



Semesterarbeit

Offline - Trajektorienoptimierung eines Bewegungssimulators

Daniel Spohr

2. Oktober 2014

Betreuer: Prof. Dr.-Ing. habil. Boris Lohmann
Prof. Dr.-Ing. Martin Otter (DLR)
Dr.-Ing. Paul Kotyczka
Dipl.-Ing. Andreas Labusch (DLR)

Institute: Lehrstuhl für Regelungstechnik
Technische Universität München;
Deutsches Zentrum für Luft- und Raumfahrt (DLR)
Institut für Systemdynamik und Regelungstechnik

Erklärung

Hiermit erkläre ich, die vorliegende Arbeit selbstständig durchgeführt zu haben und keine weiteren Hilfsmittel und Quellen als die angegebenen genutzt zu haben. Mit ihrer unbefristeten Aufbewahrung in der Lehrstuhlbibliothek erkläre ich mich einverstanden.

München, den 2. Oktober 2014

Aufgabenstellung

Problemstellung:

Interaktive Bewegungssimulatoren wie der Robotic Motion Simulator des DLR werden durch Eingaben des Piloten gesteuert. Um den Arbeitsraum des Simulators einzuhalten und gleichzeitig (möglichst) keine falschen Bewegungseindrücke zu erzeugen, werden auf Wahrnehmungsmodelle basierende Filter verwendet, die die vorgegebene Bewegung skalieren. Dadurch kann es wiederum passieren, dass Bewegungen, die im Rahmen des Arbeitsraums und der Dynamik möglich wären, unzureichend umgesetzt werden.

Aufgabe:

Für Manöver, bei denen der Verlauf des zu simulierenden Modells im Voraus bekannt ist, lassen sich Optimierungsverfahren für Trajektorien anwenden. Im Rahmen der Semesterarbeit sollen Algorithmen zur Trajektorienoptimierung recherchiert und auf den robotischen Bewegungssimulator angewendet werden. Dabei müssen kinematische und dynamische Beschränkungen beachtet werden. Zusätzlich gilt es, die menschliche Bewegungswahrnehmung zu berücksichtigen. Abschließend soll diese vorab optimierte Trajektorie eines Beispielmanövers mit der Trajektorie verglichen werden, die aus der Echtzeitbahnplanung des interaktiven Systems resultiert.

Inhaltsverzeichnis

Abbildungsverzeichnis	iv
Tabellenverzeichnis	vii
Nomenklatur	viii
1 Einleitung	1
1.1 Motivation	1
1.2 Aufbau der Arbeit	1
2 Der DLR Robot Motion Simulator	3
2.1 Softwareaufbau	3
2.2 Hardwareaufbau	4
2.2.1 Roboter auf Linearachse	4
2.2.2 Piloten-Kapsel	5
2.2.3 Roboterkinematik	6
2.2.4 Vergleich zu Hexapod-Systemen	7
2.3 Sicherheitsmaßnahmen	8
3 Bahnplanung als Optimierungsproblem mit Nebenbedingungen	9
3.1 Wahrnehmung	9
3.1.1 Wahrnehmungsschwellen	10
3.1.2 Washout-Filter und Tilt Coordination	10
3.2 Online-/Offline-Bahnplanung	12
3.3 Formulierung des Optimierungsproblems	13
3.3.1 Gütekriterium	15
3.3.2 Nebenbedingungen	17
3.3.3 Zusammenfassung des vorliegenden Optimierungsproblems	17
3.4 Quaternionen	18

4	Optimierungsalgorithmen	20
4.1	Downhill Simplex Methode	23
4.1.1	Eigenschaften	23
4.1.2	Tauglichkeit	24
4.2	Pattern Search	24
4.2.1	Eigenschaften	25
4.2.2	Tauglichkeit	25
4.3	SQP - Sequential Quadratic Programming	25
4.3.1	Eigenschaften	25
4.3.2	Tauglichkeit	27
4.4	Genetische Algorithmen	28
4.4.1	Eigenschaften	28
4.4.2	Tauglichkeit	28
5	Offline-Trajektorienoptimierung: Methoden, Ergebnisse und Vergleiche	30
5.1	Implementierung und Voraussetzungen	30
5.1.1	Dymola Optimization Library	31
5.1.2	Referenzszenario: Überholmanöver	33
5.2	Vergleich der Optimierungsalgorithmen	34
5.3	Verwendung der Online-Lösung als Starttrajektorie	35
5.3.1	Übertragung der Online-Lösung zur Verwendung als Starttrajektorie	36
5.3.2	Vergleich: Online \leftrightarrow Offline - Optimierung	36
5.3.3	Vergleich der Optimierungsalgorithmen mit der Online-Lösung als Starttrajektorie	40
5.4	Parallelisierung des Optimierungsprozesses: Cluster-Server	41
5.4.1	Cluster-Server	41
5.4.2	Performanz-Vergleich	42
5.5	Parameter-Tuning	43
5.5.1	Variation der Startkonfiguration \mathbf{q}_0	46
5.5.2	α, β, γ - Variation	48
5.6	Erhöhung der Stützstellenanzahl	53
6	Anwendung auf weitere Referenzszenarien	56
6.1	Überholmanöver mit höherer Geschwindigkeit	56
6.2	Landestoß	58
7	Zusammenfassung und Ausblick	61
7.1	Zusammenfassung	61

Inhaltsverzeichnis

iii

7.2 Ausblick

62

Literaturverzeichnis

63

Abbildungsverzeichnis

2.1	Systemübersicht des RMS: Geschlossene Kette zwischen Pilot, Dynamikmodell des zu simulierenden Fahr-/Flugzeugs, Bahnplanung und Robotersteuerung sowie visueller Darstellung der virtuellen Realität	4
2.2	Der <i>DLR Robot Motion Simulator</i> im Betrieb	5
2.3	Die Simulatorzelle des <i>DLR Robot Motion Simulators</i> mit Fahrzeuginstrumententräger und die Komponenten der Zelle	6
2.4	Nummerierung der Roboterachsen $q_1 \dots q_6$ und der zusätzlichen Linearachse q_7	6
3.1	Schematische Darstellung der Gütemaßanteile für Richtung und Absolutwert bezüglich der Referenz- und Istwerte	16
4.1	Vereinfachtes Schema des allgemeinen, iterativen Vorgehens zur Lösung von Optimierungsaufgaben.	21
4.2	Verlauf einer Extremwertfindung mit der Downhill Simplex Methode und Operatoren des Verfahrens	24
4.3	Ablaufdiagramm des Pattern Search Algorithmus	26
4.4	Ablaufdiagramm eines Genetischen Algorithmus	29
5.1	Modell des simulierten Roboters mit skalaren Ein- und Ausgängen als zugrundeliegendes System des Optimierungsproblems. Die Berechnung der Gütemaßsummanden sowie der normierten Signale der Nebenbedingungen erfolgt innerhalb des Modells.	31
5.2	Referenzszenario: Überholmanöver bzw. doppelter Spurwechsel eines PKW	33
5.3	Schema der Implementierung zur Verwendung der Online-Lösung als Starttrajektorie. Abgetastete Werte aus <i>.txt</i> -Datei werden durch <i>B-Splines</i> approximiert und in Form von <i>.mat</i> -Files an die <i>Optimization Library</i> übergeben.	36

5.4	Vergleich der Gelenkwinkelbeschleunigungen von Achse 5 und 6 aus der Online-Lösung mit $k = 500$ Stützstellen und approximiert mit $k = 21$ Stützstellen. Die charakteristische Beschaffenheit des exakten Signals kann durch die Approximation in ausreichendem Maße dargestellt werden. Hochfrequente Anteile werden aufgrund der niedrigen Stützstellenanzahl nicht vermittelt.	38
5.5	Vergleich der Querschleunigungen (Nickachse). Verglichen wird der Referenzverlauf, die Lösung aus der Online-Optimierung approximiert durch B-Splines mit $k = 500$ bzw. $k = 21$ Stützstellen sowie das Ergebnis der Offline-Optimierung, welche ebenfalls mit $k = 21$ Stützstellen arbeitet und die approximierte Online-Lösung mit $k = 21$ als Startlösung verwendet. Die enorme Verschlechterung des Gütemaßes der approximierten Online-Lösung ist in Anbetracht deren Beschleunigungsvermittlung (orange) nachvollziehbar.	39
5.6	Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den Gewichtungsfaktoren $\alpha = \beta = 0,3$; $\gamma = 0,8$ und fester Startkonfiguration \mathbf{q}_0	45
5.7	Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den Gewichtungsfaktoren $\alpha = \beta = 0,3$; $\gamma = 0,8$ und angepasster Startkonfiguration \mathbf{q}_0	47
5.8	Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit Tuning von α und β bei $\gamma = 0,5$	49
5.9	Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit Tuning von α und β bei $\gamma = 0,01$ und angepasster Startkonfiguration \mathbf{q}_0	50
5.10	Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit Tuning von α und β bei $\gamma = 0,2$ und angepasster Startkonfiguration \mathbf{q}_0	51
5.11	Vermehrung der Stützstellen zur Verfeinerung der Auflösung.	53
5.12	Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit Tuning von α und β bei $\gamma = 0,2$ und 43 Stützstellen sowie angepasster Startkonfiguration \mathbf{q}_0	54
5.13	Vergleich der Startkonfigurationen \mathbf{q}_0 aus der Online-Optimierung (links) bzw. der Offline-Optimierung inkl. Anpassung der Gewichtungsparameter und Erhöhung der Stützstellenanzahl (rechts).	55

- 6.1 Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen während des schnelleren Überholmanövers mit Tuning von α und β bei $\gamma = 0, 2$ und 43 Stützstellen sowie angepasster Startkonfiguration \mathbf{q}_0 57
- 6.2 Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit Tuning von α und β bei $\gamma = 0, 2$ und 47 Stützstellen sowie angepasster Startkonfiguration \mathbf{q}_0 59

Tabellenverzeichnis

2.1	Werte der in der Simulation verwendeten Beschränkungen für Gelenkwinkelpositionen \mathbf{q}_n , -geschwindigkeiten $\dot{\mathbf{q}}_n$ und -beschleunigungen $\ddot{\mathbf{q}}_n$	8
3.1	Eigenschaften von Orientierungsdarstellungen mit Rotationsmatrizen, Euler-Winkeln und <i>Quaternionen</i>	19
5.1	Optimierungsparameter für vergleichbare Optimierungsdurchläufe mit verschiedenen Algorithmen.	34
5.2	Vergleich der Optimierungssolver Simplex Algorithmus, SQP, Pattern Search und Genetischer Algorithmus hinsichtlich des Gütekriteriums und der Performanz; Starttrajektorie: Nullsignal	35
5.3	Vergleich der Gütemaße von Online- und Offline-Optimierung; Algorithmus: SQP.	37
5.4	Vergleich der Optimierungssolver Simplex Algorithmus, SQP, Pattern Search und Genetischer Algorithmus hinsichtlich Rechenzeit und Gütemaß; Starttrajektorie: Online-Lösung.	40
5.5	Optimierungsparameter für die Parallelisierung des Optimierungsprozesses.	42
5.6	Performanzvergleich zwischen den Berechnungen auf einem Kern und verteilten Berechnungen auf lokaler CPU sowie auf dem Cluster; Algorithmus: SQP; Starttrajektorie: Online-Lösung.	43
5.7	Optimierungsparameter für das Tuning von bisher festen Parameterwerten.	43

Nomenklatur

α	Gewichtungsfaktor des Beschleunigungsabsolutanteils
β	Gewichtungsfaktor des Drehratenabsolutanteils
γ	Parameter zur Gewichtung zwischen Beschleunigungen und Drehraten
\mathbb{H}	Gruppe der Quaternionen
$\omega_{\text{HP,rot}}$	Eckfrequenz des Hochpass-Filters
$\mathbf{a}_{\text{P,n}}$	Normierter Vektor der Beschleunigungen, die auf den Simulatorinsassen wirken
\mathbf{g}	Ungleichungsnebenbedingungen
\mathbf{h}	Gleichungsnebenbedingungen
\mathbf{H}_k	Hessematrix der Langrangefunktion
\mathbf{J}_R	Jakobi-Matrix der Rotation
\mathbf{J}_T	Jakobi-Matrix der Translation
\mathbf{n}	Einheitsvektor
\mathbf{q}	Vektor der Gelenkwinkel
\mathbf{q}_0	Vektor der Gelenkwinkel zum Startzeitpunkt
\mathbf{R}	Rotationsmatrix
$\mathbf{r}(\mathbf{q})$	Direktkinematik des Roboters
$\mathbf{u} = \ddot{\mathbf{q}}$	Eingangsvektor: Gelenkwinkelbeschleunigungen
\mathbf{v}	Vektorieller Quaternionenanteil
$\mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix}$	Zustandsvektor des Systems
ω_{abs}	Gütemaßanteil der Drehratenrichtung
ω_{dir}	Gütemaßanteil des Drehratenbetrags
ω_{thres}	Wahrnehmungsschwelle der Drehratenamplitude
\vec{Q}	Quaternion
λ	optimaler <i>Langrange-Multiplikator</i>
${}^0\mathbf{r}_B$	Ortsvektor des Punktes B im Inertialsystem
${}^{\text{EE}}\ddot{\mathbf{r}}_P$	Beschleunigung auf den Piloten bzw. Simulatorinsassen ohne Gravitationsanteil, dargestellt im Endeffektorsystem
${}^{\text{EE}}\mathbf{a}_P$	Beschleunigung auf den Piloten bzw. Simulatorinsassen, dargestellt im

	Endeffektorsystem
${}^{\text{EE}}\mathbf{g}_P$	Erdbeschleunigung auf den Piloten bzw. Simulatorinsassen, dargestellt im Endeffektorsystem
${}^{\text{EE}}\mathbf{r}_P$	Kartesische Position des Simulatorinsassen, dargestellt im Endeffektorsystem
a_{abs}	Gütemaßanteil der Beschleunigungsrichtung
a_{dir}	Gütemaßanteil des Beschleunigungsbetrags
b_k	<i>Base Point</i> des Pattern Search Algorithmus
d_i	<i>de-Boor-Punkte</i> einer Kurvenapproximation mit <i>B-Splines</i>
f_{Abtast}	Abtastfrequenz zur informationsverlustfreien Signalabtastung laut dem <i>Nyquist-Shannon'schen Abtasttheorem</i>
f_{max}	Maximale Eigenfrequenz eines Signals
k	Anzahl der Stützstellen pro Eingangsvariable
$L(\mathbf{x}, \mathbf{u})$	Gütemaß der Optimierung
$P_1(\mathbf{z}, \alpha)$	<i>L₁-Penalty-Funktion</i> zur Berücksichtigung von Nebenbedingungen im Rahmen der Downhill Simplex Methode
p_k	Lösung eines quadratischen Teilproblems innerhalb des SQP-Algorithmus
T_{higher}	Zeitbereich, in dem der Betrag der Referenzdrehrate oberhalb der Wahrnehmungsschwelle ω_{thres} liegt
T_{lower}	Zeitbereich, in dem der Betrag der Referenzdrehrate unterhalb der Wahrnehmungsschwelle ω_{thres} liegt
v_{PKW}	Geschwindigkeit des Fahrzeugs der Referenztrajektorie
w	Skalarer Quaternionenanteil
z_k	Startpunkt der SQP-Methode
Index '1'	Roll-Achse
Index '2'	Nick- bzw. Pitch-Achse
Index '3'	Gier- bzw. Yaw-Achse

1 Einleitung

1.1 Motivation

Zur Ausbildung von Piloten sowie zur realitätsnahen Demonstration der Bewegungen dynamischer Systeme wie PKW oder Fahrgeschäften wurde am *Deutschen Zentrum für Luft- und Raumfahrt (DLR)* ein Bewegungssimulator entwickelt, welcher dem Insassen visuell und akustisch, vor allem jedoch physisch entsprechende Bewegungseindrücke vermitteln soll. Im Gegensatz zum Großteil der bestehenden Bewegungssimulationskonzepte, wobei zumeist leistungsstarke Hexapodsysteme zum Einsatz gebracht werden, basiert der *DLR Robot Motion Simulator (RMS)* auf einem seriellen Roboter.

Um die Beschleunigungen bei der interaktiven Simulation eines Flug- oder Fahrmanövers korrekt wiederzugeben, werden Online-Bahnplanungsverfahren zum Einsatz gebracht. Für Szenarien mit im Voraus bekannter darzustellender Bewegung, wie z.B. bei Fahrgeschäften, wird im Folgenden ein Offline-Optimierungsverfahren angewendet. Dabei findet die Berechnung der Trajektorie des Simulators vor Beginn des Simulationsvorgangs statt.

In der vorliegenden Arbeit werden verschiedene Offline-Optimierungsalgorithmen untereinander und mit der Lösung des Online-Verfahrens verglichen. Zur effizienten Nutzung der Kenntnis des darzustellenden Manövers und der größeren zur Verfügung stehenden Rechenressourcen werden weitere Möglichkeiten im Bereich der Offline-Optimierung erarbeitet. Dabei ist allem voran die Anpassung der Startkonfiguration und von Gewichtungsparemtern innerhalb des Gütekriteriums zu nennen. Simulative Auswertungen verifizieren die vorgestellten Methoden und ermöglichen objektive Vergleiche.

1.2 Aufbau der Arbeit

Zunächst wird im Kapitel „Der DLR Robot Motion Simulator“ der am *DLR* entwickelte Bewegungssimulator und dessen hard- sowie softwareseitiger Aufbau vorgestellt.

Im Kapitel „Bahnplanung als Optimierungsproblem mit Nebenbedingungen“ werden einige wichtige Eigenschaften und Funktionen des menschlichen Wahrnehmungssystems vermittelt. Ferner werden moderne Filter erläutert, die bei Bewegungssimulationen

häufig Anwendung finden. Daraufhin folgt eine Beschreibung von Online- und Offline-Planungsverfahren sowie eine mathematische Formulierung des vorliegenden Optimierungsproblems wird erarbeitet. Eine kurze Einführung in die Orientierungsdarstellung mit *Quaternionen* beschließt das Kapitel.

Eine Klassifizierung von Optimierungsaufgaben als auch verschiedene Methoden zu deren Lösung sind Inhalt des Kapitels „Optimierungsalgorithmen“.

Nachdem die Optimierungsumgebung, die vorliegenden Rahmenbedingungen sowie die genutzte Referenztrajektorie näher beleuchtet werden, stellt das Kapitel „Offline-Trajektorienoptimierung: Methoden, Ergebnisse und Vergleiche“ die durchgeführten Vergleichsergebnisse dar und liefert Vorschläge zur weiteren Verbesserung der Optimierungsergebnisse, die ausschließlich im Fall der Offline-Planung angewendet werden können.

Abschließend bildet das Kapitel „Zusammenfassung und Ausblick“ ein Résumé der erzielten Ergebnisse und richtet den Blick auf weiterführende und zukünftige Forschungsthemen.

2 Der DLR Robot Motion Simulator

Eine Übersicht über die Funktionsweise eines Bewegungssimulators sowie der Aufbau des *DLR Robot Motion Simulator (RMS)* aus Hard- und Softwaresicht sind die Hauptbestandteile des folgenden Kapitels (siehe [BHH⁺11]).

2.1 Softwareaufbau

Um dem Insassen des Simulators das Gefühl zu vermitteln, in einem realen Flug- oder Fahrzeug zu sitzen, müssen die in Abb. 2.1 zu sehenden Teilfunktionen eine geschlossene Kette bilden (vgl. [Bel14]):

- Dem Piloten wird durch die Visualisierung seiner virtuellen Umgebung die aktuelle Position im virtuellen Raum übermittelt. Im Falle eines interaktiven Bewegungssimulators kann der Pilot durch Eingabemöglichkeiten im Cockpit Steuerkommandos abgeben.
- Über ein Dynamikmodell des simulierten Systems wird aus den Steuereingaben die zu simulierende Bewegung des Systems im virtuellen Raum in Form eines diskreten Verlaufs von Beschleunigungen und Drehraten berechnet. Das Signal wird dabei mit einer Frequenz von $f = 83.33\text{Hz}$ abgetastet, da die zugrundeliegende *KUKA*-Steuerung mit einer Taktung von 12ms arbeitet. Die Umgebungsvisualisierung wird mit Hilfe der zusätzlich ermittelten Position und Orientierung der Pilotenperspektive erzeugt. Um unerwünschte Effekte vor allem bei Bewegungen in den Überkopfzustand zu vermeiden, werden Orientierungen im vorliegenden Fall ausschließlich über *Quaternionen* und nicht über *Euler-Winkel* wiedergegeben.
- Im Rahmen der Bahnplanung des Simulators (engl. *Motion Cueing*) werden die Beschleunigungen und Drehraten vom globalen Arbeitsraum des virtuellen, zu simulierenden Systems in den Arbeitsraum des Simulators umgerechnet. Hierzu werden Ergebnisse der Wahrnehmungsforschung des Menschen verwendet, um möglichst realitätsnahe Bewegungen im meist kleineren Arbeitsraum wiederzugeben. Häufig verwendet wird hier ein sogenannter *Washout-Filter (WOF)*, der im Kapitel

„Washout-Filter und tilt coordination“ genauer erklärt wird. Bei nicht-interaktiven Offline-Verfahren kann diese *Washout-Filterung* umgangen werden, indem die Umrechnung in den Simulatorarbeitsraum beispielsweise innerhalb einer Optimierungsaufgabe erfolgt.

Die Ergebnisse der Online- bzw. Offline-Optimierung sind die Solltrajektorien der Stellglieder des Roboters.

- Die Umsetzung dieser Solltrajektorie geschieht durch die unterlagerte Motorregelung *KR C2* von *KUKA* und führt zur Bewegung der Pilotenkapsel.

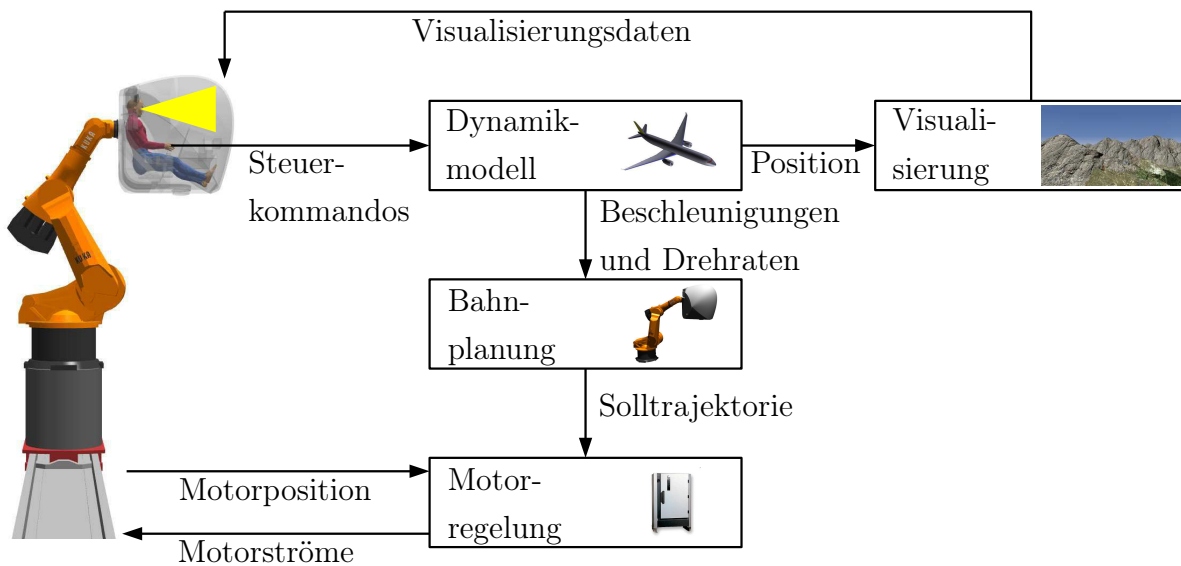


Abbildung 2.1: Systemübersicht des RMS: Geschlossene Kette zwischen Pilot, Dynamikmodell des zu simulierenden Fahr-/Flugzeugs, Bahnplanung und Robotersteuerung sowie visueller Darstellung der virtuellen Realität; Quelle: [Bel14].

2.2 Hardwareaufbau

Der *Robot Motion Simulator* besteht im Wesentlichen aus einem Industrieroboter, an den eine Cockpit-Kapsel angeflanscht ist. Die einzelnen Hardwarekomponenten werden im Folgenden vorgestellt.

2.2.1 Roboter auf Linearachse

Der *RMS* wurde auf Basis des *KUKA KR 500-2 TÜV* entwickelt. Dieser Robotertyp hat aufgrund der hohen Sicherheitsstandards die Berechtigung zum Transport von Personen.

Der vorliegende Roboter besitzt insgesamt sieben Freiheitsgrade (engl. *degrees of freedom*, DOF). Die ersten sechs Freiheitsgrade resultieren aus der seriellen Anordnung der Gelenke und der Ausrichtung der Drehachsen. Eine 10m lange Linearachse, die besonders für lange Beschleunigungsphasen von Nutzen ist, erzeugt den zusätzlichen, siebten Freiheitsgrad. Beschränkungen der Winkelstellungen aller Gelenke, der Gelenkwinkelgeschwindigkeiten und -beschleunigungen sind hard- und softwareseitig ausgeführt. Dabei wurde eine zusätzliche Gelenkwinkelbeschränkung hardwareseitig realisiert, um eine Kollision zwischen Kapsel und Roboterarm auszuschließen. Hardwareanschläge in Form von Metallpuffern verhindern Gelenkwinkelbewegungen über die vorgegeben Grenzen hinaus. Zum Tragen kommen die Sicherheitsmaßnahmen im Falle eines Getriebebruchs o.Ä.

Abb. 2.2 zeigt den *RMS* auf der zusätzlichen Linearachse in der Versuchshalle des *DLR*. Im linken Bild ist im Hintergrund die Flugsimulation auf der Leinwand zu erkennen.

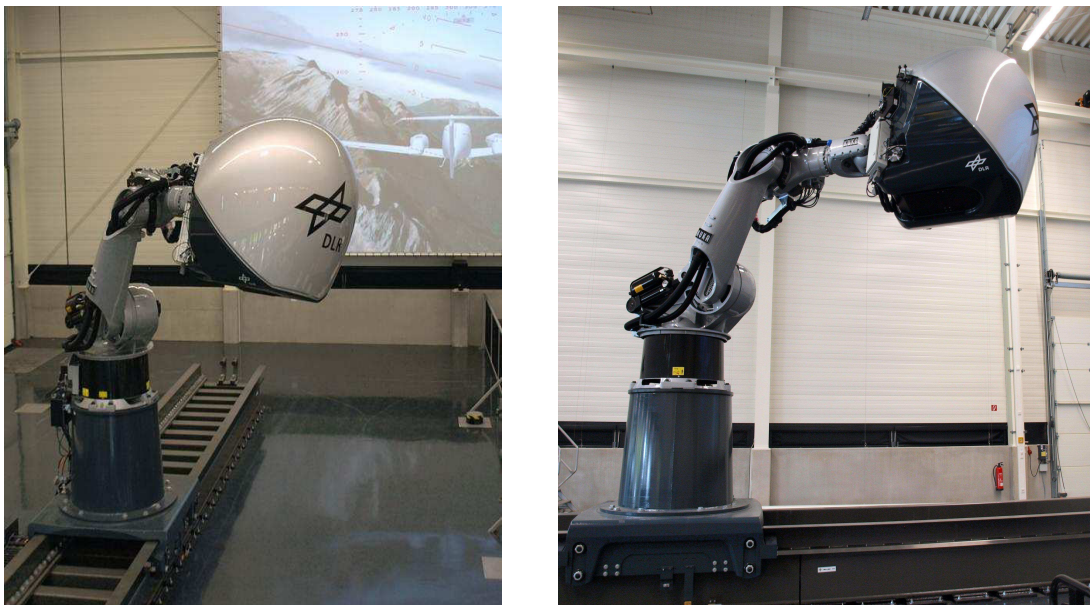


Abbildung 2.2: Der *DLR Robot Motion Simulator* im Betrieb; Quelle: *DLR*

2.2.2 Piloten-Kapsel

Am Flansch des Roboterarms befindet sich eine Kapsel für den Piloten, welche durch die mit einem Klappmechanismus ausgestatteten Gondelhaube betreten werden kann. Darin ist das Cockpit inklusive aller Instrumente integriert, welches dem vorliegenden, zu simulierenden Flug- oder Fahrzeug angepasst werden kann. Im Inneren der Kapsel befinden sich zwei High-Definition-Beamer, welche die virtuelle Aussicht des Piloten aus dem Cockpit an die Kapselinnenwand projizieren. Durch eine vom Piloten getragene 3D-

Brille entsteht für diesen ein räumlicher Eindruck der Flug- oder Fahrzeugumgebung. Die Signalübertragung zwischen dem Roboterarm und der Kapsel wird mit Hilfe von zwei Ethernetkabeln realisiert. Abb. 2.3 zeigt die Pilotengondel des *RMS* und deren Aufbau.



Abbildung 2.3: Die Simulatorzelle des *DLR Robotic Motion Simulators* mit Fahrzeuginstrumententräger (links). Die Komponenten der Zelle (rechts); Quelle: [Bel14].

2.2.3 Roboterkinematik

Der serielle Aufbau des Roboters sowie die Bezeichnungen der sieben Achsen $q_1 \dots q_7$ werden in Abb. 2.4 dargestellt.

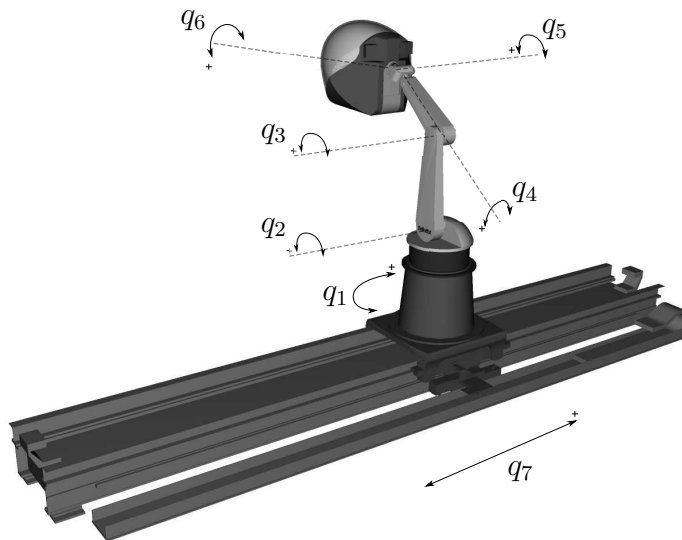


Abbildung 2.4: Nummerierung der Roboterachsen $q_1 \dots q_6$ und der zusätzlichen Linearachse q_7 .

Die Gelenkwinkel dienen als generalisierte Koordinaten des Systems, mit Hilfe derer die Systemkonfiguration vollständig und nicht-redundant beschrieben werden kann. Mit Kenntnis der Abmessungen ist somit jeder Punkt des Roboters abhängig von den generalisierten Koordinaten $\mathbf{q} \in \mathbb{R}^7$ darstellbar. Dabei wird jeder starr mit dem Roboter verbundene Punkt B mit dessen Ortsvektor ${}^0\mathbf{r}_B$ im Inertialsystem bezeichnet.

Die Funktion, welche ausgehend von den Systemabmessungen die Gelenkwinkel auf die kartesische Position eines Punktes abbildet wird als *Vorwärtslösung* oder *Direktkinematik* $\mathbf{r}(\mathbf{q})$ bezeichnet. Die Abbildung der Gelenkwinkel auf eine kartesische Position mit Hilfe der *Direktkinematik* ist eindeutig. Dynamische Zusammenhänge mit *Jakobi-Matrizen* zur Aufstellung der Bewegungsgleichungen, die auf den kinematischen Beziehungen aufbauen, werden im Abschnitt „Formulierung des Optimierungsproblems“ innerhalb des folgenden Kapitels behandelt.

Die Umkehroperation der *Direktkinematik*, mit der ausgehend von einer kartesischen Position eines Roboterpunktes und der Orientierung des ihn beschreibenden Systems die dafür nötige Gelenkwinkelkonfiguration ermittelt wird, wird als *Rückwärtslösung* oder *Inverskinematik* betitelt. Bei seriellen robotischen Systemen ist diese Abbildung im Allgemeinen nicht zwingend eindeutig.

Es sei an dieser Stelle erwähnt, dass für die Simulationen zur Erprobung der Optimierungsverfahren aus Geheimhaltungsgründen seitens des Roboterherstellers nicht das Modell des *RMS* verwendet wird. Das genutzte Simulationsmodell unterscheidet sich in der kinematischen Struktur zwar nicht vom *RMS*, dessen Parameter (Abmessungen, Massen, Massenschwerpunkte, dynamische Restriktionen) differieren jedoch von denen des *RMS*. Alle vorgestellten Methoden und Vorgehensweisen sind jedoch gänzlich unabhängig vom zugrundeliegenden System und können demzufolge bei jeglicher Art von technischen Systemen eingesetzt werden. Im vorliegenden Fall beschränkt sich die Anwendung auf serielle, robotische Systeme.

Tab. 2.1 beinhaltet die konkret vorliegenden Werte der gelenkseitigen Beschränkungen.

2.2.4 Vergleich zu Hexapod-Systemen

Hexapoden sind parallelkinematische Systeme mit sechs aktuierten und somit längenveränderlichen Beinen mit sechs DOF (vgl. [HSM02]). Die Vorteile des seriell aufgebauten *RMS* gegenüber einem Hexapod-Simulator liegen im Wesentlichen in einem deutlich größeren rotatorischen Arbeitsraum sowie in dem geringeren Anschaffungspreis aufgrund der Massenproduktion von seriellen Industrierobotern. Der vergleichsweise große Platzbedarf des *RMS*, speziell aufgrund der zusätzlichen Linearachse, kann hingegen als Nachteil bewertet werden.

Tabelle 2.1: Werte der in der Simulation verwendeten Beschränkungen für Gelenkwinkelpositionen \mathbf{q}_n , -geschwindigkeiten $\dot{\mathbf{q}}_n$ und -beschleunigungen $\ddot{\mathbf{q}}_n$.

n		1	2	3	4	5	6	7
$\mathbf{q}_n[\text{rad}]$	max	3,124	-0,698	1,222	3,124	1,047	3,124	-2,5[m]
	min	-3,124	-2,269	-0,524	-3,124	-1,047	-3,124	-8,5[m]
$\dot{\mathbf{q}}_n[\frac{\text{rad}}{\text{s}}]$	max	$\pm 1,2$	$\pm 1,0$	$\pm 1,2$	$\pm 1,3$	$\pm 1,3$	$\pm 2,0$	$\pm 1,5[\frac{\text{m}}{\text{s}}]$
	min							
$\ddot{\mathbf{q}}_n[\frac{\text{rad}}{\text{s}^2}]$	max	$\pm 1,7$	$\pm 1,3$	$\pm 2,2$	$\pm 0,7$	$\pm 1,6$	$\pm 1,1$	$\pm 0,8[\frac{\text{m}}{\text{s}^2}]$
	min							

2.3 Sicherheitsmaßnahmen

Zu den Sicherheitseinrichtungen des *RMS* zählen ein Rauchmelder, ein Temperatursensor, eine Kamera und ein Intercomsystem in der Cockpitkapsel. Ferner ist der vom Simulator zu erreichende Raum durch einen Zaun gesichert, sodass sich während der Betriebszeit keine Personen im Gefahrenbereich aufhalten können. Das gewaltsame Öffnen der Zauntür während des Betriebs führt zu einem Nothalt. Temperatur- und Rauchsensoren können den Roboter ebenso wie die Betätigung einer Notfalltaste durch den Piloten in die Ausgangsposition zurückfahren lassen.

Kommt die Kapsel kopfüber zum Stillstand, kann sie vom Kontrollraum aus in die aufrechte Lage bewegt werden. Softwareseitig werden sowohl in den vom *DLR* entwickelten Komponenten als auch in der Robotersteuerung *KR C2* von *KUKA* zusätzliche Softwareebenen zur Überwachung eingesetzt. Des Weiteren sind auf Softwareebene die Nebenbedingungen als Teil des Optimierungsproblems zu nennen (siehe Abschnitt „Nebenbedingungen“ im nachfolgenden Kapitel), welche bereits im Bahnplanungsprozess Gelenkzustände außerhalb eines bestimmten Bereiches verhindern. Diese sind auf Hardwareebene durch die genannten Anschläge am Roboterarm zusätzlich abgesichert.

3 Bahnplanung als Optimierungsproblem mit Nebenbedingungen

Grundsätzlich sind zwischen den Verfahren der Online- und Offline-Bahnplanung klare Unterschiede hervorzuheben:

Die Online-Bahnplanung berücksichtigt Steuereingaben des Insassen in Echtzeit und ermittelt die notwendigen Gelenkwinkelbeschleunigungen mit Hilfe eines *Washout-Filters* und der inversen Kinematik des Roboters. Bei der Offline-Bahnplanung erfolgt die Ermittlung der Gelenkwinkelbeschleunigungen vor Beginn des Simulationsstarts. Im vorliegenden Fall wird dabei kein komplexer *WOF* genutzt. Durch die Formulierung eines Optimierungsproblems soll die Trajektorie der Roboterinputs für die komplette Dauer der Simulation errechnet werden, wobei die hard- und softwareseitigen Begrenzungen stets in Form von Nebenbedingungen berücksichtigt werden. Im Folgenden wird genauer auf die Umwandlung der vorgegebenen Bewegung des zu simulierenden Systems in die Trajektorie des Simulators vor dem Hintergrund des menschlichen Wahrnehmungsmechanismus eingegangen. Außerdem werden die Unterschiede zwischen dem interaktiven Betrieb mit Online-Bahnplanung und der Offline-Planung bei nicht-interaktiven Manövern hervorgehoben.

3.1 Wahrnehmung

Die menschliche Wahrnehmung des Gleichgewichts erfolgt über das sogenannte Vestibularsystem und ist ein entscheidender Bestandteil für die Funktionsweise der Bewegungssimulation. Durch Drehbeschleunigungen wird innerhalb der drei orthogonal angeordneten Bogengänge des Vestibularsystems Lymphflüssigkeit in Bewegung gebracht, welche ihrerseits sensible Sinneshäärchen anregt. Durch das Nervensystem gelangt die Information ins Gehirn und wird dort als Rotation interpretiert. Translatorische Beschleunigungen werden in den Otolithen-Kammern registriert. Sinneshäärchen befinden sich dabei in einer gallertartigen Masse, welche zusätzlich Kristalle enthält. Durch Trägheitseffekte wird diese

Masse bei Einwirkung von Beschleunigungen in Bewegung versetzt. Diese Bewegung wird von den Sinneshäuten wahrgenommen und an das Nervensystem übermittelt. Empirische Kenntnisse dieses Vorgangs ermöglichen es, dem Simulatorinsassen durch geschickte Kombination aus Beschleunigungen, Drehraten sowie visuellen und akustischen Signalen eine Bewegung vorzutäuschen, die der eines Flug- oder Fahrzeugs entspricht. Allerdings gilt es festzuhalten, dass die zur Modellierung verwendeten Kennwerte dieser Wahrnehmung von Mensch zu Mensch variieren und deshalb nicht jeder Insasse ein exakt gleiches Gefühl der Bewegung vermittelt bekommt.

Neben dem vestibulären verfügt der Mensch noch über Wahrnehmungssysteme für auditive (Ohr), visuelle (Auge) und somato-sensorische (Haut, Muskeln, Gelenke, Sehnen) Reize (vgl. [Fis09]). Wichtig ist bei allen Bewegungssimulationsalgorithmen, dass die durch die verschiedenen Wahrnehmungssysteme wahrgenommenen Reize einen physikalisch sinnvollen, konfliktfreien Gesamteindruck vermitteln. Bei widersprüchlichen Informationen der verschiedenen Wahrnehmungssysteme kann es zur „Simulatorkrankheit“ kommen, welche sich als Kopfschmerz, Schwindel oder Übelkeit auswirkt.

3.1.1 Wahrnehmungsschwellen

Jeder Mensch kann eine Drehrate erst ab einer gewissen Amplitude wahrnehmen (siehe [GJ48]). Da dieser Schwellwert jedoch von Mensch zu Mensch verschieden ist, liefern verschiedene Quellen über durchgeführte Studien jeweils differierende Zahlenwerte. In [RN85] wurde als Wahrnehmungsschwelle ein Amplitudenwert von

$$\omega_{\text{thres}} = 0,05236 \frac{\text{rad}}{\text{s}} = 3,0^\circ/\text{s} \quad (3.1)$$

ermittelt. Im Rahmen der Offline-Lösung erfolgt eine gesonderte Behandlung der Anteile unterhalb des Schwellwertes innerhalb der Optimierungsaufgabe, worauf im Abschnitt „Formulierung des Optimierungsproblems“ detaillierter eingegangen wird.

Zur Filterung des Drehratenreferenzsignals wird außerdem analog zu [BOH11] ein Hochpassfilter mit einer Eckfrequenz von

$$f_{\text{HP, rot}} = 0,4 \text{ Hz} \quad (3.2)$$

angewendet, um niedrige Frequenzen des Drehratensignals zu entfernen. Die niedrigfrequenten Anteile der Drehraten sind für die Empfindung des Simulatorinsassen unerheblich.

3.1.2 Washout-Filter und Tilt Coordination

Dem *klassischen Washout-Filter (CWOFF)*, wie er bei der Online-Bahnplanung in [Bel14] zum Einsatz kommt, obliegt die Aufgabe, großskalige Bewegungen des zu simulierenden

Systems auf den eingeschränkten, im Vergleich zum realen System sehr kleinen Arbeitsraum des Simulationsroboters zu projizieren. Unter Berücksichtigung des Wahrnehmungsmechanismus des Menschen erfolgt hierbei eine Aufteilung der Eingangsgrößen. Neben einer Aufspaltung in rotatorische und translatorische Anteile werden hoch- und niederfrequente Anteile gesondert betrachtet. In [BOH11] werden den einzelnen Anteilen folgende Umsetzungen zugewiesen:

- **Translation**

- **hochfrequente** Beschleunigung:
Wenn möglich direkte Umsetzung im Simulator.
- **niederfrequente** Beschleunigung:
Berücksichtigung über *tilt coordination* → Verdrehen der Kapsel, da das menschliche Vestibularsystem eine translatorische Beschleunigung nicht von einer Neigung des Kopfes unterscheiden kann.
- Geschwindigkeit & Position:
Werden nicht berücksichtigt, da keine Kräfte auf den Insassen resultieren.

- **Rotation**

- Drehraten:
Um eine rotatorische Geschwindigkeit wahrzunehmen genügt es, die Drehrate durch einen kurzen Reiz anzudeuten und danach langsam abzubremsen. Zusammen mit dem weiterhin rotierenden visuellen Input entsteht der Eindruck, dass die Drehung im angedeuteten Maß fortgeführt wird. Demzufolge müssen lediglich **hochfrequente** Anteile berücksichtigt werden.
- Orientierung:
Entspricht einer Veränderung der Richtung des Gravitationsvektors → wird im Rahmen der **niederfrequenten**, translatorischen Beschleunigung durch die *tilt coordination* berücksichtigt.

Die *Washout-Filterung* projiziert die wahrnehmbaren Beschleunigungen und Drehraten des Manövers aus der kartesischen Referenztrajektorie auf eine vereinfachte, an den Arbeitsraum des Roboters angepasste, ebenfalls kartesische Trajektorie. Mittels der inversen Roboterkinematik kann daraufhin der Gelenkwinkel- bzw. Gelenkwinkelbeschleunigungsverlauf ermittelt werden und an die Robotersteuerung übergeben werden.

Nachteil des klassischen Washout-Filters

Die Parametrisierung des Washout-Filters hängt vom „stärksten Stoß“ ab. Deshalb ist der Bewegungsablauf, welcher den Simulator am nächsten an die Grenzen seines Arbeitsraums

bzw. am weitesten über diese Grenzen hinaus beansprucht, für die Parameter des *CWOF* maßgeblich. Damit werden jedoch alle anderen Bewegungsabläufe - auch diejenigen, die weit innerhalb der Arbeitsraumgrenzen liegen - abgeschwächt und somit nicht korrekt dargestellt.

Alternativ kann eine Limitierung der Eingangswerte erfolgen. Dies hätte jedoch ebenfalls eine Verzerrung der vermittelten Bewegungen zur Folge.

3.2 Online-/Offline-Bahnplanung

Während beim Einsatz von Robotern zum Handling von Objekten meist nur die Start- und Zielkonfiguration des Endeffektors von Relevanz ist (*point-to-point motion*), muss die Endeffektorkonfiguration bei der maschinellen Verarbeitung von Werkstücken mit Hilfe von Industrierobotern, wie beispielsweise beim Schweißen oder Zerspanen, sowie bei dem in dieser Arbeit behandelten Einsatz als Bewegungssimulator einer vorgegebenen Wunschtrajektorie möglichst exakt folgen (*path motion*) (siehe [SSVO09]). Das Ziel der Trajektorienplanung ist es, abhängig von einem vorgegebenen, gewünschten Verhalten einen zeitabhängigen Verlauf der relevanten Eingangsgrößen zu generieren. Im vorliegenden Fall werden ausgehend von den vorgegebenen Signalen der Beschleunigungen und Drehraten, welche auf den Simulatorpiloten einwirken sollen, die Verläufe der Gelenkwinkelbeschleunigungen $\ddot{\mathbf{q}}(t)$ geplant. Singularitäten im Arbeitsraum, konfigurationsabhängiges Beschleunigungsvermögen und der Umgang mit der kinematischen Redundanz des Robotersystems stellen zusätzliche Herausforderungen für den Bahnplanungsalgorithmus dar.

In der vorliegenden Arbeit werden Offline-Planungen untersucht und mit einer bestehenden Online-Lösung verglichen, bei der eine fahrbare Trajektorie ebenfalls mit Hilfe eines Optimierungsvorgangs unter Einhaltung von Nebenbedingungen ermittelt wird.

Online-Bahnplanung

Bei Online-Verfahren wird das jeweilige Problem während des laufenden Programms bzw. der laufenden Simulation gelöst. Bei der in [BOH11] vorgestellten Vorgehensweise wird zunächst auf Basis eines modifizierten *WOF* eine kartesische Trajektorie geplant. Dabei werden die Parameter des *WOF* an den Arbeitsraum des Roboters angepasst. Die Referenztrajektorie gibt diejenigen Bewegungen wieder, welche zur Erweckung der entsprechenden Eindrücke beim Insassen nötig sind. Daraufhin wird ein Optimierungsproblem ausschließlich im Rahmen eines Abtastschrittes gelöst (Ein-Schritt-Verfahren). Aus der iterativen Lösung der Ein-Schritt-Optimierung resultieren die Gelenkwinkeltrajektorien

einer fahrbaren Simulatorbewegung. Dieses Vorgehen sorgt dafür, dass die Ausführung der durch den Insassen kommandierten Eingaben innerhalb eines vorhersagbaren Zeitintervalls abgeschlossen wird. Das Verfahren ist somit echtzeitfähig.

Da dieser Algorithmus auch für interaktive Simulatoren anwendbar sein soll, können hierbei keinerlei zukünftige Referenzbewegungen berücksichtigt werden. Aufgrund dessen wird die Startkonfiguration des Roboters \mathbf{q}_0 so gewählt, dass das summierte Beschleunigungsvermögen aller Richtungen maximal ist. Liegt kein Bewegungssignal an, sorgt der *WOF* außerdem dafür, dass der Roboter immer wieder in seine Startkonfiguration zurückgeführt wird.

Offline-Bahnplanung

Vorgehensweisen, bei denen jegliche Berechnungen vor Beginn des Programm- bzw. Simulationsstarts erfolgen, werden als Offline-Verfahren bezeichnet. Bei der Offline-Bahnplanung wird die gesamte Referenztrajektorie als bekannt vorausgesetzt und der komplette Verlauf der Eingangsgrößen vor Beginn der Simulation in Form eines umfassenden Optimierungsproblems geplant. Aufgrund dessen kann hierbei eine referenzabhängige Anpassung der Startkonfiguration \mathbf{q}_0 erfolgen. Der Roboter wird demzufolge vor Beginn der Simulation in exakt die Position gebracht, von der aus die folgende Bewegung am effizientesten durchführbar ist, ohne dabei Restriktionen zu verletzen. Die Startkonfiguration ist dabei nicht - wie bei der Online-Lösung - ein Kompromiss für jegliche Trajektorien, sondern eine speziell für die vorliegende Aufgabe optimale Ausgangsposition.

Ferner werden in der vorliegenden Arbeit - im Gegensatz zur Nutzung eines *WOF* im Rahmen der Online-Optimierung - die Gelenkwinkelbeschleunigungen direkt aus der kartesischen Referenztrajektorie ermittelt. Unter Berücksichtigung der Systemgrenzen in Form von Nebenbedingungen und Nutzung der Eigenschaften des Vestibular- und Otholithsystems kann somit der „Umweg“ über die aufwendige Inverskinematik des Roboters vermieden werden. Es erfolgt ausschließlich eine möglichst genaue Anpassung der wahrnehmbaren Beschleunigungen und Drehraten an deren Referenzverläufe.

3.3 Formulierung des Optimierungsproblems

Zunächst werden die Systemvariablen des zugrundeliegenden Systems wie folgt zugewiesen:

$$\mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix}, \quad \mathbf{u} = \ddot{\mathbf{q}} \quad \rightarrow \quad \dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{u} \quad (3.3)$$

Dabei ist $\mathbf{q} \in \mathbb{R}^n$ der Vektor der Roboter-Gelenkwinkel. Im vorliegenden Fall mit zusätzlicher Linearachse gilt $n = 7$.

Der Systemzusammenhang besteht aus einer reinen Integration. Momente, Kräfte und deren dynamische Beziehungen zu den generalisierten Systemkoordination tauchen dabei nicht auf, da diese von der zugrundeliegenden Robotersteuerung berücksichtigt werden. Der Vorteil dieser reinen Kinematik-Betrachtung liegt neben der einfachen Handhabbarkeit in der Berücksichtigung der Nebenbedingungen. Die restringierten Systemgrößen \mathbf{q} , $\dot{\mathbf{q}}$ und $\ddot{\mathbf{q}}$ liegen somit in Form der Zustands- und Eingangsvariablen vor, was deren Berücksichtigung enorm vereinfacht. Restriktionen der Motormomente werden durch die Beschränkung von $\ddot{\mathbf{q}}$ automatisch eingehalten, da die Gelenkwinkelbeschleunigungen so ausgelegt sind, dass die Grenzen der Motormomente nie verletzt werden.

Die Optimierung der Bahntrajektorien basiert auf dem Vergleich von vorgegebenen und gemessenen Beschleunigungen und Drehraten, welche der Simulatorinsasse wahrnimmt. Dabei steht der Index „P“ für „Person“ bzw. „Simulator-Pilot“, während „ref“ auf die „Referenztrajektorie“ verweist. Die auf den Simulator-Piloten wirkende spezifische Beschleunigung ${}^{\text{EE}}\mathbf{a}_P(\mathbf{x}, \mathbf{u}, t)$, dargestellt im Endeffektorsystem (Index „EE“, vereinfacht als $\mathbf{a}_P(\mathbf{x}, \mathbf{u}, t)$ bezeichnet) erhält man wie folgt:

$$\mathbf{a}_P(\mathbf{x}, \mathbf{u}, t) = {}^{\text{EE}}\mathbf{a}_P(\mathbf{x}, \mathbf{u}, t) = {}^{\text{EE}}\mathbf{g}_P(\mathbf{x}, t) - {}^{\text{EE}}\ddot{\mathbf{r}}_P(\mathbf{x}, \mathbf{u}, t) \quad (3.4)$$

mit der Erdbeschleunigung dargestellt im Endeffektorsystem und vereinfacht als $\mathbf{g}_P(\mathbf{x}, t)$ bezeichnet:

$$\mathbf{g}_P(\mathbf{x}, t) = {}^{\text{EE}}\mathbf{g}_P(\mathbf{x}, t) = T_{\text{EE},0}(\mathbf{x}, t) \cdot \begin{bmatrix} 0 \\ 0 \\ -9,81 \frac{\text{m}}{\text{s}^2} \end{bmatrix} \quad (3.5)$$

wobei sich die Transformationsmatrix $T_{\text{EE},0}(\mathbf{x}, t)$ mit Hilfe von Gl. 3.34 berechnen lässt. Der Beschleunigungsanteil ohne Gravitationseffekte ${}^{\text{EE}}\ddot{\mathbf{r}}_P(\mathbf{x}, \mathbf{u}, t)$, ebenfalls im Endeffektorsystem und vereinfacht als $\ddot{\mathbf{r}}_P(\mathbf{x}, \mathbf{u}, t)$ bezeichnet, welcher von der Bewegung der Simulatorkapsel hervorgerufen wird, ergibt sich mit der direkten Kinematik $\mathbf{r}(\mathbf{q})$ des Roboters

$$\dot{\mathbf{r}} = \mathbf{J}_T \cdot \dot{\mathbf{q}}, \quad \ddot{\mathbf{r}} = \dot{\mathbf{J}}_T(\mathbf{q}, \dot{\mathbf{q}}) \cdot \dot{\mathbf{q}} + \mathbf{J}_T(\mathbf{q}) \cdot \ddot{\mathbf{q}} \quad (3.6)$$

und dem Systemzusammenhang zu

$$\ddot{\mathbf{r}}_P(\mathbf{x}, \mathbf{u}, t) = {}^{\text{EE}}\ddot{\mathbf{r}}_P(\mathbf{x}, \mathbf{u}, t) = \begin{bmatrix} \mathbf{0}_{3 \times n} & \dot{\mathbf{J}}_T(\mathbf{x}, t) \end{bmatrix} \cdot \mathbf{x}(t) + \mathbf{J}_T(\mathbf{x}, t) \cdot \mathbf{u}(t). \quad (3.7)$$

$\mathbf{J}_T(\mathbf{x}, t)$ repräsentiert hierbei die Jakobi-Matrix der Translation:

$$\mathbf{J}_T(\mathbf{x}, t) = \frac{\partial \mathbf{r}}{\partial \mathbf{q}}, \quad \mathbf{J}_T(\mathbf{x}, t) \in \mathbb{R}^{3 \times n} \quad \text{mit} \quad n = 7. \quad (3.8)$$

Die Drehraten, denen der Simulator-Pilot ausgesetzt ist, ${}^{\text{EE}}\boldsymbol{\omega}_{\text{P}}(\mathbf{x}, t)$, wiederum dargestellt im Endeffektorsystem und vereinfacht mit $\boldsymbol{\omega}_{\text{P}}(\mathbf{x}, t)$ bezeichnet, ergeben sich mit der Jakobi-Matrix der Rotation

$$\mathbf{J}_{\text{R}}(\mathbf{x}, t) = \frac{\partial \boldsymbol{\omega}}{\partial \dot{\mathbf{q}}}, \quad \mathbf{J}_{\text{R}}(\mathbf{x}, t) \in \mathbb{R}^{3 \times n} \quad \text{mit} \quad n = 7 \quad (3.9)$$

zu

$$\boldsymbol{\omega}_{\text{P}}(\mathbf{x}, t) = \mathbf{J}_{\text{R}}(\mathbf{x}, t) \cdot \dot{\mathbf{q}} = \begin{bmatrix} \mathbf{0}_{3 \times n} & \mathbf{J}_{\text{R}}(\mathbf{x}, t) \end{bmatrix} \cdot \mathbf{x}(t) \quad (3.10)$$

3.3.1 Gütekriterium

Die Minimierung des Gütemaßes sorgt dafür, dass die Kräfte bzw. Beschleunigungen sowie die Drehraten, die auf den Insassen wirken, möglichst genau den vorgegebenen Trajektorien entsprechen. Des Weiteren wird in Anbetracht der im Abschnitt „Wahrnehmungsschwellen“ genannten, minimal spürbaren Drehrate ω_{thres} der Anteil $\omega_{\text{abs}}(t)$ nur während der Abschnitte aktiv, in denen der Absolutwert der Referenzdrehrate größer als die minimal wahrnehmbare Drehrate ist. Andernfalls wird lediglich in Form einer weiteren Nebenbedingung gefordert, dass sich der Absolutwert der Ist-Drehrate unterhalb der Wahrnehmungsschwelle befindet (siehe Abschnitt „Nebenbedingungen“). Demzufolge wird zwischen folgenden Zeitabschnitten unterschieden:

$$T_{\text{higher}} : \quad \|\boldsymbol{\omega}_{\text{ref}}(t_{\text{higher}})\| > \omega_{\text{thres}}, \quad t_{\text{higher}} \in T_{\text{higher}} \quad (3.11)$$

$$T_{\text{lower}} : \quad \|\boldsymbol{\omega}_{\text{ref}}(t_{\text{lower}})\| \leq \omega_{\text{thres}}, \quad t_{\text{lower}} \in T_{\text{lower}} \quad (3.12)$$

mit

$$t_0 \leq t_{\text{lower}}, t_{\text{higher}} \leq t_{\text{E}} \quad (3.13)$$

Das Gütekriterium wird mathematisch in vier Anteile aufgespalten, welche jeweils den Wert 0 annehmen, sofern der gemessene dem Referenzwert entspricht:

- Betrag der Beschleunigungen

$$a_{\text{abs}}(\mathbf{x}, \mathbf{u}, t) = \|\mathbf{a}_{\text{ref}}(t) - \mathbf{a}_{\text{P}}(\mathbf{x}, \mathbf{u}, t)\|_2 \cdot \alpha \cdot \gamma \quad (3.14)$$

- Richtung der Beschleunigungen

$$a_{\text{dir}}(\mathbf{x}, \mathbf{u}, t) = (1 - \mathbf{a}_{\text{ref},n}(t) \bullet \mathbf{a}_{\text{P},n}(\mathbf{x}, \mathbf{u}, t)) \cdot \gamma \quad (3.15)$$

- Betrag der Drehraten

$$\omega_{\text{abs}}(\mathbf{x}, \mathbf{u}, t) = \|\boldsymbol{\omega}_{\text{ref}}(t) - \boldsymbol{\omega}_{\text{P}}(\mathbf{x}, \mathbf{u}, t)\|_2 \cdot \beta \cdot (1 - \gamma) \quad (3.16)$$

- Richtung der Drehraten

$$\omega_{\text{dir}}(\mathbf{x}, \mathbf{u}, t) = (1 - \omega_{\text{ref},n}(t) \bullet \omega_{\text{P},n}(\mathbf{x}, \mathbf{u}, t)) \cdot (1 - \gamma) \quad (3.17)$$

mit

$$\mathbf{a}_{\text{P},n} = \frac{\mathbf{a}_{\text{P}}}{\|\mathbf{a}_{\text{P}}\|_2} \quad (\mathbf{a}_{\text{ref},n}, \omega_{\text{P},n}, \omega_{\text{ref},n} \text{ analog}) \quad \text{normierte Vektoren}$$

$$0 < \alpha, \beta, \gamma < 1$$

Abb. 3.1 visualisiert die physikalische Bedeutung des Gütemaßes. Dabei stellen α und β

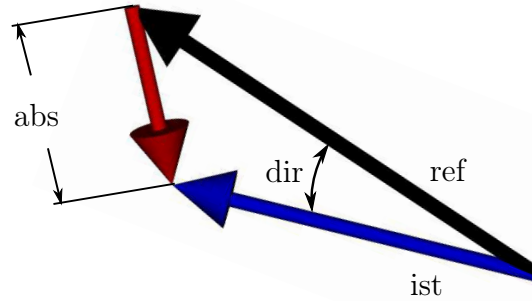


Abbildung 3.1: Schematische Darstellung der Gütemaßanteile für Richtung und Absolutwert („dir“ und „abs“) bezüglich der Referenz- und Istwerte („ref“ bzw. „ist“)

Gewichtungsfaktoren dar, mit denen die Beträge relativ zu den Richtungen der Vektoren skaliert werden können. Diese sind von besonderer Wichtigkeit, da laut der Wahrnehmungsforschung Fehler der Richtungen von Beschleunigung und Drehraten von Simulatorsassessoren stärker wahrgenommen werden als fehlerhafte Beträge der Bewegungen [Zac78]. γ ermöglicht eine Gewichtung zwischen Beschleunigungs- und Drehratenanteil.

Das Ziel der Optimierung ist es, die integrierte Summe der quadrierten Anteile des Gütekriteriums zu minimieren. Bei $a_{\text{abs}}(t)$, $a_{\text{dir}}(t)$ und $\omega_{\text{dir}}(t)$ wird über die gesamte Trajektoriendauer, bei $\omega_{\text{abs}}(t)$ lediglich über die Zeitabschnitte T_{higher} integriert. Dies führt zu folgender Optimierungsfunktion:

$$L(\mathbf{x}, \mathbf{u}) = \min_{\mathbf{u}} \left\{ \int_{t_0}^{t_E} a_{\text{abs}}^2(\mathbf{x}, \mathbf{u}, t) + a_{\text{dir}}^2(\mathbf{x}, \mathbf{u}, t) + \omega_{\text{dir}}^2(\mathbf{x}, \mathbf{u}, t) dt + \int_{T_{\text{higher}}} \omega_{\text{abs}}^2(\mathbf{x}, \mathbf{u}, t) dt \right\} \quad (3.18)$$

3.3.2 Nebenbedingungen

Die Existenz von Nebenbedingung sind der eigentliche Grund für die Konversion der Trajektorienplanung in ein Optimierungsproblem anstatt diese durch simple inverskinematische Umformungen zu bewerkstelligen. Die vorliegenden Einschränkungen führen zu einer Lösung, welche der Referenz nicht exakt folgt, hinsichtlich der Nebenbedingungen jedoch optimal ist. Folgende Einschränkungen werden berücksichtigt:

- Gelenkwinkel:

$$\mathbf{q}_{\min} \leq \mathbf{q}(t) \leq \mathbf{q}_{\max}, \quad t_0 \leq t \leq t_E \quad (3.19)$$

- Gelenkwinkelgeschwindigkeiten:

$$\dot{\mathbf{q}}_{\min} \leq \dot{\mathbf{q}}(t) \leq \dot{\mathbf{q}}_{\max}, \quad t_0 \leq t \leq t_E \quad (3.20)$$

- Gelenkwinkelbeschleunigungen:

$$\ddot{\mathbf{q}}_{\min} \leq \ddot{\mathbf{q}}(t) \leq \ddot{\mathbf{q}}_{\max}, \quad t_0 \leq t \leq t_E \quad (3.21)$$

- Absolutwertbeschränkung der Drehraten unterhalb der Wahrnehmungsschwelle:

$$\|\boldsymbol{\omega}_P(t_{\text{lower}})\| \leq \omega_{\text{thres}} \quad \forall \quad t_{\text{lower}} \in T_{\text{lower}} \quad (3.22)$$

mit

$$\omega_{\text{thres}} = 0,05236 \frac{\text{rad}}{\text{s}} = 3,0^\circ/\text{s} \quad (3.23)$$

- Systemzusammenhang:

$$\mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix}, \quad \mathbf{u} = \ddot{\mathbf{q}} \quad \rightarrow \quad \dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{u} \quad (3.24)$$

Für die Werte der gelenkseitigen Restriktionen sei auf Tab. 2.1 im Abschnitt „Roboterkinematik“ verwiesen.

3.3.3 Zusammenfassung des vorliegenden Optimierungsproblems

Es liegt ein Optimierungsproblem mit einem eingangs- und zustandsabhängigen (\rightarrow vektoriell), nichtlinearen Gütekriterium vor. Außerdem sind Ungleichungsnebenbedingungen der Gelenkwinkel und deren Ableitungen sowie des Absolutwertes der Drehraten unterhalb

der Wahrnehmungsschwelle einzuhalten. Eine weitere Gleichungsnebenbedingung besteht in Form des Systemzusammenhangs:

$$L(\mathbf{x}, \mathbf{u}) = \min_{\mathbf{u}} \left\{ \int_{t_0}^{t_E} a_{\text{abs}}^2(\mathbf{x}, \mathbf{u}, t) + a_{\text{dir}}^2(\mathbf{x}, \mathbf{u}, t) + \omega_{\text{dir}}^2(\mathbf{x}, \mathbf{u}, t) dt + \int_{T_{\text{higher}}} \omega_{\text{abs}}^2(\mathbf{x}, \mathbf{u}, t) dt \right\} \quad (3.25)$$

u.d. NB

$$\mathbf{q}_{\min} \leq \mathbf{q}(t) \leq \mathbf{q}_{\max}, \quad t_0 \leq t \leq t_E \quad (3.26)$$

$$\dot{\mathbf{q}}_{\min} \leq \dot{\mathbf{q}}(t) \leq \dot{\mathbf{q}}_{\max}, \quad t_0 \leq t \leq t_E \quad (3.27)$$

$$\ddot{\mathbf{q}}_{\min} \leq \ddot{\mathbf{q}}(t) \leq \ddot{\mathbf{q}}_{\max}, \quad t_0 \leq t \leq t_E \quad (3.28)$$

$$\|\omega_P(t_{\text{lower}})\| \leq \omega_{\text{thres}} \quad \forall \quad t_{\text{lower}} \in T_{\text{lower}} \quad (3.29)$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix}, \quad \mathbf{u} = \ddot{\mathbf{q}} \quad \rightarrow \quad \dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{u} \quad (3.30)$$

3.4 Quaternionen

Wie in [BOH11] festgestellt, kann es bei der Darstellung von Orientierungen in Form von *Euler-Winkeln* zu Problemen kommen, wenn Orientierungswerte auftreten, welche die Grenze von 360° überschreiten. Aus diesem Grund werden im vorliegenden Fall, wobei teilweise komplizierte Orientierungsverläufe darzustellen sind, *Quaternionen* genutzt. Hierbei ist eine Interpolation zwischen zwei *Quaternionen* immer auf kürzestem Wege und ohne Probleme bei einer 360° -Überschreitung möglich (vgl. [Bel14]).

Zu Ehren ihres Erfinders - *Sir William Hamilton* (siehe [Ham44]) - wird die Menge aller *Quaternionen* mit \mathbb{H} bezeichnet. Als Erweiterung der komplexen Zahlen formen sie bezüglich ihrer Multiplikatoren eine nicht-kommutative Gruppe und sind definiert als

$$\begin{aligned} \vec{Q} &= w + ix + jy + kz = [w, \mathbf{v}], \quad w, x, y, z \in \mathbb{R}, \quad \mathbf{v} = [x, y, z] \\ i^2 &= j^2 = k^2 = -1, \quad ij = k = -ji, \quad \vec{Q} \in \mathbb{H}. \end{aligned} \quad (3.31)$$

Anschaulich stellt eine Rotation via Einheitsquaternion ($w^2 + x^2 + y^2 + z^2 = 1$) eine Drehung um die Achse, die durch den Einheitsvektor $\mathbf{n} \in \mathbb{R}^3$ beschrieben wird, um den Winkel Θ dar:

$$\vec{Q} = [w, \mathbf{v}] = \left[\cos\left(\frac{\Theta}{2}\right), (\mathbf{n} \cdot \sin\left(\frac{\Theta}{2}\right)) \right] \quad (3.32)$$

Mit Hilfe weiterer Rechenvorschriften, für deren genaue Herleitung auf [DKL98] verwiesen sei, ergeben sich folgende wichtige Eigenschaften der *Quaternionen*:

- Multiplikation zweier *Quaternionen*. Damit kann eine Kombination von aufeinanderfolgenden Rotationen dargestellt werden ($\vec{Q}\vec{Q}' \rightarrow$ zunächst Rotation um \vec{Q}' , danach Rotation um \vec{Q}):

$$\vec{Q}\vec{Q}' = [ww' - \mathbf{v} \cdot \mathbf{v}', \mathbf{v} \times \mathbf{v}' + w\mathbf{v}' + w'\mathbf{v}] \quad (3.33)$$

- Darstellung von Einheitsquaternionen in Form einer Rotationsmatrix:

$$\mathbf{R}(\vec{Q}) = \begin{bmatrix} w^2 + x^2 - y^2 - z^2 & -2wz + 2xy & 2wy + 2xz \\ 2wz + 2xy & w^2 - x^2 + y^2 - z^2 & -2wx + 2yz \\ -2wy + 2xz & 2wx + 2yz & w^2 - x^2 - y^2 + z^2 \end{bmatrix} \quad (3.34)$$

- Zusammenhang zwischen Winkelgeschwindigkeit und Orientierung in Form von *Quaternionen*:

$$\frac{\partial \vec{Q}}{\partial t} = \frac{1}{2} \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix} \vec{Q} \quad (3.35)$$

Die Vor- und Nachteile der verschiedenen Möglichkeiten zur Darstellung von Orientierungen sind in Tab. 3.1 zusammenfassend aufgeführt.

Tabelle 3.1: Eigenschaften von Orientierungsdarstellungen mit Rotationsmatrizen, *Euler-Winkeln* und *Quaternionen* (vgl. [PM13])

	Redundante Elemente	Anschaulichkeit	Singuläre Stellen	Rechen- aufwand
Rotationsmatrix	6	gering	nein	klein
Euler-Winkel	0	gut	ja	groß
Quaternionen	1	sehr gering	nein	sehr klein

4 Optimierungsalgorithmen

Optimierungen werden eingesetzt, um für Fragestellungen jeglicher, nicht ausschließlich technischer Natur, die günstigste, schnellste oder gewinnbringendste Lösung zu finden. Dabei greift man häufig auf Erfahrungen und subjektive Eindrücke zurück. Insbesondere bei steigender Anzahl an Einflussfaktoren bietet die mathematische Optimierungstheorie eine fundiertere Entscheidungsgrundlage, vorausgesetzt es kann eine mathematische Formulierung des vorliegenden Problems gefunden werden. Mit Hilfe der Rechenleistung moderner Computer können durch Optimierungsalgorithmen auch exakte Lösungen technischer Probleme gefunden werden. In [BP67] wird das Vorgehen bei der Optimierung als Drei-Stufen-Prozess beschrieben.

- Systemidentifikation und mathematische Modellierung des Systems.
- Festlegung eines objektiven Bewertungskriteriums (Kosten- oder Gewinnfunktional).
- Anwendung des Optimierungsalgorithmus zur Findung einer bezüglich des Gütefunktionals optimalen Lösung durch Minimierung bzw. Maximierung der Kosten- bzw. Gewinnfunktion.

Numerische Optimierung

Ein allgemeines Optimierungsproblem kann in mathematischer Form wie folgt ausgedrückt werden:

$$L(\mathbf{z}) = \min_{\mathbf{z}} l(\mathbf{z}), \quad \mathbf{z} \in \mathbb{R}^n \quad (4.1)$$

u.d. NB

$$h_i(\mathbf{z}) = 0, \quad i = 1, 2, \dots, p \quad (4.2)$$

$$g_j(\mathbf{z}) \leq 0, \quad j = 1, 2, \dots, m \quad (4.3)$$

wobei $l, h_i, g_j : \mathbb{R}^n \rightarrow \mathbb{R}$ mindestens einmal differenzierbare Funktionen sind.

Das Ziel der Optimierung ist es, den Vektor \mathbf{z} zu finden, welcher die Zielfunktion $L(\mathbf{z})$ optimiert, im Fall eines Kostenfunktionals also minimiert, während der Lösungsraum

die Gleichheitsnebenbedingungen \mathbf{h} und Ungleichheitsnebenbedingungen \mathbf{g} berücksichtigt: Abb. 4.1 illustriert das iterative Vorgehen bei numerischen Optimierungen. Dabei werden die Optimierungsvariablen dem Modell übergeben, welches das Gütemaß und die Informationen über die Einhaltung der Nebenbedingungen an den Optimierungsalgorithmus zurückgibt. Iteratives Wiederholen dieses Vorgangs, bis ein zufriedenstellender Wert des Gütemaßes ohne Verletzung der Nebenbedingungen erreicht ist, führt zum Lösungsvektor. Allerdings ist dieser Lösungsvektor meist nicht uneingeschränkt und global als optimal zu bezeichnen. Vielmehr stellt er die unter Berücksichtigung von Fehlerquellen, Parameterwahl und Lösungsalgorithmus vergleichsweise beste Lösung des Problems dar. Mögliche Fehlerquellen sind u.A. die Konvergenz in lokale Minima, Fehlparametrisierung der Kriterien und eine ungünstige Wahl der Startlösung.

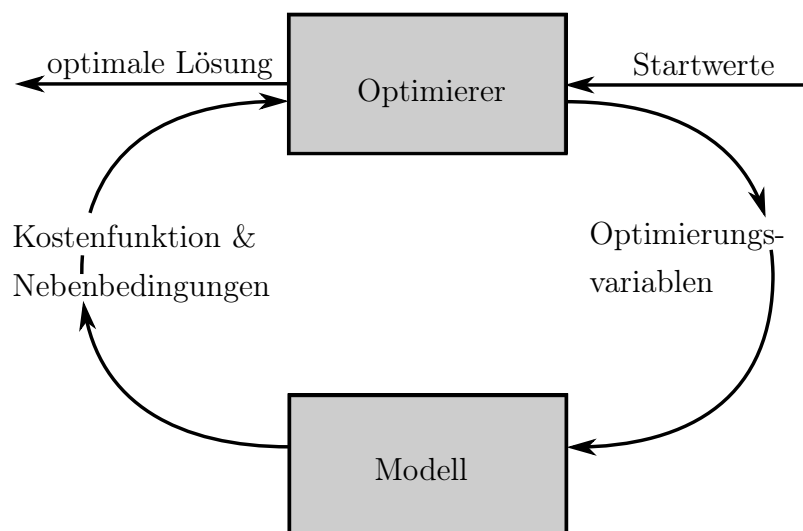


Abbildung 4.1: Vereinfachtes Schema des allgemeinen, iterativen Vorgehens zur Lösung von Optimierungsaufgaben.

Arten von Optimierungsproblemen

Ausgehend von der Art der Optimierungsvariablen, der Gütefunktion und der Nebenbedingungen können Optimierungsprobleme in verschiedene Kategorien unterteilt werden (vgl. [Diw08], [Mer03]):

- **Lineare Programmierung (LP):**
 - Gütekriterium linear
 - Nebenbedingungen linear
 - Optimierungsvariablen skalar und kontinuierlich

- **Ganzzahlige Programmierung (IP):**
 - Gütekriterium linear
 - Nebenbedingung linear
 - Optimierungsvariablen skalar und ganzzahlig
- **Nichtlineare Programmierung (NLP):**
 - Gütekriterium mindestens quadratisch
 - Nebenbedingung nichtlinear
 - Optimierungsvariablen skalar und kontinuierlich
 - Häufigste Formen: linear beschränkte Optimierung, quadratische Optimierung
- **Kombinatorische Optimierung:**
 - Mögliche Lösungen sind diskrete Mengen oder können zu solchen reduziert werden
 - Häufigste Formen: Permutations-, Reihenfolgen-, Zuordnungs-Probleme, Finden von Teilmengen
- **Mehrgrößenoptimierung:**
 - Mehrere Gütekriterien werden berücksichtigt
 - Lösung ist „Pareto-optimal“, wenn keines der Gütekriterien verbessert werden kann, ohne ein anderes zu verschlechtern
- **Dynamische Programmierung:**
 - Findung einer Lösung komplexer Optimierungsprobleme durch Zerlegung in gleichartige, kleinere Teilprobleme
 - Geprägt durch Richard Bellman → Bellman'sches Optimalitätsprinzip (siehe [\[Bel57\]](#))
- **Variationsprobleme:**
 - Es wird nach einer Funktion gesucht, welche das Gütefunktional minimiert/maximiert

Bei der im Abschnitt „Zusammenfassung des vorliegenden Optimierungsproblems“ geschilderten Optimierung handelt es sich demzufolge um eine nichtlineare Programmierung mit nichtlinearen Nebenbedingungen. Durch die Addition der Quadrate der einzelnen Gütemaßbestandteile wird das Mehrgrößenproblem auf ein nichtlineares Problem mit einem Gütekriterium reduziert.

Im Folgenden werden in Frage kommende Optimierungsalgorithmen näher beleuchtet und auf deren Tauglichkeit bezüglich der konkret vorliegenden Aufgabe untersucht. Im Voraus kann bereits der BFGS (Broyden-Fletcher-Goldfarb-Shanno)-Algorithmus als untauglich deklariert werden, da hierbei weder Gleichungs- noch Ungleichungsnebenbedingungen berücksichtigt werden können.

4.1 Downhill Simplex Methode

Bei der Downhill Simplex Methode, auch *Nelder-Mead Simplex Methode* genannt, handelt es sich, im Gegensatz zur „einfachen“ Simplex Methode, um einen Lösungsalgorithmus für nichtlineare Optimierungsprobleme. Durch die Erzeugung einer Folge von Simplexen, deren Durchmesser im Konvergenzfall abnimmt, soll dabei das gesuchte Minimum sukzessive eingegrenzt werden. Als n -Simplex wird in der Geometrie die konvexe Hülle von $(n + 1)$ Eckpunkten bezeichnet.

4.1.1 Eigenschaften

Das *Nelder-Mead*-Verfahren ist im Wesentlichen eine heuristische Methode bzw. eine direkte Suchmethode. Da es keinerlei Informationen über den Gradienten $\nabla l(\mathbf{z})$ bzw. die Hesse-Matrix $\nabla^2 l(\mathbf{z})$ benötigt, gehört es den ableitungsfreien Verfahren an. Demzufolge wird keine Glattheit (beispielsweise zweifache Differenzierbarkeit) der Zielfunktion und der Restriktionen vorausgesetzt. Insbesondere bei Optimierungsaufgaben mit mehreren Optimierungsvariablen ist die Konvergenz des Algorithmus jedoch nicht mehr garantiert.

Bei der vorliegenden Implementierung in der *Dymola Optimization Library* werden Nebenbedingungen mit Hilfe einer exakten L_1 -*Penalty-Funktion* berücksichtigt. Eine solche *Penalty-Funktion* kann wie folgt definiert werden (vgl. [GK02]):

$$P_1(\mathbf{z}, \alpha) := l(\mathbf{z}) + \alpha \sum_{i=1}^p |h_i(\mathbf{z})| + \alpha \sum_{j=1}^m \max\{0, g_j(\mathbf{z})\} \quad (4.4)$$

Vorgehen

Das Vorgehen bei der Downhill Simplex Methode wird in [Fob06] sowie [GK99] folgendermaßen beschrieben und in Abb. 4.2 dargestellt:

- Beginne mit einem Startsimplex
- Suche den Punkt mit dem schlechtesten Gütekriteriumswert (x_S) und entferne ihn
- Reflektiere den entfernten Punkt am Schwerpunkt der restlichen Punkte (x_R)

- Wiederhole den Vorgang

Neben der Reflexion wird das Verfahren durch Expansion und Kontraktion erweitert. Bei der Expansion wird versucht, durch Erhöhung der Schrittweite einen qualitativ noch besseren Punkt als den reflektierten zu finden (x_E). Führt der reflektierte Punkt zu einem schlechteren Ergebnis als der entfernte Punkt, so muss sich im Inneren des Simplex ein besserer Punkt befinden. Durch die Verkleinerung der Schrittweite kann dieser Punkt bei der Kontraktion gefunden werden (x_K).

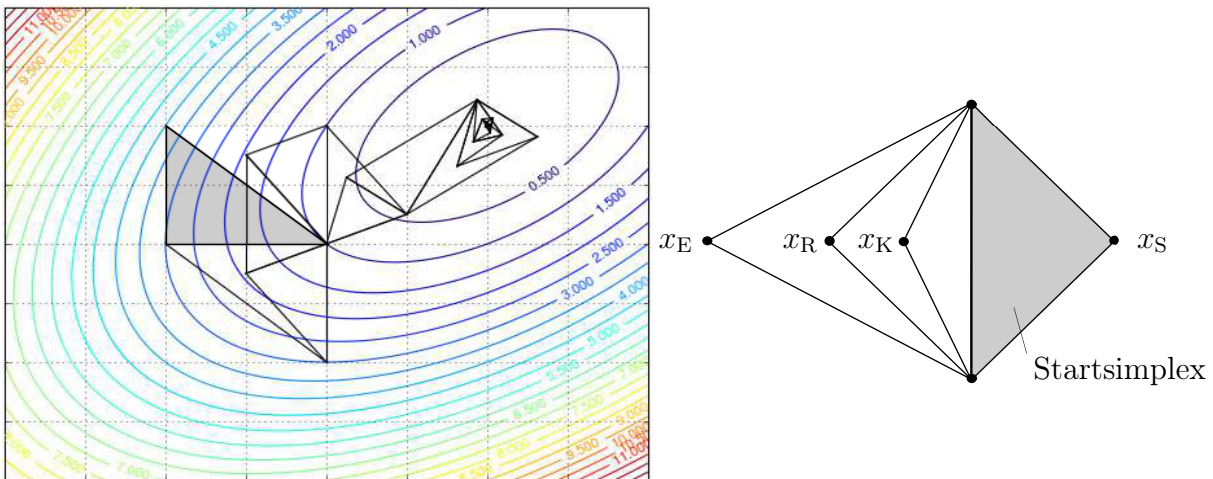


Abbildung 4.2: Verlauf einer Extremwertfindung mit der Downhill Simplex Methode (links) und Operatoren des Verfahrens (rechts); Quellen: [Kon13] (links), eigene Darstellung (rechts).

4.1.2 Tauglichkeit

Die Linearisierung des zugrundeliegenden Systems ist aufgrund des ableitungsunabhängigen Vorgehens nicht nötig, was Vorteile bezüglich der Rechenzeit mit sich bringt. Zudem gilt das Verfahren als recht robust, da z.B. Unstetigkeiten keine numerischen Probleme zur Folge haben. Häufig arbeitet der Algorithmus jedoch ineffizient und konvergiert „nur“ linear. Ferner ist eine Konvergenz in Nebenminima möglich.

4.2 Pattern Search

Der Pattern Search Algorithm (vgl. [HJ61]) ist ebenfalls ein für nichtlineare Zielfunktionen geeigneter Optimierungssolver, welcher auf einer Folge von *exploratory moves* um einen Basispunkt beruht, auf die im Konvergenzfall *pattern moves* folgen.

4.2.1 Eigenschaften

Der Pattern Search Algorithmus ist, ebenso wie die Downhill Simplex Methode, der Klasse der direkten Suchmethoden zuzuordnen. Aufgrund des ableitungsunabhängigen Vorgehens werden keine Differenzierbarkeitsvoraussetzungen an das zugrundeliegende Problem gestellt.

Vorgehen

Die in Abb. 4.3 dargestellten Schritte veranschaulichen das Prozedere des Pattern Search Algorithmus (vgl. [LC07]).

4.2.2 Tauglichkeit

In der Praxis bewährt sich die Pattern Search Methode vor allen Dingen bei der Lokalisation von Minima auf Hyperflächen mit „scharfen“ Tälern, bei der klassische Techniken an ihre Grenzen stoßen und eine Annäherung an die Minima nur langsam vonstatten geht (siehe [HJ61]). Um sich nicht aus einem vorliegenden Tal herauszubewegen, finden bei direkten Suchmethoden, welche mit simplen Schrittbewegungen arbeiten, sehr geringe Schrittweiten Anwendung. Aufgrund dessen ist die Pattern Search Methode, wie die Downhill Simplex Methode, recht ineffizient. Trotzdem können Minima auch bei Unstetigkeiten der zugrundeliegenden Funktion robust angenähert werden, während eine Konvergenz in Nebenminima ebenfalls nicht auszuschließen ist.

4.3 SQP - Sequential Quadratic Programming

Die Sequentielle Quadratische Programmierung ist ein iteratives Lösungsverfahren für nichtlineare, beschränkte Optimierungsprobleme, welches das nichtlineare Problem in jedem Iterationsschritt durch ein sogenanntes *Quadratisches Programm* approximiert (siehe [Ex12]).

4.3.1 Eigenschaften

Der SQP-Algorithmus wird auch als *Lagrange-Newton-Verfahren* bezeichnet und arbeitet mit Gradienten der zugrundeliegenden Funktion. SQP-Verfahren können als Erweiterung der *Newton-Methode* zur Lösung des Systems der *Karush-Kuhn-Tucker-Bedingungen* (KKT) aufgefasst werden. Liegen Beschränkungen vor, werden diese bei der Lösung mit der SQP-Methode linear approximiert. Zur Anwendbarkeit des SQP-Algorithmus müssen die Zielfunktion wie auch die Funktionen der Nebenbedingungen zweifach kontinuierlich

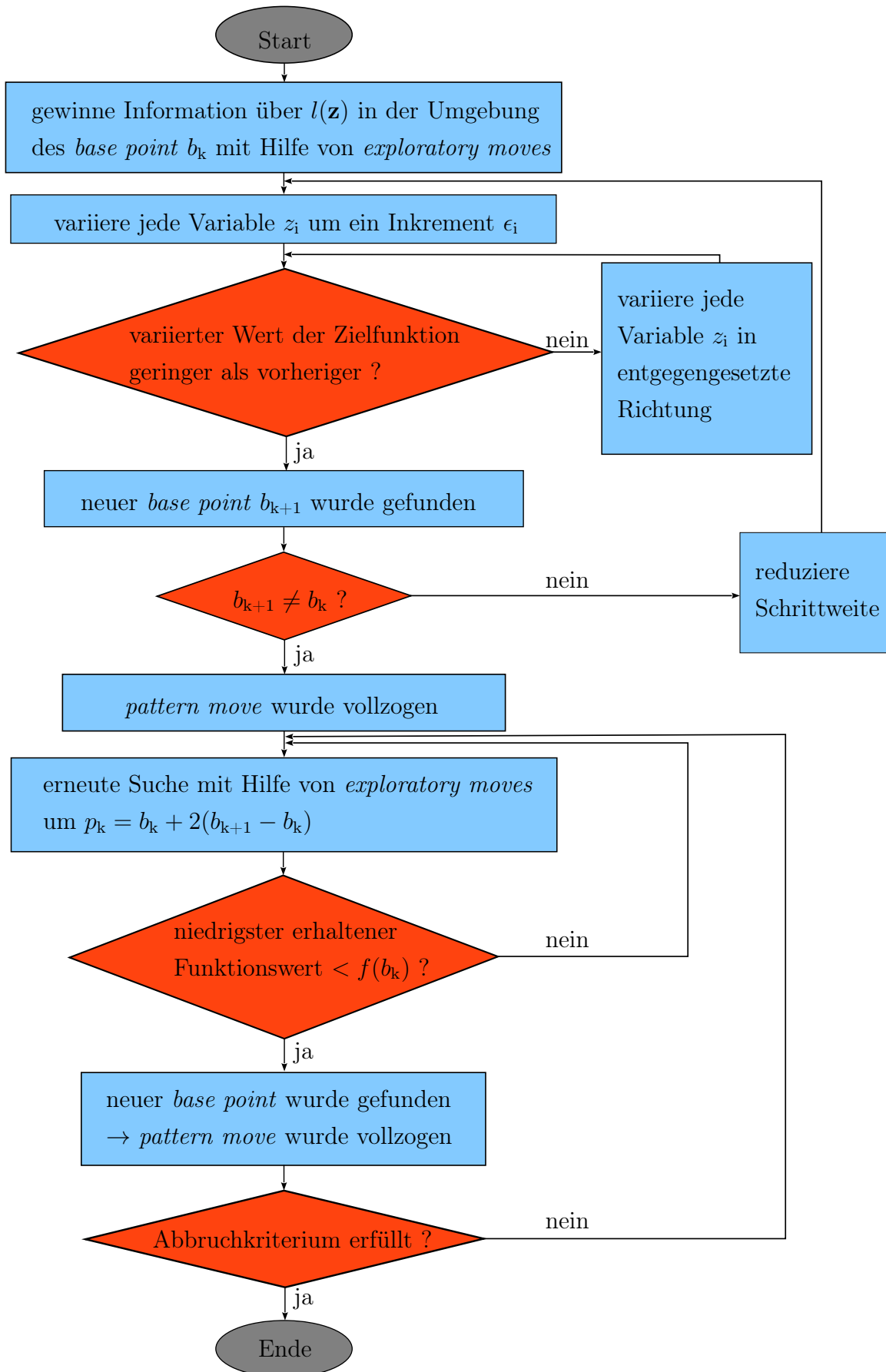


Abbildung 4.3: Ablaufdiagramm des Pattern Search Algorithmus

differenzierbar sein. Für SQP-Verfahren kann lokale, superlineare Konvergenz nachgewiesen werden.

Vorgehen

Ausgangspunkt des Verfahrens ist ein Startpunkt z_k , der „möglichst nahe“ am Optimum liegen soll. Ein verbesserter, näher am Optimum liegender Punkt z_{k+1} wird definiert als

$$z_{k+1} = z_k + p_k \quad (4.5)$$

wobei p_k die Lösung eines quadratischen Teilproblems ist.

Dieses *quadratische Programm (QP)* kann folgendermaßen beschrieben werden:

$$\min \quad \nabla l(z_k)^T \mathbf{p} + \frac{1}{2} \mathbf{p}^T \mathbf{H}_k \mathbf{p} \quad (4.6)$$

u.d. NB

$$h_i(z_k) + \nabla h_i(z_k)^T \mathbf{p} = 0, \quad i = 1, 2, \dots, p \quad (4.7)$$

$$g_j(z_k) + \nabla g_j(z_k)^T \mathbf{p} \leq 0, \quad j = 1, 2, \dots, m \quad (4.8)$$

$$(4.9)$$

mit

$$\mathbf{H}_k = \nabla_{zz}^2 \mathcal{L}(z_k, \lambda_k) \quad \text{Hessematrix der Lagrangefunktion (vgl. [Obe12])} \quad (4.10)$$

Bei positiver Definitheit von \mathbf{H}_k ist das QP streng konvex und hat demzufolge immer eine eindeutige Lösung.

Diese Vorgehensweise wird iterativ wiederholt, wobei die Lösungsfolge quadratisch gegen eine (lokal) optimale Lösung konvergiert. Neben einer Folge von Näherungen des Lösungsvektors erzeugt der SQP-Algorithmus den optimalen *Lagrange-Multiplikator* λ .

4.3.2 Tauglichkeit

SQP ist eine äußerst mächtige und effiziente Lösungsmethode für komplexe, nichtlineare Optimierungsprobleme mit nichtlinearen Gleichungs- und Ungleichungsnebenbedingungen und spielt in Industrie und Forschung eine sehr bedeutende Rolle.

Gegen den SQP-Algorithmus spricht, dass die Linearisierung von Gleichungsnebenbedingungen innerhalb des QP zu Inkonsistenzen führen kann. Informationen zweiter Ordnung sind dann lediglich in der Hessematrix der Lagrangefunktion enthalten und der nächste berechnete Schritt bewirkt keine Verbesserung des Gütemaßes, wird also nicht

übernommen (vgl. [Exl12]). Aufgrund dessen scheitert der Algorithmus häufig daran, ausgehend von einer Startlösung z_k zu beginnen, welche „weit“ vom Optimum entfernt liegt. Aufgrund der Verwendung von Gradienten der zugrundeliegenden Funktion können ferner numerische Probleme auftreten, beispielsweise bei verrauschten Signalen. Die daraus resultierenden, falschen Suchrichtungen verlangsamen die Konvergenz des Algorithmus und können diese sogar gänzlich verhindern.

4.4 Genetische Algorithmen

Ein Genetischer Algorithmus (GA) ist laut [Pop05] eine iterative Methode zur Lösung von beschränkten und unbeschränkten Optimierungsproblemen, bei der auf Basis einer „natürlichen Auslese“ ein biologischer Evolutionsprozess nachempfunden wird, wobei jeweils das „beste“ Individuum einer Population (\rightarrow die „beste“ Lösung) „überlebt“. Genetische Algorithmen werden bevorzugt bei Problemen mit diskontinuierlichen, nicht-differenzierbaren, stochastischen oder hochgradig nichtlinearen Zielfunktionen und bei großer Anzahl von Optimierungsvariablen angewendet, bei denen klassische Optimierungsalgorithmen versagen.

4.4.1 Eigenschaften

Als Untergruppe der *Evolutionären Algorithmen* werden die GA häufig als Metaheuristiken bezeichnet. Beschränkungen werden bei der vorliegenden Implementierung durch *Penalty-Funktionen* realisiert.

Vorgehen

Das iterative Vorgehen zur Annäherung an eine „optimale“ Lösung wird in Abb. 4.4 visualisiert (vgl. [Pop05]). Das Ziel besteht darin, in jedem Iterationsschritt aus einer zufälligen Menge an Punkten diejenigen zu finden, welche die besten Werte der Zielfunktion (\rightarrow *fitness*) liefern. Auf Basis dieser sogenannten *Eltern* wird daraufhin eine neue Generation der Population gebildet und das Verfahren wird wiederholt.

4.4.2 Tauglichkeit

Optimierungen mit Hilfe von Genetischen Algorithmen sind aufgrund der wiederkehrenden Auswertungen der Gütefunktion sehr ineffizient, äußerst umfangreich und benötigen große Rechenressourcen. Häufig werden GA deshalb lediglich dafür genutzt, eine grobe Lösung des Problems zu finden, woraufhin ein weiterer Algorithmus zur Findung der exakten

Lösung eingesetzt wird. Auch bei GA kann keine Konvergenz in ein globales Optimum garantiert werden. Da die Parameter der Individuen jedoch in jedem Iterationsschritt zufällig gewählt werden, ist die Gefahr verglichen mit gradientenbasierten Verfahren als gering einzustufen. Des Weiteren sind Genetische Algorithmen recht robuste Verfahren, was wiederum auf der Unabhängigkeit jeglicher Gradienten basiert.

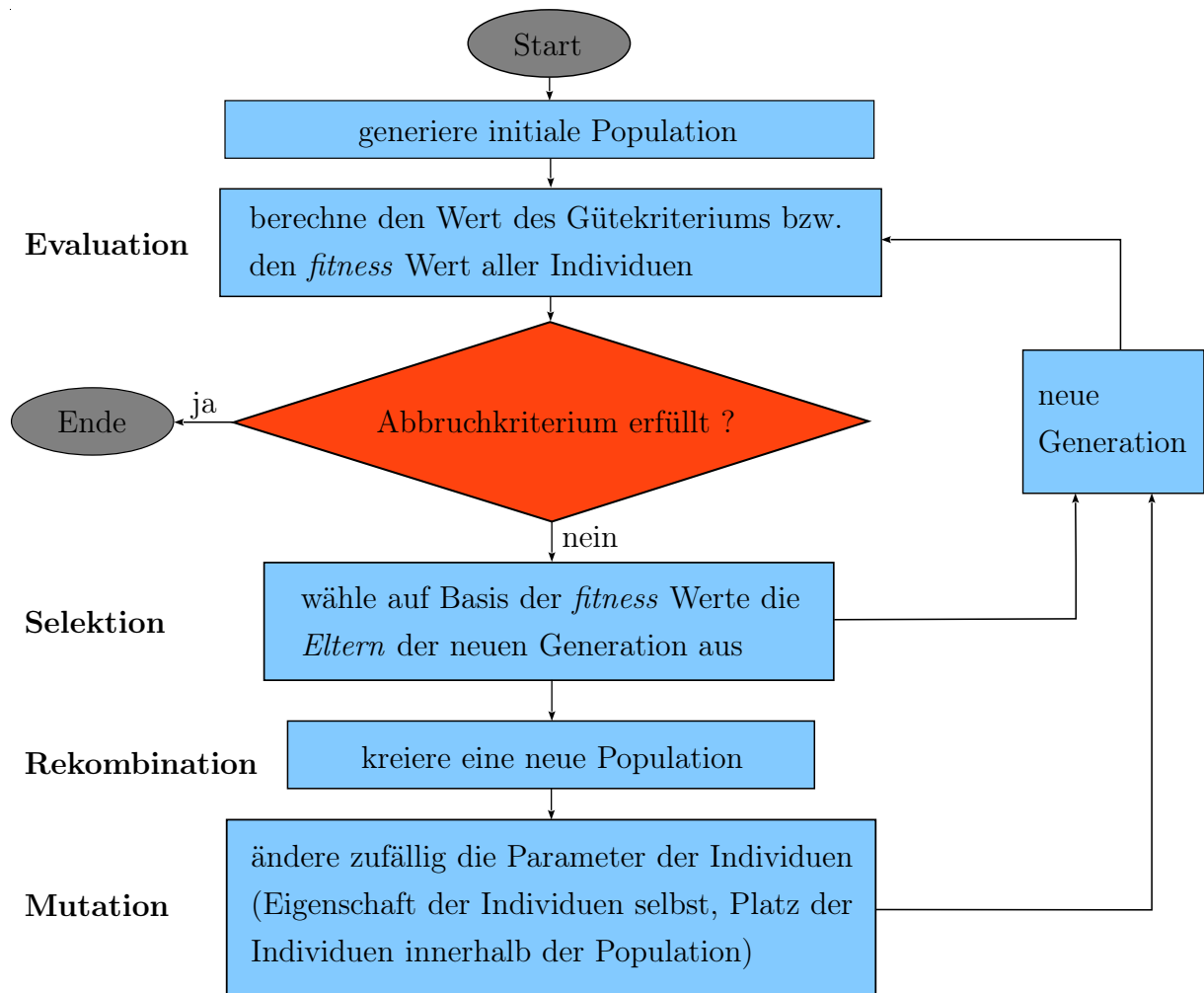


Abbildung 4.4: Ablaufdiagramm eines Genetischen Algorithmus

5 Offline - Trajektorienoptimierung: Methoden, Ergebnisse und Vergleiche

Im folgenden Kapitel werden zunächst die vorgestellten Optimierungsalgorithmen unter identischen Rahmenbedingungen verglichen. Daraufhin werden mit der Verwendung der Online-Lösung als Starttrajektorie, der Parallelisierung des Berechnungsprozesses, dem Tuning von bis dahin fixen Parametern und der Erhöhung der Stützstellenanzahl Methoden und Schritte vorgestellt, die zu einem „optimalen“ Ergebnis hinsichtlich der Performanz und der Güte der erhaltenen Trajektorie führen sollen.

Alle Algorithmen und Verfahren sind unabhängig von dem zugrundeliegenden technischen System und den Referenztrajektorien, wodurch das Vorgehen neben Simulationen von Flug- oder Fahrzeugmanövern auf jegliche Bewegungsszenarien übertragbar ist.

5.1 Implementierung und Voraussetzungen

Die Implementierung des zu optimierenden Systems erfolgt komplett in der Modellierungs- und Simulationsumgebung *Dymola*, welche auf der gleichungsorientierten Modellierungssprache *Modelica* basiert.

Das kinematische Modell des Roboters, die systematischen Zusammenhänge ($\mathbf{x} = [\mathbf{q}, \dot{\mathbf{q}}]^T$, $\mathbf{u} = \ddot{\mathbf{q}}$), die Filterung und Trennung des Drehratensignals, die Berechnung des Gütemaßes sowie sämtliche gelenkwinkelseitigen Nebenbedingungen werden in *Dymola* modelliert und zu einem Kompletmodell zusammengefügt. Die Systemeingänge ($\ddot{q}_1 \dots \ddot{q}_7$) und Ausgänge (alle - an dieser Stelle noch nicht quadrierten - Summanden des Gütemaßes und die normierten Nebenbedingungen) liegen ausschließlich in skalarer Form vor. Dabei werden Beschränkungen derart normiert, dass bei Einhaltung der Grenzen Werte < 1 erzeugt werden. Werte ≥ 1 signalisieren eine Verletzung der Nebenbedingungen. Abb. 5.1 zeigt das vereinfachte Modell des zugrundeliegenden Systems.

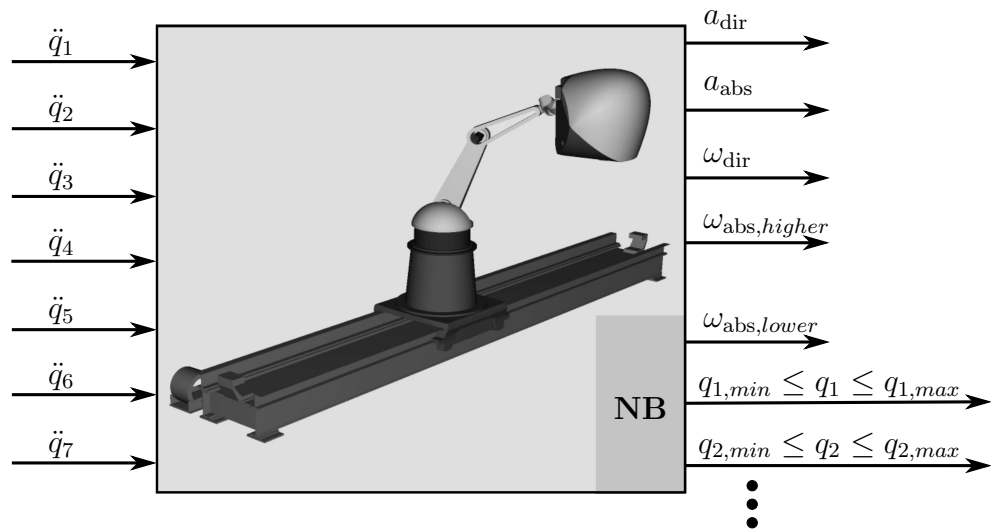


Abbildung 5.1: Modell des simulierten Roboters mit skalarer Ein- und Ausgängen als zugrundeliegendes System des Optimierungsproblems. Die Berechnung der Gütemaßsummanden sowie der normierten Signale der Nebenbedingungen erfolgt innerhalb des Modells.

5.1.1 Dymola Optimization Library

Aufgrund der Tatsache, dass die Modellierung des Optimierungsproblems vollständig in *Dymola* erfolgt und die *Optimization Library* in *Dymola* über die vorgestellten Solver verfügt, wird die GUI dieser Library zur Ausführung der folgenden Optimierungen genutzt.

Die von A. Pfeiffer am Institut für Systemdynamik und Regelungstechnik des *DLR* entwickelte Optimierungsbibliothek bietet zahlreiche numerische Optimierungsalgorithmen, welche unabhängig von dem zu optimierenden System implementiert sind (siehe [Pfe12]). Für den Optimierungsprozess mit Hilfe der Optimierungsbibliothek in *Dymola* stehen neben der Auswahl des Optimierungsalgorithmus folgende Parameter, Eigenschaften und Konfigurationen zur Verfügung:

- **Auswahl der Optimierungsvariablen:**

Aus allen skalaren Systemeingängen können diejenigen gewählt werden, welche dem gewählten Algorithmus zur Optimierung zur Verfügung stehen sollen.

- **Näherung durch B-Spline:**

Im Bereich der Trajektorienoptimierung werden die diskret abgetasteten Verläufe der Optimierungsvariablen durch *B-Splines* angenähert. Dabei kann die Anzahl der

Stützstellen (\rightarrow *de-Boor-Punkte*) an die Länge und die Beschaffenheit des Referenzsignal angepasst werden, wozu auf den Abschnitt „Vergleich der Optimierungsalgorithmen mit der Online-Lösung als Starttrajektorie“ verwiesen sei. Mit der Approximation der Signale in Form von *B-Splines* geht in den meisten Fällen eine Reduktion des Rechenaufwandes einher.

Beispiel:

Berücksichtigt man n Eingangsgrößen mit $k + 2$ Abtastschritten (inkl. Anfangs- und Endpunkt), liegen dem Algorithmus $n \cdot (k + 2)$ Optimierungsvariablen vor. Bei einer Trajektorienlänge von 12s, $n = 7$ Eingangsgrößen und einer Abtastrate von 12ms stehen demzufolge bereits $7 \cdot 1002 = 7014$ Optimierungsvariablen zur Verfügung. Nach einer Spline-Approximation mit $k = 21$ Stützstellen und den zusätzlichen Stützpunkten am Beginn sowie am Ende des Signals ergeben sich lediglich $7 \cdot 23 = 161$ Variablen. Dies entspricht einer Reduktion um 97,7%.

- **Parallelisierung des Rechenprozesses:**

Standardmäßig erfolgt die Berechnung des Optimierungsprozesses auf einem einzigen Rechenkern. Um die Rechenzeit zu verringern kann der Optimierungsprozess parallelisiert werden, d.h. auf mehrere Rechenkerne verteilt werden. Dies kann - für kleinere Probleme - auf dem lokalen Rechner, für komplexere Aufgaben auf einem Cluster-Server realisiert werden. Die Parallelisierung des Prozesses ist jedoch nicht für alle Optimierungsalgorithmen sinnvoll bzw. möglich: Bei der Downhill Simplex Methode sowie beim Pattern Search Algorithmus kann keine Parallelisierung erfolgen. Lediglich das SQP-Verfahren und die Genetischen Algorithmen bieten einer Prozessaufspaltung eine sinnvolle Grundlage.

- **Integrationsverfahren:**

Die wichtigsten in Frage kommenden Integrationsalgorithmen sind jene von *Euler* und *Dassl*. Sie unterscheiden sich grundlegend in deren Schrittweite, welche beim *Euler*-Algorithmus fest, bei der *Dassl*-Methode variabel ist. Während Methoden mit fester Schrittweite eine voraussagbare Berechnungszeit garantieren, können Algorithmen mit adaptiver Schrittweite genauere Ergebnisse erzielen, wobei die Schrittweite und somit die benötigte Rechenzeit maßgeblich vom zugrundeliegenden System und der gewählten Fehlertoleranz abhängt (siehe [DS13]). In der vorliegenden Arbeit wurde aufgrund der erhöhten Genauigkeit ausschließlich die *Dassl*-Methode verwendet.

- **Toleranzen, Abbruchkriterien:**

Für die Optimierungsprozesse kann neben einer maximalen Anzahl an Auswertungen des Gütekriteriums auch eine minimale Fehlertoleranz des Ergebnisses angegeben werden, bei der der Optimierungsprozess abgebrochen wird.

- **Tuner:**

Neben der Variation der *B-Spline*-Parameter besteht die Möglichkeit der Manipulation von Systemparametern. Hierbei kann durch Variation der Startkonfiguration oder Gewichtungswerten eine Erweiterung der Freiheitsgrade für die Optimierung erzielt werden. Speziell bei Gewichtungsparametern ist jedoch zu beachten, dass eine komplette Nicht-Berücksichtigung des gewichteten Terms verhindert wird (Gewichtung sollte nicht null werden → z.B. durch wechselseitige Verwendung von Gewichtungen mit ξ und $(1 - \xi)$ oder $\frac{1}{\xi}$).

5.1.2 Referenzszenario: Überholmanöver

Zunächst wird als beispielhaftes Referenzszenario ein Überholmanöver eines PKW gewählt, wie es in Abb. 5.2 zu sehen ist. Als Referenztrajektorie liegen die im 12ms-Takt abgetasteten Beschleunigungen und Drehraten vor, welche vom Fahrzeuginsassen während eines doppelten Spurwechsels bei einer Fahrtgeschwindigkeit von $v_{\text{PKW}} = 10,0 \frac{\text{m}}{\text{s}}$ wahrgenommen werden. Im Fall des Überholmanövers dominieren Beschleunigungen in horizontaler Richtung.

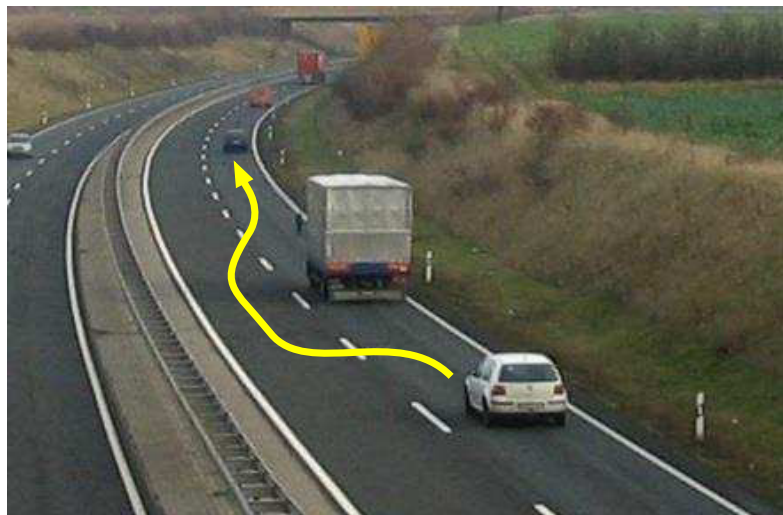


Abbildung 5.2: Referenzszenario: Überholmanöver bzw. doppelter Spurwechsel eines PKW; Quelle: de.wikipedia.org

5.2 Vergleich der Optimierungsalgorithmen

Ausgehend von der Nulllösung als Starttrajektorie und auf Basis der Referenztrajektorie des PKW-Überholmanövers werden im folgenden Abschnitt die vier vorgestellten Optimierungsmethoden hinsichtlich ihrer Performanz und der Güte des Ergebnisses miteinander verglichen und bewertet.

Rahmenbedingungen

Für den Vergleich der Algorithmen werden die in Tab. 5.1 dargestellten, einheitlichen Parameter verwendet.

Tabelle 5.1: Optimierungsparameter für vergleichbare Optimierungsdurchläufe mit verschiedenen Algorithmen.

Parameter	Wert/Auswahl
Stützstellen pro Eingangsvariable	10
Fehlertoleranz des Optimierers	1E – 6
Max. Gütemaßevaluationen	1000
Parallelisierung	keine
α	0,3
β	0,3
γ	0,8

Vergleichsergebnisse

Die Optimierungsprozesse der vier Solver unter den in Tab. 5.1 genannten Voraussetzungen ergeben die in Tab. 5.2 aufgeführten Ergebnisse. Sowohl das Simplex-Verfahren als auch die Pattern Search Methode liefern eine, im Rahmen der zur Verfügung stehenden Anzahl an Evaluationen, zufriedenstellende Verbesserung des Gütemaßes und benötigen dafür ähnliche, im Vergleich zu den weiteren Verfahren geringe, Berechnungszeiten. Beide Methoden sind ableitungsfreie Verfahren und haben keinerlei Probleme damit, von einem verhältnismäßig weit vom „Optimum“ entfernten Punkt ausgehend zu starten.

Mit dem Genetischen Algorithmus, der ebenfalls ableitungsunabhängig arbeitet, gelingt zwar eine geringe Verbesserung des Gütewertes und somit ein Verlassen des Startpunktes, um eine brauchbare Lösung zu erhalten wären jedoch deutlich mehr Evaluationen

Tabelle 5.2: Vergleich der Optimierungssolver Simplex Algorithmus, SQP, Pattern Search und Genetischer Algorithmus hinsichtlich des Gütekriteriums und der Performanz. Der Patter Search Algorithmus liefert die größte Verbesserung des Gütemaßes. Aufgrund der „ungünstigen“ Startlösung scheitert der SQP Algorithmus;
Starttrajektorie: Nullsignal; * manueller Berechnungsabbruch.

Algorithmus	Gütemaß initial: 10,271	Änderung in [%]	Rechenzeit in [s]
Simplex Methode	8,740	−14,9	4336,1
Pattern Search	5,403	−47,4	5509,7
SQP-Verfahren	10,271	±0,0	139707.7*
Genetischer Algorithmus	9,948	−3,1	5748,9

vonnöten. Im vorliegenden Fall wurde eine Populationsgröße von 5 bei einer Generationsanzahl von 200 verwendet.

Besonders auffällig ist das Ergebnis des SQP-Verfahrens. Hier tritt der erwähnte Fall ein, dass es dem gradientenbasierten Algorithmus auch nach langer Rechenzeit (Abbruch nach 38,8h) nicht gelingt, den Startpunkt in einer sinnvollen Richtung zu verlassen. Auch die Rechenzeit einer einzelnen Evaluation dauert deutlich länger als bei den ableitungsfreien Verfahren, was aufgrund der Gradientenberechnung des kompletten, zugrundeliegenden Systems in jedem Berechnungsschritt zu erwarten war.

5.3 Verwendung der Online-Lösung als Starttrajektorie

Um den SQP-Algorithmus trotz der Startproblematik einsetzen zu können, wird im folgenden Abschnitt die Lösung aus der Online-Optimierung des gewählten Beispielmanövers als Starttrajektorie der Offline-Optimierung zum Einsatz gebracht. Neben der erhofften Einsetzbarkeit des SQP-Verfahrens aufgrund der veränderten Startpunktwahl wird von diesem Vorgehen eine Verringerung der Rechenzeit erwartet, da die Startlösung dem erwünschten, „optimalen“ Ergebnis „ähnlicher“ ist.

5.3.1 Übertragung der Online-Lösung zur Verwendung als Starttrajektorie

Die Lösung des Online-Verfahrens liegt zunächst in Form von diskreten, im 12ms-Takt abgetasteten Verläufen als `.txt`-Datei vor. Ziel ist es, aus diesen Werten die *de-Boor-Punkte* desjenigen *B-Splines* zu generieren, welcher abhängig von der gewählten Zahl der Stützstellen die Signale bestmöglich approximiert (\rightarrow *curve fitting*) und diese der *Optimization Library* in Form von `.mat`-Files zu übergeben. Abb. 5.3 veranschaulicht das Vorgehen schematisch. Für die Verwendung als Startlösung werden sowohl die Abszissenwerte der k Knotenpunkte als auch die $(k + 2)$ Ordinatenwerte der *de-Boor-Punkte* (an allen k Stützstelle sowie jeweils am Start- und Endpunkt) verlangt.

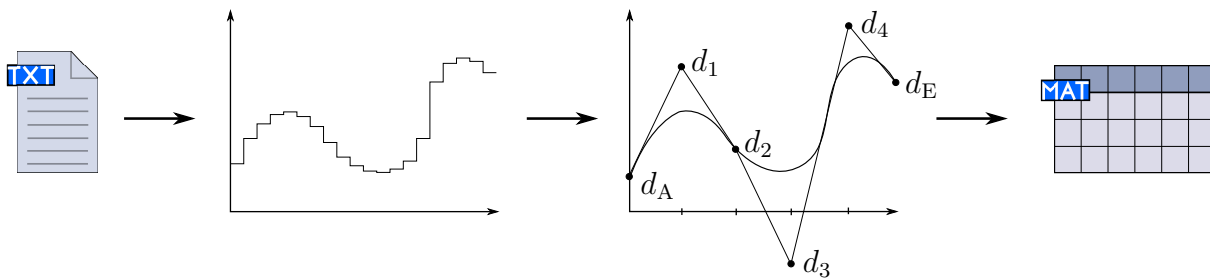


Abbildung 5.3: Schema der Implementierung zur Verwendung der Online-Lösung als Starttrajektorie. Abgetastete Werte aus `.txt`-Datei werden durch *B-Splines* approximiert und in Form von `.mat`-Files an die *Optimization Library* übergeben.

Anzugebende Parameter sind die gewünschte polynomiale Ordnung der *B-Splines*, die Anzahl der Stützstellen, die Gesamtdauer des Trajektoriensignals sowie die Abtastrate des vorliegenden Signals aus der Online-Optimierung.

5.3.2 Vergleich: Online \leftrightarrow Offline - Optimierung

Zunächst wird durch einen *B-Spline* mit $k = 500$ Stützstellen eine für den vorliegenden Fall ausreichend exakte *B-Spline*-Approximation der Online-Lösung erzeugt. Um den Rechenaufwand zu verringern, wird daraufhin nach einer minimalen Anzahl an Stützstellen gesucht, welche die Online-Lösung nach wie vor mit akzeptablem Informationsverlust darstellt. Dafür werden die Gelenkwinkelbeschleunigungssignale aus dem Online-Verfahren mit Hilfe einer *Fast-Fourier-Transformation (FFT)* analysiert. Es ergeben sich Eigenfrequenzen der Signale bei 0,28Hz, 0,45Hz, 0,65Hz und 0,77Hz. Die maximale Eigenfrequenz liegt somit bei $f_{\max} = 0,77\text{Hz}$. Laut dem *Nyquist-Shannon'schen Abtasttheorem* muss ein Signal, dessen höchste vorkommende Eigenfrequenz f_{\max} ist, mit einer Abtastfrequenz

$f_{\text{Abtast}} \geq 2 \cdot f_{\text{max}}$ abgetastet werden, um eine exakte Rekonstruktion aus dem diskretisierten Signal zu ermöglichen, einen Informationsverlust also zu verhindern. In der Praxis wird sicherheitshalber häufig ein etwas erhöhter Faktor verwendet. Im vorliegenden Fall wird dieser zu 2,2 gewählt, um bei Erhöhung der Sicherheit die Datenmenge nicht unnötig zu vergrößern. Das Theorem führt zu folgenden Werten:

$$f_{\text{Abtast}} \geq 2,2 \cdot f_{\text{max}} = 2,2 \cdot 0,77\text{Hz} = 1,694\text{Hz} \quad (5.1)$$

Bei der betrachteten Dauer des Manövers von $t_{\text{ges}} = 12\text{s}$ führt dies zu einer Anzahl an Stützstellen von

$$k \geq t_{\text{ges}} \cdot f_{\text{Abtast}} = 12\text{s} \cdot 1,694\text{Hz} = 20,328 \quad (5.2)$$

Demzufolge wird eine Stützstellenanzahl von $k = 21$ gewählt, um bei möglichst geringem Rechenaufwand eine verlustarme Approximation der Online-Lösung zu erhalten.

Zum Vergleich zwischen Online- und Offline-Optimierung wird nun der SQP-Algorithmus eingesetzt. Mit denselben Parametern wie in Tab. 5.1, abgesehen von der Anzahl der Stützstellen, erhält man die in Tab. 5.3 dargestellten Gütemaße.

Tabelle 5.3: Vergleich der Gütemaße von Online- und Offline-Optimierung;
Algorithmus: SQP.

Vorgehen	k	Gütemaß
Online-Lösung	500	6,795
Online-Lösung, approximiert	21	868,742
Offline-Lösung Startpunkt: Online-Lösung	21	0,396 (−94,17% vgl. mit exakter Online-Lsg.) (−99,95% vgl. mit approx. Online-Lsg.)

Zunächst fällt auf, dass das Gütemaß der approximierten Online-Lösung eine enorme Verschlechterung bezüglich der exakten Online-Lösung darstellt. Aufgrund der folgenden Punkte ist die Verwendung der approximierten Online-Lösung als Starttrajektorie trotzdem sinnvoll:

- Wie unter dem Punkt „Näherung durch B-Splines“ im Abschnitt „Dymola Optimization Library“ erläutert, bewirkt die Approximation durch *B-Splines* eine maßgebliche Reduktion des Berechnungsaufwandes.
- Es wird eine Starttrajektorie benötigt, von der aus ein Start des SQP-Algorithmus, anders als im Fall der Nulllösung, möglich ist. Bei Verwendung der approximierten Online-Lösung gelingt dies problemlos.

- Abb. 5.4 zeigt, dass durch die Approximation der Online-Lösung durch *B-Splines* mit $k = 21$ Stützstellen keineswegs eine exakte Abbildung des Originalsignals erreicht wird (beispielhaft für Achse 5 und 6 dargestellt). Vor allem hochfrequente Anteile des Signals können aufgrund der niedrigen Anzahl an Stützstellen nicht exakt nachgebildet werden. Festzuhalten und für die Verwendung als Starttrajektorie entscheidend ist jedoch, dass die Beschaffenheit und Charakteristik des Signals in ausreichendem Maße dargestellt wird.

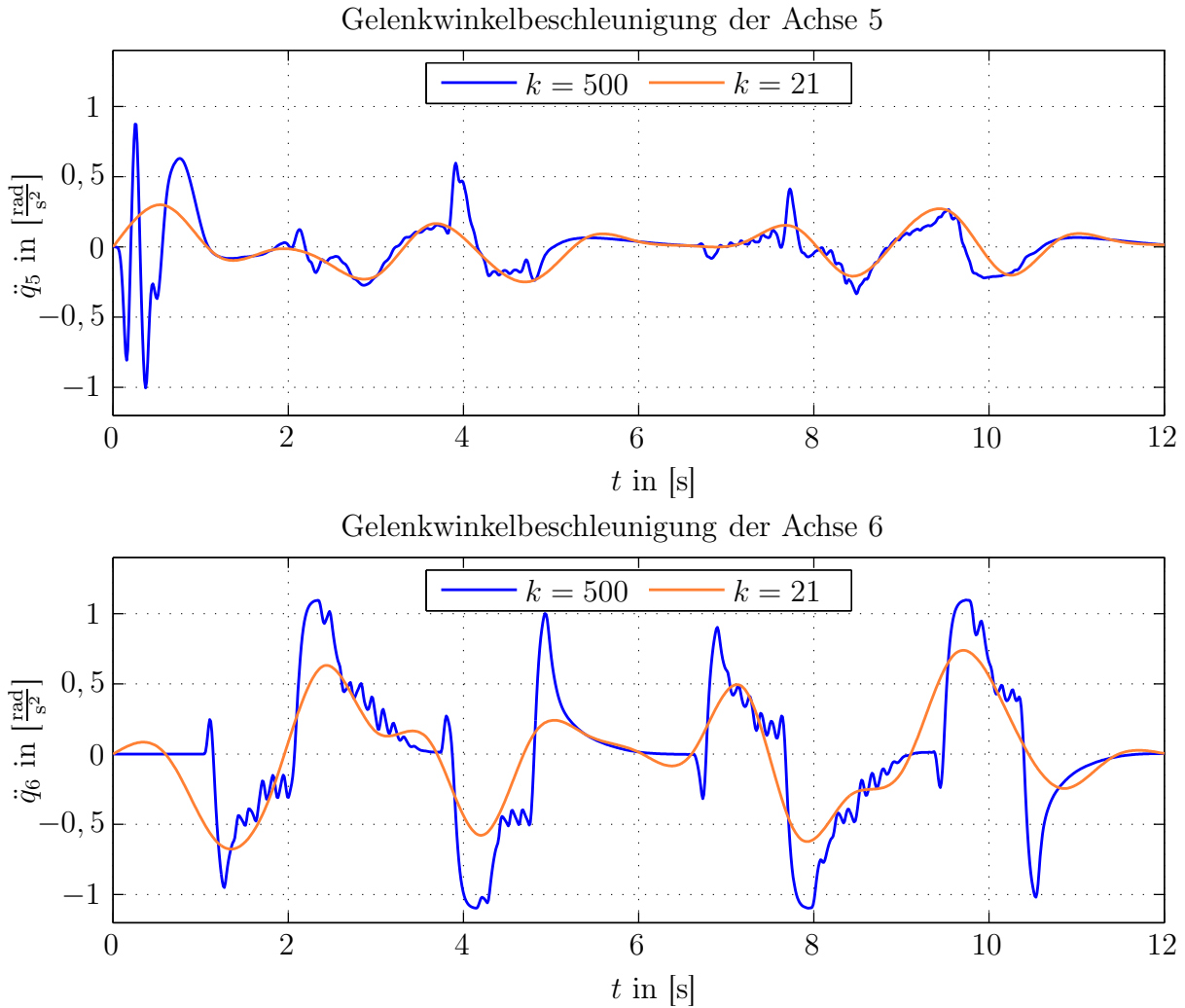


Abbildung 5.4: Vergleich der Gelenkwinkelbeschleunigungen von Achse 5 und 6 aus der Online-Lösung mit $k = 500$ Stützstellen und approximiert mit $k = 21$ Stützstellen. Die charakteristische Beschaffenheit des exakten Signals kann durch die Approximation in ausreichendem Maße dargestellt werden. Hochfrequente Anteile werden aufgrund der niedrigen Stützstellenanzahl nicht vermittelt.

- Die enorme Verschlechterung des Gütemaßes wird durch die Betrachtung der in Abb. 5.5 gezeigten Querbeschleunigungsverläufe, die auf den Simulatorinsassen wirken, nachvollziehbar. Obwohl sich die Eingangssignale $\ddot{q}_1 \dots \ddot{q}_7$ der approximierten Online-Lösung in deren grobem Verhalten nur geringfügig von denen der exakten Online-Lösung unterscheiden, führen diese zu einer komplett verfälschten Vermittlung der Beschleunigung an den Simulatorinsassen. Das gewählte Gütemaß, welches Fehler der Beschleunigungs- und Drehratendarstellung quadratisch bestraft, reagiert äußerst sensibel auf eine solche Fehldarstellung und nimmt demzufolge den deutlich erhöhten Wert an.

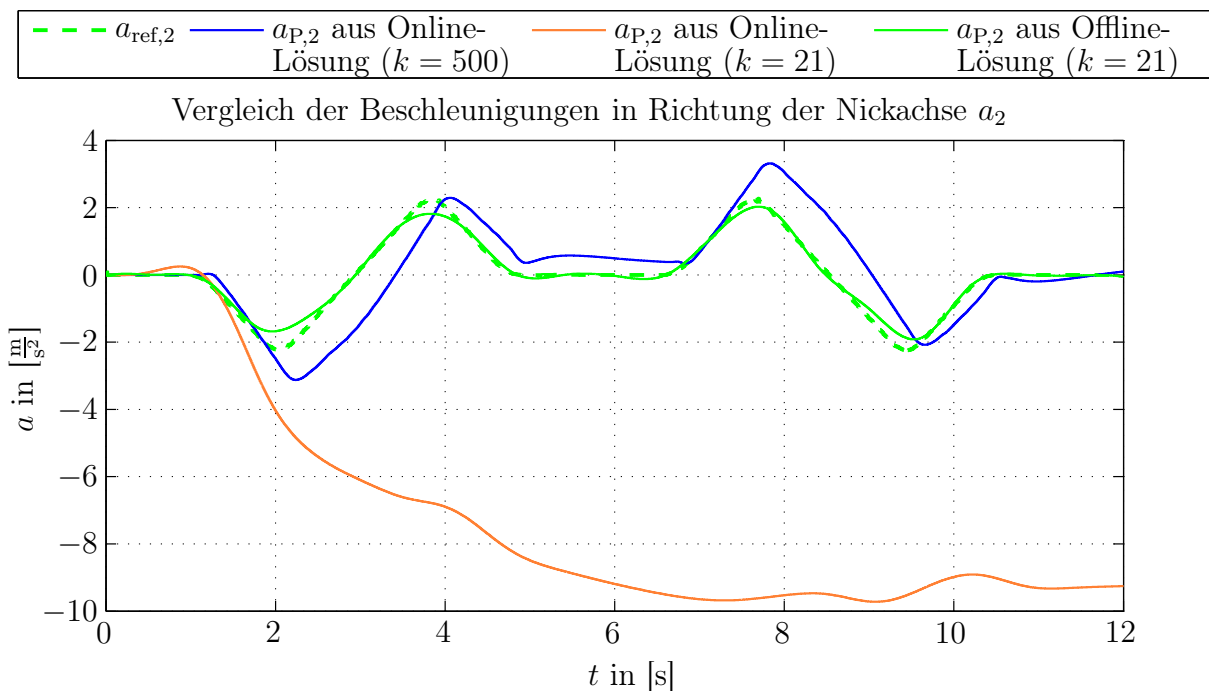


Abbildung 5.5: Vergleich der Querbeschleunigungen (Nickachse). Verglichen wird der Referenzverlauf, die Lösung aus der Online-Optimierung approximiert durch B-Splines mit $k = 500$ bzw. $k = 21$ Stützstellen sowie das Ergebnis der Offline-Optimierung, welche ebenfalls mit $k = 21$ Stützstellen arbeitet und die approximierte Online-Lösung mit $k = 21$ als Startlösung verwendet. Die enorme Verschlechterung des Gütemaßes der approximierten Online-Lösung ist in Anbetracht deren Beschleunigungsvermittlung (orange) nachvollziehbar.

Die Online-Lösung mit $k = 500$ Stützstellen liefert hinsichtlich des in der Arbeit verwendeten Gütemaßes lediglich eine geringe Verbesserung im Vergleich zur Nulllösung. Dies ist der Fall, da die Online-Optimierung auf einem *Washout-gefilterten* Signal basiert und zunächst keinerlei Kräfte und Drehraten berücksichtigt, welche innerhalb des genutzten

Gütemaßes vorrangig sind. Abb. 5.5 macht ersichtlich, dass durch die Online-Lösung eine Darstellung der auftretenden Beschleunigung nichtsdestotrotz gelingt. Die approximierte Online-Lösung bietet dem SQP-Verfahren im Gegensatz zur Nulllösung eine Starttrajektorie, von der aus ein Optimierungsstart möglich ist. Gleichzeitig wird der Rechenaufwand im Vergleich zur Verwendung der exakten Online-Lösung verringert, während aufgrund der ähnlichartigen Beschaffenheit des Eingangssignals eine effektive Lösungsfindung ermöglicht wird.

Hinsichtlich des gewählten Gütemaßes zeigt sich die Offline-Lösung als qualitativ merklich besser. Weitere Verbesserungen und Vorteile gegenüber der Online-Optimierung werden jedoch in den folgenden Abschnitten, hauptsächlich durch das Tuning der Startkonfigurations- und Gewichtungssparameter, deutlich.

5.3.3 Vergleich der Optimierungsalgorithmen mit der Online-Lösung als Starttrajektorie

Tab. 5.4 zeigt die Optimierungsergebnisse der vorgestellten Solver unter den in Tab. 5.1 genannten Voraussetzungen (Ausnahme: $k = 21$) mit der Online-Lösung als Starttrajektorie.

Tabelle 5.4: Vergleich der Optimierungssolver Simplex Algorithmus, SQP, Pattern Search und Genetischer Algorithmus hinsichtlich Rechenzeit und Gütemaß; Starttrajektorie: Online-Lösung.

Algorithmus	Gütemaß initial: 868,742	Änderung in [%]	Rechenzeit in [s]
Simplex Methode	42,150	-95,1	5469,6
Pattern Search	874,796	+0,7	6124,3
SQP-Verfahren	0,396	-99,95	392908,9
Genetischer Algorithmus	868,742	$\pm 0,0$	6075,6

In der vorliegenden Arbeit wird nach einem Verfahren gesucht, welches allgemein und somit auf jegliche Referenztrajektorie anwendbar ist. Dadurch ist nicht sichergestellt, dass eine Startlösung - unabhängig davon, ob diese aus der approximierten Online-Lösung gewonnen wird, die Nulllösung verwendet wird oder sonstige Startkonfigurationen eingesetzt werden - alle Begrenzungen einhält.

Das Scheitern des Pattern Search Algorithmus und des Genetischen Algorithmus ist im vorliegenden Fall auf eine solche Verletzung der Restriktionen zurückzuführen (hier:

$q_3(t)$ und der Drehratenanteil unterhalb der Wahrnehmungsschwelle $\|\omega_P(t_{\text{lower}})\|$). Während der Genetische Algorithmus gar nicht erst in der Lage ist, von einem „nicht gültigen“ Startpunkt aus einen verbesserten Punkt zu finden, liegt die erste Priorität der Pattern Search Methode in der Einhaltung der Begrenzung. Um dieses Ziel zu erreichen, wird im Rahmen der Optimierungsmethode vorübergehend eine geringfügige Verschlechterung des Gütemaßes in Kauf genommen.

Aufgrund fehlender Robustheit der beiden Solver gegenüber Starttrajektorien, die die Restriktionen verletzen, werden diese für ein allgemeines, nicht auf eine bestimmte Referenztrajektorie fixiertes Vorgehen nicht verwendet.

Die Simplex Methode zeigt die beste Performanz unter Berücksichtigung des erreichten Gütemaßes. Mit den zur Verfügung stehenden Evaluationen erreicht die Simplex Methode jedoch nicht die Güte des SQP-optimierten Ergebnisses, dessen immens lange Rechenzeit wiederum auffallend ist. Diese ergibt sich wiederum aus der Auswertung des Systemgradienten in jeder Evaluation. Aus Sicht des Gütemaßes liefert der SQP-Algorithmus indes ein hervorragendes Ergebnis, welches sogar im Vergleich zur exakten Online-Lösung eine Verbesserung von 94,75% darstellt. Die höhere Effizienz bezüglich der Evaluationsanzahl und Exaktheit des SQP-Algorithmus gegenüber der Simplex Methode führt trotz des enormen Rechenaufwandes zu der Entscheidung, dass im Folgenden mit dem SQP-Verfahren agiert wird. Dass der SQP-Solver anders als das Simplex-Verfahren die Möglichkeit zur parallelisierten und somit schnelleren Berechnung enthält, spricht zusätzlich für diese Wahl. Die erwähnte verteilte Optimierungsberechnung ist Inhalt des folgenden Abschnittes.

5.4 Parallelisierung des Optimierungsprozesses: Cluster-Server

Besonders zur effizienteren Nutzung des SQP-Algorithmus wird der Optimierungsprozess auf mehreren Rechenkernen parallel ausgeführt.

Rahmenbedingungen

Im Folgenden wird mit den in Tab. 5.5 dargestellten Rahmenbedingungen gearbeitet.

5.4.1 Cluster-Server

Der vom Institut für Systemdynamik und Regelungstechnik des *DLR* betriebene Cluster besteht aus insgesamt 18 Linux-Rechnern mit insgesamt 33 nutzbaren CPUs als Rechen-

Tabelle 5.5: Optimierungsparameter für die Parallelisierung des Optimierungsprozesses.

Parameter	Wert/Auswahl
Stützstellen pro Eingangsvariable	21
Fehlertoleranz des Optimierers	1E – 9
Max. Gütemaßevaluationen	1000
α	0,3
β	0,3
γ	0,8

server. Dabei werden die Jobs mit Hilfe einer Lastverteilungssoftware automatisch einem geeigneten Rechner zugewiesen. Für die Großzahl der Rechenjobs werden innerhalb des Clusters 64-Bit-Doppelprozessor-Blade-Server genutzt. Bei den im Cluster eingesetzten Prozessoren pro Knoten handelt es sich um *Intel Xeon 5160*-Prozessoren mit 2 Kernen ohne Multithreading-Technologie. Im lokalen Rechner ist ein *Intel Xeon W3530*-Prozessor mit 4 Kernen und jeweils 2 Threads pro Kern verbaut.

5.4.2 Performanz-Vergleich

Die deutlich kürzere Rechenzeit bei der parallelisierten Ausführung basiert vor allem auf der Eigenschaft des SQP-Verfahrens, überhaupt für eine verteilte Berechnung geeignet zu sein. Es gilt jedoch zu erwähnen, dass die relative Zeitersparnis mit der Anzahl an Optimierungsvariablen steigt, da der Einfluss der Zeit für die Übersetzung des Modells geringer wird. Da diese Übersetzungszeit aufgrund der Verteilung auf die Clusterrechner bei der parallelen Ausführung höher ist als bei der Berechnung auf einem lokalen Rechner, lohnt sich der Einsatz des Clusters erst ab einer gewissen Modellkomplexität. Allgemein gilt:

$$\text{Anzahl Optimierungsvariablen} = (\text{Eingänge} \cdot \text{Stützstellen}) + \text{Tuning-Variablen}$$

Dass die Zeit, die zur Verteilung der Prozesse bei der parallelisierten Ausführung nicht vernachlässigbar ist, wird durch den Vergleich der Rechenzeit zwischen lokal und auf dem Cluster parallelisierter Berechnung deutlich: Während auf dem Cluster die vierfache Anzahl an Threads zur Verfügung steht, benötigt die lokal parallelisierte Berechnung lediglich das rund 2,5-fache der Rechenzeit.

In Tab. 5.6 sind die Berechnungszeiten für die Optimierungsprozesse mit dem SQP-Verfahren unter Verwendung der in Tab. 5.5 aufgeführten Rahmenbedingungen dargestellt. Verglichen werden die Rechenzeiten auf einem einzigen lokalen Kern, parallelisiert

auf einem lokalen Rechner und parallelisiert auf dem Cluster bis ein Abbruchkriterium aktiv wird. Als Starttrajektorie wird wiederum die Online-Lösung verwendet.

Tabelle 5.6: Performanzvergleich zwischen den Berechnungen auf einem Kern und verteilten Berechnungen auf lokaler CPU sowie auf dem Cluster;
Algorithmus: SQP; Starttrajektorie: Online-Lösung.

Parallelisierung	Threads	Rechenzeit in [s]	Rechenzeit in [h]
keine	2	392908,9	109,14
lokal	8	94619,0	26,28
Cluster	32	37587,1	10,44

5.5 Parameter-Tuning

Rahmenbedingungen

Für die folgenden Schritte werden die in Tab. 5.7 dargestellten Parameter verwendet.

Tabelle 5.7: Optimierungsparameter für das Tuning von bisher festen Parameterwerten.

Parameter	Wert/Auswahl
Stützstellen pro Eingangsvariable	21
Fehlertoleranz des Optimierers	$1\text{E} - 9$
Max. Gütemaßevaluationen	5000

Die Variation von bisher festen Parametern ermöglicht es, die Startkonfiguration des Roboters sowie die Gewichtung der Gütemaßanteile (Richtung \leftrightarrow Betrag, Beschleunigungen \leftrightarrow Drehraten) zu variieren. Als Algorithmus wird für die folgenden Optimierungen innerhalb des aktuellen Kapitels ausschließlich der SQP-Algorithmus genutzt. Dabei wird die approximierte Online-Lösung als Starttrajektorie verwendet.

Um die Ergebnisse des Parameter-Tunings einordnen und bewerten zu können, sind in Abb. 5.6 die Referenz- und Istverläufe der Beschleunigungen und Drehraten dargestellt, wie sie aus der Optimierung mit dem SQP-Algorithmus und den in Tab. 5.5 genannten Parametern ohne jegliches Tuning resultieren. Die Beschleunigungen werden dabei im Hinblick auf die mit $k = 21$ recht niedrige Stützstellenanzahl bis auf die Bereiche der maximalen/minimalen Beschleunigungen zufriedenstellend dargestellt. Als unzureichend ist das Folgeverhalten der Drehratensignale zu bewerten, welche mit der Referenz vor allem während der Zeitbereiche nicht übereinstimmen, in denen sich die Querbeschleunigung $a_{P,2}$ stark ändert. Die markant fehlerhaften Drehratensignale sind mit ① gekennzeichnet.

Zu erklären sind die Differenzen zwischen Soll- und Istdrehraten, im Speziellen das Verhalten von $\omega_{P,2}$ während der Zeit zwischen $t = 2\text{s}$ und $t = 4\text{s}$, wie folgt: Der Parameter $\gamma = 0,8$ gewichtet die Beschleunigungen im Vergleich zu den Drehraten stärker. Demzufolge besteht das vorrangige Ziel der Optimierung in der korrekten Darstellung der Querbeschleunigung $a_{P,2}$. Diese Querbeschleunigung wird zunächst durch eine Bewegung der Linearachse q_7 realisiert. Die Beschleunigung dieser Achse (\ddot{q}_7) erreicht währenddessen jedoch deren Beschränkung. Folglich muss eine zusätzliche Querbeschleunigung durch eine Parallelbewegung mit Hilfe der Achsen 1, 4, 5 und 6 realisiert werden. Da \ddot{q}_4 ebenfalls die Beschränkung erreicht, kann eine resultierende Drehgeschwindigkeit um die Gierachse ($\rightarrow \omega_{P,2}$) nicht vermieden werden. Im Bereich zwischen $t = 8\text{s}$ und $t = 10\text{s}$ liegt die Problematik in analoger Form aber anderer Richtung vor.

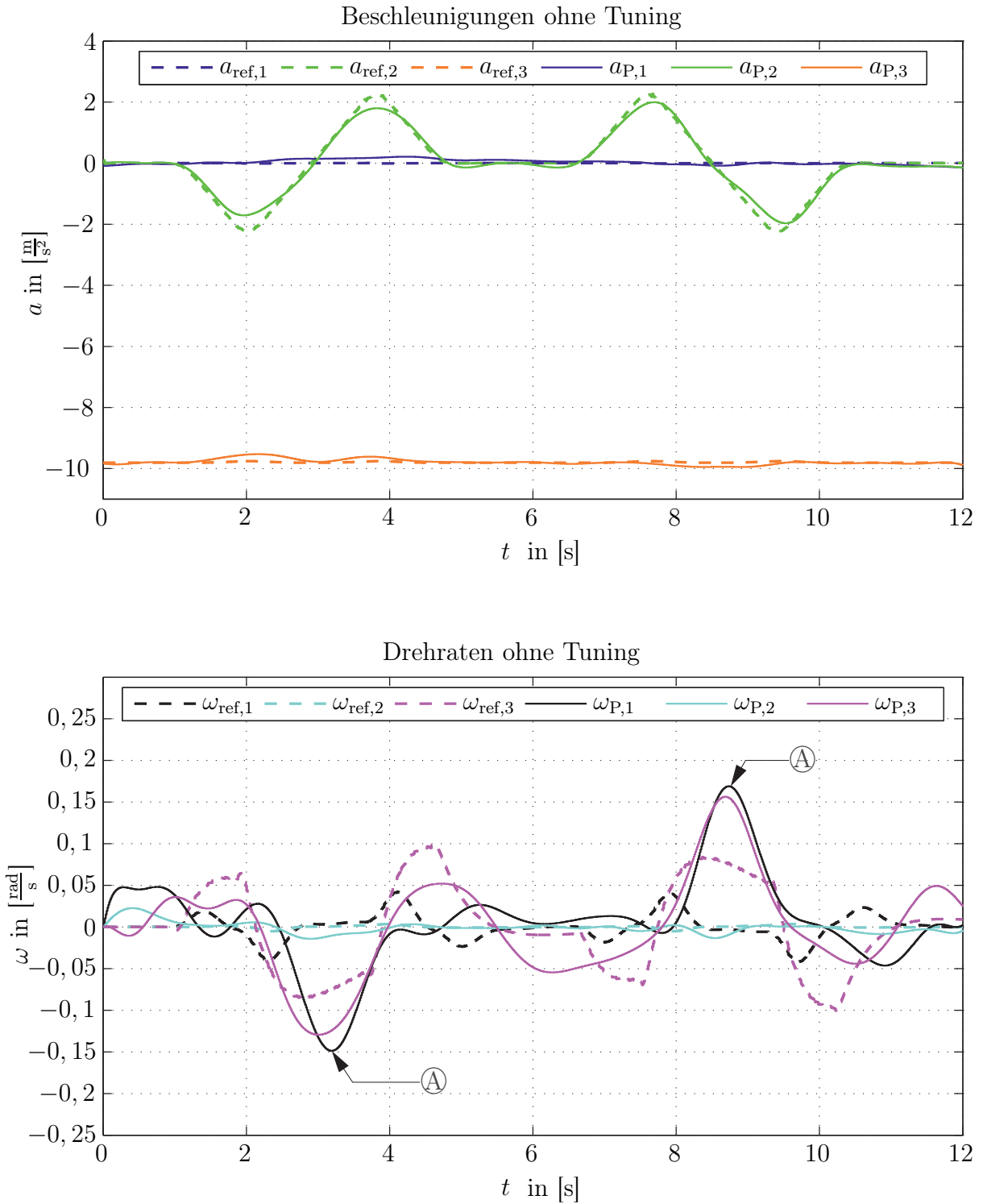


Abbildung 5.6: Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den Gewichtungsfaktoren $\alpha = \beta = 0,3$; $\gamma = 0,8$ und fester Startkonfiguration \mathbf{q}_0 . Achse 7 erreicht Beschleunigungsbeschränkung und kann Querbewegung nicht korrekt darstellen. Zusätzliche Parallelbewegung durch Achse 1, 4, 5 und 6 erzeugt resultierende Drehgeschwindigkeit ($\rightarrow \textcircled{A}$), da Achse 4 ebenfalls Beschränkung erreicht.

5.5.1 Variation der Startkonfiguration \mathbf{q}_0

Bei der Online-Lösung wurde die Startkonfiguration \mathbf{q}_0 so gewählt, dass das gewichtete Beschleunigungsvermögen aller Richtungen maximal ist. Die Startposition ist demzufolge unabhängig von der Referenz und nicht Teil der Optimierung.

Im Gegensatz dazu kann die Startkonfiguration \mathbf{q}_0 bei der Offline-Optimierung als Tuningvariable in die Optimierung miteinbezogen werden. Diese Modifikation verbessert die Optimierungsergebnisse bezüglich des Gütemaßes zunächst nicht. Ferner können in den in Abb. 5.7 veranschaulichten Verläufen der Beschleunigungen und Drehraten mit angepasster Startkonfiguration \mathbf{q}_0 kaum Unterschiede im Vergleich zur vorangegangenen Optimierung ohne Variation von \mathbf{q}_0 ausgemacht werden. Da lediglich die Startwerte der Gelenkwinkel modifiziert werden, besteht das Problem der resultierenden Drehgeschwindigkeiten weiterhin, welches durch das Erreichen mehrerer Gelenkwinkelbeschleunigungsgrenzen hervorgerufen wird und hier mit ③ markiert ist. Da die angesprochenen Beschleunigungsgrenzen noch früher als im vorigen Versuch erreicht werden, entstehen zusätzlich erhöhte Überschreitungen der Drehraten um die Gierachse (\rightarrow ④).

Ferner ist zu erkennen, dass bei ⑤ die Nebenbedingung der Drehraten unterhalb der Wahrnehmungsschwelle eingreift. Während $\omega_{P,1}$ und $\omega_{P,2} \approx 0 \frac{\text{rad}}{\text{s}}$ betragen, wird $\omega_{P,3}$ im Bereich von $\omega_{\text{thres}} = \pm 0,05236 \frac{\text{rad}}{\text{s}}$ gehalten. Dies entspricht dem gewünschten Verhalten, da sich der Betrag des Referenzdrehratenvektors während des entsprechenden Zeitbereichs ebenfalls unterhalb der Wahrnehmungsschwelle befindet.

Der entscheidende Vorteil der Startkonfigurationsanpassung liegt in der deutlich effizienteren Nutzung des Arbeitsraumes des Roboters. Sowohl zum Startzeitpunkt als auch während des Manövers erreichen die Gelenkwinkel und -geschwindigkeiten bei modifiziertem \mathbf{q}_0 häufiger deren Restriktionswerte. Während solch ein Verhalten bei interaktiven Simulationen als äußerst kritisch zu bewerten wäre, ist dies bei der Offline-Planung aufgrund der bekannten Trajektorie unproblematisch. Vielmehr bietet die Offline-Optimierung dem Simulator die Möglichkeit, sich stets in besonders effizienten und für die folgenden Bewegungen günstigen Gelenkwinkelkonfigurationen zu befinden, auch wenn sich diese nahe der Grenzen befinden. Das Erreichen mehrerer Grenzwerte ohne diese zu überschreiten ist demzufolge ein durchaus erwünschtes Verhalten.

Enorme Verbesserungen hinsichtlich des Gütemaßes zeigen sich erst in Kombination mit der Variation der Parameter α und β , wobei sich die modifizierten Werte der Tuningvariablen gegenseitig beeinflussen und für jede Kombination von α , β und γ neue \mathbf{q}_0 -Konfigurationen resultieren. Ein Vergleich der Startkonfigurationen \mathbf{q}_0 aus Online- bzw. Offline-Lösung folgt in Abb. 5.13 im Rahmen des Abschnitts „Erhöhung der Stützstellenanzahl“.

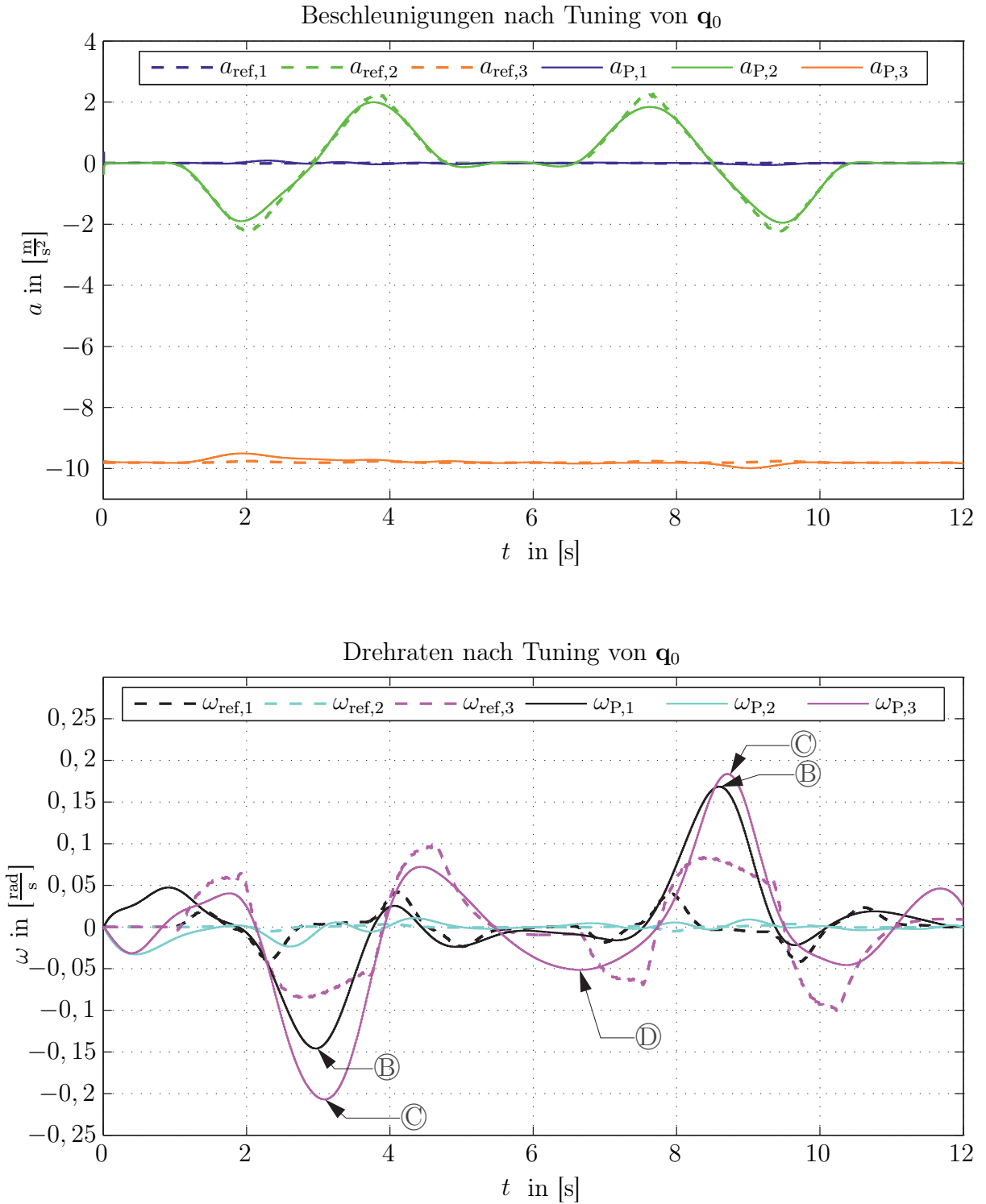


Abbildung 5.7: Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den Gewichtungsfaktoren $\alpha = \beta = 0,3$; $\gamma = 0,8$ und angepasster Startkonfiguration \mathbf{q}_0 . Beschleunigungen und Drehraten werden durch die Anpassung zunächst nicht exakter dargestellt. Fehlerhafte Drehratendarstellungen bestehen weiterhin (\rightarrow B,C). Die Nebenbedingung bezüglich der Wahrnehmungsschwelle für die Drehraten wird hingegen eingehalten (\rightarrow D).

5.5.2 α, β, γ - Variation

Wie im Abschnitt „Gütekriterium“ erwähnt, skalieren α bzw. β den Betrag der Beschleunigungen bzw. Drehraten. γ gewichtet die Beschleunigungsanteile relativ zu den Drehratenanteilen ($\gamma \rightarrow 1$: Beschleunigungen, $\gamma \rightarrow 0$: Drehraten).

Automatische Variation von α und β

Ist die vorgegebene Referenztrajektorie fahrbar, d.h. kann der Simulator ohne Verletzung jeglicher Restriktionen die vorgegebenen Beschleunigungen und Drehraten darstellen, so verschwindet der „Fehlervektor“ zwischen Referenz- und Istwerten (vgl. Abb. 3.1). α und β können sich in diesem Fall ohne maßgebliche Erhöhung des Gütemaßes dem maximalen Wert von eins annähern. In diesem Fall stimmen automatisch die Richtungen der Beschleunigungen und Drehraten überein. Kann der Simulator nicht alle Anteile der vorgegebenen Trajektorie darstellen, so werden α und β verringert, um zumindest die für den Eindruck des Insassen wichtigeren Bewegungsrichtungen korrekt zu vermitteln.

Werden α und β der Optimierung als freie Tuner übergeben, ist neben einer Begrenzung der Wertebereiche eine inverse Berücksichtigung innerhalb des Gütemaßes vonnöten:

$$L_{\alpha\beta}(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + \left(\frac{1}{\alpha} - 1\right) + \left(\frac{1}{\beta} - 1\right) = L(\mathbf{x}, \mathbf{u}) + \frac{1}{\alpha} + \frac{1}{\beta} - 2 \quad (5.3)$$

mit $0 \leq \alpha, \beta \leq 1$ und $L(\mathbf{x}, \mathbf{u})$ aus Gl. 3.25. Dabei wird durch die zusätzlichen Terme $(\frac{1}{\alpha} - 1)$ und $(\frac{1}{\beta} - 1)$ verhindert, dass α und β stets den Wert 0 annehmen, was zu einer gänzlichen Unterdrückung der Absolutanteile führen würde. Um die Gütemaßwerte im fahrbaren Fall nicht zu verfälschen wird jeweils der Wert 1 abgezogen, sodass bei $\alpha, \beta = 1$ keine Beeinflussung des Gütemaßes resultiert und somit die Optimierung des bewegungsorientierten Gütemaßes $L(\mathbf{x}, \mathbf{u})$ nicht an Gewicht verliert.

Manuelle Anpassung von γ

γ kann nicht als automatische Tuningvariable genutzt werden, da eine völlige Unterdrückung eines gesamten Anteils zu erwarten ist. Die Beschleunigungsanteile sind grundsätzlich von höherer Größenordnung (andere physikalische Bedeutung, andere Einheit) und würden durch eine freie Wahl von γ gänzlich unberücksichtigt bleiben, da das Tuning zu $\gamma = 0$ führen würde. Vielmehr ist γ als Mittel zu sehen, welches bei nicht exakt darstellbaren Referenztrajektorien die Lösungsfindung ermöglicht, welche für die beschleunigungs- und drehratenseitigen Anteile zufriedenstellende Lösungen ergibt.

Die Abbildungen 5.8, 5.9 und 5.10 zeigen die Beschleunigungen und Drehraten mit drei verschiedenen Werten für γ sowie den jeweils angepassten Tunern α, β und \mathbf{q}_0 .

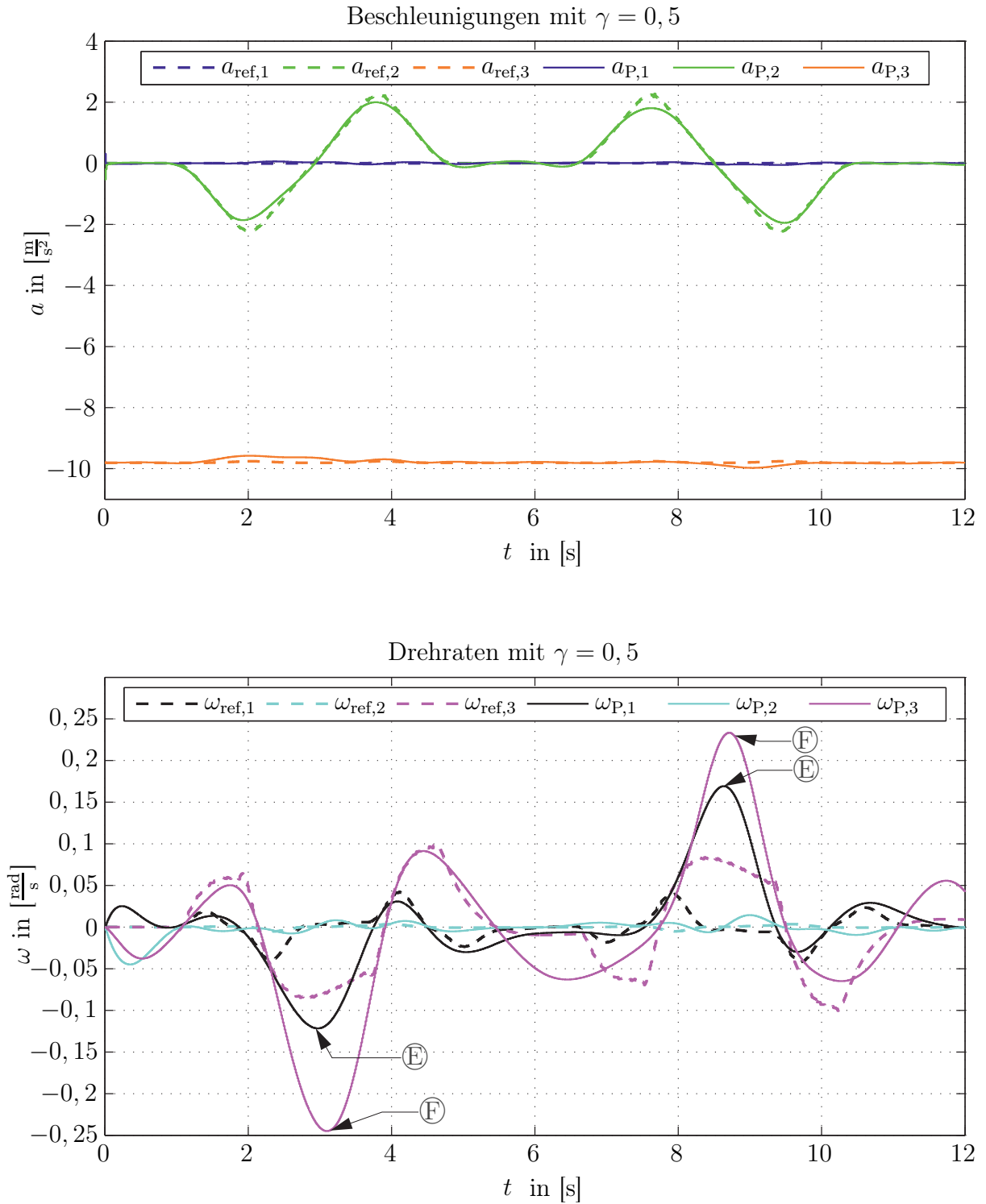


Abbildung 5.8: Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den angepassten Gewichtungsfaktoren $\alpha = 0,8105$; $\beta = 0,9562$; $\gamma = 0,5$ und angepasster Startkonfiguration \mathbf{q}_0 . Die durch \textcircled{E} und \textcircled{F} gekennzeichneten Abweichungen resultieren aus der Beschleunigungsvermittlung mit Hilfe von *tilt coordination* und der beschleunigungslastigen Gewichtung.

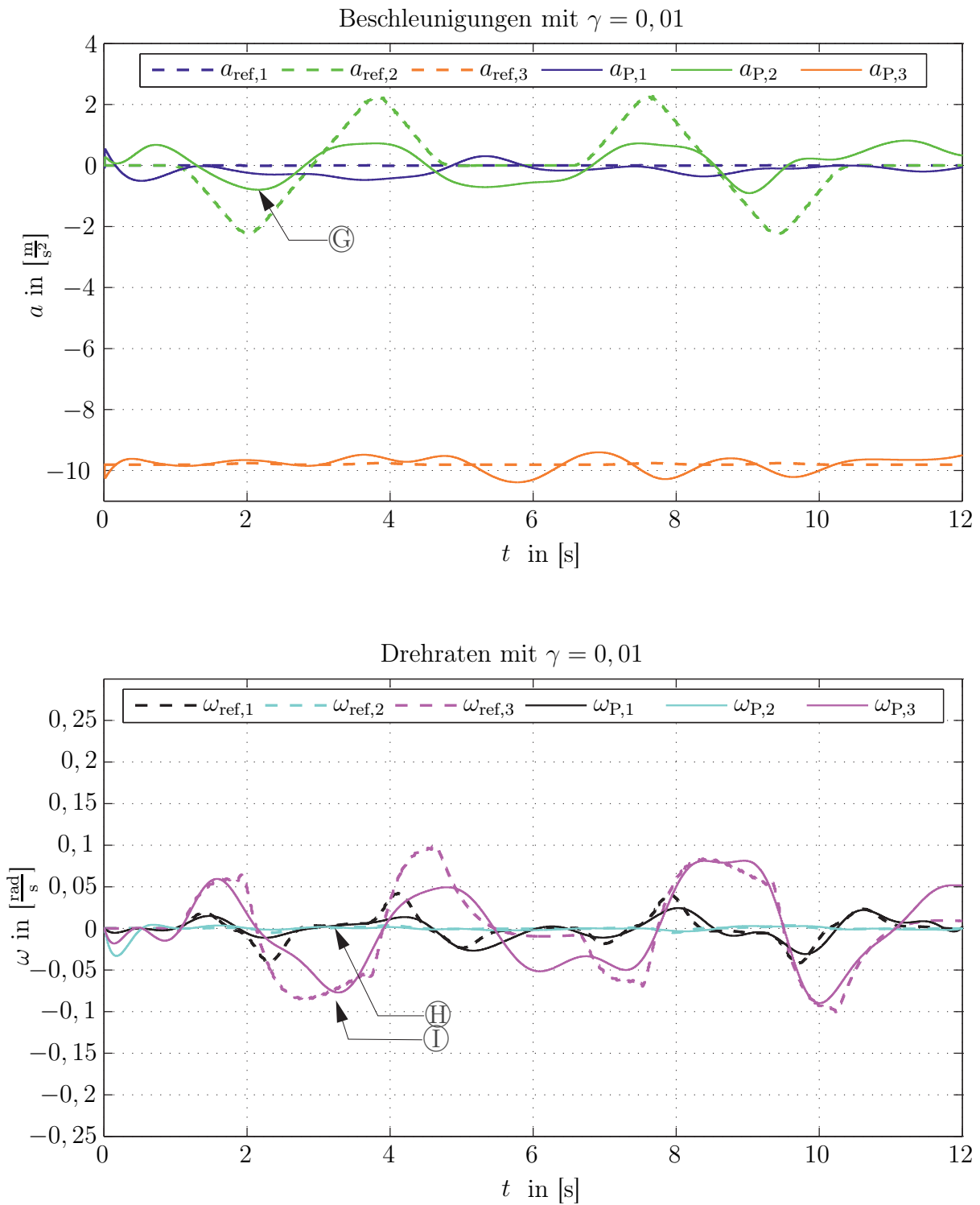


Abbildung 5.9: Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den angepassten Gewichtungsfaktoren $\alpha = 0,9905$; $\beta = 0,9803$; $\gamma = 0,01$ und angepasster Startkonfiguration \mathbf{q}_0 . Die Drehraten zeigen ein deutlich verbessertes Folgeverhalten ($\rightarrow \textcircled{H}, \textcircled{I}$) während den Beschleunigungsreferenzen nicht ausreichend exakt gefolgt werden kann ($\rightarrow \textcircled{G}$).

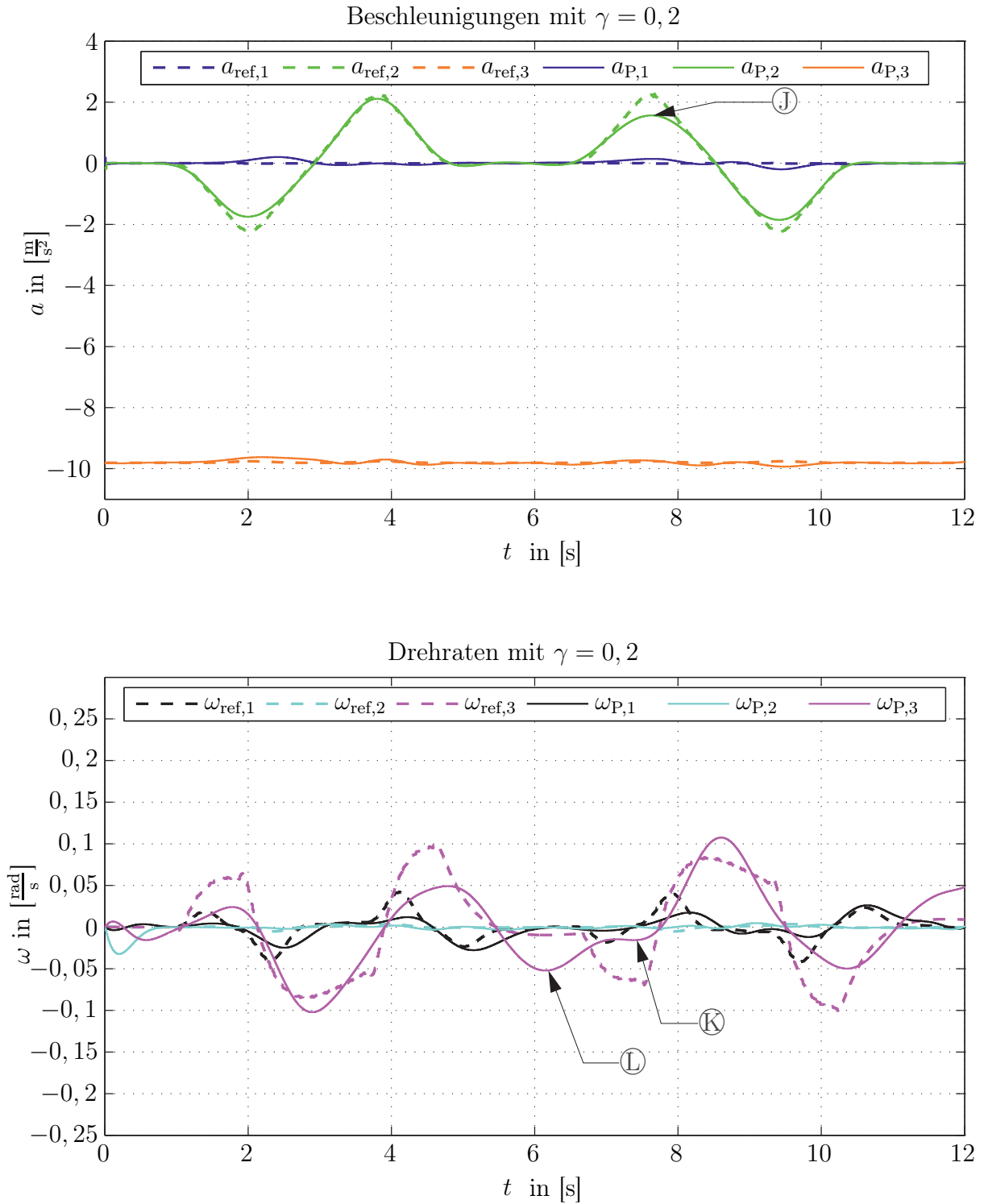


Abbildung 5.10: Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den angepassten Gewichtungsfaktoren $\alpha = 0,9369$; $\beta = 0,9792$; $\gamma = 0,2$ und angepasster Startkonfiguration \mathbf{q}_0 . *Tilt coordination* kann lediglich in geringem Maße zur Darstellung der Beschleunigungsspitzen ($\rightarrow \textcircled{J}$) genutzt werden, um den Drehratenreferenzen ($\rightarrow \textcircled{K}$) gleichzeitig zu folgen.

- $\gamma = 0,5$ (Abb. 5.8):

Bei Werten von $\gamma = 0,5$ stimmen die Beschleunigungsanteile der drei Dimensionen deutlich besser mit den Referenzverläufen überein als ohne Tuning der Gewichtungparameter, obwohl die Erniedrigung von $\gamma = 0,8$ auf $\gamma = 0,5$ eine Schwächung des Beschleunigungseinflusses bewirkt. Die Verbesserung resultiert hingegen aus der Anpassung von α , welches durch das Tuning von 0,3 auf 0,8105 ansteigt.

Die Darstellung der Drehratensignale kann nach wie vor nicht als zufriedenstellend bewertet werden. Die durch ⑤ und ⑥ markierten, fehlerhaften Peaks lassen sich wie folgt erklären: Durch die weiterhin verhältnismäßig starke Gewichtung der Beschleunigungsanteile ziehen Abweichungen der Drehratenanteile geringere Verschlechterungen des Gütemaßes nach sich, als Fehldarstellungen der Beschleunigungen. Im vorliegenden Fall wird die *tilt coordination*, also ein Verdrehen der Kapsel genutzt, um die Referenzbeschleunigungen korrekt zu übermitteln, die aufgrund der Begrenzung von \ddot{q}_7 und \ddot{q}_4 nicht ausschließlich durch translatorische Beschleunigung der Kapsel realisiert werden kann. Fehlerhafte Drehratensignale werden dafür in Kauf genommen.

- $\gamma = 0,01$ (Abb. 5.9):

Wählt man $\gamma \leq 0,05$ folgt eine sehr genaue Übermittlung der Drehraten, während den Beschleunigungsreferenzen nicht mehr korrekt gefolgt wird. Die Trajektorie bestehend aus Beschleunigungs- und Drehratenanteilen kann also nicht komplett exakt dargestellt werden, was sich auch in den Parametern α und β niederschlägt, welche stets Werte < 1 annehmen. Im vorliegenden Simulationsergebnis ist vor allem bei ⑧ und ⑨ ein äußerst zufriedenstellendes Folgeverhalten der Drehraten speziell im Vergleich zu den vorherigen Ergebnissen festzustellen. Durch die drehratenlastige Gewichtung kann die *tilt coordination* kaum mehr zur Darstellung von translatorischen Beschleunigungen genutzt werden, um die Drehratenvermittlung nicht zu verfälschen. Aus diesem Grund können die Beschleunigungen, beispielsweise bei ③, nur unzureichend dargestellt werden, da gleichzeitig die Beschleunigung der Achse 7 und teilweise der Achse 4 an deren Grenzen gelangt.

- $\gamma = 0,2$ (Abb. 5.10):

Mit $\gamma = 0,2$ werden Beschleunigungen und Drehraten äußerst gut, wenn auch nicht vollkommen exakt, dargestellt. Letztere Wahl von γ stellt einen zufriedenstellenden Kompromiss zwischen den beiden einseitigen Gewichtungen dar.

Die nicht vollständig exakt dargestellten Beschleunigungsspitzen (\rightarrow ⑪) resultieren aus den zeitgleich auftretenden, starken Änderungen der Drehratenreferenzen (\rightarrow ⑫). Um diesen zu folgen, kann eine zusätzliche Verstärkung der Beschleunigungen

durch *tilt coordination* nur in begrenztem Maße stattfinden. Neben der Beschleunigung erreicht im Bereich dieser Abweichung auch die Geschwindigkeit der Achse 7 ihren Grenzwert.

Bei ($\rightarrow \mathbb{L}$) kann erneut die Einhaltung der Nebenbedingung bezüglich der Wahrnehmungsschwelle beobachtet werden.

5.6 Erhöhung der Stützstellenanzahl

Eine Erhöhung der Anzahl an Knotenpunkten führt unmittelbar zu einer enormen Erhöhung des Rechenaufwandes. Über den Zusammenhang zwischen Anzahl an manipulierbaren Variablen und der resultierenden Rechenzeit kann jedoch keine pauschale Aussage gemacht werden, da die Topologie des zugrundeliegenden Problems, Restriktionen der Optimierungsvariablen sowie die Wahl des Startpunktes den Rechenaufwand stark beeinflussen und gleichzeitig teilweise nicht mathematisch modellierbar sind.

Für die bisherigen Optimierungen wurde eine recht geringe Zahl an Knotenpunkten genutzt, um mit niedrigerem Aufwand eine zufriedenstellende Lösung des Problems zu erhalten. Um eine noch exaktere Trajektorie zu erhalten, wird nun das Ergebnis nach der manuellen Variation von γ verwendet und dem Optimierungsalgorithmus als Startlösung übergeben. Dabei werden die Startlösungen wiederum als *B-Splines* approximiert, die Anzahl der Stützstellen erhöht und neue *de-Boor-Punkte* generiert, welche als Starttrajektorie verarbeitet werden können. Abb. 5.11 veranschaulicht diesen Vorgang.

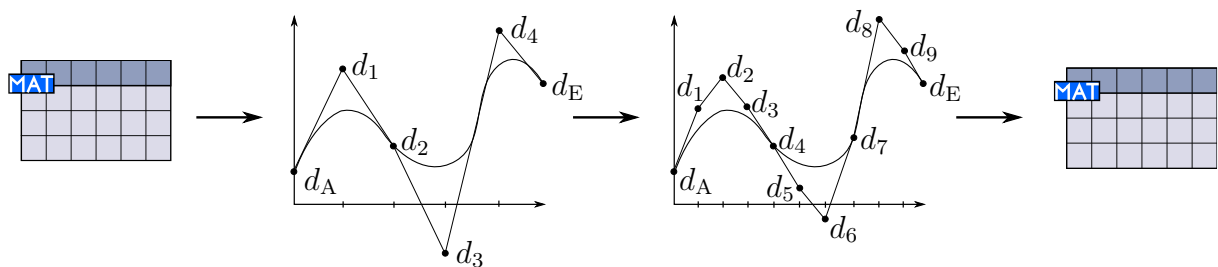


Abbildung 5.11: Vermehrung der Stützstellen zur Verfeinerung der Auflösung.

Anschließend wird die Optimierung mit 43 Stützstellen durchgeführt, welche zu dem in Abb. 5.12 dargestellten Ergebnis führt. α, β sowie \mathbf{q}_0 sind dabei wiederum frei, während $\gamma = 0,2$ beibehalten wird.

Die Zahl der Stützstellen (ohne Anfangs- und Endpunkt) wurde so gewählt, dass die Positionen der Knotenpunkte auf der Abszisse bestehen bleiben, zusätzlich indes zwischen jeweils zwei Stützstellen eine weitere hinzukommt. Daraus folgt, dass aus bisher k Stützstellen $k^* = 2 \cdot k + 1$ Stützstellen werden, was bei $k = 21$ zu $k^* = 43$ führt.

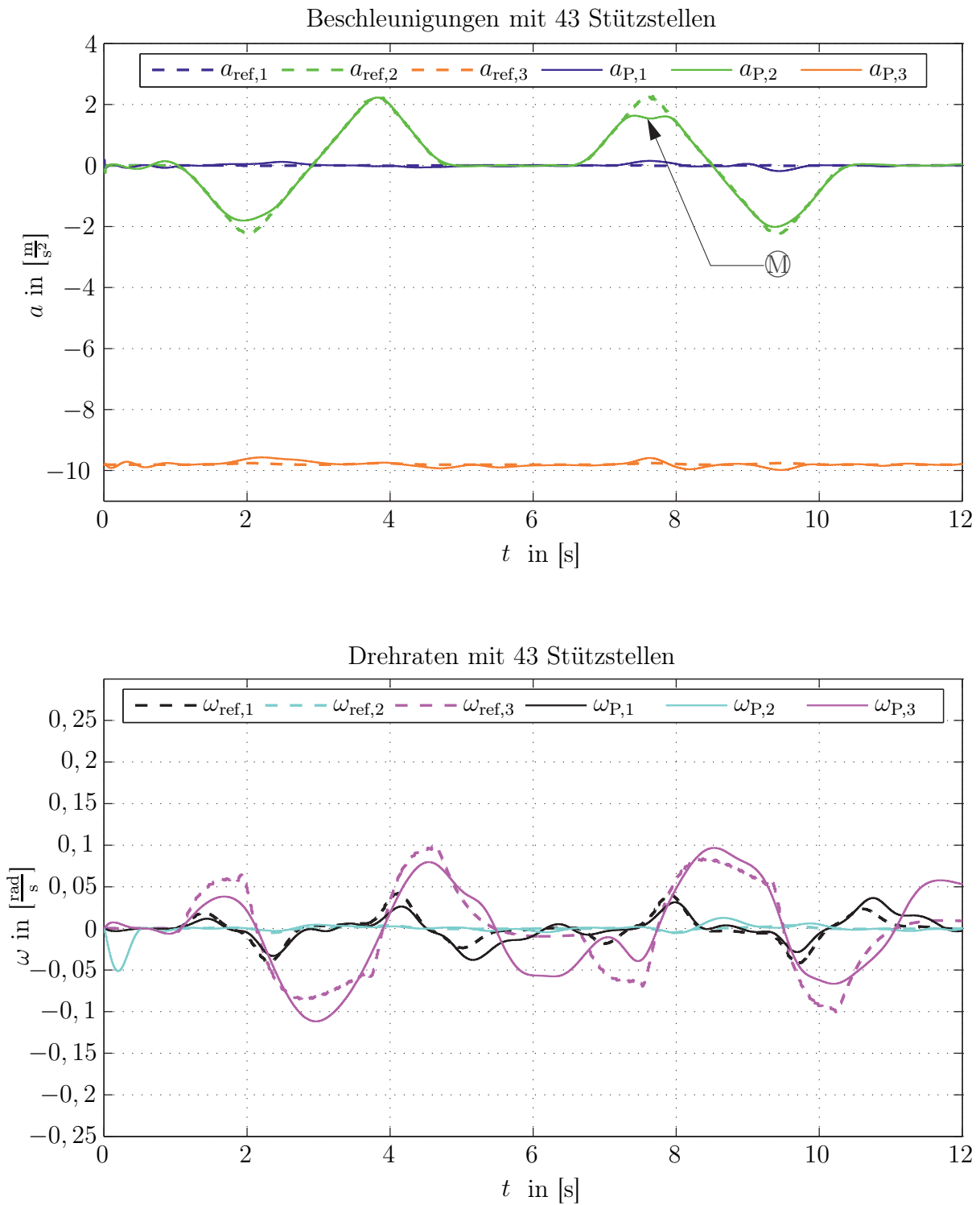


Abbildung 5.12: Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den Gewichtungsfaktoren $\alpha = 0,9485$; $\beta = 0,9928$; $\gamma = 0,2$ und 43 Stützstellen sowie angepasster Startkonfiguration \mathbf{q}_0 . Eine zusätzliche Stützstelle führt im Bereich von \textcircled{M} zu einer Verschlechterung des Beschleunigungssignals, welches durch die exaktere Darstellung der starken Drehratenänderung im selben Zeitraum akzeptabel ist.

Speziell die Drehratenanteile, welche innerhalb der Referenz häufiger hochfrequente Anteile enthalten, können durch die erhöhte Stützstellenanzahl exakter dargestellt werden.

Im Signal von $a_{P,2}$ ist die Vermehrung der Stützstellenanzahl deutlich zu erkennen. Bei $t \approx 7,5\text{s}$ ($\rightarrow \mathbb{M}$), an welcher Stelle im Fall von 21 Stützstellen ein weicher Verlauf vorherrscht, befindet sich nun ein Knick, der offensichtlich von einer zusätzlichen Stützstellen hervorgerufen wird. Aufgrund des deutlich verbesserten Folgeverhaltens von $\omega_{P,1}$ und $\omega_{P,3}$ zu diesem Zeitpunkt ist der Knick im Beschleunigungssignal problemlos zu akzeptieren.

Angepasste Startkonfiguration \mathbf{q}_0

Das Tuning der Startkonfiguration führt in erster Linie zu einer deutlich gestreckteren Konfiguration im Bereich der Achse 3. Dies ist folglich hilfreich, um durch eine Parallelbewegung die vorgegebenen Querbewegungen mit Hilfe eines „größeren Hebelarmes“ darzustellen. Außerdem kann Achse 3 aufgrund der veränderten Startkonfiguration im späteren Verlauf der Bewegung eingreifen, während sie sich im Fall der nicht angepassten Startkonfiguration ab dem Zeitpunkt $t \approx 8\text{s}$ an ihrer Grenze befindet. Ferner befindet sich die Startposition auf der Linearachse 7 bei der Lösung aus der Offline-Optimierung mittiger als im Fall der Online-Lösung.

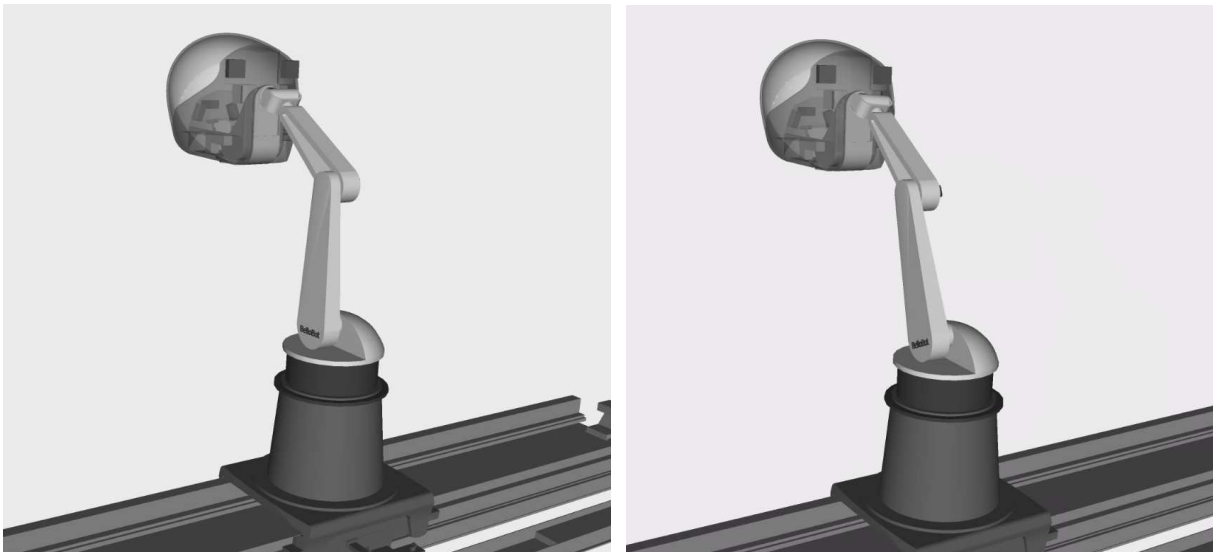


Abbildung 5.13: Vergleich der Startkonfigurationen \mathbf{q}_0 aus der Online-Optimierung (links) bzw. der Offline-Optimierung inkl. Anpassung der Gewichtsparameter und Erhöhung der Stützstellenanzahl (rechts).

6 Anwendung auf weitere Referenzszenarien

Um die Unabhängigkeit der erarbeiteten Methoden hinsichtlich der Referenztrajektorie zu verdeutlichen, werden im Folgenden zwei weitere Bewegungsszenarien behandelt.

6.1 Überholmanöver mit höherer Geschwindigkeit

Zunächst wird wiederum ein Überholmanöver in Form eines doppelten Spurwechsels eines PKW betrachtet. Im Gegensatz zum bisher betrachteten Manöver beträgt die PKW-Geschwindigkeit $v_{\text{PKW}}^* = 18 \frac{\text{m}}{\text{s}}$ statt $v_{\text{PKW}} = 10 \frac{\text{m}}{\text{s}}$. Daraus resultieren deutlich höhere Beschleunigungswerte in horizontaler Richtung, welche die Leistungsfähigkeit des Simulators übersteigt.

Vorgehen und Ergebnisse

Nachdem die Online-Lösung mit $k = 21$ Stützstellen approximiert wird, kommt das Simplex Verfahren zum Einsatz, welches nach 2000 Evaluationen abgebrochen wird, um eine grobe aber recht schnelle Annäherung des Optimums zu erhalten. Nun wird zusätzlich ein Tuning von α und β aktiviert, während γ konstant auf einem Wert von 0,2 gehalten wird. Daraufhin wird das SQP-Verfahren zum Einsatz gebracht. Nachdem die SQP-Optimierung keine bessere Lösung mehr erreicht, wird deren Stützstellenanzahl auf $k^* = 43$ erhöht und nochmals angewendet. Die erhaltenen Ergebnisse werden in Abb. 6.1 präsentiert.

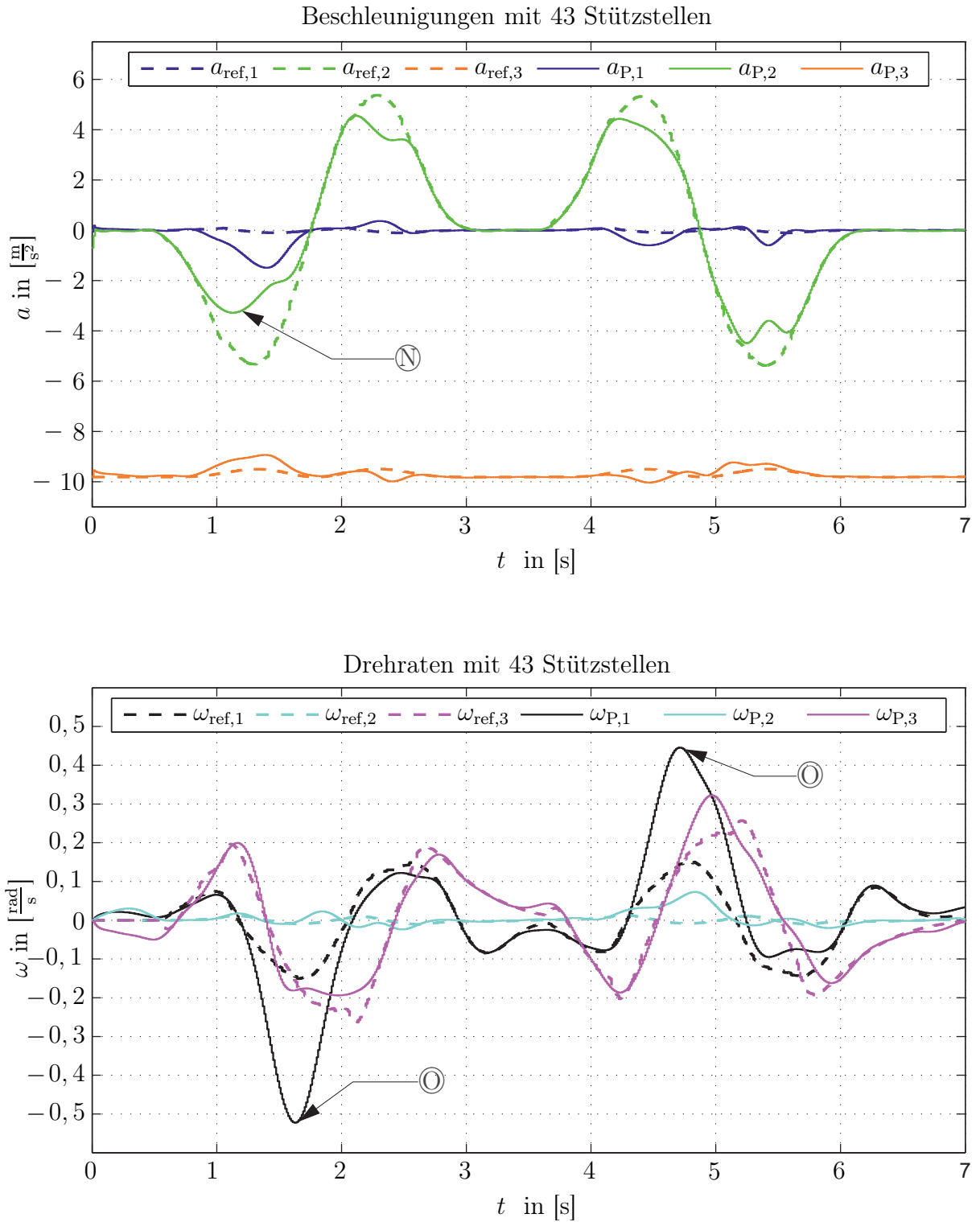


Abbildung 6.1: Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen während des schnelleren Überholmanövers mit den Gewichtungsfaktoren $\alpha = 0,8123$; $\beta = 0,9094$; $\gamma = 0,2$, 43 Stützstellen sowie angepasster Startkonfiguration \mathbf{q}_0 . Die hohen translatorischen Referenzbeschleunigungen können trotz maximalen Gelenkwinkelbeschleunigungen mehrerer Achsen nicht korrekt dargestellt werden ($\rightarrow \textcircled{N}$). Zusätzlich resultieren fehlerhafte Drehraten ($\rightarrow \textcircled{O}$).

Die Simulationsergebnisse zeigen sehr deutlich, dass die Darstellung der geforderten Bewegungen die Leistungsfähigkeit des Simulators übersteigt. Dabei erreichen die Beschleunigungen der Achsen 2, 4, 5, 6 und 7 ihre Begrenzungen.

Dies führt zu einer fehlerhaften Darstellung der translatorischen Beschleunigung in Querrichtung bei \textcircled{N} sowie bei den weiteren Beschleunigungsspitzen im Verlauf der Bewegung. Zum anderen wird versucht - analog zu den Überholmanövern mit niedrigerer Geschwindigkeit bei beschleunigungslastigen Gewichtungen - die Beschleunigungen durch eine zusätzliche Parallelbewegung des Roboters zu verstärken. Da hierbei jedoch die Begrenzungen von Achse 4, 5 und 6 erreicht werden, entsteht wiederum eine resultierende Drehbewegung ($\rightarrow \textcircled{O}$). Im Gegensatz zum langsamen Überholmanöver können hier jedoch trotz der zusätzlichen Parallelbewegung nicht die gewünschten, äußerst hohen translatorischen Beschleunigungen dargestellt werden.

6.2 Landestoß

Um die erprobten Optimierungsvorgänge objektiv bewerten zu können, wird alternativ zum bisher verwendeten Überholmanöver eine Referenztrajektorie eingesetzt, die die Beschleunigungen und Drehraten des Landestoßes eines Flugzeugs darstellt. Damit sollen die bisher erarbeiteten Methoden dahingehend untersucht werden, ob diese für andersartig beschaffene Referenzen ebenso zufriedenstellende Ergebnisse liefern und somit unabhängig von der vorgegeben Trajektorie erfolgversprechend einsetzbar sind. Anders als beim doppelten Spurwechsel, bei dem hauptsächlich horizontale Beschleunigungen auftreten, sind die Insassen eines Flugzeugs während des Landestoßes - dem Aufsetzen eines Flugzeugs auf der Landebahn - größtenteils senkrechten Beschleunigungen ausgesetzt.

Vorgehen und Ergebnisse

Nachdem die Online-Lösung mit $k = 23$ Stützstellen approximiert wird (max. Eigenfrequenz $f_{\max} = 2,60\text{Hz}$), kommt das Simplex Verfahren zum Einsatz, welches nach 3000 Evaluationen abgebrochen wird, um eine grobe aber schnelle Annäherung des Optimums zu erhalten. Nachdem bislang keinerlei Tuning aktiv war, wird im Folgenden eine Variation von \mathbf{q}_0 zugelassen. α und β stehen ab jetzt ebenfalls als Tuningvariablen zur Verfügung, während γ konstant auf einem Wert von 0,2 gehalten wird. Daraufhin wird das SQP-Verfahren mit ebenfalls $k = 23$ Stützstellen und max. 5000 Evaluationen zum Einsatz gebracht. Kann durch die SQP-Optimierung keine bessere Lösung gefunden werden, wird die Stützstellenanzahl auf $k^* = 47$ erhöht und ein weiterer Optimierungsvorgang mit wiederum max. 5000 Evaluationen gestartet. Abb. 6.2 zeigt das Optimierungsergebnis.

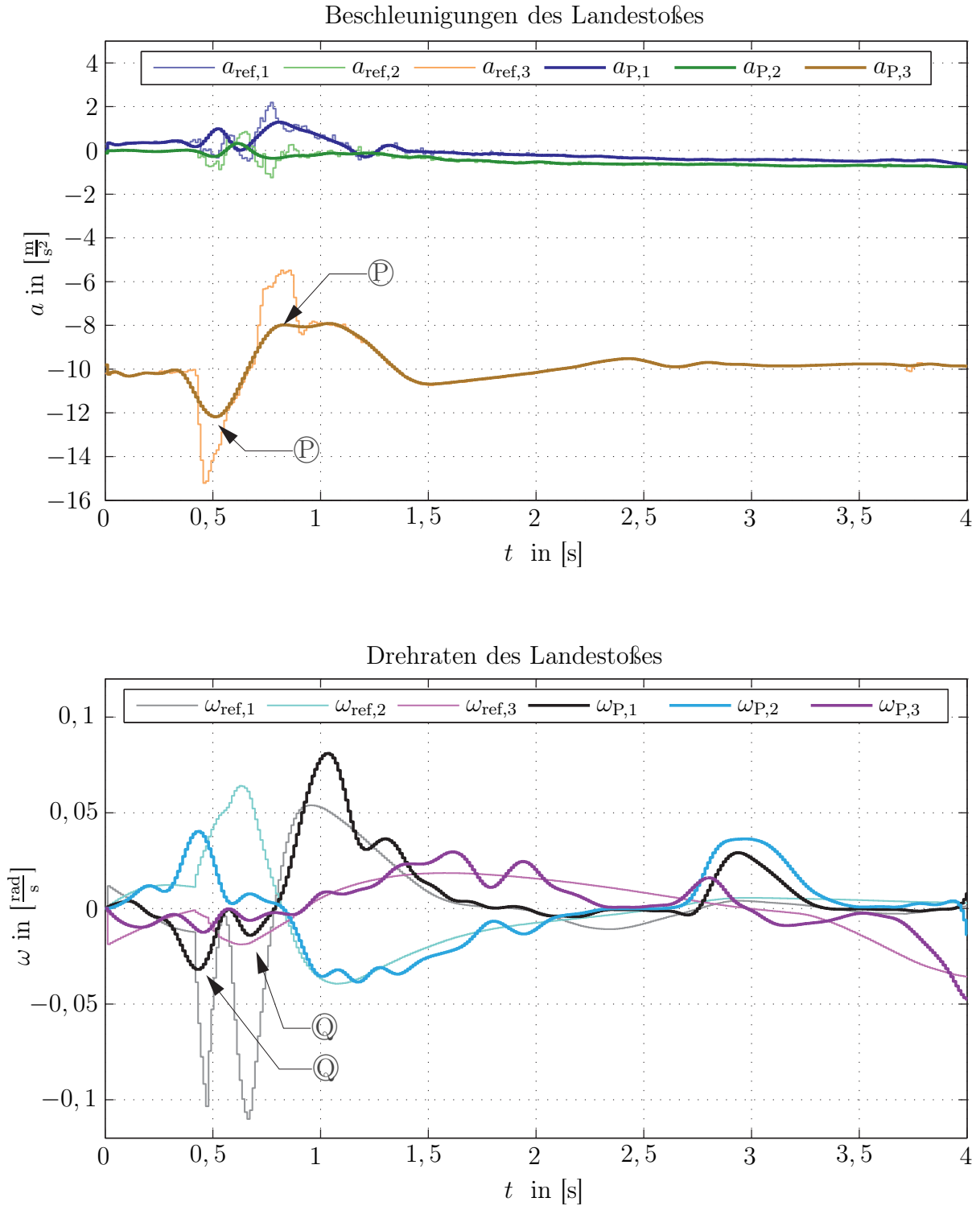


Abbildung 6.2: Gemessene und Referenzverläufe der Beschleunigungen und Drehraten auf den Simulatorinsassen mit den Gewichtungsfaktoren $\alpha = 0,9369$; $\beta = 0,9996$; $\gamma = 0,2$ und 47 Stützstellen sowie angepasster Startkonfiguration \mathbf{q}_0 . Hochfrequenten Referenzanteilen wie bei $a_{\text{ref},1}$ und $a_{\text{ref},2}$ kann aufgrund der Näherung durch *B-Splines* nicht exakt gefolgt werden. \textcircled{P} und \textcircled{Q} bringen die Beschleunigungen der Achsen 3,4 und 5 an deren Grenzen und sind somit nicht exakt darstellbar.

Der Verlauf der Referenz $a_{\text{ref},3}$ stellt sehr anschaulich den Beschleunigungsverlauf dar, der nötig ist, um dem Insassen das Gefühl eines Landestoßes zu vermitteln: Während des ersten Kontaktes des Flugzeugs mit der Fahrbahn nimmt der Betrag der in negative z -Richtung wirkenden Beschleunigung auf den Insassen zu. Dieser wird folglich durch den Landestoß stärker in seinen Sitz gedrückt. Daraufhin erfolgt eine Erniedrigung des Beschleunigungsbetrags. Die Beschleunigung in negative z -Richtung nimmt währenddessen Beträge kleiner der Erdbeschleunigung an, was dazu führt, dass der Insasse leicht aus seinem Sitz gehoben wird. Dies wird durch die Rückfederung des Flugzeugs hervorgerufen.

Die starken Peaks im Verlauf von $\omega_{\text{ref},1}$ bei \textcircled{Q} sowie der zur selben Zeit auftretende Peak von $\omega_{\text{ref},2}$ repräsentieren die Nick- und Rollbewegungen des Flugzeugs, welche aus der leicht schrägen Anflugposition und den nacheinander aufkommenden Rädern des Flugzeugs inklusive der daraus resultierenden Ausrichtung in Fahrtrichtung resultieren.

Das erhaltene Ergebnis kann als äußerst zufriedenstellend eingestuft werden, da die Beschaffenheit der erhaltenen Trajektorie dem der Referenz im Wesentlichen entspricht. Hochfrequenten Referenzanteilen, vor allem bei $a_{\text{ref},1}$ und $a_{\text{ref},2}$, kann aufgrund der Verwendung von *B-Splines* trotz der erhöhten Stützstellenanzahl von 47 nicht exakt gefolgt werden. Da die hochfrequenten Anteile der Referenz teilweise Folge von Diskretisierungsfehlern sind, ist diese Glättung des erhaltenen Verlaufs jedoch ein durchaus erwünschter Effekt. Die enormen Beschleunigungs- und Drehratenwerte und deren schnelle Änderungen bei \textcircled{P} und \textcircled{Q} bringen die Beschleunigungen der Achsen 3, 4 und 5 an deren Grenzen und sind somit nicht exakt darstellbar. Sonstige Abweichungen innerhalb der Drehraten sind durchweg akzeptabel, da sich die Referenzwerte nahezu ausschließlich unterhalb der Wahrnehmungsschwelle befinden.

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

In dieser Arbeit wurden Optimierungsalgorithmen anhand einer optimierungsbasierten Offline-Bahnplanung für einen robotischen Bewegungssimulator analysiert und miteinander verglichen. Anschließend wurden die Ergebnisse des Offline-Verfahrens denen der echtzeitfähigen Online-Bahnplanung gegenübergestellt.

Um Probleme und Potenziale der Bahnplanung hinsichtlich des zugrundeliegenden Robotersystems zu identifizieren, wurde zunächst der Aufbau sowie die kinematischen Zusammenhänge des robotischen Simulators analysiert. Für die Echtzeitplanung stellen sich speziell die redundante Inverskinematik sowie der Umgang mit gelenkwinkelseitigen Begrenzungen des seriellen Roboters als Schwierigkeiten dar.

Um die Planung der Simulatortrajektorie in Form eines Optimierungsproblems lösen zu können, wurde ein beschleunigungs- und drehratenbasiertes Gütemaß festgelegt. Ist die gewünschte Trajektorie aufgrund von Beschränkungen für den Simulator nicht exakt fahrbar, wird mit Hilfe des gewählten Gütemaßes zunächst eine korrekte Darstellung der Bewegungsrichtungen sichergestellt. Im Falle einer fahrbaren Referenztrajektorie gewinnt zusätzlich die korrekte Übermittlung der Beträge von Beschleunigungen und Drehraten an Gewicht.

Während Genetische Algorithmen sowie Pattern Search Algorithmen nur unter Einschränkungen robust einsetzbar sind, konnten mit dem Simplex Algorithmus effiziente und zufriedenstellende Optimierungsergebnisse erzielt werden. Trotz des vergleichsweise höchsten Rechenaufwandes wurden weiterführende Aufgaben jedoch mit Sequentieller Quadratischer Programmierung gelöst. Diese konnte allem voran durch die Exaktheit der erlangten Ergebnisse überzeugen.

In Anbetracht der menschlichen Wahrnehmungsmechanismen können sowohl Online- als auch Offline-Verfahren zu einer Simulatorbahn führen, die dem Insassen einen wahrheitsgetreuen Eindruck der zu simulierenden Bewegungen vermittelt. Neben dem Nachteil der maßgeblich längeren Rechenzeit der Offline-Optimierung bringt diese indes entscheidende Vorteile mit sich. Der aufwändige und problemanfällige Umweg über die Inverskinematik des seriellen Roboters kann ebenso wie die Verwendung eines parametrisierten

Washout-Filters verhindert werden, da das zugrundeliegende Optimierungsproblem ausschließlich auf die Anpassung von Beschleunigungen und Drehraten fokussiert ist. Außerdem muss aufgrund der Bekanntheit der zu fahrenden Trajektorie mit Beschränkungen nicht konservativ umgegangen werden, wie es im Rahmen der Echtzeitplanung der Fall ist. Dies ermöglicht es dem Roboter, sich stets in möglichst effizienten Bereichen des Arbeitsraumes zu befinden und bei Bedarf seine volle Leistungsfähigkeit auszuschöpfen.

Zu weiteren Verbesserungen hinsichtlich der Bewegungsdarstellung führt eine optimierungsbasierte Anpassung der Startkonfiguration des Roboters sowie der Gewichtungparameter innerhalb des verwendeten Gütemaßes. Diese Adaption hängt ebenfalls von der jeweils vorliegenden Referenztrajektorie ab. Somit erfolgt eine referenzspezifische Optimierung, welche sich stets den Eigenschaften verschiedenartiger Bewegungsmanöver anpasst.

7.2 Ausblick

Sowohl bei der Annäherung der Online-Lösung zur Verwendung als Starttrajektorie als auch bei der späteren Optimierung sind die Stützstellen der Knotenpunkte, durch die die abgetasteten Trajektorienverläufe approximiert werden, äquidistant verteilt. Im Rahmen des bearbeiteten Überholmanövers bedeutet dies, dass die Auflösung während der Geradeausfahrt, bei der nahezu keine Beschleunigungen und lediglich geringe Drehraten auf den Piloten wirken, ebenso hoch ist, wie während der Spurwechselphasen, bei denen deutlich großskaligere und sich schneller ändernde Bewegungssignale vorherrschen.

Eine simple Erhöhung der Stützstellenanzahl würde die Darstellung der Bewegungen während der Spurwechselphasen zwar verbessern, gleichzeitig jedoch den Rechenaufwand enorm steigern. Um eine Verbesserung des Folgeverhaltens bei gleichbleibender Rechenzeit zu erreichen, könnten sich zukünftige Arbeiten mit der ungleichmäßigen, der Beschaffenheit der Referenztrajektorie angepassten Verteilung der Knotenpunkte beschäftigen.

Ferner könnte die Entwicklung sinnvoller Automatismen zur Hintereinanderausführung mehrerer Optimierungsalgorithmen erfolgversprechende Forschungsinhalte darstellen. Beispielsweise wäre eine grobe Eingrenzung des Optimums mit Hilfe eines Genetischen Algorithmus gefolgt von der Simplex Methode hilfreich. Der SQP-Algorithmus könnte folglich für die exakte Lösungsfindung sorgen, ohne enorme Rechenressourcen für die Lösungsannäherung ausgehend von einem verhältnismäßig weit entfernten Startpunkt zu verbrauchen. Ein automatisiertes Vorgehen inklusive der Kriterien für einen Algorithmenwechsel könnte eine zusätzliche Performanzsteigerung bewirken.

Literaturverzeichnis

- [Bel57] BELLMAN, R.E.: *Dynamic Programming*. (1957) [4](#)
- [Bel14] BELLMANN, T.: *Optimierungsbasierte Bahnplanung für interaktive robotische Bewegungssimulatoren*, Diss., 2014 [2.1](#), [2.1](#), [2.3](#), [3.1.2](#), [3.4](#)
- [BHH⁺11] BELLMANN, T. ; HEINDL, J. ; HELLERER, M. ; KUCHAR, R. ; SHARMA, K. ; HIRZINGER, G.: The DLR Robot Motion Simulator Part I: Design and setup. In: *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011. – ISSN 1050–4729, S. 4694–4701 [2](#)
- [BOH11] BELLMANN, T. ; OTTER, M. ; HIRZINGER, G.: The DLR Robot Motion Simulator Part II: Optimization based path-planning. In: *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011. – ISSN 1050–4729, S. 4702–4709 [3.1.1](#), [3.1.2](#), [3.2](#), [3.4](#)
- [BP67] BEIGHTLER, C.S. ; PHILLIPS, D.T.: *Applied Geometric Programming*. Wiley, New York, 1967 [4](#)
- [Diw08] DIWEKAR, U.: *Introduction to Applied Optimization*. Springer, 2008. – ISBN 9780387766355 [4](#)
- [DKL98] DAM, E. B. ; KOCH, M. ; LILLHOLM, M.: Quaternions, interpolation and animation. (1998) [3.4](#)
- [DS13] DASSAULT SYSTÈMES, AB: *Dymola Dynamic Modeling Laboratory, User Manual, Volume 1*. 2013 [5.1.1](#)
- [Exl12] EXLER, O.: New Trust Region SQP Methods for Continuous and Integer Optimization. (2012) [4.3](#), [4.3.2](#)
- [Fis09] FISCHER, M.: *Motion-Cueing-Algorithmen für eine realitätsnahe Bewegungssimulation*. Technische Universität Carolo-Wilhelmina zu Braunschweig, Fakultät für Maschinenbau, 2009 [3.1](#)

- [Fob06] FÖBER, T.: Experimentelle Analyse Evolutionärer Algorithmen auf dem CEC 2005 Testfunktionensatz. (2006) [4.1.1](#)
- [GJ48] GROEN, J. J. ; JONGKEES, L. B.: The Threshold of angular acceleration perception. In: *Journal of Physiology*, 1948, S. 1–7 [3.1.1](#)
- [GK99] GEIGER, C. ; KANZOW, C.: Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben. (1999) [4.1.1](#)
- [GK02] GEIGER, C. ; KANZOW, C.: *Theorie und Numerik Restringierter Optimierungsprobleme*. Springer Berlin Heidelberg, 2002 (Springer-Lehrbuch Masterclass). – ISBN 9783540427902 [4.1.1](#)
- [Ham44] HAMILTON, W. R.: On Quaternions, or on a new System of Imaginaries in Algebra. (1844) [3.4](#)
- [HJ61] HOOKE, R. ; JEEVES, T. A.: „Direct Search“ Solution of Numerical and Statistical Problems. In: *J. ACM* (1961), S. 212–229 [4.2](#), [4.2.2](#)
- [HSM02] HEISEL, U. ; STEINLE, T. ; MAIER, W.: Werkzeugmaschinen mit Beinen - die Hexapod-Maschinen. (2002) [2.2.4](#)
- [Kon13] KONKA, J.: Unconstrained nonlinear programming: Nelder-Mead Simplex algorithm. (2013) [4.2](#)
- [LC07] LAI, L.L. ; CHAN, T.F.: The Method of Hooke and Jeeves. (2007) [4.2.1](#)
- [Mer03] MERZ, Dr. P.: *Moderne Heuristische Optimierungsverfahren: Meta - Heuristiken*. Wilhelm-Schickard-Institut für Informatik, Universität Tübingen, 2003 [4](#)
- [Obe12] OBERLE, H.J.: Optimierung, Universität Hamburg, 2012 [4.10](#)
- [Pfe12] PFEIFFER, A.: Optimization Library for Interactive Multi-Criteria Optimization Tasks, 2012, S. 669–680 [5.1.1](#)
- [PM13] PROF. MONTENEGRO, S.: Vorlesungsunterlagen „Orientierungsdarstellung mit Quaternionen“. (2013) [3.1](#)
- [Pop05] POPOV, A.: *Genetic Algorithms for Optimization*. 2005 [4.4](#), [4.4.1](#)
- [RN85] REID, L. D. ; NAHON, M. A.: Flight Simulation motion-base drive algorithms. (1985) [3.1.1](#)

-
- [SSVO09] SICILIANO, B. ; SCIavicco, L. ; VILLANI, L. ; ORIOLO, G.: *Robotics: Modeling, Planning and Control*. Springer, 2009 (Advanced Textbooks in Control and Signal Processing). – ISBN 9781846286414 [3.2](#)
- [Zac78] ZACHARIAS, G.L.: *Motion Cue Models for Pilot-Vehicle Analysis*. Defense Technical Information Center, 1978 [3.3.1](#)