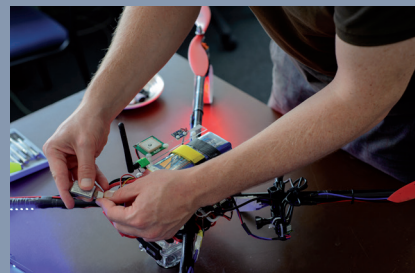
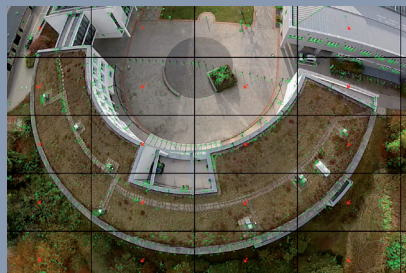
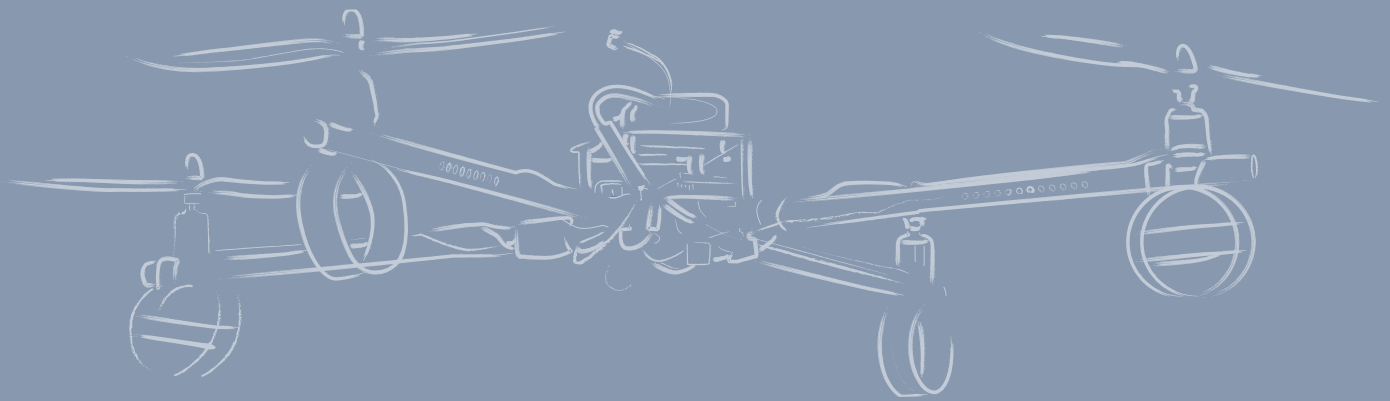


# NeuroCopter: Optimierte Verfahren zur Positions- und Lagebestimmung eines biomimetischen Quadrocopters



**Diplomarbeit**  
Fachbereich Mathematik und Informatik  
Freie Universität Berlin

Philipp Breinlinger

Erstgutachter: Prof. Dr. Raúl Rojas  
Zweitgutachter: Dr. Hamid Mobbalegh  
Betreuer: Dr. Tim Landgraf



## Erklärung

Hiermit erkläre ich, dass ich die vorliegende Diplomarbeit selbstständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich genannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht.

---

Ort, Datum

---

Unterschrift

## Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die ich mich vor und während der Zeit unterstützt haben und mir zur Seite standen.

Mein besonderer Dank gilt:

*Anja*, die meine Motivation ist, mich weiterzuentwickeln und mich dabei wohlwollend anschiebt. Ohne dich bliebe ich ewig Student 😊

*Meinen Eltern*, für Geduld und finanzielle sowie moralische Unterstützung während fast 25 Jahren Ausbildung.

*Hildegard & Gerhard*, für die Bereitschaft, mir über Jahre einen persönlichen Studienleistungsunterstützungsbeitrag zu gewähren.

*Tim Landgraf*, der mich in diese so vielfältige Arbeitsgruppe geholt und mir in Zeiten der Orientierungslosigkeit den nächsten Schritt aufgezeigt hat.

*Prof. Dr. Raúl Rojas*, für die Ermöglichung meiner Diplomarbeit und das früh im Studium geweckte Interesse an Robotern und künstlicher Intelligenz.

*Tobi*, mit dem man gleichzeitig breit und wahnsinnig tief in die Materie einsteigen kann. Vor deinen Fähigkeiten zieh ich den Hut!

*Philipp Nowak*, der mir das Fliegen beigebracht und mit mir die vielen spannenden Auftritte absolviert hat. An manche Momente werde ich mich lange erinnern!

*Benni*, der immer hilfsbereit und mir gerade in der finalen Phase durch bloße Anwesenheit und Zuhören vor dem Wahnsinn bewahrt hat.

*Elias*, bei dem ich nie Angst haben musste, auch mal blöde Fragen zu stellen. Vielen Dank für deine Geduld und die Beantwortung meiner Fragen .

## Abstract

Im interdisziplinären Projekt "NeuroCopter" der Informatik und Neurobiologie der FU Berlin werden die kognitiven Fähigkeiten der europäischen Honigbiene erforscht und mit Hilfe von "Spiking Neural Networks" simuliert. Zur Überprüfung verschiedener Hypothesen über das Navigationsverhalten und die Fähigkeiten natürlicher Organismen können autonome Agenten eingesetzt werden. Zu diesem Zweck ist es notwendig, einen Roboter zu entwickeln, der die natürlichen Fähigkeiten des Organismus auf sensorischer, motorischer und kognitiver Ebene imitiert. Im Rahmen dieser Arbeit wurden verschiedene Komponenten des Multicoptersystems weiterentwickelt und ein Verfahren zur Kamerabasierten Positionsbestimmung anhand farblich kodierter Landmarken entwickelt.

**Schlagwörter:** biomimetische roboter, biomimetik, multicopter, mav, uav, honigbiene, navigation, landmarken, computer vision, connected components, binarisierung, entzerrung, lokalisierung, richtungsfindung, opencv, kamera,

## Inhaltsverzeichnis

ERKLÄRUNG.....	3
DANKSAGUNG.....	4
<b>1 EINLEITUNG.....</b>	<b>8</b>
1.1    MOTIVATION.....	8
1.2    AUFBAU DER ARBEIT.....	9
<b>2 VERWANDTE ARBEITEN.....</b>	<b>10</b>
<b>3 SYSTEMENTWURF NEUROCOPTER.....</b>	<b>13</b>
3.1    ELEKTROMECHANISCHER FLUGGRAHMEN.....	13
3.2    SENSORIK.....	14
3.2.1 <i>Digitales 3-Achsen Gyroskop.....</i>	<i>14</i>
3.2.2 <i>Digitale 3-Achsen Beschleunigungssensor (Accelerometer).....</i>	<i>15</i>
3.2.3 <i>Luftdrucksensor (Barometer).....</i>	<i>17</i>
3.2.4 <i>Ultraschallsensor (Sonar).....</i>	<i>17</i>
3.2.5 <i>3-Achsen-Magnetometer (Kompass).....</i>	<i>18</i>
3.2.6 <i>GPS.....</i>	<i>19</i>
3.2.7 <i>Kamerasystem für optischen Fluss.....</i>	<i>19</i>
3.2.8 <i>Kamera für Landmarkenerkennung.....</i>	<i>19</i>
3.3    FLUGSTEUERUNG.....	20
3.3.1 <i>Low-Level Flugsteuerung (ArduPilot Mega 2.5).....</i>	<i>20</i>
3.3.2 <i>Konventionelle High-Level Flugsteuerung (IGEP).....</i>	<i>20</i>
3.3.3 <i>Neuronale Steuerarchitektur.....</i>	<i>21</i>
<b>4 SYSTEME ZUR POSITIONSBESTIMMUNG (NAVIGATION).....</b>	<b>23</b>
4.1    EXTERNE VERFAHREN.....	23
4.1.1 <i>VICON /Motion Capturing.....</i>	<i>23</i>
4.1.2 <i>Global Positioning System (GPS).....</i>	<i>24</i>
4.2    INTEGRIERTE VERFAHREN.....	24
4.2.1 <i>Koppelnavigation.....</i>	<i>24</i>
4.2.2 <i>Inertiale Navigation (INS).....</i>	<i>24</i>
4.3    OPTISCHE NAVIGATIONSVERFAHREN.....	25
4.3.1 <i>Kamera als Sensor.....</i>	<i>25</i>
4.3.2 <i>Egomotion via Optischem Fluss.....</i>	<i>25</i>
4.3.3 <i>Positionsbestimmung anhand von optischen Merkmalen.....</i>	<i>25</i>

<b>5</b>	<b>KAMERABASIERTE POSITIONSBESTIMMUNG ANHAND KODIERTER LANDMARKEN.....</b>	<b>27</b>
5.1	SPEZIFIKATION DER ANFORDERUNGEN.....	27
5.2	GRUNDLAGEN DES VERFAHRENS ZUR POSITIONSBESTIMMUNG.....	28
5.3	ABLAUF DER POSITIONSBESTIMMUNG.....	30
5.3.1	<i>Berechnung der Rotation.....</i>	<i>30</i>
5.3.2	<i>Rotation des Richtungsvektors.....</i>	<i>31</i>
5.3.3	<i>Ermittlung der Schnittebene und des Abstands.....</i>	<i>31</i>
5.3.4	<i>Berechnung der dreidimensionale Position des Copters.....</i>	<i>31</i>
5.4	IMPLEMENTIERUNG.....	32
5.4.1	<i>Entwicklungsumgebung / Bibliotheken.....</i>	<i>32</i>
5.4.2	<i>Feststellung der intrinsischen Kameraparameter / Kamerakalibration.....</i>	<i>32</i>
5.4.2.1	Kalibration der Kamera.....	32
5.4.2.2	Korrektur des Kamerabildes.....	33
5.4.2.3	Konvertierung der Farbräume des Kamerabilds von RGB zu HSV.....	33
5.4.3	<i>Erkennung einer Landmarke aus der Verarbeitung des Kamerabildes.....</i>	<i>34</i>
5.4.3.1	Binarisierung.....	34
5.4.3.2	Analyse der Zusammenhangskomponenten (ZHK).....	35
5.4.3.3	Filtern der Zusammenhangskomponenten.....	35
5.4.3.4	Positionsbestimmung der Zusammenhangskomponente im Kamerabild.....	36
5.4.4	<i>Berechnung der Rotation durch Landmarke im Kamerabild.....</i>	<i>36</i>
5.4.4.1	Normalisierung der Bildkoordinaten.....	36
5.4.4.2	Berechnung der Rotation aus der normalisierten Pixelposition.....	37
5.4.5	<i>Abschließende Verarbeitungsschritte.....</i>	<i>37</i>
5.5	EINSATZ DER SOFTWARE.....	37
<b>6</b>	<b>EVALUATION .....</b>	<b>39</b>
6.1	SZENARIO 1: POSITIONSBESTIMMUNG DER KAMERA.....	40
6.1.1	<i>Messreihe 1: Pitch -30°.....</i>	<i>41</i>
6.1.2	<i>Messreihe 2: Pitch -45°.....</i>	<i>42</i>
6.2	SZENARIO 2: POSITIONSBESTIMMUNG DER LANDMARKE.....	42
<b>7</b>	<b>DISKUSSION UND AUSBLICK .....</b>	<b>45</b>
<b>8</b>	<b>LITERATURVERZEICHNIS .....</b>	<b>46</b>
	ABBILDUNGSVERZEICHNIS.....	49

# 1 Einleitung

Abseits der militärischen Nutzung hat die Entwicklung kleiner, unbemannter Luftfahrzeuge (UAVs: englisch „unmanned aerial vehicles“) die Aufmerksamkeit unterschiedlicher ziviler Interessensgruppen geweckt. Durch Programmierung automatisierter Abläufe und vergleichsweise günstige Anschaffung eröffnen für UAVs vielseitige Einsatzmöglichkeiten, wie zur Erforschung und Kartographie von Gelände, dem Transport von Gütern, der Inspektion von Gebäudeschäden und vieles mehr.

Um eine autonome Arbeitsweise zu ermöglichen gilt es intelligente Prozesse auf Computern umzusetzen. Um Maschinen mit vergleichbaren Fähigkeiten eines intelligenten Organismus auszustatten müssen dabei vielfältige zentrale Herausforderungen, wie Wahrnehmung, Planung, Lernen und Entscheidungsfindung verstanden und umgesetzt werden. Auf Ebene der Wahrnehmung müssen dafür Umgebungsinformationen über Sensoren gesammelt, in einem anschließenden Prozess verarbeitet und zur Erlangung einer anschließenden intelligenten Entscheidungsfindung zu einem kohärenten Gesamtbild zusammengeführt werden. Hierfür ist es notwendig Fähigkeiten aber auch Limitierungen der Sensoren zu verstehen und entsprechend zu interpretieren.

Die besondere Herausforderung der dynamischen Interaktion mit einem natürlichen, häufig unstrukturierten Umfeld erfordert eine aufwändige und rechenintensive Verarbeitung der Umwelteindrücke. Durch Evolution erprobte Konzepte der Natur sollen die Vorlage sein, um neuartige, rechnerisch elegante Ansätze für die Lösung komplexer Aufgaben im Bereich der Roboternavigation zu entwickeln.

## 1.1 Motivation

Leitmotiv für das Projekt "NeuroCopter" und damit Motivation für diese Arbeit ist die Erforschung des Verhaltens der Honigbiene. Die Fähigkeiten dieser kleinen Lebewesen müssen selbst diejenigen erstaunen, die nur oberflächlich Ahnung über ihre Lebensweise und Verhalten haben. Mit einem Gehirn das nur wenige Milligramm wiegt, sind sie in der Lage, komplexe Aufgaben auszuführen: Über große Distanzen Nahrungsquellen aufspüren, wiederholt aufsuchen und deren Position per Bienentanz an ihre Artgenossen im Bienenstock kommunizieren. [1] Honigbienen dienen darum seit geraumer Zeit als Modellorganismus für die Forschung. Das Studium ihrer Fähigkeiten zur Navigation soll Aufschluss geben über die zugrundeliegenden Mechanismen welche es ihr ermöglichen mit begrenzten Mitteln in der Umwelt zu navigieren und sich selbst zu lokalisieren.

Die Analyse der rezeptiven und kognitiven Fähigkeiten der Honigbiene verspricht Antworten auf grundlegende Fragen: Auf welche Weise können Bienen Objekte anhand von Form, Farbe oder Textur erkennen? Wie nehmen sie den dreidimensionalen Raum ihrer Umwelt wahr? Welche visuellen Informationen erlauben ihnen, Entfernung und Richtung ihres Flugs zu ermitteln?

Aktuelle Erkenntnisse deuten darauf hin, dass Honigbienen nicht nur "erfolgreiche" Flugrouten abspeichern und abrufen, sondern fortlaufend ein kartenähnliches Bild der Umgebung in ihrem Gedächtnis entwickeln. [2] Neben der Fähigkeit, mittels Pfadintegration ihrer Flugrouten eine robuste Schätzung über Distanz und Orientierung zu ihrem Bienenstock zu erlangen, merken sie sich die Position auffälliger optischer Landmarken um ihre eigene Positionsschätzung zu verbessern. [3] [4]

Bei Bienen ist der neuronale Prozess von der sensorischen Wahrnehmung hin zur Entscheidungsfindung und Bewegungsplanung bzw. der motorischen Durchführung noch nicht verstanden. Jedoch macht die Tatsache, dass sie mit begrenzten neuronalen Ressourcen ein derart



robustes Verhalten aufweisen, sie zu einem Modell für das Studium der kognitiven Anforderungen zur Navigation. Die Erkenntnisse ermöglichen es, biologisch inspirierte, effiziente, Algorithmen und Hardware zu entwickeln und diese zur Steuerung von Robotern einzusetzen. Dies dient nicht nur dazu, vorhandene Technologien weiterzuentwickeln und zu verbessern, sondern erlaubt es, natürliches Verhalten von Lebewesen zu simulieren und dabei interne Prozesse nachzuvollziehen und dadurch besser zu verstehen.

Im interdisziplinären Projekt "NeuroCopter" der Robotik und Neurowissenschaften soll mit neuronalen Modellen des Gehirns die Fähigkeiten der Honigbiene zur Navigation und Lokalisation erforscht werden. Diese Modelle werden nicht nur innerhalb einer Computersimulation trainiert und getestet, sondern durch Steuerung eines fliegenden Roboters überprüft und weiterentwickelt. Dazu werden Verhalten und Flugrouten des Roboteragenten mit aufgezeichneten Radartracks natürlicher Bienenflüge verglichen.

Im Rahmen dieser Diplomarbeit wurden Komponenten der Roboterplattform weiterentwickelt und eine Methode zur kamerabasierten Positionsbestimmung anhand von Landmarken entworfen und umgesetzt.

## **1.2 Aufbau der Arbeit**

Die vorliegende Arbeit hat folgende Struktur: Kapitel 2 schildert den wissenschaftlichen Kontext anhand verwandter Projekte und Forschungsarbeiten. Kapitel 3 gibt eine Beschreibung des Entwurfs und der Details der Multicopter-Plattform mit mechanischen und elektronischen Komponenten. Außerdem werden die innerhalb dieser Arbeit vorgenommenen Modifikationen an der Hardware herausgestellt. Zur Einordnung der eigenen Methode zeigt Kapitel 4 in einer Übersicht Charakteristiken, wie Vor- und Nachteile der verschiedenen Navigationssysteme. In Kapitel 5 folgt die Vorstellung des implementierten Verfahrens zur Positionsbestimmung anhand von Landmarken. Die Entwicklung wird logisch nachvollzogen von der Anforderungsanalyse, über Entwurf, die mathematischen Grundlagen, Implementierung und Anwendungsmöglichkeiten. Die Evaluation des Verfahrens folgt in Kapitel 6, bevor im abschließenden Kapitel 7 die Erkenntnisse zusammengefasst werden und ein Ausblick in die Zukunft stattfindet.

## 2 Verwandte Arbeiten

Aufgrund der interdisziplinären Ausrichtung des Projekts NeuroCopter zwischen Biologie und Robotik finden sich Schnittpunkte zu Arbeiten in beiden Disziplinen.

Seitens der Robotik gibt es steigende Anzahl an Projekten mit autonomen Flugrobotern. Viele Arbeiten verwenden noch relativ große und schwere UAVs mit einem Gesamtgewicht über 5kg. Durch die entsprechende Motorik ist es möglich, ein umfangreiches Repertoire an Sensorik als Nutzlast mit an Board zu tragen, z.B. LIDAR, Radar, Kameras und Ultraschallsensoren, aber eben auch leistungsstärkere Computersysteme mitzuführen. Das erlaubt es Aufgaben, wie Abheben, Landen oder Hindernisvermeidung, auf rechenintensive Weise durchzuführen. [5] [6] Die gleichen Resultate auf einem unter 2kg schweren System durchzuführen, bedarf Anpassung der Hardware, gegebenenfalls auch der Algorithmen. Einige Projekte setzen dabei auf LIDAR-Abstandssensoren. [7] Ansätze zur Kamerasicht-basierten Flugsteuerung gehen den Weg, die Verarbeitung "off-board" - also nicht auf dem Roboter selbst, sondern auf einer via Funk verbundenen Workstation am Boden auszuführen. [8] Dies hat den Grund, dass viele der konventionellen Algorithmen wie SLAM („Simultaneous localization and mapping“) mit steigender Komplexität einen wachsenden Rechen- und Speicherbedarf haben und eine besonders strukturierte Umgebung voraussetzen. [9] [10] Fortschritte in der Weiterentwicklung der Hardware haben dazu geführt, dass die rechenintensiven Aufgaben zur Vision-basierten Navigation vermehrt ohne Unterstützung externer Hochleistungsrechner auf der fliegenden Plattform berechnet werden können. [11] [12]

Das Studium natürlicher Prinzipien um Anhaltspunkte für effizientere Verfahren zu bekommen ist nicht neu, biologische Navigationstechniken wurden an verschiedenen kleinen Organismen, wie der Wüstenameise, untersucht. [13] Eine anhand des Ameisenverhaltens entwickelte Modellstrategie sieht eine Kopplung der Bewegung und Blickrichtung voraus. Vertraute Bewegungen sind mit vertrauten Ansichten der Umgebung verknüpft, die Navigation entlang einer bestimmten Route kann als Suche nach vertrauten, assoziierten Bildern interpretiert werden. Durch spezielle Suchbewegungen können sie mögliche Routen mit gemerkten perspektivischen Ansichten ("Snapshots") abgleichen.

Eine Hypothese über die Orientierung bei Honigbienen sieht eine ähnliche Technik des systematischen Lernens und Abgleichens mit Schnappschüssen vor. Die Richtung zu einer bekannten Futterquelle kann zunächst mit einer entfernten Landmarken als Orientierungspunkt über größere Distanz bestimmt werden. Bei Annäherung wird die Position weiter präzisiert, indem auffällige Bezugspunkte in der Umgebung der Futterquelle gemerkt und bei wiederholtem Aufsuchen abgeglichen werden. [4] Basierend auf Untersuchungen der Wüstenameise wurde alternativ das "Average Landmark Vector" (ALV)-Modell entwickelt. [14] Es basiert ebenfalls auf der Positionsbestimmung anhand von Landmarken, im Gegensatz zu genannten Strategien werden aber nicht Schnappschüsse gespeichert, sondern die Summe der Richtungsinformationen zu verschiedenen Bezugspunkten gibt Aufschluss über die eigene Position. Die Anwendung der Strategien wie dem Snapshot- oder ALV-Modell und biologisch inspirierter Sensorik wurde bereits erfolgreich innerhalb einer Simulation [15] oder auf einem bodengebundenen Roboter [16] getestet.

Das Projekt NeuroCopter hat die Zielsetzung, biologisches Verhalten und Navigationsfähigkeiten auf einem autonomen Roboteragenten zu erforschen. Im Gegensatz zu vergleichbaren Projekten steht hier jedoch die Weiterentwicklung des neuronalen Programm-Paradigmas im Vordergrund. Jede Aufgabe lässt sich als vernetzter Prozess unterschiedlicher kognitiver Funktionen darstellen, die aufeinander folgen und miteinander interagieren. Grundlegende Funktionen werden individuell innerhalb neuronaler Netze trainiert, diese können dann als Bauteile wiederum in einer übergeordneten Struktur

vernetzt werden. Komplexe Tätigkeiten werden so aus verschiedenen Bestandteilen programmiert, der konkrete Verarbeitungsprozess ist dabei allein durch die Struktur des Netzwerkes definiert. Die Fähigkeit zur Navigation entwickelt sich aus der bestimmten Vernetzung vorher trainierter Aufgaben wie Lokalisation, Mapping und Planung. Die Funktionsweise biologischer Neuronaler Netzwerke wird mit komplexen, "spiking" Netzwerken simuliert und studiert. Mit konventioneller Hardware ist es noch nicht möglich die notwendige Rechenleistung in Echtzeit onboard bereitzustellen, in Zukunft verspricht jedoch die Entwicklung neuromorpher Hardware Abhilfe. [17]

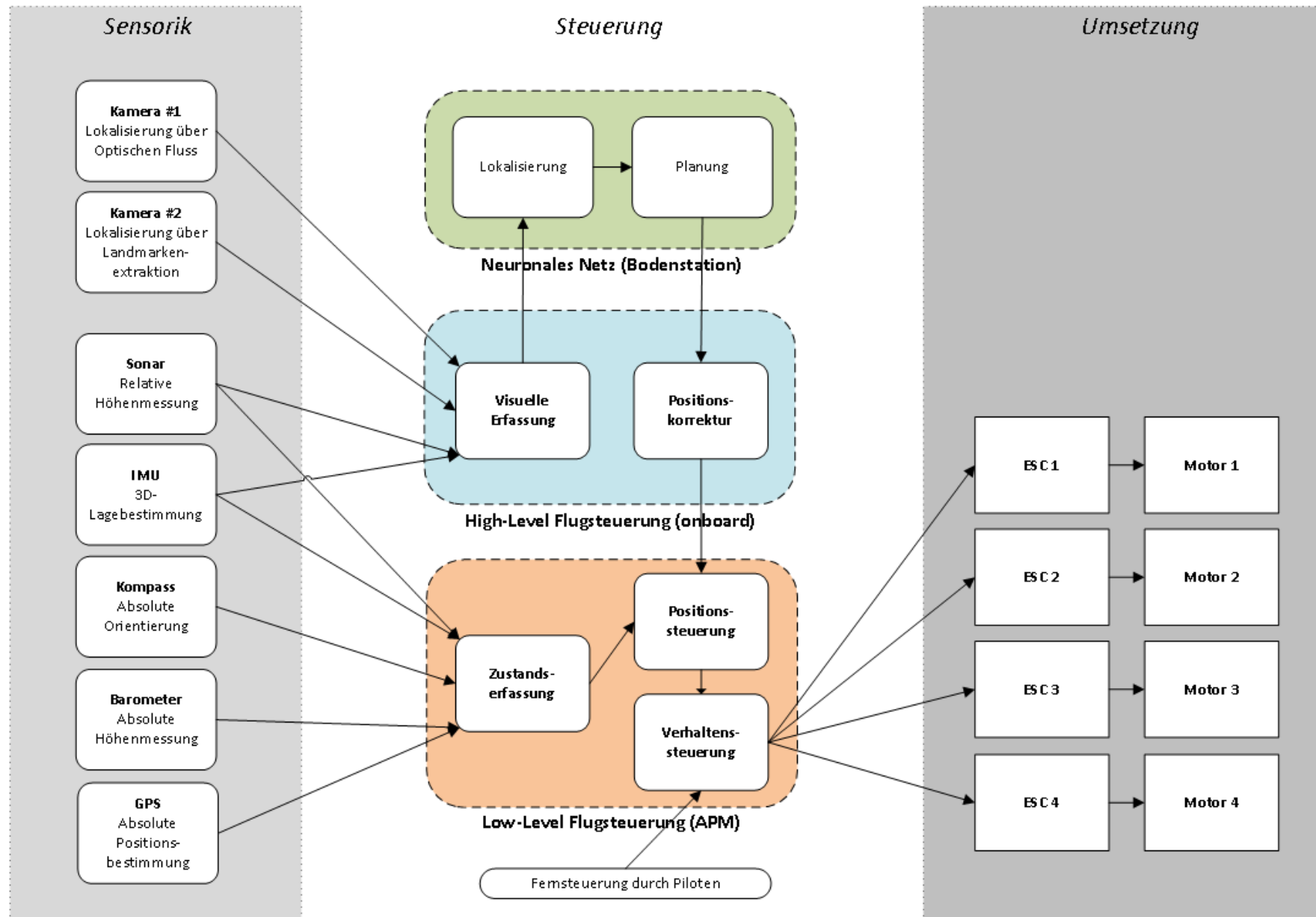


Abbildung 1: Systementwurf NeuroCopter. Die Informationen der Sensorik (links) werden innerhalb der Flugsteuerung verarbeitet und interpretiert. Lokalisierung innerhalb des neuronalen Netzes soll durch Mapping von Landmarken und Berechnung des optischen Flusses geschehen [32]. Geplante Bewegungen werden in Positionskorrekturen umgerechnet und von der Low-Level-Flugsteuerung in Motorbefehle umgesetzt.

### 3 Systementwurf NeuroCopter

Ausschlaggebend für Entwurf des Gesamtsystems war die Zielsetzung der Entwicklung eines autonomen Agenten für biomimetische Versuche. Dies bedingt Anforderungen an die Konzeption der Flugplattform, die Wahl der Sensorik und Computer-Hardware, wie auch der Navigationsverfahren. Ein wichtiger Aspekt war eine modulare Architektur zu entwickeln, um in weiteren Entwicklungsstufen Komponenten anzupassen oder Bestandteile in zukünftige Roboter portieren zu können.

Um dies hervorzuheben erfolgt eine schichtweise Trennung zwischen dem Multicopter, der Sensorik und der Flugsteuerung. Abbildung 1 zeigt das Zusammenwirken der verschiedenen Systemkomponenten innerhalb und zwischen den Ebenen. Zunächst wird die Flugplattform und ihre Bestandteile erläutert, darauf aufbauend die verfügbare Sensorik und sowohl die aktuelle, „klassische“, zweistufige Steuerarchitektur als auch deren Schnittstelle zu der anvisierten, neuronalen Steuerung vorgestellt.

#### 3.1 Elektromechanischer Flugrahmen

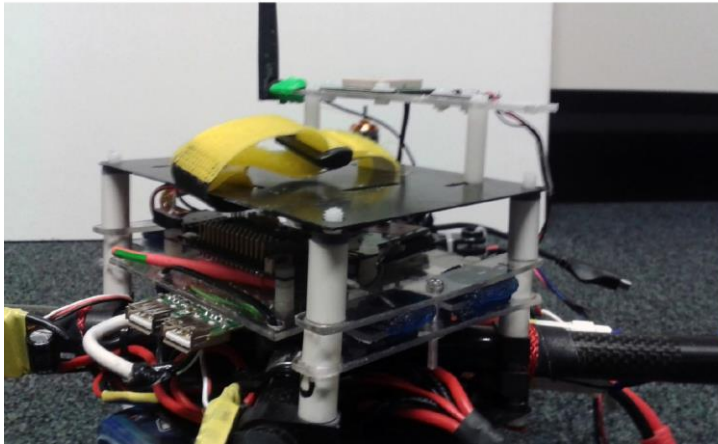
Als Äquivalent zum biologischen Körper setzt der elektromechanische Flugrahmen die Steuerbefehle des Flugcontrollers in Bewegungen um. Die Bestandteile der Plattform orientieren sich dabei weitgehend an dem typischen Aufbau handelsüblicher Multicopter, wie sie auch beim ferngesteuerten Modellflug im Hobbybereich vorzufinden sind. Bei dem Rahmen handelt es sich um einen Eigenbau, der von zwei weiteren Mitgliedern der Arbeitsgruppe (Tobias Ludwig und Philipp Nowak) für das Projekt NeuroCopter entworfen und maßgefertigt wurde. Abbildung 2 zeigt den aktuellen Quadcopter in der Gesamtansicht.



Das Gestell besteht aus vier in „X“-Formation gleichmäßig angeordneten Karbonrohren (1), die über ein Aluminiumkreuz zentral verbunden und fixiert sind. An der Unterseite der Arme befinden sich LED-Streifen mit unterschiedlichen Farben, die es einem Piloten am Boden erleichtern die Orientierung in der Luft festzustellen. Am Ende der Arme befindet sich jeweils ein elektrischer Gleichstrommotor (2), der einen 12" Kunststoff-Propeller (3) antreibt. Über elektronische Schubkontroller (4) können die Motoren individuell angesteuert werden. In der Mitte befindet sich ein dreistöckiger Käfig, bestehend aus Boden-/Deckplatte und einer gedämpft aufgehängten Plattform für Elektronik und vibrations-

empfindliche Sensorik. Eine Nahaufnahme des Käfigs sieht man in Abbildung 3\*. Der Käfig beherbergt neben dem Empfänger für die Fernsteuerung auch die zentrale Stromversorgung. Als Quelle dient ein 3-zelliger 5000mAh Lithium-Polymer-Akku (5), dessen Spannung für die Steuerelektronik und die verschiedenen Sensoren parallel zur Versorgung der Motoren über einen UBEC Spannungswandler geregelt wird.

ABBILDUNG 3: NAHANSICHT KÄFIG MIT COPTERELEKTRONIK



#### DAS RÜCKENMARK

Die Schaltzentrale regelt die Kommunikation zwischen Gehirn und Multicopter.

## 3.2 Sensorik

Jenseits der tragenden Copterplattform wurde der NeuroCopter zur Messung des eigenen Zustandes und zur Wahrnehmung der Umgebung mit elektronischer Sensorik versehen. Zum Teil sind diese direkt auf der Platine des Flugcontrollers verbaut, während weitere extern am Rahmen angebracht sind. Innerhalb der Unterpunkte dieses Kapitels werden die jeweiligen Sensoren vorgestellt. Außerdem werden Motivation für die Auswahl und vorgenommene Modifikationen aufgezeigt.

### 3.2.1 Digitales 3-Achsen Gyroskop

Um Bewegungen im freien Raum ohne externe Referenzen feststellen zu können werden Sensoren verwendet, die im weiteren Sinn auf dem mechanischen Prinzip der Trägheit basieren<sup>†</sup>. Gyroskope sind geschlossene Kreiselssysteme, deren Drehimpuls ohne Einwirkung einer äußeren Kraft konstant bleibt. Versucht eine Kraft die Drehachse des Kreisels zu kippen, resultiert ein Drehmoment. Die Integration des Moments über die Zeit ergibt den Drehwinkel und damit die Lage des Sensors. Der Beitrag einer Messung der Copter- bzw. Kameraneigung innerhalb des entwickelten Verfahrens zur Positionsbestimmung wird in Kapitel 4.3 verdeutlicht.

Auf der Flugcontrollerplatine verbaut ist ein Bewegungssensor Modell MPU-6000 der Firma Invensense. Der Microchip setzt sich aus einem digitalen 3-Achsen Gyroskop und einem 3-Achsen Beschleunigungssensor zusammen und erlaubt die Bestimmung der Bewegung in sechs Freiheitsgraden (drei Rotation, drei Translation). [18]

---

\* Auf die mittige Elektronikplattform und die Auswirkung der gelbasierten Vibrationsdämpfung in Bezug auf den Beschleunigungssensor wird in Kapitel 3.2.1.2 im Detail eingegangen.

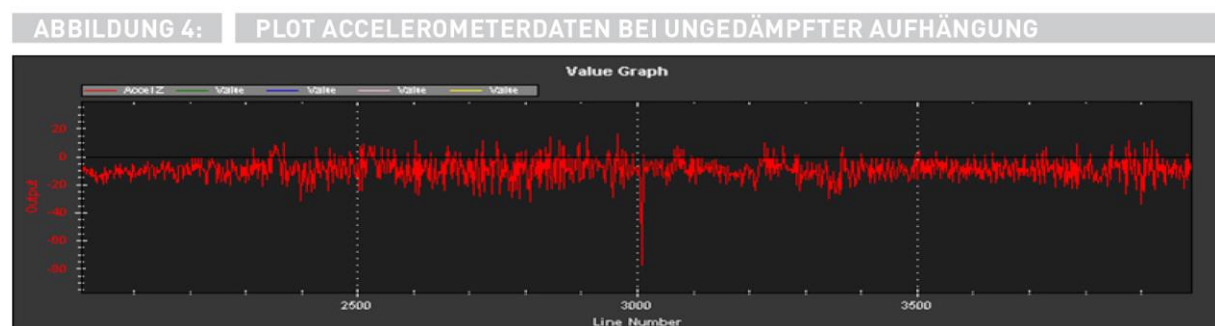
<sup>†</sup> Die Massenträgheit eines Körpers dient auch als definierende Größe bei Inertialsystemen (siehe Kapitel 4.1.2.2).

### 3.2.2 Digitaler 3-Achsen Beschleunigungssensor (Accelerometer)

Im Gegensatz zum Drehmoment beim Gyroskop misst der Beschleunigungssensor die Veränderungen der Beschleunigungskräfte auf eine Masse entlang der Raumachsen. Man kann sich die Arbeitsweise des Beschleunigungssensors anhand einer federgedämpften Masse vorstellen. Wirkt eine beschleunigende Kraft von außen auf den Sensor ein, verschiebt sich die Position dieser Masse relativ zur Neutralposition analog zur Stauchung der Feder. Die Verschiebung der Masse entspricht dann dem Maß der Beschleunigung bis die Masse auf die gleiche Geschwindigkeit wie das Gehäuse beschleunigt wird. Findet keine zusätzliche Krafteinwirkung durch Bewegung des Sensors statt, entspricht die gemessene Beschleunigung der umgekehrten Erdgravitation  $G^*$  in Z-Richtung. [19] Will man in der Praxis durch zeitliche Integration der Beschleunigung die Bewegung messen, muss für diese kompensiert werden. Über die Winkel zwischen den Sensorachsen und der vertikal wirkenden Erdbeschleunigung lässt sich außerdem die Lage des Sensors bestimmen.

Ein Problem bei der Berechnung einer Bewegung über die gemessenen Beschleunigungskräfte ist durch Vibrationen verursachtes Rauschen in den Messdaten. Dies machte sich in der Praxis bemerkbar, da in der ursprünglichen Konfiguration das Accelerometer ungedämpft auf die untere Karbonplatte des Elektronikkäfigs montiert wurden. Mit dem Release von Version 2.9 wurde die Flugcontroller-Firmware mit einem inertialen Navigationssystem erweitert<sup>†</sup>. Dies sollte vor allem die Präzision im „Position halten“-Modus (sog. „Loitern“, deutsch: „herumlungern“) verbessern. Loitern heißt, dass der Flugcontroller versucht, durch Korrekturen der Motordrehzahl die beim initiieren des Flugmodus aktuelle Position sowohl vertikal als auch horizontal beizubehalten. Die Umstellung auf Firmware-Version 2.9 resultierte zunächst bei vielen Anwendern in einer subjektiven Verschlechterung der Performance, gekennzeichnet durch teils zu starke, teils zu schwache Korrekturmanöver des Autopiloten bei der Kompensation von unerwünschten Driftbewegungen.

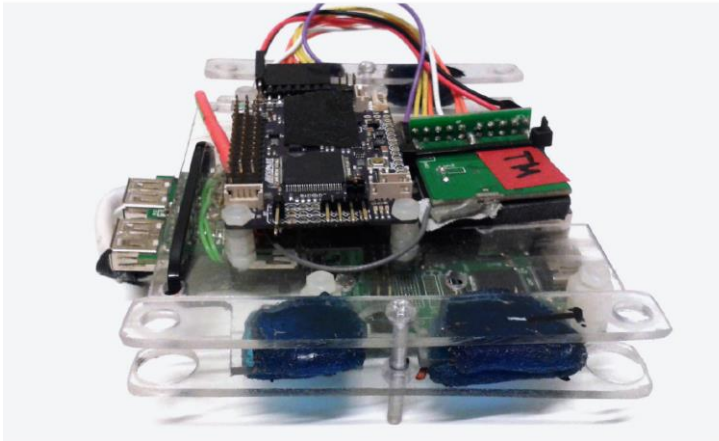
Eine Auswertung der Flug-Logdateien nach Diskussion mit den Entwicklern ergab stark verrauschte Accelerometerdaten und damit Vibrationen des Flugrahmens als potentielle Ursache. Abbildung 4 zeigt beispielhaft die starken Schwankungen der Messwerte des Beschleunigungssensors entlang der vertikalen Z-Achse während eines zwanzig Sekunden dauernden Fluges im „Loiter“-Modus. Während des Fluges wurden keine externen Steuerbefehle durch den menschlichen Piloten vorgenommen. Zu beachten ist die erwähnte Erdbeschleunigung, die ein Offset der Messungen von rund  $-10\text{m/s}^2$  bewirkt. Im Diagramm klar zu erkennen ist die Streuung der Messwerte im Bereich von  $\pm 10\text{m/s}^2$  um den Sollwert von  $-10\text{m/s}^2$ . Um dem Problem entgegenzuwirken sollte eine Lösung entwickelt werden, die den Einfluss der Vibrationen auf den Sensor dämpft. Gleichzeitig sollte der Sensor nicht zu stark entkoppelt sein, damit er die tatsächlich stattfindenden Bewegungen korrekt und möglichst verzögerungsfrei wahrnehmen kann.



\* Auf der Erdoberfläche ist  $G$  ungefähr  $9,81\text{ m/s}^2$

† Siehe Kapitel 4.1.2.2 für die Funktionsweise

ABBILDUNG 5: ELEKTRONIKPLATTE MIT MOONGEL-DÄMPFUNG



#### WIE MAN SICH BETTET...

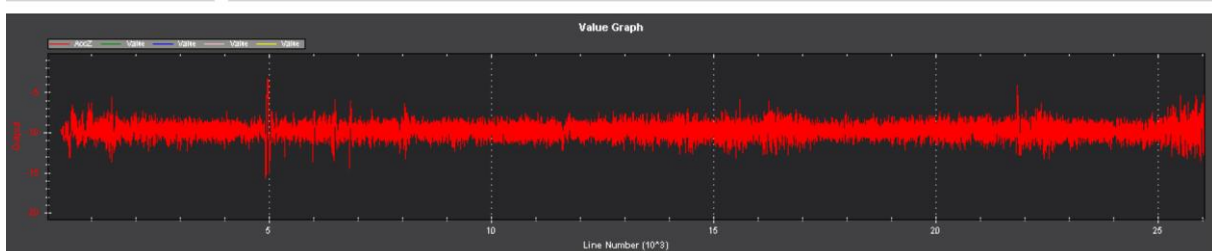
Acht Gelkissen verhindern die Übertragung von Vibrationen der Motoren auf die Beschleunigungssensoren.

Bei der Konzeption der Vibrationsdämpfung wurden die unterschiedlichen Ansätze und Erfahrungsberichte verschiedener Nutzer der Arducopter-Firmware als Richtlinie herangezogen. [20] Ein Großteil der dort diskutierten Lösungen setzen hierbei auf gel- oder schaumstoffbasierte Materialien [21].

Bei der Konzeption einer Dämpfung müssen verschiedene Kriterien berücksichtigt werden. Die effektive Wirkung hängt nicht nur von materialspezifischen Eigenschaften wie der Elastizität ab, sondern ebenfalls von der Größe der Kontaktfläche sowie der zu dämpfenden Masse. Die Masse des Flugcontroller-Board mit betroffener Sensorik von knapp 25 Gramm bedingt einen kleinen Spielraum bei der Wahl und Dimensionierung des Materials. Stattdessen wurde die zu dämpfende Masse erhöht, indem APM, Onboard-Computer und weitere Elektronik über eine Kunststoff-Platte fest aneinander gekoppelt wurden. Die insgesamt knapp 150 Gramm schwere Platte wird mit Hilfe einer justierbaren Schraube horizontal in eine Bügelkonstruktion eingeklemmt, wobei sämtliche Kontaktflächen durch acht daumennagelgroße Gel-Kissen\* von den Bügeln isoliert wurden. Abbildung 5 zeigt den Prototypen mit montierter Elektronik und der dämpfenden Bügelkonstruktion. Über Kunststoffhülsen wird die komplette Konstruktion in der Mitte des Elektronikkäfigs fixiert (siehe Abbildung 3 in Kapitel 3.1).

Zur Validierung der angestrebten Dämpfungswirkung wurde im Anschluss ein weiterer Testflug unternommen. Erneut wurde hierbei der „Position Halten“ Modus aktiviert, um ein repräsentatives Ergebnis ohne Verfälschung durch Interaktion des Piloten zu erhalten. Neben einer subjektiven Verbesserung des Verhaltens ergab die Auswertung der Logdateien des zweiten Testflugs deutlich geringere Schwankungen der Messungen des Beschleunigungssensors im Bereich von  $\pm 3m/s^2$  (Abbildung 6). Nachteile durch die indirekte Koppelung des Flugcontrollers an den Flugrahmen wurden nicht festgestellt, die vorgenommenen Maßnahmen können demnach als Erfolg angesehen werden.

ABBILDUNG 6: PLOT ACCELEROMETERDATEN BEI GEDÄMPFTER AUFHÄNGUNG



\* Das verwendete Material ist Moongel [44]. Die Gelpads werden sonst zur Dämpfung von Schlagzeugen verwendet.



### 3.2.3 Luftdrucksensor (Barometer)

Weiterer Bestandteil der Sensorik ist ein Barometer, welches Temperatur und atmosphärischen Luftdruck misst. Zwischen Temperatur, Luftdruck und Höhe auf der Erde besteht ein fester Zusammenhang, über die barometrische Höhenformel kann aus dem gemessenen Druck direkt die Flughöhe über dem Meeresspiegel ermittelt werden. Diese Umrechnung wird bei den typischen Microchips intern erledigt und nur die berechnete Höhe ausgegeben, weshalb auch die Bezeichnung Altimeter üblich ist. Auf der APM-Platine verbaut ist ein Sensor der Firma MEAS, Modell MS5611 mit einer Messgenauigkeit von maximal  $\pm 2,5\text{mbar}$  bei normalen Temperaturbedingungen im Bereich von  $-20^\circ\text{C}$  bis  $+85^\circ\text{C}$ . [22]

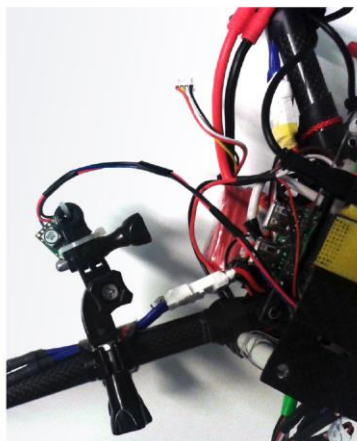
Eine geringe Fehlertoleranz für die Messung der Druckveränderungen bedingt sich, da die zurückgelegten Höhenunterschiede beim Einsatz des NeuroCopters sehr gering sind. Ein Nachteil des Messverfahrens liegt in der Anfälligkeit gegenüber Druckschwankungen durch Temperaturveränderung oder Luftzug. Erstere können die Messung beispielsweise bei direkter Sonneneinstrahlung negativ beeinflussen, wobei Letzterer durch Luftverwirbelungen der Propeller oder schnelle Bewegungen verursacht wird. Um beide Effekte zu minimieren wurde ein Wattepolster mit handelsüblichem Klebeband unmittelbar über dem Sensor angebracht und der Flugcontroller vor Sonneneinstrahlung geschützt.

### 3.2.4 Ultraschallsensor (Sonar)

Ergänzt wird die Höhenmessung durch einen senkrecht nach unten gerichteten Ultraschallsensor. Im Kontrast zum Barometer misst das Sonar statt der absoluten Höhe den lokalen Abstand zum Boden. Die Funktionsweise beruht auf der Laufzeitmessung zwischen Aussenden und Empfangen eines hochfrequenten akustischen Signals, das sich kegelförmig ausbreitet. Der verwendete MB1240 XL Sensor der Firma MaxBotix (siehe Abbildung 7 links) erlaubt eine auf wenige Zentimeter genaue Bestimmung von Abständen im Bereich von 0,2-7 Meter [23]. Für die entwickelte Positionsbestimmung und die Berechnung der Eigenbewegung aus dem optischen Fluss spielt die Distanz zum Boden eine entscheidende Rolle.

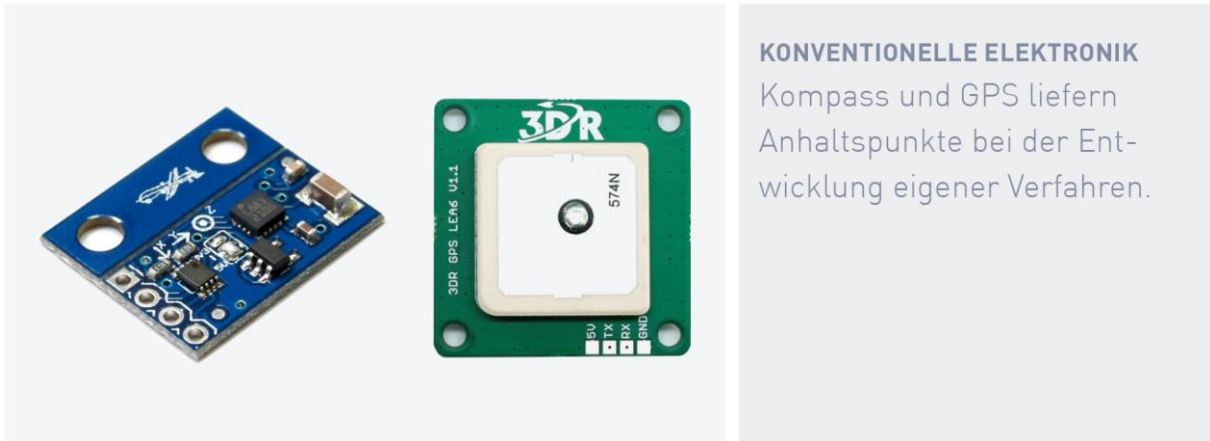
Durch ihre akustische Funktionsweise sind Ultraschallsensoren empfindlich gegenüber hohen Lärmpegeln, beispielsweise durch rotierende Propeller. Bei analoger Anbindung ist die vom Sensor ausgegebene Spannung proportional zu den Messwerten, Sensor und Kabel sollten also weitgehend frei von elektronischen Interferenzen durch sonstige Verkabelung und Akku montiert sein. Deshalb zu sehen wurde der Sensor und Verkabelung mit Abstand zur sonstigen Elektronik an einer Halterung montiert. (Abbildung 7 rechts)

ABBILDUNG 7: SONAR MB1240 UND SONARHALTERUNG



#### HÖHENMESSUNG PER ULTRASCHALL

Auf wenige Zentimeter genau wird der lokale Abstand zum Boden gemessen.



**KONVENTIONELLE ELEKTRONIK**  
 Kompass und GPS liefern Anhaltspunkte bei der Entwicklung eigener Verfahren.

### 3.2.5 3-Achsen-Magnetometer (Kompass)

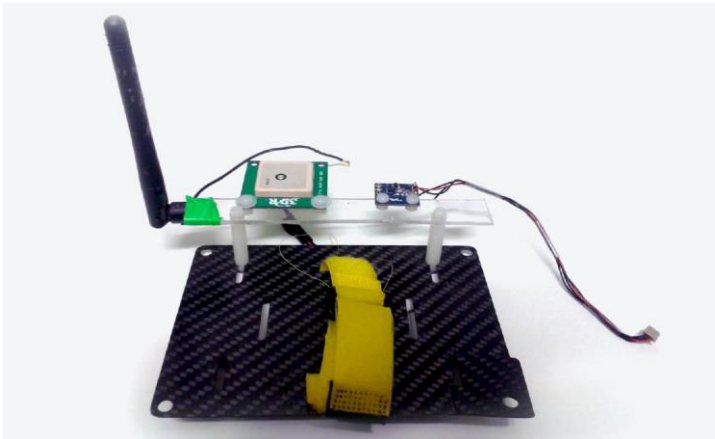
Die Aufgabe des Magnetometers ist es, Stärke und Richtung des Erdmagnetfeldes zu messen. Durch die Erfassung des Magnetfeldes entlang seiner drei Achsen ist der Sensor theoretisch imstande, seine Ausrichtung in drei Raumdimensionen zu bestimmen. Innerhalb der APM-Firmware wird der Sensor als digitaler 2D-Kompass verwendet. Um die Himmelsrichtung anhand des Magnetfeldes korrekt anzuzeigen, muss der Sensor softwareseitig mit der ortsabhängigen magnetischen Deklination korrigiert werden.

Der Einsatz eines Magnetometers auf einem Multicopter funktioniert jedoch nicht problemlos. Durch metallische Bauteile im Rahmen, sowie durch die starken erzeugten Magnetfelder in der Nähe der Motoren, ihrer Verkabelung und bei der Stromversorgung können die Messung des Erdmagnetfeldes stark verrauschen oder komplett verfälschen. Dieser Umstand äußerte sich erneut auffällig im vollständig computergesteuerten "Loiter"-Flugmodus (vgl. 3.2.2 oben über Auswirkungen des Accelerometers). Der Autopilot registrierte eine Driftbewegung, die eingeleiteten Korrekturmanöver wichen aber konsequent um mehrere Grad von der intendierten Richtung ab. Der Copter beginnt sich auf einer stetig enger werdenden Kreisbahn um die Ausgangsposition zu bewegen, als Resultat schaukeln sich registrierte Abweichung und Stärke der Manöver mit jeder versuchten Korrektur weiter auf (sog. "Toilet Bowling").

Um Störungen im Betrieb zu minimieren wurde das ursprüngliche Magnetometer auf der APM-Platine deaktiviert und durch ein externes Modul HMC5583L der Firma Honeywell [24] ersetzt (siehe Abbildung 8 links).

Zur exponierten Montage des Sensors wurde der Prototyp einer Halterung entwickelt. In Abbildung 9 ist die Karbondeckplatte des zentralen Elektronikkäfigs mit dem entwickelten Prototyp zu sehen (siehe auch Abbildung 3 für eine montierte Ansicht). Kunststoff wurde hierbei bewusst als Material für Querstrebe und Stützen ausgewählt, um magnetische Interferenzen zu vermeiden. Unter Verwendung des so montierten externen Kompassmoduls trat der beschriebene Effekt im "Loiter"-Modus nicht mehr wahrnehmbar auf, der Prototyp erfüllte somit seine Aufgabe und kann als Vorlage für eine zukünftige Anfertigung dienen.

\* Die Bezeichnung stammt von der spiralförmigen Abwärts-Flussbewegung des Wassers in der Toilettenschüssel



#### ABSTAND ZU STÖRFELDERN

Einige Sensoren profitieren von der Montage abseits der sonstigen Elektronik.

### 3.2.6 GPS

Für den konventionellen Betrieb wurde der NeuroCopter ebenfalls mit einem LEA6-H GPS-Modul der Firma U-Blox [25] ausgestattet. (Abbildung 8 rechts) Neben der Möglichkeit, automatisierte Flugrouten über GPS-Wegpunkte vorzudefinieren, greift die Firmware des Flugcontrollers auch für weitere automatisierte Flugmodi auf die Positionsbestimmung durch GPS zurück. [26] Die Positionsmessungen des Sensors bilden für sich bereits ein Navigationssystem (siehe 4.1.2) und dienen bei den Experimenten mit alternativen Navigationsstrategien als Vergleichsbasis.

Für eine korrekte Funktionsfähigkeit benötigt der Sensor ständigen Kontakt zu den GPS-Satelliten und sollte weitgehend unbehindertem Sichtkontakt haben. Er wurde deshalb ebenfalls außen an der Kunststoffbank montiert (Abbildung 9).

### 3.2.7 Kamerasystem für optischen Fluss

In Kapitel 1.1 wurde bereits die Rolle des optischen Flusses bei der Navigation der Honigbiene angedeutet, weshalb das NeuroCopter-System kürzlich durch eine PX4Flow [27] Smart Kamera ergänzt wurde (siehe Abbildung 10 rechts).

Bei PX4Flow handelt es sich um ein System, das speziell für die Bestimmung des Optischen Flusses entwickelt wurde. Auf der Platine befindet sich außer der hochfrequenten Kamera ein eigener Mikroprozessor zur Berechnung des optischen Flusses. Er greift dabei auf eine Kombination aus Lage- und Ultraschallsensor zurück, welche die notwendigen Parameter wie Bodenabstand und Neigung liefern. Mit der nach unten gerichteten Kamera wird während des Flugs eine Bildsequenz aufgenommen und aus der Verschiebung korrespondierender Bildpunkte anschließend die Eigenbewegung berechnet. Eine ausführlichere Beschreibung der Funktionsweise des optischen Flusses als Navigationssystem erfolgt in Kapitel 4.3.2.

### 3.2.8 Kamera für Landmarkenerkennung

Neben der Spezialkamera befindet sich an der Unterseite des Copters eine weitere Kamera mit anpassbarer Ausrichtung. Sie soll während des Fluges visuelle Informationen über die Umgebung zu sammeln und dient damit unter anderem als Sensor zur Erkennung von Landmarken für das im Rahmen dieser Abschlussarbeit entwickelte Verfahren. Die verwendete Kamera ist eine leicht modifizierte PS3Eye-Kamera von Sony, welche auf dem OV7221 Sensor der Firma Omnivision [28] basiert. (Abbildung 10 links)

ABBILDUNG 10: SONY PS3EYE UND PX4FLOW SMART KAMERA



Die PS3Eye eignet sich aufgrund ihrer preisgünstigen Anschaffung und gleichzeitig guten Performance von bis zu 60 Bildern/Sekunde bei einer Auflösung von 640x480. Sie lässt sich außerdem leicht mit unterschiedlichen Linsen modifizieren.

### 3.3 Flugsteuerung

Motorik und Sensorik werden in einer zweistufigen Steuerarchitektur zusammengeführt. Abbildung 1 weiter oben zeigt schematisch das Zusammenwirken der Steuerung mit den anderen Schichten.

#### 3.3.1 Low-Level Flugsteuerung (ArduPilot Mega 2.5)

Ein Kriterium bei der Auswahl des Flugcontrollers war die Verfügbarkeit einer Open-Source-Firmware, die grundlegende Funktionalität bereits bereitstellt, aber trotzdem für unsere Bedürfnisse angepasst werden kann. Im Gegensatz zu kommerziellen Lösungen ist diese auch deutlich preiswerter. Die Entscheidung fiel zugunsten eines Arduino Microcontroller-Board namens ArduPilotMega (APM) 2.5 aus, das von der Open Source Community speziell zur Robotersteuerung entworfen und von der Firma 3DRobotics gefertigt wird. (Abbildung 11 rechts)

Im Kern der APM Hardware arbeitet ein ATmel ATmega 2560 8-bit AVR-Mikroprozessor mit 16MHz Taktung und 256 Kbytes internem Flashspeicher. Bereits im Auslieferungszustand bietet der Flugcontroller außerdem drei der in Kapitel 3.2 erwähnten Sensoren: Eine 6-Achsen Gyroskop/Accelerometer, Altimeter und ein 3-Achsen Magnetometer\*. Auf der Platine befinden sich Multifunktions-Pins für Input/Output und Buchsen zur Anbindung externer Sensoren wie GPS, Sonar oder Kompass. Mit Hilfe eines ATmega32U-2 wird eine serielle Verbindung über USB bereitgestellt, über die das Board mit dem Onboard-PC verbunden ist.

Auf dem Board läuft die Open-Source ArduCopter-Firmware zur Steuerung von Multicoptern. [29] Die Firmware benutzt das MAVLink-Protokoll der ETH Zürich zur Kommunikation mit Bodenstation bzw. der High-Level-Steuerung. [30]

#### 3.3.2 Konventionelle High-Level Flugsteuerung (IGEP)

Über die serielle USB-Schnittstelle angebunden an den APM-Flugcontroller ist ein IGEPv2 Prozessor-Board der Firma ISEE. (Abbildung 11 links)

---

\* Das mitgelieferte Magnetometer wurde innerhalb dieser Arbeit durch ein externes Magnetometer ersetzt (siehe Kapitel 0)

ABBILDUNG 11: ONBOARD-PC UND FLUGCONTROLLER



#### WOHIN UND WIE

Onboard-PC und Flugcontroller sorgen für die Umsetzung der Routenbefehle in eine Bewegung.

Durch die geringen Abmessungen von 93x64mm, dem kleinen Gewicht von ca. 50 Gramm und seiner niedrigen Leistungsaufnahme von unter 7 Watt eignet sich das Board für den Einsatz als Multifunktions-PC "onboard" des Quadcopters. Er ist ausgestattet mit einer ARM Cortex A8 CPU mit 1GHz, 512MB RAM und einer Vielzahl an Schnittstellen (WLAN, Ethernet, USB, Audio, SD-Karten Leser). [31] Eine angepasste Ubuntu Linux-Distribution der BerlinUnited-Arbeitsgruppe wird als Betriebssystem verwendet.

Neben der konventionellen Berechnung einzelner Aufgaben wie die Vorverarbeitung der Kamerainformationen zur optischen Lokalisierung besteht die Hauptaufgabe aus dem Management der Kommunikation zwischen APM und den externen Teilen der High-Level-Steuerung, wie dem Neuronalen Netzwerk.

### 3.3.3 Neuronale Steuerarchitektur \*

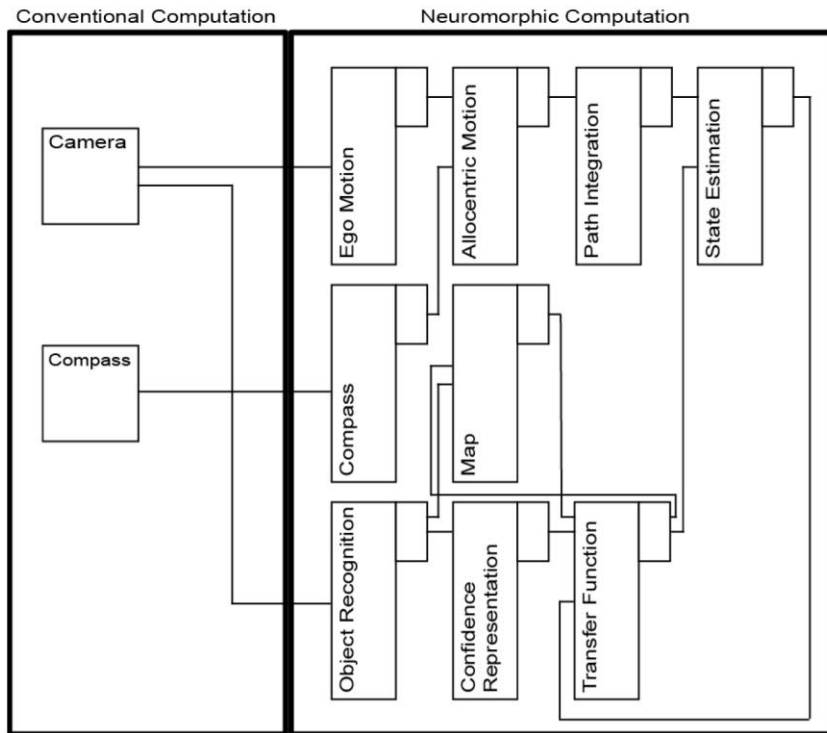
Wie am Ende von Kapitel 2 oben bereits erwähnt liegt die besondere Ausrichtung des NeuroCopters in dem Neuronalen Programmierparadigma. Die neuronale High-Level-Steuerarchitektur entwickelt sich dabei aus der Erlernung spezifischer Aufgaben, wie der Berechnung der Eigenbewegung aus dem Optischen Fluss oder der Positionsbestimmung anhand von Landmarken. Jede Aufgabe lässt sich als vernetzter Prozess unterschiedlicher kognitiver Funktionen darstellen, die aufeinander folgen und miteinander interagieren. Diese werden wiederum innerhalb einer programmierten neuronalen "Black Box" verarbeitet.

Das Schema in Abbildung 12 zeigt am Beispiel der Berechnung des Optischen Flusses und der optischen Positionsbestimmung über Landmarken wie sich komplexe Aufgaben als verschachtelte Verarbeitungsschritte einzelner kognitiver Funktionen interpretieren lassen. Erkennbar ist die komplementäre Wirkung der beiden Verfahren innerhalb eines Lokalisierungs- und Mappingprozesses. Eingaben werden von linksstehenden Modulen geliefert, anschließend verarbeitet und mittels Ausgabe-Neuronen (kleine Boxen) nach rechts weitergeleitet. Die Boxen auf der rechten Seite symbolisieren Neuro-Module, welche jeweils die einzelnen Verarbeitungsschritte repräsentieren und intern eine beliebige Struktur haben. In der Abbildung hervorgehoben ist weiterhin die Schnittstelle zwischen konventioneller und neuromorpher Berechnung, wie sie im Einsatz der Steuerarchitektur denkbar wäre.

---

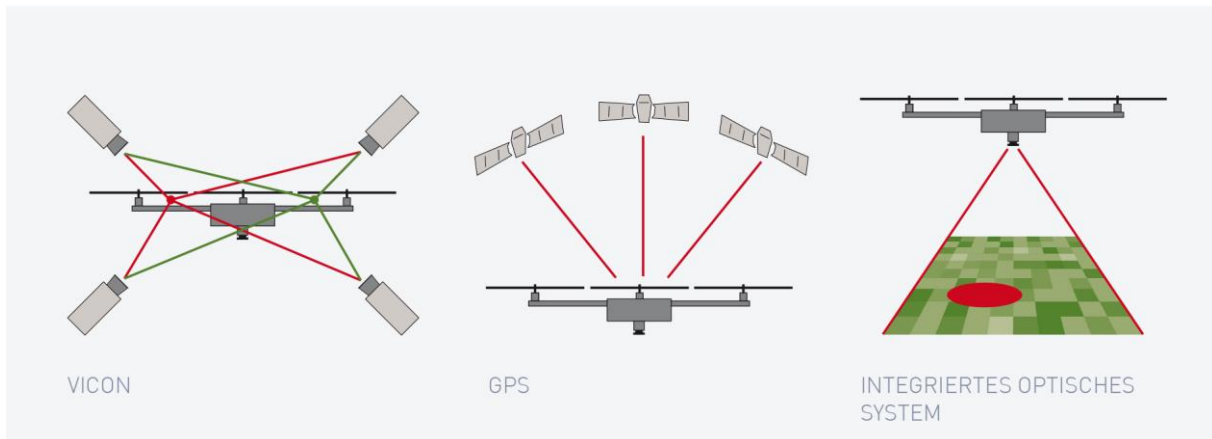
\* Die Neuronale Steuerung steht in einer frühen Phase der Entwicklung und kann sich dementsprechend ändern. Das Konzept bezieht sich auf den beschriebenen Entwurf aus [32].

ABBILDUNG 12: KONVENTIONELL-NEUROMORPHE STEUERARCHITEKTUR



Das Modul oben links ("Ego Motion") erhält aus dem Kamerabild extrahierte Daten über den optischen Fluss und ermittelt daraus die Eigenbewegung des Copters. Der berechnete Bewegungsvektor wird im nächsten Schritt zusammen mit der Orientierung aus dem Kompassmodul dazu verwendet die Bewegungsvektoren aus dem ego- in ein allozentrisches Koordinatensystem zu überführen. Per Integration über die zurückgelegten Pfade erhalten wir damit eine Schätzung über die aktuelle Position des Roboters. Aus diesen Schätzungen werden in einem weiteren (neuronalen) Modul unterschiedliche Positionshypothesen erstellt und verfolgt.

Wird eine Landmarke im Kamerabild entdeckt ermittelt zunächst das "Object Recognition"-Modul ihre Identität und überträgt diese zusammen mit ihrer Position an eine zentrale Karte. Die Karte erhält außerdem Eingaben aus dem Transfermodul, zuständig für die Integration und Verteilung der Informationen verschiedener anderer Module. Das Konfidenz-Modul weist daraufhin den auf der Karte gespeicherten Landmarkenpositionen ein Vertrauensmaß zu. Ein niedriger Wert führt dazu, dass die geschätzte Position des Copters als Ausgangswert auf die Position der Landmarke übertragen wird. Ist die wahrgenommene Landmarke bereits ausreichend vertraut und bestimmt, führt ein hoher Konfidenzwert im Umkehrschluss dazu, dass der Copter seine Position anhand der Landmarke aktualisiert.



## 4 Systeme zur Positionsbestimmung (Navigation)

Damit der Flugroboter, vergleichbar einer Biene, autonom in einem unbekanntem Umfeld navigieren, Nahrungsquellen anfliegen und die aktuelle Position über einen längeren Zeitraum beibehalten kann, muss er seine eigene Position möglichst exakt bestimmen und regeln können. Viele der herkömmlichen Lokalisierungsverfahren aus der Robotik verlangen eine besonders strukturierte Umgebung oder zusätzliche Vorkenntnisse über diese (siehe Kapitel 2). Einige dieser Systeme scheitern zudem an den besonderen Voraussetzungen für den Einsatz auf MAVs. Die begrenzte Tragkraft bedingt Einschränkungen für das Gesamtgewicht und den Energieverbrauch oder der zusätzliche Grad der Bewegungsfreiheit eine zu hohe Komplexität des zu ermittelnden Zustandes. In der Praxis werden deshalb die Einschränkungen der unterschiedlichen Sensorsysteme durch Kombination bzw. Fusion kompensiert.

In diesem Kapitel werden kurz Grundlagen über die verschiedenen Navigationssysteme vermittelt und ihre Anwendbarkeit bezüglich der Projektanforderungen bewertet. Allgemein wird hierbei zwischen externen und direkt auf der Drohne integrierten Systemen unterschieden. Im Anschluss wird unser Ansatz basierend auf zwei optischen Verfahren zur Umsetzung eines integrierten Systems dargestellt.

### 4.1 Externe Verfahren

Bei vielen bereits existierenden Systemen erfolgt die Positionsbestimmung des Roboters mit Hilfe externer Sensorik, beispielsweise einem optischen Tracking System oder GPS. Zwei populäre Verfahren werden im Folgenden repräsentativ vorgestellt.

#### 4.1.1 VICON / Motion Capturing

Das erste vorgestellte System hat sich in anderen Anwendungsbereichen jenseits der Robotik unter der Bezeichnung "Motion Capturing" bereits etabliert. Bei diesem Verfahren wird das zu beobachtende Objekt mit eindeutigen optischen Features (Markern) versehen und von mehreren räumlich verteilten, externen Kameras erfasst. Die gesammelten perspektivischen Daten über Bewegung und Position der Marker werden in einem zentralen Rechner zusammengeführt und ermöglichen so eine dreidimensionale Lokalisierung. (Abbildung 13 links)

Im Bezug auf UAVs hat dieses Verfahren mehrere Anwendungsmöglichkeiten. Neben der Positionsbestimmung liefert es genaue Daten über die Ausrichtung, ermöglicht zusätzlich eine Verfolgung der Bewegungstrajektorie oder Hindernis- bzw. Objekterkennung. Da das System aus

externen Beobachtern jedoch den vollständigen Bewegungsraum erfassen muss, werden vergleichbare Systeme beinahe ausschließlich im Innenbereich eingesetzt. Sie eignen sich somit nicht für einen autonomen Quadrocopter, der sich selbstständig in unbekanntem Terrain lokalisieren soll.

#### **4.1.2 Global Positioning System (GPS)**

Tragbare Computer, wie auch ein Großteil autonomer Roboter, vertrauen auf zur Ermittlung der Position. Ein spezieller GPS-Empfänger bekommt aus verschiedenen Richtungen Radiosignale von mehreren Satelliten, welche die Erde umkreisen und fortlaufend ihre aktuelle Position und exakte Uhrzeit ausstrahlen. (Abbildung 13 Mitte)

Anhand der unterschiedlichen Signallaufzeiten kann der Abstand zu dem jeweiligen Satellit bestimmt und folglich die eigene Position berechnet werden. Theoretisch reichen zur Bestimmung der genauen Position und Höhe die Signale von drei Satelliten aus. Da aber in der Praxis die interne Uhr der GPS-Empfänger nicht exakt genug ist, um die Laufzeiten korrekt messen zu können, wird das Signal eines vierten Satelliten verwendet. Über die Messung des sog. Dopplereffekts oder mittels Differenzierung der Positionsänderung abhängig von der Zeit lassen sich neben Position auch Bewegungsgeschwindigkeiten am Empfänger ermitteln. Damit diese Art der Lokalisierung überhaupt funktioniert, ist es notwendig, ausreichend guten Kontakt zu den GPS-Satelliten herzustellen. Dies ist bei Flügen innerhalb von Gebäuden aber auch in stark bebauten Umgebungen oder bei schlechtem Wetter nicht immer gewährleistet. Eigene Versuche haben außerdem gezeigt, dass die ermittelte Position auch bei gutem Empfang mit mehr als fünf Satelliten einer relativ großen Ungenauigkeit unterliegt.

Im Kontext der Erforschung des Navigationsverhaltens der Honigbiene innerhalb des Projekts NeuroCopter wird GPS eine ergänzende Rolle als Referenz übernehmen.

### **4.2 Integrierte Verfahren**

Systeme wie VICON und GPS erfüllen nur den Anspruch der Autonomie bezüglich Unabhängigkeit von einer menschlichen Kontrollinstanz, sind aber angewiesen auf eine ständige Verbindung zu externen Informationsquellen. Integrierte Navigationsverfahren ermöglichen es dem Roboter, sich unabhängig von den Umgebungsvoraussetzungen zu lokalisieren und erlauben insofern vollständig autonomen Flug.

#### **4.2.1 Koppelnavigation**

Ein herkömmlicher Ansatz zur Realisierung eines solchen Systems ist die sog. Koppelnavigation (engl. Dead Reckoning) mit Hilfe von Odometrie-Daten. Ein derartiges Verfahren ist in diesem Fall schwer realisierbar, da die Motoren kein Feedback zur Drehzahl liefern. Im Gegensatz zu bodengebundenen Robotern können Fluggeräte zudem aufgrund der inkonsistenten Koppelung von Motordrehzahl zu tatsächlich erfolgtem Vorschub nur unzureichende Rückschlüsse über die zurückgelegte Distanz machen.

#### **4.2.2 Inertiale Navigation (INS)**

Auf einer inertialen Messeinheit basierende Navigationssysteme arbeiten ebenfalls ohne externe Ortungssignale aus der Umgebung. Innerhalb der Messeinheit wird mit Hilfe des Gyroskops (Drehratensensors) die Winkelgeschwindigkeit bei Rotationsbewegungen, mit dem Accelerometer (Beschleunigungssensor) die Beschleunigung jeweils in den drei orthogonalen Raumrichtungen gemessen. Durch die komplementäre Nutzung beider Sensoren kann mittels Integration über die Zeit die Bewegung und Lage in allen sechs Freiheitsgraden eines frei beweglichen Körpers erfasst werden.



Ein Nachteil bei günstigen Sensoren ist der vorhandene Sensordrift, welcher sich als Fehler auf die Messung auswirkt. [33] Da das INS die gemessenen Veränderungen in seiner Position immer zu der letzten Schätzung addiert häuft sich der Fehler kumulativ an und wirkt sich durch die zeitliche Integration über die Bewegung verstärkt aus. Der Einsatz des Multicopters in natürlicher Umgebung bedingt zusätzlich störende Einflüsse (z.B. durch Wind, Vibrationen, ...) auf die Messgenauigkeit und damit die Zuverlässigkeit der Positionsermittlung. In der Praxis wird deshalb die Positionsschätzung basierend auf inertialen Messungen häufig mit anderen Sensoren korrigiert. In einer typischen Kombination liefert der GPS-Empfänger absolute Positionsangaben im Sekundenabstand während das INS Zwischenwerte in den GPS-Messintervallen interpoliert oder für Ausfälle kompensiert.

## 4.3 Optische Navigationsverfahren

Ähnlich wie der Mensch benutzen auch Honigbienen ihre Augen als wichtiges Organ zur Wahrnehmung der Umwelt und zur Orientierung. Im Kontext der Biorobotik ist es also naheliegend mit einer Kamera als sensorischem Äquivalent zu arbeiten. (siehe Abbildung 13 rechts)

### 4.3.1 Kamera als Sensor

Kameras sind in vielerlei Hinsicht optimale Sensoren für den Einsatz auf MAVs: Ihr Gewicht ist gering und der Platzbedarf klein, sie haben eine niedrige Leistungsaufnahme und kosten verhältnismäßig wenig Geld. Gleichzeitig liefern sie ein breites Spektrum an Informationen, da potentiell jeder Pixel unabhängige Daten über die Umwelt enthält. Aus einem einzelnen Kamerabild lassen sich viele Informationen extrahieren. Farbe, Form oder Oberflächenstruktur von Objekten ermöglichen es, Orientierungspunkte anhand vielfältiger Charakteristika zu spezifizieren und identifizieren. Kameras haben eine theoretisch unbegrenzte Reichweite und erlauben mit ihrer hohen Auflösung äußerst genaue Winkelmessungen im Bild. Da sie ein projektiver Sensor sind, enthalten die Bilder zunächst keine Tiefeninformationen, aus einer Sequenz von mehreren Bildern kann aber anhand der relativen Verschiebung von korrespondierenden Features die Tiefe ermittelt und die eigene Bewegung berechnet werden. Im Gegensatz zur Verwendung des Gyroskops eines INS unterliegen optische Sensoren keinem Sensordrift und sind zudem nicht auf externe Hilfsmittel angewiesen, was einen Einsatz in unbekanntem Umgebungen ermöglicht. In der Praxis bietet sich der parallele Einsatz mehrerer Kameras an, die durch unterschiedliche Ausrichtung ein größeres Sichtfeld erfassen und durch Redundanzen die Robustheit der Erfassung erhöhen. Probleme können sich durch die Bewegungsunschärfe oder durch die Verschiebung zwischen fester Framerate und variablem Fluss im Kamerabild abhängig von der Distanz zu den erfassten Objekten ergeben.

### 4.3.2 Egomotion via Optischem Fluss

Durch eine zum Boden gerichtete Kamera wird während des Fluges eine Bildsequenz aufgenommen. Aus der Verschiebung korrelierender Bildpunkte innerhalb dieser Bildsequenz kann die entsprechende Eigenbewegung berechnet und die Positionsänderung approximiert werden. Ist die Anfangsposition bekannt, verfügt man somit über eine ständige Positionsschätzung. Durch seine autonome Arbeitsweise ist ein derartiges System sowohl für den Einsatz innerhalb, als auch außerhalb eines Gebäudes geeignet. Aufgrund der Tatsache, dass die aktuelle Schätzung immer ausgehend zur vorherigen Position über Pfadintegration bestimmt wird ist das System anfällig für Fehler die sich aufsummieren. [34] [32]

### 4.3.3 Positionsbestimmung anhand von optischen Merkmalen

Ein wichtiger Beitrag zur Robustheit der Navigation leistet die Fähigkeit über einen längeren Zeitraum hinweg eine kartenähnliche Repräsentation der Umgebung zu entwickeln. Zur Navigation merkt man

sich dazu visuelle Informationen über herausragende Objekte (sog. Landmarken) an bereits besuchten Orten.

Der hier verfolgte Ansatz sieht folgenden Ablauf vor: Wird eine Landmarke zum ersten Mal entdeckt werden charakteristische optische Anhaltspunkte gespeichert und sie bekommt eine allozentrische Position ausgehend von der eigenen Positionsschätzung zugewiesen. Wenn in den Bereich einer vertrauten Landmarke zurückkehrt wird, vergleicht man die gespeicherten Informationen mit der aktuell wahrgenommenen Umgebung und sucht nach Übereinstimmungen. Wird sie wiedererkannt, kann somit mit jedem erneuten Aufsuchen die Schätzung für diese bekannte Landmarke durch diesen iterativen Prozess verfeinert werden. Daraus resultiert eine höhere Zuverlässigkeit bei der Ermittlung der eigenen Position relativ zu bekannten Orientierungspunkten. Da die Existenz auffälliger Orientierungspunkte im nahen Umfeld Voraussetzung für die Funktionalität dieser Navigationsstrategie ist ergänzt sie sich z.B. mit der Positionsbestimmung aus Optischem Fluss. Letztere stellt keine strukturellen Anforderungen an die Umwelt, weist aber gerade über längere Distanzen einen größer werdenden Fehler auf.

## 5 Kamerabasierte Positionsbestimmung anhand kodierter Landmarken

Im Rahmen dieser Diplomarbeit wurde ein Verfahren entwickelt und implementiert welches unter Verwendung der Lageinformationen des Copters eine relative Positionsbestimmung durch die Erkennung künstlicher Landmarken im Kamerabild erlaubt. Die konventionelle Umsetzung des verwendeten Ansatzes soll hierbei einen Referenzwert für zukünftig entwickelte, neuronale Methoden bieten.

Wie bei der Neuronalen Steuerarchitektur in Kapitel 3.3.3 erwähnt, ist die eigene Positionsbestimmung anhand erkannter Objekte ein iterativer Prozess. Nähert sich der Copter wiederholt einer bekannten Landmarke, kann sowohl die Positionsschätzung der Landmarke innerhalb der Karte, als auch die eigene verbessert werden. Das entwickelte Verfahren zur relativen Positionsbestimmung eignet sich gleichzeitig dazu anhand der eigenen Positionsschätzung eine Hypothese über die Position einer neu entdeckten Landmarke zu entwickeln.

In diesem Kapitel werden zunächst die Anforderungen für die Software und das Lokalisierungsverfahren aufgezeigt, gefolgt von einer Erklärung der Grundlagen des Verfahrens, der verwendeten Konventionen und Terminologie. Unter Abschnitt 5.3 wird der Ablauf der Positionsbestimmung mathematisch und umgangssprachlich in Einzelschritten nachvollzogen. Anschließend folgen Details über die technische Implementierung der Software, inklusive einer Vorstellung der verwendeten Entwicklungsumgebung, Algorithmen und Bibliotheken. Der technische Ablauf des implementierten Verfahrens wird abschließend in einzelnen Verarbeitungsschritten dargestellt und der Einsatz der Software beispielhaft erläutert.

### 5.1 Spezifikation der Anforderungen

Zu Beginn wurden die Anforderungen erhoben um Voraussetzungen an die Funktionalität der Software zu spezifizieren.

Ausgehend von der biologischen Motivation (Kapitel 1.1) wurden verschiedene Anforderungen an die Software gestellt: Um Hypothesen über die Navigationsfähigkeiten der Honigbiene erforschen zu können, ist es von Vorteil, wenn die Verfahren möglichst nah am natürlichen Vorbild sind. Eine mögliche Strategie sieht vor, den Ausflug zur Nahrungsquelle mit einer Orientierung im Umfeld mittels Kompass und Integration der Eigenbewegung aus dem Optischen Fluss (siehe Kapitel 4.3.2) zu bewerkstelligen, um anschließend die Lokalisierung bei Annäherung an eine bekannte Landmarke zu optimieren bzw. Fehler zu korrigieren. Die entwickelte verfeinerte Positionsbestimmung anhand optischer Landmarken bildet hierbei nur einen bestimmten Aspekt und muss demzufolge an den Kontext unterschiedlicher Strategien anpassbar sein.

Für die Entwicklung der Software gab es darüber hinaus weitere technische Rahmenbedingungen. Hinsichtlich der angedachten neuronalen Steuerarchitektur (Kapitel 3.2.2) galt es, bei der Implementierung die einzelnen Verarbeitungsschritte voneinander trennbar zu machen. Dies bildet die Grundlage, um beispielsweise die Extraktion der Landmarke aus dem Kamerabild mit traditionellen Verfahren zu berechnen und die Positionsermittlung anschließend innerhalb des Neuronalen Netzes durchzuführen.

Da die Positionsberechnung Informationen über die Orientierung der Kamera voraussetzt, wird sie unmittelbar von der Sensorik und damit dem aktuellen Zustand des Multicopters beeinflusst. Zugleich wirkt sie sich wiederum auf die weitere Navigation aus, weshalb es von großer Bedeutung ist, die

Informationen möglichst zeitnah zu verarbeiten. Die Anbindung an die Flugsteuerung und die notwendige Synchronisation der Sensormessungen konnte bis zum Abschluss der Arbeit leider nicht umgesetzt werden, es wird daher mit simulierten Daten gearbeitet.

## 5.2 Grundlagen des Verfahrens zur Positionsbestimmung

Zunächst sollen ein paar Grundlagen zum Verständnis des Verfahrens vermittelt werden. Da wir uns hier im Bereich der Luftfahrt befinden werden in der Erläuterung wie auch bei der Implementierung die übliche Terminologie und Konventionen verwendet. Abbildung 14 links zeigt den Verlauf der Rotationsachsen im Körpereigenen Koordinatensystem und rechts deren Interpretation in Weltkoordinaten. Beide Systeme folgen hierbei der Rechten-Hand-Regel.

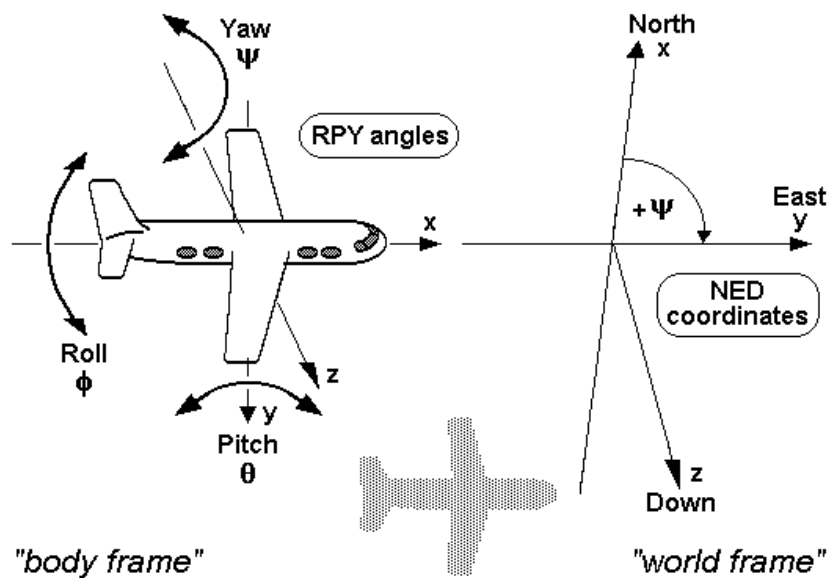


Abbildung 14: Winkelangaben und Rotationsachsen (links) bzw North-East-Down Konvention (rechts) im Bereich der Luftfahrt [Quelle: Wikipedia]

Die Orientierung eines Körpers kann vollständig durch Rotation um 3 orthogonale Achsen X/Y/Z beschrieben werden. Verwendet werden dabei folgende Bezeichnungen:

*Yaw*: Rotation mit dem Winkel  $\alpha$  im Uhrzeigersinn um die Z-Achse. Die entsprechende Rotationsmatrix sieht folgendermaßen aus:

$$R_z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Formel 1: 3D-Rotation um Z-Achse (Yaw)

*Pitch*: Rotation mit dem Winkel  $\beta$  im Uhrzeigersinn um die Y-Achse. Die entsprechende Rotationsmatrix sieht folgendermaßen aus:

$$R_y(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}$$

Formel 2: 3D-Rotation um Y-Achse (Pitch)

*Roll*: Rotation mit dem Winkel  $\gamma$  im Uhrzeigersinn um die X-Achse. Die entsprechende Rotationsmatrix sieht folgendermaßen aus:

$$R_x(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}$$

Formel 3: 3D-Rotation um X-Achse (Roll)

Durch Kombination dieser Rotationen kann ein Körper in eine beliebige Lage gebracht werden, die Gesamtrotation wird durch Multiplikation der einzelnen Matrizen ausgedrückt. Dabei ist es wichtig eine feste Reihenfolge festzulegen, da die Multiplikation von Matrizen nicht kommutativ ist und somit eine unterschiedliche Matrix resultieren würde.

Für eine Rotation um alle 3 Achsen wurde hier gemäß der Konvention die Reihenfolge 1. Yaw 2. Pitch 3. Roll gewählt. Die resultierende Matrix entspricht also:

$$R(\gamma, \beta, \alpha) = R_x(\gamma)R_y(\beta)R_z(\alpha)$$

Formel 4: Yaw-Pitch-Roll-Matrix

Durch die Matrix in Formel 4 kann statt der Rotation eines Körpers um die verschiedenen Koordinatenachsen auch die Rotation unterschiedlicher Koordinatensysteme zueinander ausgedrückt werden. Abbildung 15 zeigt den Zusammenhang zwischen einem Punkt  $y$  im ursprünglichen Koordinatensystem und dem Punkt  $y'$  im rotierten Koordinatensystem. Ebenfalls zeigt die Abbildung die Umkehrung der Rotation durch die inverse Rotationsmatrix  $R^{-1}$ .

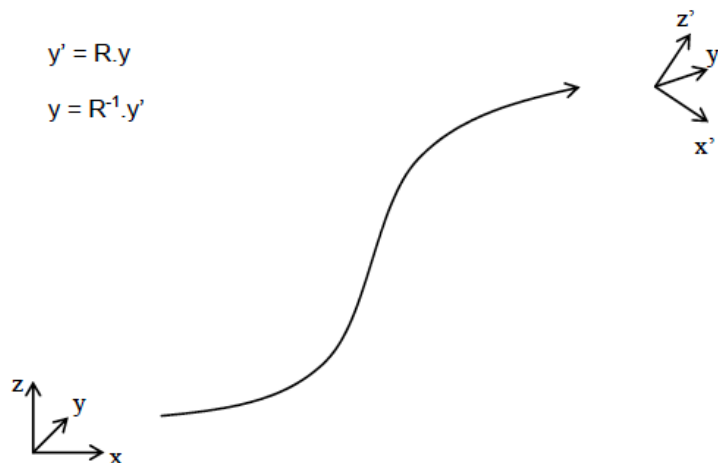


Abbildung 15: Rotation des Koordinatensystems mit Rotationsmatrix  $R$  [35]

Im weiteren Verlauf wird ein Einheitsvektor  $v_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$  aus seiner initialen Ausrichtung im Weltkoordinatensystem\* schrittweise durch drei aufeinanderfolgende Rotationen  ${}^C R_{LM}$ ,  ${}^B R_C$ ,  ${}^W R_B$  in Richtung der detektierten Landmarkenposition im Kamerabild rotiert. Die drei Rotationsmatrizen sind dabei:

${}^C R_{LM}$  die Rotation des Kamerakoordinatensystems in Richtung der Landmarke, Yaw-/Pitch-Winkel werden ermittelt aus der Position der Landmarke im Kamerabild.

${}^B R_C$  die Rotation des Körper ("Body")-Koordinatensystems zur Kamera, YPR-Winkel sind bestimmt durch die Befestigung der Kamera am Copter und damit statisch

\* Blick nach Norden entlang der positiven X-Achse, keine Rotation um eine der Achsen

${}^W R_B$  die Rotation des globalen ("Welt"-) Koordinatensystems zum Copter, Winkel werden bestimmt durch Kompass (Yaw) und IMU-Daten (Pitch + Roll)

Vor der Anwendung des eigentlichen Verfahrens muss außerdem einmalig der vertikale und horizontale Sichtbereich (FOV = Field of View) der Kamera und die vertikale und horizontale Bildauflösung bestimmt werden. Aus dem horizontalen Sichtbereich  $\varphi_{hor}$  in Grad und der horizontalen Bildauflösung  $R_{hor}$  in Pixel lässt sich dann ein Umrechnungsfaktor

$$c_{hor} = \frac{\varphi_{hor}}{R_{hor}}$$

**Formel 5: Horizontaler Umrechnungsfaktor in Grad/Pixel**

in Grad/Pixel berechnen. Analog ergibt sich der vertikale Umrechnungsfaktor:

$$c_{ver} = \frac{\varphi_{ver}}{R_{ver}}$$

**Formel 6: Vertikaler Umrechnungsfaktor in Grad/Pixel**

Innerhalb dieser Arbeit wird zur Vereinfachung dieses Verhältnis über die gesamte Bildfläche durch einen konstanten Faktor approximiert. Das ist jedoch insbesondere bei typischen sphärischen Linsen nicht exakt der Fall und berücksichtigt nicht eine eventuell intern vorgenommene Entzerrung durch die verwendete Kamera.

## 5.3 Ablauf der Positionsbestimmung

Der Ablauf des Algorithmus lässt sich in vier Verarbeitungsschritte aufteilen. Diese werden in den folgenden Unterpunkten erläutert.

### 5.3.1 Berechnung der Rotation

Wurde eine Landmarke im Kamerabild erkannt wird zunächst die Position ihres Mittelpunktes relativ zum Bildmittelpunkt in Pixeln ermittelt. Aus der Pixelposition können anschließend die Winkel zwischen der Ausrichtung der Kamera (dem Bildmittelpunkt) und Landmarke berechnet werden. Eine positive Pixeldistanz in horizontaler Richtung  $\Delta_{hor}$  im Kamerabild entspricht dabei einem positiven Rotationswinkel  $\alpha = \Delta_{hor} * c_{hor}$  um die Z-Achse des Kamerakoordinatensystems (Yaw). Ein positiver Abstand  $\Delta_{ver}$  in vertikaler Richtung der Rotation um den Winkel  $\beta = \Delta_{ver} * c_{ver}$  um die Y-Achse (Pitch). Beide Winkel werden über die Matrizen  $R_z(\alpha)$  und  $R_y(\beta)$  in der Matrix  ${}^C R_{LM}$  festgehalten.

Die ermittelte Matrix wird anschließend von rechts mit der statischen Rotationsmatrix  ${}^B R_C$  multipliziert. Somit erhalten wir eine neue Matrix:

$${}^B R_{LM} = {}^B R_C * {}^C R_{LM}$$

**Formel 7: Rotation der LM im Copter-Koordinatensystem**

Um nun die Rotation und damit die endgültige Richtung des Vektors bezüglich des Weltkoordinatensystems zu erhalten müssen wir im Anschluss analog zu Formel 7  ${}^B R_{LM}$  von rechts mit  ${}^W R_B$  multiplizieren um  ${}^W R_{LM}$  zu bekommen. Wie in Abschnitt 5.2 festgelegt, besteht  ${}^W R_B$  aus den Roll-/Pitch-/Yaw-Winkeln der IMU bzw. des Kompass und gibt die Lage des Copters im Welt-Referenzsystem an.

### 5.3.2 Rotation des Richtungsvektors

Abschließend wird die abschließende Rotationsmatrix  ${}^wR_{LM}$  mit  $v_0$  multipliziert, wir erhalten einen neuen Vektor:

$$v_{LM} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = {}^wR_{LM} * v_0$$

**Formel 8: Berechnung des neuen Richtungsvektors aus der Rotation des ursprünglichen Einheitsvektors**  
 $v_{LM}$  beschreibt nun die Richtung zur Landmarke im Weltkoordinatensystem. Durch Rotation ändert sich hierbei die Länge des Vektors nicht.

### 5.3.3 Ermittlung der Schnittebene und des Abstands

Um im nächsten Schritt die Position eindeutig bestimmen zu können ist es notwendig neben dem Ausgangspunkt und Richtungsvektor die zu schneidenden Ebene zu kennen. Da wir den vertikalen Abstand des Copters von der Bodenebene durch die Höhenmessung der Sensorik erhalten können, bestimmen wir die Schnittebene als die parallel zur Bodenebene verlaufende Ebene auf der für alle Punkte der Ebene gilt  $z = \text{Flughöhe des Copters}$ . Die im nächsten Schritt berechnete Position des Copters C ist also ein Punkt dieser Ebene.

Gegeben der vertikale Abstand  $\delta_z = z_{\text{Copter}} - z_{\text{Landmarke}}$  zwischen Copter und Landmarkenposition L lässt sich nun ein Skalar  $t$  aus der Z-Komponente  $z_v$  des Richtungsvektors  $v_{LM}$  berechnen:

$$t * z_v = \delta_z \text{ und somit für } z_v \neq 0^*$$

$$t = \frac{\delta_z}{z_v}$$

**Formel 9: Berechnung des Skalarfaktors  $t$  aus der vertikalen Distanz und dem rotierten Richtungsvektor**  
Der Skalarfaktor  $t$  beschreibt nun den euklidischen Abstand der der Copterposition C und der Landmarkenposition L im Weltkoordinatensystem.

### 5.3.4 Berechnung der dreidimensionale Position des Copters<sup>†</sup>

Um die Position C des Multicopters anhand der bekannten Landmarkenposition L zu berechnen, müssen wir zunächst den Richtungsvektor  $v_{LM}$  mit dem Skalar  $t$  multiplizieren:

$$v_D = t * v_{LM}$$

**Formel 10: Der skalierte Richtungsvektor**

Wie in Unterpunkt 3) erwähnt beschreibt  $t$  den dreidimensionalen euklidischen Abstand,  $v_{LM}$  gibt die Richtung und die Länge ( $\|v_{LM}\| = 1$ ) vor. Man kann sich die vorgenommene Multiplikation also bildlich als "gehe  $t$  Schritte der Länge  $\|v_{LM}\|$  in Richtung des Vektors  $v_{LM}$ " vorstellen.

X, Y und Z-Komponente von  $v_D$  entsprechen nun dem Abstand im Weltkoordinatensystem in allen drei Raumdimensionen des Copterposition C von der Landmarkenposition L. Die finale Bestimmung von C anhand von L ist dann nur noch eine Addition des inversen Vektors  $v_D$  auf L.

---

\* Ist  $z_v = 0$  ist  $v_{LM}$  parallel zur X-/Y-Ebene. Damit muss auch  $\delta_z = 0$  sein, Copter und Landmarke befinden sich also exakt auf einer Höhe. In diesem Fall lässt sich der Schnittpunkt nicht eindeutig bestimmen. Dieser Spezialfall wird im Algorithmus gesondert behandelt.

† Für eine Berechnung der Landmarkenposition ausgehend von einer bekannten Copterposition muss der Richtungsvektor  $v_D$  in Formel 11 nicht invertiert werden.

$$C = L + (-1 * v_D)$$

Formel 11: Die berechnete Copterposition C aus Landmarkenposition L und dem Abstandsvektor  $v_D$

## 5.4 Implementierung

Nach Grundlagen und Ablauf soll in diesem Kapitel auf die technische Implementierung und die notwendigen Bildverarbeitungsschritte eingegangen werden.

### 5.4.1 Entwicklungsumgebung / Bibliotheken

Die vorliegende Software wurde in der Programmiersprache C++ geschrieben. Als Entwicklungsumgebung wurde die Eclipse CDT IDE in Version 3.7.2 verwendet und auf einem Apple MacBook mit Intel Core 2 Duo (2.4 GHz), 4GB Arbeitsspeicher und Mac OS X 10.7 programmiert und debugged. Die Funktionsfähigkeit wurde mit verschiedenen Kameras auf dem angegebenen System überprüft.

Für die Bildverarbeitung dienten die frei verfügbare OpenCV Bibliothek (Version 2.4.1) [36] erweitert durch die cvBlob Bibliothek (Version 0.10.4) [37], die zusammen bereits eine Vielzahl an Möglichkeiten für die Bildverarbeitung und die Erkennung von Zusammenhangskomponenten bereitstellt.

### 5.4.2 Feststellung der intrinsischen Kameraparameter / Kamerakalibration

Damit die Positionsbestimmung anhand der räumlichen Lage der Kamera erfolgen kann, müssen zunächst die intrinsischen Parameter festgestellt werden. Letztere ergeben sich ausschließlich aus den spezifischen Eigenschaften der Kamera und der verwendeten Linse. Die intrinsischen Parameter definieren dabei den Zusammenhang zwischen dem dreidimensionalen Kamerakoordinatensystem und den zweidimensionalen Bildpixeln. Sie haben direkten Einfluss auf das aufgenommene Kamerabild, indem das wahrgenommene Bild z.B. verzerrt dargestellt wird. Im Gegensatz zu extrinsischen Parametern sind die intrinsischen Parameter keinen zeitlichen Veränderungen unterworfen. Sie können deshalb einmalig vorher festgestellt werden und dann abgespeichert und wiederverwendet werden.

#### 5.4.2.1 Kalibration der Kamera

Die Feststellung der kameraeigenen Parameter wird allgemein auch als Kamerakalibration bezeichnet. Hierbei werden in dem Kalibrationsprozess die Abweichungen von der idealen Norm festgestellt.

Um die intrinsischen Kameraeigenschaften durch verschiedene Kameraaufnahmen festzustellen muss der Zusammenhang zwischen den 3D-Objektkoordinaten und ihren korrespondierenden 2D-Bildprojektionen bekannt sein. Eine bewährte Methode zur Feststellung der Kameraparameter ist deshalb die Kalibration mit einem Schachbrettmuster. Das Muster als 3D-Objekt hat dabei eine sehr einfache, gleichmäßige Geometrie. Als Objektpunkte werden die internen Ecken zwischen angrenzenden Feldern gewählt. Alle diese Punkte liegen planar auf einer Ebene und haben einen konstanten Abstand zueinander in vertikale wie auch horizontale Richtung. Die Objektkoordinaten der Eckpunkte lassen sich somit einfach automatisch generieren. Die Verwendung eines Schachbrettmusters hat außerdem den Vorteil, dass die internen Eckpunkte besonders markante, leicht detektierbare Features sind. Abbildung 16 zeigt das verwendete Muster mit den erkannten Ecken. Im Anschluss kann ihre 2D-Bildprojektion leicht mit den 3D-Objektkoordinaten korreliert werden.



Zur Kalibration wurde ein kleines Programm geschrieben welches die Anpassung der Parameter durch den Nutzer erlaubt und weitere Schritte automatisiert. Intern verwendet das Programm die OpenCV-Bibliotheksmethoden `findChessboardCorners()` zur Erkennung der Ecken im Kamerabild (siehe Abbildung 16) und `calibrateCamera()` zur Kalibration der Kamera. [38] Neben der radialen und tangentialen Verzerrung durch die Linse liefert uns die Kalibration auch die Brennweite und die Position des Fokus der Kamera. Nach abgeschlossener Kalibration werden die ermittelten Parameter in einer XML-Datei gespeichert und können so jederzeit geladen und wiederverwendet werden.



#### 5.4.2 Korrektur des Kamerabildes

Durch die Bestimmung der intrinsischen Kameraparameter können Verzeichnungen im Kamerabild durch die Software ausgeglichen werden. Verzeichnung meint dabei einen geometrischen Fehler in der Abbildung, der zu einer lokalen Abweichung im Abbildungsmaßstab führt. Ein typisches Merkmal ist beispielsweise eine zunehmende Verzerrung des Kamerabildes mit zunehmendem Abstand von der optischen Achse in Richtung der Bildränder, die durch die Verwendung sphärischer Linsen bedingt ist. Diese bündeln durch ihre Wölbung das eintreffende Licht unterschiedlich stark, eine radiale Verzerrung des Bildes ist das Resultat.

Soll die photographische Abbildung wie in unserem Fall für präzise Messungen verwendet werden, ist es notwendig, diesen störenden Effekt zu korrigieren. Vor der weiteren Verarbeitung des Kamerabildes werden dazu die intrinsischen Kameraparameter zur Entzerrung mit der OpenCV-Methode `undistort()` auf jeden Kameraframe angewendet. [38]

#### 5.4.3 Konvertierung der Farbräume des Kamerabilds von RGB zu HSV

Neben der Entzerrung zur Korrektur optischer Verzeichnungen wird jeder Kameraframe außerdem von dem ursprünglichen RGB- in den HSV-Farbraum konvertiert. OpenCV stellt auch hierfür eine Methode namens `cvtColor()` bereit.

Dies hat den Grund, dass das implementierte Verfahren als primäres Kriterium den Farbton zur Unterscheidung einer Landmarke von ihrer Umgebung benutzt. Für die Verwendung innerhalb des Verfahrens ist dazu das HSV-System besser geeignet. So kann man zum Beispiel leichter ein Bild nach allen Rottönen filtern. Hierfür muss man lediglich den Bereich der HUE-Werte einschränken. Filteroperationen die auf starken Kontrasten aufbauen, wie Kantendetektion, können wiederum mit dem VALUE-Anteil durchgeführt werden. Durch Begrenzung des SATURATION-Anteils kann man die Toleranz gegenüber Änderungen in der Farbsättigung anpassen. Im RGB-Farbraum sind die R, G und

B-Anteile der Farbe eines Objekts stark voneinander und von den Beleuchtungsverhältnissen abhängig. [39] Das bedingt einen großen Unterraum, den die Schwellwerte für die Erkennung einer Landmarke auch bei schwankenden Lichtverhältnissen erzeugen und damit potentiell zu einer hohen Fehl-Erkennungsrate führt.

### 5.4.3 Erkennung einer Landmarke aus der Verarbeitung des Kamerabildes

Eine kleine Vorbemerkung: Die nachfolgenden Schritte können parallel nebenläufig zur Erkennung unterschiedlicher Landmarken auf ein einzelnes Kamerabild angewandt werden. Für die Vorstellung der Verarbeitungsschritte wird hier aber nur auf die Erkennung einer Landmarke Bezug genommen.

Die farblichen Charakteristika jeder Landmarke sind anhand von minimalen und maximalen Schwellwerten in den drei Dimensionen des HSV-Farbraums definiert. Beim Eintreffen eines neuen Kameraframes wird das Bild auf die Existenz dieser Landmarke hin analysiert. Im ersten Schritt wird das Bild dazu binarisiert.

#### 5.4.3.1 Binarisierung

Bei der Binarisierung wird ein Bild anhand gegebener Schwellwerte in einen binären Farbraum konvertiert\*. Dies hat den Vorteil, dass anschließende Verarbeitungsschritte auf binären Werten arbeiten können.

Ein Kamerabild kann man sich durch eine 2-Dimensionale<sup>†</sup> Matrix repräsentiert vorstellen, wobei jeder Eintrag dieser Matrix einem Bildpunkt entspricht. Für jeden Pixel wird nun in der Matrix überprüft, ob seine drei Farbwerte Hue, Saturation und Value innerhalb der festgelegten Schwellwerte liegt. Falls dies für alle drei Kriterien zutrifft haben wir einen Treffer und es wird ihm der neue binäre Farbwert 1 (=weiß) zugewiesen, ansonsten eine 0 (=schwarz).

Die Binarisierung des Farbbildes wird über die openCV-Methode *inRange()* realisiert. Diese akzeptiert die Übergabe mehrdimensionaler Schwellwerte. Das Bild enthält jetzt die binäre Information, ob ein gegebener Pixel des ursprünglichen Bildes der Farbe einer Landmarke entspricht. Abbildung 17 zeigt beispielhaft ein Kamerabild binarisiert anhand von sechs<sup>‡</sup> empirisch gewählten Schwellwerten für die blaue Farbe des Papierkorbs.

Im weiteren Verlauf interessieren uns nur diejenigen Pixel, welche im binarisierten Bild den Farbwert 1 erhalten haben.

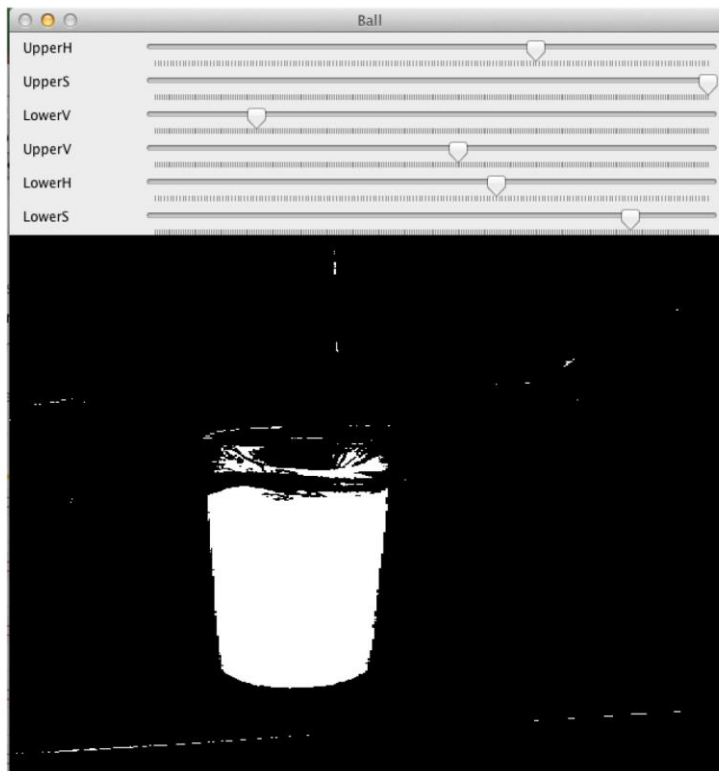
---

\* Es wird nicht das ursprüngliche Bild binarisiert, da dadurch Informationen verloren gehen. Stattdessen wird eine Kopie erstellt

† Die tatsächliche Datenstruktur der Matrix hat mehr als zwei Dimensionen, da jeder Bildpunkt wiederum drei Farbwerte für HUE, SATURATION und VALUE umfasst

‡ 3 minimale Schwellwerte ( $H_{min}, S_{min}, V_{min}$ ) und 3 maximale ( $H_{max}, S_{max}, V_{max}$ )

ABBILDUNG 17: BINARISIERTES BILD MIT SCHWELLENWERTEN FÜR DEN HSV-FARBRAUM



#### BINÄR

Bei der Binarisierung werden die Bildpunkte anhand von Schwellwerten in zwei Kategorien getrennt. Damit lässt sich beispielsweise leichter der Mittelpunkt des Objekts bestimmen.

### 5.4.3.2 Analyse der Zusammenhangskomponenten (ZHK)

Aus der pixelweise isolierten Information "Pixel hat Farbe der Landmarke" soll die Information "bin Teil der zusammenhängenden Fläche X mit der Farbe der Landmarke" entwickelt werden. Dazu müssen zusammenhängende Regionen im Bild gefunden und jeder der zugehörigen Pixel als solcher identifiziert werden. Würde man über die isolierten Informationen keine Zusammenhangskomponenten bestimmen, hätte man das Problem, dass jeder Pixel mit der Farbe der Landmarke eine eigene unabhängige Landmarke repräsentiert.

Das Prinzip der ZHK lehnt sich dabei an den Begriff aus der Graphentheorie an, wonach eine ZHK ein Subgraph ist, in dem jedes Paar zweier Knoten über einen Pfad miteinander verbunden sind. Angewandt auf das Problem der Bildverarbeitung gehören zwei Pixel (die Knoten) im binarisierten Bild genau dann zur selben Zusammenhangskomponente, wenn sie im Bild benachbart sind\*. Die Relation "ist benachbart" ist dabei reflexiv, symmetrisch und transitiv.

Zur Analyse und Bestimmung der ZHK wird die Funktion *cvLabel()* der Erweiterungsbibliothek *cvBlob* verwendet. [37] Das Verfahren basiert dabei auf dem Algorithmus von [40] und speichert weitere Informationen, wie die Größe, zugewiesenes Label, in der eigenen Blob-Datenstruktur.

### 5.4.3.3 Filtern der Zusammenhangskomponenten

Im vorangehenden Schritt wurden im Binärbild aus weißen Pixeln zusammenhängende Regionen entwickelt. Zur Bestimmung der Copterposition müssen wir jedoch in der Lage sein genau eine Landmarke fehlerfrei und eindeutig zu erkennen. Um weitgehend zu vermeiden, dass kleine Bildpunkte z.B. durch Rauschen im Kamerabild als Referenzpunkte herangezogen werden Ausreißer

\* Benachbart heißt in diesem Fall: Sie haben eine gemeinsame Kante oder Ecke (sogenannte 8-Konnektivität)

anhand ihrer Größe herausgefiltert. Die `cvBlob` Funktion `cvFilterByArea()` erlaubt dafür die Angabe einer minimalen und einer maximalen Größe in Anzahl der Pixel. [37] Ist die Anzahl der erkannten ZHK weiterhin  $>1$ , wird außerdem in einem nachgeschalteten Verarbeitungsschritt nur die Größte beibehalten. Nach dem Herausfiltern der Ausreißer verbleibt also immer höchstens ein Kandidat in der Liste.

#### 5.4.3.4 Positionsbestimmung der Zusammenhangskomponente im Kamerabild

Nachdem die Anzahl der möglichen Landmarken im Bild auf genau 1 reduziert wurde kann nun die Position im Bild bestimmt werden. Als maßgebliche Größe für die Position der Fläche wurde hierbei die Position ihres Mittelpunktes festgelegt. Die Funktion `cvLabel()` aus Abschnitt 5.4.3.2 bestimmt neben Label und Größe der Zusammenhangskomponente auch die zentralen Momente. [37]

Mit Hilfe der zentralen Momente  $M_{10}$  in X-Richtung und  $M_{01}$  in Y-Richtung und der Fläche  $M_{00}$  lässt sich die Position  $P$  des Schwerpunktes in einem Binärbild folgendermaßen berechnen:

$$P = \{x, y\} = \left\{ \frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right\}$$

Formel 12: Berechnung des Schwerpunktes

Die Koordinaten des Schwerpunktes entsprechen in dem gefilterten Bild dem Mittelpunkt der Fläche der Zusammenhangskomponente.

#### 5.4.4 Berechnung der Rotation durch Landmarke im Kamerabild

Mit der Bestimmung der Position  $P$  der Landmarke im Kamerabild sind alle erforderlichen Informationen aus dem Kamerabild extrahiert. In den weiteren Schritten wird aus der ermittelten Position im Bild die entsprechende Rotation der Kamera bestimmt.

##### 5.4.4.1 Normalisierung der Bildkoordinaten

Jedes Bild ist in OpenCV wie in Diagramm 1 repräsentiert. Die Repräsentation entspricht einem zweidimensionalen Koordinatensystem mit Ursprung links oben, die Breite  $W$  des Bildes somit der Anzahl der Pixel in X-Richtung, die Höhe  $H$  der Anzahl in Y-Richtung.

Im vorangehenden Schritt wurde mit dem Schwerpunkt der ZHK die Position  $P = \{x, y\}$  der Landmarke bestimmt. Um die Information der Bildkoordinaten in Bezug auf die Kameraräum zu interpretieren, wird die Position von  $P$  normalisiert. Die Ausrichtung des Kamerakoordinatensystems ist dabei durch die optische Achse gegeben, welche die Bildebene im Mittelpunkt  $M = \left\{ \frac{W}{2}, \frac{H}{2} \right\}$  schneidet.\* Um den normalisierten Punkt  $P' = \{\text{Abstand von } P \text{ zu } M \text{ in X-Richtung; Abstand von } P \text{ zu } M \text{ in Y-Richtung}\}$  in Pixeln zu erhalten, wird die Differenz  $P-M$  komponentenweise berechnet.

---

\* Der Mittelpunkt des Bildes als Schnittpunkt der optischen Achse mit der Bildebene wird dabei vereinfachend angenommen. Dies ist nicht unbedingt der Fall, eine präzisere Bestimmung erfolgt bei der Kalibration der Kamera.

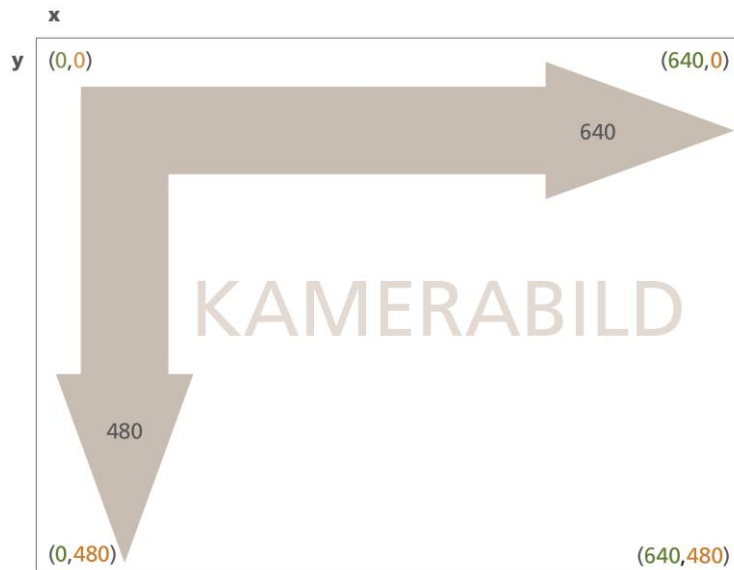


Diagramm 1: Repräsentation eines Bildes in OpenCV

#### 5.4.4.2 Berechnung der Rotation aus der normalisierten Pixelposition

Wie bei den mathematischen Grundlagen des Verfahrens in Kapitel 5.2 erläutert, lassen sich über die festen Umrechnungsfaktoren  $c_{hor}$  und  $c_{ver}$  aus dem neuen Punkt  $P' = \{x', y'\}$  zwei Winkel berechnen:

$$\omega = c_{hor} * x' \text{ und } \gamma = c_{ver} * y'$$

$\omega$  entspricht dabei einer Rotation um die Y-Achse des Bildes,  $\gamma$  einer Rotation um die X-Achse.

Damit  $\omega, \gamma$  eine Rotation in Bezug auf die festgelegte Achsenkonvention ausdrücken (siehe Abbildung 14), müssen die Bildachsen umgedeutet werden. Die vertikale Y-Achse im Bildsystem entspricht der Z-Achse, die horizontale X-Achse im Bildsystem der Y-Achse in der Standardausrichtung. Damit wird  $\omega$  als Yaw,  $\gamma$  als Pitch interpretiert. Durch die zweidimensionale Bildinformation kann kein Rückschluss über einen möglichen Rollwinkel erfolgen. Aus  $\omega, \gamma$  wird dann die 3x3 Matrix  ${}^C R_{LM}$  generiert (siehe Kapitel 5.2).

#### 5.4.5 Abschließende Verarbeitungsschritte

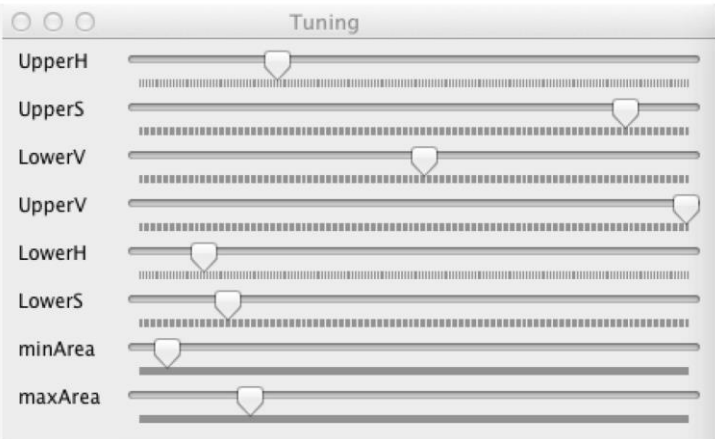
Das weitere Vorgehen innerhalb der Implementierung des Verfahrens orientiert sich an dem in 5.2 geschilderten Ablauf.  ${}^C R_{LM}$  wurde in den vorangehenden Schritten aus dem Kamerabild ermittelt,  ${}^B R_C$  ist durch die Eigenschaften der Montagevorrichtung der Kamera am Rumpf des Copters festgelegt.  ${}^W R_B$  wird in der Praxis durch die externen Winkelangaben des Flugcontrollers bestimmt, in der vorliegenden Implementierung aber simuliert. Ausgestattet mit den drei Rotationsmatrizen wird der Vektor  $v_0$  in Richtung der Landmarke gedreht und anschließend mit dem Parameter  $t$  Anhand des vertikalen Abstandes  $\delta_z$  skaliert.

### 5.5 Einsatz der Software

Die implementierte Klasse soll in Zukunft als Unterroutine in verschiedene Navigationsstrategien der High-Level Steuerarchitektur eingebunden werden (siehe Kapitel 5.1). Für die eigentliche Funktionsweise ist daher keine fortlaufende Benutzerinteraktion notwendig, weshalb auf die Entwicklung einer aufwändigen graphischen Oberfläche verzichtet wurde. Zur Überprüfung und Modifikation der empirisch bestimmten Parameter wie der Schwellwerte für die zu detektierende

Farbe (siehe Kapitel 5.4.3.1) der Landmarke und der minimalen/maximalen Größe der akzeptierten Flächen (siehe 5.4.3.3) während der Ausführung wurden einfache Schieberegler über openCV Trackbars implementiert. (Abbildung 18) Außerdem wurde die Ausgabe des modifizierten Kamerabildes inkl. der detektierten Landmarken und ihres bestimmten Mittelpunktes realisiert. (Abbildung 19)

**ABBILDUNG 18: TUNINGREGLER FÜR DIE ERKENNUNGSPARAMETER**



The image shows a window titled "Tuning" with a standard macOS-style title bar (three buttons on the left). Inside the window, there are eight horizontal sliders, each with a white triangular knob. The sliders are labeled on the left as follows: UpperH, UpperS, LowerV, UpperV, LowerH, LowerS, minArea, and maxArea. The sliders are positioned at various points along their tracks, representing the current values of these parameters.

**NOCH EIN BISSCHEN GELB?**  
Schieberegler passen die Erkennungsparameter während der Laufzeit an.

**ABBILDUNG 19: KAMERABILD MIT ERKANNTER LANDMARKE**



The image shows a window titled "Kamera" with a standard macOS-style title bar. The main content is a camera feed of a dark, textured surface. A vertical pole is visible in the center. A red and yellow rectangular box is drawn around a section of the pole, indicating a detected landmark. A small blue crosshair is centered within this box.

**ZIEL GEFUNDEN**  
Erkennt die Software eine Landmarke, wird sie in das Kamerabild eingezeichnet.

## 6 Evaluation

In diesem Kapitel wird das entwickelte Verfahren evaluiert. Um eine robuste Erkennung der künstlichen Landmarke im Kamerabild zu gewährleisten, wurden dazu zunächst empirisch die oberen und unteren Schwellwerte für die drei Farbkomponenten und die minimale und maximale Größe der Landmarke ermittelt. Im Verlauf der vorangehenden Entwicklungsphasen hatte sich dabei eine blaue, wenig reflektierende Oberfläche bewährt. Über die in Kapitel 5.5 erwähnten Schieberegler wurden zuerst die tolerierten Farbbereiche in jeder Dimension in einem iterativen Prozess expandiert und eingeschränkt, um gleichzeitig Falscherkennungen im Kamerabild zu minimieren und robust gegenüber wechselnden Lichtbedingungen zu sein. Die dabei ermittelten und übernommenen Werte sind folgende:

Blaue Landmarke	Minimaler Schwellwert	Maximaler Schwellwert
Hue	90	120
Saturation	75	175
Value	100	200

Mit diesen Werten konnte die Landmarke zuverlässig von der Umgebung unterschieden werden. Als nächstes wurde versucht, sinnvolle Grenzen für die Größe der Landmarkenprojektion im Bild zu definieren. Vergleichbar den Schwellwerten für die drei HSV Farbkomponenten wurden dabei in wiederholten Schritten die untere und obere Grenze angepasst. Es zeigte sich, dass vor Allem der minimale Wert dabei Einfluss auf die Zuverlässigkeit der korrekten Erkennung hat. Da die dargestellte Größe jedoch unmittelbar mit der tatsächlichen Größe der gewählten Marke und dem Abstand der Kamera zu ihr zusammenhängt, muss hierfür in der Praxis ein vernünftiges Maß zwischen Fehlererkennung und angestrebter Reichweite des Systems gewählt werden. In Hinblick auf die nachfolgenden Testszenarien mit einer Landmarke von knapp 20cm Durchmesser und Distanzen zwischen 1 und 5 Metern wurden folgende Werte festgelegt:

Minimale Fläche (in Pixeln)	Maximale Fläche (in Pixeln)
400	10000

Nach Ermittlung der übergreifenden Parameter für die Erkennung kann nun das Verfahren evaluiert werden.

Die Positionsbestimmung anhand der optischen Landmarken eignet sich im Einsatz für zwei symmetrische Szenarien: Bestimmung bzw. Verbesserung der eigenen Positionsschätzung anhand einer bekannten Landmarke und die Bestimmung der Landmarkenposition anhand der eigenen Position. Um die Funktionstüchtigkeit und Präzision des Algorithmus in beiden potentiellen Einsatzszenarien zu überprüfen, wurden zwei separate Szenarien entworfen.

In Szenario 1 wurde die Kamera in einzelnen Stationen um eine ortsfeste Landmarke mit bestimmter Position bewegt und dabei die notwendigen Lage- und Höhendaten der Kamera vorgegeben. Dies

simuliert den Einsatz zur eigenen Positionsermittlung auf dem Multicopter. Szenario 2 simuliert den umgekehrten Fall: Die Position der Landmarke wird während der Erfassung entlang einer vorgegebenen Kreisbahn im Bild verändert, die Kamera behält dabei eine feste voreingestellte Position und Lage.

## 6.1 Szenario 1: Positionsbestimmung der Kamera

Zur Evaluation des ersten Szenarios wird in zwei Messreihen mit unterschiedlichen Pitch-Winkeln die Position der Kamera an unterschiedlichen Orten ermittelt. Dazu wurde zunächst eine Position für die Landmarke  $L = (100,100,0)$  festgelegt und dem Programm als Referenzwert übergeben. Der erste Wert entspricht der simulierten östlichen Länge in Weltkoordinaten, der zweite Wert der nördlichen Breite. Weiterhin kam in diesem Szenario eine planare Fläche als Landmarke zum Einsatz, weshalb die vertikale Komponente auf 0 gesetzt wurde. Zur Positionierung der Kamera wurden im Anschluss vier Aufstellpositionen kreisförmig im Abstand von 100cm zur Landmarke durch Markierungen auf dem Boden gekennzeichnet. Die Verteilung entspricht dabei den vier diametral angeordneten Himmelsrichtungen, aus denen die Kamera auf die Landmarke blicken soll\*.

Durch die Befestigung der Kamera am Stativ ergab sich ein horizontaler und vertikaler Offset abhängig vom Pitch der Kamera. Um die Genauigkeit der innerhalb der Testreihe ermittelten Position der Kamera später feststellen zu können wurden diese mit einem Maßband und einer digitalen Wasserwaage ermittelt. Tabelle 3 fasst die Daten der Messvorrichtung noch einmal übersichtlich zusammen, Tabelle 4 die tatsächlichen Positionen der vier Messpunkte im globalen Bezugssystem hinsichtlich der Landmarke. Vor dem Start der Messung an jeder Position wird die jeweilige Höhe und der Yaw-Winkel der Software als Parameter übergeben. Zur Ermittlung eines repräsentativen Wertes werden 40 Samples aufgenommen und die berechneten Positionen gespeichert.

Neigungswinkel (Pitch) der Kamera	-30°	-45°
Stativentfernung von LM	100 cm	100 cm
Horizontaler Offset von Stativposition	12,5 cm	8,2 cm
Resultierender Horizontaler Messabstand	<b>112,5 cm</b>	<b>108,2 cm</b>
Stativhöhe über LM	57 cm	57 cm
Vertikaler Offset der Kamera über Stativ	18,5 cm	15,5 cm
Resultierende Vertikale Messhöhe	<b>75,5 cm</b>	<b>72,5 cm</b>

Tabelle 3: Daten der Messvorrichtung

---

\* Die vier Himmelsrichtungen sind im Folgenden durch ihre Yaw-Winkel ausgehend von der 0° Position (=Nord) angegeben. Blick in Richtung Osten entspricht 90° Yaw, Süd = 180°, West = 270°



	Yaw	Pitch 30°		Pitch 45°	
		Ost	Nord	Ost	Nord
Position 1	0°	100,0	-12,5	100,0	-8,2
Position 2	270°	212,5	100,0	208,2	100,0
Position 3	180°	100,0	212,5	100,0	208,2
Position 4	90°	-12,5	100,0	-8,2	100,0

Tabelle 4: Daten der vier Messpositionen

### 6.1.1 Messreihe 1: Pitch -30°

Die Software hat an keinem Messpunkt Probleme, die Landmarke im Bild zu erkennen und liefert umgehend Resultate. Diagramm 2 zeigt die berechneten Werte an den unterschiedlichen Positionen und die tatsächlichen Messpunkte als Referenz. Neben der durchweg vorhandenen, leichten Überschätzung der Distanz zur Landmarke ist auffällig, dass die ermittelten Werte für die jeweilige Position dabei sehr nahe beieinander liegen. Ersteres drückt sich auch in den Zahlen aus: Über alle Messpositionen hinweg liegt die mittlere absolute Abweichung der Messwerte von den Referenzpunkten in Richtung „von der Mitte weg“ bei 11,84cm während die seitliche Abweichung nur knapp 1,15cm beträgt. Die Messwerte bleiben dabei konstant stabil und es gibt kaum Ausreißer. Die gemittelte Standardabweichung der berechneten Positionsschätzungen liegt bei 0,51.

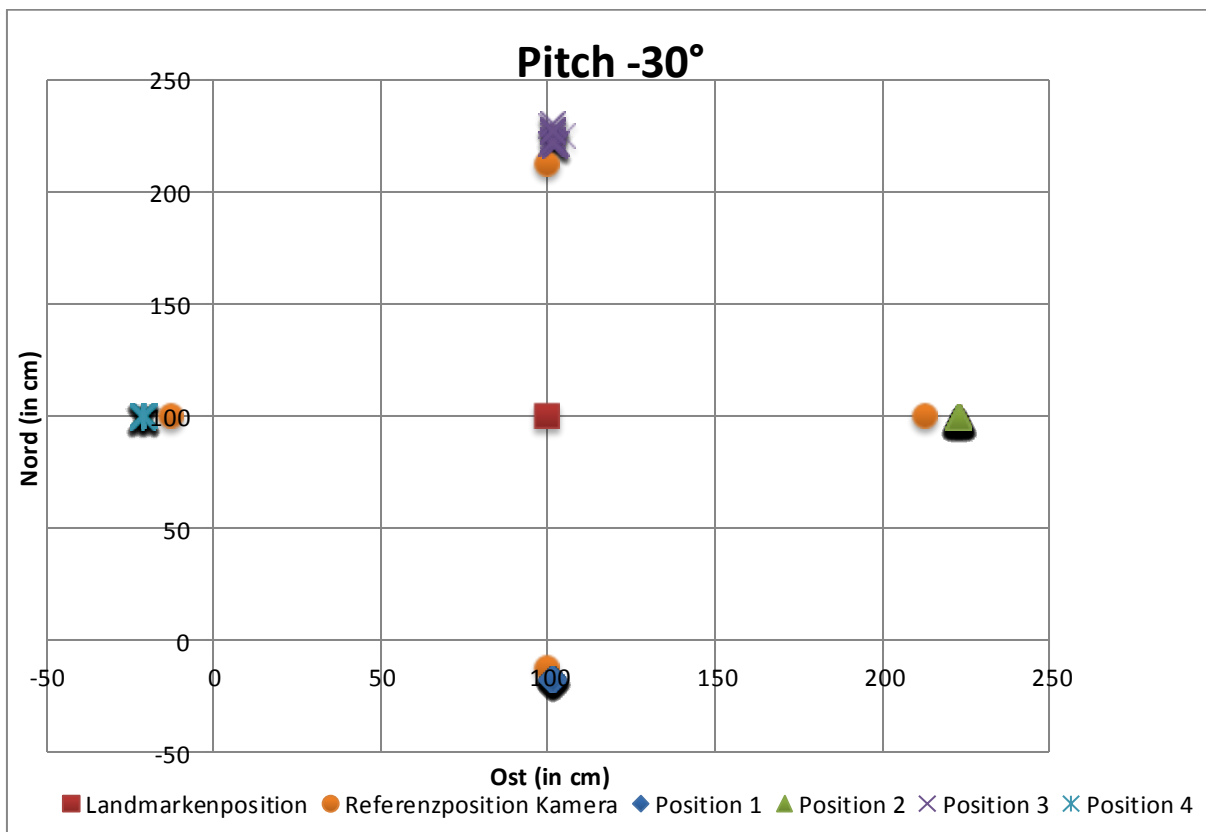


Diagramm 2: Plot der Messergebnisse aus Messreihe 1 mit Kamerapitch -30°. Die orangenen Punkte zeigen die kreisförmig angeordneten Aufstellpositionen der Kamera als Referenz. Die berechneten Positionen sind mit den Namen der assoziierten Aufstellposition bezeichnet und farbig hervorgehoben.

### 6.1.2 Messreihe 2: Pitch -45°

Im Anschluss an Messreihe 1 wird der gleiche Ablauf mit neuen Parametern wiederholt. Der Pitch der Kamera wird auf -45° erhöht und die resultierende Höhe zur Berechnung angepasst. (siehe Tabelle 3) Der Plot in Diagramm 3 zeigt die Messergebnisse. Erneut ist die starke Ballung der gemessenen Position zu bemerken, die Standardabweichung innerhalb der Werte liegt diesmal sogar nur bei 0,24cm. Ein wenig geringer, aber immer noch merklich ist die kontinuierliche Überschätzung der Distanz zur Landmarke. Dies ist besonders bei der 4. Messposition feststellbar, hier liegt die Software bei der Positionsschätzung um fast 17cm daneben. Über alle Messpunkte hinweg entsteht aber trotzdem ein verbesserter Wert und die durchschnittliche Distanz zur Landmarke wird nur um 6,2cm überschätzt.

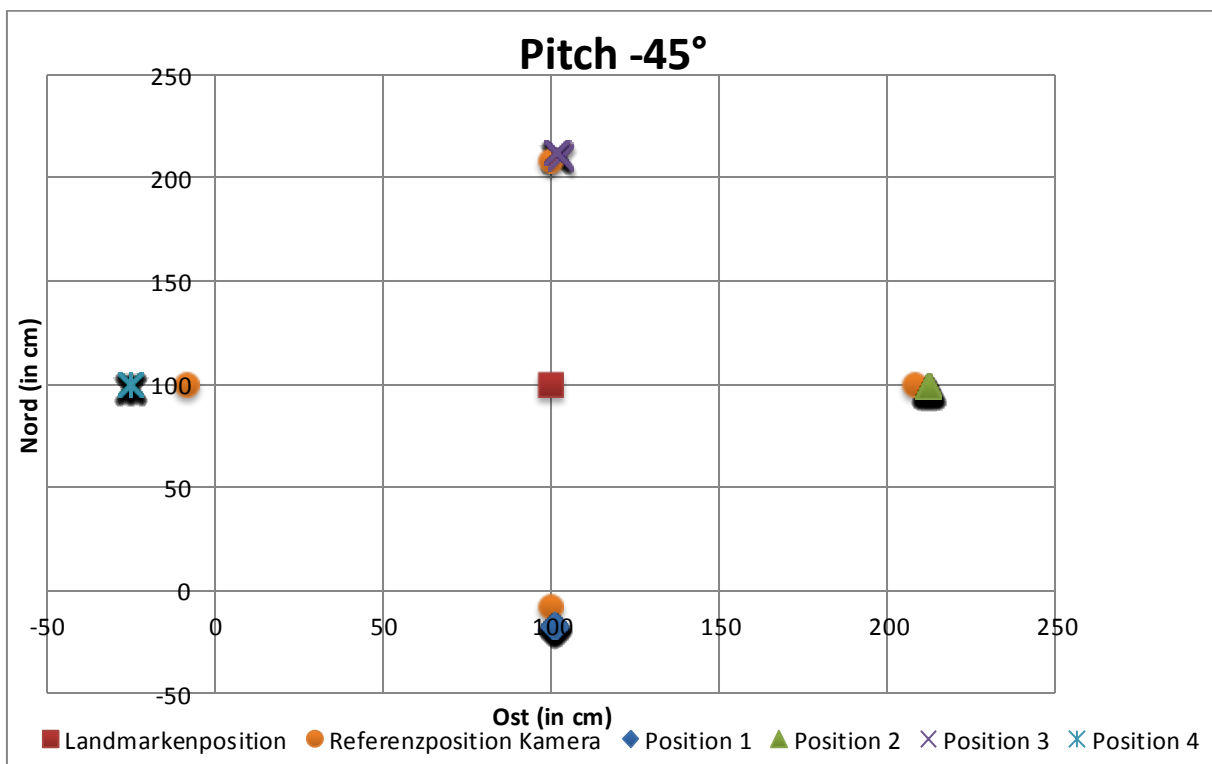


Diagramm 3: Plot der Messergebnisse aus Messreihe 2 mit Kamerapitch -45°. Die orangenen Punkte zeigen die kreisförmig angeordneten Aufstellpositionen der Kamera als Referenz. Die berechneten Positionen sind mit den Namen der assoziierten Aufstellposition bezeichnet und farbig hervorgehoben.

## 6.2 Szenario 2: Positionsbestimmung der Landmarke

Im zweiten Szenario wird der Fall simuliert, dass die Position des Copters bekannt ist und die eigene Schätzung zum Erlernen neuer Landmarken genutzt wird. Dazu wurde die Kamera an einer festen Position aufgestellt und die künstliche Landmarke anschließend im Bild bewegt. Des Weiteren sollte diesmal auch der Einsatz des Systems aus einer größeren Distanz zur Landmarke getestet werden. Ein Problem das sich spätestens bei der Analyse und Auswertung dieses Szenarios nachteilig auswirkt, besteht darin, dass keine präzise zeitliche Synchronisation zwischen den Messungen und der Bewegung der Landmarke zur Verfügung stand. Damit statt der direkten Korrelation zwischen Messung und tatsächlicher Position ein Maß für die Präzision des Verfahrens besteht, wurde die gleiche Bewegung der Landmarke über einen längeren Zeitraum wiederholt. Als Testumgebung diente

hierbei das Spielfeld der BerlinUnited-Fußball-Roboter, dessen Maße und Proportionen dem Standardregelwerk in der Small Size Robot League entspricht. [41]

Kamera Roll	0°
Kamera Pitch	-42°
Kamera Yaw	236°
Kamera Position X	405cm
Kamera Position Y	605cm
Kamera Position Z	178cm

**Tabelle 5: Positions- und Lageparameter der Kamera in Testszenario 2**

Wie in Szenario 1 wurden Kamera und Stativ aufgebaut und ihre Lage, Ausrichtung und Abstand zum Boden gemessen. Zur leichteren Orientierung wurde dafür die rechte obere Ecke des Spielfelds als Messpunkt gewählt mit Blick Richtung Anstoßpunkt. Dieser ist dabei ca. 4m von der Kamera entfernt. Tabelle 5 zeigt die bestimmte Position der Kamera, in Abbildung 20 sieht man das Bild der Kamera am vom Aufstellungsort. Außerdem im Bild zu sehen ist die bewegliche Landmarke in Form eines blauen Balles. Dieser wird während der Messung in einer gleichmäßigen Bewegung 30 Sekunden lang kreisförmig entlang der Linie des Mittelkreises bewegt.

Wie eingangs erwähnt, ist es aufgrund der mangelnden Synchronisierung zwischen tatsächlicher Position des Balles und Messung schwierig, eine exakte Analyse anhand der Korrelation zwischen Mess- und Sollwert durchzuführen. In Diagramm 4 wurde stattdessen das Spielfeld mit den ermittelten Landmarkenpositionen überlagert. Auffallend ist auch hier wieder, dass die Qualität der Positionsschätzung mit zunehmender Distanz zur Kamera nachlässt. Während die Position des Balles auf der zugewandten Seite des Mittelkreises noch näherungsweise übereinstimmt, nimmt die Genauigkeit auf der entlegenen Hälfte stark ab. Die am weitesten von der Kamera entfernte, berechnete Position hat eine Distanz von 472cm in der XY-Ebene. Nimmt man als Referenz nun den Punkt auf dem Mittelkreis mit der größten Distanz (=414cm) \* zum Aufstellort der Kamera ergibt sich eine Abweichung von 58cm.

---

\* Dieser Punkt liegt auf dem entfernten Schnittpunkt der Diagonalen zwischen den gegenüberliegenden Spielfeldecken und dem Mittelkreis.

ABBILDUNG 20: TESTSETUP FÜR SZENARIO 2



### ROBOCUP

Die bekannten Maße des Spielfeldes helfen bei der Positionsbestimmung.

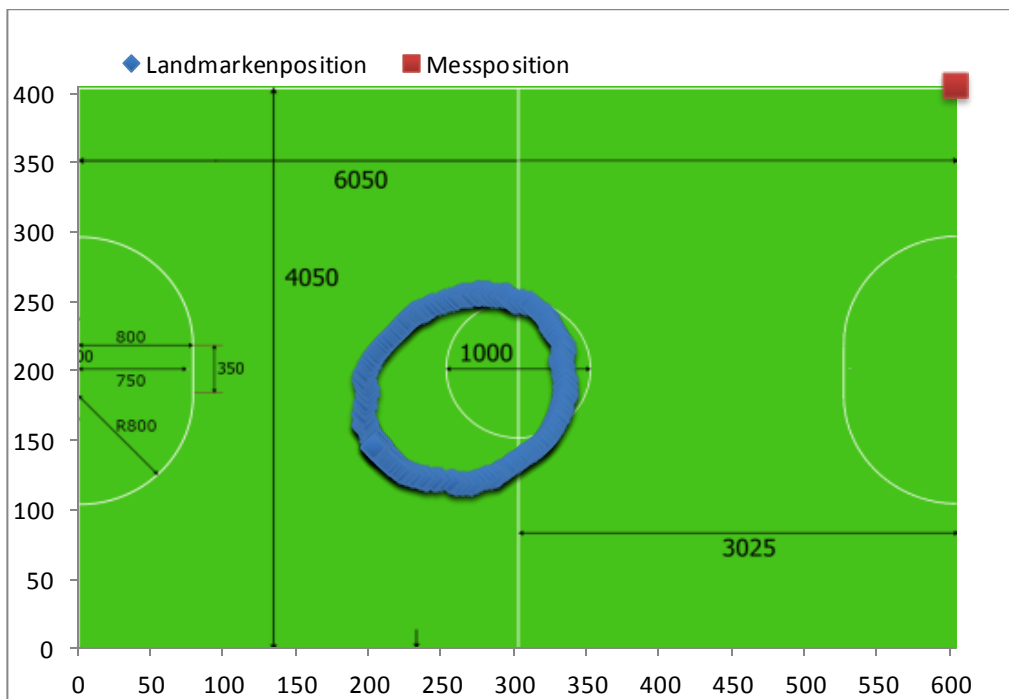


Diagramm 4: Plot der Positionsbestimmung der bewegten Landmarke (Szenario 2). Die Landmarke wurde auf der Linie des Mittelkreises bewegt. Die blauen Messpunkte zeigen die berechnete Position.

## 7 Diskussion und Ausblick

Das Fazit über die vorgenommenen Modifikationen an der NeuroCopter-Hardware fällt durchweg positiv aus. Besonders der Kompass profitiert von der exponierten Montage mit einigem Abstand zur sonstigen Elektronik. Der Einfluss der Vibrationsdämpfung auf den Beschleunigungssensor ist nicht nur optisch in Form einer Verbesserung des Loiter-Verhaltens erkennbar, sondern auch in den Messungen deutlich nachzuvollziehen.

Die Analyse der Ergebnisse der beiden Messreihen für die kamerabasierte Methode zur Lokalisierung zeigt ein zweischneidiges Bild. Die implementierten Algorithmen funktionieren besonders im nahen Bereich unter Laborbedingungen gut. Wird die Distanz erhöht ist allerdings gleichzeitig zu erkennen, dass das entwickelte Verfahren abhängig von der Kameraneigung Probleme hat, sobald die projizierte Position der Landmarke im Kamerabild "nach oben" von der optischen Achse abweicht. Im typischen Einsatzszenario ist der Blick der Kamera von schräg oben über die Bodenebene gerichtet. Die Projektionsebene ist nicht parallel zur erfassten Objektebene und damit besteht kein konstantes Verhältnis zwischen den realen Distanzen und ihrer projizierten Größe in Pixeln. Dieses Verhältnis wird neben der relativen Bildposition auch durch die Entfernung zur Kamera und ihrem Neigungswinkel beeinflusst. Angewandt auf das implementierte Verfahren bedeutet das: Pixel und Winkel haben ein konstantes Verhältnis zueinander und können deswegen umgerechnet werden. Aber kleine Fehler bei der Bestimmung der Kameralage und bei der Erkennung der Landmarke im Bild haben gegebenenfalls große Auswirkungen auf die Genauigkeit der Positionsschätzung. Für einen zukünftigen Praxiseinsatz auf dem Coptersystem wird man feststellen müssen, inwiefern verrauschte Messungen der Lage- und Höhensensoren die Funktionsweise des entwickelten Verfahrens beeinträchtigen.

Durch die Integration in eine globale Navigationsstrategie ist aber eine situationsabhängige Anwendung folgendermaßen vorstellbar: Die Orientierung zum Zielort erfolgt über die Berechnung der eigenen Ausrichtung anhand der Winkel im Kamerabild. Eine kontinuierliche Überprüfung der geflogenen Route während des Anflugs passiert dann über die Extraktion der Eigenbewegung aus dem Optischen Fluss. Ist der Copter anschließend hinreichend nah an der Landmarke, kann er seine eigene Position über die entwickelte Methode verbessern.

Weitere Optimierungen sind via Fusion der optischen Positionsschätzungen mit alternativen Navigationssystemen oder durch den Einsatz einer stärker zum Boden gerichteten Kamera möglich. Eine Ausrichtung der Kamera Richtung Boden bedingt jedoch gleichzeitig eine Beschränkung des erfassbaren Ausschnitts der Umwelt. Zusätzliche Methoden mit kamerabasierter Sensorik können in Zukunft eine wichtige Rolle innerhalb des NeuroCopter-Systems übernehmen und herkömmliche Sensorik ersetzen. Besonders für biologisch inspirierte Systeme ergeben sich vielfältige Möglichkeiten zur Entwicklung von Verfahren die miteinander harmonisieren und sich ergänzen. Ideen sind dabei eine horizontbasierte Lagebestimmung, oder die Bestimmung der Flughöhe aus dem optischen Fluss.

## 8 Literaturverzeichnis

- [1] T. Landgraf, „Analysis of the waggle dance motion of honeybees for the design of a biomimetic honeybee robot,“ PubMed, Berlin.
- [2] R. Menzel, „Honey bees navigate according to a map-like spatial memory,“ in *Proceedings of the National Academy of Sciences of the United States of America*, USA, 2005.
- [3] R. Menzel, K. Lehmann und G. Manz, „Vector integration and novel shortcutting in honeybee navigation,“ *Apidologie*, Nr. 0044-8435, pp. 229-243, 2012.
- [4] M. V. Srinivasan, „Honeybees as a model for the study of visually guided flight,“ *Physiological Reviews*, Bd. vol. 91, pp. 389-411, 2011.
- [5] G. D. P. Conte, „An integrated uav navigation system based on aerial image matching,“ in *Proceedings of the IEEE Aerospace Conference*, 2008.
- [6] T. A. O. K. Q. Kanade, „Real-time and 3d vision for autonomous small and micro air vehicles,“ in *43rd IEEE Conference on Decision and Control*, 2004.
- [7] A. d. W. A. H. R. H. G. P. S. R. N. Bachrach, „Range - robust autonomous navigation in gps-denied environments,“ in *2010 IEEE International Conference on Robotics and Automation (ICRA)*, 2010.
- [8] M. W. S. S. D. S. R. Blösch, „Vision based mav navigation in unknown and unstructured environments,“ in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010.
- [9] J. Weingarten, „EKF-based 3D SLAM for structured environment reconstruction,“ in *International Conference on Intelligent Robots and Systems*, 2005.
- [10] H. J. Chang, C. S. G. Lee, Y.-H. Lu und Y. C. Hu, „P-SLAM: Simultaneous Localization and Mapping With Environmental-Structure Prediction,“ in *IEEE TRANSACTIONS ON ROBOTICS*, VOL. 23, NO. 2, 2007.
- [11] C. Kemp, Visual control of a miniature quad-rotor helicopter, Bd. Ph.D. thesis, Churchill College: University of Cambridge, 2006.
- [12] L. Meier, P. Tanskanen, F. Fraundorfer und M. Pollefeys, „PIXHAWK: A system for autonomous flight using onboard computer vision,“ in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai , 2011.
- [13] e. a. Philippides, „A Model of Ant Route Navigation Driven by Scene Familiarity,“ *PLoS Comput Biol* 8(1), 2012.

- [14] R. L. D. R. T. P. R. & W. R. Möller, „Insect strategies of visual homing in mobile robots,“ *Proc. Computer Vision and Mobile Robotics Workshop*, Bd. 98, pp. 37-45, 2001.
- [15] V. V. e. a. Hafner, „An autonomous flying robot for testing bio-inspired navigation strategies,“ in *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*, 2010.
- [16] P. M. W. L. Lambrinos, „A mobile robot employing insect strategies for navigation,“ 1999.
- [17] G. INDIVERI und E. .. CHICCA, „A VLSI neuromorphic device for implementing spike-based neural networks,“ 2011.
- [18] Invensense, „MPU6000 Product Sheet,“ [Online]. Available: <http://www.invensense.com/mems/gyro/documents/PS-MPU-6000A-00v3.4.pdf>. [Zugriff am 15 09 2013].
- [19] Wikipedia, „Gyroscope,“ [Online]. Available: <http://en.wikipedia.org/wiki/Gyroscope>. [Zugriff am 16 09 2013].
- [20] ArduPilot, „ArduCopter Wiki - Vibration Damping,“ [Online]. Available: <http://copter.ardupilot.com/wiki/vibration-damping/>.
- [21] „DIYDrones Forum,“ [Online]. Available: <http://diydrones.com/forum/topics/vibration-isolation-and-dampening-of-apm-px4-for-version-2-9>. [Zugriff am 23 09 2013].
- [22] Measurement Specialities, „Datasheet MS5611,“ [Online]. Available: [http://www.meas-spec.com/product/t\\_product.aspx?id=8503](http://www.meas-spec.com/product/t_product.aspx?id=8503). [Zugriff am 20 09 2013].
- [23] MaxBotix, „Datenblatt XL MaxSonar-EZ,“ 2013. [Online]. Available: [http://www.maxbotix.com/documents/XL-MaxSonar-EZ\\_Datasheet.pdf](http://www.maxbotix.com/documents/XL-MaxSonar-EZ_Datasheet.pdf). [Zugriff am 22 09 2013].
- [24] Honeywell, „www.honeywell.com,“ [Online]. Available: [http://www51.honeywell.com/aero/common/documents/myaerospacecatalog-documents/Defense\\_Brochures-documents/HMC5883L\\_3-Axis\\_Digital\\_Compass\\_IC.pdf](http://www51.honeywell.com/aero/common/documents/myaerospacecatalog-documents/Defense_Brochures-documents/HMC5883L_3-Axis_Digital_Compass_IC.pdf). [Zugriff am 24 09 2013].
- [25] U-Blox, „www.u-blox.com,“ [Online]. Available: [http://www.u-blox.com/images/downloads/Product\\_Docs/LEA-6\\_DataSheet\\_%28GPS.G6-HW-09004%29.pdf](http://www.u-blox.com/images/downloads/Product_Docs/LEA-6_DataSheet_%28GPS.G6-HW-09004%29.pdf). [Zugriff am 20 08 2013].
- [26] ArduPilot, „ArduCopter Wiki - Loiter Mode,“ [Online]. Available: <http://copter.ardupilot.com/wiki/loiter-mode/>. [Zugriff am 10 09 2013].
- [27] PixHawk, „PX4FLOW Smart Camera,“ [Online]. Available: <https://pixhawk.ethz.ch/px4/modules/px4flow>. [Zugriff am 24 09 2013].
- [28] Omnivision, „Datenblatt OV7720/7221 CMOS Sensor,“ [Online]. Available: [http://www.zhopper.narod.ru/mobile/ov7720\\_ov7221\\_full.pdf](http://www.zhopper.narod.ru/mobile/ov7720_ov7221_full.pdf). [Zugriff am 24 09 2013].

- [29] ArduPilot, „ArduCopter Mainpage,“ [Online]. Available: <http://copter.ardupilot.com/>. [Zugriff am 20 09 2013].
- [30] ETH Zürich, „MAVLink Kommunikations Protokoll,“ [Online]. Available: <http://qgroundcontrol.org/mavlink/start>. [Zugriff am 15 09 2013].
- [31] ISEE, „IGEPv2 Produktübersicht,“ [Online]. Available: <http://www.isee.biz/products/processor-boards/igepv2-board>. [Zugriff am 10 09 2013].
- [32] T. Landgraf und e. al, „NeuroCopter: Neuromorphic Computation of 6D Ego-Motion of a Quadcopter,“ in *Lecture Notes in Computer Science*, 2013, pp. 143-153.
- [33] The Engineering Department, „Inertial Sensors and Systems - An Introduction,“ GeneSys Elektronik GmbH, 06 2000. [Online]. Available: <http://www.genesys-offenburg.de/en/pdfs/inerteng.pdf>. [Zugriff am 25 09 2013].
- [34] M. L. Harald Frenz, „Absolute travel distance from optic flow,“ *Vision Research* 45, Bd. 45, pp. 1679-1692, 2005.
- [35] W. a. B. P. Premerlani, „Direction cosine matrix imu: Theory,“ *DIYDRONES*, pp. 13-15, 2009.
- [36] OpenCV, „OpenCV Homepage,“ [Online]. Available: <http://opencv.org/>. [Zugriff am 29 08 2013].
- [37] cvBlob, „Blob Library for OpenCV,“ [Online]. Available: <http://code.google.com/p/cvblob/>. [Zugriff am 15 08 2013].
- [38] OpenCV, „Camera Calibration and 3D Reconstruction,“ [Online]. Available: [http://docs.opencv.org/modules/calib3d/doc/camera\\_calibration\\_and\\_3d\\_reconstruction.html](http://docs.opencv.org/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html). [Zugriff am 10 09 2013].
- [39] X. J. Y. S. J. W. H.D. Cheng, „Color image segmentation: advances and prospects,“ *Pattern Recognition*, Bd. 34, Nr. 12, pp. 2259-2281, 12 2001.
- [40] C.-j. C. „, C.-j. L. Fu Chang, „A linear-time component-labeling algorithm using contour tracing technique,“ in *Computer Vision and Image Understanding*, 2004.
- [41] RoboCup, „Small Size Robot League: Rules,“ 2013. [Online]. Available: [http://robocupssl.cpe.ku.ac.th/\\_media/rules:ssl-field-2012-final.pdf](http://robocupssl.cpe.ku.ac.th/_media/rules:ssl-field-2012-final.pdf). [Zugriff am 28 09 2013].
- [42] S. L. S. C. F. Bosch, „Autonomous detection of safe landing areas for an uav from monocular images,“ in *Intelligent Robots and Systems*, 2006.
- [43] Wikipedia, „Moment (Bildverarbeitung),“ [Online]. Available: [http://de.wikipedia.org/wiki/Moment\\_%28Bildverarbeitung%29](http://de.wikipedia.org/wiki/Moment_%28Bildverarbeitung%29). [Zugriff am 26 09 2013].
- [44] RTOM, „MoonGel Damper Pads,“ [Online]. Available: <http://www.rtom.com/moongel.htm>. [Zugriff am 20 09 2013].



## Abbildungsverzeichnis

Abbildung 1: Systementwurf NeuroCopter. Die Informationen der Sensorik (links) werden innerhalb der Flugsteuerung verarbeitet und interpretiert. Lokalisierung innerhalb des neuronalen Netzes soll durch Mapping von Landmarken und Berechnung des optischen Flusses geschehen [32] . Geplante Bewegungen werden in Positionskorrekturen umgerechnet und von der Low-Level-Flugsteuerung in Motorbefehle umgesetzt. ....	12
Abbildung 2: Gesamtansicht NeuroCopter .....	13
Abbildung 3: Nahansicht Käfig mit Copterelektronik .....	14
Abbildung 4: Plot Accelerometerdaten bei ungedämpfter Aufhängung .....	15
Abbildung 5: Elektronikplatte mit MoonGel-Dämpfung .....	16
Abbildung 6: Plot Accelerometerdaten bei gedämpfter Aufhängung .....	16
Abbildung 7: Sonar MaxBotix MB1240 und die Halterung am Copter .....	17
Abbildung 8: Grafik Kompass + GPS .....	18
Abbildung 9: Prototyp der Kunststoff-"Bank" mit Magnetometer, GPS-Sensor und WLAN-Antenne .....	19
Abbildung 10: PS3Eye und PX4 FLOW Kamera .....	20
Abbildung 11: IGEP und APM-Flugcontroller .....	21
Abbildung 12: Konventionell-Neuromorphe Steuerarchitektur [32] .....	22
Abbildung 13: Navigationssysteme Vicon/GPS/Optisch .....	23
Abbildung 14: Winkelangaben und Rotationsachsen (links) bzw North-East-Down Konvention (rechts) im Bereich der Luftfahrt (Quelle: Wikipedia) .....	28
Abbildung 15: Rotation des Koordinatensystems mit Rotationsmatrix R [35] .....	29
Abbildung 16: Schachbrettmuster mit durch findChessboardCorners() detektierten internen Ecken .....	33
Abbildung 17: Binarisiertes Bild des blauen Papierkorbs mit Schwellwerten für den HSV-Farbraum .....	35
Abbildung 18: Tuning-Fenster mit Reglern für die Bestimmung der Detektionsparameter .....	38
Abbildung 19: Kamerabild mit erkannter künstlicher Landmarke .....	38
Abbildung 20: Testsetup für die Messung in Szenario 2 .....	44