$

$$\vec{x}_t = \vec{x}_t^- + \boldsymbol{K}_t (\vec{z}_t - \boldsymbol{H} \vec{x}_t^-) \tag{3.21}$$

$$\boldsymbol{P}_t = (\boldsymbol{I} - \boldsymbol{K}_t \boldsymbol{H}) \boldsymbol{P}_t^- \tag{3.22}$$

Die Matrix $\boldsymbol{K}_t$ ist der *Kalman-Gain*, welcher für die Gewichtung der Korrektur in Gleichung (3.21) und Gleichung (3.22) benötigt wird. Die Matrix $\boldsymbol{H}$ gibt vor, wie die Sensorwerte den Zustand aktualisieren. In der Matrix $\boldsymbol{R}$ ist das Messrauschen der Sensoren enthalten.

Der Zustand nach der Korrektur $\vec{x}_t$ wird in Gleichung (3.21) mithilfe des Kalman-Gains, der Matrix H und den neuen Messdaten der Sensoren $\vec{z}_t$ berechnet. Die Kovarianzmatrix $\boldsymbol{P}_t$ wird in Gleichung (3.22) berechnet. Hierbei ist $\boldsymbol{I}$ die Einheitsmatrix.

Für die Kalibrierung des Kalman-Filters können primär die Matrix $\boldsymbol{R}$ für das Messrauschen der Sensoren sowie die Matrix $\boldsymbol{Q}$ für das Prozessrauschen bzw. die Dynamik des Systems verwendet werden. Beide Matrizen enthalten im Fall der Orientierungsbestimmung mittels Quaternion 16 Werte. Dies ermöglicht mehr Kontrolle über die Einstellung des Filters, macht eine gute Kalibrierung aber auch komplexer.

Für die Untersuchungen der Algorithmen wurden bei der Implementierung des Kalman-Filters die Matrixoperationen aus der *Eigen-Matrix-Bibliothek* verwendet [A89]. Für die inverse Matrix wird eine spezielle Methode für 4×4 Matrizen aus der *MESA* Implementierung der *OpenGL Utility Libraries* verwendet [A90], [A91].

3.6 Sensorfehler und Fehler bei der Sensordatenfusion

Sowohl bei der Messung der physischen Kräfte durch die Inertialsensoren als auch bei der Verarbeitung und Fusion der Sensordaten treten Fehler auf, die die Qualität der gewonnenen Daten negativ beeinflussen. Die Art der Fehler können unterschiedlicher Natur sein und benötigen zusätzlichen Aufwand, um genauere Ergebnisse zu erzielen. Oftmals ist es das Ziel der Sensordatenfusion, die Messfehler der Sensoren zu minimieren.

Sensorfehler

Unabhängig von der verwendeten Technologie für die Messung von Bewegungen entstehen bei der Messung selbst sowie bei der Umwandlung in elektrische Signale Fehler, die sich auf die Genauigkeit der Messdaten sowie auf die Ergebnisse der Sensordatenfusion auswirken. Auch bei der Umwandlung der analogen Spannung in ein digitales Signal entstehen Fehler, beispielsweise durch die Quantisierung der Daten.

Durch die kleinen Strukturen bei der MEMS-Technologie sind die Messungen anfällig für Rauschen. Besonders das thermische Rauschen, verursacht durch die brownsche Molekularbewegung, ist dominant vertreten [A72]. Weiterhin treten durch die Fertigungsprozesse und durch kleine Ungenauigkeiten in den Strukturen weitere Fehler auf, wie etwa ein Bias oder Skalierungsfehler [A74]. Beim Bias sind die gemessenen Werte mit einem Offset versehen, der Nullpegel ist also verschoben. Beim Skalierungsfehler sind die Messwerte um einen bestimmten Faktor kleiner oder größer als die tatsächliche mechanische Kraft. Diese Fehler können beispielsweise bei der Verlötung des Sensors durch thermischen oder mechanischen Stress entstehen und müssen kalibriert werden, bevor der Sensor verwendet werden kann. Weiterhin ist die Ausprägung von Bias und Skalierungsfehler temperaturabhängig. Dies bedeutet, dass sich die Fehlercharakteristiken des Sensors gerade beim Einschalten ändern können, bis ein thermisches Gleichgewicht erreicht ist. Moderne Inertialsensoren enthalten einen Temperatursensor, mit dessen Hilfe die temperaturabhängig vermindert werden soll.

Ein weiterer Fehler bei MEMS-Sensoren ist eine Nichtlinearität der Messungen. Dies bedeutet, dass nicht jede gemessene Kraft linear in eine äquivalente elektrische Spannung umgewandelt wird [A74]. Dies ist in Abbildung 3.11 veranschaulicht. Die grüne Linie zeigt den idealen linearen Verlauf der Umwandlung einer mechanischen Kraft in eine elektrische Spannung, die lilafarbene Linie zeigt einen möglichen tatsächlichen Verlauf. Diese Nichtlinearität kann teilweise werksseitig kompensiert werden.

Zuletzt entstehen bei der Fertigung auch Ausrichtungsfehler, bei denen entweder der gesamte Sensor im Gehäuse nicht exakt ausgerichtet ist oder die einzelnen Achsen des Sensors nicht orthogonal zueinander stehen. Dadurch wirken sich Kräfte, die auf einer Achse gemessen werden, auch auf die anderen Achsen aus, was zu einer Verfälschung

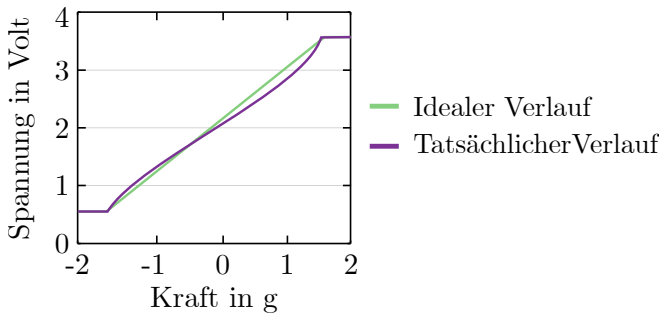
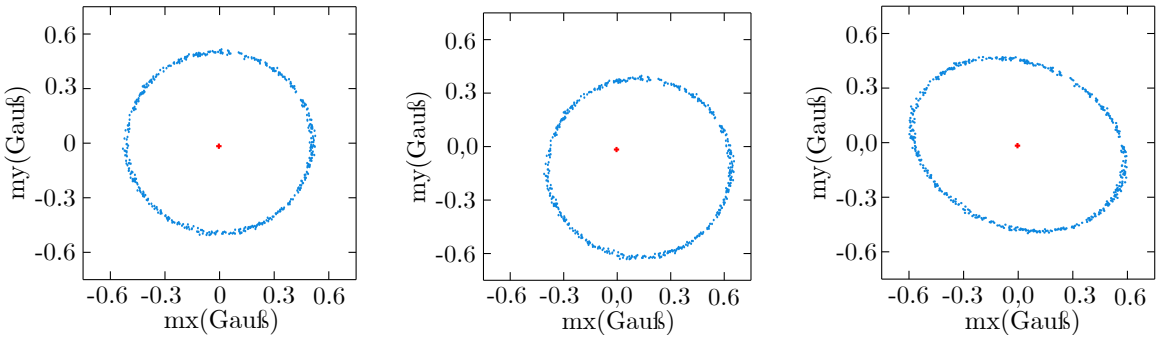


Abbildung 3.11: Linearer und nicht-linearer Verlauf

der Messung führt. Die Charakteristiken der bisher genannten Fehler sind in den Datenblättern der verwendeten Inertialsensoren genannt, meist mit typischer und maximaler Ausprägung. Anhand dieser Informationen kann man die Sensordatenfusion kalibrieren und Inertialsensoren auswählen, die sich für die geplante Verwendung eignen.

Von den in dieser Dissertationsschrift betrachteten Sensoren bildet das Magnetometer bezüglich seiner Fehler eine Ausnahme. Abgesehen von der Temperatur sind Gyroskop und Beschleunigungssensor nur wenig von externen Faktoren beeinflusst. Beim Magnetometer ist dies nicht der Fall. Es misst idealerweise das Erdmagnetfeld. Dieses wird jedoch lokal durch andere Magnetfelder überlagert und verzerrt. Diese lokalen Magnetfelder entstehen entweder durch die passive Magnetisierung ferromagnetischer Metalle durch das Erdmagnetfeld (Weicheiseneffekt) [A92] oder durch aktive elektrisch Geräte oder Permanentmagnete (Harteiseneffekt) [A92].



(a) Keine Störungen **(b)** Harteiseneffekt **(c)** Weicheiseneffekt

Abbildung 3.12: Auswirkung elektromagnetischer Störungen auf das Magnetometer, abgewandelt aus [A92]

Ohne elektromagnetische Störungen formen die Messungen des Magnetometer eine konzentrische Kugel mit dem Mittelpunkt in den Koordinaten [0,0,0]. Dies ist in Abbildung 3.12(a) dargestellt. Der Harteiseneffekt äußert sich durch eine Verschiebung der Messdaten, wie in Abbildung 3.12(b) zu sehen ist. Der Weicheiseneffekt sorgt dafür, dass die Kugel im Querschnitt zu einer Ellipse verformt wird. Dies ist in Abbildung 3.12(c)

dargestellt. Diese Fehler müssen vor einer Verwendung ermittelt und das Magnetometer entsprechend kalibriert werden. Da die elektromagnetischen Störungen orts- und zeitabhängig sind, muss eine Kalibrierung in regelmäßigen Abständen stattfinden.

Fehler bei der Sensordatenfusion

Ebenso wie bei der Aufnahme der Sensordaten können auch während der Sensordatenverarbeitung Fehler auftreten. Oftmals ist es jedoch das erklärte Ziel bei der Verarbeitung oder Fusion der Sensordaten, die Messfehler zu korrigieren oder zumindest zu verringern. Ein klassisches Beispiel für eine Verminderung der Sensorfehler ist die Rauschreduktion mittels Kalman-Filter oder die Fusion von Gyroskop und Beschleunigungssensor zu Bestimmung der räumlichen Lage. Bei der Integration der gemessenen Drehraten vom Gyroskop werden auch die Messfehler aufsummiert, was das Ergebnis über die Zeit ungenauer werden lässt. Durch die geschickte Kombination mit den Daten des Beschleunigungssensors wird dies korrigiert. Die Daten des Beschleunigungssensors sind stärker verrauscht, die Genauigkeit ist aber zeitunabhängig.

Es können durch die Fusion jedoch auch neue Fehler entstehen. Wenn beispielsweise eine Achse des Beschleunigungssensors einen großen Bias aufweist, so wird sich dieser Fehler auch auf die Bestimmung der Lage auswirken. Ein typischer Bias von $0{,}5\,m/s^2$ [A93] kann einen Winkelfehler von $2{,}5\,^\circ$ verursachen. Bei einer ungünstigen Kombination kann ein Bias auf mehreren Achsen sogar einen Fehler von bis zu $3{,}4\,^\circ$ verursachen. Weiterhin ist die Bestimmung die Lage mittels Beschleunigungssensor auf die Erdbeschleunigung angewiesen. Additive Beschleunigungen, wie sie bei der Bewegung eines Gerätes entstehen, sorgen für eine kurzfristige Verfälschung. Dies muss gerade in dynamischen Szenarien berücksichtigt werden. Die Überlagerung des Erdmagnetfelds sorgt ebenfalls für einen permanenten Fehler, wenn das Magnetometer nicht kalibriert ist.

Weiterhin gibt es auch systematische Fehler, die bei der Sensordatenverarbeitung entstehen. Wenn die Daten aus mehreren Sensorquellen stammen, ist es wahrscheinlich, dass diese technologisch bedingt einen zeitlichen Versatz aufweisen. Etwa, wenn die Daten der Inertialsensoren mit Daten aus einem Kamerasystem fusioniert werden. Dieser zeitliche Versatz kann die Qualität der Ergebnisse drastisch verringern, da die Fehler, die bei der Sensordatenfusion geschätzt werden, nicht von den Sensoren stammen, sondern durch den zeitlichen Versatz entstehen. Wie in Unterabschnitt 4.1.3 gezeigt wird, kann es vorkommen, dass die Sensoren kalibriert werden, während deren Daten genutzt werden, was zu Sprüngen in den Daten und einem Fehlverhalten der Anwendungen führen kann, die diese Daten verwenden. Zuletzt besteht die Möglichkeit, dass für die Sensordatenfusion Annahmen über dessen Verwendung getroffen werden. Treffen diese Annahmen nicht zu, kann es zu einem Fehlerverhalten der Sensordatenfusion kommen. Etwa wenn ein Gerät, das für menschliche Interaktion gedacht ist, von einem Roboter bedient wird. Mögliche Auswirkungen hiervon werden in Abschnitt 4.3 gezeigt.

3.7 Statistische Bewertungsmethoden

Bei den in dieser Dissertationsschrift beschriebenen Arbeiten werden unterschiedliche statistische Bewertungsmethoden verwendet, um die Ergebnisse der Arbeiten zu verifizieren, bestimmte Eigenschaften der Messergebnisse zu bestimmen oder Vergleiche zwischen Messungen anzustellen. Zu den verwendeten Methoden gehört die Varianz- und Messsystemanalyse, welche in Abschnitt 4.1 für die Bewertung des Messsystems verwendet wird sowie in Abschnitt 5.2 Anwendung findet, um den Einfluss unterschiedlicher Faktoren auf die Messergebnisse zu ermitteln. Weiterhin wird in Abschnitt 5.2 das Konfidenzintervall genutzt, um die Ergebnisse der Varianz- und Messsystemanalyse zu bestätigen und die Unterschiede zwischen den Messgruppen zu verdeutlichen. In Abschnitt 4.1 werden weiterhin statistische Eigenschaften wie Mittelwerte, zentrale Momente (mittlere absolute Abweichung und Varianz) und eine lineare Regressionsanalyse angewandt, um die Eigenschaften und Fehlereinflüsse in den Messdaten zu ermitteln. In diesem Kapitel werden die Eigenschaften und Ziele der verwendeten statistischen Bewertungsmethoden erläutert sowie erklärt, wie die Ergebnisse zu interpretieren sind.

Varianz- und Messsystemanalyse

Die am häufigsten in dieser Arbeit verwendete statistische Analysemethode ist die Varianzanalyse und die darauf aufbauende Messsystemanalyse für Wiederhol- und Vergleichspräzisio n (Gage Repeatability & Reproducibility, Gage R&R). Bei der Varianzanalyse wird statistisch ermittelt, ob eine oder mehrere unabhängige Variablen (Faktoren) einen statistisch signifikanten Einfluss auf eine abhängige Variable (Messgröße) haben. Für die Berechnungen werden die Mittelwerte, Residuenquadratsummen und die quadratischen Mittel der Messdaten verwendet. Aus diesen lässt sich der P-Wert bestimmen [A94], welcher für einen Hypothesentest verwendet wird. Mit dem Hypothesentest kann ermittelt werden, ob eine der unabhängigen Variablen einen statistisch signifikanten Einfluss auf die Messgröße hat. In dieser Dissertationsschrift werden zwei Arten von Varianzanalysen genutzt. In der ersten Variante, der zweifaktorielle Varianzanalyse, gibt es zwei unabhängige Variablen. Des Weiteren wird noch eine dreifaktorielle Varianzanalyse genutzt, die eine Aussage darüber treffen kann, ob die Daten durch drei unabhängige Variablen unterschieden werden können. Bei beiden Varianten ist es ebenfalls möglich, zu erkennen, ob eine Interaktion zwischen zwei oder mehr unabhängigen Variablen einen signifikanten Einfluss auf die Messwerte hat. Die Berechnungsvorschriften für die genutzten Varianzanalysen werden näher im Anhang in Abschnitt D.1 erläutert.

Messsystemanalyse

Bei der Messsystemanalyse (MSA) handelt es sich um ein Verfahren zur Beurteilung eines Messsystems, bei dem ermittelt wird, durch welche Messfaktoren bei einem Messvorgang die Streuung der Messdaten verursacht wird [A95], [A96]. Ziel ist es, zu ermitteln, ob die Messungen wiederholbar und reproduzierbar sind. Die Wiederholbarkeit sagt aus, wie gut eine Messung unter denselben Bedingungen mit gleichem Messergebnis wiederholt werden kann. Die Reproduzierbarkeit sagt aus, wie gut eine Messung für das gleiche Bauteil unter anderen Bedingungen wiederholt werden kann. Bei dieser Art der Messsystemanalyse spricht man auch von einer Messsystemanalyse für Wiederhol- und Vergleichspräzisio n.

Die Messsystemanalyse baut auf der zweifaktoriellen Varianzanalyse auf, um die Einflüsse von zwei unabhängigen Faktoren (oft prüfende Person und Bauteil) sowie den Einfluss der Interaktion beider Faktoren zu bestimmen. Zusätzlich wird mit den Ergebnissen der Varianzanalyse eine Varianzkomponentenschätzung durchgeführt. Diese schätzt, wie groß der Einfluss der einzelnen Faktoren auf die Varianz innerhalb der Messdaten ist [A97]. Der Anteil der Messvarianz, der von keinem der beiden Faktoren stammt, wird abhängig von der Signifikanz der Interaktion der Einflussfaktoren berechnet. Die Berechnungsvorschriften für die Messsystemanalyse sind im Anhang in Abschnitt D.2 beschrieben.

Konfidenzintervall

Neben der Messsystemanalyse mittels Gage R&R sowie der darin genutzten Varianzanalyse gibt es noch weitere statistische Methoden, um die Verlässlichkeit eines bestimmten Ergebnisses zu ermitteln. Eine dieser Methoden ist das Konfidenzintervall (oder Vertrauensintervall), welches bei den Untersuchungen und Tests in dieser Dissertationsschrift zum Einsatz gekommen ist. Mithilfe des Konfidenzintervalls kann aus den vorhandenen Stichproben ein Intervall bestimmt werden, in welchem der unbekannte wahre Wert wahrscheinlich liegt [A98]. Die Wahrscheinlichkeit, mit der der wahre Wert innerhalb dieses Intervalls liegt, das sogenannte Konfidenzniveau, wird vorher festgelegt. Oftmals werden 95 % oder 99 % verwendet [A99]. Die Breite des Konfidenzintervalls kann einen Aufschluss darüber geben, ob die Anzahl der genommenen Stichproben ausreichend ist, um eine Aussage treffen zu können oder ob die Stichproben eine zu große Varianz aufweisen, um eine Aussage treffen zu können. Je kleiner das Konfidenzintervall ist, desto besser kann eine Aussage getroffen werden. Die Formeln für die Berechnung des Konfidenzintervalls sind im Anhang in Abschnitt D.3 zu finden.

Lineare Regression

Die lineare Regression ist ein statistisches Verfahren, bei dem versucht wird, einen linearen Zusammenhang zwischen zwei unabhängigen Variablen zu ermitteln. Hierbei werden die Parameter a und b der Geradengleichung $y = ax + b$ für die beiden unabhängigen Variablen x und y ermittelt [A100], beispielsweise zwischen einem Fehler einer Messung und der Dauer der Messung, um einen linearen Drift festzustellen. Weiterhin wird bei der Bestimmung der linearen Regressionsanalyse noch der Korrelationskoeffizient ermittelt, mit dem eine Aussage darüber getroffen werden kann, wie gut die ermittelte Geradengleichung die Messdaten annähert. Die Berechnung der Parameter der Geradengleichung sowie des Korrelationskoeffizienten werden im Anhang in Abschnitt D.4 beschrieben.

3.8 Bewertung der Sensordaten

Im Laufe dieser Dissertationsschrift wird in mehreren der vorgestellten Arbeiten eine Bewertung der Daten von Inertialsensoren oder der Ergebnisse von Sensorfusionsalgorithmen vorgenommen. Für die Bewertung der Messdaten wird ein Vergleich mit einer parallel zur Messung aufgenommenen Referenz durchgeführt. Für die Bestimmung der Referenz wird eine Methode verwendet, die auf der Arbeit von [A4] basiert. In dieser wird beschrieben, wie anhand der Kamera eines Smartphones eine Bewertung der Inertialsensoren des Smartphones durchgeführt werden kann. Dafür wurde eine Bildverarbeitungsmethode entwickelt, mittels derer die relative Änderung der Orientierung des Smartphones als Referenz ermittelt werden kann. Für die Orientierung des Smartphones werden die Daten des virtuellen Game-Rotation-Vector Sensor verwendet. Dieser fusioniert die Daten von Gyroskop und Beschleunigungssensor, um die Orientierung des Smartphones zu bestimmen. In der beschriebenen Messmethode werden im ersten Schritt die Sensordaten und Kameradaten simultan aufgenommen und abgespeichert während die Testperson eine Bewegung mit dem Smartphone ausführt. Hierbei wird die Kamera des Smartphones permanent auf das Muster ausgerichtet, wie in Abbildung 3.13 dargestellt wird.

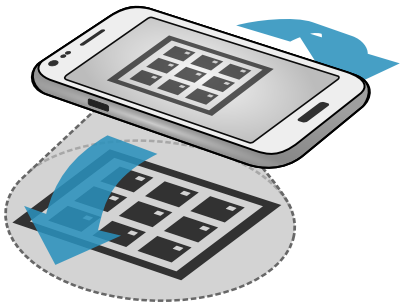


Abbildung 3.13: Veranschaulichung des Messvorgangs für die Kamera-Referenz

Im Falle eines Smartphones sind Kamera und Inertialsensoren starr miteinander verbunden und das Muster ist unbewegt. Bei den Untersuchungen in Abschnitt 5.2 wird das starr mit den Inertialsensoren verbundene Muster bewegt und die Kamera ist statisch.

Im Anschluss wird die Referenz aus den Bilddaten ermittelt und eine Kreuzkorrelation sowie anschließend der Vergleich der Daten durchgeführt. Der Ablauf dieser Messung ist in Abbildung 3.14 gezeigt.

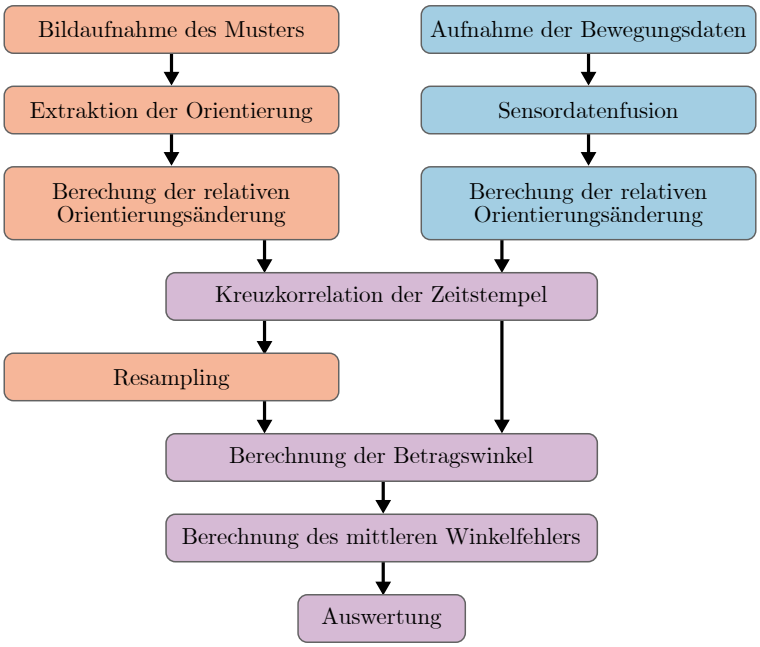


Abbildung 3.14: Ablauf einer Messung zur Bestimmung der Genauigkeit der untersuchten Sensorfusionsalgorithmen

Bestimmung der Referenz

Im ersten Schritt zur Bestimmung der Referenz-Orientierung wird das 8-Bit-Graustufenbild von der Kamera mit einem binären Schwellwert-Filter gefiltert. Anschließend werden zufällige Linien durch das Bild gezogen und nach einem abwechselndem Verhältnis zwischen den Schwarzen und weißen Sektionen von 3 : 1 gesucht (Abbildung 3.15 links), welches dem Größenverhältnis von den Quadraten zu deren Abstand zueinander entspricht. Wenn mindestens vier aufeinanderfolgende Segmente gefunden wurden, kann sich dort ein Quadrat befinden. Wenn das erste Quadrat gefunden wurde, werden von diesem aus nachfolgende Quadrate gesucht (grüne Linien in Abbildung 3.15 links). Anschließend werden die Eckpunkte mithilfe der weißen Markierung in jedem Quadrat eindeutig bestimmt. Hierfür wird die Kontur bestimmt und vereinfacht, bis sie aus nur noch vier Punkten besteht. Wenn die Vereinfachung zu einer anderen Anzahl an Punkten konvergiert, wird dieses Quadrat als Fehlerhaft angesehen und verworfen.

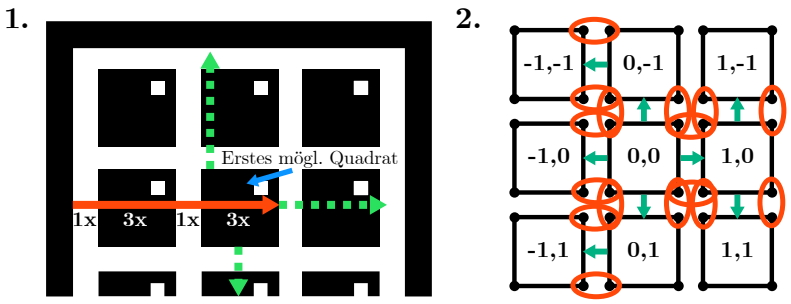


Abbildung 3.15: Schritte zum Bestimmen der Referenz, abgewandelt aus [A4]

Aus den Eckpunkten der Quadrate im Kamerabild werden die Weltkoordinaten der Punkte der Quadrate berechnet. Das mittlere Quadrat hat hierbei die Koordinaten (0,0). Wie in Abbildung 3.15 auf der rechten Seite dargestellt werden die Koordinaten der Quadrate und schließlich ihrer Eckpunkte bestimmt. Die Weltkoordinaten können auf diese Weise generiert werden, weil nur die relative Lageänderung der Kamera zu dem Muster für die Bestimmung der Orientierung notwendig ist. Nachdem die Weltkoordinaten generiert wurden, muss das Perspective-n-Point (PnP) Problem gelöst werden, um die relative Lage der Kamera aus den Weltkoordinaten und den Koordinaten der Punkte im Bild zu errechnen. Dieses Problem ist in Abbildung Abbildung 3.16 dargestellt.

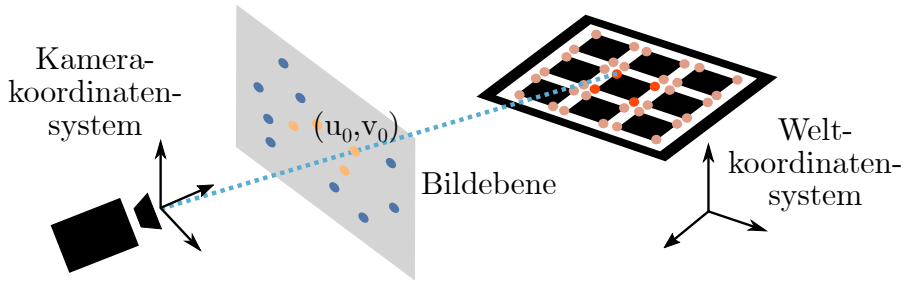


Abbildung 3.16: Beziehung zwischen Weltkoordinaten und Kamerakoordinaten, abgewandelt aus [A101]

Bei dem Perspective-n-Point Problem wird versucht, mithilfe der Punkte im Weltkoordinatensystem, den Punkten auf der Bildebene und der intrinsischen Matrix für die Kamera die räumliche Position der Kamera zu bestimmen [A66]. Grundsätzlich kann man dies als eine Umkehrung der Projektion aus der Computergrafik beschrieben. Dabei wird mittels einer Projektionsmatrix ein Punkt im 3-D Raum auf eine zweidimensionale Ebene projiziert. Es gibt unterschiedliche Methoden, um das PnP Problem zu lösen. Beispielsweise Perspective-3-Point (P3P) für drei Punkte [A102] oder RANSAC PnP [A103]. In der für die Berechnung verwendeten Software OpenCV werden vier verschiedene Lösungsmethoden angeboten [A101]. Für die Bewertung des Game-Rotation-Vectors wird lediglich die Rotation aus der ermittelten Transformationsmatrix verwendet.

Definitionen

Sowohl die Messwerte der Inertialsensoren und Sensorfusionsalgorithmen als auch die Referenzmessungen sind zeit- und wertdiskret. Die Messdaten von einem Inertialsensor S mit einer Sequenz von m Messpunkten wird definiert als:

$$S = \left(\vec{s}\left(ts_1\right), \vec{s}\left(ts_2\right), \cdots, \vec{s}\left(ts_m\right)\right) \qquad : \vec{s}\left(ts_i\right) \in \mathbb{Q}^d, i \in [1, m] \tag{3.23}$$

Hierbei ist jedes Datum $\vec{s}\left(ts_i\right)$ der Messsequenz S ein Vektor mit d Elementen, der zu einem diskreten Zeitpunkt ts_i aufgenommen wurde. Die Dimensionalität d der Vektoren ist abhängig vom genutzten Sensor. Typischerweise besteht ein Messdatum aus ein, drei oder vier Werten. Ein einzelnes Messdatum $\vec{s}\left(ts_i\right)$ kann auch beschrieben werden als:

$$\vec{s}\left(ts_i\right) = \begin{pmatrix} s\left(ts_i\right)_1 \\ s\left(ts_i\right)_2 \\ \vdots \\ s\left(ts_i\right)_d \end{pmatrix} \tag{3.24}$$

Somit ist jede der einzelnen Komponenten eines Messdatums definiert als:

$$\vec{s}\left(ts_i\right)_k \in \vec{s}\left(ts_i\right), k \in [1, d] \tag{3.25}$$

Äquivalent zu der Definition der untersuchten Sequenz S ist die ebenfalls zeit- und wertdiskrete Sequenz für die Referenz R definiert als:

$$R = \left(\vec{r}\left(tr_1\right), \vec{r}\left(tr_2\right), \cdots, \vec{r}\left(tr_n\right)\right) \qquad : \vec{r}\left(tr_j\right) \in \mathbb{Q}^d, j \in [1, n] \tag{3.26}$$

Die Sequenz R besteht hierbei aus n Messdaten $\vec{r}\left(tr_j\right)$, die zu einem diskreten Zeitpunkt tr_j aufgenommen wurden. Typischerweise haben die in R enthaltenen Vektoren dieselbe Dimensionalität d wie die Daten in S.

Ein Messdatum $\vec{r}\left(tr_j\right)$ kann auch beschrieben werden als:

$$\vec{r}\left(tr_j\right) = \begin{pmatrix} r\left(tr_j\right)_1 \\ r\left(tr_j\right)_2 \\ \vdots \\ r\left(tr_j\right)_d \end{pmatrix} \tag{3.27}$$

Somit ist jede der einzelnen Komponenten eines Messdatums definiert als:

$$\vec{r}\left(tr_j\right)_k \in \vec{r}\left(tr_j\right), k \in [1, d] \tag{3.28}$$

Vergleich von Sensordaten und Referenzdaten

Wie anfangs in diesem Abschnitt erwähnt, sollen die Daten der Inertialsensoren oder Sensorfusionsalgorithmen mit der Referenz verglichen werden, um eine Bewertung vornehmen zu können. Um die Sensordaten mit den Referenzdaten zu vergleichen, ist es notwendig, dass die Daten von S und R zuerst in ein gemeinsames Referenzsystem überführt werden. Bei den in dieser Arbeit verwendeten Messungen handelt es sich ausschließlich um Orientierungen, die als Quaternion vorliegen. Daher ist es möglich, die relative Änderung der Orientierung zu verwenden. Die Änderung ist hierbei relativ zu der ersten Orientierung, die jeweils für S und R gemessen wurde.

Die Sequenzen für relativen Orientierungen S^* wird wie folgt definiert und berechnet:

$$S^* = \left(\vec{s}^{\,*}(ts_1), \vec{s}^{\,*}(ts_2), \cdots, \vec{s}^{\,*}(ts_m)\right) \quad : \vec{s}^{\,*}(ts_i) \in \mathbb{Q}^4, i \in [1, m] \tag{3.29}$$

$$\vec{s}^{\,*}(ts_i) = \vec{s}\,(ts_i)\,\mathrm{conj}\left(\vec{s}\,(ts_1)\right) \quad : \vec{s}\,(ts_i) \in S \tag{3.30}$$

Äquivalent dazu ist die Sequenz der relativen Orientierungen der Referenz R^* definiert:

$$R^* = \left(\vec{r}^{\,*}(tr_1), \vec{r}^{\,*}(tr_2), \cdots, \vec{r}^{\,*}(tr_n)\right) \quad : \vec{r}^{\,*}(tr_j) \in \mathbb{Q}^4, j \in [1, n] \tag{3.31}$$

$$\vec{r}^{\,*}(tr_j) = \vec{r}\,(tr_j)\,\mathrm{conj}\left(\vec{r}\,(tr_1)\right) \quad : \vec{r}\,(ts_j) \in R \tag{3.32}$$

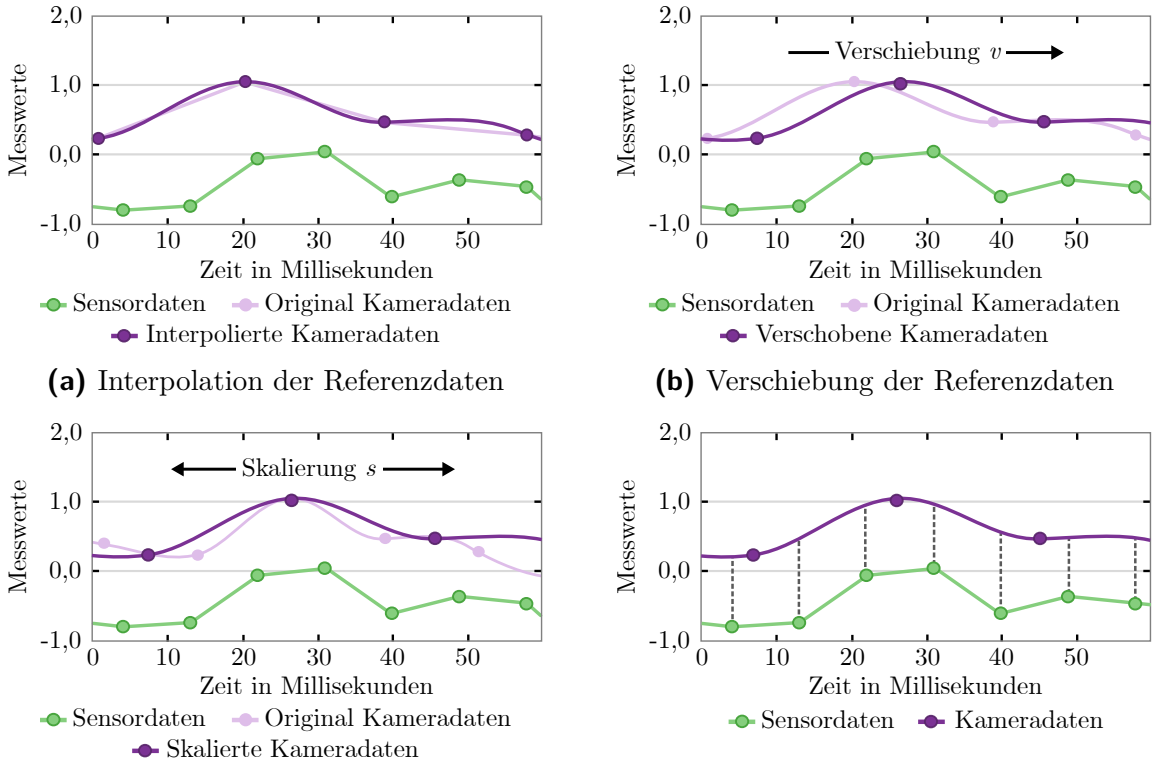
Zu erkennen ist hier, dass die Messdaten nun eine Dimensionalität von vier haben. Dies entspricht der Anzahl an Werten in einer Quaternion. Die Operation *conj* berechnet die Konjugierte der genutzten Quaternion. Bei den genutzten Einheitsquaternionen genügt es hier, die x, y und z Komponenten der Quaternion zu invertieren. Die Multiplikation von Quaternionen ist in [A104] beschrieben

Da die Daten, die in S und R gespeichert wurden aus unterschiedlichen Quellen stammen, wurden diese auch mit unterschiedlicher Frequenz aufgenommen und verwenden keine einheitliche Zeitbasis. Daher kann allgemein angenommen werden, dass:

$$j = i \implies ts_i \neq tr_r \quad : i \in [1, m], j \in [1, n] \tag{3.33}$$

Für einen Vergleich der Daten muss eine Kreuzkorrelation von S^* oder R^* und ein Resampling von R^* durchgeführt werden. Bei der Kreuzkorrelation werden die Zeitpunkte tr_j der Referenzdaten verschoben und skaliert, bis die Daten von S^* und R^* übereinander liegen. Dieser Vorgang ist in den Abbildungen 3.17(b) und 3.17(c) veranschaulicht.

Wir erwähnt, wird für die Bestimmung der Referenz ein Kamerasystem und die in [A4] beschriebene Methode für die Bestimmung der Orientierung verwendet. Daher ist es in der Regel der Fall, dass die Referenzdaten eine niedrigere Frequenz aufweisen und



(a) Interpolation der Referenzdaten

(b) Verschiebung der Referenzdaten

(c) Skalierung der Referenzdaten

(d) Upsampling der Referenzdaten

Abbildung 3.17: Kreuzkorrelation zwischen Messdaten S^* und Referenzdaten R^* sowie Upsampling der Referenzdaten

ein Upsampling von diesen stattfindet, damit keine Informationen verloren gehen. Für die Interpolation wird eine Splineinterpolation dritten Grades verwendet. Dies ist beispielhaft in Abbildung 3.17(a) dargestellt. Bei der Interpolation werden die Achsen der Quaternionen separat interpoliert, daher ist anschließend einer Normalisierung der interpolierten Referenzdaten notwendig.

Da die Referenz auf Kameradaten basiert, kann es vorkommen, dass die Zeitpunkte in den Daten von R^* aber auch von S^* nicht äquidistant sind. Beispielsweise, wenn ein Kamerabild nicht verwendet werden kann. Daher wird das Upsampling so durchgeführt, dass für jeden Zeitpunkt ts_i in der Sequenz S^* ein Wert aus den Referenzdaten für exakt diesen Zeitpunkt ermittelt wird. Dieser Schritt ist in Abbildung 3.17(d) dargestellt.

Die neu abgetasteten und durch die Kreuzkorrelation verschobenen und zeitlich skalierten Referenzdaten sind in Gleichung (3.34) definiert. Wichtig anzumerken ist, dass nun die Zeitpunkte ts_i der Sensordaten verwendet werden.

$$\hat{R}^* = \left(\vec{\hat{r}}^{\,*}(ts_o)\,,\,\vec{\hat{r}}^{\,*}(ts_{o+1})\,,\cdots,\,\vec{\hat{r}}^{\,*}(ts_p)\right) \qquad : \vec{\hat{r}}^{\,*}(ts_i) \in \mathbb{Q}^4, i \in [o,p] \tag{3.34}$$

Die neuen Werte $\vec{\hat{r}}^{\,*}$ für die Sequenz der Referenzdaten $\hat{R}^{*}$ werden wie in Gleichung (3.35) beschrieben ermittelt. Der Wert s ist hierbei der Skalierungsfaktor für die Zeitpunkte und v der Versatz.

$$\vec{\hat{r}}^{\,*}(ts_i) = \text{interpolation}\left(R^{*}, ts_i, v, s\right) \tag{3.35}$$

Die Grenzen der neu abgetasteten Referenzdaten $\hat{R}^{*}$ sind so gewählt, dass nur der überlappende Bereich zwischen Sensordaten und Referenzdaten verwendet wird. Dieser Umstand ist in Gleichung (3.36) dargestellt.

$$\forall o \in [1, m], \forall p \in [1, m] : (o < p) \tag{3.36}$$

Fehlerfunktion

Für die Berechnung des Unterschieds zwischen Sensordaten S^{*} und Referenzdaten $\hat{R}^{*}$ wird der Mittelwert der Betragswinkel aller Messpunkte im überlappenden Bereich der beiden Messsequenzen berechnet. Dies ist in Gleichung (3.37) gezeigt.

$$E = \frac{1}{p - o + 1} \sum_{i=o}^{p} e\left(ts_i\right) \tag{3.37}$$

Die Funktion für die Berechnung des Betragswinkels zwischen den einzelnen Messpunkten $\vec{\hat{r}}^{\,*}(ts_i)$ und $\vec{s}^{\,*}(ts_i)$ ist in den Gleichungen 3.38 und 3.39 beschrieben.

$$\vec{\delta}\left(ts_i\right) = \vec{\hat{r}}^{\,*}(ts_i)\,\text{conj}\left(\vec{s}^{\,*}(ts_i)\right) \tag{3.38}$$

$$e\left(ts_i\right) = \arctan\left(\vec{\delta}\left(ts_i\right)_1\right)\frac{360}{\pi} \tag{3.39}$$

Der in Gleichung (3.39) genutzte Wert $\vec{\delta}\left(ts_i\right)_1$ ist die w Komponente der Quaternionen, die für die Repräsentation der Orientierung genutzt werden.

4 Methoden und Untersuchungen auf Anwendungsebene

Der erste Kernbereich dieser Dissertationsschrift befasst sich mit Methoden zur Bewertung von Sensorsystemen auf der Anwendungsebene. Hierbei geht es speziell um die Bewertung von bereits existierenden Systemen, die ein implementiertes Sensorsubsystem enthalten. Eine Anwendung, die auf diesem System läuft, erhält die Sensorrohdaten und/oder verarbeitete Sensordaten über eine vordefinierte Schnittstelle. Gängige Beispiele sind hierbei Smartphones, die meist eine 9-DOF-IMU beinhalten, um Aufgaben wie beispielsweise die korrekte Ausrichtung des Bildschirms an die Ausrichtung des Smartphones zu erfüllen [A105] oder auch Smartwatches, die als Fitnesstracker verwenden werden können. Am Beispiel von Android-Smartphones kann man jedoch schnell erkennen, dass sich Messungen des Sensorsubsystems in ihrer Qualität stark unterscheiden. Dieser Unterschied wurde in [A106] für Android-Smartphones mithilfe einer fixen Apparatur für den Neigungs- und Rollwinkel verdeutlicht. In [A4] werden dieselben Schlüsse gezogen und am Beispiel von Indoor-Navigation veranschaulicht.

Dieser Umstand bedeutet für Nutzer:Innen, dass bestimmte Anwendungen auf einem Gerät zuverlässiger funktionieren als auf einem anderen. Für die Entwickler:Innen von Anwendungen, die Daten aus dem Sensorsubsystem nutzen, bedeutet es, dass viel Aufwand in die Implementierung und Evaluation der Anwendungen gesteckt werden muss, damit diese auf möglichst vielen Geräten zuverlässig funktionieren. Die Anzahl der möglichen Anwendungen ist hierbei vielfältig, von der Schritterkennung über Gestensteuerung für das Smartphone bis hin zu (sport-)medizinischen Anwendungen [A107].

Das Ziel der in diesem Kapitel behandelten Arbeiten ist es, Methoden zur Verfügung zu stellen, um die Qualität des Sensorsubsystems eines eingebetteten Systems auf unkomplizierte Weise zu ermitteln. Zuerst wird in Abschnitt 3.8 die in [A4] vorgestellte Methodik für die Evaluation der Genauigkeit des Sensorsubsystems von Smartphones beschrieben. In Abschnitt 4.1 und Abschnitt 4.2 werden die Probleme der Methodik beschrieben und die Erweiterungen und Verbesserungen von dieser vorgestellt. Hierbei wird auch aufgezeigt, welche Auswirkung die in Abschnitt 3.6 beschriebenen Sensorfehler auf die Sensordatenfusion haben können. Kenntnisse, die mittels der veröffentlichten MEMS Score-Anwendung gewonnen wurden, werden im Anschluss erläutert. Nachfolgend wird ein Test mit einem Roboterarm in Abschnitt 4.3 beschrieben. Schließlich wird eine Bewertungsmethode für Smartpens als Zeigegerät in Abschnitt 4.4 beschrieben.

4.1 Bewertung der Smartphone-Sensoren

Wie in Kapitel 1 erläutert, variiert die Qualität der Daten des Game-Rotation-Vectors für die Orientierung auf Smartphones stark. Dies führt zu großen Problemen bei der Entwicklung, da eine Anwendung so entwickelt werden sollte, dass sie zuverlässig auf möglichst vielen Geräten funktioniert. Aus diesem Grund wurde in [A4] eine Methodik vorgestellt, welche es ermöglicht, die Genauigkeit des Sensorsubsystems eines Android-Smartphones ohne externe Gerätschaften zu bewerten. Das Ziel dieser Arbeit ist es, allen Nutzer:Innen eines Smartphones hierfür ein einfach zu nutzendes Werkzeug zur Hand zu geben.

Diese bereits existierende Methodik wurde erweitert und verfeinert, um die Anwendung öffentlich auf Google Play bereitzustellen. Hierfür wurden drei Arbeiten durchgeführt.

- Es wurde eine Nutzerführung eingebaut, die gewährleisten soll, dass die Messungen reproduzierbar und wiederholbar sind.
- Es wurde eine vergleichende Gage R&R Analyse durchgeführt, die die Reproduzierbarkeit und Vergleichbarkeit untersuchen soll.
- Es wurde eine statistische Analyse der Messdaten hinzugefügt, um unterschiedliche Fehlerarten zu ermitteln.

Des Weiteren wurden noch kleinere Änderungen an der Verarbeitung der in Abschnitt 3.8 vorgestellten Methodik vorgenommen. Das Muster für die Bestimmung der Orientierung des Smartphones wurde abgeändert, um eine höhere Robustheit der Referenz zu erreichen. Es wurde eine weniger rechenintensive Variante für die Bestimmung der Referenz entwickelt, die es erlaubt die Referenz auch während der Messung zu bestimmen. Zuletzt wurde die lineare Interpolation der Referenzdaten durch eine Splineinterpolation ersetzt, um eine höhere Genauigkeit der errechneten Fehler zu erreichen.

Damit sollten zwei Ziele erreicht werden:

Zum einen sollte es Nutzer:Innen, aber auch Entwickler:Innen erlauben, zuverlässig die Qualität des Sensorsubsystem auf ihrem Smartphone auf einfache Weise zu testen. Diese Information ist beispielsweise sinnvoll, wenn jemand in Erfahrung bringen möchte, ob das eigene Gerät gut für Erweiterte Realität (Augmented reality, AR)-Anwendungen geeignet ist. Weiterhin werden die Messungen anonym auf einem Server der Universität Rostock gespeichert. Dies hat den Zweck, Daten über die Bandbreite der Qualität der Sensorsubsysteme von Smartphones zu sammeln. Diese Information kann Entwickler:Innen helfen, die Programme entsprechend robust zu gestalten. Weiterhin ist es dadurch möglich, in der Anwendung das eigene Gerät mit anderen zu vergleichen. Die Ergebnisse dieser Arbeit wurden 2016 auf der *Engineering Interactive Computing Systems (EICS)* Konferenz vorgestellt [B1].

4.1.1 Probleme der bestehenden Messmethode

Bei Untersuchungen der beschriebenen Bewertungsmethode für die Orientierung des Smartphones wurde schnell deutlich, dass die Ergebnisse sich trotz gleicher Ausgangsbedingungen deutlich voneinander unterschieden. Obwohl die Messungen direkt hintereinander von derselben Person auf demselben Gerät durchgeführt wurden, gab es deutliche Unterschiede in den Messungen. Diese Unterschiede sind in Abbildung 4.1 gezeigt. Diese signifikanten Messunterschiede unter gleichen Bedingungen verhindern eine zuverlässige Bewertung des Sensorsubsystems.

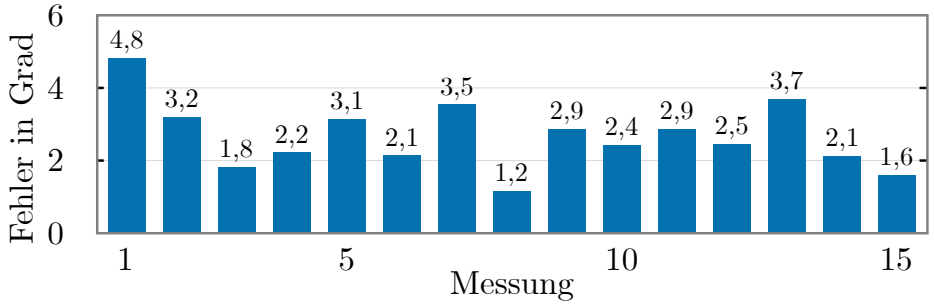


Abbildung 4.1: Messunterschiede unter gleichen Bedingungen

Weiterhin hat sich auch gezeigt, dass es signifikante Unterschiede gibt, wenn dasselbe Gerät von unterschiedlichen Personen getestet wird. Wie bei den vorherigen Messungen unter gleichen Bedingungen verhindern diese Unterschiede eine zuverlässige Bewertung der Sensordaten. Die Messunterschiede von zwei Testpersonen sind in Abbildung 4.2 dargestellt. Auch wenn eine generelle Tendenz bei einigen getesteten Geräten zu erkennen ist, sind die Ergebnisse doch zu unterschiedlich.

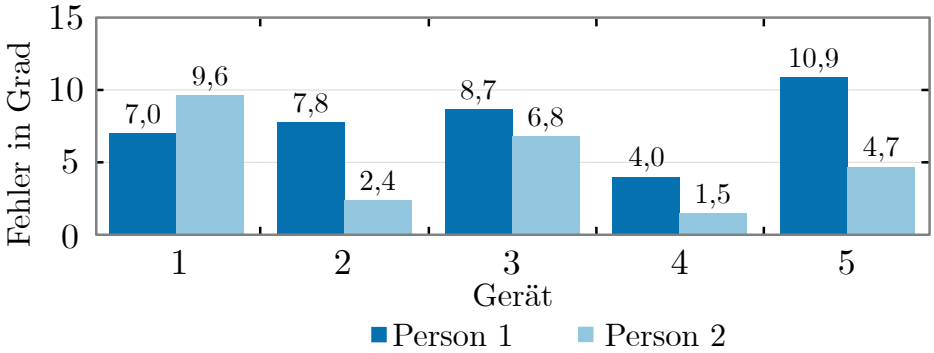


Abbildung 4.2: Messunterschiede zwischen zwei Testpersonen

Um diesen Eindruck zu untersuchen und zu verifizieren, wurde eine Gage R&R-Analyse durchgeführt, um die Wiederholbarkeit und Reproduzierbarkeit zu überprüfen. Diese Eigenschaften sind wie folgt definiert:

- **Wiederholbarkeit** ist die Möglichkeit, das Ergebnis einer Messung unter gleichen Bedingungen mit dem gleichen Ergebnis zu wiederholen. Hierbei sind Testperson, Gerät, Umwelteinflüsse und Messmethode gleich.
- **Reproduzierbarkeit** ist die Möglichkeit, das gleiche Ergebnis bei einer Messung zu erhalten, wenn diese von einer anderen Testperson bei ansonsten gleichen Bedingungen durchgeführt wird.

Das Ergebnis der Gage R&R Analyse zeigt, dass sowohl die Reproduzierbarkeit als auch die Wiederholbarkeit der Messungen nicht gewährleistet ist. Aus diesem Grund wurden weitere Untersuchungen durchgeführt, um die Ursache der Messunterschiede zu ermitteln. Drei Einflussfaktoren wurden untersucht. Hierzu zählen die Betriebstemperatur des Smartphones und die Umgebungstemperatur. Diese wurde jedoch ausgeschlossen, da die Tests direkt hintereinander durchgeführt wurden und sich weder die Betriebstemperatur der Smartphones noch die Umgebungstemperatur geändert haben. Weiterhin wurden elektromagnetische Störungen betrachtet, die jedoch ausgeschlossen werden konnten, da bei den durchgeführten Messungen kein Magnetometer verwendet wurde. Bei weiteren Untersuchungen wurde der Grund für die Messunterschiede erkannt: Die Messungen wurden zwar unter gleichen Bedingungen durchgeführt, jedoch war die Bewegung, die die Testpersonen durchführen sollten, nicht vorgegeben. Das führte dementsprechend dazu, dass die Bewegungen unterschiedlichen ausfielen. Es kann vorkommen, dass die Testperson nur sehr kleine Bewegungen durchführt, wodurch auch nur kleine Unterschiede entstehen. Andererseits ist es auch möglich, dass das Gerät zu schnell bewegt wird und dabei die Referenz nicht mehr zuverlässig ermittelt werden kann, da das Kamerabild verschwimmt. Zuletzt ist es auch möglich, dass auf einer der Drehachsen keine Bewegung durchgeführt wurde, wodurch das dynamische Verhalten dieser Achse nicht bewertet werden konnte.

4.1.2 Nutzerführung

Damit Reproduzierbarkeit und Wiederholbarkeit der Messmethode gewährleistet werden können, muss sichergestellt werden, dass bei der Durchführung der Tests immer dieselbe Bewegung durchgeführt wird. Hierfür muss während des Tests angezeigt werden, in welche Richtung das Gerät mit welcher Geschwindigkeit bewegt werden soll. Zudem muss sichergestellt werden, dass die vorgegebenen Bewegungen beim Test auch wirklich nachvollzogen werden. Hierfür musste die Anwendung um eine grafische Benutzeroberfläche erweitert werden, die eine vorgegebene Bewegungen anzeigen kann und die Testperson durch den Test leitet. Im Idealfall sollte die grafische Benutzeroberfläche bereits während des Tests eine Rückmeldung darüber geben, wie gut die Bewegungen nachvollzogen wurden.

4.1.2.1 Konzept

Die ersten Überlegungen für die grafische Vorgabe der Bewegung wurden bezüglich der visuellen Elemente getroffen. Diese mussten sowohl die aktuelle Orientierung als auch die Zielorientierung auf eine intuitiv verständliche Weise visualisieren. Ein sehr häufig für diese Aufgabe genutztes Element ist der virtuelle Horizont, der im Flugzeug dessen Lage anzeigt. Die Zielorientierung kann als Marker angezeigt werden, ähnlich wie bei einem Landeanflug. Ausgehend von diesem Konzept wurde ein Fadenkreuz entworfen, welches die aktuelle Orientierung des Gerätes sowie die Zielorientierung für alle drei Drehachsen darstellen kann. Dieser Schritt wurde gewählt, da der virtuelle Horizont nur in bestimmten Lagen des Gerätes intuitiv verständlich wäre. Der Schritt vom virtuellen Horizont zum Fadenkreuz ist in Abbildung 4.3 dargestellt. Die orangefarbenen Markierungen auf der rechten Seite zeigen die vorgegebene Orientierung für alle drei Achsen an.

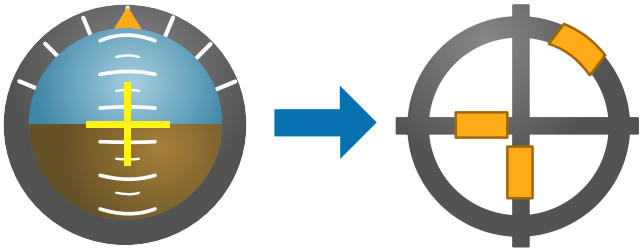


Abbildung 4.3: Umgestaltung von virtuellem Horizont zu Fadenkreuz mit Markierungen

Für die Ausgabe der aktuellen Orientierung gab es zwei Alternativen. Bei der ersten Alternative sollte die Testperson versuchen, die Marker permanent durch Rotieren des Smartphones zentriert zu halten, ähnlich wie bei einem Geschicklichkeitsspiel mit einer Murmel. Diese Variante zeigte sich bei ersten Tests aber als nicht intuitiv genug, da die senkrecht zum Display liegende Achsen so nur schwer zu erfassen ist. Die zweite

Alternative sah ein weiteres Set an Markierungen vor, welche die aktuelle Orientierung darstellen und jeweils mittig in den Markierungen für die Zielorientierung gehalten werden sollen. Die Balkenbreiten für die Zielorientierung dienen hierbei als Hinweis, wie weit maximal von der vorgegebenen Drehung abgewichen werden darf.

Zuletzt musste noch die Geschwindigkeit der vorgegebenen Bewegung festgelegt werden. Diese durfte nicht zu schnell sein, damit die Bilder der Kamera nicht verwischen, jedoch auch nicht zu langsam, da ansonsten kein Bezug zu realen Bewegungen besteht. Damit die Testperson bei der Benutzung nicht durch plötzliche Bewegungs- oder Richtungsänderungen überrascht wird und durch ruckartige Bewegungen reagiert, sollen die Bewegungen jeweils am Anfang und am Ende einer Drehung verlangsamt werden.

4.1.2.2 Implementierung

Die konzipierte grafische Oberfläche zum Anleiten der Testperson wurde dem Konzept entsprechend implementiert. Alle drei Achsen werden durch zwei Marker, die für die gewünschte Orientierung und die tatsächliche Orientierung stehen, dargestellt. Um bei der Durchführung des Tests zu unterstützen, werden die Markierungen der Achse, um die gerade aktiv gedreht werden soll, visuell durch Farbe und Form hervorgehoben. Die aktuelle Drehrichtung wird über Pfeile angezeigt. Damit die nächste Bewegung für die Testperson nicht überraschend kommt, wird an der Seite ein Pfeil mit einem Countdown angezeigt, der zeigt, wann und in welche Richtung die nächste Drehung stattfinden wird. Die Benutzeroberfläche ist in Abbildung 4.4 dargestellt.

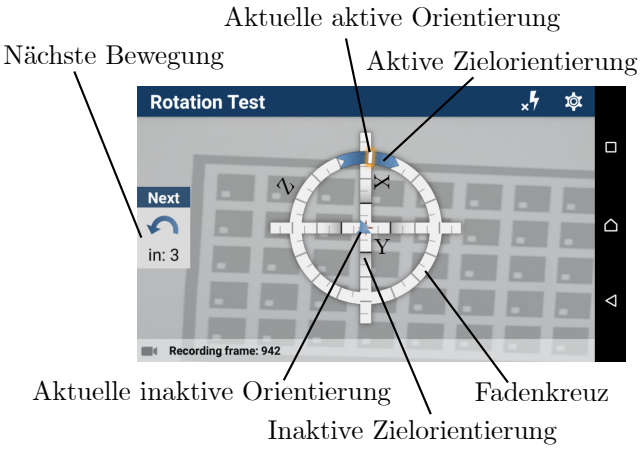
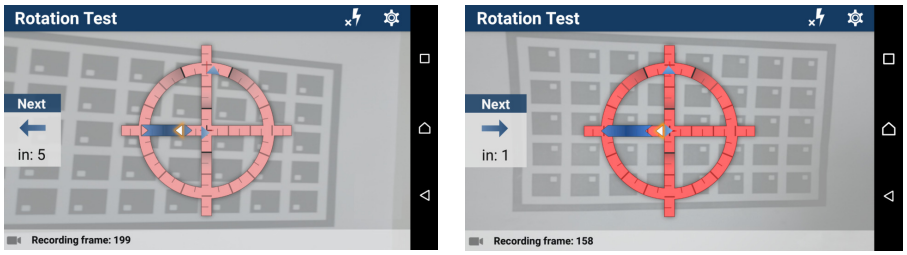


Abbildung 4.4: Benutzeroberfläche für die Drehung des Smartphones beim Sensortest

Die Rückmeldung an die Testperson über die Qualität/Güte des durchgeführten Tests geschieht auf zwei Wegen: Zum einen über die Breite der Markierungen für die gewünschte Orientierung, welche angibt, wie weit die aktuelle Orientierung abweichen darf. Die zweite Rückmeldung geschieht über eine Färbung des Fadenkreuzes. Je weiter die aktuelle

Drehung von der gewünschten Drehung abweicht, desto stärker wird das Fadenkreuz rot eingefärbt. Dies ist in Abbildung 4.5 dargestellt. Das Fadenkreuz wird rötlich eingefärbt, wenn eine Abweichung von 5 ° überschritten wird. Ab einer Abweichung von 10 ° ist das Fadenkreuz komplett rot eingefärbt und es ertönt ein Warnton. Versuche haben gezeigt, dass kleinere Grenzen für Testpersonen schnell frustrierend sein konnten. Damit eine Messung als valide angesehen wird und das Messergebnis genutzt werden kann, darf die mittlere Abweichung von der vorgegebenen Bewegung nicht mehr als 5 ° betragen. Dieser Wert hat sich als akzeptabler Mittelweg zwischen möglichst genauer Durchführung der Tests und einer einfachen Durchführbarkeit für die Testpersonen erwiesen.



Geringe Abweichung Starke Abweichung

Abbildung 4.5: Einfärbung des Fadenkreuzes bei einer Abweichung von der vorgegebenen Bewegung

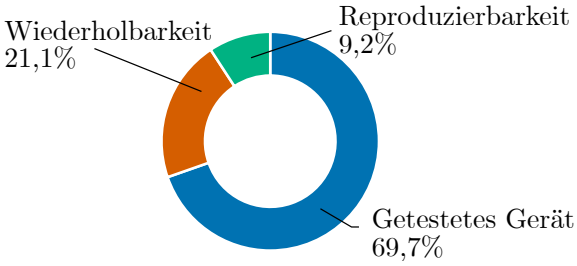
Für die Bestimmung der Orientierung während des Tests wurde eine einfachere und deutlich weniger rechenintensive Variante der Methode entwickelt, die für die Bestimmung der Referenz mittels Kamera genutzt wird. Diese einfachere Methode ist weniger genau, kann dafür jedoch während der Messung ausgeführt werden, um unabhängig von den getesteten Sensoren die aktuelle Orientierung zu bestimmen. Für die Bewertung der Sensordaten wird weiterhin die genauere Analyse benutzt.

4.1.2.3 Bewertung

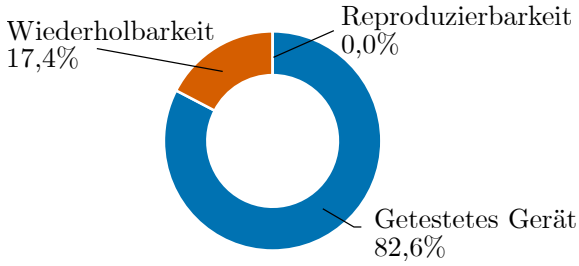
Für die Bewertung der implementierten Nutzerführung wurde eine vergleichende Gage R&R-Studie durchgeführt, welche die Wiederholbarkeit und Reproduzierbarkeit des Testverfahrens ohne und mit Nutzerführung bewertet. Hierfür wurden zwei Gage R&R Studien getrennt voneinander durchgeführt. Bei dieser Untersuchung waren die Testperson sowie das getestete Gerät die zwei unabhängigen Variablen. Die Abweichung zwischen Referenz und Sensormessung war die abhängige Testgröße. Die Berechnung der Referenz ist in Abschnitt 3.8 beschrieben.

Für die Durchführung der vergleichenden Studie wurden alle Messungen in einem gut beleuchteten Raum ohne weitere elektronische Gegenstände durchgeführt. Der Test wurde von vier Testpersonen auf fünf Smartphones (LG G3, LG Nexus 5, Motorola Moto G, Samsung Galaxy S5 und Sony XPeria Z3 Compact) durchgeführt. Die Testpersonen

sind technisch versiert, drei der Testpersonen hatten den Test vorher noch nie ausgeführt. Die vierte Person kannte den Test und galt somit auch als Kontrollinstanz. Insgesamt wurde der Test von jeder Testperson dreimal mit Nutzerführung und dreimal ohne Nutzerführung durchgeführt. Für den Test ohne Nutzerführung wurden die Testpersonen instruiert zu versuchen, alle Achsen zu testen. Beim Test mit Nutzerführung sollten die Testpersonen den Angaben auf dem Bildschirm folgen. Insgesamt wurden für beide Studien 120 Messungen durchgeführt. Die Ergebnisse der Messungen und die Auswirkungen der einzelnen Faktoren sind in den Abbildungen 4.6(a) und 4.6(b) gezeigt.



(a) Wiederholbarkeit & Reproduzierbarkeit von Messungen ohne Nutzerführung



(b) Wiederholbarkeit & Reproduzierbarkeit von Messungen mit Nutzerführung

Abbildung 4.6: Wiederholbarkeit und Reproduzierbarkeit der Sensorbewertung

An den Ergebnissen für die Messungen ohne Nutzerführung ist zu sehen, dass 9,2 % der Messunterschiede durch Probleme bei der Reproduzierbarkeit bestehen, also eine Testperson nicht in der Lage ist, die Messungen einer anderen Testperson zu wiederholen. Weiterhin werden 21,1 % der Messunterschiede durch Probleme bei der Wiederholbarkeit einer Messung unter gleichen Bedingungen verursacht. Die Analyse ergibt, dass das gemessene Ergebnis zu 69,7 % durch das getestete Gerät verursacht wird. Beides bestätigt die ursprünglich beobachteten Messergebnisse. Besonders die Probleme bei der Reproduzierbarkeit stellen ein Problem dar, weil somit das Messergebnis von der Testperson beeinflusst wird. Die Messsystemanalyse ergab hier, dass die Interaktion von Testperson und Gerät einen signifikanten Einfluss hatte ($p = 0{,}024$).

Bei den Ergebnissen mit Nutzerführung ist zu erkennen, dass die Probleme bei der Reproduzierbarkeit eliminiert werden konnten. Es gibt keine signifikante Interaktion mehr zwischen Testperson und Gerät ($p = 0{,}999$). Die Probleme bei der Wiederholbarkeit der Messungen konnten jedoch nur zu einem geringen Prozentsatz verbessert werden. Insgesamt wird das Messergebnis zu 82,6 % von dem untersuchten Gerät bestimmt.

Da vorher davon ausgegangen wurde, dass die Messunterschiede hauptsächlich von der Durchführung der Testperson verursacht werden, wurden anschließend weitere Untersuchungen zu der Ursache der noch bestehenden Varianz der Messergebnisse angestellt. Als eine der Hauptursachen wurde das Verhalten des Sensors an sich ausgemacht. Dieser liefert auch bei direkt aufeinanderfolgenden Messungen oftmals deutlich unterschiedliche Ergebnisse. Dies kann gut an den Verläufen in Abbildung 4.7 nachvollzogen werden.

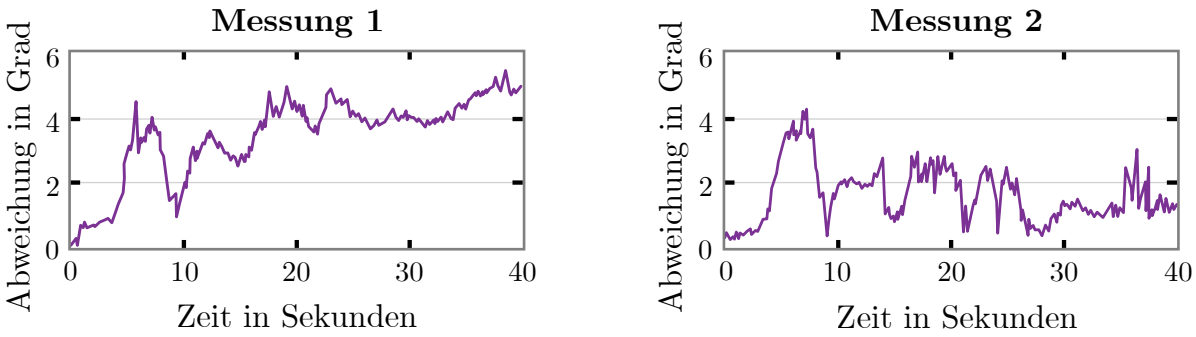


Abbildung 4.7: Unterschiedliche Ergebnisse von zwei Messungen unter gleichen Ausgangsbedingungen

Es ist klar zu erkennen, dass der Fehler bei der ersten Messung über die Zeit größer wird. Dies ist bei der zweiten Messung nicht der Fall. Der in Messung 1 zu sehende lineare Drift ist ein typisches Merkmal für einen Bias auf den Messdaten, der auf eine unzureichende Kalibrierung des Sensors zurückzuführen ist. Da der Fehler nicht direkt von einer Testperson beeinflusst werden kann, ist nicht anzunehmen, dass der Drift durch die Handhabung entstanden ist. Bei der zweiten Messung muss in der Zwischenzeit eine Kalibrierung stattgefunden haben, da hier offenbar der Bias korrigiert wurde. Da beide Messungen in kurzem Abstand hintereinander getätigt wurden, muss die Kalibrierung in der kurzen Zeit zwischen den Messungen stattgefunden haben. In weiteren Messungen konnte sogar beobachtet werden, dass eine Kalibrierung während der Messung stattfand. Dies war an einem Sprung in den Daten zu erkennen.

Sowohl der Drift der Daten, als auch eine Kalibrierung während der Nutzung sorgen für ein sehr inkonsistentes Verhalten der Sensordaten bei einigen der untersuchten Geräte. Aufgrund dieser Beobachtung und der Tatsache, dass ein Einfluss der Testperson auf das Testergebnis durch die Gage R&R Analyse ausgeschlossen werden konnte, wurde entschieden, die Konsistenz der Messgenauigkeit als ein Qualitätskriterium aufzunehmen. Um diese zu bestimmen, sollen die Messungen mindestens fünfmal durchgeführt werden, um sowohl die mittlere Messgenauigkeit, als auch deren Streuung zu ermitteln.

Besonders eine Kalibrierung während der Nutzung der Sensordaten kann das Verhalten einer Anwendung sehr negativ beeinflussen. Aber auch der Drift der Daten oder andere Fehler beeinträchtigen die Qualität der Anwendung. Um die Fehlerquellen zu ermitteln, wurde das Messverfahren um die im nächsten Kapitel beschriebenen statistischen Bewertungsmethoden erweitert.

4.1.3 Statistische Fehleranalyse

Um sowohl Anwender:Innen als auch Entwickler:Innen einen möglichst guten Überblick über die Qualität der Sensoren zu geben, wird zusätzlich zu der grafischen Nutzerführung auch eine statistische Analyse der Messungen durchgeführt. Diese Analyse soll dazu beitragen, die Gründe für die Fehler bei den Sensordaten zu ermitteln. Folgende fünf Fehlerarten, die am häufigsten anzutreffen sind, sollen hierbei ermittelt werden:

- Rekalibrierung
- Drift
- Bias
- Skalierungsfehler
- Rauschen

Für die Klassifizierung der Fehlerarten wurden insgesamt 75 Messungen auf fünf Geräten durchgeführt. Anschließend wurde ein Vergleich der Messdaten mit verschiedenen statistischen Eigenschaften der gemessenen Fehler angestellt. Hierdurch konnten die statistischen Eigenschaften ermittelt werden, die auf eine bestimmte Fehlerart hindeuten. Die in [B1] beschriebenen statistischen Auswertungsmethoden wurden überarbeitet und verbessert, um eine zuverlässigere Einschätzung vornehmen zu können. Zudem wurde die Methodik so erweitert, dass auch Skalierungsfehler und Rauschen voneinander unterscheidbar sind.

4.1.3.1 Rekalibrierung

Eine Rekalibrierung findet statt, wenn in der Firmware des Inertialsensors eine Fehlermessung durchgeführt und eine Korrektur der Daten vorgenommen wird. Der Messfehler von einem Sensor kann über die Zeit schwanken, beispielsweise durch eine Veränderung der Temperatur. Aus diesem Grund muss in der Firmware in regelmäßigen Abständen eine Kalibrierung durchgeführt werden, damit die Ergebnisse genau bleiben. Diese Korrektur kann beispielsweise mit der Hilfe anderer Sensoren oder in Ruhelage durchgeführt werden. Die genaue Funktionsweise der Kalibrierung ist oft herstellerabhängig und wird nicht öffentlich gemacht. Idealerweise sollte eine Kalibrierung nicht während der Nutzung stattfinden, da ein Sprung in den Daten oft einen sehr negativen Einfluss auf die nutzende Anwendung hat. Trotzdem wurden diese Sprünge bei einigen Smartphones während der Messung in Bewegung festgestellt. Ein Beispiel hierfür ist in Abbildung 4.8 zu sehen.

Auf der linken Seite ist die Drehung um die x-Achse dargestellt, hier findet etwa bei 32 Sekunden eine Kalibrierung statt. Die Werte springen hierbei um ca. 12 Grad und

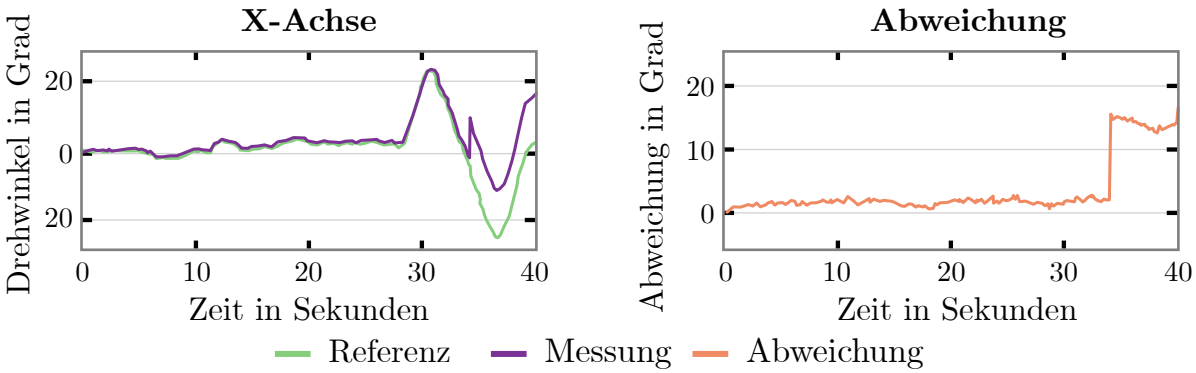


Abbildung 4.8: Visualisierung einer Kalibrierung der Sensoren während der Messung

verbleiben bei diesem Offset. Auf der rechten Seite kann man gut anhand des Fehlers erkennen, dass eine Kalibrierung stattgefunden hat.

Für die Ermittlung einer Kalibrierung wird der gleitende Median verwendet. Hierbei wird ein Ausschnitt der ermittelten Messfehler mit einer festen Länge betrachtet, der über die Messdaten geschoben wird. Nach der Berechnung aller Mediane wird die Differenz aller aufeinanderfolgenden Mediane ermittelt. Übersteigt diese Differenz ein Vielfaches der Standardabweichung der Gesamtdaten, so ist von einem Sprung in den Daten auszugehen.

$$F_{Kalibrierung} = \begin{cases} 0 & \forall\, m \in \tilde{X} : |m_t - m_{t-1}| < 50 \cdot \sigma \\ 1 & ansonsten \end{cases} \tag{4.1}$$

Für die Ermittlung, ob eine Kalibrierung während der Messung stattgefunden hat, wird die Formel von Gleichung (4.1) verwendet. Hierbei sind $\tilde{X} = \{m_1, m_2, \ldots, mn\}$ die ermittelten n Median-Werte, σ ist die Standardabweichung der Gesamtdaten.

Bei einer Kalibrierung während der Messung werden alle anderen Fehlerarten als nicht vorhanden angesehen, da die anderen statistischen Eigenschaften verfälscht sind.

4.1.3.2 Drift

Als Zweites wird der Drift ermittelt. Bei einem Drift wird der Fehler über die Zeit kontinuierlich größer. Das tritt auf, wenn der Sensor nicht ausreichend gut kalibriert ist und die Sensordaten einen Bias aufweisen. Bei der Orientierung passiert dies bei einem Bias auf den Daten des Gyroskops. Durch die Integration der Daten entsteht so ein Fehler, der linear mit der Zeit anwächst. Durch die Linearität lässt sich dieser Fehler gut mittels einer linearen Regressionsanalyse ermitteln, die in Abschnitt D.4 beschrieben wurde. Die

beiden untersuchten Variablen sind hierbei die in Abschnitt 3.8 ermittelten Betragswinkel $e\,(ts_i)$ und die Zeitpunkte ts_i. Ein Beispiel für den Drift ist in Abbildung 4.9 zu sehen.

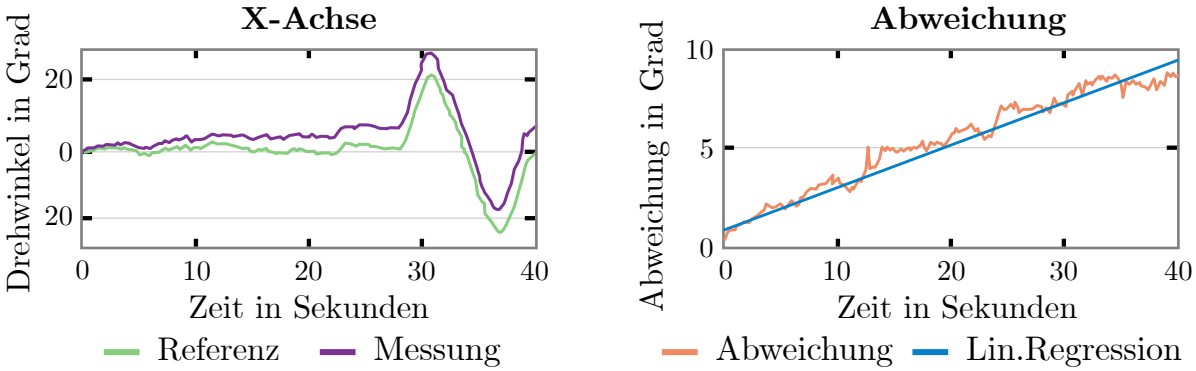


Abbildung 4.9: Drift der Sensordaten während der Messung (links), Fehler und Regressionsanalyse (rechts)

Der Faktor a_t aus der Geradengleichung der linearen Regressionsanalyse ist hierbei die Steigung und somit die Stärke des Drifts in der Messung. Mit dem Korrelationskoeffizienten r_t kann bestimmt werden, wie viel Prozent der Messunterschiede durch die Geradengleichung beschrieben werden können. Der Anteil des vom Drift verursachten Fehlers ist damit durch Gleichung (4.2) gegeben.

$$F_{Drift} = \frac{a_t r_t M}{2E} \tag{4.2}$$

Hierbei ist der Wert E die mittlere Abweichung zwischen Referenz und Sensordaten und M die Messdauer. Für den Fall, dass es einen negativen Drift gibt, wird dieser nicht berücksichtigt, da eine Verringerung des Fehlers über die Zeit unwahrscheinlich ist.

4.1.3.3 Offset

Der Offset entsteht in seltenen Fällen, wenn die Sensordaten zum Anfang der Messung wegdriften, sich aber nach einer kurzen Zeit stabilisieren. Dies kann beispielsweise passieren, wenn sich der Sensor beim Start der Messung erwärmt und dann ein thermisches Gleichgewicht erreicht. Ein Beispiel für den Offset ist in Abbildung 4.10 dargestellt.

Für die Bestimmung des Offsets wurden die aufgezeichneten Messfehler zuerst um den Drift korrigiert. Die Korrektur wurde hierbei gewichtet mit dem Korrelationskoeffizienten r_t vorgenommen. Die für die Korrektur verwendete Formel ist in Gleichung (4.3) beschrieben.

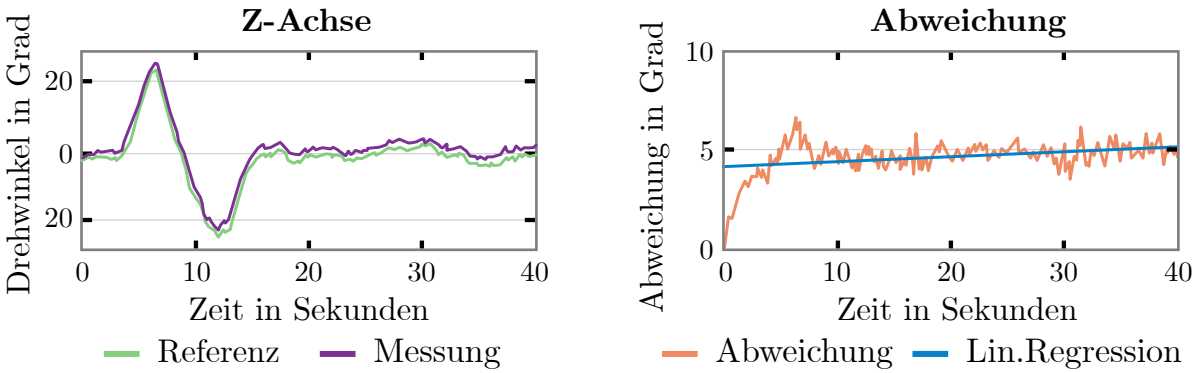


Abbildung 4.10: Offset der Sensordaten während der Messung (links), Fehler und Regressionsanalyse (rechts)

$$e'(ts_i) = e(ts_i) - r_t a(ts_i - ts_1) \tag{4.3}$$

Das Ergebnis $e'(ts_i)$ ist der korrigierte Betragswinkel für den Messwert zum Zeitpunkt ts_i.

Der Einfluss des Offsets auf den Sensorfehler lässt sich über den Mittelwert E' des korrigierten Messfehlers $e'(ts_i)$ sowie dessen Standardabweichung σ^* schätzen. Da durch den Drift zu Beginn der Messung Ausreißer in den Daten zu erwarten sind, wird der getrimmte Mittelwert verwendet, bei welchem ein festgelegter Prozentsatz der kleinsten und größten Werte nicht berücksichtigt wird. Für E' wurden die größten und kleinsten 5 % ignoriert. Damit wird der vom Drift verursachte Fehler durch Gleichung (4.4) berechnet.

$$F_{Offset} = \frac{E' - 2\sigma^*}{E} \tag{4.4}$$

Hierbei ist der Wert E wieder die mittlere Abweichung zwischen Referenz und Sensordaten, die in Gleichung (3.37) beschrieben ist. Sollte F_{Offset} kleiner als Null sein, so wird davon ausgegangen, dass es keinen Offset gibt.

4.1.3.4 Skalierungsfehler

Beim Skalierungsfehler handelt es sich um einen Fehler, bei dem die Bewegungen durch den Sensor zu einem bestimmen Prozentsatz über- oder unterschätzt werden. Hierbei gibt es einen meist linearen Zusammenhang zwischen Auslenkung der Bewegung und gemessenem Fehler. Eine typischer Skalierungsfehler ist in Abbildung 4.11 dargestellt. Hier zeigt die Auslenkung des gemessenen Drehwinkels und die Referenz eine andere Stärke der Auslenkung

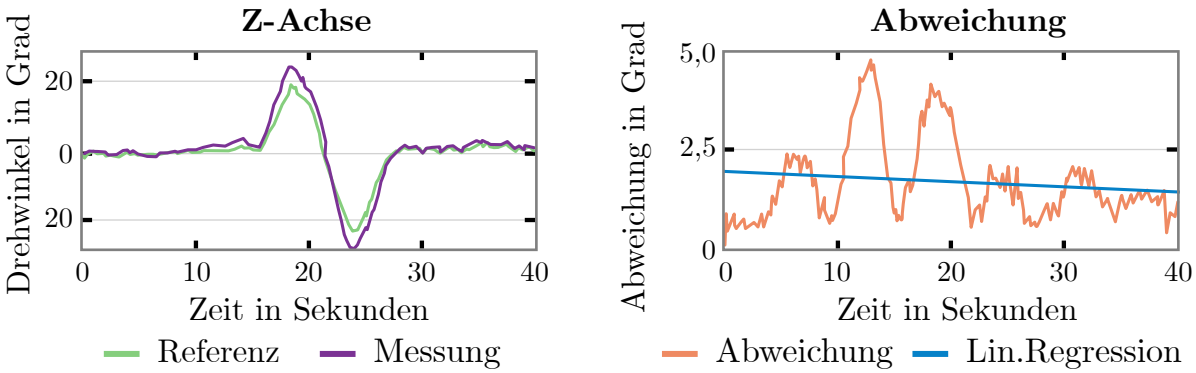


Abbildung 4.11: Skalierungsfehler der Sensordaten während der Messung (links), Fehler und Regressionsanalyse (rechts)

Mittels der um den Drift bereinigten gemessenen Abweichungen kann nun der Einfluss des Skalierungsfehlers bestimmt werden. Dies geht mittels des Korrelationskoeffizienten r_s. Für die Bestimmung von r_s wird die Korrelation zwischen den korrigierten Messfehlern $e'(ts_i)$ und der Auslenkung der durchgeführten Bewegung berechnet. Für die Stärke der Auslenkung wird der W Wert der Quaternionen für die relativen Orientierungen $\vec{s}^*(ts_i)_1$ verwendet, die in Abschnitt 3.8 in Gleichung (3.30) beschrieben werden. Der Fehleranteil durch einen Skalierungsfehler wird dann aus dem Korrelationskoeffizienten r_s und dem verbleibenden Fehleranteil nach Abzug von Offset und Bias ermittelt. Die Berechnung ist in Gleichung (4.5) zu sehen.

$$F_{Skalierung} = r_s(1 - F_{Drift} - F_{Offset}) \tag{4.5}$$

4.1.3.5 Rauschen

Nach der Bestimmung der übrigen Fehlereinflüsse lässt sich der verbleibende Fehler, der als Messrauschen angenommen wird, aus der Differenz zwischen dem Gesamtfehler und den anderen Fehlern errechnen. Dies ist in Gleichung (4.6) gezeigt.

$$F_{Rauschen} = 1 - (F_{Drift} + F_{Offset} + F_{Skalierung}) \tag{4.6}$$

Auch wenn angenommen werden muss, dass der verbleibende Fehler nicht gänzlich dem Rauschen zugeordnet werden kann, werden die verbleibenden Fehler trotzdem als Rauschen interpretiert, da eine noch tiefergehende Analyse, etwa über Nichtlinearitäten oder Achsenversatz, sehr komplex und nicht eindeutig.

4.1.3.6 Bewertung

Als Anwendungsbeispiel für die Nutzbarkeit der vorgestellten Verbesserung der Messung und der Auswertung der Fehlereigenschaften wurden beispielhaft 75 Messungen auf insgesamt fünf Geräten durchgeführt und ausgewertet, für jedes Gerät wurden 15 Messungen durchgeführt. Die gezeigten Ergebnisse sollen die Anwendung zeigen und haben nicht den Anspruch einen Überblick über den Smartphone-Markt zu geben.

Die Auswertung der Messungen hat nicht nur einen großen Unterschied bei der Qualität der Ergebnisse offenbart, sondern auch Unterschiede in den ermittelten Fehlerquellen. Die errechneten mittleren Sensorfehler in Grad sind in Tabelle 4.1 dargestellt.

Tabelle 4.1: Gemessene Fehler bei den untersuchten Smartphones

Messwert	Gerät 1	Gerät 2	Gerät 3	Gerät 4	Gerät 5
Mittelwert	4,494 °	1.716 °	2,871 °	3,220 °	0,728 °
Std. Abweichung	0,582 °	0,679 °	1,542 °	1,132 °	0,049 °

Hier sind deutliche Unterschiede bei den Geräten zu erkennen. Der Fehler von Gerät 2 ist um den Faktor 6 größer als der Fehler von Gerät 5. Beim zweiten Gerät gab es bei einer Messung eine Kalibrierung, durch die der mittlere Fehler für diese Messung bei über 90 ° lag. Diese Messung wurde in der Berechnung vom Mittelwert und Standardabweichung nicht berücksichtigt und wiederholt Die Einflüsse der einzelnen Fehlerarten sind in Abbildung 4.12 dargestellt.

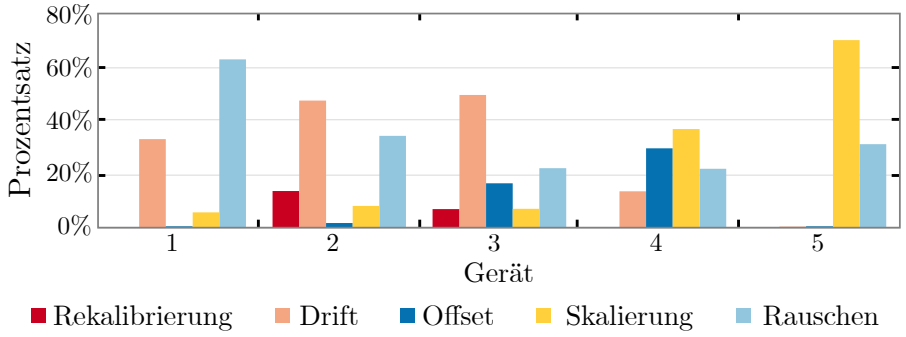


Abbildung 4.12: Verteilung der Fehlerarten bei den untersuchten Geräten

Bei den Ergebnissen sind deutliche Unterschiede zwischen den Geräten auszumachen. So zeigen die Geräte 1,2 und 3 jeweils einen starken Einfluss durch einen Drift. Dieser ist bei Gerät 5 in keiner der Messungen vorhanden. Die Geräte 2 und 3 stammen von demselben Hersteller. Bei beiden Fehlereinflüssen ist eine teilweise Überschneidung zu erkennen. Diese zeigt, dass die Fehler nicht nur abhängig vom Gerät sind, sondern auch vom Hersteller des Smartphones. Das Problem einer Kalibrierung während der Nutzung des Sensors scheint ebenfalls herstellerabhängig zu sein.

4.1.4 Weitere Tests

Neben der bisher beschriebenen Methode für die Bewertung der dynamischen Genauigkeit des Game-Rotation-Vectors wurden drei weitere Tests in der MEMS-Score-Anwendung umgesetzt. Diese Tests sollen weitere Eigenschaften des Sensorsubsystems des Smartphones bewerten. Hierbei handelt es sich um einen statischen Test des Game-Rotation-Vectors, einen statischen Test des virtuellen Sensors für die die lineare Beschleunigung sowie um eine Auswertung der die Verfügbarkeit der Sensoren und virtuellen Sensoren des Smartphones.

Statische Genauigkeit des Game-Rotation-Vectors

Der Test für die statistische Genauigkeit des Game-Rotation-Vectors bewertet denselben Sensor wie die bisher beschriebene Methode. In diesem Fall jedoch aber gemessen, wie sehr sich die gemessene Orientierung verändert, wenn das Smartphone still auf einer ebenen Oberfläche liegt. Im Idealfall sollte sich hier die Orientierung nicht ändern. (Sehr) häufig trifft dies nach Beobachtungen hingegen nicht bei Smartphones zu. Der Test dauert 30 Sekunden. Aus der gemessenen Orientierungsänderung wird die Bewertung berechnet. Aus einer geringeren Abweichung resultiert eine bessere Bewertung.

Statische Genauigkeit des Sensors für die lineare Beschleunigung

In diesem Test wird die statische Genauigkeit des virtuellen Sensors für die lineare Beschleunigung bewertet. Wie beim Test für die statische Genauigkeit des Game-Rotation-Vectors soll das Gerät für 30 Sekunden bewegungslos auf einer ebenen Oberfläche liegen. Für die Bewertung werden die Daten für die lineare Beschleunigung doppelt aufintegriert. Sobald die Positionsänderung 1 cm übersteigt, werden Geschwindigkeit und Position zurückgesetzt. Die Bewertung erfolgt anhand der mittleren Zeit, bis eine Positionsabweichung von 1 cm überschritten wird. Aus einer kurzen Zeit resultiert eine negative Bewertung.

Verfügbarkeit der (virtuellen) Sensoren

Der letzte Test bewertet, welche und wie viele der möglichen (virtuellen) Sensoren in einem Smartphone vorhanden sind. Insgesamt sind (Stand Februar 2022) inklusive der Kameras 23 Sensoren möglich. Meistens sind jedoch nicht alle möglichen Sensoren vorhanden. Neben den (virtuellen) Sensoren wird eine weitere wichtige Eigenschaft abgefragt, und zwar, ob die Inertialsensoren sogenannte *High-Fidelity*-Sensoren sind. Diese sollen einen höheren Standard bezüglich ihrer Qualität und der Latenz der Daten erfüllen. *High-Fidelity*-Sensoren sollen zudem dieselbe Basis für die Zeitstempel verwenden, wie die Kamera. Damit sind sie deutlich besser geeignet für AR- oder VR-Anwendungen.

4.1.5 Gesammelte Ergebnisse

Die Speicherung der Ergebnisse der MEMS-Score-Anwendung auf einem Server erlaubt es, einen Datensatz an Messungen aufzubauen, der einen Einblick darüber gewährt, in welchem Rahmen sich die Fehler bei der Orientierungsmessung der Smartphones bewegen. Hierbei lässt sich nicht nur eine Aussage darüber treffen, welche Fehlerarten besonders häufig auftreten, sondern auch, ob die Qualität der Sensordatenfusion oder die Häufigkeit bestimmte Fehlerarten herstellerabhängig ist. Diese Daten können dabei helfen, die für Entwicklung und Evaluation notwendigen Parameter zu bestimmen, damit die Anwendungen zuverlässig funktionieren. Wenn die Messungen von bestimmten Geräten oder Herstellern häufig Fehler aufweisen, bei denen ein korrektes Funktionieren der Anwendungen nicht möglich ist, können diese explizit als inkompatibel markiert werden. Die in diesem Abschnitt dargestellten Daten zeigen eine Zusammenfassung der gesammelten Daten mit den wichtigsten Merkmalen. Insgesamt wurde mit der MEMS-Score Anwendung 977 Messungen durchgeführt und gespeichert.

4.1.5.1 Verfügbarkeit von Sensoren und virtuellen Sensoren

Der erste grundlegende Aspekt, der bei der Betrachtung der Smartphones wichtig ist, ist die Verfügbarkeit von Inertialsensoren sowie den aus diesen abgeleiteten virtuellen Sensoren. Hierfür wurden die Daten von insgesamt 99 Geräten ausgewertet. Auf der linken Seite in Tabelle 4.2 ist aufgezeigt, wie viel Prozent der Geräte einen der in dieser Dissertationsschrift betrachteten Inertialsensoren enthalten.

Tabelle 4.2: Verfügbarkeit von Inertialsensoren und virtuellen Sensoren in Smartphones

Beschleunigungssensor	Gyroskop	Magnetometer	Rotation Vector	Game Rotation Vector	Lineare Beschleunigung	High Fidelity Sensoren
100 %	73 %	85 %	73 %	56 %	75 %	14 %

Bei 100 % der Geräte ist ein Beschleunigungssensor vorhanden. Dieser ist zwingend notwendig, um den Bildschirm entsprechend der Ausrichtung des Gerätes zu drehen. Das Magnetometer ist mit 85 % häufiger als das Gyroskop vorhanden, da es oftmals für Navigationsaufgaben benötigt wird.

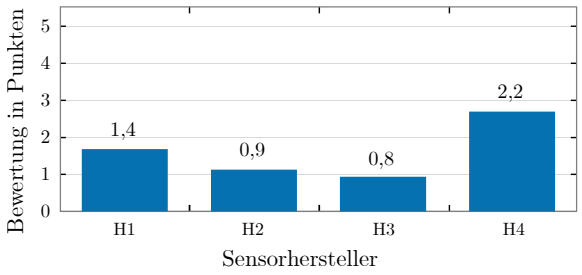
Auf der rechten Seite in Tabelle 4.2 ist die Verfügbarkeit ausgewählter virtueller Sensoren von Android-Smartphones zu sehen. Von den virtuellen Sensoren ist der Rotation-Vector, welcher die absolute Orientierung ermittelt, mit 73 % vorhanden. Dieser Wert war zu erwarten, da für die zugrundeliegende Sensordatenfusion zusätzlich zu dem Beschleunigungssensor das Gyroskop und das Magnetometer benötigt werden. Bemerkenswert ist

hier, dass der Game-Rotation-Vector, welcher nur Beschleunigungssensor und Gyroskop verwendet, deutlich seltener vorhanden ist. Ein wichtiger Faktor für die Genauigkeit der Sensordaten ist die Definition von sogenannten *High-Fidelity (Hi-Fi)-Sensoren*. Dieses Attribut sagt aus, dass die Sensoren in dem Smartphone höhere Anforderungen an Genauigkeit und Latenz erfüllen, also besser für Anwendungen geeignet sind, die eine genaue Sensordatenfusion erfordern [A108]. Zudem werden bei den High-Fidelity-Sensoren dieselben Zeitstempel wie für die Kamerabilder verwendet. Dies soll deutlich bessere Ergebnisse insbesondere für hybride Systeme mit Kamera und Sensordaten, wie etwa bei AR-Anwendungen, liefern. Von den untersuchten Smartphones besitzen jedoch nur 14 % High-Fidelity-Sensoren.

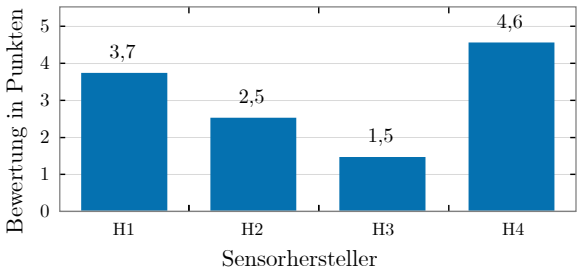
4.1.5.2 Ergebnisse der Tests

Für die Analyse der Messungen, die mit der Score-Anwendung durchgeführt worden sind, wurden die Ergebnisse nach Sensorhersteller, genutzter Sensordatenfusion sowie Hersteller des Smartphones gruppiert. Insgesamt werden vier Sensorhersteller bewertet: Bosch Sensortec, InvenSense, STMicroelectronics und HTC. Die verwendeten Sensordatenfusionen stammen von sechs Herstellern: Google, InvenSense, QTI, HTC, Samsung und AOSP. AOSP ist das *Android Open Source Project*, die Fusion stammt also ursprünglich von Google. Die MEMS-Score-Anwendung bewertet die virtuellen Sensoren mit Punkten von null bis fünf. Je höher die Punktezahl, desto besser ist das Ergebnis. Für die Orientierung des Game-Rotation-Vektors wird die mittlere Abweichung in Grad von der Referenz verwendet. Die Punkte werden beginnend von 0 Punkten für eine Abweichung von 10 ° oder mehr bis zu 5 Punkten für eine Abweichung von 1 ° oder weniger vergeben. Zwischen diesen Werten skaliert die Bewertung linear. Bei der linearen Beschleunigung wird die mittlere Zeit verwendet, nach der die Position des Gerätes nach einer doppelten Integration der Beschleunigungsdaten um mehr als 1 cm abweicht. Bei einer Zeit von 200 ms oder weniger werden 0 Punkte vergeben, bei einer Zeit von 10 s oder mehr werden 5 Punkte vergeben. Die Zwischenwerte werden über eine Wurzelfunktion bestimmt, um dem quadratischen Anstieg des Fehlers durch die doppelte Integration zu berücksichtigen. Die Ergebnisse, gruppiert nach Sensorhersteller sind in Abbildung 4.13 aufgezeigt. In Abbildung 4.14 sind die Ergebnisse, gruppiert nach Hersteller der Sensordatenfusion aufgezeigt. Zu beachten ist hier, dass nur Ergebnisse für Hersteller verwendet werden, für die mindestens 10 Messungen vorliegen. Deshalb sind nicht alle Hersteller bei allen Ergebnissen aufgelistet. Weiterhin bieten nicht alle Hersteller eine Sensordatenfusion für alle virtuellen Sensoren an.

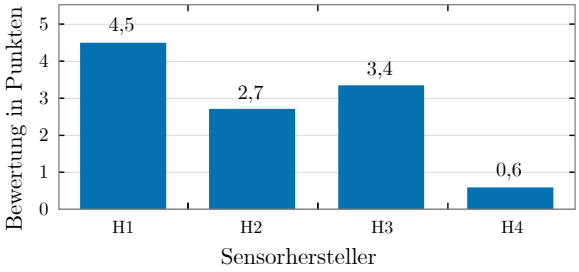
Bei den Herstellern der Sensorhardware lässt sich eine klare Tendenz sehen. Für die statischen Tests ist die relative Bewertung der Qualität des Sensorsystems für die lineare Beschleunigung sowie dem Game-Rotation-Vector gleich. Interessant ist hier, dass das Sensorsubsystem von Hersteller H4 in den statischen Tests am besten abschneidet, in dem dynamischen Test jedoch die schlechtesten Ergebnisse liefert. Dies kann damit erklärt



(a) Ergebnisse nach Hersteller für die lineare Beschleunigung



(b) Ergebnisse nach Hersteller für den Game-Rotation-Vector (statisch)

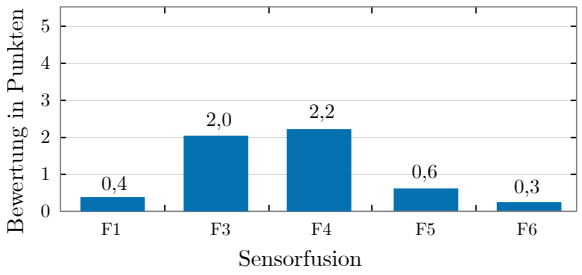


(c) Ergebnisse nach Hersteller für den Game-Rotation-Vector (dynamisch)

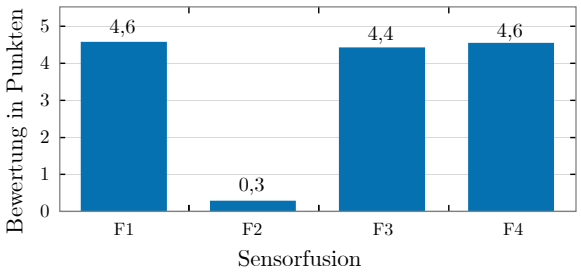
Abbildung 4.13: Ergebnisse der MEMS-Score-Anwendung gruppiert nach Hersteller

werden, dass die Sensorfusion von Hersteller H4 erkennt, wenn das Gerät in Ruhe ist und die Sensordaten entsprechend korrigiert. Erkennbar ist hier, dass die Sensoren von Hersteller H1 im Mittel die besten Ergebnisse liefern.

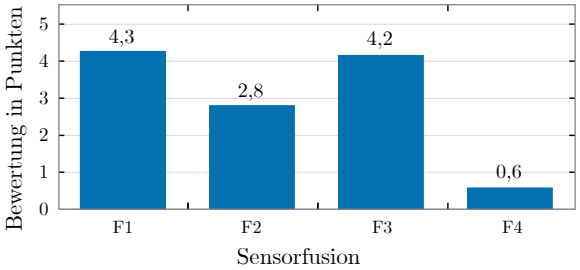
Gruppiert nach dem Hersteller der Sensordatenfusion ergibt sich ein weniger klares Bild. Zum einen ist zu erkennen, dass für die lineare Beschleunigung eine Sensordatenfusion von anderen Herstellern genutzt wird als für den Game-Rotation-Vector. Bei der Fusion F4 zeigt sich, äquivalent zu Hersteller H4, dass diese in den statischen Tests deutlich besser als in den dynamischen Tests abschneidet. Die Sensordatenfusion von F3 scheidet bei allen Tests gut ab, die Fusion von F1 nur bei der Rotation. Beim statischen Test der linearen Beschleunigung sind die Ergebnisse für F1 und F6 nahezu identisch. Dies entspricht den Erwartungen, da die Sensordatenfusionen F1 und F6 vom selben Hersteller stammen. Für die Sensordatenfusion von F5 zeigt sich durchgehend ein schlechtes Ergebnis. Dies ist wahrscheinlich der Grund dafür, dass die Fusion dieses Herstellers nur auf älteren Smartphones vorzufinden ist und auf neueren Geräten nicht mehr verwendet wird. Eine Zusammenfassung der gesammelten Daten der MEMS-Score-Anwendung ist im Anhang in den Tabellen D.4 bis D.9 zu finden.



(a) Ergebnisse für die *lineare Beschleunigung*



(b) Ergebnisse für den Game-Rotation-Vector (statisch)



(c) Ergebnisse für den Game-Rotation-Vector (dynamisch)

Abbildung 4.14: Ergebnisse der MEMS-Score-Anwendung gruppiert nach Sensorfusion

4.1.5.3 Einfluss der *High-Fidelity*-Sensoren

Die Einführung der *High-Fidelity*-Sensoren in Android 6 ist ein Schritt in die Richtung akkuraterer und besser nutzbarer Inertialsensoren für komplexe Anwendungen auf Smartphones. Sieben Jahre nach der Einführung der *High-Fidelity*-Sensoren (stand Februar 2022) sind diese jedoch bei gerade einmal 14 % der untersuchten Smartphones vorhanden. Von Interesse ist die Untersuchung, ob die virtuellen Sensoren für die lineare Beschleunigung und den Rotation-Vector qualitativ bessere Ergebnisse liefern als die normalen Sensoren.

Die Ergebnisse der Auswertung der Daten für die drei vorgestellten Tests sind in Abbildung 4.15 zu sehen. Die gemessenen Werte sind im Anhang in Tabelle D.3 zu finden.

In den Messungen ist zu erkennen, dass die Ergebnisse des Game-Rotation-Vectors für die Orientierung etwas besser ausfallen, als bei den normalen Sensoren. Bei der linearen Beschleunigung ist ein gegenteiliger Effekt zu erkennen. Hier sind die Messfehler von den *High-Fidelity*-Sensoren sogar etwas größer. Generell lässt sich erkennen, dass die Ergebnisse für die lineare Beschleunigung generell noch nicht ausreichend sind, um beispielsweise die räumliche Position für AR-Anwendungen zu bestimmen. Für die Verwendung zur Erkennung von Gesten oder einfachen Bewegungen einer Person sollten sich die Daten aber in beiden Fällen eignen.

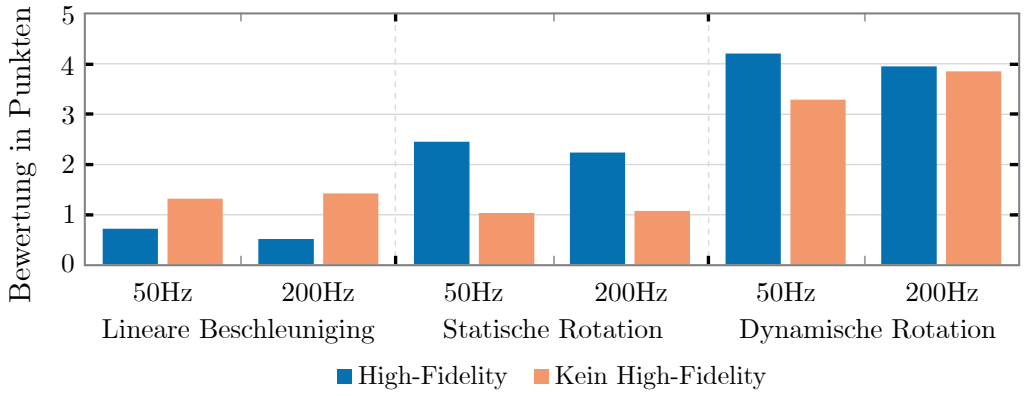


Abbildung 4.15: Vergleich von *High-Fidelity* und normalen Sensoren

4.1.6 Zusammenfassung und Bewertung

Die MEMS-Score-Anwendung für die Bewertung der Smartphone-Sensoren ist ein sinnvolles Werkzeug, das es Nutzer:Innen und Entwickler:Innen ermöglicht, die Qualität des im Smartphone verbauten Sensorsubsystems zu ermitteln. Es werden mehrere Testmethoden angeboten, um die statische Genauigkeit der virtuellen Sensoren für die lineare Beschleunigung und Rotation (Game-Rotation-Vector) sowie die dynamische Genauigkeit der Rotation zu ermitteln. Für den dynamischen Test kann ermittelt werden, von welcher Art der Fehler bei den Messdaten ist. Diese Information kann bei der Entwicklung von Smartphone-Anwendungen helfen, da sich Entwickler:Innen somit auf die am häufigsten auftretenden Fehler fokussieren können. Wenn beispielsweise eine Anwendung für die Erkennung von Gesten nicht ordnungsgemäß funktioniert, kann die Information über die Art der Fehler nützlich sein, um die Ursache für das Fehlverhalten schneller zu ermitteln und die Anwendung gegenüber diesen Fehlern robuster zu gestalten.

Die Daten über die Qualität der Sensordaten wurden in einer Datenbank gespeichert, die es erlaubt, das getestete Gerät mit anderen Geräten aus der Datenbank zu vergleichen. Dies gibt Nutzer:Innen eine einfache Möglichkeit abzuschätzen, wie gut die Sensoren des Smartphones wirklich sind. Auch welche Sensoren vorhanden sind und ob diese als *High-Fidelity*-Sensoren ausgezeichnet sind, kann einfach in Erfahrung gebracht werden.

Die vorgestellte Methode für die Bewertung der Smartphone-Sensoren zeigt aber auch einige Defizite. Aufgrund der auftretenden Unterschiede zwischen mehreren Messungen, die in Unterunterabschnitt 4.1.2.3 beschrieben sind, muss jeder Test mindestens fünfmal durchgeführt werden, um sowohl die mittlere Genauigkeit der virtuellen Sensoren als auch deren Streuung zu ermitteln. Für viele Nutzer:Innen, die schnell in Erfahrung bringen wollen, wie genau die Sensoren des Smartphones sind, scheint der damit einhergehende Aufwand zu groß zu sein. Für viele der in der Datenbank vorhandenen Smartphones sind insgesamt weniger als fünf Messungen vorhanden, weshalb diese nicht in der Gesamtwertung genutzt werden können. Besonders bei dem aufwendigen Test für die

dynamische Genauigkeit des Game-Rotation-Vectors gibt es deutlich weniger Messungen. Weiterhin sind zwar Ergebnisse von insgesamt 99 Smartphones vorhanden, aber bei aktuell 16,525 Modellen [A109], auf denen die MEMS-Score-Anwendung laufen kann, ist eine Aussage über den gesamten Smartphone-Markt nicht möglich. Hierfür müsste die Anwendung deutlich populärer sein, um einen geeignet großen Datensatz aufzubauen.

Ein weiteres Problem ist, dass der Test für die Rotation mit dem Game-Rotation-Vectors durchgeführt wird, dieser aber nur bei ca. 56 % der untersuchten Smartphones vorhanden ist. Der Grund hierfür ist, dass für den Test die relative Änderung der Orientierung untersucht wurde, beim Rotation-Vector aber die absolute Orientierung wichtiger ist. Diese Entscheidung schließt jedoch etwa die Hälfte aller Smartphones von der Bewertung aus. Eine Abhilfe hierfür wird im nächsten Abschnitt behandelt, welcher eine neue Testmethode für den Rotation-Vector vorstellt.

4.2 Bewertung des Magnetometers

Die bisher betrachtete Methode für die Bewertung der Sensordatenfusion zur Bestimmung der Orientierung eines Smartphones bewertet die Genauigkeit der relativen Änderung der Orientierung. Diese Methode eignet sich, um die dynamische Genauigkeit zu bestimmen. Was bei diesem Test jedoch nicht untersucht werden kann, ist die Korrektheit des Rotation-Vectors, der zusätzlich die Daten des Magnetometers verwendet. In den gesammelten Daten der MEMS-Score-Anwendung ist erkennbar, dass deutlich mehr Smartphones den Rotation-Vector unterstützen und der Game-Rotation-Vector nur in etwa der Hälfte der Geräte verfügbar ist. Der Rotation-Vector beschreibt die absolute Orientierung, bei der die Drehung um die zum Erdboden senkrecht stehenden z-Achse relativ zu den Polen des Erdmagnetfelds ist. Diese Drehung wird oft auch als Himmelsrichtung bezeichnet. Das Magnetometer misst für die Bestimmung der Himmelsrichtung das Erdmagnetfeld und fungiert so als digitaler Kompass. Die absolute Orientierung ist z.B. bei der Navigation zur Bestimmung der Blickrichtung oder bei der Lagebestimmung von unbemannten Luftfahrzeugen (Unmanned Aerial Vehicles, UAVs) wichtig.

Aus diesen Gründen wurde eine Testmethode entworfen, die die absolute Orientierung bewerten soll. Das Konzept und die Implementierung der Testmethode sind in Zusammenarbeit mit Studierenden in zwei Masterarbeiten entstanden [C1], [C2]. Teile dieser Arbeiten wurden 2019 auf der *IEEE Sensors Konferenz* [B2] vorgestellt und veröffentlicht. Die Tests hatten abermals zum Ziel, sowohl Nutzer:Innen als auch Entwickler:Innen ein Werkzeug an die Hand zu geben, das die Bewertung der Sensordatenfusion ohne zusätzliches und teures Equipment ermöglicht.

4.2.1 Konzept

Beim Entwurf der Messmethode mussten mehrere Faktoren beachtet werden. Zum einen werden die Daten des Magnetometers sehr stark durch elektromagnetische Störungen in der Umgebung beeinflusst, wie in Abschnitt 3.6 beschrieben ist. Dies erfordert, dass vor jedem Test sichergestellt wird, dass das Magnetometer kalibriert ist. Zudem wird eine Referenz benötigt, die zuverlässig die korrekte Orientierung relativ zum geografischen Nordpol angibt. Ein externer Kompass wäre beispielsweise ebenfalls von den elektromagnetische Störungen in der Umgebung betroffen. Hinzu kommt noch die ortsabhängige Deklination, welche den Unterschied zwischen den geografischen und magnetischen Polen beschreibt. Diese muss bei der Bestimmung der Referenz beachtet werden.

Als Methode für die Bestimmung der Referenz wurde ein kartenbasierter Ansatz ausgewählt. Hierbei wird eine Abwandlung der bewährten Methode der Peilung verwendet, die bei der Positionsbestimmung in der Schifffahrt Anwendung findet. Bei der Peilung wird die Position anhand von zwei oder mehr markanten Objekten am Horizont und

dem relativen Blickwinkel zu diesen bestimmt. Umgekehrt kann diese Methode genutzt werden, um anhand der bekannten Position zweier Objekte den Blickwinkel relativ zum geografischen Norden zwischen diesen Punkten zu bestimmen. Der Test wurde so entworfen, dass die Testperson einen markanten Punkt, beispielsweise einen Funkmast oder einen Fernsehturm, auswählt. Die Koordinaten dieses Objektes sowie die Koordinaten der Testperson können dann für die Bestimmung der Referenz genutzt werden. Die Kamera des Smartphones soll bei der Durchführung des Tests mittig auf das markante Objekt gerichtet werden.

Für die Kompensation der Deklination sowie für die Überprüfung, ob die Daten vom Magnetometer kalibriert sind, kann das World Magnetic Model (WMM) verwendet werden [A110]. Über die Koordinaten der Testperson lassen sich Deklination und erwartete Magnetfeldstärke bestimmen. Die Magnetfeldstärke kann hierbei genutzt werden, um zu ermitteln, ob das Magnetometer vom Smartphone ausreichend kalibriert ist, indem die gemessene Magnetfeldstärke mit der erwarteten Magnetfeldstärke verglichen wird.

4.2.2 Implementierung

Der Test für die absolute Orientierung wurde so implementiert, dass er in drei Phasen abläuft. Diese Phasen sind in Abbildung 4.16 dargestellt. In der ersten Phase wird eine Kalibrierung durchgeführt. Dies geschieht auf Android-Smartphones in der Regel, indem eine Figure-8 Bewegung durchgeführt wird. Nach der Durchführung der Bewegung wird geprüft, ob die gemessene Magnetfeldstärke mit der erwarteten Magnetfeldstärke übereinstimmt. Wenn dies der Fall ist, kann mit Phase 2 weiter gemacht werden. Ansonsten wird die Kalibrierung wiederholt.

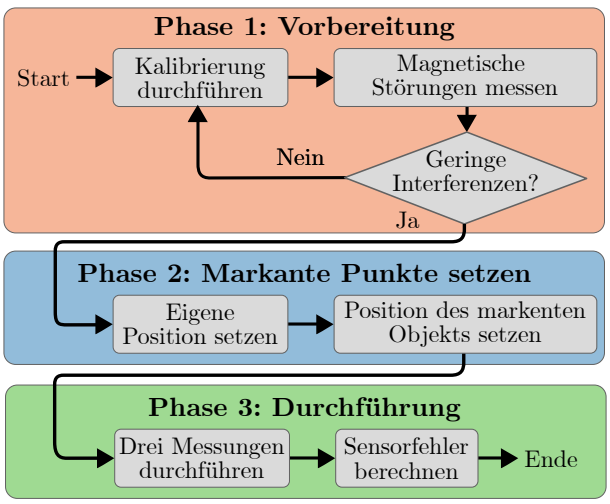


Abbildung 4.16: Ablauf des Tests für die absolute Orientierung

In Phase 2 werden die für die Durchführung des Tests notwendigen Koordinaten für die Referenz festgelegt. Hierfür steht der Testperson eine Karte zur Verfügung, auf der die Marker platziert werden können. Die Position der Testperson kann auch automatisch mithilfe des GPS ermittelt und gesetzt werden. Dieser Schritt ist auf der linkten Seite in Abbildung 4.17 zu sehen. Auf der rechten Seite wird der Zusammenhang zwischen magnetischer Peilung, echter Peilung und Sensorfehler veranschaulicht.

Abbildung 4.17: Phase 2 des Tests: Auswahl der Bezugspunkte

Sobald beide Positionen gesetzt sind, wird mithilfe des WMM und der Kartenpositionen die erwartete magnetische Peilung und echte Peilung ermittelt. In Phase 3 soll die Testperson nun drei Messungen durchführen, bei der eine auf dem Bildschirm dargestellte Markierung auf das in Phase 2 ausgewählte Objekt gerichtet werden soll. Dies ist für die ersten beiden Messungen in Abbildung 4.18 dargestellt.

Erste Messung Zweite Messung

Abbildung 4.18: Phase 3 des Tests: Messung. Links ist die erste Messung mit der Markierung in der Mitte zu sehen, rechts die zweite Messung mit der Markierung auf der linken Seite.

Es sollen drei Messungen mit versetzten Markierungen durchgeführt werden. Zum einen, um Durchführungsfehler zu erkennen, zum anderen um verbleibende Einflüsse von Hart- und Weicheiseneffekten zu erkennen, welche bei nur einer Messung nicht erkannt werden können. Während der Messung wird weiterhin die Magnetfeldstärke bewertet, um etwaige kurzfristig auftretende Magnetfeldstörungen zu erkennen, die die Messung verfälschen könnten.

4.2.3 Ergebnisse

Für die Bewertung der entwickelten Methode wurden Messungen mit zwei Geräten durchgeführt. Die Messungen fanden in einer üblichen Büroumgebung statt, um die Bewertung der Messmethode möglichst realitätsnah zu gestalten. Um nur die vorgestellte Methode zu bewerten, wurden die Magnetometer der Geräte vor den Messungen jeweils kalibriert. In Tabelle 4.3 werden beispielhaft die Ergebnisse von drei Messungen dargestellt. Die gesamten Messergebnisse für beide Messdurchläufe sind im Anhang in den Tabellen D.10 und D.11 zu finden. Diese Messungen wurden von dem Studenten Marcel Stieringer als Teil seiner Masterarbeit durchgeführt [C1].

Tabelle 4.3: Beispielhafte Ergebnisse des Magnetometer-Tests

Messung links	Messung Mitte	Messung rechts	Nutzerfehler links	Nutzerfehler rechts	Mittlere Abweichung
302,00 °	285,93 °	269,94 °	-0,95 °	-1,02 °	-4,86 °
306,91 °	289,36 °	272,60 °	0,54 °	-0,25 °	-1,20 °
312,58 °	298,80 °	278,70 °	-3,22 °	3,01 °	5,87 °

Die Kamera des Smartphones, dessen Daten in Tabelle 4.3 gezeigt sind, hatte ein Sichtfeld (Field of View, FOV) von 64,26 °. Das Sichtfeld ist hierbei der Bildwinkel, der horizontal von der Kamera erfasst werden kann. Die linke Messung soll so durchgeführt werden, dass die Markierung bei 25% der Bildbreite ist, der erwartete Winkel ist also um ¼des FOV, also 16,07 ° größer als jener der mittleren Messung. Die rechte Messung soll bei 75% der Bildbreite sein. Dementsprechend ist der erwartete Winkel um 16,07 ° kleiner. Dies lässt sich auch in Tabelle 4.3 anhand der Ergebnisse gut erkennen. Die Nutzerfehler für links und rechts ergeben sich aus der verbleibenden Abweichung zur mittleren Messung.

Aus den durchgeführten Messungen für beide Geräte ergeben sich mittlere Messfehler von -2,64 ° und -2,85 °. Beim zweiten Gerät ist die Varianz der gemessenen Abweichungen jedoch deutlich höher. Dies kann durch die verwendete Sensordatenfusion verursacht werden und ist ebenfalls ein Indikator für die Qualität des virtuellen Rotation-Vector Sensors.

Wie hier an den Beispieldaten bereits zu erkennen ist, unterscheiden sich sowohl die gemessenen Sensorfehler als auch die ermittelten Nutzungsfehler teilweise stark. Dies kann zum einen durch kurzfristig auftretende elektromagnetische Störungen verursacht werden, aber auch auf Fehler bei der Benutzung zurückzuführen sein. So kann bei der Benutzung die Drehung nicht korrekt durchgeführt worden sein, weil das Gerät nicht um die eigene Achse gedreht wurde. Weiterhin können nicht nur Nutzungsfehler die

Abweichung für die linke und rechte Messung verursachen, sondern auch eine unzureichende Kalibrierung. Mögliche Fehler bei der Benutzung sowie Möglichkeiten für die Trennung von Sensorfehlern und Nutzungsfehlern werden in den folgenden Abschnitten erläutert.

4.2.4 Bewertung und Abschwächung von Nutzungsfehlern

Die vorgestellte Messmethode für die Bewertung der absoluten Orientierung setzt voraus, dass die Messungen genau und fehlerfrei durchgeführt werden. Es ist jedoch zu erwarten, dass Nutzer:Innen bei der Durchführung der Messungen Fehler unterlaufen. Für eine exakte Messung müsste die Drehachse bei den Messungen durch den Mittelpunkt der Kamera verlaufen. Dies wird höchstwahrscheinlich nur selten passieren, da das Smartphone meist um seinen Mittelpunkt gedreht wird. Es kann auch sein, dass das Smartphone seitlich bewegt wird, anstatt es zu drehen. Eine Verschiebung der Position der Kamera verändert hierbei auch den gemessenen Winkel und verursacht eine Abweichung bei der Messung. Die Auswirkungen dieses Fehlers und wie diese minimiert werden können, wird folgend erklärt.

Off-Axis Rotation

Eine Drehung, dessen Rotationsachse nicht in der Kamera zentriert ist, eine sogenannte Off-Axis Rotation, verursacht einen Fehler bei der Messung, der das Ergebnis beeinflusst. Dieser Fehler ist mit hoher Wahrscheinlichkeit zu erwarten. Entweder wird das Smartphone um das eigene Zentrum gedreht oder es wird am ausgestreckten Arm gehalten und die messende Person dreht sich um ihre eigene Achse. Der hierdurch entstehende Fehler ist in Abbildung 4.19 dargestellt. Der Fehler, der hierdurch entsteht, kann durch die Formel in Gleichung (4.7) errechnet werden.

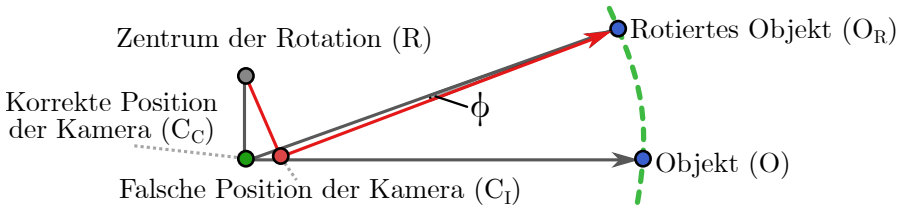


Abbildung 4.19: Auswirkung einer Drehung um das falsche Rotationszentrum

$$\phi = \arccos \frac{\vec{r}_{CR} \cdot \vec{r}_{IR}}{|\vec{r}_{CR}||\vec{r}_{IC}|} \tag{4.7}$$

Der Einfluss des fehlerhaften Rotationszentrums verursacht eine Parallaxe des Zielobjektes und hängt von dem Verhältnis zwischen der Distanz zum anvisierten Objekt und der Distanz zwischen dem tatsächlichen und dem korrekten Rotationszentrum ab. Für die Berechnung wurde ein Winkel von ±15 ° für die linke und rechte Messung angenommen. Wenn um das Zentrum des Smartphones gedreht wird und die Distanz von Kamera zum Drehpunkt 4 cm beträgt und das anvisierte Objekt 10 m entfernt ist, ergibt sich ein Fehler von 0,07 °. Dieser Fehler ist bei einer Messung vertretbar, da die erwarteten Fehler der Orientierung deutlich größer sind. Wenn die Testperson sich um die eigene Achse dreht und die Distanz zwischen Kamera und Drehpunkt 0,5 m beträgt, ist der Fehler deutlich größer. Bei einer Distanz zum anvisierten Objekt von 10 m beträgt der Fehler 0,73 °. Bei einer Distanz von 100 m liegt der Fehler nur noch bei 0,073 °, bei 1000 m bei 0,0073 °. Damit der Fehler vernachlässigbar klein ist, sollte die Distanz zwischen der testenden Person und dem anvisierten Objekt mindestens 100 m betragen.

Seitliche Bewegung des Smartphones

Eine seitliche Bewegung des Smartphones führt auch zu einer Verfälschung der Messergebnisse. Eine seitliche Verschiebung des Gerätes verursacht eine Parallaxe des anvisierten Objektes, wodurch sich dessen Position im Kamerabild ändert. Wenn die Kamera weit genug seitlich bewegt wird, könnte so das Objekt anvisiert werden, ohne die Kamera zu drehen. Der durch eine seitliche Verschiebung verursachte Fehler lässt sich anhand der Formel von Gleichung (4.8) ermitteln.

$$\Delta x = \cos(90° - \alpha)|\vec{r}_{CO}| \tag{4.8}$$

Der Winkel α wird wieder mit 15 ° angenommen. Die Distanz zwischen Kamera und anvisierten Objekt ist mit $|\vec{r}_{CO}|$ angegeben. Bei einer Distanz von 10 m zum anvisierten Objekt muss die Kamera dementsprechend 2,59 m seitlich bewegt werden. Bei einer Distanz von 100 m beträgt diese Strecke bereits 25,9 m. Eine solche Verschiebung kann mittels GPS leicht ermittelt werden, um den Test gegebenenfalls abzubrechen.

4.2.5 Mögliche Verbesserungen durch Bildanalyse

Wie bereits bei der Beschreibung des Testablaufes erläutert, werden drei Messungen durchgeführt, um die Messungen genauer zu gestalten und Fehler bei der Benutzung, speziell beim Anvisieren des Objektes, zu erkennen. Hierfür wird die Differenz zischen erwarteter und gemessener Orientierung für die linke und rechte Messung bestimmt. Diese Differenz kann jedoch nicht nur durch Fehler bei der Benutzung entstehen, sondern auch durch eine fehlende oder unzureichende Kalibrierung des Magnetometers. Die Auswirkungen einer unzureichende Kalibrierung sind in Abbildung 4.20 dargestellt.

Der durch eine unzureichende Kalibrierung verursachte Fehler ist abhängig vom Drehwinkel und äußert sich auf dieselbe Weise wie der Bias des Beschleunigungssensors, der in Abschnitt 3.6 beschrieben ist. Bei einem Harteiseneffekt von 0,2 Gauß und einem Winkel von 15 ° ergibt sich bereits eine Abweichung von 9,34 °. Hier ist leicht zu erkennen, dass in so einem Fall der Fehler bei der Benutzung nicht von einem echten Sensorfehler unterschieden werden kann. Eine Unterscheidung zwischen Nutzungsfehler und Sensorfehler muss hier über eine Erweiterung der Testmethode geschehen.

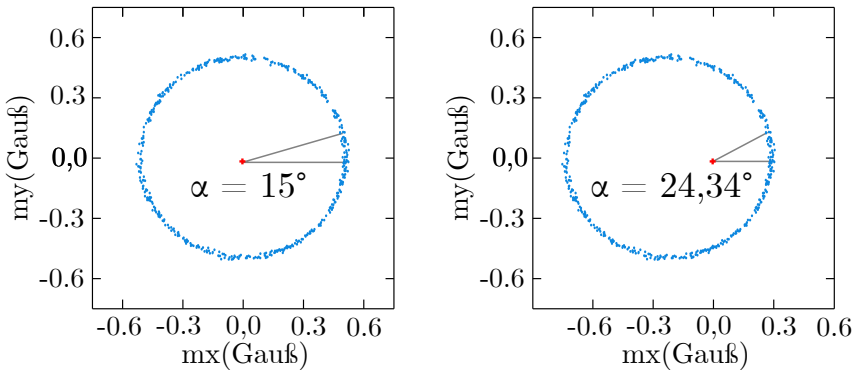
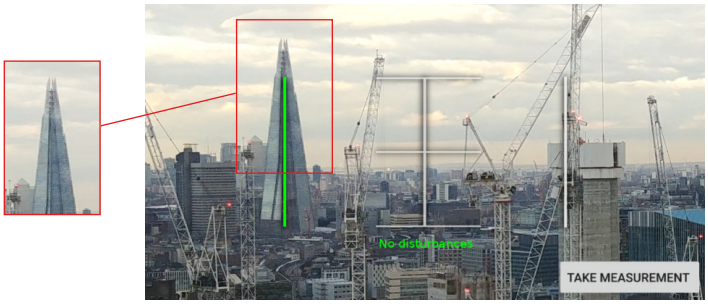


Abbildung 4.20: Beispiel einer Fehlkalibrierung. Links ist das Magnetometer kalibriert, rechts weist es einen Bias von -0,2 Gauß auf der x-Achse auf.

Für die Unterscheidung von Nutzungsfehler und Sensorfehler lassen sich hier gut Verfahren aus der Bildverarbeitung verwenden. Eines dieser Verfahren ist das sogenannte *Template Matching*. Bei diesem Verfahren wird ein Bildausschnitt von einem Bild in einem anderen Bild gesucht. Dabei wird der Bildausschnitt über das größere Bild geschoben und die Differenzen der einzelnen Bildpunkte ermittelt. Wenn die Differenz einen Schwellenwert unterschreitet, ist der Bildausschnitt im anderen Bild gefunden worden. Die Position des Bildausschnittes lässt mittels der Position mit dem geringsten Unterschied ermitteln [A111]. Ein Beispiel, wie das Template Matching bei der beschriebenen Messmethode verwendet werden kann, ist in Abbildung 4.21 veranschaulicht.

Bei der ersten Messung wird ein Ausschnitt des Bildes aus der Mitte ausgewählt, welcher als Template für die linke und rechte Messung verwendet wird. Das Template wird



Template Linke Messung

Abbildung 4.21: Bestimmung der Template-Position bei der zweiten Messung

während der Messung der Sensordaten im Kamerabild gesucht und dessen Position innerhalb des Bildes bestimmt. Anhand der Auflösung des Bildes, der vertikalen Position des Templates in der zweiten und dritten Messung sowie dem FOV der Kamera lässt sich die tatsächliche Orientierungsänderung relativ zur mittleren Messung bestimmen. Die Formeln für diese Berechnungen sind in den Gleichungen 4.9 und 4.10 dargestellt.

$$\Delta x_l = x_t - (w_b \cdot 0,25 - w_t \cdot 0,5) \tag{4.9}$$

$$\Delta x_r = x_t - (w_b \cdot 0,75 - w_t \cdot 0,5) \tag{4.10}$$

Die linke Abweichung Δx_l, gemessen von der linken oberen Ecke ergibt sich aus der Differenz der ermittelten horizontale Position des Templates im Bild x_t zu der erwarteten Position des Templates relativ zur Mitte des Bildes. Hierbei ist w_b die Breite des Originalbildes in Pixel, w_t die Breite des ausgewählten Templates. Die rechte Abweichung Δx_r wird äquivalent berechnet. Der Koordinatenursprung für Bild und Template liegt wie in der Bildverarbeitung üblich links oben. Der Messfehler in Grad für die linke und die rechte Messung lässt sich durch die Formel in Gleichung (4.11) ermitteln. Mittels der daraus berechneten Winkel lassen sich nun Nutzungsfehler und Messfehler voneinander unterscheiden.

$$E = \frac{\Delta x \cdot A}{w_b} \tag{4.11}$$

Mithilfe der Bildverarbeitung lässt sich folglich die Genauigkeit der absoluten Orientierung des Rotation-Vectors zuverlässiger bestimmen. Die Verwendung von Bildverarbeitung mit Template Matching setzt jedoch voraus, dass das bei der Messung anvisierte Objekt markant genug ist, um in den beiden nachfolgenden Messungen zuverlässig erkannt zu werden. Bei dem in Abbildung 4.21 dargestellten Gebäude wäre das der Fall. Sind jedoch mehrere gleichartige Objekte in dem Bild vorhanden, kann es zu Verwechslungen kommen. Dies kann durch heuristische Methoden zumindest teilweise korrigiert

werden. Etwa, wenn es mehrere potenzielle Übereinstimmungen gibt, sollte die Position für das Template gewählt werden, die am nächsten zu der erwarteten Position liegt. Fehler können hier aber nicht gänzlich ausgeschlossen werden.

Die in Unterabschnitt 4.2.4 beschriebenen Nutzungsfehler können sich auch auf die Genauigkeit der Bildanalyse auswirken. Durch eine veränderte Position der Kamera verändert sich auch die Perspektive (Parallaxe) [A112]. Gerade bei nahen Objekten kann eine Verschiebung der Position dazu führen, dass Objekte, die im Template sichtbar sind, bei der zweiten und dritten Messung durch ein anderes Objekt verdeckt werden. Dadurch besteht die Möglichkeit, dass das Template in der zweiten und dritten Messung nicht mehr gefunden werden kann, oder es entsteht ein Versatz bei der Analyse, der das Ergebnis verfälscht. Durch die in Unterabschnitt 4.2.4 bereits ermittelte minimale Distanz von Testperson und anvisierten Objekt von 100 m ist die perspektivische Verschiebung jedoch vernachlässigbar klein.

4.3 Bewertung mittels Roboterarm

Bei den Methoden, die bisher für die Bewertung des Sensorsubsystems beschrieben wurden, werden die benötigten Bewegungen von einer Testperson durchgeführt. Erwartungsgemäß unterscheiden sich die durchgeführten Bewegungen jeder Messung voneinander. Ein möglicher nächster Schritt wäre, die vordefinierten Bewegungen mit der Hilfe eines Roboterarms durchzuführen. Dadurch fällt der Einflussfaktor der testenden Person weg, da die Bewegungen jedes Mal exakt wiederholt werden. Hiermit ließe sich ermitteln, wie weit die Testperson trotz Nutzerführung den Test noch beeinflussen kann, indem man die Testergebnisse mit denen eines Roboters vergleicht. Die Nutzung eines Roboterarms bei der Bewertung widerspricht hier zwar dem Grundgedanken der Testmethode, ohne teures zusätzliches Equipment auszukommen, kann jedoch gut für die Bewertung der Messmethode an sich verwendet werden. Es wurden jeweils 20 Messungen mit fünf Smartphones durchgeführt. Bei zehn der Messungen wurden die Smartphones um ihre eigene Achse gedreht, bei fünf Messungen entlang einer Halbkugel, was der echten Bewegung beim Test durch eine Testperson am nächsten kommt.

Bei der Auswertung der Messungen, die mit dem Roboterarm durchgeführt wurden, war schnell offensichtlich, dass dieser bei der Bewertung der Sensordatenfusion von Smartphones nicht ohne Weiteres genutzt werden kann. Der Grund hierfür ist, dass die einprogrammierten Bewegungen und Drehungen mit dem Roboterarm zu linear und gleichmäßig sind. Bei mehreren der Geräte hat die Sensordatenfusion die gleichmäßige Winkelgeschwindigkeit als Sensorfehler interpretiert und versucht, diesen zu korrigieren. Das Ergebnis ist in Abbildung 4.22 zu erkennen. Die gestrichelten Linien entsprechen der tatsächlich durchgeführten Bewegung. Es bestände die Möglichkeit, die Bewegungen weicher zu gestalten, oder den Roboter menschliche Bewegungen nachahmen zu lassen. Dies hätte jedoch die Komplexität der Testdurchführung deutlich erhöht und widerspricht dem Ziel, die Testmethode ohne großen Aufwand zur Verfügung zu stellen.

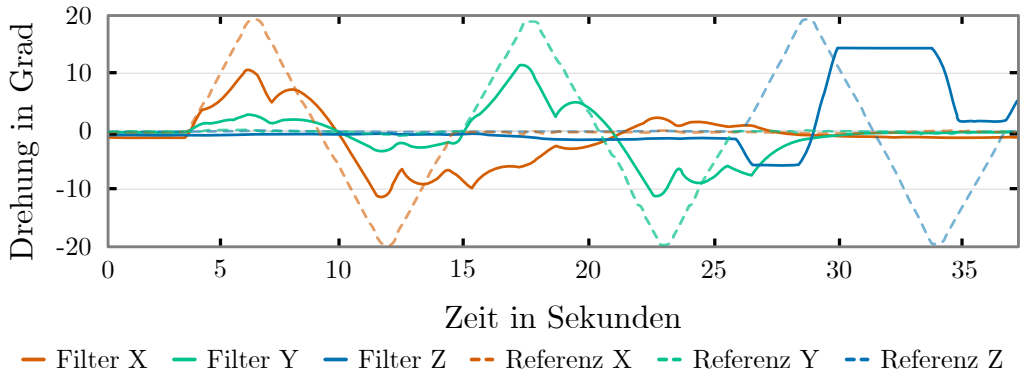


Abbildung 4.22: Ergebnisse der Messung der Sensordatenfusion mit einem Roboterarm

In Kombination mit der Korrektur durch die Gravitation entstehen bei der x- und y-Achse ein Wellenmuster. Der Winkel für die z-Achse springt, da hier keine Korrektur durch die Gravitation stattfinden kann. Dieses Verhalten ist nicht einheitlich, aber auf allen untersuchten Geräten zu beobachten. Infolgedessen wurden keine weiteren Messungen mit einem Roboterarm durchgeführt, da die Ergebnisse nicht verwertbar sind.

4.4 Bewertung von digitalen Zeigegeräten

Neben den bisher behandelten Smartphones gibt es noch eine Vielzahl anderer digitaler Geräte, die mittels Inertialsensoren Bewegungen erfassen und so eine Mensch-Maschine-Interaktion ermöglichen. Zu diesen gehören auch die bereits erwähnten Smartwatches, aber auch Smartpens können hier als Alternative zu herkömmlichen digitalen Zeigegeräten wie der Computermaus verwendet werden. Oftmals werden sie auch als digitaler Laserpointer für Präsentationen eingesetzt. Das prominenteste Beispiel eines Smartpens ist der S-Pen von Samsung [A113]. Smartpens haben den Vorteil, dass sie aufgrund der Nutzung von Inertialsensoren mehrere Aufgaben erfüllen können. Meistens vereinen sie folgende Funktionalitäten:

1. Gestensteuerung, mittels derer es möglich ist Programmen einfache Befehle zu geben, wie etwa eine PowerPoint-Präsentation weiter zu schalten oder die Lautstärke bei einem Video- oder Musikplayer zu verändern.
2. Als digitales Zeigegerät, das als Ersatz für eine Computermaus oder einen Laserpointer bei Präsentationen dienen soll.

Bei beiden oben genannten Funktionalitäten ist eine zuverlässige Gestenerkennung und eine gute Handhabbarkeit bei der Benutzung als Zeigegerät sehr wichtig, um bei Nutzer:Innen eine möglichst hohe Akzeptanz zu erzielen. Ein wichtiger Aspekt hierbei ist, dass eine horizontale Bewegung des Stiftes sich auch als eine horizontale Bewegung des Zeigers widerspiegelt. Selbiges gilt natürlich auch für vertikale Bewegungen. Da der Smartpen bei der Benutzung in jeder möglichen Lage gehalten und währenddessen auch gedreht werden kann, ist es notwendig, eine Kompensation für diese Verdrehung zu implementieren. Im Zuge der Entwicklung einer solchen Kompensation ist die Patentschrift [B4] entstanden. Diese beschreibt, wie mittels einer gezielten Nutzung von Beschleunigungssensor und Gyroskop eine energiesparende und trotzdem schnelle Kompensation für die Verdrehung eines Smartpens erreicht werden kann.

Je nach Beschaffenheit ist es möglich, dass ein Smartpen als *Human Interface Device* alternativ zu einer Maus verwendet werden kann. Es ist jedoch auch möglich, dass Anwendungen direkt die Daten des Smartpens erhalten und dann angepasst in die jeweilige Anwendung verarbeiten. Gerade bei der Benutzung als digitales Zeigegerät ist es wichtig, dass die Bewegungen verzögerungsfrei umgesetzt werden und der Zeiger nicht springt oder sich unvorhersehbar bewegt. Dies erfordert viel Arbeit und Feinabstimmung, damit das Zeigegerät für eine großer Nutzerbasis anwendbar ist. In [A114] wird gezeigt, dass viele der alternativen Zeigegeräte weit von der Nutzbarkeit einer Computermaus entfernt sind und noch viel Arbeit notwendig ist, dieselbe Handhabbarkeit zu erreichen. Es ist jedoch nicht einfach, die Handhabbarkeit eines digitalen Zeigegerätes zu ermitteln oder zu verbessern, da zeitgleich mehrere Aspekte berücksichtigt werden müssen. Am wichtigsten ist jedoch die subjektive Bewertung der Handhabbarkeit durch die Nutzer:Innen.

Diese subjektive Bewertung wird maßgeblich beeinflusst von der Präzision des Smartpens, die aussagt, wie zuverlässig ein Zeiger von einem Punkt zu einem anderen Punkt bewegt werden kann. Ein anderer wichtiger Aspekt ist die Reaktionsgeschwindigkeit. Es ist wichtig, dass die Bewegung des Zeigers nicht merklich später geschieht als die durchgeführte Bewegung. Andernfalls kann dies als störend empfunden werden und führt zu dem Eindruck, dass sich das Zeigegerät träge oder schwammig verhält.

Um die Nutzbarkeit eines digitales Zeigegerät bewerten zu können, wurde eine ISO-Norm [A27] eingeführt. Diese Norm soll sicherstellen, dass die digitalen Zeigegeräte einen Mindeststandard erfüllen. Es wurden jedoch in [A28] gezeigt, dass es mit diesem Standard nicht möglich ist, zuverlässige Rückschlüsse auf die Nutzererfahrung zu ziehen. Die in Abschnitt 3.1 vorgestellten Methoden Fitt's Law und Accot-Zhai Steering Law sind zwar im MMI Bereich für die Bewertung von digitalen Zeigegeräten etabliert, nutzen beide jedoch nur die Geschwindigkeit als Bewertungskriterium. Technische Aspekte der Zeigegeräte werden nicht berücksichtigt

Um auch technisch bedingten Eigenschaften in die Bewertung einfließen zu lassen und auf diesem Anwendungsgebiet Entwickler:Innen eine Möglichkeit zu geben, die Qualität der Messungen und Daten eines Smartpens bewerten zu können, wurde eine spezielle Testmethode entworfen. Diese hat zum Ziel, anhand von quantitativen Messwerten eine Aussage über die subjektive Nutzungserfahrung von Smartpens aufgrund von dessen technischen Eigenschaften machen zu können. Es soll somit auch ermöglicht werden, die Nutzbarkeit von mehreren untersuchten Smartpens oder anderen digitalen Zeigegeräten miteinander vergleichen zu können. Weiterhin war es wie bei den vorherigen Arbeiten wichtig, dass die Messmethode möglichst einfach durchzuführen ist und kein zusätzliches Equipment erfordert. Zur Bewertung des Smartpens wurden zu diesem Zweck drei separate Tests entworfen, die jeweils einen Aspekt der Nutzbarkeit bewerten sollen:

Test für Handhabbarkeit: Dieser Test bestimmt, wie gut einfache und häufige Aktionen mit dem Smartpen ausgeführt werden können.

Test für Reaktionsfähigkeit & Geschwindigkeit: Mit diesem Test kann bestimmt werden, wie schnell Nutzer:Innen einfache und häufige Aktionen durchführen können und wie schnell der Smartpen reagiert.

Test für Präzision & Stabilität: In diesem Test soll ermittelt werden, wie genau der Zeiger geführt werden kann und komplexe Aktionen durchgeführt werden können.

Der Testaufbau ist entworfen, dass mehrere Geräte von den Testpersonen getestet werden. Nach jedem Testdurchlauf soll die Testperson eine Bewertung des jeweils getesteten Zeigegerätes abgeben, um die quantitative Auswertung aus den Tests mit der subjektiven Bewertung der Testpersonen vergleichen zu können und daraus Rückschlüsse zu ziehen. Im Folgenden werden die drei Tests und deren Durchführung beschrieben. Anschließend folgt die Zusammenfassung der Ergebnisse und eine Bewertung. Die entwickelte Messmethode wurde 2019 auf der *PECCS Konferenz* in Wien vorgestellt [B3].

4.4.1 Test für Handhabbarkeit

Der erste Test soll bewerten, wie genau die durchgeführten Bewegungen von den Inertialsensoren des Smartpens ermittelt und in Bewegungen des Zeigers umgesetzt werden können. Dies spiegelt sich in der Handhabbarkeit des Smartpens wider. Wie dargelegt beschreibt die Handhabbarkeit hier, wie präzise häufig wiederhole Aktionen mit dem Gerät ausgeführt werden können. Dabei handelt es sich im Falle eines Zeigegerätes meist um die Fähigkeit, den Zeiger direkt von einem Punkt auf dem Bildschirm zu einem anderen Punkt bewegen zu können. Die für eine gute Handhabbarkeit wichtigen Faktoren sind, dass der Zeiger möglichst gradlinig bewegt werden kann und auf dem anvisierten Ziel zum Stehen kommt. Gerade, wenn Nutzer:Innen Probleme dabei haben, einen Punkt gezielt anzusteuern, kann dies frustrierend sein. Die Fähigkeit, gradlinig und gezielt ein Ziel auf dem Bildschirm anzusteuern wird in diesem Test bewertet. Hierfür soll die Testperson mit dem Zeiger zufällig auf dem Bildschirm verteile Punkte ansteuern. Sobald ein Punkt erreicht ist, soll der Zeiger dort 2 Sekunden verbleiben, bis der nächste Punkt angesteuert werden kann.

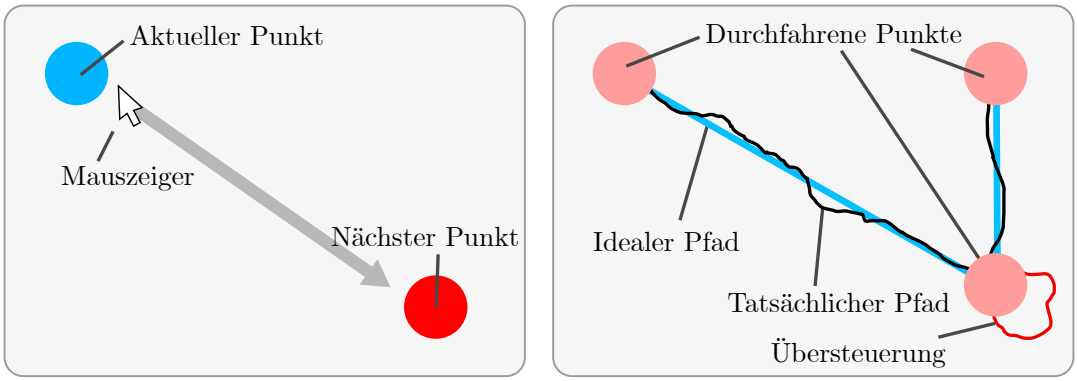


Abbildung 4.23: Links: GUI für den Test zur Handhabbarkeit während des Tests. Rechts: Ergebnisse des Tests für die Handhabbarkeit.

Die grafische Benutzeroberfläche für diesen Test in ist Abbildung 4.23 gezeigt. Der aktuelle Punkt, bei dem sich der Zeiger gerade befindet, wird in Blau dargestellt; der nachfolgende Punkt, der angesteuert werden soll, in Rot. Der Test ist beendet, sobald die Testperson eine vorher festgelegte Anzahl an Punkten erreicht hat. Die Bewertung für die Handhabbarkeit wird anhand zweier Faktoren berechnet.

Der erste Faktor ist das Verhältnis des (kürzesten) Weges zwischen allen Punkten und dem Weg, den der Zeiger tatsächlich bewegt wurde. Hiermit soll bewertet werden, wie gezielt eine Person den Zeiger von einem Punkt zu einem anderen bewegen kann. Der kürzeste Weg zwischen den Punkten wird als Referenz verwendet, damit die Berechnungen unabhängig von der Auflösung des Bildschirms sind. Ein fester Wert würde zu unterschiedlichen Ergebnisse bei unterschiedlichen Auflösungen führen [A115].

In Gleichung (4.12) wird gezeigt, wie das Verhältnis zwischen kürzester und tatsächlich zurückgelegter Distanz berechnet wird.

$$s_{dist} = 1 - \alpha \cdot \frac{d_{bewegt} - d_{min}}{d_{min}} \tag{4.12}$$

Das Ergebnis s_{dist} ist das Verhältnis zwischen minimaler Distanz d_{min} und der zurückgelegten Distanz d_{bewegt}. Das Ergebnis wird gewichtet mit dem Faktor α für eine geeignete Skalierung dieses Bewertungskriteriums. Ohne α würde die Bewertung des Tests gegen null laufen, wenn die zurückgelegte Distanz mindestens zweimal so groß ist wie die minimale Distanz. Dies ist jedoch sehr unwahrscheinlich. Basierend auf den ersten Tests wurde das α auf 3 festgelegt, sodass die Bewertung gegen null geht, wenn die zurückgelegte Distanz $33.\overline{3}\,\%$ größer als die minimale Distanz ist.

Der zweite Faktor bewertet die Möglichkeit, mit dem Zeiger gezielt auf einem Punkt zum Stehen zu kommen. Wenn der Zielpunkt erreicht wurde, sich der Zeiger aber über diesen hinaus bewegt, wird dieses Übersteuern gemessen. Eine solche Bewegung ist in Abbildung 4.23 auf der rechten Seite als roter Pfad dargestellt. Ein Übersteuern passiert häufig, wenn das Gerät zu langsam auf die Benutzereingabe reagiert oder generell nicht gut handhabbar ist. Die Berechnung für die Bewertung eines Übersteuerns wird in Gleichung (4.13) beschrieben.

$$s_{über} = 1 - \beta \cdot \frac{d_{über}}{d_{min}} \tag{4.13}$$

Ein größeres Übersteuern wird in einer niedrigeren Bewertung resultieren. Die Bewertung des Übersteuerns wird als Verhältnis zwischen minimaler Distanz d_{min} und der zurückgelegten Distanz beim Übersteuern $d_{über}$ berechnet. Für eine Gewichtung der Ergebnisse wird, wie bei Gleichung (4.12) auch, in Gleichung (4.13) ein Gewichtungsfaktor β verwendet. Dieser wurde nach initialen Tests auf 4 festgelegt. Dies bedeutet, dass die Bewertung für das Überschwingen Null erreicht, wenn beim Überschwingen mindestens 25 % der minimalen Distanz zurückgelegt wurde.

Die Berechnung der Gesamtbewertung für den Test zur Handhabbarkeit wird in Gleichung (4.14) gezeigt.

$$s_h = s_{dist} \cdot s_{über} \tag{4.14}$$

Die Werte für s_{dist} und $s_{über}$ werden jeweils auf das Intervall $0 \leq s \leq 1$ begrenzt. Für die letztendliche Bewertung werden die beiden Teilergebnisse miteinander multipliziert. Dies berücksichtigt, dass die Handhabbarkeit auch stark beeinträchtigt ist, wenn lediglich in einem der Teilergebnisse eine geringe Punktzahl erreicht wurde.

4.4.2 Test für Reaktionsfähigkeit & Geschwindigkeit

Der Aufbau des *Tests für Reaktionsfähigkeit & Geschwindigkeit* des Zeigegerätes ist sehr ähnlich wie der Aufbau des *Tests für Handhabbarkeit*. Die Testperson muss wieder mithilfe des Zeigegerätes mehrere Punkte auf dem Bildschirm abfahren. Anders als beim *Test für Handhabbarkeit* kommt es in diesem Test jedoch nicht darauf an, den Zeiger in einer möglichst geraden Linie zwischen den Punkten zu bewegen, sondern diese in möglichst kurzer Zeit zu erreichen. Damit soll gemessen werden, wie schnell mit dem Zeigegerät navigiert werden kann. Ein langsames oder träges Verhalten wird von Nutzer:Innen oftmals als störend empfunden, da Aufgaben nicht schnell genug erledigt werden können.

Hierbei berücksichtigt der Test nur die Zeit, die eine Testperson tatsächlich benötigt, um von einem Punkt zu einem nächsten Punkt zu kommen. Sobald der Zeiger einen Punkt berührt, wird die Zeit angehalten. Dies soll verhindern, dass die Reaktionsfähigkeit der Testperson einen Einfluss auf das Testergebnis hat. Wenn ein neuer Punkt an einer zufälligen Stelle auftaucht und angesteuert werden muss, kann die Testperson sich demzufolge erst darauf einstellen und anschließend den Zeiger bewegen. Die Bewertung dieses Tests wird anhand von Gleichung (4.15) berechnet.

$$s_{geschw} = \frac{n_{punkte} \cdot \gamma \cdot f_{distanz}}{t} \tag{4.15}$$

Der Parameter t ist hierbei die Gesamtzeit des Tests in Millisekunden. Der Faktor γ wird für die Gewichtung der Anzahl der Punkte relativ zur Zeit t verwendet. Dies ermöglicht es, festzulegen, wie viele Punkte pro Sekunde erreicht werden müssen, um die höchste Punktzahl zu erreichen. Basierend auf ersten Tests wurde γ auf einen Wert von 500 festgelegt, was zwei Punkten pro Sekunde entspricht. Abermals ist das Ergebnis auf das Intervall $0 \leq s_{geschw} \leq 1$ begrenzt.

Der Faktor dieses Tests $f_{distanz}$ spielt eine wichtige Rolle für die Wiederholbarkeit und Genauigkeit für diesen Test für die Geschwindigkeit des Zeigegerätes. Der Test ist so entworfen, dass die Punkte, die eine Testperson mit dem Zeiger ansteuern muss, an zufälligen Stellen platziert werden, um einen Trainings- oder Gewöhnungseffekt zu vermeiden. Durch die zufällige Platzierung der Punkte unterscheidet sich jedoch die Distanz, die bei jedem Test zwischen den Punkten liegt. Dies ist beispielsweise problematisch, wenn bei einem Test alle Punkte recht nah beieinander liegen und bei einem anderen Test sehr weit auseinander. Die schnellere Erreichbarkeit der dichter liegenden Punkte führt zu einer Verzerrung der Ergebnisse. Der Faktor $f_{distanz}$ ist ein Gewichtungsfaktor, der die Distanz zwischen den Punkten berücksichtigt. Die Berechnung von $f_{distanz}$ ist in Gleichung (4.16) aufgezeigt.

$$f_{distanz} = \sqrt{\frac{d_{test}}{n_{punkte} \cdot d_{mittel}}} \tag{4.16}$$

Der Wert d_{mittel} ist die mittlere Distanz zwischen den möglichen Punkten und kann vor dem Test unter Berücksichtigung der Bildschirmauflösung berechnet werden. Die tatsächliche Distanz zwischen den Punkten ist d_{test}. Wenn dieser Wert größer als d_{mittel} ist; die Testperson mit dem Zeiger also eine größere Distanz zurücklegen muss, wird das Ergebnis nach oben korrigiert. Bei kleineren Distanzen erfolgt eine Korrektur nach unten. Nach ersten Tests wurde entschieden, die Quadratwurzel des Verhältnisses von d_{test} zu $dist_{mittel}$ zu verwenden, da bei einer größeren Distanz zwischen den Punkten auch eine höhere Geschwindigkeit des Zeigers erreicht werden kann. Dies bliebe bei einem linearen Verhältnis unberücksichtigt. Die Anzahl der erreichten Punkte ist mit n_{punkte} angegeben.

4.4.3 Test für Präzision & Stabilität

Im *Tests für Präzision & Stabilität* soll bewertet werden, wie genau sich der Zeiger mit einem Zeigegerät bewegen lässt. Ein präzises Bewegen des Zeigers ist für die Interaktion von kleinen Objekten auf dem Bildschirm von Relevanz, beispielsweise beim Zeichnen einer Form oder beim Markieren von Text. Wenn ein genaues Navigieren mit einem Zeigegerät nicht möglich ist, ist anzunehmen, dass das Zeigegerät von Nutzer:Innen nicht gut angenommen wird. Gemäß dem Entwurf dieses Tests bewegt die Testperson den Zeiger möglichst genau entlang der Kanten mehrerer Objekte, um deren Form nachzuzeichnen. Der Unterschied zwischen der tatsächlichen und der gezeichneten Form wird genutzt, um die Bewertung für diesen Test zu berechnen. Die grafische Benutzeroberfläche für diesen Test ist in Abbildung 4.24 dargestellt.

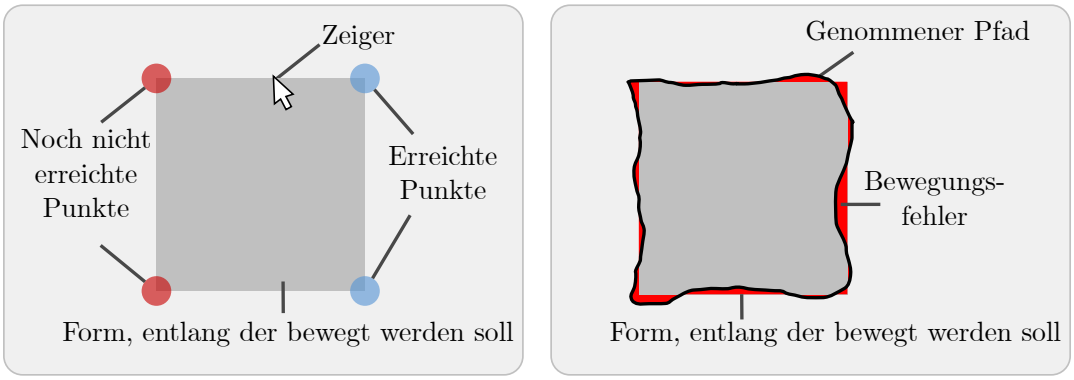


Abbildung 4.24: Links: GUI für den *Test für Genauigkeit* während des Tests. Rechts: Anzeige der Ergebnisse. Die rote Fläche ist der Bewegungsfehler.

Die roten und blauen Punkte auf der linken Seite in Abbildung 4.24 zeigen die Punkte, die die Testperson während des Tests durchlaufen muss, um sicherzustellen, dass auch tatsächlich die Form nachgezeichnet wurde. Für die Bestimmung der Präzision und Stabilität des Zeigegerätes werden insgesamt drei unterschiedliche Formen verwendet, welche verschiedene Aspekte bewerten können:

Quadrat Mit dem Quadrat kann bewertet werden, wie gut der Zeiger mit dem Zeigegerät in vertikaler und horizontaler Richtung bewegt werden kann.

Dreieck Das Dreieck erlaubt es zu bewerten, wie gut der Zeiger entlang diagonaler Linien bewegt werden kann. Dies unterscheidet sich bei einigen Geräten von horizontalen und vertikalen Bewegungen.

Kreis Mit dem Kreis wird getestet, wie gut runde und fließende Bewegungen mit dem Zeigegerät nachgezeichnet werden können.

Mit den drei oben beschriebenen Formen ist es möglich, die wichtigen Aspekte der Genauigkeit des Zeigegerätes zu bestimmen. Für die Berechnung der Bewertung dieses Tests werden zwei Teilbewertungen ermittelt.

Für die erste Teilbewertung wird ermittelt, wie groß die Differenz zwischen der tatsächlichen Form und der gezeichneten Form ist. Dies gibt Aufschluss darüber, wie gut es Nutzer:Innen möglich ist, die Formen an sich nachzuzeichnen. Die Berechnung erfolgt anhand der Formeln in Gleichung (4.17) und Gleichung (4.18).

$$d_{mittel} = \frac{\sum_{n=0}^{N} e_n}{N} \tag{4.17}$$

Der Wert e_n ist die kürzeste Distanz zwischen der $n - ten$ Zeigerposition und der Außenlinie der Form. N ist die Anzahl der Zeigerposition, die in dem Test aufgezeichnet wurden.

$$s_{form} = 1 - \frac{\delta \cdot d_{mittel}}{u_{form}} \tag{4.18}$$

Die mittlere Distanz zwischen der Form und dem nachgezeichneten Pfad wird mit dem Faktor δ skaliert, um den Test gewichten zu können. Das δ wurde nach initialen Tests auf einen Wert von 25 festgelegt. Der Umfang der Form ist mit u_{form} angegeben.

Das zweite wichtige Kriterium, das in diesem Test bewertet wird, ist die Stabilität des Zeigegerätes. Für Nutzer:Innen ist es nicht nur wichtig, dass der Zeiger sich genau entlang einer Form führen lässt, sondern auch, dass die Position des Zeigers nicht springt. Hierfür wird ein zweites Bewertungskriterium genutzt, welches die Summe aller Winkel zwischen den Positionen des Zeigers auf dem Bildschirm berechnet. Im Optimalfall

würde ein Winkel von 360 ° (bzw. 180 ° beim Dreieck) benötigt, um eine der Formen nachzuzeichnen. Wenn sich der Zeiger jedoch nicht stabil verhält und häufig springt, ist die Summe der Winkel deutlich größer. Dieser Effekt wird in Abbildung 4.25 dargestellt, welche den optimalen Pfad in Blau und den tatsächlichen Pfad des Zeigers in Rot darstellt.

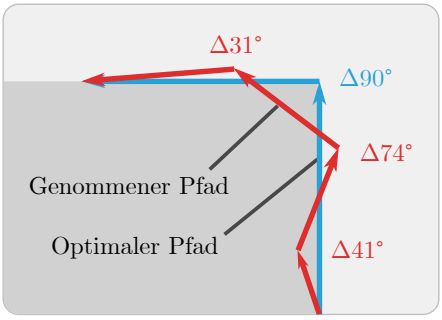


Abbildung 4.25: Optimaler Pfad entlang der Form (blau) und gemessener Pfad eines instabilen Zeigers (rot)

Summiert man alle Winkel des blauen Pfades auf, erhält man einen Winkel von 90 °. Für den roten Pfad liegt die Summer aller Winkel bei 146 °. Je höher die Summe aller Winkel bei der Bewegung sind, desto schlechter ist die Stabilität des Zeigegerätes. Bei sehr kleinen Bewegungen, die nur wenige Pixel weit sind, kann es jedoch dazu kommen, dass große Fehler von bis zu 180 ° hinzugefügt werden. Daher wird der Fehler erst berechnet, wenn sich der Zeiger mindestens 25 Pixel vom letzten Punkt entfernt hat. Bei kleineren Distanzen hat sich gezeigt, dass ansonsten auch eine unruhige Hand das Ergebnis beeinflusst. Mit den ermittelten Winkeln lässt sich mittels Gleichung (4.19) die Bewertung für die Stabilität des Zeigegerätes errechnen.

$$s_{winkel} = 1 - 0.5 \cdot \left(1 - \frac{w_{min}}{w_{bewegt}}\right) \tag{4.19}$$

Der Wert w_{min} ist 360 ° (bzw. 180 °) geteilt durch die Anzahl der berechneten Winkel. w_{bewegt} ist der mittlere Winkel, der aus den Bewegungen des Zeigers errechnet wurde. Das Gesamtergebnis für den *Test für Präzision & Stabilität* wird mit der Gleichung (4.20) berechnet.

$$s_{genauigkeit} = s_{form} \cdot s_{winkel} \tag{4.20}$$

Das Ergebnis $s_{genauigkeit}$ ist das Produkt von s_{form} und s_{winkel}. Wie bei dem *Test für Handhabbarkeit* werden beide Teilergebnisse miteinander multipliziert, da eine schlechte Bewertung bei einem der Teilergebnisse die gesamte Präzision und Stabilität beeinträchtigt.

4.4.4 Testdurchführung

Für die Evaluation der beschriebenen Testmethoden wurden Messungen mit 10 Personen im Alter von 29 bis 58 Jahren durchgeführt. Drei der Testpersonen waren weiblich, sieben männlich. Eine große Vielfalt bei den Testpersonen ist wichtig, da gezeigt wurde, dass diese Faktoren gerade bei einer Mensch-Maschine-Interaktion einen großen Einfluss haben [A116]. Für den Test wurden folgende Geräte verwendet:

- eine Computermaus
- ein kommerziell erhältliches digitales Zeigegerät
- ein Smartphone
- ein Prototyp eines Smartpens

Der Prototyp für einen Smartpen bestand aus einem 1,5 cm x 3,0 cm großem Sensorboard, welches fest an einem Kugelschreiber befestigt war. Die Maus wurde auf einem Mauspad auf einem ebenen Tisch benutzt, die anderen Zeigegeräte wurden wie ein Laserpointer gehandhabt. Der Bildschirm, auf dem der Test ausgeführt wurde, war ein 55 Zoll großer LCD-Monitor in 3 Meter Entfernung zu den Testpersonen. Die Computermaus wurde als Referenz verwendet, da sie das meistgenutzte Eingabegerät für Computer ist [A117], [A118]. Für die Geschwindigkeit des Mauszeigers wurden die Standardeinstellungen des Betriebssystems verwendet. Bei ausführlicheren Untersuchungen der Eingabegeräte sollten mehrere Messdurchläufe mit unterschiedlichen Einstellungen der Zeigergeschwindigkeit durchgeführt werden.

Um einem Trainings- oder Gewöhnungseffekt bei der Benutzung der Zeigegeräte vorzubeugen, wurde die Reihenfolge, in der die Testpersonen die Zeigegeräte getestet haben, zufällig festgelegt. Insgesamt wurden 160 Messungen für alle Tests zusammen durchgeführt. Nach der Durchführung der Tests wurden die Testpersonen nach ihrer subjektiven Bewertung des Zeigegerätes gefragt, indem Sie die Geräte nach ihrer Nutzbarkeit von „am besten nutzbar“ nach „am schlechtesten nutzbar“ sortieren sollten.

Erwartete Ergebnisse

Um zu zeigen, dass die quantitativen Bewertungen der Tests das subjektive Empfinden der Nutzer:Innen widerspiegelt, sollten die Bewertungen der Tests und die Bewertungen der Nutzer:Innen die gleichen Tendenzen und Abstufungen zeigen. Gleichartige Ergebnisse würden die Annahme stützen, dass die entworfenen Tests geeignet sind, das subjektive Empfinden der Testpersonen anhand der festgelegten Kriterien zu ermitteln.

Es war weiterhin zu erwarten, dass die Ergebnisse für die Computermaus die besten sein würde, da jede der Testpersonen mit dieser vertraut war. Zudem hat sich die Computermaus in der Vergangenheit als eine der präzisesten Eingabegeräte für Computer erwiesen [A29]. Daher wird die Computermaus als Referenz verwendet, gegen die die anderen Geräte verglichen werden.

4.4.5 Ergebnisse

In diesem Abschnitt werden die Ergebnisse der durchgeführten Tests aufgezeigt. Zuerst werden die (Teil-)Ergebnisse der einzelnen Tests erläutert. Anschließend werden die Ergebnisse zusammengefasst und mit den subjektiven Bewertungen der Testpersonen ins Verhältnis gesetzt.

Test für Handhabbarkeit

Die Resultate des *Tests für Handhabbarkeit* werden in Abbildung 4.26 dargestellt. Es ist deutlich zu erkennen, dass die Computermaus die höchste Punktzahl erreicht hat. Der Prototyp eines Zeigegerätes erreichte die zweithöchste Punktzahl. Das Smartphone hat in der Gesamtwertung dieses Tests mit 0,67 am schlechtesten abgeschnitten. Diese Abstufung entspricht den subjektiven Bewertungen der Testpersonen nach der Durchführung dieses Tests.

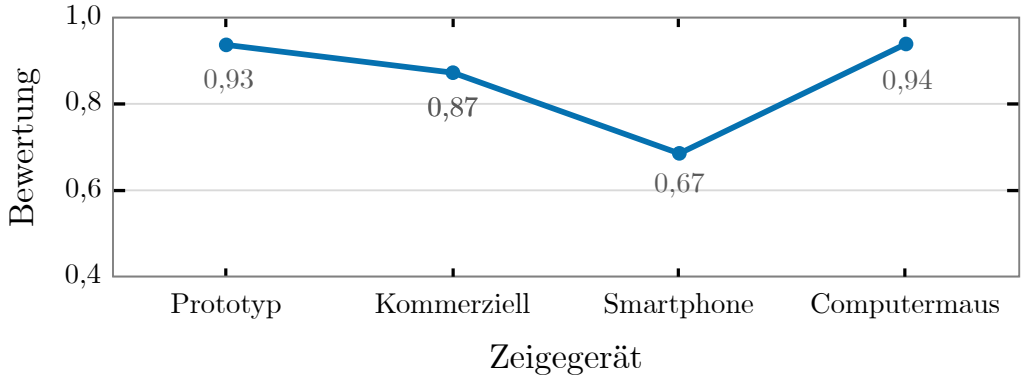


Abbildung 4.26: Gesamtwertung der Handhabbarkeit der Zeigegeräte

Die Teilergebnisse dieses Tests sind in Tabelle 4.4 aufgelistet. Tabelle 4.4 zeigt die Bewertung von 0 bis 1 für die vier untersuchten Geräte für die zwei Teilbewertungen „Bewegung“ und „Übersteuerung“ und die sich daraus für diesen Test ergebende Gesamtbewertung. Es ist zu erkennen, dass die Fehler durch ein Übersteuern nur gering vorhanden sind und die größeren Unterschiede bei der eigentlichen Bewegung entstehen. Hier unterscheiden sich die Bewertungen für die vier Geräte deutlicher. Trotzdem sind beim Übersteuern ebenfalls dieselben qualitativen Unterschiede zu erkennen, speziell das Smartphone hat wieder das schlechteste Ergebnis.

Tabelle 4.4: Bewertungen der Handhabbarkeit der Zeigegeräte

	Prototyp	Kommerziell	Smartphone	Computermaus
Bewegung	0,93	0.90	0.71	0.94
Übersteuern	0,99	0,97	0,96	0,99
Gesamt	0,93	0,87	0,68	0,94

An den Daten ist zu sehen, dass es möglich ist, zwischen geeigneten und weniger geeigneten Geräten zu unterscheiden. Alle untersuchten Geräten waren nutzbar, unterschieden sich jedoch deutlich. Das schlechte Abschneiden des Smartphones ist auf eine sehr hohe Latenz und eine weitere Charakteristik des Smartphones zurückzuführen, welche in den Ergebnissen des *Tests für Genauigkeit und Präzision* näher erklärt wird.

Test für Reaktionsfähigkeit & Geschwindigkeit

Die Ergebnisse des Tests für Reaktionsfähigkeit & Geschwindigkeit zeigen die gleiche Tendenz bei der Bewertung der getesteten Geräte wie beim Test für der Handhabbarkeit. Es wird jedoch schnell in Abbildung 4.27 ersichtlich, dass diese Mal die Computermaus mit Abstand am besten abgeschnitten hat. Eine klare Überlegenheit der Computermaus war in diesem Test zu erwarten, jede der Testpersonen diese täglich nutzt und dementsprechend mit der Handhabung vertraut ist. Das kommerzielle Produkt und das Smartphone kommunizierten per Bluetooth mit dem Computer verbunden, der Prototyp über ein sekundäres Shuttleboard. Das führt bei allen drei Geräten zu einer höheren Latenz. Weiterhin musste die Maus relativ zur zurückgelegten Strecke auf dem Bildschirm eine kürzere Distanz zurücklegen. Dies kann zwar durch eine höhere Empfindlichkeit der anderen Zeigegeräte ebenfalls erreicht werden. Dies führt jedoch in Kombination mit einer höheren Latenz zu einer erschwerten Handhabbarkeit.

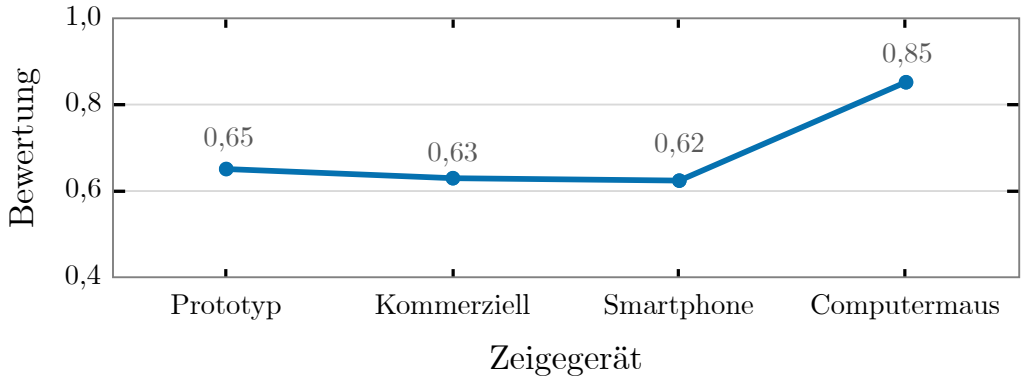


Abbildung 4.27: Bewertung der Reaktionsfähigkeit & Geschwindigkeit der Zeigegeräte

Wie beim *Test für Handhabbarkeit* schnitt der Prototyp am zweitbesten ab, gefolgt von dem kommerziellen Produkt. Das Smartphone erreicht wieder die geringste Punktzahl. Jedoch ist in diesem Test der Unterschied zwischen den Geräten geringer. Dies ist auch darauf zurückzuführen, dass die Gewichtung in diesem Test so gewählt wurde, dass auch die Ergebnisse von der Computermaus abgebildet werden können. Weiterhin wird in diesem Test die Anzahl der erreichten Punkte für die Bewertung verwendet, was zu einer geringeren Abweichung zwischen den Geräten führte, da die Ergebnisse eine höhere Granularität aufweisen.

Test für Präzision & Stabilität

Wie bereits in Unterabschnitt 4.4.3 beschrieben, besteht der *Test für Präzision & Stabilität* aus drei Teilbewertungen für die drei unterschiedlichen Formen. In diesem Test ist bei den Ergebnissen ein Unterschied zu den bisherigen Ergebnissen der anderen beiden Tests zu erkennen. Die Einzelergebnisse sowie das Gesamtergebnis werden in Abbildung 4.28 dargestellt. Der genaue Wert der Gesamtbewertung ist für jedes Gerät als Zahl angegeben.

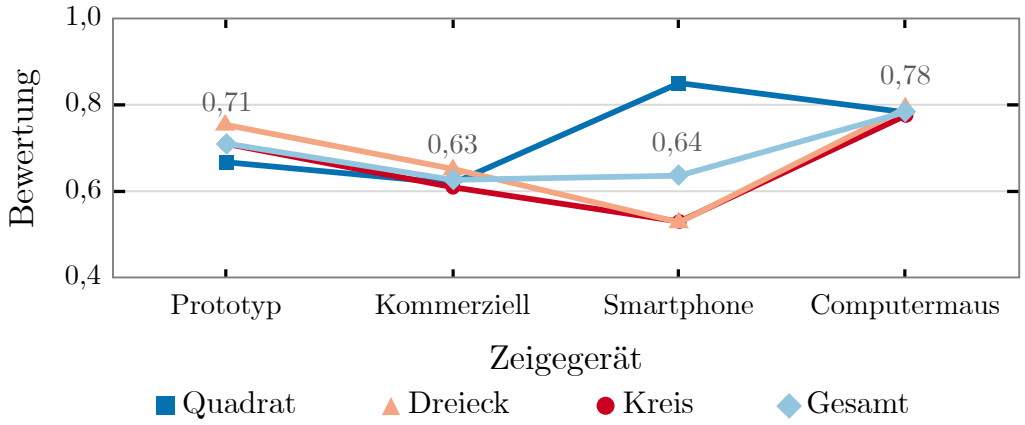


Abbildung 4.28: Gesamtbewertung der Genauigkeit der Zeigegeräte

Die Teilergebnisse für das Dreieck sowie den Kreis zeigen vergleichbare Ergebnisse zu den beiden vorherigen Tests. Bei dem Quadrat ist jedoch zu sehen, dass das Smartphone am besten abgeschnitten hat, besser sogar als die Computermaus. Besonders unter dem Gesichtspunkt, dass das Smartphone in allen anderen Tests am schlechtesten abgeschnitten hat, ist dieses Ergebnis eine Anomalie.

Nach einer detaillierteren Analyse wurde festgestellt, dass es bei dem Smartphone einen Schwellenwert für die Bewegungen gibt, bevor sich der Zeiger bewegt. Die ist ebenfalls beim Prototyp und (höchstwahrscheinlich) den anderen Geräten der Fall. Jedoch ist beim

Smartphone der Schwellenwert sehr hoch gesetzt und wird für jede Achse separat angewandt. Dies führt dazu, dass gerade bei horizontalen und vertikalen Bewegungen eine der Achsen „blockiert“ und der Zeiger in einer perfekt geraden Linie bewegt werden kann. Da das Quadrat nur aus vertikalen und horizontalen Bewegungen besteht, ist dadurch das Smartphone klar im Vorteil. Beim Dreieck und beim Kreis führt der hohe Schwellenwert bei den diagonalen Bewegungen zu einem Treppeneffekt. In bestimmten Anwendungen kann das Blockieren einer Achse auch von Vorteil sein, da es vergleichbar mit der „Ausrichten am Raster“-Funktion von Zeichenprogrammen ist. Da das Smartphone jedoch beliebig drehbar ist, können auch die fixierten Achsen eine beliebige Ausrichtung haben. Weiterhin unterstützen die meisten Zeichenprogramme „Ausrichten am Raster“ als eigene Funktion. Beim Smartphone lässt sich diese Verhalten aber nicht steuern.

Das Gesamtergebnis dieses Tests zeigt trotz des Ausreißers eine sehr ähnliche Bewertung der getesteten Geräte, wie der *Test für Handhabbarkeit* und der *Test für Reaktionsfähigkeit & Geschwindigkeit*. Die Computermaus schneidet wieder am besten ab, gefolgt vom Prototyp. Lediglich die Reihenfolge vom kommerziellen Produkt und dem Smartphone haben gewechselt.

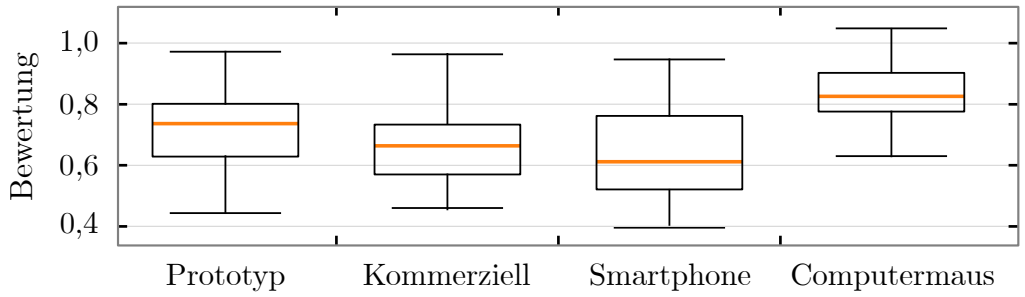


Abbildung 4.29: Verteilung der Testergebnisse der Zeigegeräte für die Genauigkeit

Dass es beim Smartphone Ausreißer in den Daten gibt, lässt sich auch anhand der Verteilung der Messwerte erkennen. Diese ist in dem Box-Plot in Abbildung 4.29 für alle vier Testgeräte dargestellt. Hier liegen das untere und obere Quartil bei dem Smartphone deutlich weiter auseinander als bei den anderen Geräten. Die Ergebnisse werden von dem Test mit dem Quadrat deutlich nach oben gezogen. Weiterhin ist die breite Verteilung der Messergebnisse beim Smartphone ein Indikator dafür, dass Nutzer:Innen weniger zuverlässig präzise und genaue Bewegungen durchführen können.

Gesamtergebnis der Tests

Für den Vergleich der Ergebnisse der Testmethoden mit der subjektiven Nutzerbewertung wurden die Ergebnisse für jedes Gerät normalisiert, sodass sie zwischen null und eins liegen. Das resultierende Ergebnis ist in Abbildung 4.30 dargestellt.

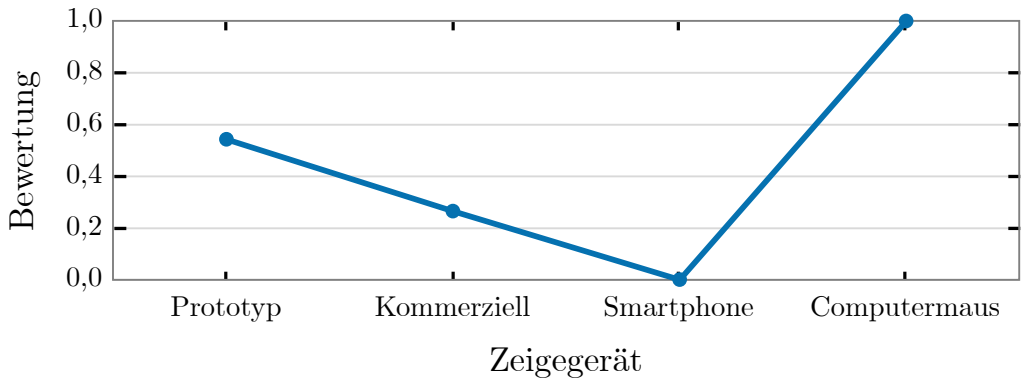


Abbildung 4.30: Normalisierter Mittelwert für die Gesamtbewertung

Es ist zu erkennen, dass der Ausreißer des *Tests für Genauigkeit & Präzision* keinen großen Einfluss auf das Gesamtergebnis hatte. Mit anderen Werten für die Skalierungsfaktoren α, β, γ und δ könnten die Ergebnisse auch noch gewichtet und verändert werden. Eine Änderung dieser Parameter würde jedoch nicht das relative Verhältnis der Geräte zueinander verändern.

Subjektive Nutzerbewertung

Die subjektive Bewertung der Nutzbarkeit der Zeigegeräte hilft dabei zu bewerten, ob die entworfenen Testmethoden in der Lage sind, nicht nur eine quantitative, aber auch eine qualitative Bewertung der getesteten Geräte zu ermöglichen. Im Idealfall sollte die Einschätzung der Testpersonen mit der Bewertung der Tests übereinstimmen, um mit den Untersuchungen eine verlässliche Aussage treffen zu können.

Für die subjektive Beurteilung der Geräte sollten die Testpersonen diesen nach Nutzbarkeit klassifizieren. Das beste Gerät sollte dabei drei Punkte erhalten, das zweitbeste Gerät zwei Punkte, das drittbeste Gerät einen Punkt und das letzte Gerät null Punkte. Die normalisierte Gesamtwertung der Geräte mit Werten zwischen null und eins ist in Abbildung 4.31 dargestellt.

Wie zu erwarten war, wurde die Computermaus von allen Testern ausnahmslos am besten bewertet. Es ist gut zu erkennen, dass die subjektive Bewertung der Testpersonen stark der quantitativen Bewertung der Tests ähnelt. Ohne dem Ausreißer in den Messungen für *Präzision & Stabilität* wären die Testergebnisse sogar noch ähnlicher.

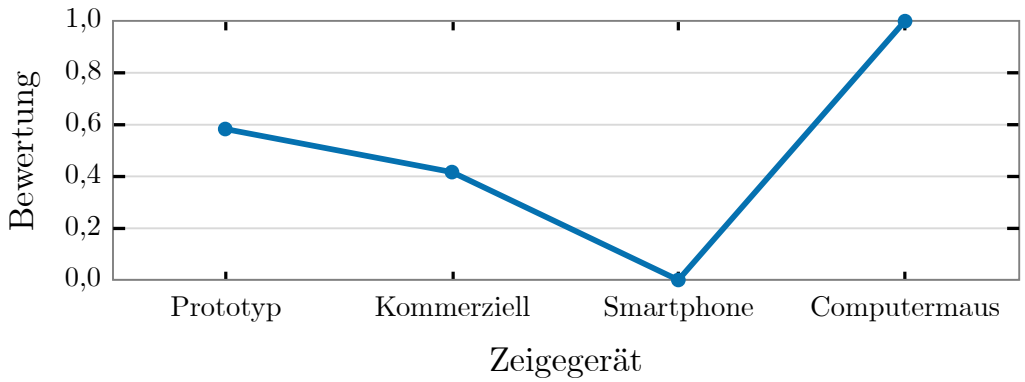


Abbildung 4.31: Normalisierte subjektive Bewertung der Testpersonen

Vergleich

Die quantitativen Ergebnisse der entworfenen Tests sowie der subjektiven Nutzerbewertung zeigen eine starke Korrelation. Das legt den Schluss nahe, dass die entworfenen Tests ein gutes Maß dafür sein können, die Qualität und Nutzererfahrung der Geräte zu bewerten. Abgesehen von dem Ausreißer zeigen alle Tests dieselbe Tendenz zwischen subjektiver Bewertung und den Ergebnissen der Tests. Für eine genauere Feinabstimmung der Tests und einem noch aussagekräftigeren Vergleich wird vorgeschlagen, künftig die subjektive Bewertung zu verfeinern. Beispielsweise, indem die Testpersonen eine Bewertung zwischen 0 und 10 für jeden einzelnen Test abgeben müssen anstatt einer Gesamtbewertung für die Geräte. Dies würde auch eine Normalisierung der Ergebnisse für einen Vergleich überflüssig machen.

4.5 Zusammenfassung

In diesem Kapitel wurde der Schwerpunkt auf Methoden und Untersuchungen auf der Anwendungsebene gelegt. Hierbei wurde zuerst die MEMS-Score-Anwendung vorgestellt, welche es Nutzer:Innen eines Smartphones auf einfache Weise ermöglicht, die Qualität der in dem Gerät verbauten Inertialsensoren sowie deren Sensordatenverarbeitung zu bewerten. Die Anwendung erlaubt es auch, die Qualität der Sensoren des eigenen Smartphones mit anderen Smartphones zu vergleichen. Für Entwickler:Innen biete die MEMS-Score-Anwendung die Möglichkeit, zu überprüfen, ob die verbauten Sensoren des Smartphones gut genug sind, um in einer Anwendung genutzt werden zu können. Die Testergebnisse der MEMS-Score-Anwendung werden in einer Datenbank gespeichert, die es auf lange Sicht erlauben soll, das Spektrum der Qualitäten der verbauten Bewegungssensoren zu erfassen. Die bisher gesammelten Ergebnisse aus der MEMS-Score-Anwendung reichen aktuell jedoch noch nicht aus, um einen guten Überblick über die Verteilung der Qualität der Sensordaten zu erlangen. Es hat sich gezeigt, dass nur wenige Nutzer:Innen alle notwendigen Tests durchführen, um eine abschließende Bewertung der Smartphones durchführen zu können.

Weiterhin wurden Erweiterungen für die MEMS-Score-Anwendung vorgestellt, welche es ermöglichen sollen, weitere Sensoren der Smartphones in die Bewertung einzubeziehen. Hierfür wurde ein Test für die absolute Orientierung vorgestellt, der mittels des Prinzips der Peilung eine Bewertung erlaubt. Dieser Test ist jedoch noch nicht öffentlich verfügbar.

Zuletzt wurde in diesem Kapitel ein Verfahren für die Bewertung von digitalen Zeigegeräten und insbesondere Smartpens vorgestellt. Es wurde gezeigt, anhand welcher Kriterien es möglich ist, anhand einer qualitativen Bewertung Rückschlüsse auf die subjektive Bewertung von Nutzer:Innen zu ziehen. Hierbei wurde ebenfalls darauf geachtet, dass für die Durchführung der Text kein externes Equipment benötigt wird, sodass die beschriebenen Tests auch von Dritten einfach durchgeführt werden können. Es wurden mehrere Kriterien, wie Reaktionsfähigkeit, Genauigkeit und Handhabbarkeit ermittelt und Testverfahren speziell für diese Kriterien entworfen und implementiert. Für die Verifizierung der entwickelten Testverfahren wurden Untersuchungen angestellt, bei der Testpersonen Tests mit unterschiedlichen digitalen Zeigegeräten durchführen und eine subjektive Bewertung der Zeigegeräte abgeben sollen. Die Ergebnisse zeigen, dass die qualitative Bewertung der digitalen Zeigegeräte durch die Tests der subjektiven Bewertung der Testpersonen entspricht. Es hat sich jedoch auch offenbart, dass die Granularität der subjektiven Bewertung für die Zukunft überarbeitet werden sollte, um die Aussagekraft der Ergebnisse noch zu erhöhen.

5 Methoden und Untersuchungen auf Firmware-Ebene

Die Methoden zur Untersuchung der Eigenschaften von Sensorsubsystemen auf Anwendungsebene, die in Kapitel 4 vorgestellt wurden, erlauben es, einen guten Überblick über die Qualität der Sensordaten bestehender Systeme zu erhalten. Mit ihnen ist es möglich, Systeme direkt und ohne viel Aufwand miteinander zu vergleichen. Dies kann besonders aus Sicht eines Anwenders interessant sein. Die Bewertungsmethode für den Smartpen kann jedoch auch während der Entwicklung eines solchen Gerätes genutzt werden, um dessen Handhabbarkeit zu überprüfen und diese mit anderen Geräten zu vergleichen. Die Möglichkeit, die Sensorsubsysteme und Sensordatenverarbeitung während der Implementierung beurteilen zu können, ist essenziell, um eine hohe Qualität auf der Anwendungsebene zu erreichen. Daher werden in diesem Kapitel Untersuchungen und Methoden vorgestellt, die sich mit der Bewertung von Sensorfusionsalgorithmen und Sensordaten auf der Firmware-Ebene befassen oder die Bewertung vereinfachen sollen.

Die in diesem Kapitel durchgeführten Untersuchungen erweitern oder verwenden in Teilen die Sensor-in-the-Loop-Architektur, die von Herrn Daniel Gis entwickelt und an der mitgewirkt wurde. Die Ergebnisse wurden 2020 in dem *International Workshop of Rapid System Prototyping* vorgestellt [B5]. Weiterführend Untersuchungen wurden 2021 im *MDPI Sensor Journal* veröffentlicht [B6]. Die Funktionsweise der SiL-Architektur wird in Abschnitt 5.1 vorgestellt. Weitere Grundlagen, die zum Verständnis dieses Kapitels notwendig sind, werden in Kapitel 3 erläutert.

Die Untersuchungen, die in diesem Kapitel durchgeführt wurden, befassen sich mit den funktionalen und extrafunktionalen Eigenschaften von Sensorfusionsalgorithmen für die Bestimmung der Orientierung, welche auch in Abschnitt 4.1 Teil der Untersuchung sind. 2021 wurden die Ergebnisse dieser Untersuchungen im *MDPI Sensor Journal* veröffentlicht [B8].

Neben der Untersuchung der Eigenschaften der Sensorfusionsalgorithmen wird in Abschnitt 5.3 ein Framework vorgestellt, das auf der SiL-Architektur aufbaut und es erlaubt, eine Manipulation sowie Erweiterung der Sensordaten durchzuführen. Es kann dazu verwendet werden, Sensorfusionsalgorithmen robuster zu entwickeln, zu testen oder auch, um Algorithmen zur Gestenerkennung zu trainieren. Das Framework wurde 2021 auf dem *International Workshop of Rapid System Prototyping* vorgestellt.

5.1 Sensor-in-the-Loop

Die in diesem Kapitel beschriebenen Arbeiten nutzen die Sensor-in-the-Loop-Architektur, welche von Herrn Daniel Gis entwickelt und 2020 auf dem *International Workshop on Rapid System Prototyping (RSP)* vorgestellt wurde [B5]. Ein Beitrag zu dieser Arbeit wurde in Form von der Implementierung einer GUI für die Betrachtung sowie Interaktion mit der Architektur sowie der Durchführung von Tests geleistet. Eine erweiterte Version der Arbeit mit einer genaueren Untersuchung des Zeitverhaltens wurde 2021 im *MDPI Sensors Journal* veröffentlicht [B6]. Auch hier wurde in bei der Durchführung von Test und Auswertung der statistischen Analyse mitgewirkt.

Bei der SiL-Architektur handelt es sich um eine Plattform oder Debugging-Architektur, die es erlaubt, Inertialsensordaten aus einem Smart-Sensor auszulesen, abzuspeichern und später wiederzuverwenden. Die Besonderheit liegt darin, dass die Wiederverwendung der Daten direkt auf der zu testenden Hardware geschieht. Hierfür werden die vorher aufgezeichneten Daten der Inertialsensoren über eine Kommunikationsschnittstelle wieder in den Smart-Sensor eingespeist. Dieser behandelt die erhaltenen Daten nun so, als hätte er sie direkt von den Sensoren und nicht von der Kommunikationsschnittstelle erhalten. Dies ermöglicht es, die Firmware für den Smart-Sensor direkt auf der Hardware zu testen. Im Gegensatz zu der direkten Verwendung von neuen Sensordaten sind die Ergebnisse der Tests jedoch reproduzierbar und wiederholbar, da die exakt gleichen Daten verwendet werden können. Dies ermöglicht eine deutlich gezieltere und zuverlässigere Entwicklung und Evaluation der Sensor Firmware. Der Datenfluss für das Aufzeichnen und Injizieren der Daten ist in Abbildung 5.1 dargestellt.

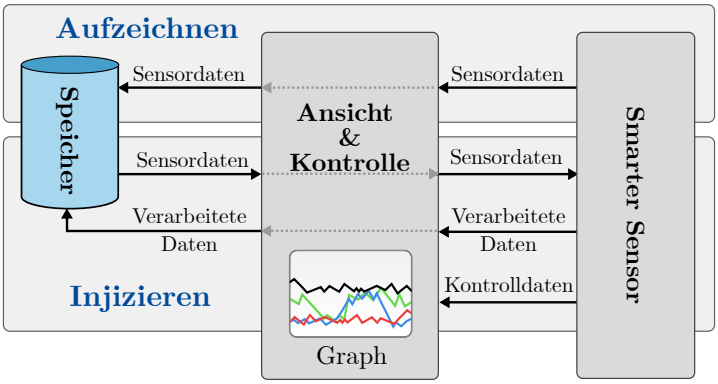


Abbildung 5.1: Datenfluss für das Aufzeichnen und Injizieren der Daten mit der Sensor-in-the-Loop-Plattform

Mittels der SiL-Plattform lässt sich auch zuverlässig die Rechenlast unterschiedlicher Implementierungen und Algorithmen gut miteinander vergleichen. Dieser Umstand wird im folgenden Abschnitt genutzt, um die extrafunktionalen Eigenschaften der betrachteten Algorithmen zu untersuchen.

5.2 Funktionale und extrafunktionale Eigenschaften

In diesem Abschnitt wird die Untersuchung der funktionalen sowie extrafunktionalen Eigenschaften der in Unterabschnitt 3.5.2 beschriebenen Sensorfusionsalgorithmen vorgestellt. Das Ziel dieser Untersuchung ist es, in Erfahrung zu bringen, welche der vorgestellten Sensorfusionsalgorithmen sich besonders gut für ein ressourcenbeschränktes eingebettetes System eignen. Hierbei wird auch untersucht, wie gut die Algorithmen mit anderen Datentypen als Gleitkommazahlen mit einfacher Genauigkeit funktionieren. Gerade wenn hohe Datenraten benötigt werden oder möglichst viel Rechenaufwand eingespart werden soll, ohne die Qualität der Ergebnisse zu stark zu beeinträchtigen, können hier Festkommazahlen eine valide Alternative darstellen. Für diese Untersuchung werden 16 Bit-Festkommazahlen und 32 Bit-Festkommazahlen verwendet. Die vorgestellten Algorithmen gibt es in zwei Varianten, von denen beide untersucht werden. In der ersten Variante werden nur Beschleunigungssensor und Gyroskop verwendet, was eine Bestimmung der Orientierung erlaubt. Die vertikale Achse kann jedoch nicht korrigiert werden. In der zweiten Variante wird zusätzliche das Magnetometer verwendet, was eine absolute Orientierungsbestimmung ermöglicht. Die vertikale Achse wird mittels Magnetometer korrigiert und ist an den Polen des Erdmagnetfelds ausgerichtet.

Die in diesem Abschnitt untersuchten extrafunktionalen Eigenschaften umfassen Codegröße sowie Ausführungsdauer. Die funktionalen Eigenschaften, also die Genauigkeit der Sensorfusionsalgorithmen, verwenden eine Referenz für den Vergleich der Ausgabe der Fusionsalgorithmen. Die verwendete Referenz wird auf dieselbe Weise ermittelt wie bei der in Abschnitt 4.1 vorgestellten Bewertungsmethode für Smartphones. Jedoch ist in diesem Fall die Kamera starr und das Muster wird bewegt.

5.2.1 Genutzte Hardware

Die Untersuchung der funktionalen und extrafunktionalen Eigenschaften wurde auf der Sensorhardware durchgeführt, um möglichst realitätsnahe Ergebnisse zu erhalten. Der hierfür verwendete intelligente Sensor ist der BMF055 von Bosch Sensortec [A13]. Der BMF055 enthält einen SAM D20 Mikrocontroller mit einem ARM Cortex M0+ Rechenkern von Microchip Technology (ehemals Atmel). Dieser auf geringen Energieverbrauch optimierte Mikrocontroller taktet mit einer Frequenz von 48 MHz und besteht aus einer 32-Bit Architektur mit zwei Pipeline-Stufen [A119]. Für die korrekte Einordnung der Ergebnisse für die extrafunktionalen Eigenschaften sind folgende Eigenschaften des SAM D20 relevant:

- Der SAM D20 besitzt eine arithmetisch-logische Einheit (Arithmetic Logic Unit, ALU), bei der eine Multiplikationen von 32 Bit Integer-Zahlen in einem Taktzyklus durchgeführt wird. Daher brauchen Addition und Multiplikation gleich lang.

- Der SAM D20 hat keine Hardware-Unterstützung für Fließkommaarithmetik, besitzt also keine Gleitkommaeinheit (Floating Point Unit, FPU). Mathematische Operationen für Fließkommazahlen müssen daher in der Software emuliert werden. Dies wird automatisch vom Compiler vorgenommen.

Das Gyroskop, das im BMF055 Verwendung findet, ist baugleich mit dem BMI055 mit einem Messbereich von ±2000 °/s und einer Datenbreite von 16-Bit [A13],[A120]. Der Beschleunigungssensor des BMF055 ist baugleich mit dem BMA280 mit einem Messbereich von ±16 g und einer Datenbreite von 14-Bit [A93], [A13]. Das Magnetometer ist baugleich mit dem BMM150 mit einer Auflösung von 0,3 µT und einer Datenbreite von 13 Bit [A13], [A121]. Der µC des SAM D20 benötigt im Ruhezustand einen Versorgungsstrom von 1 mA und bis zu 5,9 mA unter Volllast. Der BMF055 als Ganzes hat einen Stromverbrauch vom 13,7 mA, wenn alle Sensoren im normalen Modus aktiv sind und mit einer Datenrate von 100 Hz messen. Wenn der Energiesparmodus genutzt wird, liegt der Verbrauch mit 100 Hz-Datenrate bei 2,6 mA [A13].

5.2.2 Extrafunktionale Eigenschaften

Ein Hauptaspekt der Sensorfusionsalgorithmen in dieser Untersuchung sind die extrafunktionalen Eigenschaften *Codegröße* und *Rechenaufwand*. Sie bestimmen, im Besonderen bei einem eingebetteten System, wie hoch der Energieverbrauch ist, wie schnell die Daten verarbeitet werden können oder wie viel anderer Programmcode noch auf dem System implementiert werden kann. In diesem Abschnitt werden daher die extrafunktionalen Eigenschaften der vier genannten Sensorfusionsalgorithmen in Kombination mit den drei unterschiedlichen Datenformaten betrachtet. Bevor die Fusionsalgorithmen untersucht wurden, wurde sichergestellt, dass diese in der implementierten Konfiguration funktionsfähig sind und empirisch korrekte Ergebnisse liefern. Eine genaue Analyse der funktionalen Eigenschaften, als der Qualität der Ergebnisse der Sensordatenfusion, wird anschließend in Unterabschnitt 5.2.3 vorgestellt. Die Untersuchungen wurden auf der in Unterabschnitt 5.2.1 vorgestellten Hardware durchgeführt. Die Kompilierung des Programmcodes wurde mit dem GNU C Compiler vorgenommen. Für die Untersuchungen wurden die zwei Compiler-Optimierungsoptionen *-O0* und *-O3* verwendet.

5.2.2.1 Codegröße

Um die Codegröße der Programme zu ermitteln, wurde das *Size Tool* aus der GNU ARM Embedded Toolchain verwendet. Die Ergebnisse umfassen die Größe der Binärdatei im ROM sowie Speichergröße im RAM für Variablen und Funktionsstack. Da keiner der verwendeten Algorithmen dynamische Speicherallokationen oder rekursive Funktionsaufrufe verwendet, entspricht die Größe, die mit dem *Size Tool* ermittelt wurde, der

maximalen Größe im RAM des µC. Die Codegröße wurde für jede Kombination aus Sensorfusionsalgorithmus und Datenformat durchgeführt. In diesem Abschnitt werden die Tabellen und Abbildungen für die Ergebnisse mit Compiler-Optimierung (-O3) gezeigten. Die Ergebnisse der Kompilierung ohne Optimierung (-O0) sind im Anhang in den Tabellen D.13 bis D.16 zu finden. Wenn es signifikante Unterschiede zwischen den Varianten gibt, werden diese hier mit Verweis auf die Ergebnisse im Anhang genannt. Bei den ermittelten Größen handelt es sich um die reine Größe der Algorithmen. Der Speicherverbrauch der Firmware von 19,1 kB wurde bereits abgezogen.

Größen der Binärdatei

Die ermittelten Größen der Binärdateien sind in Tabelle 5.1 für Sensorfusionsalgorithmen ohne Magnetometer aufgeführt. Für eine bessere Vergleichbarkeit der Größen werden diese als Graph in Abbildung 5.2 dargestellt. Die jeweiligen Äquivalente für die Sensorfusionsalgorithmen mit Magnetometer sind in Tabelle 5.2 und Abbildung 5.3 zu finden.

Tabelle 5.1: Größe der Binärdatei in Byte im ROM ohne Nutzung des Magnetometers

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	11360	11840	10872
Madgwick-Filter	8128	5956	6176
Mahony-Filter	7852	6524	5988
Komplementärfilter	7424	7360	6276

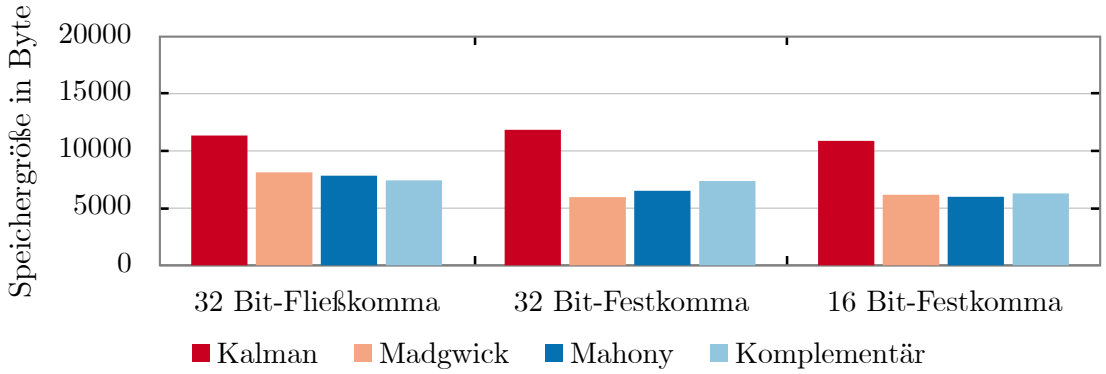


Abbildung 5.2: Größe der Binärdateien im ROM ohne Nutzung des Magnetometers

Die Größe der Algorithmen liegt etwa zwischen 5 kB und 15 kB. Relativ zur Gesamtgröße des Flash Speichers vom SAMD20 Mikrocontroller von 256 kB belegen die Algorithmen also zwischen 2,0 % und 5,9 %. Es ist zu erkennen, dass es im Mittel nur geringe Unterschiede zwischen den Datenformaten gibt.

Tabelle 5.2: Größe der Binärdatei in Byte im ROM mit Nutzung des Magnetometers

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	12676	13576	11592
Madgwick-Filter	9208	8308	7308
Mahony-Filter	8580	7476	6172
Komplementärfilter	8716	9100	7000

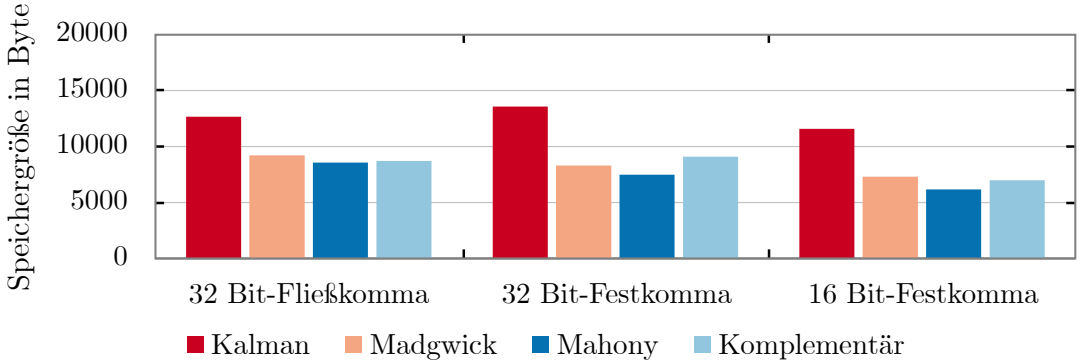


Abbildung 5.3: Größe der Binärdateien im ROM mit Nutzung des Magnetometers

Die Algorithmen mit den 16 Bit-Festkommazahlen haben meist den geringsten Speicherverbrauch. Ohne Verwendung des Magnetometers benötigen die Varianten mit 32 Bit-Fließkommazahlen den meisten Speicher. Wenn das Magnetometer verwendet wird, ist mal die Variante mit 32 Bit-Fließkommazahlen am größten, mal die Variante mit 32 Bit-Festkommazahlen. Der größte erkennbare Unterschied ist zwischen dem Kalman-Filter und den anderen Filtern. Dies ist primär auf die Verwendung von Matrizen und Matrixoperationen zurückzuführen, welche beim Kalman-Filter notwendig sind. Weiterhin werden auch mehr statische Werte benötigt, welche auch im ROM abgespeichert werden müssen. Bei der Version mit Magnetometer sind die Unterschiede zwischen den Algorithmen bei der Implementierung mit 32 Bit-Festkommazahlen deutlicher als bei den Implementierungen ohne Magnetometer. Der Unterschied fällt besonders zwischen Kalman-Filter und Madgwick-Filter geringer aus.

Benötigter Arbeitsspeicher

Der Speicherverbrauch der Sensorfusionsalgorithmen im RAM setzt sich zusammen aus Konstanten, Variablen für Werte und Zwischenergebnisse sowie dem Funktionsstack. Der Arbeitsspeicher, der für die jeweiligen Algorithmen benötigt wird, ist für die Variante ohne Magnetometer in Tabelle 5.3 und Abbildung 5.4 aufgezeigt. Die jeweiligen Äquivalente für die Sensorfusionsalgorithmen mit Magnetometer sind in Tabelle 5.4 zu finden. Auf die grafische Darstellung der Werte aus Tabelle 5.4 wurde verzichtet, da die Werte nahezu identisch mit denen aus Tabelle 5.3 sind.

Tabelle 5.3: Benötigter Arbeitsspeicher in Byte ohne Verwendung des Magnetometers

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	3272	3432	3368
Madgwick-Filter	3184	3344	3328
Mahony-Filter	3200	3360	3336
Komplementärfilter	3184	3344	3320

Tabelle 5.4: Benötigter Arbeitsspeicher in Byte mit Verwendung des Magnetometers

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	3280	3440	3376
Madgwick-Filter	3192	3352	3336
Mahony-Filter	3208	3368	3344
Komplementärfilter	3192	3352	3336

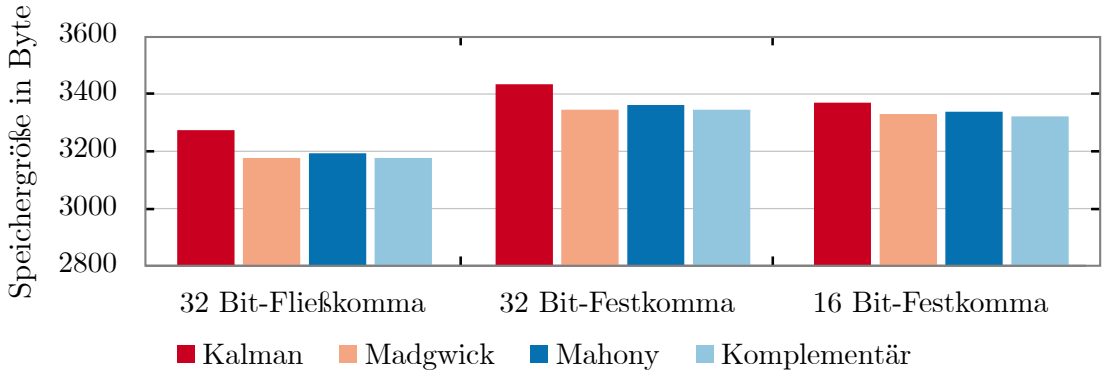


Abbildung 5.4: Benötigter Arbeitsspeicher ohne Verwendung des Magnetometers

Beim benötigten Arbeitsspeicher ist zu sehen, dass das Kalman-Filter auch hier mehr Arbeitsspeicher als die anderen Filter benötigt. Der Unterschied ist jedoch deutlich geringer als beim Festspeicher. Der geringere Unterschied ist damit zu erklären, dass Madgwick-, Mahony- und Komplementärfilter für eine möglichst geringe Ausführungszeit optimiert sind und viele Zwischenergebnisse gespeichert werden, um mehrfache Berechnungen zu vermeiden.

Deutlich zu erkennen ist, dass die Verwendung von Festkommazahlen zu einem höheren Speicherverbrauch führt, der sich zwischen 16 Bit und 32 Bit nur gering unterscheidet. Es wurde ursprünglich erwartet, dass der benötigte Arbeitsspeicher bei der Verwendung der Fließkommazahlen größer ausfallen würde, da hierfür mehr Operationen und Zwischenergebnisse notwendig sind. Es wurden mehrere Gründe ermittelt, die einen

höheren Speicherverbrauch der Algorithmen mit Festkommazahlen verursachen. Zuerst sind zusätzliche Methodenaufrufe für die Festkommaarithmetik notwendig, welche auch eine Rundung bei Multiplikation und Division durchführen. Bei Division und Multiplikation muss zudem das nächstgrößere Datenformat verwendet werden, um einen Überlauf zu verhindern, dies erfordert mehr Speicher und auch mehr Rechenoperationen. Zusätzlich wird für die Wurzelfunktion eine Lookup-Tabelle benötigt, welcher bei den Fließkommazahlen nicht benötigt wird. Es ist weiterhin anzunehmen, dass die Software-Emulation der Fließkommaarithmetik ein hoch optimierter Assemblercode ist, was den benötigten Speicherverbrauch stark reduzieren kann.

Der Unterschied ist insgesamt recht gering. Die Variante mit Festkommazahlen braucht im Schnitt 5 % mehr Arbeitsspeicher für 32 Bit-Festkommazahlen und 4 % mehr Arbeitsspeicher für 16 Bit-Festkommazahlen, verglichen mit der Fließkommaarithmetik.

5.2.2.2 Ausführungszeit

Die wichtigste extrafunktionale Eigenschaft der Sensorfusionsalgorithmen ist deren Ausführungszeit. Dies spiegelt den Rechenaufwand wider, der dafür benötigt wird, die Orientierung anhand der neuen Daten von den Sensoren zu aktualisieren. Der Rechenaufwand bestimmt hierbei sowohl die maximale Datenrate, mit der die Orientierung aktualisiert werden kann, als auch die Höhe des Energieverbrauchs des Systems.

Die Messung der Ausführungszeit der Algorithmen wurde auf dem SAMD20 µC durchgeführt. Hierfür wurden vorher aufgezeichnete Daten verwendet, welche mittels der SiL-Plattform in die Sensorhardware eingespeist wurden, um die Ausführungszeiten bestmöglich vergleichen zu können. Für die eigentliche Messung der Ausführungszeit wurde ein Oszilloskop an einen Pin des Mikrocontrollers angeschlossen. Der Aufbau des Messvorgangs ist in Abbildung 5.5 dargestellt.

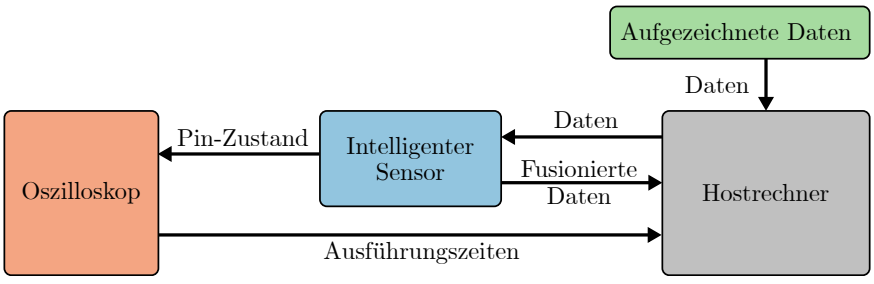


Abbildung 5.5: Messaufbau zum Ermitteln der Ausführungszeit der Fusionsalgorithmen

Sobald ein Sensorwert beim µC ankommt, wird direkt vor der Ausführung des Sensorfusionsalgorithmus der angeschlossene Pin gesetzt. Direkt nach der Rückkehr aus der Funktion wird der Pin wieder zurückgesetzt. Mittels des hochfrequenten Oszilloskops

lässt sich so die Ausführungszeit ermitteln. Bevor die Messungen durchgeführt worden sind, wurde die Zeit ermittelt, die zum Umschalten des Pins benötigt wird, um diesen Offset aus den Messungen herauszurechnen. Die Ergebnisse der Laufzeitmessungen ohne Magnetometer sind in Tabelle 5.5 und Abbildung 5.6 dargestellt. Die Ergebnisse gelten für die Firmware mit Compiler-Optimierung (*-O3*) dargestellt. Für die Firmware ohne Optimierung sind die Ergebnisse im Anhang in den Tabellen D.17 und D.18 zu finden.

Tabelle 5.5: Mittlere Ausführungszeit der Fusionsalgorithmen ohne Magnetometer

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	2,698 ms	1,492 ms	0,305 ms
Madgwick-Filter	0,762 ms	0,283 ms	0,104 ms
Mahony-Filter	0,682 ms	0,207 ms	0,105 ms
Komplementärfilter	0,666 ms	0,226 ms	0,114 ms

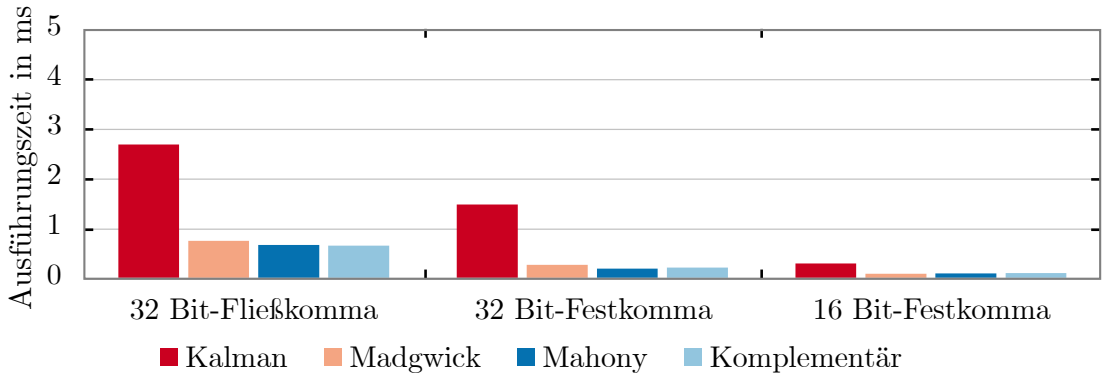


Abbildung 5.6: Mittlere Ausführungszeit der Fusionsalgorithmen ohne Magnetometer

Es ist klar erkennbar, dass das Kalman-Filter mit Abstand die meiste Rechenzeit in allen Datenformaten benötigt. Weiterhin ist offensichtlich, dass die Verwendung der Festkommaarithmetik zu einer deutlichen Verringerung der Rechenzeit führt. Die Berechnungen mit 32 Bit-Festkommazahlen benötigt etwa die Hälfte der Zeit, die bei Fließkommaarithmetik benötigt wird. Für die Berechnungen mit 16 Bit-Festkommazahlen wird sogar nur etwa ein Sechstel der Zeit benötigen. Vergleichbare Ergebnisse zeigen sich auch bei den Filtervarianten mit Magnetometer, welche in Tabelle 5.6 und Abbildung 5.7 zu sehen sind. Die generelle Rechenzeit ist etwas höher. Es ist jedoch deutlich zu erkennen, dass das Datenformat eine signifikante Auswirkung auf die Rechenzeit hat. Ebenfalls ist zu erkennen, dass das Kalman-Filter die mit Abstand meiste Rechenzeit benötigt. Vergleicht man die benötigte Ausführungszeit der Variante mit Magnetometer mit der Variante ohne Magnetometer, so ist eine Erhöhung der Rechenzeit von ca. 30 % zu erkennen. Die relativen Verhältnisse der benötigten Rechenzeit zwischen den Algorithmen und Datenformaten bleiben sehr ähnlich. Lediglich das Madgwick-Filter benötigt, vergleichen mit

Mahony-Filter und Komplementärfilter, etwas mehr Rechenzeit.

Tabelle 5.6: Mittlere Ausführungszeit der Fusionsalgorithmen mit Magnetometer

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	3,963 ms	1,923 ms	0,621 ms
Madgwick-Filter	1,142 ms	0,560 ms	0,166 ms
Mahony-Filter	0,758 ms	0,350 ms	0,121 ms
Komplementärfilter	0,782 ms	0,382 ms	0,123 ms

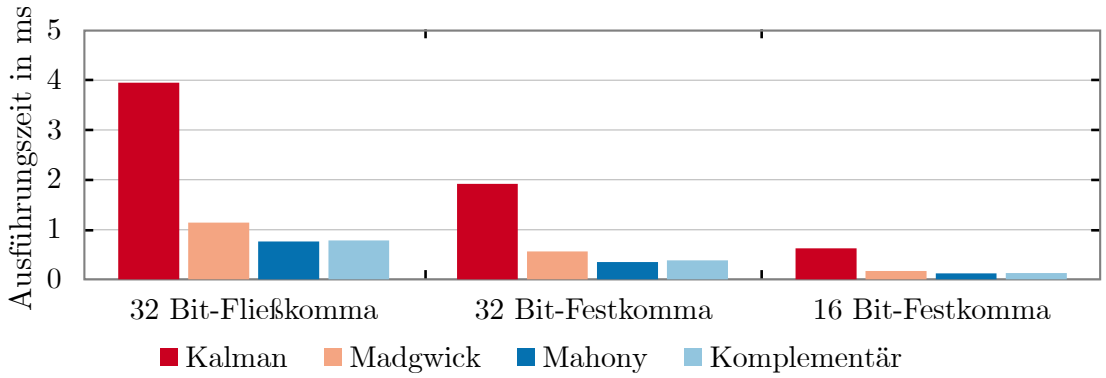


Abbildung 5.7: Mittlere Ausführungszeit der Fusionsalgorithmen mit Magnetometer

Verglichen mit den anderen Filtern benötigt das Kalman-Filter mindestens 3,4-mal länger als die anderen Filter. Dies hat besonders bei der Variante mit Fließkommaarithmetik gravierende Auswirkungen, bei der das Kalman-Filter 3,96 ms für jede Aktualisierung benötigt. Dies limitiert die maximale Datenrate auf etwa 250 Hz. Wenn berücksichtigt wird, dass auch noch zusätzliche Aufgaben oder Kontrollfunktionen auf dem µC ausgeführt werden müssen, ist die tatsächlich mögliche Datenrate noch geringer.

Varianz der Ausführungszeit

Zusätzlich zu der Betrachtung der mittleren Ausführungszeit wurde auch untersucht, wie gleichmäßig die Ausführungszeiten sind. Hierfür wurde die Standardabweichung der Ausführungszeiten ermittelt, welche in Abbildung 5.8 aufgezeigt ist. Generell ist zu sehen, dass die Standardabweichung recht gering ist und im Mittel bei 0,015 ms liegt.

Dieses Verhalten war zu erwarten, da der Kontrollfluss in keinem der verwendeten Filter datenabhängig ist, was die Anzahl der ausgeführten Instruktionen pro Aktualisierungsschritt nahezu konstant hält. Eine Ausnahme bei den Standardabweichungen stellt das Kalman-Filter in der Variante mit Fließkommaarithmetik dar. Hier ist die Standardabweichung mit etwa 0,025 ms um etwa 40 % höher. Für eine genauere Analyse wurde

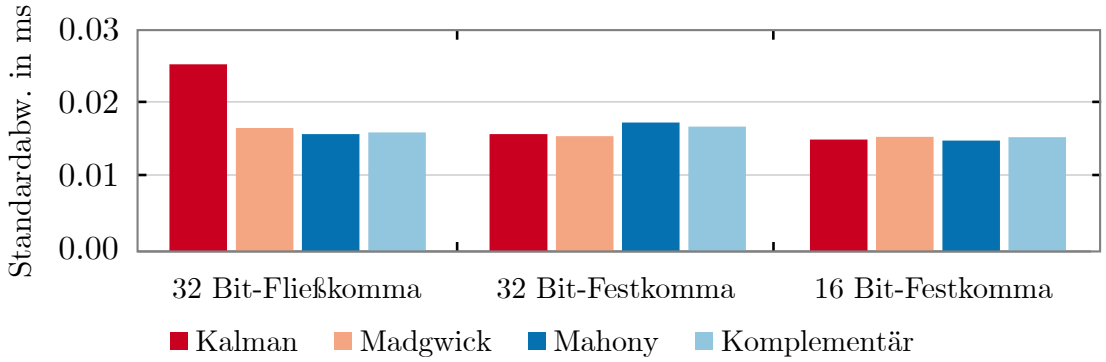


Abbildung 5.8: Standardabweichung der Ausführungszeiten der Algorithmen

für die Varianten mit Fließkommaarithmetik ein Box-Plot angefertigt, welcher in Abbildung 5.9 zu finden ist.

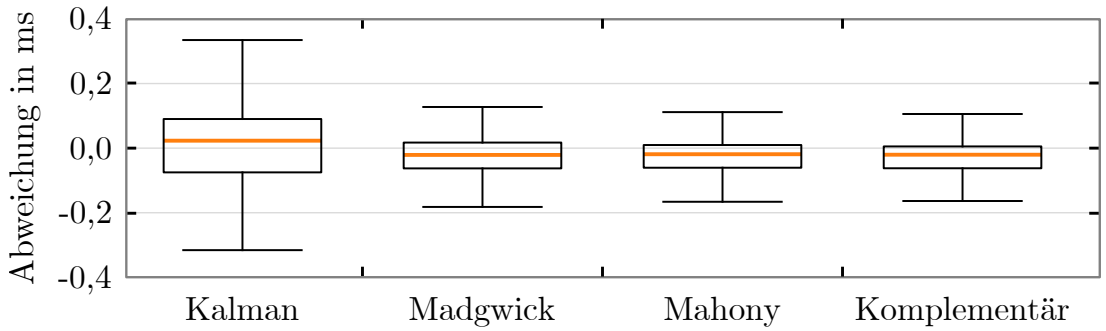


Abbildung 5.9: Verteilung der Ausführungszeiten der Algorithmen

Der Grund für diesen Ausreißer ist die Kombination aus zwei Faktoren. Erstens werden die arithmetischen Operationen bei den Fließkommazahlen in Software ausgeführt, da der SAMD20 keine FPU besitzt. Bei mathematischen Operationen, beispielsweise einer Multiplikation oder Division, ist die Anzahl der benötigten Instruktionen datenabhängig. Dies führt zu einer generell höheren Varianz in den Ausführungszeiten. Dieser Effekt zeigt sich jedoch nur verstärkt beim Kalman-Filter. Dies liegt an dem zweiten Faktor. Der mögliche Wertebereich der Zahlen ist beim Madgwick-Filter, Mahony-Filter und Komplementärfilter recht klein. Durch die Verwendung von Matrizen ist der Wertebereich beim Kalman-Filter deutlich größer, wodurch die Anzahl der Operationen für eine mathematische Operation stärker schwankt. Weiterhin werden beim Kalman-Filter deutlich mehr mathematische Operationen benötigt, was diesen Effekt noch verstärkt.

5.2.3 Funktionale Eigenschaften

Neben den extrafunktionalen Eigenschaften der Sensorfusionsalgorithmen müssen auch die funktionalen Eigenschaften von diesen betrachtet werden. Besonders wichtig ist hier zu untersuchen, wie sich die unterschiedlichen Filter in Kombination mit den genutzten Datenformaten verhalten und ob es signifikante Unterschiede in den Ergebnissen gibt.

Für die Evaluation der Sensorfusionsalgorithmen wurden Messungen mit dem BMF055 auf einem Shuttle-Board durchgeführt. Als Referenz zur Bewertung wurde dasselbe Konzept verwendet, welches in Abschnitt 3.8 vorgestellt worden ist. Im Gegensatz zu den Messungen in Abschnitt 4.1 wird bei diesen Messungen jedoch das Muster bewegt, welches starr an dem Sensor befestigt ist. Das Smartphone mit der Kamera ist stationär. Die GUI des Smartphones wurde genutzt, um die von einem Menschen ausgeführten Bewegungen bei allen Messungen möglichst vergleichbar in Auslenkung und Dauer zu gestalten. Es wurden jedoch Bewegungen in drei Geschwindigkeiten durchgeführt. Die mittlere Drehgeschwindigkeit betrug bei der langsamen Bewegung etwa 7,5 °/s, bei der mittelschnellen Bewegung etwa 15 °/s und bei der schnellen Bewegung 30 °/s. Aufgrund der Ausführung der Trajektorien durch einen Menschen ist es möglich, eine Aussage darüber zu erhalten, wie sich die Sensorfusionsalgorithmen bei menschlichen Bewegungen verhalten. Durch die Vorgabe der Bewegung ist eine statistische Analyse der Messungen möglich. Insgesamt wurden jeweils zehn Messungen für die drei Geschwindigkeiten durchgeführt. Kombiniert mit den vier Sensorfusionsalgorithmen und den drei Datenformaten ergibt das insgesamt 360 Messungen. Dadurch, dass die Sensorfusionsalgorithmen sowohl mit als auch ohne Magnetometer untersucht werden, besteht der Datensatz der Messungen aus insgesamt 720 Einzelergebnissen.

5.2.3.1 Ergebnisse ohne Verwendung des Magnetometers

Bei dem Vergleich der Sensorfusionsalgorithmen ohne Verwendung des Magnetometers wurden zuerst die Ergebnisse direkt miteinander vergleichen. Hier zeigte sich jedoch, dass die Daten eine deutliche Abweichung von der Referenz haben, die durch den Drift der z-Achse zustande kommt, da z-Achse ohne Magnetometer nicht korrigiert werden kann. Abbildung 5.10 zeigt die Ergebnisse. Im Vergleich mit den Ergebnissen der Untersuchung mit Verwendung des Magnetometers in Unterunterabschnitt 5.2.3.2 ist die Varianz der Messungen deutlich höher.

Es ist zu erkennen, dass der Winkelfehler mit 4,14 ° für Fließkommazahlen, 3,69 ° für die 32 Bit-Festkommazahlen und 4,90 ° für die 16 Bit-Festkommazahlen verhältnismäßig hoch ist. Zusätzlich zeigt das Kalman-Filter einen deutlich höheren Fehler in der 16 Bit-Festkomma-Variante. Der Grund hierfür wird in Unterunterabschnitt 5.2.3.3 näher erläutert.

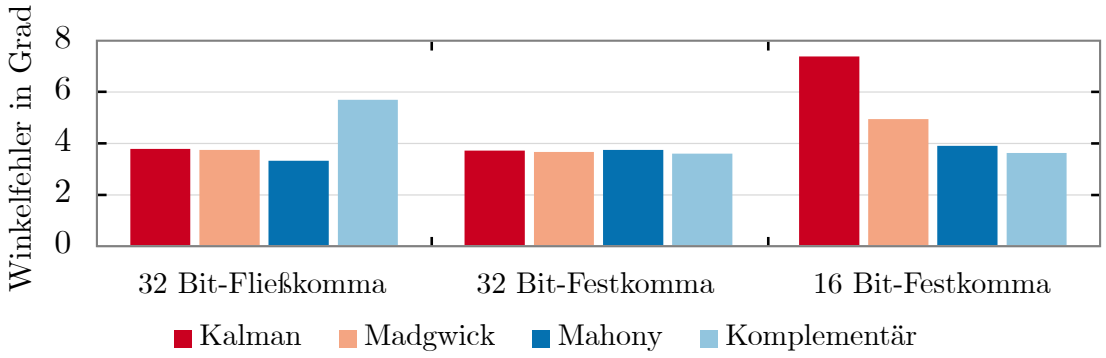


Abbildung 5.10: Mittlerer Winkelfehler beeinflusst vom Drift auf der z-Achse

Wie bereits erwähnt, ist der Fehler darin begründet, dass ohne Magnetometer keine Korrektur auf der z-Achse vorgenommen werden kann. Dies führt dazu, dass es zu einem Drift auf der z-Achse kommt. Dieser fällt je nach Messung und Filter teils sehr unterschiedlich aus. Beispielhaft ist der Drift in Abbildung 5.11 veranschaulicht. Die durchgezogenen Linien repräsentieren die Orientierung, die von der Referenz stammt. Die gestrichelten Linien zeigen das Ergebnis einer Sensordatenfusion. Es ist erkennbar, dass sich die Werte auf der z-Achse über die Zeit immer weiter von der Referenz entfernen.

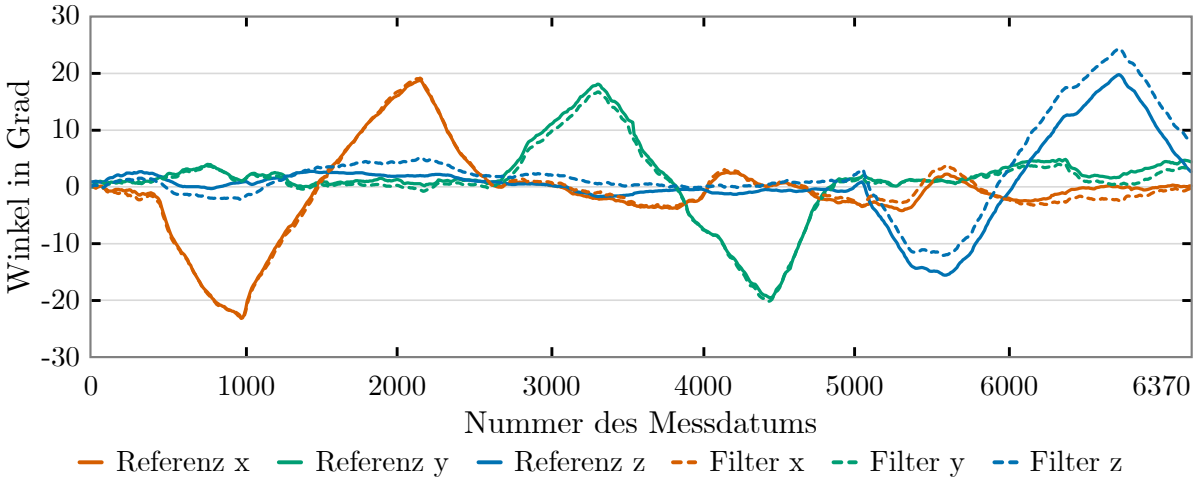


Abbildung 5.11: Beispiel einer Messung mit Drift auf der z-Achse (in Euler-Winkeln)

Da die Filter jedoch keinen direkten Einfluss auf den Drift der z-Achse haben können, führt dies zu einem verzerrten Ergebnis, da der Drift das Ergebnis mitbestimmt. Aus diesem Grund wurde die z-Achse aus der Bewertung entfernt bzw. gemäß der Referenz korrigiert. Dadurch werden für die Bewertung der Orientierung nur die Achsen verwendet, welche auch von den Sensorfusionsalgorithmen korrigiert werden können.

Die Ergebnisse der vorläufigen Untersuchung mit Korrektur der z-Achse ist in Abbildung 5.12 zu sehen. Er wird deutlich, dass der mittlere Fehler mit Ausnahme des 16 Bit-Kalman-Filters nun deutlich unter 2 ° liegt. Zudem ist zu erkennen, dass sich die Fehler zwischen den verwendeten Fusionsalgorithmen sowie zwischen den Datenformaten nur noch geringfügig unterscheiden. Das Kalman-Filter zeigt in der 16 Bit-Festkomma-Variante jedoch weiterhin einen deutlich größeren Fehler als alle anderen Varianten.

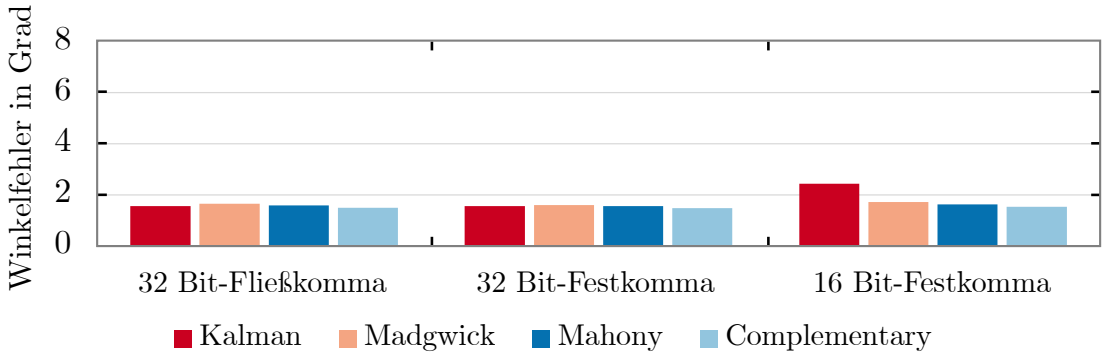


Abbildung 5.12: Mittlerer Winkelfehler nach der Korrektur des Drifts

In Tabelle 5.7 sind die genauen Mittelwerte aus 15 Messungen dargestellt. Hier ist zu erkennen, dass die Fehler mit 32 Bit-Festkommazahlen sogar geringfügig kleiner sind als die Fehler mit 32 Bit-Fließkommazahlen. Auch die Fehler der Algorithmen mit 16 Bit-Festkommazahlen sind nur geringfügig größer.

Tabelle 5.7: Mittlerer Winkelfehler bei 15 Messungen mit Korrektur der z-Achse

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	1,557 °	1,562 °	2,429 °
Madgwick-Filter	1,657 °	1,603 °	1,722 °
Mahony-Filter	1,588 °	1,557 °	1,626 °
Komplementärfilter	1,489 °	1,478 °	1,539 °

Im Anschluss an die erste oberflächliche Untersuchung wurde eine statistische Analyse der 360 Messungen vorgenommen. Es wurde eine dreifaktorielle Varianzanalyse mit anschließender Varianzkomponentenschätzung vorgenommen. Die Ergebnisse der Messungen sind im Anhang in Tabelle D.19 zu finden. Aufbauend auf den Ergebnissen wurden separate zweifaktorielle Varianzanalysen gruppiert nach den Datentypen vorgenommen. Aus der Analyse ergibt sich, dass insgesamt vier Faktoren einen signifikanten Einfluss auf das Ergebnis haben: der Algorithmus ($P = 8.57 \cdot 10^{-6}$), die Bewegungsgeschwindigkeit ($P = 0$), das Datenformat ($P = 2.30 \cdot 10^{-2}$) und die Interaktion aus Datenformat und Algorithmus ($P = 6.20 \cdot 10^{-8}$). Die anschließende Varianzkomponentenschätzung zeigt, dass

der Einfluss von Algorithmus und Datenformat fast ausschließlich auf deren Interaktion zurückzuführen ist. Lediglich der Algorithmus hat noch einen geschätzten Einfluss von 0,1 % auf die Ergebnisse. Die geschätzten Einflüsse der Faktoren sind in Abbildung 5.13 in Prozent dargestellt. Faktoren, die keinen Einfluss haben, sind nicht aufgeführt.

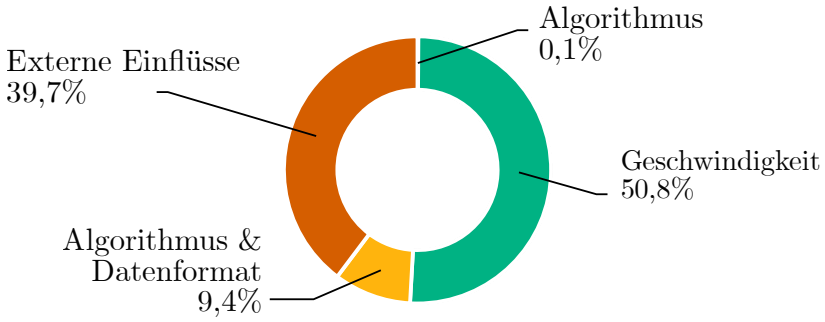
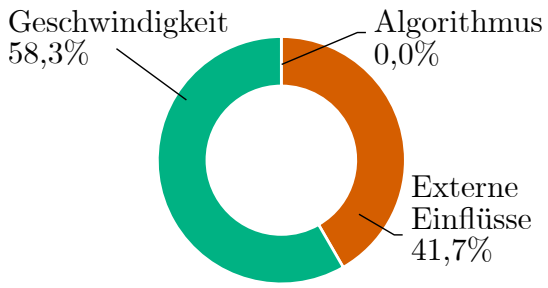


Abbildung 5.13: Einfluss der Messfaktoren auf das Ergebnis der Sensordatenfusion ohne Magnetometer

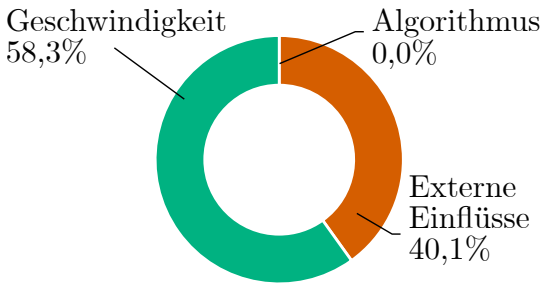
Es ist deutlich zu erkennen, dass zwei Hauptfaktoren die Unterschiede in den ermittelten Fehlern verursachen; zum einen die Geschwindigkeit, mit der das Gerät bewegt wurde, zum anderen externe Einflüsse. Externe Einflüsse sind hierbei alle Faktoren, die nicht in der Analyse untersucht wurden, also Messunterschiede, die nicht von Algorithmus, Datenformat, Bewegungsgeschwindigkeit oder einer Kombination aus diesen verursacht werden. Mögliche Ursachen für diese Unterschiede werden detailliert in Unterunterabschnitt 5.2.3.3 betrachtet. Ein weiterer Einflussfaktor ist die Interaktion zwischen dem gewählten Algorithmus und dem verwendeten Datenformat, welcher für 9,4 % der Messunterschiede verantwortlich ist. Dieses Merkmal ist bereits bei den vorläufigen Untersuchungen in Abbildung 5.12 erkennbar. Die Kombination aus Kalman-Filter und 16 Bit Datenformat verursacht einen deutlich höheren mittleren Fehler.

Für eine genauere Untersuchung dieses Effektes wurden daher zweifaktorielle Varianzanalysen getrennt für jedes Datenformat durchgeführt. Die Ergebnisse sind in den Abbildungen 5.14(a), 5.14(b) und 5.14(c) zu sehen.

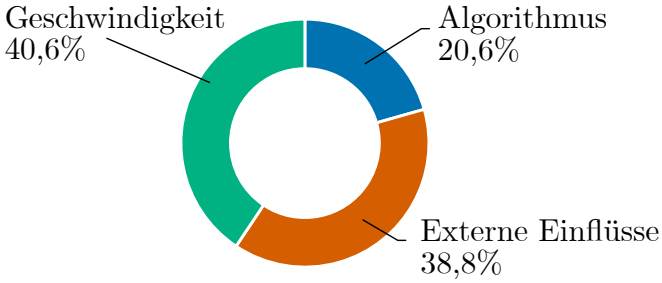
Es ist erkennbar, dass sich die Einflüsse der Faktoren zwischen den 32 Bit-Fließkommazahlen und den 32 Bit-Festkommazahlen kaum unterscheiden. Bei den 16 Bit-Festkommazahlen ist jedoch ein Einfluss des Algorithmus von 20,6 % erkennbar. Dieser wird durch den deutlich höheren Fehler beim Kalman-Filter verursacht. Aus den geringen Unterschieden zwischen den Varianten mit den 32 Bit-Fließkommazahlen und den 32 Bit-Festkommazahlen lässt sich schließen, dass es keine signifikanten Unterschiede in dem Verhalten und der Genauigkeit beider Varianten gibt. Die Ergebnisse können als quasi gleichwertig angesehen werden.



(a) Verteilung der Auswirkungen der Faktoren bei 32 Bit-Fließkommazahlen



(b) Verteilung der Auswirkungen der Faktoren bei 32 Bit-Festkommazahlen



(c) Verteilung der Auswirkungen der Faktoren bei 16 Bit-Festkommazahlen

Abbildung 5.14: Einfluss der Faktoren gruppiert nach Datenformat bei Sensorfusionsalgorithmen ohne Magnetometer

5.2.3.2 Ergebnisse mit Verwendung des Magnetometers

In einer weiterführenden Untersuchung wurden die Ergebnisse der Sensordatenfusion mit Verwendung des Magnetometers betrachtet. Wie in Abschnitt 3.6 beschrieben, stellt hier die initiale Kalibrierung des Magnetometer eine Herausforderung dar, da sich die elektromagnetischen Störungen kontinuierlich verändern und ortsabhängig sind. Für die Durchführung der Messungen wurden diese in einer durchgehenden Messreihe durchgeführt. Die Messungen wurden in drei Gruppen nach ihrer Durchführungsgeschwindigkeit aufgeteilt. Zwischen jeder Gruppe sowie am Anfang und Ende der Messungen wurden Kalibrierungsmessungen durchgeführt. Anhand der Messungen wurden die Daten des Magnetometers für den Harteiseneffekt und den Weicheiseneffekt kalibriert.

Die Resultate einer vorläufigen Messung zum direkten Vergleich der Ergebnisse der Sensordatenfusionen sind in Abbildung 5.15 sehen. Die Ergebnisse zeigen ein sehr ähnliches Verhalten wie die Messungen ohne Verwendung des Magnetometers. Jedoch sind die Fehler geringfügig, aber statistisch signifikant größer. Dies liegt vor allem daran, dass die z-Achse in der vorherigen Bewertung außer Acht gelassen wurde.

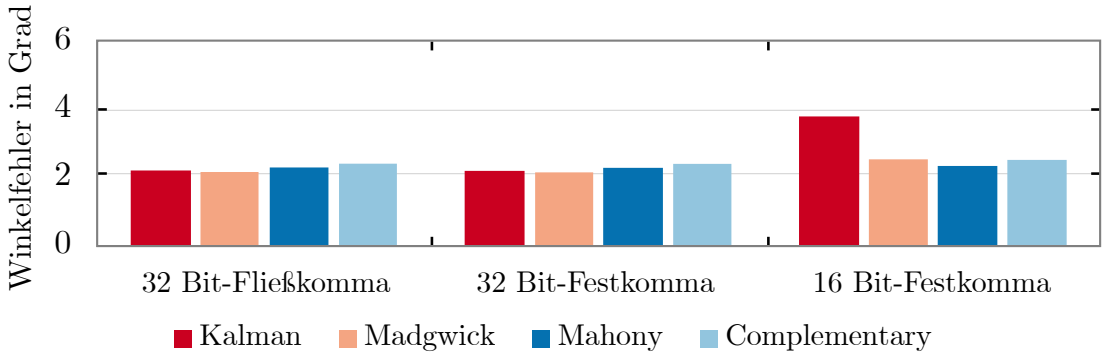


Abbildung 5.15: Mittlerer Winkelfehler mit Verwendung des Magnetometers

Es zeigt sich ebenfalls, dass das Kalman-Filter in der 16 Bit-Festkomma Variante deutlich höhere Fehler aufweist. Zudem sind die Fehler bei den 16 Bit Varianten generell etwas größer als bei anderen Datenformaten. In Tabelle 5.8 sind die genauen Mittelwerte der jeweils 15 Messungen dargestellt. Äquivalent zu den Messungen ohne Magnetometer lässt sich erkennen, dass die Ergebnisse der Variante mit 32 Bit-Festkommazahlen minimal besser sind als bei der Variante mit 32 Bit-Fließkommazahlen.

Tabelle 5.8: Mittlerer Winkelfehler bei 15 Messungen mit Magnetometer

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	2,1781 °	2,1694 °	3,7550 °
Madgwick-Filter	2,1343 °	2,1256 °	2,5054 °
Mahony-Filter	2,2664 °	2,2577 °	2,3134 °
Komplementärfilter	2,3793 °	2,3706 °	2,4868 °

Für die Analyse der Sensorfusionsalgorithmen mit Verwendung des Magnetometers wurden ebenfalls 360 Messungen durchgeführt und eine dreifaktorielle Varianzanalyse auf den Daten angewandt. Die Ergebnisse der Messungen sind im Anhang in Tabelle D.20 zu finden. Anschließend wurden ebenfalls separate Analysen gruppiert nach Datenformat durchgeführt. Aus der Analyse ergibt sich, dass insgesamt fünf Faktoren einen signifikanten Einfluss auf das Messergebnis haben: der Algorithmus ($P = 1.82 \cdot 10^{-8}$), die Bewegungsgeschwindigkeit ($P = 0$), das Datenformat ($P = 0$) und die Interaktion aus Datenformat und Algorithmus ($P = 0$) und die Interaktion aus Algorithmus und Bewegungsgeschwindigkeit ($P = 2.56 \cdot 10^{-9}$). In der durchgeführten Varianzkomponentenschätzung zeigt sich jedoch, dass weder der Algorithmus allein noch die Interaktion aus Algorithmus und Bewegungsgeschwindigkeit einen messbaren Einfluss haben. In beiden Fällen ergibt sich ein Einfluss von unter 0,1 %.

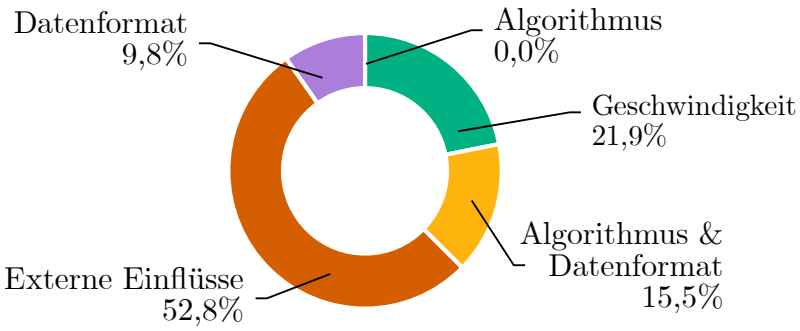
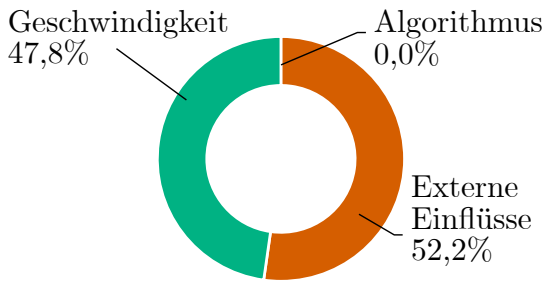


Abbildung 5.16: Prozentualer Einfluss der Messfaktoren auf das Ergebnis der Sensordatenfusion mit Magnetometer

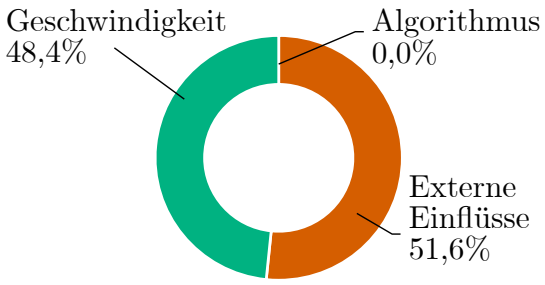
In den Ergebnissen lassen sich deutlichere Unterschiede zu der vorherigen Versuchsreihe erkennen. Auffällig ist hier, dass der Gesamteinfluss der Durchführungsgeschwindigkeit deutlich geringer ist. Die Einflüsse durch die Interaktion aus Algorithmus und Datenformat sowie der Einfluss des Datenformats an sich haben zugenommen. Anhand der vorläufigen Messungen, deren Ergebnis in Abbildung 5.15 gezeigt ist, war dieses Ergebnis zu erwarten. Zum einen sind alle gemessenen Fehler der Algorithmen mit 16 Bit-Festkomma-Datenformat etwas größer als die Äquivalente in den anderen Datenformaten. Zum anderen fallen die Unterschiede, gerade in Bezug auf das Kalman-Filter deutlicher aus. Weiterhin ist auch zu erkennen, dass ein größerer Anteil der Messunterschiede keinem der untersuchten Faktoren zugeordnet werden kann. Die Unterschiede können hier mehrere Ursachen haben. Der größere Einfluss der externen Faktoren lässt sich darauf zurückführen, dass eine perfekte Kalibrierung des Magnetometers sehr schwierig ist. Je nach Ausrichtung des Sensors im Magnetfeld können so von der Messung abhängige Fehler entstehen. Die größeren Unterschiede bei dem 16 Bit-Datenformat ergeben sich aus einer Kombination der geringeren Genauigkeit des Datenformats und der Tatsache, dass die Daten aus dem Magnetometer eine deutlich geringere Abtastrate von 20 Hz haben, vergleichen mit den 200 Hz der Daten von Beschleunigungssensor und Gyroskop.

Für eine genauere Untersuchung der Einflüsse innerhalb eines Datenformats wurden ebenfalls separate zweifaktorielle Varianzanalysen durchgeführt. Die Ergebnisse sind in den Abbildungen 5.17(a), 5.17(b) und 5.17(c) zu sehen.

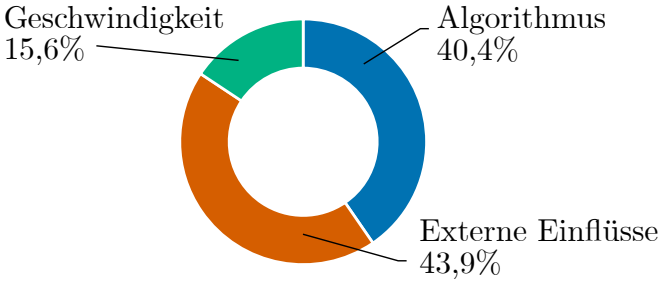
Es ist erkennbar, dass sich die Einflüsse der Faktoren zwischen den 32 Bit-Fließkommazahlen und den 32 Bit-Festkommazahlen kaum unterscheiden. Im Vergleich zu der Untersuchung ohne Magnetometer fällt der Einfluss der Bewegungsgeschwindigkeit geringer aus, was auf die Kalibrierung des Magnetometers und die Ausrichtung des Sensors im Magnetfeld zurückzuführen ist. Die größten Unterschiede zeigen sich abermals bei den 16 Bit-Festkommazahlen. Die Unterschiede durch externe Einflüsse sind mit 42,9 % etwas höher. Der verwendete Algorithmus verursacht 40,4 % der Messunterschiede. Dieses Ergebnis war ebenfalls zu erwarten, da die in Abbildung 5.15 gezeigten Messergebnisse einen deutlicheren Unterschied zwischen den Algorithmen zeigen.



(a) Verteilung der Auswirkungen der Faktoren bei 32 Bit-Fließkommazahlen



(b) Verteilung der Auswirkungen der Faktoren bei 32 Bit-Festkommazahlen



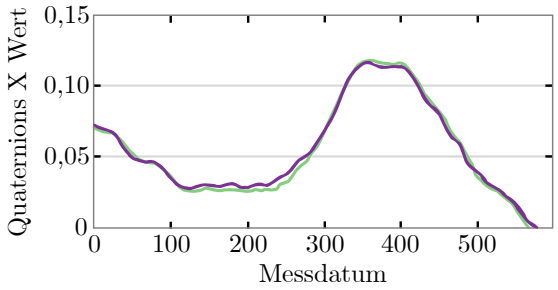
(c) Verteilung der Auswirkungen der Faktoren bei 16 Bit-Festkommazahlen

Abbildung 5.17: Einfluss der Faktoren gruppiert nach Datenformat bei Sensorfusionsalgorithmen mit Magnetometer

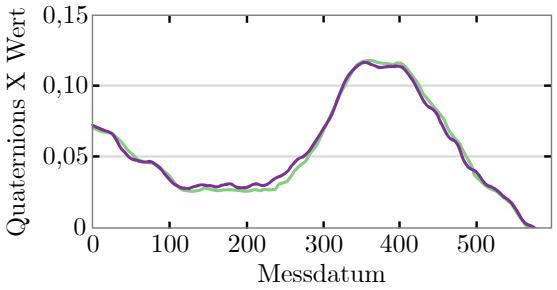
5.2.3.3 Analyse der Filterergebnisse

Eines der auffälligsten Merkmale, das bei Untersuchung der Sensorfusionsalgorithmen mit unterschiedlichen Datenformaten beobachtet wurde, ist der signifikant erhöhte Fehler des Kalman-Filters bei den 16 Bit-Festkommazahlen. Dieses Phänomen wird dadurch verursacht, dass der mögliche Wertebereich beim Kalman-Filter deutlich größer ist als bei den anderen Sensorfusionsalgorithmen. Gerade bei dem 16 Bit Datenformat resultiert dies in einem stark erhöhten Quantisierungsfehler. Beim Kalman-Filter musste aufgrund des großen Wertebereichs die Bitbreite des Nachkommaanteils auf 8 Bit-festgelegt werden. Hierdurch entstehen klar sichtbare Quantisierungsfehler, welche auf Abbildung 5.18 gut zu erkennen sind.

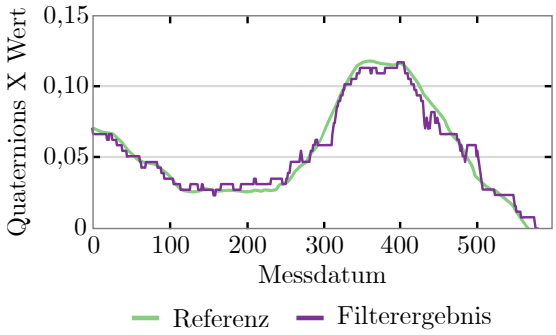
Für die 32 Bit-Fließkommazahlen und die 32 Bit-Festkommazahlen sind keine Quantisierungsfehler zu erkennen. Für die anderen Filter konnte aufgrund des deutlich kleineren Wertebereichs ein Nachkommaanteil von 12 Bit gewählt werden, was die Quantisierungsfehler deutlich verringert. Die Genauigkeit des Kalman-Filters ist in der 16 Bit Variante so gering, dass Winkelgeschwindigkeiten des Gyroskops unter 90°/s auf null gerundet werden. Hierdurch entstehen deutlich höhere Fehler und die Lage wird fast ausschließlich von den Daten des Beschleunigungssensors und des Magnetometers bestimmt.



(a) Vergleich von Referenz und Ergebnis des Kalman-Filters mit 32 Bit-Fließkommazahlen



(b) Vergleich von Referenz und Ergebnis des Kalman-Filters mit 32 Bit-Festkommazahlen



(c) Verteilung der Auswirkungen der Faktoren bei 16 Bit-Festkommazahlen

Abbildung 5.18: Auswirkung des Datenformats auf den Quantisierungsfehler

Eine weitere Beobachtung, die sich durch Quantisierungsfehler erklären lässt, ist die etwas höhere Genauigkeit der Filter in der Variante mit 32 Bit-Festkommazahlen. Die 32 Bit-Fließkommazahlen haben eine Mantisse von 23 Bit, was aufgrund der Normalisierung der Zahlen die Genauigkeit auf 23 Bit begrenzt. Die 32 Bit-Festkommazahlen haben einen Nachkommaanteil von 24 Bit. Hierdurch sind sie geringfügig genauer und dadurch besser für das untersuchte Szenario geeignet als 32 Bit-Fließkommazahlen.

Weiterhin kann an den Ergebnissen gesehen werden, dass mehr als ein Drittel der Messunterschiede von externen Einflüssen verursacht werden. Diese Messunterschiede lassen sich also nicht auf einen der untersuchten Faktoren zurückführen. Für die Untersuchung wurde die Bewegungsgeschwindigkeit in drei Gruppen aufgeteilt. Langsame Bewegungen mit einer Winkelgeschwindigkeit von etwa 7,5 °/s, mittelschnelle Bewegungen mit 15 °/s und schnelle Bewegungen mit 30 °/s. Da die Testbewegungen alle manuell durchgeführt wurden, besteht die Möglichkeit, dass es auch innerhalb der Gruppen zu Geschwindigkeitsunterschieden kommt, welche dann indirekt zu den externen Einflüssen beitragen. Um dies auszuschließen, wurden die mittleren Fehler sowie die mittleren Winkelgeschwindigkeiten in einem Graphen aufgetragen, der in Abbildung 5.19 zu sehen ist.

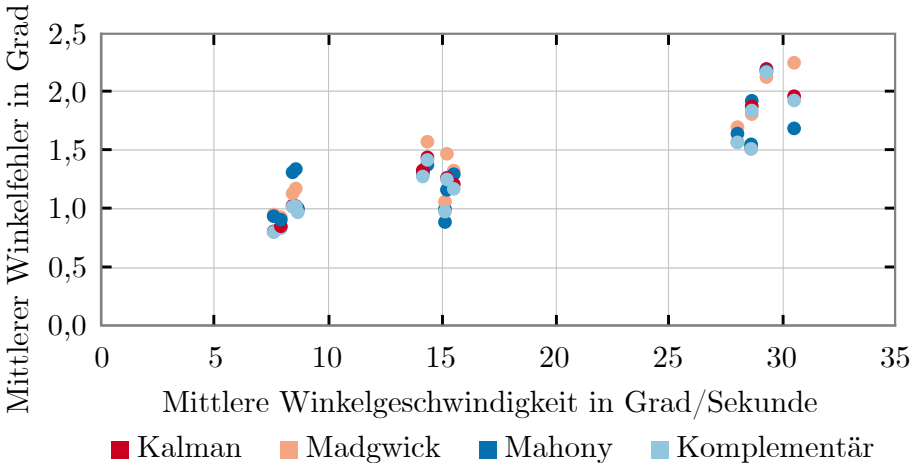


Abbildung 5.19: Vergleich von mittlerer Winkelgeschwindigkeit zu mittlerem Fehler

Ersichtlich ist eine deutliche Korrelation zwischen der mittleren Winkelgeschwindigkeit und dem gemessenen Fehler. Weiterhin ist eine klare Unterscheidung der drei Geschwindigkeitsgruppen möglich. Die Winkelgeschwindigkeiten innerhalb der Gruppen unterscheiden sich nicht stark. Innerhalb der Gruppe mit 30 °/s ist eine schwache Korrelation zwischen Winkelfehler und Winkelgeschwindigkeiten zu erkennen. Diese kann aber nur zu einem geringen Teil die externen Fehler erklären. In den anderen beiden Gruppen ist eine Korrelation nicht erkennbar, welche die Streuung der Messfehler innerhalb der Gruppen erklärt. Die Messunterschiede können folglich nicht mit den verbleibenden Geschwindigkeitsunterschieden erklärt werden.

Ein weiterer Faktor, der zu den externen Fehler beitragen könnte, ist die Genauigkeit, mit der die vorgegebenen Bewegungen durchgeführt worden sind. Da die Messungen von einem Menschen durchgeführt wurden, ist es verständlicherweise nicht möglich, alle Messungen mit exakt derselben Bewegung durchzuführen. Um diesen Fehler zu analysieren, wurde untersucht, ob die Genauigkeit der Bewegungsdurchführung einen Einfluss auf den gemessenen mittleren Winkelfehler hat. Hierfür wurden der gemessene Winkelfehler und die Stärke der Abweichung von der vorgegebenen Bewegung miteinander in Beziehung gesetzt. Das Ergebnis ist in Abbildung 5.19 zu sehen.

Bei diesem Vergleich ist keine Korrelation zwischen der Abweichung von der vorgegebenen Bewegung, verglichen mit dem mittleren Winkelfehler, zu erkennen. Die Durchführungsgenauigkeit der vorgegebenen Bewegung kann folglich als Ursache für die Messunterschiede ausgeschlossen werden.

Wie in Abbildung 5.19 zu erkennen ist, kann der Geschwindigkeitsunterschiede innerhalb der Geschwindigkeitsgruppen nur einen kleinen Teil der externen Einflüsse erklären. Auch der menschliche Einfluss auf die Messdurchführung konnte ausgeschlossen werden. Die externen Einflüsse müssen folglich aus anderen Quellen stammen. Der Messvorgang

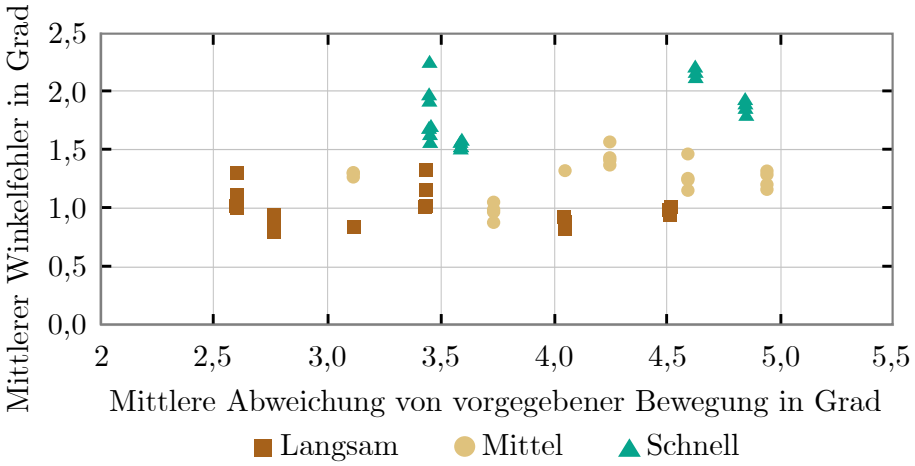


Abbildung 5.20: Vergleich von Durchführungsgenauigkeit zu mittlerem Fehler

an sich kann einen Einfluss auf das Ergebnis haben. Zum einen hat die Genauigkeit der genutzten Bildanalyse einen Einfluss. Diese hängt von der Auflösung des Kamerabildes und bei Bewegungen auch besonders von der Bildrate ab. Zum anderen können Messabweichungen dadurch entstehen, dass die Messungen mit zwei unabhängigen Systemen durchgeführt wurden, welche eine eigene Taktfrequenz und Referenzzeit benutzten. Für einen Vergleich der Daten muss eine Kreuzkorrelation durchgeführt werden, um Taktfrequenz und Referenzzeit aneinander anzupassen. Hier können auch Fehler entstehen, wenn die Daten nicht perfekt übereinandergelegt werden können. Am plausibelsten ist, dass die genutzten Inertialsensoren selbst einen Teil der externen Fehler ausmachen. Hierbei sind besonders Fehler wie Bias, Skalierungsfehler, Temperaturdrift und Rauschen relevant. Diese können sich auch geringfügig zwischen den Messungen verändern.

5.2.4 Zusammenfassung

Die Untersuchung der unterschiedlichen Sensorfusionsalgorithmen in Kombination mit unterschiedlichen Datenformaten hat viele hilfreiche Erkenntnisse geliefert, die dabei helfen können, das richtige Filter für einen Einsatzzweck zu wählen. Es konnte gezeigt werden, dass die Verwendung von 32 Bit-Festkommazahlen den Rechenaufwand um etwa 50 % reduziert, ohne dabei Einbußen bei der Genauigkeit der Sensorfusionsalgorithmen zu verursachen. Weiterhin sind der Speicherbedarf und die Codegröße durch die Verwendung von 32 Bit-Festkommazahlen nur geringfügig höher. Der Rechenaufwand bezieht sich hier auf ein System, welches keine FPU besitzt. Bei einem System mit FPU werden die Unterschiede zwischen Fließkommazahlen und Festkommazahlen deutlich geringer ausfallen. Die für die komplexe Berechnung notwendige Hardware wird jedoch trotzdem einen höheren Energieverbrauch verursachen.

In den Untersuchungen konnte gezeigt werden, dass es bei der Messung von menschlichen Bewegungen nur geringen Unterschiede bei den untersuchten Filtern gibt. Die einfacheren Filter wie Mahony-Filter, Madgwick-Filter oder Komplementärfilter benötigen jedoch nur etwa 30 % der Ausführungszeit des Kalman-Filters. Dies führt zu dem Schluss, dass diese Filter in einem ressourcenbeschränkten eingebetteten System eher ausgewählt werden sollten. Dies gilt vor allem, wenn nur 16 Bit Datenformate verwendet werden können, da das Kalman-Filter aufgrund des benötigten größeren Wertebereiches in diesem Fall nicht nutzbar ist.

Wichtig bei den Ergebnissen dieser Untersuchungen ist, dass diese sich auf die Bestimmung der Orientierung bei einer menschlichen Interaktion beziehen. Wie in Abschnitt 4.3 gezeigt wurde, können sich die Ergebnisse bei der Interaktion mit Robotern oder im industriellen Umfeld deutlich unterscheiden. Es ist zu erwarten, dass hier das deutlich aufwendigere Kalman-Filter bessere Ergebnisse liefert.

5.2.5 Weiterführende Untersuchungen

Bei den Untersuchungen ist deutlich geworden, dass für die Bestimmung der Orientierung mit Verwendung des Magnetometers eine genaue Kalibrierung der Daten des Magnetometers benötigt wird. Die Fehler, die durch eine ungenaue Kalibrierung entstehen, können deutlich die der Fusionsalgorithmen und Datenformate übersteigen. Hier wäre eine weiterführende Untersuchung interessant, die ermitteln soll wie sensibel die untersuchten Sensorfusionsalgorithmen auf die Kalibrierung des Magnetometers reagieren. In einem ressourcenbeschränkten eingebetteten System ist es wahrscheinlich, dass die Kalibrierung des Magnetometer nicht zu aufwendig ausfällt. Weiterhin kann auch die Verwendung anderer Varianten des Kalman-Filters sowohl bessere Ergebnisse liefern als auch den Rechenaufwand verringern [A122],[A123]. In den durchgeführten Untersuchungen wurde ein erweitertes Kalman-Filter verwendet.

5.3 Augmentation von Inertialsensor-Daten

Wie in den bisherigen Arbeiten bereits erkenntlich wurde, können viele der Arbeitsschritte von der Vorverarbeitung der Messdaten über Sensordatenfusion bis hin zur Gestenerkennung direkt auf der Hardware des intelligenten Sensors ausgeführt werden. Auch auf modernen Smartphones kann ein großer Teil der Sensordatenfusion direkt auf dem Inertialsensor durchgeführt werden. Dies hat den Vorteil, dass Energie eingespart werden kann, wenn diese Verarbeitungsschritte nicht auf dem deutlich weniger energieeffizienten Hauptprozessor durchgeführt werden müssen. Für bestimme Verarbeitungsschritte können zudem bereitgestellte Programmbibliotheken verwendet werden, was den Entwicklungsaufwand reduzieren kann.

Es hat sich jedoch auch gezeigt, dass deutlich mehr Aufwand bei der Entwicklung und insbesondere der Verifikation der Firmware für diese intelligenten Sensoren notwendig ist. Besonders wichtig ist hier die Verfügbarkeit von Messdaten während der Entwicklung. Beispielsweise werden im Bereich der Gestenerkennung viele Daten benötigt, um die Algorithmen für die Gestenerkennung zu trainieren und zu verifizieren [A124]. Weiterhin ist es gerade bei sicherheitskritischen Anwendungen, etwa in Industrieanlagen oder im Automobilsektor wichtig, dass das System zuverlässig und vor allem sicher ist. Zu diesem Zweck existieren Normen und Vorschriften für die Verifikation und Validierung [A125], [A126]. Insbesondere, wenn mit Daten aus Inertialsensoren gearbeitet wird, ist es jedoch schwer oder unmöglich, Daten für alle vorgegebenen oder denkbaren Szenarien mit einem Sensor aufzuzeichnen.

Für diesen Zweck wurde ein Framework entwickelt, welches es erlaubt, auf der Basis von bereits aufgezeichneten Sensordaten neue Testfälle und Sensordaten zu generieren. Dieses Framework ermöglicht es, die aufgezeichneten Sensordaten interaktiv mittels vordefinierter Manipulationsschritte zu verändern oder neue Daten zu erzeugen. Ähnliche Verfahren werden bereits im Bereich des maschinellen Lernens und insbesondere bei der Bildverarbeitung verwendet [A41], [A43]. Neben der Verwendung von vorher aufgezeichnete Sensordaten können aber auch künstlich generierte Trajektorien verwendet werden, etwa aus der *Sensor Fusion and Tracking Toolbox* von MathWorks Inc. [A127] oder des *GNSS + inertial navigation, sensor fusion simulator* von Aceinna Inc.[A38].

Die Besonderheit des in diesem Kapitel vorgestellten Frameworks für die Manipulation und Augmentation von Sensordaten ist, dass die Auswirkung der veränderten Sensordaten direkt zur Laufzeit auf dem intelligenten Sensor betrachtet werden können. Dies ermöglicht eine deutlich direktere und auch gezieltere Entwicklung und Verifikation des Sensorsubsystems. Neue Testfälle oder Sensordaten können binnen weniger Minuten generiert und direkt auf dem Sensor untersucht werden. Das Framework wurde 2021 auf dem *Rapid System Prototyping (RSP)*-Workshop auf der Embedded Systems Week (ESWEEK)-Konferenz vorgestellt [B9].

5.3.1 Konzept

Das Ziel des Frameworks für die Datenaugmentation ist es, Testdaten für eine große Bandbreite an Anwendungsfällen und Versuchsaufbauten zu generieren. Die Daten sollen sowohl aus bereits aufgenommenen als auch aus künstlichen Daten generiert werden können. Die Manipulation der Sensordaten soll hierbei auf dem Host-Computer durchgeführt werden, der mit dem zu untersuchenden System verbunden ist. Eine Manipulation der Sensordaten auf dem Sensorsubsystem selbst würde das Laufzeitverhalten des untersuchten Sensors zu stark beeinflussen. Insbesondere Energieverbrauch, aber auch Ausführungszeit der untersuchten Algorithmen ließen sich so nur schwer untersuchen. Das Framework soll drei grundlegende Bedingungen erfüllen:

1. Testdaten sollen sowohl aus bereits existierenden, aufgenommen oder künstlichen Daten erzeugt werden könnten. Auch die Generierung von komplett künstlichen Testdaten soll möglich sein.
2. Die Erzeugung und Veränderung der Testdaten soll auch „live“ mit gestreamten Daten möglich sein. Das Framework muss also in der Lage sein, für jedes neue Sensordatum direkt die veränderten Daten zu erzeugen, ohne Zugriff auf die gesamte Messung zu benötigen. Zudem muss die Generierung möglichst ohne große Verzögerung stattfinden.
3. Es soll auf schnelle und einfache Weise möglich sein, neue Testdaten zu generieren; möglichst ohne Code kompilieren zu müssen.

Daher wurde die Verarbeitung des Frameworks nach dem „Pipes und Filter“-Architekturmuster (auch Datenfluss-System) entworfen. Mit diesem ist es möglich, alle der oben genannten Kriterien zu erfüllen. Das Framework ist so entworfen, dass es aus einer Vielzahl einzelner, *Funktionsblöcke* genannten Komponenten besteht. Jeder Funktionsblock (FB) erfüllt dabei eine spezifische Aufgabe bei der Manipulation der Sensordaten und entspricht einem Filter in dem „Pipes und Filter“-Architekturmuster. Die Funktionsblöcke können mittels der „Pipes“ miteinander verbunden werden, um eine komplexe Verarbeitungskette zu erzeugen. Um unnötige Berechnungsschritte zu vermeiden, sollen die Ergebnisse eines FBs von mehreren nachfolgenden FBs verwendet werden können. Die Übertragung der Daten geschieht pull-basiert, die veränderten Daten werden also am Ausgang des Frameworks abgefragt und erst auf diese Anfrage hin berechnet. Dies soll unnötige Berechnungen vermeiden und erleichtert es, das Framework in andere Systeme zu integrieren. Der Datenfluss innerhalb des Frameworks wird beispielhaft in Unterabschnitt 5.3.2 beschrieben.

5.3.1.1 Mögliche Versuchsaufbauten

Das Framework für die Manipulation und Augmentation von Sensordaten wurde so entworfen, dass es ohne viel Aufwand in einen Versuchsaufbau integriert werden kann, sei es offline, um die Daten vor der Nutzung zu generieren, oder live mit einem angeschlossenen Sensor. Insbesondere in Kombination mit der SiL-Plattform [B5] können unterschiedliche Versuchsaufbauten realisiert werden. Drei der gängigsten Versuchsaufbauten sowie die Aufnahme der Sensordaten sind in Abbildung 5.21 aufgezeigt.

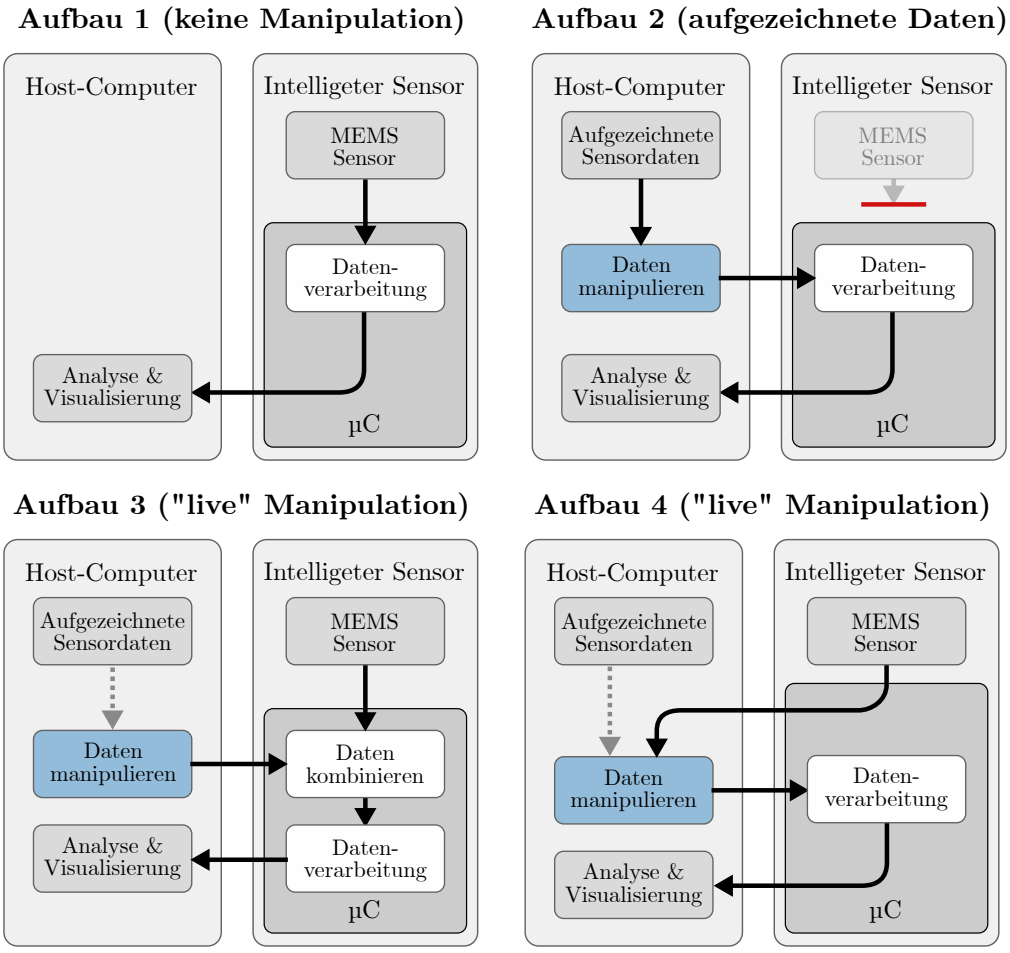


Abbildung 5.21: Mögliche Versuchsaufbauten für die Anwendung des Frameworks für Datenaugmentation in Verbindung mit der SiL-Plattform

In Aufbau 1 werden Messungen mit dem intelligenten Sensor durchgeführt. Diese Messungen werden für die spätere Verwendung gespeichert oder auf dem Host ohne vorherige Veränderung der Daten analysiert. Dieser Vorgang entspricht der „Record“-Phase der SiL-Plattform und bezieht das Framework nicht ein, wird aber benötigt, um die Daten für eine spätere Generierung der Testdaten aufzunehmen.

Aufbau 2 ist der typische Messaufbau, für den das Framework entworfen wurde. Er entspricht der „Inject“ Phase der SiL-Plattform. Bereits aufgezeichnete Daten werden auf dem Host-Computer zur Laufzeit mithilfe des Frameworks verändert und über die SiL-Plattform in den Sensor gespeist. Der intelligente Sensor behandelt die eingespeisten Daten so, als würden sie direkt von den Sensoren kommen und führt die Verarbeitung der Daten durch. Die Ergebnisse der Datenverarbeitung werden zurück an den Host-Computer gesendet. Mit diesem Aufbau ist es möglich, das Verhalten des Sensors sehr gezielt zu untersuchen. Beispielsweise lässt sich ermitteln, unter welchen Bedingungen eine zuverlässige Sensordatenfusion nicht mehr möglich ist. Durch die Verwendung von aufgezeichneten Daten ist eine Reproduzierbarkeit der Ergebnisse sichergestellt.

Im Gegensatz zu Aufbau 2 werden in Aufbau 3 Live-Daten der Inertialsensoren des intelligenten Sensors verwendet. Das Framework wird genutzt, um Daten zu generieren, die auf dem Sensor mit den eigentlichen Messdaten kombiniert und dann verarbeitet werden. Mit diesem Aufbau ist es möglich im getesteten Sensor gezielt Fehler einzuspeisen oder die Eigenschaften der Sensordaten zu verändern, um die Robustheit zu untersuchen. Die Ergebnisse der Verarbeitung des Sensors werden zurück an den Host-Computer gesendet, damit sie dort visualisiert und analysiert werden können.

Aufbau 4 ist der zweite mögliche Aufbau, bei dem Sensordaten live verwendet werden. Der intelligente Sensor sendet die Daten, die er von den Inertialsensoren erhält, an den Host-Computer, auf welchem die Daten mithilfe des Frameworks verändert werden. Die veränderten Daten werden für die Verarbeitung zurück an den intelligenten Sensor geschickt. Die Ergebnisse der Verarbeitung des Sensors werden wiederum zurück an den Host-Computer gesendet. Gegenüber Aufbau 3 hat diese Variante den Vorteil, dass keine zusätzlichen Operationen auf dem Sensor stattfinden müssen, was die Ergebnisse verfälschen kann. Es wird jedoch durch die Kommunikation und Verarbeitung eine Verzögerung verursacht, die bei der Bewertung der Ergebnisse beachtet werden muss.

Sowohl für Aufbau 3 als auch Aufbau 4 ist es wichtig anzumerken, dass diese eine Verzögerung bei der Verarbeitung der Daten sowie einen Overhead für die Kommunikation im intelligenten Sensor verursachen. Die Manipulation der Daten geschieht dementsprechend nicht in Echtzeit. Bei Untersuchung der SiL-Plattform konnte gezeigt werden, dass es mit dieser möglich ist, gleichzeitig Daten von drei dreiachsigen Sensoren mit jeweils 1,6 kHz in beide Richtungen zu übertragen. In Aufbau 4 müssen potenziell 50 % mehr Daten übertragen werden. Daher ist zu erwarten, dass die maximale Datenrate der Sensoren geringer ausfallen wird.

5.3.1.2 Funktionsblöcke

Die Funktionsblöcke sind die Grundbausteine des Frameworks. Jeder FB erfüllt hierbei eine spezifische Aufgabe bei der Manipulation der Sensordaten, wie beispielsweise eine mathematische oder logische Operation, temporäres Speichern oder die Filterung der Daten. Die meisten Funktionsblöcke sind so konzipiert, dass sie sowohl mit einzelnen Werten als auch mit Vektoren beliebiger Größe arbeiten. Lediglich spezielle Operationen, beispielsweise für Quaternionen oder Vektorrotation setzen eine fixe Größe des Vektors voraus. Folgende Funktionsblöcke wurden entworfen:

- **Dateneingang**: Sensordaten, Vektor, Quaternion oder Matrix, künstliche Signale (Rauschen, Sinusfunktion, Rechtecksignal, Dreiecksignal, Sägezahn)
- **Mathematische Operationen**: Arithmetische Operationen, Integration, Differentiation, Rundung, Sättigung und trigonometrische Funktionen, Maximum, Minimum, Mittelwert
- **Logische Operationen**: Selektion von Signalen, Schwellenwert-Überprüfung und Vergleich von Werten.
- **Manipulation des Datenstroms**: Zusammenführen von Signalen, Aufsplitten von Vektoren, Verzögerung der Daten, temporärer Speicher.
- **Spezielle Operationen**: Operationen für Quaternionen, Vektorrotationen, Nichtlinearität, FIR- und Filter mit unendlicher Impulsantwort (Infinite Impulse Response, IIR)-Filter, Sensordatenfusion, Interpolation, Normalisierung, Vektorlänge
- **Datenausgang**: Ausgabe der veränderten Sensorsignale.

Insgesamt wurden 27 Funktionsblöcke entworfen, welche in die oben genannten Kategorien aufgeteilt wurden. Durch die Kombination der genannten Funktionsblöcke sollte es möglich sein, eine sehr große Bandbreite an Manipulationen und Augmentationen der Sensordaten durchzuführen. Da sowohl Speicher als auch Verzweigungen/Selektionen möglich sind, lassen sich auch komplexe Abläufe modellieren.

Die meisten der FBs führen allgemein bekannte mathematische oder logische Operationen durch. Wie beispielsweise Vektorarithmetik, Schwellenwerterkennung oder die Integration von Messwerten. In komplexeren FBs werden mehrere Operationen vereint. So kann die Rotation von Vektoren etwa anhand von Euler-Winkeln (ZXY), Rotationsmatrizen oder Quaternionen erfolgen. Einige der FBs speichern zudem ihren Zustand, wenn dies erforderlich ist. Dies ist beispielsweise bei den FIR- und IIR-Filtern der Fall.

5.3.2 Implementierung

Das Framework für die Manipulation und Augmentation von Sensordaten wurde in Java als Add-on für die Eclipse IDE implementiert. Dies erlaubt eine reibungslose Interaktion mit der Sensor-in-the-Loop-Plattform, welche ebenfalls als Eclipse-Add-on entwickelt wurde. Hierbei besteht die Implementierung aus zwei Einzelkomponenten. In der ersten Komponente ist das Framework an sich implementiert, dies umfasst die Funktionalität der Funktionsblöcke sowie einen Node-Editor, welcher es erlaubt, festzulegen, wie die Daten manipuliert werden sollen. Die zweite Komponente ist ein Graph, der den Verlauf der ursprünglichen und der veränderten Sensordaten anzeigt. Dieser Graph ist eine Erweiterung des Graphen, der auch von der Sensor-in-the-Loop-Plattform genutzt wird, um Sensordaten anzuzeigen, zu speichern oder wieder in den Sensor zu speisen. Aus dem Node-Editor heraus lassen sich die Manipulationsschritte direkt auf die Daten anwenden, die gerade im Graphen geladen sind.

5.3.2.1 Framework und Datenfluss

Wie bereits in Unterabschnitt 5.3.1 ausgeführt wurde, wurde das Framework nach dem „Pipes und Filter“-Architekturmuster entworfen. Dieses datenflussbasierte Modell erlaubt es, die Verarbeitung der Sensordaten schrittweise als Konkatenation von Verarbeitungsschritten durchzuführen. Die Propagierung der Daten soll dabei pull-basiert ablaufen. Für jeden Zeitpunkt, an dem neue Daten benötigt werden, werden diese an den Ausgängen des Frameworks angefragt und darauf hin berechnet.

Das Framework wurde so entworfen, dass es ohne viel Aufwand in andere Anwendungen integriert werden kann, um die Manipulation der Sensordaten vorzunehmen. Propagierung und Manipulation der Daten werden intern durchgeführt. Um das Framework zu nutzen, müssen lediglich folgende Schritte durchgeführt werden:

1. Die Daten müssen entweder aus einer Datei ausgelesen oder von den Sensoren empfangen werden.
2. Die Messdaten der Sensoren müssen an die Sensor-Eingänge übergeben werden.
3. Die veränderten Messdaten müssen an den Sensor-Ausgängen abgefragt werden.
4. Wenn der genutzte Versuchsaufbau die Sensordaten verändert, müssen diese verwendet werden, ansonsten die originalen Daten aus Schritt 1.

Bei den genannten Schritten ist zu beachten, dass sowohl bei der Übergabe der Sensordaten als auch bei der Abfrage der Sensordaten ein Zeitstempel angegeben werden muss. Dieser sorgt im Framework dafür, dass Berechnungen nicht mehrfach durchgeführt werden. Das Programm, welches das Framework verwendet, muss sicherstellen, dass diese Zeitstempel valide und monoton steigend sind.

Um den Datenfluss innerhalb des Frameworks zu veranschaulichen, ist in Abbildung 5.22 ein einfaches Beispiel gegeben. Die Daten des Gyroskops werden erst mit einem Skalar multipliziert, anschließend wird ein weißes Rauschen auf die Daten addiert. Das Resultat wird über den Ausgang des Gyroskops ausgegeben. Der Datenfluss innerhalb des Frameworks für dieses Beispiel ist in Abbildung 5.23 dargestellt.

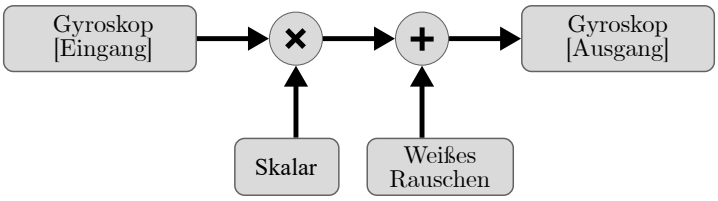


Abbildung 5.22: Beispiel für eine Manipulation der Daten.

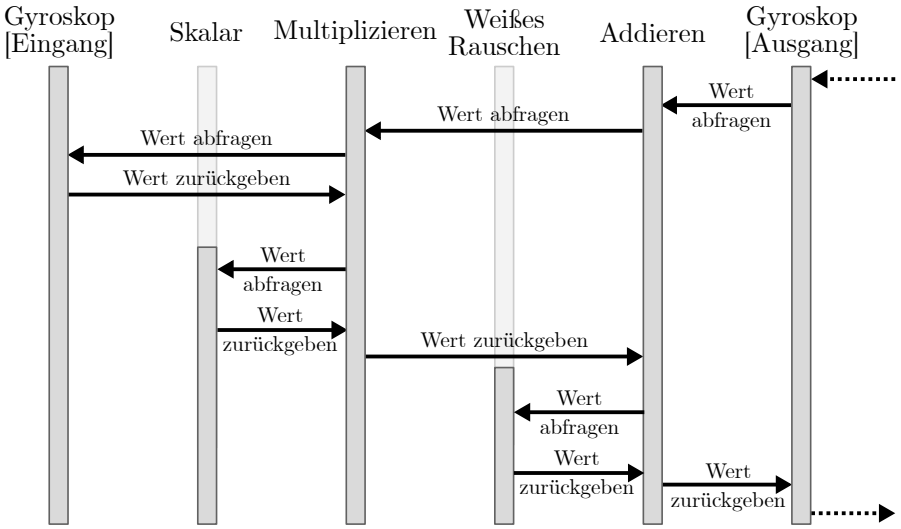


Abbildung 5.23: Datenfluss des Beispiels von Abbildung 5.22

Wenn ein neues Messdatum am *Gyroskop [Ausgang]* angefragt wird, fragt dieser Block die Daten von der Addition als einzigem Dateneingang ab. Für die Addition wiederum werden für die Berechnung die Daten von der Multiplikation und von dem Block, der weißes Rauschen generiert, benötigt und abgefragt. Die Multiplikation benötigt wiederum weitere Daten als Input. Die Abfrage der Daten wird so weit propagiert, bis keiner der Funktionsblöcke mehr Daten von einem anderen Funktionsblock benötigt. Wenn für einen Zeitpunkt mehrere Funktionsblöcke die Daten von einem anderen Funktionsblock abfragen, werden die Daten nach der ersten Berechnung zwischengespeichert. Dies verhindert überflüssige Berechnungen. Zudem wird so verhindert, dass sich digitale Filter unerwartet verhalten, da für denselben Zeitpunkt mehrmals ein Update des Zustands durchgeführt wird.

5.3.2.2 Aufbau eines Funktionsblocks

Die Funktionsblöcke sind so entworfen, dass sie unabhängig voneinander funktionieren und selbstständig neue Ergebnisse berechnen. Sie fragen Daten aus ihren Quellen ab, wenn diese benötigt werden. Der Kontrollfluss bei der Abfrage eines neuen Datums eines FBs ist in Abbildung 5.24 gezeigt.

Wenn ein neues Datum abgefragt wird, wird überprüft, ob das Ergebnis für diesen Zeitpunkt schon berechnet wurde. Ist dies der Fall, wird das bereits berechnete Ergebnis ausgegeben. Wenn das Ergebnis noch nicht vorliegt, werden die Daten für den angegebenen Zeitpunkt von allen Datenquellen angefragt, die von dem Funktionsblock genutzt werden. Sobald alle Daten vorhanden sind, wird das Ergebnis berechnet, abgespeichert und letztendlich ausgegeben.

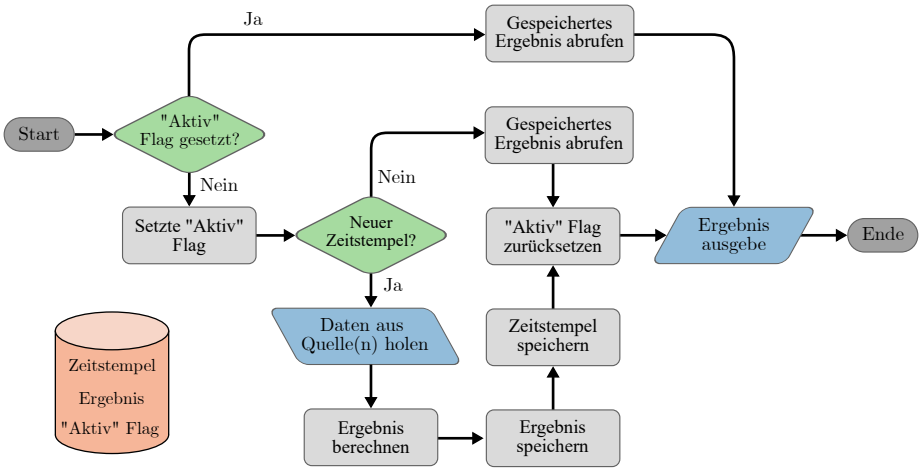


Abbildung 5.24: Kontrollfluss eines Funktionsblocks bei einer Datenabfrage

Wie in Abbildung 5.24 zu erkennen ist, wird zusätzlich in einem Funktionsblock ein „Aktiv“-Flag verwendet. Dieses speichert, ob der Funktionsblock gerade dabei ist, eine Datenanfrage zu verarbeiten. Benötigt wird dieses Flag, wenn in der Verarbeitungskette Zirkelschlüsse vorhanden sind. Das Flag sorgt in diesem Fall dafür, dass keine unendliche Rekursion entsteht. Wenn ein FB gerade dabei ist, Daten abzufragen und währenddessen eine Datenanfrage erhält, so wird das letzte abgespeicherte Ergebnis zurückgegeben. Zirkelschlüsse erzeugen somit einen temporären Speicher, welcher gut für rekursive Filter oder andere komplexere Verarbeitungsmechanismen genutzt werden kann. Beispielsweise ist ein rekursives Tiefpassfilter möglich, das in Gleichung (5.1) beschrieben wird.

$$x'_{(t)} = (1 - \alpha) \cdot x_{(t)} + \alpha \cdot x'_{(t-1)} \tag{5.1}$$

Hierbei ist $x'_{(t)}$ das neue Ergebnis, $x_{(t)}$ das aktuelle Datum vom Sensor und $x'_{(t-1)}$ das Ergebnis der vorherigen Berechnung. α ist der Gewichtungsfaktor des Tiefpasses.

5.3.2.3 Grafische Oberfläche

Für eine einfache und schnelle Erstellung einer Verarbeitungskette für die Manipulation der Sensordaten wurde eine grafische Oberfläche in der Form eines Node-Editors entwickelt. Jeder Knoten entspricht hier einem Verarbeitungsschritt, der von einem Funktionsblock durchgeführt wird. Kompatible Dateneingänge und Datenausgänge können direkt per Drag-and-drop miteinander verbunden werden, um die Verarbeitungskette zu erstellen. Der Ausgang eines jeden Knoten kann gleichzeitig mit mehreren Dateneingängen anderer Knoten verbunden werden. Die grafische Oberfläche des Node-Editors ist in Abbildung 5.25 dargestellt. Die Knoten wurden so entworfen, dass diese je nach Art der Verarbeitung eine andere Farbe haben. Dies soll die Übersichtlichkeit und Navigation innerhalb des Editors verbessern. Sobald das Setup für die Verarbeitung der Sensordaten erstellt wurden, kann daraus das funktionale Modell generiert werden, welches die Sensordaten verändert.

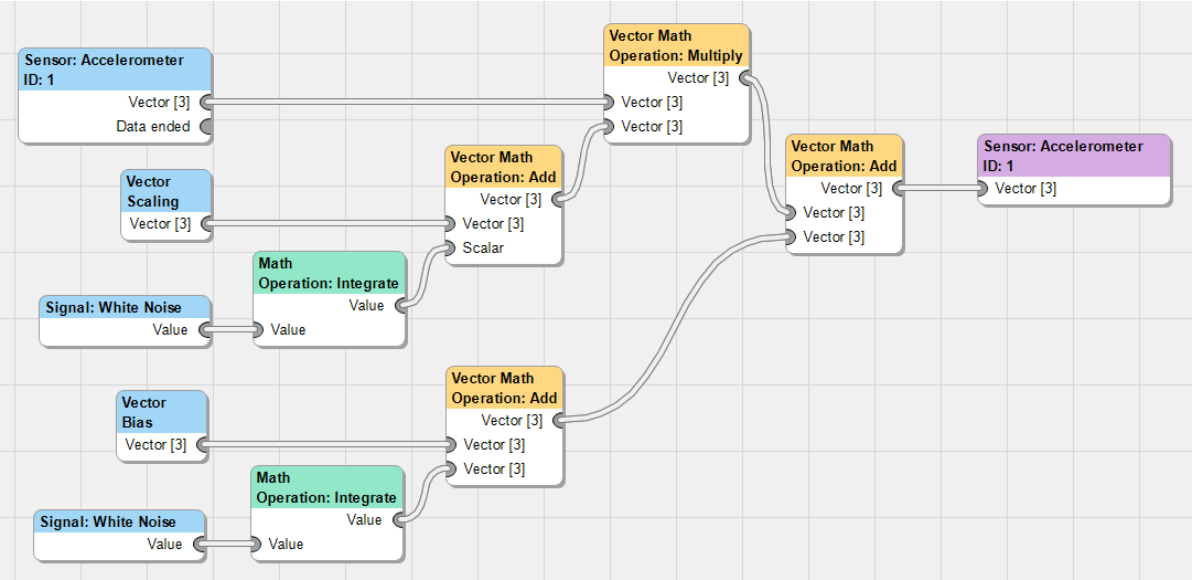


Abbildung 5.25: grafische Benutzeroberfläche des Node-Editors für die Manipulation der Sensordaten

Um die einfache Nutzbarkeit des Frameworks zu demonstrieren und zu verifizieren, wurde das Plugin der SiL-Plattform erweitert, sodass es das Framework für die Manipulation der Daten nutzen kann. Damit lassen sich dann die Versuchsaufbauten, die in Unterabschnitt 5.3.1 genannt wurden, umsetzen. Explizit umgesetzt wurden Aufbau 1 und Aufbau 2. Über den Graphen lassen sich die veränderten Sensordaten wieder in den Sensor einspeisen, um diesen auf sein Verhalten hin zu untersuchen. Es ist jedoch auch möglich, den Graphen zu nutzen, um aufgezeichnete Sensordaten einzulesen und die veränderten Daten für eine spätere Verwendung wieder abzuspeichern.

Der Graph zum Anzeigen der Sensordaten wurde so erweitert, dass die originalen und die veränderten Sensordaten gleichzeitig angezeigt werden können. Dies ermöglicht es, direkt zu inspizieren, wie die Sensordaten manipuliert werden. Ein Beispiel des Graphen ist in Abbildung 5.26 zu sehen.

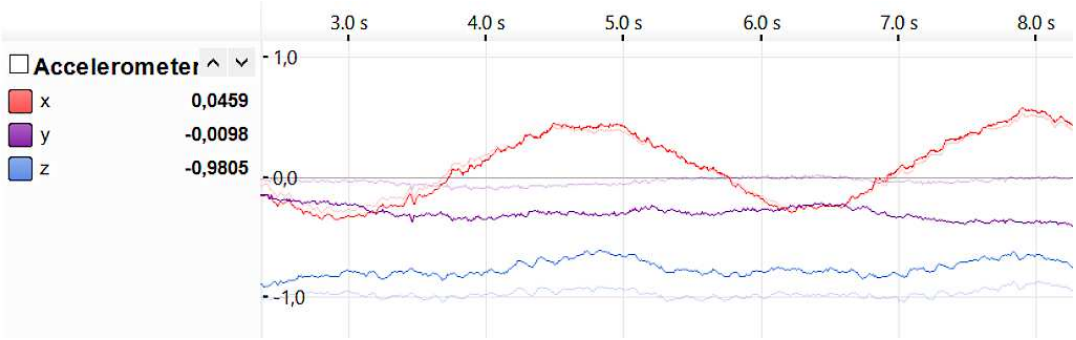


Abbildung 5.26: Graph zum Anzeigen der originalen und der veränderten Sensordaten

In der Darstellung werden die originalen Sensordaten blass dargestellt. Die veränderten Sensordaten werden in kräftigen Farben dargestellt. Welcher der Datensätze in kräftigen Farben erscheint, hängt davon ab, ob die originalen oder veränderten Daten in den Sensor eingespeist werden sollen.

5.3.3 Anwendungsbeispiele

Das Framework für die Manipulation und Augmentation von Sensordaten soll eine möglichst große Bandbreite an Veränderungen durchführen können. Mit den hierfür entworfenen Funktionsblöcken ist es möglich, zahlreiche einfache und komplexe Verarbeitungsschritte zu generieren. Einige mögliche Anwendungsbeispiele für das Framework werden in diesem Kapitel beschrieben. Obwohl das Framework für die Veränderung der Daten aus Inertialsensoren entworfen wurde, lassen sich die meisten der Verarbeitungsschritte auch auf Daten aus anderen Quellen anwenden. Beispielsweise auf die Daten von Radar- oder LIDAR-Sensoren.

5.3.3.1 Erzeugung von Rauschen

Ein wichtiger Einflussfaktor bei der Nutzung von MEMS-Inertialsensoren ist das Rauschen der Sensordaten. Durch die physikalischen Eigenschaften der Sensoren entsteht bei den Messungen mit MEMS-Sensoren oftmals ein pinkes $(1/f)$ Rauschen oder rotes $(1/f^2)$ Rauschen. Besonders das rote Rauschen ist durch die thermische Brownsche Bewegung der Teilchen prominent vorhanden. Unkorreliertes weißes Rauschen lässt sich durch einen Funktionsblock generieren. Rotes Rauschen lässt sich durch die Integration

des weißen Rauschens generieren [A128]. Das pinke Rauschen kann durch die Verwendung eines IIR-Filters auf dem weißen Rauschen angenähert werden [A129]. In [A130] wurde gezeigt, dass die Annäherung des pinken Rauschens mittels IIR Filters ausreichend genau ist.

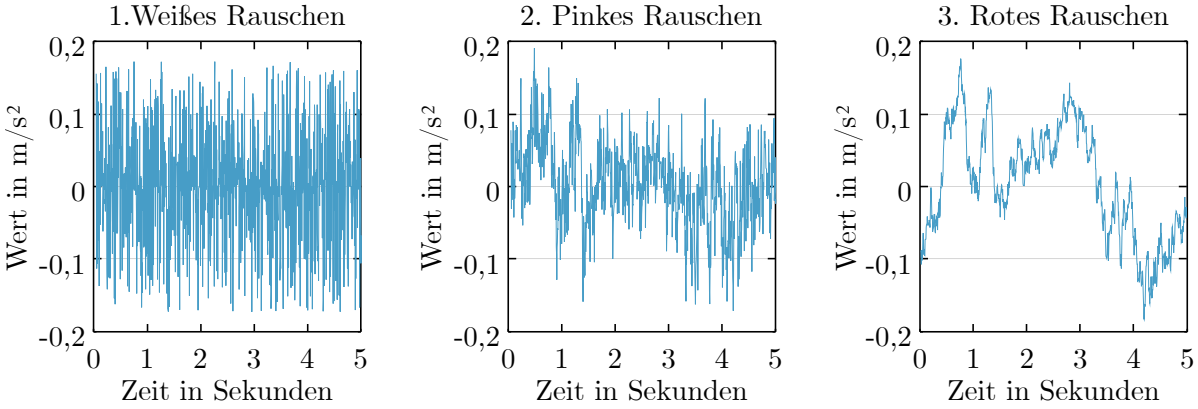


Abbildung 5.27: Weißes, pinkes und rotes Rauschen

In Abbildung 5.27 sind die drei genannten Färbungen von Rauschen, die mittels des Frameworks generiert wurden, dargestellt. Das pinke Rauschen und das rote Rauschen lassen sich jeweils mit zwei Funktionsblöcken erzeugen, das weiße Rauschen mit einem.

5.3.3.2 Simulation typischer MEMS-Sensor Fehler

Für die Evaluation von Sensorfusionsalgorithmen ist es besonders wichtig, zu bewerten, wie robust diese gegenüber der Messfehler der Sensoren sind. Bei der Verwendung von Hardware müssten die verwendeten Prototypen sorgfältig ausgesucht werden, um die gesamte Bandbreite an möglichen Fehler abzudecken. Hierfür ist potenziell die Anschaffung von sehr vielen Sensoren nötig. Mit dem Framework ist es möglich, die häufigsten Fehler der MEMS-Inertialsensoren zu modellieren und mit echten Sensordaten zu vermischen. Dadurch wird die Evaluation der Sensorfusionsalgorithmen deutlich einfacher und effizienter. Abgesehen vom Rauschen, welches in Unterunterabschnitt 5.3.3.1 erklärt wurde, treten, wie in Abschnitt 3.6 erwähnt, folgende Fehler häufig bei MEMS-Inertialsensoren auf [A74]:

- Bias
- Skalierungsfehler
- Temperaturdrift
- Nichtlinearität
- Ausrichtungsfehler
- Übersprechen von Achsen

Der *Bias* sowie die *Skalierungsfehler* lassen sich durch die Addition eines konstanten Wertes (Bias) oder der Multiplikation mit einem konstanten Wert (Skalierungsfehler) realisieren. Der Temperaturdrift, der sowohl den Bias, als auch den Skalierungsfehler über die Zeit verändern kann, lässt sich jeweils durch die Addition von rotem Rauschen auf den konstanten Wert modellieren. In Abbildung 5.28 werden die Schritte aufgezeigt, die für die Modellierung von Bias und Skalierungsfehler notwendig sind.

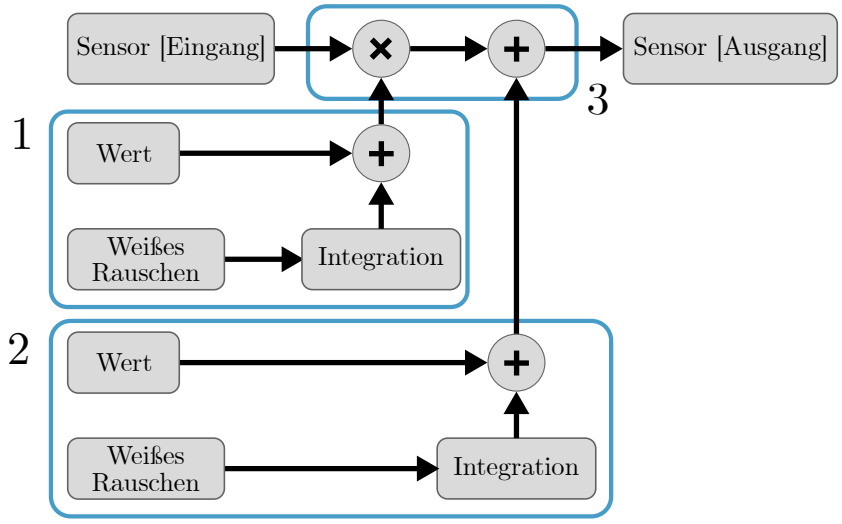


Abbildung 5.28: Funktionsblöcke zum Generieren von Bias und Skalierungsfehler inklusive Temperaturdrift

In den Bereichen **1** und **2** werden die Werte für den Bias und den Skalierungsfehler generiert. Diese Werte werden dann in Bereich **3** zu den Sensordaten hinzugefügt. Zuerst werden die Sensordaten mit dem Skalierungsfehler multipliziert, anschließend wird der Bias hinzugefügt.

Ein nicht lineares Verhalten der Sensordaten lässt sich durch einen speziellen Funktionsblock hinzufügen, welcher Eingangswerte in einem bestimmten Bereich auf Ausgangswerte projiziert. Dies geschieht über die Nutzung einer Bézierkurve, die durch vier Punkte definiert ist. Hierbei stellen die Endpunkte der Kurve den Maximalwert und den Minimalwert der Sensordaten dar.

In Abbildung 5.29 ist auf der linken Seite die Funktion gezeigt, mit der die Eingangswerte auf die Ausgangswerte projiziert werden. Die x-Achse ist der relative Wert des Sensordatums am Eingang. Die y-Achse zeigt den Wert, der von dem Funktionsblock für den jeweiligen Eingangswert ausgegeben wird. Die Kurve wird bestimmt durch die Endpunkte und die Kontrollpunkte P1 bei $[0,5; 0,25]$ und P2 bei $[0,75; 0,95]$. Auf der rechten Seite sieht man den Funktionsblock, der hinzugefügt werden muss, um ein nicht lineares Verhalten der Sensordaten zu erzeugen. In dem gezeigten Beispiel wird die Nichtlinearität vor der Multiplikation mit dem Skalierungsfehler hinzugefügt.

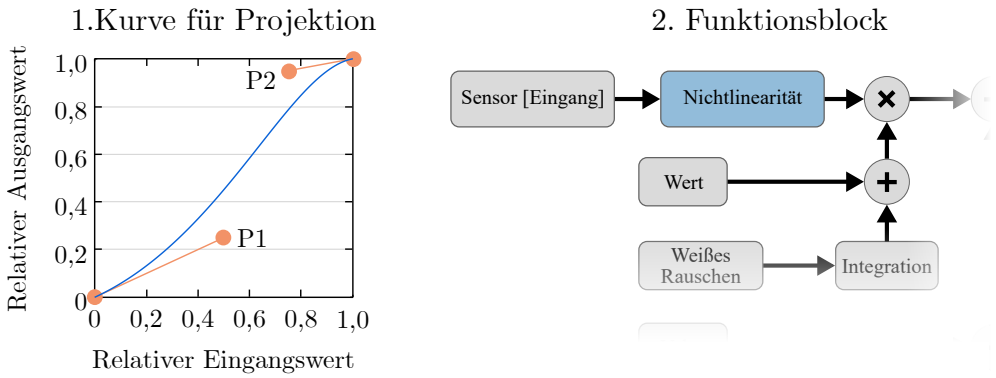


Abbildung 5.29: Funktionsblock zum Generieren von Nichtlinearität in den Sensordaten

Ausrichtungsfehler entstehen bei der Fertigung des Sensors und können in zwei Typen unterschieden werden. Der erste Typ ist der Ausrichtungsfehler im Gehäuse, welcher entsteht, wenn der Sensor mit einer leichten Verdrehung auf der Platine platziert wurde. Dieser Fehlertyp kann durch das Rotieren des Eingangsvektors der Sensordaten modelliert werden. Der zweite Fehler ist der orthogonale Ausrichtungsfehler. Hierbei stehen die Achsen des Sensors nicht exakt senkrecht aufeinander. Um diesen Fehler zu modellieren, müssen die Achsen separat rotiert werden, was von einem speziellen Funktionsblock durchgeführt werden kann. In Abbildung 5.30 ist dargestellt, wie der Gehäuse-Ausrichtungsfehler zu den Sensordaten hinzugefügt werden kann. Die Rotation der Achsen geschieht nach dem Hinzufügen der anderen Fehler, damit diese sich ausschließlich auf die jeweilige Achse auswirken.

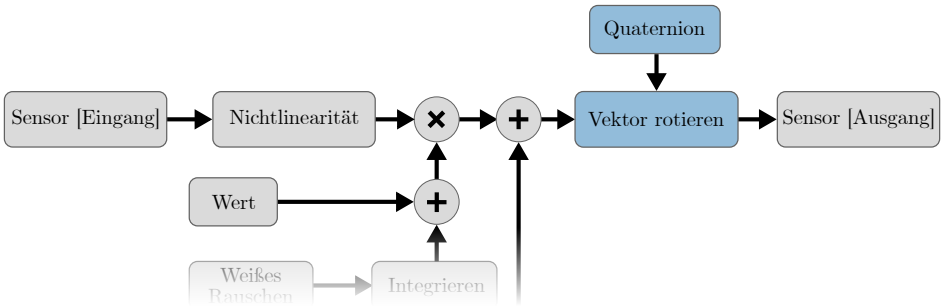


Abbildung 5.30: Generieren des Package-Ausrichtungsfehlers

Der letzte Fehler ist das Übersprechen von Achsen. Bei diesem Fehler werden Beschleunigungen oder Drehungen auf den Achsen gemessen, die senkrecht zu der Beschleunigung stehen und diese nicht messen können sollten. Dieser Fehler wird durch mehrere Faktoren hervorgerufen, beispielsweise auch durch den Ausrichtungsfehler. Weiterhin können auch Ungenauigkeiten bei der Fertigung oder ein elektrisches Übersprechen der Leitungen diesen Fehler verursachen. Um diesen Fehler zu modellieren, können separiert vom Ausrichtungsfehler Teilwerte einer Achse mit den Messwerten der anderen Achsen kombiniert werden.

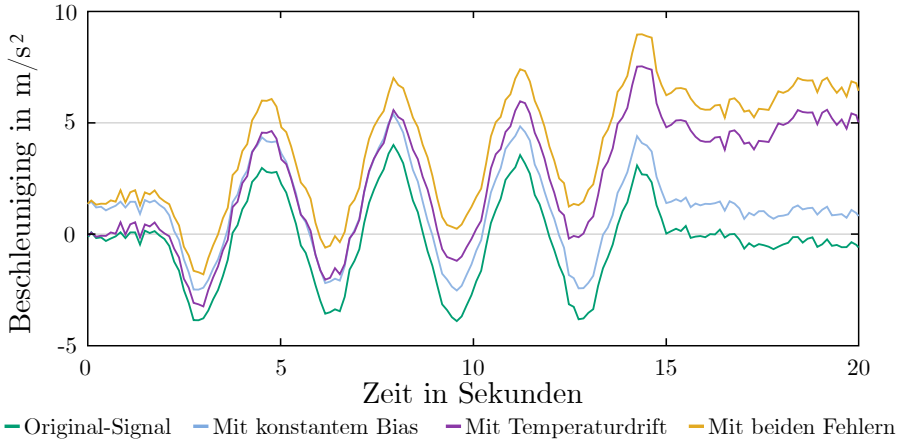


Abbildung 5.31: Generierte Fehler auf x-Achse für Bias und Temperaturdrift

In Abbildung 5.31 ist beispielhaft dargestellt, wie die veränderten Sensorsignale aussehen können, wenn ein Bias inklusive Temperaturdrift hinzugefügt wird. Das hellblaue Signal zeigt das Sensorsignal mit Bias, das Signal in Lila zeigt die Sensordaten mit Temperaturdrift aber ohne initialen Bias. Bei dem gelben Signal sind beide Fehler vorhanden. Die in Abbildung 5.31 gezeigten Fehler sind normalerweise geringer als dargestellt und dienen hier einer besseren Erkennbarkeit.

5.3.3.3 Augmentation von Sensordaten für Gestenerkennung

Intelligente Sensoren werden oftmals auch für Anwendungen verwendet, die Gestenerkennung nutzen, um mit Nutzer:Innen zu interagieren. Beispielsweise in Smartpens, für welche in Abschnitt 4.4 eine Bewertungsmethode entwickelt wurde. Bei diesen Anwendungen kann entweder die Vorverarbeitung der Daten oder die Gestenerkennung selbst direkt auf dem intelligenten Sensor durchgeführt werden. Bei der Gestenerkennung werden viele Messungen der Gesten benötigt, um die Algorithmen für die Gestenerkennung zu trainieren und zu verifizieren [A124]. Damit die Gesten unter möglichst vielen Bedingungen zuverlässig erkannt werden, muss das Training der Gestenerkennung mit Daten

aus einer Vielzahl an Situationen durchgeführt werden. Beispielsweise, wenn die nutzende Person läuft, Fahrrad fährt oder sich in einem Zug befindet. Alle diese Daten aufzunehmen ist sehr aufwendig und zeitintensiv. Um diesen Prozess zu beschleunigen, ist es mit dem entwickelten Framework möglich, die Daten von einer Geste mit denen von unterschiedlichen Situationen zu überlagern und somit neue Daten für das Training und die Verifikation der Gestenerkennung zu generieren. Diese Methodik wird Augmentation genannt und ebenfalls bei bildverarbeitenden Anwendungen genutzt [A41]. Die Daten können hierbei separat voneinander aufgenommen und beliebig kombiniert werden.

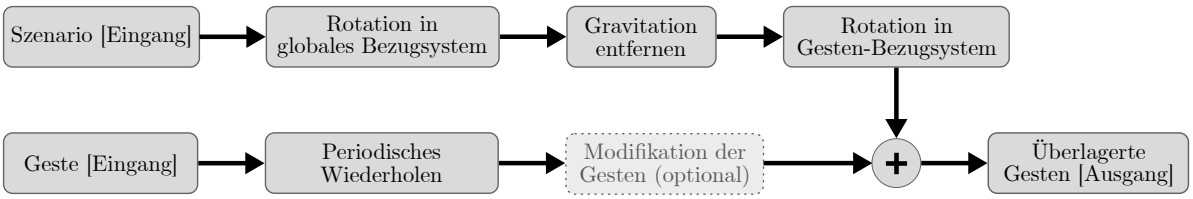
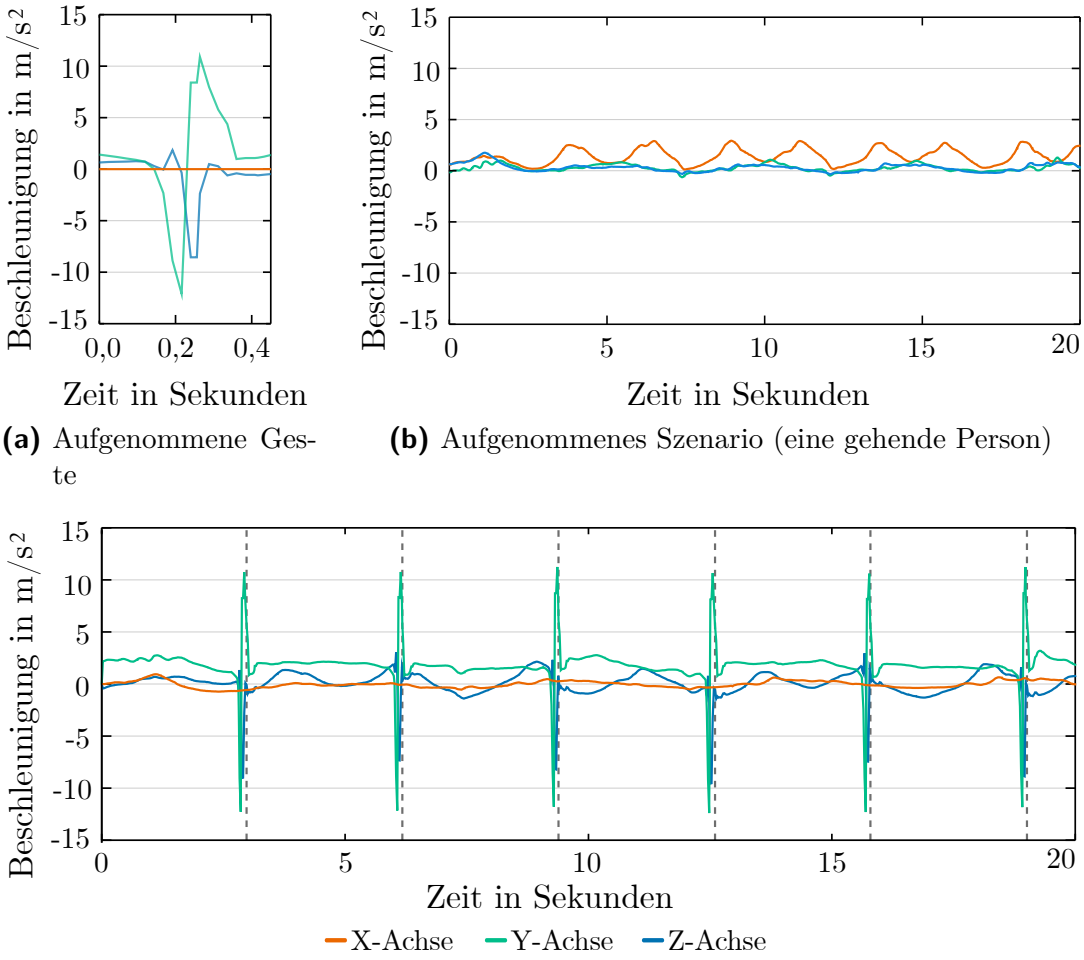


Abbildung 5.32: Verarbeitungsschritte für die Überlagerung von Sensordaten

Ein genereller Überblick, wie diese Daten erzeugt werden können, ist in Abbildung 5.32 gezeigt. Für die Augmentation der Sensordaten müssen zwei primäre Arbeitsschritte ausgeführt werden. Zuerst muss die aufgenommene Geste periodisch wiederholt werden. Dies ist notwendig, da die Aufnahmen der Szenarien deutlich länger als die Aufnahmen der Gesten sind. Je öfter eine Geste innerhalb eines Szenarios wiederholt wird, desto besser kann diese letztendlich erkannt werden. Beispielsweise gibt es Unterschiede bei der Gestenerkennung, wenn das Szenario von einer laufenden Person aufgenommen wurde und die Geste gerade beim Auftreten oder mitten im Schritt erkannt werden soll. Durch die Wiederholung der Geste kann sichergestellt werden, dass der Algorithmus für beide Situationen trainiert wird. Im zweiten Schritt müssen die Daten des Szenarios so aufbereitet werden, dass sie für die Überlagerung der Sensordaten genutzt werden können. Wenn die Aufnahmen von einem Beschleunigungssensor stammen, muss die Gravitation entfernt werden, da diese sonst zweifach in dem Ergebnis vorhanden wären. Weiterhin müssen die Daten des Szenarios vor der Überlagerung mit der Geste in das Bezugssystem der aufgenommenen Geste transformiert werden.

Die Wiederholung der Geste kann in dem Framework mit einer optionalen Verzögerung zwischen jeder Wiederholung realisiert werden. Für das Padding zwischen den Wiederholungen der Geste sind mehrere Optionen möglich, etwa Zero-Padding oder eine Interpolation zwischen Anfangs- und Endwert der wiederholten Gesten. In Abbildung 5.33(a) werden die Daten einer Geste und in Abbildung 5.33(b) die Daten eines Szenarios dargestellt. Das Ergebnis der *Superposition* ist in Abbildung 5.33(c) zu sehen. Die Geste wird periodisch nach 120 Samples wiederholt. Bei den Daten handelt es sich um die lineare Beschleunigung, aus der die Erdgravitation bereits entfernt wurde.



(a) Aufgenommene Geste

(b) Aufgenommenes Szenario (eine gehende Person)

(c) Ergebnis der Überlagerung von Geste und Szenario

Abbildung 5.33: Daten und Ergebnisse bei der Augmentation von Sensordaten

Die senkrecht gestrichelten Linien in Abbildung 5.33(c) zeigen an, wenn der Algorithmus für die Gestenerkennung die in Abbildung 5.33(a) gezeigte Geste erkennt. Wie zu sehen ist, werden die Gesten bei allen Wiederholungen korrekt erkannt. Das Framework kann also gut dazu verwendet werden, um die Robustheit der Gestenerkennung zu beurteilen oder die Gestenerkennung mit den Daten zu trainieren.

5.3.3.4 Vergleich von Sensordatenverarbeitungsschritten

Ein weiterer sinnvoller Einsatzbereich für das Framework sind Szenarien, in denen es für die Lösung einer Aufgabe, wie etwa der Extraktion von Features aus den Daten, mehrere Lösungswege gibt. Mit dem Framework ist es möglich, diese Lösungen auf ihre Machbarkeit hin zu untersuchen, bevor sie auf dem intelligenten Sensor implementiert werden. Hierfür kann die Verarbeitung der Daten in dem Framework durchgeführt werden, bevor diese Daten dann an den Sensor gesandt werden. In Abbildung 5.34 sind als Beispiel zwei Methoden für die Extraktion der Beschleunigung des Smartphones gezeigt, die für die Erkennung einer Schüttel-Geste genutzt werden kann.

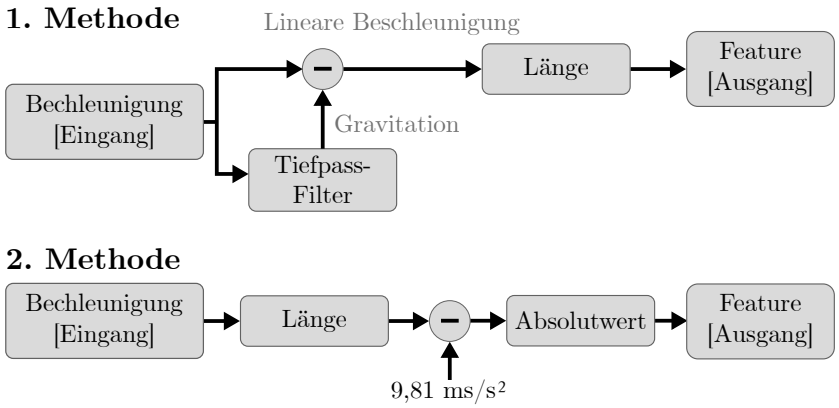


Abbildung 5.34: Verarbeitungsschritte für die Überlagerung von Sensordaten

Die erste Methode nutzt einen Tiefpass-Filter, um die Erdbeschleunigung zu entfernen und berechnet aus dem Ergebnis die Länge des Vektors. In der zweiten Methode wird die Länge des Vektors berechnet und davon die Gravitationskonstante abgezogen. Vom Ergebnis ist der absolute Wert das Ergebnis. Beide Methoden ermitteln dasselbe Feature, die Ergebnisse unterscheiden sich jedoch leicht, wie Abbildung 5.35 zu sehen ist.

Bei der Erkennung der Geste wird deutlich, dass sich der Zeitpunkt je nach Methode unterscheidet. Dies veranschaulicht gut, wie das Framework dafür genutzt werden kann, Verarbeitungsschritte zu untersuchen, bevor diese in der Firmware implementiert werden. Anhand der Ausgaben des Sensors und der visuellen Darstellung lassen sich die Ergebnisse und Unterschiede einfach vergleichen.

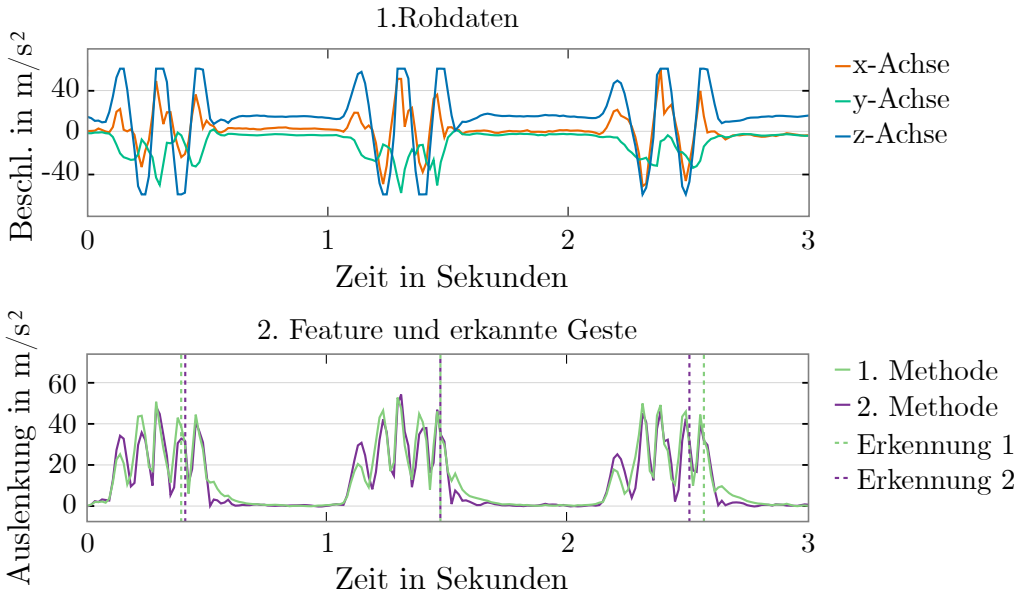
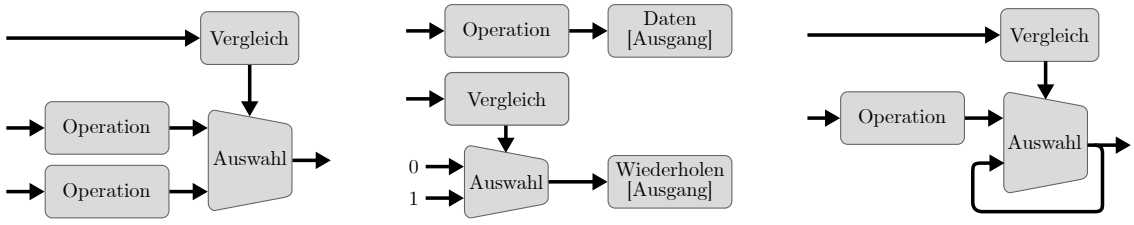


Abbildung 5.35: Verarbeitungsschritte für die Überlagerung von Sensordaten

5.3.3.5 Turing-Vollständigkeit

Die in dem vorgestellten Framework enthaltenen Funktionsblöcke reichen aus, um mithilfe des Frameworks jede berechenbare Funktion zu implementieren. Das vorgestellte Framework ist also Turing-vollständig. Dies kann anhand des *Structured program theorem* [A131], [A132] gezeigt werden. Hierfür werden drei Eigenschaften benötigt: sequentielle Ausführung von Operationen, bedingte Ausführung und bedingte Wiederholungen. Die sequentielle Ausführung von Operationen ist durch das „Pipes und Filter"-Architekturmuster gegeben.



(a) Bedingtes Ausführen

(b) Wiederholtes Ausführen mit Bedingung

(c) Speichern von Variablen oder Zustand

Abbildung 5.36: Elemente für Turing-Vollständigkeit des Frameworks

Ein bedingtes Ausführen lässt sich durch Funktionsblöcke für Vergleiche und Auswahl von Werten realisieren. Dies ist in Abbildung 5.36(a) dargestellt. Bedingte Wiederholungen sind möglich, indem man dem Framework als Datenausgang mitteilt, ob die aktuellen Daten nochmals abgefragt werden sollen, ohne die nächsten Eingangsdaten zu

verwenden. Hierdurch kann eine Berechnung beliebig oft für dieselben Daten wiederholt werden. Dies ist in Abbildung 5.36(b) dargestellt. Das Speichern von Zuständen oder Variablen kann, wie in Abbildung 5.36(c) dargestellt, durch den in Unterunterabschnitt 5.3.2.2 erklärte temporären Speicher implementiert werden. Mittels dieser vier Blöcke lässt sich entsprechend [A132] jede berechenbare Funktion errechnen.

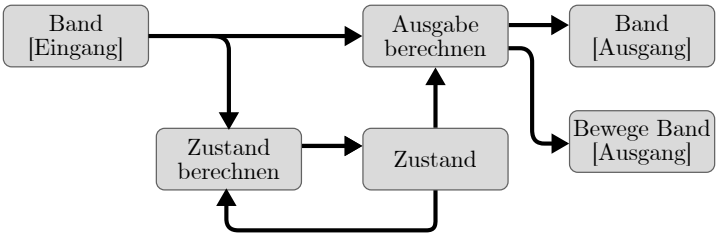


Abbildung 5.37: Beispiel einer minimalen Turing-Maschine

Ein simples abstraktes Modell einer Turing-Maschine ist in Abbildung 5.37 gezeigt. Die Dateneingänge und Datenausgänge fungieren hier als Band. Der Block „Bewege Band [Ausgang]“ bestimmt, in welche Richtung und wie weit das Band bewegt werden soll. Ausgänge und der nächste Zustand werden anhand von Eingangsdaten und aktuellem Zustand berechnet. Der Endzustand kann über ein spezielles Datum an einem der Ausgänge ausgegeben werden.

5.3.4 Zusammenfassung

Das in diesem Abschnitt vorgestellte Framework für die Veränderung und Augmentation von Inertialsensor-Daten stellt ein mächtiges Werkzeug bereit, mit dessen Hilfe es möglich ist, das Verhalten von Sensorfusionsalgorithmen oder der Vorverarbeitung von Sensordaten gezielt zu untersuchen. In Kombination mit der SiL-Plattform lässt sich so die Firmware direkt auf der Hardware des intelligenten Sensors gezielt und reproduzierbar untersuchen. Anhand der gezeigten Beispiele konnte veranschaulicht werden, dass das Framework in einer Vielzahl unterschiedlicher Szenarien bei der Bewertung und Entwicklung von Sensorsubsystemen genutzt werden kann. Dies geht von der einfachen Manipulation der Daten über die Simulation von Sensorfehlern bis hin zur Generierung von neuen Sensordaten durch die Kombination mehrerer Aufnahmen. Es konnte gezeigt werden, dass es mittels des Frameworks möglich ist, jede berechenbare Manipulation der Daten durchzuführen. Das Framework ist nicht auf Daten aus Bewegungssensoren beschränkt, sondern kann auch für die Manipulation von anderen Daten oder Messgrößen verwendet werden. Etwa für die Daten von Radar- oder LIDAR-Sensoren von selbstfahrenden Autos. Dies kann besonders bei der Validierung der Straßentauglichkeit helfen, um Fehlerquellen oder auch nur möglichst viele Situationen während der Fahrt zu simulieren [A133].

Mittels des implementierten Prototyps in Form einer Eclipse-Erweiterung mit grafischer Oberfläche lassen sich die Manipulationsschritte schnell und einfach generieren. Die veränderten Daten sind mittels des Graphen direkt erkennbar und können mit den Originaldaten verglichen werden.

5.3.5 Zukünftige Entwicklung

Zukünftig soll es auch möglich sein, in dem Framework eigene Funktionsblöcke per Plug-ins hinzuzufügen. Dies soll eine einfache Implementierung häufig genutzter Kombinationen von Manipulationsschritten oder anderer komplexer Verarbeitungsschritte ermöglichen. Weiterhin ist es sinnvoll, die Durchführung der Manipulation durch das Framework für andere Systeme oder in anderen Programmiersprachen zu implementieren. So kann das Framework etwa auf Smartphones genutzt werden, um zu untersuchen, wie sich eine Anwendung bei unterschiedlichen Qualitäten der Sensordaten verhält, ohne eine große Anzahl an unterschiedlichen Smartphones zum Testen zu benötigen. Dies knüpft an das Ziel der Arbeiten aus Abschnitt 4.1 an und kann die gesammelten Erkenntnisse und Daten aus diesen als Grundlage für die Modellierung der Sensorfehler verwenden. Weiterhin ist eine denkbare Erweiterung, das Framework in einer Skriptsprache wie Python zu implementieren. Dies würde zur Vereinfachung der automatischen Manipulation einer großen Anzahl an Datensätzen beitragen.

5.4 Zusammenfassung

In diesem Kapitel wurde der Schwerpunkt auf Methoden und Untersuchungen auf der Firmware-Ebene gelegt. Hierbei wurden zunächst die funktionalen und extrafunktionalen Eigenschaften mehrere Sensorfusionsalgorithmen mit unterschiedlichen Datentypen untersucht. Aus den Ergebnissen der Untersuchungen konnte die Erkenntnis gewonnen werden, dass die Verwendung von 32 Bit-Festkommazahlen nicht nur deutlich die Rechenzeit und damit den Energieverbrauch verringern kann, sondern auch die Ergebnisse aus den Sensorfusionsalgorithmen qualitativ gleichwertig sind. Dies ist auf die Genauigkeit und den verwendeten Wertebereich zurückzuführen. Die Untersuchungen der Einflüsse auf die Qualität der Ergebnisse der Sensorfusionsalgorithmen zeigten, dass der verwendete Algorithmus, zumindest im Kontext der Verwendung mit menschlichen Interaktionen, nur einen geringen Einfluss hat. Auch hier hat sich ein Potenzial für eine deutliche Reduktion des Rechenaufwandes offenbart, welcher nur zu geringfügigen Qualitätsverlusten führt. Die in dieser Untersuchung gewonnen Erkenntnisse können gerade bei der Entwicklung von Anwendungen für ressourcenbeschränkte eingebettete Systeme wertvolle Informationen liefern, die dabei helfen, die passendsten Sensorfusionsalgorithmen und Datenformate auszuwählen.

Weiterhin wurde für die Entwicklung auf der Firmware-Ebene eine Framework vorgestellt, welches es Entwickler:Innen ermöglicht, unterschiedliche Charakteristiken der verwendeten Sensoren direkt zur Laufzeit auf der Hardware zu untersuchen. Hierfür kann in einem Editor anhand von Funktionsblöcken eine Verarbeitungskette der Sensordaten erstellt werden. Aufgenommene und durch das Framework veränderte Sensordaten können dann mithilfe der Sensor-in-the-Loop-Plattform direkt in den Sensor injiziert werden. Das Framework kann in zahlreichen Entwicklungsszenarien eingesetzt werden: von der Simulation von Sensorfehlern, über die Augmentation von Sensordaten für Training und Verifizierung von Algorithmen zur Gestenerkennung bis hin zur Untersuchung unterschiedlicher Vorverarbeitungsschritte.

Beide in diesem Kapitel vorgestellte Arbeiten liefern einen Beitrag dazu, die für einen bestimmten Einsatzzweck optimalen Sensorfusionsalgorithmen auszuwählen und Entwickler:Innen bei der Programmierung und Verifikation von Anwendungen zu unterstützen, in denen Sensordaten verwendet werden.

6 Zusammenfassung der Dissertationsschrift

Mit den in dieser Dissertationsschrift beschriebenen Arbeiten konnten nützliche Konzepte, Methoden und Anwendungen für die Bewertung von Inertialsensorsubsystemen vorgestellt werden, die Entwickler:Innen dabei helfen sollen, einfacher zuverlässige und genaue Anwendungen zu entwickeln. Sie bilden eine solide Basis, um Entwickler:Innen eine Hilfestellung bei der Entwicklung von Anwendungen zu geben, die Daten aus Bewegungssensoren nutzen. Es wurden Methoden, Konzepte und Anwendungen vorgestellt, mittels derer es auf einfache Weise möglich ist, die Qualität der Sensordatenfusion oder des Inertialsensorsubsystems zu beurteilen. Weiterhin wurden Ergebnisse von Untersuchungen der funktionalen und extrafunktionalen Eigenschaften von Sensorfusionsalgorithmen vorgestellt. Die Dissertationsschrift ist in zwei Hauptkapitel unterteilt, die sich der Thematik aus verschiedenen Perspektiven annehmen. Zunächst wird die Anwendungsebene betrachtet. In dieser wird während der Entwicklung auf bereits bestehende Sensorsubsysteme zurückgegriffen, etwa bei Smartphones. Hierbei kommt es darauf an, die Qualität der erhaltenen Daten einzuschätzen zu können und die Anwendungen entsprechend zu entwickeln. Anschließend wird die Firmware-Ebene betrachtet, auf der es darum geht, möglichst energieeffiziente und trotzdem qualitativ gute Sensordatenfusion zu implementieren. Hierfür wurden sowohl Untersuchungen von Sensorfusionsalgorithmen als auch Anwendungen zur Unterstützung bei Entwicklung und Verifikation von diesen vorgestellt.

Das erste Hauptkapitel befasst sich mit Konzepten, Methoden und Anwendungen für die Analyse der Inertialsensorsubsysteme auf Anwendungsebene. Auf dieser Abstraktionsebene können Entwickler:Innen auf bereits implementierte Sensorsubsysteme zugreifen und verarbeitete oder fusionierte Sensordaten verwenden. Dies ist beispielsweise in Smartphones, aber auch in anderen Wearables und Geräten wie Smartpens der Fall. Für diesen Zweck wurden zwei Methoden vorgestellt, mittels derer sich solche Systeme, beziehungsweise die Qualität von deren Sensorsubsystem gut bewerten lassen. Zuerst wurde die MEMS-Score-Anwendung vorgestellt, welche es auf einfache Weise und ohne die Verwendung von teurem zusätzlichen Equipment ermöglicht, das Sensorsubsystem zu bewerten. Es wurde ermöglicht, mittels der Anwendung Daten über die Qualität der verwendeten Sensorsubsysteme zu sammeln, um Richtwerte für die Entwicklung auf sensordatenbasierter Anwendungen zu ermitteln. Weiterhin wurden für die Bewertung

von digitalen Zeigegeräten Konzepte und Methden vorgestellt, mittels derer es möglich ist, quantitativ die Nutzbarkeit von digitalen Zeigegeräten zu ermitteln. Hierfür wurden die Bewertungsmethoden so konzipiert, dass die ermittelten quantitativen Ergebnisse der subjektiven Bewertung der Testpersonen entsprechen.

Das zweite Hauptkapitel befasst sich mit Konzepten, Methoden und Anwendungen für die Analyse von Inertialsensorsubsystemen auf der Firmware-Ebene. Zuerst wurden die Ergebnisse von Untersuchungen vorgestellt, in denen die funktionalen und extrafunktionalen Eigenschaften mehrerer Sensorfusionsalgorithmen für die Orientierung in Kombination mit unterschiedlichen Datentypen untersucht wurden. Hierbei wurden Rechenzeit, Speicherverbrauch, Genauigkeit der Ergebnisse sowie die Einflüsse von Sensorfusionsalgorithmus, Datentyp und Bewegungsgeschwindigkeit untersucht. Die Ergebnisse bieten eine große Unterstützung bei der Auswahl der für einen Einsatzzweck passendste Kombination aus Fusionsalgorithmus und Datenformat. Zuletzt wurde ein Framework vorgestellt, mit dessen Hilfe es auf einfache Weise möglich ist, bereits aufgenommene Sensordaten zu verändern, zu augmentieren oder zu verarbeiten und die veränderten Daten direkt in der Firmware auf dem Sensor zu testen. Dies soll Entwickler:Innen bei der Entwicklung von robusten und akkuraten Sensorsubsystemen unterstützen, indem es die Verfügbarkeit von geeigneten Daten verbessert.

6.1 Bewertung der Ergebnisse

Auf der Anwendungsebene ist es gelungen, mit der MEMS-Score-Anwendung eine Anwendung zu entwickeln, die es erlaubt, die Qualität des Sensorsubsystems zu ermitteln, ohne hierfür weiteres Equipment zu benötigen. Die Untersuchungen, die mit der MEMS-Score-Anwendung durchgeführt wurden, haben gezeigt, dass diese für die Bewertung des Sensorsubsystems geeignet ist. Das in der Anwendung implementierte Ranking erlaubt es Nutzer:Innen, die Qualität des eigenen Smartphones im Vergleich zu anderen Geräten zu ermitteln. Seit der Veröffentlichung der Anwendung konnten bisher 977 Messungen in die Datenbank aufgenommen werden, die Rückschlüsse auf die Qualität der Sensoren, die Verfügbarkeit unterschiedlicher (virtueller) Sensoren sowie deren Charakteristiken zulassen. Besonders bei den aufwendigeren Tests, die mehrere Testläufe benötigen, wurden jedoch oftmals nicht ausreichend Messungen durchgeführt, um eine zuverlässige Aussage über ein bestimmtes Gerät zu treffen. Um einen besseren Überblick zu erlangen, müsste die Anwendung eine größere Nutzer:Innenbasis erlangen und der Aufwand für die Durchführung der Tests reduziert werden. Die Wichtigkeit zuverlässiger Inertialsensoren ist auch von Google erkannt worden. So wurde der CTS-Verifier mittlerweile um eine Testmethode erweitert, die der MEMS-Score Anwendung sehr ähnlich ist und die Genauigkeit der virtuellen Sensoren für die Rotation bewertet [A134].

Bei den Bewertungsmethoden der digitalen Zeigegeräte konnte dargelegt werden, wie Tests für Geräte es ermöglichen können, anhand quantitativer Messwerte die subjektiven Bewertungen der Nutzer:Innen abzuschätzen. Es wurde anhand von Untersuchungen gezeigt, dass die quantitativen Unterschiede gemäß der Tests mit den subjektiven Bewertungen der Testpersonen übereinstimmen. Es hat sich jedoch auch gezeigt, dass die Bewertungen durch die Testpersonen nicht differenziert genug gewählt waren, um eine detailliertere Aussage über Korrelation zwischen Testergebnis und subjektiver Bewertung machen zu können.

Die auf der Firmware-Ebene vorgestellten Untersuchungen von Sensorfusionsalgorithmen in Kombination mit unterschiedlichen Datenformaten können Entwicklern:Innen dabei helfen, für das in Entwicklung befindliche System die richtige Kombination aus Algorithmus und Datenformat auszuwählen. In den Ergebnissen wird detailliert auf unterschiedliche Einflussfaktoren sowie Rechenaufwand und Speicherverbrauch der untersuchten Algorithmen eingegangen. In den Ergebnissen konnte gezeigt werden, dass die Sensorfusionsalgorithmen bei menschlicher Interaktion vergleichbare und vor allem nutzbare Ergebnisse liefern. Zentral ist hier jedoch anzumerken, dass dies nur für die Bestimmung der Orientierung bei menschlicher Interaktion gilt. Andere Untersuchungen haben gezeigt, dass das Kalman-Filter in zahlreichen technischen Prozessen, etwas bei UAVs zuverlässigere Ergebnisse liefert.

Das ebenfalls für die Firmware-Ebene vorgestellte Framework zur Daten-Augmentation ist in der Lage, für eine große Bandbreite an Anwendungsfällen eine Hilfe bei der Entwicklung von Sensorsubsystemen zu sein. Es konnte anhand konkreter Beispiele gezeigt werden, dass es möglich ist, Sensorfehler zu simulieren, Sensordaten zu augmentieren und Verarbeitungsschritte zu evaluieren, bevor diese in der echten Firmware implementiert werden. Weiterhin konnte gezeigt werden, dass das Framework Turing-vollständig ist und somit jede berechenbare Veränderung der Sensordaten vorgenommen werden kann. Die Turing-Vollständigkeit lässt jedoch keine Aussage darüber zu, wie einfach es ist, die Manipulation der Daten vorzunehmen. Dies muss sich noch in der Praxis zeigen. Es wurde Wert darauf gelegt, übliche und häufig vorkommende Veränderungen der Daten mit geringem Aufwand zu ermöglichen.

Die in dieser Dissertationsschrift vorgestellten Konzepte, Methoden und praktischen Anwendungen liefern eine solide Basis, um Entwickler:Innen zentrale Möglichkeiten zu eröffnen: Sie erleichtern die Entwicklung zuverlässiger Sensorsubsysteme und weiterer Anwendungen sowie Entscheidungen über die zu verwendenden Algorithmen. Zugleich wird es Entwickler:Innen erleichtert, die Algorithmen zu bewerten und zu verifizieren.

6.2 Ausblick

Mit den beschriebenen Arbeiten konnten nützliche Konzepte, Methoden und Anwendungen für die Bewertung von Inertialsensorsubsystemen vorgestellt werden, die Entwickler:Innen dabei helfen sollen, einfacher zuverlässige und genaue Anwendungen zu entwickeln. Es hat sich jedoch auch gezeigt, dass es an mehreren Stellen Potenzial für Verbesserungen gibt.

So hat sich bei der Auswertung der gesammelten Daten der MEMS-Score-Anwendung gezeigt, dass die Durchführung der komplexeren Tests für viele Nutzer:Innen zu zeitaufwändig war und diese dadurch nicht oft genug wiederholt wurden, um die Daten aus den Inertialsensorsubsystemen zuverlässig bewerten zu können. An dieser Stelle sollte die Anwendung noch überarbeitet werden. Bei den Bewertungsmethoden für die digitalen Zeigegeräte wäre es für zukünftige Untersuchungen sinnvoll, die Granularität der Fragen über die subjektive Einschätzung durch die Testpersonen zu verbessern. Dies würde nicht nur eine qualitative, sondern auch eine quantitative Aussage über die Nutzbarkeit der untersuchten digitalen Zeigegeräte ermöglichen.

Bei der Untersuchung der funktionalen und extrafunktionalen Eigenschaften der Sensorfusionsalgorithmen sind weitergehende Untersuchungen denkbar, die Ausführungszeit, Energieverbrauch und Speicherverbrauch in einem System bewerten, in welchem eine FPU verbaut ist. Weiterhin kann es hier auch sinnvoll sein, die Untersuchungen auf andere Arten von Sensorfusionsalgorithmen zu erweitern. Etwa auf die Extraktion der Gravitation aus den Daten des Beschleunigungssensors oder die Bestimmung der Position unter Verwendung von GPS.

Auch für das Framework zur Augmentation von Sensordaten bietet sich eine Weiterentwicklung an, die es Entwicklern erlaubt, eigene Funktionsblöcke einzubinden und zu nutzen. Weiterhin können Portierungen auf anderes Systeme oder in andere Programmiersprachen das Anwendungsgebiet des Frameworks drastisch erhöhen. Etwa für Android-Smartphones, um zur Laufzeit aufgenommene Sensordaten zu manipulieren und für die Evaluation von Anwendungen zu nutzen. Auch eine Portierung in Python bietet sich an, da sich so auch eine große Anzahl an Datensätzen problemlos verändern ließe.

Abgesehen von diesen Erweiterungen und Verfeinerungen gibt es noch viel Potenzial, weitere Bewertungsmethoden zu entwickeln oder andere Eigenschaften der Sensorsubsysteme zu untersuchen. Beispielsweise bei Methoden für die Bestimmung der Position, die etwa in UAVs genutzt werden. Dies ist sehr wichtig, da die Anzahl der verwendeten Sensoren und Anwendungsszenarien in der Zukunft noch deutlich steigen wird und Nutzer:Innen sich immer mehr darauf verlassen werden, dass die Systeme zuverlässig funktionieren. Die hier vorgestellten Arbeiten können dazu beitragen zukünftig dabei zu helfen, die Zuverlässigkeit zu erhöhen oder bei Systemintegratoren (Smartphone-Herstellern) das Bewusstsein zu stärken, damit die Qualität der Sensorsubsysteme steigt.

A Literaturverzeichnis

[A1] T. Seel, M. Kok und R. S. McGinnis, „Inertial Sensors—Applications and Challenges in a Nutshell,“ *Sensors*, Jg. 20, Nr. 21, 2020. DOI: 10.3390/s20216221. Adresse: https://www.mdpi.com/1424-8220/20/21/6221.

[A2] A. RajKumar, F. Vulpi, S. R. Bethi, H. K. Wazir, P. Raghavan und V. Kapila, „Wearable Inertial Sensors for Range of Motion Assessment,“ *IEEE Sensors Journal*, Jg. 20, Nr. 7, S. 3777–3787, 2020. DOI: 10.1109/JSEN.2019.2960320.

[A3] A. Godfrey, R. Conway, D. Meagher und G. ÓLaighin, „Direct measurement of human movement by accelerometry,“ *Medical Engineering & Physics*, Jg. 30, Nr. 10, S. 1364–1386, 2008, Special issue to commemorate the 30th anniversary of Medical Engineering & Physics, ISSN: 1350-4533. DOI: 10.1016/j.medengphy.2008.09.005.

[A4] L. Middendorf, R. Dorsch, R. Bichler, C. Strohrmann und C. Haubelt, „A Mobile Camera-Based Evaluation Method of Inertial Measurement Units on Smartphones,“ in *Internet of Things. IoT Infrastructures*, Cham: Springer International Publishing, 2016, S. 362–372, ISBN: 978-3-319-47075-7. DOI: 10.1109/TIM.2015.2498998.

[A5] A. Kos, S. Tomažič und A. Umek, „Evaluation of Smartphone Inertial Sensor Performance for Cross-Platform Mobile Applications,“ *Sensors*, Jg. 16, Nr. 4, 2016, ISSN: 1424-8220. DOI: 10.3390/s16040477. Adresse: https://www.mdpi.com/1424-8220/16/4/477.

[A6] Qualcomm developer network. „Gesture Recognition Dataset: Jester.“ (2022), Adresse: https://developer.qualcomm.com/software/ai-datasets/jester (besucht am 01. 03. 2022).

[A7] J. R. Blum, D. G. Greencorn und J. R. Cooperstock, „Smartphone Sensor Reliability for Augmented Reality Applications,“ in *Mobile and Ubiquitous Systems: Computing, Networking, and Services*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, S. 127–138.

[A8] R. Jackermeier und B. Ludwig, „Exploring the limits of PDR-based indoor localisation systems under realistic conditions,“ *Journal of Location Based Services*, Jg. 12, Nr. 3-4, S. 231–272, 2018. DOI: 10.1080/17489725.2018.1541330.

[A9] C. Brewer und M. Harrower. „COLORBREWER 2.0.“ (2021), Adresse: https://colorbrewer2.org (besucht am 04. 08. 2021).

[A10] Infineon. „Mensch-Maschine-Interaktion heute und in Zukunft.“ (2021), Adresse: https://www.infineon.com/cms/de/discoveries/human-machine-interaction/ (besucht am 19. 07. 2021).

[A11] S. Zimmermann. „Elektrische Messtechnik.“ (2021), Adresse: http://campus.fh-worms.de/~stzimmer/TI-3/Skript/Folien_1_40.pdf (besucht am 02. 12. 2021).

[A12] T. Sauer. „Digitale Signalverarbeitung.“ (2021), Adresse: https://www.fim.uni-passau.de/fileadmin/dokumente/fakultaeten/fim/lehrstuhl/sauer/geyer/DigiSig.pdf (besucht am 02. 12. 2021).

[A13] Bosch Sensortec. „Smart Sensor: BMF055.“ (2021), Adresse: https://www.bosch-sensortec.com/products/smart-sensors/bmf055.html (besucht am 03. 08. 2021).

[A14] Bosch Sensortec. „BHI260AP.“ (2023), Adresse: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bhi260ap-ds000.pdf (besucht am 27. 02. 2023).

[A15] STMicroelectronics. „The Onlife era of MEMS.“ (2023), Adresse: https://www.st.com/content/st_com/en/campaigns/ispu-ai-in-sensors.html (besucht am 27. 02. 2023).

[A16] Analog Devices. „Embedded Sensor Platform with AI Algorithms—Locally from Big Data to Smart Data.“ (2023), Adresse: https://www.analog.com/en/technical-articles/embedded-sensor-platform-withai-algorithms-locally-from-big-data-to-smart-data.html (besucht am 08. 03. 2023).

[A17] A. G. Cutti, A. Giovanardi, L. Rocchi und A. Davalli, „A simple test to assess the static and dynamic accuracy of an inertial sensors system for human movement analysis,“ in *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, 2006, S. 5912–5915. DOI: 10.1109/IEMBS.2006.260705.

[A18] M. Caruso, A. M. Sabatini, M. Knaflitz, M. Gazzoni, U. D. Croce und A. Cereatti, „Accuracy of the Orientation Estimate Obtained Using Four Sensor Fusion Filters Applied to Recordings of Magneto-Inertial Sensors Moving at Three Rotation Rates,“ in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2019, S. 2053–2058. DOI: 10.1109/EMBC.2019.8857655.

[A19] P. Patonis, P. Patias, I. N. Tziavos und D. Rossikopoulos, „A methodology for the performance evaluation of low-cost accelerometer and magnetometer sensors in geomatics applications,“ *Geo-spatial Information Science*, Jg. 21, Nr. 2, S. 139–148, 2018. DOI: 10.1080/10095020.2018.1424085.

[A20] L. Wang, Y. Hao, Z. Wei und F. Wang, „A calibration procedure and testing of MEMS inertial sensors for an FPGA-based GPS/INS system," in *2010 IEEE International Conference on Mechatronics and Automation*, 2010, S. 1431–1436. DOI: 10.1109/ICMA.2010.5589034.

[A21] Google Inc. „Using CTS Verifier." (2021), Adresse: https://source.android.com/compatibility/cts/verifier.html (besucht am 30. 09. 2021).

[A22] A. Gabel, S. Uellenbeck und T. Holz, „Android Applikation zur Bestimmung der Genauigkeit von Smartphone Sensoren," 2012. Adresse: https://www.syssec.ruhr-uni-bochum.de/media/emma/arbeiten/2013/02/18/ba_Gabel_Andreas.pdf (besucht am 30. 11. 2021).

[A23] F. Koushanfar, M. Potkonjak und A. Sangiovanni-Vincentelli, „Error models for light sensors by statistical analysis of raw sensor measurements," in *SENSORS, 2004 IEEE*, Bd. 3, 2004, S. 1472–1475. DOI: 10.1109/ICSENS.2004.1426465.

[A24] Q. Mourcou, A. Fleury, C. Franco, F. Klopcic und N. Vuillerme, „Performance Evaluation of Smartphone Inertial Sensors Measurement for Range of Motion," *Sensors*, Jg. 15, S. 23 168–23 187, Sep. 2015. DOI: 10.3390/s150923168.

[A25] Android Authority. „What is image stabilization? OIS, EIS, and HIS explained." (2023), Adresse: https://www.androidauthority.com/image-stabilization-1087083/ (besucht am 08. 03. 2023).

[A26] STMicroelectronics. „Optical Image Stabilization (OIS)." (2023), Adresse: https://www.st.com/content/ccc/resource/technical/document/white_paper/c9/a6/fd/e4/e6/4e/48/60/ois_white_paper.pdf/files/ois_white_paper.pdf/jcr:content/translations/en.ois_white_paper.pdf (besucht am 08. 03. 2023).

[A27] ISO, *9241-400: Ergonomics of human–system interaction – Part 400: Principles and requirements for physical input devices*, ISO, 2007.

[A28] S. A. Douglas, A. E. Kirkpatrick und I. S. MacKenzie, „Testing pointing device performance and user assessment with the ISO 9241, Part 9 standard," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, ACM, 1999, S. 215–222. DOI: 10.1145/302979.303042.

[A29] I. S. MacKenzie, T. Kauppinen und M. Silfverberg, „Accuracy measures for evaluating computer pointing devices," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 2001, S. 9–16. DOI: 10.1145/365024.365028.

[A30] R. W. Soukoreff und I. S. MacKenzie, „Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI," *International Journal of Human-Computer Studies*, Jg. 61, Nr. 6, S. 751–789, 2004, Fitts' law 50 years later: applications and contributions from human-computer interaction, ISSN: 1071-5819. DOI: 10.1016/j.ijhcs.2004.09.001.

[A31] S. Zhai, „Characterizing computer input with Fitts' law parameters: The information and non-information aspects of pointing," *International Journal of Human-Computer Studies*, Jg. 61, S. 791–809, Dez. 2004. DOI: 10.1016/j.ijhcs.2004.09.006.

[A32] J. Accot und S. Zhai, „Performance Evaluation of Input Devices in Trajectory-Based Tasks: An Application of the Steering Law," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Ser. CHI '99, Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 1999, S. 466–472, ISBN: 0201485591. DOI: 10.1145/302979.303133.

[A33] F. Alam, Z. ZhaiHe und H. Jiajia, „A Comparative Analysis of Orientation Estimation Filters using MEMS based IMU," in *2nd International Conference on Research in Science, Engineering and Technology*, 2014. DOI: 10.15242/iie.e0314552.

[A34] P. Gui, L. Tang und S. Mukhopadhyay, „MEMS based IMU for tilting measurement: Comparison of complementary and kalman filter based data fusion," in *2015 IEEE 10th Conference on Industrial Electronics and Applications (ICIEA)*, 2015, S. 2004–2009. DOI: 10.1109/ICIEA.2015.7334442.

[A35] H. Teague, „Comparison of Attitude Estimation Techniques for Low-cost Unmanned Aerial Vehicles," 2016. Adresse: https://arxiv.org/abs/1602.07733.

[A36] A. Cavallo, A. Cirillo, P. Cirillo u. a., „Experimental Comparison of Sensor Fusion Algorithms for Attitude Estimation," *IFAC Proceedings Volumes*, Jg. 47, Nr. 3, S. 7585–7591, 2014, 19th IFAC World Congress. DOI: 10.3182/20140824-6-ZA-1003.01173.

[A37] M. Buchmayr, W. Kurschl und J. Küng, „A Simulator for Generating and Visualizing Sensor Data for Ambient Intelligence Environments," *Procedia Computer Science*, Jg. 5, S. 90–97, 2011. DOI: https://doi.org/10.1016/j.procs.2011.07.014.

[A38] Aceinna. „Open-source GNSS + inertial navigation, sensor fusion simulator." (2021), Adresse: https://github.com/Aceinna/gnss-ins-sim (besucht am 02. 07. 2021).

[A39] G. Ligorio und A. M. Sabatini, „A Simulation Environment for Benchmarking Sensor Fusion-Based Pose Estimators," *Sensors*, Jg. 15, Nr. 12, S. 32 031–32 044, 2015. DOI: 10.3390/s151229903.

[A40] A. D. Young, M. J. Ling und D. K. Arvind, „IMUSim: A simulation environment for inertial sensing algorithm design and evaluation," in *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2011, S. 199–210.

[A41] N. Dawar, S. Ostadabbas und N. Kehtarnavaz, „Data Augmentation in Deep Learning-Based Fusion of Depth and Inertial Sensing for Action Recognition," *IEEE Sensors Letters*, Jg. 3, Nr. 1, S. 1–4, 2019. DOI: 10.1109/LSENS.2018.2878572.

[A42] L. Tran und D. Choi, „Data Augmentation for Inertial Sensor-Based Gait Deep Neural Network," *IEEE Access*, Jg. 8, S. 12 364–12 378, 2020. DOI: 10.1109/ACCESS.2020.2966142.

[A43] O. Steven Eyobu und D. Han, „Feature Representation and Data Augmentation for Human Activity Classification Based on Wearable IMU Sensor Data Using a Deep LSTM Neural Network," *Sensors*, Jg. 18, Aug. 2018. DOI: 10.3390/s18092892.

[A44] Interaction Design Foundation. „Human-Computer Interaction (HCI)." (2021), Adresse: https://www.interaction-design.org/literature/topics/human-computer-interaction (besucht am 19. 07. 2021).

[A45] T. T. Hewett, R. Baecker, S. Card u. a., „ACM SIGCHI Curricula for Human-Computer Interaction," New York, NY, USA, Techn. Ber., 1992.

[A46] L. Clark, P. Doyle, D. Garaialde u. a., „The State of Speech in HCI: Trends, Themes and Challenges," *Interacting with Computers*, Jg. 31, Nr. 4, S. 349–371, Sep. 2019. DOI: 10.1093/iwc/iwz016.

[A47] R. P. Sharma und G. K. Verma, „Human Computer Interaction using Hand Gesture," *Procedia Computer Science*, Jg. 54, S. 721–727, 2015, Eleventh International Conference on Communication Networks, ICCN 2015, August 21-23, 2015, Bangalore, India Eleventh International Conference on Data Mining and Warehousing, ICDMW 2015, August 21-23, 2015, Bangalore, India Eleventh International Conference on Image and Signal Processing, ICISP 2015, August 21-23, 2015, Bangalore, India, ISSN: 1877-0509. DOI: 10.1016/j.procs.2015.06.085.

[A48] R. Ibañez, Á. Soria, A. Teyseyre und M. Campo, „Easy gesture recognition for Kinect," *Advances in Engineering Software*, Jg. 76, S. 171–180, 2014, ISSN: 0965-9978. DOI: 10.1016/j.advengsoft.2014.07.005.

[A49] Ultraleap. „Leap Motion Controller." (2021), Adresse: https://www.ultraleap.com/product/leap-motion-controller/ (besucht am 19. 07. 2021).

[A50] Samsung. „S Pen Air Actions." (2021), Adresse: https://www.samsung.com/at/support/mobile-devices/enjoy-air-actions-with-enhanced-s-pen-of-galaxy-note20-ultra/ (besucht am 19. 07. 2021).

[A51] T. Schlömer, B. Poppinga, N. Henze und S. Boll, „Gesture Recognition with a Wii Controller," *First publ. in: Proceedings of the 2nd International Conference on Tangible and Embedded Interaction 2008, Bonn, Germany, February 18-20, 2008, pp. 11-14*, Jan. 2008. DOI: 10.1145/1347390.1347395.

[A52] M. Kim, J. Cho, S. Lee und Y. Jung, „IMU Sensor-Based Hand Gesture Recognition for Human-Machine Interfaces," *Sensors*, Jg. 19, Nr. 18, 2019, ISSN: 1424-8220. DOI: 10.3390/s19183827.

[A53] Tutorialspoint. „Embedded Systems - Overview." (2021), Adresse: https://www.tutorialspoint.com/embedded_systems/es_overview.htm (besucht am 28. 07. 2021).

[A54] Embedded-Software-Engineering. „Was ist ein Embedded System?" (2021), Adresse: https://www.embedded-software-engineering.de/was-ist-ein-embedded-system-a-665424/ (besucht am 28. 07. 2021).

[A55] M. Barr. „Is a Smartphone an Embedded System?" (2021), Adresse: https://embeddedgurus.com/barr-code/2011/01/is-an-iphone-an-embedded-system/ (besucht am 28. 07. 2021).

[A56] „ISO/IEC/IEEE International Standard - Floating-point arithmetic," *ISO/IEC 60559:2020(E) IEEE Std 754-2019*, S. 1–86, 2020. DOI: 10.1109/IEEESTD.2020.9091348.

[A57] IBM Support. „Single-precision floating point accuracy." (2018), Adresse: https://www.ibm.com/support/pages/single-precision-floating-point-accuracy (besucht am 29. 07. 2021).

[A58] H. So. „Introduction to Fixed Point Number Representation," Berkeley University, California. (2006), Adresse: https://inst.eecs.berkeley.edu/~cs61c/sp06/handout/fixedpt.html (besucht am 29. 07. 2021).

[A59] E. Oberstar, „Fixed-point representation & fractional math (This is the Old Previous Copy See updated Link in Abstract)," Jg. 1, Jan. 2007.

[A60] C. Corsi, „Smart Sensors," *Infrared Physics & Technology*, Jg. 49, Nr. 3, S. 192–197, 2007, ISSN: 1350-4495. DOI: 10.1016/j.infrared.2006.06.002.

[A61] IT-Wissen. „Intelligenter Sensor." (2021), Adresse: https://www.itwissen.info/smart-sensor-Intelligenter-Sensor.html (besucht am 28. 07. 2021).

[A62] D. Marsh, „Applied Geometry for Computer Graphics and CAD," in März 2006, Bd. 2, ISBN: 978-1-84628-109-9. DOI: 10.1007/b138823.

[A63] J. Bloomenthal und J. Rokne, „Homogeneous coordinates," *The Visual Computer*, Jg. 11, S. 15–26, Jan. 1994. DOI: 10.1007/BF01900696.

[A64] Riesenfeld, „Homogeneous Coordinates and Projective Planes in Computer Graphics," *IEEE Computer Graphics and Applications*, Jg. 1, S. 50–55, 1981.

[A65] H. Ivan, „The Use of Projective Geometry in Computer Graphics," in Juli 2005, Bd. 1, ISBN: 978-3-540-55075-4. DOI: 10.1007/3-540-55075-5.

[A66] E. Marchand, H. Uchiyama und F. Spindler, „Pose Estimation for Augmented Reality: A Hands-On Survey," *IEEE Transactions on Visualization and Computer Graphics*, Jg. 22, Nr. 12, S. 2633–2651, Dez. 2016. DOI: 10.1109/TVCG.2015.2513408.

[A67] M. Lust, *Quaternionen - mathematischer Hintergrund und ihre Interpretation als Rotationen*, Computergrafik-Seminar, Universität Koblenz-Landau, 2001. Adresse: https://userpages.uni-koblenz.de/~cg/veranst/ws0001/sem/Lust_quaternion.pdf.

[A68] E. B. Dam, M. Koch und M. Lillholm, „Quaternions, Interpolation and Animation," University of Copenhagen, Techn. Ber. DIKU-TR-98/5, 1998. Adresse: https://web.mit.edu/2.998/www/QuaternionReport1.pdf.

[A69] Y.-B. Jia, *Quaternions and Rotations**, Com S 477/577 Notes, Stanford University, 2013. Adresse: https://graphics.stanford.edu/courses/cs348a-17-winter/Papers/quaternion.pdf.

[A70] Online-Kompendium Messtechnik und Sensorik. „8. Sensoren für Drehzahl, Geschwindigkeit, Beschleunigung und Position im Raum." (2021), Adresse: https://messtechnik-und-sensorik.org/8-sensoren-fuer-drehzahl-geschwindigkeit-beschleunigung-und-position-im-raum/ (besucht am 19. 07. 2021).

[A71] K. Maenaka, „MEMS inertial sensors and their applications," in *2008 5th International Conference on Networked Sensing Systems*, 2008, S. 71–73. DOI: 10.1109/INSS.2008.4610859.

[A72] F. Mohd-Yasin, D. Nagel und C. Korman, „Topical Review: Noise in Mems," *Measurement Science & Technology - MEAS SCI TECHNOL*, Jg. 21, Jan. 2010. DOI: 10.1088/0957-0233/21/1/012001.

[A73] A. G. Quinchia, G. Falco, E. Falletti, F. Dovis und C. Ferrer, „A Comparison between Different Error Modeling of MEMS Applied to GPS/INS Integrated Systems," *Sensors*, Jg. 13, Nr. 8, S. 9549–9588, 2013. DOI: 10.3390/s130809549.

[A74] Analog Devices. „Accelerometer Specifications - Quick Definitions." (2021), Adresse: https://www.analog.com/en/products/landing-pages/001/accelerometer-specificatios-definitions.html (besucht am 17. 06. 2021).

[A75] J. Lenz und S. Edelstein, „Magnetic sensors and their applications," *IEEE Sensors Journal*, Jg. 6, Nr. 3, S. 631–649, 2006. DOI: 10.1109/JSEN.2006.874493.

[A76] M. Ghose und A. Pradhan, „A Hand Gesture Recognition using Feature Extraction," *International Journal of Current Engineering and Technology (IJCET)*, Jg. 2, S. 323–327, Jan. 2012. DOI: 10.1016/j.procs.2020.06.022.

[A77] D. Laidig, S. Trimpe und T. Seel, „Event-Based Sampling for Reducing Communication Load in Realtime Human Motion Analysis by Wireless Inertial Sensor Networks," *Current Directions in Biomedical Engineering (accepted)*, Jg. 2, Sep. 2016. DOI: 10.1515/cdbme-2016-0154.

[A78] D. LI, W. Gao, Z. Li, Y. Yang und G. Chen, „Noise suppression method of rotating accelerometer gravity gradiometer instrument based on oversampling," in *2018 DGON Inertial Sensors and Systems (ISS)*, 2018, S. 1–14. DOI: 10.1109/InertialSensors.2018.8577130.

[A79] G. de Alteriis, D. Accardo, C. Conte und R. Schiano Lo Moriello, „Performance Enhancement of Consumer-Grade MEMS Sensors through Geometrical Redundancy," *Sensors*, Jg. 21, Nr. 14, 2021, ISSN: 1424-8220. DOI: 10.3390/s21144851.

[A80] W. Geiger, J. Bartholomeyczik, U. Breng u. a., „MEMS IMU for AHRS applications," in *2008 IEEE/ION Position, Location and Navigation Symposium*, 2008, S. 225–231. DOI: 10.1109/PLANS.2008.4569973.

[A81] R. Mahony, T. Hamel und J.-M. Pflimlin, „Nonlinear Complementary Filters on the Special Orthogonal Group," *IEEE Transactions on Automatic Control*, Jg. 53, Nr. 5, S. 1203–1218, 2008. DOI: 10.1109/TAC.2008.923738.

[A82] S. O. H. Madgwick, A. J. L. Harrison und R. Vaidyanathan, „Estimation of IMU and MARG orientation using a gradient descent algorithm," in *2011 IEEE International Conference on Rehabilitation Robotics*, 2011, S. 1–7. DOI: 10.1109/ICORR.2011.5975346.

[A83] G. Baldwin, R. Mahony, J. Trumpf, T. Hamel und T. Cheviron, „Complementary filter design on the Special Euclidean group SE(3)," in *2007 European Control Conference (ECC)*, 2007, S. 3763–3770. DOI: 10.23919/ECC.2007.7068746.

[A84] R. G. Valenti, I. Dryanovski und J. Xiao, „A Linear Kalman Filter for MARG Orientation Estimation Using the Algebraic Quaternion Algorithm," *IEEE Transactions on Instrumentation and Measurement*, Jg. 65, Nr. 2, S. 467–481, 2016. DOI: 10.1109/TIM.2015.2498998.

[A85] Q. Li, R. Li, K. Ji und W. Dai, „Kalman Filter and Its Application," in *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, 2015, S. 74–77. DOI: 10.1109/ICINIS.2015.35.

[A86] R. G. Valenti, I. Dryanovski und J. Xiao, „Keeping a Good Attitude: A Quaternion-Based Orientation Filter for IMUs and MARGs," *Sensors*, Jg. 15, Nr. 8, S. 19 302–19 330, 2015, ISSN: 1424-8220. DOI: 10.3390/s150819302.

[A87] S. O. Madgwick. „An efficient orientation filter for inertial and inertial/magnetic sensor arrays." (2010), Adresse: https://www.x-io.co.uk/res/doc/madgwick_internal_report.pdf (besucht am 29. 07. 2021).

[A88] E. Wan, „Sigma-Point Filters: An Overview with Applications to Integrated Navigation and Vision Assisted Control," in *2006 IEEE Nonlinear Statistical Signal Processing Workshop*, 2006, S. 201–202. DOI: 10.1109/NSSPW.2006.4378854.

[A89] „Eigen." (2021), Adresse: http://eigen.tuxfamily.org/index.php?title=Main_Page (besucht am 03. 08. 2021).

[A90] „GLUT and OpenGL Utility Libraries." (2021), Adresse: https://www.opengl.org/resources/libraries/ (besucht am 03. 08. 2021).

[A91] „The Mesa 3D Graphics Library." (2021), Adresse: https://www.mesa3d.org (besucht am 03. 08. 2021).

[A92] C. Konvalin. „Compensating for Tilt, Hard-Iron, and Soft-Iron Effects." (2009), Adresse: https://www.fierceelectronics.com/components/compensating-for-tilt-hard-iron-and-soft-iron-effects (besucht am 22. 11. 2021).

[A93] Bosch Sensortec. „BMA280 Data sheet." (2021), Adresse: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bma280-ds000.pdf (besucht am 03. 08. 2021).

[A94] A. Gelman, „Analysis of variance," *New Palgrave dictionary of economics*, 2006. DOI: 10.1057/9780230280816_37.

[A95] Quality-One. „Gage Repeatability & Reproducibility (Gage R&R)." (2021), Adresse: https://quality-one.com/grr/ (besucht am 27. 07. 2021).

[A96] Quality Services & Wissen GmbH. „Messsystemanalyse." (2021), Adresse: https://www.quality.de/lexikon/messsystemanalyse/ (besucht am 27. 07. 2021).

[A97] Engineering Statistics Handbook. „What are variance components?" (2021), Adresse: https://www.itl.nist.gov/div898/handbook/prc/section4/prc44.htm (besucht am 27. 07. 2021).

[A98] Yale University, Department of Statistics and Data Science. „Confidence Intervals." (2021), Adresse: http://www.stat.yale.edu/Courses/1997-98/101/confint.htm (besucht am 20. 07. 2021).

[A99] Investopedia. „Confidence Interval." (2021), Adresse: https://www.investopedia.com/terms/c/confidenceinterval.asp (besucht am 21. 07. 2021).

[A100] M. Müller. „Angewandte statistische Regression." (2021), Adresse: https://stat.ethz.ch/education/semesters/WS_2006_07/asr/Stat.pdf (besucht am 21. 07. 2021).

[A101] OpenCV. „Perspective-n-Point (PnP) pose computation." (2023), Adresse: https://docs.opencv.org/4.x/d5/d1f/calib3d_solvePnP.html (besucht am 21. 05. 2023).

[A102] B. M. Haralick, C.-N. Lee, K. Ottenberg und M. Nölle, „Review and analysis of solutions of the three point perspective pose estimation problem," *International Journal of Computer Vision*, Jg. 13, S. 331–356, Dez. 1994. DOI: 10.1007/BF02028352.

[A103] L. Ferraz, X. Binefa und F. Moreno-Noguer, „Very Fast Solution to the PnP Problem with Algebraic Outlier Rejection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Juni 2014.

[A104] EuclideanSpace. „Maths - Quaternion Arithmetic." (2022), Adresse: https://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/index.htm (besucht am 11. 02. 2022).

[A105] TechRepublic. „How to use Android sensors to detect orientation changes." (2021), Adresse: https://www.techrepublic.com/article/pro-tip-use-android-sensors-to-detect-orientation-changes/ (besucht am 21. 07. 2021).

[A106] T. Kuhlmann, P. Garaizar und U.-D. Reips, „Smartphone sensor accuracy varies from device to device in mobile research: The case of spatial orientation," *Behavior Research Methods*, Jg. 53, Mai 2020. DOI: 10.3758/s13428-020-01404-5.

[A107] D. James, „The Application of Inertial Sensors in Elite Sports Monitoring," in Jan. 2006, Bd. 3, S. 289–294, ISBN: 978-0-387-34680-9. DOI: 10.1007/978-0-387-45951-6_52.

[A108] Google Inc. „Android 6.0 Compatibility Definition." (2015), Adresse: https://source.android.com/compatibility/6.0/android-6.0-cdd.pdf (besucht am 26. 10. 2021).

[A109] Google Inc. „Gerätekatalog." (2021), Adresse: https://play.google.com/console/about/devicecatalog/ (besucht am 29. 10. 2021).

[A110] NOAA National Centers for Environmental Information. „World Magnetic Model." (2021), Adresse: https://www.ngdc.noaa.gov/geomag/WMM/ (besucht am 11. 10. 2021).

[A111] OpenCV. „Template Matching." (2014), Adresse: https://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html (besucht am 23. 11. 2021).

[A112] Lexico. „Meaning of parallax in English." (2014), Adresse: https://www.lexico.com/definition/parallax (besucht am 23. 11. 2021).

[A113] Samsung. „Alles rund um den S Pen." (2009), Adresse: https://www.samsung.com/de/support/mobile-devices/alles-rund-um-den-s-pen/ (besucht am 09. 12. 2021).

[A114] I. S. MacKenzie und S. Jusoh, „An evaluation of two input devices for remote pointing," in *IFIP International Conference on Engineering for Human-Computer Interaction*, Springer, 2001, S. 235–250. DOI: 10.1007/3-540-45348-2_21.

[A115] M. Oehl, C. Sutter und M. Ziefle, „Considerations on efficient touch interfaces–how display size influences the performance in an applied pointing task," in *Symposium on Human Interface and the Management of Information*, Springer, 2007, S. 136–143. DOI: 10.1007/978-3-540-73345-4_17.

[A116] N. Charness, P. Holley, J. Feddon und T. Jastrzembski, „Light pen use and practice minimize age and hand performance differences in pointing tasks," *Human Factors*, Jg. 46, Nr. 3, S. 373–384, 2004. DOI: 10.1518/hfes.46.3.373.50396.

[A117] P. Atkinson, „The best laid schemes o' mice and men : the evolution of the computer mouse," in *Design and Evolution : Proceedings of Design History Society Conference*, 2006.

[A118] L. Greenemeier. „The Origin of the Computer Mouse." (2009), Adresse: https://www.scientificamerican.com/article/origins-computer-mouse (besucht am 28. 07. 2021).

[A119] Microchip Technology. „ATSAMD20G18 - Arm Cortex-M3 MCU." (2021), Adresse: https://www.microchip.com/wwwproducts/en/ATSAMD20G18 (besucht am 03. 08. 2021).

[A120] Bosch Sensortec. „BMI055 Data sheet." (2021), Adresse: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmi055-ds000.pdf (besucht am 03. 08. 2021).

[A121] Bosch Sensortec. „BMM150 Data sheet." (2021), Adresse: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmm150-ds001.pdf (besucht am 03. 08. 2021).

[A122] Z. Zhang, Z. Zhou, S. Du, C. Xiang und C. Kuang, „Unscented Kalman Filter Based Attitude Estimation with MARG Sensors," in Mai 2019, S. 490–502, ISBN: 978-981-13-7758-7. DOI: 10.1007/978-981-13-7759-4_43.

[A123] Z. Dai und L. Jing, „Lightweight Extended Kalman Filter for MARG Sensors Attitude Estimation," *IEEE Sensors Journal*, Jg. 21, Nr. 13, S. 14 749–14 758, 2021. DOI: 10.1109/JSEN.2021.3072887.

[A124] J. Brownlee. „How Much Training Data is Required for Machine Learning?" (2017), Adresse: https://machinelearningmastery.com/much-training-data-required-machine-learning/ (besucht am 01. 07. 2021).

[A125] *Sicherheit von Maschinen, Sicherheitsbezogene Teile von Steuerungen*, Ser. Deutsche Normen DIN EN ISO 13849. Berlin: Beuth, 2013, Bd. Teil 2, 98 S., Ersatz für DIN EN ISO 13849-2:2008-09 und DIN EN ISO 13849-2 Berichtigung 1:2009-01.

[A126] *DIN EN IEC 61508/IEC Funktionale Sicherheit in der Prozess-Instrumentierung zur Risikoreduzierung*, Internationale Elektrotechnische Kommission, 2011.

[A127] Mathworks. „Sensor Fusion and Tracking Toolbox.“ (2021), Adresse: `https://de.mathworks.com/products/sensor-fusion-and-tracking.html` (besucht am 02. 07. 2021).

[A128] P. Bourke. „Generating noise with different power spectra laws.“ (1998), Adresse: `http://paulbourke.net/fractals/noise/` (besucht am 16. 06. 2021).

[A129] J. O. Smith III. „Example: Synthesis of 1/F Noise (Pink Noise).“ (2020), Adresse: `https://ccrma.stanford.edu/~jos/sasp/Example_Synthesis_1_F_Noise.html` (besucht am 16. 06. 2021).

[A130] J. O. Smith III. „Example: Pink Noise Analysis.“ (2020), Adresse: `https://ccrma.stanford.edu/~jos/sasp/Example_Pink_Noise_Analysis.html` (besucht am 16. 06. 2021).

[A131] D. Kozen und W.-L. D. Tseng, „The Böhm–Jacopini Theorem Is False, Propositionally,“ in *Mathematics of Program Construction*, P. Audebaud und C. Paulin-Mohring, Hrsg., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, S. 177–192, ISBN: 978-3-540-70594-9.

[A132] W. J. Rapaport. „The Boehm-Jacopini Theorem and Structured Programming.“ (2004), Adresse: `https://cse.buffalo.edu/~rapaport/111F04/greatidea3.html` (besucht am 18. 11. 2021).

[A133] K. Neubauer, M. Rumez, H. Tremmel u. a., „Virtual Verification of E/E Architectures for Secure Automated Driving Functions,“ in *2021 IEEE International Symposium on Systems Engineering (ISSE)*, 2021, S. 1–8. DOI: `10.1109/ISSE51541.2021.9582552`.

[A134] Google Inc. „Rotation vector CV crosscheck.“ (2023), Adresse: `https://source.android.com/docs/compatibility/cts/rotation-vector` (besucht am 02. 05. 2023).

[A135] REAL STATISTICS. „Two Factor ANOVA with Replication.“ (2021), Adresse: `https://www.real-statistics.com/two-way-anova/two-factor-anova-with-replication/` (besucht am 26. 07. 2021).

[A136] S. S. SHAPIRO und M. B. WILK, „An analysis of variance test for normality (complete samples)†,“ *Biometrika*, Jg. 52, Nr. 3-4, S. 591–611, Dez. 1965, ISSN: 0006-3444. DOI: `10.1093/biomet/52.3-4.591`.

[A137] StatSoft. „Levene und Brown-Forsythe-Tests für Varianzhomogenität.“ (2021), Adresse: `https://www.statsoft.de/glossary/L/LeveneandBrownForsythetestsforhomogeneityofvariancesHOV.htm` (besucht am 26. 07. 2021).

B Liste der Veröffentlichungen und Fachvorträge auf Tagungen

[B1] N. Büscher, L. Middendorf, C. Haubelt, R. Dorsch und F. Wegelin, „Statistical analysis and improvement of the repeatability and reproducibility of an evaluation method for IMUs on a smartphone," in *Proceedings of the Symposium on Engineering Interactive Computing Systems (EICS'16)*, Brüssel, Belgien: ACM, Juni 2016, S. 149–158, ISBN: 978-1-4503-4322-0. DOI: 10.1145/2933242.2933255.

[B2] N. Büscher, M. Stieringer und C. Haubelt, „Evaluation Method for the absolute Orientation from the Rotation Vector on Mobile Devices," in *Proceedings of the IEEE International Symposium on Inertial Sensors and Systems (INERTIAL)*, Naples, Florida, USA: IEEE, Apr. 2019, S. 179–183. DOI: 10.1109/ISISS.2019.8739713.

[B3] N. Büscher, D. Gis, S. Stieber und C. Haubelt, „Multi-Aspect Evaluation Method for Digital Pointing Devices," in *Proceedings of Pervasive and Embedded Computing and Communication Systems*, Bd. 8, Wien, Österreich: PECCS, Sep. 2019, S. 128–135. DOI: 10.5220/0008355701280135.

[B4] R. Dorsch, C. Haubelt, S. Stieber, D. Gis und N. Büscher, „Verfahren zum Verarbeiten der Sensorsignale einer Drehratensensoranordnung, Vorrichtung und Verfahren zum Betreiben der Vorrichtung," DE102020205943B3, Mai 2021.

[B5] D. Gis, N. Büscher und C. Haubelt, „Investigation of Timing Behavior and Jitter in a Smart Inertial Sensor Debugging Architecture," *MDPI Sensors*, Nr. 14, 2021. DOI: 10.3390/s21144675.

[B6] D. Gis, N. Büscher und C. Haubelt, „Advanced Debugging Architecture for Smart Inertial Sensors using Sensor-in-the-Loop," in *International Workshop of Rapid System Prototyping (RSP)*, IEEE, Sep. 2020, S. 1–7. DOI: 10.1109/RSP51120.2020.9244851.

[B7] D. Gis, N. Büscher und C. Haubelt, „Real-Time Power Analysis of Smart Sensors Using Advanced Debugging Methods," *MDPI Micromachines*, Nr. 12, 2021. DOI: 10.3390/mi12111276.

[B8] N. Büscher, D. Gis, V. Kühn und C. Haubelt, „On the Functional and Extra-Functional Properties of IMU Fusion Algorithms for Body-Worn Smart Sensors," *MDPI Sensors*, Nr. 8, 2021. DOI: 10.3390/s21082747.

[B9] N. Büscher, D. Gis, J. P. Wolff und C. Haubelt, „Data Augmentation Framework for Smart Sensor System Development Using the Sensor-in-the-Loop Prototyping Platform," in *International Workshop of Rapid System Prototyping (RSP)*, Paris, France: IEEE, 2021. DOI: 10.1109/RSP53691.2021.9806209.

[B10] D. Gis, C. Haubelt und N. Büscher, „Verfahren zum Verarbeiten der Sensorsignale einer Drehratensensoranordnung, Vorrichtung und Verfahren zum Betreiben der Vorrichtung," DE102021200244A1, März 2022.

[B11] D. Gis, C. Haubelt und N. Büscher, „Method for Checking, Evaluation, and/or Error Diagnosis of a Sensor System, Sensor System, and System," US000011589-139B2, Feb. 2023.

[B12] D. Gis, C. Haubelt und N. Büscher, „Method for Testing, Evaluation, and/or Error Diagnosis of a Sensor System, Sensor System, and System," CN0001142538-31A, 2022.

C Liste der betreuten studentischen Arbeiten

[C1] M. Stieringer, „Konzeption und Implementierung eines Testablaufes für die Bewertung von Rotationssensoren mit Magnetometer," *Masterarbeit*, 2018.

[C2] M. N. Abbasi, „Enhancing the robustness of a Test for the Rotation-Vector via Image Analysis," *Masterarbeit*, 2019.

[C3] A. John, „Extraction of the Characteristics of Typical Measurement Errors from Inertial Sensors and Generation of their Artificial Counterpart," *Masterarbeit*, 2017.

[C4] D. Adrians, „Effiziente Konsolidierung von Tiefenkarten," *Masterarbeit*, 2017.

[C5] J. Lossner, „Design and Implementation of a Calibration Model for Multi-aperture Sensors," *Masterarbeit*, 2018.

[C6] A. Neskovic, „Untersuchung der Eigenschaften verschiedener Fusionsalgorithmen in einem eingebetteten System," *Bachelorarbeit*, 2019.

[C7] J. Marx, „Analyse verschiedener Techniken zur Extraktion der Erdbeschleunigung aus Inertialsensordaten," *Bachelorarbeit*, 2019.

[C8] S. Sailik, „Analysis of various Techniques for the Extraction of the Gravity Component from Measurements of an Accelerometer," *Masterarbeit*, 2020.

[C9] B. Stüpmann, „Implementierung eines Software-Dienstes zur Parametrisierung und Kontrolle einer Steuereinheit zur Generierung von Triggersignalen," *Bachelorarbeit*, 2020.

[C10] O. Mobayed, „Analysis and evaluation of the precision of an image based algorithm to detect position and orientation of a device," *Masterarbeit*, 2021.

[C11] M. Alattar, „Analysis of a superposition methodology for training and using gesture recognition algorithms," *Masterarbeit*, 2021.

[C12] M. Alahmad, „Implementation and evaluation of multiple calibration methods for Magnetometer data," *Masterarbeit*, 2022.

D Anhang

D.1 Berechnung der Varianzanalysen

Wie in Abschnitt 3.7 beschrieben, werden bei der Bewertung der Sensordaten oder der Ergebnisse der Sensordatenfusionen statistische Methoden verwendet. Wie diese verwendeten Methoden arbeiten, wird in diesem Anhang kurz vorgestellt.

Berechnung bei der einfaktoriellen Varianzanalyse

Im Folgenden wird erklärt, wie der Einfluss einer unabhängigen Variable mit der einfaktoriellen Varianzanalyse berechnet wird. Angenommen, man hat g Gruppen (unterteilt nach der unabhängigen Variable) mit jeweils n Messwerten für jede Gruppe. Zuerst muss der Erwartungswert $\overline{x}$ für die Gesamtheit der Messwerte sowie die Erwartungswerte $\overline{x}_j$ für jede Gruppe berechnet werden. Dies ist in Gleichung (D.1) gezeigt. Hierbei stehen $j \in \{1, \ldots, g\}$ jeweils für den Index der Gruppe, $x_{i,j}$ ist der i-te Wert in der Gruppe mit dem Index j.

$$\overline{x}_j = \frac{1}{n} \sum_{i=1}^{n} x_{i,j} \tag{D.1}$$

Anschließend wird für jede Gruppe die Residuenquadratsumme innerhalb der Gruppe gebildet, wie in Gleichung (D.2) gezeigt

$$S_j = \sum_{i=1}^{n} (x_{i,j} - \overline{x}_j)^2 \tag{D.2}$$

Weiterhin müssen die Residuenquadratsumme S_G aller Messwerte (Gleichung (D.3)), die Summe der Quadrate innerhalb der Gruppen S_I (Gleichung (D.4)), sowie die Summe der Quadrate S_Z (Gleichung (D.4)) zwischen den Gruppen gebildet werden. Hierbei ist $\overline{x}$ der Mittelwert aller $g \cdot n$ Messungen.

$$S_G = \sum_{i=1}^{n} \sum_{j=1}^{g} (x_{i,j} - \overline{x})^2 \tag{D.3}$$

$$S_I = \sum_{j=1}^{g} S_j \tag{D.4}$$

$$S_Z = \sum_{j=1}^{g} n\left(\overline{x}_j - \overline{x}\right)^2 \tag{D.5}$$

Für die Berechnung des F-Wertes werden die quadratischen Mittel von S_G, S_I und S_Z benötigt, die in den Gleichungen D.6, D.7 und D.8 berechnet werden. Für S_G ergeben sich $f_G = (n \cdot g) - 1$ Freiheitsgerade, für S_Z ergeben sich $f_Z = n - 1$ Freiheitsgerade und für S_I ergeben sich $f_I = (n \cdot g) - n$ Freiheitsgerade.

$$Q_G = S_G / d_G \tag{D.6}$$

$$Q_I = S_I / d_I \tag{D.7}$$

$$Q_Z = S_Z / d_Z \tag{D.8}$$

Der F-Wert kann nun mittels Gleichung (D.9) ermittelt werden.

$$F = \frac{Q_Z}{Q_I} \tag{D.9}$$

Anhand des F-Wertes und der Freiheitsgrade f_Z und f_I lässt sich mittels der F-Verteilung der P-Wert berechnen. Liegt der P-Wert über dem Signifikanzniveau α, gibt es keinen Einfluss der unabhängigen Variablen, die Nullhypothese kann also nicht verworfen werden. Zudem kann mittels des F-Tests geprüft werden, ob die unabhängige Variable einen Einfluss auf die Messwerte hat, wenn der berechnete F-Wert unter dem kritischen Wert F_{krit} liegt.

Berechnung bei der zweifaktoriellen Varianzanalyse

Die Berechnung der Einflüsse bei der zweifaktoriellen Varianzanalyse ist äquivalent zu der Berechnung bei der einfaktoriellen Varianzanalyse. Weiterhin werden S_G für die Varianz aller Messwerte und S_I für die Varianz innerhalb der Gruppen benötigt. Da es mehrere Gruppen gibt, wird S_I jeweils gruppiert nach dem ersten Faktor (A) und den zweiten Faktor (B) berechnet [A135]. Weiter kommt der Faktor für die Interaktion zwischen den Faktoren A und B hinzu, dessen Berechnung in der Gleichung (D.10) gezeigt ist.

$$S_{A\times B} = m \sum_{i=1}^{a} \sum_{j=1}^{b} \left(\overline{x}_{i,j} - \overline{x}_{Bi} - \overline{x}_{Aj} - \overline{x}\right)^2 \tag{D.10}$$

Hierbei ist a die Anzahl der Gruppen von Faktor A und b die Anzahl der Gruppen in Faktor B, $\overline{x}_{i,j}$ ist der Erwartungswert alle Werte der i-ten Gruppe von Faktor A und der j-ten Gruppe von Faktor B. $\overline{x}_{Bi}$ ist der Erwartungswert für die i-ten Gruppe von Faktor A für alle Gruppen von Faktor B, $\overline{x}_{Aj}$ ist der Erwartungswert für die j-ten Gruppe von Faktor B für alle Gruppen von Faktor A.

Um zuverlässige Ergebnisse für die zweifaktorielle Varianzanalyse zu erhalten, müssen die Messdaten bestimmte statistische Voraussetzungen erfüllen. Die erste Voraussetzung ist die Normalverteilung der Grundgesamtheit der Daten, aus denen die Messwerte genommen werden. Die Normalverteilung kann mittels eines Shapiro-Wilk-Tests [A136] überprüft werden. Zweitens muss die Varianz innerhalb der Gruppen homogen sein (Varianzhomogenität). Dies kann mittels eines Levene-Tests [A137] ermittelt werden.

D.2 Berechnungen der Messsystemanalyse

Bei der Messsystemanalyse wird zusätzlich zu der zweifaktoriellen Varianzanalyse eine Varianzkomponentenschätzung durchgeführt. Diese schätzt, wie groß der Einfluss der einzelnen Faktoren auf die Varianz innerhalb der Messdaten ist. Der Anteil der Messvarianz V_U, der von keinem der beiden Faktoren stammt, wird abhängig von der Signifikanz der Interaktion der Faktoren berechnet. Die Formel hierfür ist in Gleichung Gleichung (D.11) gegeben.

$$V_U = \begin{cases} \dfrac{S_A + S_B - S_G}{f_I + f_A + f_B} & \text{wenn } P_{A\times B} > 0.05 \\ Q_{A\times B} & \text{wenn } P_{A\times B} \leq 0.05 \end{cases} \tag{D.11}$$

Die Berechnungen der geschätzten Varianzkomponenten sind in den Gleichungen D.12, D.13 und Gleichung (D.14) beschrieben.

$$V_A = \frac{Q_A - V_U}{bn} \tag{D.12}$$

$$V_B = \frac{Q_B - V_U}{an} \tag{D.13}$$

$$V_{A\times B} = \frac{Q_{A\times B} - Q_I}{n} \tag{D.14}$$

Der Einfluss einer einzelnen Komponente sowie von Wiederholbarkeit und Reproduzierbarkeit ergibt sich aus dem Verhältnis der einzelnen Komponente zur Summe aller Varianzkomponenten V_{Ges}. Die Berechnungen der Einflussfaktoren für Wiederholbarkeit und Reproduzierbarkeit sind in Gleichung (D.15) und Gleichung (D.16) gegeben.

$$Wiederholbarkeit = \frac{V_U}{V_{Ges}} \tag{D.15}$$

$$Reproduzierbarkeit = \frac{V_B + V_{A\times B}}{V_{Ges}} \tag{D.16}$$

D.3 Berechnungen des Konfidenzintervalls

Mithilfe des Konfidenzintervalls kann aus den vorhandenen Stichproben ein Intervall bestimmt werden, in welchem der unbekannte wahre Wert mit einer bestimmten Wahrscheinlichkeit (oft 95 % oder 99 %) liegt [A98]. Bei dem Konfidenzintervall werden die obere und die unter Grenze wie in den Gleichungen D.17 und D.18 ermittelt.

$$k_{oben} = \overline{x} + t^* \frac{\sigma}{\sqrt{n}} \tag{D.17}$$

$$k_{unten} = \overline{x} - t^* \frac{\sigma}{\sqrt{n}} \tag{D.18}$$

Hierbei ist $\overline{x}$ der Mittelwert der Stichproben, σ die Standardabweichung der Stichproben und n die Anzahl der Stichproben. Der Wert für t^* wird aus einer Tabelle abgelesen und hängt von der Anzahl der Stichproben sowie dem erwünschten Konfidenzniveau ab. Je nach Anzahl der Stichproben wird entweder eine studentsche t-Verteilung oder eine Standardnormalverteilung angenommen und die entsprechende Tabelle verwendet.

D.4 Berechnungen der linearen Regression

Bei der linearen Regression wird versucht, einen linearen Zusammenhang zwischen zwei unabhängigen Variablen zu ermitteln. Hierbei werden die Parameter a und b der Geradengleichung $y = ax + b$ für die beiden unabhängigen Variablen x und y ermittelt [A100],

$$a = \frac{n \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \tag{D.19}$$

$$b = \frac{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} x_i y_i \sum_{i=1}^{n} x_i}{n \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \tag{D.20}$$

Die Werte x_i und y_i sind die Messwerte für x und y an der i-ten Stelle. Die Anzahl der Messpunkte ist mit n angegeben. Neben der Werte für a und b ist natürlich auch eine Information notwendig, die angibt, wie stark die Korrelation zwischen den unabhängigen Variablen ist. Hierfür wird der Korrelationskoeffizient r verwendet, der sich wie in Gleichung (D.21) beschrieben ermitteln lässt.

$$r = \frac{n \sum_{i=1}^{n} x_i y_i - \left(\sum_{i=1}^{n} x_i\right)\left(\sum_{i=1}^{n} y_i\right)}{\sqrt{n \left(\sum_{i=1}^{n} x_i^2\right) \left(\sum_{i=1}^{n} x_i\right)^2} \sqrt{n \left(\sum_{i=1}^{n} y_i^2\right) \left(\sum_{i=1}^{n} y_i\right)^2}} \tag{D.21}$$

Der Korrelationskoeffizient liegt in einem Intervall von $-1 \leq r \leq 1$. Wenn der r-Wert gegen 0 geht, besteht kein linearer Zusammenhang zwischen Varianz und Steigung der Geraden. Bei einem positiven r-Wert gibt es einen positiven Zusammenhang, bei einem negativen r-Wert einen negativen Zusammenhang. Je näher der r-Wert gegen -1 oder 1 tendiert, desto unabhängiger sind die Variablen x und y voneinander.

D.5 Messungen für die MEMS-Score-Anwendung

Gage R&R Analyse der MEMS-Score-Anwendung

Tabelle D.1: Daten für die Gage R&R-Analyse ohne Nutzerführung

Testperson	Wiederholung	Gerät 1	Gerät 2	Gerät 3	Gerät 4	Gerät 5
A	1	3,8	5,04	1,65	11,01	1,03
A	2	4,49	2,98	1,56	13,6	2,13
A	3	3,62	5,12	1,64	10,2	2,00
B	1	0,90	1,20	1,24	5,67	9,55
B	2	2,38	1,31	1,17	9,94	2,70
B	3	1,68	1,14	1,07	6,04	2,38
C	1	1,29	1,21	0,88	12,84	1,43
C	2	1,43	2,21	1,44	14,67	1,36
C	3	1,79	2,83	1,40	7,80	1,45
D	1	2,97	2,87	0,95	12,45	2,67
D	2	4,31	5,21	1,03	9,14	2,14
D	3	4,17	3,27	1,15	3,44	2,95

Tabelle D.2: Daten für die Gage R&R-Analyse mit Nutzerführung

Testperson	Wiederholung	Gerät 1	Gerät 2	Gerät 3	Gerät 4	Gerät 5
A	1	1,83	2,41	0,76	7,50	1,55
A	2	2,65	3,38	0,73	8,29	1,58
A	3	1,21	2,24	0,81	6,38	2,63
B	1	1,5	2,07	0,59	11,55	3,55
B	2	1,09	4,39	0,76	4,26	1,18
B	3	1,44	2,25	0,70	6,69	1,40
C	1	3,00	2,94	0,73	7,59	2,48
C	2	2,27	4,42	0,74	10,67	2,61
C	3	1,55	3,49	0,80	4,84	1,92
D	1	2,79	2,95	0,88	5,47	1,84
D	2	3,46	3,28	0,65	9,86	2,44
D	3	2,64	3,62	0,81	7,14	1,58

Gesammelte Ergebnisse der MEMS-Score-Anwendung

Tabelle D.3: Daten über die *High-Fidelity*-Sensoren in Android-Smartphones

Test	Frequenz	Hi-Fi	Bewertung	# Messungen	Geräte
Lineare Beschl.	50Hz	Nein	0,72	143	47
Lineare Beschl.	50Hz	Ja	1,32	67	11
Lineare Beschl.	200Hz	Nein	0,52	118	35
Lineare Beschl.	200Hz	Ja	1,43	57	12
Statische Rotation	50Hz	Nein	2,45	103	25
Statische Rotation	50Hz	Ja	1,04	63	10
Statische Rotation	200Hz	Nein	2,24	82	17
Statische Rotation	200Hz	Ja	1,08	55	12
Dyn. Rotation	50Hz	Nein	4,20	39	9
Dyn. Rotation	50Hz	Ja	3,29	47	8
Dyn. Rotation	200Hz	Nein	3,95	32	7
Dyn. Rotation	200Hz	Ja	3,85	43	9

Tabelle D.4: Bewertung der Sensorsubsysteme gruppiert nach Sensorhersteller mit 50 Hz-Datenrate

Hersteller	Bewertung	Anz.	Bewertung	Anz.	Bewertung	Anz.
	Lineare Beschl.		*Statische Rotation*		*Dynamische Rotation*	
Bosch	1,39	84	1,38	71	4,51	47
InvenSense	0,93	50	2,62	43	2,71	19
STM	0,77	58	3,70	42	3,35	21
HTC	2,22	8	0,50	8	0,59	5
MTK	0,19	8	-	-	-	-

Tabelle D.5: Bewertung der Sensorsubsysteme gruppiert nach Sensorhersteller mit 200 Hz-Datenrate

Hersteller	Bewertung	Anz.	Bewertung	Anz.	Bewertung	Anz.
	Lineare Beschl.		*Statische Rotation*		*Dynamische Rotation*	
Bosch	1,39	75	0,65	59	4,51	40
InvenSense	0,80	45	2,52	40	3,94	16
STM	1,21	41	3,48	29	3,31	13
HTC	3,10	6	0,27	6	1,25	5
MTK	0,25	5	-	-	-	-

Tabelle D.6: Bewertung der Sensorsubsysteme gruppiert nach Sensorfusion mit 50 Hz Datenrate

Hersteller	Bewertung	Anz.	Bewertung	Anz.	Bewertung	Anz.
	Lineare Beschl.		*Statische Rotation*		*Dynamische Rotation*	
Samsung	0,62	38	-	-	-	-
AOSP	0,25	25	-	-	-	-
Google	0,39	40	4,59	40	4,29	37
QTI	2,05	74	4,40	57	4,10	24
HTC	2,22	8	4,56	8	0,59	5
Motorola	0,39	7	-	-	-	-
Invensense	-	-	0,29	46	2,82	24

Tabelle D.7: Bewertung der Sensorsubsysteme gruppiert nach Sensorfusion mit 200 Hz Datenrate

Hersteller	Bewertung	Anz.	Bewertung	Anz.	Bewertung	Anz.
	Lineare Beschl.		*Statische Rotation*		*Dynamische Rotation*	
Samsung	0,44	33	-	-	-	-
AOSP	0,25	18	-	-	-	-
Google	0,36	38	4,54	38	3,94	30
QTI	1,94	69	4,59	48	4,36	22
HTC	3,10	6	4,83	6	1,25	5
Motorola	0,37	5	-	-	-	-
Invensense	-	-	0,40	40	3,62	17

Tabelle D.8: Bewertung der Sensorsubsysteme gruppiert nach Smartphone-Hersteller mit 50 Hz Datenrate

Hersteller	Bewertung	Anz.	Bewertung	Anz.	Bewertung	Anz.
	Lineare Beschl.		*Statische Rotation*		*Dynamische Rotation*	
LGE	1,44	41	4,47	41	4,41	37
Sony	1,86	34	4,69	34	3,15	10
Google	0,51	17	4,86	17	4,38	12
Samsung	0,59	54	0,63	42	3,28	22
Motorola	0,44	16	-	-	-	-
Huawei	0,52	16	2,64	11	4,42	5
HTC	2,09	9	4,05	9	0,59	5
JYT	0,21	5	-	-	-	-
Xiaomi	3,11	5	-	-	-	-

Tabelle D.9: Bewertung der Sensorsubsysteme gruppiert nach Smartphone-Hersteller mit 200 Hz Datenrate

Hersteller	**Bewertung**	**Anz.**	**Bewertung**	**Anz.**	**Bewertung**	**Anz.**
	Lineare Beschl.		*Statische Rotation*		*Dynamische Rotation*	
LGE	1,69	37	4,47	36	4,64	29
Sony	1,73	33	4,59	33	4,12	10
Google	0,44	17	4,98	17	3,83	11
Samsung	0,86	43	0,00	33	3,56	14
Motorola	1,15	11	-	-	-	-
Huawei	0,54	9	4,38	7	4,16	5
HTC	3,10	6	4,83	6	1,25	5
JYT	1,78	5	-	-	-	-
Xiaomi	0,25	5	-	-	-	-

D.6 Messungen für den Test des Magnetometers

Tabelle D.10: Ergebnisse der ersten Messreihe für den Magnetometer-Test

Messung links	Messung mitte	Messung rechts	Nutzer:Innen-fehler links	Nutzer:Innen-fehler rechts	Mittlere Abweichung
298,36 °	283,70 °	269,54 °	0,61 °	0,11 °	-6,95 °
299,48 °	286,67 °	272,43 °	-1,24 °	0,19 °	-4,63 °
302,81 °	289,41 °	275,76 °	-0,65 °	0,41 °	-1,49 °
302,52 °	289,26 °	273,69 °	-0,79 °	1,52 °	-2,33 °
306,18 °	290,91 °	276,31 °	1,22 °	0,55 °	0,31 °
301,97 °	289,28 °	272,57 °	-1,36 °	2,66 °	-2,88 °
299,67 °	289,30 °	273,28 °	-3,69 °	1,97 °	-3,40 °
302,27 °	288,08 °	273,01 °	0,14 °	1,01 °	-3,04 °
301,91 °	288,01 °	274,59 °	-0,16 °	-0,62 °	-2,65 °
301,05 °	289,90 °	277,10 °	-2,90 °	-1,25 °	-1,47 °

Tabelle D.11: Ergebnisse der zweiten Messreihe für den Magnetometer-Test

Messung links	Messung mitte	Messung rechts	Nutzer:Innen-fehler links	Nutzer:Innen-fehler rechts	Mittlere Abweichung
302,09 °	288,38 °	262,05 °	-3,30 °	9,30 °	-6,65 °
302,00 °	285,93 °	269,94 °	-0,95 °	-1,02 °	-4,86 °
306,91 °	289,36 °	272,60 °	0,54 °	-0,25 °	-1,20 °
303,80 °	285,27 °	268,02 °	1,52 °	0,23 °	-5,12 °
312,58 °	298,80 °	278,70 °	-3,22 °	3,01 °	5,87 °
306,48 °	290,39 °	272,65 °	-0,93 °	0,73 °	-0,98 °
301,67 °	288,77 °	264,88 °	-4,11 °	6,89 °	-5,72 °
302,60 °	287,27 °	267,11 °	-1,68 °	3,14 °	-5,16 °
306,05 °	294,36 °	270,51 °	-5,33 °	6,84 °	-0,51 °
305,55 °	291,15 °	269,60 °	-2,62 °	4,54 °	-2,06 °

D.7 Ergebnisse des Tests für digitale Zeigegeräte

Tabelle D.12: Ergebnisse des Tests für die Genauigkeit und die Präzision von digitalen Zeigegeräten

Gerät	**Form**	**Genauigkeit** ($score_{form}$)	**Stabilität** ($score_{winkel}$)	**Gesamt-bewertung**
Prototyp	Würfel	0.81	0.81	0,67
Prototyp	Dreieck	0,83	0,83	0,69
Prototyp	Kreis	0,84	0,88	0,74
Kommerziell	Würfel	0,76	0,81	0,62
Kommerziell	Dreieck	0,77	0,79	0,61
Kommerziell	Kreis	0,82	0,80	0,65
Smartphone	Würfel	0,90	0,94	0,85
Smartphone	Dreieck	0,77	0,69	0,53
Smartphone	Kreis	0,79	0,67	0,53
Computermaus	Würfel	0,88	0,88	0,78
Computermaus	Dreieck	0,86	0,90	0,78
Computermaus	Kreis	0,87	0,91	0,79

D.8 Messergebnisse der Untersuchung der Sensorfusionsalgorithmen

Codegröße ohne Compiler-Optimierung

Tabelle D.13: Größe der Binärdatei in Byte im Festwertspeicher ohne Verwendung des Magnetometers ohne Compiler-Optimierung

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	17580	17144	17252
Madgwick-Filter	11464	9132	10412
Mahony-Filter	10896	9560	10072
Komplementärfilter	10564	9604	9960

Tabelle D.14: Größe der Binärdatei in Byte im Festwertspeicher mit Verwendung des Magnetometers ohne Compiler-Optimierung

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	19600	19312	18828
Madgwick-Filter	14984	11476	13252
Mahony-Filter	12416	9864	10748
Komplementärfilter	12488	11744	11552

Benötigter Arbeitsspeicher mit -O3 Compiler-Optimierung

Tabelle D.15: Benötigter Arbeitsspeicher in Byte ohne Verwendung des Magnetometers mit Compiler-Optimierung (-O3)

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	3272	3432	3368
Madgwick-Filter	3176	3344	3328
Mahony-Filter	3192	3360	3336
Komplementärfilter	3176	3344	3320

Tabelle D.16: Benötigter Arbeitsspeicher in Byte mit Verwendung des Magnetometers mit Compiler-Optimierung (-O3)

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	3280	3440	3376
Madgwick-Filter	3192	3352	3336
Mahony-Filter	3208	3368	3344
Komplementärfilter	3192	3352	3336

Laufzeit ohne Compiler-Optimierung

Tabelle D.17: Durchschnittliche Ausführungszeit der Sensorfusionsalgorithmen für eine Aktualisierung ohne Magnetometer und ohne Compiler-Optimierung

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	2,698 ms	1,492 ms	0,305 ms
Madgwick-Filter	0,762 ms	0,283 ms	0,104 ms
Mahony-Filter	0,682 ms	0,207 ms	0,105 ms
Komplementärfilter	0,666 ms	0,226 ms	0,114 ms

Tabelle D.18: Durchschnittliche Ausführungszeit der Sensorfusionsalgorithmen für eine Aktualisierung mit Magnetometer und ohne Compiler-Optimierung

Filter	32 Bit-Float	32 Bit-Festkomma	16 Bit-Festkomma
Kalman-Filter	3,963 ms	1,923 ms	0,621 ms
Madgwick-Filter	1,142 ms	0,560 ms	0,166 ms
Mahony-Filter	0,758 ms	0,350 ms	0,121 ms
Komplementärfilter	0,782 ms	0,382 ms	0,123 ms

Messungen der Fusionsalgorithmen ohne Magnetometer

Tabelle D.19: Messergebnisse Filteralgorithmen ohne Magnetometer

Filter	Format	Geschw.	Messung									
Kompl.	F16	Langsam	1,074	1,281	1,235	1,572	1,158	2,160	2,659	1,721	1,502	1,709
Mahony	F16	Langsam	0,990	1,037	1,404	1,423	1,213	1,087	1,201	0,957	0,812	1,437
Madgwick	F16	Langsam	1,004	0,882	1,234	1,174	1,158	1,016	1,129	1,025	0,833	1,478
Kalman	F16	Langsam	1,013	0,864	1,179	1,115	1,153	0,983	1,003	1,249	0,765	1,413
Kompl.	F16	Mittel	1,543	1,484	2,166	1,489	1,643	3,329	2,142	2,670	2,666	2,354
Mahony	F16	Mittel	1,460	1,421	1,279	1,315	1,444	1,907	1,390	2,146	1,852	1,074
Madgwick	F16	Mittel	1,495	1,268	1,237	1,342	1,404	1,694	1,407	1,858	1,908	1,116
Kalman	F16	Mittel	1,410	1,332	1,166	1,412	1,569	1,817	1,262	1,699	1,906	1,104
Kompl.	F16	Schnell	2,228	3,431	3,047	2,360	2,296	2,959	3,293	2,541	4,672	5,474
Mahony	F16	Schnell	1,861	2,310	1,904	2,304	1,934	1,407	1,843	1,168	4,170	2,594
Madgwick	F16	Schnell	1,920	2,571	2,116	2,143	1,840	1,385	1,711	1,043	2,466	2,618
Kalman	F16	Schnell	1,893	2,446	2,203	2,142	1,759	1,508	1,671	1,051	1,911	2,670
Kompl.	F32	Langsam	1,005	0,933	1,153	1,119	1,134	1,027	0,926	1,034	0,783	1,735
Mahony	F32	Langsam	1,106	0,958	1,177	1,147	1,158	1,145	0,929	1,114	0,820	1,629
Madgwick	F32	Langsam	1,082	1,044	1,557	1,417	1,252	1,002	1,079	0,988	0,730	1,288
Kalman	F32	Langsam	0,990	0,872	1,198	1,177	1,133	0,996	0,980	1,130	0,748	1,371
Kompl.	F32	Mittel	1,474	1,338	1,207	1,459	1,584	1,877	1,292	1,631	1,831	1,148
Mahony	F32	Mittel	1,437	1,393	1,202	1,563	1,662	1,971	1,332	1,895	1,866	1,267
Madgwick	F32	Mittel	1,439	1,504	1,034	1,372	1,474	1,703	1,236	2,086	1,725	0,978
Kalman	F32	Mittel	1,385	1,299	1,126	1,374	1,521	1,772	1,242	1,764	1,868	1,090
Kompl.	F32	Schnell	1,948	2,521	2,303	2,180	1,780	1,593	1,721	1,135	3,269	2,715
Mahony	F32	Schnell	2,048	2,414	2,414	2,123	1,761	1,642	1,764	1,182	3,188	2,774
Madgwick	F32	Schnell	1,939	2,250	1,938	2,216	1,903	1,322	1,751	1,187	3,638	2,563
Kalman	F32	Schnell	1,867	2,409	2,108	2,100	1,741	1,443	1,652	1,119	2,232	2,629
Kompl.	Float	Langsam	1,030	0,870	1,178	1,127	1,165	1,012	0,946	1,014	0,783	1,691
Mahony	Float	Langsam	1,109	1,047	1,310	1,251	1,172	1,336	1,021	1,330	0,888	1,625
Madgwick	Float	Langsam	1,075	1,060	1,585	1,430	1,256	1,097	1,052	0,978	0,776	1,437
Kalman	Float	Langsam	1,013	0,864	1,179	1,115	1,153	0,983	1,003	1,249	0,765	1,413
Kompl.	Float	Mittel	1,470	1,372	1,191	1,432	1,593	1,866	1,284	1,658	1,838	1,136
Mahony	Float	Mittel	1,470	1,505	1,258	1,650	1,741	1,988	1,335	1,646	1,915	1,462
Madgwick	Float	Mittel	1,460	1,482	1,088	1,315	1,548	1,708	1,239	2,113	1,867	1,002
Kalman	Float	Mittel	1,410	1,332	1,166	1,412	1,569	1,817	1,262	1,699	1,906	1,104
Kompl.	Float	Schnell	1,942	2,497	2,267	2,188	1,786	1,578	1,706	1,116	3,244	2,708
Mahony	Float	Schnell	2,068	2,426	2,568	2,136	1,763	1,706	1,831	1,186	3,120	2,826
Madgwick	Float	Schnell	1,972	2,417	1,929	2,217	1,891	1,395	1,772	1,140	3,776	2,567
Kalman	Float	Schnell	1,893	2,446	2,203	2,142	1,759	1,508	1,671	1,051	1,911	2,670

Die Werte der Messung sind die mittleren quadratischen Fehler im Vergleich zur Referenz. F16 sind 16 Bit-Festkommazahlen, F32 die 32 Bit-Festkommazahlen und Float die 32 Bit-Fließkommazahlen. Die *langsame* Geschwindigkeit bezieht sich auf die Messungen mit einer durchschnittlichen Winkelgeschwindigkeit von 7,5 °/s, die mittlere Geschwindigkeit auf eine durchschnittliche Winkelgeschwindigkeit von 15 °/s und die schnelle Geschwindigkeit auf eine durchschnittliche Winkelgeschwindigkeit von 30 °/s.

Messungen der Fusionsalgorithmen mit Magnetometer

Tabelle D.20: Messergebnisse Filteralgorithmen mit Magnetometer

Algorithmus	Format	Geschw.	Messung									
Kompl.	F16	Langsam	1,5118	1,2727	1,4042	2,7488	1,7143	1,7561	1,5106	2,2960	1,9352	1,2326
Mahony	F16	Langsam	2,6758	2,2489	2,7843	2,9490	1,1167	1,9675	1,7345	2,3612	2,5173	2,8325
Madgwick	F16	Langsam	1,7555	1,2369	1,9253	3,6366	1,7200	2,4381	1,4730	2,2141	2,1321	1,7722
Kalman	F16	Langsam	2,8191	2,5441	2,8329	4,3606	3,9305	3,2192	2,6507	3,6365	3,7911	2,6547
Kompl.	F16	Mittel	2,9442	3,0047	3,5563	2,9116	2,8137	2,6014	2,7379	2,3820	2,9831	2,8208
Mahony	F16	Mittel	2,4335	3,2914	1,6679	2,0397	2,3261	2,4382	1,8401	2,0479	2,1892	3,2406
Madgwick	F16	Mittel	2,3160	3,0293	2,0613	3,2115	2,6849	2,2586	2,4098	2,4160	3,2873	3,9706
Kalman	F16	Mittel	3,5779	4,7074	6,3270	2,8572	2,7518	2,9572	3,4248	4,4246	3,1835	4,0574
Kompl.	F16	Schnell	3,5584	3,1622	2,7673	2,7124	2,3943	3,0070	2,9912	2,5998	2,8754	2,3972
Mahony	F16	Schnell	2,3144	1,8138	2,0979	1,5976	2,5041	3,7759	2,3395	1,3919	1,9570	2,9084
Madgwick	F16	Schnell	1,9189	3,1053	2,4996	1,6882	3,2182	4,1923	3,1680	2,0271	1,8307	3,5632
Kalman	F16	Schnell	3,9136	3,0759	4,4764	3,8389	4,1699	4,8080	4,7583	2,9049	5,7603	4,2354
Kompl.	F32	Langsam	1,4474	1,1773	1,4644	2,7146	1,5492	1,8194	1,4317	1,6823	1,7087	1,1410
Mahony	F32	Langsam	2,7700	1,8835	2,7014	2,4467	1,7414	2,1282	2,0355	1,9480	2,4769	2,5015
Madgwick	F32	Langsam	1,4706	1,1635	1,6575	2,8638	1,6740	1,9011	1,5277	1,7414	1,7693	1,0812
Kalman	F32	Langsam	1,4691	1,2376	1,5218	3,0611	1,6935	1,8155	1,4009	1,8127	1,7600	1,3070
Kompl.	F32	Mittel	2,7765	2,7469	2,7948	2,9772	2,9181	2,5356	2,6356	2,4786	2,8604	2,9966
Mahony	F32	Mittel	2,1261	2,9328	2,1133	1,9755	1,8978	1,7703	2,0383	2,5314	2,2391	3,4192
Madgwick	F32	Mittel	2,3067	3,0271	2,1934	1,9986	2,0704	1,9756	2,2591	2,4345	2,3614	3,1539
Kalman	F32	Mittel	2,0656	2,7014	2,2301	1,9168	1,8318	1,6937	2,0824	2,4858	2,2496	3,1796
Kompl.	F32	Schnell	2,9041	2,9545	2,7265	2,6462	2,4335	3,0886	2,8351	2,3511	2,7277	2,5939
Mahony	F32	Schnell	2,4075	1,7048	1,9828	1,7547	2,4432	2,8385	2,3117	1,4576	2,3069	2,8476
Madgwick	F32	Schnell	2,2161	1,8581	2,3220	1,7022	2,7812	2,8305	2,3367	1,7638	2,1671	3,1605
Kalman	F32	Schnell	2,4739	2,1842	2,4700	2,1269	2,8437	3,1719	2,5306	1,9505	2,4518	3,3631
Kompl.	Float	Langsam	1,4395	1,2039	1,4699	2,7684	1,4929	1,8446	1,4561	1,7150	1,6351	1,1977
Mahony	Float	Langsam	2,7996	1,9308	2,7055	2,4979	1,6642	2,1467	2,0621	1,9673	2,4012	2,5450
Madgwick	Float	Langsam	1,4786	1,1864	1,6193	2,9153	1,6542	1,9334	1,5176	1,7788	1,7227	1,1308
Kalman	Float	Langsam	1,5015	1,2863	1,5834	3,1146	1,5892	1,8463	1,4597	1,7646	1,6805	1,3401
Kompl.	Float	Mittel	2,7639	2,7640	2,7699	3,0319	2,9831	2,6082	2,5840	2,3341	2,9311	3,0371
Mahony	Float	Mittel	2,1097	2,9322	2,1370	2,0447	1,9507	1,8322	1,9709	2,3637	2,3264	3,4632
Madgwick	Float	Mittel	2,3286	3,0873	2,1734	2,0395	2,1219	2,0589	2,2393	2,2936	2,4226	3,1026
Kalman	Float	Mittel	2,0526	2,6102	2,1118	2,0048	1,9153	1,7225	2,0951	2,4062	2,3296	3,2758
Kompl.	Float	Schnell	2,6338	3,0783	2,7681	2,6318	2,5443	3,1081	2,8336	2,4832	2,5553	2,7118
Mahony	Float	Schnell	2,3986	1,8048	1,9859	1,6998	2,5198	2,8632	2,2687	1,5690	2,0964	2,9361
Madgwick	Float	Schnell	2,2025	1,9210	2,3620	1,6585	2,8539	2,8044	2,3328	1,8363	2,0498	3,2040
Kalman	Float	Schnell	2,5155	2,3050	2,4205	2,1222	2,9021	3,1763	2,4513	2,0671	2,1997	3,4938

Die Werte der Messung sind die mittleren quadratischen Fehler im Vergleich zur Referenz. F16 sind 16 Bit-Festkommazahlen, F32 die 32 Bit-Festkommazahlen und Float die 32 Bit-Fließkommazahlen. Die *langsame* Geschwindigkeit bezieht sich auf die Messungen mit einer durchschnittlichen Winkelgeschwindigkeit von 7,5 °/s, die mittlere Geschwindigkeit auf eine durchschnittliche Winkelgeschwindigkeit von 15 °/s und die schnelle Geschwindigkeit auf eine durchschnittliche Winkelgeschwindigkeit von 30 °/s.