Bachelorarbeit am Institut für Informatik der Freien Universität Berlin

Arbeitsgruppe Intelligente Systeme und Robotik - Biorobotics Lab

Bestimmung der 6D-Eigenbewegung eines fliegenden Roboters anhand von monokularen Messungen des optischen Flusses

Benjamin Wild

Betreuer: Dr. Tim Landgraf

Gutachter: Prof. Dr. Raúl Rojas, Dr. Hamid Mobalegh

21. Februar 2014

Eidesstattliche Erklärung

Ich versichere hiermit an Eides statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder Ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremdenA rbeiten sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

Benjamin Wild

21. Februar 2014

Inhaltsverzeichnis

1 Einleitung

In dieser Arbeit, die im Rahmen des Projektes NeuroCopter entstanden ist, wird ein Verfahren vorgestellt, mit dem sich die Eigenbewegung eines fliegenden Roboters anhand des optischen Flusses rekonstruieren lässt. Als Sensor wird eine handelsübliche Kamera verwendet. Im Gegensatz zu den meisten ähnlichen Verfahren in der Robotik wird dabei vollständig auf weitere Sensoren wie Gyroskope oder Accelerometer verzichtet.
Das Verfahren verwendet ein künstliches neuronales Netz und bildet die Grundlage für die Implementierung von neuromorphen Navigationsmodellen im Projekt NeuroCopter.

1.1 Motivation

1.1.1 Projekt NeuroCopter

Abbildung 1: NeuroCopter im Flug

NeuroCopter ist ein Gemeinschaftsprojekt des Biorobotics Lab und der Neuroinformatikgruppe an der Freien Universität Berlin. Ziel dieses interdisziplinären Projektes ist es, einen autonom fliegenden Roboter mit einer biologisch inspirierten, neuromorphen Gehirnsimulation zu kontrollieren. Dabei sollen Modelle für komplexe, abstrakte Aufgaben wie Lernen, Gedächtnis und Navigation entwickelt und getestet werden. Diese Modelle sollen ausgehend von den behavioralen und neuronalen Kenntnissen über Honigbienen entwickelt werden. In Kombination oder ergänzend zu funktionalen Modellen sollen biologisch akkurate spikende neuronale Netzwerke entwickelt werden, welche klar abgrenzte Teilprobleme lösen und sich später zu neuromorphen

Strukturen zusammenfügen lassen. In Abbildung 2 ist ein Konzept von einem solchen Modell abgebildet.

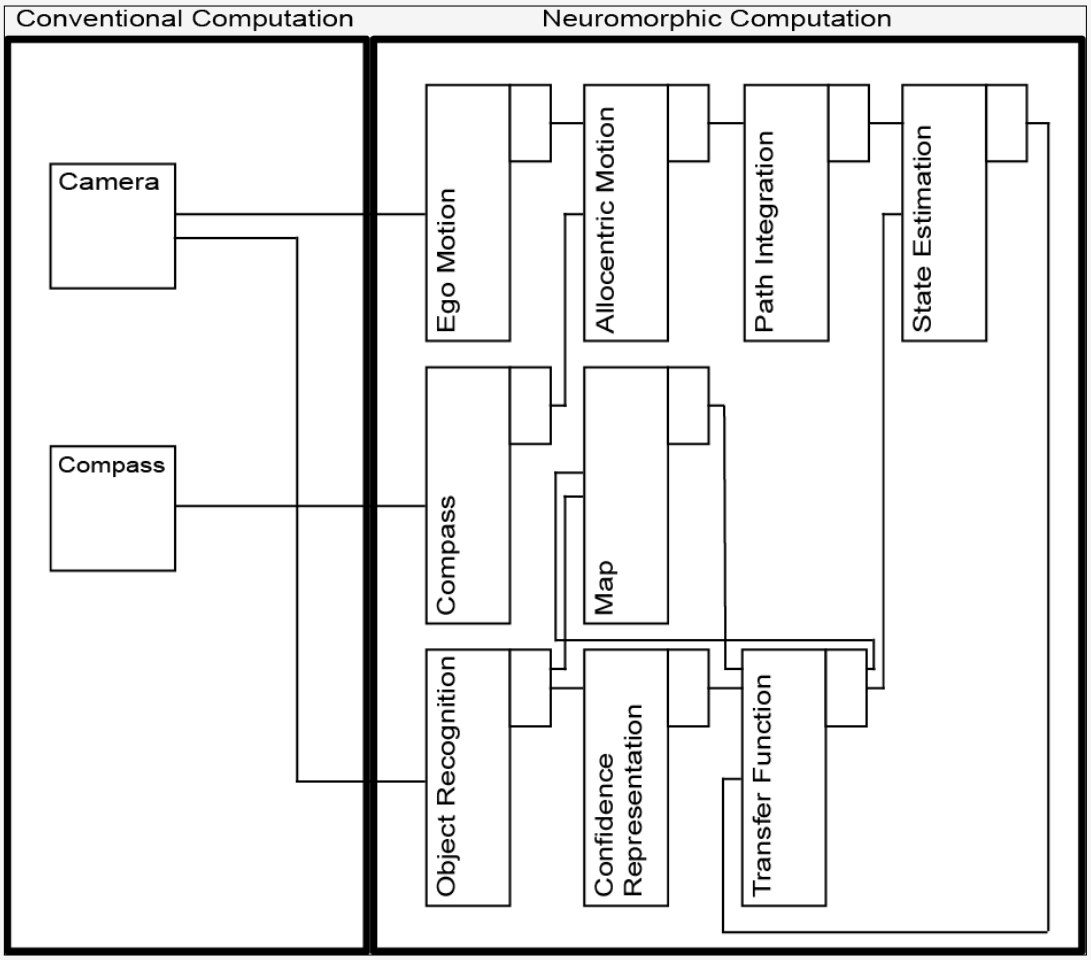

Abbildung 2: Ein Navigationsmodell im Projekt NeuroCopter [LWL13a]. Der aktuelle Zustand wird anhand einer Transferfunktion aus einer kamerabasierten Eigenbewegungsbestimmung und einem Modell der Umgebung ermittelt. Dabei hat die gespeicherte Position in dem Umgebungsmodell einen umso höheren Einfluss auf die Einschätzung des aktuellen Zustandes, je höher die Konfidenz in die gespeicherten Positionen von momentan erkannten Landmarken bzw. je höher die erwartete Ungenauigkeit der integrierten Eigenbewegung ist. Die aufgeführten Module sollen dabei zunächst unabhängig von den anderen Komponenten entwickelt und evaluiert werden.

Diese neuromorphen Modelle sollen einen tieferen Einblick in die unter anderem von Honigbienen verwendeten kognitiven Prozesse liefern. Honigbienen sind insbesondere für ihre robusten Navigationsfähigkeiten bekannt und werden in diesem Kontext seit Jahrzehnten untersucht. Durch die Verwendung von spikenden neuronalen Netzwerken eröffnet sich mittelfristig die

Möglichkeit, energieeffiziente neuromorphe Chips anstelle von traditionellen Prozessoren mit einer Von-Neumann-Architektur zu verwenden. Heutzutage ist es noch nicht möglich, komplexe spikende neuronale Netze auf analoger neuromorpher Hardware an Bord einer fliegenden Drohne einzusetzen. Aufgrund der momentan schnell voranschreitenden Entwicklungen in diesem Bereich (sowohl auf akademischer als auch industrieller Seite) ist jedoch zu vermuten, dass die Verwendung von neuromorphen Chips in der Robotik in absehbarer Zeit praktikabel wird [ILBH+11, PGJ+13, PZ11].

Die im Projekt NeuroCopter erstellten Modelle werden zunächst in einer Computersimulation getestet. Dadurch können sie mit virtuellen sensorischen Daten trainiert und getestet werden, ohne dabei die Hardware der Drohne zu gefährden. Eine so erstellte „Hirnsimulation" kann dann anschließend auf den Roboter übertragen und an echte sensorische Daten angepasst werden.

Technische Details Bei der Drohne handelt es sich um einen in dem Projekt eigens angefertigten Quadrocopter. Als Gestell werden vier Kohlefaserrohre verwendet, die mit einer aus Aluminium gefrästen Querverbindung befestigt sind. An den Enden eines jeden Rohres ist ein bürstenloser Gleichstrommotor (AC2836-358) angebracht. Die Motoren werden mit einer elektronischen Drehzahlregelung (jDrones AC20) angesteuert, welche mit einem kommerziell erwerblichen APM-Board[1] verbunden ist. Als Sensoren werden ein GPS-Gerät (Mtek GTPA010), Inertialsensoren sowie ein Barometer und Sonar verwendet. Das APM kann zur manuellen Kontrolle über einen 2,4GHz 8-Kanal-Funkempfänger (Turnigy 9X8CV2) mit einer Fernbedienung (Turnigy TGY 9X) angesteuert werden. Zusätzlich dazu ist das APM mit einem Cortex-A8 basierten PC (IGEP v2 DM3730) über eine serielle Schnittstelle verbunden, auf dem ein Embedded-Linux läuft. Auf dem IGEP läuft eine Software, welche UDP Pakete, die von einer Bodenstation über WLAN gesendet werden, entweder an das APM oder an andere Programme, die simultan ausgeführt werden, weiterleitet. Ebenso kann das APM über die serielle Schnittstelle Nachrichten an den Nachrichtenverteiler auf dem IGEP schicken, die dann entsprechend weitergeleitet oder verarbeitet werden. Eine handelsübliche Webcam (Logitech Pro 9000) ist über USB mit dem IGEP verbunden. Die Bodenstation ist dafür vorgesehen, abstrakte Kommandos (Start, Landung etc.) an die Drohne zu senden. Des weiteren werden bei aufwändigen Berechnungen Sensordaten an die Bodenstation gesendet und dort verarbeitet. Die Drohne wiegt 1340g und kann eine Nutzlast von ungefähr 1.5kg tragen.

[1] ArduPilotMega

1.1.2 Bestimmung der Eigenbewegung

Notwendige Bedingung für die meisten Navigationsmodelle ist ein hinreichend robustes Verfahren zur Bestimmung der Eigenbewegung, welches anschließend von dem Navigationsmechanismus, möglicherweise in Kombination mit weiteren sensorischen Daten, verwendet werden kann. In dem hier vorgestellten Ansatz wird dazu ausschließlich eine einzelne Kamera verwendet. Es ist bekannt, dass Honigbienen den optischen Fluss unter anderem verwenden, um ihre Fluggeschwindigkeit zu regulieren und Distanzen einzuschätzen [Sri03, PRRF11]. Es ist daher denkbar, dass sie den optischen Fluss auch für die Pfadintegration verwenden.
Das hier vorgestellte Verfahren soll folglich die Grundlagen für einen Navigationsmechanismus im Projekt NeuroCopter schaffen und auf einer funktionalen Ebene zeigen, dass der optische Fluss für die von Honigbienen demonstrierten Pfadintegrationsfähigkeiten ausreichend ist. In der Robotik könnte die Methode ergänzend oder als Ersatz für herkömmliche Ansätze zur Bestimmung der Eigenbewegung (zum Beispiel die Verwendung von Inertialsensoren) eingesetzt werden.

2 Bezug zu vorherigen Arbeiten

Robotik In der Robotik sind verschiedene Verfahren zur Abschätzung der Eigenbewegung gebräuchlich. SLAM[2]-Algorithmen verwenden in der Regel Inertialsensoren und ein Bewegungsmodell, um die Bewegung des Roboters zu bestimmen und im Zusammenhang mit weiteren sensorischen Daten eine Karte der Umgebung zu erstellen [DwB06]. Weiterhin ist die Verwendung von (biologisch nicht plausiblen) Sensoren wie GPS oder LiDAR üblich.
Auch visuelle Informationen können zur Abschätzung der Eigenbewegung verwendet werden. Einen Überblick über solche Ansätze bietet [Pan08]. Die Verwendung des optischen Flusses bei UAVs[3] wird unter anderem in [Wei12] und [GBR12] behandelt.
Frühere Verfahren zur Bestimmung der Eigenbewegung mit Hilfe des optischen Flusses erzielen gute Ergebnisse, funktionieren aber nur in Kombination mit weiteren Sensoren (in der Regel Gyrometer) oder unter bestimmten Einschränkungen (zum Beispiel bei einer konstanten Flughöhe und keiner Rotation der Kamera) [GBR12, HM13].

Biologie Die Navigationsfähigkeiten der Honigbiene werden seit Jahrzehnten untersucht [CC02, Men12]. Durch Integration ihrer Eigenbewegung sind Bienen in der Lage, eine robuste Abschätzung ihrer Position in Relation zum Bienenstock zu bestimmen [WS03]. Es wird vermutet, dass Honigbienen bei ihren Erkundungsflügen ein komplexes, kartenähnliches Modell ihrer Umgebung generieren [MGS$^+$05, MKH$^+$11, MLM$^+$12]. Es gibt bislang wenige Erkenntnisse über die konkreten neuronalen Mechanismen hinter diesen Fähigkeiten.
Honigbienen verwenden visuelle Informationen (insbesondere den optischen Fluss), um eine Vielzahl von Navigations- und Flugproblemen zu lösen [SZLC96, EZST01, ERN12]. Es ist daher denkbar, dass sie den optischen Fluss (möglicherweise in Kombination mit weiteren sensorischen Daten) auch nutzen, um ihre Eigenbewegung im Flug abzuschätzen. Ein detailliertes Modell von einem solchem Mechanismus wurde jedoch bisher weder auf neuronaler noch auf funktionaler Ebene veröffentlicht.

[2] Simulataneous Localization and Mapping
[3] Unmanned aerial vehicle

3 Umsetzung

In diesem Kapitel werden der Ansatz des Verfahrens sowie die technischen Details der Implementierung vorgestellt.

3.1 Ansatz

Eine Kamera sei an einem fliegenden Roboter befestigt und perpendikular zum Boden ausgerichtet. Gegeben sei ein Punkt p in dem Referenzsystem der Kamera sowie die translationale (v) und rotationale Komponente (ω) der relativen Bewegung zwischen der Kamera und dem Punkt p.

$$
\begin{aligned}
p &= \begin{pmatrix} x \\ y \\ z \end{pmatrix} \\
\omega &= \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \\
v &= \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix}
\end{aligned}
$$

Der projizierte Punkt $\tilde{p}$ auf der Bildebene zu einem Zeitpunkt ist dann gegeben durch:

$$
\tilde{p} = f\frac{p}{z}
$$

Die Gesamtbewegung zwischen der Kamera und dem Punkt p ergibt sich aus der translationalen und der rotationalen Komponente der Bewegung [HS81].

$$
V = -v - \omega \times p
$$

Durch Ableitung nach der Zeit kommt man zu der Relation zwischen dem optischen Fluss (der Ableitung der Projektion) und der Bewegung.

$$\frac{flow}{\Delta t} = \Delta\tilde{p} \approx \frac{zV - V_z p}{z^2}$$

Wie in [HM13] gezeigt lässt sich unter der Annahme, dass sowohl die rotationale Komponente der Bewegung als auch die vertikale Bewegung null ist, die Bewegung direkt aus dem optischen Fluss berechnen.

$$\begin{aligned} \omega &= 0 \\ v_z &= 0 \\ v &= \Delta\tilde{p}\frac{z}{f} \end{aligned}$$

Ziel dieser Arbeit war es, ein Verfahren zu finden, welches aus dem optischen Fluss alleine sowohl die translationale als auch die rotationale Komponente der Bewegung abschätzen kann.

Es ist intuitiv ersichtlich, dass sich bei einer Bewegung, bei der $\omega \neq 0$ oder $v_z \neq 0$ ist, der optische Fluss von der Position des Punktes p auf der Bildebene abhängt. Eine Rotation um die z-Achse (yaw) bei einer perpendikular zum Boden gerichteten Kamera sollte zum Beispiel zu einem optischen Fluss führen, bei dem die Vektoren auf den entgegengesetzten Seiten der Bildebene um 180° rotiert sind. Dabei sollte die konkrete Ausrichtung und Länge der Vektoren abhängig von der Richtung und Geschwindigkeit der Rotation sein.

Des weiteren sollte eine Bewegung, bei der nur $v_z \neq 0$ ist, zu einem optischen Fluss führen, bei dem alle Vektoren je nach der Ausrichtung von v_z entweder nach innen (in Richtung des Zentrums der Bildebene) oder nach außen zeigen.

Es soll nun ermittelt werden, inwiefern eine (möglicherweise hochgradig nichtlineare) Abbildung von gleichmäßig auf der Bildebene verteilten Flussvektoren auf die sechsdimensionale Eigenbewegung der Kamera existiert.

$$\begin{aligned} \Delta\tilde{P} &= \begin{pmatrix} \Delta\tilde{p}_x^1 & \Delta\tilde{p}_y^1 \\ \dots & \dots \\ \Delta\tilde{p}_x^n & \Delta\tilde{p}_y^n \end{pmatrix} \\ \begin{pmatrix} v \\ \omega \end{pmatrix} &\approx f(\Delta\tilde{P}) \end{aligned}$$

Wobei n die Gesamtanzahl von verwendeten Vektoren ist und f gesucht wird. Ausgehend davon ließe sich dann die Bewegung der Kamera berechnen.

$$s \approx \sum_{t=0}^{T} f(\Delta\tilde{P}(t))$$

Dabei stellt sich die Frage, inwiefern es eine eindeutige Abbildung von dem optischen Fluss auf die sechsdimensionale Eigenbewegung der Kamera gibt. Es scheint intuitiv ersichtlich, dass es durch den Verlust der Tiefeninformation bei der Projektion durch die Kamera dazu kommen kann, dass unterschiedliche Bewegungen zu einer ähnlichen oder gar gleichen Kombination von Flussvektoren auf der Bildebene führen können.
Ausgehend davon war der erste Schwerpunkt dieser Arbeit, zu bestimmen, unter welchen Bedingungen unterschiedliche Bewegungen ähnliche Flussvektoren erzeugen. Anschließend daran ist es möglich zu beurteilen, inwiefern solche eventuell vorhandenen Paare von Bewegungen in einem realistischen Flugszenario relevant sind und ob der Fehler, der dadurch bei der Abschätzung der Eigenbewegung auftreten muss, in einem akzeptablen Rahmen liegt.
Unter der Voraussetzung, dass der erwartete (minimale) Fehler der Abbildung nicht zu groß ist, muss geklärt finden, wie eine solche Abbildung gefunden werden kann.

Simulation des optischen Flusses In einer simulierten 3D-Welt lässt sich der erwartete optische Fluss bei der Bewegung einer Kamera ermitteln. Mit Hilfe einer solchen Simulation ließe sich nun der optische Fluss für alle möglichen Bewegungen (alle Kombinationen der Bewegungskomponenten mit uniform verteiltem Abstand) berechnen. Ausgehend von diesen Daten könnte bestimmt werden, welche Paare von maximal unterschiedlichen Bewegungen zu minimal verschiedenen Paaren von Flussvektorfeldern führen. In Abhängigkeit von der Ähnlichkeit der Flussvektorfelder und der praktischen Relevanz der Bewegungen, wie vorher erläutert, ließe sich nun die erwartete Praxistauglichkeit des Verfahrens evaluieren.

Ermittlung der Abbildung Es ist zu erwarten, dass eine solche Abbildung nicht trivial zu ermitteln ist. Einerseits steigt die Genauigkeit vermutlich nur bis zu einer gewissen Grenze (unter anderem aufgrund von Rauschen und Ausreißern bei der Ermittlung des optischen Flusses) mit der Anzahl von evaluierten Flussvektoren. Andererseits ist es wahrscheinlich, dass die Abbildung hochgradig nichtlinear und nicht eindeutig ist. Des weiteren ist im Hinblick auf realistische Anwendungsfälle des Verfahrens (Implementierung auf leistungsschwacher Embedded-Hardware) auch auf die Geschwindigkeit bei der Abschätzung der Eigenbewegung

zu achten. Es ist also ein Verfahren gesucht, welches automatisch eine solche Abbildung mit einem möglichst minimalen Fehler bestimmt und gleichzeitig ausreichend schnell angewandt werden kann.

3.2 Implementierung

3.2.1 Simulation des optischen Flusses

Generierung der 3D-Welt Die Simulation der 3D-Welt wurde in Matlab implementiert. Die Landschaft wird durch eine zweidimensionale Matrix repräsentiert, wobei die Indizes die (gleichmäßig verteilten) x- und y-Koordinaten der Landschaft und die Werte der Matrix die z-Koordinaten darstellen. Zunächst wurde die Matrix mit uniform verteilten Zufallswerten initialisiert. Anschließend wurden die Werte mit einem zweidimensionalen Tiefpassfilter geglättet. Daraufhin wurden die Werte mit (erneut uniform verteilten) Zufallswerten leicht verrauscht. Die sich ergebende Landschaftsmatrix wurde nun noch einmal geringfügig mit einem Tiefpassfilter geglättet. Eine mit diesem Verfahren erzeugte Landschaft ist in Abbildung 3 dargestellt.

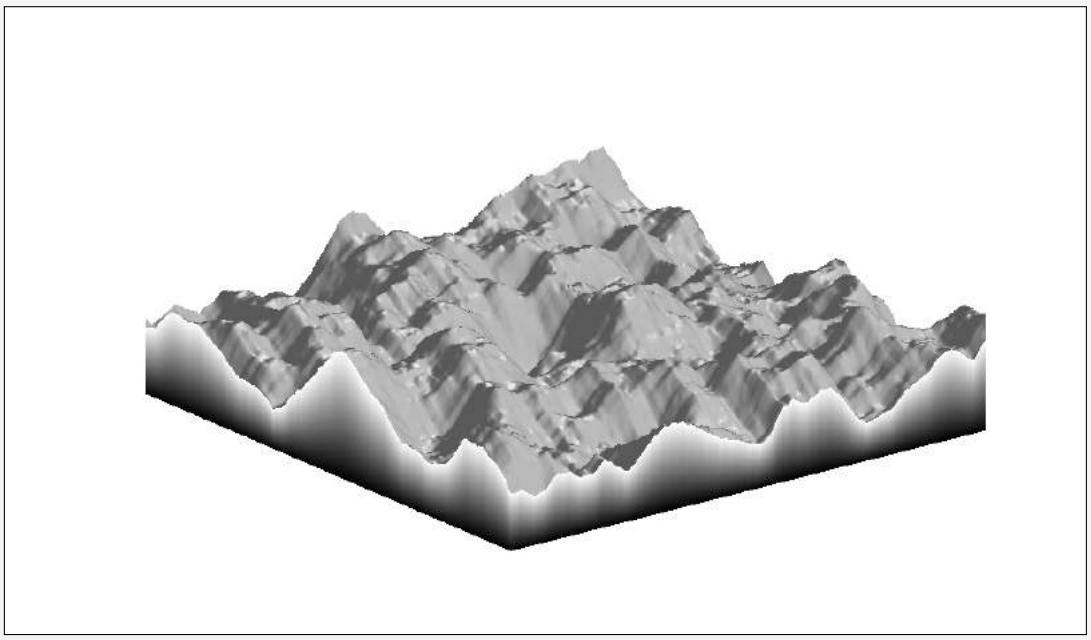

Abbildung 3: Eine algorithmisch generierte Punktlandschaft

Aufwendigere Verfahren zur Generierung der Landschaften (zum Beispiel mit Perlin noise) wurden in Betracht gezogen, aber aufgrund der zufriedenstellenden Ergebnisse der oben beschriebenen Methode nicht verwendet.

Simulation des optischen Flusses In der simulierten Welt wurde nun eine virtuelle Kamera platziert. Für jede Position der Kamera wurde die Landschaftsmatrix in das Koordinatensystem der Kamera projiziert und anschließend auf die Bildebene abgebildet. Aus den projizierten Punkten der Landschaft zwischen Paaren von Kamerapositionen wurden dann die resultierenden Flussvektoren für die Bewegung zwischen den Positionspaaren abgeleitet. Da die Indizes der Punkte vor und nach der Projektion bekannt waren, konnte diese Zuordnung ohne die normalerweise auftretenden Fehler bei der Bestimmung des optischen Flusses bei nichtsimulierten Bilddaten vorgenommen werden.

In Hinblick auf die Performance und Generalisierung des später implementierten Verfahrens wurde das berechnete Flussvektorfeld nun in gleichmäßig verteilte Gridzellen unterteilt. In jeder der Zellen wurde der Mittelwert der darin enthaltenen Vektoren berechnet. Bei einem Grid mit n Elementen ergibt sich somit ein $n \times 2$ großer Merkmalsvektor, der die x- und y-Komponenten von jedem gemittelten Flussvektor von den Gridzellen enthält. Durch spätere Experimente wurde bestimmt, dass ein Grid mit 5×5 Elementen zu guten Ergebnissen führt. Ein wesentlich feiner aufgelöstes Grid würde dazu führen, dass die gemittelten Vektoren übermäßig stark durch Ausreißer bei der Berechnung des optischen Flusses bei realen Kamerabildern verfälscht werden können. Des weiteren würde ein größerer Merkmalsvektor die Geschwindigkeit des Verfahrens verringern, wodurch es nicht mehr möglich wäre, es auf Embedded-Hardware in Echtzeit zu verwenden. Bei einem zu kleinen Grid ist zu wenig Information in dem Vektor enthalten, um zufriedenstellende Ergebnisse zu erreichen.

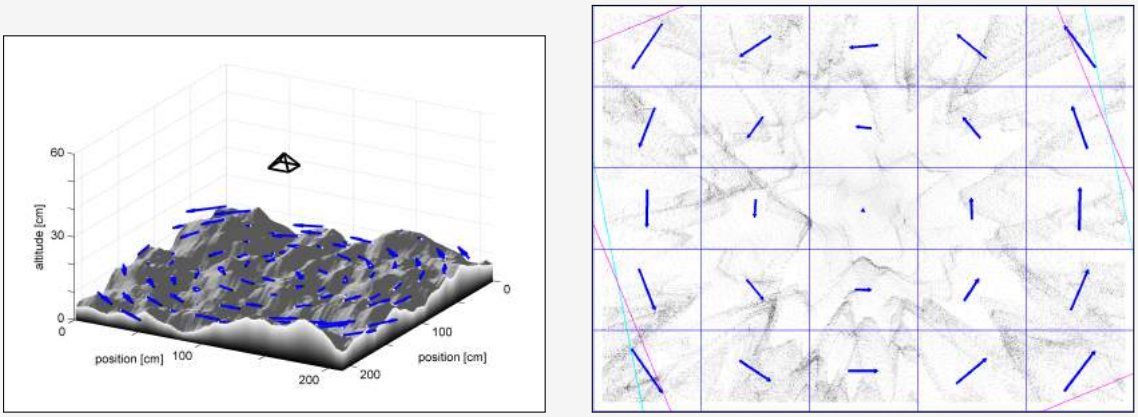

Abbildung 4: Die Rotation der Kamera (links) führt zu einer dreidimensionalen Bewegung der Landschaftspunkte im Koordinatensystem der Kamera. Durch die Projektion der Bildpunkte auf die Bildebene der Kamera lässt sich der optische Fluss bei einer Bewegung in der Simulation ermitteln. Zur späteren Verarbeitung werden die Flussvektoren auf einem Grid mit 5×5 Zellen gemittelt (rechts).

Ein Beispiel für den entstehenden optischen Fluss bei der Rotation der virtuellen Kamera über der Landschaft ist in Abbildung 4 auf der vorherigen Seite zu sehen. Weitere Beispiele sind im Anhang abgebildet.

3.2.2 Bestimmung des optischen Flusses bei echten Kamerabildern

Auf den Kamerabildern werden Bildmerkmale bestimmt [ST94]. Ausgehend davon wird der optische Fluss mit der pyramidalen Version der Lucas-Kanade Methode abgeschätzt [LK81]. Die so ermittelten Vektoren werden wie in der Simulation auf einem 5×5 Grid gemittelt. Abbildung 5 zeigt eine Beispielaufnahme von einem Testflug mit den erkannten Features und den gemittelten Flussvektoren.

Es ist anzumerken, dass aufgrund der begrenzten Rechenleistung auf der im Projekt Neuro-Copter verwendeten CPU die Bestimmung des optischen Flusses nicht mit der gewünschten Qualität in Echtzeit vorgenommen werden konnte. Lösungsansätze für dieses Problem werden in Abschnitt 5 auf Seite 20 diskutiert.

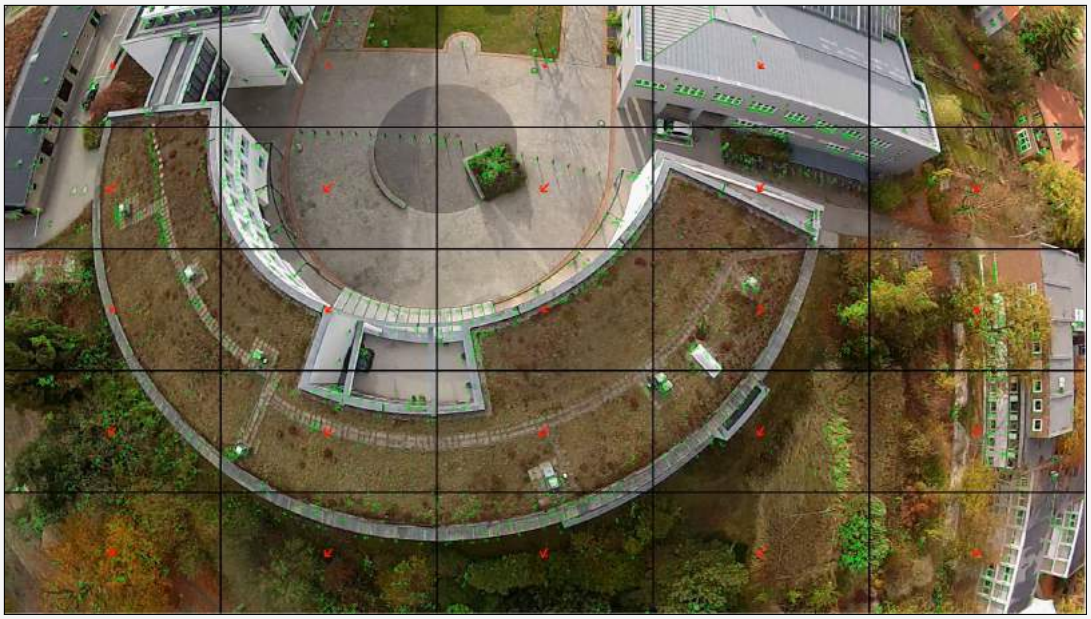

Abbildung 5: Ein Testflug über dem Institut für Informatik der Freien Universität Berlin. Die erkannten Bildmerkmale sind mit grünen Punkten gekennzeichnet, die gemittelten Flussvektoren werden durch rote Pfeile dargestellt.

3.2.3 Lernverfahren

Erzeugung der Trainingsdaten Die Trainingsdaten wurden mit der in Abschnitt 3.2.1 auf Seite 9 vorgestellten Simulation generiert. Dabei wurden alle Kombinationen von plausiblen Bewegungen mit gleichmäßig verteilten Werten für die z-Komponente, also die „Flughöhe“ der Kamera, generiert. Als plausibel galten hierbei Bewegungen, die ein physikalisch realistisches Maximum in den Einzelkomponenten der Bewegung nicht überschritten.

Insgesamt wurden so über 10.000 Eigenbewegungen mit den dazugehörigen Merkmalsvektoren generiert. Ein solcher Vektor enthält die x- und y-Komponenten der auf den 25 Gridzellen berechneten gemittelten Flussvektoren, also insgesamt 50 Werte.

Training Bei der Suche nach einer geeigneten Abbildung, welche die Abweichung der berechneten Eigenbewegungen und den in den Trainingsdaten enthaltenen Bewegungen möglichst stark minimiert und außerdem gut auf weitere Daten generalisiert (also nicht zu stark auf die Trainingsdaten angepasst ist), bieten sich verschiedene Verfahren aus dem Bereich der Mustererkennung an.

Ein weitverbreitetes Lernverfahren sind mehrschichtige neuronale Netzwerke (ANN[4]), deren Gewichte so angepasst werden, dass die Abweichung zwischen den Ausgabewerten und den gewünschten Ausgaben möglichst minimal wird. In [Cyb89] und anderen Veröffentlichungen wurde gezeigt, dass ein mehrschichtiges ANN unter einigen (relativ schwachen) Voraussetzungen ein universeller Funktionsapproximator ist. Mit modernen Trainingsverfahren wie der Levenberg-Marquardt Methode ([Mar63]) lassen sich auch bei großen Datensätzen und vielen Neuronen schnell gute Trainingsergebnisse erreichen. Die Anwendung von einem einmal trainierten Netzwerk ist performant, da es sich dabei hauptsächlich um eine Reihe von Matrixadditionen und Multiplikationen handelt.

Im Kontext des Projektes NeuroCopter bot sich die Wahl von einem solchen neuronalen Netz außerdem an, da dadurch möglicherweise die Portierung des Verfahrens auf ein (biologisch akkurateres) spikendes neuronales Netzwerk (SNN) vereinfacht wird [SGGW11].

Die Eingabewerte (die gemittelten Flussvektoren) wurden vor dem Training normalisiert. Als Fehlerfunktion wurde die mittlere quadratische Abweichung (MSE) verwendet. Das Netzwerk wurde mit der Levenberg-Marquardt Methode trainiert [Mar63]. Die Größe des Hidden Layers wurde experimentell ermittelt, um einen möglichst guten Kompromiss zwischen Generalisierung und Genauigkeit des Verfahrens zu erreichen. Ein solcher Kompromiss wurde bei 250

[4] Artificial neural network

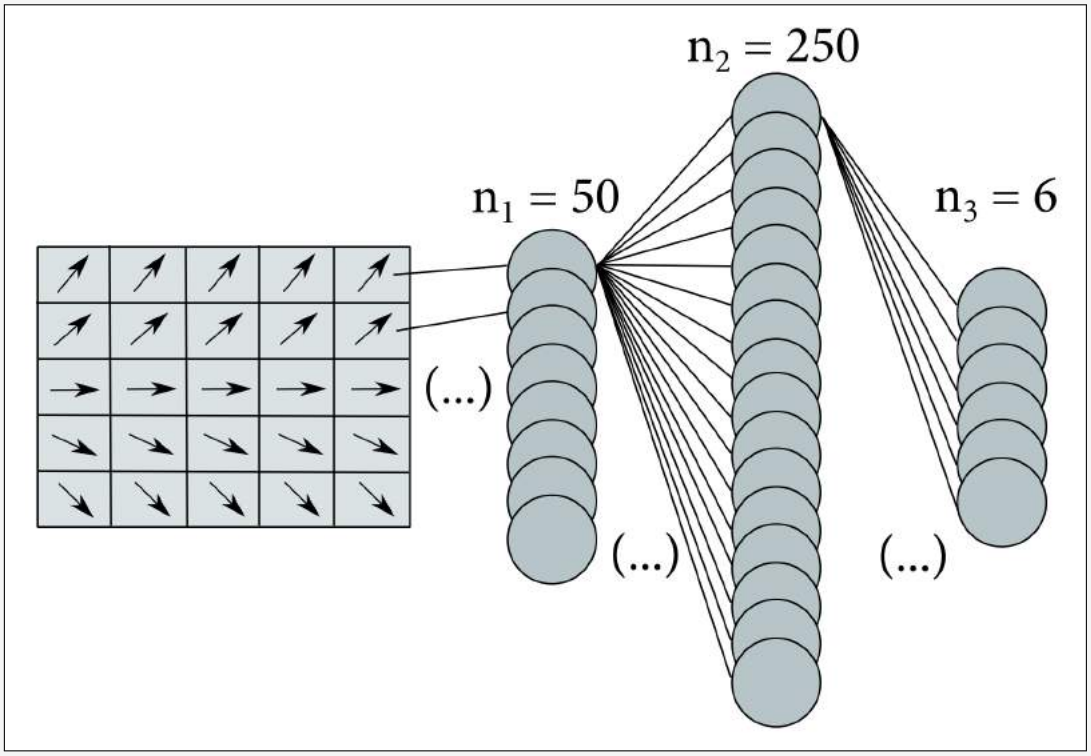

Abbildung 6: Schema des verwendeten neuronalen Netzes. Die x- und y-Komponente von jedem vorher ermittelten Flussvektor wird in das Netzwerk eingegeben. Mit der Levenberg-Marquardt Methode werden die Gewichte zwischen den drei Schichten iterativ so angepasst, dass die Abweichungen zwischen den Trainingsdaten (der Eigenbewegung der Drohne) und den Ausgaben des Netzwerkes möglichst minimal werden. Durch Integration der Ausgaben von einem solchen Netzwerk lässt sich die Flugbahn der Drohne rekonstruieren.

Neuronen gefunden. Um Overfitting zu vermeiden, wurden die Ergebnisse mit einer 6-Fold Cross-Validation überprüft [Koh95].

Die Trainingszeit ist abhängig von den (zufällig gewählten) Startwerten für die Gewichte des Netzwerks und damit von der A nzahl an Iterationsschritten, die benötigt werden, bis ein (eventuell lokales) Minimum gefunden wird. Das Training wurde abgebrochen, sobald nach 5 aufeinanderfolgenden Iterationsschritten eine Verschlechterung der Ergebnisse bei einem separat ausgewerteten Testdatensatz aufgetreten ist. Die in dem Testdatensatz enthaltenen Merkmalsvektoren waren dabei nicht in den Trainingsdaten vorhanden. Der Trainingsvorgang wurde fünf Mal mit verschiedenen Startgewichten ausgeführt, um ein möglichst gutes (lokales) Minimum zu finden. Die Trainingszeit betrug dabei auf einem PC mit einem modernen Intel Core i5 Prozessor in jedem Fall deutlich weniger als eine Stunde.

4 Ergebnisse

4.1 Analyse der Simulationsdaten

Die mit der in Abschnitt 3.2.1 auf Seite 9 beschriebenen Simulation erzeugten Daten wurden nun zunächst quantitativ untersucht. Dabei sollte die Frage geklärt werden, inwiefern es unterschiedliche Bewegungen der virtuellen Kamera im 6D-Raum gibt, die zu einem ähnlichen optischen Fluss führen.

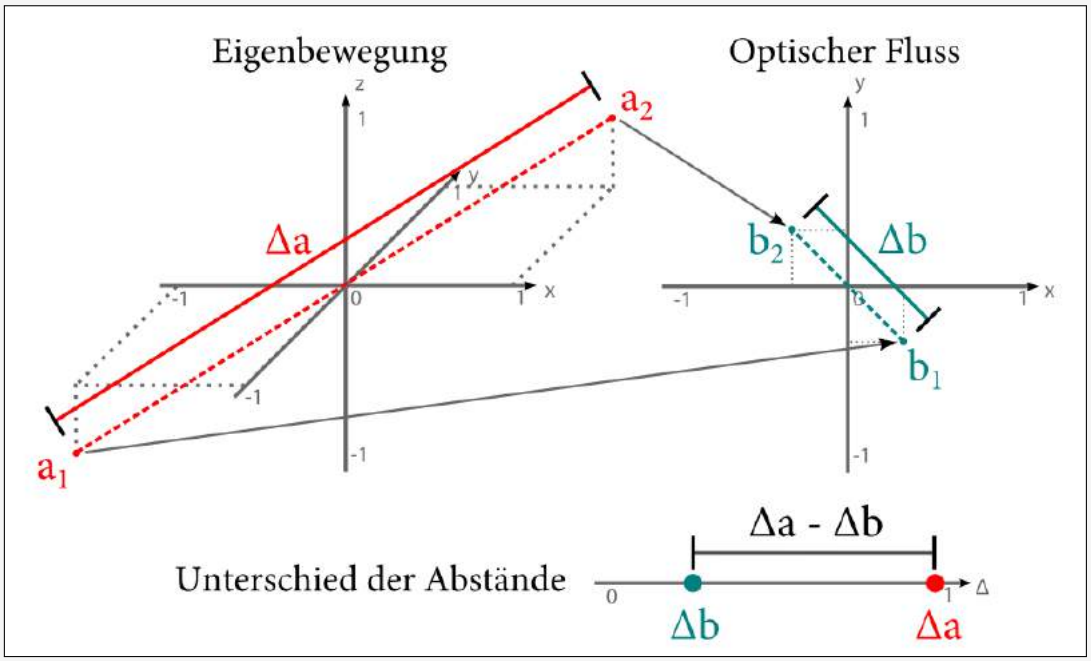

Abbildung 7: Der euklidische Abstand von zwei Bewegungen (hier vereinfacht für den dreidimensionalen Fall dargestellt) wird ermittelt (Δa). Anschließend wird der Abstand von den durch die Bewegungen verursachten (eigentlich 50-dimensionalen) Merkmalsvektoren berechnet (Δb). Die Distanz zwischen den beiden Abständen ($\Delta b - \Delta a$) ist ein Maß für die Unterschiede von den Ähnlichkeiten der beiden Bewegungen und Merkmalsvektoren zueinander. Mit diesem Maß lassen sich Paare von unterschiedlichen Bewegungen ermitteln, die zu einem ähnlichen optischen Fluss führen.

Allen möglichen Paaren von generierten Bewegungen $(\vec{a}_i, \vec{a}_j)$ wurden zunächst die entsprechenden Paare von resultierenden Merkmalsvektoren $(\vec{b}_i, \vec{b}_j)$ zugeordnet. Die Merkmalsvektoren enthalten die 25 gemittelten Flussvektoren in den Gridzellen. Die Daten wurden anschließend in ihren jeweiligen Vektorräumen auf Einheitslänge normiert. Mit der euklidischen Distanz wurde nun die Ähnlichkeit der Bewegungen in jedem Bewegungspaar und der Merkmalsvektoren in

jedem Merkmalspaar ermittelt. Eine geringe euklidischeD istanz zwischen zwei Bewegungsvektoren signalisiert eine hohe Ähnlichkeit der beiden Bewegungen. Alle so ermittelten Distanzen wurden nun erneut normalisiert, so dass jeweils in beiden Vektorräumen die maximale Distanz den Wert 1 beträgt. Anschließend wurden die Abstände zwischen den euklidischen Distanzen der Bewegungen und den zugehörigen euklidischen Distanzen der Merkmalsvektoren ermittelt. Es wurde also berechnet, in welchem Maße sich die Ähnlichkeit der Paare von Bewegungen zueinander von der Ähnlichkeit der Merkmalsvektoren zueinander unterscheidet. Ein hoher Abstand in diesem Maß bedeutet, dass ein Paar von Bewegungen mit einer hohen Ähnlichkeit ein Paar von Flussfeldern verursacht, welche eine sehr geringe Ähnlichkeit haben (oder umgekehrt).

Durch dieses Abstandsmaß wurden Paare von Bewegungen ermittelt, die zu einem ähnlichen optischen Fluss führen. Dies ist vor allem bei einigen komplexen Bewegungen, bei denen die Rotationskomponenten entgegen der Translationskomponenten wirken, der Fall. So führt zum Beispiel eine Vorwärtsbewegung der Kamera bei einer gleichzeitig stattfindenden positiv gerichteten Rotation auf der perpendikularen Achse zu zwei entgegengesetzt gerichteten Flussfeldern. In bestimmten Fällen (abhängig von der Flughöhe, und den Geschwindigkeiten) führt dies dazu, dass sich die Flussfelder nahezu aufheben. Es gibt also einige Fälle von sehr unterschiedlichen Bewegungen, die fast keinen optischen Fluss erzeugen (vergleichbar mit der Rotationskompensation der Iris bei Kopfbewegungen eines Menschen).

Solche Bewegungen sollten allerdings in realen Flugszenarien selten auftreten. So verursacht zum Beispiel eine Rotation eine Translationsbewegung auf der perpendikularen Achse der Rotation. Ein gleichzeitiges Auftreten der beiden Bewegungskomponenten sollte daher auf einen kurzen Zeitraum beschränkt sein.

Aus dieser Analyse lässt sich schlussfolgern, dass es (wie erwartet) nicht möglich ist, eine fehlerfreie Abbildung von dem optischen Fluss auf die sechsdimensionale Eigenbewegung der Kamera zu ermitteln. Ein solches Verfahren könnte dennoch zu einer Abschätzung der Eigenbewegung führen, die von einem darauf aufbauenden Navigationsverfahren genutzt werden kann, falls der Fehler nicht zu hoch ist. Ein gewisser Fehler tritt bei jeder Methode zur Bestimmung der Eigenbewegung auf. Beispielsweise akkumulieren die unter anderem in SLAM-Algorithmen oft verwendeten Inertialsensoren über die Zeit einen Messfehler (Sensordrift).

4.2 Rekonstruktion der Eigenbewegung in der Simulation

Wie in Abschnitt 3.2.2 auf Seite 11 beschrieben, wurde nun ein neuronales Netz zur Bestimmung der Eigenbewegung anhand der aus den Flussvektoren ermittelten Merkmalsvektoren trainiert.

Dies führte zunächst zu einer Abbildung mit einem hohen mittleren quadratischen Fehler. Als anschaulicher Test wurden Flussdaten für einen realistischen Spiralflug generiert. In Abbildung 8 ist zu sehen, dass ein mit diesem Netz rekonstruierter Flug einen hohen Fehler aufweist. Eine Ähnlichkeit mit der echten Flugbahn ist jedoch bereits deutlich erkennbar.

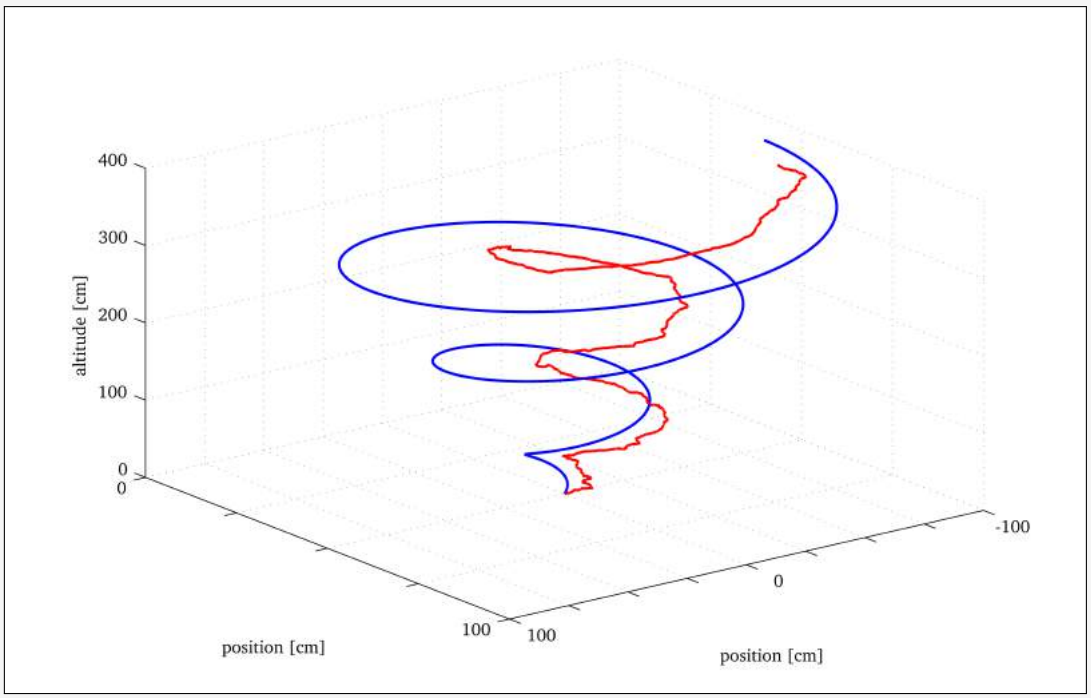

Abbildung 8: Fehlerbehaftete Rekonstruktion eines Spiralflugs mit einem Netzwerk, das nur die gemittelten Flussvektoren als Eingaben erhält (siehe Abbildung 6 auf Seite 13). Die von dem Netzwerk ermittelte Flugbahn entspricht nur in etwa der eigentlichen Trajektorie.

Durch weitere Analysen wurde ermittelt, dass der Fehler hauptsächlich ein Resultat der fehlenden Höheninformation in den Merkmalsvektoren ist. Es ist intuitiv ersichtlich, dass der optische Fluss bei einer Bewegung maßgeblich von der aktuellen Flughöhe abhängt.D ie Länge der entstehenden Vektoren ist dabei umgekehrt proportional zu der Höhe der Kamera.

Die Höheninformation kann entweder durch Integration der vorher ermittelten Bewegungen ode r durch einen externen Sensor als weitere Eingabe in das Netzwerk eingespeist werden. In den untersuchten Szenarien war die aus den Netzwerkausgaben integrierte Höheninformation ausreichend, um ein gutes Resultat zu erzielen. Dieses erweiterte Schema ist in Abbildung 9 auf der nächsten Seite abgebildet.

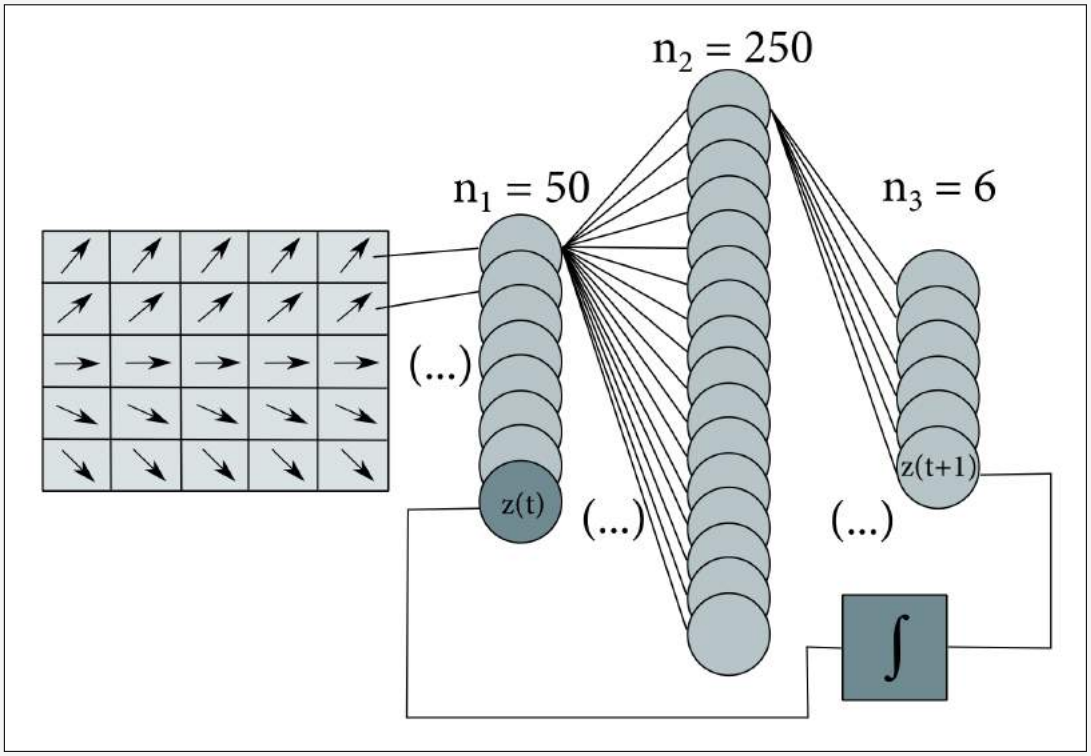

Abbildung 9: Neuronales Netz mit Flussvektoren und integrierter vertikaler Bewegung als Eingaben und der Eigenbewegung als Ausgabe. Durch die Verwendung der aktuellen Flughöhe als weiteres Merkmal konnte die Genauigkeit des Verfahrens deutlich verbessert werden.

In Abbildung 10 auf der nächsten Seite ist zu sehen, wie sich das Ergebnis des Verfahrens durch die Höheninformation verbessert. In der Simulation kann die originale Flugbahn nahezu fehlerfrei durch Integration der Netzwerkausgaben rekonstruiert werden.

4.3 Rekonstruktion der Eigenbewegung von echten Flugdaten

Anschließend wurde das Netzwerk verwendet, um einen Testflug mit dem NeuroCopter zu rekonstruieren. Ein Video von dem Testflug kann online unter [LWL+13b] eingesehen werden. Wie auch bei der Simulation wurden außer dem optischen Fluss keine weiteren Daten für die Bestimmung der Eigenbewegung verwendet. Die Höheninformation wurde aus den Ausgaben des Netzwerkes in den vorherigen Schritten integriert. Als Ground-Truth-Daten wurden die Sensoren des NeuroCopters verwendet. Die Intertialsensoren (Gyroskope und Accelerometer) wurden dabei mit einer Frequenz von 50Hz abgetastet, das GPS mit 5Hz. In Abbildung 11 auf Seite 19 ist das Ergebnis der Rekonstruktion zu dargestellt. Der Fehler ist (wie zu erwarten)

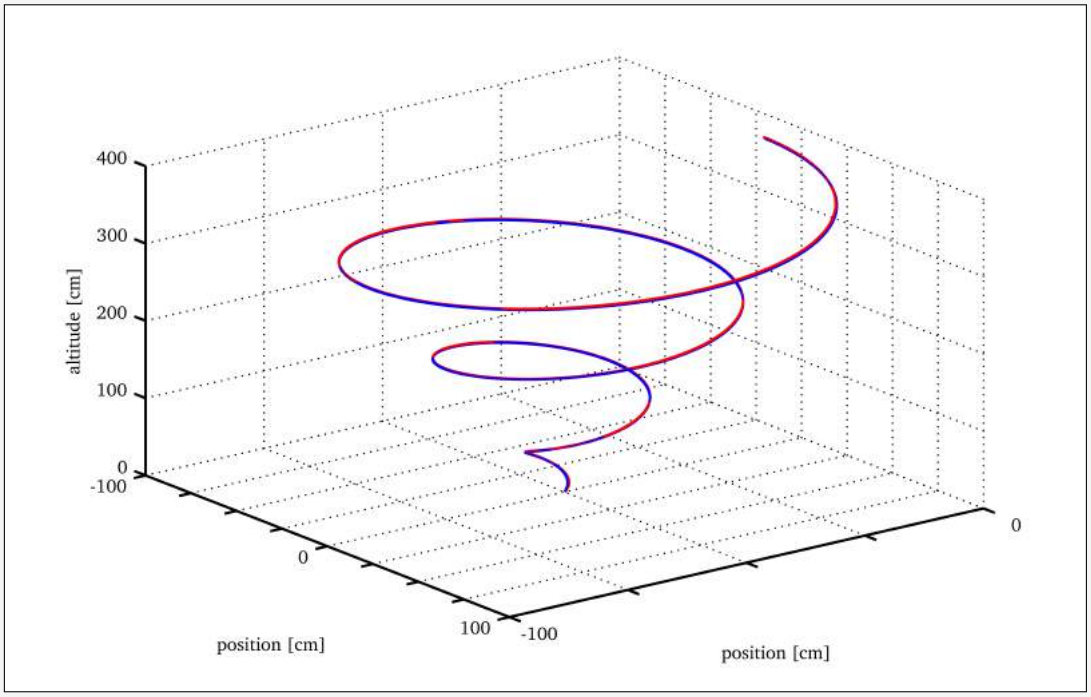

Abbildung 10: Eine Rekonstruktion des Spiralflugs mit dem Netzwerk aus Abbildung 9 auf der vorherigen Seite. Mit der Höheninformation als zusätzliches Merkmal lässt sich der Testflug in der Simulation nahezu fehlerfrei rekonstruieren.

höher als bei den simulierten Daten. Die rekonstruierte Flugbahn weist allerdings eine hohe Ähnlichkeit zu den GPS Daten auf. Es ist anzumerken, dass die GPS Daten nicht fehlerfrei sind. Eine exakte Quantisierung des Fehlers ist daher mit den gegebenen Ground-Truth-Daten nicht möglich. In Abbildung 12 auf der nächsten Seite ist der Abstand zwischen den GPS Daten und der rekonstruierten Trajektorie auf den Translationsachsen separat dargestellt.

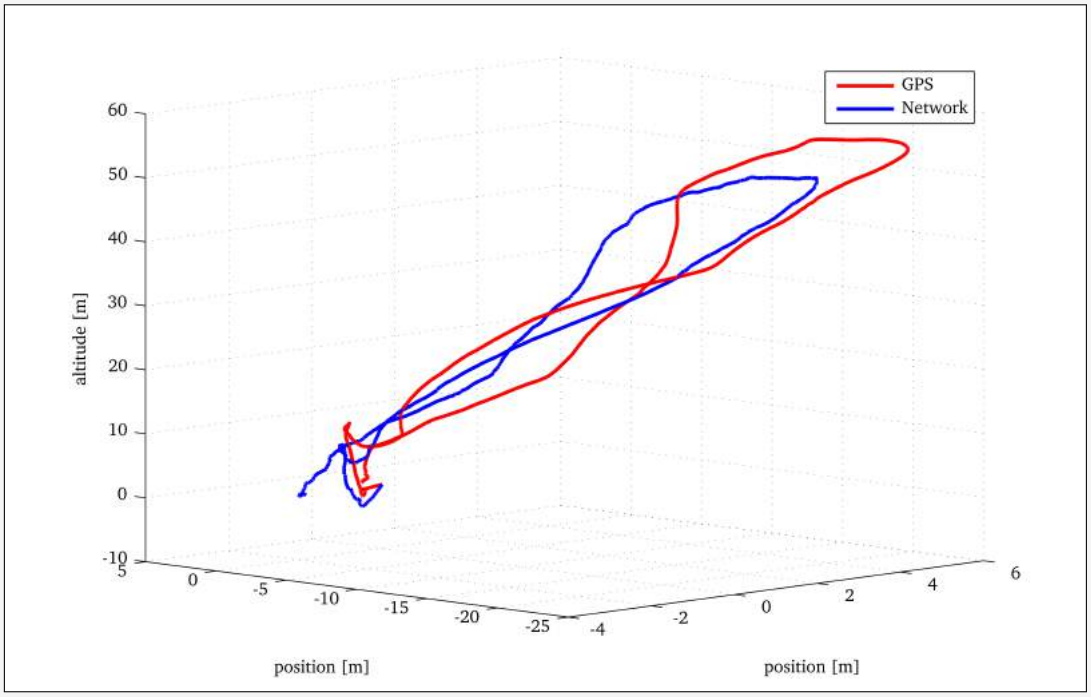

Abbildung 11: Rekonstruktion eines echten Testfluges (70 Sekunden) mit dem optischen Fluss. Die berechnete Flugbahn hat eine deutliche Ähnlichkeit zu der GPS-Trajektorie. Mögliche Fehlerquellen werden in Abschnitt 5 auf der nächsten Seite diskutiert.

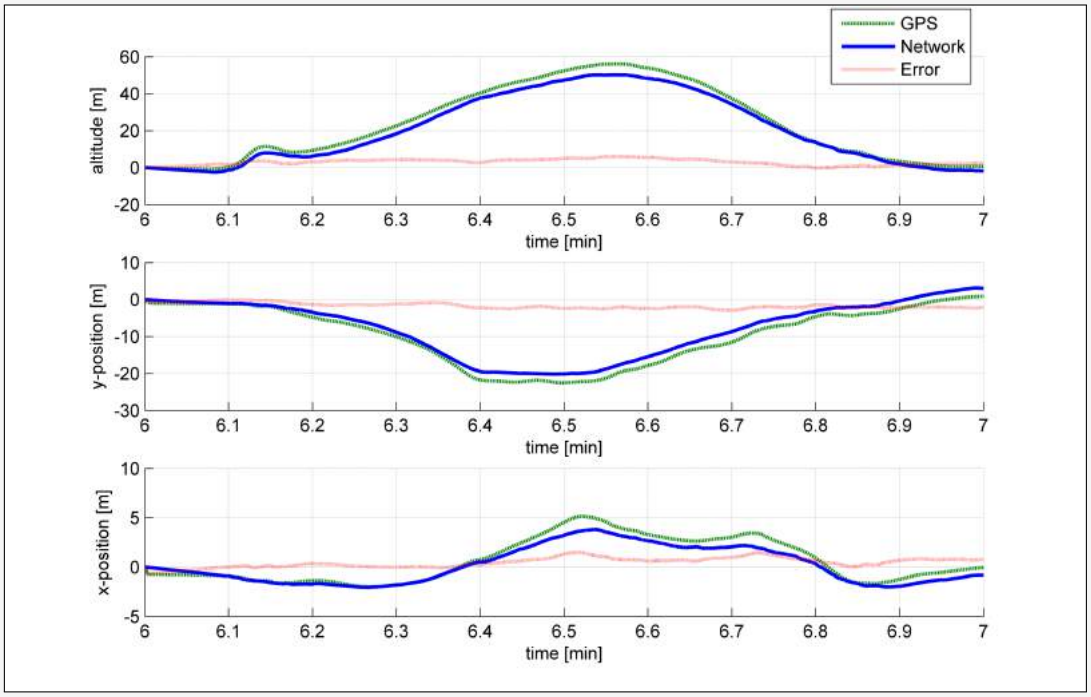

Abbildung 12: Abweichungen zwischen der Rekonstruktion der Flugbahn und der GPS-Trajektorie (Abbildung 11). Der maximale Unterschied bei der Flughöhe beträgt 5.8m. Die maximale vertikale Abweichung beträgt 2.9m beziehungsweise 1.5m.

5 Diskussion

In dieser Arbeit wurde gezeigt, dass sich die Eigenbewegung eines fliegenden Roboters unter bestimmten Einschränkungen mit dem aus Bildern von einer einzigen Kamera berechneten optischen Fluss abschätzen lässt. Anhand der Simulationsdaten wurde gezeigt, dass diese Abschätzung unter optimalen Bedingungen und einer bei fehlerfreien Bestimmung der Flussvektoren sehr genau sein kann. Bei der Rekonstruktion von einem echten Testflug konnte weiterhin gezeigt werden, dass das Verfahren auch in realen Szenarien verwendet werden kann, wenn auch mit einer höheren Ungenauigkeit. Mögliche Fehlerquellen und Verbesserungsansätze dazu werden in den nächsten Abschnitten diskutiert.

Im Projekt NeuroCopter wurde mit dieser Arbeit die Grundlage für ein biologisch inspiriertes, neuromorphes Navigationsverfahren gelegt. Ausgehend von der Integration der aus dem optischen Fluss abgeschätzten Bewegungen der Drohne lassen sich nun Modelle entwickeln, die zusammen mit anderen sensorischen Daten und einem Landmarkenerkennungsverfahren eine neuronale Karte der Umgebung generieren. Durch die Wahl eines künstlichen neuronalen Netzes wurde des weiteren die Möglichkeit geschaffen, das Verfahren auf eine biologisch akkuratere Abbildung eines Bienengehirns wie ein spikendes neuronales Netzwerk zu übertragen. Da es im Projekt NeuroCopter noch kein fertiges Navigationsmodell gibt, kann zu diesem Zeitpunkt noch nicht mit Sicherheit gesagt werden, ob die Genauigkeit des hier vorgestellten Verfahrens hoch genug ist, um eine qualitativ zufriedenstellende Navigation zu ermöglichen.

Aus biologischer Sicht hat diese Arbeit gezeigt, dass es möglich ist, dass Honigbienen und andere Organismen ihre Eigenbewegung anhand des optischen Flusses bestimmen. In Kombination mit einem biologisch inspirierten Navigationsmodell ließen sich so behaviorale Voraussagen über das Verhalten von Honigbienen unter kontrollierten Bedingungen schaffen, welche anschließend mit aufgezeichneten Daten von echten Honigbienen verglichen werden könnten.

5.1 Fehlerquellen

Aufgrund der Ungenauigkeit der auf dem NeuroCopter verwendeten Sensoren (GPS und Intertialsensoren) war es nicht möglich, die Genauigkeit des Verfahrens in realen Bedingungen mit einer zufriedenstellenden Präzision zu evaluieren. Es lässt sich feststellen, dass die Ergebnisse des Verfahrens in der realen Welt qualitativ schlechter sind als in der Simulation.

Eine mögliche Ursache sind Fehler bei der Bestimmung des optischen Flusses. Bei der verwendeten Lucas-Kanade Methode kommt es zu Ungenauigkeiten und Ausreißern bei den ermittelten Flussvektoren, die in Abhängigkeit von der Geschwindigkeit der Drohne unterschiedlich stark

ausgeprägt sind. Auch die geringe Framerate der bei dem Testflug verwendeten Kamera und die optischen Verzerrungen des Objektives hatten einen negativen Einfluss auf die Ergebnisse. Durch Umwelteinflüsse kommt es außerdem, zum Beispiel durch Windstöße, zu Bewegungen, die in einer Simulation bei einem idealen Flugmodell nicht auftreten können. Wie in Abschnitt 4 auf Seite 14 gezeigt, sind bestimmte Bewegungen besonders schwer zu rekonstruieren. Weiterhin können Fehler durch die Oberflächenstruktur oder (hohe) Objekte in der Landschaft auftreten. Bei der Verwendung von künstlichen neuronalen Netzen besteht die Gefahr, ein lokales Optimum bei dem Training zu finden. Durch das mehrmalige Trainieren des Netzes wurde das Risiko, auf ein sehr schlechtes lokales Minimum zu konvergieren, reduziert. Dennoch lässt sich nicht ausschließen, dass bei den vorhandenen Trainingsdaten und der Netzwerkstruktur ein besseres Ergebnis möglich ist.
Die als zusätzliches Merkmal verwendete Höheninformation wurde bei der Rekonstruktion von dem echten Testflug aus den Ausgaben des Netzwerkes integriert, was im Laufe der Zeit zu einer Akkumulation von Fehlern bei der Bestimmung der Eigenbewegung geführt hat.

5.2 Verbesserungsansätze

Eine genauere Untersuchung der Ergebnisse mit einer Analyse der Signifikanz der einzelnen Fehlerquellen könnte durch bessere Ground-Truth-Daten ermöglicht werden. Denkbar ist zum Beispiel die Verwendung von besseren bzw. weiteren Sensoren auf der Drohne, um die Genauigkeit der Messdaten zu erhöhen. Des weiteren wäre ein Testflug in einer kontrollierten Umgebung mit einem externen Tracking denkbar. In einer solchen Umgebung ließen sich auch Umwelteinflüsse wie Windstöße vermeiden bzw. besser kontrollieren.
Durch die Verwendung einer dedizierten Kamera zur Bestimmung des optischen Flusses mit einer hohen Framerate ließe sich die Qualität der Eingabedaten deutlich verbessern. Des weiteren würde dadurch der rechnerische Aufwand für die Berechnung des optischen Flusses auf der Embedded-CPU der Drohne reduziert werden. Im Projekt NeuroCopter ist geplant, zu diesem Zweck die kommerziell erwerbliche Kamera PX4FLOW [HM13] einzusetzen.
Bei der Bestimmung des optischen Flusses könnte man die Bildmerkmale danach klassifizieren, ob sie in etwa auf der Oberflächenebene liegen oder nicht [GBR12] und somit bei der Berechnung der gemittelten Flussvektoren nur Features mit einer konstanten Höhe verwenden.
Es gibt eine Reihe von Verfahren, mit denen das Training von mehrschichtigen neuronalen Netzen verbessert werden kann. So könnte man unter anderem die Wahrscheinlichkeit, beim Training in ein lokales Minimum zu konvergieren, durch Ansätze wie in [AS07, LGP12] vorgestellt, verringern.

Durch die Einbeziehung von weiteren sensorischen Daten könnten die Ergebnisse möglicherweise deutlich verbessert werden. So könnte man zum Beispiel Intertialsensoren verwenden, um Ausreißer zu erkennen und kompensieren. Auch eine Mittelung der Ausgaben von dem neuronalen Netz und den Messwerten von Intertialsensoren, die Verwendung von einem Bewegungsmodell (zum Beispiel ausgehend von den aktuellen Motorgeschwindigkeiten) oder die Einbeziehung von zeitlichen Informationen ist denkbar.

5.3 Ausblick

Die erforderliche Genauigkeit des Verfahrens hängt von dem gewählten Einsatzzweck ab und kann som it nicht ohne Kontext bewertet werden. In einem Navigationsverfahren wird die Abschätzung der Eigenbewegung in der Regel mit anderen sensorischen Daten kombiniert. In dem angestrebten Navigationsmodell im Projekt NeuroCopter ist es vorgesehen, die ermittelte Eigenbewegung mit der gespeicherten Position von in der Landschaft erkannten Landmarken zu kombinieren. In Abhängigkeit von der Gewissheit des Modells, die es bezüglich der Position individueller Landmarken hat, könnte so zum Beispiel der Einfluss des hier vorgestellten Verfahrens auf die Zustandsabschätzung der Drohne adaptiv angepasst werden. Wenn eine bereits oft gesehene Landmarke erkannt wird, dann ist es wahrscheinlich, dass die gespeicherte Position der Landmarke eine hinreichende Genauigkeit hat. Andernfalls ließe sich die Genauigkeit der gespeicherten Position iterativ durch Verfeinerungen mit der integrierten Eigenbewegung verbessern.

Behaviorale Daten über das Verhalten der Drohne können erhoben werden, sobald die im Abschnitt 1.1.1 auf Seite 1 vorgestellten Neuromodule implementiert sind. Ausgehend davon wäre es möglich, das Verhalten von Honigbienen mit dem NeuroCopter zu vergleichen und so Aussagen über die biologische Plausibilität der verwendeten Modelle zu treffen [CW11].

Es ist absehbar, dass man in Zukunft energieeffiziente neuromorphe Chips verwenden kann, um die im Projekt NeuroCopter vorgesehenen neuronalen Module zu implementieren. Dafür ist es erforderlich, das in der Arbeit vorgestellte Verfahren mit einem spikenden neuronalen Netz umzusetzen. Es ist schwer vorherzusehen, inwiefern die unterschiedlichen Charakteristika von einem solchen Netzwerk eine solche Portierung erschweren würden.

6 Anhang

An dieser Stelle sind weitere Beispiele für die gemittelten Flussvektoren bei einigen ausgewählten Bewegungen der Kamera abgebildet.

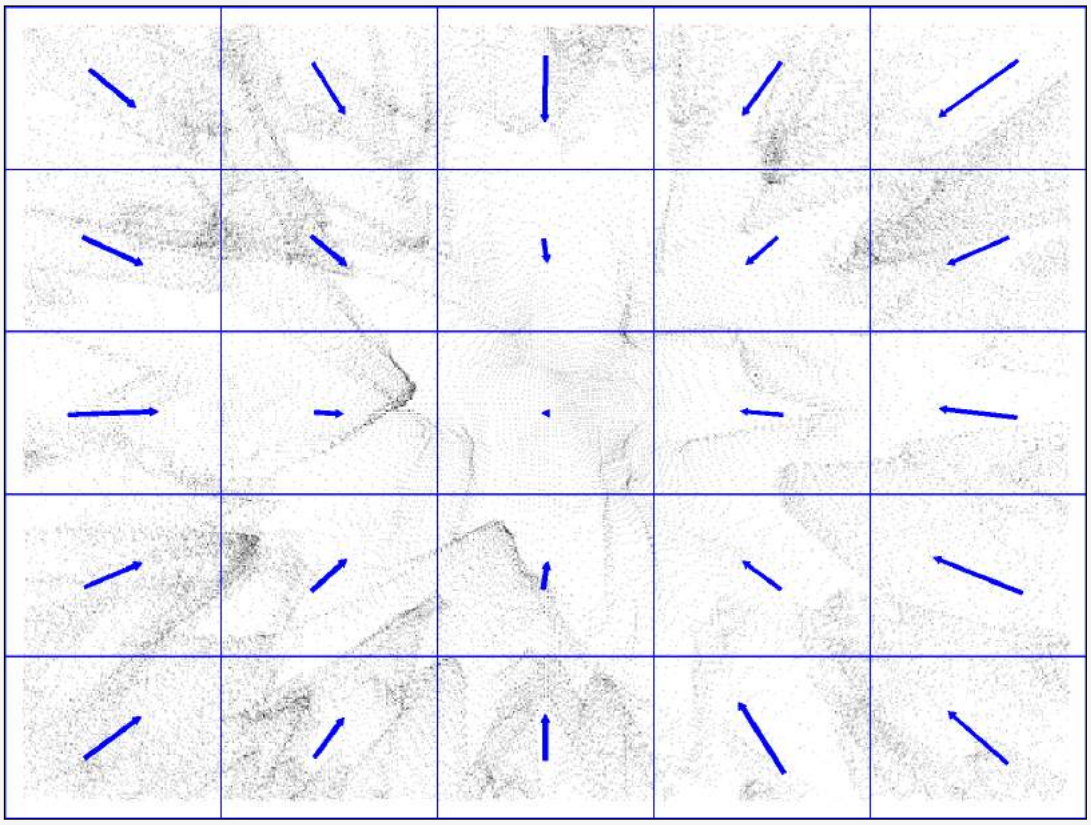

Abbildung 13: Die gemittelten Flussvektoren, die durch eine vertikal steigende Bewegung verursacht werden. Die Vektoren sind in Richtung des Zentrums der Bildebene ausgerichtet und ihre Länge ist proportional zu der Entfernung von der Bildmitte.

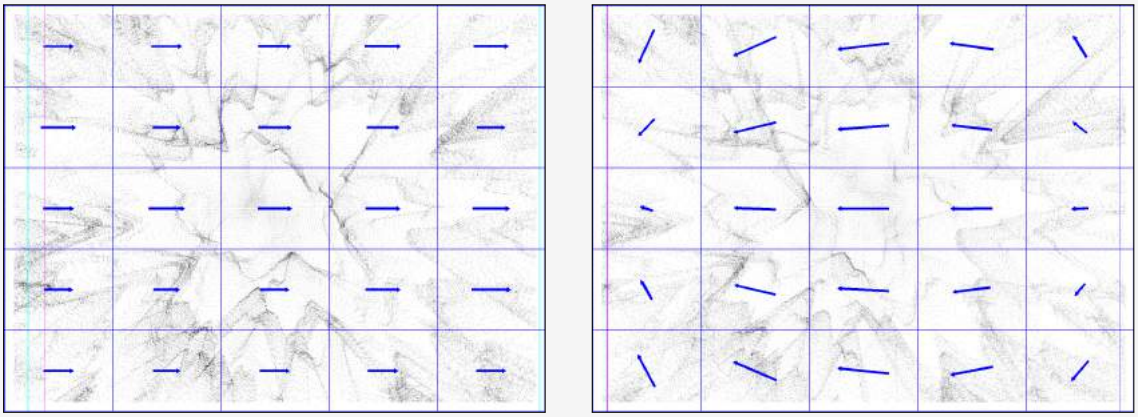

Abbildung 14: Links: Eine horizontale Bewegung auf der x-Achse. Rechts: Eine Drehung um die y-Achse (pitch).

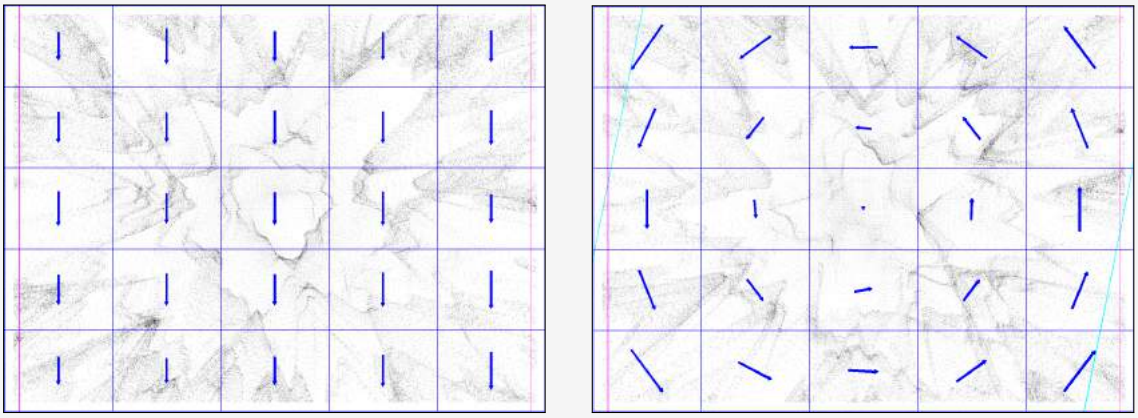

Abbildung 15: Links: Eine horizontale Bewegung auf der y-Achse. Rechts: Eine Drehung um die z-Achse (roll).

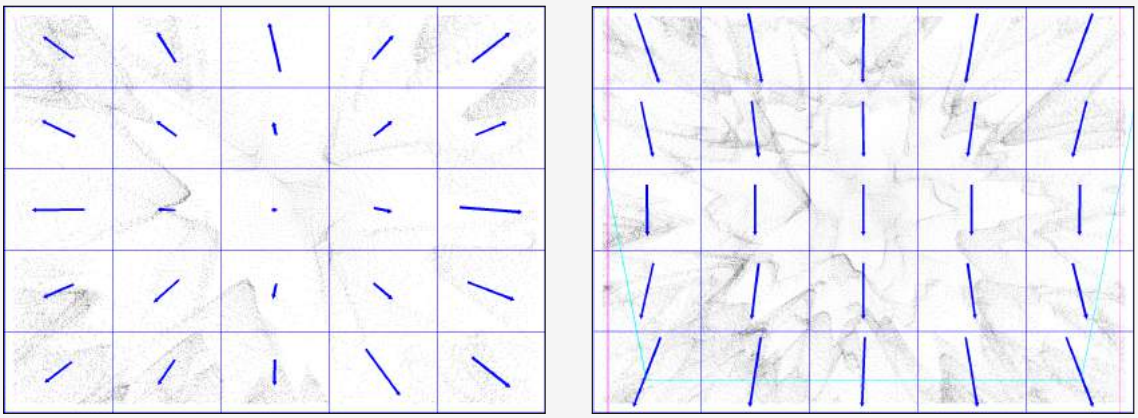

Abbildung 16: Links: Eine vertikal sinkende Bewegung. Rechts: Eine Drehung um die x-Achse (yaw).

Literatur

[AS07] Akarachai Atakulreka and Daricha Sutivong. Avoiding local minima in feedforward neural networks by simultaneous learning. *AI 2007: Advances in Artificial Intelligence*, 2007.

[CC02] Thomas S Collett and Matthew Collett. Memory use in insect visual navigation. *Nature reviews. Neuroscience*, 3(7):542–52, July 2002.

[CW11] Holk Cruse and Rüdiger Wehner. No need for a cognitive map: decentralized memory for insect navigation. *PLoS computational biology*, 7(3):e1002009, March 2011.

[Cyb89] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, pages 303–314, 1989.

[DwB06] Hugh Durrant-whyte and Tim Bailey. Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. pages 1–9, 2006.

[ERN12] M a Eckles, D W Roubik, and J C Nieh. A stingless bee can use visual odometry to estimate both height and distance. *The Journal of experimental biology*, 215(Pt 18):3155–60, September 2012.

[EZST01] H E Esch, S Zhang, M V Srinivasan, and J Tautz. Honeybee dances communicate distances measured by optic flow. *Nature*, 411(6837):581–3, May 2001.

[GBR12] Volker Grabe, Heinrich H. Bulthoff, and Paolo Robuffo Giordano. Robust optical-flow based self-motion estimation for a quadrotor UAV. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2153–2159, October 2012.

[HM13] Dominik Honegger and Lorenz Meier. An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications. *Intl. Conf. Robotics and …*, 2013.

[HS81] Berthold K.P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, August 1981.

[ILBH+11] Giacomo Indiveri, Bernabé Linares-Barranco, Tara Julia Hamilton, André Van Schaik, Ralph Etienne-Cummings, Tobi Delbruck, Shih-Chii Liu, Piotr Dudek, Philipp

Häfliger, Sylvie Renaud, et al. Neuromorphic silicon neuron circuits. *Frontiers in neuroscience*, 5, 2011.

[Koh95] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. *IJCAI*, 1995.

[LGP12] JTH Lo, Yichuan Gui, and Yun Peng. Overcoming the local-minimum problem in training multilayer perceptrons with the NRAE training method. *Advances in Neural Networks ISNN 2012*, pages 1–8, 2012.

[LK81] Bruce D Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. *Imaging*, 130(x):674–679, 1981.

[LWL13a] Tim Landgraf, Benjamin Wild, and Tobias Ludwig. NeuroCopter: Neuromorphic Computation of 6D Ego-Motion of a Quadcopter. *…and Biohybrid Systems*, pages 1–12, 2013.

[LWL+13b] Tim Landgraf, Benjamin Wild, Tobias Ludwig, Philipp Nowak, Benjamin Daumenlang, and Philipp Breinlinger. Neurocopter - optical flow. https://www.youtube.com/watch?v=KvmmQc9p-xA, 2013.

[Mar63] DW Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial & Applied …*, (Nlin 2):1970, 1963.

[Men12] Randolf Menzel. The honeybee as a model for understanding the basis of cognition. *Nature Reviews Neuroscience*, 13(11):758–768, 2012.

[MGS+05] Randolf Menzel, Uwe Greggers, Alan Smith, Sandra Berger, Robert Brandt, Sascha Brunke, Gesine Bundrock, Sandra Hülse, Tobias Plümpe, Frank Schaupp, Elke Schüttler, Silke Stach, Jan Stindt, Nicola Stollhoff, and Sebastian Watzl. Honey bees navigate according to a map-like spatial memory. *Proceedings of the National Academy of Sciences of the United States of America*, 102(8):3040–5, February 2005.

[MKH+11] Randolf Menzel, Andreas Kirbach, Wolf-Dieter Haass, Bernd Fischer, Jacqueline Fuchs, Miriam Koblofsky, Konstantin Lehmann, Lutz Reiter, Hanno Meyer, Hai Nguyen, et al. A common frame of reference for learned and communicated vectors in honeybee navigation. *Current Biology*, 21(8):645–650, 2011.

[MLM+12] Randolf Menzel, Konstantin Lehmann, Gisela Manz, Jacqueline Fuchs, Miriam Koblofsky, and Uwe Greggers. Vector integration and novel shortcutting in honeybee navigation. *Apidologie*, 43(3):229–243, March 2012.

[Pan08] RM Panish. Vehicle egomotion estimation using computer vision. 2008.

[PGJ+13] Thomas Pfeil, Andreas Grübl, Sebastian Jeltsch, Eric Müller, Paul Müller, Mihai A Petrovici, Michael Schmuker, Daniel Brüderle, Johannes Schemmel, and Karlheinz Meier. Six networks on a universal neuromorphic computing substrate. *Frontiers in neuroscience*, 7, 2013.

[PRRF11] Geoffrey Portelli, Franck Ruffier, Frédéric L Roubieu, and Nicolas Franceschini. Honeybees' speed depends on dorsal as well as lateral, ventral and frontal optic flows. *PloS one*, 6(5):e19486, January 2011.

[PZ11] Chi-Sang Poon and Kuan Zhou. Neuromorphic silicon neurons and large-scale neural networks: challenges and opportunities. *Frontiers in neuroscience*, 5:108, January 2011.

[SGGW11] David Sterratt, Bruce Graham, Andrew Gillies, and David Willshaw. *Principles of computational modelling in neuroscience*. 2011.

[Sri03] Zhang Srinivasan. Honeybee navigation: properties of the visually driven 'odometer'. *Journal of Experimental Biology*, 206(8):1265–1273, April 2003.

[ST94] Jianbo Shi and Carlo Tomasi. Good features to track. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94*, pages 593–600. IEEE Comput. Soc. Press, 1994.

[SZLC96] M Srinivasan, S Zhang, M Lehrer, and T Collett. Honeybee navigation en route to the goal: visual flight control and odometry. *The Journal of experimental biology*, 199(Pt 1):237–44, January 1996.

[Wei12] SM Weiss. Vision based navigation for micro helicopters. 2012.

[WS03] Rüdiger Wehner and Mandyam V Srinivasan. Path integration in insects. 2003.