Hochschule **Rosenheim**
University of Applied Sciences

Optische Positions- und Lagebestimmung einer Drohne im geschlossenen Raum

Projektarbeit 2:

Masterstudiengang
Angewandte Forschung & Entwicklung in den Ingenieurwissenschaften

vorgelegt von

Michael Spagl
Matr.-Nr.: 873259

am 2. März 2021
an der Hochschule Rosenheim

Erstprüfer:	Prof. Dr.-Ing. Peter Zentgraf, M.Sc.
Zweitprüfer:	Prof. Dr.-Ing. Michael Wagner

Kurzfassung

Die vorliegende Arbeit beschäftigt sich mit der Erarbeitung einer optischen Positions- und Lagebestimmung eines Quadrokopters mit Hilfe einer zum Boden gerichteten Kamera und den daraus berechneten optischen Fluss.

Nach der Hinführung zum Thema und dem Aufzeigen der genauen Fragestellung werden im ersten Schritt alle hierzu notwendigen Grundlagen dargestellt. Hierzu zählt ein kurzer Überblick im Bereich der optischen Bildverarbeitung, das Modell der Lochkamera, die geometrische Kamerakalibrierung sowie die Definition und Berechnung des optischen Flusses. Es wird ebenso auf die Horn-Schunck Methode sowie die Lucas-Kanade Methode näher eingegangen. Am Ende des zweiten Kapitels wird der in dieser Arbeit verwendete Quadrokopter und dessen Sensoren kurz dargestellt.

Im Anschluss wird eine von MathWorks Inc. bereits implementierte Positions- und Lagebestimmung in einer Versuchsreihe auf dessen Genauigkeit untersucht. Da mit Hilfe dieser keine aussagekräftigen Ergebnisse erzielt werden konnten, wurde im 4. Kapitel eine eigene optische Positions- und Lageregelung erfolgreich erarbeitet. Diese basiert ebenfalls auf der Berechnung des optischen Flusses und der Methode der kleinsten Quadrate bzw. dem Kalman Filter. Im letzten Teil wird das zuvor erarbeitete Verfahren näher untersucht, der Einfluss von Messungenauigkeiten sowie Messrauschen dargestellt und simuliert.

Abstract

The present thesis deals with the development of an optical position and orientation determination of a quadcopter. This should be done with the help of a camera directed towards the ground and the calculation of the optical flow.

After introducing the topic and describing the exact question, in the first step all the necessary basics are presented. This includes a brief overview in the field of optical image processing, the model of the pinhole camera, the geometric camera calibration and the definition and calculation of the optical flow. The Horn-Schunck method and the Lucas-Kanade method are also discussed in more detail. At the end of the second chapter, the used quadcopter and its sensors are briefly discussed.

Afterwards, a position and attitude determination already implemented by MathWorks Inc. is examined in a series of tests for its accuracy. Since unusable results could be achieved, a separate optical position and orientation determination was successfully developed in Chapter 4. This method is based on the calculation of the optical flow data and the least squares method or the Kalman filter. In the last part, the previously developed procedure is examined, and the influence of measurement errors and measurement noise is shown and simulated.

Inhaltsverzeichnis

Kurzfassung ..... **I**

Abstract ..... **I**

Inhaltsverzeichnis ..... **II**

1 Einleitung **1**

1.1 Motivation und Problemstellung 1

1.2 Zielsetzung der Arbeit 2

2 Grundlagen **3**

2.1 Überblick 3

2.2 Lochkamera 4

2.3 Geometrische Kamerakalibrierung 5

2.4 Bewegungsfelder und optischer Fluss 10

2.4.1 Berechnung des optischen Flusses 12

2.4.2 Lucas-Kanade Methode 14

2.4.3 Horn-Schunck Methode 16

2.5 Bilderfassung mit Parrot Minidrones 19

3 Vorhandene optische Positions- und Lagebestimmung **20**

3.1 Qualifizierung der vorhandenen optischen Positions- und Lagebestimmung 22

3.1.1 Ziel 22

3.1.2 Versuchsaufbau 22

3.1.3 Versuchsdurchführung 23

3.2 Ergebnis 24

4 Optische 3D Positions- und Lagebestimmung **31**

4.1 Algebraische Berechnung eines Geschwindigkeitsvektorfeldes 32

4.2 Erweiterung des algebraischen Geschwindigkeitsvektorfeldes 38

4.2.1 Bekannte Größen 40

4.2.2 Koordinatentransformation zwischen Inertial- und Kamera-Koordinatensystem 41

4.2.3 Geometrischer Zusammenhang 41

4.2.4 Linearisierung der Drehwinkel 45

4.3 Schätzung der Kamerageschwindigkeiten sowie der Kameralage 48

4.4 Simulation der Geschwindigkeits- und Lagebestimmung 52
4.4.1 Qualifizierung unter idealen Bedingungen 53
4.4.2 Qualifizierung unter realistischen Bedingungen 57
5 Zusammenfassung und Ausblick 66
5.1 Zusammenfassung der Ergebnisse 66
5.2 Ausblick 67
Literaturverzeichnis IV
Abbildungsverzeichnis VI
Tabellenverzeichnis VIII
Formelverzeichnis IX
Anhang XI
Anhang A: Umformung der Gleichung (4.30) XI
Anhang B: Die Berechnung von alfa XIII
Anhang C: Koeffizienten der Gleichung (4.34) XVII
Anhang D: Lösungen der unbekannten Größen vx, vy, vz, ωx, ωy, ωz, β XIX
Anhang E: Verwendete Software und Hardware XX

1 Einleitung

1.1 Motivation und Problemstellung

„Die Drone-Economy ist ein Markt mit Zukunft. Weltweit sprießen Visionen für innovative Drohnenservices wie Pilze aus dem Boden. Von der Paketzustellung bis hin zu Flugtaxis – die Möglichkeiten, die Drohnenanwendungen und automatisierte Mobilitätslösungen bieten, sind ebenso zahlreich wie faszinierend. Und die Potenziale sind gewaltig: Alle Prognosen gehen von einem stetigen Wachstum des Marktes aus.“ [1]

Abbildung 1: ... mit Drohnen fliegen – BMWi [1]

Mit diesen Worten leitet der Koordinator der Bundesregierung für die Deutsche Luft- und Raumfahrt Thomas Jarzombek in der vom Bundesministerium für Wirtschaft und Energie die 2019 veröffentlichten Broschüre „… mit Drohnen - Unbemanntes Fliegen im Dienst von Mensch, Natur und Gesellschaft“ ein. In dieser Broschüre werden beispielhafte Einsatzbereiche von Drohen vorgestellt [1].

Doch betrachtet man die dort aufgelisteten Projekte genauer so fällt auf, dass all diese für dein Einsatz für den Outdoor-Bereich konzipiert sind. Wie sieht es im Indoor-Bereich aus? GPS-Signale, die eine Wand durchdringen, werden in Abhängigkeit von den Baumaterialien zusätzlich um den Faktor 100 und mehr gedämpft. Gegenüber einem Outdoor-Empfänger, der das Signal innerhalb weniger Sekunden empfängt, benötigt derselbe Empfänger unter den Bedingungen des Innenraumes sehr viel länger, ein brauchbares Signal- Rauschverhältnis zu erhalten [2].

Damit eine Drohe einen vorgegebenen Pfad im Raum abfliegen kann, ist somit eine schnellere und effektivere Möglichkeit zur Positions- und Lagebestimmung des Flugkörpers nötig. Eine Möglichkeit ist die optische Positions- und Lagebestimmung.

Genau mit dieser Frage beschäftigt sich die nachfolgende Projektarbeit, in welcher eine funktionsfähige optische Positions- und Lagebestimmung erarbeitet werden soll.

1.2 Zielsetzung der Arbeit

Das oberste Ziel dieser Arbeit ist es eine funktionsfähige Positions- und Lagebestimmung eines Quadrokopters im Raum zu erarbeiten, zu entwerfen und zu testen. Diese Positions- und Lagebestimmung soll mit Hilfe einer am Quadrokopter befestigten und vertikal nachunten gerichteten Kamera realisiert werden. Für die Realisierung dieser optischen Positions- und Lagebestimmung wird der Quadrokopter Rolling Spider der Firma Parrot Drones SAS sowie die Programmierumgebungen Simulink und MATLAB der Firma MathWorks, Inc. verwendet.

2 Grundlagen

2.1 Überblick

Generell ist Licht eine Form von elektromagnetischer Strahlung. Hierbei ist nur ein bestimmtes Spektrum dieser elektromagnetischen Strahlung für das menschliche Auge sichtbar. Zur Vereinfachung von Berechnungen wird in der Strahlenoptik der Verlauf des Lichtes ausschließlich durch einen Lichtstrahl beschrieben. Dabei wird die seitliche Ausdehnung der elektromagnetischen Wellen vernachlässigt [3].

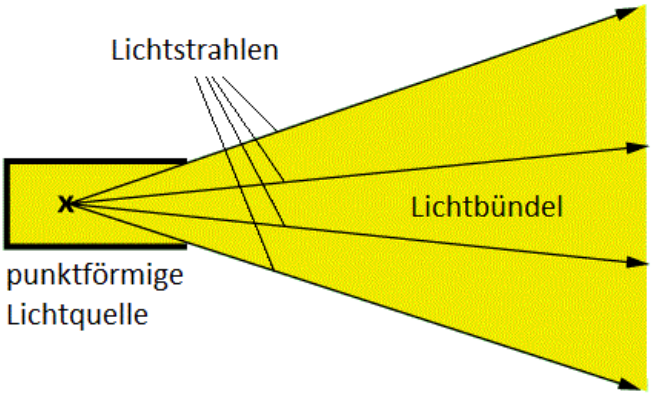


Abbildung 2: Lichtstrahlen und Lichtbündel Quelle: www.lernhelfer.de/schuelerlexikon/physik/artikel/lichtstrahlen-und-lichtbuendel

In der optischen Bildverarbeitung wird meistens das emittierte Licht einer Bildquelle bzw. das reflektierte Licht eines Objekts auf einen Kamerasensor[1] über ein optisches System abgebildet. Um den Verlauf des Lichts, welches durch das Linsensystem auf den optischen Sensor fällt, beschreiben zu können, werden auch hier die elektromagnetischen Wellen des Lichtes durch eine einfache Linie angenähert [4].

In Abbildung 3 ist beispielhaft ein Linsensystem gezeigt, bei welchen ein Objekt auf die Bildebene des Kamerasensors mit Hilfe einer Linse projiziert wird.

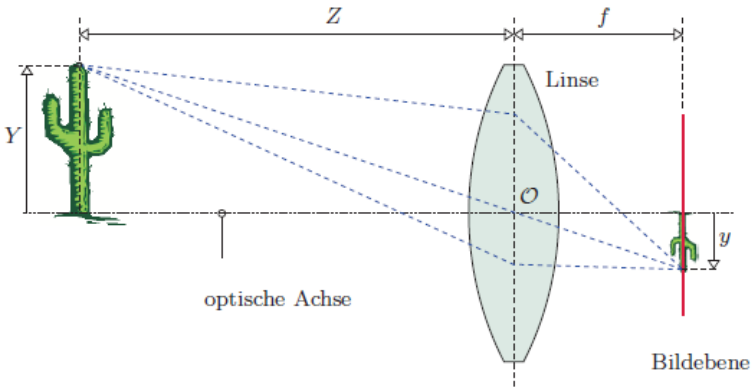


Abbildung 3: Einfaches Linsensystem [5]

Bei mehreren Linsen und komplizierteren Linsensystemen wird die Berechnung des Verhältnisses zwischen einem beliebigen Weltpunkt und der dazugehörigen Projektion auf der Bildebene des Sensors sehr aufwendig und zeitintensiv. Daraus folgend ist bei unbekannten Linsensystemen die Rückführung des Bildpunktes in einen Weltpunkt nicht möglich.

[1] Meist CCD-Sensoren (Charge Coupled Device) oder CMOS-Sensoren (Complementary Metal Oxide Semiconductor)

2.2 Lochkamera

Eine Lochkamera stellt dabei das einfachste Prinzip einer Kamera dar. Dieses Modell ist bereits seit dem 13. Jahrhundert bekannt. Sie hat zwar heute keinerlei praktische Bedeutung, dient jedoch als brauchbares Modell, um wesentliche Elemente der optischen Abbildung ausreichend genau beschreiben zu können [5].

Eine Lochkamera besteht im Wesentlichen aus einer geschlossenen Box mit einem kleinen Loch an der Vorderseite und der Bildebene an der Rückseite. Die Bildebene stellt dabei die lichtempfindliche Ebene des Kamerasensors dar. Lichtstrahlen, welche ausgehend von einem Objektpunkt Richtung der Kamera fallen, werden geradlinig durch die Öffnung auf die Bildebene projiziert. Wie in Abbildung 4 dargestellt, entsteht so ein verkleinertes und gespiegeltes Abbild des Objekts auf der Sensoroberfläche [5].

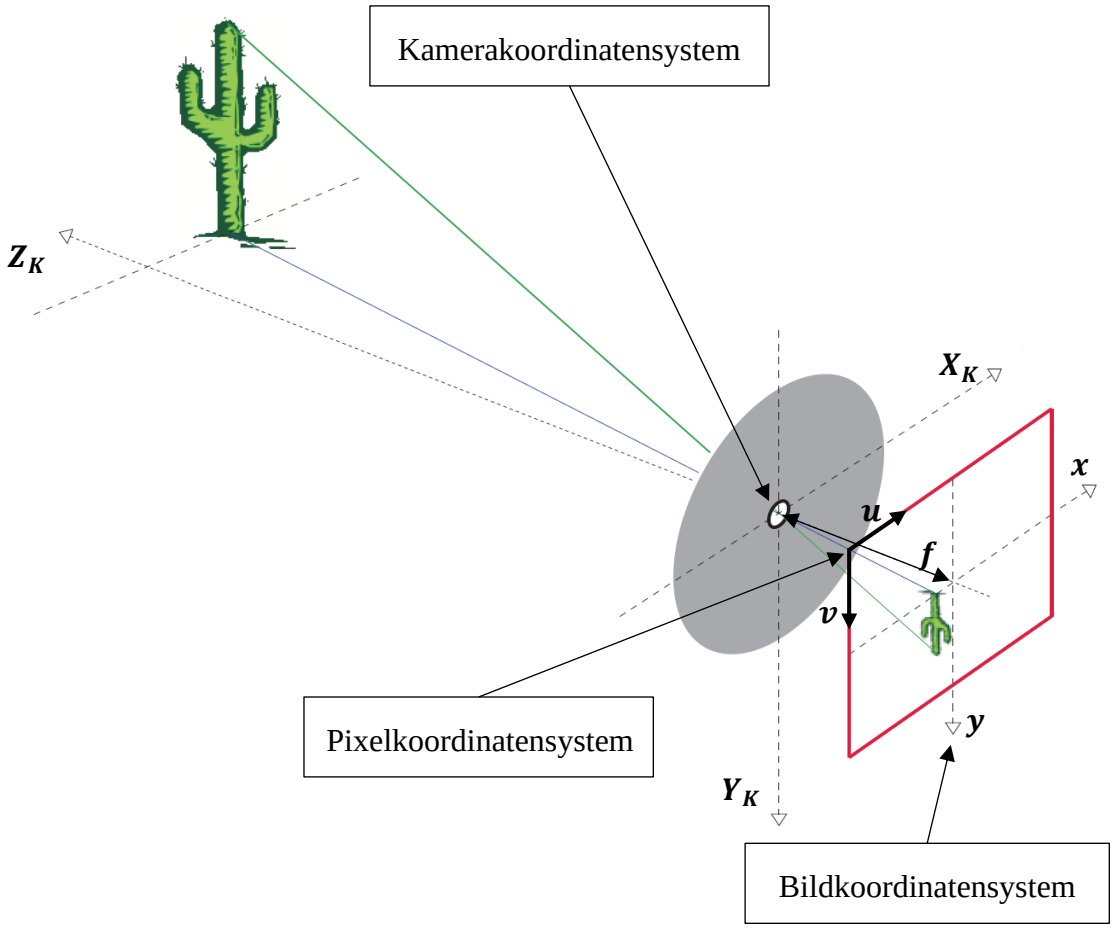


Abbildung 4: Geometrie einer Lochkamera [5]

Die geometrischen Verhältnisse einer Lochkamera lassen sich einfach beschreiben. Die „optische Achse“ verläuft gerade durch die Lochöffnung der Kamera und trifft rechtwinklig auf die Bildebene des Sensors. Die Brennweite f gibt den Abstand der Bildebene zur Lochöffnung (≙ Fokuspunkt) an. Z_C beschreibt den Abstand des Weltpunktes zum Koordinatenursprung des Koordinatensystems [5].

Durch den Vergleich der einzelnen Abstände auf der Bildseite und der Objektseite der Kamera lassen sich folgende Zusammenhänge erkennen.

$$x = f \cdot \frac{X_C}{Z_C} \tag{2.1}$$

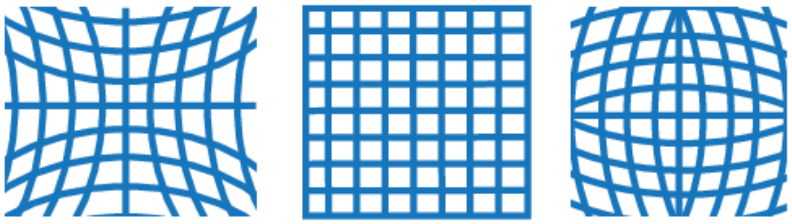
$$y = f \cdot \frac{Y_C}{Z_C} \tag{2.2}$$

Die Koordinaten (x,y) beschreiben die Position des Bildpunktes auf der Bildebene mit einem Abstand f zum Ursprung des Kamerakoordinatensystem. (X_C,Y_C, Z_C) geben die Position des dazugehörigen Weltpunktes an

2.3 Geometrische Kamerakalibrierung

Wie bereits beschrieben bildet eine Kamera die dreidimensionale Welt auf ein zweidimensionales Bild ab. Es werden 3D-Weltpunkte auf einen 2D-Bildpunkt projiziert. Ein zentraler Bereich der digitalen Bildverarbeitung ist die dreidimensionale Interpretation von Bildern. Hierzu zählt die Korrektur der optischen Verzerrung, das Abschätzen von der Entfernung eines Weltpunktes zur Kamera und das Vermessen von Objekten. Für alle diese Aufgaben ist es notwendig alle unbekannten Parameter des verwendeten Kameramodells zu bestimmen. Zu den Kameraparametern gehören die intrinsischen Parameter, die extrinsischen Parameter und Verzerrungskoeffizienten. Die Bestimmung dieser Parameter nennt man Kamerakalibrierung. Die durch eine Kalibrierung zu bestimmenden Parameter hängen dabei von dem verwendeten Kameramodell ab.

Um eine möglichst einfache Rechengrundlage zu erreichen, wird als Basis der Kamerakalibrierung das Kameramodell deiner Lochkamera verwendet. Um eine Vielzahl von verschiedensten Kameramodellen und Kamerasystemen auf das einfachste Modell zurück führen zu können, wird das Lochkameramodell um eine Linsenverzerrung in radialer und tangentialer Richtung erweitert [6].

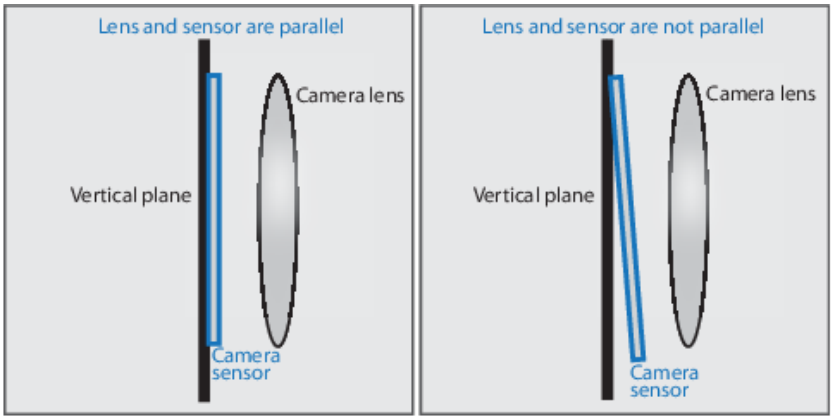


Negative radiale Verzerrung | *Keine radiale Verzerrung* | *Positive radiale Verzerrung*

Abbildung 5: Radiale Linsenverzerrung [7]

Die tangentiale Verzerrung ist hierbei auf eine unvollständige Zentrierung der Linsenkomponenten und andere Herstellungsfehler in einem Linsensystem zurückzuführen [6].

Dieses Verzerrungsmodell wurde erstmals 1966 von Brown eingeführt und wird als "Plumb Bob"-Modell bezeichnet. Dieses Verfahren wird oft im Bereich der Kamerakalibrierung eingesetzt [6].



Keine tangentiale Verzerrung | *tangentiale Verzerrung*

Abbildung 6: Tangentiale Bildverzerrung [7]

Die Brennweiten des Kameramodells $f_x\ [pixel]$ und $f_y\ [pixel]$ sowie der Kamerahauptpunkt[2] $c_x\ [pixel]$ und $c_y\ [pixel]$ bilden die intrinsischen Parameter einer Kamera. Sie können in der Kamerakalibriermatrix K zusammengefasst werden. Der Vektor $[x_n \quad y_n \quad 1]^T$ gibt die auf $z_n = 1$ normierte Position des Pixels u und v im Kamerakoordinatensystem an.

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \cdot \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} \tag{2.4}$$

Bei realen Kameras kann es vorkommen, dass ein Bildsensor nicht aus quadratischen Pixeln aufgebaut ist. Dies bedeutet, dass ein Bild in vertikaler und horizontaler Richtung unterschiedlich skaliert ist. Somit ergeben sich unterschiedliche Brennweiten der Kamera in X- und Y-Richtung. Sind die Pixel einer Kamera quadratisch so gilt $f_x = f_y$.

Die extrinsischen Parameter beschreiben die Position und die Orientierung der Kamera im dreidimensionalen Raum. Sie stellen einen Zusammenhang zwischen dem Kamerakoordinatensystem und dem Weltkoordinatensystem des betrachteten Objektes her. Diese sind vor allem für die Kalibrierung von Stereokameras notwendig und werden deshalb im Folgenden nicht näher betrachtet [7].

Der Zusammenhang zwischen den intrinsischen und extrinsischen Parametern sowie den Welt-, Kamera- und Pixelkoordinatensystemen ist in Abbildung 7 und in Abbildung 8 dargestellt [7].

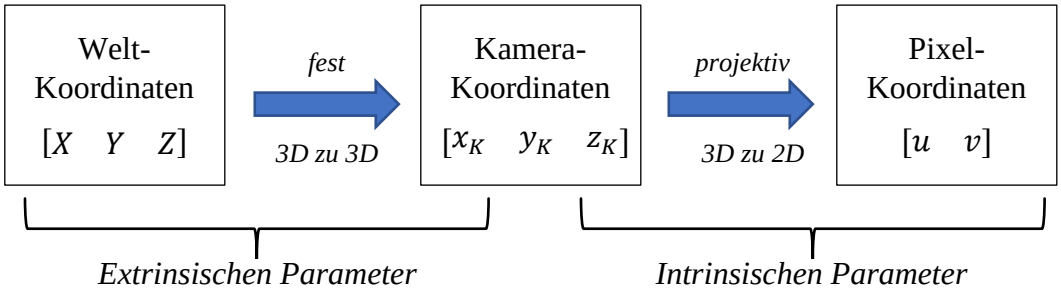


Abbildung 7: Koordinatentransformation zwischen Welt-, Kamera- und Pixelkoordinatensystem

[2] Siehe Abbildung 8

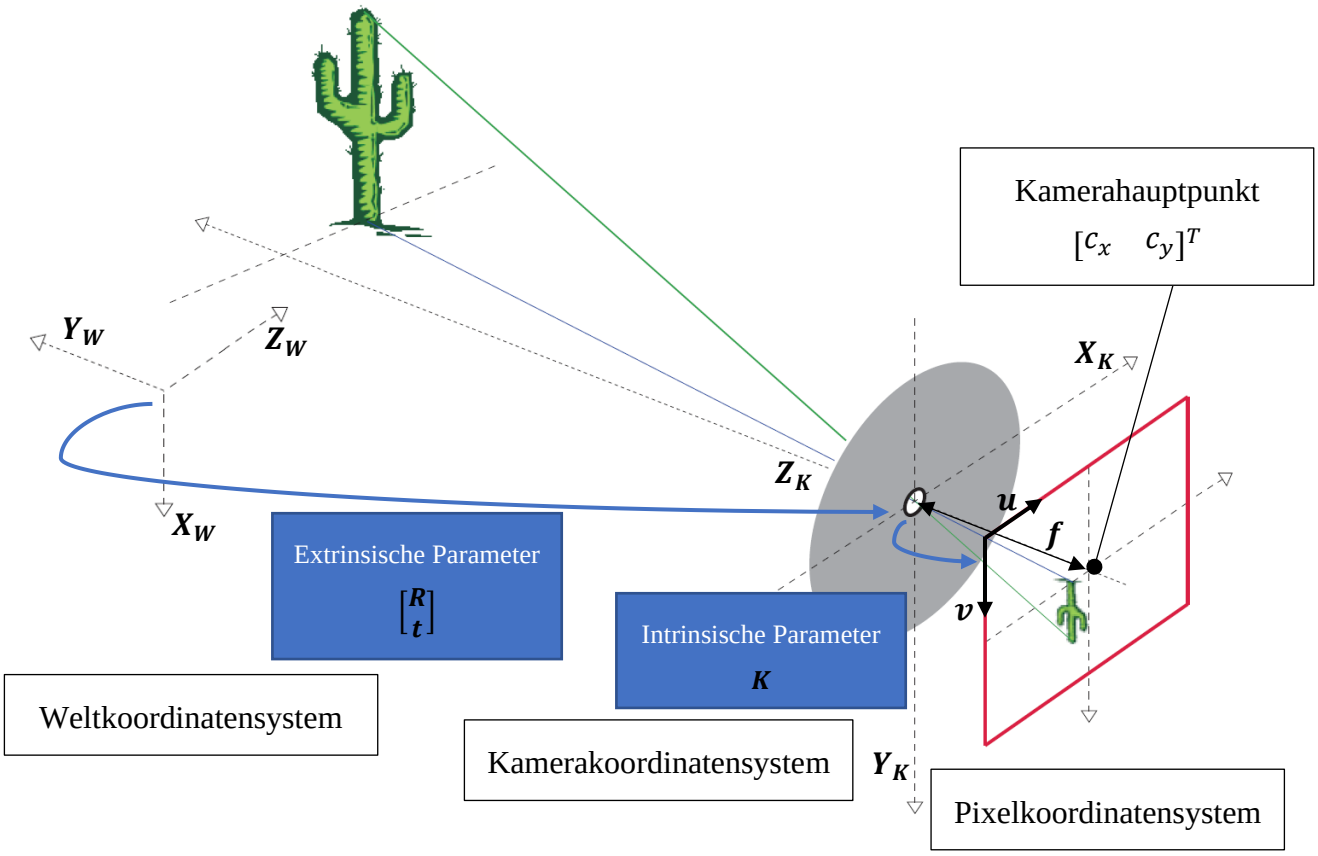


Abbildung 8: Zusammenhang zwischen intrinsischen und extrinsischen Kameraparametern sowie dem Welt-, Kamera- und Pixelkoordinatensystem [5]

Dabei ist zu erwähnen ist, dass sich der Ursprung des Pixelkoordinatensystems in der Mitte des linken oberen Pixels eines Bildes befindet [6].

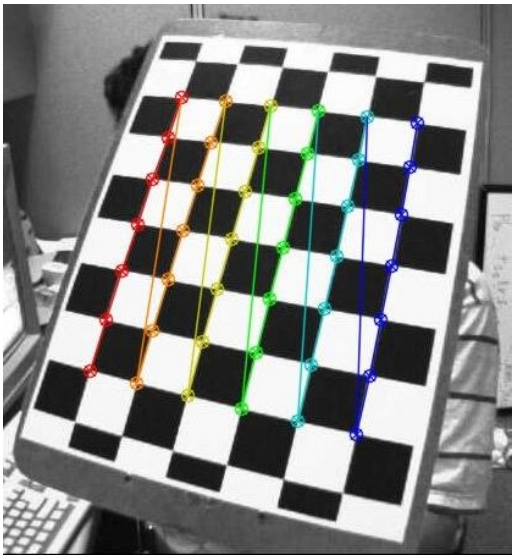


Abbildung 9: Beispiel Kalibriermuster [8]

Um die unbekannten intrinsischen und extrinsischen Kameraparameter sowie die Verzerrungskoeffizienten einer Kamera schätzen zu können, ist ein bekanntes Kalibriermuster notwendig. Dabei sind die relativen Positionen von spezifischen Punkten bekannt. Hierzu eignen sich besonders die quadratischen Ecken eins Schachbrettmusters oder die Mittelpunkte eines symmetrischen bzw. unsymmetrischen Punktemusters. Dadurch sind die Positionen der markanten Punkte im Raum sowie deren Koordinaten im Pixelkoordinatensystem bekannt. Anschließend können mit Hilfe eines Kalibrierverfahrens die Verzerrungskoeffizienten sowie die intrinsischen und extrinsischen Parameter der Kamera geschätzt werden. Für eine ausreichend genaue Schätzung wird ein Kalibriermuster mit mindestens 10 spezifischen Punkten benötigt. Ebenso sollte das Kalibriermuster in mindestens 3 unterschiedlichen Positionen im Raum von der Kamera erfasst werden [8].

Nachfolgend wird für die Schätzung der intrinsischen Kameraparameter und der Verzerrungskoeffizienten die *„Camera Calibration Toolbox“* von MATLAB sowie ein Schachbrettmuster mit einer Kantenlänge der Quadrate von $40mm$ verwendet.

Nachdem alle unbekannten Größen ermittelt worden sind, kann mit Hilfe der Verzerrungskoeffizienten und numerischen Methoden die tangentiale sowie die radiale Linsenverzerrung in einem Bild korrigiert werden. In dem so erhaltenen, unverzerrten Bild kann nun jede Pixelposition mit Hilfe der intrinsischen Parameter vom Pixelkoordinatensystem in das Kamerakoordinatensystem transformiert werden[3]. Alle weiteren Berechnungen werden nachfolgend im Kamerakoordinatensystem nach den Rechenregeln des Lochkameramodells durchgeführt.

[3] Siehe Formel (2.3) und Formel (2.4)

2.4 Bewegungsfelder und optischer Fluss

Die Bewegung ist eines der wichtigsten Forschungsthemen im Bereich der Bildverarbeitung. Es ist die Basis einer Vielzahl von anderen Problemstellungen. Ein wichtiger Teil der Bewegungsanalyse ist hierbei der optische Fluss. Als optischer Fluss wird in der Bildverarbeitung ein 2D-Vektorfeld bezeichnet, welches für jeden Punkt einer Bildsequenz die Bewegungsrichtung und die Bewegungsgeschwindigkeit beschreibt. Die Berechnung des optischen Flusses findet in verschiedensten Bereichen Anwendung. Hierzu gehört die autonome Navigation von Robotern, die 3D-Bildrekonstruktion, die Objekt-Verfolgung in einer Bildfolge sowie bei der Komprimierung von Videodateien [9].

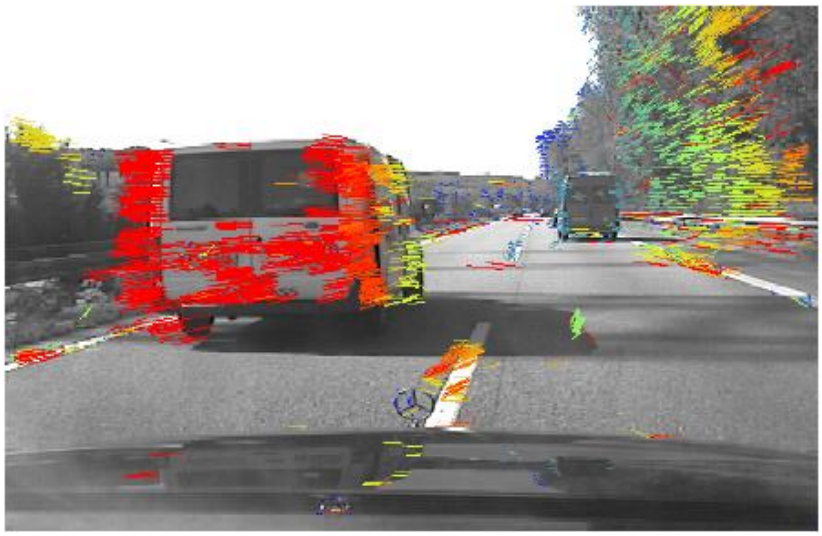


Abbildung 10: optischer Fluss bei Fahrassistenzsystemen [10]

Die Abbildung 10 zeigt beispielhaft das 2D-Vektorfeld des berechneten optischen Flusses eines Fahrerassistenzsystems. Hierbei stellt jeder Strich die Bewegungsrichtung und die Farbe die Bewegungsgeschwindigkeit des betrachteten Pixels dar.

Das Bewegungsfeld eines Objekts im Raum kann nicht direkt mit Hilfe einer Kamera beobachtet werden, da das Bewegungsmuster des 3D-Punktes im Raum unbekannt ist. Mit Hilfe des optischen Flusses versucht man das zweidimensionale Geschwindigkeitsfeld, welches von dem 3D-Punkt auf die Bildebenen projiziert wird, mit Hilfe einer Bildfolge und einem sich bewegendem Helligkeitsmuster anzunähern. Der optische Fluss entspricht nicht immer dem tatsächlichen Geschwindigkeitsfeld des Objektes auf der Bildebene [9].

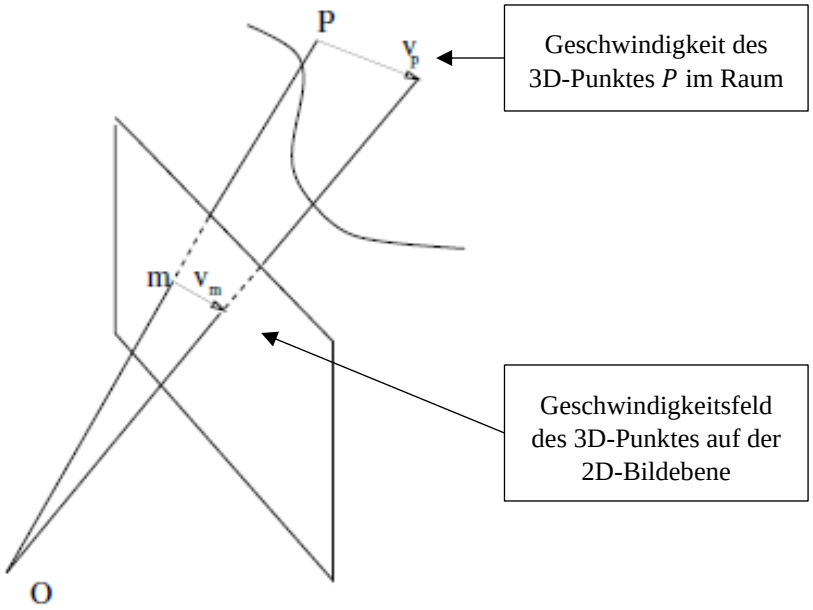


Abbildung 11: Projektion einer Bewegung eines 3D-Punktes im Raum auf die 2D-Bildebene [9]

Betrachten man eine vollkommen gleichmäßige Kugel, welche einen Schatten auf die Bildfläche der Kamera projiziert. Dreht man die Kugel, so bewegt sich der projizierte Schatten der Kugel nicht. In diesem Fall ist der optische Fluss Null, das tatsächliche Geschwindigkeitsfeld jedoch nicht [9].

Hält man die Kugel ruhig bewegt aber die Lichtquelle, so ändert sich der Schatten der Kugel auf der Bildebene. In diesem Fall ist der optische Fluss nicht Null, sondern die tatsächliche Bewegung des Objekts [9].

Jedoch wird im Bereich der Bildverarbeitung davon ausgegangen, dass sich der optische Fluss und das tatsächliche Bewegungsfeld in dem meisten Situationen nicht unterscheiden und somit identisch sind.

2.4.1 Berechnung des optischen Flusses

Zur Berechnung des optischen Flusses wurden in den letzten Jahren eine Vielzahl von unterschiedlichsten Algorithmen entwickelt. Zu den meist verbreitetsten zählen hierzu die Horn-Schunck Methode sowie die Lucas-Kanade Methode. Beide Methoden zählen zu den differentiellen Verfahren [11].

Bewegt sich ein Weltpunkt auf einen dreidimensionalen Pfad im Raum und wird dieser auf die Bildebene der Kamera projiziert. So ergibt sich der zweidimensionale Pfad $\vec{x}(t)$. Die Geschwindigkeit des projizierten Weltpunktes kann durch eine Differentiation berechnet werden [11].

$$\dot{\vec{x}}(t) = \begin{bmatrix} \dot{\vec{x}}(t) \\ \dot{\vec{y}}(t) \end{bmatrix} = \frac{d\vec{x}(t)}{dt} = \begin{bmatrix} \frac{d\vec{x}(t)}{dt} \\ \frac{d\vec{y}(t)}{dt} \end{bmatrix} \tag{2.5}$$

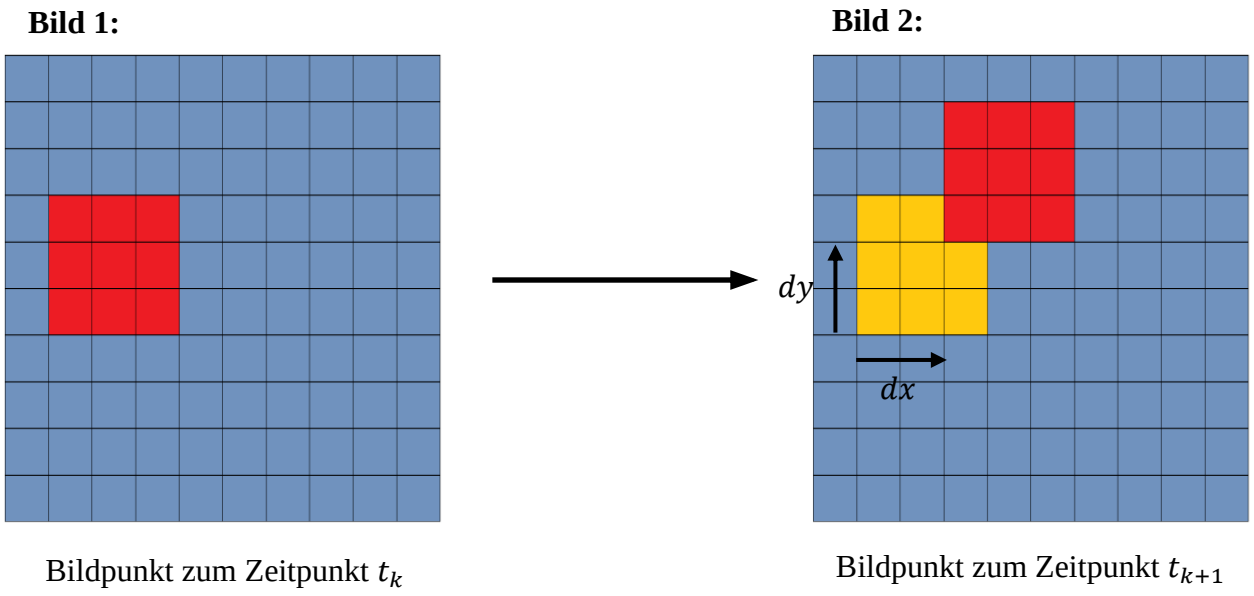


Abbildung 12: Grundlegende Funktionsweise von differentiellen Verfahren zur Berechnung des optischen Flusses

Die Gesamtheit aller Bildpunkte mit deren Geschwindigkeiten zum Zeitpunkt t ergibt das zweidimensionale Geschwindigkeitsvektorfeld des optischen Flusses. Das Ziel aller differentiellen Verfahren ist, das Geschwindigkeitsvektorfeld mit Hilfe der Änderung der Intensitätswerte einer Bildfolge zu schätzen [11].

Betrachtet man Intensität $I(x, y, t)$ und die Geschwindigkeit v_m des Pixels $m = [x \quad y]^T$,

$$v_m = \dot{m} = \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{bmatrix} \tag{2.6}$$

so gehen die Horn-Schunck Methode als auch die Lucas-Kanade Methode davon aus, dass sich der Intensitätswert I eines betrachteten Bildpixels über die Zeit dt nicht ändert. Diese Annahme wird oft als *„Brightness Constancy Constraint“* bezeichnet [9; 12].

$$I(x + v_x \cdot dt, y + v_x \cdot dt, t + dt) = I(x, y, t) \tag{2.7}$$

Geht man ebenso davon aus, dass sich die Position des betrachteten Pixels von Zeitpunkt t zu Zeitpunkt $t + dt$ nur geringfügig ändert, so lässt sich die linke Seite der Gleichung (2.7) mit Hilfe einer Taylor-Reihe erster Ordnung linearisieren. So ergibt sich der in Gleichung (2.9) dargestellte Zusammenhang. Diese Gleichung wird oft als „*motion constraint equation*“ oder auch „gradient-constraint equation“ bezeichnet [9; 12].

$$\begin{gathered} I(x, y, t) + \frac{\partial I}{\partial x} \cdot v_x \cdot dt + \frac{\partial I}{\partial y} \cdot v_y \cdot dt + \frac{\partial I}{\partial t} \cdot dt + O(dt^2) \\ = I(x, y, t) \end{gathered} \tag{2.8}$$

$$\Leftrightarrow \frac{\partial I}{\partial x} \cdot v_x + \frac{\partial I}{\partial y} \cdot v_y + \frac{\partial I}{\partial t} = 0$$

$$\Leftrightarrow I_x \cdot v_x + I_y \cdot v_y + I_t = 0 \tag{2.9}$$

Die Schätzung des Geschwindigkeitsvektors mit Hilfe der „*motion constraint equation*“ ist jedoch nicht allein möglich, da die Gleichung zwei unbekannte Variablen besitzt und somit unterbestimmt ist. Dies führt zu einen Blendenfehler.

Die Bekannten I_x und I_y berechnen sich mit Hilfe eines Sobel-Konvolution-Filters für jedes Pixel zum Zeitpunkt t. I_t ergibt sich durch $I(x, y, t) - I(\, x, y, t + dt)$ [13].

Für die Berechnung der beiden Unbekannten v_x und v_y der Gleichung (2.9) verfolgen die Lucas-Kanade Methode und die Horn-Schunck Methode unterschiedliche Ansätze.

2.4.2 Lucas-Kanade Methode

Die Lucas-Kanade Methode betrachtet hierzu nicht nur ein Pixel, sondern auch dessen Nachbarpixel. Das Originalbild wird Schritt für Schritt durch eine Maske betrachtet. Man nimmt in jedem Abschnitt eine konstante Geschwindigkeit an. $q_1, q_2, q_3, \cdots, q_n$ beschreiben die Pixel im inneren der Maske. Somit ergeben sich n Gleichungen [14; 15].

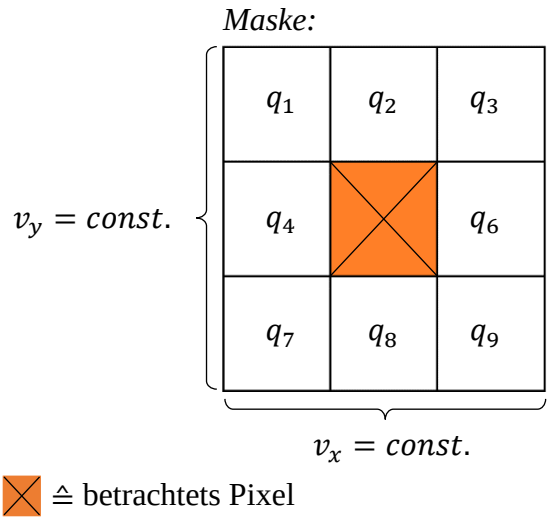


Abbildung 13: Beispiel eines 3x3 Maske mit 9 Gleichungen

$$
\begin{aligned}
I_x(q_1) \cdot v_x + I_y(q_1) \cdot v_y &= -I_t(q_1) \\
I_x(q_2) \cdot v_x + I_y(q_2) \cdot v_y &= -I_t(q_2) \\
&\vdots \\
I_x(q_n) \cdot v_x + I_y(q_n) \cdot v_y &= -I_t(q_n)
\end{aligned}
$$

Diese Gleichungen können in der Matrix-Form $y = A \cdot x$ beschrieben werden [14; 15].

$$
A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix} \qquad x = \begin{bmatrix} v_x \\ v_y \end{bmatrix} \qquad y = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_n) \end{bmatrix} \tag{2.10}
$$

Nun ist eine Schätzung des Vektors x mit Hilfe der Methode der kleinsten Quadrate[4] möglich [14; 15].

[4] Weitere Infos zur Methode der kleinsten Quadrate: Vorlesung „Experimental Modelling and Simulation“ von Herrn Prof. Dr. Peter Zentgraf an der Technischen Hochschule Rosenheim

$$x = (A^T \cdot A)^{-1} \cdot A^T \cdot y \tag{2.11}$$

$$\vdots$$

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} I_x(q_i)^2 & \sum_{i=1}^{n} I_x(q_i) \cdot I_y(q_i) \\ \sum_{i=1}^{n} I_y(q_i) \cdot I_x(q_i) & \sum_{i=1}^{n} I_y(q_i)^2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} -\sum_{i=1}^{n} I_x(q_i) \cdot I_t(q_i) \\ -\sum_{i=1}^{n} I_y(q_i) \cdot I_t(q_i) \end{bmatrix} \tag{2.12}$$

Die in Gleichung (2.11) beschriebene Lösung für x nach der Methode der kleinsten Quadrate gewichtet alle betrachteten Pixel in der Maske mit gleicher Gewichtung. In der Praxis ist es vorteilhafter die Pixel, welche näher im Zentrum der Maske liegen, mehr zu gewichten. Hierzu verwendet man die gewichtete Version der Methode der kleinsten Quadrate. W ist die $n \times n$ diagonale Gewichtungsmatrix. Sie enthält dabei die zum Pixel q_i dazugehörige Gewichtung w_i [13].

$$W = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & w_n \end{bmatrix} \tag{2.13}$$

$$x = (A^T \cdot W \cdot A)^{-1} \cdot A^T \cdot W \cdot y \tag{2.14}$$

$$\vdots$$

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} w_i \cdot I_x(q_i)^2 & \sum_{i=1}^{n} w_i \cdot I_x(q_i) \cdot I_y(q_i) \\ \sum_{i=1}^{n} w_i \cdot I_y(q_i) \cdot I_x(q_i) & \sum_{i=1}^{n} w_i \cdot I_y(q_i)^2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} -\sum_{i=1}^{n} w_i \cdot I_x(q_i) \cdot I_t(q_i) \\ -\sum_{i=1}^{n} w_i \cdot I_y(q_i) \cdot I_t(q_i) \end{bmatrix} \tag{2.15}$$

Für die Gewichtung der einzelnen Pixel in der Maske wird meistens ein Gauß-Filter verwendet [13].

Grundsätzlich ist die Lucas-Kanade Methode ein einfaches und leicht zu implementierendes Verfahren, welches ein breites Einsatzfeld aufweist. Die Annahme eines konstanten optischen Flusses in einer kleinen Umgebung um das betrachtete Pixel ist dabei oft tragbar. Da es sich aber um eine lokale Methode zur Schätzung der Pixelgeschwindigkeit handelt, wird meistens kein

dichtes Vektorfeld errechnet. Mit diesem Verfahren können besonders gut die Flussinformationen von Objekträndern geschätzt werden. Mit wachsendem Abstand zu den Rändern wird die Berechnung immer schwieriger [16].

2.4.3 Horn-Schunck Methode

Im Gegensatz zur Lucas-Kanade Methode ist die Horn-Schunck Methode ein globales Verfahren. Hierbei wird angenommen, dass sich das Vektorfeld über das Bild nur gleichmäßig ändert, woraus sich ein glattes Feld ergibt. Würde sich jeder Punkt unabhängig von den umgebenen Punkten bewegen, gäbe es keine Möglichkeit eine Bewegung zu schätzen. Dazu wird eine Glattheitsbedingung definiert. Als Kriterium sollen hierbei die Quadrate der Gradientenbeträge der Verschiebungsvektoren minimiert werden [17].

$$|\nabla v_x| = \sqrt{\left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2} \qquad |\nabla v_y| = \sqrt{\left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2} \tag{2.16}$$

Zusätzlich zu den Glattheitstermen (2.16) muss auch die Gleichung (2.9) minimiert werden. Daraus ergibt sich die für alle Pixel Ω das zu minimierende Funktional $E(v_x, v_y)$ [17].

$$E(v_x, v_y) = \iint_{\Omega} \left(\left(I_x \cdot v_x + I_y \cdot v_y + I_t\right)^2 + \alpha^2 \cdot \left(|\nabla v_x|^2 + |\nabla v_y|^2\right)\right) dxdy$$

$$\Leftrightarrow E(v_x, v_y) = \iint_{\Omega} \left(\left(I_x \cdot v_x + I_y \cdot v_y + I_t\right)^2 + \alpha^2 \cdot \left(\left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2 + \left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2\right)\right) dxdy \tag{2.17}$$

Der konstante Faktor α^2 gibt die Gewichtung des zu minimierenden Glattheitsterms an. Je größer α^2 umso stärker wird das Vektorfeld geglättet. Mit Hilfe der zugehörigen Euler-Lagrange Gleichungen kann das Funktional $\mathcal{L}\left(x, y, v_x, v_y, \frac{\partial v_x}{\partial x}, \frac{\partial v_x}{\partial y}, \frac{\partial v_y}{\partial x}, \frac{\partial v_y}{\partial y}\right)$ für alle unbekannten minimiert werden [18].

$$\mathcal{L}\left(x, y, v_x, v_y, \frac{\partial v_x}{\partial x}, \frac{\partial v_x}{\partial y}, \frac{\partial v_y}{\partial x}, \frac{\partial v_y}{\partial y}\right) =$$

$$= \left(I_x \cdot v_x + I_y \cdot v_y + I_t\right)^2 + \alpha^2 \cdot \left(\left(\frac{\partial v_x}{\partial x}\right)^2 + \left(\frac{\partial v_x}{\partial y}\right)^2 + \left(\frac{\partial v_y}{\partial x}\right)^2 + \left(\frac{\partial v_y}{\partial y}\right)^2\right) \quad (2.18)$$

Hierbei ergeben sich folgende Gleichungen.

$$I_x{}^2 \cdot v_x + I_x \cdot I_y \cdot v_y = \alpha^2 \cdot \frac{\partial}{\partial x}\left(\frac{\partial v_x}{\partial x}\right) + \alpha^2 \cdot \frac{\partial}{\partial y}\left(\frac{\partial v_x}{\partial y}\right) - I_x \cdot I_t$$

$$I_x \cdot I_y \cdot v_x + I_y{}^2 \cdot v_y = \alpha^2 \cdot \frac{\partial}{\partial x}\left(\frac{\partial v_y}{\partial x}\right) + \alpha^2 \cdot \frac{\partial}{\partial y}\left(\frac{\partial v_y}{\partial y}\right) - I_y \cdot I_t \quad (2.19)$$

Durch weiteres Umformen, einer Transformation in den Laplace-Bereich und dem Lösen der Gleichungen mit Hilfe des iterativen Gauß-Seidel Verfahren erhält man folgenden Zusammenhang zur Berechnung des optischen Flusses [18; 17].

$$v_x{}^{n+1} = \bar{v}_x{}^n - I_x \cdot \frac{I_x \cdot \bar{v}_x{}^n + I_y \cdot \bar{v}_y{}^n + I_t}{\alpha^2 + I_x{}^2 + I_y{}^2}$$

$$v_y{}^{n+1} = \bar{v}_y{}^n - I_y \cdot \frac{I_x \cdot \bar{v}_x{}^n + I_y \cdot \bar{v}_y{}^n + I_t}{\alpha^2 + I_x{}^2 + I_y{}^2} \quad (2.20)$$

$\bar{v}_x{}^n$ und $\bar{v}_y{}^n$ ist die durchschnittliche geschätzte Geschwindigkeit der Nachbarpixel. Diese kann mit Hilfe eines Konvolution-Filters und dem Kernel [0 1 0; 1 0 1; 0 1 0] berechnet werden [13].

Die genauen Rechenschritte können in den Literaturen [17] und [18] nachgeschlagen werden.

Generell ist die Horn-Schunck Methode ein sehr rechenintensives Verfahren. Hierbei werden oft hunderte Iterationen für ein genügend dichtes Vektorfeld benötigt. Durch den Glattheitsterm werden jedoch die Geschwindigkeitsvektoren von Stellen mit höheren Gradientenbeträgen in Bereiche mit schwachen Gradienten projiziert. Dadurch ergibt sich ein dichteres Vektorfeld. Ebenso wurde herausgefunden, dass das Horn-Schunck Verfahren eine höhere Anfälligkeit auf Rauschen im Gegensatz zu lokalen Methoden aufweist [16].

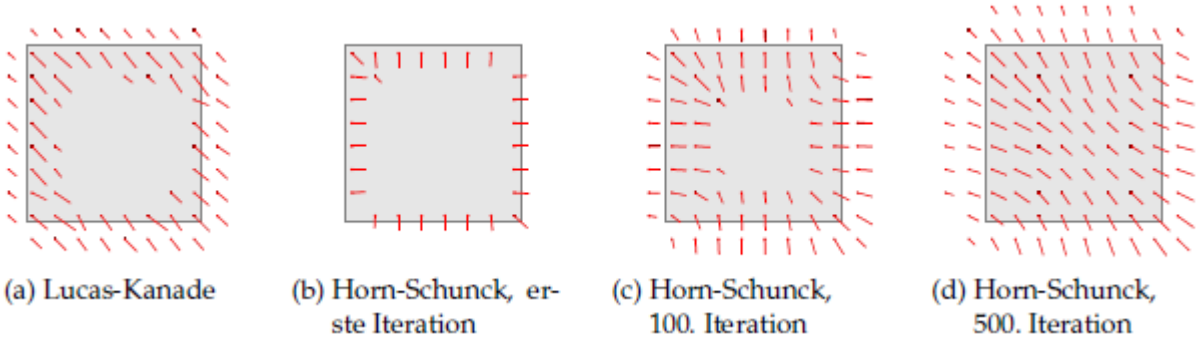


Abbildung 14: Vergleich der Lucas-Kanade Methode (a) mit der Horn-Schunck Methode (b, c, d) mit drei unterschiedlichen Iterationsstufen [19]

2.5 Bilderfassung mit Parrot Minidrones

Die Basis für die Bilderfassung bildet die Minidrone RollingSpider der amerikanischen Firma Parrot SAS.

Abbildung 15: Parrot Rolling Spieder
Quelle: https://upload.wikimedia.org/wikipedia/commons/6/66/Rolling_Spider.jpg

Der Vorteil dieses Quadrokopters ist, dass hierfür bereits ein Matlab sowie Simulink Support Package existiert. Dies ermöglicht eine einfache Programmierung mit den verwendeten Softwarekomponenten. Des Weiteren können so erarbeitete Algorithmen schnell und einfach an dem realen Quadrokopter getestet werden. Die Drohne ist ebenso bereits mit zahlreichen Sensoren ausgestattet, welche mit Hilfe des Simulink Support Package einfach ausgelesen und eingebunden werden können. Die Sensoren sind in der nachfolgenden Tabelle aufgelistet.

Tabelle 1: Sensoren der Minidrone RollingSpider

	Wirkrichtung	**Abtastzeit**	**Sonstiges**
RGB-Kamera	In Richtung der Z-Achse	200ms	120x160 Pixel
Beschleunigungssensor	Um die X-, Y,- und Z-Achse	5ms	$\left[{}^{m}/_{s^2}\right]$
Gyroskop	Um die X-, Y,- und Z-Achse	5ms	$\left[{}^{rad}/_{s}\right]$
Ultraschallsensor	In Richtung der Z-Achse	5ms	$[m]$
Drucksensor	/	5ms	$[Pa]$
Temperatursensor	/	5ms	$[°C]$

3 Vorhandene optische Positions- und Lagebestimmung

Die Firma MathWorks ermöglicht mit ihrem MATLAB Add-on „*Simulink Support Package for Parrot Minidrones*“ Flugsteuerungen für Parrot Minidrones zu entwickeln, zu entwerfen und zu testen. Es werden zahlreiche vorprogrammierte Modell-Beispiele zur Verfügung gestellt. Diese erleichtern die Entwicklung von eigenen Algorithmen. In dem Beispielmodell „*Follow Set of Waypoints or Follow Orbit Using Parrot Minidrone*” wurde bereits von MathWorks eine Positions- und Lagebestimmung des Quadrokopters mit Hilfe dessen eingebauten Sensoren implementiert [20].

Diese Positions- und Lagebestimmung gereift auf die Sensordaten des eingebauten Beschleunigungssensors, Gyroskops, Umgebungsdrucksensors, Ultraschallsensors sowie der durch die Firmware ermittelten X- und Y-Verschiebung des Quadrokopters zurück. Die in der Firmware ermittelten X- und Y-Verschiebung wird mit Hilfe der nach unten gerichteten Kamera des Quadrokopters und einem optischen Fluss Algorithmus[5] ermittelt. Eine genauere Beschreibung der einzelnen von der Firmware des Quadrokopters bereitgestellten Sensordaten des Quadrokopters sind in der Tabelle 1 dargestellt. Abbildung 16 gibt einen kurzen Überblick über den grundsätzlichen Aufbau dieser Positions- und Lagebestimmung [20; 21].

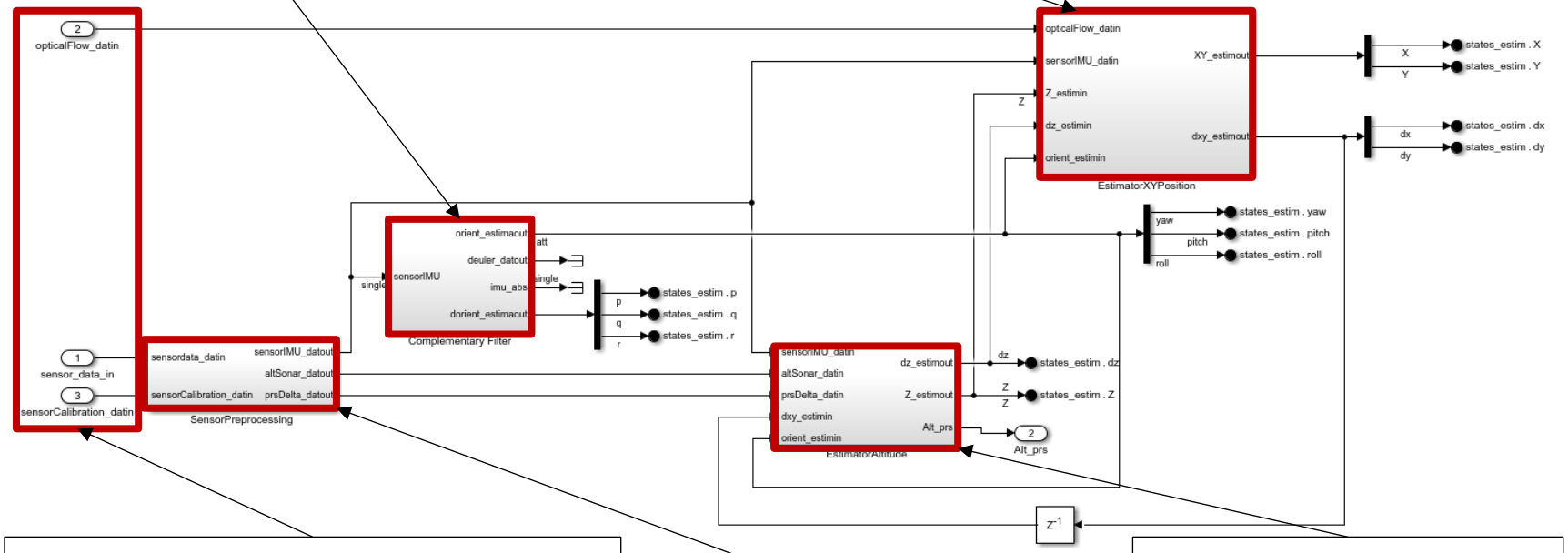
[5] Siehe Kapitel 2.4

Lagebestimmung:

Schätzung der Quadrokopterlage mit Hilfe der Beschleunigungssensoren und des Gyroskops

XY-Positionsbestimmung:

Schätzung der XY-Position des Quadrokopters mit Hilfe von Bildverarbeitung, der ermittelten Lage, Geschwindigkeit und Z-Position



Input:

- Optischer Fluss Daten
- Sensor Daten (Beschleunigungsmesser, Gyroskop, Ultraschall, Umgebungsdruck)
- Kalibrierdaten der Sensoren

Sensor-Preprocessing:

- Kalibrierung der Sensordaten
- Filterung der Sensordaten

Höhenberechnung:

Schätzung der Quadrokopterhöhe mit Hilfe des Umgebungsdrucks, des Ultraschallsensors sowie der Quadrokopter-Lage

Abbildung 16: Übersicht des Aufbaus der Positions- und Lagebestimmung

3.1 Qualifizierung der vorhandenen optischen Positions- und Lagebestimmung

Um die Funktionsweise sowie die Genauigkeit der hier verwendeten Algorithmen zur Positions- und Lagebestimmung des Quadrokopters im Raum ermitteln zu können, wird vorab ein Versuch durchgeführt.

3.1.1 Ziel

Hierbei wir die Genauigkeit der XY-Positionsbestimmung des Quadrokopters in verschiedenen Höhen im Raum näher analysiert, ohne auf die einzelnen Algorithmen näher einzugehen.

Anhand der Ergebnisse dieses Versuchs soll entschlossen werden, ob die bereits von MathWorks implementierten Algorithmen weiter untersucht, optimiert oder ein eigener Ansatz zur optischen Positions- und Lagebestimmung des Quadrokopters erarbeitet wird.

3.1.2 Versuchsaufbau

Zur Qualifizierung der bereits implementierten XY-Positionsbestimmung des Quadrokopters wird dieser an einer beweglichen Vorrichtung befestigt. Mit Hilfe dieser Vorrichtung kann der Quadrokopter in einer konstanten Höhe im Raum bewegt werden. Die Höhe des Quadrokopters ist in verschiedenen Stufen einstellbar.

Bei der Befestigung des Quadrokopters an der Vorrichtung wird darauf geachtet, keine Sensoren zu beeinträchtigen. Des Weiteren wird der Quadrokopter so montiert, dass sich keine Bauteile der Vorrichtung im Blickfeld der nach unten gerichteten Kamera des Quadrokopters befinden.

Darüber hinaus wird ein möglichst strukturreicher Untergrund gewählt, um eine bestmögliche Positions- und Lagebestimmung zu ermöglichen.

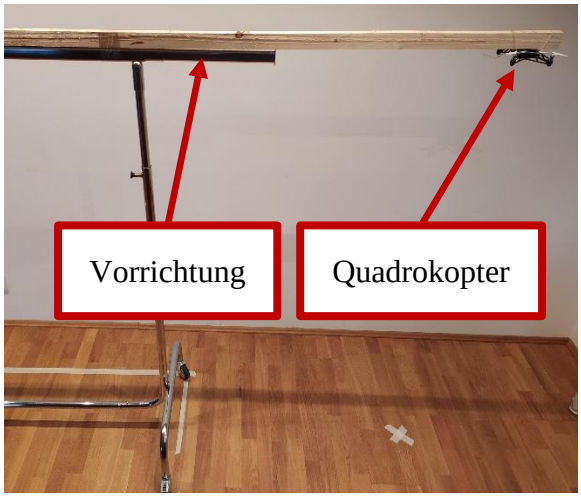


Abbildung 17: Versuchsaufbau des Versuch 1

3.1.3 Versuchsdurchführung

Zur Messdatenaufnahme wird der Quadrokopter in drei verschiedenen Höhen jeweils dreimal entlang dessen X-Achse und dreimal entlang dessen Y-Achse um 2 Meter verschoben. Es wird versucht mit Hilfe von Markierungen am Boden bei allen Messdatenaufnahmen eine möglichst geradlinige, konstante, gleichmäßige und schwingungsfreie Bewegung zu erzielen. Alle Einzelversuche werden ebenso mit möglichst gleicher Geschwindigkeit und Beschleunigung durchgeführt. Abbildung 18 zeigt die einzelnen Bewegungsrichtungen des Quadrokopters während des Versuchs.

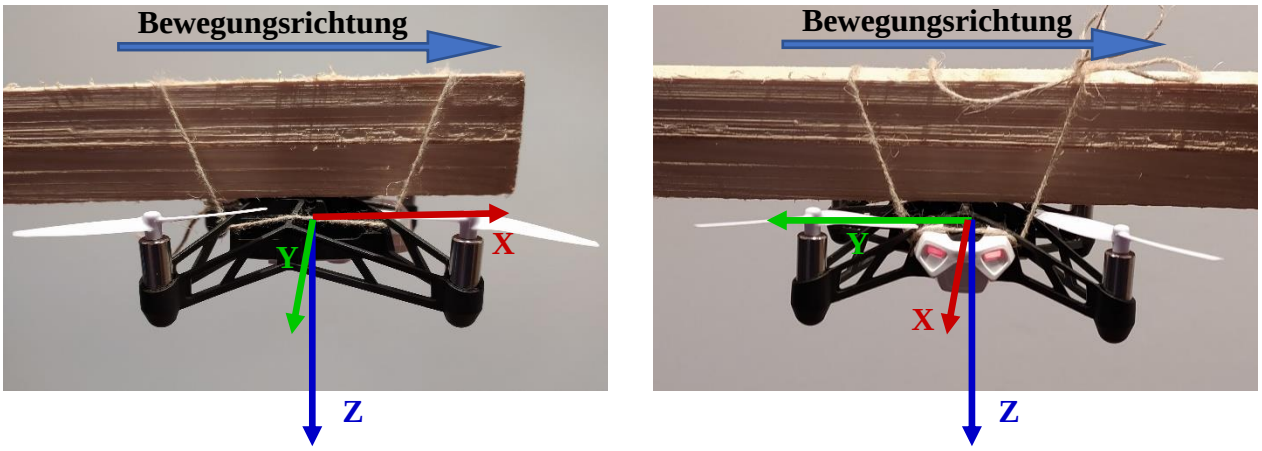


Abbildung 18: Bewegungsrichtungen des Quadrokopters

Tabelle 2 listet nachfolgend die exakte Durchführung des Versuches auf.

Tabelle 2: Versuchsdurchführung

Nr.:	Bewegungsrichtung	Höhe	Strecke $P_{START} - P_{STOPP}$	Anzahl
1.1	positive X-Richtung	1,5 Meter	2 Meter	3-mal
1.2	positive X-Richtung	1,75 Meter	2 Meter	3-mal
1.3	positive X-Richtung	2 Meter	2 Meter	3-mal
2.1	negative Y-Richtung	1,5 Meter	2 Meter	3-mal
2.2	negative Y-Richtung	1,75 Meter	2 Meter	3-mal
2.3	negative Y-Richtung	2 Meter	2 Meter	3-mal

In Abbildung 19 werden die verwendeten Bodenmarkierungen dargestellt. Die aufgeklebten Kreuze stellen den Start-Punkt P_{START} und den Stopp-Punkt P_{STOPP} des Quadrokopters dar.

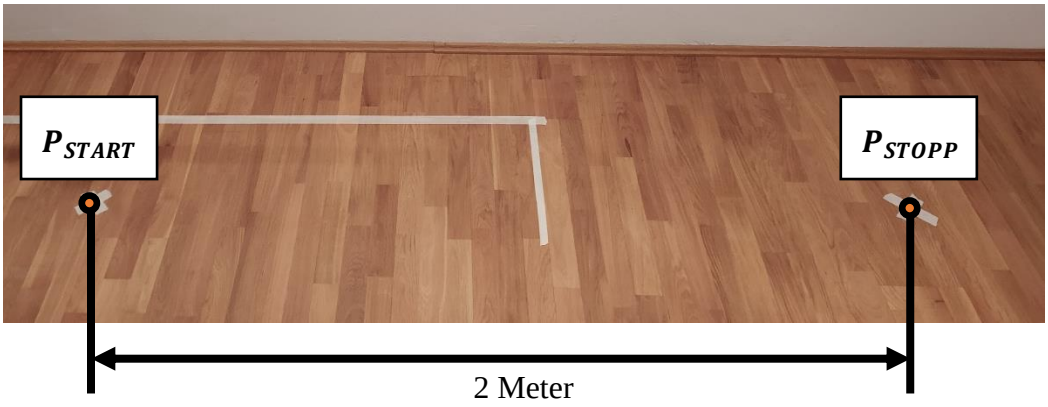


Abbildung 19: Bodenmarkierungen des 1. Versuchs

Die Ergebnisse jedes Versuchs werden mit Hilfe von Matlab aufgezeichnet, ausgelesen und anschließend gespeichert. Die Rohdaten sind in dem Ordner „*03_Datensätze\01_Bestandsaufnahme_XY-Verschiebung_Simulink*“ beigefügt.

3.2 Ergebnis

Nach der Versuchsdurchführung werden die Messdaten sortiert. Die von dem oben dargestellten Algorithmus ermittelten Positionen des Quadrokopters im Raum werden in Abhängigkeit von der Zeit dargestellt. Die Ergebnisse sind in Abbildung 20 und Abbildung 21 dargestellt.

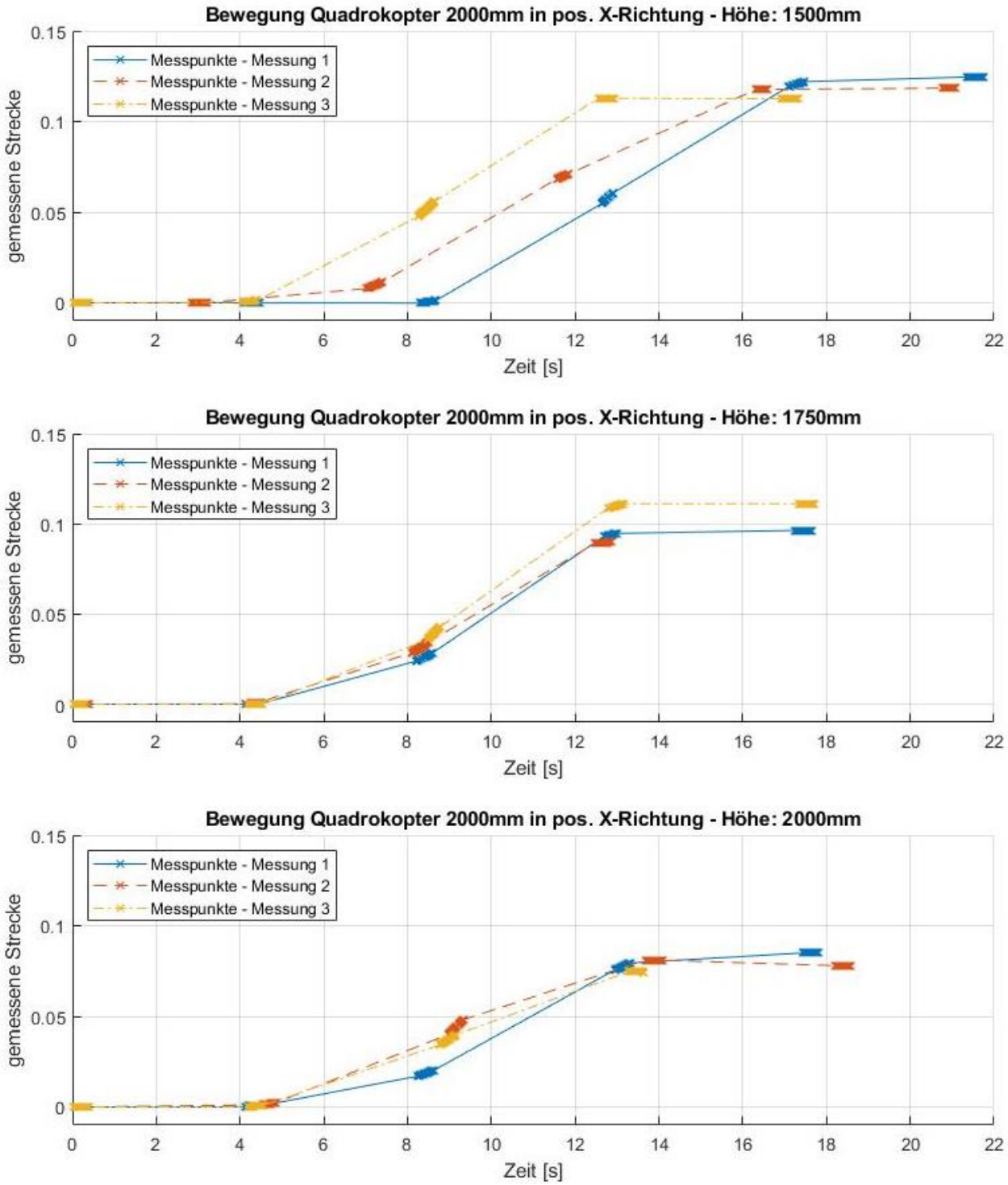


Abbildung 20: Ergebnisse der Position entlang der X-Achse

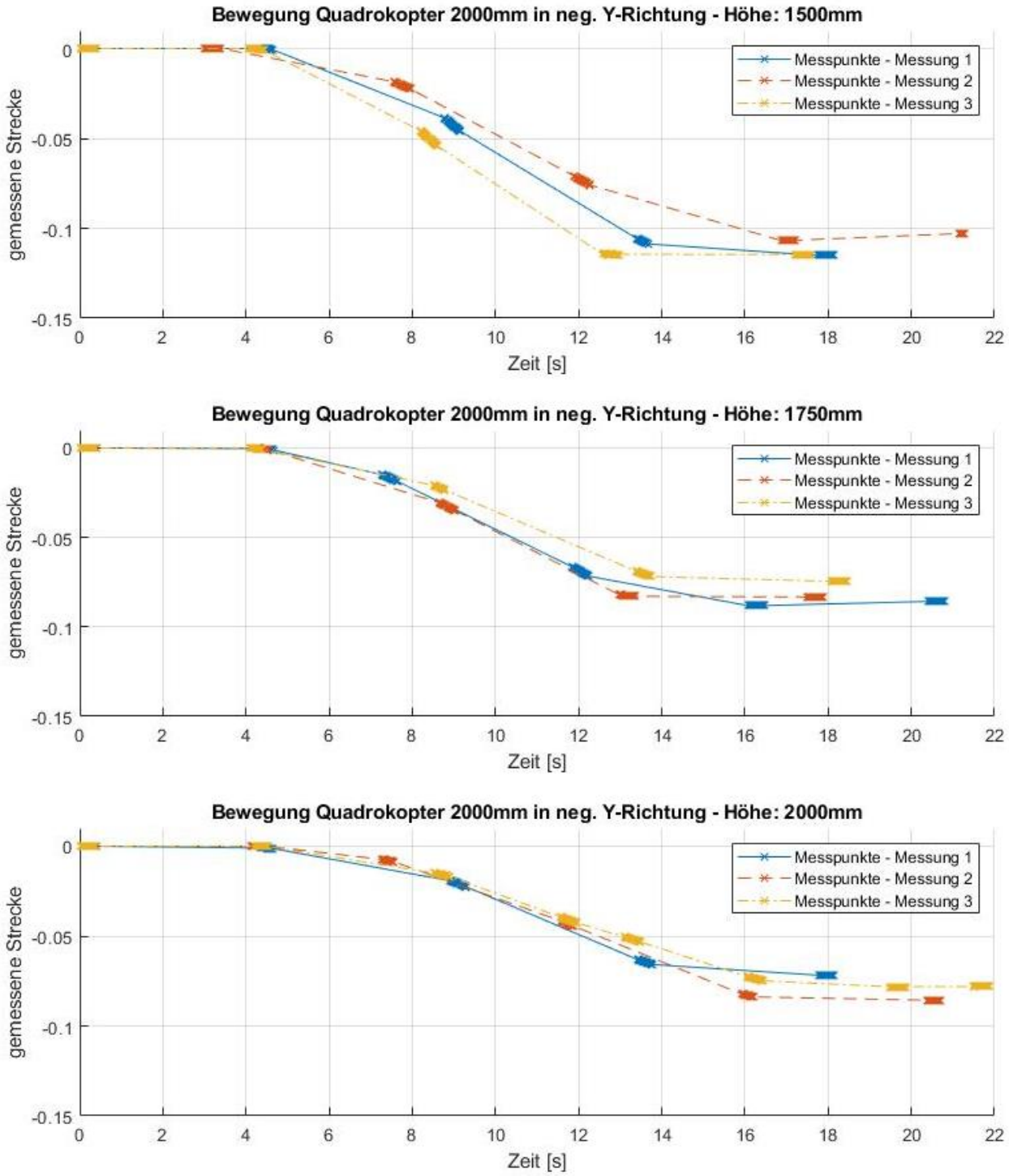


Abbildung 21: Ergebnisse der Position entlang der Y-Achse

Betrachtet man die Ergebnisse genauer so fällt auf, dass die geschätzte Position des Quadrokopters abhängig von dessen Höhe im Raum ist. Dies lässt sich sowohl in X-Richtung als auch in Y-Richtung des Body-Koordinatensystems beobachten.

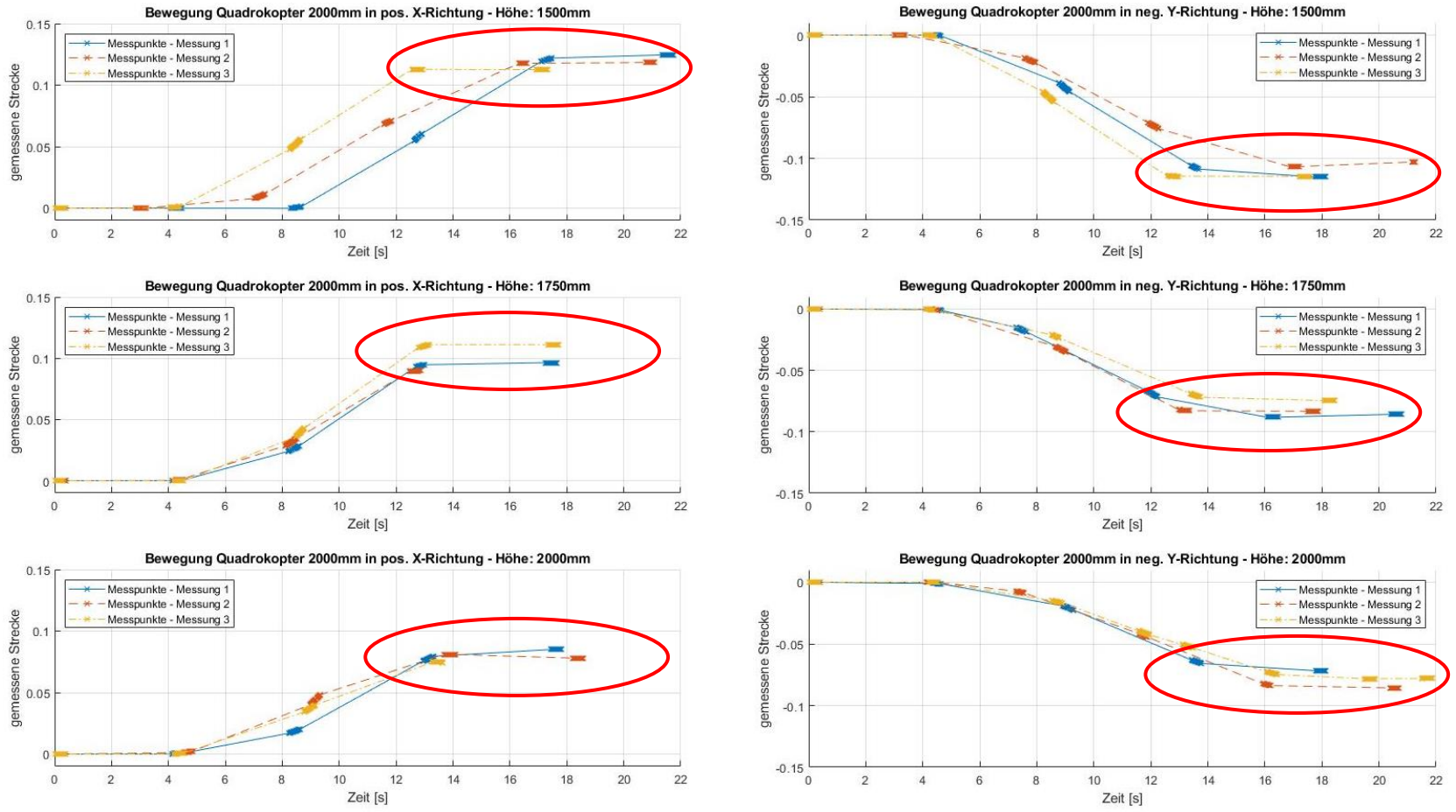


Abbildung 22: Höhenabhängigkeit der Messdaten

Des Weiteren ist in den Versuchen keine konstante Aufnahme von Messdaten zu erkennen. Die Abtastzeit variiert in allen Versuchsreihen von 5 ms bis zu 4565 ms.

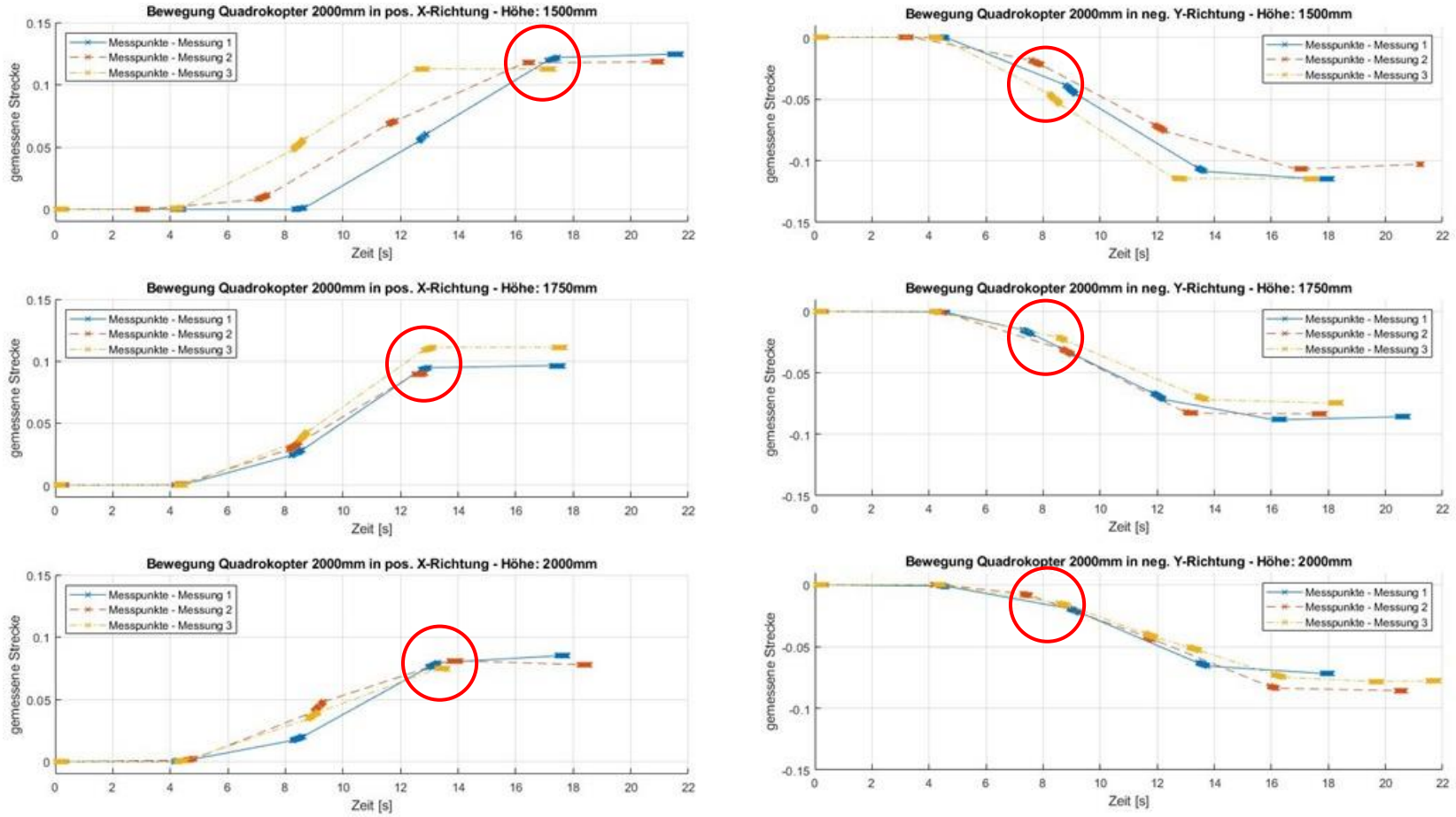


Abbildung 23: Keine konstante Messdatenaufnahme

Außerdem geht aus dem von MathWorks bereitgestellten Daten und dem Simulink-Modell nicht die genaue Einheit der geschätzten Position des Quadrokopters in X- und Y-Richtung hervor. Es wird angenommen, dass die Position in Meter geschätzt wird. Die in den Versuchsreihen bestimmten Positionen stehen somit in keiner Relation zu der tatsächlichen Strecke von 2 m.

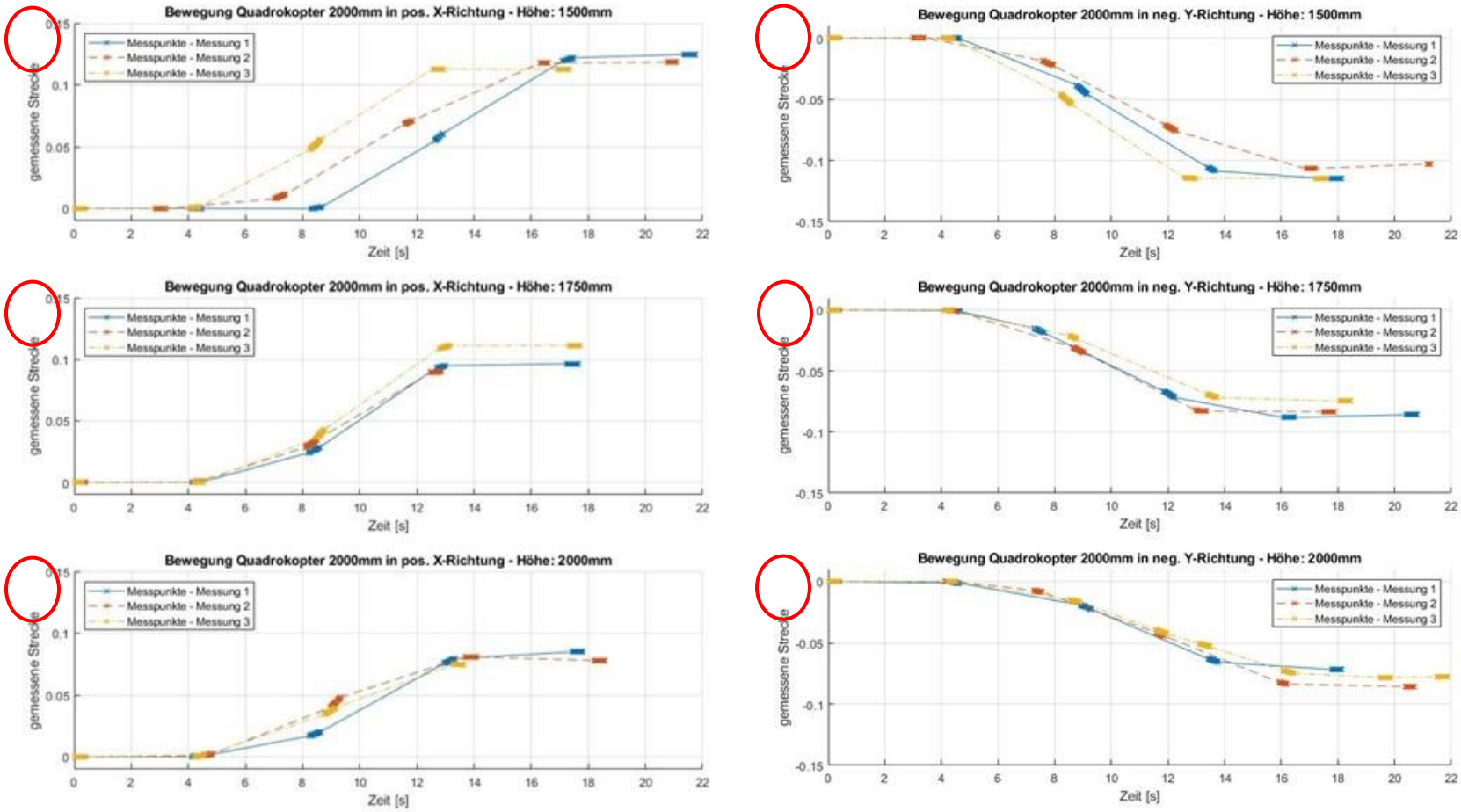


Abbildung 24: Keine eindeutige Einheit

Zusätzlich zu den oben aufgeführten Unsicherheiten ist die genaue Berechnung des optischen Flusses sowie die hierfür verwendeten Algorithmen nicht bekannt. Die hier verwendeten optischen Fluss Daten werden bereits in der Firmware des Quadrokopters berechnet. Auch nach einer Kontaktaufnahme mit dem Hersteller sind diese nicht zugänglich.

Aus diesen Gründen wird nachfolgend, unabhängig von dem von MathWorks implementierten Ansatz, ein Verfahren zur Positions- und Lagebestimmung eines Quadrokopters erarbeitet. Dieses soll mit Hilfe einer entlang der Z-Achse des Quadrokopters gerichteten Kamera geschätzt werden. Basis der optischen Positions- und Lagebestimmung ist die in Abschnitt 0 beschriebene Parrot Minidrone.

4 Optische 3D Positions- und Lagebestimmung

Betrachtet man die Bewegung einer Kamera in einem dreidimensionalen Raum genauer, so lässt sich mit Hilfe des durch einen optischen Fluss Algorithmus berechneten Geschwindigkeitsvektorfeldes, viel über die Bewegung des Quadrokopters ableiten.

Bewegt sich die Kamera ausschließlich geradlinig entlang deren Blickrichtung, so ist das in Abbildung 25 dargestellte Vektorfeld zu erkennen. Dieses breitet sich radial von einem bestimmten Punkt aus. Dieser Punkt wird „*Focus of Expension*“ genannt. Er zeigt die Bewegungsrichtung der Kamera an [22].

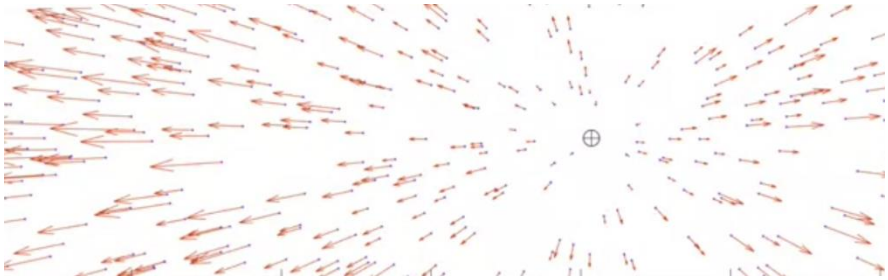


Abbildung 25: Geschwindigkeitsvektorfeld einer translatorischen Bewegung einer Kamera [22]

Dreht dich die Kamera nun schwenkend nach rechts, so erhält man das in Abbildung 26 dargestellte Vektorfeld. Es entsteht ein gleichmäßiges, fast horizontales Vektorfeld von rechts nach links. Am linken und rechten Rand beginnt sich das Vektorfeld zu krümmen und die Vektoren werden größer [22].

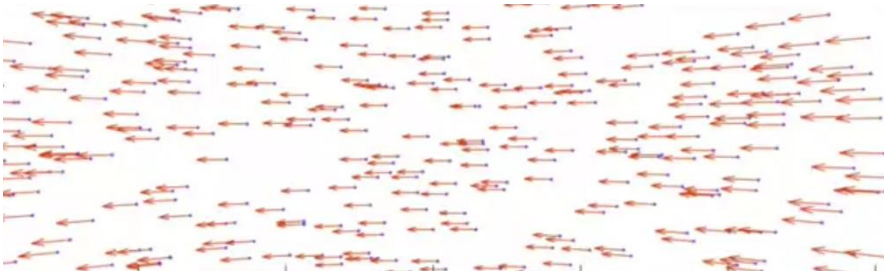


Abbildung 26: Geschwindigkeitsvektorfeld einer rotatorischen Bewegung einer Kamera [22]

Dreht man die Kamera ausschließlich um die optische Achse, so entsteht ein rotierendes Vektorfeld. Auch in diesem Fall ist die Richtung der optischen Achse gut zu erkennen [22].

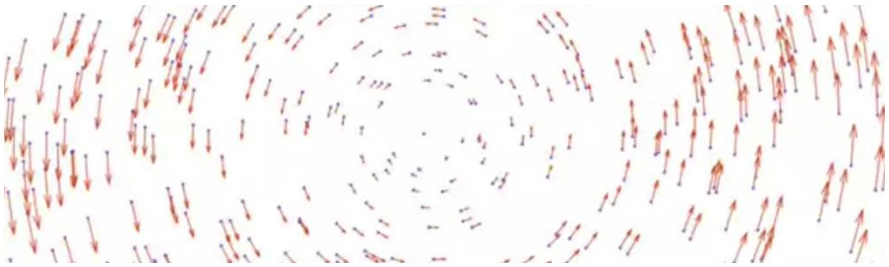


Abbildung 27: Geschwindigkeitsvektorfeld einer um die optische Achse rotierende Kamera [22]

Kombiniert man unterschiedliche rotatorische und translatorische Bewegungen einer Kamera, so lässt sich in dessen Geschwindigkeitsvektorfeld, wie in Abbildung 28 zu erkennen, kein klares Muster beobachten. Um trotzdem aus diesem Vekorfeld Informationen über die Bewegung der Kamera zu erhalten, wird im nächsten Schritt die algebraische Berechung eines auf die Bildebene projizierten Geschwindigkeitsvektorfeldes genauer betrachtet [22].

Abbildung 28: Geschwindigkeitsvektorfeld einer beliebigen Bewegung einer Kamera [22]

4.1 Algebraische Berechnung eines Geschwindigkeitsvektorfeldes

Eine Bewegung einer Kamera kann mit Hilfe einer translatorischen Geschwindigkeit $v = [v_x \quad v_x \quad v_z]^T$ sowie einer rotatorischen Winkelgeschwindigkeit $\Omega = [\omega_x \quad \omega_y \quad \omega_z]^T$ beschrieben werden. Betrachtet man einen fixen Punkt P im Raum, so bewegt sich dieser im bewegten Kamera-Koordinatensystem mit der Geschwindigkeit $\dot{P}$. Diese Geschwindigkeit kann durch die translatorische und rotatorische Geschwindigkeit des Kamerakoordinatensystems sowie der Position des Punktes P im Kamera-Koordinatensystem dargestellt werden [22].

$$\dot{P} = -\underbrace{\Omega \times P}_{\text{rotatorische Geschwindigkeit}} - \underbrace{v}_{\text{translatorische Geschwindigkeit}} \tag{4.1}$$

$$\begin{bmatrix}\dot{X}\\ \dot{Y}\\ \dot{Z}\end{bmatrix} = -\begin{bmatrix}0 & -\omega_z & \omega_y\\ \omega_z & 0 & -\omega_x\\ -\omega_y & \omega_x & 0\end{bmatrix} \times \begin{bmatrix}X\\ Y\\ Z\end{bmatrix} - \begin{bmatrix}v_x\\ v_y\\ v_z\end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix}\dot{X}\\ \dot{Y}\\ \dot{Z}\end{bmatrix} = \begin{bmatrix}\omega_z \cdot Y - \omega_y \cdot Z - v_x\\ -\omega_z \cdot X + \omega_x \cdot Z - v_y\\ \omega_y \cdot X - \omega_x \cdot Y - v_z\end{bmatrix} \qquad (4.2)$$

Ebenso ist der Zusammenhang zwischen der Position des Punktes sowie dessen Position auf der Bildebene des Kamerasensors, durch eine zuvor durchgeführte geometrische Kamerakalibrierung, bekannt. Wie bereits in Abschnitt 2.3 gezeigt, lässt sich mit Hilfe einer geometrischen Kamerakalibrierung das vorliegende Kamerasystem auf das Modell einer Lochkamera zurückführen. Hit Hilfe der intrinsischen Kalibiermatrix K [6] können die Positionen der einzelnen Pixel des Pixel-Koordinatensystem in das Kamera-Koordinatensystem transformiert werden.

Die Kalibiermatrix K enthält dabei die geschätzten Brennweiten f_x und f_y der Kamera. Mit den Gleichungen (2.1) und (2.2) kann so folgendes Verhältnis aufgestellt werden.

$$x_d(t) = f_x \cdot \frac{X(t)}{Z(t)} \qquad y_d(t) = f_y \cdot \frac{Y(t)}{Z(t)} \qquad (4.3)$$

$$\Leftrightarrow \begin{bmatrix}x_d(t)\\ y_d(t)\end{bmatrix} = \frac{1}{Z(t)} \cdot \begin{bmatrix}f_x \cdot X(t)\\ f_y \cdot Y(t)\end{bmatrix} \qquad (4.4)$$

$$\Leftrightarrow \begin{bmatrix}X(t)\\ Y(t)\end{bmatrix} = Z(t) \cdot \begin{bmatrix}\frac{1}{f_x} \cdot x_d(t)\\ \frac{1}{f_y} \cdot y_d(t)\end{bmatrix} \qquad (4.5)$$

[6] Siehe Formel (2.3)

Setzt man nun die Gleichung (4.5) in die Gleichung (4.2) ein, erhält man:

$$\begin{bmatrix} \dot{X}(t) \\ \dot{Y}(t) \\ \dot{Z}(t) \end{bmatrix} = \begin{bmatrix} \omega_z \cdot \frac{y_d(t)}{f_y} \cdot Z(t) - \omega_y \cdot Z(t) - v_x \\ -\omega_z \cdot \frac{x_d(t)}{f_x} \cdot Z(t) + \omega_x \cdot Z(t) - v_y \\ \omega_y \cdot \frac{x_d(t)}{f_x} \cdot Z(t) - \omega_x \cdot \frac{y_d(t)}{f_y} \cdot Z(t) - v_z \end{bmatrix} \tag{4.6}$$

Leitet man zusätzlich die Gleichung (4.4) nach der Zeit ∂t ab, so errechnet sich die Geschwindigkeit $\dot{p}$ des Punktes p im Kamera-Koordinatensystem.

$$\frac{\partial}{\partial t}\begin{bmatrix} x_d(t) \\ y_d(t) \end{bmatrix} = \frac{\partial}{\partial t}\left(\frac{1}{Z(t)} \cdot \begin{bmatrix} f_x \cdot X(t) \\ f_y \cdot Y(t) \end{bmatrix}\right)$$

$$\Leftrightarrow \begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \end{bmatrix} = \begin{bmatrix} f_x \cdot \frac{\dot{X}(t) \cdot Z(t) - X(t) \cdot \dot{Z}(t)}{Z(t) \cdot Z(t)} \\ f_y \cdot \frac{\dot{Y}(t) \cdot Z(t) - Y(t) \cdot \dot{Z}(t)}{Z(t) \cdot Z(t)} \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \end{bmatrix} = \begin{bmatrix} f_x \cdot \frac{\dot{X}(t) \cdot \cancel{Z(t)}}{Z(t) \cdot \cancel{Z(t)}} - f_x \cdot \frac{X \cdot \dot{Z}(t)}{Z(t) \cdot Z(t)} \\ f_y \cdot \frac{\dot{Y}(t) \cdot \cancel{Z(t)}}{Z(t) \cdot \cancel{Z(t)}} - f_y \cdot \frac{Y(t) \cdot \dot{Z}(t)}{Z(t) \cdot Z(t)} \end{bmatrix}$$

Mit Gleichung (4.4):

$$\begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \end{bmatrix} = \begin{bmatrix} f_x \cdot \frac{\dot{X}(t)}{Z(t)} - x_d(t) \cdot \frac{\dot{Z}(t)}{Z(t)} \\ f_y \cdot \frac{\dot{Y}(t)}{Z(t)} - y_d(t) \cdot \frac{\dot{Z}(t)}{Z(t)} \end{bmatrix} \tag{4.7}$$

Um nun den Geschwindigkeitsvektor $\dot{p}$ eines definierten Pixels im Kamera-Koordinatensystem berechnen zu können, muss die Gleichung (4.6) in die Gleichung (4.7) eingesetzt werden. Der daraus folgende Zusammenhang ist dabei unabhängig von der X- und Y- Position des Fixpunktes P im Raum.

$$\Leftrightarrow \begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \end{bmatrix} =$$

$$= \begin{bmatrix} \frac{f_x}{Z(t)}\left(\omega_z \cdot \frac{y_d(t)}{f_y} \cdot Z(t) - \omega_y \cdot Z(t) - v_x\right) - x_d(t) \cdot \frac{\omega_y \cdot \frac{x_d(t)}{f_x} \cdot Z(t) - \omega_x \cdot \frac{y_d(t)}{f_y} \cdot Z(t) - v_z}{Z(t)} \\ \frac{f_y}{Z(t)}\left(-\omega_z \cdot \frac{x_d(t)}{f_x} \cdot Z(t) + \omega_x \cdot Z(t) - v_y\right) - y_d(t) \cdot \frac{\omega_y \cdot \frac{x_d(t)}{f_x} \cdot Z(t) - \omega_x \cdot \frac{y_d(t)}{f_y} \cdot Z(t) - v_z}{Z(t)} \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \end{bmatrix} =$$

$$= \begin{bmatrix} \frac{f_x \cdot y_d(t)}{f_y}\omega_z - f_x \cdot \omega_y - \frac{f_x}{Z(t)} \cdot v_x - \frac{{x_d}^2(t)}{f_x} \cdot \omega_y + \frac{x_d(t) \cdot y_d(t)}{f_y} \cdot \omega_x + \frac{x_d(t)}{Z(t)} \cdot v_z \\ \left(-\frac{f_y \cdot x_d(t)}{f_x}\omega_z + f_y \cdot \omega_x - \frac{f_y}{Z(t)} \cdot v_y\right) - \frac{x_d(t) \cdot y_d(t)}{f_x} \cdot \omega_y + \frac{{y_d}^2(t)}{f_y} \cdot \omega_x + \frac{y_d(t)}{Z(t)} \cdot v_z \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \end{bmatrix} =$$

$$\begin{bmatrix} -\frac{f_x}{Z(t)} \cdot v_x + \frac{x_d(t)}{Z(t)} \cdot v_z + \frac{x_d(t) \cdot y_d(t)}{f_y} \cdot \omega_x - \left(f_x + \frac{{x_d}^2(t)}{f_x}\right) \cdot \omega_y + \frac{f_x \cdot y_d(t)}{f_y}\omega_z \\ -\frac{f_y}{Z(t)} \cdot v_y + \frac{y_d(t)}{Z(t)} \cdot v_z + \left(f_y + \frac{{y_d}^2(t)}{f_y}\right) \cdot \omega_x - \frac{x_d(t) \cdot y_d(t)}{f_x} \cdot \omega_y - \frac{f_y \cdot x_d(t)}{f_x}\omega_z \end{bmatrix} \quad (4.8)$$

$$\Leftrightarrow \begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{f_x}{Z(t)} & 0 & \frac{x_d(t)}{Z(t)} \\ 0 & -\frac{f_y}{Z(t)} & \frac{y_d(t)}{Z(t)} \end{bmatrix}}_{translatorischer\ Fluss} \cdot \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$$

$$+ \underbrace{\begin{bmatrix} \frac{x_d(t) \cdot y_d(t)}{f_y} & -\left(f_x + \frac{{x_d}^2(t)}{f_x}\right) & \frac{f_x \cdot y_d(t)}{f_y} \\ \left(f_y + \frac{{y_d}^2(t)}{f_y}\right) & -\frac{x_d(t) \cdot y_d(t)}{f_x} & -\frac{f_y \cdot x_d(t)}{f_x} \end{bmatrix}}_{rotatorischer\ Fluss} \cdot \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \quad (4.9)$$

Alle für die algebraische Berechnung des projizierten Geschwindigkeitsvektorfeldes eines dreidimensionalen Körpers benötigten Größen werden in der Abbildung 29 zur Veranschaulichung skizzenhaft dargestellt.

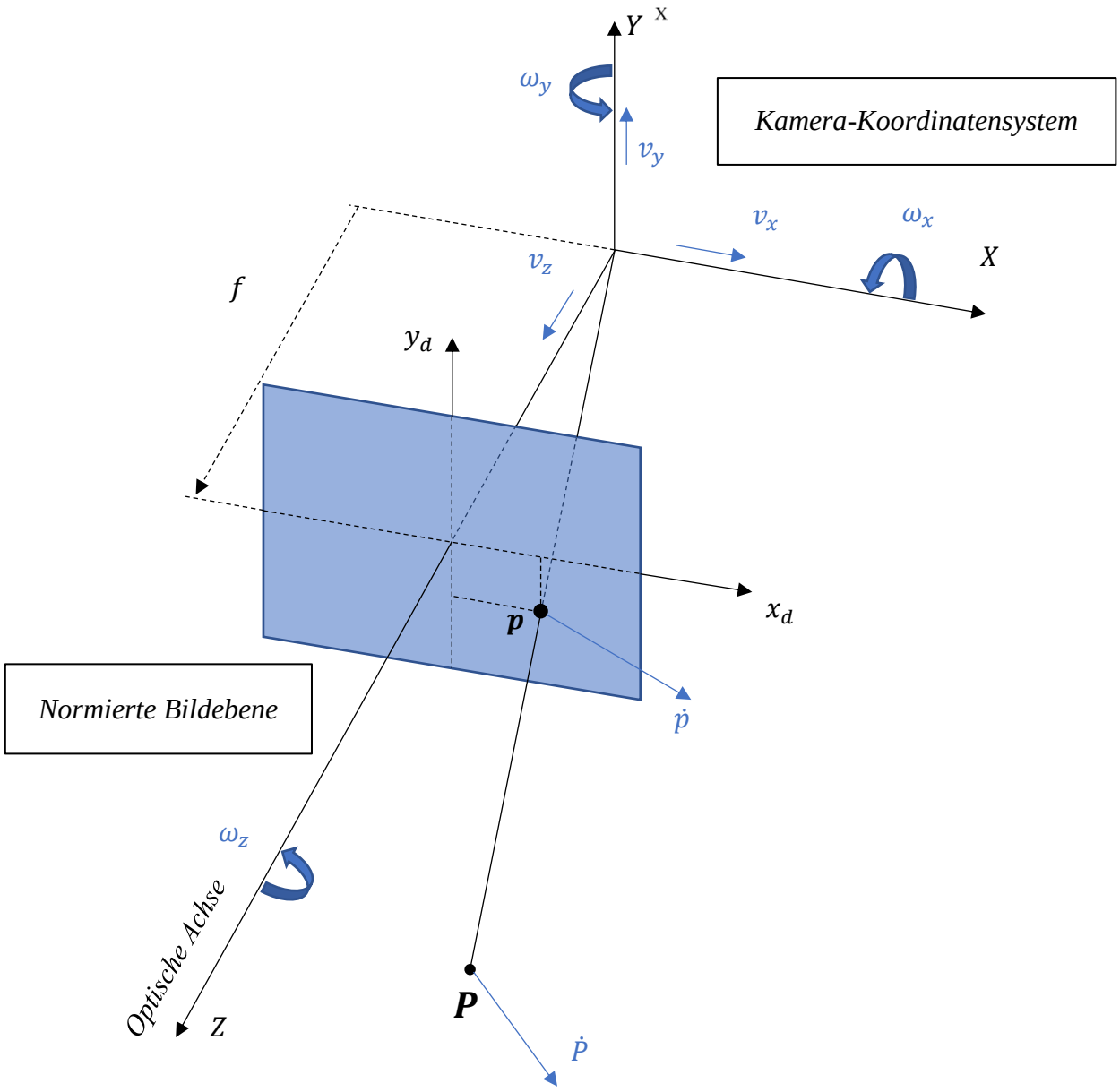


Abbildung 29: Skizze zur algebraischen Berechnung des projizierten Geschwindigkeitsvektorfeldes eines dreidimensionalen Körpers

Betrachtet man die Gleichung (4.9) genauer so fällt auf, dass der rotatorische Fluss nicht abhängig vom Abstand Z des stationären Punktes P zum Ursprung des Kamera-Koordinatensystems ist. Im Gegensatz dazu fließt dieser Abstand in die Berechnung des translatorischen Flusses ein.

4.2 Erweiterung des algebraischen Geschwindigkeitsvektorfeldes

In dem in Abschnitt 0 beschriebenen Fall, zeigt die fest am Quadrokopter montierte Kamera immer entlang deren positiven Z-Achse. Zusätzlich wird davon ausgegangen, dass der von der Kamera betrachtete Boden eine ebene Fläche darstellt und alle betrachteten Punkte P_i auf dieser liegen.

Ist die Lage der XY-Ebene der Kamera und somit auch die der Drohne parallel zum Inertial-Koordinatensystem, so ist der Abstand jedes Punkts P_i zum Boden identisch. Dieser kann mit Hilfe eines im Quadrokopter montierten Ultraschallsensors oder einem Umgebungsdrucksensors ermittelt werden.

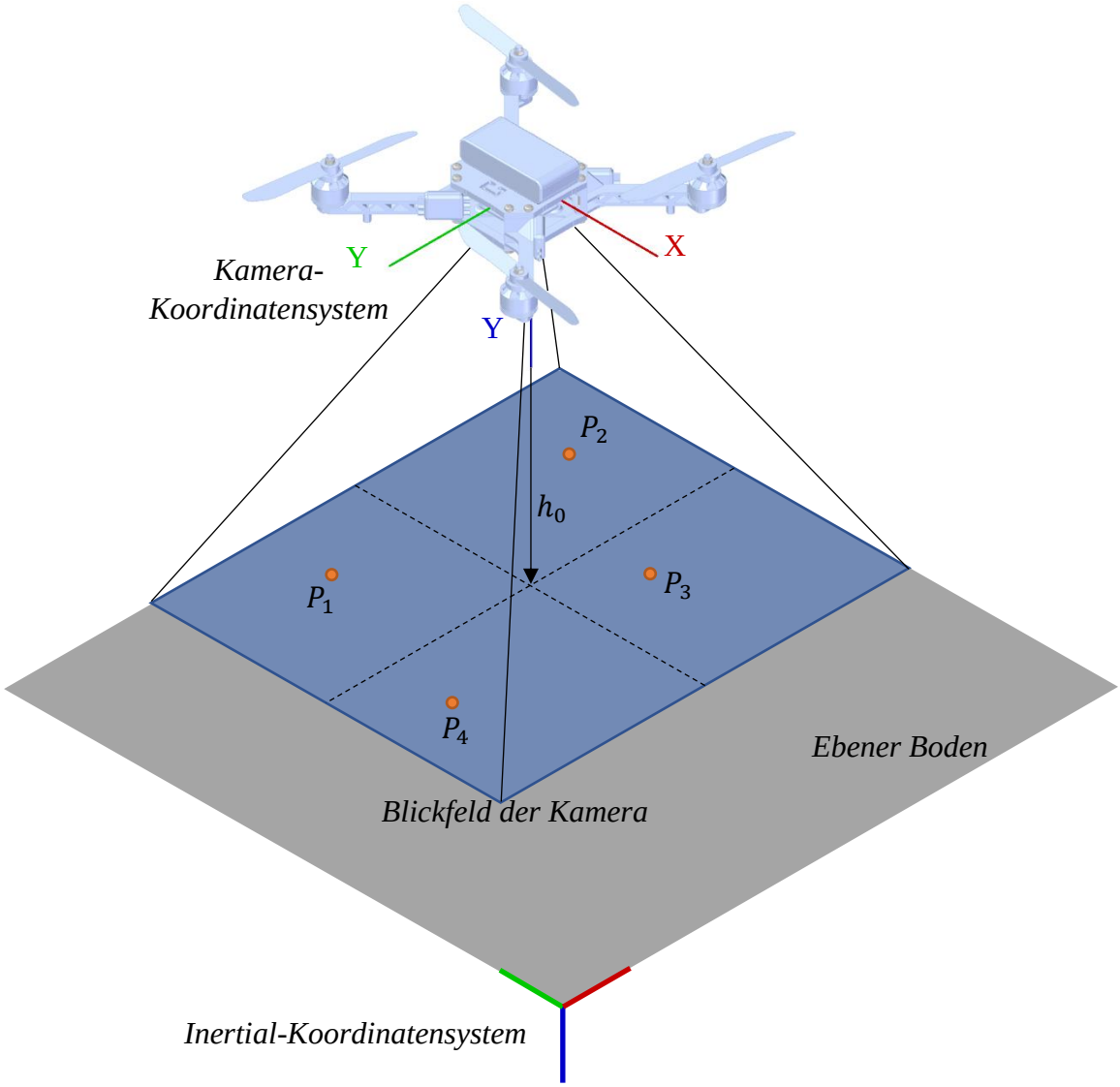


Abbildung 30: Skizze - Quadrokopter parallel zum Inertial-Koordinatensystem

Da eine Positionsänderung des Quadrokopters immer durch eine Lageänderung des Quadrokopters verursacht wird, kann nicht davon ausgegangen werden, dass sich das Kamera-Koordinatensystem immer parallel zum Inertial-Koordinatensystem befindet. Auch in dem nicht parallelen Zustand des Quadrokopters soll dessen Lange sowie dessen Geschwindigkeit mit Hilfe des hier erarbeiteten Verfahrens angenähert werden können. Hierzu müssen für einen gegebenen Abstand h_0 zwischen dem Kamera-Koordinatensystem und dem Boden die einzelnen Abstände jedes beliebigen Punktes P_i zum Kamera-Koordinatensystem berechnet werden. Es wird davon ausgegangene, dass sich alle Punkte P_i auf einer Ebene befinden und die Höhe h_0 senkrecht auf dieser steht. Nachfolgend wird ein repräsentativer Punkt P genauer betrachtet.

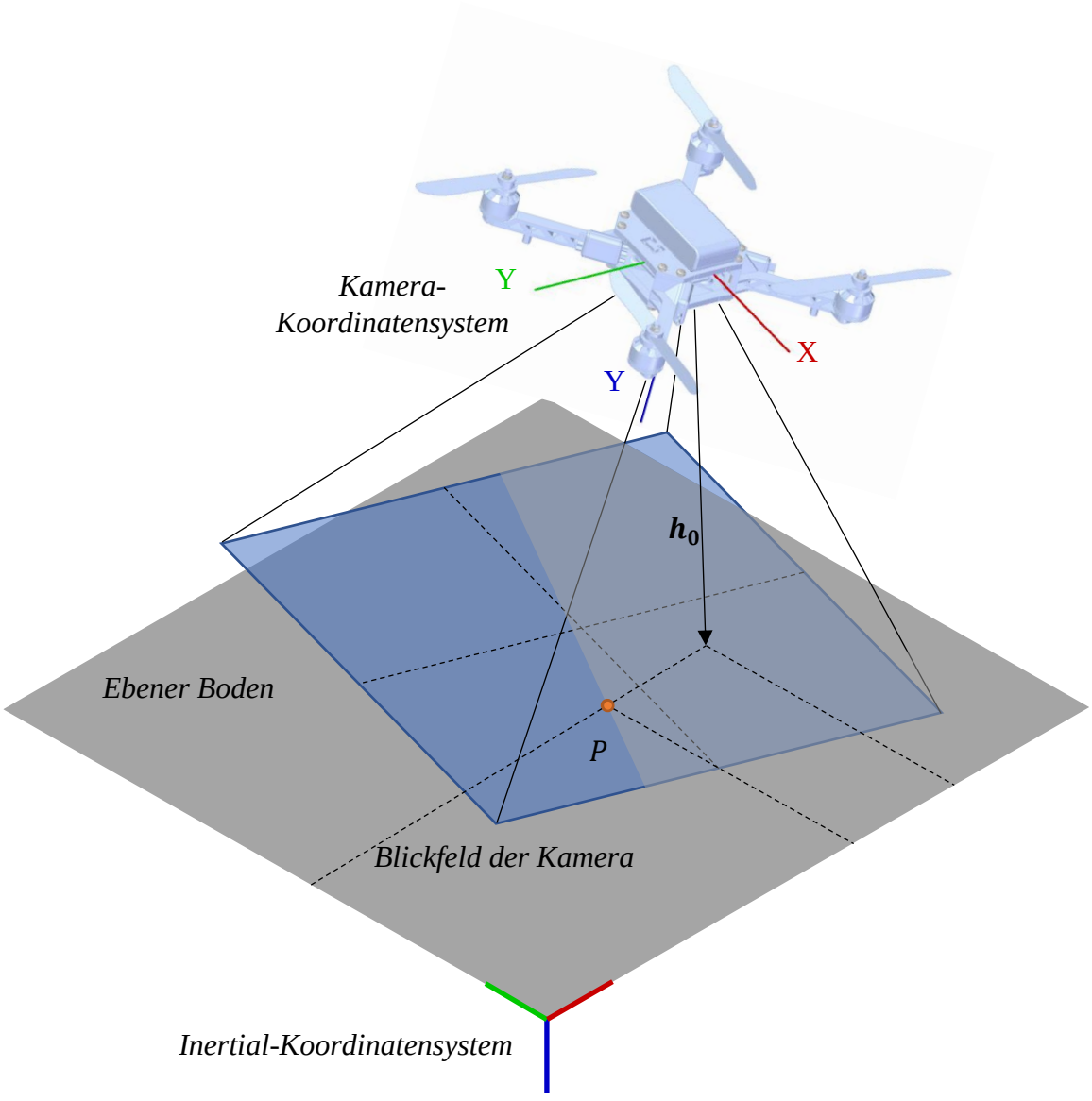


Abbildung 31: Skizze - Quadrokopter nicht parallel zum Inertial-Koordinatensystem

4.2.1 Bekannte Größen

Die Position des Punktes P kann im Kamera-Koordinatensystem als Vektor $\vec{P}_K = [X_K \quad Y_K \quad Z_K]^T$ dargestellt werden. Der tiefgestellte Buchstabe gibt an, in welchem Koordinatensystem der Vektor beschrieben wird. Hierbei steht K für das Kamera-Koordinatensystem in I für das Inertial-Koordinatensystem.

Der Vektor $\vec{P}_K$ ist ein vielfachtes des Vektors $\vec{p}_K = [x_n \quad y_n \quad 1]^T$.

$$\vec{P}_K = n \cdot \vec{p}_K \tag{4.10}$$

$\vec{p}_K = [x_n \quad y_n \quad 1]^T$ ist die normierte Position eines Pixels (u, v) im Kamerakoordinatensystem. Die Berechnung des Vektors $\vec{p}_K$ erfolgt nach Gleichung (2.4) mit Hilfe der intrinsischen Kamera-Matrix K und der Pixelposition $[u \quad v \quad 1]^T$.

H_I steht senkrecht auf der XY-Ebene des Inertial-Koordinatensystems und gibt den aktuell gemessenen Abstand der Kamera zur Grundfläche an.

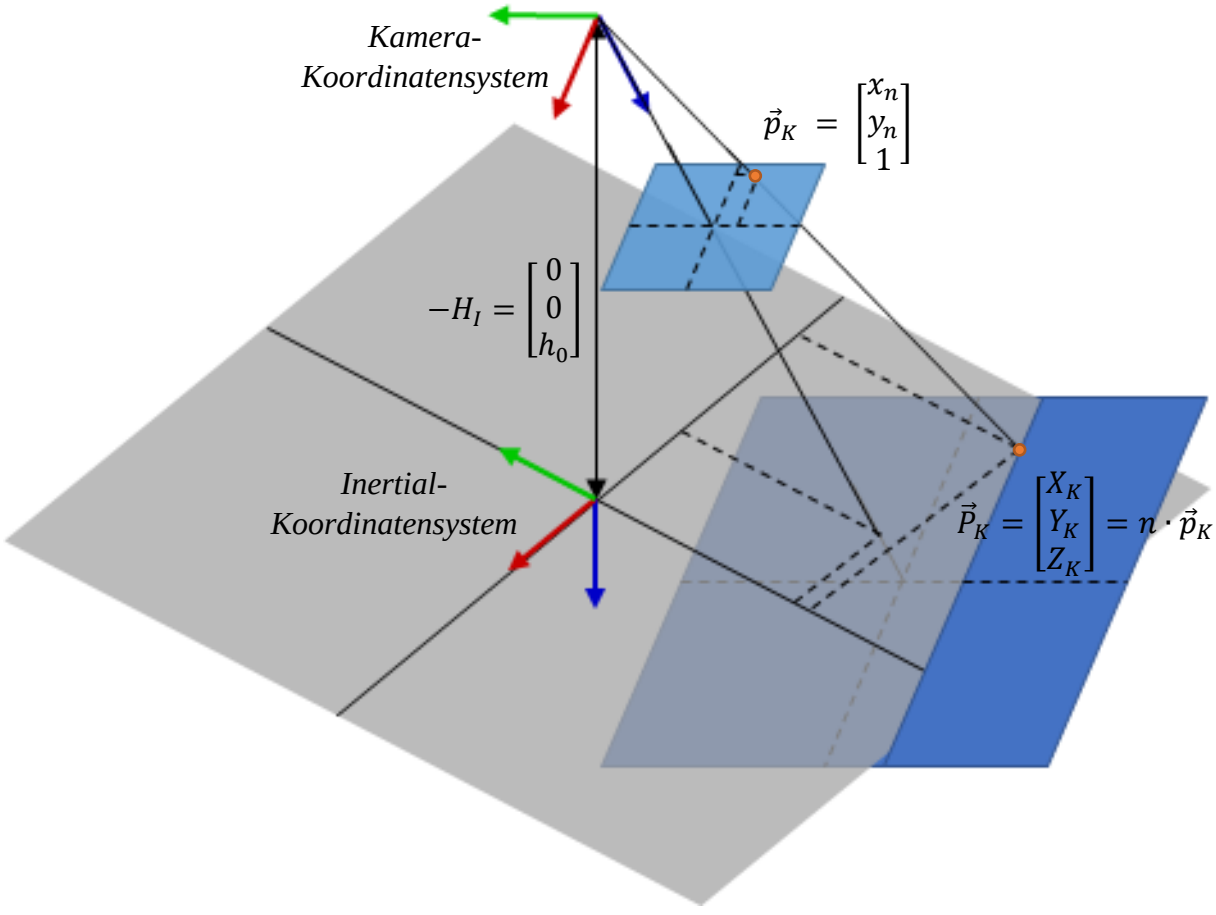


Abbildung 32: Skizze der bekannten Vektoren

4.2.2 Koordinatentransformation zwischen Inertial- und Kamera-Koordinatensystem

Eine allgemeine Drehung eines Körpers im Raum kann durch drei aufeinanderfolgende Elementardrehungen um die Koordinatenachsen des Körpers beschrieben werden. Die eindeutige Definition der Drehreihenfolge wird mit Hilfe einer kinematischen Kette definiert. Je nach Drehreihenfolge ergibt sich eine unterschiedliche Transformationsmatrix.

Ganz allgemein und unabhängig von den unterschiedlichen Drehreihenfolgen kann die Drehung des Kamera-Koordinatensystem gegenüber dem Inertial-Koordinatensystem mit der Transformationsmatrix T_K^I beschrieben werden. Der obere Index beschreibt dabei das Start-Koordinatensystem, der untere Index das Ziel-Koordinatensystem in wessen Abhängigkeit der Vektor dargestellt werden soll.

$$T_K^I = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{4.11}$$

4.2.3 Geometrischer Zusammenhang

Transformiert man den Vektor $H_I = [0 \quad 0 \quad h_0]^T$ mit Hilfe der Transformationsmatrix T_K^I vom Inertial-Koordinatensystem in das Kamera-Koordinatensystem, so kann die Höhe der Kamera im Kamera-Koordinatensystem dargestellt werden. Der Abstand h_0 lässt sich beispielsweise mit Hilfe der barometrischen Höhenformel sowie den Messdaten des Umgebungsdrucksensors berechnen.

$$H_K = T_K^I \cdot H_I$$

$$\Leftrightarrow H_K = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ h_0 \end{bmatrix}$$

$$\Leftrightarrow H_K = \begin{bmatrix} a_{13} \cdot h_0 \\ a_{23} \cdot h_0 \\ a_{33} \cdot h_0 \end{bmatrix} = \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \end{bmatrix} \cdot h_0 \tag{4.12}$$

Der Einheitsvektor $[a_{13} \quad a_{23} \quad a_{33}]^T$ beschreibt die aktuelle Verdrehung zwischen dem Inertial-Koordinatensystem und dem Kamera-Koordinatensystem. Der Betrag dieses Vektors ist $|[a_{13} \quad a_{23} \quad a_{33}]^T| = 1$.

Mit den trigonometrischen Funktionen können rechnerische Zusammenhänge zwischen einem Winkel und den Seitenverhältnissen eines rechtwinkligen Dreiecks aufgestellt werden. Betrachtet man die in Abbildung 33 skizzierten Vektoren, so lässt sich durch deren Beträge der Winkel θ folgendermaßen beschreiben.

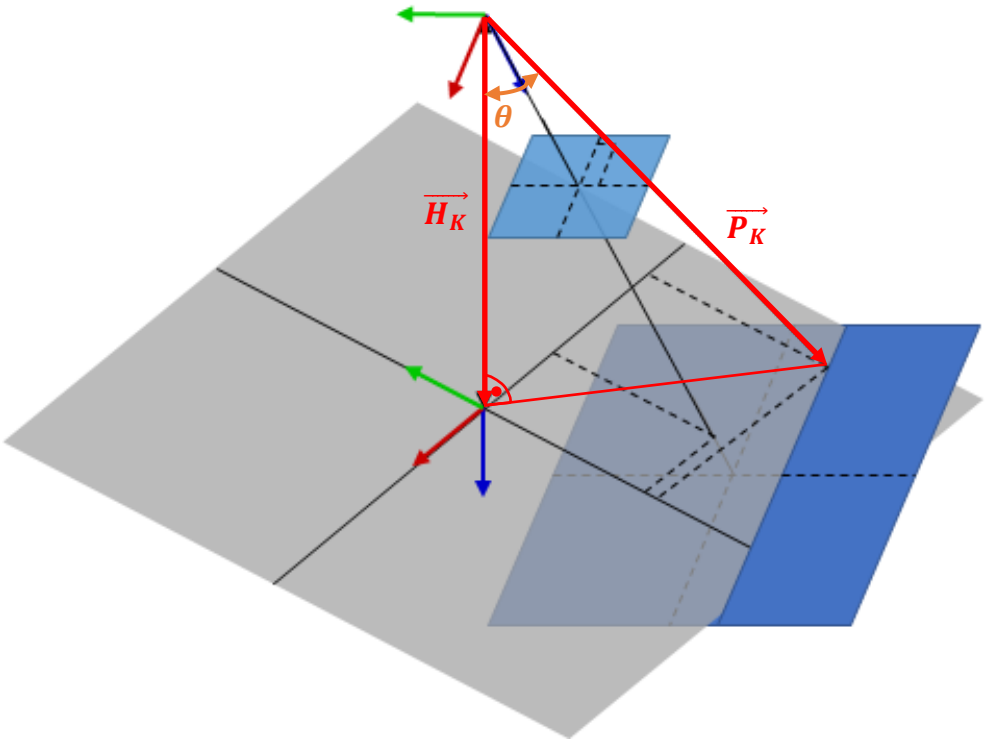


Abbildung 33: Geometrischer Zusammenhang in Bezug auf die Höhe H_K

$$\cos(\theta) = \frac{AK}{HY} = \frac{|\overrightarrow{H_K}|}{|\overrightarrow{P_K}|} \tag{4.13}$$

Mittels der Gleichung (4.10) kann der Vektor $\overrightarrow{P_K}$ als Vielfaches des bekannten Vektors $\overrightarrow{p_K}$ angegeben werden.

$$\cos(\theta) = \frac{|\overrightarrow{H_K}|}{|\overrightarrow{p_K} \cdot n|} = \frac{h_0}{|\overrightarrow{p_K}| \cdot n} \tag{4.14}$$

Ebenso ist die Berechnung des durch die beiden Vektoren $\overrightarrow{H_K}$ und $\overrightarrow{P_K}$ aufgespannten Winkel θ mit Hilfe des Skalarproduktes möglich.

$$\cos(\theta) = \frac{\overrightarrow{H_K} \circ \overrightarrow{P_K}}{|\overrightarrow{H_K}| \cdot |\overrightarrow{P_K}|} = \frac{\left(\overrightarrow{H_K}\right)^T \cdot (\overrightarrow{p_K} \cdot n)}{h_0 \cdot |\overrightarrow{p_K}| \cdot n} \quad f\text{ür } \left(\overrightarrow{H_K} \neq 0,\ \overrightarrow{P_K} \neq 0\right) \tag{4.15}$$

Durch das Einsetzen der Gleichung (4.12) für $\overrightarrow{H_K}$ folgt:

$$\cos(\theta) = \frac{\left(\begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \end{bmatrix} \cdot h_0\right)^T \cdot \left(\begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} \cdot n\right)}{h_0 \cdot |\overrightarrow{p_K}| \cdot n}$$

$$\Leftrightarrow \cos(\theta) = \frac{\cancel{h_0} \cdot \cancel{n} \cdot (a_{13} \cdot x_n + a_{23} \cdot y_n + a_{33})}{\cancel{h_0} \cdot |\overrightarrow{p_K}| \cdot \cancel{n}}$$

$$\Leftrightarrow \cos(\theta) = \frac{(a_{13} \cdot x_n + a_{23} \cdot y_n + a_{33})}{|\overrightarrow{p_K}|} \tag{4.16}$$

Setzt man die beiden Gleichungen (4.14) und (4.16) gleich, erhält man:

$$\frac{h_0}{|\overrightarrow{p_K}| \cdot n} = \frac{(a_{13} \cdot x_n + a_{23} \cdot y_n + a_{33})}{|\overrightarrow{p_K}|}$$

$$\Leftrightarrow \frac{h_0}{n} = (a_{13} \cdot x_n + a_{23} \cdot y_n + a_{33}) \tag{4.17}$$

Um die Z-Koordinaten des betrachteten Punktes P in Abhängigkeit der Höhe h_0 sowie der Lage des Quadrokopters berechnen zu können, muss ein weiterer Zusammenhang für das Vielfache n des Vektors $|\overrightarrow{p_K}|$ erarbeitet werden.

Betrachtet man die Abbildung 34, so können weitere zwei Gleichungen zur Berechnung des Winkels ϑ zwischen den Vektoren $\overrightarrow{Z_K}$ und $\overrightarrow{P_K}$ sowie zwischen $\overrightarrow{e_{z\_K}}$ und $\overrightarrow{p_K}$ aufgestellt werden.

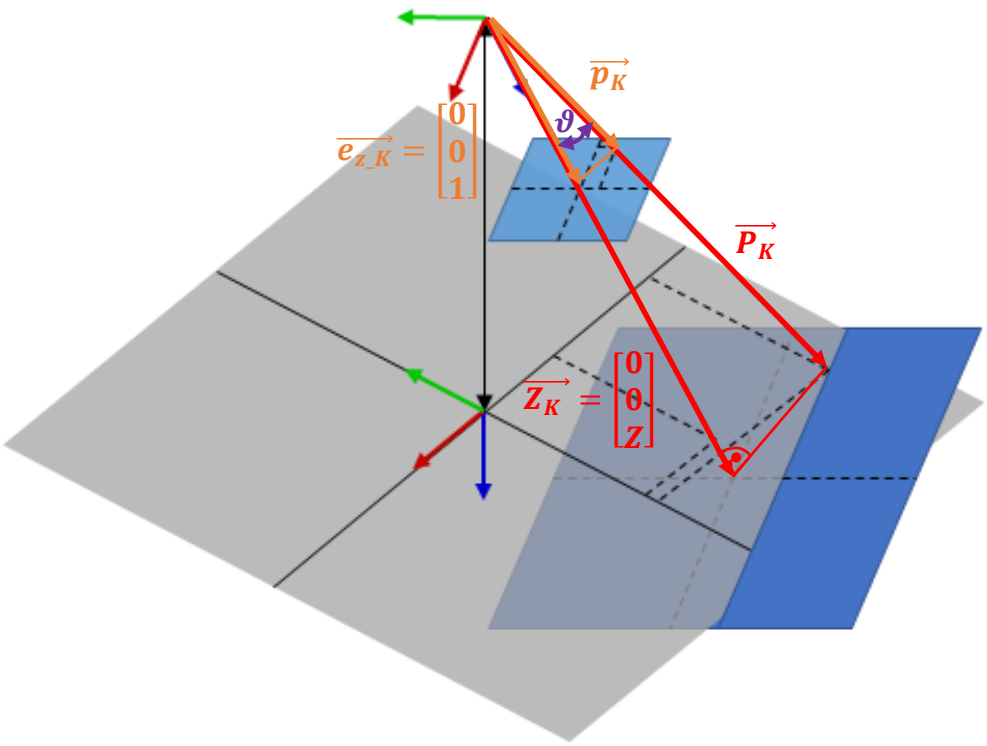


Abbildung 34: Geometrischer Zusammenhang in Bezug auf den Abstand Z_K

Durch den trigonometrischen Zusammenhang $\cos(\vartheta) = \frac{AK}{HY}$ ergibt sich:

$$\cos(\vartheta) = \frac{AK}{HY} = \frac{|\overrightarrow{Z_K}|}{|\overrightarrow{P_K}|} = \frac{\left\|\begin{bmatrix}0\\0\\Z\end{bmatrix}\right\|}{|\overrightarrow{p_K}| \cdot n} = \frac{Z}{|\overrightarrow{p_K}| \cdot n} \tag{4.18}$$

Mit Hilfe des Skalarproduktes aus dem Einheitsvektor $\overrightarrow{e_{z\_K}}$ und dem bekannten Vektor $\overrightarrow{p_K}$ folgt:

$$\cos(\vartheta) = \frac{\overrightarrow{e_{z\_K}} \circ \overrightarrow{p_K}}{|\overrightarrow{e_{z\_K}}| \cdot |\overrightarrow{p_K}|} = \frac{[0 \quad 0 \quad 1] \cdot \begin{bmatrix}x_n\\y_n\\1\end{bmatrix}}{|\overrightarrow{p_K}|} = \frac{1}{|\overrightarrow{p_K}|} \tag{4.19}$$

Durch das Gleichsetzen der Gleichungen (4.18) und (4.19) berechnet sich:

$$\Leftrightarrow \frac{1}{|\overrightarrow{p_K}|} = \frac{Z}{|\overrightarrow{p_K}| \cdot n}$$

$$\Leftrightarrow n = Z \qquad (4.20)$$

Setzt man die Gleichung (4.20) in die Gleichung (4.17) ein, erhält man den in Gleichung (4.21) beschriebenen Zusammenhang. Dieser ermöglicht die Berechnung der Z-Koordinate des betrachteten Raumpunktes in Abhängigkeit des dazugehörigen Pixels auf der Bildebene, der Höhe des Quadrokopters über dem Boden sowie der Lage des Kamera-Koordinatensystems gegenüber des Inertial-Koordinatensystems.

$$\frac{h_0}{Z} = (a_{13} \cdot x_n + a_{23} \cdot y_n + a_{33})$$

$$\Leftrightarrow Z = \frac{h_0}{(a_{13} \cdot x_n + a_{23} \cdot y_n + a_{33})} \qquad (4.21)$$

4.2.4 Linearisierung der Drehwinkel

Wie bereits in Abschnitt 4.2.2 dargestellt, können mit Hilfe einer Transformationsmatrix Vektoren zwischen verschiedenen Koordinatensystemen transformiert werden. Diese Transformationsmatrix enthält dabei die Verdrehung des Start-Koordinatensystems gegenüber dem Ziel-Koordinatensystem. Diese kann durch drei Elementardrehungen beschrieben werden. Die eindeutige Definition der Drehreihenfolge wird durch eine kinematische Kette definiert. Je nach Drehreihenfolge ergibt sich eine unterschiedliche Transformationsmatrix. Eine Transformationsmatrix ist dabei folgendermaßen aufgebaut.

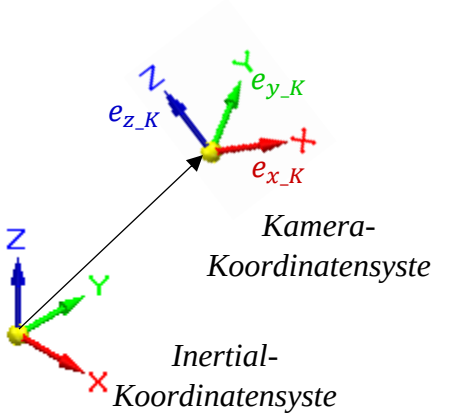


Abbildung 35: Einheitsvektoren des Kamera-Koordinatensystems im Inertial-Koordinatensystem

$$T_K^I = \left[\underbrace{\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix}}_{e_{x\_K}} \quad \underbrace{\begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{bmatrix}}_{e_{y\_K}} \quad \underbrace{\begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}}_{e_{z\_K}} \right] \qquad (4.22)$$

e_x, e_y und e_z beschreiben die Einheitsvektoren des Ziel-Koordinatensystems im Start-Koordinatensystem[7]. Deren Betrag ist immer 1.

Nachfolgend soll mit Hilfe der Methode der kleinsten Quadrate[8] die Lage des Quadrokopters geschätzt werden. Zur Vereinfachung der Schätzung der einzelnen Drehwinkel wird von nur sich geringfügig ändernden Drehwinkeln ausgegangen. Dies ermöglicht eine Linearisierung der Transformationsmatrix.

Es wird in dieser Arbeit die x-y-z Drehreihenfolge der Elementardrehungen betrachtet. Der Winkel α beschreibt die Drehung um die X-Achse, β die Drehung um die Y-Achse und γ die Drehung um die Z-Achse. T_K^I ist folgendermaßen definiert.

$$T_K^I = \begin{bmatrix} \cos(\beta)\cos(\gamma) & \cos(\gamma)\sin(\alpha)\sin(\beta) + \sin(\gamma)\cos(\alpha) & -\cos(\gamma)\sin(\beta)\cos(\alpha) + \sin(\gamma)\sin(\alpha) \\ -\cos(\beta)\sin(\gamma) & -\sin(\gamma)\sin(\alpha)\sin(\beta) + \cos(\gamma)\cos(\alpha) & \sin(\gamma)\sin(\beta)\cos(\alpha) + \cos(\gamma)\sin(\alpha) \\ \sin(\beta) & -\cos(\beta)\sin(\alpha) & \cos(\alpha)\cos(\beta) \end{bmatrix} \quad (4.23)$$

Zur Linearisierung können die Sinus- und Cosinus-Funktionen näherungsweise mit einer Taylorentwicklung ersten Grades angenähert werden. Hierzu wird $sin(\varphi) \approx \varphi$ und $cos(\varphi) \approx 1$ approximiert. Glieder höherer Ordnung, wie zum Beispiel $\alpha \cdot \beta$ oder $\alpha \cdot \beta \cdot \beta$, werden vernachlässigt. So ergibt sich eine Transformationsmatrix der Form:

$$T_K^I = \begin{bmatrix} 1 & \gamma & -\beta \\ -\gamma & 1 & \alpha \\ \beta & -\alpha & 1 \end{bmatrix} \quad (4.24)$$

Mit Gleichung (4.11) ergeben sich für a_{13}, a_{23} und a_{33} folgende Zusammenhänge.

$$a_{13} = -\beta \qquad a_{23} = \alpha \qquad a_{33} = 1 \quad (4.25)$$

Setzt man diese in die Gleichung (4.21) ein, ergibt sich für Z:

$$Z = \frac{h_0}{(-\beta \cdot x_n + \alpha \cdot y_n + 1)} \quad (4.26)$$

[7] Siehe Abbildung 35

[8] Siehe Vorlesung „Experimental Modelling and Simulation“ von Herrn Prof. Dr. Peter Zentgraf an der Technischen Hochschule Rosenheim

Auffällig ist in Gleichung (4.26), dass die Rotation des Kamera-Koordinatensystems gegenüber dem Inertial-Koordinatensystem um die Z-Achse γ nicht in die Berechnung der Z-Position des betrachteten Punktes P eingeht.

Aktuell ist die Gleichung (4.26) noch von den normierten Pixelpositionen im Kamera-Koordinatensystem x_n und y_n abhängig. Mit Hilfe der Horn-Schunck Methode bzw. der Lucas-Kanade Methode werden die einzelnen Pixelgeschwindigkeiten $[\dot{x}_d \quad \dot{y}_d]^T$ in der tatsächlichen Position der Bildebene berechnet. Um alle Geschwindigkeiten und Pixelpositionen in der gleichen Relation zueinander betrachten zu können, müssen die normierten Pixelpositionen x_n und y_n in die Pixelpositionen x_d und y_d mit Abstand f_x bzw. f_y zum Ursprung des Kamera-Koordinatensystem umgerechnet werden. Dies ist mit den Gleichungen (2.1) und (2.2) möglich. Dadurch erhält man folgende Zusammenhänge:

$$\frac{x_n}{1} = \frac{x_d}{f_x} \qquad \Leftrightarrow \qquad x_n = \frac{x_d}{f_x} \tag{4.27}$$

$$\frac{y_n}{1} = \frac{y_d}{f_y} \qquad \Leftrightarrow \qquad y_n = \frac{y_d}{f_y} \tag{4.28}$$

Setzt man diese in Gleichung (4.26) ein, folgt:

$$Z = \frac{h_0}{\left(-\beta \cdot \frac{x_d}{f_x} + \alpha \cdot \frac{y_d}{f_y} + 1\right)} \tag{4.29}$$

4.3 Schätzung der Kamerageschwindigkeiten sowie der Kameralage

Die in Kapitel 4.2.4 hergeleitete Gleichung (4.29) wird zur algebraischen Berechnung des Geschwindigkeitsvektorfeldes in die in Abschnitt 4.1 beschriebene Gleichung (4.9) eingesetzt. Die Geschwindigkeit eines Pixels $[\dot{x}_d \quad \dot{y}_d]^T$ im Kamera-Koordinatensystem wird folgendermaßen berechnet.

$$\begin{bmatrix}\dot{x}_d\\ \dot{y}_d\end{bmatrix} = \begin{bmatrix} -\frac{f_x \cdot \left(-\beta \cdot \frac{x_d}{f_x} + \alpha \cdot \frac{y_d}{f_y} + 1\right)}{h_0} \cdot v_x + \frac{\left(-\beta \cdot \frac{x_d}{f_x} + \alpha \cdot \frac{y_d}{f_y} + 1\right) \cdot x_d}{h_0} \cdot v_z + \\ -\frac{f_y \cdot \left(-\beta \cdot \frac{x_d}{f_x} + \alpha \cdot \frac{y_d}{f_y} + 1\right)}{h_0} \cdot v_y + \frac{\left(-\beta \cdot \frac{x_d}{f_x} + \alpha \cdot \frac{y_d}{f_y} + 1\right) \cdot y_d}{h_0} \cdot v_z + \\ \frac{x_d \cdot y_d}{f_y} \cdot \omega_x - \left(f_x + \frac{{x_d}^2}{f_x}\right) \cdot \omega_y + \frac{f_x \cdot y_d}{f_y} \cdot \omega_z \\ \left(f_y + \frac{{y_d}^2}{f_y}\right) \cdot \omega_x - \frac{x_d \cdot y_d}{f_x} \cdot \omega_y - \frac{f_y \cdot x_d}{f_x} \cdot \omega_z \end{bmatrix} \tag{4.30}$$

Die Pixelgeschwindigkeit jedes Pixels ist durch die Berechnung des optischen Flusses $[\dot{x}_d \quad \dot{y}_d]^T$ bekannt. Darüber hinaus sind durch die geometrischen Kamera-Kalibrierung die intrinsischen Parameter f_x, f_y, c_x und c_y bestimmt worden. Dies ermöglicht die Berechnung der Position jedes Pixels im Kamera-Koordinatensystem $[x_d \quad y_d]^T$ nach Gleichung (4.31).

$$x_d = u - c_x$$
$$y_d = v - c_y \tag{4.31}$$

Die aktuelle Höhe h_0 des Quadrokopters gegenüber dem ebenen Boden kann beispielsweise durch den sich verändernden Luftdruck berechnet werden. Somit sind die unbekannten Parameter die translatorische $[v_x \quad v_y \quad v_z]^T$ und rotatorische $[\omega_x \quad \omega_y \quad \omega_z]^T$ Geschwindigkeit sowie die Lage $[\alpha \quad \beta]^T$ der Kamera.

Für die Schätzung dieser unbekannten Größen wird nachfolgend die Methode der kleinsten Quadrate bzw. dessen rekursive Form, der Kalman-Filter, verwendet. Hierzu muss die Gleichung (4.30) in die Form $\bar{y} = \bar{\bar{H}} \cdot \bar{x}$ umgestellt werden. Die Methode der kleinsten Quadrate ist ein Verfahren, welches zur Schätzung von Systemzuständen sowie zur Parameteridentifikation

verwendet wird. Der x-Vektor des oben beschriebenen Zusammenhangs wird so berechnet, dass die Funktion $\tilde{y} = \bar{\bar{H}} \cdot \hat{x} + e$ den kleinstmöglichen Fehler e aufweist[9].

Um mit diesem Verfahren die acht unbekannten Größen bestimmen zu können, darf der Vektor $\bar{y}$ sowie die Matrix $\bar{\bar{H}}$ sich ausschließlich aus bekannten Größen zusammensetzen. Für eine eindeutige Lösungen für $[v_x \quad v_y \quad v_z \quad \omega_x \quad \omega_y \quad \omega_z \quad \alpha \quad \beta]^T$ muss der Vektor $\bar{x}$ eine Größe von $[8 \times 1]$ aufweisen. Dies führt zu acht Gleichungen für acht unbekannte Parameter. Darüber hinaus müssen die einzelnen Spalten der H-Matrix linear unabhängig voneinander sein.

Durch das Umstellen der Gleichung (4.30) nach den eben Beschriebenen Kriterien, erhält man:

$$\begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix} = \begin{bmatrix} \frac{x_d}{h_0} & \frac{f_x \cdot y_d}{f_y} & -f_x & 0 & 0 & 0 & -\frac{{x_d}^2}{f_x} & \frac{x_d \cdot y_d}{f_y} \\ 0 & 0 & 0 & -\frac{f_y \cdot x_d}{f_x} & \frac{y_d}{h_0} & f_y & -\frac{x_d \cdot y_d}{f_x} & \frac{{y_d}^2}{f_y} \end{bmatrix} \cdot \begin{bmatrix} (v_z + \beta \cdot v_x) \\ \left(-\frac{\alpha \cdot v_x}{h_0} + \omega_z\right) \\ \left(\frac{v_x}{h_0} + \omega_y\right) \\ \left(-\frac{\beta \cdot v_y}{h_0} + \omega_z\right) \\ (v_z - \alpha \cdot v_y) \\ \left(-\frac{v_y}{h_0} + \omega_x\right) \\ \left(\frac{\beta \cdot v_z}{h_0} + \omega_y\right) \\ \left(\frac{\alpha \cdot v_z}{h_0} + \omega_x\right) \end{bmatrix} \tag{4.32}$$

$$\underbrace{\quad}_{\bar{y}} = \underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\bar{\bar{H}}} \cdot \underbrace{\qquad\qquad}_{\bar{x}}$$

Die einzelnen Rechenschritte sind im Anhang A abgebildet.

Nach der Schätzung des Vektors $\hat{x} = [x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_7 \quad x_8]^T$ durch die Methode der kleinsten Quadrate bzw. dem Kalman-Filter kann dieser nach den einzelnen unbekannten Größen $[v_x \quad v_y \quad v_z \quad \omega_x \quad \omega_y \quad \omega_z \quad \alpha \quad \beta]^T$ entsprechend der Gleichung (4.33) berechnet werden.

[9] Siehe Vorlesung „Experimental Modelling and Simulation“ von Herrn Prof. Dr. Peter Zentgraf an der Technischen Hochschule Rosenheim

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix}}_{\hat{x}} = \underbrace{\begin{bmatrix} (v_z + \beta \cdot v_x) \\ \left(-\frac{\alpha \cdot v_x}{h_0} + \omega_z\right) \\ \left(\frac{v_x}{h_0} + \omega_y\right) \\ \left(-\frac{\beta \cdot v_y}{h_0} + \omega_z\right) \\ (v_z - \alpha \cdot v_y) \\ \left(-\frac{v_y}{h_0} + \omega_x\right) \\ \left(\frac{\beta \cdot v_z}{h_0} + \omega_y\right) \\ \left(\frac{\alpha \cdot v_z}{h_0} + \omega_x\right) \end{bmatrix}}_{\tilde{x}} \qquad (4.33)$$

Löst man die Gleichung (4.33) im ersten Schritt nach α, so erhält man ein Polynom 6. Grades der Form:

$$0 = a_6 \cdot \alpha^6 + a_5 \cdot \alpha^5 + a_4 \cdot \alpha^4 + a_3 \cdot \alpha^3 + a_2 \cdot \alpha^2 + a_1 \cdot \alpha + a_0 \qquad (4.34)$$

Die Koeffizienten $a_0 \cdots a_6$ der Gleichung (4.34) sind in Anhang C aufgelistet. Die Berechnung des Polynoms ist in Anhang B dargestellt.

Mit den bekannten Koeffizienten können alle sechs Nullstellen „$alfa\_null$“ des Polynoms berechnet werden. Die Nullstellen sind dabei die Eigenwerte der nachfolgenden charakteristischen Matrix.

$$alfa\_null = eig\left(\begin{bmatrix} \frac{\alpha^5}{\alpha^6} & \frac{\alpha^4}{\alpha^6} & \frac{\alpha^3}{\alpha^6} & \frac{\alpha^2}{\alpha^6} & \frac{\alpha^1}{\alpha^6} & \frac{\alpha^0}{\alpha^6} \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}\right) \qquad (4.35)$$

Darauffolgend ist eine Filterung nach der richtigen Nullstelle für α aus den sechs berechneten Nullstellen des Polynoms nötig. Im ersten Schritt können alle komplexen Nullstellen ausgeschlossen werden. Aufgrund der Linearisierung der Transformationsmatrix T_K^I sind nur kleine Winkel für α, β und γ zulässig. So wird nachfolgend die Nullstelle für α gewählt, welche sich am nächsten an Null befindet. Diese wird als α_{est} bezeichnet.

Darauffolgend können die Größen v_x, v_y, v_z, ω_x, ω_y, ω_z und β eindeutig berechnet werden. Die entsprechenden Gleichungen sind im Anhang D zu finden.

Nachdem die rotatorische und die translatorische Geschwindigkeit sowie die Lage des Quadrokopters gegenüber dem Inertial-Koordinatensystem mit Hilfe des optischen Flusses und dem oben beschriebenen Verfahren geschätzt worden ist, kann dessen aktuelle Position Pos und Lage ermittelt werden. Hierzu ist eine Integration der Größen v_{x_est}, v_{y_est}, v_{z_est} und ω_{z_est} über die Zeit notwendig. Eine Überprüfung der Lage des Quadrokopters um die X und Y-Achse ist durch eine Integration der Größen ω_{x_est} und ω_{y_est} möglich.

$$\overrightarrow{Pos} = \int_0^t \begin{bmatrix} v_{x\_est} \\ v_{y\_est} \\ v_{z\_est} \end{bmatrix} dt + \overrightarrow{Pos_0} \tag{4.36}$$

$$\vec{\varphi} = \int_0^t \begin{bmatrix} \omega_{x\_est} \\ \omega_{y\_est} \\ \omega_{z\_est} \end{bmatrix} dt + \overrightarrow{\varphi_0} \tag{4.37}$$

4.4 Simulation der Geschwindigkeits- und Lagebestimmung

Um das oben erarbeitete Verfahren zur Positions- und Lagebestimmung mittels des optischen Flusses qualifizieren zu können, wurde ein Matlab-Skript programmiert. Mit Hilfe dieses Skriptes kann für vorgegebene Werte das dazugehörige ideale Geschwindigkeitsvektorfeld berechnet werden. Zur Qualifizierung der in 4.3 beschriebenen Methode zur Schätzung der Werte für v_x, v_y, v_z, ω_x, ω_y, ω_z, α und β, werden aus dem idealen Geschwindigkeitsvektorfeld die vorgegebenen, idealen Werte zurückgerechnet. Zusätzlich besteht die Möglichkeit Messrauschen im berechneten optischen Fluss sowie Differenzen in der Höhenmessung zu simulieren. Dabei wird von einer Standardnormalverteilung der Rauschsignale ausgegangen. In der nachfolgenden Abbildung ist der grundlegende Aufbau des verwendeten Matlab-Skriptes „*testAlgorithmus.m*“ dargestellt.

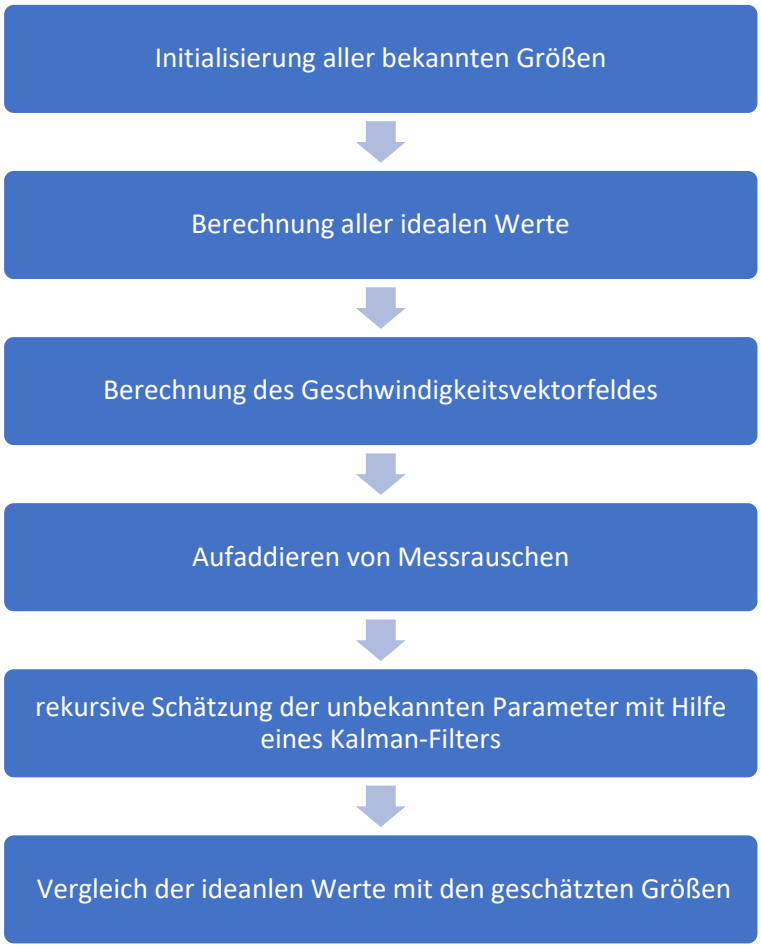


Abbildung 36: Aufbau des Matlab-Skriptes

4.4.1 Qualifizierung unter idealen Bedingungen

Im ersten Schritt wird ausschließlich die Genauigkeit des Verfahrens unter idealen Bedingungen untersucht. Hierbei werden aus einem idealen Geschwindigkeitsvektorfeld die zuvor vorgegebenen Größen zurückgerechnet. Es werden folgende Werte für v_x, v_y, v_z, ω_x, ω_y, ω_z, α, β und h_0 als ideale Werte definiert.

Tabelle 3: Werte für vX, vY, vZ, omegaX, omegaY, omegaZ, alfa, beta und h0

Variable	Wert
$\boldsymbol{v_x}$	$1\ m/s$
$\boldsymbol{v_y}$	$1\ m/s$
$\boldsymbol{v_z}$	$1\ m/s$
$\boldsymbol{\omega_x}$	$0{,}1\ m/s^2$
$\boldsymbol{\omega_y}$	$0{,}1\ m/s^2$
$\boldsymbol{\omega_z}$	$0{,}1\ m/s^2$
$\boldsymbol{\alpha}$	10°
$\boldsymbol{\beta}$	10°
$\boldsymbol{h_0}$	$1{,}2\ m$

Zur rekursiven Schätzung des $\hat{x}$-Vektors mit Hilfe des Kalman-Filters wird als Anfangsbedingung für $\hat{x}$ der Nullvektor und für die Kalman-Update-Matrix P die Einheitsmatrix verwendet. Des Weiteren ist zu erwähnen, dass die Fehler, welche durch die Linearisierung der Transformationsmatrix T_K^I entstehen, nicht in der nachfolgenden Betrachtung berücksichtigt werden. Um die Linearisierungsfehler berücksichtigen zu können, muss in dem beiliegenden Matlab-Skript „*testAlgorithmus.m*“ die Variable „*linFehler*“ auf 1 gesetzt werden.

Die Ergebnisse sind in Abbildung 37 , Abbildung 38 und Abbildung 39 dargestellt.

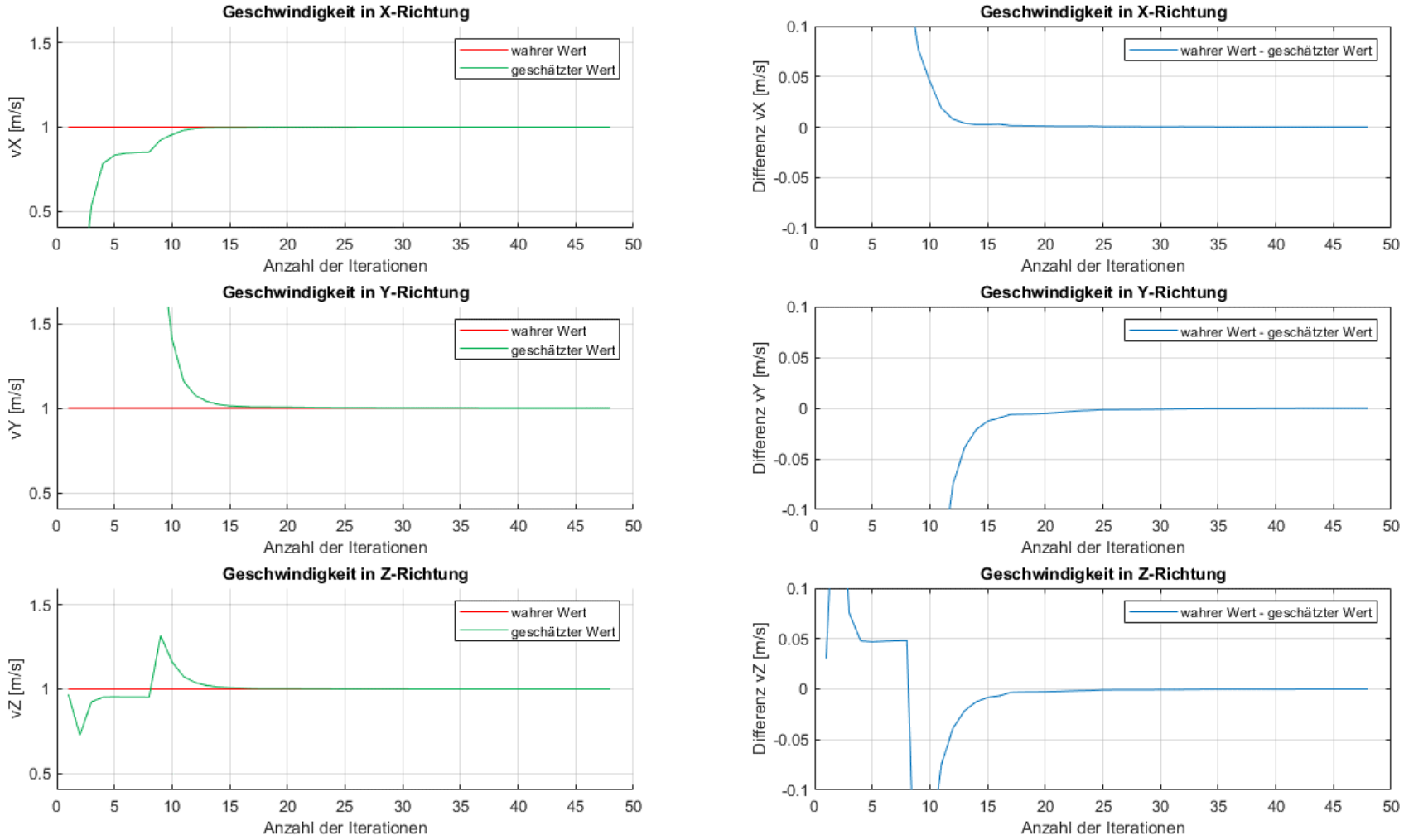


Abbildung 37: Berechnung von vX, vY und vZ unter idealen Bedingungen

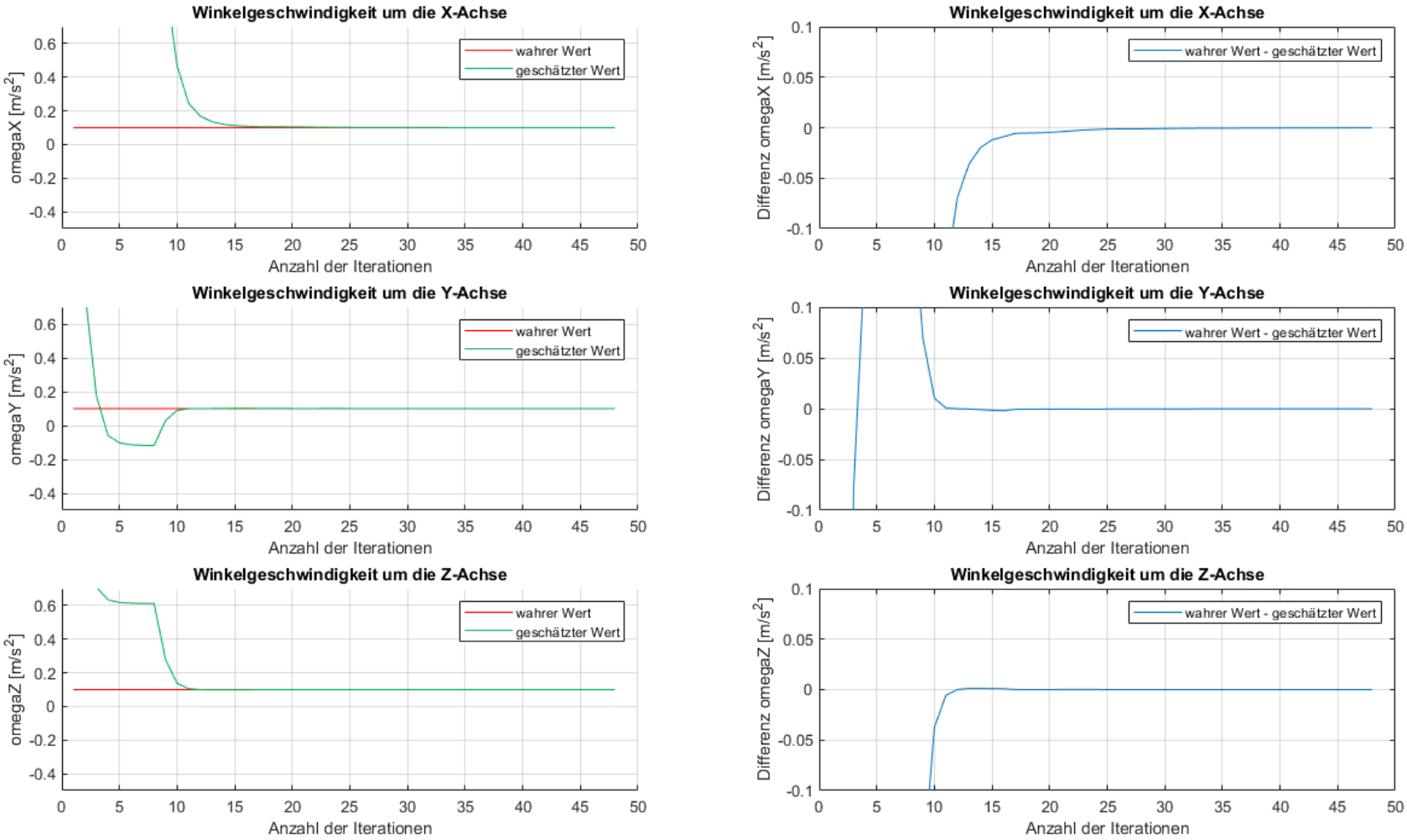
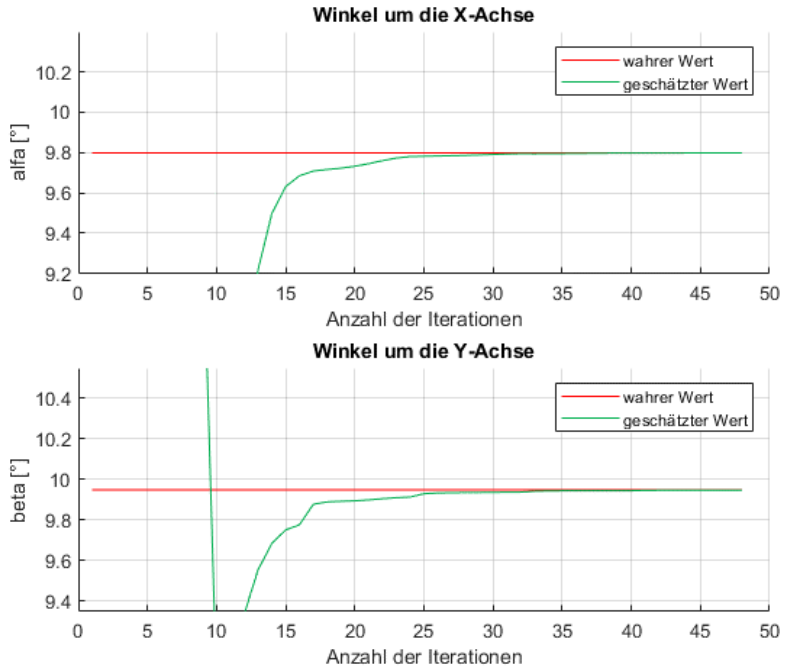


Abbildung 38: Berechnung von omegaX, omegaY und omegaZ unter idealen Bedingungen



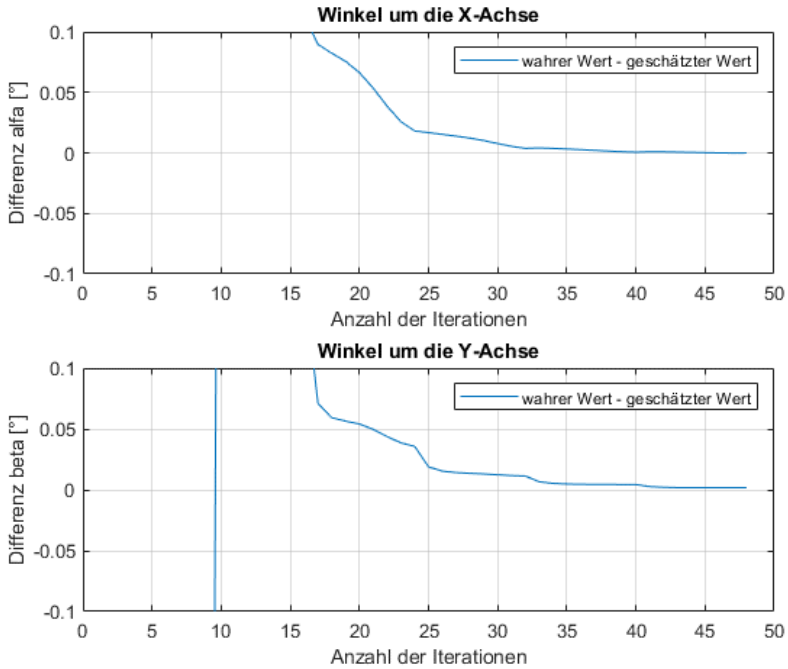


Abbildung 39: Berechnung von alfa und beta unter idealen Bedingungen

Zu beachten ist, dass sich aus einem betrachteten Pixel zwei Iterationen des Kalman-Filters ergeben. Die erste Iteration betrachtet die Pixelgeschwindigkeit in X-Richtung. Die zweite Iteration die Pixelgeschwindigkeit desselben Pixels in Y-Richtung.

In den oben dargestellten Abbildungen ist klar zu erkennen, dass auf die einzelnen Größen sehr gut zurückgerechnet werden kann. Nach 25 bis 30 Iterationen konvergieren alle Unbekannten mit einem geringen Fehler gegen den entsprechenden idealen Wert. Nach 48 Iterationen ist ein maximaler Fehler in allen acht Größen von $0{,}1141\%$ erreicht. Werden alle möglichen Iterationen berücksichtigt, so ist ein maximaler Fehler von $0{,}0002231\%$ zu beobachten.

Ebenso sind nur geringe maximale Fehler vom Idealwert des x-Vektors gegenüber dem geschätzten $\hat{x}$-Vektors zu erkennen. Der maximale Fehler bei 48 Iterationen liegt bei $0{,}0357\%$. Einen 0.0001060% Fehler für $\hat{x}$ erhält man bei der Berücksichtigung aller möglichen Iterationen.

Es ist klar zu erkennen, dass eine Erhöhung der Iterationen eine klare Verbesserung der Schätzung verursacht. Theoretisch sollten die idealen Werte für v_x, v_y, v_z, ω_x, ω_y, ω_z, α und β bereits nach acht Iterationen erreicht werden. Die hier erkennbaren Differenzen sind dabei auf eine nicht ideale Wahl der Anfangsbedingung für $\hat{x}$ sowie der Kalman-Update-Matrix P zurückzuführen.

4.4.2 Qualifizierung unter realistischen Bedingungen

Durch Störeinflüsse, Fertigungstoleranzen usw. addieren sich Rauschsignale und Offset-Fehler auf die wahren Messwerte jedes einzelnen Signals. Die Einflüsse dieser Ungenauigkeiten auf die Schätzung der translatorischen und rotatorischen Geschwindigkeit sowie der Lage der Kamera wird nachfolgend näher untersucht.

4.4.2.1 Einfluss von h0 auf das Ergebnis

Zunächst wird der Einfluss von ungenauen Messwerten der Höhe h_0 betrachtet. Hierzu wird ein ideales Geschwindigkeitsvektorfeld berechnet. Es werden die gleichen idealen Werte wie in Abschnitt 4.4.1 mit einer idealen Höhe von $h_0 = 1{,}20\,m$ definiert.

Nach der Berechnung des Geschwindigkeitsvektorfeldes wird die „gemessene“ Höhe der Kamera von $h_0 = 0{,}80\,m$ auf $h_0 = 1{,}60\,m$ mit einem Inkrement von $0{,}05m$ erhöht. Mit diesen falschen Abständen der Kamera zur Bodenoberfläche wird anschließend der dazugehörige Vektor $\hat{x}$ über 48 Iterationen geschätzt. Als Kenngröße der Genauigkeit werden die prozentualen Fehler von v_x, v_y, v_z, ω_x, ω_y, ω_z, α und β verwendet.

Da der translatorische Teil der Gleichung (4.9) nicht abhängig vom Abstand Z des betrachteten Punktes zum Ursprung des Kamera-Koordinatensystems ist, wird ausschließlich ein Einfluss des Höhenfehlers auf die rotatorischen Geschwindigkeit und somit der Lage der Kamera erwartet.

Die Ergebnisse sind in Abbildung 40, Abbildung 41 und Abbildung 42 dargestellt.

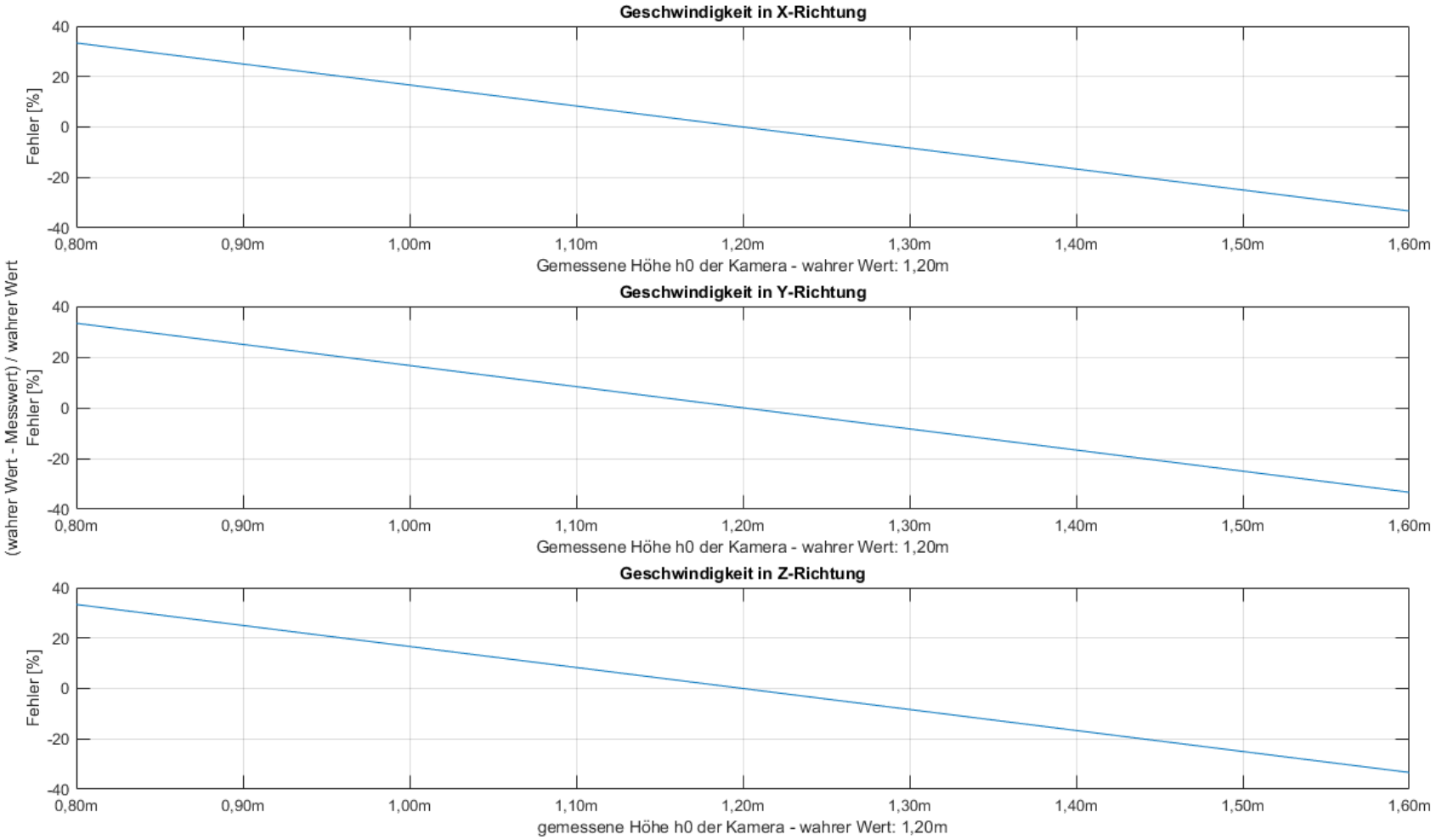


Abbildung 40: Einfluss des Höhenmessfehlers bei der Berechnung der translatorischen Geschwindigkeit

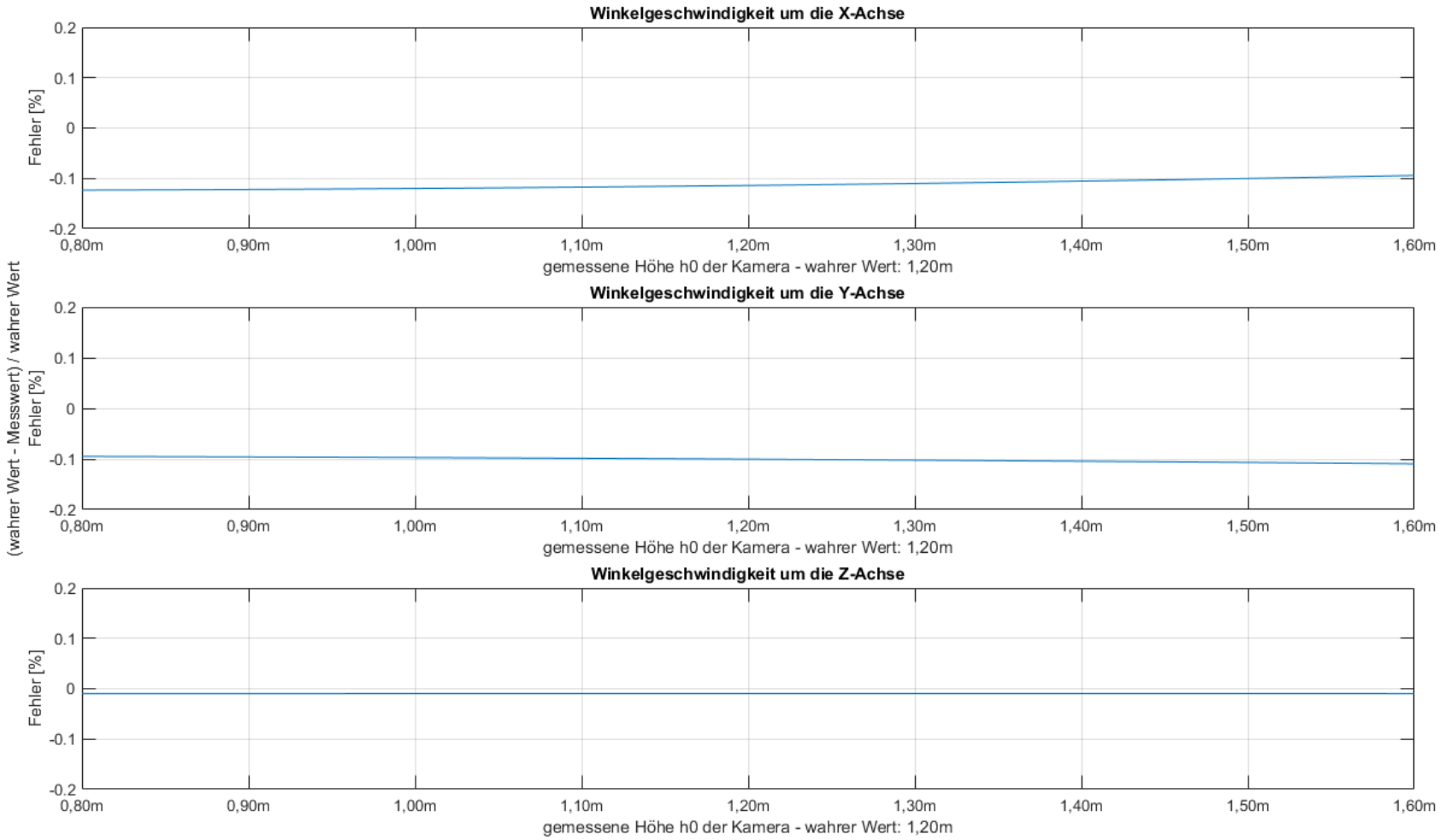


Abbildung 41: Einfluss des Höhenmessfehlers bei der Berechnung der rotatorischen Geschwindigkeit

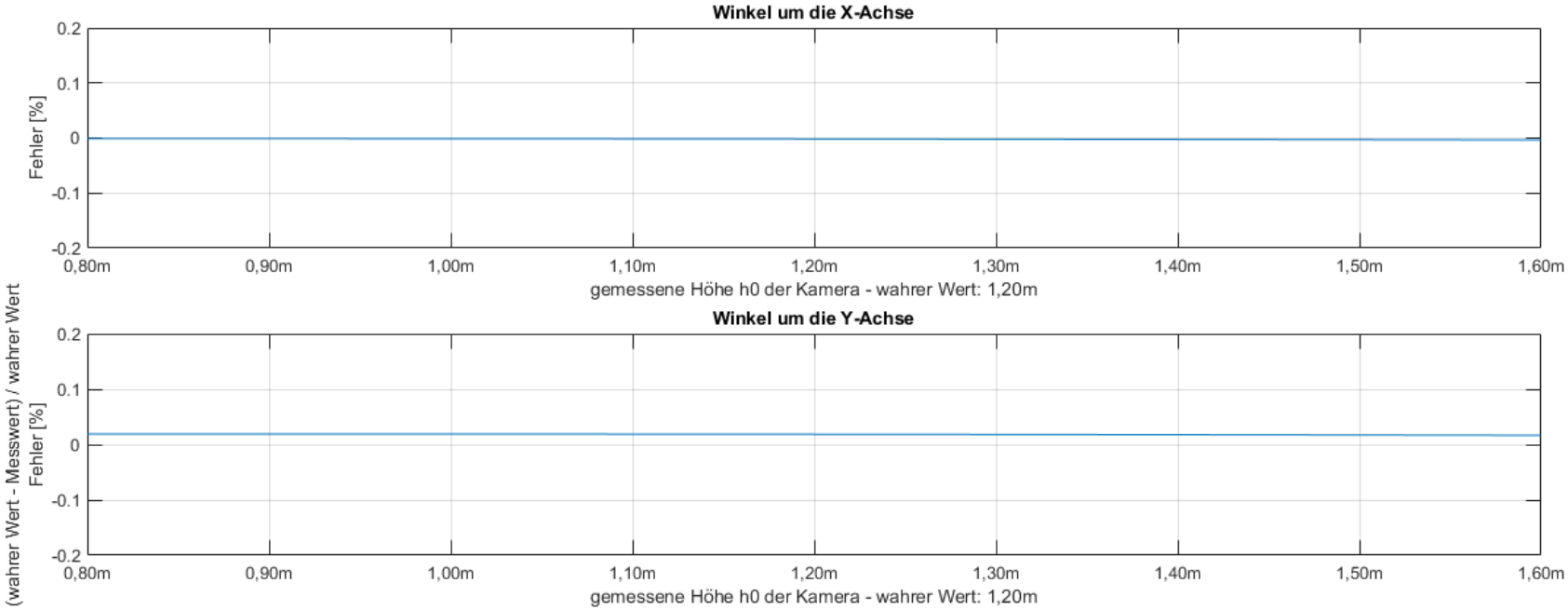


Abbildung 42: Einfluss des Höhenmessfehlers bei der Berechnung der Lage

In den oben dargestellten Abbildungen ist klar zu erkennen, dass die Höhendifferenz zwischen dem realen und dem idealen Wert einen eindeutig linearen Einfluss auf die Berechnung der translatorischen Geschwindigkeit in X-, Y- sowie Z-Richtung haben. In der Schätzung der rotatorischen Geschwindigkeit sowie der Lage des Quadrokopters sind nur geringfügige Einflüsse zu erkennen. Das oben vermutete Verhalten der Ergebnisse bestätigt sich hiermit.

4.4.2.2 Einfluss von Messrauschen im Geschwindigkeitsvektorfeld

Im letzten Schritt soll der Einfluss von Messrauschen im Geschwindigkeitsvektorfeld auf die Schätzung der translatorischen und rotatorischen Geschwindigkeiten sowie der Lage der Kamera näher untersucht werden.

Nach der Berechnung des idealen Geschwindigkeitsvektorfeldes mit Hilfe der in Abschnitt 4.4.1 dargestellten idealen Werte für v_x, v_y, v_z, ω_x, ω_y, ω_z, α, β und h_0 werden auf die einzelnen Pixelgeschwindigkeiten standardnormalverteilte Rauschwerte aufaddiert. Anschließend wird der unbekannte $\hat{x}$-Vektor über 48 Iterationen geschätzt. Um den Einfluss des Messrauschen auf die Berechnung der einzelnen Größen kompakt darstellen zu können, werden die prozentualen Abweichungen von v_x, v_y, v_z, ω_x, ω_y, ω_z, α und β gegenüber den idealen Werten berechnet. Diese Kennzahlen werden für Standardabweichungen von $\sigma = 0\,pixel/s$ bis $\sigma = 5\,pixel/s$ in den nachfolgenden Abbildungen dargestellt.

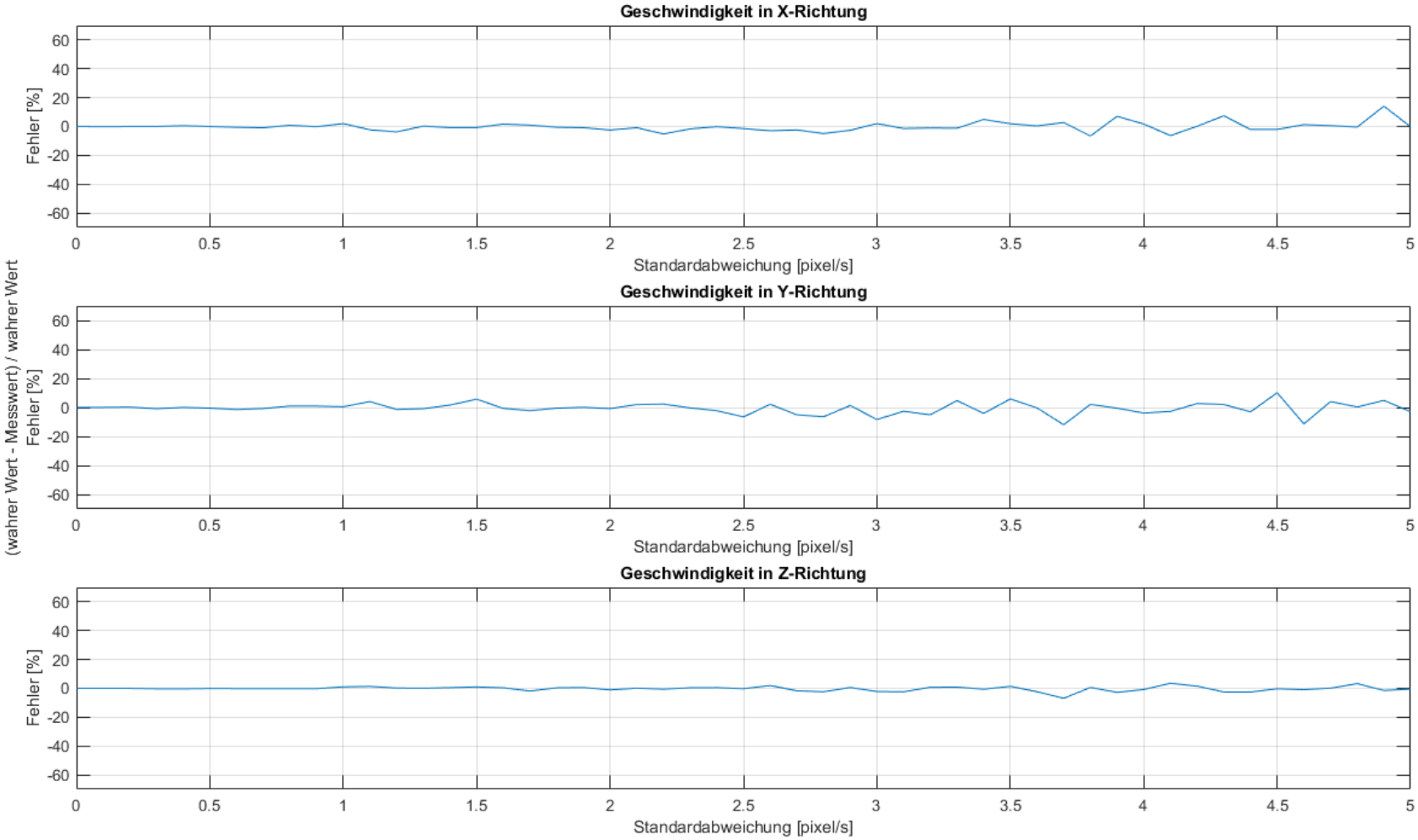


Abbildung 43: Einfluss von Messrauschen auf die Berechnung der translatorischen Geschwindigkeit

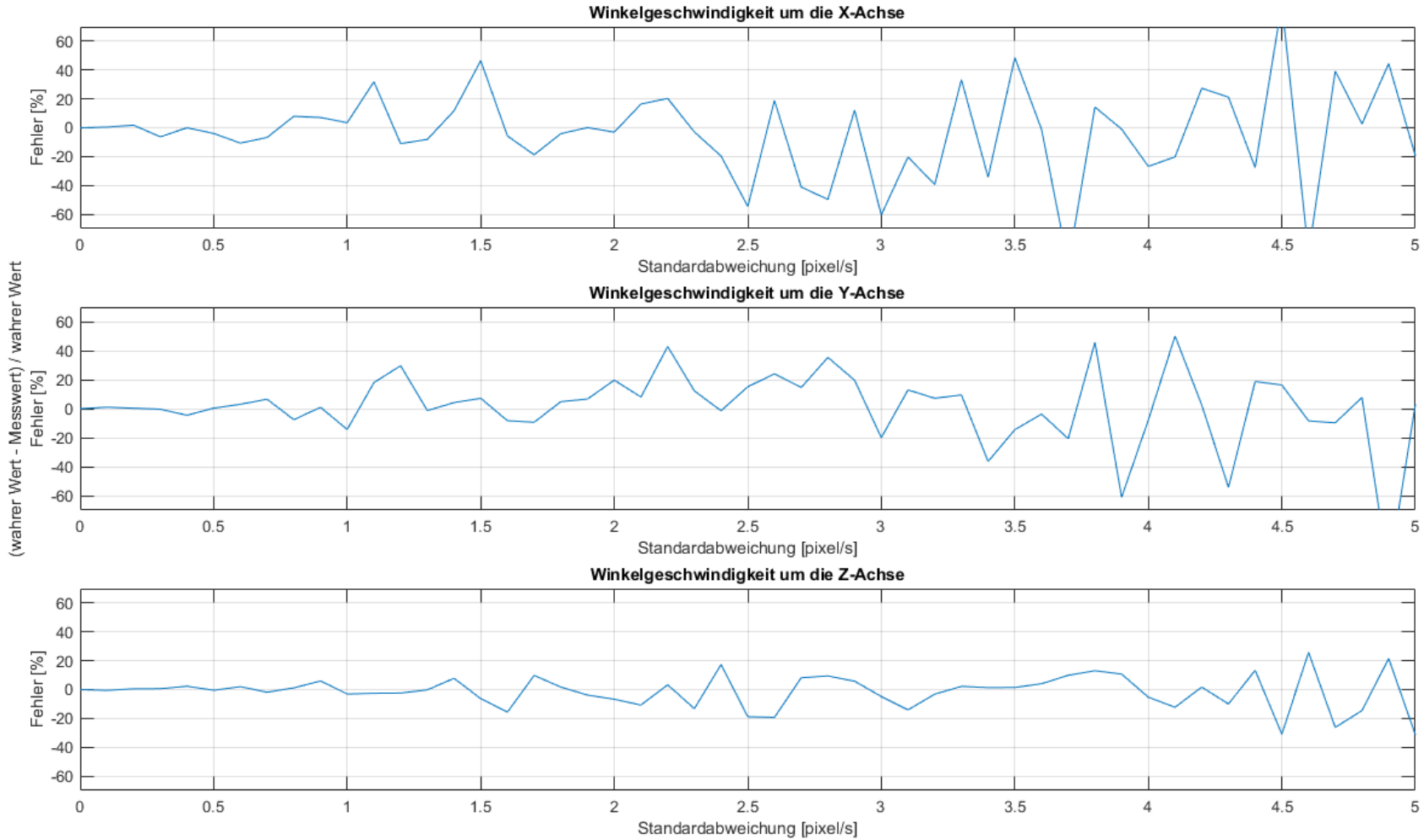


Abbildung 44: Einfluss von Messrauschen auf die Berechnung der rotatorische Geschwindigkeit

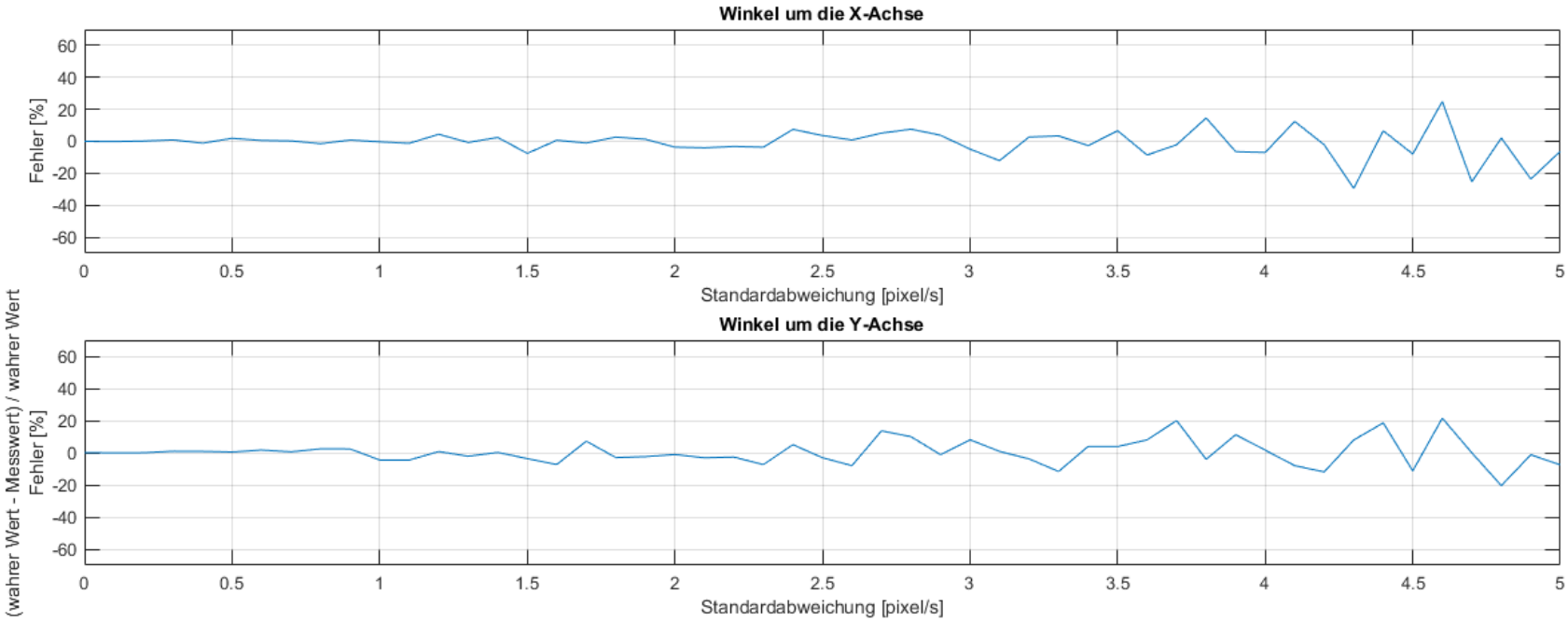


Abbildung 45: Einfluss von Messrauschen auf die Berechnung der Lage

In allen oben dargestellten Abbildungen ist eine klare Zunahme der Schätzfehler mit zunehmender Standardabweichung zu erkennen. Das Rauschen weist vor allem einen großen Einfluss auf Berechnung der Winkelgeschwindigkeiten ω_x und ω_y auf. Der geringste Einfluss ist in der Schätzung der translatorischen Geschwindigkeit der Kamera zu erkennen. Werden mehrere Iterationen für die Schätzung des $\hat{x}$-Vektors mit Hilfe des Kalman-Filters verwendet, so kann die Robustheit des Verfahrens deutlich erhöht werden. Jedoch ist dabei mit einer deutlichen Erhöhung der notwendigen Rechenkapazität zu rechnen.

Bevor das Verfahren in praktischen Einsatz verwendet werden kann, sollte das Rauschen der optischen Fluss-Daten in einer Messdatenstudie untersucht werden. Auf Basis dieser Erkenntnisse ist eine Abschätzung der idealen Anzahl von Schätziterationen besser möglich.

5 Zusammenfassung und Ausblick

Im Folgenden sollen die zentralen Ergebnisse dieser Arbeit zusammengefasst sowie ein Ausblick in die Zukunft gegeben werden.

5.1 Zusammenfassung der Ergebnisse

Ausgehend von der zentralen Problemstellung, eine funktionsfähige optische Positions- und Lagebestimmung eines Quadrokopters im Raum zu erarbeiten, zu entwerfen und zu testen, wurden im ersten Schritt die allgemeinen Grundlagen erarbeitet. Nach einem kurzen generellen Überblick über den Bereich der optischen Bildverarbeitung wurde auf die Lochkamera, als einfachstes Modell einer Kamera, hingearbeitet. Ausgehend von der Berechnungsgrundlage dieses Kameramodells wurde den Lesern im Abschnitt 2.3 die geometrische Kamerakalibrierung nähergebracht. In diesem Teil der Arbeit wurde auf die Notwendigkeit dieser Kalibrierung, deren generelle Funktionsweise sowie auf die einzelnen mathematischen Zusammenhänge und Größen eingegangen. Da die optische Positions- und Lagebestimmung auf Basis der einzelnen Pixelgeschwindigkeiten einer Bildfolge bestimmt werden soll, wurden im Kapitel 2.4 Bewegungsfelder sowie der optische Fluss näher erklärt. Aufbauend auf den dort dargestellten Grundlagen wurde nachfolgend auf zwei Methoden zur Berechnung des optischen Flusses eingegangen. Die Lucas-Kanade Methode als auch die Horn-Schunck Methode spiegeln dabei, die beiden meist verbreiteten Verfahren zur Berechnung eines Geschwindigkeitsvektorfeldes wider. Am Ende des 2. Abschnittes wurde kurz der, in dieser Arbeit verwendete Quadrokopter sowie dessen eingebaute Sensoren beschrieben.

Nach diesen allgemeinen Grundlagen wurde im Kapitel 3 eine von MathWorks Inc. bereits auf Basis des optischen Flusses implementierte Positions- und Lagebestimmung näher betrachtet. Die Genauigkeit der hier implementierten Positions- und Lagebestimmung wurde in einer Versuchsreihe untersucht. Hierbei konnten enorme Differenzen zwischen den berechneten und den realen Werten in der Position des Quadrokopters aufgezeigt werden.

Aus dem oben genannten Grund und der nicht transparenten Implementierung einiger Algorithmen wurde entschlossen in Kapitel 4 eine eigene Methode zur optischen Positions- und Lagebestimmung auf Basis des optischen Flusses zu erarbeiten. Hierzu wurden in Abschnitt 4.1 die algebraische Berechnung eines Geschwindigkeitsvektorfeldes hergeleitet. Dieser mathematische Zusammenhang wurde anschließend in Kapitel 4.2 erweitert, sodass die aktuelle Drehung des Quadrokopters um die X- und Y-Achse sowie dessen translatorische und rotatorische Geschwindigkeit berechnet werden können.

Ausgehend von den in Kapitel 4.1 und Kapitel 4.2 erarbeiteten Formeln wurden diese in Abschnitt 4.3 so angepasst, dass mit Hilfe der Methode der kleinsten Quadrate bzw. dem Kalman-Filter die translatorische und rotatorische Geschwindigkeit der Kamera sowie deren Lage um die X- und Y-Achse geschätzt werden konnte.

Hierzu wird die Höhe der Kamera über dem Boden als auch das, mit Hilfe dem optischen Fluss berechneten Geschwindigkeitsvektorfeld, als bekannte Größen benötigt.

Letztendlich wurde in Kapitel 4.4 das oben erarbeitete Verfahren auf dessen Genauigkeit untersucht. Ebenso wurde das Verhalten der unbekannten Parameter auf mögliche Messfehler aufgezeigt.

5.2 Ausblick

Mit dieser Arbeit wurde gezeigt, dass theoretisch die aktuelle Geschwindigkeit, Position sowie Lage eines Quadrokopters unabhängig von GPS-Daten bestimmt werden kann. Dies ermöglicht den Einsatz von Drohnen nicht nur im Outdoor-Bereich, sondern auch im geschlossenen Raum. Um hierzu eine klare Aussage treffen zu können sollte das hier erarbeitete Verfahren im nächsten Schritt auch im praktischen Einsatz getestet werden.

Die erarbeitete Methode kann individuell angewandt werden. Ebenso ist die Berechnung der translatorischen und rotatorischen Geschwindigkeit sowie der Lage eines Körpers unter der Verwendung einer beliebigen Methode zur Berechnung des optischen Flusses möglich. Das beste Verfahren zur Bestimmung des optischen Flusses sollte individuell für das jeweilige Einsatzgebiete in einer Versuchsreihe evaluiert werden.

Wie in Abschnitt 4.4 zu erkennen ist, wurden die Anfangsbedingungen des Kalman-Filters in den ersten Versuchsreihen nicht ideal gewählt. Dies führte zu einem hohen Rechenaufwand, welcher in Zukunft durch eine gezielte Wahl der Anfangsbedingungen signifikant reduziert werden kann. Diese könnten durch eine relativ ungenaue Schätzung mit Hilfe von vier ausgewählten Pixeln optimiert werden.

Ebenso kann das Einsatzgebiet der hier erarbeiteten optischen Positions- und Lagebestimmung enorm durch eine intelligente Auswahl, der für die Berechnung verwendeten Pixel, gesteigert werden. Durch den Einsatz von künstlicher Intelligenz könnten beispielsweise Stufen, Tische und sonstige Unregelmäßigkeiten mit unterschiedlichen Höhen im Bild erkannt und die dazugehörigen Pixel nicht bei der Berechnung berücksichtigt werden.

Auch ist der Einsatz von mehreren Kameras für eine robustere Schätzung der unbekannten Größen ist denkbar.

Literaturverzeichnis

[1] Bundesministerium für Wirtschaft und Energie: …mit Drohnen: Unbemanntes Fliegen im Dienst von Mensch, Natur und Gesellschaft. Berlin 2019.

[2] Eissfeller, B.; Teuber, A. und Zucker, P.: Indoor-GPS: Ist der Satellitenempfang in Gebäuden möglich?, zfv 130 (2005). S. 226–234.

[3] Kuypers, F.: Physik für Ingenieure und Naturwissenschaftler: Band 2: Elektrizität, Optik und Wellen. Weinheim: Wiley 2012.

[4] Erhardt, A.: Einführung in die Digitale Bildverarbeitung: Grundlagen, Systeme und Anwendungen. Wiesbaden: Vieweg+Teubner / GWV Fachverlage GmbH Wiesbaden 2008.

[5] Burger, W. und Burge, M. J.: Digitale Bildverarbeitung: Eine algorithmische Einführung mit Java. 3. Auflage. Berlin: Springer Vieweg 2015.

[6] Bouguet, J.-Y.: Camera Calibration Toolbox for Matlab. http://www.vision.caltech.edu/bouguetj/index.html, Zugriff am: 08.02.2021.

[7] The MathWorks Inc.: Single and Stereo Camera Calibration. https://www.mathworks.com/help/vision/single-and-stereo-camera-calibration.html?s_tid=CRUX_lftnav, Zugriff am: 08.02.2021.

[8] -,-: Camera Calibration. https://docs.opencv.org/master/dc/dbb/tutorial_py_calibration.html, Zugriff am: 08.02.2021.

[9] Wu, Y.: Optical Flow and Motion Analysis. Evanston 2007.

[10] Alberts, J.: Bildverarbeitung für das Projekt FAUST: Focus of Expansion / optischer Fluss 2007.

[11] Hiebl, M.: Realtime Optical Flow 2011.

[12] Baker, S.; Scharstein, D.; Lewis, J. P.; Roth, S.; Black, M. J. und Szeliski, R.: A Database and Evaluation Methodology for Optical Flow. 2010. Auflage 20 September 2010.

[13] The MathWorks Inc.: Optical Flow. https://de.mathworks.com/help/vision/ref/opticalflow.html?searchHighlight=Lucas-Kanade&s_tid=srchtitle, Zugriff am: 10.02.2021.

[14] Lucas, B. D. und Kanade, T.: An Iterative Image Registration Technique with an Application to Stereo Vision. Pennsylvania 1981.

[15] Rojas, R.: Lucas-Kanade in a Nutshell. Freie Universität Berlin.

[16] Barron, J. L.; Fleet, D. J. und Beauchemin, S. S.: Performance Of Optical Flow Techniques, International Journal of Computer Vision (1994) H. 12. S. 43–77.

[17] Horn, B. K. P. und Schunck, B. G.: Determining optical flow, Artifical Intelligence (1981) H. 17. S. 185–203.

[18] Seppke, B.: Herleitung der Euler-Lagrange Gleichungen für Optical Flow Constraints 2010.

[19] Bestmann, O.: Konzeption und Implementierung eines Verfahrens zur Messung von Verschiebungsvektoren in Multispektralbildern. Hamburg 2012.

[20] The MathWorks, I.: Simulink Support Package for Parrot Minidrones. https://de.mathworks.com/help/supportpkg/parrot/index.html?s_tid=CRUX_lftnav, Zugriff am: 20.11.2020.

[21] The MathWorks, I.: Simulink Support Package for Parrot: User's Guide. 2020. Auflage. Natick, MA 01760-2098 2020.

[22] Daniilidis, K. und Shi, J.: 3D Velocities from Optical Flow. www.coursera.org 2016.

Abbildungsverzeichnis

Abbildung 1: ... mit Drohnen fliegen – BMWi 1

Abbildung 2: Lichtstrahlen und Lichtbündel 3

Abbildung 3: Einfaches Linsensystem 3

Abbildung 4: Geometrie einer Lochkamera 4

Abbildung 5: Radiale Linsenverzerrung 6

Abbildung 6: Tangentiale Bildverzerrung 6

Abbildung 7: Koordinatentransformation zwischen Welt-, Kamera- und Pixelkoordinatensystem 7

Abbildung 8: Zusammenhang zwischen intrinsischen und extrinsischen Kameraparametern sowie dem Welt-, Kamera- und Pixelkoordinatensystem 8

Abbildung 9: Beispiel Kalibriermuster 9

Abbildung 10: optischer Fluss bei Fahrassistenzsystemen 10

Abbildung 11: Projektion einer Bewegung eines 3D-Punktes im Raum auf die 2D-Bildebene ... 11

Abbildung 12: Grundlegende Funktionsweise von differentiellen Verfahren zur Berechnung des optischen Flusses 12

Abbildung 13: Beispiel eines 3x3 Maske mit 9 Gleichungen 14

Abbildung 14: Vergleich der Lucas-Kanade Methode (a) mit der Horn-Schunck Methode (b, c, d) mit drei unterschiedlichen Iterationsstufen 18

Abbildung 15: Parrot Rolling Spieder 19

Abbildung 16: Übersicht des Aufbaus der Positions- und Lagebestimmung 21

Abbildung 17: Versuchsaufbau des Versuch 1 22

Abbildung 18: Bewegungsrichtungen des Quadrokopters 23

Abbildung 19: Bodenmarkierungen des 1. Versuchs 24

Abbildung 20: Ergebnisse der Position entlang der X-Achse 25

Abbildung 21: Ergebnisse der Position entlang der Y-Achse 26

Abbildung 22: Höhenabhängigkeit der Messdaten 27

Abbildung 23: Keine konstante Messdatenaufnahme 28

Abbildung 24: Keine eindeutige Einheit 29

Abbildung 25: Geschwindigkeitsvektorfeld einer translatorischen Bewegung einer Kamera 31

Abbildung 26: Geschwindigkeitsvektorfeld einer rotatorischen Bewegung einer Kamera 31

Abbildung 27: Geschwindigkeitsvektorfeld einer um die optische Achse rotierende Kamera 32

Abbildung 28: Geschwindigkeitsvektorfeld einer beliebigen Bewegung einer Kamera 32

Abbildung 29: Skizze zur algebraischen Berechnung des projizierten Geschwindigkeitsvektorfeldes eines dreidimensionalen Körpers..... 36

Abbildung 30: Skizze - Quadrokopter parallel zum Inertial-Koordinatensystem..... 38

Abbildung 31: Skizze - Quadrokopter nicht parallel zum Inertial-Koordinatensystem..... 39

Abbildung 32: Skizze der bekannten Vektoren..... 40

Abbildung 33: Geometrischer Zusammenhang in Bezug auf die Höhe HK 42

Abbildung 34: Geometrischer Zusammenhang in Bezug auf den Abstand ZK 44

Abbildung 35: Einheitsvektoren des Kamera-Koordinatensystems im Inertial-Koordinatensystem 45

Abbildung 36: Aufbau des Matlab-Skriptes..... 52

Abbildung 37: Berechnung von vX, vY und vZ unter idealen Bedingungen 54

Abbildung 38: Berechnung von omegaX, omegaY und omegaZ unter idealen Bedingungen 55

Abbildung 39: Berechnung von alfa und beta unter idealen Bedingungen..... 56

Abbildung 40: Einfluss des Höhenmessfehlers bei der Berechnung der translatorischen Geschwindigkeit 58

Abbildung 41: Einfluss des Höhenmessfehlers bei der Berechnung der rotatorischen Geschwindigkeit 59

Abbildung 42: Einfluss des Höhenmessfehlers bei der Berechnung der Lage..... 60

Abbildung 43: Einfluss von Messrauschen auf die Berechnung der translatorischen Geschwindigkeit 62

Abbildung 44: Einfluss von Messrauschen auf die Berechnung der rotatorische Geschwindigkeit 63

Abbildung 45: Einfluss von Messrauschen auf die Berechnung der Lage..... 64

Tabellenverzeichnis

Tabelle 1: Sensoren der Minidrone RollingSpider 19
Tabelle 2: Versuchsdurchführung 23
Tabelle 3: Werte für vX, vY, vZ, omegaX, omegaY, omegaZ, alfa, beta und h0 53

Formelverzeichnis

(2.1) 5
(2.2) 5
(2.3) 7
(2.4) 7
(2.5) 12
(2.6) 13
(2.7) 13
(2.8) 13
(2.9) 13
(2.10) 14
(2.11) 15
(2.12) 15
(2.13) 15
(2.14) 15
(2.15) 15
(2.16) 16
(2.17) 16
(2.18) 17
(2.19) 17
(2.20) 17
(4.1) 32
(4.2) 33
(4.3) 33
(4.4) 33
(4.5) 33
(4.6) 34
(4.7) 34
(4.8) 35
(4.9) 35

(4.10) 40
(4.11) 41
(4.12) 41
(4.13) 42
(4.14) 42
(4.15) 43
(4.16) 43
(4.17) 43
(4.18) 44
(4.19) 44
(4.20) 45
(4.21) 45
(4.22) 45
(4.23) 46
(4.24) 46
(4.25) 46
(4.26) 46
(4.27) 47
(4.28) 47
(4.29) 47
(4.30) 48
(4.31) 48
(4.32) 49
(4.33) 50
(4.34) 50
(4.35) 50
(4.36) 51
(4.37) 51

Anhang

Anhang A: Umformung der Gleichung (4.30)

$$\begin{bmatrix}\dot{x}_d\\ \dot{y}_d\end{bmatrix} = \begin{bmatrix} -\frac{f_x\cdot\left(-\beta\cdot\frac{x_d}{f_x}+\alpha\cdot\frac{y_d}{f_y}+1\right)}{h_0}\cdot v_x + \frac{\left(-\beta\cdot\frac{x_d}{f_x}+\alpha\cdot\frac{y_d}{f_y}+1\right)\cdot x_d}{h_0}\cdot v_z + \frac{x_d\cdot y_d}{f_y}\cdot\omega_x - \left(f_x+\frac{{x_d}^2}{f_x}\right)\cdot\omega_y + \frac{f_x\cdot y_d}{f_y}\cdot\omega_z \\ -\frac{f_y\cdot\left(-\beta\cdot\frac{x_d}{f_x}+\alpha\cdot\frac{y_d}{f_y}+1\right)}{h_0}\cdot v_y + \frac{\left(-\beta\cdot\frac{x_d}{f_x}+\alpha\cdot\frac{y_d}{f_y}+1\right)\cdot y_d}{h_0}\cdot v_z + \left(f_y+\frac{{y_d}^2}{f_y}\right)\cdot\omega_x - \frac{x_d\cdot y_d}{f_x}\cdot\omega_y - \frac{f_y\cdot x_d}{f_x}\cdot\omega_z \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix}\dot{x}_d\\ \dot{y}_d\end{bmatrix} = \begin{bmatrix} \frac{\cancel{f_x}\cdot x_d}{\cancel{f_x}\cdot h_0}\cdot\beta\cdot v_x - \frac{f_x\cdot y_d}{f_y\cdot h_0}\cdot\alpha\cdot v_x - \frac{f_x}{h_0}\cdot v_x - \frac{x_d\cdot x_d}{f_x\cdot h_0}\cdot\beta\cdot v_z + \frac{x_d\cdot y_d}{f_y\cdot h_0}\cdot\alpha\cdot v_z + \frac{x_d}{h_0}\cdot v_z + \frac{x_d\cdot y_d}{f_y}\cdot\omega_x - f_x\cdot\omega_y - \frac{{x_d}^2}{f_x}\cdot\omega_y + \frac{f_x\cdot y_d}{f_y}\cdot\omega_z \\ \frac{f_y\cdot x_d}{f_x\cdot h_0}\cdot\beta\cdot v_y - \frac{\cancel{f_y}\cdot y_d}{\cancel{f_y}\cdot h_0}\cdot\alpha\cdot v_y - \frac{f_y}{h_0}\cdot v_y - \frac{y_d\cdot x_d}{f_x\cdot h_0}\cdot\beta\cdot v_z + \frac{y_d\cdot y_d}{f_y\cdot h_0}\cdot\alpha\cdot v_z + \frac{y_d}{h_0}\cdot v_z + f_y\cdot\omega_x + \frac{{y_d}^2}{f_y}\cdot\omega_x - \frac{x_d\cdot y_d}{f_x}\cdot\omega_y - \frac{f_y\cdot x_d}{f_x}\cdot\omega_z \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix}\dot{x}_d(t)\\ \dot{y}_d(t)\end{bmatrix} = \begin{bmatrix} \frac{x_d(t)}{h_0}\cdot(v_z+\beta\cdot v_x) + \frac{f_x\cdot y_d(t)}{f_y}\cdot\left(-\frac{\alpha\cdot v_x}{h_0}+\omega_z\right) - f_x\cdot\left(\frac{v_x}{h_0}+\omega_y\right) - \frac{{x_d}^2(t)}{f_x}\cdot\left(\frac{\beta\cdot v_z}{h_0}+\omega_y\right) + \frac{x_d(t)\cdot y_d(t)}{f_y}\cdot\left(\frac{\alpha\cdot v_z}{h_0}+\omega_x\right) \\ -\frac{f_y\cdot x_d(t)}{f_x}\cdot\left(-\frac{\beta\cdot v_y}{h_0}+\omega_z\right) + \frac{y_d(t)}{h_0}\cdot(v_z-\alpha\cdot v_y) + f_y\cdot\left(-\frac{v_y}{h_0}+\omega_x\right) - \frac{x_d(t)\cdot y_d(t)}{f_x}\cdot\left(\frac{\beta\cdot v_z}{h_0}+\omega_y\right) + \frac{{y_d}^2(t)}{f_y}\cdot\left(\frac{\alpha\cdot v_z}{h_0}+\omega_x\right) \end{bmatrix}$$

$$\begin{bmatrix}\dot{x}_d\\ \dot{y}_d\end{bmatrix} = \begin{bmatrix}\frac{x_d}{h_0} & \frac{f_x \cdot y_d}{f_y} & -f_x & 0 & 0 & 0 & -\frac{{x_d}^2}{f_x} & \frac{x_d \cdot y_d}{f_y}\\ 0 & 0 & 0 & -\frac{f_y \cdot x_d}{f_x} & \frac{y_d}{h_0} & f_y & -\frac{x_d \cdot y_d}{f_x} & \frac{{y_d}^2}{f_y}\end{bmatrix} \cdot \begin{bmatrix}(v_z + \beta \cdot v_x)\\ \left(-\frac{\alpha \cdot v_x}{h_0} + \omega_z\right)\\ \left(\frac{v_x}{h_0} + \omega_y\right)\\ \left(-\frac{\beta \cdot v_y}{h_0} + \omega_z\right)\\ (v_z - \alpha \cdot v_y)\\ \left(-\frac{v_y}{h_0} + \omega_x\right)\\ \left(\frac{\beta \cdot v_z}{h_0} + \omega_y\right)\\ \left(\frac{\alpha \cdot v_z}{h_0} + \omega_x\right)\end{bmatrix}$$

Anhang B: Die Berechnung von alfa

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} = \begin{bmatrix} (v_z + \beta \cdot v_x) \\ \left(-\frac{\alpha \cdot v_x}{h_0} + \omega_z\right) \\ \left(\frac{v_x}{h_0} + \omega_y\right) \\ \left(-\frac{\beta \cdot v_y}{h_0} + \omega_z\right) \\ (v_z - \alpha \cdot v_y) \\ \left(-\frac{v_y}{h_0} + \omega_x\right) \\ \left(\frac{\beta \cdot v_z}{h_0} + \omega_y\right) \\ \left(\frac{\alpha \cdot v_z}{h_0} + \omega_x\right) \end{bmatrix} \Leftrightarrow \begin{bmatrix} x_1 = (v_z + \beta \cdot v_x) \\ x_2 = \left(-\frac{\alpha \cdot v_x}{h_0} + \omega_z\right) \\ x_3 = \left(\frac{v_x}{h_0} + \omega_y\right) \\ x_4 = \left(-\frac{\beta \cdot v_y}{h_0} + \omega_z\right) \\ x_5 = (v_z - \alpha \cdot v_y) \\ x_6 = \left(-\frac{v_y}{h_0} + \omega_x\right) \\ x_7 = \left(\frac{\beta \cdot v_z}{h_0} + \omega_y\right) \\ x_8 = \left(\frac{\alpha \cdot v_z}{h_0} + \omega_x\right) \end{bmatrix} \Leftrightarrow \begin{bmatrix} x_1 = (v_z + \beta \cdot v_x) \\ \omega_z = x_2 + \frac{\alpha \cdot v_x}{h_0} \\ \omega_y = x_3 - \frac{v_x}{h_0} \\ x_4 = \left(-\frac{\beta \cdot v_y}{h_0} + \omega_z\right) \\ x_5 = (v_z - \alpha \cdot v_y) \\ \omega_x = x_6 + \frac{v_y}{h_0} \\ x_7 = \left(\frac{\beta \cdot v_z}{h_0} + \omega_y\right) \\ x_8 = \left(\frac{\alpha \cdot v_z}{h_0} + \omega_x\right) \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} v_z + \beta \cdot v_x = x_1 \\ x_2 + \frac{\alpha \cdot v_x}{h_0} - \frac{\beta \cdot v_y}{h_0} = x_4 \\ v_z - \alpha \cdot v_y = x_5 \\ x_3 - \frac{v_x}{h_0} + \frac{\beta \cdot v_z}{h_0} = x_7 \\ x_6 + \frac{v_y}{h_0} + \frac{\alpha \cdot v_z}{h_0} = x_8 \end{bmatrix} \Leftrightarrow \begin{bmatrix} v_z + \beta \cdot v_x = x_1 \\ x_2 + \frac{\alpha \cdot v_x}{h_0} - \frac{\beta \cdot v_y}{h_0} = x_4 \\ v_z - \alpha \cdot v_y = x_5 \\ v_x = x_3 \cdot h_0 - x_7 \cdot h_0 + \beta \cdot v_z \\ v_y = x_8 \cdot h_0 - x_6 \cdot h_0 - \alpha \cdot v_z \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} v_z + \beta \cdot (\beta \cdot v_z + h_0 \cdot x_3 - h_0 \cdot x_7) = x_1 \\ x_2 + \frac{\alpha \cdot (\beta \cdot v_z + h_0 \cdot x_3 - h_0 \cdot x_7)}{h_0} + \frac{\beta \cdot (\alpha \cdot v_z + h_0 \cdot x_6 - h_0 \cdot x_8)}{h_0} = x_4 \\ v_z + \alpha \cdot (\alpha \cdot v_z + h_0 \cdot x_6 - h_0 \cdot x_8) = x_5 \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} v_z + \beta \cdot (\beta \cdot v_z + h_0 \cdot x_3 - h_0 \cdot x_7) = x_1 \\ x_2 + \frac{\alpha \cdot (\beta \cdot v_z + h_0 \cdot x_3 - h_0 \cdot x_7)}{h_0} + \frac{\beta \cdot (\alpha \cdot v_z + h_0 \cdot x_6 - h_0 \cdot x_8)}{h_0} = x_4 \\ v_z = \frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1} \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1} + \beta \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right) = x_1 \\ x_2 + \frac{\alpha \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right)}{h_0} + \frac{\beta \cdot \left(\alpha \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_6 - h_0 \cdot x_8\right)}{h_0} = x_4 \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1} + \beta \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right) = x_1 \\ x_2 \cdot h_0 + \alpha \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right) + \beta \cdot \left(\alpha \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_6 - h_0 \cdot x_8\right) = x_4 \cdot h_0 \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1} + \beta \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right) = x_1 \\ x_2 \cdot h_0 - x_4 \cdot h_0 + \alpha \cdot \beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + \alpha \cdot h_0 \cdot x_3 - \alpha \cdot h_0 \cdot x_7 + \beta \cdot \left(\alpha \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_6 - h_0 \cdot x_8\right) = 0 \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1} + \beta \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right) = x_1 \\ h_0 \cdot (x_2 - x_4 + \alpha \cdot x_3 - \alpha \cdot x_7) + \alpha \cdot \beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + \beta \cdot \left(\alpha \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_6 - h_0 \cdot x_8\right) = 0 \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1} + \beta \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right) = x_1 \\ h_0 \cdot (\alpha^2 + 1) \cdot (x_2 - x_4 + \alpha \cdot x_3 - \alpha \cdot x_7) + \alpha \cdot \beta \cdot (x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8) + \beta \cdot \left(\alpha \cdot (x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8) + (\alpha^2 + 1) \cdot (h_0 \cdot x_6 - h_0 \cdot x_8)\right) = 0 \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1} + \beta \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right) = x_1 \\ \beta = -\frac{h_0 \cdot (\alpha^2 + 1) \cdot (x_2 - x_4 + \alpha \cdot x_3 - \alpha \cdot x_7)}{\alpha \cdot (x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8) + \left(\alpha \cdot (x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8) + (\alpha^2 + 1) \cdot (h_0 \cdot x_6 - h_0 \cdot x_8)\right)} \end{bmatrix}$$

$$\Leftrightarrow \begin{bmatrix} \frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1} + \beta \cdot \left(\beta \cdot \left(\frac{x_5 - \alpha \cdot h_0 \cdot x_6 + \alpha \cdot h_0 \cdot x_8}{\alpha^2 + 1}\right) + h_0 \cdot x_3 - h_0 \cdot x_7\right) = x_1 \\ \beta = -\frac{h_0 \cdot (\alpha^2 + 1) \cdot (x_2 - x_4 + \alpha \cdot x_3 - \alpha \cdot x_7)}{2 \cdot \alpha \cdot x_5 + h_0 \cdot x_6 - h_0 \cdot x_8 - \alpha^2 \cdot h_0 \cdot x_6 + \alpha^2 \cdot h_0 \cdot x_8} \end{bmatrix}$$

Mit Hilfe der Matlab Symbolic Math Toolbox:

$$
\begin{aligned}
\Leftrightarrow 0 = {} & -\alpha^6 {h_0}^3 x_2 x_3 x_6 + \alpha^6 {h_0}^3 x_2 x_3 x_8 + \alpha^6 {h_0}^3 x_2 x_6 x_7 - \alpha^6 {h_0}^3 x_2 x_7 x_8 + \alpha^6 {h_0}^3 x_3 x_4 x_6 - \alpha^6 {h_0}^3 x_3 x_4 x_8 - \alpha^6 {h_0}^3 x_4 x_6 x_7 + \alpha^6 {h_0}^3 x_4 x_7 x_8 \\
& - \alpha^6 {h_0}^2 {x_3}^2 x_5 + 2\,\alpha^6 {h_0}^2 x_3 x_5 x_7 - \alpha^6 {h_0}^2 x_5 {x_7}^2 - x_1\,\alpha^6 {h_0}^2 {x_6}^2 + 2\,x_1\,\alpha^6 {h_0}^2 x_6 x_8 - x_1\,\alpha^6 {h_0}^2 {x_8}^2 - \alpha^5 {h_0}^3 {x_2}^2 x_6 + \alpha^5 {h_0}^3 {x_2}^2 x_8 \\
& + 2\,\alpha^5 {h_0}^3 x_2 x_4 x_6 - 2\,\alpha^5 {h_0}^3 x_2 x_4 x_8 - \alpha^5 {h_0}^3 {x_3}^2 x_6 + \alpha^5 {h_0}^3 {x_3}^2 x_8 + 2\,\alpha^5 {h_0}^3 x_3 x_6 x_7 - 2\,\alpha^5 {h_0}^3 x_3 x_7 x_8 - \alpha^5 {h_0}^3 {x_4}^2 x_6 \\
& + \alpha^5 {h_0}^3 {x_4}^2 x_8 - \alpha^5 {h_0}^3 {x_6}^3 + 3\,\alpha^5 {h_0}^3 {x_6}^2 x_8 - \alpha^5 {h_0}^3 x_6 {x_7}^2 - 3\,\alpha^5 {h_0}^3 x_6 {x_8}^2 + \alpha^5 {h_0}^3 {x_7}^2 x_8 + \alpha^5 {h_0}^3 {x_8}^3 + 4\,x_1\,\alpha^5 h_0 x_5 x_6 \\
& - 4\,x_1\,\alpha^5 h_0 x_5 x_8 - 3\,\alpha^4 {h_0}^3 x_2 x_3 x_6 + 3\,\alpha^4 {h_0}^3 x_2 x_3 x_8 + 3\,\alpha^4 {h_0}^3 x_2 x_6 x_7 - 3\,\alpha^4 {h_0}^3 x_2 x_7 x_8 + 3\,\alpha^4 {h_0}^3 x_3 x_4 x_6 - 3\,\alpha^4 {h_0}^3 x_3 x_4 x_8 \\
& - 3\,\alpha^4 {h_0}^3 x_4 x_6 x_7 + 3\,\alpha^4 {h_0}^3 x_4 x_7 x_8 + \alpha^4 {h_0}^2 {x_2}^2 x_5 - 2\,\alpha^4 {h_0}^2 x_2 x_4 x_5 - 2\,\alpha^4 {h_0}^2 {x_3}^2 x_5 + 4\,\alpha^4 {h_0}^2 x_3 x_5 x_7 + \alpha^4 {h_0}^2 {x_4}^2 x_5 \\
& + 5\,\alpha^4 {h_0}^2 x_5 {x_6}^2 - 10\,\alpha^4 {h_0}^2 x_5 x_6 x_8 - 2\,\alpha^4 {h_0}^2 x_5 {x_7}^2 + 5\,\alpha^4 {h_0}^2 x_5 {x_8}^2 + x_1\,\alpha^4 {h_0}^2 {x_6}^2 - 2\,x_1\,\alpha^4 {h_0}^2 x_6 x_8 + x_1\,\alpha^4 {h_0}^2 {x_8}^2 \\
& - 4\,x_1\,\alpha^4 {x_5}^2 - 2\,\alpha^3 {h_0}^3 {x_2}^2 x_6 + 2\,\alpha^3 {h_0}^3 {x_2}^2 x_8 + 4\,\alpha^3 {h_0}^3 x_2 x_4 x_6 - 4\,\alpha^3 {h_0}^3 x_2 x_4 x_8 - 2\,\alpha^3 {h_0}^3 {x_3}^2 x_6 + 2\,\alpha^3 {h_0}^3 {x_3}^2 x_8 \\
& + 4\,\alpha^3 {h_0}^3 x_3 x_6 x_7 - 4\,\alpha^3 {h_0}^3 x_3 x_7 x_8 - 2\,\alpha^3 {h_0}^3 {x_4}^2 x_6 + 2\,\alpha^3 {h_0}^3 {x_4}^2 x_8 + 2\,\alpha^3 {h_0}^3 {x_6}^3 - 6\,\alpha^3 {h_0}^3 {x_6}^2 x_8 - 2\,\alpha^3 {h_0}^3 x_6 {x_7}^2 \\
& + 6\,\alpha^3 {h_0}^3 x_6 {x_8}^2 + 2\,\alpha^3 {h_0}^3 {x_7}^2 x_8 - 2\,\alpha^3 {h_0}^3 {x_8}^3 - 8\,\alpha^3 h_0 {x_5}^2 x_6 + 8\,\alpha^3 h_0 {x_5}^2 x_8 - 3\,\alpha^2 {h_0}^3 x_2 x_3 x_6 + 3\,\alpha^2 {h_0}^3 x_2 x_3 x_8 \\
& + 3\,\alpha^2 {h_0}^3 x_2 x_6 x_7 - 3\,\alpha^2 {h_0}^3 x_2 x_7 x_8 + 3\,\alpha^2 {h_0}^3 x_3 x_4 x_6 - 3\,\alpha^2 {h_0}^3 x_3 x_4 x_8 - 3\,\alpha^2 {h_0}^3 x_4 x_6 x_7 + 3\,\alpha^2 {h_0}^3 x_4 x_7 x_8 + 2\,\alpha^2 {h_0}^2 {x_2}^2 x_5 \\
& - 4\,\alpha^2 {h_0}^2 x_2 x_4 x_5 - \alpha^2 {h_0}^2 {x_3}^2 x_5 + 2\,\alpha^2 {h_0}^2 x_3 x_5 x_7 + 2\,\alpha^2 {h_0}^2 {x_4}^2 x_5 - 6\,\alpha^2 {h_0}^2 x_5 {x_6}^2 + 12\,\alpha^2 {h_0}^2 x_5 x_6 x_8 - \alpha^2 {h_0}^2 x_5 {x_7}^2 \\
& - 6\,\alpha^2 {h_0}^2 x_5 {x_8}^2 + x_1\,\alpha^2 {h_0}^2 {x_6}^2 - 2\,x_1\,\alpha^2 {h_0}^2 x_6 x_8 + x_1\,\alpha^2 {h_0}^2 {x_8}^2 + 4\,\alpha^2 {x_5}^3 - 4\,x_1\,\alpha^2 {x_5}^2 - \alpha\,{h_0}^3 {x_2}^2 x_6 + \alpha\,{h_0}^3 {x_2}^2 x_8 \\
& + 2\,\alpha\,{h_0}^3 x_2 x_4 x_6 - 2\,\alpha\,{h_0}^3 x_2 x_4 x_8 - \alpha\,{h_0}^3 {x_3}^2 x_6 + \alpha\,{h_0}^3 {x_3}^2 x_8 + 2\,\alpha\,{h_0}^3 x_3 x_6 x_7 - 2\,\alpha\,{h_0}^3 x_3 x_7 x_8 - \alpha\,{h_0}^3 {x_4}^2 x_6 + \alpha\,{h_0}^3 {x_4}^2 x_8 \\
& - \alpha\,{h_0}^3 {x_6}^3 + 3\,\alpha\,{h_0}^3 {x_6}^2 x_8 - \alpha\,{h_0}^3 x_6 {x_7}^2 - 3\,\alpha\,{h_0}^3 x_6 {x_8}^2 + \alpha\,{h_0}^3 {x_7}^2 x_8 + \alpha\,{h_0}^3 {x_8}^3 + 4\,\alpha\,h_0 {x_5}^2 x_6 - 4\,\alpha\,h_0 {x_5}^2 x_8 - 4\,x_1\,\alpha\,h_0 x_5 x_6 \\
& + 4\,x_1\,\alpha\,h_0 x_5 x_8 - {h_0}^3 x_2 x_3 x_6 + {h_0}^3 x_2 x_3 x_8 + {h_0}^3 x_2 x_6 x_7 - {h_0}^3 x_2 x_7 x_8 + {h_0}^3 x_3 x_4 x_6 - {h_0}^3 x_3 x_4 x_8 - {h_0}^3 x_4 x_6 x_7 + {h_0}^3 x_4 x_7 x_8 \\
& + {h_0}^2 {x_2}^2 x_5 - 2\,{h_0}^2 x_2 x_4 x_5 + {h_0}^2 {x_4}^2 x_5 + {h_0}^2 x_5 {x_6}^2 - 2\,{h_0}^2 x_5 x_6 x_8 + {h_0}^2 x_5 {x_8}^2 - x_1\,{h_0}^2 {x_6}^2 + 2\,x_1\,{h_0}^2 x_6 x_8 - x_1\,{h_0}^2 {x_8}^2
\end{aligned}
$$

Anhang C: Koeffizienten der Gleichung (4.34)

$$a_0 = h_0^{\,2}\,x_2^{\,2}\,x_5 - h_0^{\,2}\,x_1\,x_6^{\,2} - h_0^{\,2}\,x_1\,x_8^{\,2} + h_0^{\,2}\,x_4^{\,2}\,x_5 + h_0^{\,2}\,x_5\,x_6^{\,2} + h_0^{\,2}\,x_5\,x_8^{\,2} - 2\,h_0^{\,2}\,x_2\,x_4\,x_5 - h_0^{\,3}\,x_2\,x_3\,x_6 + h_0^{\,3}\,x_2\,x_3\,x_8 + h_0^{\,3}\,x_3\,x_4\,x_6 + 2\,h_0^{\,2}\,x_1\,x_6\,x_8 + h_0^{\,3}\,x_2\,x_6\,x_7 - h_0^{\,3}\,x_3\,x_4\,x_8 - h_0^{\,3}\,x_2\,x_7\,x_8 - h_0^{\,3}\,x_4\,x_6\,x_7 - 2\,h_0^{\,2}\,x_5\,x_6\,x_8 + h_0^{\,3}\,x_4\,x_7\,x_8$$

$$a_1 = -h_0^{\,3}\,x_2^{\,2}\,x_6 + h_0^{\,3}\,x_2^{\,2}\,x_8 + 2\,h_0^{\,3}\,x_2\,x_4\,x_6 - 2\,h_0^{\,3}\,x_2\,x_4\,x_8 - h_0^{\,3}\,x_3^{\,2}\,x_6 + h_0^{\,3}\,x_3^{\,2}\,x_8 + 2\,h_0^{\,3}\,x_3\,x_6\,x_7 - 2\,h_0^{\,3}\,x_3\,x_7\,x_8 - h_0^{\,3}\,x_4^{\,2}\,x_6 + h_0^{\,3}\,x_4^{\,2}\,x_8 - h_0^{\,3}\,x_6^{\,3} + 3\,h_0^{\,3}\,x_6^{\,2}\,x_8 - h_0^{\,3}\,x_6\,x_7^{\,2} - 3\,h_0^{\,3}\,x_6\,x_8^{\,2} + h_0^{\,3}\,x_7^{\,2}\,x_8 + h_0^{\,3}\,x_8^{\,3} + 4\,h_0\,x_5^{\,2}\,x_6 - 4\,h_0\,x_5^{\,2}\,x_8 - 4\,x_1\,h_0\,x_5\,x_6 + 4\,x_1\,h_0\,x_5\,x_8$$

$$a_2 = -3\,h_0^{\,3}\,x_2\,x_3\,x_6 + 3\,h_0^{\,3}\,x_2\,x_3\,x_8 + 3\,h_0^{\,3}\,x_2\,x_6\,x_7 - 3\,h_0^{\,3}\,x_2\,x_7\,x_8 + 3\,h_0^{\,3}\,x_3\,x_4\,x_6 - 3\,h_0^{\,3}\,x_3\,x_4\,x_8 - 3\,h_0^{\,3}\,x_4\,x_6\,x_7 + 3\,h_0^{\,3}\,x_4\,x_7\,x_8 + 2\,h_0^{\,2}\,x_2^{\,2}\,x_5 - 4\,h_0^{\,2}\,x_2\,x_4\,x_5 - h_0^{\,2}\,x_3^{\,2}\,x_5 + 2\,h_0^{\,2}\,x_3\,x_5\,x_7 + 2\,h_0^{\,2}\,x_4^{\,2}\,x_5 - 6\,h_0^{\,2}\,x_5\,x_6^{\,2} + 12\,h_0^{\,2}\,x_5\,x_6\,x_8 - h_0^{\,2}\,x_5\,x_7^{\,2} - 6\,h_0^{\,2}\,x_5\,x_8^{\,2} + x_1\,h_0^{\,2}\,x_6^{\,2} - 2\,x_1\,h_0^{\,2}\,x_6\,x_8 + x_1\,h_0^{\,2}\,x_8^{\,2} + 4\,x_5^{\,3} - 4\,x_1\,x_5^{\,2}$$

$$a_3 = -2\,h_0^{\,3}\,x_2^{\,2}\,x_6 + 2\,h_0^{\,3}\,x_2^{\,2}\,x_8 + 4\,h_0^{\,3}\,x_2\,x_4\,x_6 - 4\,h_0^{\,3}\,x_2\,x_4\,x_8 - 2\,h_0^{\,3}\,x_3^{\,2}\,x_6 + 2\,h_0^{\,3}\,x_3^{\,2}\,x_8 + 4\,h_0^{\,3}\,x_3\,x_6\,x_7 - 4\,h_0^{\,3}\,x_3\,x_7\,x_8 - 2\,h_0^{\,3}\,x_4^{\,2}\,x_6 + 2\,h_0^{\,3}\,x_4^{\,2}\,x_8 + 2\,h_0^{\,3}\,x_6^{\,3} - 6\,h_0^{\,3}\,x_6^{\,2}\,x_8 - 2\,h_0^{\,3}\,x_6\,x_7^{\,2} + 6\,h_0^{\,3}\,x_6\,x_8^{\,2} + 2\,h_0^{\,3}\,x_7^{\,2}\,x_8 - 2\,h_0^{\,3}\,x_8^{\,3} - 8\,h_0\,x_5^{\,2}\,x_6 + 8\,h_0\,x_5^{\,2}\,x_8$$

$$a_4 = -3\,h_0^{\,3}\,x_2\,x_3\,x_6 + 3\,h_0^{\,3}\,x_2\,x_3\,x_8 + 3\,h_0^{\,3}\,x_2\,x_6\,x_7 - 3\,h_0^{\,3}\,x_2\,x_7\,x_8 + 3\,h_0^{\,3}\,x_3\,x_4\,x_6 - 3\,h_0^{\,3}\,x_3\,x_4\,x_8 - 3\,h_0^{\,3}\,x_4\,x_6\,x_7 + 3\,h_0^{\,3}\,x_4\,x_7\,x_8 + h_0^{\,2}\,x_2^{\,2}\,x_5 - 2\,h_0^{\,2}\,x_2\,x_4\,x_5 - 2\,h_0^{\,2}\,x_3^{\,2}\,x_5 + 4\,h_0^{\,2}\,x_3\,x_5\,x_7 + h_0^{\,2}\,x_4^{\,2}\,x_5 + 5\,h_0^{\,2}\,x_5\,x_6^{\,2} - 10\,h_0^{\,2}\,x_5\,x_6\,x_8 - 2\,h_0^{\,2}\,x_5\,x_7^{\,2} + 5\,h_0^{\,2}\,x_5\,x_8^{\,2} + x_1\,h_0^{\,2}\,x_6^{\,2} - 2\,x_1\,h_0^{\,2}\,x_6\,x_8 + x_1\,h_0^{\,2}\,x_8^{\,2} - 4\,x_1\,x_5^{\,2}$$

$$a_5 = -{h_0}^3\,{x_2}^2\,x_6 + {h_0}^3\,{x_2}^2\,x_8 + 2\,{h_0}^3\,x_2\,x_4\,x_6 - 2\,{h_0}^3\,x_2\,x_4\,x_8 - {h_0}^3\,{x_3}^2\,x_6 + {h_0}^3\,{x_3}^2\,x_8 + 2\,{h_0}^3\,x_3\,x_6\,x_7 - 2\,{h_0}^3\,x_3\,x_7\,x_8 - {h_0}^3\,{x_4}^2\,x_6 + {h_0}^3\,{x_4}^2\,x_8 - {h_0}^3\,{x_6}^3 + 3\,{h_0}^3\,{x_6}^2\,x_8 - {h_0}^3\,x_6\,{x_7}^2 - 3\,{h_0}^3\,x_6\,{x_8}^2 + {h_0}^3\,{x_7}^2\,x_8 + {h_0}^3\,{x_8}^3 + 4\,x_1\,x_5\,h_0\,x_6 - 4\,x_1\,x_5\,h_0\,x_8$$

$$a_6 = {h_0}^3\,x_2\,x_3\,x_8 - {h_0}^2\,{x_3}^2\,x_5 - {h_0}^2\,x_1\,{x_8}^2 - {h_0}^2\,x_5\,{x_7}^2 - {h_0}^3\,x_2\,x_3\,x_6 - {h_0}^2\,x_1\,{x_6}^2 + {h_0}^3\,x_3\,x_4\,x_6 + 2\,{h_0}^2\,x_1\,x_6\,x_8 + 2\,{h_0}^2\,x_3\,x_5\,x_7 + {h_0}^3\,x_2\,x_6\,x_7 - {h_0}^3\,x_3\,x_4\,x_8 - {h_0}^3\,x_2\,x_7\,x_8 - {h_0}^3\,x_4\,x_6\,x_7 + {h_0}^3\,x_4\,x_7\,x_8$$

Anhang D: Lösungen der unbekannten Größen v_x, v_y, v_z, ω_x, ω_y, ω_z, β

Lösung für β:

$$\beta_{est} = -\frac{h_0\,({\alpha_{est}}^2 + 1)\,(x_2 - x_4 + \alpha_{est}\,x_3 - \alpha_{est}\,x_7)}{2\,\alpha_{est}\,x_5 + h_0\,x_6 - h_0\,x_8 - {\alpha_{est}}^2\,h_0\,x_6 + {\alpha_{est}}^2\,h_0\,x_8}$$

Lösung für v_z:

$$v_{z\_est} = \frac{x_5 - \alpha_{est}\,h_0\,x_6 + \alpha_{est}\,h_0\,x_8}{{\alpha_{est}}^2 + 1}$$

Lösung für v_y:

$$v_{y\_est} = h_0\,x_8 - h_0\,x_6 - \alpha_{est}\,v_{z\_est}$$

Lösung für v_x:

$$v_{x\_est} = \beta_{est}\,v_{z\_est} + h_0\,x_3 - h_0\,x_7$$

Lösung für ω_z:

$$\omega_{z\_est} = x_2 + \frac{\alpha_{est}\,v_{x\_est}}{h_0}$$

Lösung für ω_y:

$$\omega_{y\_est} = x_3 - \frac{v_{x\_est}}{h_0}$$

Lösung für ω_x:

$$\omega_{x\_est} = x_6 + \frac{v_{y\_est}}{h_0}$$

Anhang E: Verwendete Software und Hardware

Für die hier beschrieben Projektarbeit wurden nachfolgende Software und Hardware verwendet:

Software	**Version**
MATLAB	Version 9.8 (R2020a)
Simulink	Version 4.3 (R2020a)
Computer Vision Toolbox	Version 9.2 (R2020a)
Symbolic Math Toolbox	Version 8.5 (R2020a)
Simulink Support Package for Parrot Minidrones	Version 20.1.3

Hardware	**Modell**
Quadrokopter	Parrot Rolling Spider