Multi-Sensor Data Fusion zur optischen Navigation im Kontext unbemannter Luftfahrzeuge

Bachelorarbeit

von

Eike Harm Otto Bremers

geboren am 24. Dezember 1997 in Braunschweig

24. April 2020

Betreuer:
Prof. Dr.-Ing Stefan Levedag
Prof. Dr.-Ing Peter Hecker
M.Sc. Nikolaus Ammann

Technische Universität Braunschweig
in Zusammenarbeit mit dem
Deutschen Zentrum für Luft- und Raumfahrt
Institut für Flugsystemtechnik
Abteilung Unbemannte Luftfahrzeuge

Hiermit versichere ich, dass ich die schriftliche Hausarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Die Stellen meiner Arbeit, die dem Wortlaut oder dem Sinne nach anderen Werken und Quellen, einschließlich Quellen aus dem Internet, entnommen sind, habe ich in jedem Fall unter Angabe der Quelle deutlich als Entlehnung kenntlich gemacht. Dasselbe gilt sinngemäß für Tabellen, Karten und Abbildungen.

Zusammenfassung

Im Rahmen dieser Bachelorarbeit wird die Erweiterung eines Zustandsschätzers um weitere Sensoren angestrebt. Hierbei beruht der Zustandsschätzer auf einem Fehlerzustandsschätzer, welcher unabhängig von der aktuellen Zustandsgenerierung den aktuellen Fehler abschätzt und korrigiert. Zunächst soll die Erweiterung des Zustandsschätzers vorgenommen werden, anschließend soll dann die Funktionalität der neuen Sensoren überprüft werden.
Bei der Bearbeitung der Arbeit wurde auf ein bestehendes Framework aufgebaut, welches die Funktionalität eines Fehlerzustandsschätzers bereits implementiert hatte. Die Berechnung des aktuellen Systemzustandes teilt sich dabei in zwei Teile auf, zum einen die aktuelle Zustandsschätzung, welche aus einem einfachen Inertialen Navigationsalgorithmus(INS) besteht, zum anderen dem tatsächlichen Fehlerschätzer. Dieser besteht dabei aus einem Unscented Kalman Filter, in welchem der aktuelle Fehler mithilfe von weiteren Sensoren berechnet wird. Anschließend wird mithilfe des Fehlerzustandes der tatsächliche Fehler korrigiert. Im Rahmen dieser Arbeit wurden dabei weitere Sensoren implementiert. Ursprünglich war hierbei nur die reine Fusionierung der aktuellen Position implementiert. Da diese jedoch von der Verfügbarkeit von GNSS Signalen abhängig ist, ist diese störanfällig. Somit wurden im Folgenden vier Sensoren implementiert. Nämlich die Geschwindigkeit im erdfesten Koordinatensystem, die Lage im Inertialsystem wie es zum Beispiel von Sternensensoren genutzt wird, das Magnetfeld, sowie die relative Messung der Position mithilfe von visueller Odometrie. Hierbei mussten jeweils zwei Stellen erweitert werden. Zum einen die Generierung der tatsächlichen Messwerte bzw. die Umwandlung in den Fehlerzustandsraum, zum anderen die Generierung der geschätzten Messwerte im Kalman Filter.
Nachdem die Sensoren implementiert waren, wurden diese getestet. Hierzu wurde innerhalb einer bestehenden Testumgebung eine Flugtrajektorie genutzt. Hierbei wurden die Sensoren zunächst einzeln getestet, wobei alle der implementierten Sensoren einzeln betrachtet für eine deutliche Verbesserung der Zustandsschätzung sorgen. Dabei zeigten sowohl die Geschwindigkeit als auch die Lage eine enge Korrelation zwischen Messtoleranz und geschätzter dreifacher Standardabweichung. Dabei kann jedoch die Fusionierung der Geschwindigkeit etwaige zu Beginn auftretende Positionsabweichungen nicht ausgleichen. Weiterhin bleibt bei reiner Fusionierung des Magnetfeldes ein gewisser Offset bestehen, dieser tritt jedoch nicht auf, falls ein weiterer Sensor fusioniert wird. Dieser lässt sich dadurch erklären, dass das lokale Magnetfeld von der Position abhängig ist. Die Fusionierung der relativen Position lieferte bezüglich der Positionsbestimmung die zu erwartende Verbesserung, jedoch ist die geschätzte Standardabweichung deutlich zu gering.

Inhaltsverzeichnis

Institut für Flugsystemtechnik

Deutsches Zentrum für Luft- und Raumfahrt

Technische Universität Braunschweig

DLR e. V. Institut für Flugsystemtechnik
Lilienthalplatz 7, 38108 Braunschweig

An die
Fakultät Maschinenbau
der Technischen Universität Braunschweig

Name Prof. Dr.-Ing. Stefan Levedag

Telefon +49 (0)531 295 2600
Telefax +49 (0)531 295 2864
E-Mail Stefan.Levedag@dlr.de

07.01.2020

Thema der Bachelorarbeit für Herrn Eike Harm Otto Bremers

Matrikel-Nr.: 4815680 Studiengang: Luft- und Raumfahrttechnik

E-Mail: e.bremers@tu-braunschweig.de

Sprache der Arbeit: Deutsch

Titel:

- Multi-Sensor Data Fusion zur optischen Navigation im Kontext unbemannter Luftfahrzeuge
- Multi-sensor data fusion for optical navigation in the context of unmanned aircraft

Erläuterung:

Unbemannte Luftfahrzeuge (UAVs) nutzen zur Selbstlokalisierung vor allem globale Navigationssatelliten-systeme, wie z.B. GPS, GLONASS oder Galileo. Um robuster gegen Störungen der GNSS zu sein, wird an alternativen Navigationssystemen (engl.: Alternative Positioning Navigation and Time (APNT)) geforscht. Die Nutzung von optischen Sensordaten, wie Kamerabildern, ist dabei eine große Forschungsrichtung. Basierend auf den Bilddaten, wird dabei die Eigenbewegung geschätzt.

Das DLR untersucht die Auswirkungen, der Fusion von unterschiedlichsten Sensoren auf die Positionsbestimmung im Kontext von unbemannten Luftfahrzeugen. Zu diesen Sensoren gehören neben den Inertialsensoren auch typische Sensoren aus der Luftfahrt, wie Barometer, Altimeter, GNSS oder Geschwindigkeitsmesser. Darüber hinaus werden aber auch Sensoren aus der Raumfahrt fusioniert. Zu diesen Sensoren gehören beispielsweise die Sternensensoren, welche eine Lage bzgl. des Sonnensystems messen. Durch die Fusion dieser Sensordaten ist die entwickelte Zustandsschätzung in unterschiedlichsten Projekten in er Luft- und Raumfahrt zum Einsatz gekommen. Die Evaluation dieses Systems erfolgt in einer Simulation, in der die Sensoren

phänomenologisch modelliert sind, einer Hardware-in-the-Loop Postprocessing Emulation, welche reale Flugversuchsdaten nutzt, und im realen Flugversuch.
Die aktuelle Zustandsschätzung wird kontinuierlich weiter entwickelt. Zu diesem Zweck sollen verschiedene Sensordaten neu modelliert und fusioniert werden. Die Auswirkungen auf die Genauigkeit und Beobachtbarkeit, welche durch die Fusion der einzelnen Sensoren resultiert, ist dabei besonders zu evaluieren.

Aufgabenstellung:

Die aktuelle Zustandsschätzung soll im Rahmen dieser Bachelorarbeit erweitert werden. Die Zustandsschätzung, basierend auf einem Unscented Kalman Filter, soll um weitere Messgleichungen zur Fusion weiterer Sensoren erweitert werden. Dazu soll zunächst eine Literaturrecherche durchgeführt werden, bei der die gängigen Verfahren zur Zustandsschätzung genannt und deren Unterscheide herausgearbeitet werden. Es soll auf die Vor- und Nachteile des Unscented Kalman Filter und der Fehlerzustandsschätzung eingegangen werden. Weiter sollen die Messmodelle für die einzelnen Sensoren erarbeitet und anschließend implementiert werden. Die Implementierung erfolgt dabei in C++. Für die Evaluation soll die Simulationsumgebung genutzt werden. Hier sollen die Auswirkungen der Sensorparameter auf die Genauigkeit und Beobachtbarkeit der Zustandsschätzung untersucht werden. Abschließend besteht die Möglichkeit den Algorithmus mit Flugversuchsdaten im Postprocessing oder im Flugversuch zu testen.

Teilaufgaben:

- Literaturrecherche
- Erweiterung der Messgleichungen des Zustandsschätzers für verschiedene Sensordaten
- Evaluation der Fusion in der Simulation mit einem Fokus auf Genauigkeit und Beobachtbarkeit
- Optional: Evaluation des Algorithmus mit Flugversuchsdaten im Playback oder im Flugversuch

Literatur:

[1] Ammann, Nikolaus Alexander und Andert, Franz (2017) Visual Navigation for Autonomous, Precise and Safe Landing on Celestial Bodies using Unscented Kalman Filtering. In: IEEE Aerospace Conference Proceedings

[2] S. J. Julier, J. K. Uhlmann und H. F. Durrant-Whyte (1995) A new approach for filtering nonlinear systems. In Proceedings of the 1995 American Control Conference

Zeitrahmen:

Beginn: 24.01.2019

Abgabe 24.04.2019

Die Bachelorarbeit wird beim Deutschen Zentrum für Luft- und Raumfahrt e.V. (DLR) bearbeitet. Alle im Laufe der Arbeit zugänglich gemachten Informationen sind vertraulich zu behandeln.

Bearbeitungszeit: ☒ 3 Monate (Bachelor) ☐ 6 Monate (Master)

☐ Projektarbeit (Bachelor) (2 Monate) ☐ Studienarbeit (Master) (3-4 Monate)

Betreuer:

Nikolaus Ammann, M.Sc.
Institut für Flugsystemtechnik / Abteilung Unbemannte Luftfahrzeuge
Deutsches Zentrum für Luft- und Raumfahrt e.V.
Lilienthalplatz 7
38108 Braunschweig

Erstprüfer:

Professor Dr.-Ing. Levedag
Deutsches Zentrum für Luft- und Raumfahrt e.V.
Institut für Flugsystemtechnik
Lilienthalplatz 7
38108 Braunschweig

Zweitprüfer:

Professor Dr.-Ing. P. Hecker
Technische Universität Braunschweig
Institut für Flugführung
Hermann-Blenk-Straße 27
38108 Braunschweig

Prof. Dr.-Ing. Levedag

1 Einleitung

Unbemannte Luftfahrzeuge erfreuen sich immer größer werdender Beliebtheit in verschiedensten Anwendungsgebieten. Dabei gibt es diverse Möglichkeiten zur Positions- und Lagebestimmung. Die am weitesten verbreiteten Möglichkeiten zur Bestimmung der aktuellen Position sind dabei die verschiedenen Navigationssatellitensysteme(GNSS) wie GPS, Galileo, Beidou oder GLONASS. Um die vergleichsweise hohe Störanfälligkeit von GNS Systemen zu umgehen, werden unterschiedliche Sensoren untersucht. Hierbei bieten sich speziell visuelle Sensoren wie Kameras oder Lidare an, welche mithilfe der visuellen Odometrie ausgewertet werden. Daneben gibt es allerdings noch andere Sensoren wie Magnetometer, Geschwindigkeitsmesser, Barometer und auch Sternenverfolger.
Um diese Sensoren auszuwerten gibt es unterschiedlichste Methoden. Eines der aktuellen Forschungsthemen zum Thema Sensorfusion ist die Auswertung mithilfe eines Fehlerzustandsschätzers. Dabei wird die Berechnung der aktuellen Position und die Korrektur getrennt voneinander vorgenommen. Dieses Verfahren bietet zahlreiche Vorteile. Ziel dieser Arbeit ist es, die Funktionalität eines Fehlerzustandsschätzers um weitere Sensoren zu erweitern. Anschließend soll die Funktionalität der implementierten Sensoren untersucht werden. Im Rahmen dieser Arbeit werden vier verschiedene Sensoren ausgewählt, die im Folgenden näher betrachtet werden sollen. Bei diesen Sensoren handelt es sich um einen Sternenverfolger und einen Magnetfeldsensor. Darüber hinaus werden noch Messwerte der Geschwindigkeit sowie die relative Position, welche mithilfe visueller Sensoren generiert wird, fusioniert. Da weiterhin die absolute Position bereits implementiert wurde, sind somit alle Systemzustände vertreten, um den Zustand des Flugobjektes vollständig zu beschreiben.
Im Folgenden sollen zunächst die verschiedenen benötigten Grundlagen vorgestellt werden. Dazu werden die verschiedenen Koordinatensysteme und ihre Transformationen vorgestellt. Danach werden die spezifischeren Grundlagen wie der Aufbau von Kalman Filtern sowie die einzelnen Sensoren näher beleuchtet. Weiterhin sollen die verschiedenen genutzten Tools und die Umgebung beschrieben werden. Abschließend werden die Sensoren einzeln implementiert und getestet.

2 Grundlagen

Im Folgenden werden die verschiedenen Grundlagen vorgestellt. Hierbei wird zunächst auf die mathematischen Grundlagen, wie die verschiedenen Koordinatensysteme sowie die Transformation zwischen ihnen eingegangen. Anschließend wird auf die Zustandsschätzung eingegangen und diese auf kinematische Systeme angewendet. Zum Schluss werden die verschiedenen genutzten Sensoren vorgestellt.

2.1 Mathematische Grundlagen

2.1.1 Koordinatensysteme

Im Folgenden werden die verschiedenen genutzten Koordinatensysteme vorgestellt.

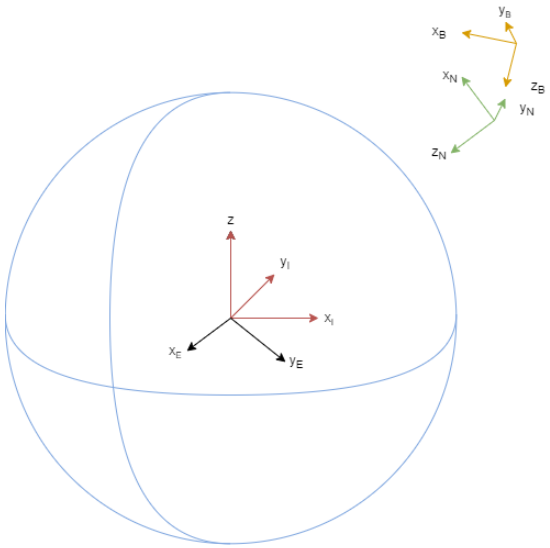


Abbildung 1: Überblick über die verschiedenen Koordinatensysteme

ECEF Das geozentrische Koordinatensystem (ECEF) ist ein kartesisches Koordinatensystem, dessen Ursprung im Erdmittelpunkt liegt, weiterhin verläuft die x-Achse durch den Schnittpunkt aus Nullmeridian und Äquator. Die z-Achse bildet in diesem Fall die Rotationsachse der Erde, mit positiver Achse in Richtung des Nordpols. Die y-Achse vervollständigt das rechtshändige kartesische Koordinatensystem. Dieses Koordinatensystem bildet die Grundlage zur Satellitennavigation. [8] Der Index bei Transformationen beziehungsweise Größenangaben ist E. Eine weitere Darstellungsmöglichkeit ist hierbei das sogenannte geografische Koordinatensystem (LLA), bei dem es sich um ein elliptisches Koordinatensystem mit denselben Referenzpunkten handelt. Allerdings verläuft in diesem Fall der 0. Längengrad durch Greenwich und der Äquator bildet den 0. Breitengrad.

ECI Beim erdzentrierten inertialen Koordinatensystem (earth-centered inertial) handelt es sich um ein kartesisches Koordinatensystem, dessen Ursprung im Mittelpunkt der Erde liegt, und welches nicht gegenüber dem Firmament rotiert. Dabei bildet die Rotationsachse der Erde die z-Achse. Weiterhin gibt es verschiedene Modelle zur Bestimmung der x-Achse. Diese gleichen sich jedoch alle darin, dass diese am Äquinoktium, also der Tag-Nacht-Gleiche, in Richtung Sonne zeigt. Das am weitesten verbreitete Modell ist dabei das J2000 Modell. Die x-Achse ist hierbei der Schnittpunkt zwischen der Äquatorial- und der Ekliptikebene am 1. Januar 2000 um 11:59 UTC. [7] J2000 beschreibt die entsprechende Epoche. Der Index bei Transformationen beziehungsweise Größenangaben ist I .

NED Das lokale Tangentialebenen-Koordinatensytem (North-East-Down) ist ein lokales kartesisches Koordinatensystem, dessen z-Achse senkrecht auf der lokalen Tangentialebene steht und mit positiver Koordinatenachse in Richtung Erde zeigt. Weiterhin zeigt die x-Achse nach Norden und die y-Achse nach Osten. Der Index bei Transformationen beziehungsweise Größenangaben ist N.

BFF Das körperfeste Koordinatensystem (body-frame-fixed) ist ein in diesem Zusammenhang körperfestes, lokales Koordinatensystem, dessen x-Achse in Richtung der Flugrichtung zeigt, die y-Achse geht nach rechts beziehungsweise steuerbord. Die z-Achse vervollständigt das rechtshändige Koordinatensystem und steht somit senkrecht auf der xy-Ebene. Der Index bei Transformationen beziehungsweise Größenangaben ist B.

2.1.2 Koordinatentransformationen

Im Folgenden werden die Umwandlungen zwischen den verschiedenen Koordinatensystemen vorgestellt. Dies ist nötig, weil zum Beispiel die Sensoren in unterschiedlichen Koordinatensystemen gegeben sind. Hierbei werden zunächst die verschiedenen Rotationsformalismen vorgestellt, da diese unterschiedliche Vor- und Nachteile haben. Anschließend werden die entsprechenden Zusammenhänge zwischen den zuvor genannten Koordinatensystemen vorgestellt.

Rotationen Die Transformation der Koordianatensysteme besteht im Allgemeinen aus zwei Operatoren, der Rotation und in einigen Fällen der Translation. Im Folgenden werde ich näher auf die Rotation zwischen zwei Koordinatensystemen eingehen. Dabei gilt, dass $R_{\alpha 2\beta}$ die Rotation vom α-Koordinatensystem in das β-Koordinatensystem ist. Weiterhin gilt, dass sich die Rotationen invertieren lassen, sodass gilt,

$$R_{\alpha 2\beta} = R^{-1}_{\beta 2\alpha} = R^{T}_{\beta 2\alpha}. \tag{2.1}$$

Um diese Rotationen mathematisch zu beschreiben, werden in dieser Arbeit vier verschiedene Ansätze genutzt. Wichtig ist, dass bei allen vorgestellten Rotationstypen das Kommutativ-Gesetz nicht gilt.

Euler Winkel Bei den sogenannten Euler Winkeln handelt es sich um das anschaulichste Prinzip der hier genannten. Hierbei wird eine Rotation als die drei Winkel dargestellt, die jeweils für die Rotation um eine der drei Achsen des kartesischen Koordinatensystems stehen. Da die Reihenfolge der Rotation um die einzelnen Achsen jedoch nicht kommutativ ist, wurde die Reihenfolge in der Luftfahrt durch die Luftfahrtnorm DIN 9300 festgelegt. [16] Hierbei wird zunächst um die Gierachse gedreht, dann um die Nickachse und zuletzt um die Rollachse. Ein weiteres Problem ist hierbei der Gimbal Lock, bei dem, wenn einer der Nickwinkel 90° erreicht, die Rotation nicht mehr eindeutig ist.

Rotationsmatrizen Eine Rotationsmatrix bildet die direkte mathematische Beschreibung der Rotation eines Vektors um die entsprechenden Euler Winkel. Die Generierung einer Rotationsmatrix ist dabei von der Reihenfolge der Rotationen um die einzelnen Achsen abhängig, eine Rotationsmatrix selber ist allerdings eindeutig. Weiterhin gilt, da die Matrizen orthogonal sind, dass $R^T = R^{-1}$. Die Rotation um in diesem Fall die z-Achse lässt sich wie folgt aufstellen:

$$R_z(\alpha) = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

Die um die y-Achse bildet sich zu

$$R_y(\alpha) = \begin{bmatrix} \sin(\alpha) & 0 & \cos(\alpha) \\ 0 & 1 & 0 \\ \cos(\alpha) & 0 & -\sin(\alpha) \end{bmatrix} \tag{2.3}$$

Und die Rotation um die x-Achse ist

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{2.4}$$

Quaternionen Bei Quaternionen handelt es sich grundsätzlich um eine Erweiterung des Zahlenraums der komplexen Zahlen. Mit ihnen lassen sich Rotationen im dreidimensionalen Raum darstellen, ohne die oben beschriebenen Probleme wie Gimbal Lock zu erhalten. Wichtig hierbei ist, dass es sich um normalisierte Quaternionen handelt, da es ansonsten nicht zu der gewünschten Rotation kommt. Hiebei besteht ein Quaternion aus einem Realteil sowie drei Imaginärteilen. Die Imaginärteile stellen hierbei die Rotationsachse dar und der Realteil den zur Rotation gehörigen Winkel um ebenjene.

$$\mathbf{Q} = \mathrm{s} + \mathbf{x} = (w, x, y, z) \tag{2.5}$$

Aus diesen kann man mithilfe von

$$\theta = 2 \arccos\ s \tag{2.6}$$

$$\mathbf{w} = \begin{cases} \frac{\mathbf{x}}{|\mathbf{x}|}, & \theta \neq 0 \\ 0, & sonst. \end{cases} \tag{2.7}$$

die Rotation um einen Winkel θ und eine Rotationsachse $\mathbf{w}$ darstellen [14].

Angle Axis Ein ähnliches Konzept verfolgen die Angle Axis nach Olinde Rodriguez. Bei ihnen wird eine Rotation im dreidimensionalen Raum ebenfalls als eine Rotation um eine bestimmte Achse dargestellt. Hierbei macht man sich zunutze, dass sich jede Rotation im dreidimensionalen Raum als eine einzelne Rotation um eine entsprechende Achse darstellen lässt.

$$\theta = \theta\hat{e}, \quad \hat{e} \in \mathbb{R}^3, \ \hat{e}^T * \hat{e} = 1 \tag{2.8}$$

Eine entsprechende Rotation besteht dabei nach [1] aus dem Produkt von Rotationsachse $\hat{e}$ mit dem entsprechenden Winkel θ. Der Vorteil hierbei ist, dass es sich um eine einzelne Rotation handelt, wodurch die Nachteile der gewöhnlichen Euler Rotation aufgehoben werden. Die Angle Axis können dabei nach Gleichung 2.9 aus den Quaternionen berechnet werden.

$$\omega = \frac{2 * \arccos(\sqrt{1 - |\mathbf{s}|^2})}{|\mathbf{s}|} * \mathbf{s} \tag{2.9}$$

Im Folgenden werden die konkreten Transformationen zwischen den einzelnen Koordinatensystemen vorgestellt.

ECI2ECEF Bei der Transformation vom inertialen zum erdfixierten Koordinatensystem handelt es sich um eine einfache Rotation, da beide Koordinatensysteme denselben Ursprung haben. Grundsätzlich handelt es sich dabei nahezu um eine Rotation um die z-Achse der Koordinatensysteme. Da die Erde jedoch nicht exakt um diese Achse dreht, muss es einen gewissen Korrekturterm geben. Dieser lässt sich mathematisch beschreiben, wurde hier jedoch zugunsten der Übersichtlichkeit nicht angewendet. Stattdessen wird die Rotation als eine initiale Rotation sowie eine reine Rotation um die z-Achse dargestellt. Diese initiale Rotation ist die vollständige Rotation zu einem bestimmten Zeitpunkt. Mithilfe der Zeitdifferenz seit dieser Referenzrotation und der Winkelgeschwindigkeit der Erde $\omega_{ie} = 7,292115 * 10^{-5} rad * s^{-1}$ lässt sich nun die aktuelle Rotation zwischen inertialem und erdfestem Koordinatensystem bestimmen zu

$$R_{i2e} = \begin{bmatrix} \cos(\omega_{ie}dt) & -\sin(\omega_{ie}dt) & 0 \\ \sin(\omega_{ie}dt) & \cos(\omega_{ie}dt) & 0 \\ 0 & 0 & 1 \end{bmatrix} * R_0. \tag{2.10}$$

ECEF2LLA Bei der Transformation zwischen ECEF und LLA Koordinatensystem handelt es sich um die Transformation von einem kartesischen Koordinatensystem in ein elliptisches. [18] Hierbei muss zunächst der Längengrad berechnet werden, nach

$$\lambda = \arctan\left(\frac{X_y}{X_x}\right). \tag{2.11}$$

Die Berechnung des Breitengrades gestaltet sich ungleich komplexer, da hier die Erde in Form eines Ellipsoiden angenommen wird. Hierbei wird zunächst eine initiale, reduzierte Breite β sowie die geodätische Breite ϑ nach

$$\beta = \arctan\left(\frac{X_z}{(1-f)s}\right) \tag{2.12}$$

$$\vartheta = \arctan\left(\frac{X_z + \frac{e^2(1-f)}{1-e^2}R(\sin\beta)^3}{s - e^2R(\cos\beta)^3}\right) \tag{2.13}$$

berechnet, wobei f die Abflachung der Erde ist, und für die Parameter s und e^2 gilt:

$$s = \sqrt{X_x^2 + X_y^2}$$
$$e^2 = 1 - (1-f^2)$$

Nach der initialen Berechnung wird β mithilfe der geodätischen Breite korrigiert.

$$\beta = \arctan\left(\frac{(1-f)\sin\vartheta}{\cos\vartheta}\right) \tag{2.14}$$

Die beiden Formeln 2.13 und 2.14 werden nacheinander iterativ ausgeführt bis ϑ konvergiert. Somit ist die geodätische Breite ϑ bekannt und es muss noch die Höhe über dem Ellipsoiden berechnet werden,

$$h = s\cos\vartheta + (X_z + e^2N\sin\vartheta)\sin\vartheta - N, \tag{2.15}$$

wobei N definiert ist als

$$N = \frac{R}{\sqrt{1 - e^2(sin\vartheta)^2}}.$$

NED2ECEF Bei der Transformation zwischen NED und ECEF Koordinatensystem handelt es sich sowohl um eine Translation als auch eine Rotation. Wobei sich beide aus der aktuellen Position im ECEF bzw. LLA Koordinatensystem ableiten lassen. Die Rotationsmatrix ist dabei

$$R_{n2e} = \begin{bmatrix} -\sin\phi\cos\lambda & -\sin\lambda & -\cos\phi\cos\lambda \\ -\sin\phi\sin\lambda & \cos\lambda & -\cos\phi\sin\lambda \\ \cos\phi & 0 & -\sin\phi \end{bmatrix}. \tag{2.16}$$

Weiterhin muss die Position noch translatorisch in das lokale NED Koordinatensystem überführt werden,

$$X_{NED} = R_{e2n}(X_e - X_{Ref}), \tag{2.17}$$

wobei X_{Ref} der Ursprungspunkt des NED Koordinatensystems im ECEF Koordinatensystem ist.

NED2BFF Bei der Transformation vom NED ins BFF Koordinatensystem handelt es sich für den Fall eines mitgeführten Koordinatensystems um eine reine Rotation um die bekannten Fluglagewinkel. Für den Fall, dass es sich um ein lokales Referenzkoordinatensystem handelt, kommt noch eine Translation hinzu. Die Namen der einzelnen Rotationen sind dabei Nicken Θ, Rollen Φ und Gieren Ψ. Diese Winkel und die dazugehörige Rotationsmatrix sind allgemein in der Luftfahrtnorm LN9300 [16] definiert zu

$$R_{n2b} = \begin{bmatrix} \cos\Theta\cos\Psi & \cos\Theta\sin\Psi & -\sin\Theta \\ \sin\Phi\sin\Theta\cos\Psi - \cos\Phi\sin\Psi & \sin\Phi sin\Theta\sin\Psi + \cos\Phi\cos\Psi & \sin\Psi\cos\Theta \\ \cos\Phi\sin\Theta\cos\Psi + \sin\Psi\sin\Psi & \cos\Phi\sin\Theta\sin\Psi - \sin\Phi\cos\Psi & \cos\Phi\cos\Theta \end{bmatrix}. \tag{2.18}$$

ECEF2BFF Die Transformation zwischen körperfestem und erdfixiertem Koordinatensystem setzt sich ebenfalls aus einer Rotation und einer Translation zusammen. Im Allgemeinen lässt sich mit der Transformation vom erdfeste ins körperfeste Koordinatensystem die Position und Lage eines Objektes beschreiben.

$$X_E = X_0 + R_{e2b}X_B \tag{2.19}$$

2.1.3 Zufallsgrößen

Für Berechnungen mit zufälligen Ergebnissen werden Zufallsvariablen benötigt, die einem Elementarergebnis $\omega \in \Omega$ eine reale Zahl zuordnen.

$$X : \Omega \to \mathbb{R} \tag{2.20}$$

$$\omega \to X(\omega) \tag{2.21}$$

Gleichzeitig gibt es häufig zwei Kennwerte, die die Zufallsvariable beschreiben. Dies sind der Erwartungswert, welcher mathematisch als $\mathbb{E}(X)$ definiert ist [22] und die Varianz $var(X)$. Für den Fall, dass es sich um eine mehrdimensionale Zufallsgröße handelt, werden die einzelnen Zufallsvariablen in eine Vektorform gebracht, den Zufallsvektor

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n. \end{bmatrix}. \tag{2.22}$$

2.1.4 Gauß Verteilung

In der Wahrscheinlichkeitstheorie gibt es viele verschiedene Dichtefunktionen um die Varianz bzw. Verteilung einer Zufallsvariablen zu beschreiben. Viele dieser Zufallsprozesse in der Natur lassen sich dabei als Normalverteilung oder auch Gaußsche Normalverteilung (GRV) beschreiben. So können zum Beispiel keine absoluten Messwerte generiert werden, sondern diese schwanken immer um den absoluten Messwert, da es immer zu

einem gewissen Fehler kommt. Die zwei Hauptparameter einer Gaußschen Normalverteilung sind der Erwartungswert μ und die Standardabweichung σ. Der Erwartungswert bildet dabei zum Beispiel bei einer Messung den absoluten Wert, während die Standardabweichung die Verteilung der Messwerte um diesen charakterisiert. Mithilfe dieser beiden Parameter lässt sich die entsprechende Dichtefunktion f der Normalverteilung bilden.

$$f(x)0\frac{1}{\sigma\sqrt{2*\pi}}e^{\frac{(x-\mu)^2}{2\sigma^2}} \quad \textit{für } \sigma > 0 \tag{2.23}$$

Eine weitere wichtige Eigenschaft der Gaußschen Normalverteilung ist, dass sich inner-

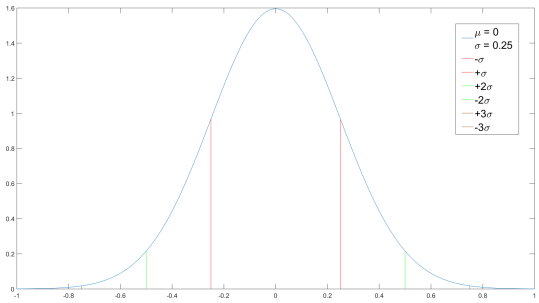


Abbildung 2: Gaußsche Normalverteilung
mit $\mu = 0$ und $\sigma = 0,25$

halb eines Gebietes von $\pm\sigma$ um den Erwartungswert $68,27\%$ der Zufallszahlen befinden. Weiterhin sind in einem Gebiet von $\pm 2\sigma$ $95,45\%$ der Zufallszahlen sowie in einer $\pm 3\sigma$-Umgebung $99,73\%$ der Zufallszahlen. Dementsprechend werden in vielen Anwendungen Ergebnisse außerhalb des 3σ-Bereiches ignoriert, da die entsprechende Wahrscheinlichkeit geringer als $0,003$ ist. Außerdem ist die Varianz der Gaußverteilung definiert zu $Var(X) = \sigma^2$. [17,21] Gleichzeitig gibt es zwei wichtige Kennwerte, die die Zufallsvariable beschreiben. Dies sind der Erwartungswert, welcher mathematisch als $\mathbb{E}(X)$ definiert ist [22] und die Varianz $var(X)$, die die Verteilung charakterisiert und definiert ist als

$$Var(X) = \mathbb{E}(X - \mathbb{E}(X))^2. \tag{2.24}$$

2.1.5 Kovarianz

Die Kovarianz oder auch Varianz-Kovarianz Matrix stellt einen Zusammenhang zwischen den jeweiligen Bestandteilen einer mehrdimensionalen Zufallsverteilung her. Grundsätzlich sind sie symmetrisch und positiv semi-definit. Auf der Hauptachse liegen dabei die Varianzen der Zufallsvariablen. Abseits dieser liegen die Kovarianzen oder auch Korrelationskoeffizienten. Die Kovarianzen liefern dabei ein Maß für den linearen statistischen Zusammenhang zwischen jeweils zwei Zufallsvariablen X und Y. Die allgemeine Defini-

tion der Kovarianz lautet dabei nach [4]

$$Cov(X,Y) = \mathbb{E}[(X - \mathbb{E}(X))(Y - \mathbb{E}(Y))] \tag{2.25}$$
$$Cov(X,Y) = \mathbb{E}(X * Y) - \mathbb{E}(X) * \mathbb{E}(Y).$$

Für den Fall, dass es sich bei X um eine n-dimensionle Zufallsgröße handelt, kann eine Kovarianz Matrix gebildet werden. Entsprechend wird für einen n-dimensionalen Zustandsvektor auch eine nxn-Matrix benötigt. In diesem Fall ist die Kovarianz Matrix nach [5]

$$C = \mathbb{E}[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T], \tag{2.26}$$

wobei die Elemente der Matrix σ_{ij} jeweils nach Gleichung 2.25 gebildet werden mit $X = X_i$ und $Y = X_j$. Somit stehen entlang der Hauptachse die entsprechenden Varianzen und abseits davon die Kovarianzen.

Für den Fall, dass die Zufallsvariable in ein anderes Koordianatensystem transformiert werden soll, muss die Kovarianz ebenfalls transformiert werden. Dies geschieht dabei nach dem folgenden Schema, wobei es sich bei R um die Transformationsmatrix handelt. [24]

$$\begin{aligned} C' &= \mathbb{E}(X'X'^T) - \mathbb{E}(X') * \mathbb{E}(X'^T) \\ &= \mathbb{E}(RXX^TR^T) - \mathbb{E}(RX)\mathbb{E}(X^TR^T) \\ &= R\mathbb{E}(XX^T)R^T - R\mathbb{E}(X)\mathbb{E}(X^T)R^T \\ &= R(\mathbb{E}(XX^T) - \mathbb{E}(X)\mathbb{E}(X^T))R^T \end{aligned} \tag{2.27}$$
$$C' = RCR^T \tag{2.28}$$

Außerdem ist C die Kovarianz vor der Transformation und C' die transformierte Kovarianz.

2.2 Zustandsschätzung

Die Zustandsschätzung ist die Abschätzung des aktuellen Systemzustands aus verschiedenen Messungen. Im Allgemeinen kann der gesuchte Systemzustand nicht direkt gemessen werden, sondern muss über andere beobachtbare Systemgrößen berechnet werden.

Beim Systemzustand handelt es sich um alle Informationen, die zu einer vollständigen Beschreibung sämtlicher Eigenschaften des Systems nötig sind.

$$\mathbf{x}_t = \mathbf{F}(\mathbf{x}_{t-1}, \mathbf{z}_t) + Rauschen \tag{2.29}$$

Dabei ist der aktuelle Systemzustand $\mathbf{x}_t$ eine Funktion $\mathbf{F}$ vom vorhergegangenen Systemzustand $\mathbf{x}_{t-1}$, sowie der Messgrößen $\mathbf{z}_t$. Da der Zustand und seine Änderung jedoch nicht absolut bekannt sind, kommt immer eine entsprechende Unsicherheit durch das Systemrauschen hinzu. Meist kann dieses Systemrauschen als Gaußsche Normalverteilung angenommen werden.
Dabei gibt es verschiedene Modelle, die im Folgenden vorgestellt werden sollen.

- Partikel Navigation / Monte-Carlo-Methoden
- Kalman Filter
- Unscented Kalman Filter

Sie unterscheiden sich sowohl in Rechenaufwand und Präzision, als auch in der Anfälligkeit für äußere Störungen. Hierzu sollen zunächst die Systemzustände vorgestellt werden. Anschließend werden dann einige allgemeine Zustandsschätzungs-Methoden vorgestellt. Im darauffolgenden Kapitel werden diese dann auf die Problematik eines kinematischen Körpers angewendet.

2.2.1 Systemzustände

Im folgenden Abschnitt sollen die beiden verwendeten Zustandssysteme vorgestellt werden. Ein Systemzustand ist dabei die Summe aller Informationen, die zu einer vollständigen Beschreibung des Systems notwendig sind. Hierbei gibt es zwei Arten von Zuständen. Zum einen die totalen Zustände (true State) X, zum anderen die Fehlerzustände (Error State) δX.

True State Beim totalen Zustand oder True State handelt es sich um den wahren Zustand des Systems. Für den Fall eines kinematischen Systems sind dies die Position, die Geschwindigkeit sowie die Lage.

$$X = \begin{bmatrix} \mathbf{x}_E & \mathbf{v}_E & \mathbf{q}_{E2B} \end{bmatrix}^T \in \mathbb{R}^{10} \tag{2.30}$$

Error State Der Fehlerzustand oder auch Error State ist eine Möglichkeit, den Fehler des wahren Systemzustandes darzustellen. Somit liegen im Falle eines kinematischen Systems die Zustände hier in Form eines Fehlers vor.

$$\delta X = \begin{bmatrix} \delta X & \delta v & R_e 2e & b_a & b_w & sfe_a & sfe_w & b_g \end{bmatrix}^T \in R^{24} \tag{2.31}$$

Weiterhin wird die Abweichung bzw Bias von Beschleunigung b_a, Drehrate b_w sowie Erdbeschleunigung b_g als Systemzustamd mit abgeschätzt. Außerdem wird der Skalierungsfehler (sfe) ebenfalls für den Beschleunigungssensor und Drehratensensor mit abgeschätzt.

2.2.2 Koppelnavigation

Bei der Koppelnavigation (en. Dead Reckoning) handelt es sich um eine bereits alte Möglichkeit, einen nicht direkt messbaren Systemzustand zu berechnen. Der Hauptnachteil ist jedoch, dass sich die Messfehler, die unweigerlich auftreten, mit der Zeit aufsummieren. Sie wurde schon früh in der Seefahrt eingesetzt um die aktuelle Position zu bestimmen. Hierbei wurde aus der Messung von Fahrtrichtung, Geschwindigkeit und vergangener Zeit die zurückgelegte Strecke berechnet. Dieses Prinzip wird ebenfalls in

der Luftfahrt eingesetzt. Es bildet die einfachste Möglichkeit, um unabhängig von visuellen Referenzpunkten die aktuelle Position zu bestimmen. Für ein kinematisches System bildet sich dabei

$$x_e(t + \Delta t) = x_e(t) + v_e(t)\Delta t + R_{eb}a_b\Delta t^2 + \gamma_e - coriolis_e \tag{2.32}$$

$$v_e(t + \Delta t) = v_e(t) + (R_{eb}a_b + \gamma_e - coriolis_e)\Delta t \tag{2.33}$$

$$R_{eb}(t + \Delta t) = R_{eb}(t)\omega_{ib}\Delta t - \Omega_{ie}R_{eb}(t)\Delta t \tag{2.34}$$

Hierbei sind $x_e(t)$ und $R_{eb}(t)$ die aktuellen Systemzustände in Form von Position und Lage. Weiterhin hat das System eine aktuelle Geschwindigkeit v_e, sowie die aktuellen Sensordaten Drehrate ω_{ib} und Beschleunigung a_b. Außerdem müssen die Drehrate der Erde Ω_{ie}, die Erdbeschleunigung γ_e und die Coriolis Kraft in Folge der Erdrotation und der Geschwindigkeit bekannt sein. Hiermit kann anschließend der Systemzustand für den nächsten Zeitschritt bestimmt werden. Da es sich jedoch nur um eine Abschätzung handelt, summieren sich die numerischen Fehler mit der Zeit auf, sodass der Fehler wächst.

2.2.3 Kalman Filter

Der Kalman- oder auch Kalman-Bucy-Filter ist ein mathematisches Verfahren, um den Systemzustand mithilfe von fehlerbehafteten Messungen iterativ zu bestimmen. Er wurde 1960 von R. E. Kalman entwickelt und findet heute in abgewandelter Form häufig Anwendung in verschiedensten Regelanlagen. Er besteht dabei aus einem Vorhersage- und einem Korrektur-Schritt. Diese werden nacheinander ausgeführt, wobei auch mehrere Vorhersageschritte nacheinander ausgeführt werden können, ohne einen Korrekturschritt. Der umgekehrte Fall ist jedoch nicht sinnvoll. Durch das abwechselnde Abschätzen und Korrigieren des Systemzustands können zusätzlich Informationen über das Systemrauschen erlangt werden, was die Zustandsschätzung verbessert. [13] Zunächst muss der Filter initialisiert werden. Wobei hier, ähnlich wie beispielsweise bei der Koppelnavigation, der ursprüngliche Systemzustand $\mathbf{x}_{k=0}$ bekannt sein muss.

$$\mathbf{x}_{k=0} = \mathbf{x}_0 \tag{2.35}$$

$$\mathbf{P}_{k=0} = \mathbf{P}_0 \tag{2.36}$$

Weiterhin muss eine anfängliche Kovarianz $\mathbf{P}_{k=0}$ bestimmt werden, diese beschreibt die Standardabweichung der initialen Systemzustände. Anschließend wird die Prädiktion des Systemzustandes durchgeführt.

$$\mathbf{x}_k^* = \mathbf{F}\ \mathbf{x}_{k-1} + \mathbf{B}_{k-1}\ \mathbf{u}_{k-1} \tag{2.37}$$

$$\mathbf{P}_k^* = \mathbf{F}\ \mathbf{P}_{k-1}\ \mathbf{F}^{\mathrm{T}} + \mathbf{Q}_{k-1} \tag{2.38}$$

Dabei ist $\mathbf{x}_k^*$ der vom vorherigen Systemzustand $\tilde{\mathbf{x}}_{k-1}$ geschätzte Systemzustand.
$\mathbf{F}_{k-1}$ ist die Systemmatrix, die den Systemzustand vom Zeitpunkt $k-1$ zum aktuellen

Systemzustand propagiert.
$\mathbf{B}_{k-1}\ \mathbf{u}_{k-1}$ stellt den Einfluss der Störungen auf den Systemzustand dar.
$\mathbf{P}_k^*$ ist die geschätzte Kovarianz des aktuellen Systemzustands.
Diese wird aufgrund von Prozessunsicherheiten Q verfälscht.

Wenn nun ein neuer Messwert $\mathbf{z}_k$ vorliegt, kann der vorhergesagte Systemzustand korrigiert werden und es folgt der Korrektur-Schritt.

$$\mathbf{K}_k = \mathbf{P}_{k-1}^*\ \mathbf{H}_k^{\mathrm{T}}\ (\mathbf{H}_k\ \mathbf{P}_{k-1}\ \mathbf{H}^{\mathrm{T}} + \mathbf{R})^{-1} \tag{2.39}$$

$$\mathbf{x}_k = \mathbf{x}_{k-1}^* + \mathbf{K}_k\ (\mathbf{z}_k - \mathbf{H}_k\ \mathbf{x}_{k-1}^*) \tag{2.40}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\hat{\mathbf{P}}_{k-1} \tag{2.41}$$

Hierbei ist $\mathbf{H}_k$ die Beobachtungsmatrix, welche die Messwerte mit dem entsprechenden Systemzustand verbindet. [17]
Aus der Messfunktion und den Kovarianzen wird die sogenannte Kalman-Gain-Matrix $\mathbf{K}_k$ gebildet. Bei der Berechnung der Kalman-Matrix kommt die Kovarianz des Messrauschens $\mathbf{R}$ hinzu. Mithilfe der Kalman-Matrix kann nun aus der Differenz zwischen Messung und tatsächlicher Messung der vorhergesagte Systemzustand korrigiert werden. Weiterhin kann mit der Kalman-Matrix auch die Systemkovarianz $\mathbf{P}_k$ korrigiert werden. Für den Fall, dass die Varianzen nicht zeitabhängig sind, können diese auch nur zu Beginn einmal abgeschätzt werden, und im weiteren Verlauf als konstant angenommen werden.

Grundsätzlich liefert der Kalman Filter nur für lineare Funktionen eine gute Lösung, weshalb es viele Erweiterungen ebenjenes gibt. Hierbei können sowohl die Systemfunktion als auch die Messfunktion nichtlinear sein. Als Beispiele sollten hier der Extended Kalman Filter oder der im Folgenden vorgestellte Unscented Kalman Filter angeführt werden. Der Extended Kalman Filter basiert darauf, dass die nichtlinearen Funktionen um den aktuellen Systemzustand linearisiert werden. Diese Linearisierung kann jedoch bei höheren Nichtlinearitäten schnell zu instabilem Verhalten führen. Ein weiterer Nachteil des Extended Kalman Filters ist, dass für jeden Prädiktionsschritt die Jacobi-Matrix berechnet werden muss. Da um einen neuen Punkt linearisiert werden muss und sich somit auch die Jacobi-Matrix ändert.

2.2.4 Unscented Kalman Filter

Da im Folgenden nichtlineare Funktionen genutzt werden sollen, kann der gewöhnliche Kalman Filter (2.2.3) nicht verwendet werden. Um die Problematik der Nichtlinearität zu umgehen, wird nun der Unscented Kalman Filter vorgestellt, der von Julier und Uhlmann entwickelt wurde. Er verwendet sogenannte Sigma Punkte, die mit Hilfe der Unscented Transformation bestimmt werden und dann mit dem gleichnamigen Filter in einen Fehlerzustand umgewandelt werden. [12] Der Name Unscented Kalman Filter wurde dabei willkürlich gewählt und hat keine direkte Bedeutung.

Unscented Transformation Um eine Zufallsverteilung mithilfe einer nicht linearen Funktion zu transformieren wurde die Unscented Transformation entwickelt. [11] Die Unscented Transformation bietet die Möglichkeit, aus einem Mittelwert und einer Kovarianz eine Menge an charakteristischen Punkten zu entwickeln. Diese sogenannten *sigma*-Punkte können in nichtlineare Funktionen eingesetzt werden. Mithilfe dieser Referenzpunkte können sowohl die tatsächlichen Werte als auch die entsprechenden Kovarianzen durch die nichtlinearen Funktionen geführt werden, ohne dass sie linearisiert werden müssen. Da hier im Gegensatz zu bekannten Monte-Carlo-Methoden keine zufälligen Punkte verwendet werden, sondern diese nach einem speziellen deterministischen Algorithmus ausgewählt werden. Der Vorteil ist hierbei, dass die sehr wenigen Punkte einen sehr hohen Informationsgehalt haben, was dem Rechenaufwand zugute kommt. Hierbei ist $\mathbf{x}$ ein n-dimensionaler Zufallsvektor, welche den Mittelwert $\overline{\mathbf{x}}$ und die Kovarianz $\mathbf{P}_{xx}$ besitzt. Diese lassen sich als eine Matrix X darstellen, die $2n$ $+1$ *sigma*-Vektoren beinhaltet. Dabei lassen sich die *sigma*-Vektoren

$$X_0 = \overline{\mathbf{x}} \tag{2.42}$$

$$X_i = \overline{\mathbf{x}} + (\sqrt{(n+\lambda) + P_{xx}})_i \quad i = 1, ..., n \tag{2.43}$$

$$X_i = \overline{\mathbf{x}} - (\sqrt{(n+\lambda) + P_{xx}})_{i-n} \quad i = n+1, ..., 2n \tag{2.44}$$

und die dazugehörigen Gewichte

$$W_0^{(m)} = \lambda/(n+\lambda) \tag{2.45}$$

$$W_0^{(c)} = \lambda/(n+\lambda) + (1 - \alpha^2 + \beta) \tag{2.46}$$

$$W_i^{(m)} = W_i^{(c)} = 1/(2(n+\lambda)) \quad i = 1, ..., 2n \tag{2.47}$$

bestimmen, wobei W_i^m das Gewicht für den entsprechenden Mittelwert und W_i^c das Gewicht für die entsprechende Kovarianz ist. Hierbei ist λ der Skalierungsfaktor und als

$$\lambda = \alpha^2(n + \kappa) - n \tag{2.48}$$

definiert, wobei α die Verteilung der *sigma*-Punkte angibt und im Allgemeinen auf $1e-3$ gesetzt wird. Der weitere Paramter κ wird in diesem Fall nicht benötigt und wird deshalb auf 0 gesetzt. Zuletzt beschreibt β die Art der Verteilung von $\mathbf{x}$. Da diese hier als Gaußsche Normalverteilung angenommen wird, gilt hier $\beta = 2$.
Nachdem die *sigma*-Punkte nun bekannt sind, können diese in die nichtlineare Funktion

$$Y_i = \mathbf{F}(X_i) \quad i = 1, .., 2n \tag{2.49}$$

eingesetzt werden. Aus den errechneten Werten für Y_i lassen sich mit den Gewichten die entsprechenden Mittelwerte und die Kovarianz von $\mathbf{y}$ bestimmen.

$$\overline{\mathbf{y}} \approx \sum_{i=0}^{2n} W_i^{(m)} Y_i \tag{2.50}$$

$$\mathbf{P}_{yy} \approx \sum_{i=0}^{2n} W_i^{(c)} (Y_i - \overline{\mathbf{y}})(Y_i - \overline{\mathbf{y}})^T \tag{2.51}$$

Der hier gewählte Ansatz approximiert bei Gaußverteilungen Momente bis zur Dritten Ordnung hinreichend genau. [11, 25]

Unscented Kalman Filter Grundsätzlich ist die Struktur des Unscented Kalman Filters der des normalen Kalman Filters sehr ähnlich, auch hier wird zunächst der aktuelle Systemzustand mithilfe der Systemgleichung f berechnet. Der Unterschied ist nur, dass hier im Gegensatz zum Kalman Filter die *sigma* Punkte in die Funktion eingesetzt werden. Diese werden nach Gleichung 2.44 berechnet und anschließend mit den entsprechenden Gewichten nach Gleichung 2.47 zur entsprechenden Kovarianz $\mathbf{P}$ umgewandelt.

$$X_i^* = f(X_{i-1}) \tag{2.52}$$

$$\overline{\mathbf{x}}^* = \sum_{i=0}^{2n} W_i^{(m)} X_i^* \tag{2.53}$$

$$\mathbf{P}^* = \sum_{i=0}^{2n} W_i^{(c)} (X_i^* - \overline{\mathbf{x}}^*)(X_i^* - \overline{\mathbf{x}}^*)^T \tag{2.54}$$

Hierbei sind wie zuvor die vorausgesagten Zustände mit einem Stern gekennzeichnet. Aus diesen geschätzten Systemzuständen können dann die entsprechenden Vorhersagen für die Messwerte getroffen werden. Dabei wird wieder wie beim Kalman Filter vorgegangen. Der Unscented Kalman Filter unterscheidet sich in diesem Fall nur darin, dass hier *sigma* in *gamma* Punkte transformiert werden. Diese sogenannten *gamma* Punkte entsprechen den Messwerten der transformierten *sigma* Punkte. Weiterhin wird ihr Mittelwert $\overline{\mathbf{z}}_i$ berechnet.

$$Z_i = \mathbf{H}(X_i^*) \tag{2.55}$$

$$\overline{\mathbf{z}}_i = \sum_{i=0}^{2n} W_i^{(m)} Z_i \tag{2.56}$$

Im Folgenden kann nun die Kovarianz der *gamma*-Punkte bestimmt werden. Außerdem wird die Kreuzkovarianz zwischen den transformierten *sigma*-Punkten sowie den *gamma*-Punkten berechnet. Mithilfe der Kovarianz und der Kreuzkovarianz kann nun die entsprechende Kalman-Matrix K berechnet werden.

$$\mathbf{P}_{zz} = \sum_{i=0}^{2n} W_i^{(c)} [Z_i - \overline{\mathbf{z}}_k][Z_i - \overline{\mathbf{z}}_k]^T \tag{2.57}$$

$$\mathbf{P}_{xz} = \sum_{i=0}^{2n} W_i^{(c)} [X_i - \overline{\mathbf{x}}][Z_i - \overline{\mathbf{z}}_k]^T \tag{2.58}$$

$$K = \mathbf{P}_{xz} \mathbf{P}_{zz}^{-1} \tag{2.59}$$

Mit dieser kann anschließend der Korrekturschritt durchgeführt und die neue Kovarianz der Zustandsschätzung berechnet werden.

$$x_k = \overline{\mathbf{y}}_k + K(z_k - \overline{\mathbf{z}}_k) \tag{2.60}$$

$$\mathbf{P}_k = \mathbf{P}^* - K\mathbf{P}K^T \tag{2.61}$$

Auch hier fällt wieder die starke Ähnlichkeit zum Kalman Filter auf. [12, 25]

2.3 Zustandsschätzung kinematischer Systeme

Im folgenden Kapitel wird die Zusammensetzung der verschiedenen Konzepte zur Fehlerzustandsschätzung eines kinematischen Systems mit den entsprechenden Freiheitsgraden vorgestellt.

Im Fall eines kinematischen Systems kann die Propagierung des Systemzustands über die inertiale Navigation durchgeführt werden. Dieser driftbehaftete Zustand wird dann mit deutlich verrringerter Frequenz mithilfe eines Fehlerzustandsschätzers korrigiert. Dies hat mehrere Vorteile. Zunächst kann der aktuelle Systemzustand über die deutlich weniger rechenintensive inertiale Navigation berechnet werden, wodurch weniger Rechenressourcen benötigt werden. Dadurch kann die Berechnung deutlich höherfrequenter durchgeführt werden und somit auch schnelle Systemänderungen noch mitverfolgen. Ein weiterer Vorteil ist, dass bei Ausfall der Korrektur immernoch eine verschlechterte Positionsbestimmung vorhanden ist. [23]

Weiterhin können durch die nachträgliche Korrektur des Systemzustandes auch nicht

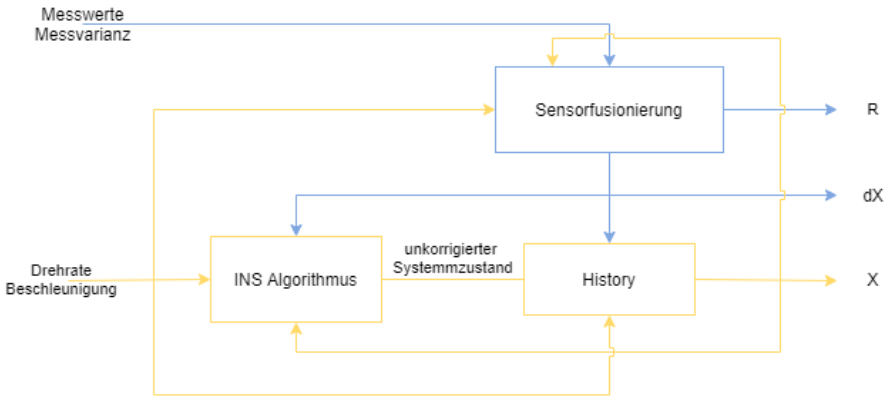


Abbildung 3: Überblick über einen Fehlerzustandsschätzer nach [1]

echtzeifähige Sensoren verwendet werden. Dies gilt im Speziellen für visuelle Sensoren. Diese Korrektur geschieht dabei innerhalb der sogenannten History. Ein wichtiger Punkt ist dabei, dass die Korrektur des Zustandes auch a-postiori geschehen kann, da die Messwerte der Trägheitsnavigation in der History gespeichert werden. Hierzu wird innerhalb der History, von der aktuellen Berechnungsposition des Fehlerzustandsschätzers aus, die inertiale Navigation mit den bereits vorliegenden Messwerten noch einmal durchgeführt,

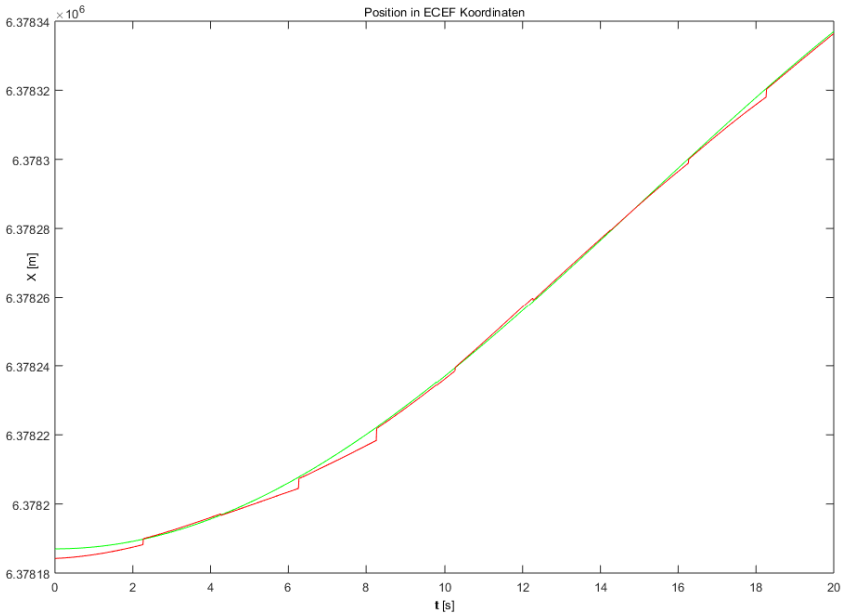


Abbildung 4: Prinzipbild der Fehlerkorrektur des Errorestimator

woduch die aktuelle Position in korrigierter Form vorliegt. [23]
Einen Überblick über die Zusammensetzung des Fehlerzustandsschätzers gibt dabei Abbildung 3. Hierbei ist der hochfrequente INS-Teil mit History in orange dargestellt und die Sensorfusion mit Fehlerzustandsschätzer in blau.
Die unterschiedlichen Frequenzen von Fehlerkorrektur und inertialer Navigation sind gut in dem exemplarischen Verlauf der Positionslösung in Abbildung 4 zu sehen. Hierbei ist der grüne Graph der wahre Systemzustand, während es sich bei der roten Linie um die geschätzte Positionslösung handelt. Hier ist der Sensordrift der inertialen Navigationsehr gut zu sehen, welcher in diesem Fall alle zwei Sekunden korrigiert wird.

Im Weiteren soll näher auf den Fehlerzustandsschätzer eingegangen werden. Da es sich bei dem hier beschriebenen Filter um einen Fehlerzustandsschätzer handelt, werden die Fehler der Zustände sowie deren Änderungsraten als Systemgrößen genutzt. Somit ergeben sich die Fehlergrößen zu

$$\delta X = \begin{bmatrix} \delta X & \delta v & R_e 2e & b_a & b_w & sfe_a & sfe_w & b_g \end{bmatrix}^T . \tag{2.62}$$

Die Abschätzung des eigentlichen Fehlers basiert dabei auf dem zuvor vorgestellten Unscented Kalman Filter [1].
Um die Funktionalität des UKF herzustellen müssen die Systemfunktion **f** sowie die Messfunktion **H** definiert werden. Dabei handelt es sich bei der Systemfunktion **F** um

die Fehlerfortpflanzung der Trägheitsnavigation.

$$d\delta X_e = \delta V_e \tag{2.63}$$

$$\begin{aligned} d\delta V_e = {} & (R_{e2e} - \mathtt{I}) * R_{b2e} * a_{bias} + R_{e2e} * R_{b2e} * a_{bias} \\ & + \gamma(X_e + \delta X_e) - \gamma(X_e) + \gamma_{bias} - 2\omega_{ie} \times (\delta V_e) \end{aligned} \tag{2.64}$$

$$dR_{e2e} = R_{e2e} * (R_{b2e} * W_{bias} * R_{b2e}^T + \Omega_{ie}) - \Omega_{ie} * R_{e2e} \tag{2.65}$$

Hierbei werden jeweils zunächst die Änderungsraten der Fehlerzustandsgrößen bestimmt. Anschließend müssen diese integriert werden, um die tatsächlichen Fehlerzustände zu erhalten.

$$\delta X_e = \delta X_e + d\delta X_e * dt \tag{2.66}$$

$$\delta v_e = \delta v_e + d\delta v_e * dt \tag{2.67}$$

$$R_{e2e} = R_{e2e} + dR_{e2e} * dt \tag{2.68}$$

$$\tag{2.69}$$

Weiterhin muss die entsprechende **h**-Funktion noch definiert werden. Diese ist allerdings stark von den verwendeten Sensoren abhängig, weshalb an dieser Stelle noch keine allgemeine Definition der **h**-Funktion gegeben werden kann.

2.4 Sensoren

Im Folgenden werden die verschiedenen genutzten Sensoren vorgestellt.

2.4.1 Globales Navigationssatellitensystem

Die absolute Position und Geschwindigkeit kann auf der Erde mithilfe eines der Globalen Navigationssatellitensysteme bestimmt werden. Hierbei wird von den Satelliten kontinuierlich ein Signal ausgesendet, das vom Empfänger entsprechend ausgewertet werden kann. Dabei gibt es vier funktionsfähige, unabhängige Globale Systeme,

- Global Positioning Network (GPS) der USA
- GLONAS der Russischen Förderation
- Galileo der Europäischen Union
- Beidou der Volksrepublik China

Weiterhin betreiben sowohl Japan als auch Indien ein regionales Navigationssatellitensystem. Diese dienen dazu, die Genauigkeit des amerikanischen GNSS Systems lokal zu verbessern [20].
Ein Satellitennavigationssystem basiert dabei auf drei Untergruppen, dem Weltraumsegment, welches die Satelliten des Systems beinhaltet, dem Kontrollsegment, welches für das Hochladen und Korrigieren der Signale zuständig ist, und dem Nutzersegment,

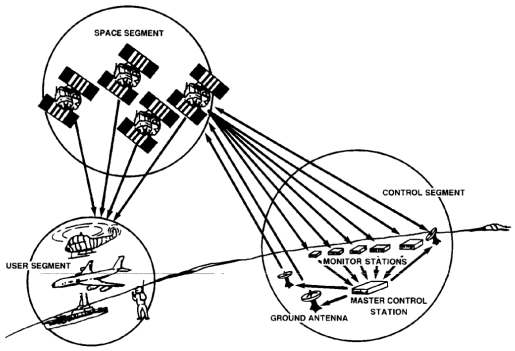


Abbildung 5: Überblick über die GPS Segmente [3]

welches die letztendliche Nutzung darstellt. Die Satelliten senden dabei kontinuierlich die Uhrzeit, sowie Informationen zur Fehlerkorrektur und den Almanach mit einer Liste aller Satelliten und ihrer Orbits. Da die Berechnung der Position auf der Laufzeitberechnung zwischen Satellit und Empfänger basiert, müsste die Uhrzeit des Empfängers bis auf wenige Nanosekunden genau bekannt sein. Da dies nicht gewährleistet sein kann, muss neben den drei translatorischen Freiheitsgraden die aktuelle Uhrzeit ebenfalls mit abgeschätzt werden, womit sich ein Gleichungssystem mit vier Unbekannten bildet. Deshalb werden zur Positionsbestimmung auch immer mindestens vier Satelliten im Sichtfeld benötigt. [3] Mithilfe dieser Signale wird somit die aktuelle Position im ECEF Koordinatensystem x_E bestimmt.

Geschwindigkeit Mithilfe der aktuellen Position sowie einer vergangenen Position und dem Zeitschritt zwischen diesen beiden, kann die Änderungsrate bestimmt werden, und somit die aktuelle Geschwindigkeit näherungsweise bestimmt werden.

$$v_E = \frac{x_e(t) - x_e(t - \Delta t)}{\Delta t} \tag{2.70}$$

Hierbei hängt die Präzision der geschätzten Geschwindigkeit sowohl von der Positionsgenauigkeit als auch von der Größe des Zeitschritts Δt ab. Eine weitere Möglichkeit zur Bestimmung der aktuellen Geschwindigkeit ist, die Dopplerverschiebung der GNSS Signale zu untersuchen. Da die Geschwindigkeit der Satelliten aus dem Almanach abgelesen werden kann, kann somit die Relativgeschwindigkeit zu den einzelnen Sensoren bestimmt werden, und damit die eigene Geschwindigkeit.

2.4.2 Sternenverfolger

Die absolute Orientierung gegenüber dem Inertialsystem R_i2b kann zum Beispiel mithilfe eines Sternenverfolgers gemessen werden. Hierbei wird mithilfe eines Bildsensors ein Bild des Nachthimmels erstellt, und durch die Lage der Sterne im Bild kann anschließend die Lage im System bestimmt werden. Ein Prinzipbild ist dabei in Abbildung 6 zu sehen. Dabei handelt es sich um einen modernen Sternenverfolger, bestehend aus

der Bildhardware und der Auswertungshardware. Die Bildhardware dient dazu, das Bild zu generieren und besteht aus Blendschutz, verschiedenen Linsen und dem Bildsensor. Demgegenüber steht die Auswertungshardware, die zur Auswertung der Bilder genutzt wird. Diese besteht aus einem Microcomputer und verschiedenen Datenbanken. Hierbei konnten frühe Sensoren nur einzelne Sterne verfolgen, womit nur unter Kenntnis der aktuellen Position eine genaue Lagebestimmung möglich war. Im Gegensatz dazu beinhalten heutige Sternenverfolger dank moderner Technik vollständige Sternenkataloge und können somit direkt die Lage bestimmen. Hierbei wird in zwei unterschiedlichen

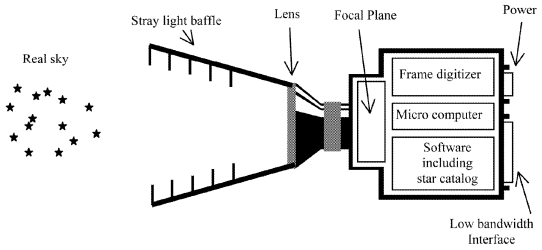


Abbildung 6: Schema eines Automatischen Sternenverfolgers nach [15]

Modi vorgegangen. Zunächst wird die initiale Lage bestimmt, danach muss diese weiter verfolgt werden. Die Bestimmung der initialen Lage kann dabei einige Sekunden in Anspruch nehmen, während die weitere Verfolgung der Lage im Bereich von einigen Hertz durchgeführt werden kann. Dabei treten verschiedene Fehler auf. Diese wurden jedoch im Rahmen dieser Arbeit nicht näher abgeschätzt und finden daher hier keine nähere Betrachtung. Allgemein werden Fehler des Sternensensors als Gaußverteilung um den tatsächlichen Messwert angenommen. Darüberhinaus können noch weitere Fehler bei der letztendlichen Messung durch verschiedene Lichtstörungen wie zum Beispiel Reflektionen hinzukommen. [15]

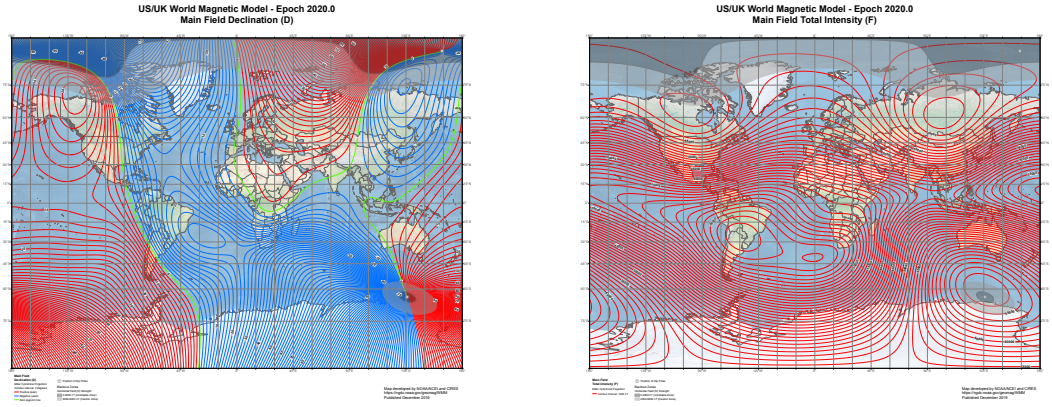
2.4.3 Magnetometer

Das Magnetometer bietet eine Möglichkeit das lokale Magnetfeld der Erde zu messen. Hierbei ist der integrierte Hall-Effekt-Sensor der häufigste, da er sehr robust gegenüber äußeren Kräften ist. Weiterhin hat er Vorteile bei Größe und Preis. [10] Bei der Messung mit ihm treten ebenfalls Fehler auf,

- Fehlausrichtung
- Störung durch Magnetische Stoffe im Umfeld
- Produktionsfeler des Sensors
- Rauschen .

Da aus dem gemessenen Magnetfeld eine Lage bestimmt werden soll, muss das Magnetfeld an der aktuellen Position bekannt sein. Da es sich beim Magnetfeld der Erde

nicht um einen einfachen Dipol handelt, gibt es verschiedene Modelle. Die bestimmenden Größen des Modells sind hierbei Inklination, Deklination sowie Intensität. Diese Parameter ändern sich allerdings mit der Zeit und sind ortsabhängig. Das meistgenutzte Modell ist dabei das World Magnetic Model des amerikanischen National Geospatial-Intelligence Agency (NGA) und bem britischen Defence Geographic Centre (DGC) herausgegeben. Dieses wird alle 5 Jahre aktualisiert und das aktuellste Modell ist von 2020. [19] Dabei



(a) Deklination des Erdmagnetfelds (b) Intensität des Erdmagentfelds

Abbildung 7: Überblick über das Erdmagnetfeldmodel 2020

werden die Parameter des Modells durch sich aktualisierende Koeffizientenlisten dargestellt. Mit ihrer Hilfe kann daraufhin ein lokales Magnetfeld im NED Koordinatensystem gebildet werden.

$$\mathbf{B}_N = f(x_e, t) \tag{2.71}$$

Aus diesem kann anschließend zum Beispiel die aktuelle Lage bestimmt werden.

2.4.4 Optische Messungen

Eine weitere Art der Sensoren sind optische Sensoren wie Kameras und Lidare. Aus diesen kann mithilfe der Algorithmen der visuellen Odometrie die relative Position bestimmt werden. Hierbei wird mithilfe von aufeinander folgenden Kamerabildern die Lage beziehungsweise Positionsänderung bestimmt. Klassischerweise wird dabei merkmalfokussiert vorgegangen, sodass verschiedene Merkmale von einem zum nächsten Bild verfolgt werden. Anschließend wird aus der Bewegung dieser Merkmale die entsprechende Positionsänderung bestimmt. [2]

relative Postion Bei der relativen Position handelt es sich dementsprechend um eine Positionsdifferenz zwischen zwei Zeitpunkten. Dabei wird zum aktuellen Zeitpunkt immer

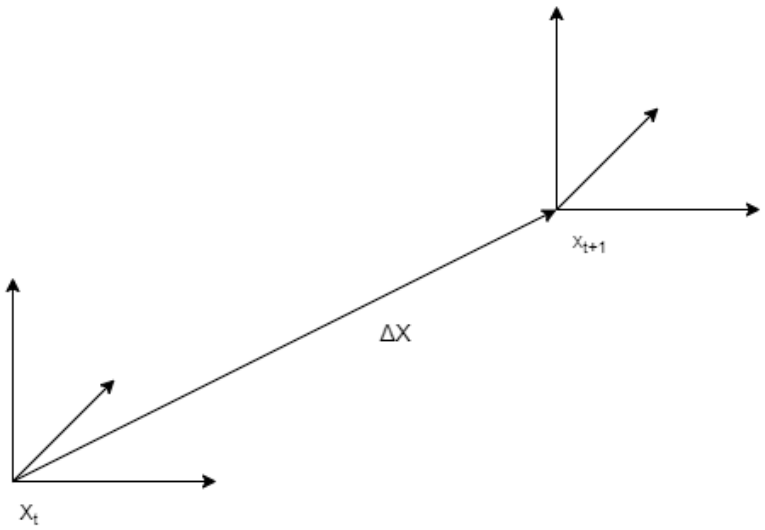


Abbildung 8: Relative Änderung der Position

die Strecke vom letzten Zeitpunkt zum aktuellen genutzt.

$$\Delta X_t = X_t - X_{t-1} \tag{2.72}$$

Da diese Art der Messwerte stark von der Lage der visuellen Sensoren abhängig ist, werden die Messwerte allgemein im sensorfesten Koordinatensystem angegeben. Da die Transformation vom sensorfesten in das körperfeste Koordinatensystem bekannt sein sollte, kann der Messwert zumeist im körperfesten Koordinatensystem angegeben werden.

3 Umgebung

In diesem Kapitel werden die allgemeinen Umgebungen der Bachelorarbeit beschrieben. Hierbei werden zunächst die genutzten Entwicklungsumgebungen vorgestellt. Anschließend wird das sogenannte Navigation Framework beschrieben, welches in dieser Arbeit erweitert wurde. Zuletzt wird dann die dazugehörige Testumgebung mit ihren Möglichkeiten und Limitationen thematisiert.

3.1 Entwicklungsumgebung

Im folgenden Abschnitt werden die beiden genutzten Entwicklungsumgebungen vorgestellt, die während dieser Bachelorarbeit genutzt wurden.

3.1.1 Codeblocks

Bei Codeblocks (Code::Blocks) handelt es sich um eine quelloffene, freie Entwicklungsumgebung, die für die Hochsprachen C und C++, sowie für FORTRAN entwickelt wurde. Hierbei können einzelne Projekte in eine Arbeitsumgebung (Workspace) eingebunden

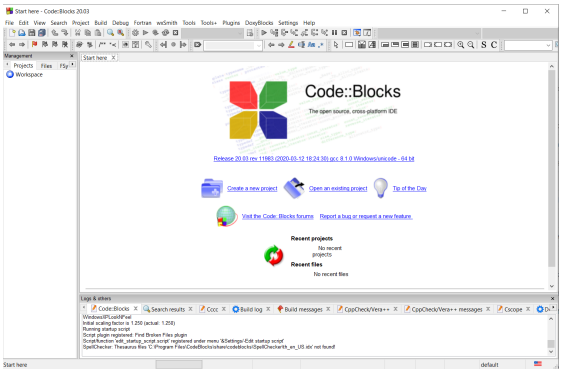


Abbildung 9: Überblick über Code::Blocks

werden. Daduch können einfache Abhängigkeiten zwischen den einzelnen Programmteilen oder auch externen Projekten erstellt werden. Weiterhin beherrscht Codeblocks die klassischen Features moderner Entwicklungsumgebungen, wie Syntax Highlighting und centext basierter, automatischer Wortvervollständigung. Als Compiler wurde während der Entwicklung der Visual C++ Compiler in der Version von 2008 verwendet.

3.1.2 Matlab

Bei Matlab handelt es sich um ein kommerziell vertriebenes Computeralgebra System, das vornehmlich zur Lösung verschiedener numerischer Probleme entwickelt wurde. Es besteht zum einen aus der Allgemeinen Algebra Umgebung, die speziell zur Lösung von numerischen Problemen genutzt werden kann. Zum anderen bildet es die Grundlage für Simulink, eine datenflussorientierte Simulationsumgebung.

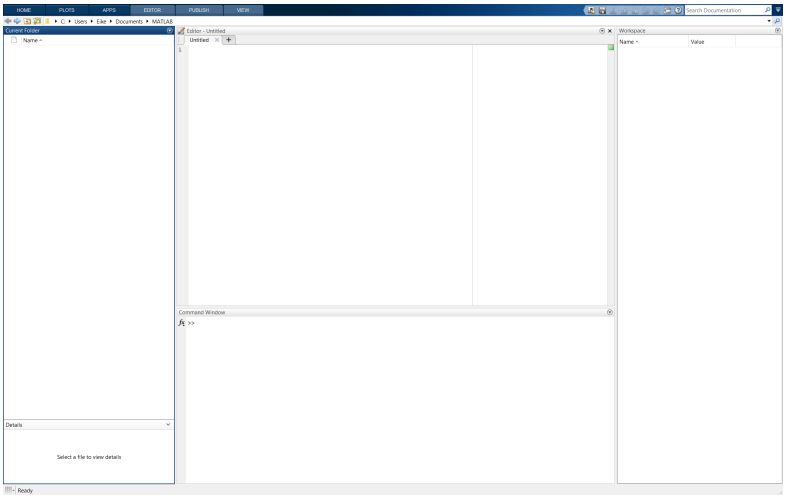


Abbildung 10: Überblick über Matlab

Simulink Bei Simulink handelt es sich um eine Simulationsumgebung, die mithilfe von graphischen Blöcken programmieren lässt. Diese Blöcke können dabei verschiedenste Funktionen beinhalten. Weiterhin lassen sich komplexere Modelle hierarchisch als Block in anderen Modellen einbinden. Dabei müssen die Ein- und Ausgangsparameter der Blöcke spezifiziert werden.

sFunction Die sogenannten sFunctions bilden eine Möglichkeit, externe Algorithmen in Simulink Simulationen einzubinden. Dies hat den Vorteil, dass verschiedene Programmteile als einzelne Module in die Simulink Simulation eingebaut werden können. Somit wird die Kommunikation zwischen den einzelnen Modulen stark vereinfacht. Ein weiterer Vorteil ist dabei die Flexibilität auch verschiedene andere Programmiersprachen mit einzubinden, wobei hier vor allem C++ sowie FORTRAN unterstützt werden. Die sFunctions bilden dabei die Schnittstelle zwischen dem Programm und der Simulation. Dafür werden in ihnen die verschiedenen Ein- und Ausgänge des Blocks und die Initialisierungswerte festgelegt.

Grafiken Matlab bietet verschiedene Möglichkeiten, um sowohl zwei- als auch dreidimensionale Grafiken zu erstellen. Die Grundfunktionen sind dabei bereits vorhanden. Diese wurden der später vorgestellten Testumgebung genutzt, um die verschiedenen Systemzustände direkt anzeigen und auswerten zu können.

3.2 ATONTasking

Beim ATONTasking System handelt es sich um ein datenflussorientiertes Framework, ähnlich zur Simulink Umgebung. In diesem System können verschiedene Einzelmodule eingebunden werden, die die entsprechenden Daten verarbeiten. Diese können dabei,

ähnlich wie die Blöcke in Simulink verschiedenste Funktionen und Komplexitäten annehmen. Eine Möglichkeit ist dabei auch, dass die Daten direkt in einem Logger abgespeichert werden können. Diese können bei der im Folgenden vorgestellten Playback Simulation genutzt werden.

3.3 Navigation Framework

Beim sogenannten Navigation Framework handelt es sich um ein Modul, das in das ATONTasking Framework eingebunden werden kann, das wiederum die Zustandsbestimmung zum Ziel hat. Es besteht aus einem hochfrequenten Inertialnavigationsteil und einem niedrigfrequenten Fehlerzustandsschätzer. Die Hauptfunktion ist die optische Navigation, welche z.B. auch bei fehlender GNSS Abdeckung genutzt werden kann. Es besteht aus vier Kernbausteinen, dem Sensorpool, der History sowie dem Zustandsschätzer, bestehend aus einem INS Algorithmus und einem Fehlerzustandsschätzer.

Sensorpool Der Sensorpool bildet eine der wichtigsten Klassen des Frameworks, in ihm werden sämtliche Sensordaten mit dem gleichen Zeitstempel abgespeichert. In ihm werden weiterhin die entsprechenden Messvektoren sowie ihre Kovarianzen gebildet.

History Die History Funktion beinhaltet die Funktionalität, den berechneten Systemzustand für einen gegebenen Zeitraum zu speichern. Außerdem wird für den Fall, dass ein neuer Fehlerzustand vorliegt, in ihr der aktuelle Systemzustand korrigiert.

Inertiale Navigation Die Inertiale Navigation beinhaltet die Implementierung eines Inertial Navigation Systems, zum Beispiel in Form eines Euler Integrators wie er in Kapitel 2.2.2 vorgestellt wurde. Dieser wird zum Propagieren des Systemzustands genutzt. Weiterhin wird hiermit im Falle eines in der Vergangenheit laufenden Fehlerzustandsschätzers dieser auf die aktuelle Position propagiert.

Fehlerzustandschätzer In diesem Modul befindet sich der Fehlerzustandsschätzer in Form eines Unscented Kalman Filters wie er in Kapitel 2.3 vorgestellt wurde. Dieser wird jeweils aufgerufen, falls neue Sensordaten vorliegen, um dann einen neuen Fehlerzustand zu generieren. Dabei werden jeweils die neuesten Sensordaten aus dem Sensorpool entnommen und verarbeitet.

Weiterhin kommen einige externe Programmteile zum Einsatz. Dabei handelt es sich zum einen um das Magnetfeldmodell, das die Parameter des WorldMagneticModel in einen lokalen Magnetfeldvektor umrechnet, und zum anderen um die sFunktions. Diese dienen, wie bereits zuvor beschrieben, dazu, die Zustandsschätzung mit der Simulink Umgebung zu verbinden.

3.4 Simulink Testumgebung

Bei dem so genannten NavigationFramework Simulink Testing Enviroment handelt es sich um eine Simulink Testumgebung, in der das NavigationFramework getestet werden kann. Grundsätzlich besteht es aus den folgenden Teilen:

- Generierung der Trajektorie
- Generierung des Zustandsvektors
- Simulation der Sensordaten
- Emulation optischer Sensoren inklusive Feature Tracking
- Berechnung der Systemzustände
- Erstellen der Auswertungsgraphen

Diese sind abgesehen von der Erstellung der Graphen in Abbildung 11 zu sehen. Für diese Testumgebung können verschiedenste Parameter festgelegt werden. Da die Gesamtheit der Parameter zu groß ist, wird hier nur auf die im weiteren Verlauf genutzten Parameter eingegangen. Eine wichtige Eigenschaft der Testumgebung ist, dass die Zustandsschätzung so lange unterbrochen wird bis die zu dem Zeitsstempel gehörigen Messwerte vorliegen. Dies führt dazu, dass die Simulation nicht in Echtzeit läuft.

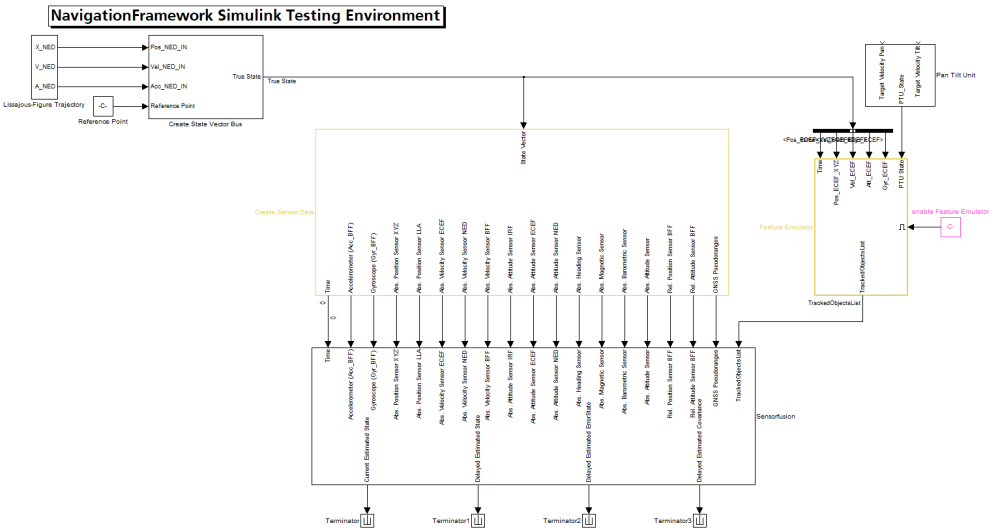


Abbildung 11: Überblick über das Navigation Testing Framework

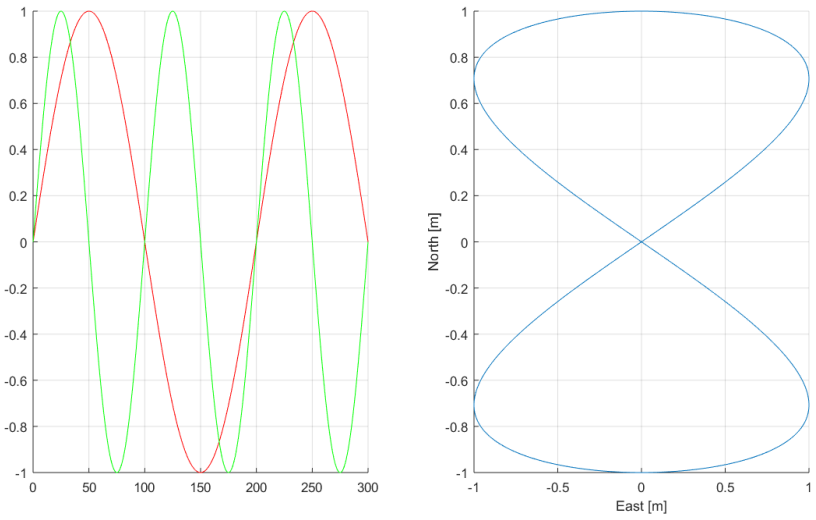


Abbildung 12: Beispiel einer Lissajous Figur mit Frequenzverhälnis 2:1

3.4.1 Generierung der Trajektorie

Zunächst werden mithilfe von Sinusfunktionen sogenannte Lissajous-Figuren gebildet, die sich jeweils durch eine Sinusfunktion in die jeweilige Koordinatenrichtung auszeichnen. Diese Figuren bilden die abgeflogene Trajektorie im NED Koordinatensystem.

$$X_N = X_N^0 + A_N * \sin(F_N * t + P_N) \tag{3.1}$$
$$X_E = X_E^0 + A_E * \sin(F_E * t + P_E) \tag{3.2}$$
$$X_D = X_D^0 + A_D * \sin(F_D * t + P_D) \tag{3.3}$$

Wobei für jede der Sinusfunktionen jeweils die Nulllage X_0, eine Amplitude A, eine Frequenz F und eine Phasenverschiebung P definiert werden kann. Da sich die Sinusfunktionen analytisch differenzieren lassen, können somit die Geschwindigkeit v_{NED} sowie die Beschleunigung a_{NED} im NED Koordinatensystem einfach berechnet werden.

3.4.2 Generierung des Zustandsvektors

Im nächsten Abschnitt werden aus der gegebenen Trajektorie sämtlche Positionen, Geschwindigkeiten, Beschleunigungen, Lagen sowie Drehraten berechnet. Darüberhinaus werden die entsprechenden Koordinatensystem-Transformationen durchgeführt, sodass diese Zustände in weitestgehend allen in Kapitel 2.1.1 vorgestellten Koordinatensystemen vorliegen, siehe Tabelle 1. Hierbei wird die Lage unter der Annahme eines Coordinated Turns berechnet. Ein Coordinated Turn ist dabei der Kurvenflug, bei dem der Beschleunigungsvektor in y-Richtung im Körperfesten Koordinatensystem genau Null ist. Aus der Lage wird anschließend die Drehrate diskret abgeleitet.

X_{NED}	X_{ECEF}	X_{BFF}
v_{NED}	v_{ECEF}	v_{BFF}
a_{NED}	a_{ECEF}	a_{BFF}
$\psi_{ECEF2BFF}$	$\psi_{NED2BFF}$	$\psi_{ECI2BFF}$
$\omega_{ECEF2BFF}$	$\omega_{NED2BFF}$	$\omega_{ECI2BFF}$

Tabelle 1: Überblick über die Systemzustände

3.4.3 Simulation der Sensordaten

In der Simulation der Sensordaten werden die Sensordaten aus den damit korrelierten Systemzuständen generiert. Hierbei liegen sämtliche Sensoren als einzelne Blöcke vor, in denen die berechneten Systemzustände um die definierten Fehler verfälscht werden. Wobei neben den aus der Luftfahrt bekannten Sensoren wie Beschleunigungsmesser, Gyro, Höhenmesser, Kompass und GNSS auch absolute Sensoren von Position, Geschwindigkeit, Beschleunigung, Lage und Drehrate implementiert sind. Weiterhin werden hier auch

Größe	Sensor
Position	GNSS
Geschwindigkeit	Pitot Rohre, Indirekt über GNSS, visuelle Odometrie
Beschleunigung	Accelerometer,
Lage	Magnetfeldsensor, Sternensensor, Visuell
Drehrate	Gyro, visuelle Odometrie

Tabelle 2: Übersicht der Systemgrößen und der zugehörigen Sensoren

die relativen Messwerte von Position und Lage simuliert.
Dabei können die folgenden Fehlertypen für die IMU definiert werden:

- Sensor Bias (Bias)
- Dead Zone
- Scale Factor Error (SFE)
- Messungenauigkeit bzw. Rauschen

Dabei werden die Sensoren nach folgendem Schema verfälscht

$$Z = SFE * Z_{true} + Bias + Rauschen. \tag{3.4}$$

Somit wird der wahre Wert zunächst um den Skalierungsfaktor verfälscht, anschließend werden Bias und Rauschen hinzuaddiert. Beim Rauschen handelt es sich um eine Gaußverteilung mit der entsprechenden Standardabweichung. Da das Framework bereits auf Flughardware getestet wurde, werden hier die durch die verbaute IMU festgelegten Parameter verwendet. [9] Für die weiteren Sensoren kann jeweils die Sensorvarianz und eine Ausgangsverzögerung definiert werden.

3.4.4 Berechnung der Systemzustände

Im folgenden Block ist das Navigation Framework als Funktion eingebunden. Somit wird hier aus den gegebenen Sensorwerten der Zustandsvektor geschätzt. Da das NavigationFramework die Systemzustände nur in einem Koordinatensystem ausgibt, werden anschließend noch die Zustände in den anderen Koordinatensystemen berechnet.

3.4.5 Erstellen der Auswertungsgraphen

Als letzte Teilfunktion werden die generierten Daten automatisch graphisch aufgearbeitet und angezeigt. Hierbei können nicht nur die Daten aus der Simulation genutzt werden. Es können auch die Daten, die mithilfe der Logging Funktionalität des ATON-Tasking Frameworks aufgezeichnet wurden, graphisch dargestellt werden. Hierbei ist es unerheblich, ob die Daten durch die im Folgenden vorgestellte Playback Simulation aufgezeichnet wurden oder aus einem realen Flugversuch stammen. Die Daten werden dabei zunächst eingelesen. Da aber nicht alle Werte in den Protokollen vorhanden sind, werden sie teilweise noch in die entsprechenden Koordinatensysteme transformiert. Hiermit können sämtliche in Tabelle 1 vorgestellten Systemgrößen dargestellt werden. Weiterhin können einige spezielle Größen wie die Fehlerzustände oder die einzelnen Messwerte dargestellt werden. Ein Beispiel für die Fehlergröße der Position ist in Abbildung 13 zu sehen. Hierbei ist $X - \hat{X}$ jeweils der tatsächliche Fehler, δX der geschätzte Fehler. Weiterhin ist die vom Unscented Kalman Filter abgeschätzte dreifache Standardabweichung als 3σ ebenfalls dargestellt, da der Zustand mit einer Wahrscheinlichkeit von $99,73\%$ innerhalb dieser Umgebung um den Messwert liegen sollte.

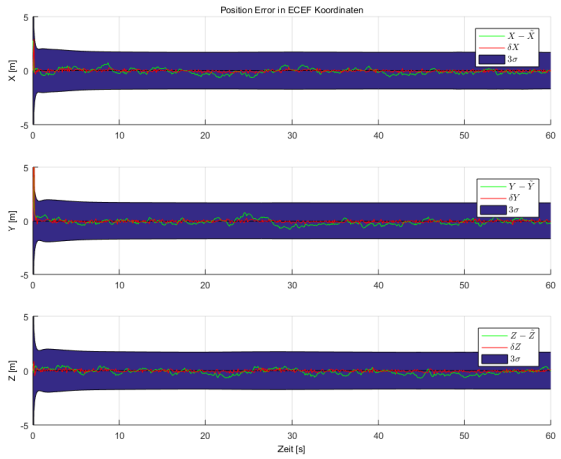


Abbildung 13: Beispiel des Positionsfehlers

3.5 Playback Simulation

Die Playback oder Prozessor-in-a-Loop Simulation bietet die Möglichkeit, neue Funktionen direkt mit echten Messdaten zu testen. Dies hat gegenüber dem Test Framework den Vorteil, dass die Daten wie im Flug nicht perfekt synchronisiert sind. Dabei werden die Messdaten aus vorherigen Flugversuchen, welche mit Zeitstempeln versehen sind, zu den korrespondierenden Zeitpunkten an das Framework gesendet. Dadurch können die in-flight Bedingungen sehr genau nachgebildet werden. Ein weiterer Vorteil ist, dass die Daten direkt vom logging-tool des Frameworks eingelesen werden können und es somit keiner Konvertierung bedarf. Da es sich um die Wiedergabe von tatsächlichen Zuständen handelt, kann die Simulation nur in Echtzeit laufen. Damit ist sie schneller als die Simulink Umgebung. Anschließend können die Daten ebenfalls in Matlab eingelesen werden und dieselben Graphen wie mit der Simulink Umgebung erstellt werden.

4 Implementierung

Im Folgenden werden die Implementierungen der verschiedenen Sensoren vorgestellt. Hierbei wird zunächst in den Abschnitten 4.1 und 4.2 das allgemeine Vorgehen beschrieben und anschließend die einzelnen Sensoren implementiert. Da der Grundaufbau des Unscented Kalman Filters bereits implementiert war, mussten hier nur die Generierung der entsprechenden Messwerte und die Abschätzung der dazugehörigen Messunsicherheiten vorgenommen werden. Außerdem musste die Generierung der geschätzten Messwerte innerhalb der h-Funktion für die einzelnen Sensoren implementiert werden. Dies wird im Folgenden für die einzelnen Sensoren durchgeführt.

4.1 Sensordaten

Zunächst muss für jeden der Sensoren die Generierung des Messvektors erweitert werden. Dabei wird erst der Messvektor nach der entsprechenden Bildungsvorschrift gebildet, und dieser dann in den Gesamtmessvektor übertragen. Die einzelnen Messwerte haben jeweils die Dimension n, wobei sich sämtliche der im Folgenden genutzten Sensoren aus jeweils drei unabhängigen Messgrößen zusammensetzen und daher dreidimensional sind. Somit bildet sich der Gesamtmessvektor zu

$$z = \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_a \end{bmatrix}, \tag{4.1}$$

wobei $z_0 \dots z_a$ die einzelnen Messvektoren sind.
Die Generierung der einzelnen Messvektoren soll im Folgenden am Beispiel der bereits implementierten Position im ECEF Koordinatensystem vorgestellt werden. In Anlehnung an diese wurden anschließend die neuen Sensoren implementiert.

$$z_a = z_{Position} = X_{Position} - X^*_{Position}. \tag{4.2}$$

Wobei z_a immer der jeweilige generierte Messwert im Fehlerzustandsraum ist. Weiterhin ist X der tatsächliche absolute Messwert und X^* der geschätzte absolute Systemzustand.

Außerdem müssen die Messungenauigkeiten der einzelnen Messungen ebenfalls in eine Gesamtmatrix übertragen werden. Dabei sind die einzelnen Kovarianzen R_a jeweils $n \times n$ Matrizen. Diese werden jeweils entlang der Hauptdiagonalen in die Gesamtmatrix übertragen.

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_0 & 0 & \dots & 0 \\ 0 & \mathbf{R}_1 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{R}_a \end{bmatrix} \tag{4.3}$$

Die Dimension der Gesamtunsicherheit ist von der Menge an aktiven Sensoren abhängig.

4.2 Messfunktion

Im Rahmen der Messfunktionen werden aus den, mithilfe der Systemfunktion geschätzten, Systemzuständen die dazugehörigen Messwerte berechnet. Hierbei handelt es sich entsprechend um eine Art umgekehrte Messfunktion. Weiterhin müssen die einzelnen in der h-Funktion geschätzten Sensorwerte in den Gesamtmessvektor z^* übertragen werden. Dies geschieht analog zur Generierung des Gesamtmessvektors im Sensorpool

$$Z_i^* = \begin{bmatrix} Z_{i,0}^* \\ Z_{i,1}^* \\ \vdots \\ Z_{i,a}^* \end{bmatrix}, \tag{4.4}$$

allerdings diesmal für sämtliche *gamma*-Punkte i. Ebenfalls am Beipiel der Fusion der Position soll hier einmal die Generierung des Messvektors in der h-Funktion vorgestellt werden.

$$Z_{i.Position} = X_{i.Position}^* \tag{4.5}$$

Hierbei fällt auf, dass der Systemzustand der Position dem damit korrelierenden Messwert entspricht, sodass hier keinerlei weitere Transformation benötigt wird.

4.3 Geschwindigkeit

Zunächst wird die Geschwindigkeit implementiert. Da diese im ECEF Koordinatensystem genutzt werden soll, kann hier ähnlich wie bei der bereits implementierten Position vorgegangen werden. Hierbei muss zunächst die Umwandlung vom Zustandsraum in den Fehlerzustandsraum der Messwerte vorgenommen werden, weiterhin muss die Kovarianzmatrix der Messunsicherheit in die Gesamtmessunsicherheitsmatrix übertragen werden. Anschließend muss die h-Funktion um eine Umwandlung der transformierten *sigma*-Punkte in die *gamma*-Punkte erweitert werden.

4.3.1 Sensordaten

Zunächst muss aus den entsprechenden Messwerten der gemessene Fehlerzustand generiert werden.

$$z_a = z_{geschwindigkeit} = X_{geschwindigkeit} - X_{geschwindigkeit}^*. \tag{4.6}$$

Hierbei wird vom gemessenen Systemzustand X der geschätzte Systemzustand X^* abgezogen. Diese Differenz bildet den gemessenen Fehlerzustand $z_{geschwindigkeit}$. Dieser dient im weiteren Verlauf als Messwert der Fehlerzustandsgröße $dX_{geschwindigkeit}$. Da allerdings ein Gesamtmessvektor gebildet werden muss, muss der Messvektor z_a noch in den Gesamtmessvektor z eingebracht werden. Weiterhin muss die Messunsicherheit R in die zusammengesetzte Gesamtunsicherheit übertragen werden.

$$\mathbf{R}_a = \mathbf{R}_{Geschwindigkeit} \tag{4.7}$$

Für den Fall, dass nur der Geschwindigkeitssensor aktiv ist, entspricht die Gesamtmatrix $\mathbf{R}$ der Matrix der Messunsicherheit $\mathbf{R}_{Geschwindigkeit}$.

4.3.2 Messfunktion

Weiterhin muss noch die Übertragung von transformierten *sigma*-Punkten in die *gamma*-Punkte implementiert werden.

$$Z_{i.Geschwindigkeit} = X^*_{i.Geschwindigkeit} \tag{4.8}$$

Bei der Generierung der *gamma*-Punkte $Z_{i.Geschwindigkeit}$ fällt auf, dass diese einem Systemzustand entsprechen und somit nicht weiter transformiert werden müssen.

Aus diesen kann dann die Kalmanmatrix berechnet werden und somit der Fehlerzustand korrigiert werden.

4.4 Lage im Inertialsystem

Bei der Lage im Inertialsystem wird die Lage relativ zum sich nicht mitdrehenden Koordinatensystem im Erdmittelpunkt (ECI) bestimmt. Da es sich um einen Lagefehler handelt, ist der Systemzustand `dX` definiert als R_{e2e} bzw selbiges als AngleAxis. Grundsätzlich muss dabei innerhalb der *h*-Funktion eine Transformation der *sigma*-Punkte vom Erdfesten in das Inertiale Koordinatensystem vorgenommen werden. Da diese Transformation jedoch vom eigentlichen Systemzustand, der Lage im ECEF Koordinatensystem, unabghängig ist, kann die entsprechende Transformation bereits im Sensorpool durchgeführt werden.

4.4.1 Sensordaten

Grundsätzlich lässt sich der Lagefehler nach dem folgenden Schema bestimmen.

$$Q^*_{b2e} = Q_{e2e} * Q_{b2e} \tag{4.9}$$

$$Q_{e2e} = Q^*_{b2e} * Q^T_{b2e} \tag{4.10}$$

Dabei ist Q_{b2e} der entsprechende Messwert, und Q^*_{b2e} der geschätzte Systemzustand. Da die Lage jedoch im ECI Koordinatensystem vorliegt, muss sie zunächst in das ECEF Koordinatensystem transformiert werden. Entsprechend muss der Messwert noch mit einer Rotation vom ECI in das ECEF Koordinatensystem multipliziert werden. Damit lässt sich der Fehlerzustand bestimmen zu

$$Q^*_{b2e} = Q_{e2i} * Q_{e2e} * Q_{b2e} \tag{4.11}$$

$$Q_{e2e} = Q^*_{b2e} * Q^T_{b2i} * Q_{e2i}. \tag{4.12}$$

Q^*_{b2e} ist dabei der geschätzte Systemzustand, und Q_{b2i} ist die Messung der Lage im Inertialsystem. Die Rotation zwischen Inertialsystem und Erdfestem Koordinatensystem erfolgt dabei nach 2.1.2. Da die rotatorischen Systemzustände alle als Rotationsachsen

nach Rodriguez angegeben sind, muss das Fehlerquaternion Q_{e2e} noch in eine Rotationsachse umgewandelt werden.

$$\omega_{e2e} = \frac{2 * \arccos(\sqrt{1 - |\mathbf{s}|^2})}{|\mathbf{s}|} * \mathbf{s} \tag{4.13}$$

Wobei auch hier das eigentliche Rotationsquaternion in eine Rotationsachse $\mathbf{s}$ und einen Winkel w aufgeteilt werden kann. Nach dieser Umwandlung muss dieser Vektor noch in den Gesamtmessvektor übertragen werden.

$$z_a = z_{Lage} = \omega_{e2e} \tag{4.14}$$

Da die Kovarianz der Messung auch im ECI Koordinatensystem vorliegt, muss diese ebenfalls noch ins ECEF Koordinatensystem transformiert werden. Dies geschieht wie bereits vorgestellt nach

$$\mathbf{R}_a = \mathbf{R}_{ECEF} = R_{i2e} * \mathbf{R}_{ECI} * R_{i2e}^T. \tag{4.15}$$

Wobei $\mathbf{R}$ hier die Kovarianzen in den entsprechenden Koordinatensystemen und R die Rotationen zwischen den Koordinatensystemen angibt. Anschließend wird die Kovarianz ähnlich der Geschwindigkeit an der entsprechenden Stelle in die Gesamtkovarianzmatrix geschrieben.

4.4.2 Messfunktion

Da die Messgröße jetzt der Lage im ECEF Koordinatensystem entspricht, muss diese innerhalb der h-Funktion nicht mehr weiter transformiert werden, sondern es kann direkt die Systemgröße genutzt werden. Somit bildet sich die h-Funktion zu

$$Z_{i.Lage_{e2b}} = X^*_{i.Lage_{e2b}}. \tag{4.16}$$

4.5 Magnetfeld

Im Folgenden wird die Implementierung des Magnetfeldsensors erklärt. Da es sich hierbei um keinen der Systemzustände handelt, hätte die Betrachtung als Fehlerzustand keinen Vorteil. Somit wurde dieser als absoluter Messwert übertragen und nicht als Fehlergröße.

4.5.1 Sensordaten

Der Übersicht halber wurden im Folgenden aber die selben Bezeichnungen verwendet.

$$z_a = z_{Magnetfeld} = X_{Magnetfeld} \tag{4.17}$$

Damit kann auch die Kovarianz der Messung direkt in die Gesamtmatrix übertragen werden

$$\mathbf{R}_a = R_{Magnetfeld}. \tag{4.18}$$

4.5.2 Messfunktion

Allerdings ist die Generierung der vorausgesagten Messwerte ungleich komplexer. Zunächst muss hierbei aus der aktuellen Position und Uhrzeit der lokale Magnetfeldvektor im NED Koordinatensystem bestimmt werden. Dies kann unter anderem mithilfe des in Kapitel 2.4.3 vorgestellten World Magnetic Model geschehen. Eine andere Alternative für erste Tests ist, zunächst ein konstantes Magnetfeld anzunehmen. Dies hat den Vorteil, dass der Einfluss der Position auf die Lagebestimmung nicht mehr vorhanden ist. Allerdings kann dieses nur für lokale Tests genutzt werden. Im Folgenden soll daher der lokale Magnetfeldvektor an der aktuellen Position bestimmt werden. Da die lokale Änderungsrate des Magnetfeldes jedoch sehr gering ist, kann das lokale Magnetfeld mit der geschätzten Position bestimmt werden. Somit muss der Magnetfeldvektor nicht für jeden *sigma*-Punkt einzeln berechnet werden. Weiterhin hat dies den Vorteil, dass die Magnetfeldbestimmung unabhängig vom Positionsfehler ist. Der so generierte Magnetfeldvektor muss anschließend in das BFF Koordinatensystem transferiert werden. Hierzu muss zunächst die aktuelle Rotation zwischen BFF und NED Koordinatensystem bestimmt werden.

$$R_{i,n2b} = X_{i,Rotation}^{-1} * R_{n2e} \tag{4.19}$$

Bei $X_{i,Rotation}$ handelt es sich dabei um den korrigierten Zustand. Dabei wird jeweils der unkorrigierte TrueState mit den entsprechenden *sigma*-Punkten korrigiert. Die Rotation von NED ins ECEF Koordinatensystem wird dabei ebenfalls aus der geschätzten Position ohne *sigma*-Punkte berechnet. Daraus werden im Anschluss die entsprechenden *gamma*-Punkte generiert.

$$Z_{i,Magnetfeld}^{*} = R_{i,n2b} * \mathbf{B}(X_{Position}^{*}) \tag{4.20}$$

4.6 Relative Position

Im Folgenden soll die Fusionierung der relativen Position implementiert werden. Da es sich auch hier um keinen der Systemzustände handelt, wird auch dieser nicht in den Fehlerzustand übertragen, sondern in Form des tatsächlichen Zustandes übertragen. Somit muss auch hier zunächst der Messwert in den Gesamtmessvektor übertragen werden, sowie die Kovarianz in die Gesamtmessunsicherheitsmatrix. Anschießend folgt die Transformation innerhalb der h-Funktion, bei welcher aus jeweils zwei Positionen mit bestimmtem zeitlichen Abstand die relative Position berechnet wird.

4.6.1 Sensordaten

Zunächst muss hier wie bei den vorherigen Sensoren ebenfalls der Messvektor in den Gesamtvektor übertragen werden.

$$z_a = z_{RelativePosition} = X_{RelativePosition} \tag{4.21}$$

Darüberhinaus muss die Kovarianz in die Gesamtmessunsicherheitsmatrix übertragen werden.

$$\mathbf{R}_a = \mathbf{R}_{RelativePosition} \tag{4.22}$$

4.6.2 Messfunktion

In der Messfunktion muss aus den geschätzten Fehlerzuständen ein geschätzter Messvektor generiert werden. Dies geschieht ebenfalls für sämtliche *gamma*-Punkte, ist hier allerdings exemplarisch nur einmal dargestellt. Hierbei wird zunächst die relative Position im ECEF Koordinatensystem gebildet und anschließend in das körperfeste Koordinatensystem transformiert. Die relative Position setzt sich hierbei aus der Summe aus geschätztem Zustand und Fehlerzustand zusammen, von welcher die Position zum Zeitpunkt $t - \Delta t$ abgezogen wird.

$$Z^*_{RelativePosition} = R^T_{e2b}(t - \Delta t) * (X^*_{Position}(t) - X_{Position}(t - \Delta t)) \tag{4.23}$$

Da es sich um den Positionsfortschritt von der ersten Position aus handelt, muss auch die Lage der ersten Position zur Transformation verwendet werden. Dies hat weiterhin den Vorteil, dass die relative Position vom aktuellen Fehlerzustand der Lage unabhängig ist.

5 Auswertung

Im folgenden Kapitel wird zuerst das Vorgehen beim Testen der verschiedenen Implementierungen vorgestellt und anschließend durchgeführt. Dabei wird zunächst die Funktionalität der einzelnen Sensoren überprüft. Im Anschluss werden diese dann im Verbund untersucht.

5.1 Testmethodik

Die Überprüfung der einzelnen Teilfunktionen wird zunächst mit dem Navigation Testing Framework aus Kapitel 3.4 vorgenommen. Dabei muss als erstes eine Flugtrajektorie gewählt werden, anhand derer die einzelnen Versuche durchgeführt werden, um eine möglichst gute Vergleichsbasis zu erreichen. Da es sich bei der Flugtrajektorie um eine Lissajous-Figur handelt, können die in Kapitel 3.4.1 vorgestellten Parameter frei gewählt werden. Hierbei muss darauf geachtet werden, dass diese auch in einem realistischen Flugszenario erfliegbar ist, und es sich somit um eine geschlossene Figur handelt. Es werden die Parameter aus Tabelle 3 genutzt. Weiterhin wird der Schnittpunkt von

	N	E	D
x_0	0	0	-150
A	1000 m	1000 m	-100 m
F	3	4	1
P	0	0	1.5 π

Tabelle 3: Parameter der Testtrajektorie

Nullmeridian und Äquator als Referenzposition im geographischen Koordinatensystem gewählt. Dies hat den Vorteil, dass sich für das ECEF Koordinatensystem ebenfalls Null auf zwei der drei Achsen ergibt. Somit sind die Werte auf diesen Achsen vergleichsweise klein. Außerdem kann das Verhalten der Lösung bei Vorzeichenwechseln beobachtet werden. Es wird eine Starthöhe von 250 Metern genutzt. Folglich bildet sich die Trajektorie aus Abbildung 14. Allerdings wird diese Position nicht direkt übergeben, sondern wird durch initiale Varianzen verfälscht. Dies sorgt zum einen dafür, dass bereits zu Beginn eine gewisse Abweichung der Position vorliegt, zum anderen wird diese ebenfalls als initiale Standardabweichung des Kalman Filters verwendet. Mithilfe der Flugtrajektorie wird im Folgenden eine Grundlage für die weiteren Versuche entwickelt.

Anschließend wird noch ein Gesamttest durchgeführt, bei dem sämtliche implementierte Sensoren zusammen getestet werden. Weiterhin wird die Fusionierung der Geschwindigkeit in der Playback Simulation getestet.

5.2 Baseline

Im foglenden Kapitel wird die Grundlage zur weiteren Einordnung der Messung entwickelt. Dazu werden zwei Tests durchgeführt. Zum einen wird die reine Inertiale Navi-

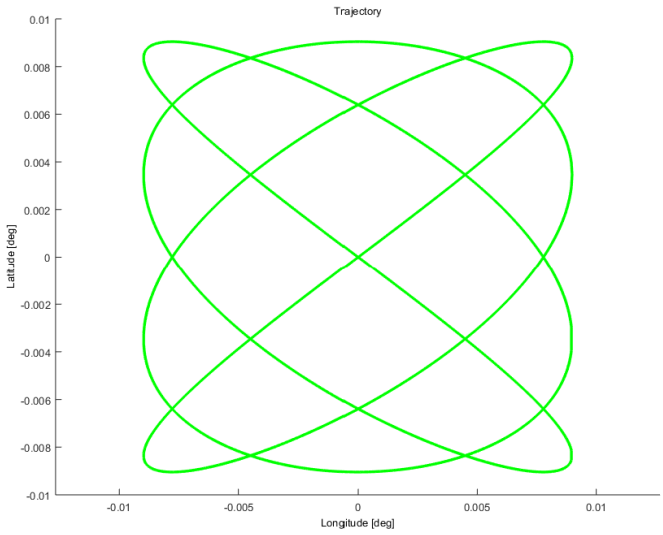


Abbildung 14: Die verwendete Lissajousfigur

gation (INS) getestet, zum anderen die bisherige Implementierung mit der Fusionierung der Position.

Inertiale Navigation Zunächst wird die Inertiale Navigation getestet, um ein Gefühl für die unkorrigierte Inertiale Navigation und ihre Abweichungen zu erhalten. Hierbei ist zu erwarten, dass die Zustandsschätzung mit der Zeit relativ stark degeneriert und nach einiger Zeit keine sinnvolle Lösung mehr liefert. Dabei werden die allgemeinen Parameter für die Flugtrajektorie genutzt. Weiterhin müssen einige Parameter für die Inertiale Messeinheit festgelegt werden. Die hierfür verwendeten Parameter sind aus vorherigen Projekten festgelegt, wie im Abschnitt 3.4 bereits vorgestellt. Die wichtigsten Parameter sind dabei in der folgenden Tabelle dargestellt.

	Beschleunigungsmesser	Gyro
Frequenz	100 Hz	100 Hz
Varianz	$9.62361e{-}05$ $[m^2/s^4]$	$2.350443e{-}07$ $[rad^2/s^2]$
DeadBand	0.0	0.0
Sättigung	∞	∞

Tabelle 4: Parameter der IMU

Damit bildet sich die in Graph 15 gezeigte Abweichung der Position. Wie zu erkennen ist, wächst diese sehr stark mit der Zeit, sodass nach zehn Minuten keine nutzbare Positionslösung mehr vorhanden ist. Im Gegensatz dazu degeneriert die Lagelösung nicht so stark, wie in Abbildung A.1 zu sehen ist.

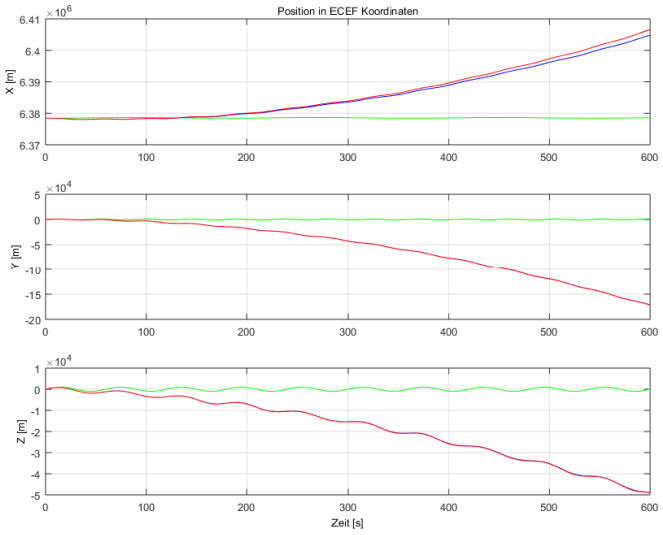


Abbildung 15: Abweichung der Position bei reiner Inertialnavigation

Fusionierung der Position Im folgenden Kapitel wird die Fusionierung der absoluten Position als Grundlage verwendet, um eine weitere Vergleichsbasis zu erhalten. Hierbei werden die erwartbaren Präzisionen eines GPS Empfängers genutzt, um die Messabweichung zu bestimmen. Es ist zu erwarten, dass der Positionsfehler sich stark verkleinert und gleichzeitig die berechnete Standardabweichung der Positionsbestimmung ebenfalls auf einen kleinen Wert sinkt.
Die Varianz beträgt somit

$$\sigma = \begin{bmatrix} 0,5 \\ 0,5 \\ 0,5 \end{bmatrix}^2 [m^2].$$

Die Messwerte der Position werden dabei mit 20 Hertz generiert, und entsprechend oft werden diese fusioniert. Wie in Abbildung 16 zu sehen ist, gibt es nur eine sehr geringe Abweichung in der Position. Weiterhin konvergiert die Gesamtunsicherheit der Positionslösung sehr schnell. Auffällig ist dabei, dass die geschätzte Standardabweichung der Wurzel der Standardmessabweichung entspricht. Wie in Grafik A.2 zu sehen ist, degeneriert der Lagefehler deutlich langsamer als bei reiner Inertialnavigation. Dies hängt damit zusammen, dass die Beschleunigung im BFF Koordinatensystem gemessen wird und somit aus Position und Beschleunigung indirekt die Lage bestimmt werden kann.

5.3 Geschwindigkeit

Im folgenden Abschnitt wird die Fusionierung der Geschwindigkeit getestet. Dabei ist zu erwarten, dass der Geschwindigkeitsfehler in einem im Rahmen der Sensorqualität

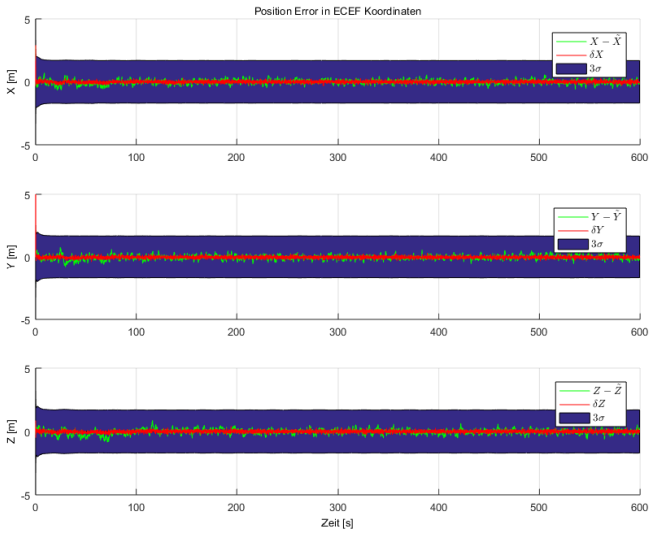


Abbildung 16: Positionsfehler mit Standardabweichung bei Positionsfusionierung

liegenden Bereich ist. Dadurch, dass die Geschwindigkeit nun bekannt ist, sollte sich der Positionsfehler um eine Größenordnung verbessern. Allerdings sollte er nicht unter die Anfangsabweichung fallen. Weiterhin sollte die geschätzte Standardabweichung auf etwa die Wurzel der Standardabweichung fallen.
Hierbei werden dieselben Grundparameter wie in Tabelle 3 genutzt. Dafür werden die folgenden Varianzen verwendet:

$$\sigma_{vel} = \begin{bmatrix} 0,5 \\ 0,5 \\ 3 \end{bmatrix}^2 \begin{bmatrix} m^2 \\ \overline{s} \end{bmatrix} \tag{5.1}$$

Außerdem werden die Sensordaten mit 10 Hz in die Fusionierung übertragen. Mithilfe dieser Parameter wird nun der zehnminütige Test durchgeführt. Wie in der Abbildung 17 zu sehen ist, konvergiert der 3σ-Bereich, sodass dieser jeweils der Wurzel der einzelnen Sensorvarianzen entspricht. Weiterhin verbleibt sowohl der geschätzte als auch der tasächliche Geschwindigkeitsfehler innerhalb der 3σ-Umgebung. Bei der Betrachtung des Positionsfehlers A.3 ist auffällig, dass die Position ebenfalls deutlich schwächer divergiert als zuvor. Dies ist allerdings zu erwarten, da die numerischen Fehler nun um eine Größenordnung gesunken sind. Somit muss nur noch einfach integriert werden um die Position zu erhalten. Allerdings kann dieser Fehler niemals die initiale Abweichung der Positionslösung korrigieren. Dies ist speziell beim Fehler in y-Richtung zu sehen, da dieser von Beginn an einen Fehler von etwa 15 Metern aufweist und dieser im Verlauf nur unwesentlich wächst. Der Lagefehler im ECEF Koordinatensystem verhält sich dabei sehr ähnlich wie bei der Positionskorrektur, wie in Abbildung A.4 zu sehen ist.

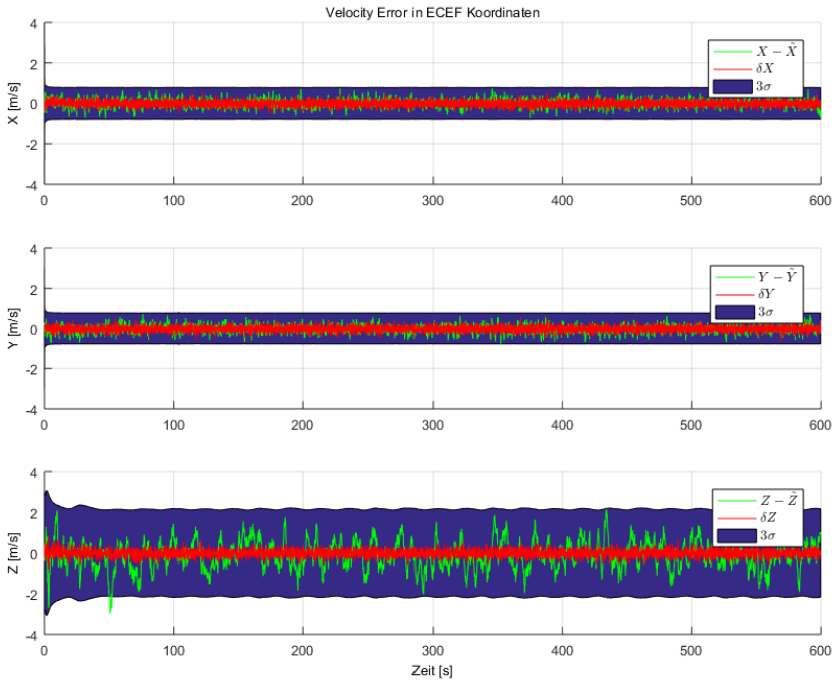


Abbildung 17: Geschwindigkeitsfehler

5.4 Lage

In diesem Kapitel wird die Lage im ECI Koordinatensystem untersucht. Hierbei soll ebenfalls die Funktionalität der Implementierung untersucht werden. Dies sollte somit zu einer deutlichen Verbesserung des Lagefehlers führen. Weiterhin sollte die Standardabweichung ebenfalls auf einen wie zu erwartenden geringen Wert, von etwa der Wurzel der Standardabweichung, sinken. Außerdem sollte sich die Positionslösung auch verbessern, da diese um eine Fehlergröße geringer wird.
Hierbei wird die folgende Varianz verwendet

$$\sigma = \begin{bmatrix} 0,1 \\ 0,1 \\ 0,1 \end{bmatrix}^2 [rad^2]. \tag{5.2}$$

Es wird eine Abtastfrequenz von 2 Hz gewählt. Aufgrund der Rotationsdefinition zwischen inertialem und erdfixiertem Koordinatensystem muss ebenfalls eine Anfangsrota-

tion R_0 gewählt werden.

$$R_0 = \begin{bmatrix} 0,58730600 & -0,809365 & -0,00100481 \\ 0,80936300 & 0,587307 & -0,00131845 \\ 0,00165724 & -3,89217e-05 & 0,999999 \end{bmatrix} \tag{5.3}$$

$$time_{R_0} = 1489395600 \approx Monday,\ 13.03.2017\ 09:00:00\ GMT + 0000 \tag{5.4}$$

Diese initiale Rotation sollte aber zumindest während der Versuche in Matlab keinerlei Einfluss haben, da sowohl die Generierung als auch die Auswertung auf derselben Bildungsvorschrift beruhen. Allerdings sollte sie für einen Realtest auf einen möglichst aktuellen Zeitpunkt gestellt werden. Mit diesen Parametern wird nun dieselbe Trajektorie wie bei den vorherigen Versuchen abgeflogen. Wie in Abbildung 18 zu sehen ist,

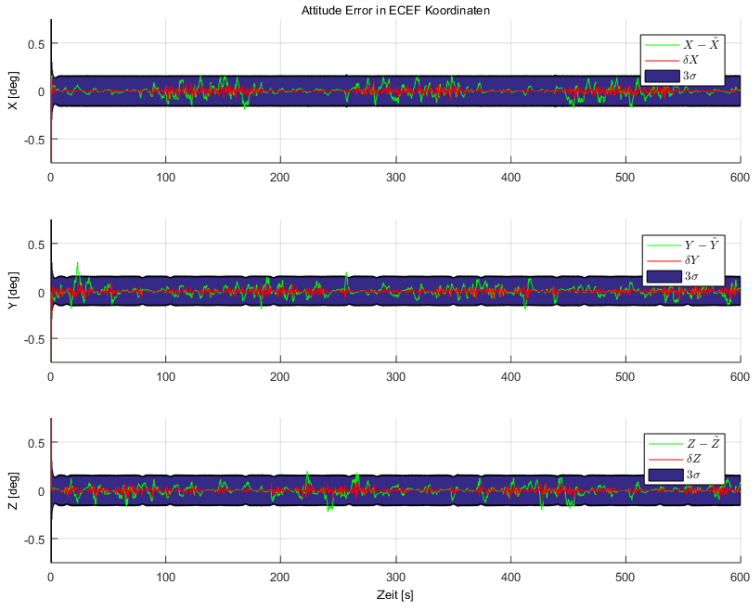


Abbildung 18: Lagefehler bei reiner Lagefusion

sorgt die Fusionierung der Lage für eine signifikante Verbesserung des Lagefehlers. Weiterhin ist zu sehen, dass die geschätzte Abweichung ebenfalls auf einen sehr geringen Wert konvergiert. Ebenfalls fällt bei der Betrachtung des Positionsfehlers auf, dass die Standardabweichung zwar wie erwartet nicht weniger stark, der tatsächliche Positionsfehler allerdings deutlich weniger stark divergiert. A.5 Dies ist darauf zurück zu führen, dass mit der Kenntnis der Lage eine der Einflussgrößen der Inertialnavigation bekannt ist. Und somit ein Parameter weniger mit der Zeit divergiert.

5.5 Magnetfeld

Im folgenden Kapitel wird die Fusionierung des Magnetfeldes durchgeführt. Hiebei soll, wie bereits in den vorangegangenen Kapiteln, die Funktionalität der Implementierung überprüft werden. Um dies zu zeigen, soll sich der Lagefehler signifikant verringern und nicht divergieren. Weiterhin soll die berechnete Standardabweichung ebenfalls konvergieren. Da die Messgröße jedoch nicht direkt der Zustandsgröße entspricht, kann die Standardabweichung nicht direkt aus der Messabweichung abgeschätzt werden.
Hierzu wird die Sensorvarianz auf

$$\sigma = \begin{bmatrix} 100 \\ 100 \\ 100 \end{bmatrix}^2 [nT^2] \tag{5.5}$$

festgelegt. Die Sensordaten werden mit einer Frequenz von 20 Hz generiert. Weiterhin wird das World Magnetic Modell in der aktuellsten Version von 2020 sowohl zur Sensorgenerierung als auch in der Fusionierung verwendet 7. Wie in Abbildung 19 zu sehen

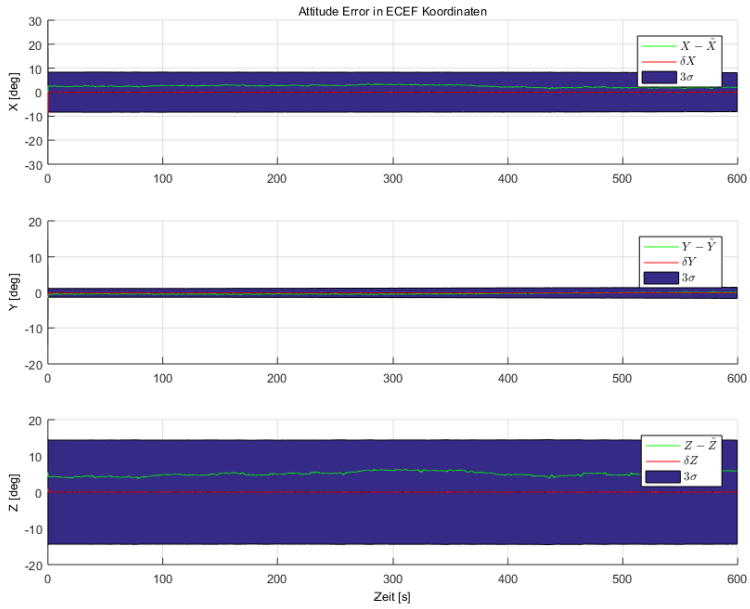


Abbildung 19: Lagefehler bei Fusion des Erdmagnetfelds

ist, sinkt hier ebenfalls die Abweichung der Lage wie erwartet. Dabei ist im Gegensatz zu den vorherigen Sensoren keine direkte Korrelation zwischen der eingestellten Messvarianz und der geschätzten Standardabweichung zu erkennen. Weiterhin verbleibt ein gewisser Offset in der Lage. Dieser verhält sich über die Laufzeit nahezu konstant, somit handelt es sich um einen systematischen Fehler. Weiterhin wird das Magnetfeld nur durch zwei Größen charakterisiert, nämlich der Richtung bzw. Deklination sowie

der Inklination, also die Neigung zur Waagerechten. Damit sind jedoch nur zwei der drei rotatorischen Freiheitsgrade festgelegt. Durch diese Unterbestimmtheit kommt es trotz Sensorfusionierung zu einer gewissen Abweichung in der Lagebestimmung. Die entsprechende Positionsbestimmung wird dabei, wie in A.6 im Anhang zu sehen ist, auch vergleichbar mit der Lage im ECI Koordinatensystem verbessert, was ebenfalls mit der Verknüpfung in der Systemfunktion bzw. innerhalb der Inertialnavigation zu erklären ist.

Da das Magnetfeld jedoch von der aktuellen Position abhängig ist, wurde im Folgenden die Fusionierung des Magnetfeldes zusätzlich noch einmal bei gleichzeitiger Positionsfusion durchgeführt. Dieser Test wurde durchgeführt um zu überprüfen, ob sich der systematische Fehler mit dem Positionsfehler erklären lässt. Hierzu wurden ebenfalls die bereits zuvor verwendeten Parameter verwendet. Hiermit bildet sich der in Abbildung 20 dargestellte Verlauf. Hierbei fällt auf, dass im Gegensatz zur reinen Fusionierung des

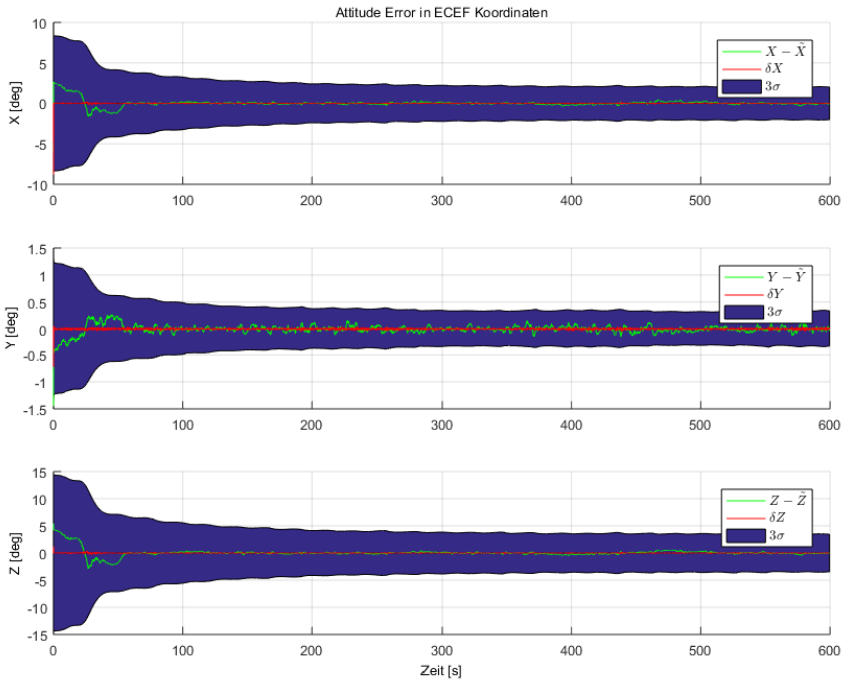


Abbildung 20: Fusionierung von Position und Magnetfeld

Magnetfeldes der systematische Fehler verschwindet. Stattdessen liegt die Abweichung nach einer Anfangsphase von etwa einer Minute bei einer Abweichung von unter einem Grad. Weiterhin sinkt die berechnete Standardabweichung ebenfalls deutlich. Weiterhin ist in Abbildung A.7 im Anhang zu erkennen, dass die Positionsabweichung sich ähnlich wie bei reiner Positionsfusion verhält.
Somit lässt sich der systematische Fehler mit der unbekannten Position erklären. Wei-

terhin lassen sich die Unterschiede zwischen den geschätzten Standardabweichungen der Lage wieder darauf zurückführen, dass das Magnetfeld nur durch zwei Parameter beschrieben wird.

5.6 Visuelle Position

Im folgenden Kapitel soll die relative Position untersucht werden. Ziel dabei ist es, die allgemeine Funktionalität zu überprüfen und zu verifizieren. Allgemein ist dabei zu erwarten, dass die Positionslösung sich verbessert und der Geschwindigkeitsfusion ähnelt. Dabei soll die Standardabweichung der Positionslösung nicht konvergieren, da die relativen Messungen die absolute Position nicht verbessern können, sondern nur die Divergenz abschwächen.
Hierbei werden die folgenden Parameter verwendet.

$$\sigma = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}^2 [m^2]$$
$$f = 5\text{Hz}$$
$$\Delta t = 0.5\text{s}$$

Bei einem ersten Test mit reinen relativen Messungen fällt auf, dass die Positionsbe-

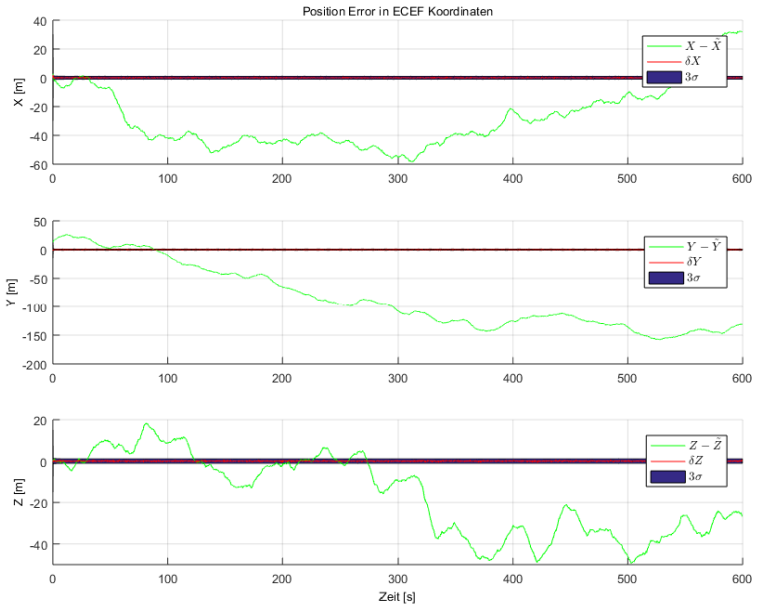


Abbildung 21: Fusionierung der relativen Position

stimmung deutlich stärker degeneriert als erwartet.

Dies liegt daran, dass die relative Position im körperfesten Koordinatensystem bestimmt wird. Somit kann diese nur bei Kenntnis der Lage zu einer deutlichen Verbesserung der Positionslösung beitragen. Aufgrund dieser Verknüpfung wird im Folgenden die Testtrajekorie noch einmal verwendet, allerdings diesmal mit der aktuellen Lage im ECI Koordiantensystem. Hierfür werden die im Kapitel 5.4 genutzten Parameter verwendet. Hiermit werden die folgenden Ergebnisse generiert.

Wie in Abbildung 21 zu erkennen ist, sorgt die kombinierte Fusion von relativer Position und Lage für die zu erwartende Verbesserung der Positionslösung. Diese entspricht dabei genau der Anfangsabweichung. Allerdings fällt auf, dass die Standardabweichung der Position nicht konstant ist, sondern auf einen sehr kleinen Wert konvergiert. Dieser liegt bei $3\sigma = \sqrt{0,5}$, was der festgelegten Messabweichung entspricht. Dies ist jedoch wie zuvor beschrieben falsch. Anstattdessen sollte die Standardabweichung der Positionslösung mit der Zeit divergieren, da keine absoluten Messwerte vorliegen.

5.7 Gesamttest

Im folgenden Kapitel wird nun ein Gesamttest durchgeführt, um auszuschließen, dass es zu unerwünschten Interaktionen zwischen den verschiedenen Sensoren kommt. Dieser wird zum einen im Testing Enviroment durchgeführt und zum anderen in der Playback Simulation.

Testing Environment Im folgenden Abschnitt werden sämtliche bisher installierte Sensoren zusammengeführt. Dabei ist zu erwarten, dass sämtliche Fehlergrößen auf einen sehr kleinen Wert fallen. Weiterhin sollten somit auch die entsprechenden geschätzten Standardabweichungen auf diesen Wert fallen.
Hierbei werden wie zuvor auch jeweils die Parameter der einzelnen Sensoren aus den vorangegangen Kapiteln verwendet. Wie bei den vorherigen Versuchen wird auch hier die zuvor vorgestellte Flugtrajektorie genutzt.

Wie in Abbildung 22 zu sehen ist, ist der relative Fehler bezogen auf die Gesamttrajektorie sehr klein. Die kleine Abweichung zu Beginn des Fluges lässt sich dabei auf die initiale Varianz zurückführen. Dies wird noch deutlicher, wenn man den direkten Positionsfehler betrachtet. Weiterhin ist ebenfalls zu erkennen, dass die geschätzte Standardabweichung auf den zu erwartenden Wert fällt. Selbiges gilt für die in A.11 dargestellte Geschwindigkeit wie ach die Lage. Bei beiden geht sowohl der Fehler als auch die Standardabweichung auf die erwartete Größe zurück.

Playback Simulation Im folgenden Abschnitt wird die Implementierung einer Art Realtest unterzogen. Hierzu wird das Navigation Framework in das ATONTasking Framework eingebunden, um dieses an aufgezeichneten Messdaten zu testen. Hierbei werden zunächst ebenfalls generierte Daten verwendet, diese wurden von Johann Dambeck von der TU München vorgeneriert und anschließend anhand dieser die Sensordaten generiert. [6] Hierbei konnten jedoch nicht sämtliche Sensoren generiert werden, sondern nur

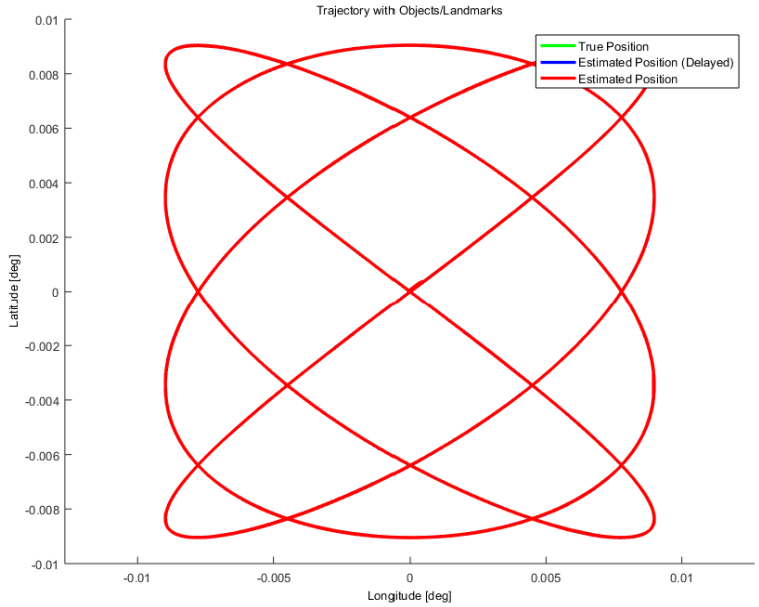
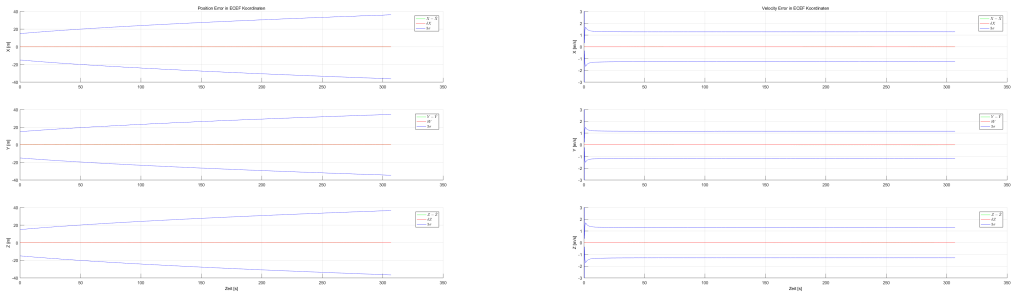


Abbildung 22: Trajektorie mit Abweichung

die folgenden

- Position im Erdfesten Koordinatensystem
- Geschwindigkeit im ECEF-Koordinatensystem
- Lage relativ zum NED-Koordinatensystem
- Höhe
- Heading

Somit kann zunächst nur die Geschwindigkeitsfusion sowie die bereits vorliegende Positionsfusion untersucht werden. Dementsprechend wird im Folgenden die Funktion der Implementierung der Geschwindigkeit gezeigt. Hierzu werden vergleichbare Parameter zu den zuvor gegebenen verwendet, um die bisher unverfälschten Messwerte mit einer Gaußverteilung zu verfälschen. Hierbei sollten vergleichbare Ergebnisse zum Test der Geschwindigkeit im Testing Framework entstehen. Somit sollte sowohl die Position im ECEF Koordinatensystem als auch die Geschwindigkeitsabschätzung konvergieren. Gleichzeitig sollte die Standardabweichung des Geschwindigkeitsfehlers auf einen geringen Wert konvergieren, während die Standardabweichung der Positionslösung langsam divergieren sollte, da diese nicht verbessert wird. Zeitlgeich wird sie allerdings Gleichzeitg um den Messfehler der Geschwindigkeit verfälscht. Außerdem sollte der Lagefehler ähnlich wie in Abschnitt 5.3 beschrieben ebenfalls weniger stark divergieren. Wie in Ab-



(a) Position (b) Geschwindigkeitsfehler

Abbildung 23: Fusionierung der Position im Playback Versuch

bildung 23a zu sehen ist, divergiert die Standardabweichung der Positionsbestimmung mit der Zeit von einem Startwert von etwa 15 Metern aus konstant mit der Zeit. Weiterhin bleibt der tatsächliche Positionsfehler in einem sehr kleinen Bereich. Gleichzeitig ist in Abbildung 23b zu sehen, dass der Fehler der Geschwindigkeitsabschätzung ebenfalls sehr gering ist, und die Standardabweichung auf einen Wert von etwa $1,3$ konvergiert. Dabei verhalten sich sowohl bei der Position als auch bei der Geschwindigkeit alle drei Dimensionen weitestgehend gleich.
Wie somit zu sehen ist, liefert die Fusionierung der Geschwindigkeit die zu erwartenden Werte. Hierbei liegt die Divergenz der Standardabweichung der Position im zu erwartenden Bereich. Weiterhin verhalten sich der Geschwindigkeitsfehler und seine Standardabweichung wie erwartet.

6 Fazit

In dieser Arbeit wurde die Zustandsschätzung mithilfe eines Fehlerzustandsschätzers und Inertialer Navigation erweitert. Der Fehlerzustandsschätzer basiert dabei auf einem Unscented Kalman Filter. Hierzu wurden zunächst die benötigten Grundlagen sowie die Programmierumgebungen vorgestellt. Zusätzlich wurden die bereits erstellten Programme vorgestellt, auf welchen die Arbeit aufbaut. Bei den Grundlagen wurde speziell auf die in der Arbeit implementierten Sensoren eingegangen, nämlich der Geschwindigkeit im erdfesten Koordinatensystem, der Lage in Relation zum Inertialsystem, ein Magnetfeldsensor und die Messung der relativen Position mithilfe von visueller Odometrie. Diese wurden im Folgenden einzeln implementiert. Dabei musste für die Sensoren jeweils die Generierung des Messvektors sowie die Berechnung des geschätzten Messwertes implementiert werden.
Anschließend wurden die Sensorfusionierungen untersucht, um ihre Funktionalität zu testen. Hierbei wurden die Implementierungen zunächst einzeln und danach im Verbund getestet. Hierbei hat die Fusionierung der Position die zu erwartenden Ergebnisse geliefert, indem die Position deutlich verbessert wurde. Ferner lag die berechnete Standardabweichung ebenfalls im erwartbaren Bereich. Daraufhin wurde die Lage im inertialen Koordinatensystem untersucht. Auch diese lieferte die erwartbare Performance und hatte eine Standardabweichung der erwarteten Größe. Bei der Fusionierung des Magnetfeldvektors fiel auf, dass die geschätzte Standardabweichung im Gegensatz zu den bisherigen Sensoren keine direkte Korrelation mit der eingestellten Varianz besaß. Weiterhin wurde bei der Lage ein gewisser Offset festgestellt, dieser tritt jedoch nicht auf, falls ein zusätzlicher positionsverbessernder Sensor fusioniert wird. Dies lässt sich somit durch die Positionsabhängigkeit des Magnetfeldes erklären. Allerdings tritt dennoch ein Unterschied in den Standardabweichungen auf. Der Grund dafür ist, dass die Richtung des Magnetfeldes nur durch zwei Größen charakterisiert wird, die Inklination und die Deklination. Somit beinhaltet die Lagebestimmung mithilfe des Magnetfeldes einen Freiheitsgrad, die Rotation um die Feldlinien. Nun wurde noch die Fusionierung der relativen Position untersucht. Diese konnte jedoch nur in Kombination mit einem weiteren lagebestimmenden Sensor genutzt werden, da die Sensordaten im körperfesten Koordinatensystem vorliegen. Sie können somit die Position nur bei Kenntnis der Lage verbessern. Ein weiteres Problem bei der Fusionierung der relativen Position ist, dass die Standardabweichung nicht steigt. Die geschätzte Standardabweichung der Position sollte ähnlich wie bei der Fusionierung der Geschwindigkeit mit der Zeit divergieren. Die Begründung dafür ist die noch fehlerhafte Implementierung der Fusionierung, bzw. dass die genutzte Sensorvarianz direkt als Unsicherheit für die Position übernommen wurde.

7 Ausblick

Im Folgenden könnten noch verschiedene Problematiken gelöst werden. Zum einen muss die relative Position derartig geändert werden, dass die Standardabweichung der Position nicht sinkt. Hierzu bieten sich zwei verschiedene Lösungsansätze an. Bei der ersten Möglichkeit könnte die Navigation History derartig geändert werden, dass in ihr die aktuelle Unsicherheit gespeichert wird. Anschließend müsste diese aktuelle Unsicherheit bei der Übertragung der Messunsicherheit im Sensorpool hinzuaddiert werden. Dadurch würde diese mit der Zeit wachsen und folglich auch die geschätzte Standardabweichung. Die zweite und auch mathematisch korrektere Möglichkeit wäre, die relative Position als einen weiteren Systemzustand einzuführen. Hierdurch müsste allerdings neben dem Systemzustand auch die Systemfunktion erweitert werden, um den Systemzustand zu propagieren. Eine andere Erweiterungsmöglichkeit wäre, die Fusionierung weiterer Sensoren zu implementieren.

Quellenverzeichnis

Literatur

[1] Nikolaus Ammann and Franz Andert. Visual navigation for autonomous, precise and safe landing on celestial bodies using unscented kalman filtering. In *2017 IEEE Aerospace Conference*, pages 1–12. IEEE, 2017.

[2] Nikolaus Ammann and Stephan Theil. Using an uav for testing an autonomous terrain-based optical navigation system for lunar landing. In *2018 IEEE Aerospace Conference*, pages 1–9. IEEE, 2018.

[3] US Army. Navstar gps user equipment introduction. Technical report, US Government, 1996.

[4] A.V.Prokhorov. Covariance. encyclopedia of mathematics. `http://www.encyclopediaofmath.org/index.php?title=Covariance&oldid=33783`.

[5] A.V.Prokhorov. Covariance matrix. encyclopedia of mathematics. `http://www.encyclopediaofmath.org/index.php?title=Covariance_matrix&oldid=13365`.

[6] Dr.-Ing. Johann Dambeck. Inertial reference trajectories. `https://www.fsd.lrg.tum.de/research/sensors-data-fusion-and-navigation/downloads/`.

[7] ESA. Conventional celestial reference system. `https://gssc.esa.int/navipedia/index.php/Conventional_Celestial_Reference_System`. abgerufen 13.4.2020.

[8] National Imagery and Mapping Agency. World geodetic system 1984 - its definition and relationships with local geodetic systems. Technical report, Department of Defense, 2000.

[9] iMAR Navigation GmbH. *Informationen zu iTraceRT-F400-E*, 2012. `http://imar-navigation.de/downloads/TraceRT-F400_de.pdf`.

[10] Josef Janisch. Was sie schon immer über hallsensoren wissen wollten kleiner effekt – große wirkung. `https://www.all-electronics.de/wp-content/uploads/migrated/article-pdf/75149/ei06-07-067.pdf`. abgerufen 23.4.2020.

[11] Simon J Julier. The scaled unscented transformation. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, volume 6, pages 4555–4559. IEEE, 2002.

[12] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–193. International Society for Optics and Photonics, 1997.

[13] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[14] Jack B. Kuipers. *Quaternions and rotation sequences : a primer with applications to orbits, aerospace, and virtual reality*. Princeton University Press, Princeton, N.J., 1999.

[15] Carl Christian Liebe. Accuracy performance of star trackers-a tutorial. *IEEE Transactions on aerospace and electronic systems*, 38(2):587–599, 2002.

[16] Normenstelle Luftfahrt. Luftfahrt norm 9300 flugmechanik: Begriffe, benennungen, zeichen.

[17] Reiner Marchthaler and Sebastian Dingler. *Kalman-Filter*. Springer, 2017.

[18] Mathworks. *Matlab Documentation to ECEF Position to LLA*. abgerufen 23.4.2020.

[19] Defence Geographic Centre National Geospatial-Intelligence Agency. The us/uk world magnetic model for 2020-2025. `https://www.ngdc.noaa.gov/geomag/WMM/DoDWMM.shtml`, 2020.

[20] Unite Nations. Current and planned global and regional navigation satellite systems and satellite-based augmentations systems. In *Proceedings of ICG*, pages 15–40, 2010.

[21] Yu.V. Prokhorov. Normal distribution. encyclopedia of mathematics. `http://www.encyclopediaofmath.org/index.php?title=Normal_distribution&oldid=33876`.

[22] Yu.V. Prokhorov. Random variable. encyclopedia of mathematics. `http://www.encyclopediaofmath.org/index.php?title=Random_variable&oldid=43639`.

[23] Stergios I Roumeliotis, Gaurav S Sukhatme, and George A Bekey. Circumventing dynamic modeling: Evaluation of the error-state kalman filter applied to mobile robot localization. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 2, pages 1656–1663. IEEE, 1999.

[24] Vincent Spruyt. A geometric interpretation of the covariance matrix. `https://www.visiondummy.com/2014/04/geometric-interpretation-covariance-matrix/`. abgerufen 23.4.2020.

[25] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, pages 153–158. Ieee, 2000.

Abbildungs- und Tabellenverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

A Diagramme

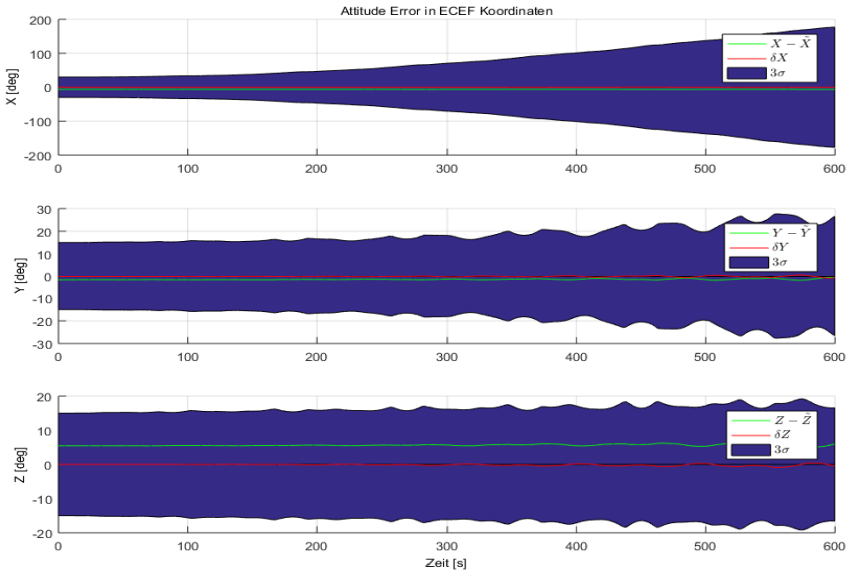


Abbildung A.1: Lage im ECEF Koordinatensystem bei reiner Inertialer Navigation

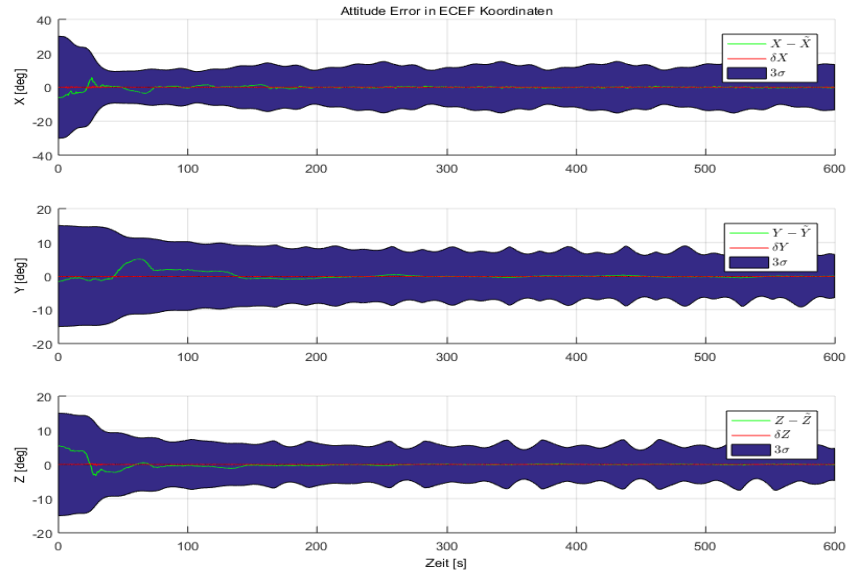


Abbildung A.2: Lage im ECEF Koordinatensystem bei Positionsfusion

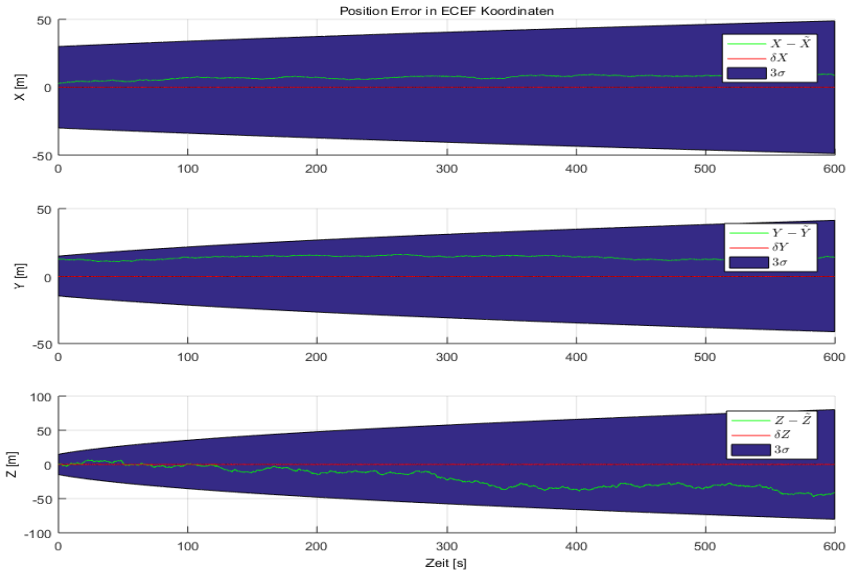


Abbildung A.3: Position im ECEF Koordinatensystem bei Geschwindigkeitsfusion

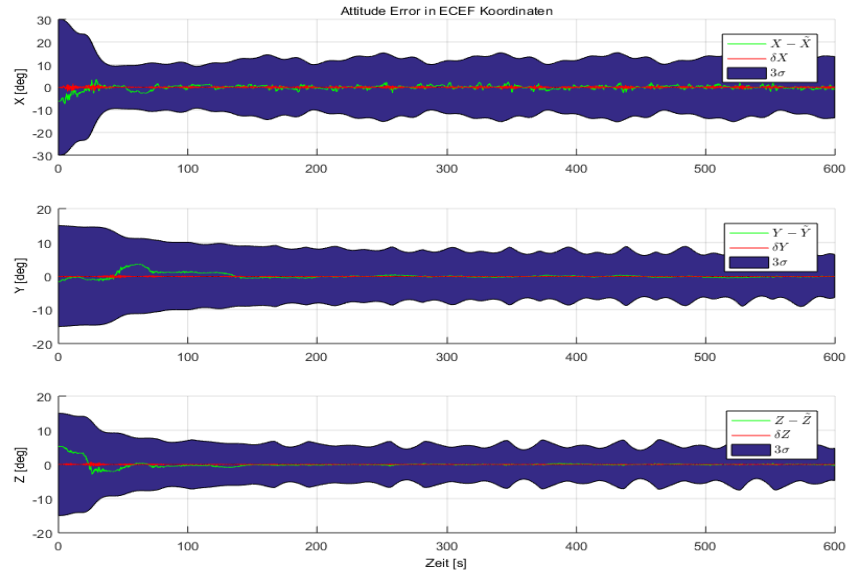


Abbildung A.4: Lage im ECEF Koordinatensystem bei Geschwindigkeitsfusion

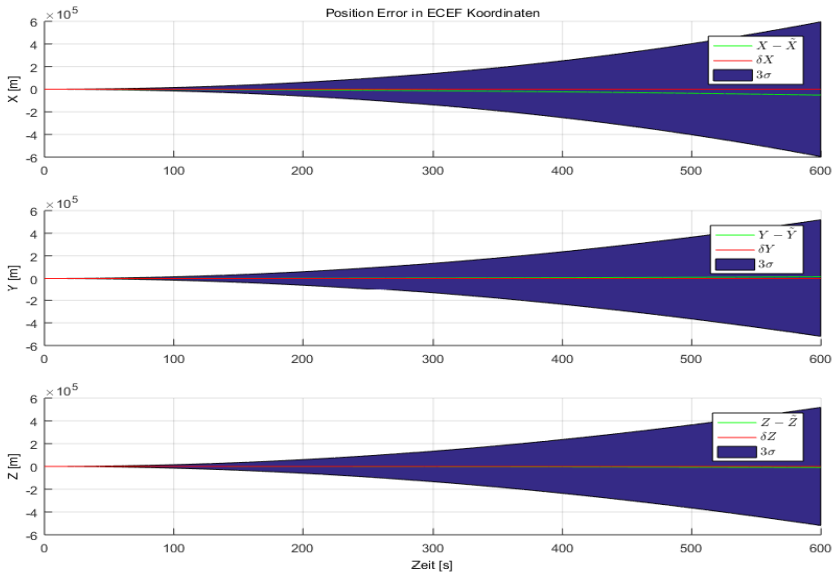


Abbildung A.5: Positionsfehler im ECEF Koordianatensystem bei Lagefusion

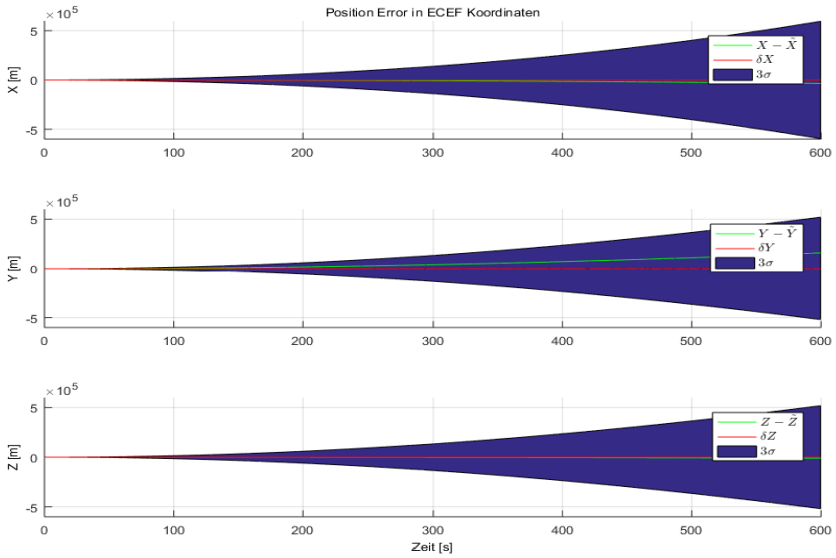


Abbildung A.6: Positionsfehler im ECEF Koordinatensystem bei Magnetfeldfusion

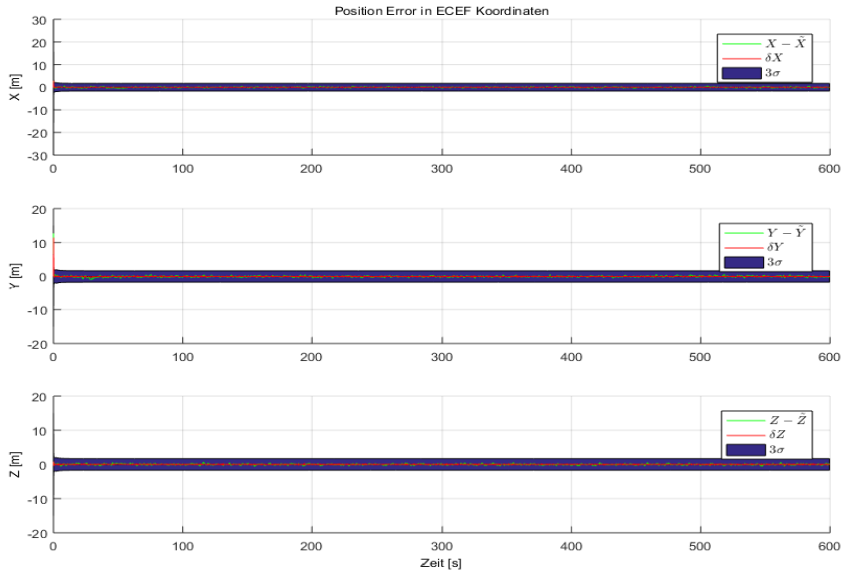


Abbildung A.7: Positionsfehler im ECEF Koordinatensystem bei Magnetfeldfusion und Positionsfusion

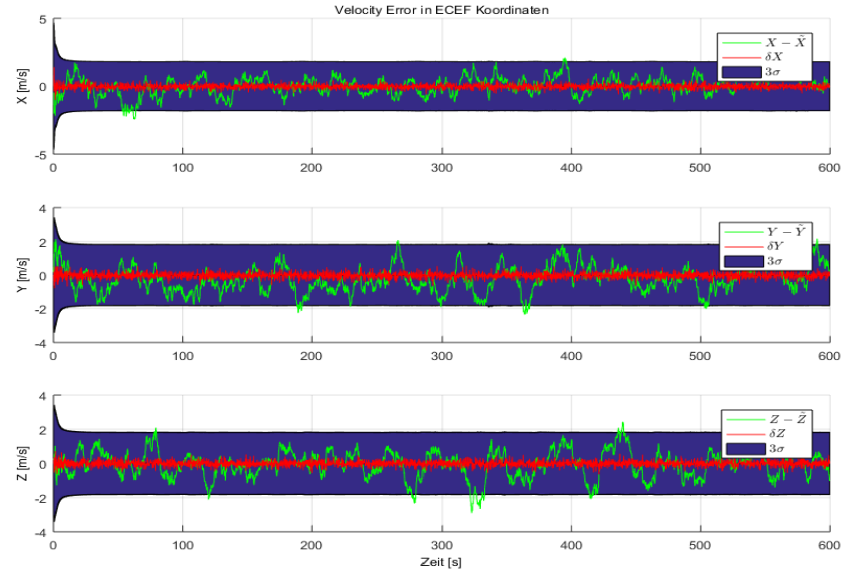


Abbildung A.8: Geschwindigkeitsfehler bei Fusionierung der relativen Position mit Lagefusion

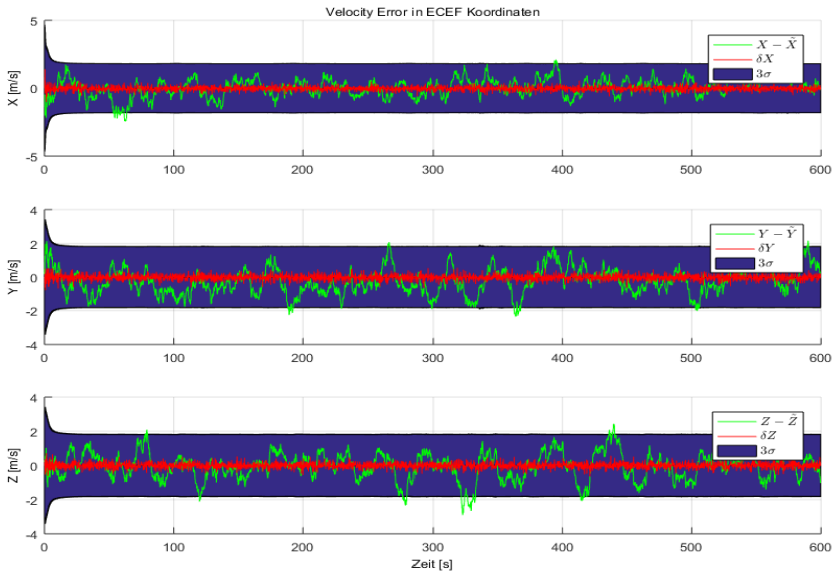


Abbildung A.9: Lagefehler bei Fusionierung der relativen Position mit Lagefusion

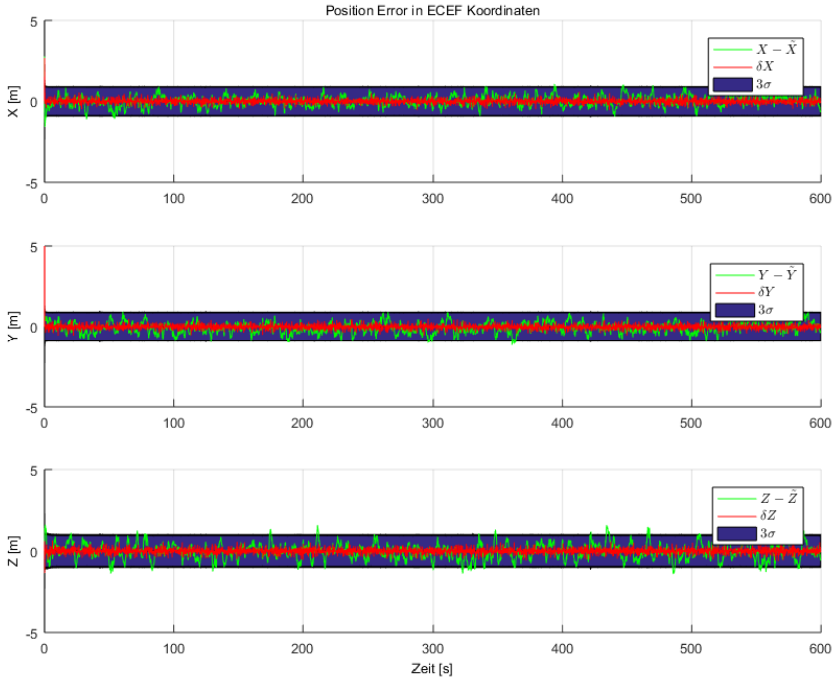


Abbildung A.10: Fehler der Positionslösung im Gesamttest

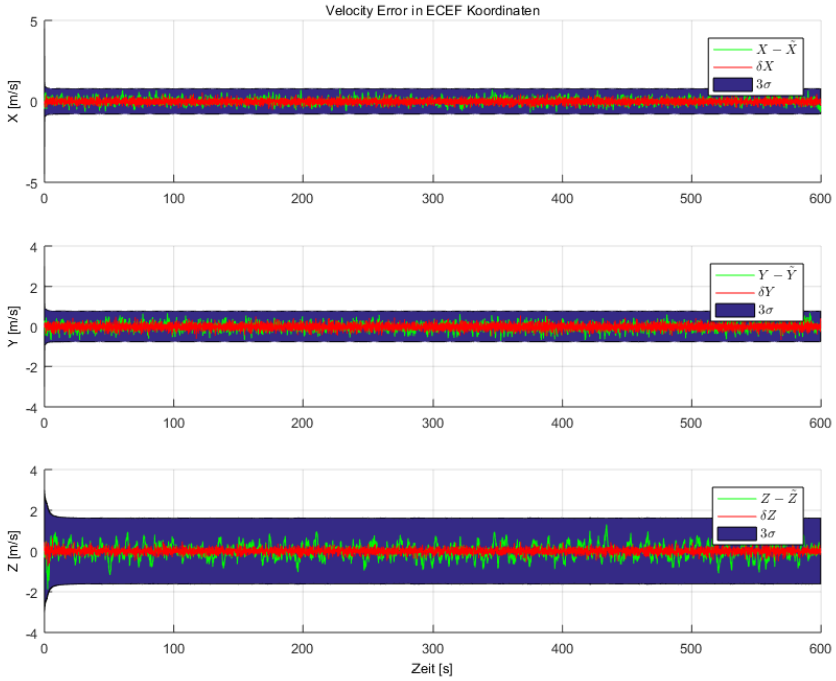


Abbildung A.11: Fehler der Geschwindigkeit im Gesamttest

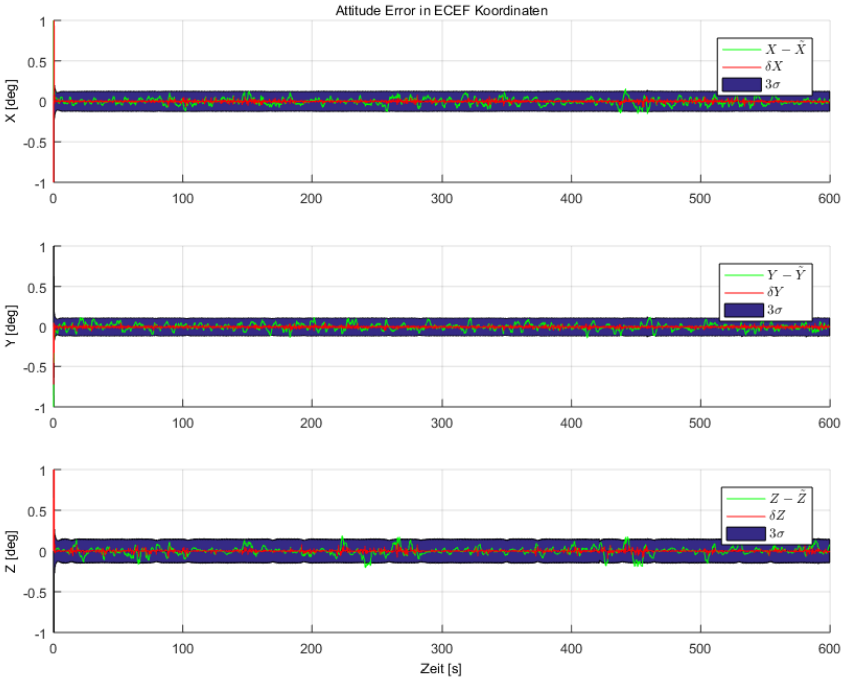


Abbildung A.12: Fehler der Lage im Gesamttest