Vorwärtssimulation der Riesenfelge am Hochreck mittels Jerk-Optimierung?

$$Jerk = \frac{1}{2} \int_{t_{start}}^{t_{end}} \left[\frac{d^3\alpha}{dt^3} \right]^2 dt \ \ = Minimum?$$

Diplomarbeit

Lehr- und Forschungsbereich Theoretische Astrophysik
der Eberhard-Karls-Universität Tübingen
vorgelegt von

Karel Kaufmann

1999

Vorwärtssimulation der Riesenfelge am Hochreck mittels Jerk-Optimierung

Diplomarbeit
von
Karel Kaufmann

Lehr- und Forschungsbereich Theoretische Astrophysik
der Eberhard-Karls-Universität Tübingen

August 1999

Die vorliegende Arbeit wurde im Zeitraum von Februar 1998 bis August 1999 am Lehrstuhl für Theoretische Astrophysik der Universität Tübingen unter Obhut von Herrn Prof. Ruder angefertigt.

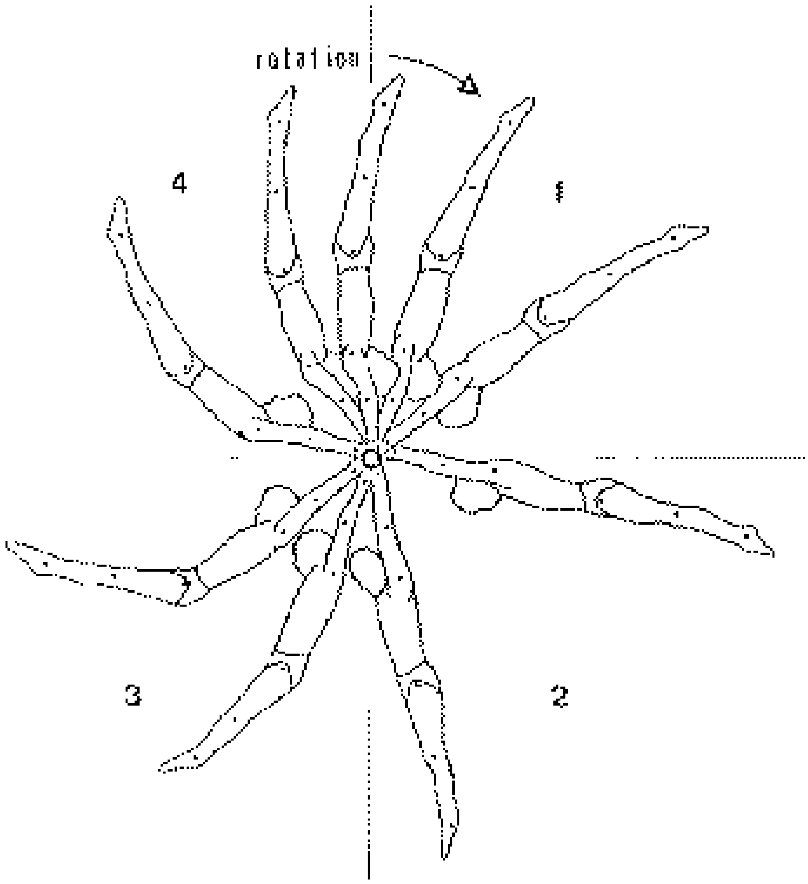
Diese Arbeit wurde von mir selbständig und nur unter Verwendung der angegebenen Hilfsmittel ausgeführt.

August 1999

Inhaltsverzeichnis

Kapitel 1

Einleitung



Die Riesenfelge am Hochreck ist für jeden Hobby-Turner der große Traum. Eine einerseits einfache und harmonische und doch nicht ganz ungefährliche Bewegung, ausgeführt in 2.5 Metern Höhe.

Erfahrungen mit Inverser Dynamik - Vorwärtsdynamik

Die Idee zu einer Vorwärtssimulation der Riesenfelge am Hochreck kam mir erstmals im Frühjahr 1997 in den Sinn, als Harald Böhm an der TAT in einem Seminarvortrag über seine Diplomarbeit berichtete. Er hatte mit Hilfe des MKS(Mehr-Körper-Simulations)-Programms DADS ein 2-dimensionales 6-segmentiges Menschenmodell mit Schwabbelmassen [Gru87] [Wid90] erstellt und mit der Methode der Inversen Dynamik eine Reihe von Meßdaten der Kölner Arbeitsgruppe von Herrn Prof. Brüggemann ausgewertet. Bei den untersuchten Bewegungen handelte es sich um hochdynamische Riesenfelgen vor Mehrfach-Salto-Abgängen vom Reck.
[Bei der verwendeten Methode der Inversen Dynamik werden die gemessenen Ortskurven für die Modell-Segmente zweimal numerisch nach der Zeit differentiert und in die Bewegungsgleichungen eingesetzt. Man erhält so — zumindest theoretisch — die Kraft- und Momentverläufe, durch die die gemessene Bewegung erzeugt wurde.]

Böhm hatte die von DADS inversdynamisch berechneten Kräfte und Momente als Eingabe für eine Vorwärts-Simulation (Integration der Bewegungsgleichungen) verwendet und feststellen müssen, daß die so erhaltene Bewegung mit der Zeit immer stärker von der ursprünglich analysierten abwich. Die Gründe dafür lagen in Ungenauigkeiten bei der zweifachen numerischen Ableitung der Ortskurven durch DADS und in der Empfindlichkeit des Systems gegenüber kleinen Störungen während der Integration. Bereits kleinste Abweichungen von der ursprünglichen Bewegung führen dazu, daß die vom Turner einst genau abgestimmten Kräfte nicht mehr optimal zu der Bewegung passen. Da das Programm im Gegensatz zu einem Menschen die fest vorgegebenen Kräfte während der Bewegung nicht an die neue Situation anpaßt, kommt es zu einer sich schnell verstärkenden Störung, die schließlich zu einer unkontrollierten Bewegung führt [Böh97].

Die Riesenfelge als Lösung einer Optimierungsaufgabe

Durch Harald Böhm stieß ich auf die Arbeit von W.L.Bauer aus dem Jahre 1976. Bauer hatte für ein 2-dimensionales 2-segmentiges Menschenmodell mit Reibung und elastischer Reckstange die Bewegungsgleichungen aufgestellt. Unter der Nebenbedingung des minimalen Energieverlustes durch Reibung optimierte er mit dem *Gradientenverfahren 1. Ordnung* die 2. Zeitableitung des Hüftwinkels $\ddot{\alpha}(t)$, die er als Steuerungsfunktion für die Vorwärtssimulation der Riesenfelge verwendete. Ziel der Optimierung war eine Riesenfelgbewegung vom Handstand in den Handstand ohne dabei durch Reibung an Kinetischer Energie zu verlieren. Den Optimierungvorgang interpretierte Bauer als Lernvorgang, da sein Programm genau wie ein Mensch jede Bewegung nach bestimmten vorher festgelegten Kriterien bewertete und im Hinblick auf diese zu verbessern (optimieren) suchte. Leider wurde die Freiheit der Bewegung von Bauer durch die Annahme, die Riesenfelge verlaufe in der ersten Hälfte der Bewegung komplett gestreckt ($\ddot{\alpha}(t) = 0$), etwas eingeschränkt[Bau76].

Variations-Prinzipien

Die Idee, die Bewegung eines physikalischen Systems aus einem Extremal- bzw. Variations-Prinzip abzuleiten, ist in der Physik spätestens seit *Lagrange* (1736–1813) und *Hamilton* (1805–1865) bekannt.

Das *Hamilton*sche Prinzip besagt, daß die Bewegung eines Systems $\vec{q}(t)$ zwischen zwei festen Zuständen $\vec{q}(t_1)$ und $\vec{q}(t_2)$ genau so verlaufen wird, daß für die durchlaufene Kurve $\vec{q}(t)$ das Integral über die *Lagrange*-Funktion $\int L dt$, auch Wirkungsintegral genannt, ein Extremum annehmen wird. Die Variation des Wirkungsintegrals verschwindet :

$$\delta \int_{t_1}^{t_2} L(\vec{q}, \dot{\vec{q}}, t)\, dt = 0. \tag{1.1}$$

Das physikalische System folgt in diesem Fall rein passiv den wirkenden äußeren Kräften, wie es die *Newton*schen Bewegungsgleichungen vorhersagen [Gol89].

Das Redundanzproblem

Die Bewegung eines Turners wird gleichfalls durch äußere Kräfte beeinflußt, doch hat der Turner im Gegensatz zu dem passiven mechanischen System die Möglichkeit, seine Bewegung aktiv durch den gezielten Einsatz von Muskelkräften zu ändern und zu steuern. Für den Turner wird es daher meist beliebig viele Wege $\vec{q}(t)$ geben, auf denen er sich von einem Zustand $\vec{q}(t_1)$ zu einem zweiten $\vec{q}(t_2)$ bewegen kann. (Redundanzproblem)

Das *Minimum-Jerk-Modell*

Beobachtet man Menschen beim Erlernen neuer Bewegungen in Sport und Alltag, so fällt auf, daß ihre Bewegungen mit zunehmendem Lernerfolg immer geschmeidiger und natürlicher wirken. Natürliche Bewegungen zeichnen sich durch ihre Einfachheit und Harmonie aus : Keine unnötigen Wege oder Beschleunigungen, kein unnötiger Energieverbrauch. Diese Eigenschaften lassen sich in zwei Minimierungskonzepten zusammenfassen : Dem *Minimum-Jerk-Modell* und dem bekannten Prinzip der Energie-Minimierung.

Unter *Jerk* versteht man allgemein die Ableitung der Beschleunigung $a(t)$ nach der Zeit. $\dot{a}$ ist ein Maß für die Änderung der Beschleunigung und somit auch ein Maß für die Glattheit der Kurve $a(t)$.

In Analogie zum *Hamilton*schen Prinzip formuliert man das *Minimum-Jerk*-Prinzip :

Für eine Bewegung mit festem Start- und Endpunkt und feststehender Gesamtzeit werde diejenige Trajektorie gewählt, bei der das Integral über die quadratische Abweichung des Jerk von der Nullfunktion, also

$$\int_{t_{start}}^{t_{end}} \dot{a}^2(t)dt \tag{1.2}$$

minimal ist. In der Literatur wird das Integral gewöhnlich mit einem Faktor 1/2 multipliziert und als *Gesamt-Jerk* bezeichnet [FH85] [SZ89] [Boo97].

Konzept für eine Vorwärtssimulation der Riesenfelge

In Hinblick auf eine Simulation der Riesenfelge war nun klar, daß nach Böhm's schlechten Erfahrungen die Methode der *Inversen Dynamik* wegen ihrer Schwächen nicht in Frage kam. Die *Inverse Dynamik* hätte zwar bei bekannter Bewegung die ungefähren Momentverläufe liefern können, aber wohl kaum solche, die als Vorgabe für eine vernünftige Vorwärtssimulation hätten dienen können. Der Gedanke, die Bewegung der Riesenfelge stattdessen durch Lösung eines Variations- bzw. Optimierungs-Problems finden zu können, schien wesentlich interessanter. Bauer hatte ja gezeigt, daß es prinzipiell möglich war. Als Bewertungskriterien boten sich schließlich neben der Einhaltung der Randbedingungen die Minimierung des Energieverbrauchs (= Reibungsminimierung) und das Minimum-Jerk-Prinzip an, wobei das letztere das effektivere war.

Kapitel 2

Modellierung

2.1 Erste Schritte

In der Anfangsphase des Projekts bestand noch die Absicht, mit Hilfe des MKS-Programms DADS von der Firma CADSI ähnlich wie Böhm [Böh97] eine detaillierte, 3-dimensionale Vorwärtssimulation der Riesenfelge an einem Spannreck durchzuführen. Dieses Vorhaben wurde jedoch nach einigen Fehlversuchen aufgegeben. Zwar ist die Modellierung komplexer Anordnungen 3-dimensionaler Körper mit DADS relativ einfach, doch steigen mit jedem neuen Körper auch der Umfang der zu integrierenden Gleichungen und die Zahl der einzuhaltenden Zwangsbedingungen und somit die Rechenzeit und Fehleranfälligkeit.

Zu Testzwecken wurde ein Spannreck mit realistischen Abmessungen, Abspannseilen und einer Reckstange aus mehreren Teilsegmenten gebaut. Das Modell bestand aus einem Bodenstück, 2 beweglichen Pfosten und einer 5-teiligen Reckstange, die durch 8 lange Federn (Abspannseile), 8 Kugelgelenke und zahlreiche weitere Federn aneinandergekoppelt waren. Für jede Feder konnten bzw. mußten die Parameter *Federkonstante* und *Ruhelänge* vorgegeben werden. Weitere Parameter wie z.B. nichtlineare Kraftgesetze und Dämpfung waren optional. Es war relativ leicht, alle Federn so einzustellen, daß das Reck sich halbwegs realistisch verhielt und nicht sofort unter seinem eigenen Gewicht zusammenbrach oder bei der Bewegung eines angebrachten Pendels nach einer Seite einfach umfiel, doch war es wegen der großen Zahl an Parametern kaum möglich, das Reck an bekannte Meßdaten (z.B. [Böh97]) anzupassen. Auch waren die Rechenzeiten für ein einzelnes Pendel, das an der Reckstange seine Runden drehte, bereits relativ groß, was für ein komplexeres Menschenmodell Schlimmes befürchten ließ.

Für erste Bewegungsversuche wurde deshalb ein neues Modell aus einer fest im Raum schwebenden Reckstange und einem zweisegmentigen Turner-Modell (Dop-

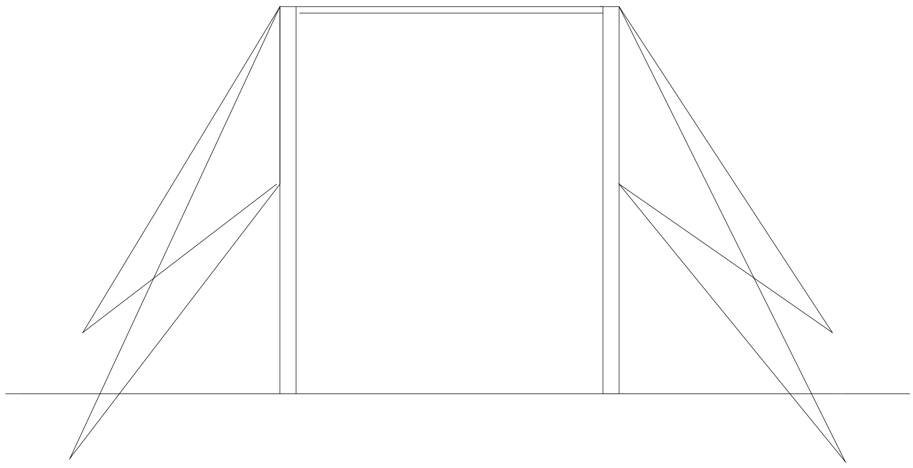


Abbildung 2.1: Nicht-Maßstabsgetreue Skizze des mit DADS nachgebauten Spannrecks mit jeweils 4 Drahtseil-Abspannungen (Federn) rechts und links.

pelpendel) gebaut. Durch Vorgabe einer durch Trial-and-Error entstandenen Momentkurve $M_\alpha(t)$ für das dem Hüftgelenk entsprechende Pendelgelenk konnte relativ schnell eine einfache Riesen-Felge erzeugt werden. Bei dem Versuch, über Kontrollmechanismen dem Hüftgelenk einen bestimmten Winkel $\alpha(\varphi)$ in Abhängigkeit von der Orientierung φ des ersten Pendelkörpers zuzuweisen, kam es jedoch relativ schnell zu Unstimmigkeiten. Bei der Integration tauchten regelmäßig plötzliche, kurze, steile Anstiege in den Winkelverläufen auf, die in den Ableitungen große Spitzen hinterließen. Vermutlich waren während der Integration die eigentlich durch Zwangsbedingungen aneinandergeketteten Körper soweit von einander weggedriftet, daß die Zwangsbedingungen als verletzt galten und DADS das System mit einem plötzlichen Ruck wieder in eine, die Zwangsbedingungen erfüllende Position brachte. Dieses Problem ließ sich bei einem Wechsel zu einem äquivalenten 2-Dimensionalen Modell eliminieren, wobei bei gleichen Modell- und Eingabedaten aus der Integration eine von der ersten (3D) stark abweichende, aber realistisch aussehende Kurve folgte. Die Verwendung von DADS-3D war somit ausgeschlossen.

Da das bisherige Doppelpendel-Turner-Modell relativ einfach und die Kontroll- und Steuerungsmöglichkeiten in DADS andererseits relativ begrenzt, umständlich und unsicher waren, wurde kurzerhand beschlossen, ein eigenes Programm unter Verwendung eines Standard-Integrators zu schreiben. Wie sich später zeigte, wurde dadurch die Rechenzeit für die Integration einer kompletten Riesenfelge um einen Faktor 10-100 auf wenige Zehntel Sekunden verbessert. Außerdem waren so alle Größen der Bewegung direkt und ohne Probleme zugänglich.

2.2 Das verwendete Turner-Reck-Modell

Das bereits oben angedeutete zweidimensionale zweigliedrige Turner-Modell besteht aus zwei Starr-Körpern, dem Oberkörper (Kopf, Rumpf, Arme) und dem Unterkörper (Beine), die über ein Scharniergelenk im Bereich der Hüfte gekoppelt sind. Über ein zweites Scharniergelenk ist der Oberkörper an einer starren Reckstange aufgehängt. (Die Bewegungsgleichungen für das Modell werden im Anhang A.2 ausführlich hergeleitet. Eine Skizze (Abb.3.1) und die Angabe der Bewegungsgleichungen folgen im nächsten Kapitel) Eine Behandlung des Problems in drei Dimensionen ist vorerst nicht notwendig, da die Bewegung sich bei korrekter Ausführung nur in der Sagittal-Ebene abspielt. Um dem Modell-Turner einen Anreiz zum Bewegen zu geben, wurde zusätzlich ein Reibungsmoment M_R eingebaut, das die Gleitreibung zwischen Reckstange und Reckriemen (Händen) simuliert und die Bewegung soweit abbremst, daß ohne aktives Eingreifen des Turners keine Umrundung der Stange möglich ist. (siehe auch [Bau76])

Auf weitere, eigentlich stark interessierende Einzelheiten der Modellierung wie eine elastische Reckstange und die Unterscheidung des Oberkörpers in Arme, Rumpf und Kopf bzw. des Unterkörpers in Ober- und Unterschenkel wurde vorerst verzichtet, um das Modell nicht zu kompliziert zu gestalten. Im Falle eines schnellen Erfolges bei der Vorwärtssimulation war geplant, diese Elemente nachträglich einzubauen und ihren Einfluß auf die Bewegung und die steuernden Momente zu untersuchen.

Die anthropometrischen Daten (Massen, Trägheitsmomente und Längen) der einzelnen Körperteile wurden mittels des Computerprogramms *Calcman* von U.Hahn ermittelt und auf das zwei-segmentige Modell umgerechnet [Hah93]. *Calcman* berechnet für ein durch Geschlecht, Körpergröße und Körpergewicht charakterisiertes Modell die durchschnittlichen anthropometrischen Daten an Hand von Regressionsgleichungen, die auf den von der NASA im *Anthropometric Source Book* zur Verfügung gestellten Daten beruhen [NAS78]. (Die Modelldaten und Rechnungen befinden sich im Anhang.)

Kapitel 3

Bewegungsgleichungen und Bewegungsaufgabe

3.1 Bewegungsgleichungen

[Die ausführliche Herleitung der Bewegungsgleichungen und eine Erläuterung der Variablen und Parameter in Abb 3.1 findet sich im Anhang A.2.]

Die Winkel φ und $\psi = \varphi + \alpha$ sind die einzigen Freiheitsgrade des Turnermodells. φ ist der Orientierungswinkel des Oberkörpers und wurde so gewählt, daß φ in der Ausgangsstellung (Handstand) der Riesenfelge den Wert Null hat. ψ ist der Orientierungswinkel des Unterkörpers. Anstelle von ψ wird jedoch im folgenden meist der Hüftwinkel α verwendet. α wurde so definiert, daß bei vollkommen gestreckter Haltung (wie z.B. im Handstand) $\alpha = 0$ gilt.

Mit den folgenden Abkürzungen

$$\begin{aligned} \Theta_1 &= I_1 + m_1 S_1^2 + m_2 L_1^2 \\ \Theta_2 &= I_2 + m_2 S_2^2 \\ \chi_1 &= m_2 L_1 S_2 \\ \chi_2 &= m_1 S_1 + m_2 L_1 \\ \chi_3 &= m_2 S_2 \end{aligned} \tag{3.1}$$

lauten die Bewegungsgleichungen :

$$\begin{pmatrix} \Theta_1 + \chi_1 \cos\alpha & \chi_1 \cos\alpha \\ \Theta_2 + \chi_1 \cos\alpha & \Theta_2 \end{pmatrix} \begin{pmatrix} \ddot{\varphi} \\ \ddot{\alpha} \end{pmatrix} =$$

$$\begin{pmatrix} M_R - M_\alpha + \chi_2 g \, \sin\varphi + \chi_1 \sin\alpha \, (\dot{\varphi} + \dot{\alpha})^2 \\ M_\alpha - \chi_1 \sin\alpha \, \dot{\varphi}^2 + \chi_3 g \sin(\varphi + \alpha) \end{pmatrix}, \tag{3.2}$$

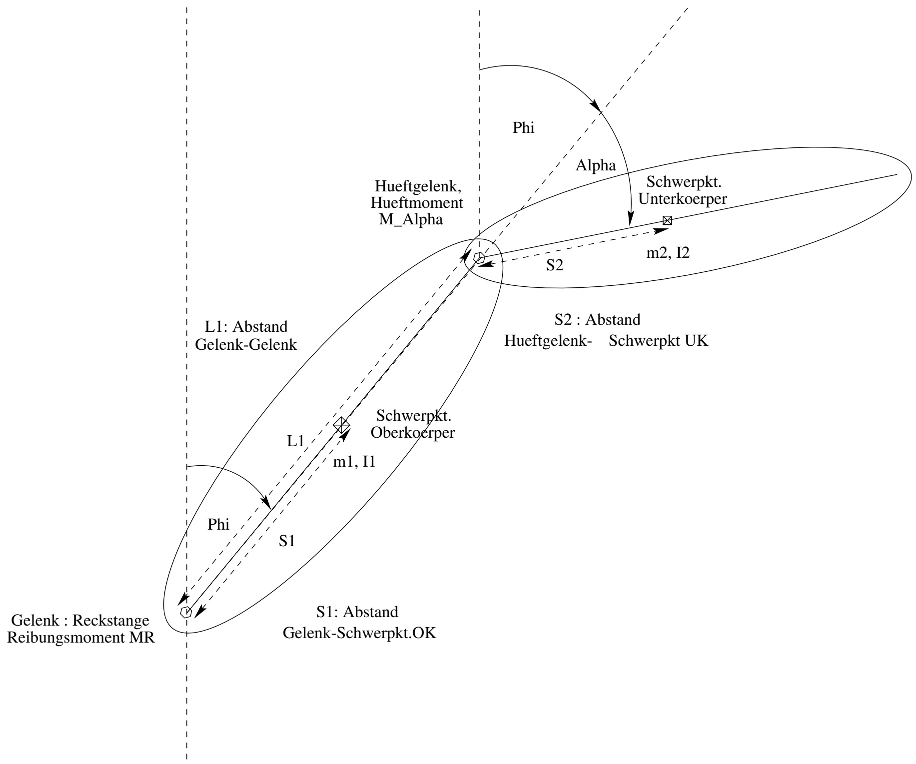


Abbildung 3.1: Skizze des verwendeten 2-dimensionalen, 2-gliedrigen Turnermodells. Oberkörper (Kopf, Rumpf, Arme) und Unterkörper (Beine) sind über ein Scharniergelenk im Hüftbereich miteinander gekoppelt. Über ein zweites Scharniergelenk ist der Oberkörper an einer starren Reckstange aufgehängt. (Die Bewegungsgleichungen für das Modell werden im Anhang A.2 ausführlich hergeleitet.)

wobei das Reibungsmoment M_R als kleine Störung der Gesamtbewegung angesehen werden kann und sich über die Schwerpunktsbeschleunigung $(\ddot{x}_{sp}, \ddot{y}_{sp})$ berechnet :

$$M_R \propto -(m_1 + m_2)\sqrt{\ddot{x}_{sp}^2 + (\ddot{y}_{sp} + g)^2}. \tag{3.3}$$

Addiert man die Bewegungsgleichungen für beide Körper, so erhält man die Differentialgleichung für den Gesamt-Drehimpuls L_{ges} :

$$\begin{aligned} L_{ges} &= \Theta_1\,\dot{\varphi} + \chi_1 \cos\alpha\,(2\dot{\varphi} + \dot{\alpha}) + \Theta_2\,(\dot{\varphi} + \dot{\alpha}) && (3.4)\\ \dot{L}_{ges} &= M_R + g\,\{\chi_2\,\sin\varphi + \chi_3 \sin(\varphi + \alpha)\}. && (3.5)\end{aligned}$$

Auf den Gesamt-Drehimpuls wirken nur die äußeren Drehmomente M_R (Reibungsmoment) und M_G (Schwerkraftmoment) :

$$\begin{aligned} M_G &= g\,\{\chi_2\,\sin\varphi + \chi_3 \sin(\varphi + \alpha)\} && (3.6)\\ M_{ext} &= M_R + M_G. && (3.7)\end{aligned}$$

Die inneren Drehmomente M_{in} gehen wegen *actio* = *reactio* in die Gleichungen für beide Körper mit unterschiedlichem Vorzeichen ein und fallen deshalb in der Summe wieder heraus. Es sind das Hüftmoment M_α und ein durch die (auf den zweiten Körper wirkende) Zentrifugalkraft verursachtes Moment :

$$M_{in} = M_\alpha - \chi_1 \sin\alpha\,\dot{\varphi}^2. \tag{3.8}$$

Das Hüftmoment M_α in Abhängigkeit der anderen Bewegungsgrößen lautet :

$$M_\alpha = (\Theta_2 + \chi_1\cos\alpha)\ddot{\varphi} + \Theta_2\ddot{\alpha} + \chi_1 \sin\alpha\,\dot{\varphi}^2 - \chi_3 g \sin(\varphi + \alpha). \tag{3.9}$$

3.2 Bewegungsaufgabe

2π-periodische Bewegung

Man kann die Riesenfelge als eine Umrundung der Stange durch den Turner aus der Position des Handstands in den Handstand definieren, wobei der Winkel φ ohne Unterbrechung den Bereich $\varphi = 0, ..., 2\pi$ durchlaufen muß. Ohne Unterbrechung bedeutet : $\dot{\varphi} > 0$. Es besteht daher während einer Riesenfelge ein eindeutiger (aber komplizierter) Zusammenhang zwischen $\varphi(t)$ und t.
Die Start- und Endbedingung der Bewegung (Handstand) lautet :

$$\alpha(\varphi = 0) = \alpha(\varphi = 2\pi) = 0. \tag{3.10}$$

Möchte man, daß der Turner für eine kurze Zeit im Handstand verweilt, so kommt eine weitere Bedingung hinzu :

$$\dot{\alpha}(\varphi = 0) = \dot{\alpha}(\varphi = 2\pi) = 0. \tag{3.11}$$

Fordert man zusätzlich, daß der Turner während der Bewegung keine Energie verliert, so folgt mit der Start-Winkelgeschwindigkeit ω :

$$\dot{\varphi}(\varphi = 0) = \dot{\varphi}(\varphi = 2\pi) = \omega. \tag{3.12}$$

Damit ist die Bewegung so festgelegt, daß sie (bei vollkommener Einhaltung der Randbedingungen) 2π-periodisch in φ fortgesetzt werden kann. Würde man die Zeit für einen Umlauf $T_{2\pi}$ im Vorraus kennen, so könnte man die Funktion auch in Abhängigkeit von der Zeit definieren. Da sich $T_{2\pi}$ jedoch bei Variation der Hüftbewegung ändert und erst nach Integration der Bewegung bekannt ist, ist die Definition einer Funktion auf dem Intervall $\varphi = 0, ..., 2\pi$ einfacher als auf dem unbekannten Intervall $t = 0, ..., T_{2\pi}$.

Die Bewegung wird vom Turner durch Variation des Hüftwinkels α bzw. durch das im Hüftgelenk wirkende Drehmoment M_α gesteuert. Gesucht ist somit eine 2π-periodische Steuer-Funktion $M_\alpha(\varphi)$, $\alpha(\varphi)$ oder $\ddot{\alpha}(\varphi)$, die die oben definierte Riesenfelgbewegung erzeugt.

Lösungsansatz und Lösungsproblematik

Welche der genannten Funktionen $M_\alpha(\varphi)$, $\alpha(\varphi)$ oder $\ddot{\alpha}(\varphi)$ man als Lösung bestimmt, ist egal. Durch jede wird die Bewegung des Systems in der Zeit eindeutig festgelegt und somit lassen sich (notfalls durch numerische Integration der Bewegungsgleichungen) alle anderen interessierenden Größen (im Rahmen der numerischen Genauigkeit) berechnen.

Da das Problem keine analytische Lösung zu besitzen scheint, wurde der Weg über die Optimierung eines parametrisierten Lösungsansatzes für die Funktion $M_\alpha(\varphi)$ gewählt:

$$M_\alpha(\varphi) = f(\vec{z}, \varphi) \tag{3.13}$$

wobei der Vektor $\vec{z}$ den verwendeten Parametersatz (bzw die Entwicklungskoeffizienten) enthält.

Ein naheliegender und interessanter Ansatz, der aber nicht weiter verfolgt wurde, ist die Entwicklung von $M_\alpha(\varphi)$ in eine endliche *Fourierreihe* :

$$M_\alpha(\varphi) = \sum_{n=0}^{N} (a_n cos(n\varphi) + b_n sin(n\varphi)). \tag{3.14}$$

Zeitmangel und die Ungewißheit, ob man bereits mit wenigen Parametern eine gute, nicht zu stark oszillierende Bewegung erzeugen kann, waren die Hauptgründe dafür, einen anderen Weg einzuschlagen.

Stattdessen wurde für $M_\alpha(\varphi)$ der einfache Ansatz einer *Linearen Interpolation* mit 2^n äquidistanten Stützstellen auf dem Intervall von 0 bis 2π gewählt. Die Funktionswerte an den Stützstellen bilden den oben angesprochenen Parametervektor $\vec{z}$ und sind die zu optimierenden Parameter. Die Idee ist, ausgehend von 4 Stützstellen (= 4 Parameter) durch Optimierung eine gute Lösung des Problems zu finden und diese durch Einfügen weiterer Stützstellen zu verfeinern. Die neuen Stützstellen werden dabei jeweils in der Mitte zwischen den bisherigen eingefügt und erhalten den Mittelwert der Nachbarwerte als Startwert. Verfeinerungen von 4 über 8, 16 bis 32 Stützstellen wurden durchgeführt.

Ein allgemeines Problem bei dieser Art von Aproximation ist der Nebeneffekt, daß abgeleitete und verwandte Größen durch die Aproximation beeinflußt werden. Im untersuchten Fall konnten bei der Funktion $\ddot{\alpha}(\varphi)$ an den Stützstellen kleine Spitzen beobachtet werden, die jedoch bei Verbesserung (Glättung) der Lösung nachließen.

Erstes Ziel der Optimierung war die möglichst exakte Einhaltung der formulierten Randbedingungen.

Redundanz des Problems

Da der Turner die Hüftbewegung (zumindest in einem gewissen Rahmen) beinahe beliebig gestalten kann, kann es unendlich viele Lösungen zu dem Problem geben. Es interessieren aber nur die biomechanisch sinnvollen. Man ist daher gezwungen, durch die Wahl einer oder mehrerer begründbarer Zusatzbedingungen die Lösungsmenge einzuschränken und den Optimierer bei der Parametersuche in die richtige Richtung zu lenken. Zu nennen sind dabei zum einen sinnvolle Einschränkungen für die Bewegungsgrößen $\alpha, \dot{\alpha}, \ddot{\alpha}$ und M_α und zum anderen bestimmte Minimierungskriterien wie die Minimierung der verbrauchten Energie oder die Minimierung des Jerk. Alle diese Zusatzbedingungen können in die Bewertungsfunktion für den Optimierer eingebaut werden.

Kapitel 4

Programmierung

Aufgabe des Programms ist es, ausgehend von einem Start-Parametervektor $\vec{p}_0$, der die Werte von $M_\alpha(\varphi)$ an äquidistanten Stützstellen im Intervall $[0, 2\pi]$ für die Lineare Interpolation von $M_\alpha(\varphi)$ enthält, durch Optimierung einen neuen Vektor zu finden, der die Bewegungsaufgabe im Sinne der Bewertungsfunktion besser löst als $\vec{p}_0$.

4.1 Der Optimierer

Verwendet wurde der *Downhill-Simplex*-Algorithmus aus dem Buch *Numerical Recipes in C* [PTVF89]. Die dazugehörigen C-Routinen heißen *amoeba.c* und *amotry.c* und sind im Anhang aufgelistet. [Weitere Informationen zu den *Numerical Recipes* finden Interessierte im Internet unter der Adresse : *http://www.nr.com.*]

Das allgemeine Optimierungsproblem

Beim allgemeinen Optimierungsproblem versucht man, zu einer gegebenen, positiven Funktion $f(\vec{z}) \geq 0$, der *Bewertungsfunktion*, ein globales oder lokales Minimum zu finden. Hat $\vec{z}$ die Dimension zwei, so kann man sich das Problem an Hand eines dreidimensionalen Surface-Plots $(x, y, f(x, y))$ veranschaulichen. Die Minima liegen dann in den Tälern und Mulden der durch die Funktion $f(x, y)$ definierten Landschaft. Hat der Parameterraum eine höhere Dimension als zwei, so versagt die Anschauung. Man kann in diesem Fall aber an Hand eines Schnittes $(\eta, f(\vec{z}(\eta)))$ entlang einer Parameterlinie wie beispielsweise

$$\vec{z}(\eta) = \vec{z}_a + \eta\,(\vec{z}_b - \vec{z}_a)$$

einen kleinen Eindruck vom Funktionsverlauf gewinnen.

Im behandelten Fall ist die Bewertungsfunktion $f(\vec{z})$ nicht bekannt. Es existiert lediglich ein Programm, das nach einigen Sekunden Rechenzeit zu einem gegebenen Eingabe-Vektor $\vec{z}_{try}$ die zugehörige Bewertung $(f(\vec{z}_{try}))$ liefert. Die Dimension des Parameterraums variiert zwischen $n = 4$ und $n = 64$. Das Problem kann daher weder analytisch durch Differenzieren von $f(\vec{z})$ noch durch Durchprobieren aller möglichen Vektoren gefunden werden. Als einzige Möglichkeit bleibt die Verwendung eines möglichst *intelligenten* Programms, das durch geschicktes Probieren das Minimum selbstständig findet.

Der *Downhill-Simplex*-Algorithmus

Beim verwendeten *Downhill-Simplex*-Algorithmus werden $n+1$ Startpunkte $\vec{p}_i$ als Eingabe verlangt, die einen nicht-degenerierten Simplex formen müssen. (Ein *Simplex* ist ein Körper aus $n+1$ Punkten in n Dimensionen wie z.B. ein Dreieck in der Ebene. *Degenerierte* Dreiecke sind Punkte und Geraden.) Ausgehend von einem einzelnen Startvektor $\vec{p}_0$ kann man leicht durch Addition von Basisvektoren $\vec{e}_i$ einen akzeptablen Simplex erzeugen :

$$\vec{p}_i = \vec{p}_0 + \lambda_i\ \vec{e}_i \qquad (i = 1, ..., n), \tag{4.1}$$

wobei die Parameter $\lambda_i \neq 0$ dem Problem angepaßte Faktoren sein sollten. Im Programm wurde der Start-Simplex mit einem einheitlichen *Variationsradius* λ_0 erzeugt, d.h.

$$\vec{p}_i = \vec{p}_0 + \lambda_0\ \vec{e}_i \qquad (i = 1, ..., n). \tag{4.2}$$

Der $(n+1)$-dimensionale Vektor $\vec{y}$ enthält die zu den einzelnen Punkten gehörenden Funktionswerte :

$$\vec{y} = (f(\vec{p}_0), f(\vec{p}_1), ..., f(\vec{p}_n)). \tag{4.3}$$

Der Einfachheit halber definiert man noch die Größen :

$$\vec{p}_{sum} = \sum_{i=0}^{n} \vec{p}_i \tag{4.4}$$

$$\vec{p}_s = 1/n\ (\vec{p}_{sum} - \vec{p}_{ihi}). \tag{4.5}$$

$\vec{p}_{sum}$ ist das $(n+1)$-fache des Mittelpunktvektors für den *Simplex*, $\vec{p}_{ihi}$ ist der schlechteste Punkt von allen und $\vec{p}_s$ ist der Mittelpunkt (oder Schwerpunkt) des durch die n besten Punkte definierten n-Ecks.

Das Ausprobieren eines neuen Punktes geschieht durch die Routine *amotry.c*. Variiert wird während eines Optimierschritts nur $\vec{p}_{ihi}$, der zu Beginn des Optimierschritts schlechteste Punkt. Die Variation $\vec{p}_{try}(c)$ verläuft auf der Verbindungsgeraden zwischen $\vec{p}_{ihi}$ und $\vec{p}_s$:

$$\vec{p}_{try}(c) = \vec{p}_s + c\ (\vec{p}_{ihi} - \vec{p}_s) \tag{4.6}$$

Der Parameter c kann je nach Situation die Werte -1, 0.5 und 2 annehmen. Die Bewertung wird in der Variablen y_{try} gespeichert :

$$y_{try} = f(\vec{p}_{try}). \tag{4.7}$$

Ist $\vec{p}_{try}$ besser als $\vec{p}_{ihi}$, so wird $\vec{p}_{ihi}$ sofort durch $\vec{p}_{try}$ ersetzt und $\vec{y}$ aktualisiert. Während eines Optimierschritts wird die Bezeichnung $\vec{p}_{ihi}$ für den neuen Punkt vorerst beibehalten, auch wenn dieser möglicherweise nicht mehr der schlechteste ist. Wird eine weitere Variation $\vec{p}_{try}$ berechnet, so wird der neue Wert für $\vec{p}_{ihi}$ und der alte für $\vec{p}_s$ genommen.

Ablauf eines Optimierungs-Schritts

Zu Beginn eines jeden Optimierungs-Schritts werden der beste $\vec{p}_{ilo}$, der schlechteste $\vec{p}_{ihi}$ und der zweit-schlechteste Punkt $\vec{p}_{inhi}$ bestimmt und die Abbruchbedingungen überprüft.
Wird keine der Abbruchbedingungen erfüllt, so erfolgt eine Variation mit $c = -1$, was einer Spiegelung von $\vec{p}_{ihi}$ an $\vec{p}_s$ entspricht (*Reflektion*, siehe Abbildung 4.1).

- Ist $\vec{p}_{try}$ besser als $\vec{p}_{ilo}$, so wird $\vec{p}_{ihi}$ durch $\vec{p}_{try}$ ersetzt und mit $c = 2$ ein weiterer Schritt in die gleiche Richtung probiert. Ist der neue Punkt wiederum besser als der vorherige, so kommt es zu einer Vergrößerung des Simplex. (*Reflektion* und *Expansion*).

- Ist $\vec{p}_{try}$ besser als $\vec{p}_{inhi}$, dann bleibt es bei der Spiegelung.

- Ist $\vec{p}_{try}$ schlechter als $\vec{p}_{inhi}$, so erfolgt der nächste Versuch mit $c = 0.5$ und liegt genau zwischen dem aktuellen $\vec{p}_{ihi}$ und $\vec{p}_s$) (*Kontraktion*). Fällt das Ergebnis schlechter aus als der Wert von $\vec{p}_{ihi}$, so zieht sich der *Simplex* um $\vec{p}_{ilo}$ herum zusammen und $\vec{y}$ wird aktualisiert (*Multiple Kontraktion*):

$$\vec{p}_i \longmapsto 1/2\ (p_i + p_{ilo}). \tag{4.8}$$

Ende des Optimierungs-Schritts.

Der Optimierer wird diesen Algorithmus solange wiederholen, bis eine vorgegebene Toleranzgrenze für die Bewertungsunterschiede aller Punkte unterschritten oder eine bestimmte Anzahl von Funktionsauswertungen gemacht wurde.

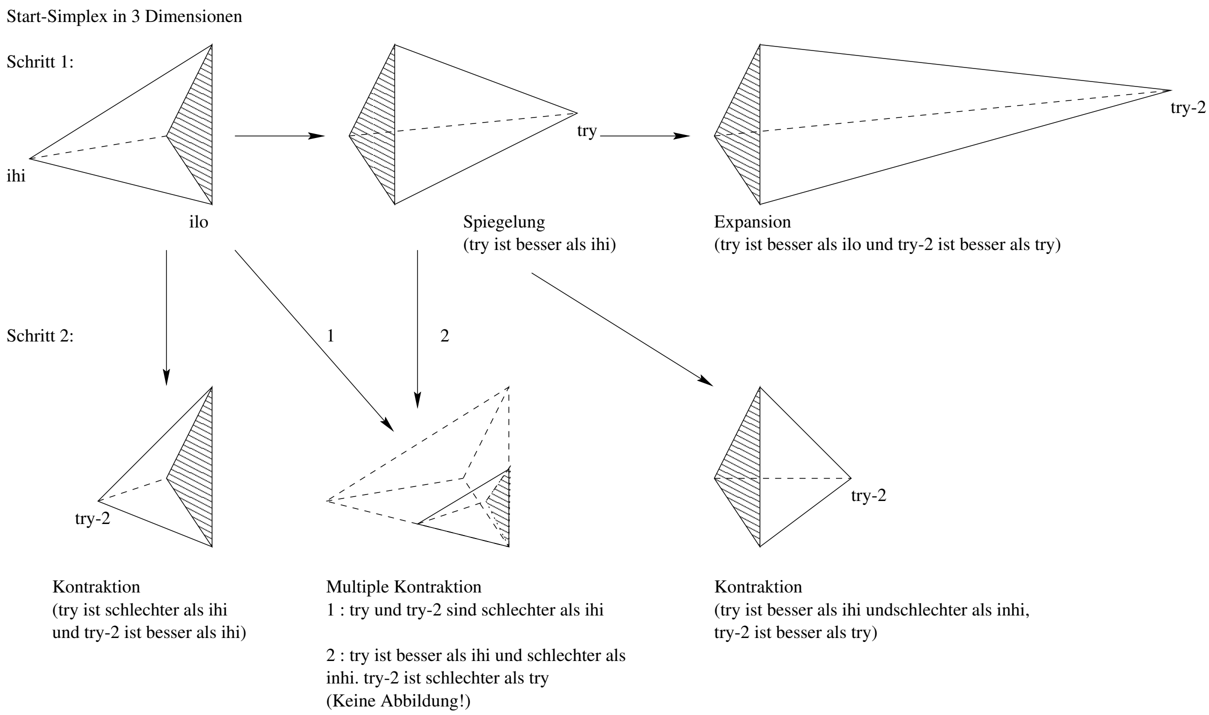


Abbildung 4.1: Optimierung in 3 Dimensionen : Der Simplex besteht aus 4 Punkten. Die Namen *ihi, inhi, ilo* und *try* werden wie im Programm verwendet und bezeichnen den schlechtesten, zweitschlechtesten, besten und den Probier-Punkt. Am Anfang des Optimierschritts probiert die *Amoeba* eine Spiegelung (Reflektion) von *ihi* am Mittelpunkt der schraffierten Fläche und bei einem guten Ergebnis (*try* besser als *ilo*) sogar eine Expansion. Ist hingegen *try* schlechter als *inhi*, so wird eine Kontraktion probiert. Geht diese Kontraktion schief, so folgt eine *Multiple Kontraktion* und der Simplex verkleinert sein Volumen und zieht sich um den Punkt *ilo* zusammen.

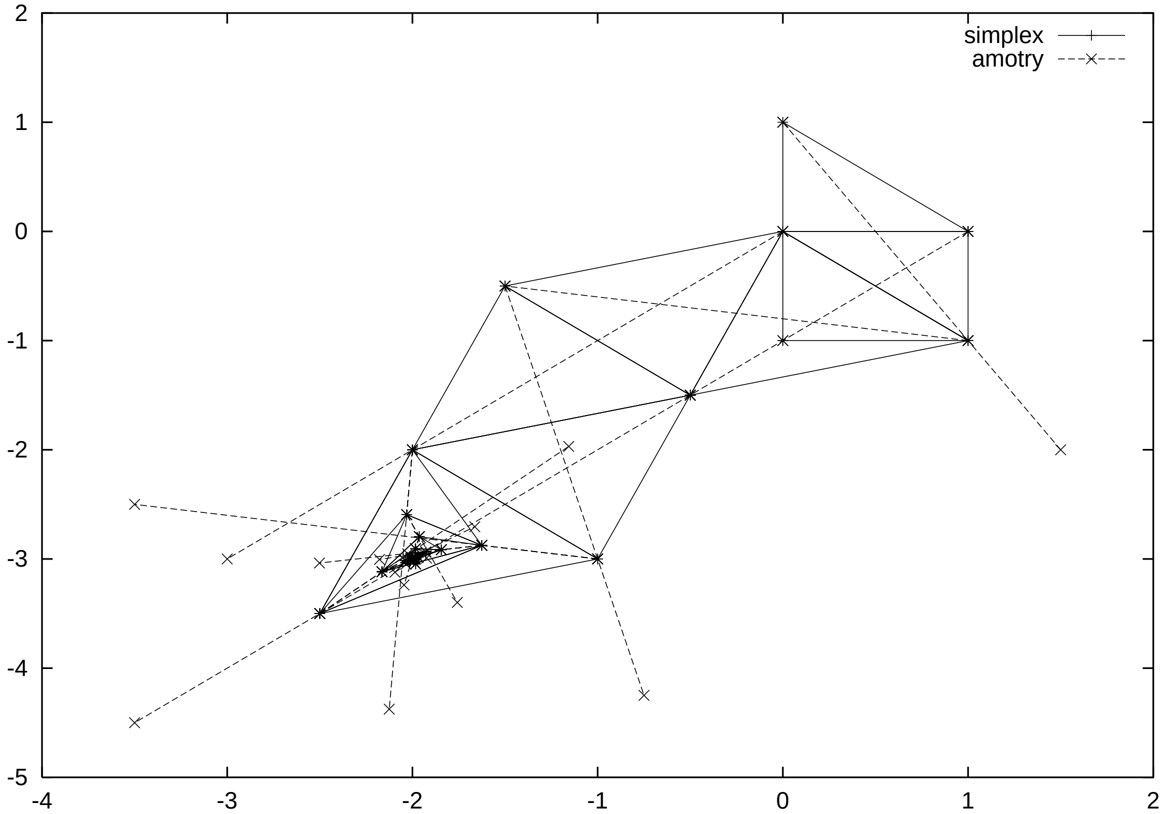


Abbildung 4.2: Der *Downhill-Simplex*-Algorithmus wurde mit der Funktion $f(x,y) = (x+2)^2 + (y+3)^2$ getestet. Der Start-Simplex besteht aus den 3 Punkten $(0,0), (1,0)$ und $(0,1)$ und bildet das kleine Dreieck ganz oben rechts. Die durchgezogenen Linien markieren die verschiedenen Entwicklungsstufen des *Simplex*, während er sich langsam dem Minimum bei $(-2,-3)$ nähert. Die gestrichelten Linien verdeutlichen die Variationsrichtung und führen vom jeweils schlechtesten Punkt $\vec{p}_{ihi}$ zum neuausprobierten Punkt $\vec{p}_{try}$ (*Spiegelung*) und eventuell weiter (mögliche *Expansion*).

4.2 Die Bewertungsfunktion

Die Aufgabe der Bewertungsfunktion ist es, die von einem Vektor $\vec{z}$ über das Hüftmoment $M_\alpha(\vec{z},\varphi)$ erzeugte Bewegung in Hinblick auf die Aufgabenstellung und weiterer Kriterien zu bewerten. $M_\alpha(\vec{z},\varphi)$ hängt von $\vec{z}$ in der Weise ab, daß die Komponenten von $\vec{z}$ die Funktionswerte für die Lineare Interpolation von $M_\alpha(\varphi)$ an äquidistanten Stützstellen im Intervall $[0, 2\pi]$ definieren.

Die Integration mit $M_\alpha(\vec{z},\varphi)$ liefert eine Bewegung $(\varphi(t), \alpha(t))$, die nach den folgenden Regeln bewertet wird:

Bewertung einer einfachen Riesenfelge

Der Modell-Turner startet zur Zeit $t = 0$ mit den Anfangswerten :

$$\begin{array}{ll} \varphi_0 = 0 & \alpha_0 = 0 \\ \dot{\varphi}_0 = \omega & \dot{\alpha}_0 = 0 \end{array}$$

[Referenzgeschwindigkeit : $\omega = 40 Grad/sec$]

Ziel des Turners ist es, nach einer Riesenfelge an der Stelle $\varphi = 2\pi$ die Werte

$$\dot{\varphi}/\omega = 1 \qquad \alpha = 0 \qquad \dot{\alpha} = 0$$

möglichst genau zu erreichen, ohne während der Bewegung die folgenden Wertebereiche zu verlassen.

$$\dot{\varphi} > 0 \qquad -\pi/4 \leq \alpha \leq \pi/2 \qquad \|\ddot{\alpha}\| \leq \text{Grenzwert}$$

[$\dot{\varphi} > 0$ verhindert eine Bewegungsumkehr, die bei einer Riesenfelge den Abbruch bedeutet und eine Unterscheidung zwischen Haftreibung und Gleitreibung voraussetzt. Haftreibung wurde bisher nicht in die Simulation integriert. Die Einschränkungen für α und $\ddot{\alpha}$ sind Vorgaben für die Körper-Haltung und die Hüftbeschleunigung. Bei anfänglichen Optimier-Versuchen ohne Jerk kam es immer wieder zu Lösungen mit stark oszillierendem α und Beschleunigungen im Bereich von 1000 bis 10000 Grad/sec^2, was nicht immer sehr realistisch schien. Durch Festlegung eines Grenzwerts können diese Bewegungen sofort als Lösung ausgeschlossen werden.]

Um möglichst effektiv zu arbeiten, unterscheidet die verwendete Bewertungsfunktion zwar zwischen erfolgreichen und fehlgeschlagenen Versuchen, bewertet aber beide. Wird während der Riesenfelge an einer Stelle φ_x eine Randbedingung verletzt oder die Bewegung abgebrochen, so erhält die Bewegung die Bewertung

$$F_{Bew} = 1000 * \{1 + (2\pi - \varphi_x)\} \tag{4.9}$$

Im Falle einer erfolgreichen Riesenfelge werden die Kontrollparameter an der Stelle $\varphi = 2\pi$ bestimmt und mit individuell einstellbaren Gewichten $c_{\dot{\varphi}}$, c_{α}, $c_{\dot{\alpha}}$, c_{Reib}, c_{Jerk}, c_{Moment} aufsummiert.

$$\begin{aligned} F_{Bew} \quad = \quad & c_{\dot{\varphi}} * \|\dot{\varphi}/\omega - 1\| \\ & + (180/\pi) * \{c_{\alpha} * \|\alpha\| + c_{\dot{\alpha}} * \|\dot{\alpha}\|\} \\ & (\, + c_{Reib} * \int M_R \, \dot{\varphi} \, dt \;\;) \\ & (\, + c_{Jerk} * 1/2 \int \left(\frac{d^3\alpha}{dt^3}\right)^2 dt \;\;) \\ & (\, + c_{Moment} * 1/2 \int M_{\alpha}^2 \, dt \;\;) \end{aligned} \tag{4.10}$$

Neben der Abweichung von den festgelegten Sollwerten können bei Bedarf noch die folgenden zusätzlichen Bewertungskriterien berücksichtigt werden.

$$Reibungsterm = \int M_R \, \dot{\varphi} \, dt \tag{4.11}$$

Der *Reibungsterm* beschreibt die dem Turner während der Bewegung durch Reibung verlorengegangene Energie. Reibungsminimierung bedeutet Minimierung der verbrauchten Energie = Energieminimierung.

$$Jerkterm = 1/2 \int \left(\frac{d^3\alpha}{dt^3}\right)^2 dt \tag{4.12}$$

Der Jerkterm integriert das Quadrat der 3. Zeitableitung (*Jerk*) des Hüftwinkels α über die ganze Bewegung. Je kleiner der *Jerk*, desto geringer ist die Änderung von $\ddot{\alpha}$ und desto ruhiger verläuft die Bewegung.

$$Momentterm = 1/2 \int M_{\alpha}^2 \, dt. \tag{4.13}$$

Der Momentterm ist ein Maß für den Kraftaufwand des Turners. Nicht alle Momente, die der Turner aufbringt, verrichten Arbeit und lassen sich mit einem Energie-Kriterium erfassen. Im Sport dienen Momente oft dazu, Bewegungen, die durch äußere und Zwangskräfte verursacht werden, durch Aufbringen eines Gegenmoments (Körperspannung) zu verhindern.

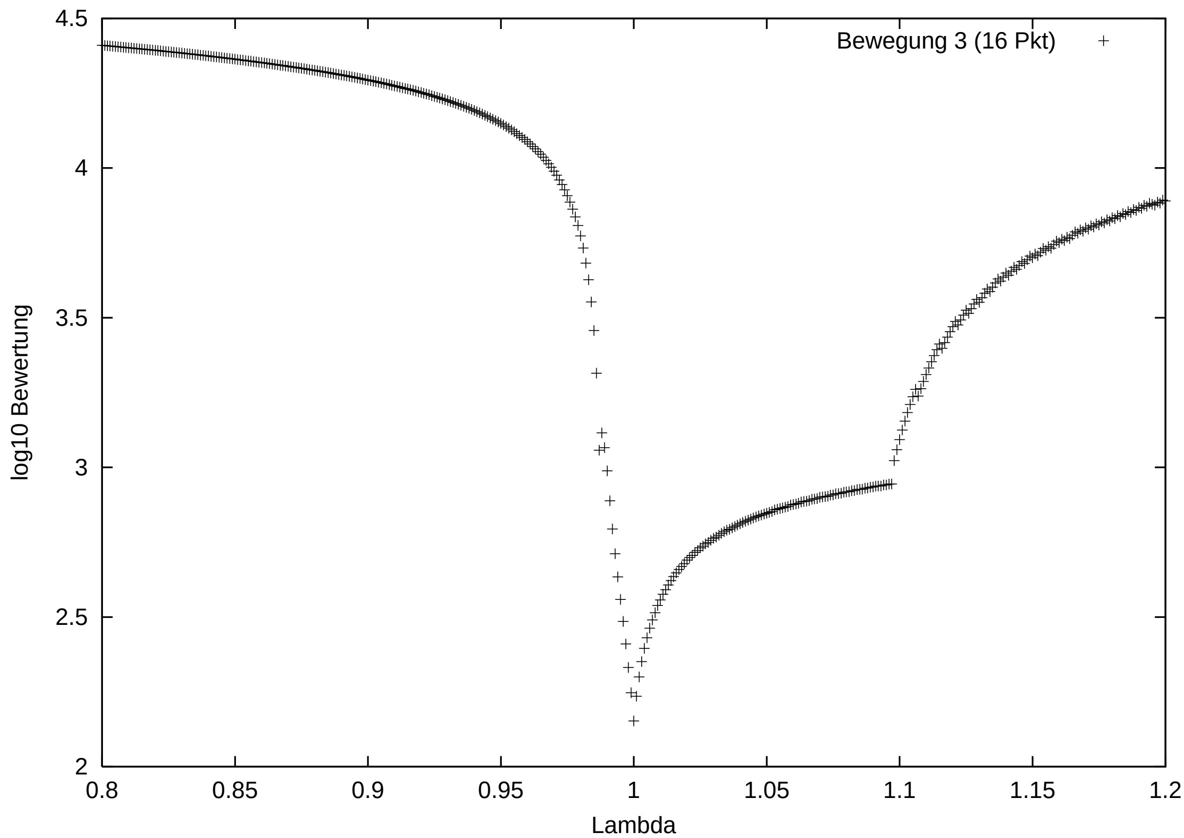


Abbildung 4.3: Um das Aussehen der Bewertungsfunktion zu veranschaulichen, wurde ein Schnitt entlang des Weges $\vec{z}(\lambda) = \lambda * \vec{z}_3$ mit $\lambda = 0.8, ..., 1.2)$ aufgenommen. $\vec{z}_3$ ist der Parametervektor, der die im folgenden Kapitel als *Bewegung 3* definierte Lösung mit 16 Stützstellen erzeugt. Die Schrittweite im Diagramm beträgt $\Delta\lambda = 0.001$ zwischen zwei benachbarten Punkten. Die Skaleneinteilung wurde zwecks besserer Übersichtlichkeit logarithmisch (log10) gewählt. Das Minimum liegt in einem sehr schmalen Bereich. Die Bewertungsfunktion wurde mit den folgenden Gewichtungen berechnet: $c_{\dot{\varphi}} = 100$, $c_{\alpha} = 10$, $c_{\dot{\alpha}} = 1$, $c_{Reib} = 0$, $c_{Jerk} = 0.01$, $c_{Moment} = 0$.

Bewertung einer Folge von Riesenfelgen

Ursprünglich wurde angenommen, daß ein Hüftmoment $M_\alpha(\varphi)$, dessen Bewegung die Anfangsbedingungen nach einer Riesenfelge optimal wiederherstellt, automatisch eine auf lange Sicht stabile (periodische) Bewegung erzeugt. Das stimmt leider nicht immer. Die meisten 'einfach-optimierten' Bewegungen brechen bereits nach zwei oder drei Felgen ab. Um stabilere Felg-Bewegungen zu erhalten, wurde die bisherige Bewertungsfunktion etwas modifiziert:

$$F^*_{Bew} = (\sum_{n=1}^{R_{Max}} F_{Bew}|_{(\varphi=2n\pi)}) * 10^{(20-3*R_{Max})}. \tag{4.14}$$

Die alte Bewertungsfunktion wird für jede erfolgreiche Runde einzeln ausgewertet und aufsummiert. R_{Max} bezeichnet die Anzahl der innerhalb der zulässigen Grenzen vollendeten Runden und legt bei Abbruch den Multiplikator fest. Dieser ist als eine Art Belohnung für besonders stabile Bewegungen (viele Runden) gedacht und sorgt für eine stärkere Abstufung (Niveau-Bildung) in der Bewertung. Bewegungen mit dem gleichen R_{Max} werden im Vergleich zu Bewegungen mit anderem R_{Max} alle ungefähr auf dem gleichen Niveau liegen.

Wie man in Diagramm 4.4 gut sieht, ist das lokale Minimum (5 Runden) so eng, daß eine Variation von $\vec{z}$ um 1 Promille bereits ausreicht, die Bewegung auf nur 1 Runde zu verkürzen.

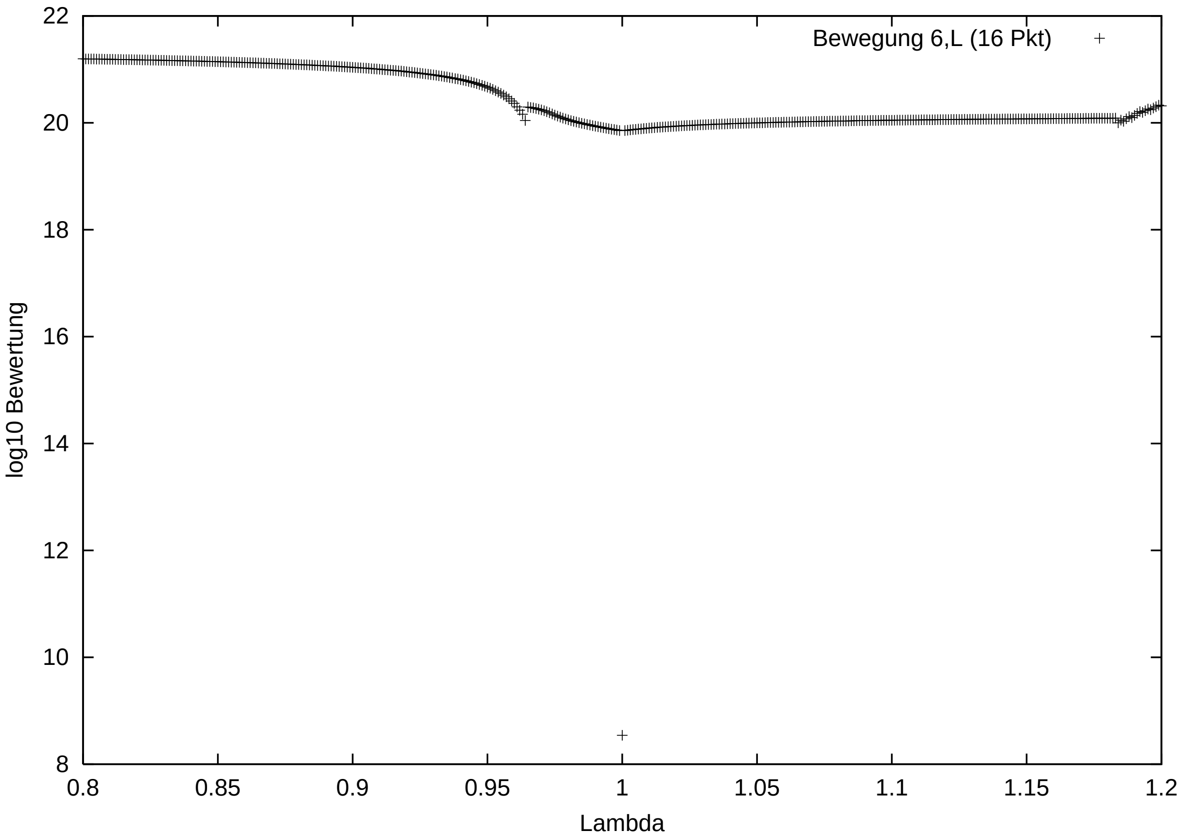


Abbildung 4.4: Log. Querschnitt durch $F^*_{Bew}(\vec{z})$ entlang des Weges $\vec{z}(\lambda) = \lambda * \vec{z}_6$ ($\lambda = 0.8;\ ...;\ 1.2$) mit Schrittweite $\Delta\lambda = 0.001$ zwischen benachbarten Punkten. $\vec{z}_6$ ist das Ergebnis von Optimier-Durchgang 6,L. Bewertet wurden bis zu 10 Riesenfelgen. Das Minimum ist ein einzelner Punkt (5 Runden), der sich deutlich von den anderen Punkten (alle nur 1 Runde) unterscheidet.

4.3 Das Hauptprogramm

Das verwendete Programm besteht im wesentlichen aus verschiedenen Ein- und Ausgaberoutinen, einem Integrator auf Basis des *odeint* zwecks Integration der Bewegungsgleichungen, an den die Bewertungsfunktion gekoppelt ist, sowie einer mehrstufigen Optimierungs-Schleife, die das Unterprogramm *amoeba* verwendet. Die Programmteile *odeint* und *amoeba* wurden aus dem Buch *Numerical Recipes in C* übernommen [PTVF89].

Im Kopf des Programms besteht die Möglichkeit, die Einstellung der Gewichte für die einzelnen Terme der Bewertungsfunktion zu verändern und einen Startvektor für die Optimierung von Hand einzugeben bzw einen File mit dem Startvektor zu definieren. Dieser Vektor repräsentiert die Funktion $M_\alpha(\varphi)$.

Die mehrstufige Optimierschleife baut sich wie folgt auf :

1. Die innerste Schleife erzeugt ausgehend vom Startvektor einen Startsimplex mit Kantenlänge λ. Dieser wird an den Optimierer übergeben, der nach einiger Zeit den besten Vektor zurückgibt, den er finden konnte. Startvektor und Ergebnis (Endvektor) werden miteinander verglichen. Sind die beiden Vektoren voneinander verschieden, so übernimmt der Endvektor die Rolle des Startvektors und die Prozedur wird wiederholt.

2. Sind Start- und Endvektor gleich, so wird das bisherige Ergebnis der Optimierung als File ausgegeben und die Kantenlänge λ für den Startsimplex halbiert und die innere Schleife 1 wiederholt. λ wird solange halbiert, bis eine untere Grenze für λ unterschritten wird. (Wie effektiv diese Schleife ist, konnte aus Zeitgründen noch nicht näher untersucht werden.)

3. Wurde die untere Grenze für λ unterschritten, wird, falls vorgesehen, die Anzahl der Stützstellen (Dimension des Vektors) verdoppelt und ausgehend vom besten Ergebnis ein neuer Startvektor erzeugt, der die gleiche Funktion $M_\alpha(\varphi)$ (mit doppelt soviel Stützstellen) repräsentiert. λ wird auf den Startwert zurückgesetzt und die Optimierung beginnt von Neuem bei Schritt 1.

In regelmäßigen Abständen wird das momentan beste Ergebnis nocheinmal integriert und bewertet und anschließend mit allen Einstellungen, Bewertungen und den Daten der erzeugten Bewegung in mehreren Files ausgegeben.

Kapitel 5

Ergebnisse

Im Verlauf der Arbeit wurden folgende Aufgabenstellungen untersucht :

- Einfache Riesenfelge : Die Bewegung wird optimiert, sodaß nach einer einzelnen Riesenfelge die Anfangsbedingungen bestmöglichst wiederhergestellt werden.
- Periodische Folge von Riesenfelgen : Ziel ist eine stabile Bewegung mit möglichst vielen, gleichbleibenden Riesenfelgen.
- Beschleunigende Riesenfelge : Der Turner startet mit einer geringen Winkelgeschwindigkeit und versucht, innerhalb einer Riesenfelge auf die vorgeschriebene Geschwindigkeit zu beschleunigen.
- Abbremsende Riesenfelge : Der Turner startet mit einer hohen Winkelgeschwindigkeit und versucht, innerhalb einer Riesenfelge auf die vorgeschriebene Geschwindigkeit abzubremsen.

Zu jeder Aufgabenstellung wurden mehrere Optimierdurchgänge mit verschiedenen Randbedingungen, Bewertungskoeffizienten und Stützstellenzahlen durchgeführt und in Form von *Stroboskop*-Diagrammen die besten Bewegungslösungen veranschaulicht. Die Bewegung des Turners verläuft in diesen Diagrammen immer im Uhrzeigersinn.

5.1 Einfache Riesenfelge

Der Turner versucht, nach einer Riesenfelge die Anfangswerte wieder möglichst genau zu erreichen, ohne während der Bewegung den erlaubten Wertebereich (Randbedingungen) zu verlassen.

1. Serie

M_α wird linear mit 4 Stützstellen aproximiert. Der Startvektor $\vec{p}_0$ für das Programm ist der Nullvektor. Der maximale Variationsradius für die Erzeugung des Startsimplexs wird mit 1 bzw 20 festgelegt, die Obergrenze für $\ddot{\alpha}$ mit 600, 2.000 bzw 10.000 Grad/sec^2. Mit diesen Einstellungen wurden 6 Optimierungen durchgeführt. Die Bewertungsfunktion lautete :

$$F_{Bew} = 100 * \|\dot{\varphi}/\omega - 1.0\| + (180/\pi) * (10\|\alpha\| + \|\dot{\alpha}\|)$$

und verzichtete auf eine Bewertung von Reibung, Jerk oder anderen Größen.

$\|\ddot{\alpha}\| \leq 600$ und Variationsradius 20

Die Bewegung scheitert knapp vor Vollendung der Riesenfelge. Durch die Einschränkung für $\ddot{\alpha}$ verläuft die Kurve $\alpha(\varphi)$ sehr flach und kann dem System nicht genügend Energie zuführen.

$\|\ddot{\alpha}\| \leq 600$ und Variationsradius 1

Der Optimierer findet keine Lösung, da die Einschränkung für $\ddot{\alpha}$ eine Barriere darstellt, die von der Amoebe mit einem Schritt der Länge 1 nicht überwunden werden kann.

$\|\ddot{\alpha}\| \leq 2.000$ und Variationsradius 20

Diese Bewegung zeichnet sich durch ein extremes Hüftmoment (Körperspannung) aus, das zum Großteil der Konstanthaltung des Hüftwinkels α dient, um die Einschränkung für $\ddot{\alpha}$ nicht zu verletzen. (Bewegung 3)

$\|\ddot{\alpha}\| \leq 2.000$ und Variationsradius 1

Der Optimierer findet aus dem gleichen Grund wie oben keine Lösung.

$\|\ddot{\alpha}\| \leq 10.000$ und Variationsradius 20

Der Turner erreicht $\varphi = 2\pi$. Die Bewegung verläuft mit einem sehr kleinen Moment M_α und sehr großer Beschleunigung $\ddot{\alpha}$ im Hüftgelenk. Der Turner folgt der Bewegung mangels Körperspannung in erster Linie passiv.

$\|\ddot{\alpha}\| \leq 10.000$ und Variationsradius 1

Ähnliches Ergebnis wie das vorige. Die *Amoebe* ist in der Lage, mit kleinen Schritten den Bereich hoher Beschleunigungen zu überwinden. (Bewegung 6)

Bemerkung

Bei diesen ersten Rechnungen kann man bereits zwei typische Extreme erkennen :

- Zum einen Bewegungen mit kleinem Hüftmoment M_α (geringe Körperspannung) und hohen (passiven) Beschleunigungen $\ddot{\alpha}$ und
- zum anderen Bewegungen mit großem Hüftmoment M_α (hoher Körperspannung) und kleiner (minimaler) Bewegung in der Hüfte ($\|\alpha\|$ klein).

Es ist anzunehmen, daß die natürliche Bewegung eines guten Turners irgendwo zwischen den beiden Extremen liegen wird.

Durch Einschalten der Zusatzbedingungen kann man dem Optimierer beibringen, auf diese Eigenschaften der Bewegung zu achten. Das folgende (Moment-)Integral über das Quadrat des Hüftmoments

$$1/2 \int M_\alpha^2 dt \tag{5.1}$$

ist ein Maß für die Körperspannung. Je kleiner, desto geringer die Belastung für den Turner. Das (Jerk-)Integral

$$1/2 \int (d\ddot{\alpha}/dt)^2 dt \tag{5.2}$$

hingegen ist ein Maß für die Änderung der Beschleunigung und führt zu einer Bewegung mit kleinem $\ddot{\alpha}$.

Beide Bedingungen schränken sich gegenseitig ein. Eine Bewegung mit kleinem Maximalwert für $\ddot{\alpha}$ hat meist eine hohe Körperspannung und somit ein großes Hüftmoment und umgekehrt. Man muß daher bei der Gewichtung der beiden Integrale einen Kompromiß finden, sodaß die resultierenden Lösungen der Bewegung eines guten Turners möglichst nahe kommen. In den folgenden Optimierungen wurde, wenn überhaupt, nur das Jerk-Integral mitbewertet.

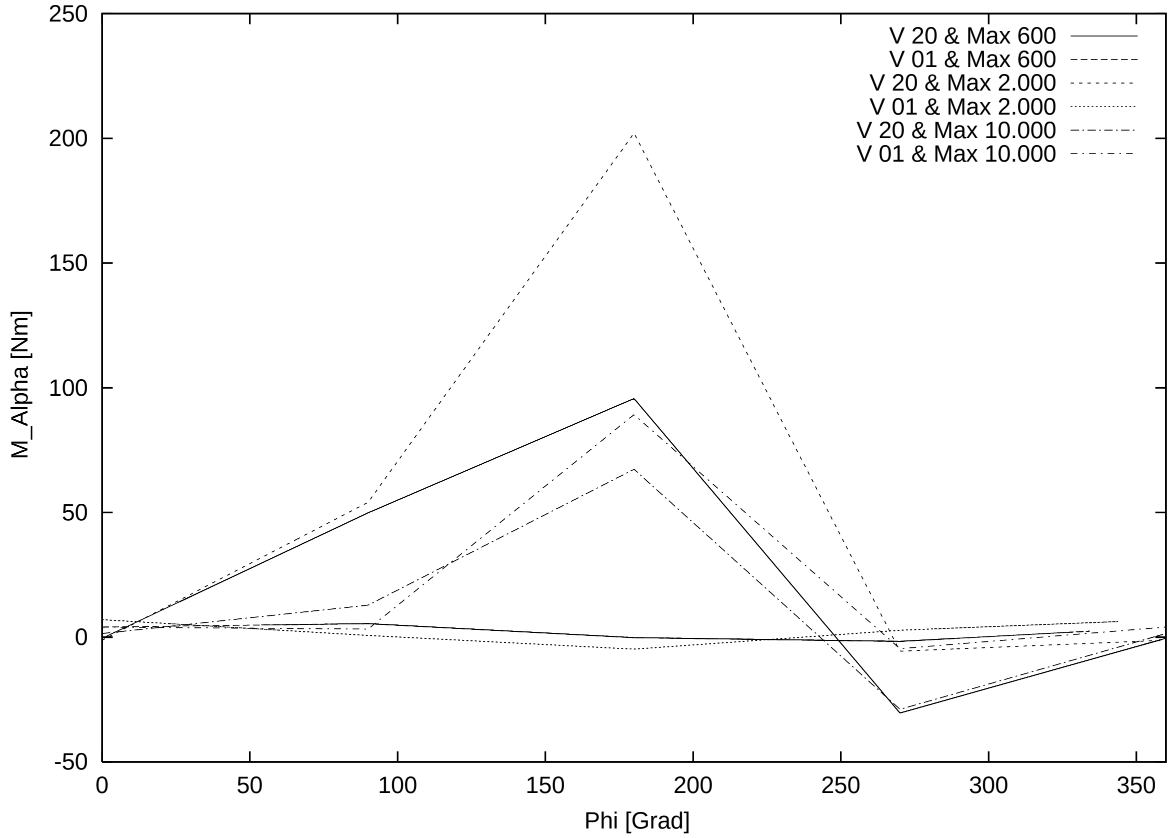


Abbildung 5.1: Hüftmoment $M_\alpha(\varphi)$: Die 6 Kurven sind das Ergebnis einer Optimierung mit verschiedenen Werten für die obere Schranke von $\ddot{\alpha}$ (600, 2.000 und 10.000 $Grad/sec^2$) und mit verschiedenen Variationsradien (20 und 1). Ausgangsfunktion war in allen Fällen die Null-Funktion. Die Kurven 3, 5 und 6 schaffen eine ganze Riesenfelge, Kurve 1 fast und die Kurven 2 und 4 gar nicht.

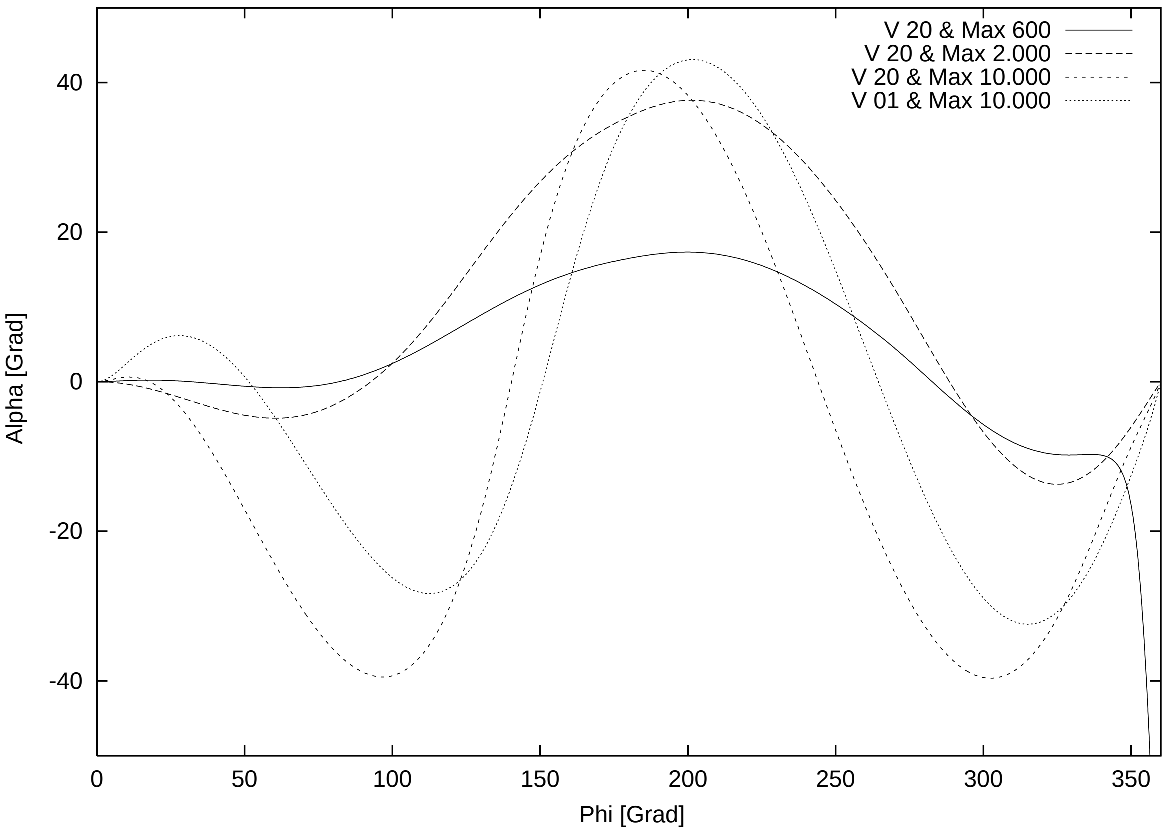


Abbildung 5.2: Winkel $\alpha(\varphi)$: Die Beschränkung für $\ddot{\alpha}$ läßt sich deutlich in den Kurven erkennen. Die ersten beiden Kurven haben eine wesentlich geringere Amplitude.

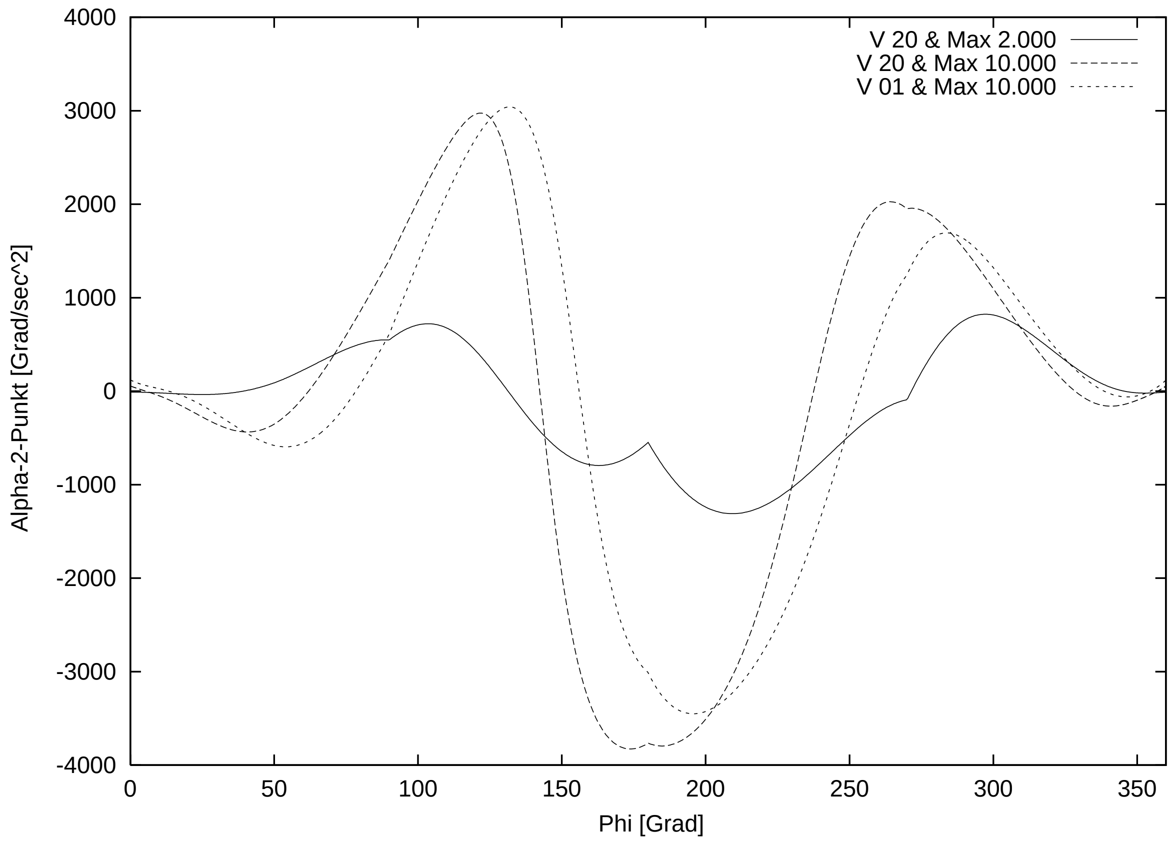


Abbildung 5.3: Winkelbeschleunigung $\ddot{\alpha}(\varphi)$: Die kleinen Spitzen sind eine direkte Folge der linearen Interpolation von M_α und treten nur an den Stützstellen auf.

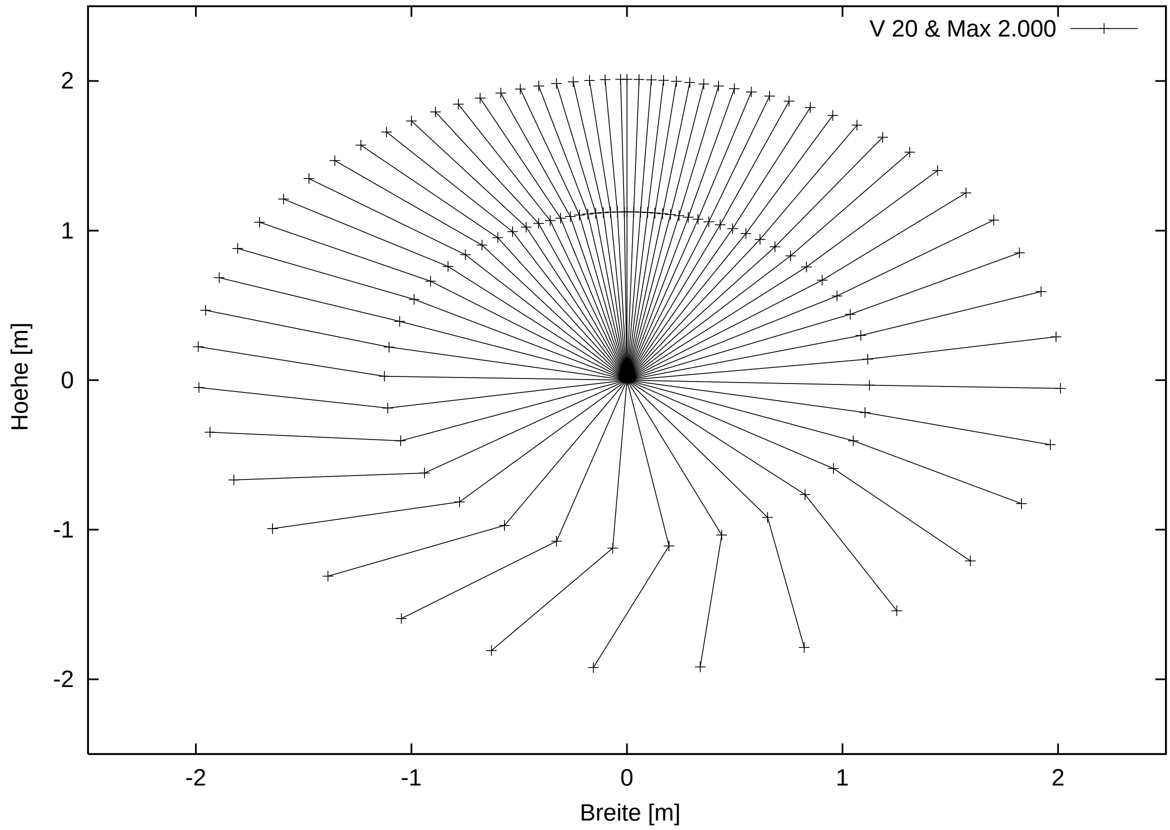


Abbildung 5.4: Stroboskop-Aufnahme von Bewegung 3 (25 Hz)

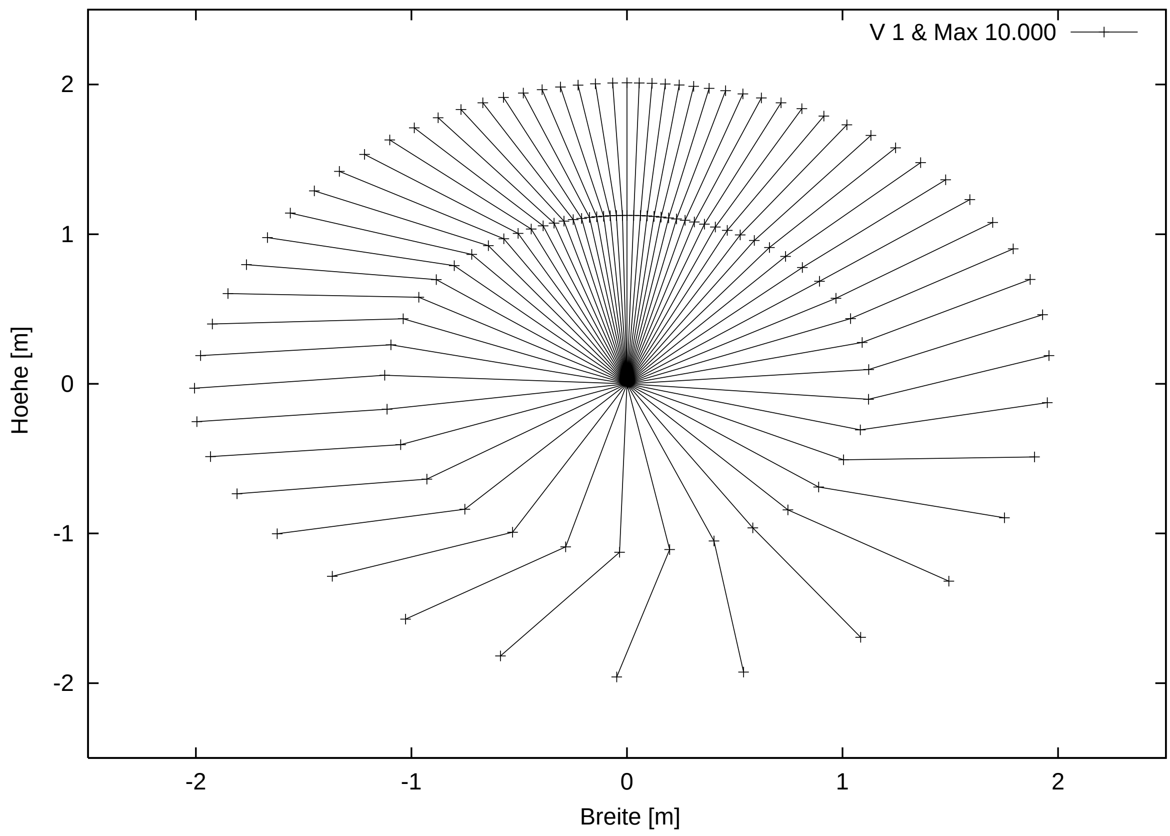


Abbildung 5.5: Stroboskop-Aufnahme von Bewegung 6 (25 Hz)

2. Serie

Um Konflikte zwischen dem Jerkterm und dem Momentterm in der Bewertungsfunktion zu vermeiden, wurde der Momentterm bei den folgenden Optimierungen nicht berücksichtigt.

Bewegung 3

Die Beschränkung für $\ddot{\alpha}$ wurde aufgehoben. Kurve 3 wurde auf 8 und 16 Stützstellen verfeinert. Das Ergebnis ist eine sehr genaue Reproduktion der Anfangswerte. Anschließend wurde eine Jerk-Optimierung durchgeführt.

Parameter und Ergebnisse

	Kurve 3	Kurve 3 (16 Pkt)	Kurve 3 (32 Pkt,J)
Startvektor	0	$\vec{z}_3$ (4 Pkt)	$\vec{z}_3$ (16 Punkte)
$c_{\dot{\varphi}}$	100	100	100
c_{α}	10	10	10
$c_{\dot{\alpha}}$	1	1	1
c_{Reib}	0	0	0
c_{Jerk}	0	0.0	0.01
Stützstellen	4	4,8,16	16,32
$\dot{\varphi}/\omega$	0.9995	1	0.999996
α (in Grad)	-0.00014	3.216e-15	-4.0978e-05
$\dot{\alpha}$ (in Grad/s)	0.4207	1.328e-16	7.484e-06
Reibung (in Nm)	61.9185	61.147	61.54
Jerk (in $s-5$)	7164.74	14219.5	763.31
Moment (in $(Nm)^2sec$)	4734.15	4759.3	4984.52

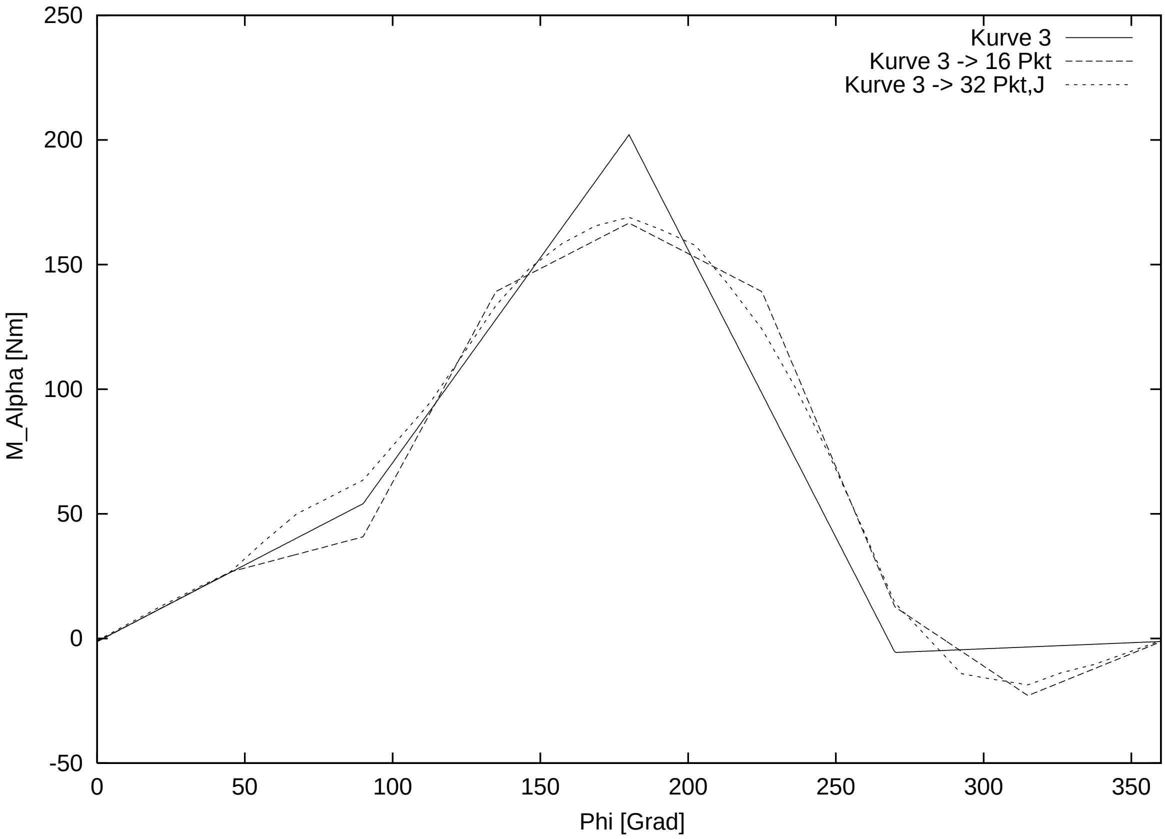


Abbildung 5.6: Hüftmoment $M_\alpha(\varphi)$: Die Vervierfachung bzw Verachtfachung der Stützstellenzahl führt zu einer runderen Kurve.

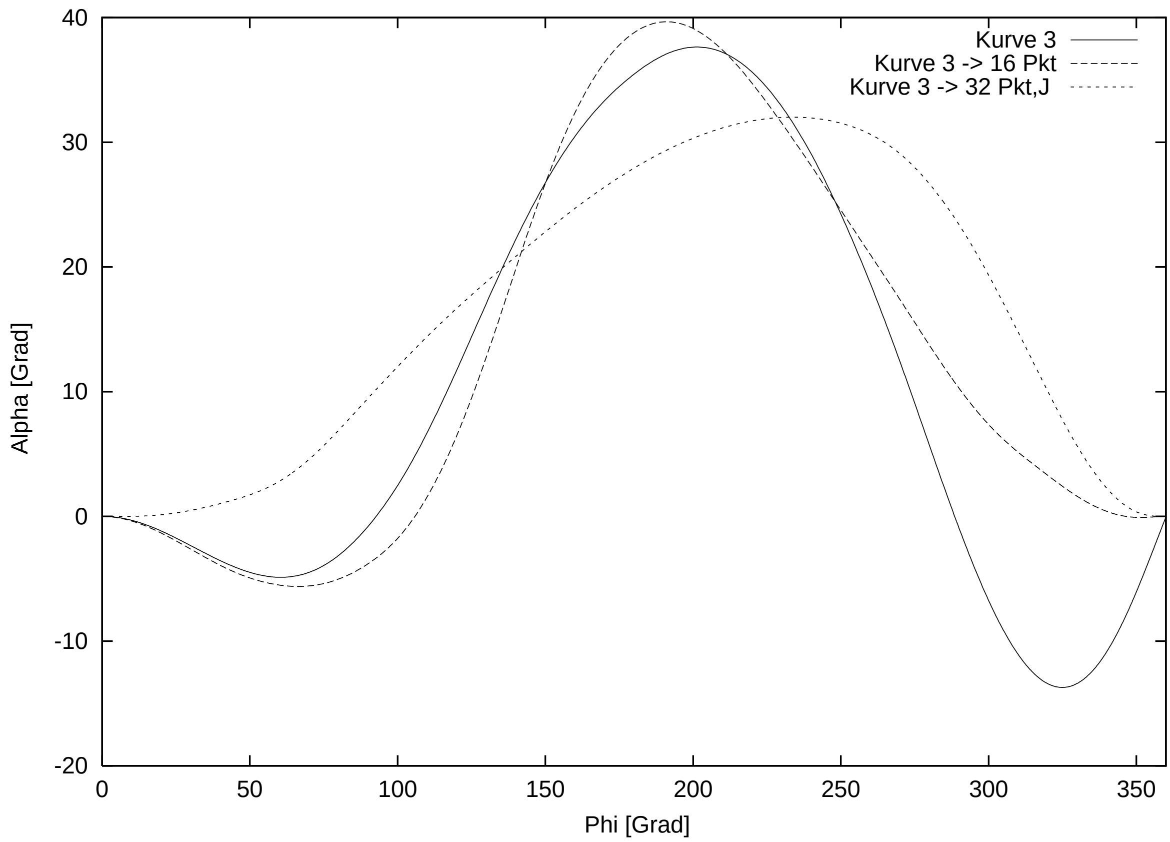


Abbildung 5.7: Winkel $\alpha(\varphi)$: Die Jerk-optimierte Kurve bewegt sich nur noch im Bereich $\alpha \geq 0$. Auch die 16-Punkte-Kurve ist ruhiger als die ursprüngliche Ausgangsfunktion.

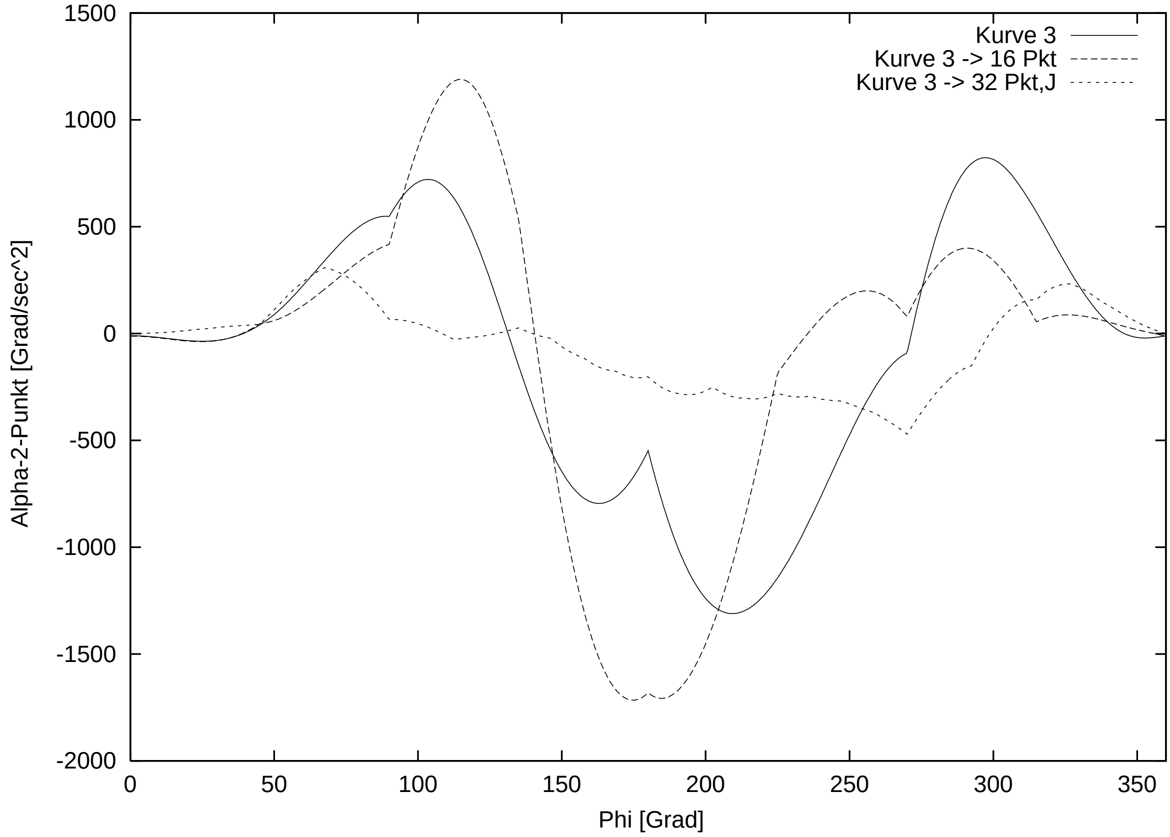


Abbildung 5.8: Winkelbeschleunigung $\ddot{\alpha}(\varphi)$: Die kleinen Spitzen sind eine direkte Folge der linearen Interpolation von M_α und treten nur an den Stützstellen auf. Die Jerk-optimierte Kurve zeichnet sich durch wesentlich geringere Beschleunigung aus.

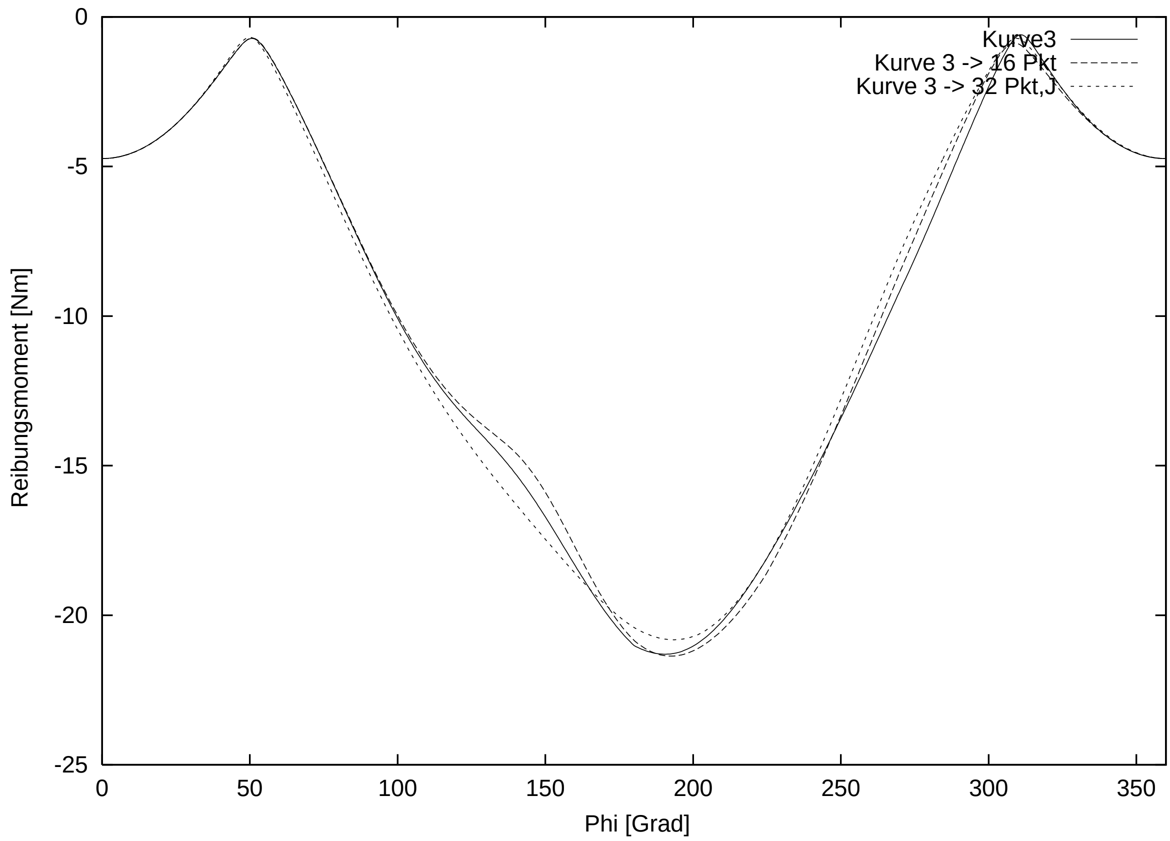


Abbildung 5.9: Reibungsmoment M_R

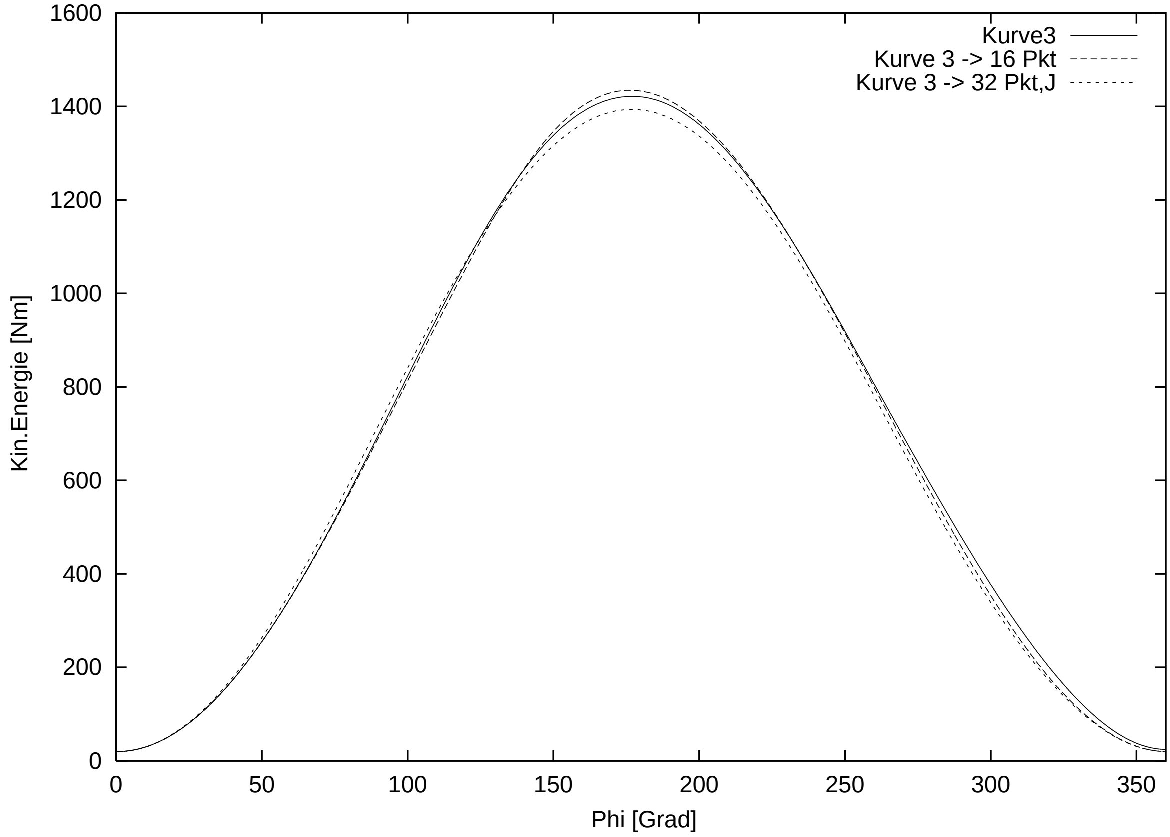


Abbildung 5.10: Kinetische Energie

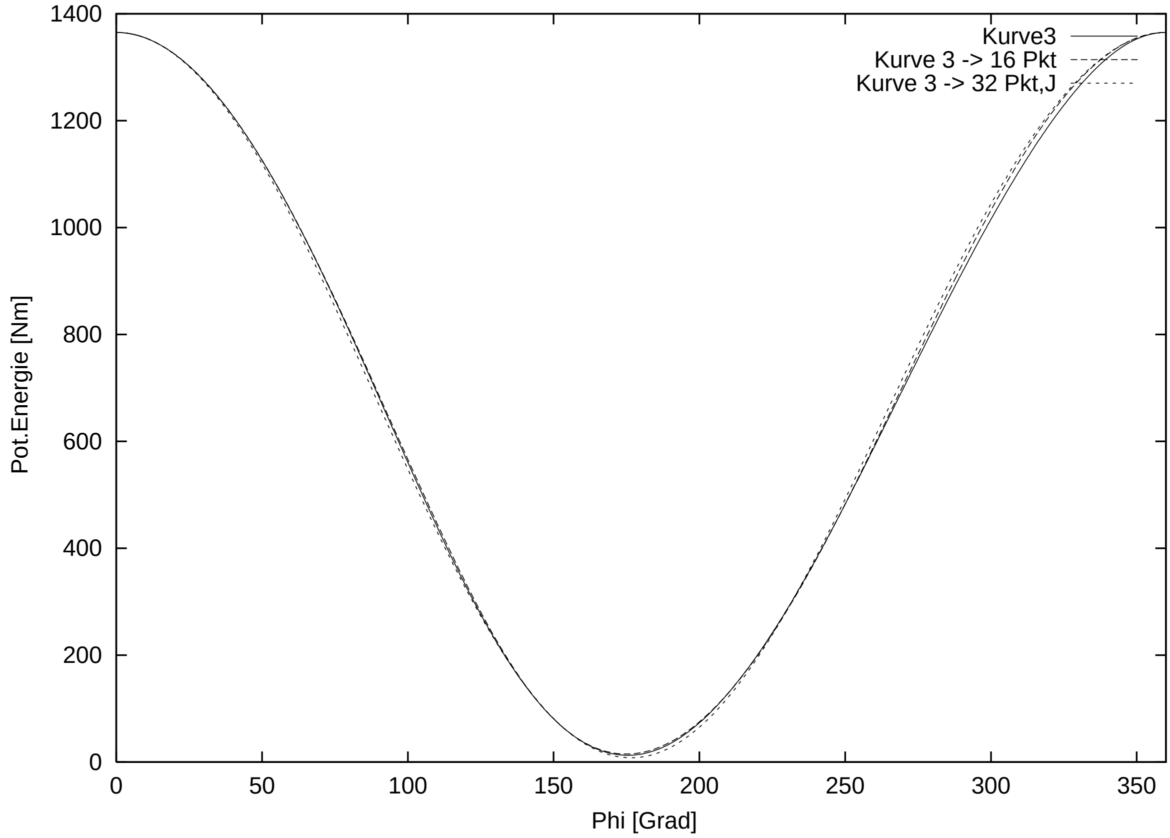


Abbildung 5.11: Potentielle Energie

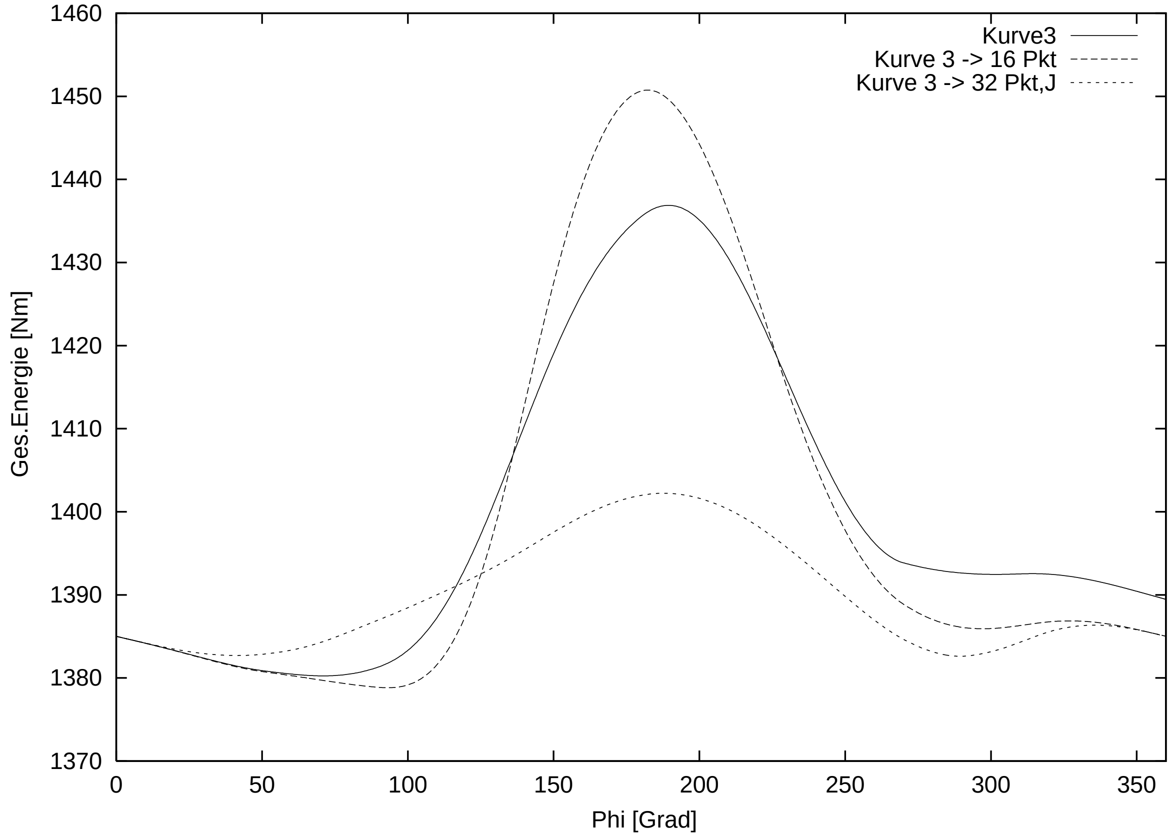


Abbildung 5.12: Gesamt-Energie

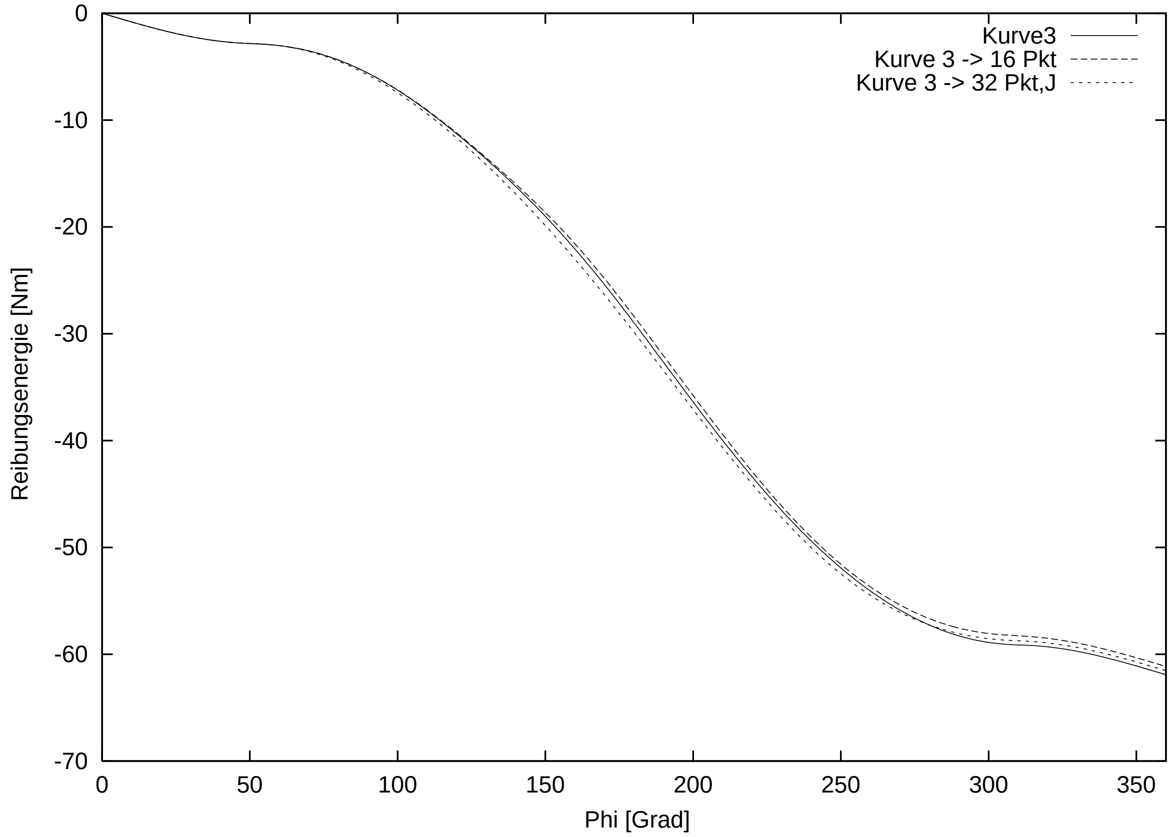


Abbildung 5.13: Reibungsverlust

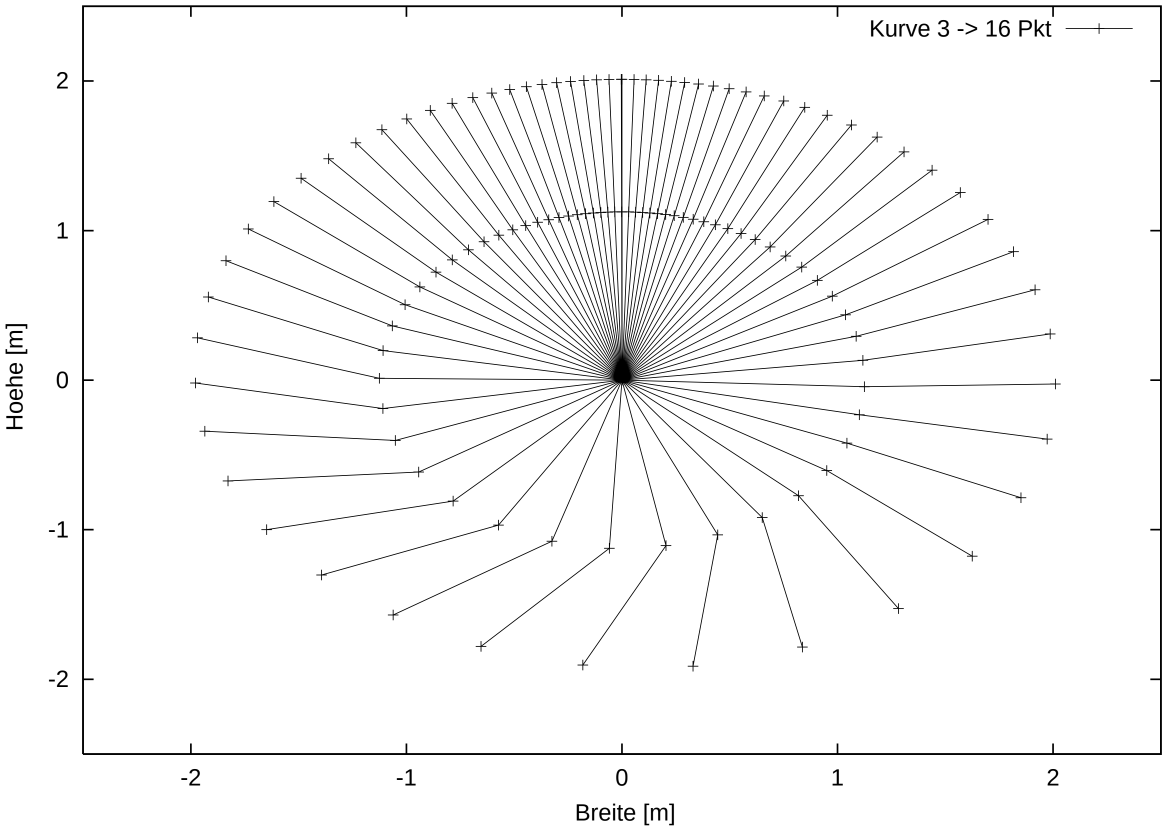


Abbildung 5.14: Stroboskop-Aufnahme von Bewegung 3 - 16 Pkt (25 Hz)

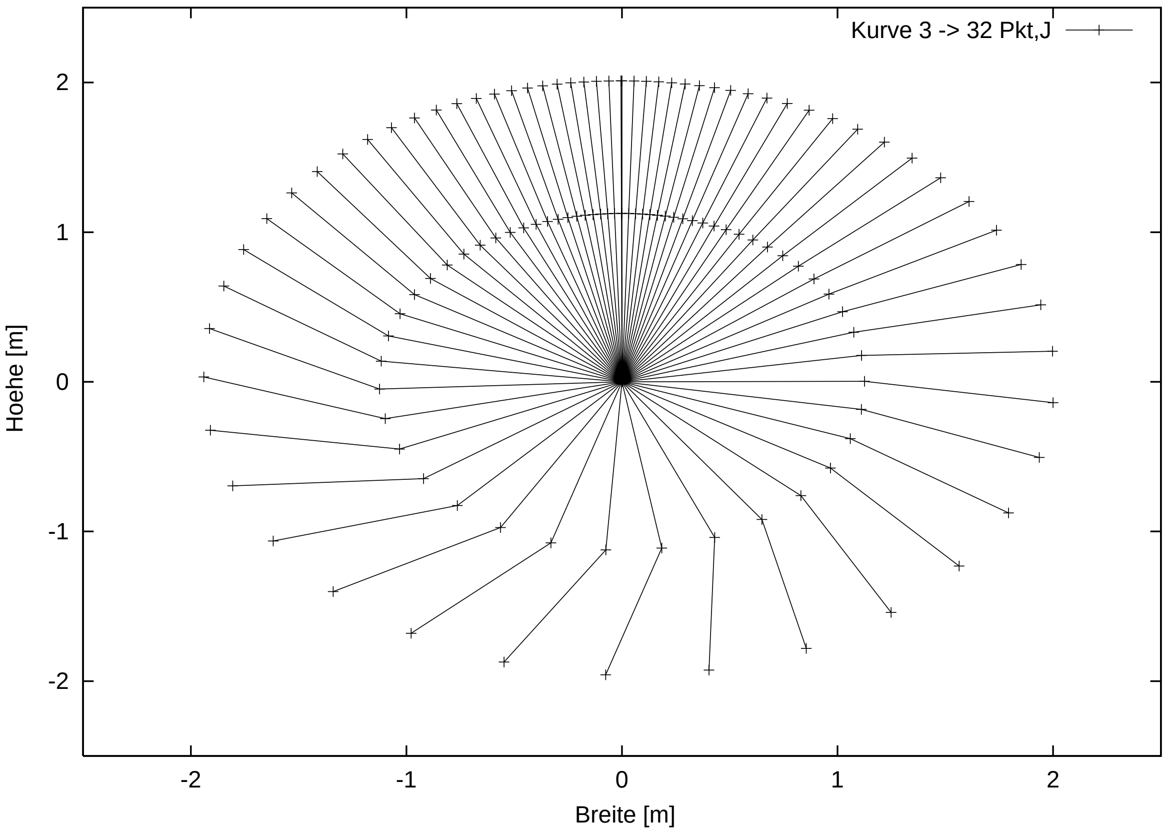


Abbildung 5.15: Stroboskop-Aufnahme von Bewegung 3 - 32 Pkt,J (25 Hz)

Bewegung 6

Kurve 6 wurde auf 8 und 16 Stützstellen verfeinert. Das Ergebnis ist genau wie bei Kurve 3 eine sehr genaue Reproduktion der Anfangswerte. Anschließend wurde eine Jerk-Optimierung durchgeführt, die zu einer leicht abgebremsten Bewegung führte. Die aus den Rechnungen resultierenden Bewegungen zeigen keine schönen Riesenfelgen, da der Turner seine Körperspannung im aufsteigenden Teil der Felge schnell verliert und ins Hohlkreuz fällt.

Parameter und Ergebnisse

	Kurve 6	Kurve 6 (16 Pkt)	Kurve 6 (32 Pkt,J)
Startvektor	0	$\vec{z}_6$ (4 Pkt)	$\vec{z}_6$ (16 Punkte)
$c_{\dot{\varphi}}$	100	100	100
c_{α}	10	10	10
$c_{\dot{\alpha}}$	1	1	1
c_{Reib}	0	0	0
c_{Jerk}	0	0	0.01
Stützstellen	4	4,8,16	16,32
$\dot{\varphi}/\omega$	0.8408	1	0.901731
α (in Grad)	-0.00115	-8.12262e-15	-0.00142648
$\dot{\alpha}$ (in Grad/s)	0.95988	2.35995e-15	2.86541e-05
Reibung (in Nm)	58,475	60.7276	61.3912
Jerk (in $s-5$)	70657.5	70720.8	5666.75
Moment (in $(Nm)^2 sec$)	807.32	1618.09	2422.3

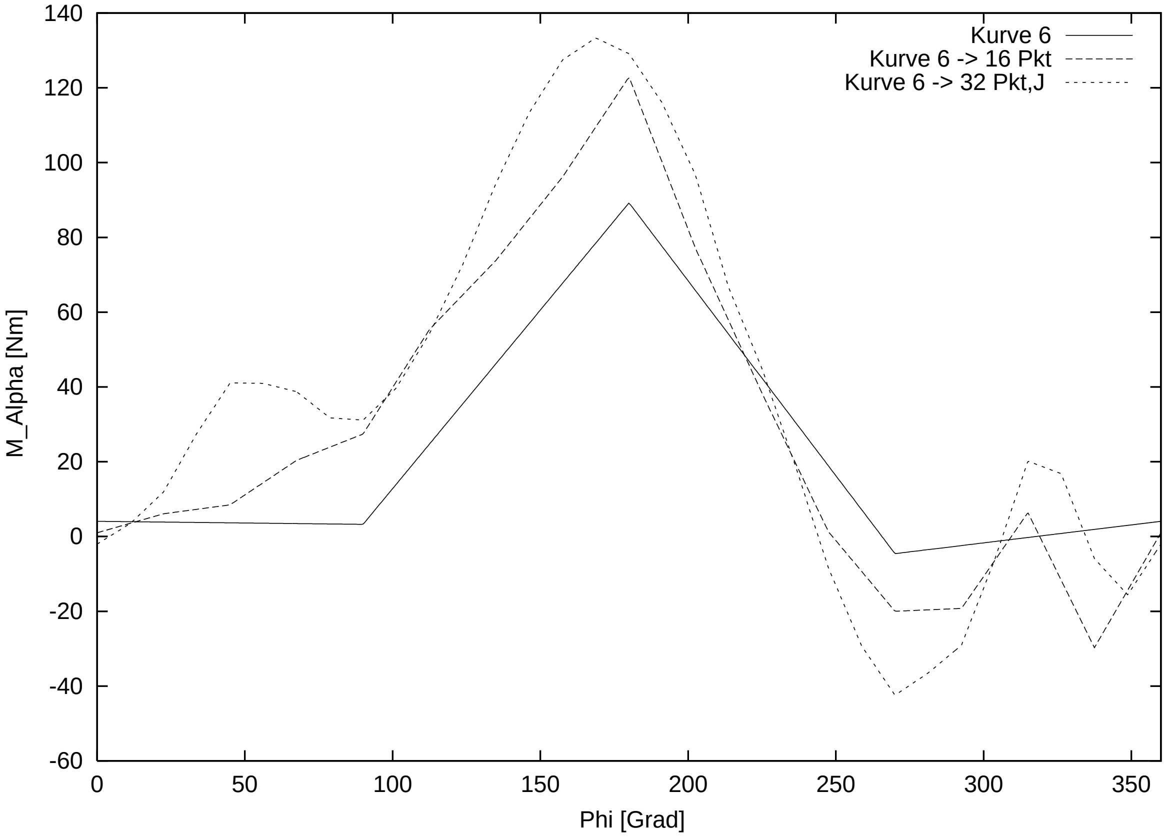


Abbildung 5.16: Hüftmoment $M_\alpha(\varphi)$: Die Jerk-Optimierung führt zu einer Abrundung der Kanten. Die zusätzlich an einigen Stellen eingebrachten Momente sorgen für eine starke Änderung der Hüftbewegung gegenüber den nicht-Jerk-Kurven.

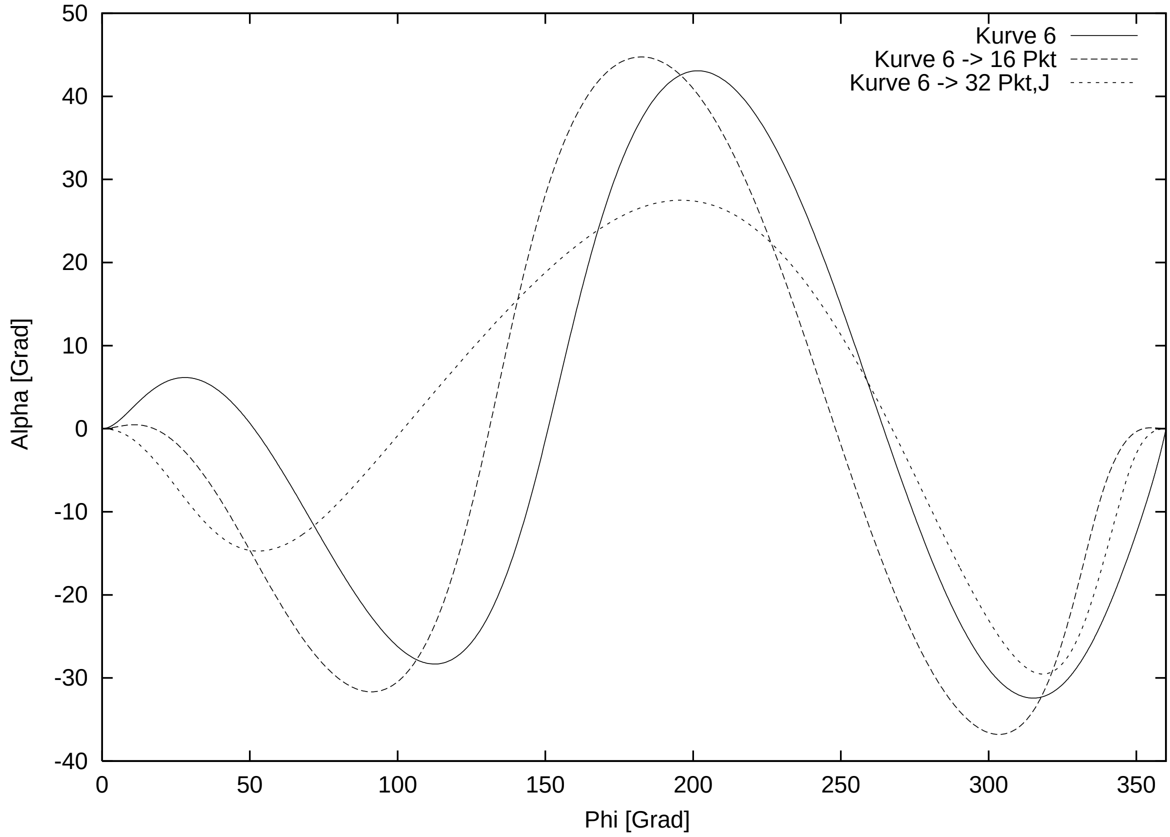


Abbildung 5.17: Winkel $\alpha(\varphi)$: Alle drei Bewegungen tendieren zu einem Hohlkreuz in der Aufstiegsphase, was normal bei der Riesenfelge vermieden werden sollte.

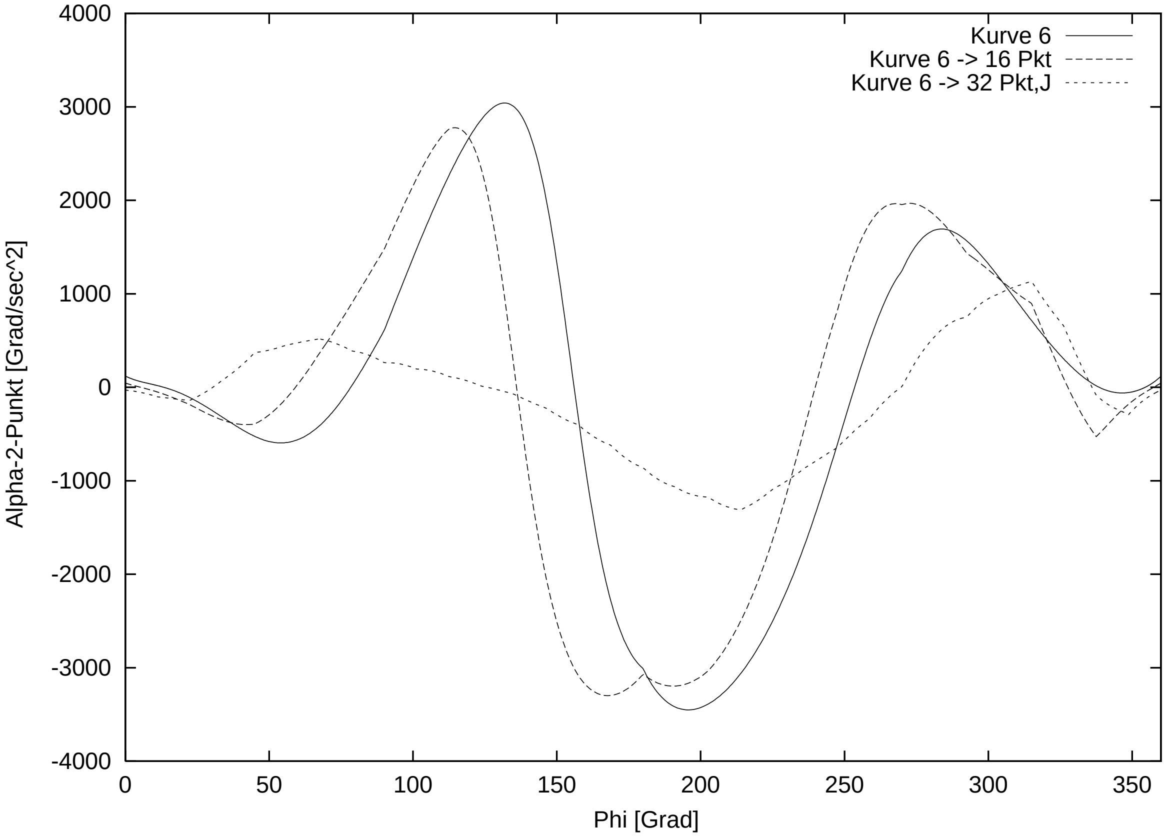


Abbildung 5.18: Winkelbeschleunigung $\ddot{\alpha}(\varphi)$: Die Jerk-optimierte Kurve zeigt bis auf die Phase kurz vor Ende der Bewegung eine geringere Beschleunigung als die anderen Kurven. Im Vergleich zu Bewegung 3 sind ihre Amplituden ungefähr doppelt so groß.

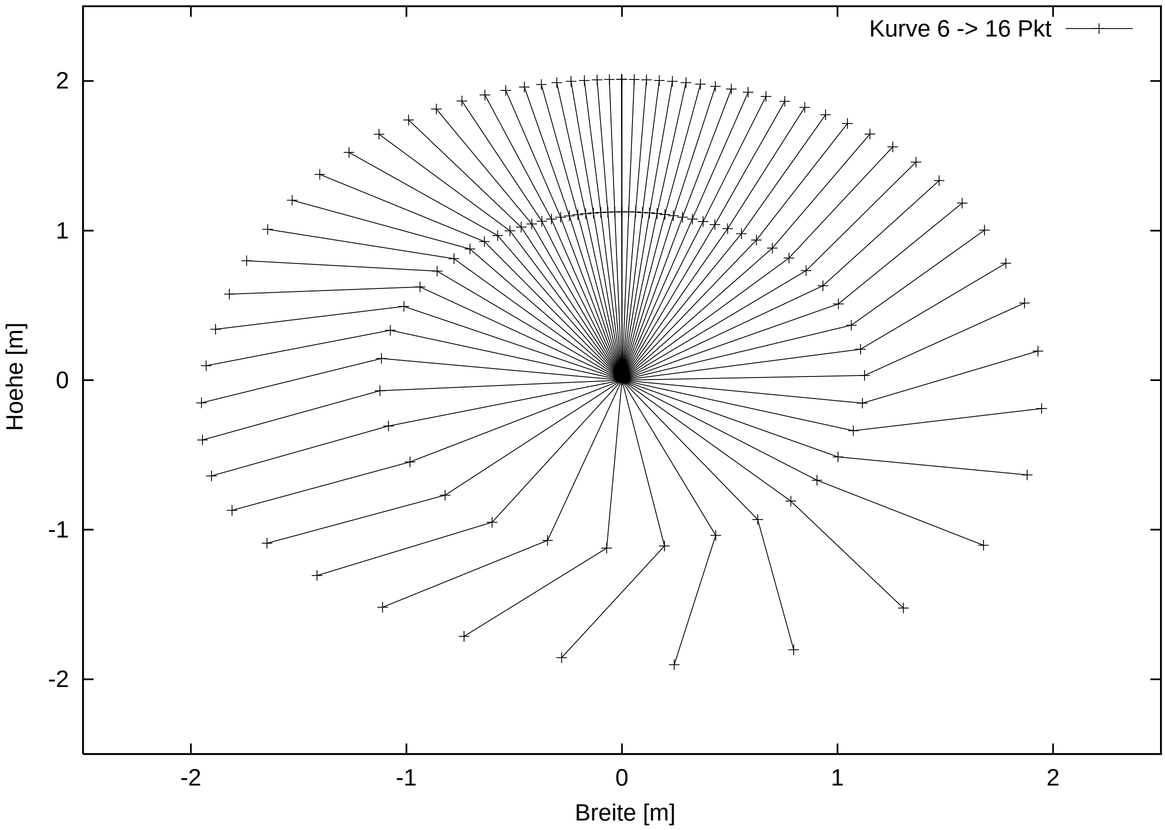


Abbildung 5.19: Stroboskop-Aufnahme von Bewegung 6 - 16 Pkt (25 Hz)

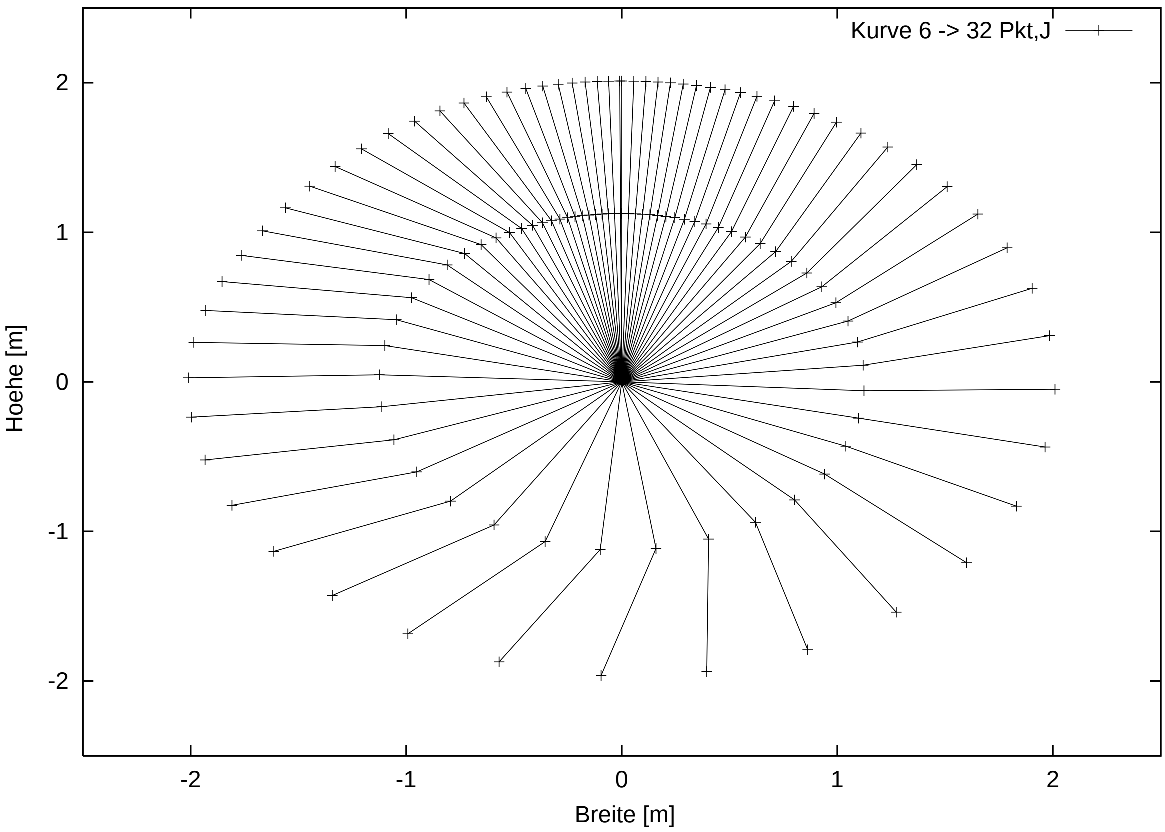


Abbildung 5.20: Stroboskop-Aufnahme von Bewegung 6 - 32 Pkt,J (25 Hz)

5.2 Periodische Folge von Riesenfelgen

Die ursprüngliche Idee ging davon aus, daß ein Hüftmoment $M_\alpha(\varphi)$, das dazu in der Lage ist, nach einer Riesenfelge die Anfangsbedingungen optimal wieder herzustellen, auch sehr viele Felgen hintereinander erzeugen kann. Diese Vermutung hat sich leider nicht bestätigt. Tatsächlich brechen die meisten Bewegungen, die für die erste Felge gute bis hervorragende Bewertungen haben, spätestens nach 2-3 Felgen ab. Paßt man jedoch die Bewertungsfunktion an die neue Aufgabenstellung an, so lassen sich nahe den bekannten Lösungen neue Funktionen finden, die bis zu 6 Riesenfelgen in Folge erzeugen.

Leider hat sich herausgestellt, daß diese Lösungen mehr oder weniger instabil sind, wie das folgende Beispiel zeigt :

Folge von Riesenfelgen

Ausgehend vom 16-Stützstellen-Spline aus Bewegung 6 wurde eine Optimierung auf 10 Runden durchgeführt, deren bestes Ergebnis eine Folge von 6 Riesenfelgen war. (Kurve 6,L) Aufgrund eines Programmierfehlers wurden die intern als 'double' behandelten Splineparameter nach der Optimierung nur mit 10 Stellen in eine Datei geschrieben, um sie später als Startvektor für neue Durchgänge des Integrators bzw. Optimierers verwenden zu können. Eine erneute Integration (Replay) mit den gespeicherten Parametern brach bereits nach 5 Riesenfelgen vorzeitig ab, da die anfangs nicht zu bemerkenden Unterschiede in den Bewegungen sich im Laufe der Integration immer weiter aufschaukelten und schließlich zu komplett unterschiedlichen Bewegungsverläufen führten. Dieses Verhalten ist typisch für chaotische bzw. instabile Systeme. Um eine langfristig stabile, periodische Bewegung zu gewinnen, wird man daher in jedem Fall zusätzliche Kontroll- und Steuerungsmechanismen einbauen müssen.

Ein möglicher Ansatz für einen solchen Regler ist ein kleines Unterprogramm, das in Abhängigkeit von φ die Bewegungsgrößen $\dot\varphi$, α, $\dot\alpha$ ständig mit den Werten einer Sollbewegung vergleicht und auf Grundlage der Abweichungen $\delta\dot\varphi$, $\delta\alpha$ und $\delta\dot\alpha$ ein Korrekturmoment $\delta M_\alpha(\varphi)$ berechnet, das korrigierend in die Bewegung eingreift.

$$\delta M_\alpha(\varphi) = f(\varphi, \delta\dot\varphi, \delta\alpha, \delta\dot\alpha) \tag{5.3}$$

$$M_\alpha(\varphi) = M_{\alpha 0}(\varphi) + \delta M_\alpha(\varphi, \delta\dot\varphi, \delta\alpha, \delta\dot\alpha) \tag{5.4}$$

Theoretisch kann man δM_α für kleine Abweichungen $\delta\dot\varphi$, $\delta\alpha$ und $\delta\dot\alpha$ entwickeln und versuchen, die ersten Koeffizienten der *Taylor*-Reihe zu bestimmen. Da δM_α als

φ-abhängiges Skalarfeld im $\mathbb{R}^3$ betrachtet werden kann, werden die Entwicklungskoeffizienten jedoch wiederum Funktionen von φ sein. Aus Zeitgründen wurde der Anzsatz nicht weiter verfolgt.

Eine Optimierung, die ohne Regler mehr als einige wenige Runden bewertet, kann wegen der Instabilität der Lösung nicht empfohlen werden.

Parameter und Ergebnisse

	Kurve 6 (16 Pkt)	Kurve 6,L	Replay
Startvektor	$\vec{z}_6$	$\vec{z}_6(16Pkt)$	
$c_{\dot{\varphi}}$	100	100	
c_α	10	10	
$c_{\dot{\alpha}}$	1	1	
c_{Reib}	0	0	
c_{Jerk}	0	0	
Max. bew.Runden	1	10	
Stützstellen	16,32	16	
Werte nach 1 Runde :			
$\dot{\varphi}/\omega$	1	1.0054	1.0054
α (in Grad)	-8.12262e-15	-0.107659	-0.107659
$\dot{\alpha}$ (in Grad/s)	2.35995e-15	0.00561976	0.00561977
Reibung (in Nm)	60.7276	60.7279	60.7279
Jerk (in $s{-}5$)	70720.8	70477.9	70477.9
Moment (in $(Nm)^2sec$)	1618.09	1620.74	1620.74
Max. Runden	3	6	5

Nach einer Umrundung der Stange haben die Bewegung 6,L und ihre Wiederholung noch auf 7 Stellen genau die gleichen Werte. Trotzdem bricht die Wiederholung bereits nach 5 Runden ab.

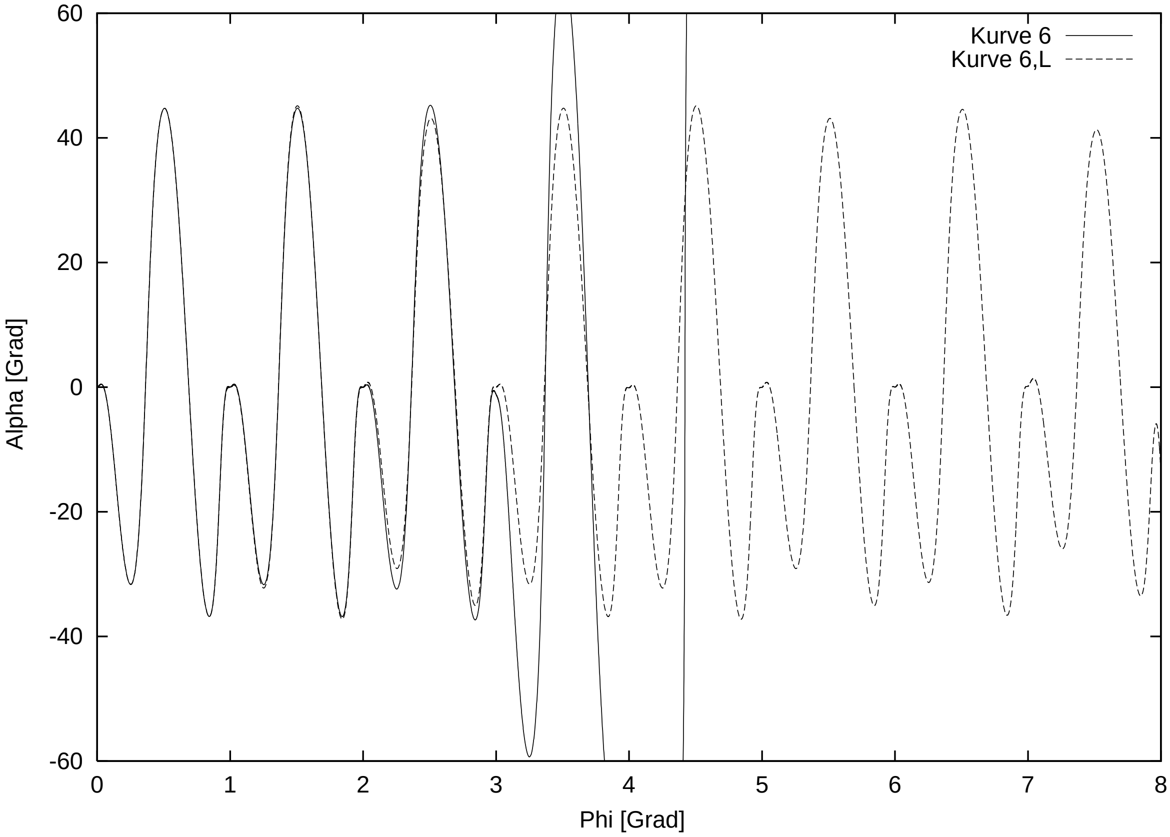


Abbildung 5.21: $\alpha(\varphi)$: Die eigentlich sehr gute Bewegung 6 (16 Pkt) bricht nach wenigen Felgen ab. Bewegung 6,L wurde auf möglichst viele Runden optimiert.

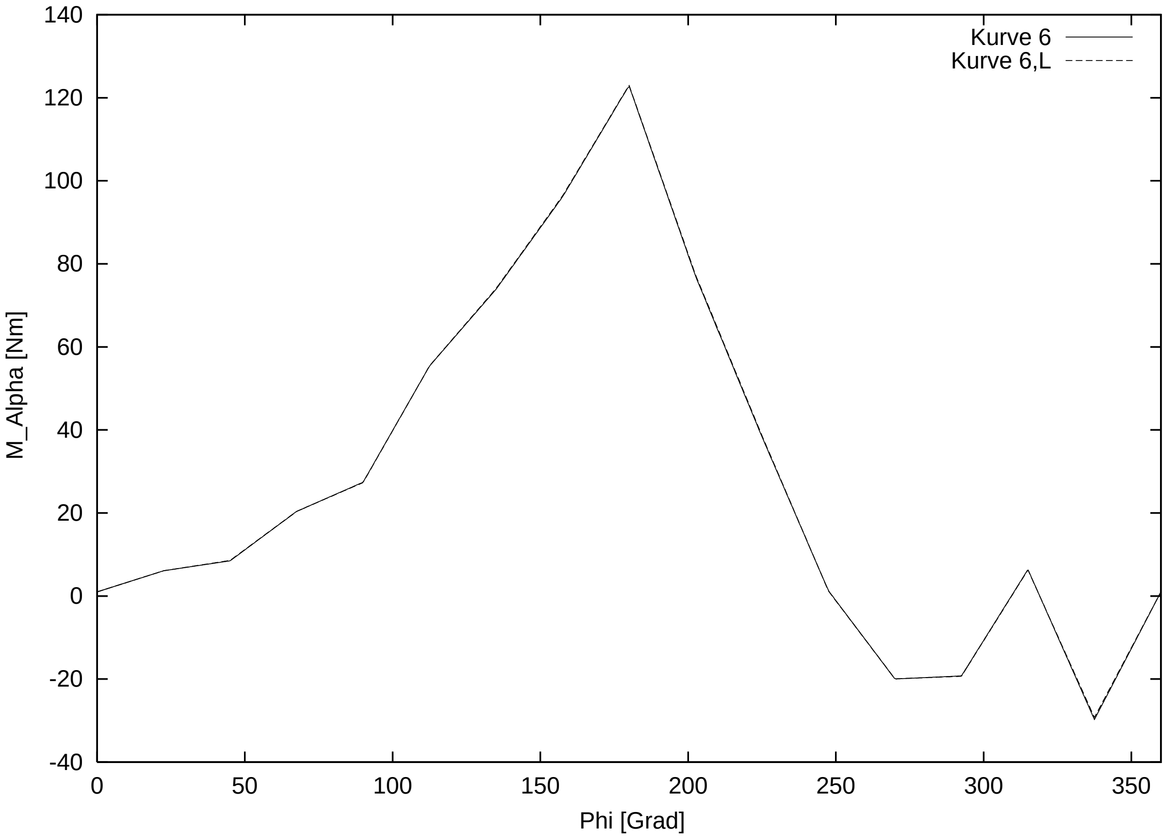


Abbildung 5.22: $M_\alpha(\varphi)$: Die Unterschiede zwischen den Momenten der beiden Bewegungen sind minimal.

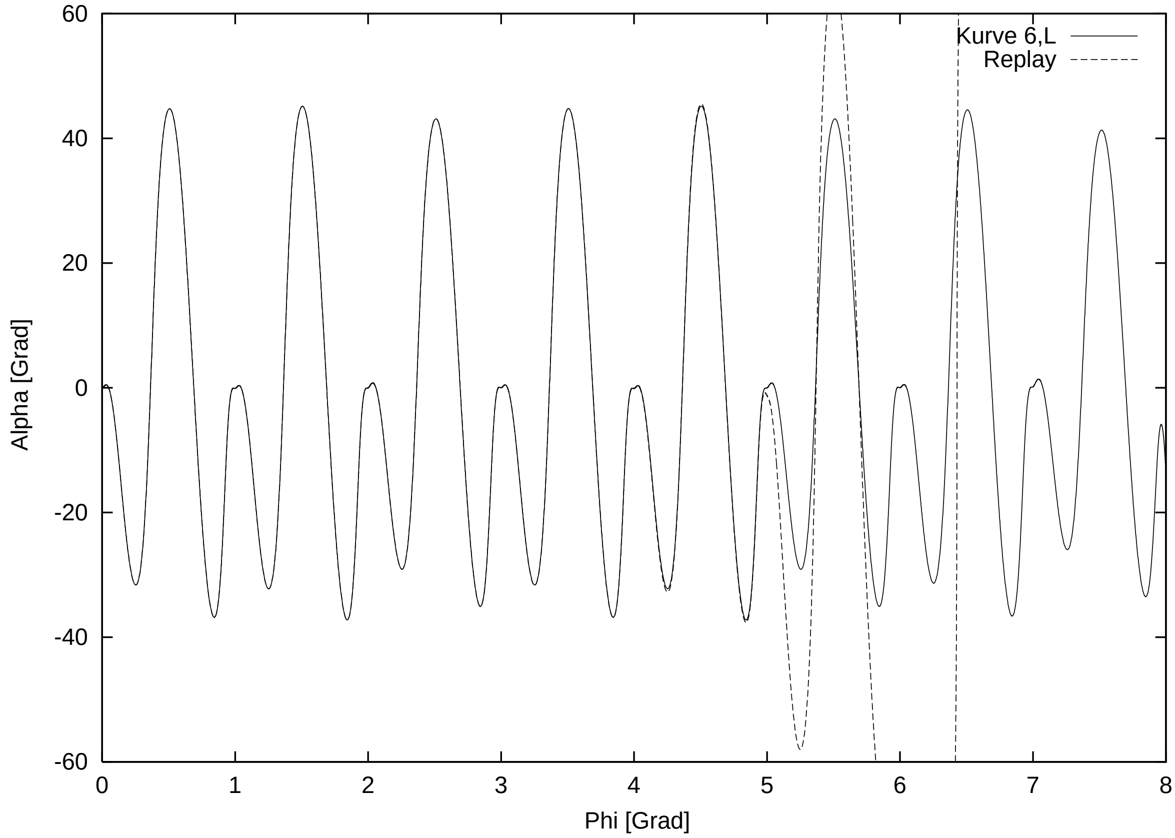


Abbildung 5.23: $\alpha(\varphi)$: Ein Hinweis für die Sensibilität der Bewegung. Die Splineparameter wurden nach der Optimierung mit 10 Nachkommastellen in einer Datei gespeichert. Aufgrund des Informationsverlusts durch die fehlenden Stellen läßt sich die erste Kurve beim Replay nicht mehr exakt reproduzieren. Minimale Abweichungen schaukeln sich auf und führen schließlich zum Abbruch nach nur 5 Runden.

5.3 Beschleunigende Riesenfelge

Der Turner startet mit der niedrigen Winkelgeschwindigkeit von $\dot{\varphi}_0 = 20Grad/sec$ $(= \omega/2)$ und versucht, innerhalb einer Riesenfelge auf die vorgeschriebene Geschwindigkeit von $\omega = 40Grad/sec$ zu beschleunigen.

Optimierung ohne Jerk

$M_\alpha(\varphi)$ startet wie gewohnt mit der Null-Funktion und wird nacheinander durch Splines mit 4, 8 und 16 Stützstellen aproximiert. Der Jerk wird nicht mitbewertet. Man erhält aus der Optimierung eine gezackte Funktion $M_\alpha(\varphi)$, die eine typische Riesenfelge mit den 3 Phasen 'Abbeugen, Überstrecken, Beugen' erzeugt. Aufgrund des zackigen Momentverlaufs fällt auch $\ddot{\alpha}(\varphi)$ sehr unruhig aus.

Optimierung mit Jerk

Die Optimierung berücksichtigt jetzt den Jerk und findet ausgehend von der obigen Funktion mit 16 Stützstellen einen glatteren Momentverlauf.

	ohne Jerk	mit Jerk
Startvektor	0	$\vec{z}(16Pkt)$
$c_{\dot{\varphi}}$	100	100
c_α	10	10
$c_{\dot{\alpha}}$	1	1
c_{Reib}	0	0
c_{Jerk}	0.0	0.01
Stützstellen	4,8,16	16,32
Werte nach 1 Runde :		
$\dot{\varphi}/\omega$	1	1.00004
α (in Grad)	-4.728e-15	6.15114e-05
$\dot{\alpha}$ (in Grad/s)	2.892e-14	1.84324e-05
Reibung (in Nm)	58.4351	59.231
Jerk (in $s{-}5$)	45530.1	1382.6
Moment (in $(Nm)^2 sec$)	2399.3	2985.54

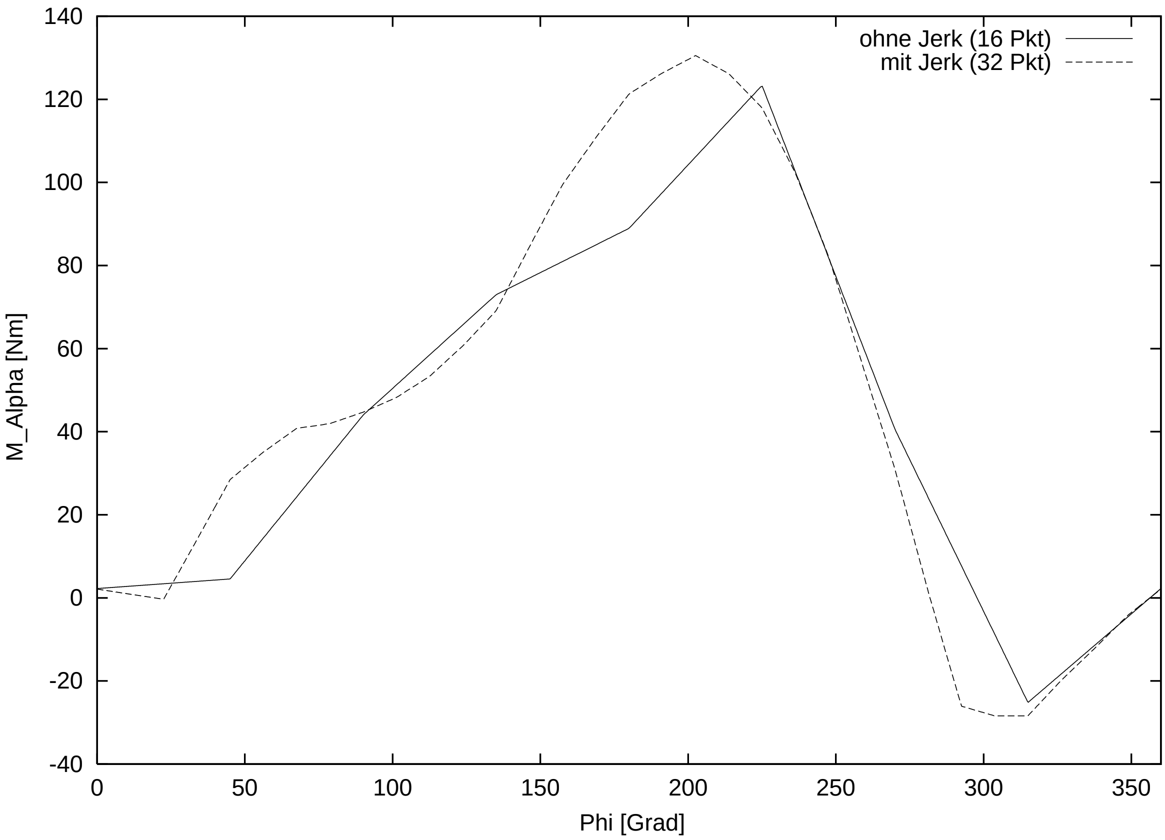


Abbildung 5.24: $M_\alpha(\varphi)$: Dank der Jerk-Optimierung wird der ursprünglich gezackte Momentverlauf ein bißchen harmonischer und glatter. Leider verschwinden nicht alle Kanten.

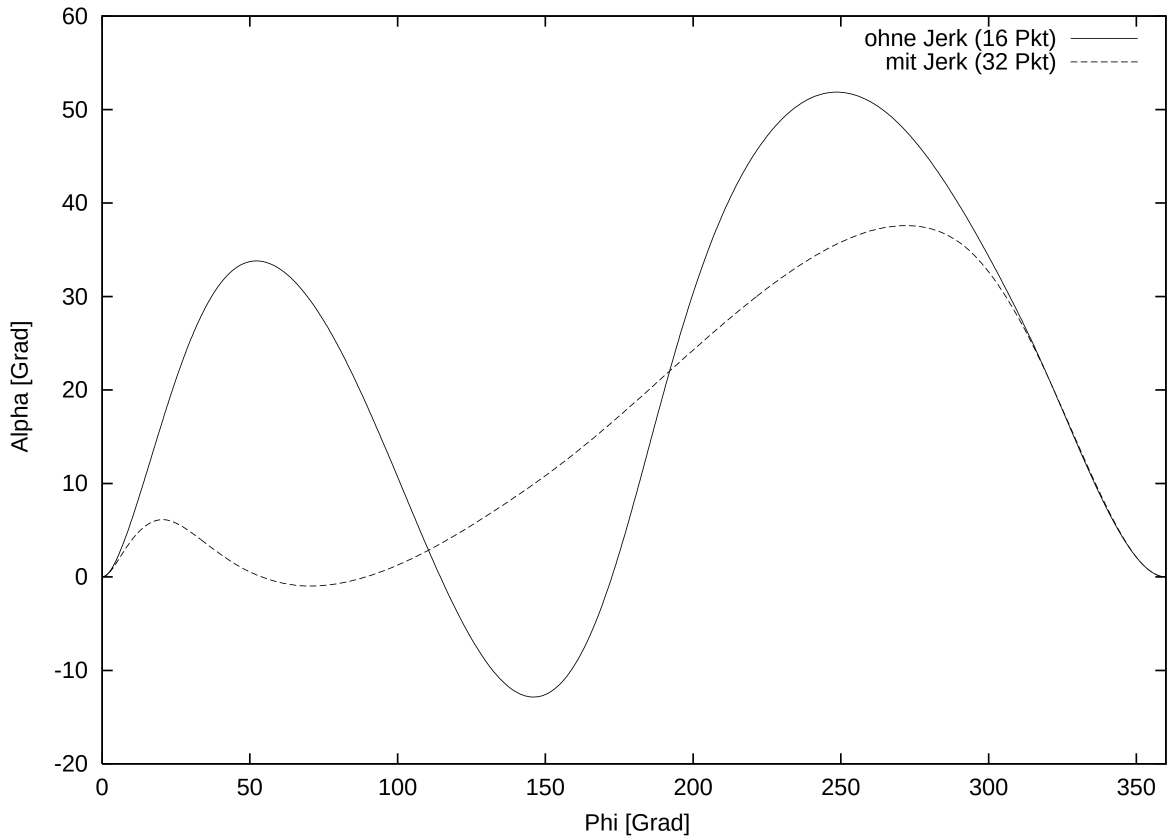


Abbildung 5.25: $\alpha(\varphi)$: Ohne Jerk erhält man die typische Riesenfelgbewegung mit 'Abbeugen, Überstrecken und Beugen'. Mit Jerk kommt es nur zu einem leichten Abbeugen, das in der Aufstiegsphase verstärkt wird.

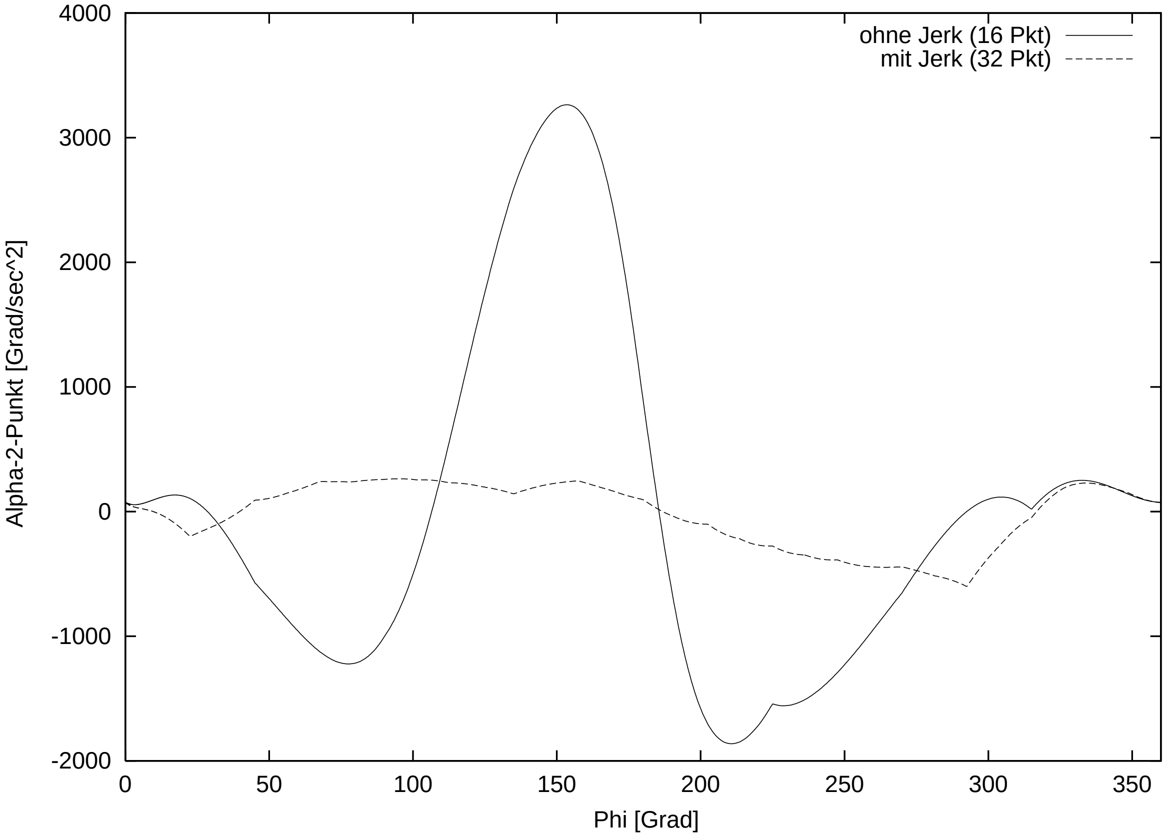


Abbildung 5.26: $\ddot{\alpha}(\varphi)$: Der kantige Momentverlauf der nicht-Jerk-optimierten Bewegung beeinflußt auch $\ddot{\alpha}$ stark. Die zweite Kurve (mit Jerk) verläuft wesentlich flacher, hat aber immer noch ein paar unschöne Knicke.

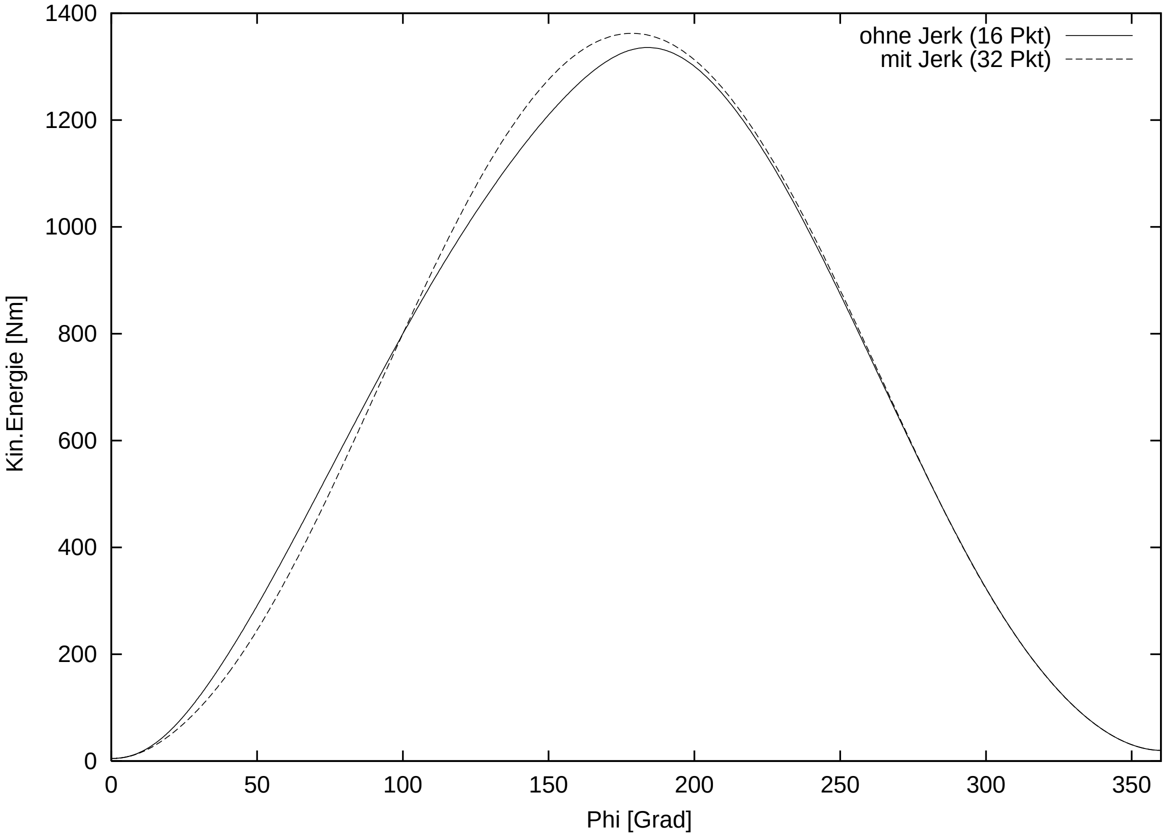


Abbildung 5.27: Kinetische Energie

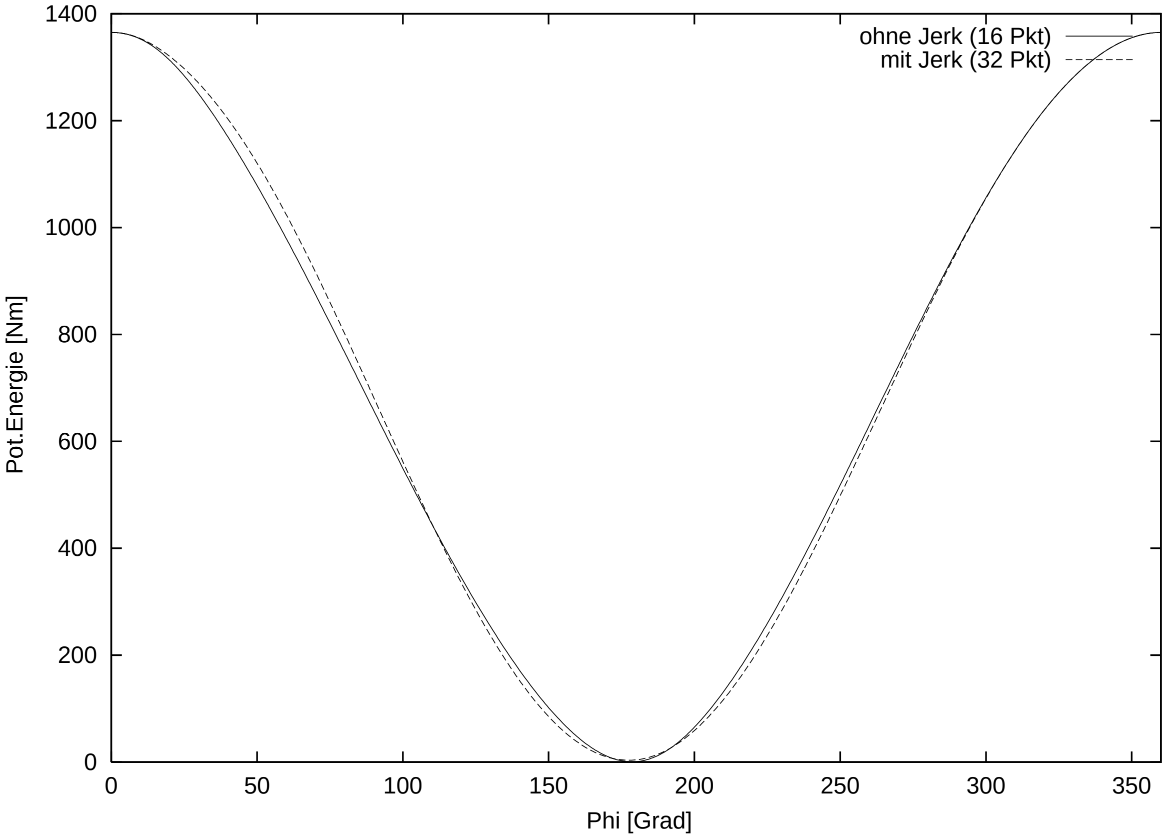


Abbildung 5.28: Potentielle Energie

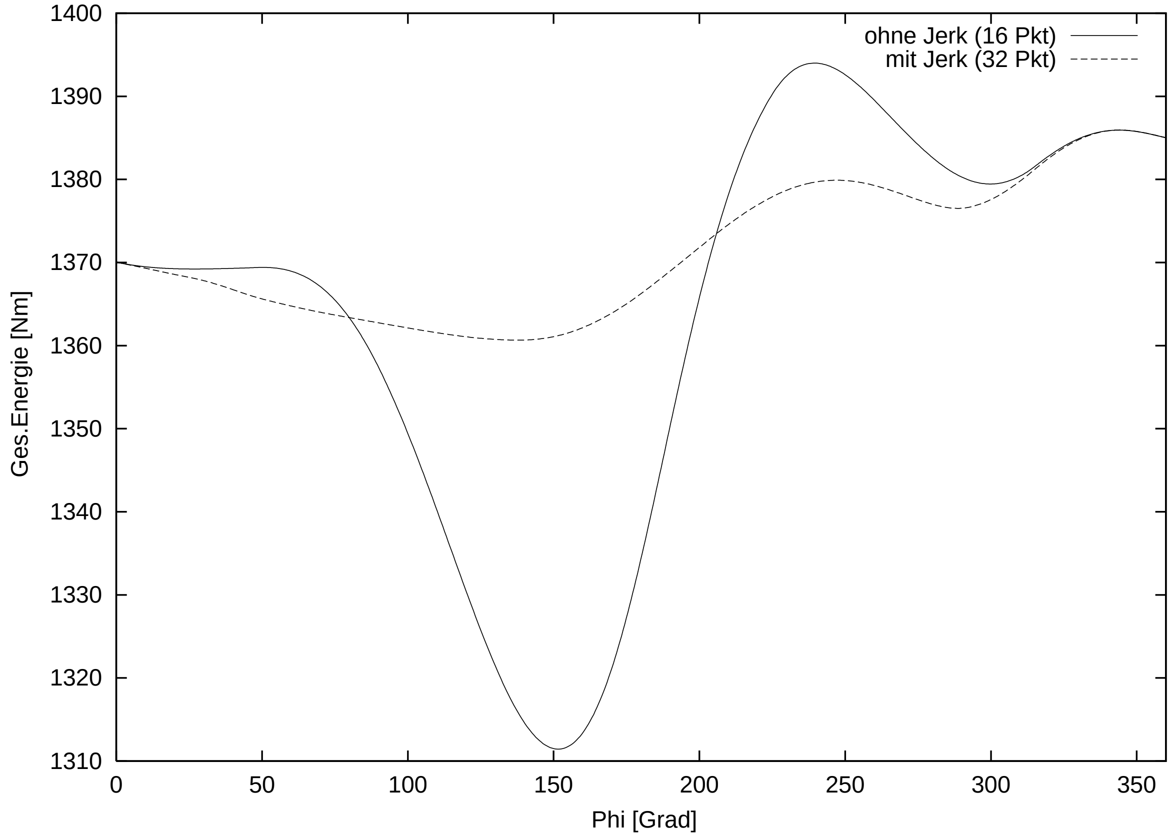


Abbildung 5.29: $E_{ges}(\varphi)$: Man erkennt sehr gut, in welchen Phasen der Bewegung dem System Energie hinzugefügt wird. Der Energie-Abfall in der letzten Phase der Bewegung kommt durch die Reibung zustande.

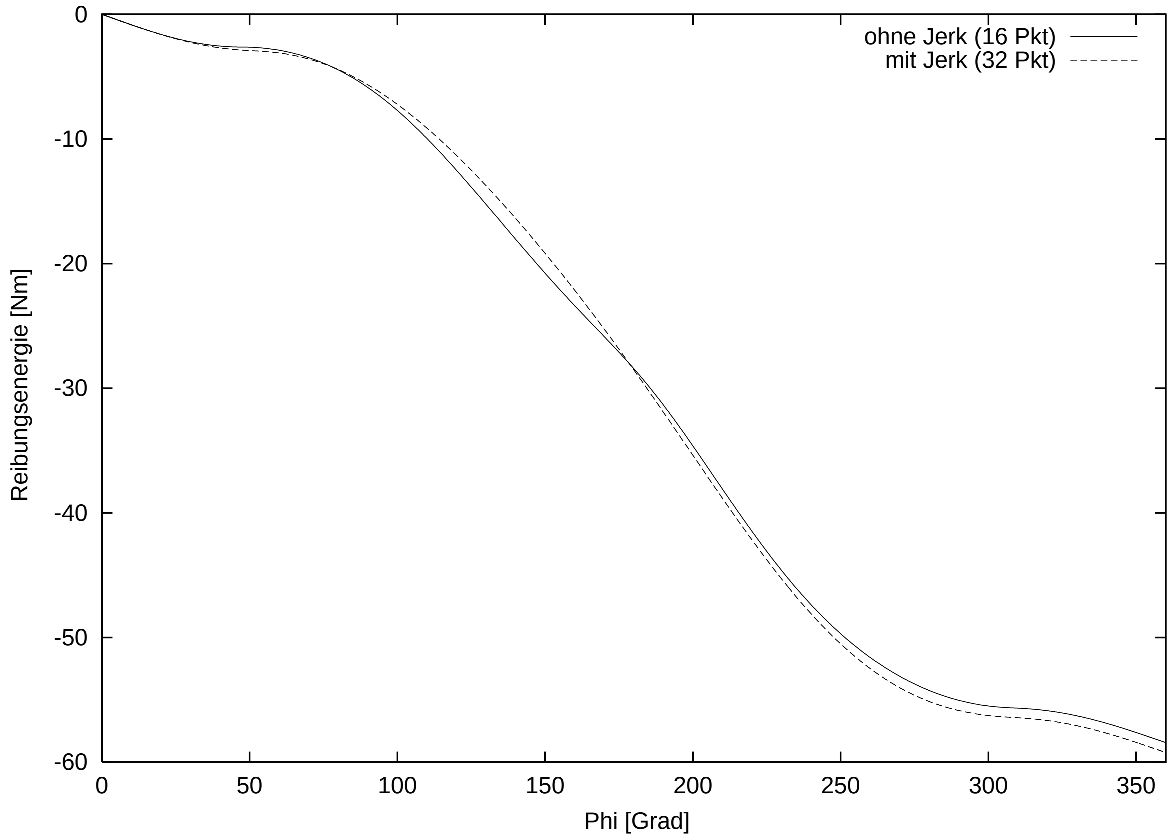


Abbildung 5.30: Reibungsverlust

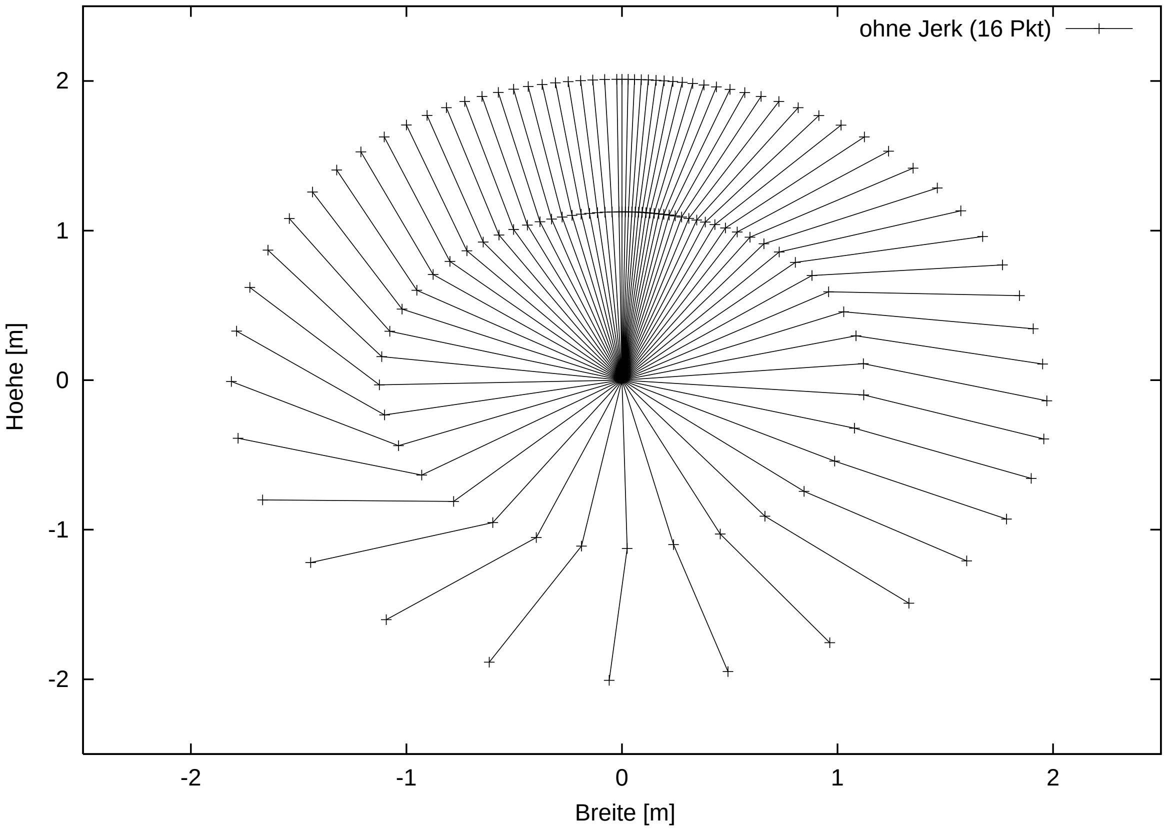


Abbildung 5.31: Stroboskop-Aufnahme der Bewegung mit großem Jerk (25 Hz)

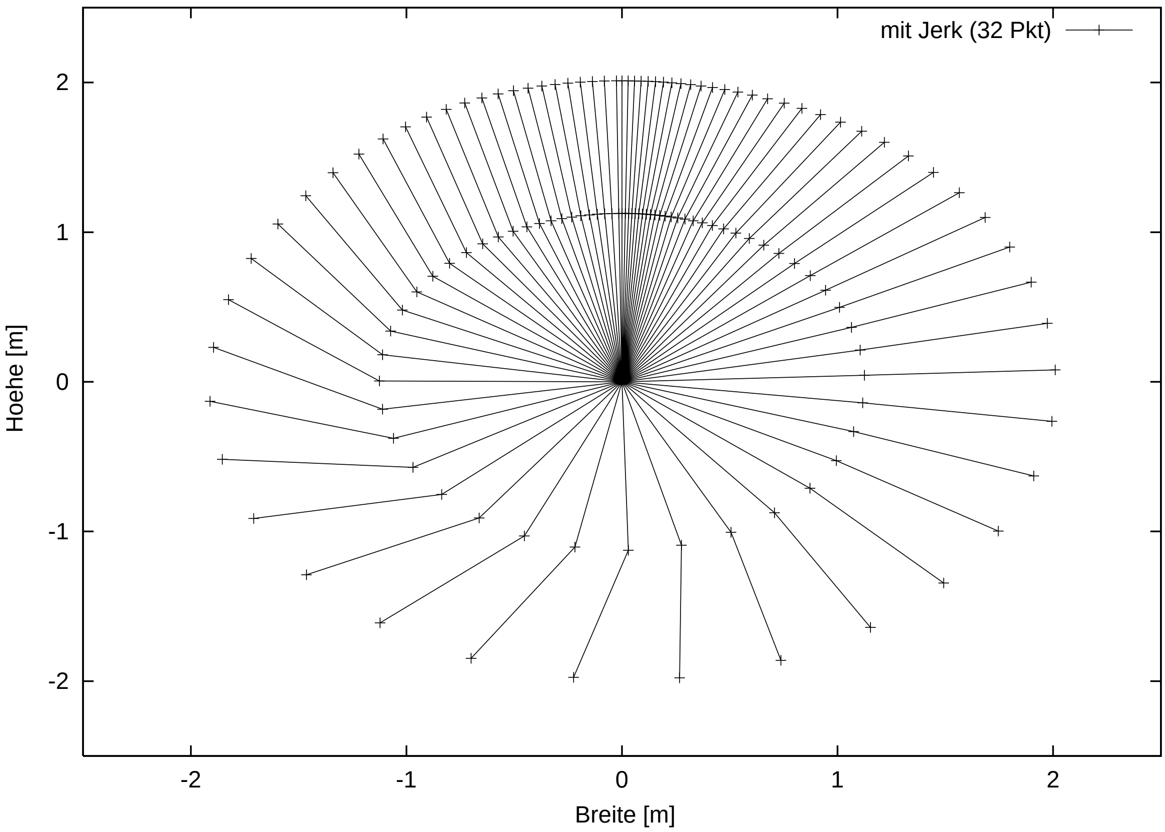


Abbildung 5.32: Stroboskop-Aufnahme der Bewegung mit kleinem Jerk (25 Hz)

5.4 Abbremsende Riesenfelge

Der Turner startet mit einer hohen Winkelgeschwindigkeit von $\dot{\varphi}_0 = 80 Grad/sec$ ($= 2\omega$) und versucht, innerhalb einer Riesenfelge auf die vorgeschriebene Geschwindigkeit von $\omega = 40 Grad/sec$ abzubremsen.

Optimierung ohne Jerk

$M_\alpha(\varphi)$ startet wie gewohnt mit der Null-Funktion und wird nacheinander durch Splines mit 4, 8 und 16 Stützstellen aproximiert. Der Jerk wird nicht mitbewertet. Das Ergebnis ist eine zackige Sinus-ähnliche Kurve, unter der $\ddot{\alpha}$ besonders leidet.

Verfeinerung mit Jerk

Die Optimierung berücksichtigt jetzt auch den Jerk und findet ausgehend von der 16-Stützstellen-Lösung von oben einen halbwegs glatten Momentverlauf. Bei 32 Stützstellen ist klar, daß die Lösung einer echten Sinus-Kurve sehr nahe kommt. Die Bewegung $\alpha(\varphi)$ verläuft nahezu gestreckt. $\ddot{\alpha}$ verschwindet dabei so gut wie vollständig.

	ohne Jerk	mit Jerk
Startvektor	0	$\vec{z}(16Pkt)$
$c_{\dot{\varphi}}$	100	100
c_α	10	10
$c_{\dot{\alpha}}$	1	1
c_{Reib}	0.0	0.0
c_{Jerk}	0.0	0.01
Stützstellen	4,8,16	16,32
Werte nach 1 Runde :		
$\dot{\varphi}/\omega$	1	1
α (in Grad)	6.77159e-16	2.84998e-05
$\dot{\alpha}$ (in Grad/s)	7.95844e-17	1.66189e-06
Reibung (in Nm)	62.9376	62.5517
Jerk (in $s{-}5$)	5338.3	66.9785
Moment (in $(Nm)^2 sec$)	725.891	623.887

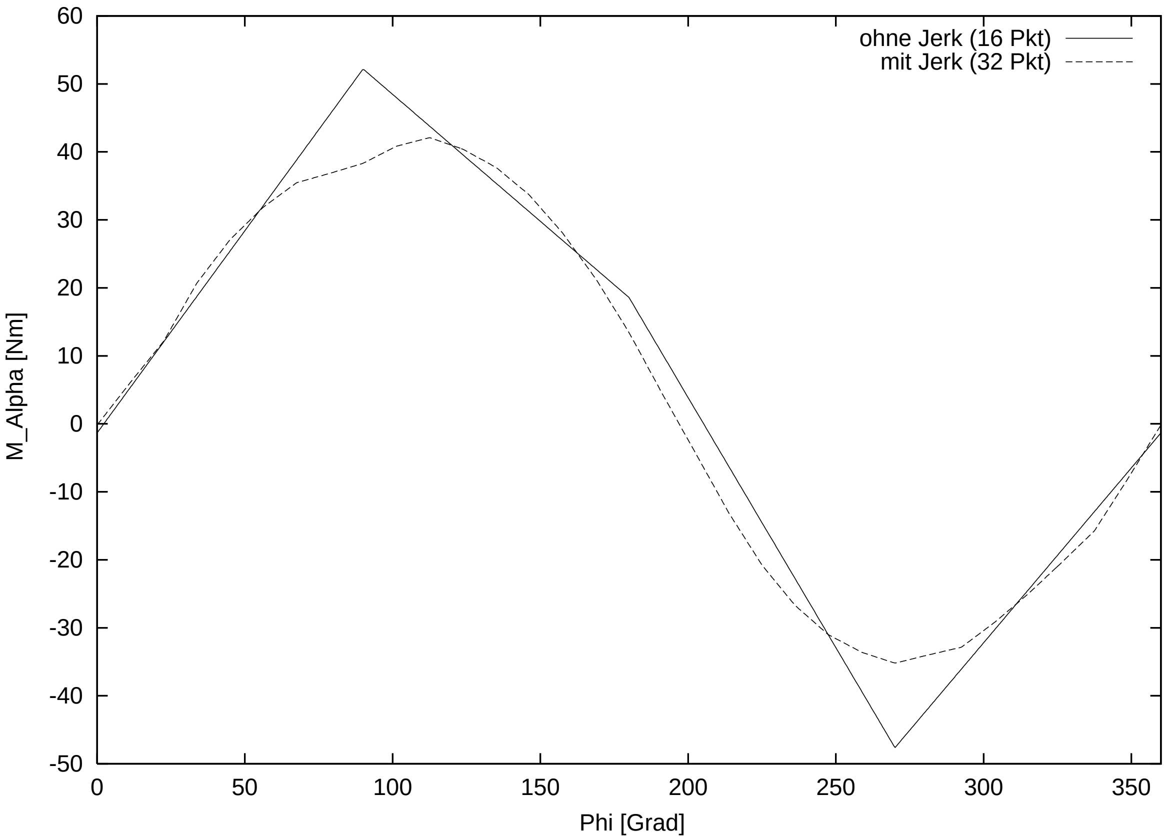


Abbildung 5.33: $M_\alpha(\varphi)$: Die Jerk-Optimierung nähert sich einer reinen Sinus-Funktion.

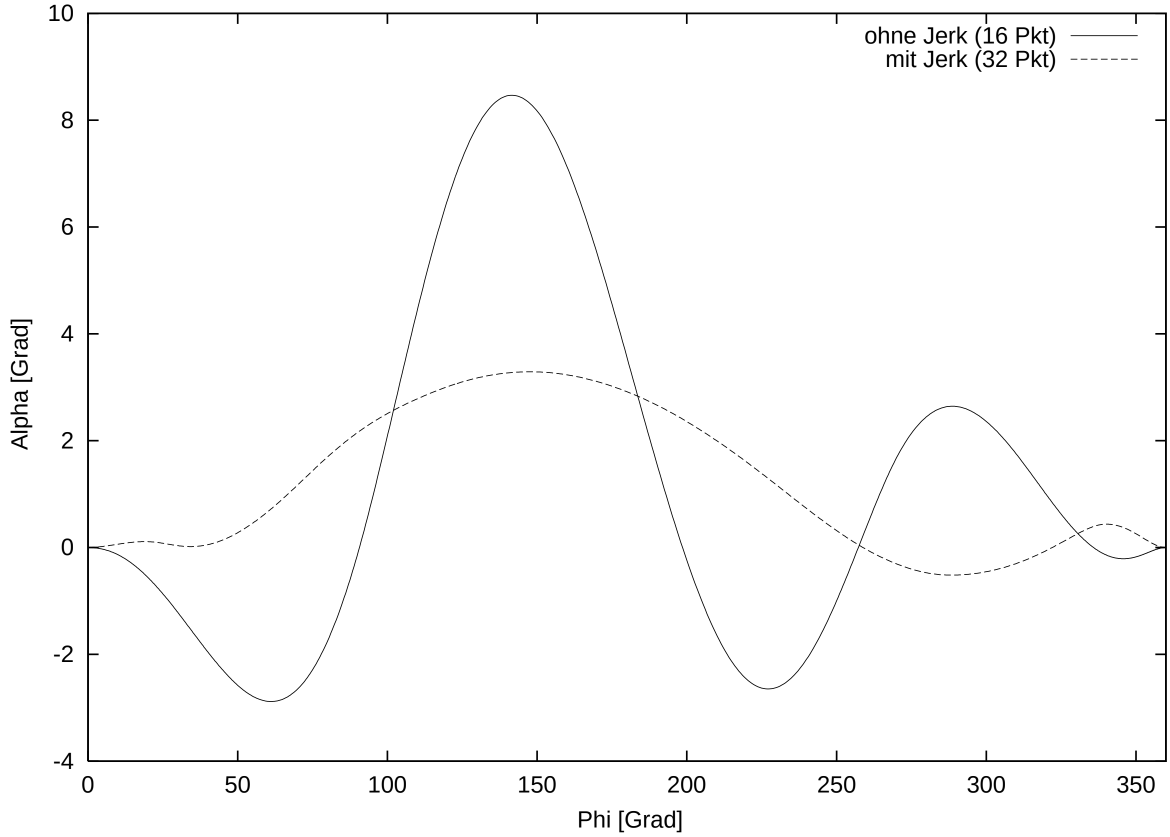


Abbildung 5.34: $\alpha(\varphi)$: Die Bewegung-mit-Jerk verläuft nahezu gestreckt. Der Turner verzichtet darauf, durch eine Anhebung seines Schwerpunkts im Bereich von 180-200 Grad Energie in das System zu pumpen. Stattdessen vernichtet er in diesem Bereich durch das Öffnen des Hüftwinkels zusätzlich Energie.

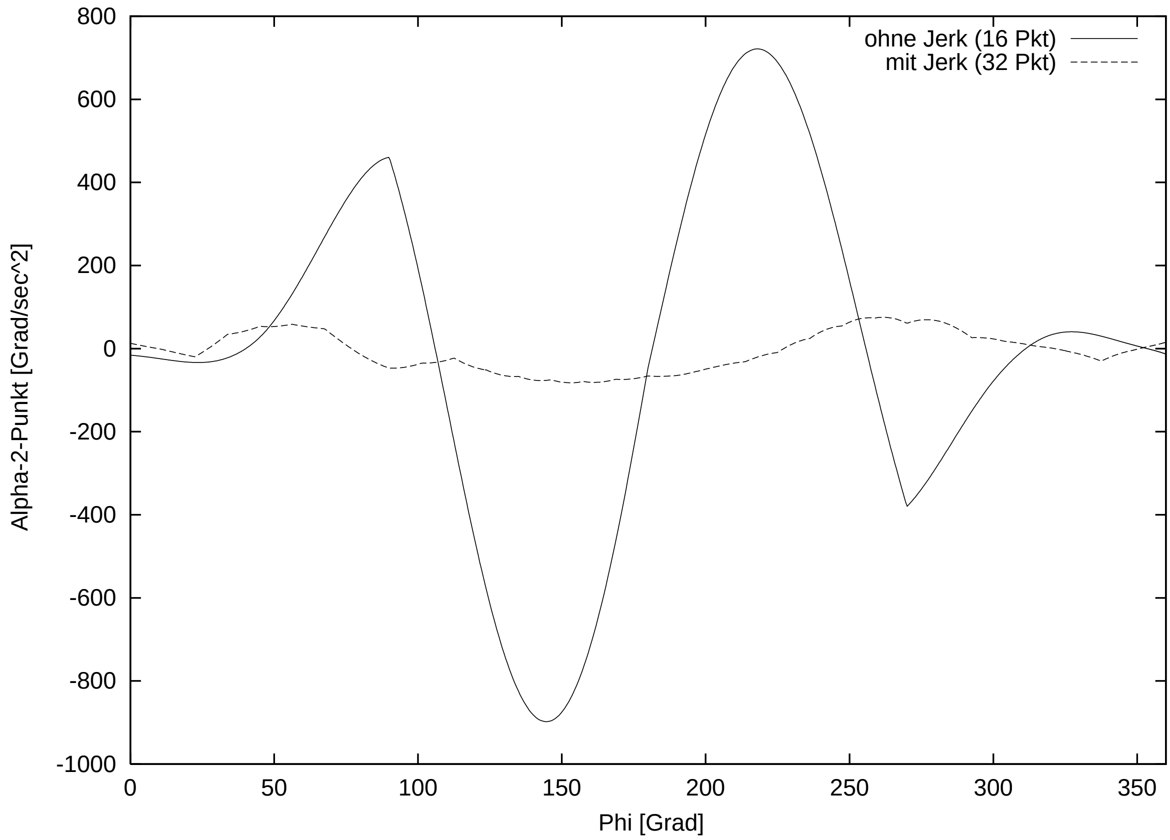


Abbildung 5.35: $\ddot{\alpha}(\varphi)$: Ein großer Unterschied zwischen 'mit Jerk' und 'ohne Jerk'.

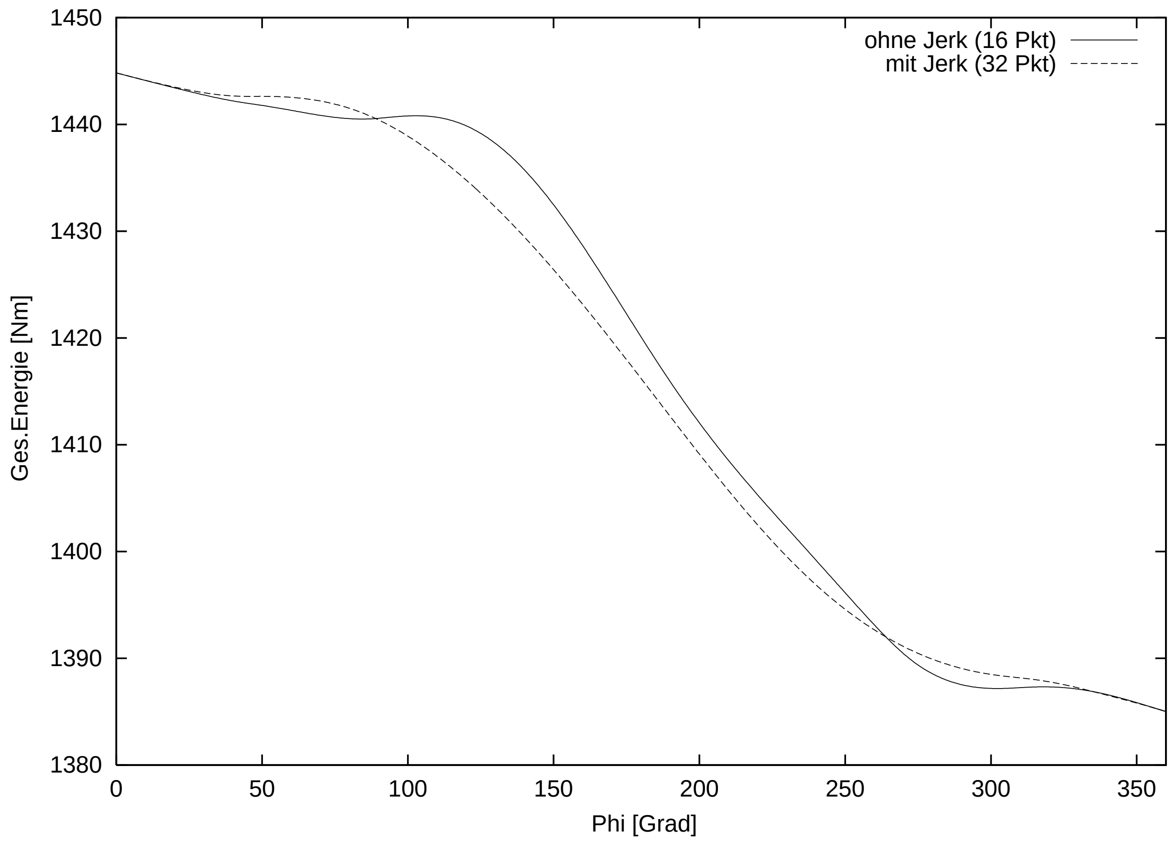


Abbildung 5.36: $E_{ges}(\varphi)$

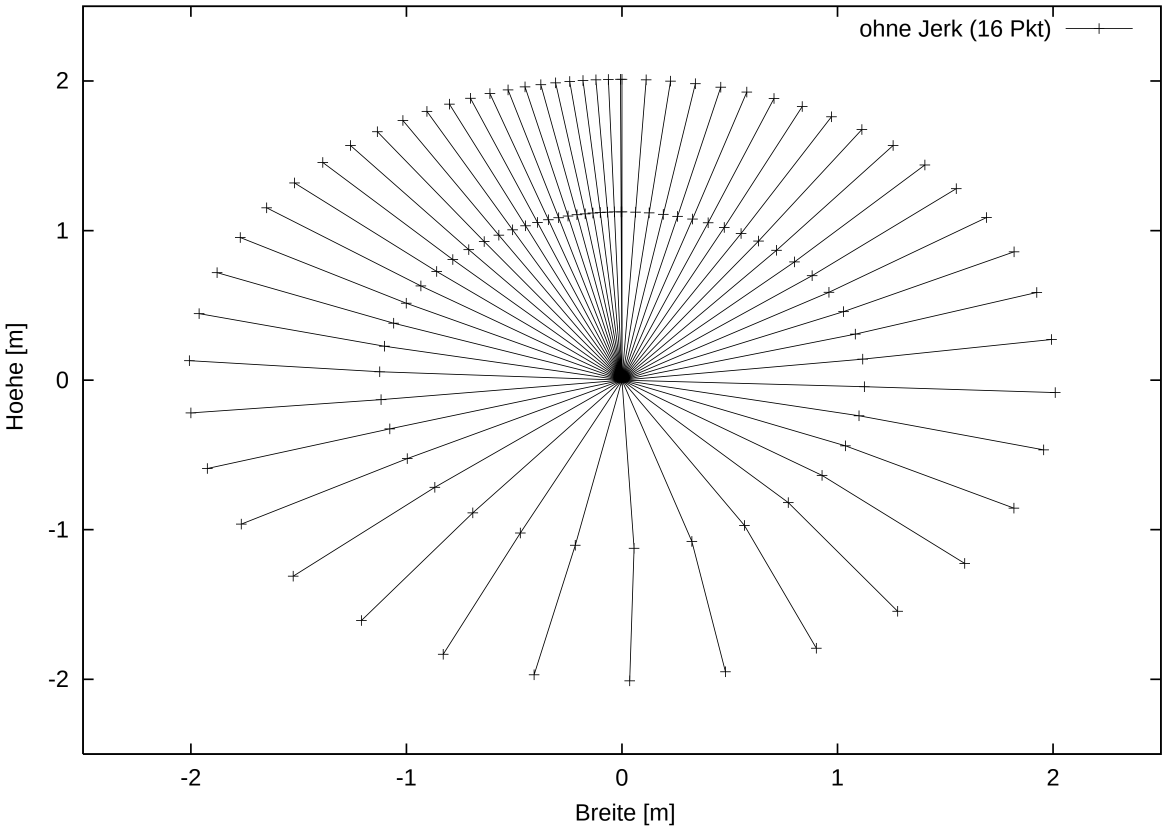


Abbildung 5.37: Stroboskop-Aufnahme - ohne Jerk - 16 Pkt (25 Hz)

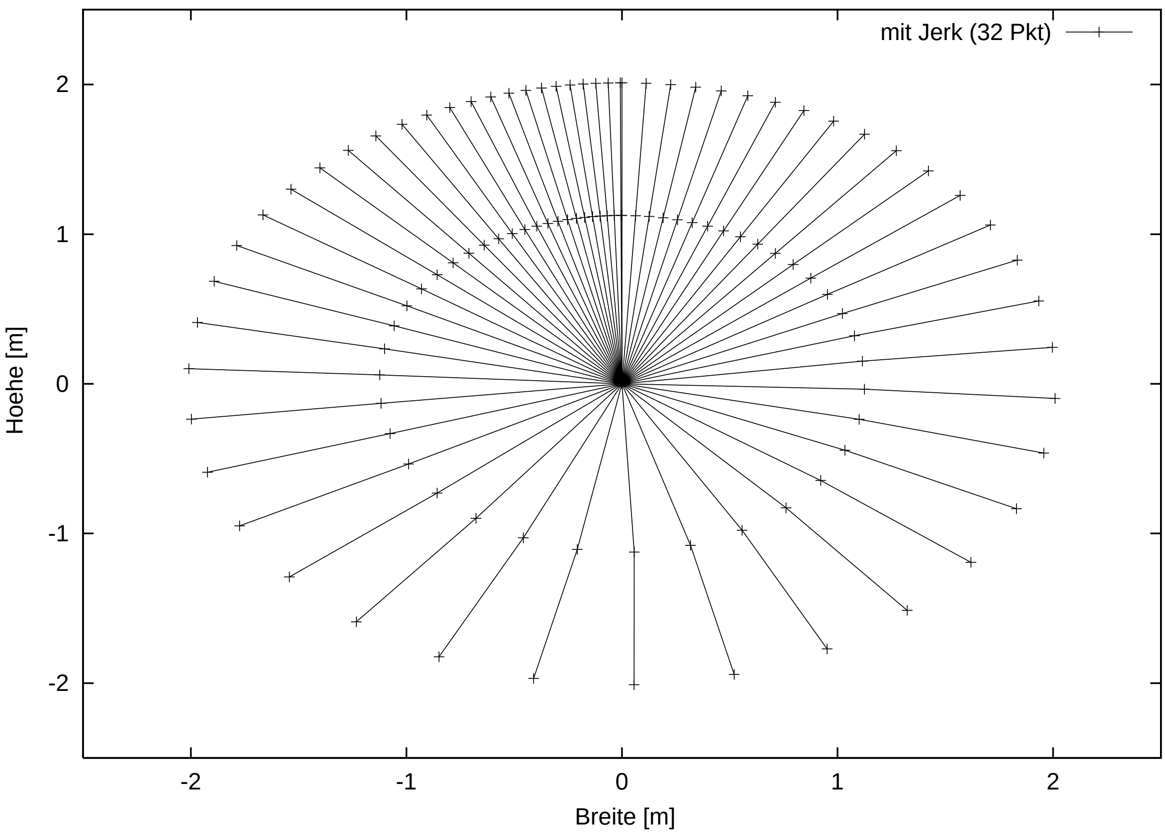


Abbildung 5.38: Stroboskop-Aufnahme - mit Jerk - 32 Pkt (25 Hz)

5.5 Diskussion

Die verwendete Optimier-Methode hat sich nur bedingt als brauchbar erwiesen, um realistische Turnbewegungen in einem einfachen Modell-System (Doppelpendel mit Reibung) zwischen zwei fest vorgegebenen Zuständen zu berechnen. Zwar kann das Turnermodell in allen Fällen die geforderten Randbedingungen mit beliebiger Genauigkeit einhalten, doch entsprechen die gefundenen Lösungen mit ihrem kantigen Aussehen keiner real zu erwartenden Momentvorgabe durch einen Turner. Hat die *Amoebe* einmal ein Minimum gefunden, so ändert sich leider (aber verständlicherweise) die Form der Momentkurve trotz Verdoppelung der Stützstellen nicht mehr. Die erhoffte Ausschmierung der Kurven findet erst statt, wenn zusätzliche Randbedingungen wie z.B. die Jerk-Minimierung gefordert werden. Dadurch verschlechtert sich wiederum die Qualität der Lösung, da die Optimierung der Randbedingungen und die Jerk-Optimierung konkurrierende Prozesse sind. Insgesamt ist der Jerk ein interessantes Hilfsmittel zur Harmonisierung (Glättung) und Stabilisierung der Bewegung.

Die Vorgabe der 2π-periodischen Funktion $M_\alpha(\varphi)$ anstelle von $M_\alpha(t)$ brachte keine Nachteile und hat sich bei der Programmierung als praktisch erwiesen. Die Steuerung einer Folge von mehreren Riesenfelgen durch die 2π-periodische Fortsetzung des Momentverlaufs $M_\alpha(\varphi)$ der ersten Felge funktionierte wegen des instabilen Charakter des Problems aber nur bedingt.

Durch die Vorgabe von $M_\alpha(\varphi)$ durch einen Linearen Spline kommt es leider zwangsläufig zu Artefakten (Spitzen) in der 2. Ableitung von α. Wären die gefundenen Momentverläufe früher bekannt gewesen, hätte sich vielleicht eine Approximation durch *Fourier-Reihen* angeboten. Durch Verdoppelung der Stützstellen und die Minimierung des Jerks läßt sich dieser Effekt etwas abmildern.

Eine Validierung der gefunden Winkelverläufe durch Vermessung der Riesenfelgen menschlicher Turner konnte aus Zeitmangel nicht durchgeführt werden. Die gefundenen Ergebnisse stimmen jedoch in etwa mit den Erfahrungen des Autors aus Beobachtung und eigenem Reckturnen überein und repräsentieren Bewegungslösungen sowohl guter als auch schlechter Turner, wobei man die Jerk-optimierten Kurven meist eher den guten Turnern zurechnen kann.

Es wird nicht bezweifelt, daß von Seiten der Mathematik sicherlich wesentlich effektivere Optimier-Algorithmen zur Lösung dieses Problem existieren als der verwendete Ansatz. Der einfache 'Downhill-Simplex'-Algorithmus wurde nur gewählt, um möglichst schnell an Hand von einfachen Rechnungen ein besseres Gefühl für das Mathematische Problem und das mögliche Aussehen der Lösungen zu bekommen.

5.6 Ausblick auf die weitere Entwicklung von Modell und Simulation

Die nächsten Entwicklungsschritte hängen im wesentlichen davon ab, ob eine einzelne oder mehrere aufeinanderfolgende Riesenfelgen simuliert werden sollen. Im zweiten Fall ist es zwingend notwendig, einen geeigneten Kontroll- und Regelmechanismus in das Modell zu integrieren, der die Bewegung stabilisiert, bevor weitere Freiheitsgrade und Verfeinerungen das Steuerungsproblem weiter komplizieren. Interessiert nur eine einzelne Riesenfelge, so kann man auf eine Regelung verzichten und sich gleich der Verfeinerung des Modells widmen.

Ein wichtiges Element, das leider nicht mehr in das Programm eingebaut werden konnte, sind die elastischen Eigenschaften des Spannrecks. Sie ermöglichen einen Energietransfer zwischen Turner und Reck, der besonders bei hochdynamischen Riesenfelgen (z.B. vor Flugteilen) vom Turner genutzt wird [Ara98]. Einfache Reckmodelle gehen von einer linearen Feder aus, kompliziertere Modelle berücksichtigen die geometrische Anordnung der Abspannung, Materialeigenschaften des Recks sowie nichtlineare Federeigenschaften und Dämpfung. Eine vollständige Simulation eines realistischen Spannrecks beispielsweise mittels der Finite-Elemente-Methode (FEM) und eine anschließende Reduktion auf die für biomechanische Rechnungen relevanten Eigenschaften fehlt aber meines Wissens noch.

Desweiteren ist eine schrittweise Verfeinerung des Turnermodells notwendig. Das Menschen-Modell der Kölner Arbeitsgruppe von Prof. Brüggemann besteht beispielsweise aus 15 Segmenten (Kopf, 2 Oberarme, 2 Unterarme, 2 Hände, Oberkörper, Hüfte, 2 Oberschenkel, 2 Unterschenkel, 2 Füsse) und dient hauptsächlich zur Analyse von Turnbewegungen mittels Inverser Dynamik [Ara98]. Eine optimierte Vorwärtssimulation mit diesem 15-Segmente-Modell dürfte sich jedoch wegen der vielen Parameter und Randbedingungen als äußerst schwierig erweisen. Kleine Schritte wie die anfängliche Unterteilung des Oberkörpers in {Rumpf mit Kopf} und {Arme} wären für das eigene Modell in Anbetracht der unterschiedlichen Aufgabenstellung (Inverse Dynamik - Vorwärtsdynamik) passender.

Da von Seiten der Mathematik und Informatik eine Vielzahl von Optimier-Algorithmen existieren, ist ein Vergleich des verwendeten Verfahrens mit anderen Programmen auf Effektivität und Schnelligkeit sinnvoll, um auch bei komplexeren Modellen gute Lösungen bei wenig Rechenzeit zu erhalten.

Kapitel 6

Nachbetrachtung zum *Minimum-Jerk*-Prinzip

Die ursprüngliche Formulierung des *Minimum-Jerk*-Prinzips lautete :

Für eine Bewegung x(t) mit festem Ziel- und Startpunkt und außerdem feststehender Gesamtzeit T werde diejenige Trajektorie gewählt, für die das Integral

$$\int_0^T \dot{a}^2(t)dt \tag{6.1}$$

minimal ist. Dabei ist $a(t) = \ddot{x}(t)$ die Beschleunigung.

Angenommen, x(t) ließe sich auf dem Intervall [0,T] in eine *Fourierreihe* der Form :

$$x(t) = a_0 + \sum_{n=1}^{\infty} \{a_n \cos(\omega_n t) + b_n \sin(\omega_n t)\} \qquad \text{mit} \qquad \omega_n = 2\pi n/T \tag{6.2}$$

entwickeln, dann kann man auch $\dot{x}$ und alle höheren Ableitungen in der *Fourier*darstellung schreiben:

$$\dot{x}(t) = \sum_{n=1}^{\infty} \omega_n \{-a_n \sin(\omega_n t) + b_n \cos(\omega_n t)\} \tag{6.3}$$

$$\ddot{x}(t) = -\sum_{n=1}^{\infty} \omega_n^2 \{a_n \cos(\omega_n t) + b_n \sin(\omega_n t)\} \tag{6.4}$$

$$\dot{a}(t) = \sum_{n=1}^{\infty} \omega_n^3 \{a_n \sin(\omega_n t) - b_n \cos(\omega_n t)\}. \tag{6.5}$$

Für $\dot{a}^2(t)$ folgt:

$$\begin{aligned}
\dot{a}^2(t) &= \sum_{n=1}^{\infty}\sum_{m=1}^{\infty} \omega_n^3\omega_m^3 \{a_n \sin(\omega_n t) - b_n \cos(\omega_n t)\}\{a_m \sin(\omega_m t) - b_m \cos(\omega_m t)\} \\
&= \sum_{n=1}^{\infty}\sum_{m=1}^{\infty} \omega_n^3\omega_m^3 \{a_n a_m \sin(\omega_n t)\sin(\omega_m t) - a_n b_m \sin(\omega_n t)\cos(\omega_m t) \\
&\qquad - a_m b_n \sin(\omega_m t)\cos(\omega_n t) + b_n b_m \cos(\omega_n t)\cos(\omega_m t)\}.
\end{aligned}$$

Setzt man diese Darstellung für $\dot{a}^2(t)$ in das obige Integral ein, so erhält man

$$\begin{aligned}
\int_0^T \dot{a}^2(t)dt &= \sum_{n=1}^{\infty}\sum_{m=1}^{\infty} \omega_n^3\omega_m^3 \{\, a_n a_m \int_0^T \sin(\omega_n t)\sin(\omega_m t)dt \\
&\quad - a_n b_m \int_0^T \sin(\omega_n t)\cos(\omega_m t)dt - a_m b_n \int_0^T \sin(\omega_m t)\cos(\omega_n t)dt \\
&\quad + b_n b_m \int_0^T \cos(\omega_n t)\cos(\omega_m t)dt \,\}.
\end{aligned}$$

Diese Integrale sind bekannt [BS85].

$$\int_0^T \sin(\omega_n t)\sin(\omega_m t)dt = \int_0^T \cos(\omega_n t)\cos(\omega_m t)dt = \delta_{nm}T/2 \tag{6.6}$$

$$\int_0^T \sin(\omega_n t)\cos(\omega_m t)dt = 0. \tag{6.7}$$

Man erhält so die einfache Form

$$\int_0^T \dot{a}^2 \, dt = T/2 \sum_{n=1}^{\infty} \omega_n^6 \, \{a_n^2 + b_n^2\}. \tag{6.8}$$

Das *Minimum-Jerk*-Prinzip lautet nun:

$$\sum_{n=1}^{\infty} n^6 \{a_n^2 + b_n^2\} \overset{!}{=} \mathit{Minimum}. \tag{6.9}$$

Aufgrund der Gewichtung von n^6 für die höheren *Fourier*-Koeffizienten ist sofort einsichtig, daß das *Fourier*spektrum einer *Minimum-Jerk*-Trajektorie möglichst wenige oder gar keine hochfrequenten Anteile enthalten wird.

Anhang A

Ausführliche Herleitung der Bewegungsgleichungen

A.1 Grundlagen

A.1.1 Allgemeine Behandlung eines n-Teilchen-Systems mit Zwangsbedingungen

Ein Punktteilchen der Masse m hat 3 translatorische Freiheitsgrade. Seine Position im Raum wird durch den Ortsvektor $\vec{r}(t) = (x, y, z)$ und seine Geschwindigkeit durch den Vektor $\dot{\vec{r}}(t) = d\vec{r}/dt$ beschrieben. Ist die auf das Teilchen wirkende Kraft $\vec{F}(\vec{r}, \dot{\vec{r}}, t)$ bekannt, so läßt sich die Bewegung des Teilchens durch zweimalige Integration der *Newton*'schen Differentialgleichung

$$m\ddot{\vec{r}} = \vec{F}(\vec{r}, \dot{\vec{r}}, t) \tag{A.1}$$

berechnen.

Analog besitzen n Teilchen 3n Freiheitsgrade. Ordnet man die 3n Ortskoordinaten des Systems alle in einem (Orts-)Vektor

$$\vec{x} = (x_1, x_2, \dots x_{3n}), \tag{A.2}$$

die Massen in einer (Massen-)Matrix

$$M = \begin{pmatrix} m_1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & m_1 & 0 & \cdots & \cdots & \cdots & \vdots \\ \vdots & 0 & m_1 & 0 & \cdots & \cdots & \vdots \\ \vdots & \vdots & 0 & m_2 & 0 & \cdots & \vdots \\ \vdots & \vdots & \vdots & 0 & \ddots & \cdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & m_n \end{pmatrix} \tag{A.3}$$

und die einzelnen Kraftkomponenten in einem (Kraft-)Vektor

$$\vec{F} = (F_1, F_2, \ldots F_{3n}) \tag{A.4}$$

an, so kann man die Bewegungsgleichungen wieder in der bekannten *Newton*'schen Form schreiben :

$$M\ddot{\vec{x}} = \vec{F}(\vec{x}, \dot{\vec{x}}, t) \tag{A.5}$$

Häufig existieren jedoch Zwangsbedingungen wie zum Beispiel feste Abstände zwischen den einzelnen Teilchen, die die Bewegungsfreiheit des Systems einschränken und die Zahl der Freiheitsgrade reduzieren. In diesem Fall lassen sich die Bewegungsgleichungen nicht mehr ohne weiteres integrieren, da zu den 'eingeprägten' Kräften $\vec{F}^{(e)}$ nun noch Zwangskräfte $\vec{F}'$ hinzukommen, die für die Einhaltung der Zwangsbedingungen sorgen und in der Regel nicht bekannt sind.

$$\vec{F} = \vec{F}^{(e)} + \vec{F}' \tag{A.6}$$

Im günstigsten Fall lassen sich die Zwangsbedingungen in Form von $r\,(\leq 3n)$ holonomen Gleichungen der Art

$$f_k(\vec{x}, t) = 0 \qquad (k = 1, 2, \ldots, r) \tag{A.7}$$

angeben. Sind die Funktionen $f_k(\vec{x}, t)$ nicht explizit von der Zeit t abhängig, so nennt man das System holonom-skleronom, andernfalls holonom-rheonom. Jede der r Bedingungen $f_k(\vec{x}, t) = 0$ schränkt die Bewegung des Systems auf eine $(3n-1)$-dimensionale Hyperfläche ein. Die Schnittfläche aller r Hyperflächen repräsentiert den $(3n-r)$-dimensionalen Koordinatenraum, in dem sich das System noch bewegen kann.

Da bei einer Bewegung innerhalb der Schnittfläche keine Zwangsbedingungen verletzt werden, dürfen dort keine Zwangskräfte wirken. Es kann daher angenommen werden, daß die Zwangskräfte senkrecht auf den Zwangsflächen stehen. Da der

Vektor $\vec{\nabla} f_k(\vec{x}, t)$ in jedem Punkt senkrecht auf der Fläche $f_k(\vec{x}, t) = 0$ steht, ist der Ansatz

$$\vec{F}'_k = \lambda_k \vec{\nabla} f_k(\vec{x}, t) \tag{A.8}$$

naheliegend. Die *Lagrange*-Multiplikatoren $\lambda_k \quad (k = 1, ..., r)$ werden i.a. komplizierte Funktionen $\lambda_k(\vec{x}, \dot{\vec{x}}, t)$ sein. Die Zwangskräfte schreiben sich dann

$$\vec{F}' = \sum_{k=1}^{r} \lambda_k \vec{\nabla} f_k(\vec{x}, t). \tag{A.9}$$

Setzt man diesen Ausdruck in die *Newton*'sche Bewegungsgleichung ein, so erhält man

$$M\ddot{\vec{x}} = \vec{F}^{(e)} + \sum_{k=1}^{r} \lambda_k \vec{\nabla} f_k(\vec{x}, t) \tag{A.10}$$

bzw umgestellt und komponentenweise

$$m_i \ddot{x}_i - F_i^{(e)} - \sum_{k=1}^{r} \lambda_k \frac{\partial f_k}{\partial x_i} = 0 \qquad (i = 1, 2, ..., 3n). \tag{A.11}$$

Um die noch unbekannten *Lagrange*-Multiplikatoren zu bestimmen, leitet man die r Zwangsbedingungen zweimal vollständig nach der Zeit ab :

$$f_k(\vec{x}, t) = 0 \tag{A.12}$$

$$\frac{df_k}{dt} = \frac{\partial f_k}{\partial t} + \sum_{i=1}^{3n} \frac{\partial f_k}{\partial x_i} \dot{x}_i = 0 \tag{A.13}$$

$$\frac{d^2 f_k}{dt^2} = \frac{\partial^2 f_k}{\partial t^2} + 2\sum_{i=1}^{3n} \frac{\partial^2 f_k}{\partial x_i \partial t} \dot{x}_i + \sum_{i,j=1}^{3n} \frac{\partial^2 f_k}{\partial x_i \partial x_j} \dot{x}_i \dot{x}_j + \sum_{i=1}^{3n} \frac{\partial f_k}{\partial x_i} \ddot{x}_i \tag{A.14}$$

Zusammen mit den 3n Bewegungsgleichungen stehen so $3n + r$ Gleichungen zur Bestimmung der gesuchten *Lagrange*-Multiplikatoren $\lambda_1, ..., \lambda_r$ und der 2.Ableitungen der Koordinaten $\ddot{x}_1, ..., \ddot{x}_{3n}$ zur Verfügung.

Bei geometrisch einfachen, holonomen Zwangsbedingungen ist es oft schneller, von den allgemeinen zu einem Satz generalisierter Koordinaten überzugehen, die die Zwangsbedingungen automatisch erfüllen, und über die *Lagrange*-Funktion $L(\vec{q}, \dot{\vec{q}}, t)$ und die *Euler-Lagrange*-Gleichungen 2.Art die Bewegungsgleichungen aufzustellen. Ein kleiner Nachteil dieser Methode liegt darin, daß die Zwangskräfte nicht mehr explizit in die Rechnung eingehen und dadurch, falls benötigt, nachträglich berechnet werden müssen.

$$(x_1, ..., x_{3n}) \rightarrow (q_1, ..., q_{3n-r}) \tag{A.15}$$

$$L(\vec{q}, \dot{\vec{q}}, t) = T(\dot{\vec{q}}) - V(\vec{q}, t) \tag{A.16}$$

$$\frac{\partial L}{\partial q_i} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) = 0 \qquad (i = 1, 2, ..., 3n - r). \tag{A.17}$$

T und V bezeichnen hierbei die kinetische bzw potentielle Energie des Systems als Funktionen der generalisierten Koordinaten $\vec{q}$, ihrer Zeitableitungen $\dot{\vec{q}}$ und der Zeit t selbst.

A.1.2 Bewegungsgleichungen eines Starrkörpers in 2 Dimensionen

Ein Starrkörper in 2 Dimensionen hat 3 Freiheitsgrade, 2 translatorische (Schwerpunktskoordinaten (x,y)) und einen rotatorischen (Orientierungswinkel ϕ). Die freien Bewegungsgleichungen lauten :

$$m\ddot{x} = F_x \tag{A.18}$$

$$m\ddot{y} = F_y \tag{A.19}$$

$$I_{zz}\ddot{\phi} = M \tag{A.20}$$

wobei

$$I_{zz} = \int_V \rho(\vec{r})(x^2 + y^2)dV \tag{A.21}$$

das Trägheitsmoment des Starrkörpers im Schwerpunktsystem um die z-Achse bezeichnet und

$$M = \vec{e}_z\vec{M} = \vec{e}_z(\vec{r} \times \vec{F}) \tag{A.22}$$

das wirkende Drehmoment darstellt.

A.1.3 Invertierung einer 2×2-Matrix

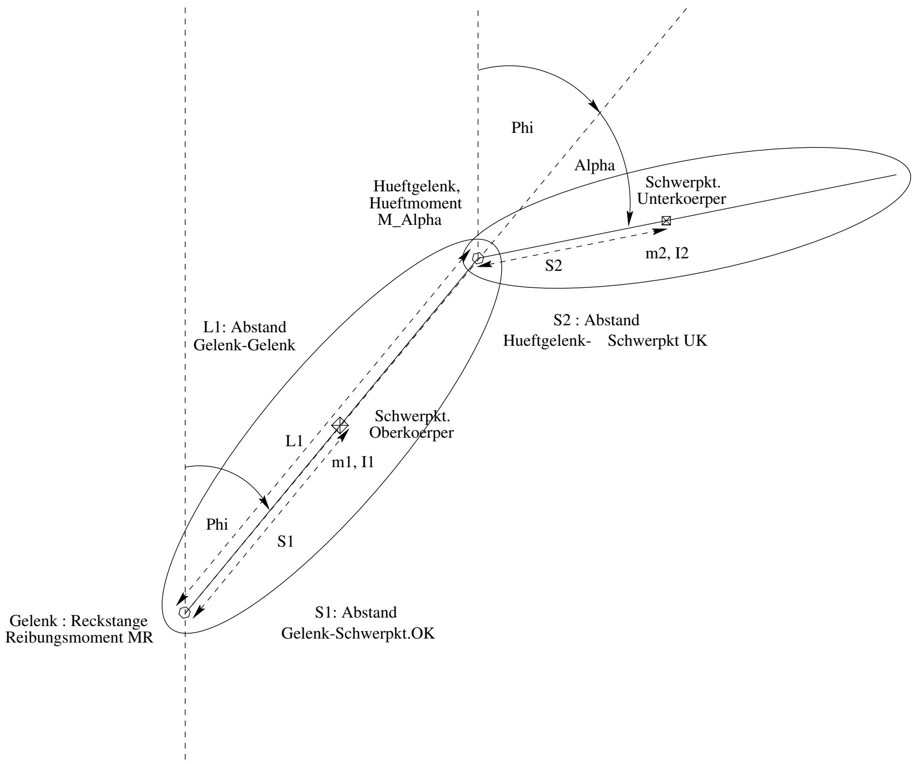
Die Invertierung einer 2×2-Matrix mit nichtverschwindender Determinante wird im nächsten Abschnitt bei der Entkopplung des Differentialgleichungssystems für φ und α benötigt.

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \qquad und \qquad detA = ad - bc \neq 0$$

$$A^{-1} = \frac{1}{detA}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \tag{A.23}$$

A.2 Bewegungsgleichungen für das Turner-Modell

Aus Symmetriegründen (die Bewegung erfolgt in der Sagittal-Ebene) werden die Bewegungsgleichungen nur in 2 Dimensionen aufgestellt. Das Turnermodell besteht aus 2 Starrkörpern (Oberkörper und Unterkörper), die durch ein Schaniergelenk im Hüftbereich (Hüftgelenk) gekoppelt sind. Die Reckstange dient als fester Aufhängepunkt und Drehachse für Körper 1.



m_1, m_2	:	Massen der Körper 1 bzw 2
S_1, S_2	:	Abstände Gelenk - Schwerpunkt
L_1	:	Abstand Reckstange (Gelenk) - Hüftgelenk
I_1, I_2	:	Trägheitsmomente der Körper bezogen auf den Schwerpunkt und die senkrecht zur Papierebene stehende Drehachse
$(x_1, y_1), (x_2, y_2)$	:	Kartes. Schwerpunktskoordinaten
φ	:	Winkel zwischen der Vertikalen und Starrkörper 1 (im Uhrzeigersinn) = Orientierungswinkel Körper 1
α	:	Hüftwinkel
M_R	:	Reibungsmoment (Handreibung an der Reckstange)
M_α	:	Hüftmoment
$\psi = \varphi + \alpha$	:	Orientierungswinkel Körper 2

Die freien Bewegungsgleichungen (mit Drehmomenten) für die Körper 1 und 2 lauten :

$$\begin{pmatrix} m_1 & 0 & 0 \\ 0 & m_1 & 0 \\ 0 & 0 & I_1 \end{pmatrix} \begin{pmatrix} \ddot{x}_1 \\ \ddot{y}_1 \\ \ddot{\varphi} \end{pmatrix} = \begin{pmatrix} 0 \\ -m_1 g \\ M_R - M_\alpha \end{pmatrix} \tag{A.24}$$

$$\begin{pmatrix} m_2 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & I_2 \end{pmatrix} \begin{pmatrix} \ddot{x}_2 \\ \ddot{y}_2 \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} 0 \\ -m_2 g \\ M_\alpha \end{pmatrix} \tag{A.25}$$

Die Zwangsbedingungen und
die Ableitungen der Kartes. Schwerpunktskoordinaten :

$$f_1 = x_1 - S_1 \sin\varphi = 0 \tag{A.26}$$

$$f_2 = y_1 - S_1 \cos\varphi = 0 \tag{A.27}$$

$$f_3 = x_2 - (L_1 \sin\varphi + S_2 \sin\psi) = 0 \tag{A.28}$$

$$f_4 = y_2 - (L_1 \cos\varphi + S_2 \cos\psi) = 0 \tag{A.29}$$

$$x_1 = S_1 \sin\varphi \tag{A.30}$$

$$\dot{x}_1 = S_1 \cos\varphi\, \dot{\varphi} \tag{A.31}$$

$$\ddot{x}_1 = S_1(\cos\varphi\, \ddot{\varphi} - \sin\varphi\, \dot{\varphi}^2) \tag{A.32}$$

$$y_1 = S_1 \cos\varphi \tag{A.33}$$

$$\dot{y}_1 = -S_1 \sin\varphi\, \dot{\varphi} \tag{A.34}$$

$$\ddot{y}_1 = -S_1(\sin\varphi\, \ddot{\varphi} + \cos\varphi\, \dot{\varphi}^2) \tag{A.35}$$

$$x_2 = L_1 \sin\varphi + S_2 \sin\psi \tag{A.36}$$

$$\dot{x}_2 = L_1 \cos\varphi\, \dot{\varphi} + S_2 \cos\psi\, \dot{\psi} \tag{A.37}$$

$$\ddot{x}_2 = L_1(\cos\varphi\, \ddot{\varphi} - \sin\varphi\, \dot{\varphi}^2) + S_2(\cos\psi\, \ddot{\psi} - \sin\psi\, \dot{\psi}^2) \tag{A.38}$$

$$y_2 = L_1 \cos\varphi + S_2 \cos\psi \tag{A.39}$$

$$\dot{y}_2 = -L_1 \sin\varphi\, \dot{\varphi} - S_2 \sin\psi\, \dot{\psi} \tag{A.40}$$

$$\ddot{y}_2 = -L_1(\sin\varphi\, \ddot{\varphi} + \cos\varphi\, \dot{\varphi}^2) - S_2(\sin\psi\, \ddot{\psi} + \cos\psi\, \dot{\psi}^2) \tag{A.41}$$

Die Bewegungsgleichungen mit Zwangskräften :

$$
\begin{aligned}
m_1 \ddot{x}_1 &= \lambda_1 \rightarrow \lambda_1 = m_1 \ddot{x}_1 &\quad& \text{(A.42)}\\
m_1 \ddot{y}_1 &= \lambda_2 - m_1 g \rightarrow \lambda_2 = m_1 \ddot{y}_1 + m_1 g && \text{(A.43)}\\
I_1 \ddot{\varphi} &= M_R - M_\alpha - \lambda_1 S_1 \cos\varphi + \lambda_2 S_1 \sin\varphi - \lambda_3 L_1 \cos\varphi + \lambda_4 L_1 \sin\varphi && \text{(A.44)}
\end{aligned}
$$

$$
\begin{aligned}
m_2 \ddot{x}_2 &= \lambda_3 \rightarrow \lambda_3 = m_2 \ddot{x}_2 &\quad& \text{(A.45)}\\
m_2 \ddot{y}_2 &= \lambda_4 - m_2 g \rightarrow \lambda_4 = m_2 \ddot{y}_2 + m_2 g && \text{(A.46)}\\
I_2 \ddot{\psi} &= M_\alpha - \lambda_3 S_2 \cos\psi + \lambda_4 S_2 \sin\psi && \text{(A.47)}
\end{aligned}
$$

An dieser Stelle ist es geschickt, die Lagrange-Multiplikatoren als Funktionen der Winkel-Koordinaten zu schreiben.

$$
\begin{aligned}
\lambda_1 &= m_1 \ddot{x}_1 = m_1 S_1 (\cos\varphi\, \ddot{\varphi} - \sin\varphi\, \dot{\varphi}^2) &\quad& \text{(A.48)}\\
\lambda_2 &= m_1 \ddot{y}_1 + m_1 g = m_1 g - m_1 S_1 (\sin\varphi\, \ddot{\varphi} + \cos\varphi\, \dot{\varphi}^2) && \text{(A.49)}\\
\lambda_3 &= m_2 \ddot{x}_2 = m_2 L_1 (\cos\varphi\, \ddot{\varphi} - \sin\varphi\, \dot{\varphi}^2) + m_2 S_2 (\cos\psi\, \ddot{\psi} - \sin\psi\, \dot{\psi}^2) && \text{(A.50)}\\
\lambda_4 &= m_2 \ddot{y}_2 + m_2 g \\
&= m_2 g - m_2 L_1 (\sin\varphi\, \ddot{\varphi} + \cos\varphi\, \dot{\varphi}^2) - m_2 S_2 (\sin\psi\, \ddot{\psi} + \cos\psi\, \dot{\psi}^2) && \text{(A.51)}
\end{aligned}
$$

$\ddot{\varphi}$ und $\ddot{\psi}$ werden später ersetzt. Unter Berücksichtigung der Beziehung $\psi = \varphi + \alpha$ ergibt Einsetzen in die Gleichung für $\ddot{\varphi}$:

$$
\begin{aligned}
I_1 \ddot{\varphi} &= M_R - M_\alpha \\
&\quad -\lambda_1 S_1 \cos\varphi + \lambda_2 S_1 \sin\varphi - \lambda_3 L_1 \cos\varphi + \lambda_4 L_1 \sin\varphi \\
&= M_R - M_\alpha \\
&\quad - S_1 \cos\varphi \{ m_1 S_1 (\cos\varphi\, \ddot{\varphi} - \sin\varphi\, \dot{\varphi}^2) \} \\
&\quad + S_1 \sin\varphi \{ m_1 g - m_1 S_1 (\sin\varphi\, \ddot{\varphi} + \cos\varphi\, \dot{\varphi}^2) \} \\
&\quad - L_1 \cos\varphi \{ m_2 L_1 (\cos\varphi\, \ddot{\varphi} - \sin\varphi\, \dot{\varphi}^2) + m_2 S_2 (\cos\psi\, \ddot{\psi} - \sin\psi\, \dot{\psi}^2) \} \\
&\quad + L_1 \sin\varphi \{ m_2 g - m_2 L_1 (\sin\varphi\, \ddot{\varphi} + \cos\varphi\, \dot{\varphi}^2) - m_2 S_2 (\sin\psi\, \ddot{\psi} + \cos\psi\, \dot{\psi}^2) \} \\
&= M_R - M_\alpha \\
&\quad - m_1 S_1^2 (\cos^2\varphi + \sin^2\varphi)\, \ddot{\varphi} + m_1 g S_1 \sin\varphi \\
&\quad - m_2 L_1^2 (\cos^2\varphi + \sin^2\varphi)\, \ddot{\varphi} + m_2 g L_1 \sin\varphi \\
&\quad - m_2 L_1 S_2 \{ (\cos\varphi\cos\psi + \sin\varphi\sin\psi)\, \ddot{\psi} + (\sin\varphi\cos\psi - \cos\varphi\sin\psi)\, \dot{\psi}^2 \} \\
&= M_R - M_\alpha - (m_1 S_1^2 + m_2 L_1^2)\, \ddot{\varphi} \\
&\quad + (m_1 S_1 + m_2 L_1)\, g\, \sin\varphi \\
&\quad - m_2 L_1 S_2 \{ \cos(\psi - \varphi)\, \ddot{\psi} - \sin(\psi - \varphi)\, \dot{\psi}^2 \} \\
&= M_R - M_\alpha - (m_1 S_1^2 + m_2 L_1^2)\, \ddot{\varphi} + (m_1 S_1 + m_2 L_1)\, g\, \sin\varphi \\
&\quad - m_2 L_1 S_2 \{ \cos\alpha\, (\ddot{\varphi} + \ddot{\alpha}) - \sin\alpha\, (\dot{\varphi} + \dot{\alpha})^2 \}
\end{aligned}
$$

Mit den Abkürzungen

$$\Theta_1 = I_1 + m_1 S_1^2 + m_2 L_1^2 \tag{A.52}$$
$$\Theta_2 = I_2 + m_2 S_2^2 \tag{A.53}$$
$$\chi_1 = m_2 L_1 S_2 \tag{A.54}$$
$$\chi_2 = m_1 S_1 + m_2 L_1 \tag{A.55}$$
$$\chi_3 = m_2 S_2 \tag{A.56}$$

folgt schließlich :

$$(\Theta_1 + \chi_1 \cos\alpha)\, \ddot{\varphi} + \chi_1 \cos\alpha\, \ddot{\alpha} = M_R - M_\alpha + \chi_2 g\, \sin\varphi + \chi_1 \sin\alpha\, (\dot{\varphi} + \dot{\alpha})^2 \tag{A.57}$$

Analog gewinnt man aus der Gleichung für $\ddot{\psi}$ ...

$$\begin{aligned}
I_2\ddot{\psi} &= M_\alpha - \lambda_3 S_2 \cos\psi + \lambda_4 S_2 \sin\psi \\
&= M_\alpha - S_2 \cos\psi \{m_2 L_1(\cos\varphi\, \ddot{\varphi} - \sin\varphi\, \dot{\varphi}^2) + m_2 S_2(\cos\psi\, \ddot{\psi} - \sin\psi\, \dot{\psi}^2)\} \\
&\quad + S_2 \sin\psi \{m_2 g - m_2 L_1(\sin\varphi\, \ddot{\varphi} + \cos\varphi\, \dot{\varphi}^2) - m_2 S_2(\sin\psi\, \ddot{\psi} + \cos\psi\, \dot{\psi}^2)\} \\
&= M_\alpha - \chi_1\{(\cos\varphi\cos\psi + \sin\varphi\sin\psi)\, \ddot{\varphi} - (\sin\varphi\cos\psi - \cos\varphi\sin\psi)\, \dot{\varphi}^2\} \\
&\quad - m_2 S_2^2(\cos^2\psi + \sin^2\psi)\, \ddot{\psi} + \chi_3 g \sin\psi \\
&= M_\alpha - \chi_1\{\cos(\psi - \varphi)\, \ddot{\varphi} + \sin(\psi - \varphi)\, \dot{\varphi}^2\} - m_2 S_2^2 \ddot{\psi} + \chi_3 g \sin\psi \\
&= M_\alpha - \chi_1\{\cos\alpha\, \ddot{\varphi} + \sin\alpha\, \dot{\varphi}^2\} - m_2 S_2^2 \ddot{\psi} + \chi_3 g \sin(\varphi + \alpha)
\end{aligned}$$

... die 2. Gleichung :

$$(\Theta_2 + \chi_1 \cos\alpha)\, \ddot{\varphi} + \Theta_2\, \ddot{\alpha} = M_\alpha - \chi_1 \sin\alpha\, \dot{\varphi}^2 + \chi_3 g \sin(\varphi + \alpha) \tag{A.58}$$

Das Gleichungssystem der gekoppelten Differentialgleichungen lautet in Matrix-Vektor-Schreibweise :

$$\begin{pmatrix} \Theta_1 + \chi_1 \cos\alpha & \chi_1 \cos\alpha \\ \Theta_2 + \chi_1 \cos\alpha & \Theta_2 \end{pmatrix} \begin{pmatrix} \ddot{\varphi} \\ \ddot{\alpha} \end{pmatrix} =$$
$$\begin{pmatrix} M_R - M_\alpha + \chi_2 g\, \sin\varphi + \chi_1 \sin\alpha\, (\dot{\varphi} + \dot{\alpha})^2 \\ M_\alpha - \chi_1 \sin\alpha\, \dot{\varphi}^2 + \chi_3 g \sin(\varphi + \alpha) \end{pmatrix} \tag{A.59}$$

Die Determinate der Massen-Matrix ist im Rahmen der physikalisch sinnvollen Parameterwerte immer positiv definit.

$$\begin{aligned}
&(\Theta_1 + \chi_1 \cos\alpha)\, \Theta_2 - (\Theta_2 + \chi_1 \cos\alpha)\, \chi_1 \cos\alpha \\
&= \Theta_1\Theta_2 - (\chi_1 \cos\alpha)^2 \qquad\qquad (A.60) \\
&= (I_1 + m_1 S_1^2 + m_2 L_1^2)(I_2 + m_2 S_2^2) - (m_2 L_1 S_2)^2 \cos^2\alpha \\
&= \ldots + (m_2 L_1 S_2)^2 (1 - \cos^2\alpha) > 0
\end{aligned}$$

Durch Invertieren der Matrix nach Gleichung A.23 läßt sich das Gleichungssystem entkoppeln :

$$\begin{pmatrix} \ddot{\varphi} \\ \ddot{\alpha} \end{pmatrix} = (\Theta_1\Theta_2 - \chi_1^2 \cos^2\alpha)^{-1} \cdot \tag{A.61}$$
$$\begin{pmatrix} \Theta_2 & -\chi_1\cos\alpha \\ -\Theta_2 - \chi_1\cos\alpha & \Theta_1 + \chi_1\cos\alpha \end{pmatrix} \begin{pmatrix} M_R - M_\alpha + \chi_2 g \sin\varphi + \chi_1 \sin\alpha\,(\dot{\varphi}+\dot{\alpha})^2 \\ M_\alpha - \chi_1\sin\alpha\,\dot{\varphi}^2 + \chi_3 g\sin(\varphi+\alpha) \end{pmatrix}$$

Drehimpulsgleichungen

Die gewonnen Gleichungen A.57 und A.58 enthalten die zeitlichen Ableitungen der Drehimpulse. Durch Umstellen und Zusammenfassen der Drehimpulsterme kann man die Gleichungen teilweise integrieren :

$$\begin{gathered}
(\Theta_1 + \chi_1\cos\alpha)\,\ddot{\varphi} + \chi_1\cos\alpha\,\ddot{\alpha} = M_R - M_\alpha + \chi_2 g \sin\varphi + \chi_1\sin\alpha\,(\dot{\varphi}+\dot{\alpha})^2 \\
\Theta_1\,\ddot{\varphi} + \chi_1\cos\alpha\,(\ddot{\varphi}+\ddot{\alpha}) - \chi_1\sin\alpha\,(\dot{\varphi}+\dot{\alpha})\,\dot{\alpha} = \\
M_R - M_\alpha + \chi_2 g \sin\varphi + \chi_1\sin\alpha\,(\dot{\varphi}+\dot{\alpha})\,\dot{\varphi} \\
\frac{d}{dt}\{\Theta_1\,\dot{\varphi} + \chi_1\cos\alpha\,(\dot{\varphi}+\dot{\alpha})\} = M_R - M_\alpha + \chi_2 g \sin\varphi + \chi_1\sin\alpha\,(\dot{\varphi}+\dot{\alpha})\,\dot{\varphi}.
\end{gathered}$$

Analog erhält man aus der zweiten Gleichung :

$$\begin{gathered}
(\Theta_2 + \chi_1\cos\alpha)\,\ddot{\varphi} + \Theta_2\,\ddot{\alpha} = M_\alpha - \chi_1\sin\alpha\,\dot{\varphi}^2 + \chi_3 g\sin(\varphi+\alpha) \\
\Theta_2\,(\ddot{\varphi}+\ddot{\alpha}) + \chi_1\cos\alpha\,\ddot{\varphi} - \chi_1\sin\alpha\,\dot{\alpha}\,\dot{\varphi} = M_\alpha - \chi_1\sin\alpha\,(\dot{\varphi}+\dot{\alpha})\dot{\varphi} + \chi_3 g\sin(\varphi+\alpha) \\
\frac{d}{dt}\{\Theta_2\,(\dot{\varphi}+\dot{\alpha}) + \chi_1\cos\alpha\,\dot{\varphi}\} = M_\alpha - \chi_1\sin\alpha\,(\dot{\varphi}+\dot{\alpha})\dot{\varphi} + \chi_3 g\sin(\varphi+\alpha).
\end{gathered}$$

Addiert man beide Gleichungen, so fallen die Inneren Drehmomente heraus und man erhält die Differentialgleichung für den Gesamt-Drehimpuls L_{ges} :

$$L_{ges} = \Theta_1\,\dot{\varphi} + \chi_1\cos\alpha\,(2\dot{\varphi}+\dot{\alpha}) + \Theta_2\,(\dot{\varphi}+\dot{\alpha}) \tag{A.62}$$
$$\dot{L}_{ges} = M_R + g\,\{\chi_2\sin\varphi + \chi_3\sin(\varphi+\alpha)\}. \tag{A.63}$$

Durch Integration folgt

$$L_{ges}(t) - L_{ges}(t_0) = \int_{t_0}^{t}\{M_R + g\,(\chi_2\sin\varphi + \chi_3\sin(\varphi+\alpha))\}dt. \tag{A.64}$$

Diese Beziehung eignet sich gut als Kontrolle für die Genauigkeit der Integration und die Korrektheit der verwendeten Bewegungsgleichungen.

A.3 Das Reibungsmoment M_R

Zwischen den Händen des Turners und der Reckstange kommt es zu Gleitreibung F_R, die nach dem *Coulomb*schen Reibungsgesetz

$$F_R = \mu\ F_N \tag{A.65}$$

proportional dem Betrag der Normalkraft $\vec{F}_N$ ist, die der Turner auf die Reckstange ausübt. Wegen *actio = reactio* muß diese entgegengesetzt gleich der Zwangskraft sein, mit der die Reckstangenaufhängung den Turner hält. Diese Zwangskraft wurde bei der Ableitung der Bewegungsgleichungen über die *Lagrange*-Multiplikatoren bestimmt und lautet in kartesischen Koordinaten :

$$\vec{F}_{zwang} = \begin{pmatrix} m_1\ddot{x}_1 + m_2\ddot{x}_2 \\ m_1\ (\ddot{y}_1 + g) + m_2\ (\ddot{y}_2 + g) \end{pmatrix} = (m_1 + m_2) \begin{pmatrix} \ddot{x}_{sp} \\ \ddot{y}_{sp} + g \end{pmatrix}. \tag{A.66}$$

Damit folgt :

$$F_N = \|\vec{F}_{zwang}\| = (m_1 + m_2)\ \sqrt{\ddot{x}_{sp}^2 + (\ddot{y}_{sp} + g)^2}. \tag{A.67}$$

Für das Reibungsmoment M_R folgt somit :

$$\vec{M}_R = \vec{r} \times \vec{F}_R \qquad bzw. \qquad M_R = -r\ \mu\ F_N \tag{A.68}$$

und

$$M_R = -r\ \mu\ \sqrt{(m_1\ddot{x}_1 + m_2\ddot{x}_2)^2 + (m_1\ddot{y}_1 + m_2\ddot{y}_2 + (m_1 + m_2)\ g)^2}, \tag{A.69}$$

wobei zu beachten ist, daß die zweiten Ableitungen der kartesischen Koordinaten alle Funktionen der Winkelkoordinaten φ und α und ihrer ersten und zweiten Ableitungen sind. Durch die Quadrierung und anschließende Wurzelziehung werden die einzelnen Terme derart miteinander vermischt, daß ein Einsetzen von $M_R(\ddot{\varphi}, \ddot{\alpha})$ in die Bewegungsgleichungen und anschließendes Auflösen nach $\ddot{\varphi}$ und $\ddot{\alpha}$ nicht möglich ist.

Verwendet man während der Integration ausreichend kleine Schrittweiten, so kann man annehmen, daß sich $\ddot{\varphi}$ und $\ddot{\alpha}$ zwischen zwei Integrationsschritten nur wenig oder gar nicht ändern und die folgende Näherung zulässig ist :

$$M_R(t_n) \approx M_R(\varphi(t_n), \dot{\varphi}(t_n), \ddot{\varphi}(t_{n-1}), \alpha(t_n), \dot{\alpha}(t_n), \ddot{\alpha}(t_{n-1})). \tag{A.70}$$

Für den Hebelarm r und den Reibungskoeffizient μ wurden nach [Bau76] fogende Werte verwendet :

$$r = \quad 0.014\text{m} \qquad \text{(Reckstangenradius)} \tag{A.71}$$

$$\mu = \quad 0.5 \qquad \text{(Stahl auf Leder)} \tag{A.72}$$

Anhang B

Anthropometrische Daten

B.1 Parameter-Berechnung

Für das Turnermodell wurden die Eigenschaften :

Geschlecht	:	männlich
Körpergröße	:	1.74 m
Gewicht	:	72.0 kg

gewählt. Es handelt sich dabei um den Durchschnitt der Werte für 3 mänliche Turner, die in der Arbeit von Arampatzis [Ara95] untersucht wurden.

Mit dem an der Uni Tübingen von Hahn entwickelten Programm *calcman* [Hah93] wurden auf der Grundlage von NASA-Daten [NAS78] mittels Regressionsgleichungen die weiteren anthropometrischen Daten berechnet. Die Werte gelten für einen typischen erwachsenen Westeuropäer.

Aus den anthropometrischen Daten wurden die in das Modell eingehenden Größen $m_1, m_2, S_1, S_2, L_1, L_2, I_1, I_2$ bestimmt. Die Trägheitsmomente wurden nach dem *Satz von Steiner* (Goldstein, S.167)[Gol89] auf die Schwerpunktsysteme von Ober- und Unterkörper umgerechnet.

Da die Tabellenwerte für die Hände auf einer ausgestreckten Haltung basieren (Länge 18.64 cm), der Turner sich aber an der Stange festhält und dabei seine Finger krümmt, wurde die Annahme gemacht, daß die Reckstange sich ungefähr 3cm (statt 13.06 cm) vom Hand-Schwerpunkt entfernt befindet.

$$
\begin{aligned}
m_1 &= 48.8092 \text{ kg} \\
m_2 &= 23.1868 \text{ kg} \\
S_1 &= 0.7242 \text{ m} \\
S_2 &= 0.3507 \text{ m} \\
L_1 &= 1.1254 \text{ m} \\
L_2 &= 0.8862 \text{ m} \\
I_1 &= 3.9258 \text{ Nm} \\
I_2 &= 1.8044 \text{ Nm}
\end{aligned}
$$

B.2 *Maple*-Notizen zur Parameter-Berechnung

Oberkörper

```
> Masse_Torso := 36.7; Masse_Head := 4.706; Masse_Upperarm := 2.001;
> Masse_Forearm := 1.227; Masse_Hand := .4736;

> Torso_Head1 := [0.01456 , 0.0 , 0.30012]; Torso_Head2 := [0.0 , 0.0 ,
> 0.11892]; Torso_Thigh := [.01456, .1136, -.28834];

> Torso_Upperarm1a := [0.01456 , 0.18179 , 0.18444]; Torso_Upperarm1b :=
> [0.01456 , -0.18179 , 0.18444];Torso_Upperarm2 := [0.0 , 0.0 ,
> 0.14394];

> Upperarm_Forearm1 := [0.0, 0.0, 0.15593]; Upperarm_Forearm2 := [0.0,
> 0.0, 0.10946];

> Forearm_Hand1 := [0.0, 0.0, 0.15751]; Forearm_Hand2 := [0.0, 0.0,
> 0.055794];

> Gesamt_Masse := Masse_Torso + Masse_Head + 2 * (Masse_Upperarm +
> Masse_Forearm + Masse_Hand);

                       Gesamt_Masse := 48.8092

> CG_Torso := [0.0, 0.0, 0.0]; CG_Head := Torso_Head1 + Torso_Head2;
> CG_Upperarma := Torso_Upperarm1a + Torso_Upperarm2; CG_Upperarmb :=
> Torso_Upperarm1b + Torso_Upperarm2; CG_Forearma := CG_Upperarma +
> Upperarm_Forearm1 + Upperarm_Forearm2; CG_Forearmb := CG_Upperarmb +
> Upperarm_Forearm1 + Upperarm_Forearm2; CG_Handa := CG_Forearma +
> Forearm_Hand1 + Forearm_Hand2; CG_Handb := CG_Forearmb + Forearm_Hand1
> + Forearm_Hand2;
```

```
>
> Schwerpunkt := 1/Gesamt_Masse * ( Masse_Head * CG_Head +
> Masse_Upperarm * (CG_Upperarma + CG_Upperarmb) + Masse_Forearm *
> (CG_Forearma + CG_Forearmb) + Masse_Hand * (CG_Handa + CG_Handb));
>
>

                      CG_Torso := [0, 0, 0]

                 CG_Head := [.01456, 0, .41904]

           CG_Upperarma := [.01456, .18179, .32838]

          CG_Upperarmb := [.01456, -.18179, .32838]

           CG_Forearma := [.01456, .18179, .59377]

           CG_Forearmb := [.01456, -.18179, .59377]

            CG_Handa := [.01456, .18179, .807074]

            CG_Handb := [.01456, -.18179, .807074]

        Schwerpunkt := [.003612227859, 0, .1128424779]

>
> Ss1 := 0.03 + CG_Handa[3] - Schwerpunkt[3];
>

                        Ss1 := .7242315221

> Ll1 := 0.03 + CG_Handa[3] - Torso_Thigh[3];

                         Ll1 := 1.125414

> Ii_Torso := 1.233558; Ii_Head := 0.01998076; Ii_Upperarm :=
> 0.01421709; Ii_Forearm := 0.007134753; Ii_Hand := 0.0007727524;

> Ii1 := Ii_Torso + Masse_Torso * ((CG_Torso[1] - Schwerpunkt[1])^2 +
> (CG_Torso[3] - Schwerpunkt[3])^2) + Ii_Head + Masse_Head *
> ((CG_Head[1] - Schwerpunkt[1])^2 + (CG_Head[3] - Schwerpunkt[3])^2) +
> 2 * (Ii_Upperarm + Masse_Upperarm * ((CG_Upperarma[1] -
> Schwerpunkt[1])^2 + (CG_Upperarma[3] - Schwerpunkt[3])^2) + Ii_Forearm
```

```
> + Masse_Forearm * ((CG_Forearma[1] - Schwerpunkt[1])^2 +
> (CG_Forearma[3] - Schwerpunkt[3])^2) + Ii_Hand + ((CG_Handa[1] -
> Schwerpunkt[1])^2 + (CG_Handa[3] - Schwerpunkt[3])^2));

                          Ii1 := 3.925803327
```

Unterkörper

```
> Masse_Thigh := 7.485; Masse_Shank := 3.132; Masse_Foot := 0.9764;

> Gesamtmasse := 2 * (Masse_Thigh + Masse_Shank + Masse_Foot);

                        Gesamtmasse := 23.1868

> Torso_Thigh1a := [0.01456, 0.11360, -.28834]; Torso_Thigh1b :=
> [0.01456, -0.11360, -.28834]; Torso_Thigh2 := [0,0,-0.17406];
> Thigh_Shank1 := [0, 0, -0.25047]; Thigh_Shank2 := [0, 0, -0.18142];
> Shank_Foot1 := [0, 0, -0.2309]; Shank_Foot2 := [.4930312534e-1,
> 0,-.4930312534e-1];

> CG_Torso := [0.,0.,0.]; CG_Thigha := Torso_Thigh1a + Torso_Thigh2;
> CG_Thighb := Torso_Thigh1b + Torso_Thigh2; CG_Shanka :=CG_Thigha +
> Thigh_Shank1 + Thigh_Shank2; CG_Shankb :=CG_Thighb + Thigh_Shank1 +
> Thigh_Shank2; CG_Foota := CG_Shanka + Shank_Foot1 + Shank_Foot2;
> CG_Footb := CG_Shankb + Shank_Foot1 + Shank_Foot2;

                        CG_Torso := [0, 0, 0]

                CG_Thigha := [.01456, .11360, -.46240]

               CG_Thighb := [.01456, -.11360, -.46240]

                CG_Shanka := [.01456, .11360, -.89429]

               CG_Shankb := [.01456, -.11360, -.89429]

          CG_Foota := [.06386312534, .11360, -1.174493125]

         CG_Footb := [.06386312534, -.11360, -1.174493125]

> Schwerpunkt := 1/Gesamtmasse * (Masse_Thigh * (CG_Thigha + CG_Thighb)
> + Masse_Shank * (CG_Shanka + CG_Shankb) + Masse_Foot * (CG_Foota +
```

```
> CG_Footb));

            Schwerpunkt := [.01871232560, 0, -.6390494048]

> Ss2 := Torso_Thigh1a[3] -  Schwerpunkt[3];

                          Ss2 := .3507094048

> Ll2 := Torso_Thigh1a[3] - CG_Footb[3];

                          Ll2 := .886153125

> Ii_Thigh := .1339172; Ii_Shank := .04482954; Ii_Foot := .003743683;

> Ii2 := 2 * (Ii_Thigh + Masse_Thigh * ((CG_Thigha[1]-Schwerpunkt[1])^2
> + (CG_Thigha[3]-Schwerpunkt[3])^2) + Ii_Shank + Masse_Shank *
> ((CG_Shanka[1]-Schwerpunkt[1])^2 + (CG_Shanka[3]-Schwerpunkt[3])^2) +
> Ii_Foot + Masse_Foot * ((CG_Foota[1]-Schwerpunkt[1])^2 +
> (CG_Foota[3]-Schwerpunkt[3])^2));

                          Ii2 := 1.804420253
```

B.3 *Calcman*-Daten

```
Anthropometric data of the human body
=======================================

  (Data given for LEFT hand side of the body
   -> symmetry w.r. to sagittal plane assumed)

All values are SI-values, all angles eularian!
length [m], angles [deg], mass [kg], moments of inertia [kgm^2]

Reference points for centers of gravity (CG) as described in
the source code (function seg_center())

Input data: Sex: male,  stature: 1.74 m,  total mass: 72.00 kg

**************************************************************
* Head:
```

```
  Mass: 4.706e+00 kg
  Length: 2.2909e-01 m
  Center of gravity w.r. to reference point:
x:  7.9069e-02   y:  0.0000e+00   z: -1.1017e-01
  Inertia tensor w.r. to center of gravity:
xx:  2.147842e-02   yy:  1.998076e-02   zz:  2.743748e-02
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00
  Occipital condyle/C1 interspace joint center w.r. to CG
x:      0.0000e+00   y:      0.0000e+00   z:     -1.1892e-01
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00

**************************************************************
* Torso (pelvis, lumbus, thorax, neck):
  Mass: 3.670e+01 kg
  Length: 5.8847e-01 m
  Center of gravity w.r. to reference point:
x:  1.0677e-01   y:  0.0000e+00   z: -2.0880e-01
  Inertia tensor w.r. to center of gravity:
xx:  1.770936e+00   yy:  1.233558e+00   zz:  4.298609e-01
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00
  Occipital condyle/C1 interspace joint center w.r. to CG
x:      1.4560e-02   y:      0.0000e+00   z:      3.0012e-01
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
  Glenohumeral joint center w.r. to CG
x:      1.4560e-02   y:      1.8179e-01   z:      1.8444e-01
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
  Hip joint center w.r. to CG
x:      1.4560e-02   y:      1.1360e-01   z:     -2.8834e-01
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00

**************************************************************
* Upper arm:
  Mass: 2.001e+00 kg
  Length: 2.9987e-01 m
  Center of gravity w.r. to reference point:
x:  0.0000e+00   y:  0.0000e+00   z: -1.4394e-01
  Inertia tensor w.r. to center of gravity:
xx:  1.529826e-02   yy:  1.421709e-02   zz:  2.594819e-03
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00
  Glenohumeral joint center w.r. to CG
x:      0.0000e+00   y:      0.0000e+00   z:      1.4394e-01
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
  Ellbow joint center w.r. to CG
```

```
x:      0.0000e+00   y:      0.0000e+00   z:     -1.5593e-01
eul1:   0.0000e+00   eul2:  -9.0000e+01   eul3:   0.0000e+00

**************************************************************
* Forearm:
  Mass: 1.227e+00 kg
  Length: 2.6696e-01 m
  Center of gravity w.r. to reference point:
x:  0.0000e+00   y:  0.0000e+00   z: -1.0946e-01
  Inertia tensor w.r. to center of gravity:
xx:  7.378079e-03   yy:  7.134753e-03   zz:  1.065444e-03
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00
  Ellbow joint center w.r. to CG
x:      0.0000e+00   y:      0.0000e+00   z:      1.0946e-01
eul1:   0.0000e+00   eul2:  -9.0000e+01   eul3:   0.0000e+00
  Wrist joint center w.r. to CG
x:      0.0000e+00   y:      0.0000e+00   z:     -1.5751e-01
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00

**************************************************************
* Hand:
  Mass: 4.736e-01 kg
  Length: 1.8640e-01 m
  Center of gravity w.r. to reference point:
x:  0.0000e+00   y:  0.0000e+00   z: -5.5794e-02
  Inertia tensor w.r. to center of gravity:
xx:  9.045808e-04   yy:  7.727524e-04   zz:  2.962932e-04
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00
  Wrist joint center w.r. to CG
x:      0.0000e+00   y:      0.0000e+00   z:      5.5794e-02
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
  Palmlength: 0.109 m

**************************************************************
* Thigh:
  Mass: 7.485e+00 kg
  Length: 4.2453e-01 m
  Center of gravity w.r. to reference point:
x:  0.0000e+00   y:  0.0000e+00   z: -1.7406e-01
  Inertia tensor w.r. to center of gravity:
xx:  1.290498e-01   yy:  1.339172e-01   zz:  2.675346e-02
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00
  Hip joint center w.r. to CG
```

```
x:      0.0000e+00   y:      0.0000e+00   z:      1.7406e-01
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
  Knee joint center w.r. to CG
x:      0.0000e+00   y:      0.0000e+00   z:     -2.5047e-01
eul1:   0.0000e+00   eul2:  -9.0000e+01   eul3:   0.0000e+00

**************************************************************
* Shank:
  Mass: 3.132e+00 kg
  Length: 4.1232e-01 m
  Center of gravity w.r. to reference point:
x:  0.0000e+00   y:  0.0000e+00   z: -1.8142e-01
  Inertia tensor w.r. to center of gravity:
xx:  4.534649e-02   yy:  4.482954e-02   zz:  3.891769e-03
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00
  Knee joint center w.r. to CG
x:      0.0000e+00   y:      0.0000e+00   z:      1.8142e-01
eul1:   0.0000e+00   eul2:  -9.0000e+01   eul3:   0.0000e+00
  Ankle joint center w.r. to CG
x:      0.0000e+00   y:      0.0000e+00   z:     -2.3090e-01
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00

**************************************************************
* Foot, one segment (NASA x- and z-axis interchanged here!):
  Mass: 9.764e-01 kg
  Length: 2.6211e-01 m
  Center of gravity w.r. to reference point:
x:  5.5044e-02   y:  0.0000e+00   z: -4.2799e-02
  Inertia tensor w.r. to center of gravity:
xx:  4.112351e-03   yy:  3.743683e-03   zz:  9.521994e-04
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00
  Ankle joint center w.r. to CG
x:     -5.5044e-02   y:      0.0000e+00   z:      4.2799e-02
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
  Ground contact triad heel w.r. to CG
x:     -8.9119e-02   y:      0.0000e+00   z:     -4.2799e-02
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
  Ground contact triad medial ball w.r. to CG
x:      8.1255e-02   y:     -2.8833e-02   z:     -4.2799e-02
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
  Ground contact triad lateral ball w.r. to CG
x:      6.8150e-02   y:      3.4075e-02   z:     -4.2799e-02
eul1:   0.0000e+00   eul2:   0.0000e+00   eul3:   0.0000e+00
```

```
  Ankle height:    8.559800e-02 m
  Ankle distance from heel:    6.028622e-02 m

**************************************************************
* Total body (suspicious data!):
  Mass: 7.200e+01 kg
  Length: 1.7400e+00 m
  Center of gravity w.r. to reference point:
x:  0.0000e+00   y:  0.0000e+00   z:  9.5700e-01
  Inertia tensor w.r. to center of gravity:
xx:  0.000000e+00   yy:  0.000000e+00   zz:  0.000000e+00
xy:  0.000000e+00   xz:  0.000000e+00   yz:  0.000000e+00

**************************************************************
* Other data:
  Shoulder height: 1.420 m
  Shoulder joint height: 1.395 m
  Head circumference: 0.566 m
  Neck circumference: 0.372 m
  Shoulder breadth (biacromical breadth): 0.403 m
  Chest/bust depth: 0.243 m
  Chest breadth: 0.324 m
  Hip breadth: 0.350 m
```

Anhang C

Numerical Recipes - Routinen

Der verwendete C-Code für den 'Downhill Simplex'-Algorithmus stammt aus dem Buch 'Numerical Recipes in C' von Press, Flannery, Teukolsky & Vetterling. [PTVF89]

C.1 amotry.c

```
#define NRANSI
#include "nrutil.h"

float amotry(float **p, float y[], float psum[], int ndim,
float (*funk)(float []), int ihi, float fac)
{
int j;
float fac1,fac2,ytry,*ptry;

ptry=vector(1,ndim);
fac1=(1.0-fac)/ndim;
fac2=fac1-fac;
for (j=1;j<=ndim;j++) ptry[j]=psum[j]*fac1-p[ihi][j]*fac2;
ytry=(*funk)(ptry);
if (ytry < y[ihi]) {
y[ihi]=ytry;
for (j=1;j<=ndim;j++) {
psum[j] += ptry[j]-p[ihi][j];
p[ihi][j]=ptry[j];
}
}
```

```
free_vector(ptry,1,ndim);
return ytry;
}

#undef NRANSI
/* (C) Copr. 1986-92 Numerical Recipes Software W73. */
```

C.2 amoeba.c

```
#include <math.h>
#define NRANSI
#include "nrutil.h"
#define NMAX 5000
#define GET_PSUM \
for (j=1;j<=ndim;j++)
   {
       for (sum=0.0,i=1;i<=mpts;i++) sum += p[i][j];
       psum[j]=sum;
   }
#define SWAP(a,b) {swap=(a);(a)=(b);(b)=swap;}

float amotry(float **p, float y[], float psum[], int ndim,
float (*funk)(float []), int ihi, float fac)
{
int j;
float fac1,fac2,ytry,*ptry;

ptry=vector(1,ndim);
fac1=(1.0-fac)/ndim;
fac2=fac1-fac;
for (j=1;j<=ndim;j++) ptry[j]=psum[j]*fac1-p[ihi][j]*fac2;
ytry=(*funk)(ptry);
if (ytry < y[ihi]) {
y[ihi]=ytry;
for (j=1;j<=ndim;j++) {
psum[j] += ptry[j]-p[ihi][j];
p[ihi][j]=ptry[j];
}
```

```
}
free_vector(ptry,1,ndim);
return ytry;
}

void amoeba(float **p, float y[], int ndim, float ftol,
float (*funk)(float []), int *nfunk)
{
float amotry(float **p, float y[], float psum[], int ndim,
float (*funk)(float []), int ihi, float fac);
int i,ihi,ilo,inhi,j,mpts=ndim+1;
float rtol,sum,swap,ysave,ytry,*psum;

psum=vector(1,ndim);
*nfunk=0;
GET_PSUM
for (;;) {
ilo=1;
ihi = y[1]>y[2] ? (inhi=2,1) : (inhi=1,2);
for (i=1;i<=mpts;i++) {
if (y[i] <= y[ilo]) ilo=i;
if (y[i] > y[ihi]) {
inhi=ihi;
ihi=i;
} else if (y[i] > y[inhi] && i != ihi) inhi=i;
}
rtol=2.0*fabs(y[ihi]-y[ilo])/(fabs(y[ihi])+fabs(y[ilo]));
if (rtol < ftol) {
SWAP(y[1],y[ilo])
for (i=1;i<=ndim;i++) SWAP(p[1][i],p[ilo][i])
break;
}
if (*nfunk >= NMAX) nrerror("NMAX exceeded");
*nfunk += 2;
ytry=amotry(p,y,psum,ndim,funk,ihi,-1.0);
if (ytry <= y[ilo])
ytry=amotry(p,y,psum,ndim,funk,ihi,2.0);
else if (ytry >= y[inhi]) {
ysave=y[ihi];
ytry=amotry(p,y,psum,ndim,funk,ihi,0.5);
if (ytry >= ysave)
   {
     for (i=1;i<=mpts;i++)
```

```
  {
  if (i != ilo)
  {
  for (j=1;j<=ndim;j++)
  p[i][j]=psum[j]=0.5*
(p[i][j]+p[ilo][j]);
  y[i]=(*funk)(psum);
}
}
*nfunk += ndim;
GET_PSUM
}
} else --(*nfunk);
}
free_vector(psum,1,ndim);
}
#undef SWAP
#undef GET_PSUM
#undef NMAX
#undef NRANSI
/* (C) Copr. 1986-92 Numerical Recipes Software W73. */
```

C.3 integrat.h

Standard-Integrator (*Runge-Kutta*-4.Ordnung) mit variabler Schrittweitensteuerung nach [PTVF89].

```
 #define SIGN(a,b) ((b) >= 0.0 ? fabs(a) : -fabs(a))

static  FLOAT maxarg1,maxarg2;
#define FMAX(a,b)
  (maxarg1=(a),maxarg2=(b),(maxarg1) > (maxarg2) ? (maxarg1) : (maxarg2))

static  FLOAT minarg1,minarg2;
#define FMIN(a,b)
  (minarg1=(a),minarg2=(b),(minarg1) < (minarg2) ? (minarg1) : (minarg2))
```

```
void nrerror(char error_text[])
{
    fprintf(stderr, "Numerical Recipes run-time error... \n ");
    fprintf(stderr, "%s \n ",error_text);
    fprintf(stderr, "...now exiting to system... \n ");
    exit(1);
}

 FLOAT dym_rk4[N+1], dyt_rk4[N+1], yt_rk4[N+1];
void rk4( FLOAT y[],  FLOAT dydx[], int n,  FLOAT x,  FLOAT h,  FLOAT yout[],
    void (*derivs)( FLOAT,  FLOAT [],  FLOAT []))
{
    int i;    FLOAT xh,hh,h6;
    hh=h*0.5;
    h6=h/6.0;
    xh=x+hh;
    for (i=1;i<=n;i++) yt_rk4[i]=y[i]+hh*dydx[i];
    (*derivs)(xh,yt_rk4,dyt_rk4);
    for (i=1;i<=n;i++) yt_rk4[i]=y[i]+hh*dyt_rk4[i];
    (*derivs)(xh,yt_rk4,dym_rk4);
    for (i=1;i<=n;i++) {
        yt_rk4[i]=y[i]+h*dym_rk4[i];
        dym_rk4[i] += dyt_rk4[i];
    }
    (*derivs)(x+h,yt_rk4,dyt_rk4);
    for (i=1;i<=n;i++)
        yout[i]=y[i]+h6*(dydx[i]+dyt_rk4[i]+2.0*dym_rk4[i]);
}
 FLOAT ak2_rkck[N+1], ak3_rkck[N+1], ak4_rkck[N+1],
      ak5_rkck[N+1], ak6_rkck[N+1], ytemp_rkck[N+1];
void rkck( FLOAT y[],  FLOAT dydx[], int n,  FLOAT x,  FLOAT h,  FLOAT yout[],
     FLOAT yerr[], void (*derivs)( FLOAT,  FLOAT [],  FLOAT []))
{
    int i;
    static  FLOAT a2=0.2,a3=0.3,a4=0.6,a5=1.0,a6=0.875,b21=0.2,
        b31=3.0/40.0,b32=9.0/40.0,b41=0.3,b42 = -0.9,b43=1.2,
        b51 = -11.0/54.0, b52=2.5,b53 = -70.0/27.0,b54=35.0/27.0,
        b61=1631.0/55296.0,b62=175.0/512.0,b63=575.0/13824.0,
        b64=44275.0/110592.0,b65=253.0/4096.0,c1=37.0/378.0,
        c3=250.0/621.0,c4=125.0/594.0,c6=512.0/1771.0,
        dc5 = -277.00/14336.0;
/*  FLOAT dc1=c1-2825.0/27648.0,dc3=c3-18575.0/48384.0,
```

```
dc4=c4-13525.0/55296.0,dc6=c6-0.25;  */
    static  FLOAT dc1=(37.0/378.0)-2825.0/27648.0,
  dc3=(250.0/621.0)-18575.0/48384.0,
        dc4=(125.0/594.0)-13525.0/55296.0,dc6=(512.0/1771.0)-0.25;

    for (i=1;i<=n;i++)
        ytemp_rkck[i]=y[i]+b21*h*dydx[i];
    (*derivs)(x+a2*h,ytemp_rkck,ak2_rkck);
    for (i=1;i<=n;i++)
        ytemp_rkck[i]=y[i]+h*(b31*dydx[i]+b32*ak2_rkck[i]);
    (*derivs)(x+a3*h,ytemp_rkck,ak3_rkck);
    for (i=1;i<=n;i++)
        ytemp_rkck[i]=y[i]+h*(b41*dydx[i]+b42*ak2_rkck[i]+b43*ak3_rkck[i]);
    (*derivs)(x+a4*h,ytemp_rkck,ak4_rkck);
    for (i=1;i<=n;i++)
        ytemp_rkck[i]=y[i]+h*(b51*dydx[i]+b52*ak2_rkck[i]+b53*ak3_rkck[i]+
          b54*ak4_rkck[i]);
    (*derivs)(x+a5*h,ytemp_rkck,ak5_rkck);
    for (i=1;i<=n;i++)
        ytemp_rkck[i]=y[i]+h*(b61*dydx[i]+b62*ak2_rkck[i]+b63*ak3_rkck[i]+
          b64*ak4_rkck[i]+b65*ak5_rkck[i]);
    (*derivs)(x+a6*h,ytemp_rkck,ak6_rkck);
    for (i=1;i<=n;i++)
        yout[i]=y[i]+h*(c1*dydx[i]+c3*ak3_rkck[i]
+c4*ak4_rkck[i]+c6*ak6_rkck[i]);
    for (i=1;i<=n;i++)
        yerr[i]=h*(dc1*dydx[i]+dc3*ak3_rkck[i]+dc4*ak4_rkck[i]
   +dc5*ak5_rkck[i]+dc6*ak6_rkck[i]);
}

#define SAFETY 0.9
#define PGROW -0.2
#define PSHRNK -0.25
#define ERRCON 1.89e-4
 FLOAT yerr_rkqs[N+1], ytemp_rkqs[N+1];
void rkqs( FLOAT y[],  FLOAT dydx[], int n,  FLOAT *x,  FLOAT htry,
FLOAT eps,  FLOAT yscal[],  FLOAT *hdid,  FLOAT *hnext,
    void (*derivs)( FLOAT,  FLOAT [],  FLOAT []))
{
    int i;
     FLOAT errmax,h,htemp,xnew;

    h=htry;
```

```
    for (;;) {
        rkck(y,dydx,n,*x,h,ytemp_rkqs,yerr_rkqs,derivs);
        errmax=0.0;
        for (i=1;i<=n;i++)
    errmax=FMAX(errmax,fabs(yerr_rkqs[i]/yscal[i]));
        errmax /= eps;
        if (errmax > 1.0) {
            htemp=SAFETY*h*pow(errmax,PSHRNK);
            h=(h >= 0.0 ? FMAX(htemp,0.1*h) : FMIN(htemp,0.1*h));
            xnew=(*x)+h;
            if (xnew == *x) nrerror( "stepsize underflow in rkqs ");
            continue;
        } else {
            if (errmax > ERRCON) *hnext=SAFETY*h*pow(errmax,PGROW);
            else *hnext=5.0*h;
            *x += (*hdid=h);
            for (i=1;i<=n;i++) y[i]=ytemp_rkqs[i];
            break;
        }
    }
}
#undef SAFETY
#undef PGROW
#undef PSHRNK
#undef ERRCON

#define MAXSTP 10000
#define TINY 1.0e-30
 FLOAT yscal_odeint[N+1],y_odeint[N+1],dydx_odeint[N+1];
void odeint( FLOAT ystart[], int nvar,  FLOAT x1,  FLOAT x2,  FLOAT eps,
FLOAT *h1, FLOAT hmin, int *nok, int *nbad,
        void (*derivs)( FLOAT,  FLOAT [],  FLOAT []),
        void (*rkqs)( FLOAT [],  FLOAT [], int,  FLOAT *,  FLOAT,  FLOAT,
              FLOAT [], FLOAT *,  FLOAT *, void (*)( FLOAT,  FLOAT [],
              FLOAT [])))
{
    int nstp,i;
     FLOAT x,hnext,hdid,h;

    x=x1;
    h=SIGN(*h1,x2-x1);
    /* *nok = (*nbad) = 0; */
    for (i=1;i<=nvar;i++) y_odeint[i]=ystart[i];
```

```
    for (nstp=1;nstp<=MAXSTP;nstp++) {
        (*derivs)(x,y_odeint,dydx_odeint);
        for (i=1;i<=nvar;i++)
            yscal_odeint[i]=fabs(y_odeint[i])+fabs(dydx_odeint[i]*h)+TINY;
        if ((x+h-x2)*(x+h-x1) > 0.0) h=x2-x;
        (*rkqs)(y_odeint,dydx_odeint,nvar,&x,h,eps,yscal_odeint,&hdid,
        &hnext,derivs);
        if (hdid == h) ++(*nok); else ++(*nbad);
        *h1 = hnext;
        if ((x-x2)*(x2-x1) >= 0.0) {
            for (i=1;i<=nvar;i++) ystart[i]=y_odeint[i];
            return;
        }
        if (fabs(hnext) <= hmin) nrerror( "Step size too small in odeint ");
        h=hnext;
    }
    nrerror( "Too many steps in routine odeint ");
}
#undef MAXSTP
#undef TINY

/* (C) Copr. 1986-92 Numerical Recipes Software t4. */
```

Literaturverzeichnis

[AB93] A. Arampatzis and G.-P. Brüggemann. *Optimierung der Vorbereitungsphase für Abgänge und Flugteile beim Reck*, pages 181–192. 1993.

[Alp93] Alp, Y. and Brüggemann, G.-P. and Cheetham, P. *Kinematische Analyse von Flugteilen am Reck während der XXV. Olympischen Sommerspiele in Barcelona 1992*, pages 27–37. 1993.

[Ara95] A. Arampatzis. *Mathematische Modellierung zur Überprüfung und Identifikation von Bewegungslösungen im Kunstturnen - angewandt auf die Riesenfelgen am Reck*, volume 6 of *Arbeiten zur angewandten Trainings- und Bewegungswissenschaft.* Sport und Buch Strauss - Edition Sport Köln, 1995.

[Ara98] Arampatzis, A. and Brüggemann, G.-P. Mechanical Energetic Process during the Giant Swing Exercise before Dismounts and Flight Elements on the High Bar and the Uneven Parallel Bars. 1998.

[Aus88] Auspurg,D. Computersimulation von Bewegungsabläufen als ein Beitrag für die Erarbeitung wirksamer sporttechnischer Lösungsverfahren im Gerätturnen. *Theorie und Praxis des Leistungssports,26,46-54*, 1988.

[Aus90] Auspurg,D. Modellierung und Computersimulation im Gerätturnen am Beispiel des Gaylord-Saltos am Reck. *Training und Wettkampf, 28,2/3,76-86*, 1990.

[Bau76] W.L. Bauer. Mathematische Modellierung und Optimierung als Hilfsmittel zur Aufklärung des Lernvorgangs bei der Turnübung 'Riesenfelge am Reck'. *Regelungstechnik Band:10,11,12*, 1976.

[BMH76] J. Borms, R. Moers, and M. Hebbelinck. Biomechanical study of forward and backward swings. *Komi, P.V. (Hrsg.): Biomechanics V-B. University Park Press, Baltimore, 309-313*, 1976.

[Böh97] H. Böhm. Dynamik der Riesenfelge am Reck im Rahmen von Mehrkörpersystemen. *Diplomarbeit Universität Tübingen*, 1997.

[Boo97] A. Boose. Biomechanisches Modellieren als Mittel zur Untersuchung der Koordination von Mehrgelenksbewegungen bei Kleinhirnpatienten und Gesunden. *Dissertation Universität Tübingen*, 1997.

[BS85] Bronstein-Semendjajew. *Taschenbuch der Mathematik.* Verlag Harri Deutsch, 1985.

[Bud90] A. Budo. *Theoretische Mechanik.* Deutscher Verlag der Wissenschaften, 1990.

[dad94] *DADS Users Manual.* CADSI, 1994.

[Den98] J. Dengler. Vorwärtsdynamik in der Biomechanik des Sports. *Wissenschaftliche Arbeit Universität Tübingen*, 1998.

[Ehr92] Ehrhardt, M. Computersimulation für lernfähige biomechanische Modelle. *Diplomarbeit Universität Stuttgart*, 1992.

[FH85] T. Flash and N. Hogan. The coordination of arm movements : An experimentally confirmed mathematical model. *J Neurosci,5,1688-703*, 1985.

[Gol89] H. Goldstein. *Klasische Mechanik.* AULA-Verlag, Wiesbaden, 10. edition, 1989.

[Gru87] K. Gruber. Entwicklung eines Modells zur Berechnung der Kräfte im Knie- und Hüftgelenk bei hohen Beschleunigungen. *Dissertation Universität Tübingen*, 1987.

[Gue97] M. Guenther. Computersimulationen zur Synthetisierung des muskulär erzeugten menschlichen gehens unter Verwendung eines biomechanischen Mehrkörpermodells. *Dissertation Universität Tübingen*, 1997.

[Hah93] Hahn, U. Entwicklung mehrgliedriger Modelle zur realistischen Simulation dynamischer Prozesse in biologischen Systemen. *Dissertation Universität Tübingen*, 1993.

[Hau89] E.J. Haug. *Computer-Aided Kinematics and Dynamics of Mechanical Systems*, volume 1 of *Allyn and Bacon series in Engineering.* Allyn and Bacon Boston, 1989.

[Hoc67] G. Hochmuth. *Biomechanik Sportlicher Bewegungen.* Wilhelm Limpert-Verlag GmbH, Frankfurt (Main), 1967.

[Kas93] G. Kassat. *biomechanik für nicht-biomechaniker.* fitness-Contur-Verlag, 1993.

[Kni94] K. Knirsch. *Lehrbuch des Gerät- und Kunstturnens*, volume 2. Knirsch-Verlag, 1994.

[KR80] P.M. Kopp and G.J. Reid. A force and torque analysis of giant swings on the horizontal bar. *Can.J.Appl.Spt.Sci.,5(2),98-102*, 1980.

[KR90] Kernigan and Ritchie. *Programmieren in C.* Hanser, 1990.

[Kri92] Krieg, M. Simulation und Steuerung biomechanischer Mehrkörpersysteme. *Diplomarbeit Universität Tübingen*, 1992.

[Mut97] Mutschler, H. Biomechanische Simulation zielgerichteter Bewegungsabläufe, gesteuert durch neuronale Netze. *Diplomarbeit Universität Tübingen*, 1997.

[NAS78] NASA Reference Publication. Anthropometric Source Book. Technical Report 1024, NASA Scientific and Technical Information Office, Springfield, 1978.

[OSIY89] A. Okamoto, S. Sakurai, Y. Ikegami, and K. Yabe. The changes in mechanical energy during the giant swing backward on the horizontal bar. *Tsarouchas, L., Terauds, J., Gowitzke, B.A., Holt, L.E. (Hrsg.): Biomechanics in Sports V, hellenic Sports Research Institute, Athens, 338-345*, 1989.

[PTVF89] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C - The Art of Scientific Computing.* Cambridge University Press, 1989.

[REG+94] H. Ruder, T. Ertl, K. Gruber, M. Günther, F. Hospach, M. Ruder, J. Subke, and K. Widmayer. Kinematics and dynamics for computer animation. *Coquillart, S., Strasser, W., Stucki, P. (Hrsg.): From Object Modelling to Advanced Visual Communication, Springer Verlag*, 1994.

[Sch88] H. Schildt. *C-Befehlsbibliothek.* McGraw-Hill Book Company GmbH, 1988.

[SZ89] K. Schneider and R.F. Zernicke. Jerk-cost modulations during the practice of rapid arm movements. *Biol Cybern,60,221-30.*, 1989.

[Wid90] K. Widmayer. Simulation von Bewegungsabläufen beim Menschen mit Hilfe von Mehrkörpersystemen. *Diplomarbeit Universität Tübingen*, 1990.

[YKO79] N. Yamashita, M. Kumamoto, and T. Okamoto. Electromyographic study of forward and backward giant swing on the horizontal bar. *Terauds,J.,Daniels, D.B. (Hrsg.):Science in Gymnastics, Academic Publishers, Del Mar, California*, 1979.

Danksagung

Herrn Prof. Ruder danke ich für die freundliche Aufnahme in seinem Institut und dafür, daß er immer Zeit gefunden hat, über physikalische Probleme zu diskutieren und mir mit wertvollen Tips weiterzuhelfen.

Des weiteren gilt mein spezieller Dank allen Mitgliedern des Instituts für Theoretische Astrophysik und Computational Physics Tübingen (TAT).

Für ihre Diskussionsbereitschaft und Hilfe bei Problemen mit Betriebssystemen, Bewegungsgleichungen, C-Programmierung, Druckern, Kaffeemaschinen, LATEX, dem Lösen von Differentialgleichungen, und sonstigen Problemen aller Art, möchte ich

Harald Riffert, Andreas Boose, Christian Götz, Michael Günther, Roland Speith, Harald Böhm, Arnim Henze, Valentin Keppler, Helmut Mutschler und Erik Schnetter

besonders danken.

Herzlichen Dank auch an Adamantios Arampatzis aus Köln für die Hilfe bei der Suche nach neuer Literatur zum Thema 'Riesenfelge'.