Fachhochschule Bielefeld
Fachbereich Elektrotechnik
Labor für Prozeßmeßtechnik
und Automatisierungstechnik

Diplomarbeit

Evolutionsstrategische Optimierung der Regelung („Fuzzy–Control “) für ein Stab–Wagen–System in Simulation und Experiment

Edwin Bühler
Mat.Nr.:286692

13. Januar 1993

Inhaltsverzeichnis

1 Einleitung **4**

2 Simulation des Stab-Wagen-Systems **5**
2.1 Einführung 5
2.2 Das Physikalische System 6
2.3 Diskretisierung 9
2.3.1 Die Struktogramme der Unit Matrix 12
2.4 Visualisierung 15
2.4.1 Stab und Wagen darstellen 15
2.4.2 Neuen Winkel bestimmen 15
2.4.3 Aufruf vom Hauptprogramm 16
2.5 Hauptprogramm der Simulation 16

3 Fuzzy – Logik **19**
3.1 Allgemeines 19
3.2 Fuzzy–Sets 20
3.3 Linguistische Variablen 21
3.4 Operatoren 23
3.5 Regeln 25
3.6 Inferenz 27
3.6.1 Defuzzifizierung 27
3.7 Struktogramme der Fuzzy–Logik 28

4 Evolutionstrategie **31**
4.1 Einführung 31
4.2 (1+5)–gliedrige Evolutionsstrategie 31
4.3 Die Programmteile der ES 32
4.4 Optimierung der Variablen 34
4.5 Kennlinienfelder 38
4.6 Zusammenfassung und Ausblick 47

5 Das reale System **52**
5.1 Anbindung der Hardware 53
5.1.1 Die analogen und digitalen Signale 53
5.1.2 Ansteuerung der AD/DA Karte 54
5.2 Das Regelungs–Programm 54

A Die Programmlistings **57**
A.1 Hauptprogramm der Simulation 57
A.2 Die Unit Stab Sim 64
A.3 Die erweiterte Unit FuzzyE 69
A.4 Die Unit Matrix 84
A.5 Die Unit Fuzzycon 87
A.6 Das Programm Stabreg 88
A.7 Schaltpläne 93

Abbildungsverzeichnis **98**

Tabellenverzeichnis **99**

Literaturverzeichnis **100**

Kapitel 1

Einleitung

Mit Hilfe unscharfer Logik („fuzzy control “) soll die Regelung eines Stab–Wagen–Systems („inverted pendulum “–Problem) vorgenommen werden. Das Ziel der Regelung besteht darin, durch die Ortsänderung des Wagens den Auslenkwinkel $\varphi(t)$ des Stabes zu Null zu machen, d. h. den Stab in Balance zu halten. Für eine Optimierung können die Zugehörigkeitsmaße der unscharfen Regeln, die Regeln selber, die Anzahl der Linguistischen Variablen oder die Anzahl der unscharfen Mengen variiert werden. Für die Festlegung dieser Parameter gibt es bisher noch keine geeigneten Verfahren. Daher liegt es nahe, diese Parameter durch Anwendung von Monte–Carlo–Verfahren (z. B. „Evolutionsstrategie “) zu optimieren.

Der erste Teil der Diplomarbeit besteht aus der Simulation des Stab–Wagen–Systems . Dazu wird das pysikalische System mit der Zustandsraumbeschreibung simuliert, und auf dem Monitor durch eine Animation dargestellt. Für die Regelung wird die Fuzzy–Logik eingesetzt.

Im zweiten Teil werden die unscharfen Mengen mit Hilfe der Evolutionsstrategie optimiert. In geringen Umfang können durch diese Optimierung auch die Regeln nachträglich korrigiert werden.

Und im dritten Teil wird der Rechner mit dem realen Stab–Wagen–System verbunden. Mit Hilfe des Hardwareaufbaus kann dadurch die Fuzzy–Regelung ausgetestet werden.

Kapitel 2

Simulation des Stab-Wagen-Systems

2.1 Einführung

Um das Stab–Wagen–System zu simulieren ist eine Systembeschreibung notwendig. Zunächst wird das Physikalische System mathematisch untersucht .

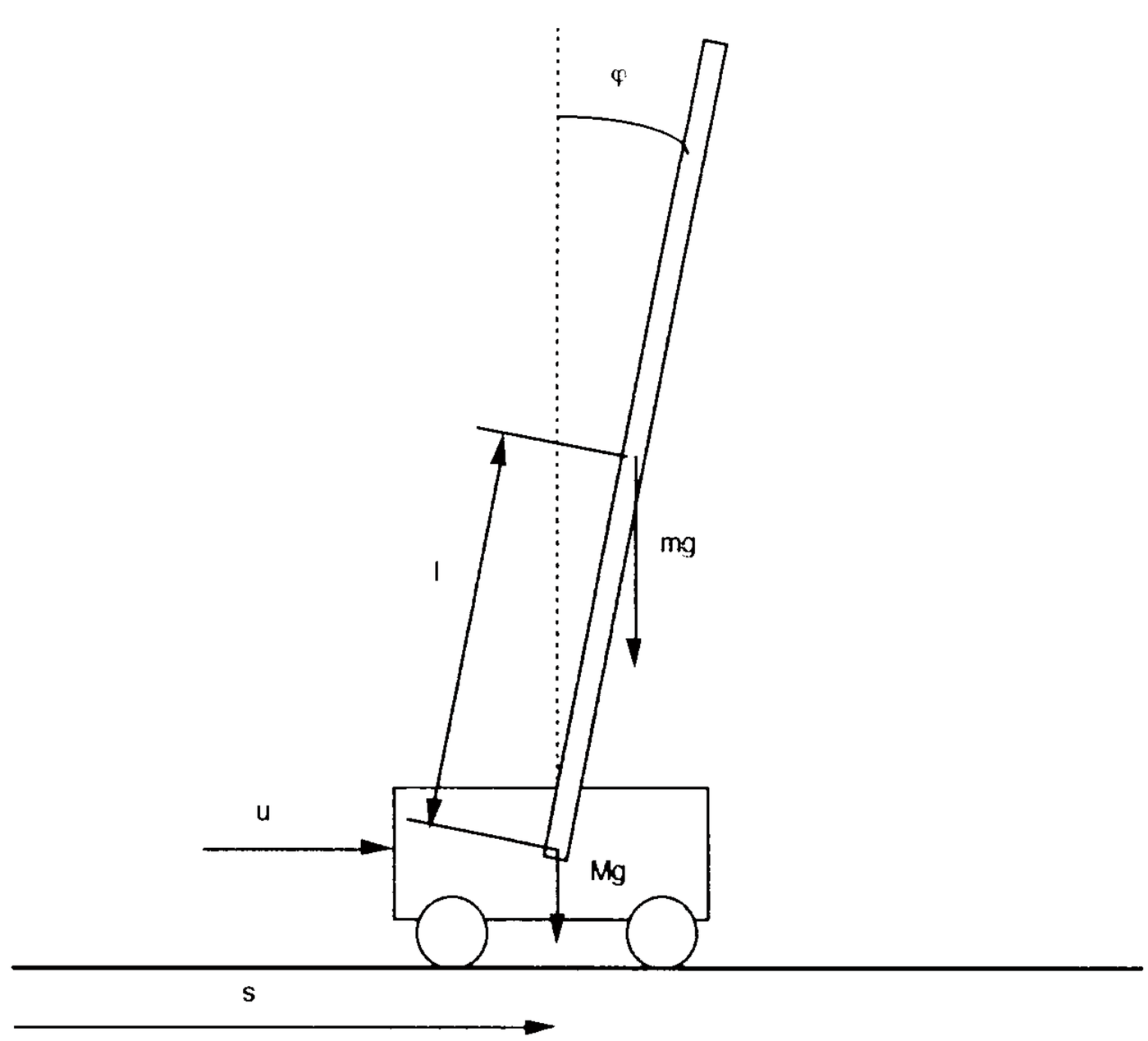

Abbildung 2.1: Stab-Wagen-System

Die so gewonnene Zustandsraumbeschreibung ist für die Diskretisierung erforder-

lich. Mit ihr kann für jeden Zeitpunkt, der sich aus der eingestellten Schrittweite ergibt, der momentane Zustand des Stab–Wagen–Systems berechnet und dargestellt werden.

Das Prinzip des Stab–Wagen–Systems und die verwendeten Größen ist in Bild (2.1) dargestellt. Der Wagen wird durch die Kraft u beschleunigt. Diese Größe entspricht der Stellgröße des Systems. Die Beschleunigung des Wagens hat Auswirkungen auf die Position s des Wagens und den Winkel φ des Stabes. Wobei bei senkrecht stehendem Stab der Winkel gleich null ist, und die Startposition ebenfalls null ist.

Der Schwerpunkt eines homogenen Stabes liegt auf halber Stablänge, die mit l bezeichnet wird. M entspricht der Masse des Wagens und m der Masse des Stabes.

2.2 Das Physikalische System

Um die Kräfte, die auf den Stab wirken zu verdeutlichen, sind diese in Bild (2.2) noch einmal dargestellt.

Es werden im Drehpunkt des Stabes folgende Kräfte übertragen.

H = Horizontalkraft des Wagens auf den Stab

V = Vertikalkraft des Wagens auf den Stab

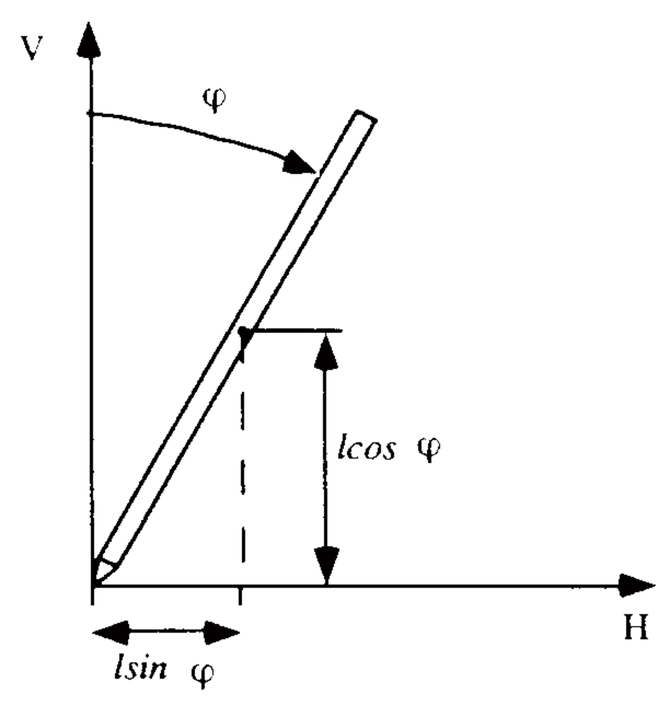

Abbildung 2.2: Kräfteverteilung

Horizontalbewegung des Stabschwerpunktes:

$$H = m\frac{d^2}{dt^2}(s + lsin\varphi) \tag{2.1}$$

Vertikalbewegung des Stabschwerpunktes:

$$V - mg = m\frac{d^2}{dt^2}(lcos\varphi) \tag{2.2}$$

Drehbewegung des Stabes um seinen Schwerpunkt:

$$\theta \frac{d^2\theta}{dt^2} = Vlsin\varphi - Hlcos\varphi \tag{2.3}$$

Trägheitsmoment des Stabes um seinen Schwerpunkt bei homogener Massenverteilung $\theta = ml^2/3$.
Horizontalbewegung des Wagens:

$$M\frac{d^2s}{dt^2} = u - H \tag{2.4}$$

Aus Gleichung (2.1) und (2.2) wird nach der Differentation der Winkelfunktionen:

$$V = mg - ml(\ddot{\varphi}sin\varphi + \dot{\varphi}^2cos\varphi) \tag{2.5}$$

$$H = m(\ddot{s} + \ddot{\varphi}cos\varphi - l\dot{\varphi}^2sin\varphi) \tag{2.6}$$

Nun wird in Gleichung (2.3) V und H aus Gl.(2.5) und (2.6) eingesetzt.

$$\begin{aligned} \theta\frac{d^2\varphi}{dt^2} &= [mg - ml(\ddot{\varphi}sin\varphi + \dot{\varphi}^2cos\varphi)]lsin\varphi \\ &\quad - [m(\ddot{s} + \ddot{\varphi}cos\varphi - l\dot{\varphi}^2sin\varphi)]lcos\varphi \end{aligned} \tag{2.7}$$

Nun werden die Gleichungen (2.4) und (2.6) zusammen gefaßt.

$$M\frac{d^2s}{dt^2} = u - m(\ddot{s} + \ddot{\varphi}cos\varphi - l\dot{\varphi}^2sin\varphi) \tag{2.8}$$

$$u = M\ddot{s} + m(\ddot{s} + l\ddot{\varphi}cos\varphi - l\dot{\varphi}^2sin\varphi)$$

In Gleichung (2.7) wird $\theta = ml^2/3$ eingesetzt.

$$\begin{aligned} \frac{1}{3}\ddot{\varphi} &= [g - l\ddot{\varphi}sin\varphi - l\dot{\varphi}^2cos\varphi]sin\varphi \\ &\quad - [s + l\varphi cos\varphi - l\varphi sin\varphi]cos\varphi \\ &= gsin\varphi - \ddot{s}cos\varphi - l\ddot{\varphi}sin^2\varphi - l\ddot{\varphi}cos^2\varphi \\ &= gsin\varphi - \ddot{s}cos\varphi - l\ddot{\varphi} \end{aligned}$$

Das ergibt:

$$\frac{4l}{3}\ddot{\varphi} + \ddot{s}cos\varphi - gsin\varphi = 0 \tag{2.9}$$

Die Funktionen Gleichung(2.8) und (2.9) werden linearisiert, da φ und $\dot{\varphi}$ nur kleine Werte sind.(Linearisierung:$sin\varphi \approx \varphi, cos \approx 1, \dot{\varphi}^2\varphi \approx 0$). Nach der Linearisierung ergeben die beiden Funktionen :

$$\begin{aligned} u &= (M + m)\ddot{s} + ml\ddot{\varphi} \\ \ddot{s} &= \frac{u - ml\ddot{\varphi}}{M + m} \end{aligned} \tag{2.10}$$

$$\frac{4l}{3} + \ddot{s} - g\varphi = 0 \tag{2.11}$$

Um die Bestimmungsgleichung für $\ddot{\varphi}$ zu bekommen, setzt man in die Gleichung (2.11) die für $\ddot{s}$ gefundene Gleichung (2.10) ein.

$$\begin{aligned}
\frac{4l}{3}\ddot{\varphi} - g\varphi + \frac{u - ml\ddot{\varphi}}{M+m} &= 0 \\
\frac{\ddot{\varphi}4l(M+m) - 3g\varphi(M+m) + 3(u - ml\ddot{\varphi})}{3(M+m)} &= 0 \\
\ddot{\varphi}4l(M+m) - 3g\varphi(M+m) + 3(u - ml\ddot{\varphi}) &= 0 \\
\ddot{\varphi} &= \frac{3g\varphi(M+m) - 3u}{4l(M+m) - 3ml} \\
\ddot{\varphi} &= \frac{3g(M+m)}{l(4M+m)}\varphi - \frac{3u}{l(4M+m)} \qquad (2.12)
\end{aligned}$$

Setzt man die Gleichung (2.12) in die Gleichung (2.10) ein , so erhält man die Bestimmungsgleichung für $\ddot{s}$:

$$\begin{aligned}
\ddot{s} &= \frac{u}{M+m} - \frac{ml(3g(M+m)\varphi - 3u)}{l(4M+m)(M+m)} \\
\ddot{s} &= \frac{(4M+m)u + 3mu}{(4M+m)(M+m)} - \frac{3gm\varphi(M+m)}{(4M+m)(M+m)} \\
\ddot{s} &= \frac{u((4M+m)+3m)}{(4M+m)(M+m)} - \frac{3gm\varphi(M+m)}{(4M+m)(M+m)} \\
\ddot{s} &= \frac{4M+4m}{(4M+m)(M+m)} - \frac{3gm}{4M+m}\varphi \qquad (2.13)
\end{aligned}$$

Die Koeffizienten der Zustandsraumbeschreibung erhält man , indem man die Konstanten der beiden Gleichung (2.12) und Gleichung (2.13) wie folgt zusammenfaßt:

$$a_{21} = \frac{3g(M+m)}{l(4M+m)} \qquad a_{41} = \frac{-3gm}{4M+m}$$

$$b_2 = \frac{-3}{l(4M+m)} \qquad b_4 = \frac{4}{4M+m}$$

Somit lauten die Bestimmungsgleichungen:

$$\ddot{\varphi} = a_{21}\varphi + b_2 u \tag{2.14}$$

$$\ddot{s} = a_{41}\varphi + b_4 u \tag{2.15}$$

Mit diesen beiden Gleichungen läßt sich nun die Beschreibung im Zustandsraum durchführen.

2.3 Diskretisierung

Für die Simulation muß aus der Zustandsraumbeschreibung der jeweils neue Zustand berechnet werden. Das heißt, aus $\underline{x}_{(k)}$ ergibt sich durch die Diskretisierung der Zustandraumbeschreibung $\underline{x}_{(k+1)}$.

$$\begin{aligned} \underline{\dot{x}}(t) &= \underline{Fx}(t) + \underline{g}u(t) \\ \underline{y}(t) &= \underline{Cx}(t) + \underline{Dx}(t) \end{aligned} \tag{2.16}$$

Von dieser Beschreibung des Systems ausgehend, wird nach folgender Gleichung der nächste Zustandsvektor $\underline{x}_{(k+1)}$ berechnet.

$$\underline{x}_{(k+1)} = \underline{Ax}_{(k)} + \underline{b}u_{(k)} \tag{2.17}$$

Wobei die Vektoren in Gleichung (2.17) folgendermaßen gebildet werden.

$$\begin{aligned} \underline{b} &= \underline{Rg} \\ \underline{A} &= \underline{I} + \underline{FR} \end{aligned} \tag{2.18}$$

Die benötigte Reihe $\underline{R}$ wird abgebrochen, wenn eine genügende Genauigkeit erreicht ist. Die verwendete Abbruchbedingung ist:

$$max|R_{neu} - R_{alt}| \leq Schwelle(z.B.10^{-6}) \tag{2.19}$$

Die Reihe lautet dann folgendermaßen:

$$\underline{R} = T \sum_{K=0}^{\infty} (\underline{F}T)^K \frac{1}{(K+1)!} \tag{2.20}$$

Dabei gilt: $T = Abtastschrittweite$(z.B. 0.002sec)

In die Gleichung (2.17) werden nun die Vektoren aus (2.18) eingesetzt, woraus sich dann die folgende Gleichung ergibt.

$$\underline{x}_{(K+1)} = \underline{Ix}_{(K)} + \underline{FRx}_{(K)} + \underline{Rg}u_{(K)} \tag{2.21}$$

Aus dem Zustandsvektor ergibt sich der Ausgangsvektor $\underline{y}_{(K)}$.

$$\underline{y}_{(K)} = \underline{C}x_{(K)} \tag{2.22}$$

Procedure Y-Werte — berechnet die Ausgangs Y-Matrix

mul5(C,xk) (Yk := C * xk)
mul1(E,xk) (ze1 := E * xk)
xk1 := ze1
mul2(F,R) (ze4 := F * R)
mul1(ze4,xk) (ze1 := ze4 * xk)
add1(ze1,xk1) (ze1 := ze1 + xk1)
xk1 := ze1
mul1(R,b) (ze1 := R * b)
mul3(ze1,u) (zeu := ze1 *uk)
add(zeu,xk1) (ze1 := zeu + xk1)
xk1 := ze1
xk := xk1

Abbildung 2.3: Struktogramm Procedure Y-Werte

Initmatrix — Setzt die Startwerte und die notwendigen Matrizen.

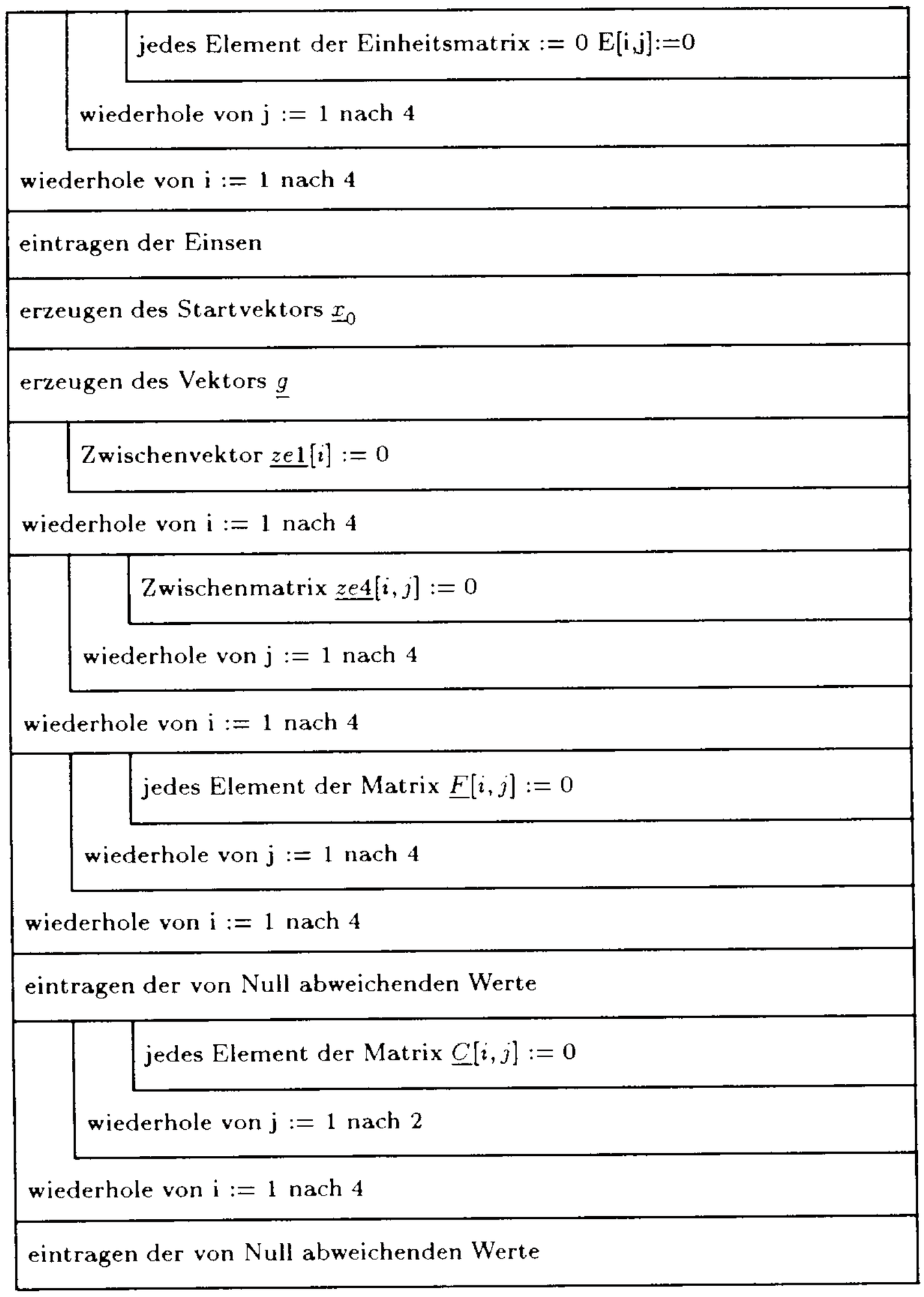

Abbildung 2.4: Struktogramm Procedure Initmatrix

2.3.1 Die Struktogramme der Unit Matrix

In dieser Unit werden die Vektoren und Matritzen erzeugt, die für die Diskretisierung benötigt werden und sich nicht mehr verändern.

Procedure mul1 — multipliziert eine [4,4] mit einer [1,4] Matrix

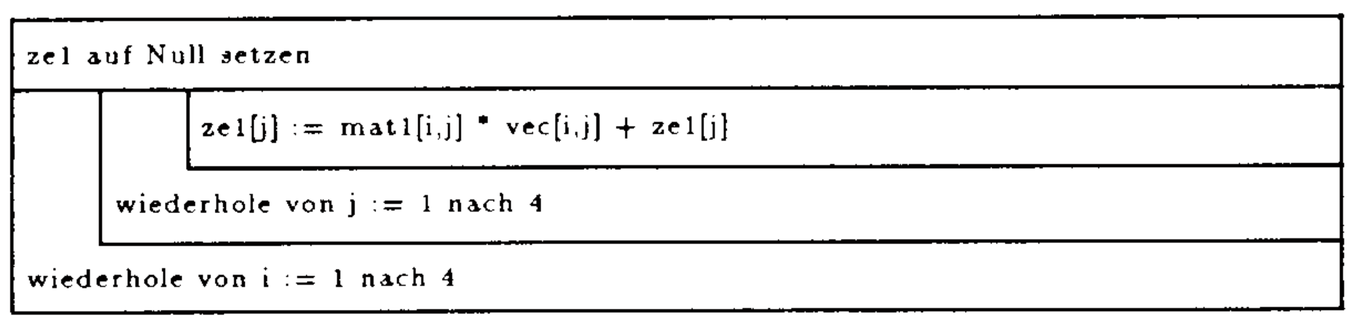

Abbildung 2.5: Struktogramm Procedure mul1

Procedure mul2 — multipliziert eine [4,4] mit einer [4,4] Matrix

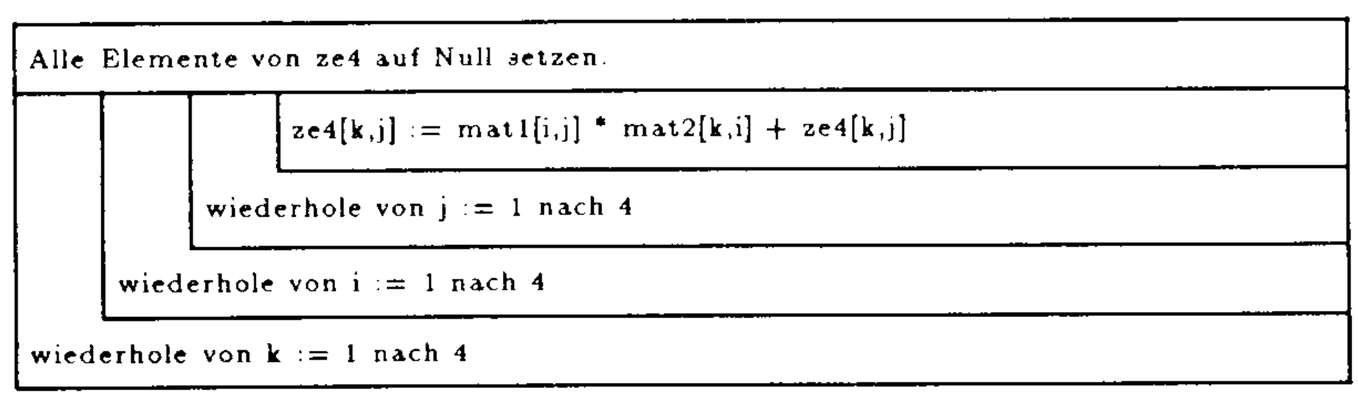

Abbildung 2.6: Struktogramm Procedure mul2

Procedure mul3 — multipliziert einen vektor mit einem Skalar

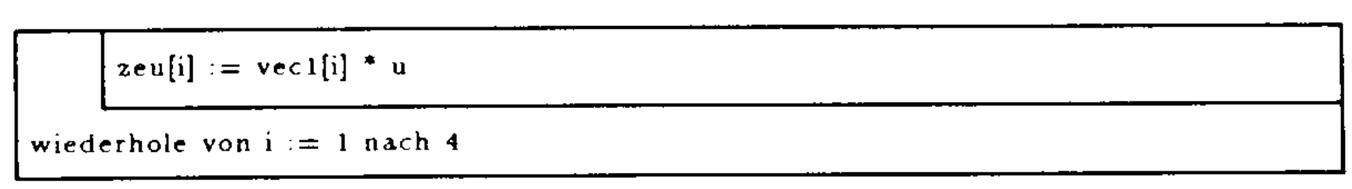

Abbildung 2.7: Struktogramm Procedure mul3

Procedure mul4 — multipliziert eine [4,4] mit einem Skalar

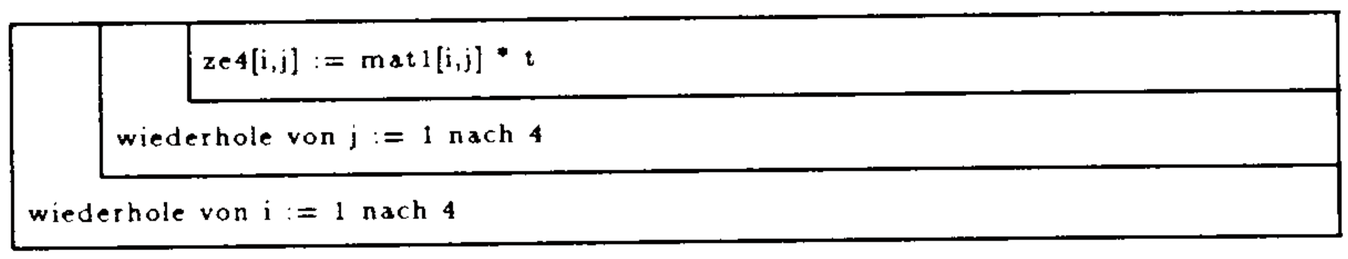

Abbildung 2.8: Struktogramm Procedure mul4

Procedure mul5 — multipliziert eine [4,2] mit einer [1,4] Matrix

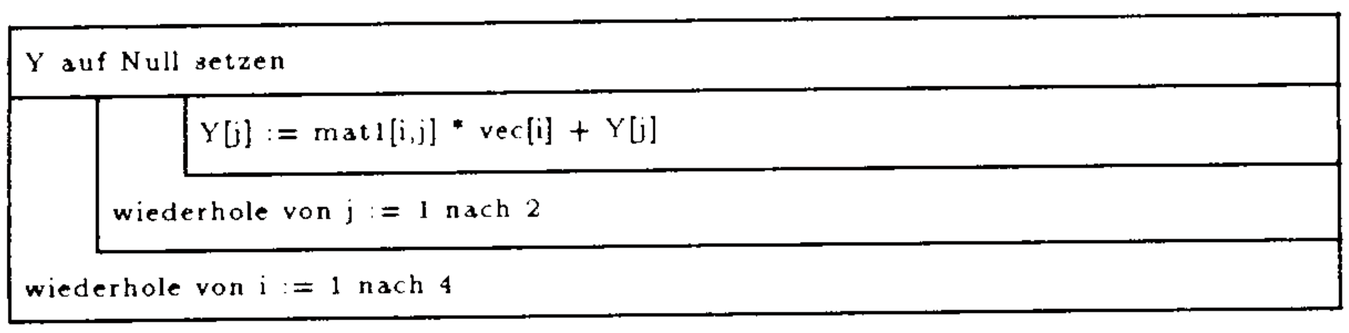

Abbildung 2.9: Struktogramm Procedure mul5

Procedure add1 — addiert zwei [1,4] Vektoren

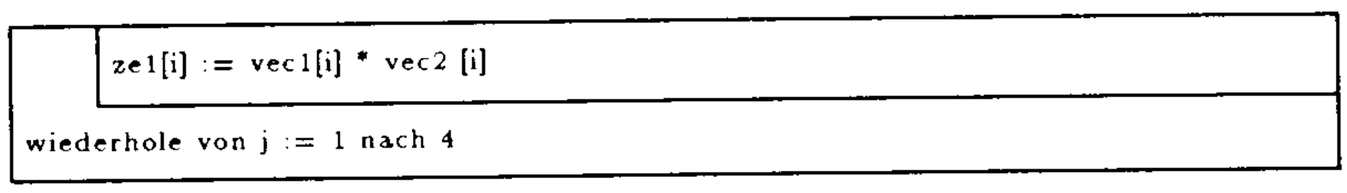

Abbildung 2.10: Struktogramm Procedure add1

Procedure add4 — addiert zwei [4,4] Matrizen

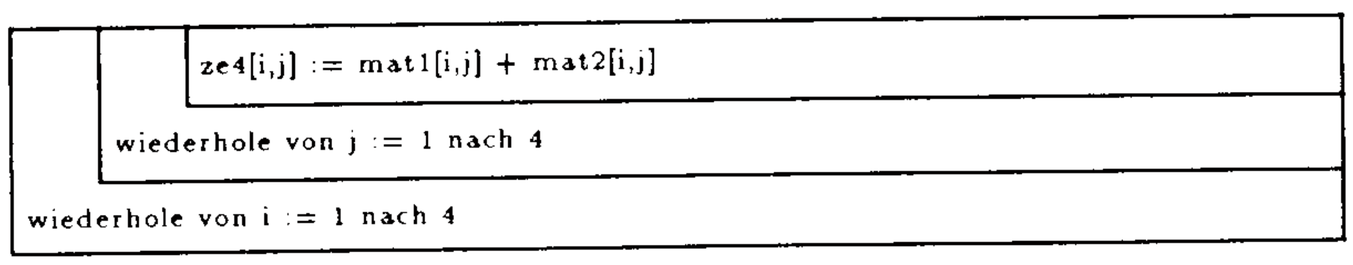

Abbildung 2.11: Struktogramm Procedure add4

Procedure R-matrix — berechnet die Matrix R

mul4(E,T) (ze4 := E * T)
R := ze4
T2 := T * T
mul4(F,T2) (ze4 := F * T2)
mul4(ze4,0.5) (ze4 := ze4 * 0.5)
add4(ze4,R) (ze4 := ze4 + R)
R := ze4
T3 := T * T * T
mul2(F,F) (ze4 := F * F)
mul4(ze4,T3) (ze4 := ze4 * T3)
mul4(ze4,0.16666 (ze4 := ze4 * 0.16666)
add4(ze4,R) (ze4 := ze4 + R)
R := ze4

Abbildung 2.12: Struktogramm Procedure R-matrix

2.4 Visualisierung

Das Stab-Wagen-System wird auf dem Bildschirm durch eine Animation dargestellt. Zusätzlich wird der sich ändernde Winkel in Form einer Kurve gezeichnet. Beides zusammen ermöglicht es die Simulation optisch zuverfolgen.

Die Proceduren, die für das Zeichnen und Löschen der grafischen Elemente zuständig sind, sind in der `Unit Stab-Sim` enthalten. Die meisten Variablen und Proceduren sind Bestandteil des Objects `Wagenstab`. So wird der momentane Zustand in diesen Variablen festgehalten.

Die Position der gesamten Darstellung wird durch die beiden Variablen `Bildposx` und `Bildposy` festgelegt. Diese beiden Werte sind absolute Koordinaten. Alle anderen Positionsangaben sind relative Koordinaten, die sich auf diese Beiden beziehen.

2.4.1 Stab und Wagen darstellen

Um den Stab zu zeichnen, muß die Position der Stabendpunkte berechnet werden. Der Drehpunkt des Stabes ergibt sich aus der konstanten Koordinate `y1` und der momentanen Wagenposition. Aus dem Stabwinkel muß nun die Spitze des Stabes berechnet werden. Dazu wird die Stablänge als Hypotenuse angenommen und dann aus dieser und dem Stabwinkel die Ankathete und Gegenkathete berechnet. Zu den Werten `x1` und `y1` werden nun die Katheten addiert bzw. subtrahiert. Daraus ergibt sich die Position der Stabspitze und der Stab kann nun in der für den Stab vorgesehenen Farbe gezeichnet werden.

Der Wagen wird gezeichnet , in dem an der aktuellen Wagenposition ein Rechteck in der entsprechenden Größe und Farbe gezeichnet wird.

Die Proceduren zum Löschen des Stabes und des Wagens entsprechen denen zum zeichnen. Nur wird die Zeichenfarbe in die Hintergrundfarbe umgewandelt.

2.4.2 Neuen Winkel bestimmen

In der `Procedure neuerwinkel` wird die Kraft, die auf den Wagen einwirkt der `Procedure y-werte` übergeben und in dieser diskretisiert. Danach wird aus der Ausgangsmatrix der neue Winkel in die Variable Stabwinkel übertragen. Damit der Winkel in dem Bereich zwischen +90[°] und −90[°] bleibt, wird der Stabwinkel auf +90[°]

Procedure Stabzeichnen — Der Stab wird an der aktuellen Position im richtigen Winkel gezeichnet.

Ankathete := Hypotenuse * cos(Stabwinkel)
Gegenkathete := Hypotenuse * sin(Stabwinkel)
x1 := BildpositionX + Wagenposition
y1 := BildpositionY
x2 := x1 + Gegenkathete
y2 := y1 - Ankathete
line(x1,y1,x2,y2) (Liniezeichnen)

Abbildung 2.13: Struktogramm Procedure Stabzeichnen

oder $-90[°]$ gesetzt wenn ein größerer bzw. kleinerer Winkel durch die Diskretisierung ermittelt wurde.

Wenn der Schalter `Caos` auf true steht, so wird zu dem Winkel ein kleiner Zufallswert addiert. Dieser Wert entsteht aus einer normal verteilten Zufallsfunktion. Diese Abänderung des Winkels simuliert die Störungen, die auf das System einwirken.

2.4.3 Aufruf vom Hauptprogramm

Im Hauptprogramm werden zuerst alle Variablen initialisiert. Anschließend wird die R–Matrix mit $t_{step} = 0,002sec$ berechnet.

Aus der Hauptschleife wird die `Procedure Wagensteht` aufgerufen. Diese Procedure ist mit den beiden Proceduren `rechtsfahren` und `linksfahren` zuständig für den Aufruf der Animationsproceduren.

2.5 Hauptprogramm der Simulation

Für die Erzeugung der Regelbasis und der Fuzzy–Sets werden Routinen des Fuzzy–Editors [16] verwendet. Da die Sets gleichmäßig über die jeweilige linguistische Variable verteilt sind, liegt es nahe diese automatisch zu erzeugen. Dazu existiert die Fuzzy–Datei „`Auto` “, die nur die drei linguistischen Variablen ohne Sets enthält.

Nach dem Start des Programms kann nun die Anzahl der Sets je Variable angegeben werden.Danach muß die Regelbasis eingegeben werden. Dies geschieht in Form einer Matrix, wobei für die Sets nur Zahlen eingegeben werden. Bei einer Set-Anzahl von 7 je Variable gelten die in Tabelle (2.1) angegebenen Vereinbarungen.

Linguistische Variable „Winkel “				
0	=	ssn	=	sehr-stark-negativ
1	=	sn	=	stark-negativ
2	=	n	=	negativ
3	=	null		
4	=	p	=	positiv
5	=	sp	=	stark-positiv
6	=	ssp	=	sehr-stark-positiv
Linguistische Variable „Omega “				
0	=	ssn	=	sehr-stark-negativ
1	=	sn	=	stark-negativ
2	=	n	=	negativ
3	=	null		
4	=	p	=	positiv
5	=	sp	=	stark-positiv
6	=	ssp	=	sehr-stark-positiv
Linguistische Variable „Wagen “				
0	=	ssl	=	sehr-stark-links
1	=	sl	=	stark-links
2	=	l	=	links
3	=	steht		
4	=	r	=	rechts
5	=	sr	=	stark-rechts
6	=	ssr	=	sehr-stark-rechts

Tabelle 2.1: Vereinbarung der Setbezeichnungen

In der Tabelle (2.1) sind die unterschiedlichen Bezeichnungen die für jeweils ein Set auftreten aufgeführt. Die Abkürzungen sind in Grafiken übersichtlicher und die Zahlen werden verwendet, da das Programm keine sinnvollen Namen automatisch erzeugen kann.
In den Abbildungen der Linguistischen Variablen sind die Sets mit den in Tab.(2.1) gegebenen Abkürzungen bezeichnet. Auch in anderen Grafiken und Tabellen wird auf diese Vereinbarungen zurückgegriffen. Die Zahlen für die Sets werden bei der automatischen Erzeugung der Sets generiert und im Programm verwendet. Die vollständigen Namen werden im Text benutzt.

Stellgröße		Wagen
0	⇒	sehr–stark–links
2048	⇒	steht
4095	⇒	sehr–stark–rechts

Für die Stellgröße werden Integer Zahlen verwendet, da diese an die AD/DA Schnittstelle übergeben werden.Die Aufteilung in einen negativen und positiven Bereich erfolgt im bipolarem Leistungsverstärker.

Aus den eingegebenen Daten wird von der `procedure autoset` die Regelbasis und die linguistischen Variablen in der in Bild (4.5) gezeigten Form erzeugt.

Soll die Optimierung beendet werden, so muß der Fuzzy–Editor aufgerufen und die Fuzzy–Datei „Elter "geladen werden. Nach dem Schließen des Editors und der Simulation mit „v "und „ESC "besteht die Möglichkeit die Regel–Matrix zu berechnen. Bei 101 Werten für den Winkel und für Omega dauert dies über drei Stunden.

Sobald die zu simulierende Zeit oder ein Winkel von ±20[°] erreicht wird, ruft das Programm die `procedure Naechsteskind` (im erweiterten Fuzzy–Editor) auf.

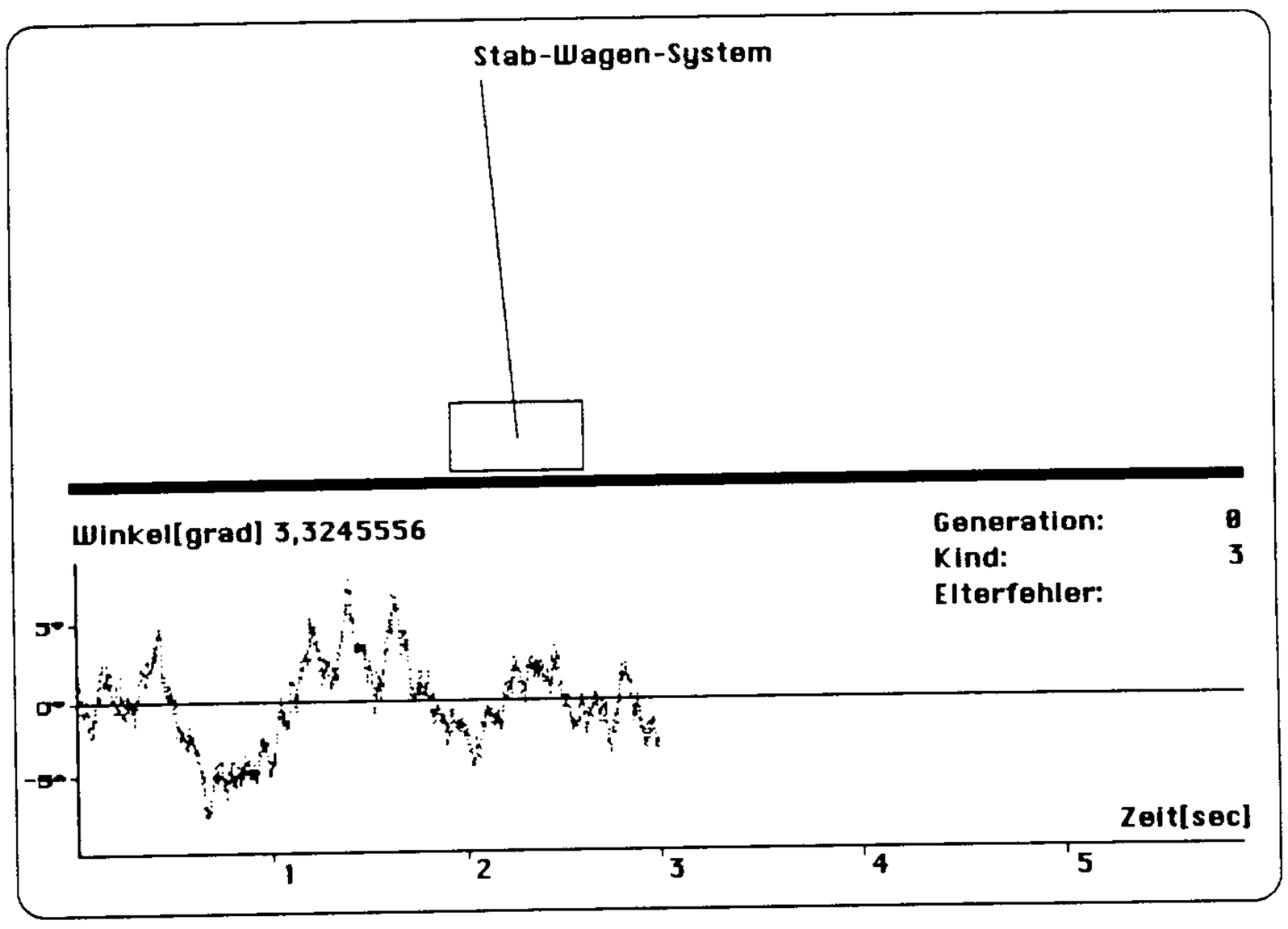

Abbildung 2.14: Darstellung der Simulation auf dem Monitor

Kapitel 3

Fuzzy – Logik

3.1 Allgemeines

Konventionelle Regler basieren auf einer Modellierung mit Differenzialgleichungen und akzeptieren daher nur diskrete (scharfe) Werte. Bei der Fuzzy-Logik wird dagegen mit unscharfen Werten gearbeitet. Diese Werte bestehen aus Symbolen des menschlichen Denkens, die in ihrer Bedeutung unscharf sind. Solche sprachlichen Wertzuweisungen wie z.B. „klein“, „groß“, „kalt“, „sehr kalt“ oder „extrem kalt“ werden bei der Fuzzy-Logik als unscharfe Menge (Fuzzy-Set) bezeichnet.

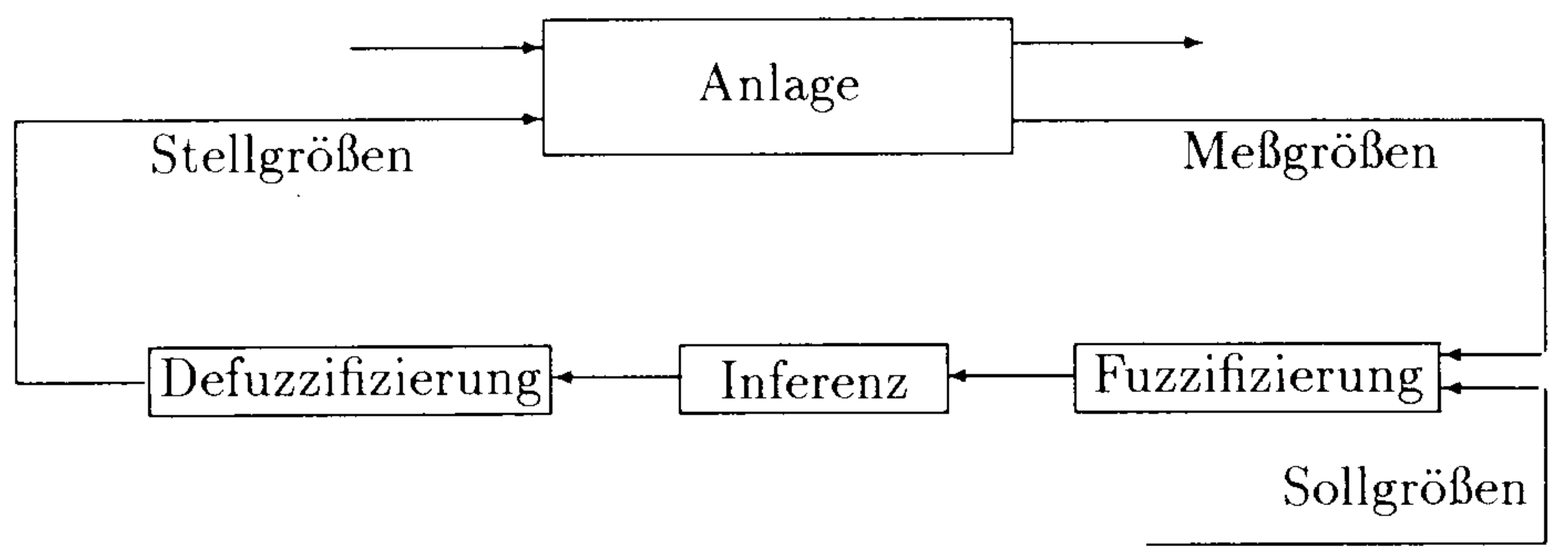

Abbildung 3.1: Grundprinzip eines Fuzzy–Reglers

Durch diese sprachliche Beschreibung der Zustände können die Steuerregeln als Wenn–Dann–Regeln (z.B. WENN Temperatur = hoch DANN Klimaanlage = kühlen) definiert werden. Mehrere dieser Regeln bilden dann den Steueralgorithmus des unscharfen Reglers. Die Grundlage des unscharfen Reglers ist also im Gegensatz zum konventionellen Regler kein exaktes mathematisches Modell. Darin liegt auch die größte Bedeutung der Fuzzy–Logik. Denn bei Prozessen für die entweder gar kein exaktes mathematisches Modell existiert oder aber ein zu komplexes, kann der Steu-

eralgorithmus besser durch linguistische Regeln beschrieben werden.

3.2 Fuzzy–Sets

Eine unscharfe Menge, die durch ein sprachliches Symbol wie z.B. „kalt“ ausgedrückt wird, muß mathematisch exakt definiert werden. Das heißt, es muß festgelegt werden welche Werte zu dieser Menge gehören. Im Gegensatz zur konventionellen Mengenlehre wird die Mitgliedschaft zu einer Menge nicht nur mit „Ja“oder „Nein“ (0 oder 1) beantwortet, sondern die Mitgliedschaft zu einer unscharfen Menge kann einen Wert zwischen 0 und 1 annehmen.

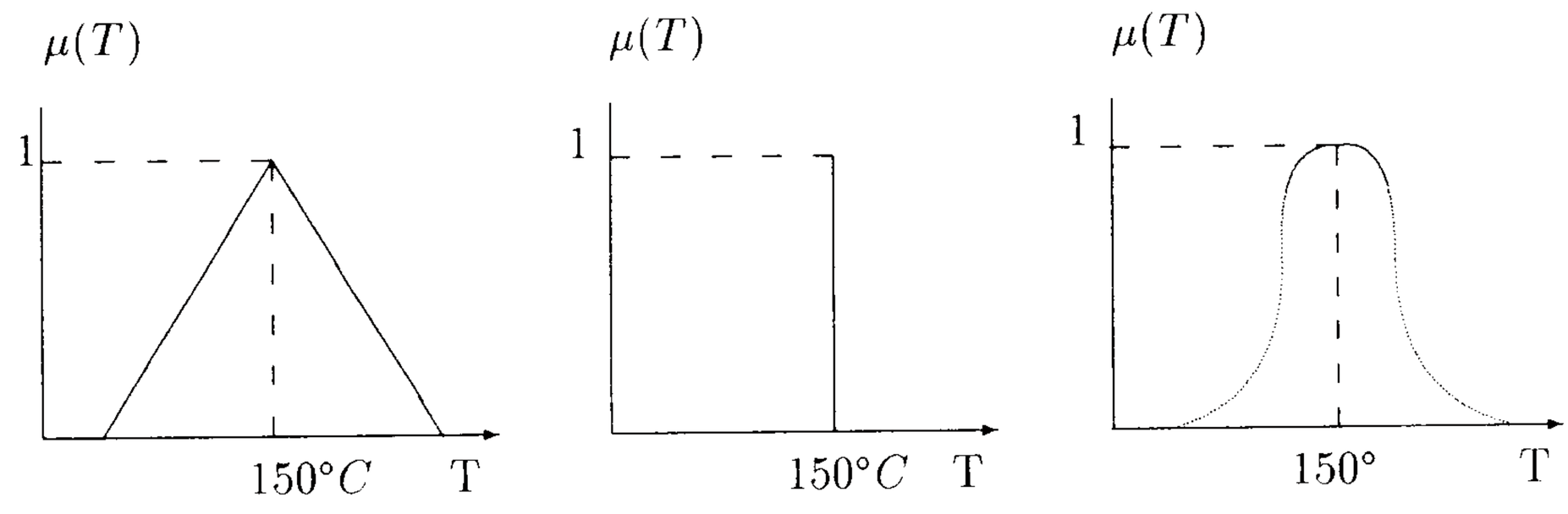

Abbildung 3.2: Beispiele für Mitgliedschaftsfunktionen

Damit für jeden Wert der Grad der Mitgliedschaft eindeutig bestimmt ist, wird der unscharfen Menge A deshalb eine Mitgliedschaftsfunktion $\mu_A(x)$ zugeordnet. Diese Funktion liefert die Mitgliedschaftsgrade aller Elemente x für die unscharfe Menge A. Ein Mitgliedschaftsgrad kann zwischen 0 und 1 liegen .Es gilt:

$$\mu_A : U \rightarrow [0,1]$$
$$A = \{(x, \mu_A(x)) \mid x \in X\}$$

Angrenzende Mitgliedsfunktionen können sich natürlich auch überlappen, da ein Fuzzy-Set in der Regel nicht am Ende eines anderen beginnt. Die Übergänge zwischen angrenzenden Sets sind also meistens ebenfalls unscharf. In Bild (3.3) ist dies anhand der Sets „warm“ und „heiß“ dargestellt. Bei einer Temperatur von $50°C$ ist der Grad der Mitgliedschaft zur Menge „warm“ $\mu(50°C) = 0.8$ und zur Menge „heiß“ $\mu(50°C) = 0.2$. Eine Temperatur von $50°C$ kann so als „ziemlich hoch und kaum noch warm “bezeichnet werden.

In den meisten Fällen ist es sogar sinnvoll, daß sich die Zugehörigkeitsfunktionen überlappen, damit keine Lücken im Reglerverhalten auftreten. Wenn Lücken in der

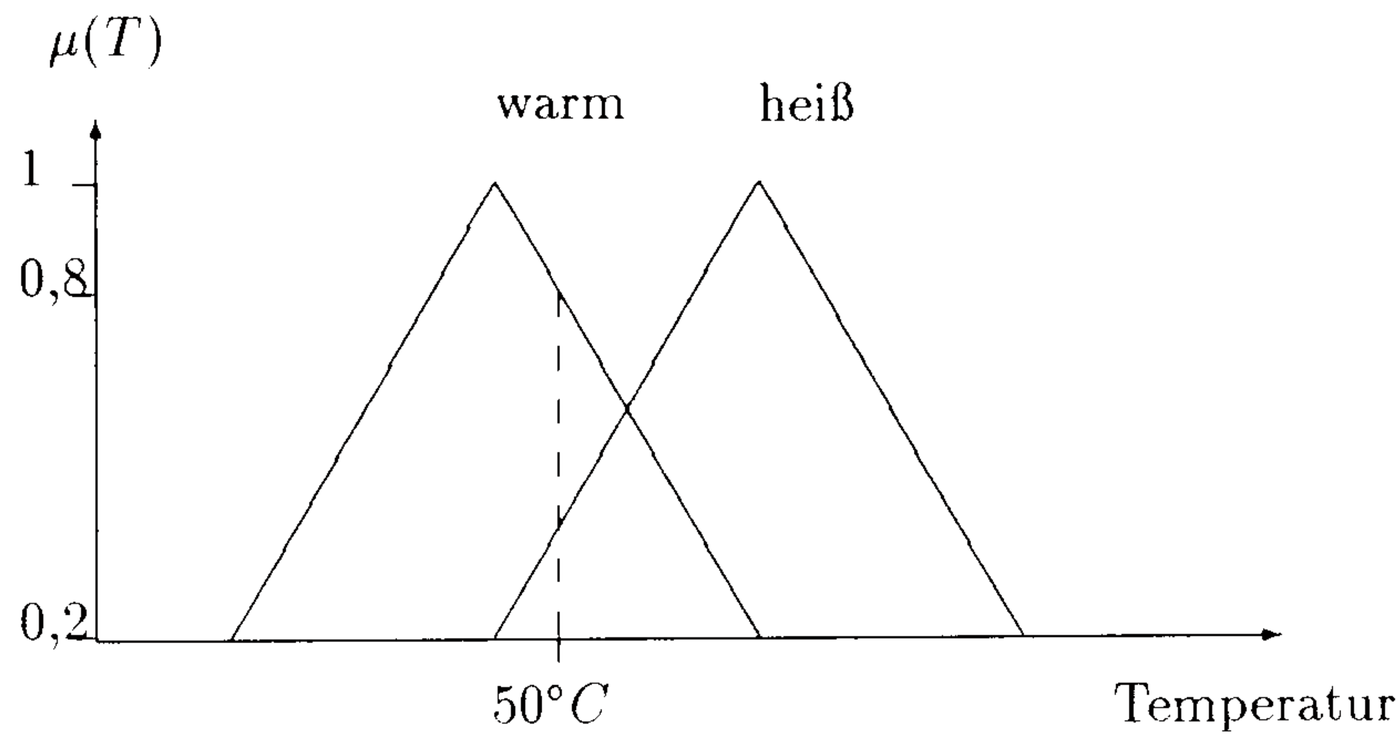

Abbildung 3.3: Überlappungen

Definition der linguistischen Variablen vorhanden sind oder für einen Bereich keine Regeln definiert wurden , kann der Regler keine Enscheidung treffen. Bei einigen Systemen wie dem Stab–Wagen–System , kann die Zeit in der keine Regelung erfolgt ausreichen, daß das System in einen Zustand gerät in dem keine Regelung mehr möglich ist.

3.3 Linguistische Variablen

Werden die unscharfen Mengen einer bestimmten Variablen zugeordnet, so entsteht eine linguistische Variable.

Für die Regelung des Stab–Wagen–Systems werden drei linguistische Variablen verwendet. Es ist eine Variable für die Ausgangsgröße, also die Stellgröße, die angibt mit welcher Kraft und welche Richtung der Wagen bewegt werden soll. In Bild (3.4) ist die linguistische Variable 'Wagen' mit ihren fünf unscharfen Mengen dargestellt. Der x–Wertebereich wird von $-5 * 10^3[\frac{kg*m}{s^2}]$ bis $+5 * 10^3[\frac{kg*m}{s^2}]$ festgelegt.

Die beiden anderen linguistischen Variablen sind Eingangsvariablen. Einmal für den Winkel φ und einmal für die Winkelgeschwindigkeit $\dot{\varphi}$. Auch diese beiden linguistischen Variablen haben je fünf unscharfe Mengen. Der x–Wertebereich der linguistische Variable 'Winkel' geht von $-10[°]$ bis $+10[°]$ und der x–Wertebereich der linguistische Variable 'Omega' geht von $-1[\frac{rad}{s}]$ bis $+1[\frac{rad}{s}]$. Die Eingangsvariablen sind in Bild (3.5) und (3.6) dargestellt. Bei allen drei linguistischen Variablen sind die unscharfen Mengen um den Nullpunkt symetrisch angeordnet.

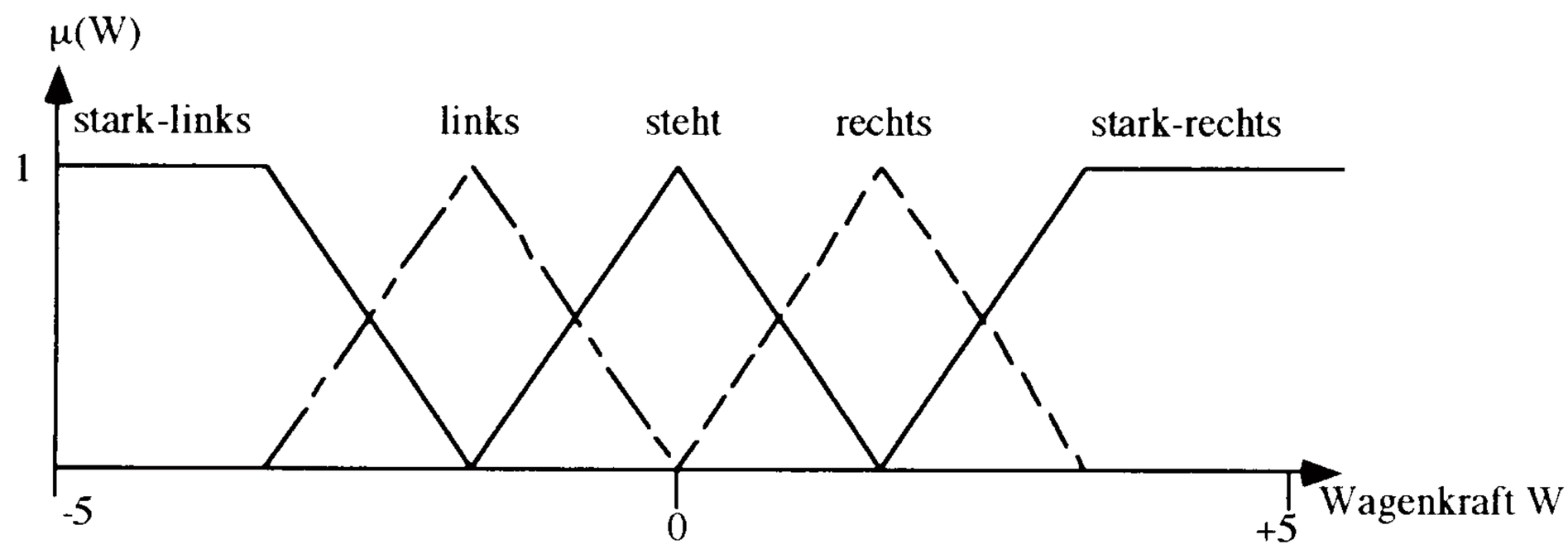

Abbildung 3.4: Linguistische Variable 'Wagen'

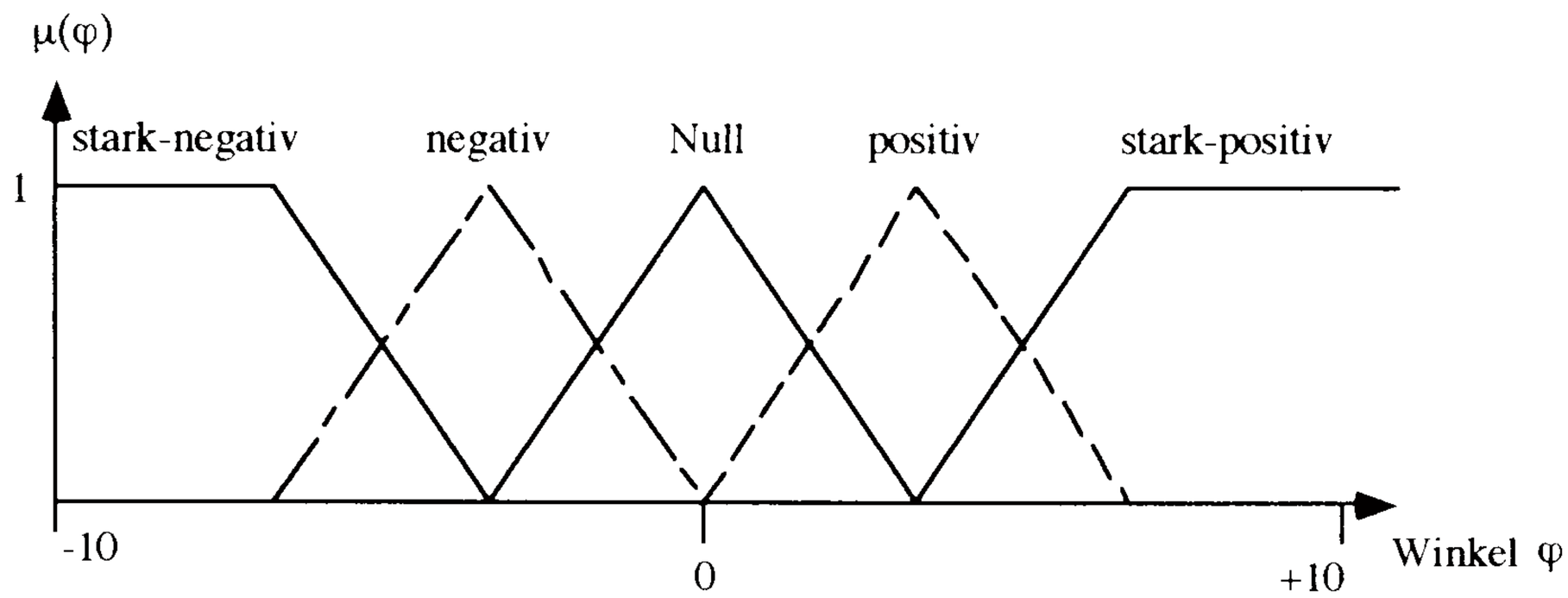

Abbildung 3.5: Linguistische Variable 'Winkel'

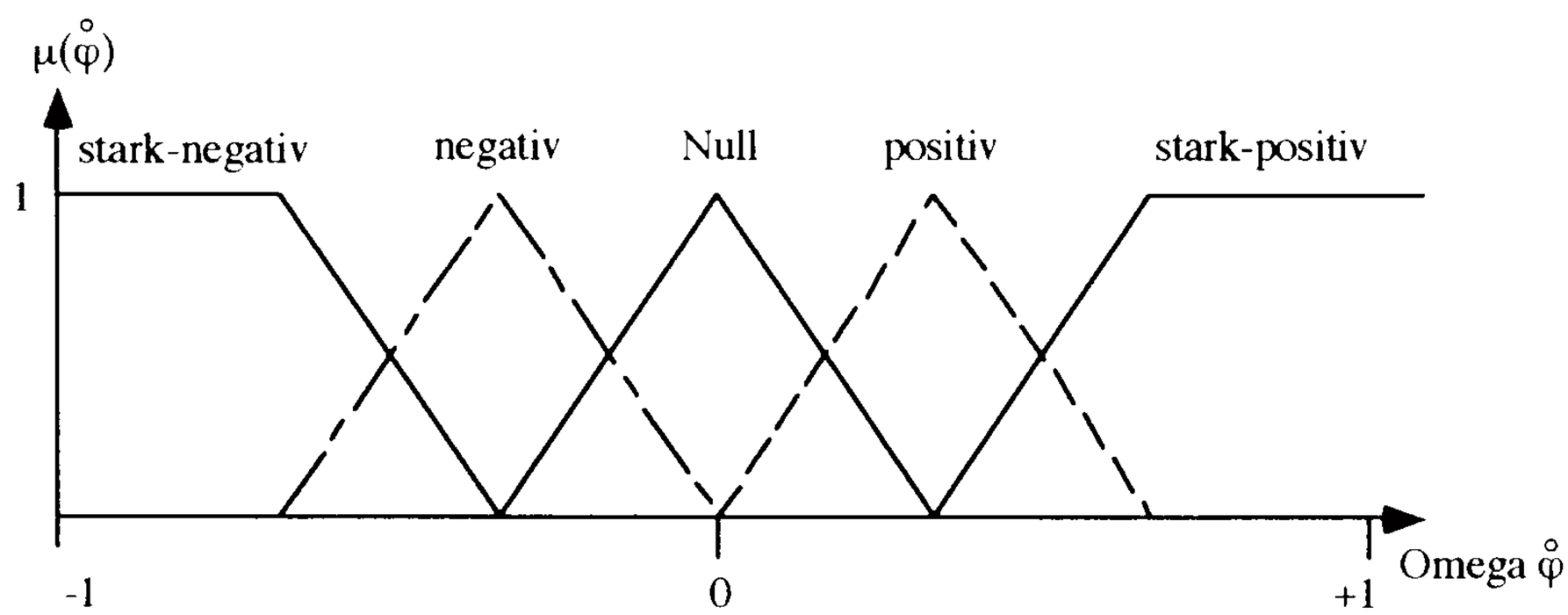

Abbildung 3.6: Linguistische Variable 'Omega'

3.4 Operatoren

Wenn für technische Größen menschliche Begriffe (linguistische Variablen) eingeführt werden, so müssen diese verknüpfbar sein. So können in der Prämisse einer Regel mehrere unscharfe Mengen über Fuzzy–Operatoren verknüpft werden. Diese Fuzzy–Operatoren sind teilweise analog zu den Boolschen Operatoren.

Es folgen nun einige der wichtigsten Mengenoperatoren:

1. Der Minimumoperator für den Durchschnitt von zwei Fuzzy–Sets $C = A \cap B$.

$$\mu_C(x) = min\{\mu_A(x), \mu_B(x)\} \quad x \in X \tag{3.1}$$

Der Minimumoperator entspricht dem logischem UND der Boolschen Algebra.

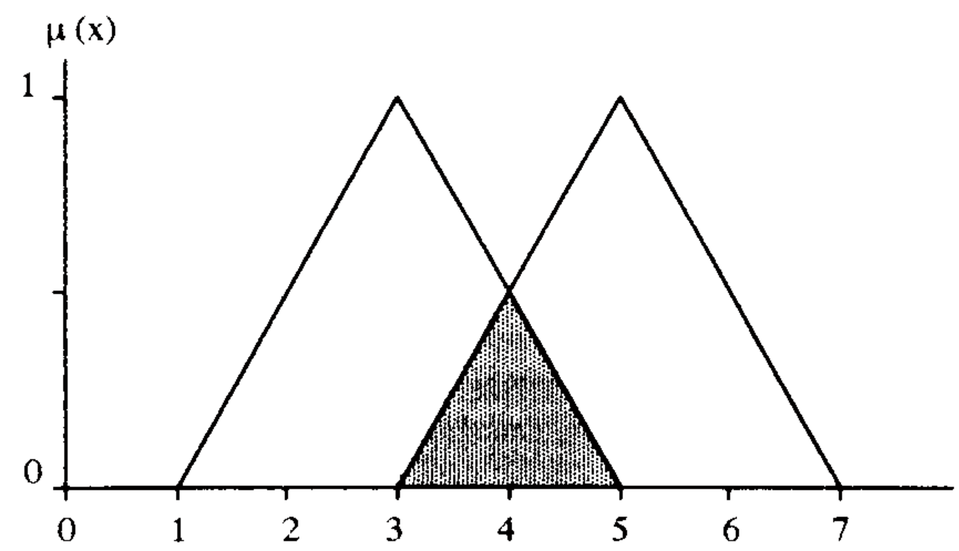

Abbildung 3.7: UND Operator

2. Der Maximumoperator für die Vereinigung von zwei Fuzzy–Sets $C = A \cup B$.

$$\mu_C(x) = max\{\mu_A(x), \mu_B(x)\} \quad x \in X \tag{3.2}$$

Der Maximumoperator entspricht dem logischem ODER der Boolschen Algebra.

3. Der γ –Operator ermöglicht eine beliebige Einstellung zwischen „UND “und „ODER “.

$$\mu_{comp}(x) = (\prod \mu_i(x))^{(1-\gamma)}(1 - \prod(1 - \mu_i(x)))^{\gamma} \quad i = 1 \cdots m, \quad \gamma \in [0,1] \tag{3.3}$$

oder

$$\mu_{comp}(x) = (\mu_A(x)\mu_B(x))^{(1-\gamma)}(1 - (1 - \mu_A(x))(1 - \mu_B(x)))^{\gamma}$$

Das Verhalten des γ –Operators ist in Bild (3.9) dargestellt.

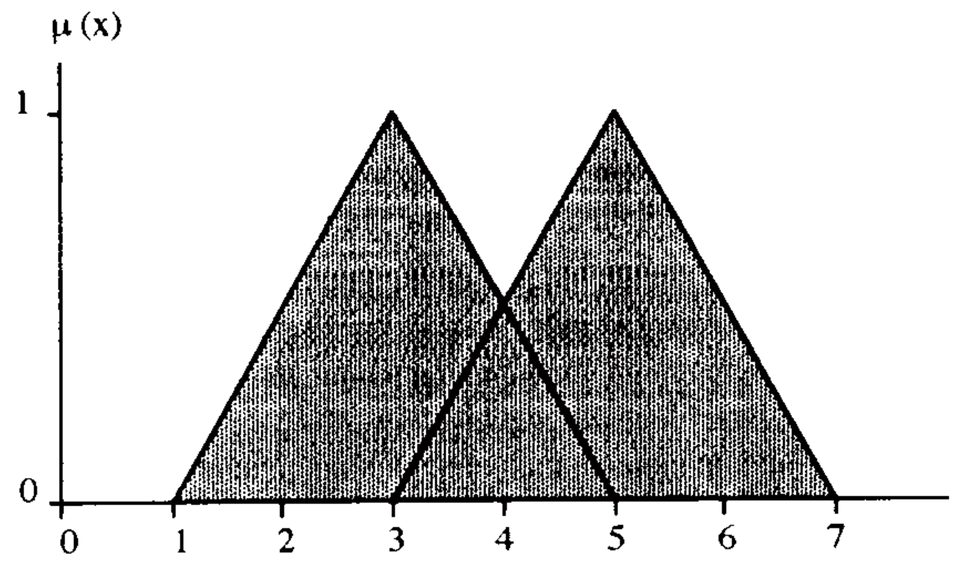

Abbildung 3.8: ODER Operator

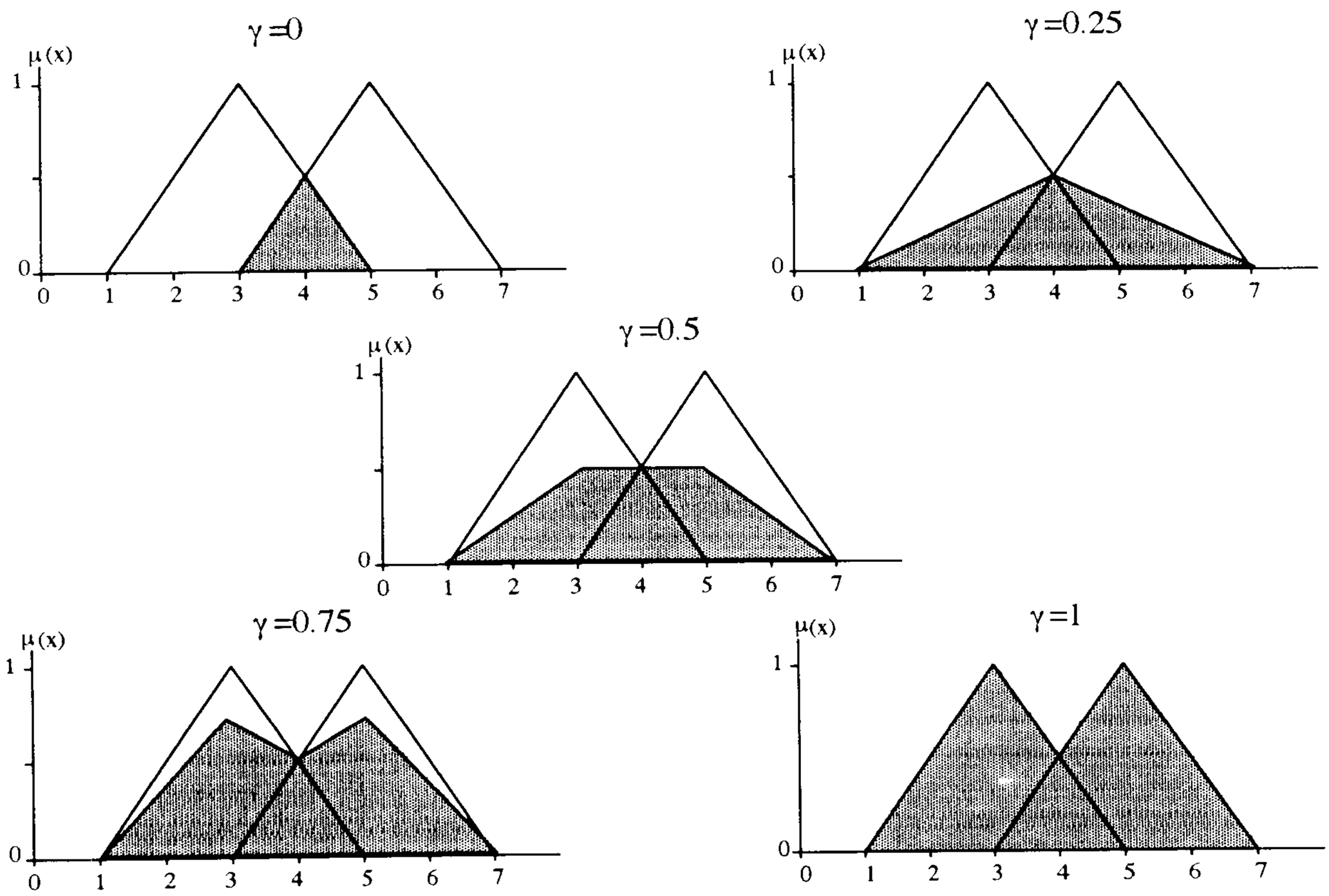

Abbildung 3.9: GAMMA Operator

3.5 Regeln

Ein Grundelement der Verarbeitung unscharfer Informationen mit Fuzzy-Logik ist die Produktionsregel. Sie besteht aus einem WENN-Teil (Vorbedingung) und einem DANN-Teil (Schlußfolgerung). Im WENN-Teil kann nun eine oder mehrere Größen stehen. Bei mehreren Größen müssen diese über Operatoren verknüpft werden.Die Regeln werden nacheinander abgearbeitet und auf ihren Wahrheitsgrad überprüft. Das geschieht in der `Procedure Regelpruefen`.

Zunächst erfolgt für jede linguistische Variable die Fuzzyfizierung. Wobei Fuzzyfizierung die linguistische Interpretation einer technischen Größe bedeutet. Für jeden gemessenen Wert wird der Zugehörigkeitgrad zu dem in der Regel genannten Fuzzy-Set ermittelt. Nun werden die Zugehörigkeitsgrade mit dem Operator der Vorbedingung verknüpft. Die Schlußfolgerung einer Regel ist zum gleichen Grad erfüllt wie ihre Vorbedingung.

Beispiel: Der Winkel beträgt $5,3[°]$ und die Winkelgeschwindigkeit beträgt $-0,5[\frac{rad}{s}]$. Eine der Regeln lautet folgendermaßen:
WENN φ = positiv UND $\dot{\varphi}$ = negativ DANN **Wagen** rechts.
Zuerst wird für die Sets der Eingangsvariablen der Grad der Zugehörigkeit ermittelt

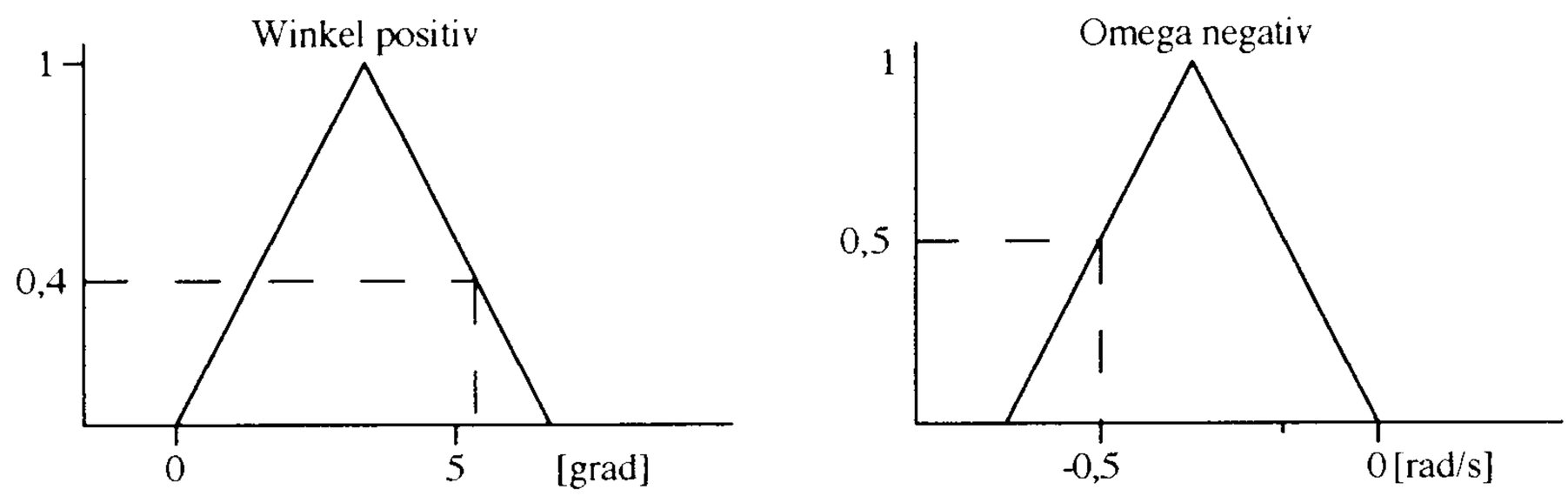

Abbildung 3.10: Prämissenberechnung

(Bild 3.10). Anschließend wird die Prämisse berechnet, was in diesem Beispiel zu folgendem Ergebnis führt: $min\{0,4 : 0,5\} = 0,4$. Die Vorbedingung ist also mit einem grad von 0,4 erfüllt, und damit auch die entsprechende Schlußfolgerung.

Die Regelbasis des Stab-Wagen-Systems besteht aus 49 Regeln, womit alle Kombinationen zwischen den Sets der beiden Eingangsvariablen abgedeckt sind. Alle Regeln werden geprüft und anschließend mit der Max-Prod Inferenz weiterverarbeitet. In Tabelle (3.1) sind die Regeln für das Stab-Wagen-System aufgeführt.

WENN $\varphi =$	sehr-negativ	UND $\dot{\varphi} =$	sehr-negativ	DANN	**Wagen**	stark-links
WENN $\varphi =$	sehr-negativ	UND $\dot{\varphi} =$	negativ	DANN	**Wagen**	stark-links
WENN $\varphi =$	sehr-negativ	UND $\dot{\varphi} =$	null	DANN	**Wagen**	stark-links
WENN $\varphi =$	sehr-negativ	UND $\dot{\varphi} =$	positiv	DANN	**Wagen**	links
WENN $\varphi =$	sehr-negativ	UND $\dot{\varphi} =$	sehr-positiv	DANN	**Wagen**	links
WENN $\varphi =$	negativ	UND $\dot{\varphi} =$	sehr-negativ	DANN	**Wagen**	stark-links
WENN $\varphi =$	negativ	UND $\dot{\varphi} =$	negativ	DANN	**Wagen**	links
WENN $\varphi =$	negativ	UND $\dot{\varphi} =$	null	DANN	**Wagen**	links
WENN $\varphi =$	negativ	UND $\dot{\varphi} =$	positiv	DANN	**Wagen**	links
WENN $\varphi =$	negativ	UND $\dot{\varphi} =$	sehr-positiv	DANN	**Wagen**	links
WENN $\varphi =$	null	UND $\dot{\varphi} =$	sehr-negativ	DANN	**Wagen**	stark-links
WENN $\varphi =$	null	UND $\dot{\varphi} =$	negativ	DANN	**Wagen**	links
WENN $\varphi =$	null	UND $\dot{\varphi} =$	null	DANN	**Wagen**	steht
WENN $\varphi =$	null	UND $\dot{\varphi} =$	positiv	DANN	**Wagen**	rechts
WENN $\varphi =$	null	UND $\dot{\varphi} =$	sehr-positiv	DANN	**Wagen**	stark-rechts
WENN $\varphi =$	positiv	UND $\dot{\varphi} =$	sehr-negativ	DANN	**Wagen**	rechts
WENN $\varphi =$	positiv	UND $\dot{\varphi} =$	negativ	DANN	**Wagen**	rechts
WENN $\varphi =$	positiv	UND $\dot{\varphi} =$	null	DANN	**Wagen**	rechts
WENN $\varphi =$	positiv	UND $\dot{\varphi} =$	positiv	DANN	**Wagen**	rechts
WENN $\varphi =$	positiv	UND $\dot{\varphi} =$	sehr-positiv	DANN	**Wagen**	stark-rechts
WENN $\varphi =$	sehr-positiv	UND $\dot{\varphi} =$	sehr-negativ	DANN	**Wagen**	rechts
WENN $\varphi =$	sehr-positiv	UND $\dot{\varphi} =$	negativ	DANN	**Wagen**	rechts
WENN $\varphi =$	sehr-positiv	UND $\dot{\varphi} =$	null	DANN	**Wagen**	stark-rechts
WENN $\varphi =$	sehr-positiv	UND $\dot{\varphi} =$	positiv	DANN	**Wagen**	stark-rechts
WENN $\varphi =$	sehr-positiv	UND $\dot{\varphi} =$	sehr-positiv	DANN	**Wagen**	stark-rechts

Tabelle 3.1: Regeln des Stab-Wagen-Systems

3.6 Inferenz

Die Schlußfolgerung einer Regel wird durch die MAX-MIN- oder die MAX-PROD-Inferenz weiterverarbeitet. Die MAX-MIN-Inferenz begrenzt die Sets der linguistischen Variable „Wagen "jeweils auf den Wahrheitsgehalt der Vorbedingung. Anschließend werden die unscharfen Mengen zu einer einzigen zusammengefaßt.

Bei der MAX-PROD-Inferenz wird das Produkt aus Wahrheitsgrad der Vorbedingung und der unscharfen Menge der Schlußfolgerung gebildet (Bild 3.11). Die so erhaltenen Sets werden ebenfalls zu einer einzigen zusammengefaßt (Bild 3.12). Bei dem Stab-Wagen-System wird die MAX-PROD-Inferenz verwendet.

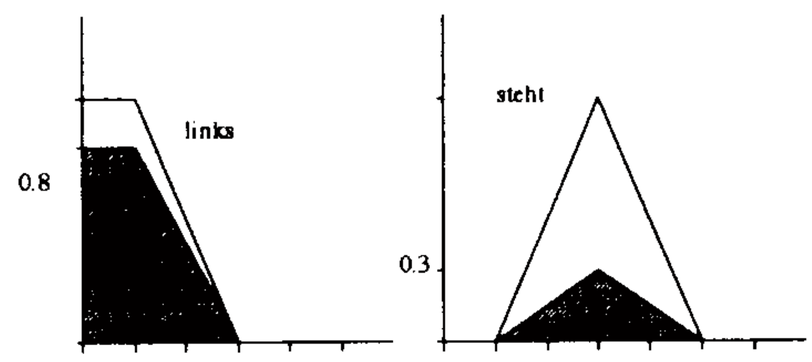

Abbildung 3.11: Max-Prod Inferenz

3.6.1 Defuzzifizierung

Für eine Regelung ist es nun notwendig die unscharfe Menge, die sich aus der Inferenz ergibt, in eine diskrete Stellgröße umzuwandeln (defuzzifizieren). Die am häufigsten verwendete Methode ist das Schwerpunktverfahren. Dabei wird nach Gleichung (3.4) der Schwerpunkt der resultierenden unscharfen Menge berechnet (Bild 3.12):

$$x_s = \frac{\sum_{i=1}^{n} x_i \mu(x)}{\sum_{i=1}^{n} \mu(x)} \tag{3.4}$$

Die X-Koordinate des Schwerpunktes entspricht der Stellgröße.

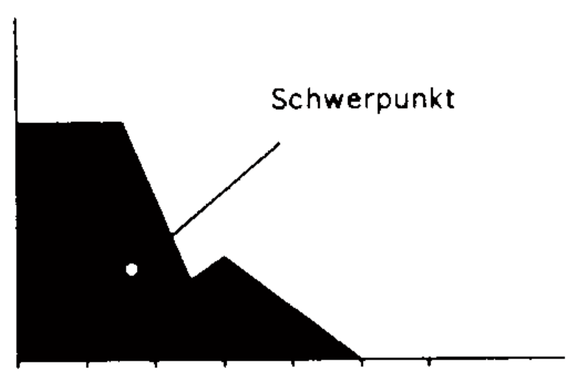

Abbildung 3.12: Defuzzifizierung

3.7 Struktogramme der Fuzzy–Logik

Alle Proceduren, die für die Fuzzifizierung und Defuzzifizierung notwendig sind, wurden in die `Unit Fuzzy-E` implementiert. Vom Hauptprogramm wird die `Proc. Regelprüfen` aufgerufen.

Procedure Regelprüfen — Alle Regeln werden nacheinander abgearbeitet

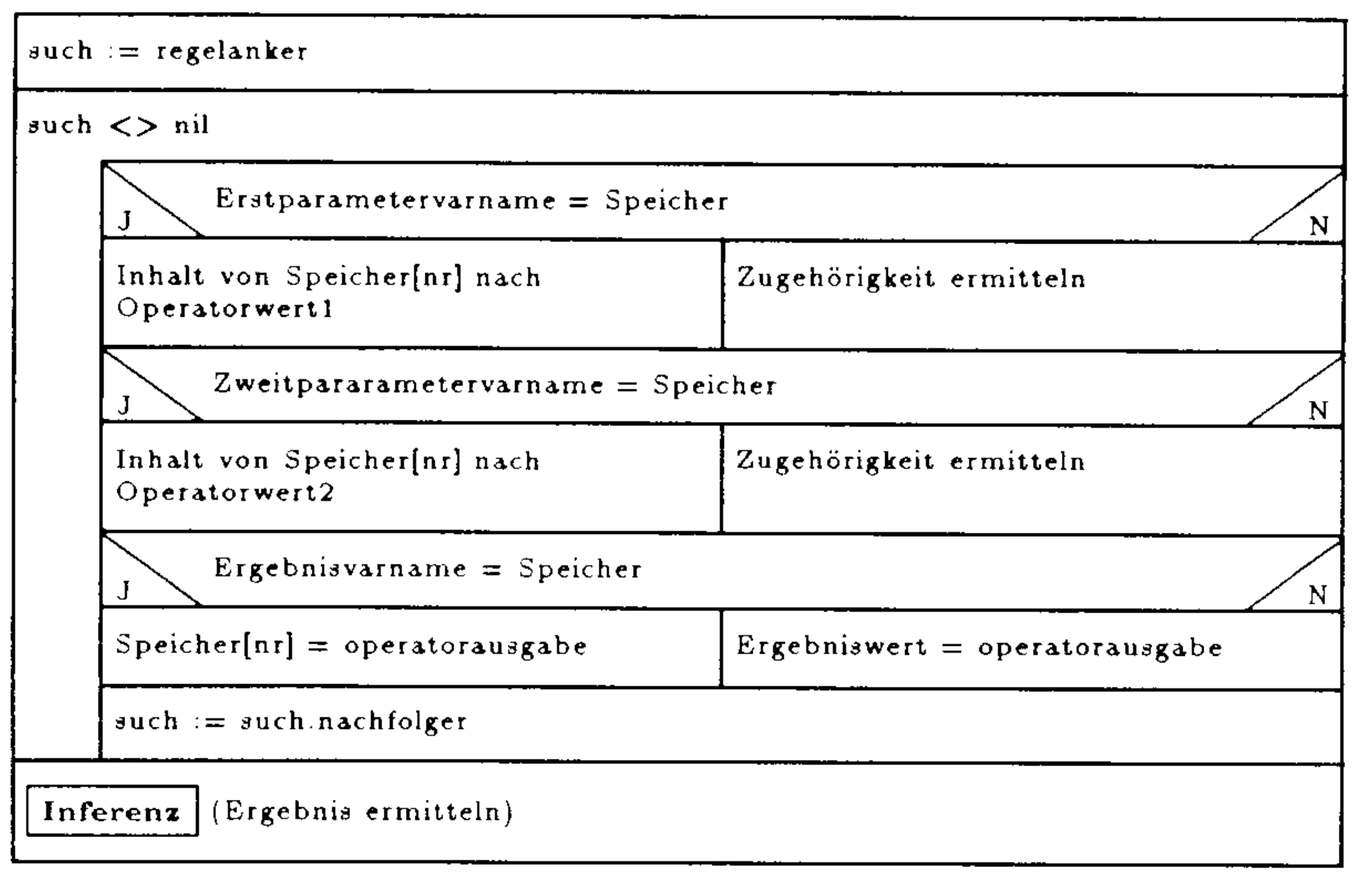

Abbildung 3.13: Struktogramm Regelprüfen

Diese Procedure prüft alle Regeln auf den Wahrheitsgehalt ihrer Schlußfolgerungen. Anschließend wird die `Procedure Inferenz` aufgerufen.

Diese führt für alle Regeln die MAX–PROD–Inferenz durch, mit der `Procedure MaxProd`, indem in ein Array der Breite `f_breite` die Fläche der resultierenden unscharfen Menge gespeichert wird. Nach Gleichung (3.4) wird nun der Schwerpunkt berechnet und in die Variable `EinAus[Nummer]` geschrieben. Je größer die Konstante `f_breite` gewählt wird, desto genauer wird die Schwerpunktberechnung durchgeführt (Mindestwert = 11).

In der `Procedure MaxProd` wird mit der entsprechenden Auflösung, für jeden x–Wert die Zugehörigkeit mit dem Wahrheitsgrad der Schlußfolgerung multipliziert. Außerdem wird die aus der Regelbasis resultierende unscharfe Menge erzeugt.

Procedure Inferenz — Unscharfes Schließen

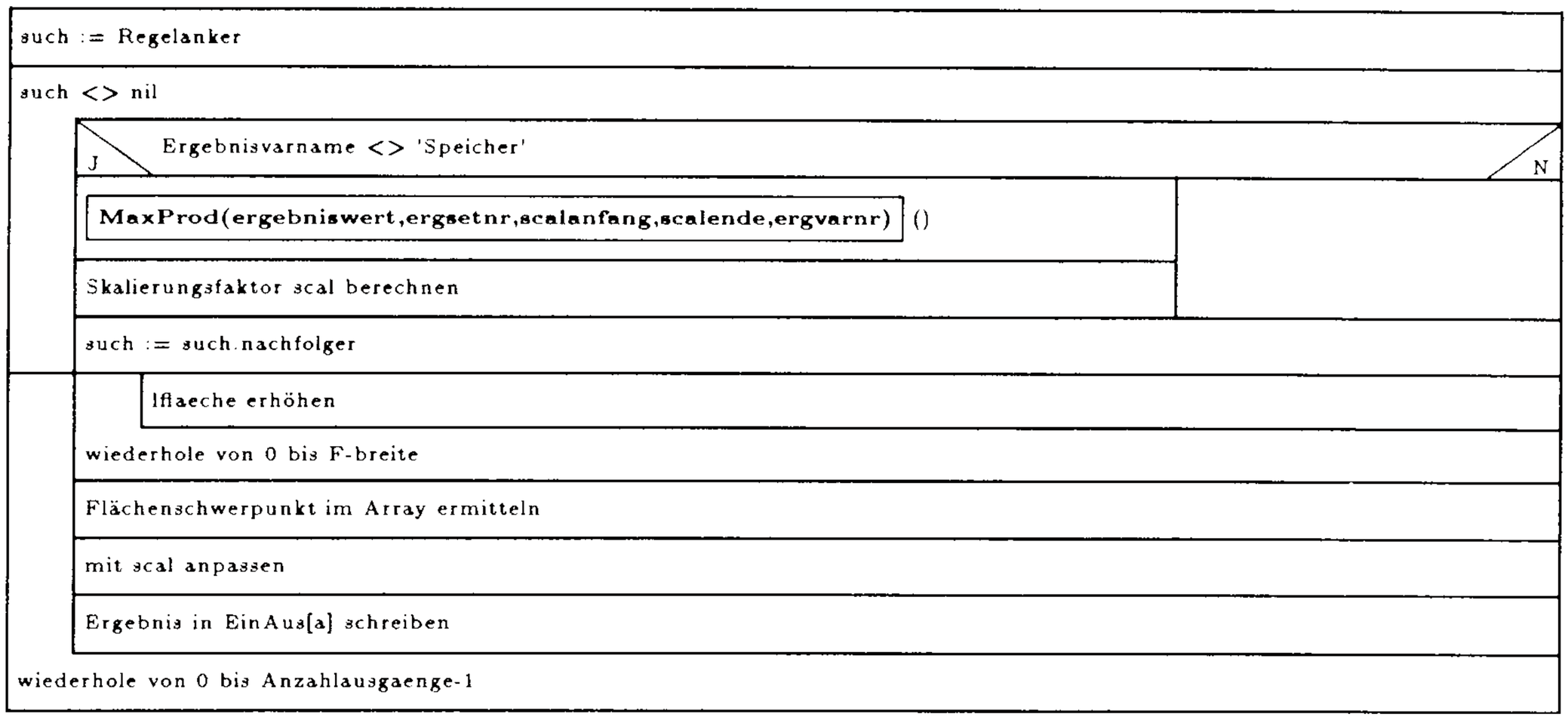

Abbildung 3.14: Struktogramm Inferenz

Pocedure MaxProd — Erzeugt die Ergebnisfläche eines Sets.

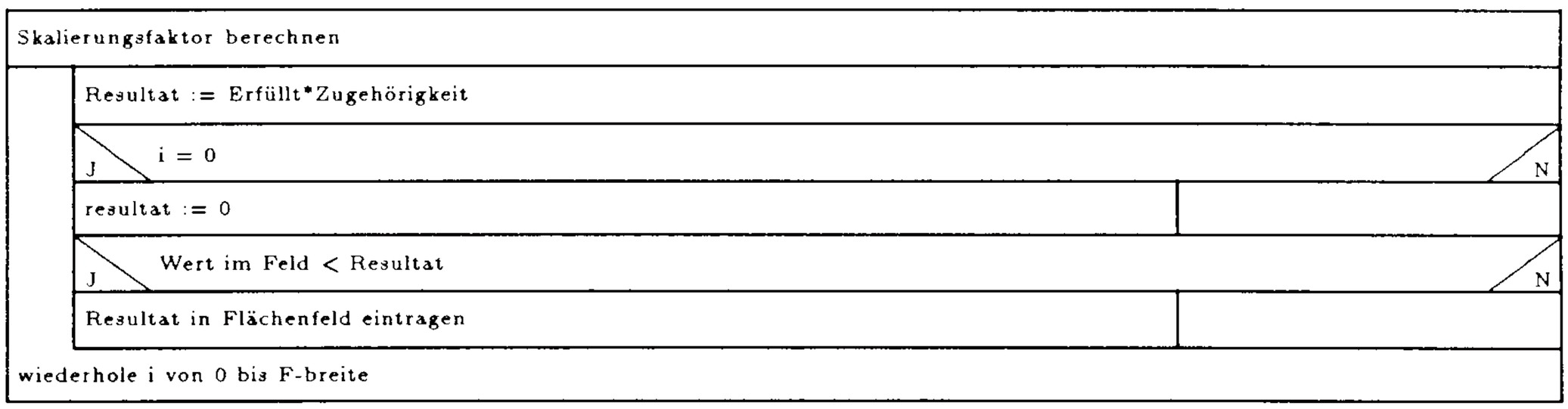

Abbildung 3.15: Struktogramm MaxProd

Function welchezugehoerigkeit — Berechnet die Zugehörigkeit eines Eingangs oder Ausgangswertes

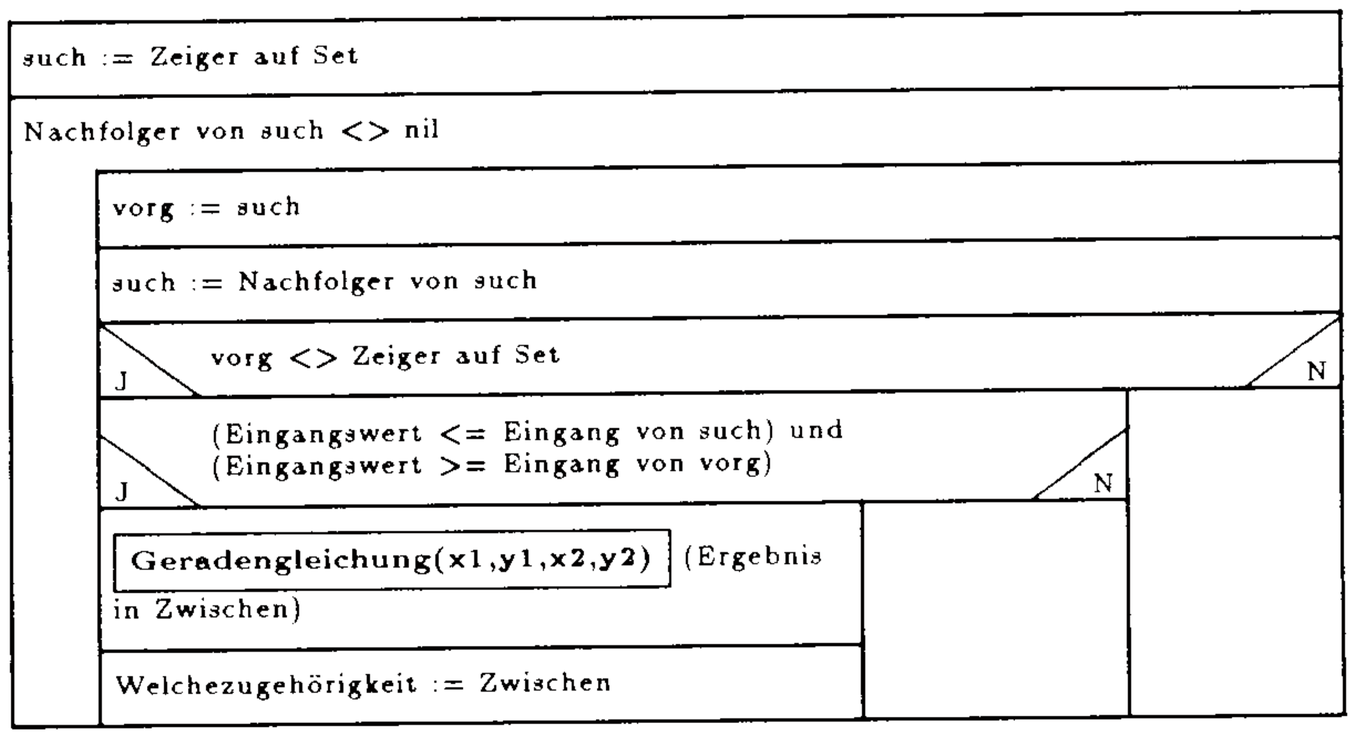

Abbildung 3.16: Struktogramm Welchezugehörigkeit

Kapitel 4

Evolutionsstrategie

4.1 Einführung

Für die Festlegung der unscharfen Mengen und der Regelbasis gibt es noch kein geeignetes Verfahren. Im Normalfall wird dies vom Entwickler empirisch festgelegt und anschließend durch Beobachtung des Regelverhaltens nachkorrigiert. Dieses Vorgehen ist jedoch sehr zeitaufwendig, und vor allen Dingen durch die hohe Anzahl der Parameter kaum überschaubar.

Eine mögliche Lösung des Problems ist die Optimierung mittels Evolutionsstrategien. Entsprechende Grundlagen sind in [8] [10] u.a. beschrieben. Die Optimierung mit Evolutionsstrategien ist zwar nicht perfekt, bietet aber für ein derart komplexes System eine annähernd beste Lösung. Daß heißt die Entwicklungsdynamik läßt das Erreichen einer zweifelsfrei besten Lösung nicht zu, sondern kommt nur in die Nähe des Optimums.

Die Methode der Evolutionsstrategie ähnelt dem oben beschriebenen Verfahren der Beobachtung und Nachkorrektur durch den Entwickler. Nur werden bei der Evolutionsstrategie die Parameter um einen zufälligen Wert abgeändert und nicht empirisch nachgebessert.

4.2 (1+5)–gliedrige Evolutionsstrategie

Die wichtigsten Grundelemente der Evolution sind die Mutation und die Selektion. Bei dem Stab–Wagen–System werden die Fuzzy-Sets der linguistischen Variablen durch Mutation verändert. Dabei werden die Eckpunkte, die die unscharfe Menge begrenzen, um einen Zufallswert verschoben (Beispiel in Bild 4.1).

Bevor die Sets verändert werden, muß der Variablensatz eines Elters dupliziert werden. Die Duplikate werden als Nachkommen bezeichnet. Sobald dies geschehen ist erfolgt die Mutation der Sets. Da die linguistischen Variablen in einen positiven und

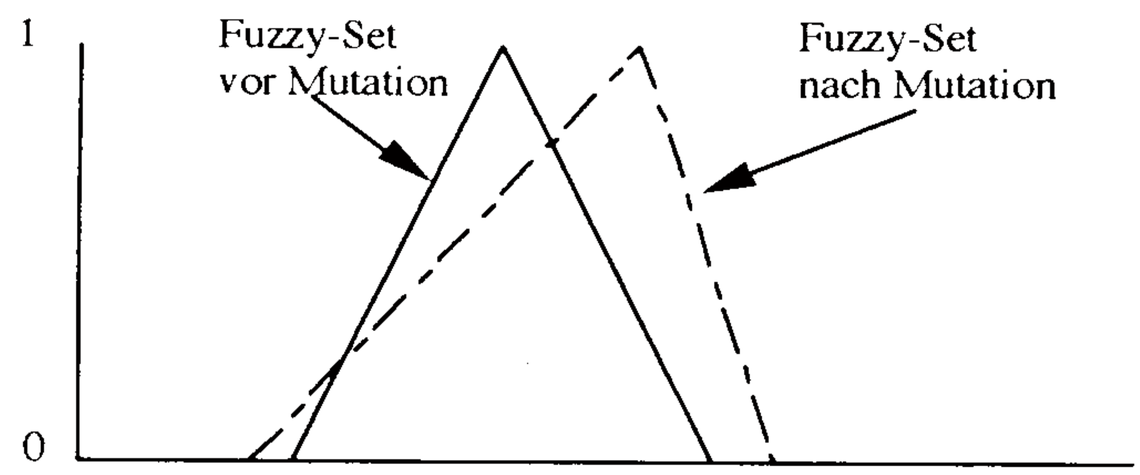

Abbildung 4.1: Mutation eines Fuzzy–Sets

einen gleich großen negativen Bereich aufgeteilt sind, erfolgt die Mutation symetrisch zum Nullpunkt.

Nun erfolgt für alle Nachkommen ein Simulationslauf über 3 Sekunden. Dabei wird jeweils der quadratische Regelfehler ermittelt.

$$QuadratischerRegelfehler = \sum_{t=0}^{n} \varphi^2(t) \tag{4.1}$$

Dieser Regelfehler dient der Selektion als Gütekriterium. Bei der Selektion wird das Gütekriterium der 5 erzeugten Nachkommen und dem Elter verglichen [1]. Zum neuen Elter wird der Nachkomme oder der alte Elter mit dem kleinsten Regelfehler.

Damit die Simulation realistischer ist, wird zu jedem $\varphi(t)$ eine Störung addiert. Da ein Vergleich des Gütekriteriums nur sinnvoll ist wenn die Simulation bei allen Nachkommen unter gleichen Bedingungen abläuft, müssen die Störungen bei jedem Simulationslauf gleich sein. Die `Störung(t)` wird deshalb immer aus einer Tabelle ausgelesen.

4.3 Die Programmteile der ES

Alle Proceduren der Evolutionsstrategie sind in der `Unit Fuzzy-E` implementiert. Diese Unit enthält die Programmteile des Fuzzy-Editors [16]. Durch diese Zusammenfassung können für die Manipulation der linguistischen Variablen entsprechende Funktionen genutzt werden (z.B. entfernen und hinzufügen von Stützpunkten der Fuzzy–Sets).

Nachdem bei allen Nachkommen die Simulation erfolgte, wird durch Vergleichsoperationen der Nachkommen mit dem kleinsten Regelfehler ermittelt. Ist dieser besser wie der Elter, so wird er zum neuen Elter. Im anderen Fall wird der alte Elter weiter verwendet. Anschließend wird die `Procedure Optimieren` aufgerufen.

Jetzt wird der Variablensatz des Elter dupliziert. Die Fuzzy–Sets der 5 Nachkommen werden nun durch Mutation verändert. Dies geschieht in der `Function Mutation`.

[1] (1 Elter + 5 Nachkommen = (1+5)–gliedrige Evolutionsstrategie

Procedure Optimieren() — Dupliziert Elter und sorgt für die Mutation

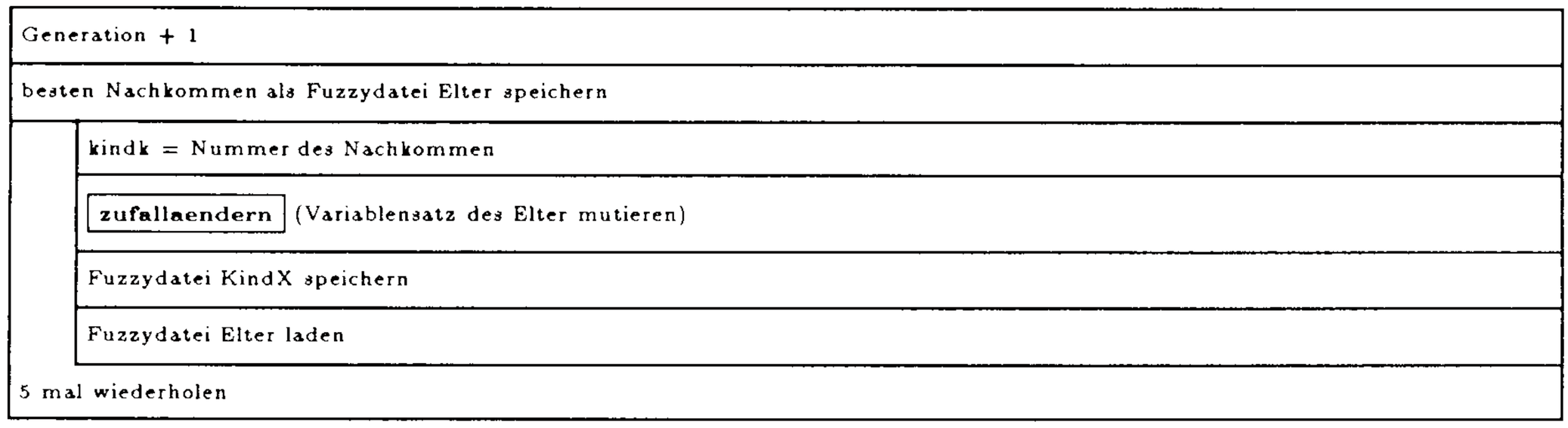

Abbildung 4.2: Struktogramm Optimieren

Procedure Naechsteskind — Selektiert den besten Nachkommen aus.

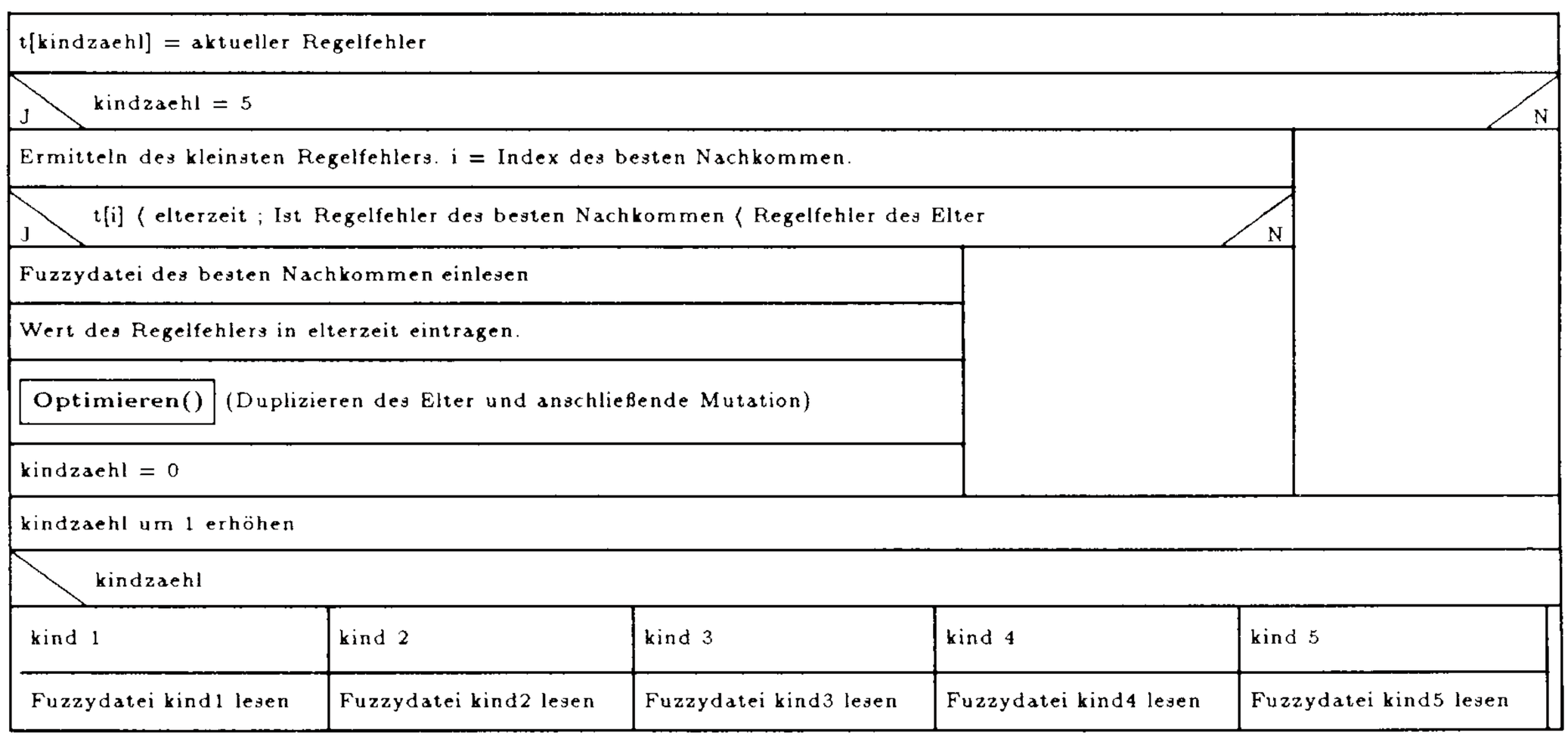

Abbildung 4.3: Struktogramm Naechsteskind

Wie schon beschrieben, werden die Eckpunkte der Sets um einen Zufallswert verschoben. In der `Function Gaussrandom` wird nach der aus [10] entnommenen Gleichung 4.2 ein normalverteilter Zufallswert erzeugt.

$$Gaussrandom = \sqrt{ln\frac{1}{Random^2}}\, sin(2\,\pi\, Random) \tag{4.2}$$

Genau wie in der Natur erfolgen also häufig kleine und selten große Mutationen. Der von der `Function Gaussrandom` erzeugte Wert wird danach noch an den Wertebereich der linguistischen Variablen angepaßt.

In der `Procedure Zufallaendern` werden nun, mit Proceduren des Fuzzy–Editors [16], die Eckpunkte der Sets entfernt und durch die mutierten Eckpunkte ersetzt. Dies geschieht, wie in Bild 4.6, 4.7 und 4.8 zu sehen ist, symetrisch zum Nullpunkt der linguistischen Variablen.

Wird die Optimierung abgebrochen, so muß die Fuzzy–Datei „Elter“eingeladen werden. Danach kann sie unter dem gewünschten Namen gesichert werden. Die Frage, ob die Regelmatrix erzeugt werden soll, ist nur mit „Ja“zu beantworten wenn die Optimierung vollständig abgeschlossen wurde. Die Rechenzeit beträgt dann noch ca. 4 Stunden.

4.4 Optimierung der Variablen

Die bei der Evolutionsstrategie vorhandene Regelbasis in Tabelle 4.1 ist empirisch festgelegt worden.

		φ						
		ssn	*sn*	*n*	*mitte*	*p*	*sp*	*ssp*
$\dot{\varphi}$	*ssn*	ssl	ssl	ssl	ssl	sr	sr	sr
	sn	ssl	ssl	sl	sl	r	sr	sr
	n	ssl	sl	l	l	r	r	sr
	null	ssl	sl	l	null	r	sr	ssr
	p	sl	l	l	r	r	sr	ssr
	sp	sl	sl	l	sr	sr	ssr	ssr
	ssp	sl	sl	sl	ssr	ssr	ssr	ssr

Tabelle 4.1: Regelbasis der Stellgröße für 7 Sets pro Variable

Da die Sets vor der Optimierung automatisch erzeugt werden, ist es am einfachsten die Sets gleichmäßig über den ganzen Wertebereich der linguistischen Variablen zu verteilen. Die drei gleich aussehenden linguistischen Variablen sind zusammen in Bild 4.5 dargestellt.

Nach 55 Generationen sehen die drei linguistischen Variablen aus, wie es in den Bildern 4.6, 4.7 und 4.8 dargestellt ist. Das erzielte Ergebnis ist natürlich nur ein

Procedure zufallaendern — mutiert die Fuzzy-Sets

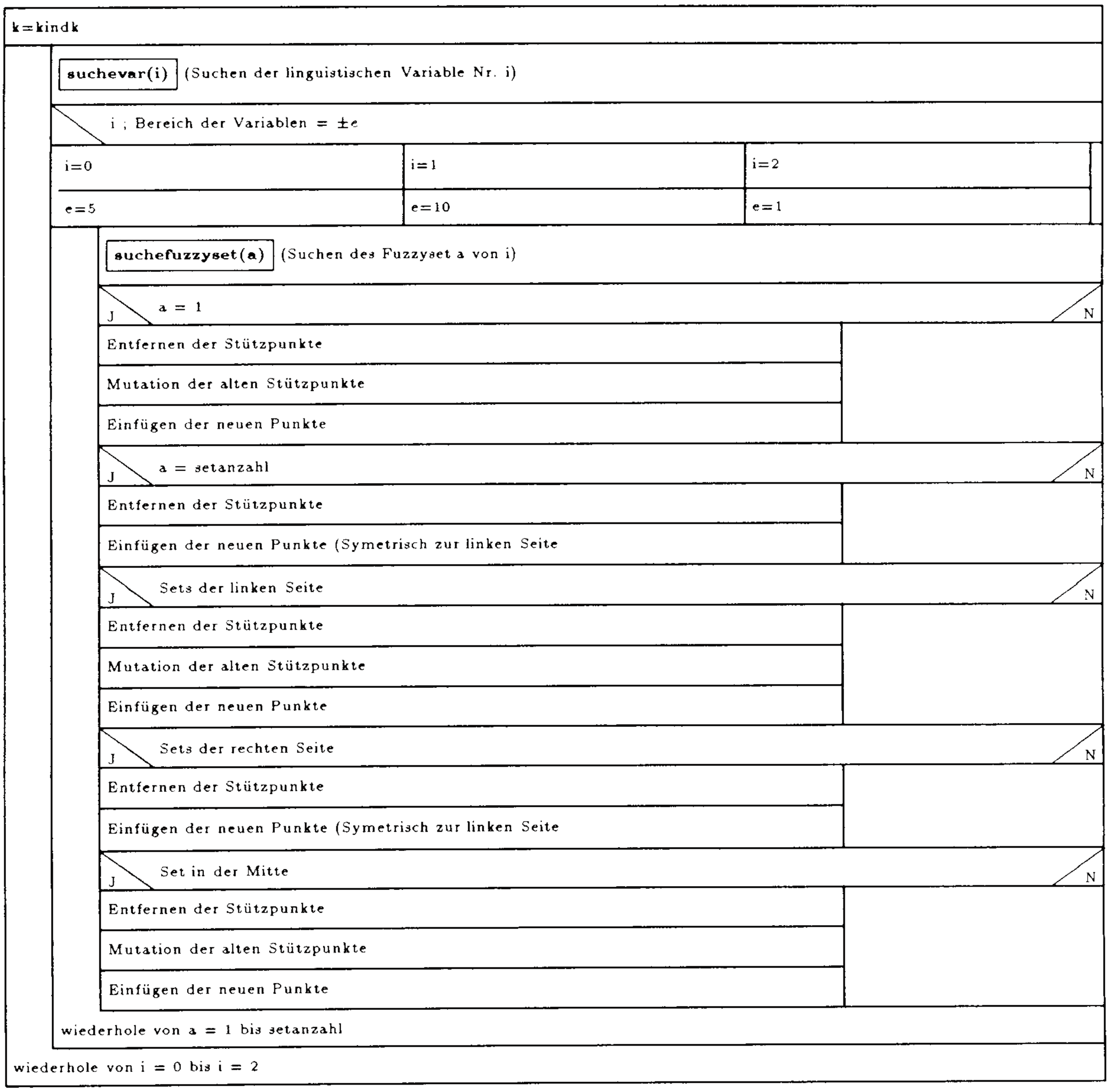

Abbildung 4.4: Struktogramm zufallaendern

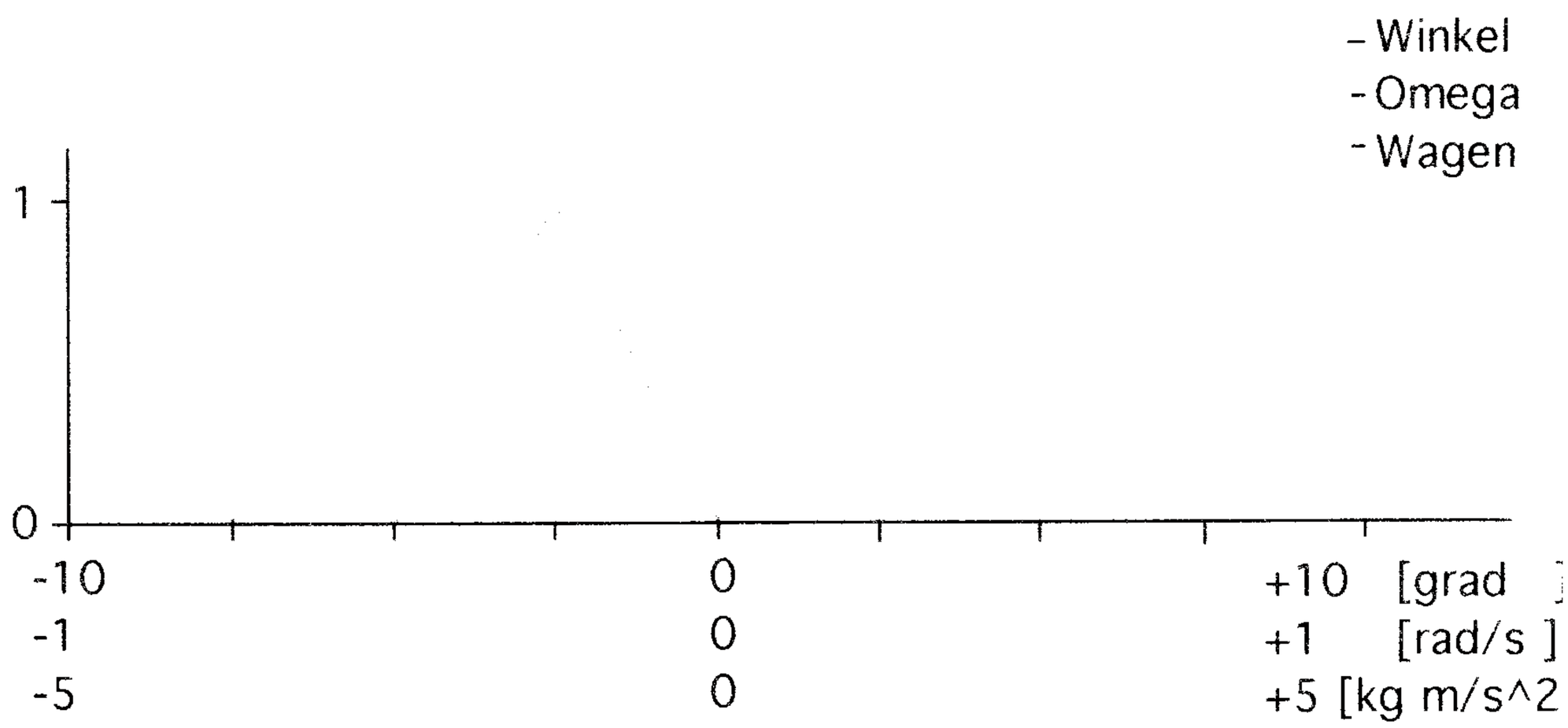

Abbildung 4.5: Linguistische Variablen vor Optimierung

Suboptimum. Die Verteilung der Sets bei der Variablen „Winkel"entspricht etwa dem erwartetem Ergebnis. Die Sets konzentrieren sich zur Mitte der Variablen, also um einen Winkel von null Grad. Diese Anordnung bedeutet eine Reaktion schon bei kleinen Winkeln. Daß bei den anderen Variablen nicht eine ähnliche Anordnug erzeugt wurde, kann daran liegen, daß das Ergebnis nur ein Suboptimum ist.

Durch die Optimierung mit der Evolutionsstrategie kann aber auch ein unerwartetes Ergebnis entstehen. Wenn nicht alle Bedingungen, unter denen dieses Ergebnis zustande kam, bekannt sind oder nicht im richtigen Zusammenhang gesehen werden, ist für das Ergebnis keine plausible Erklärung möglich.

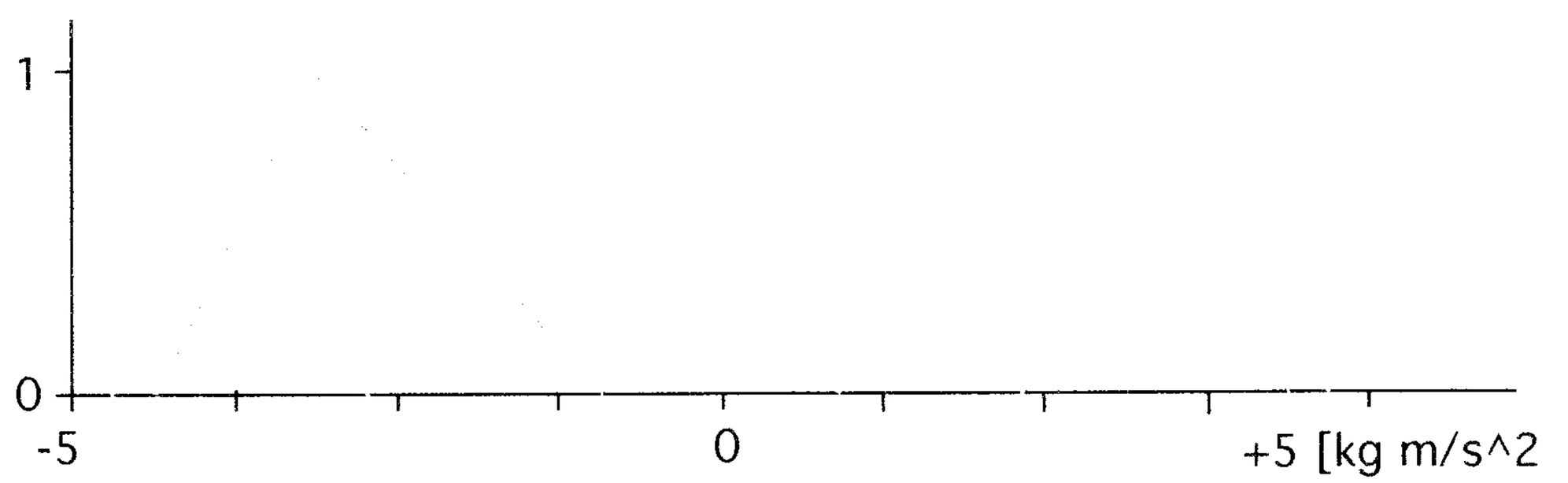

Abbildung 4.6: „Wagen"nach Optimierung

In den Bildern 4.9 und 4.10 sind die Zeitverläufe des Winkels jeweils vor und nach der Optimierung dargestellt. Die Zeitverläufe sind sich sehr ähnlich, da die

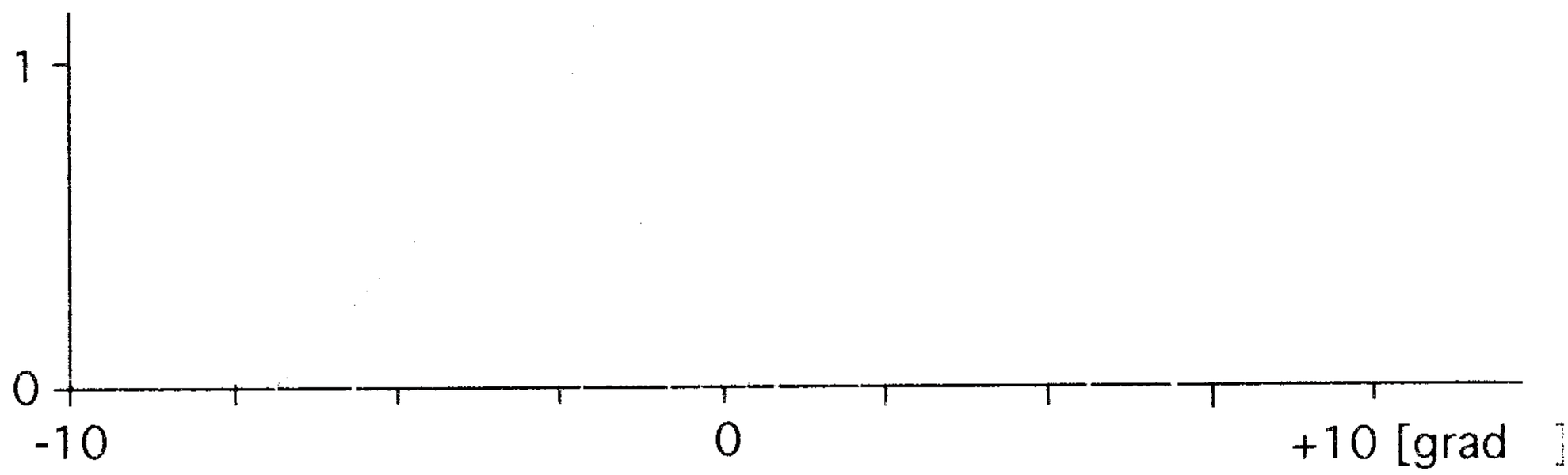

Abbildung 4.7: „Winkel" nach Optimierung

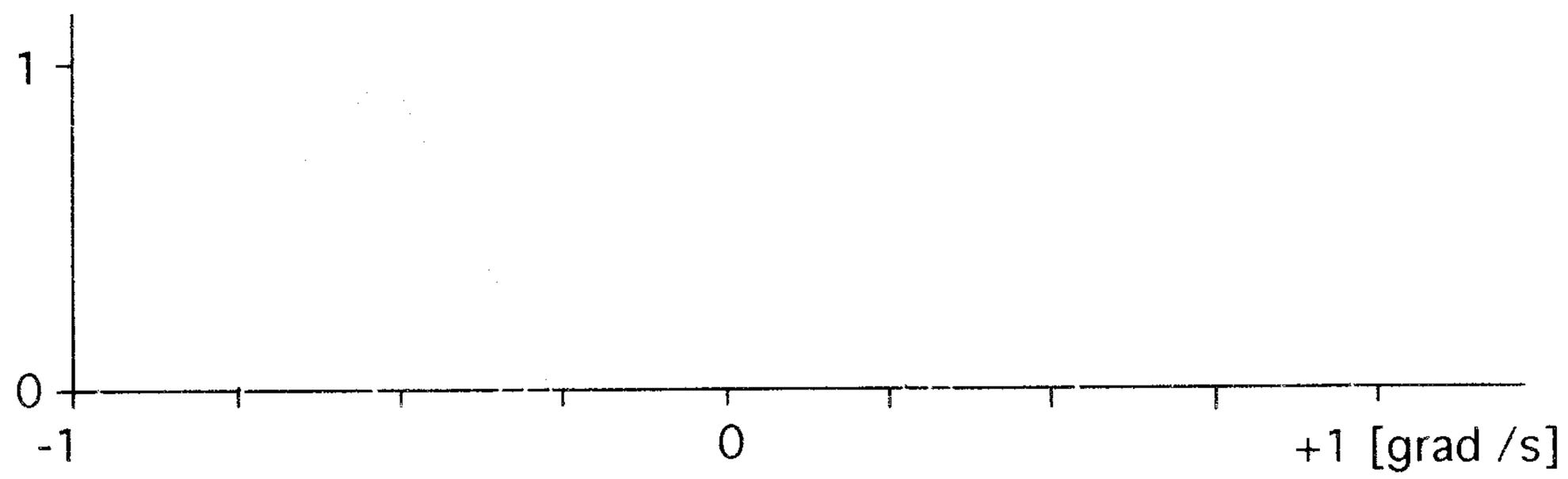

Abbildung 4.8: „Omega" nach Optimierung

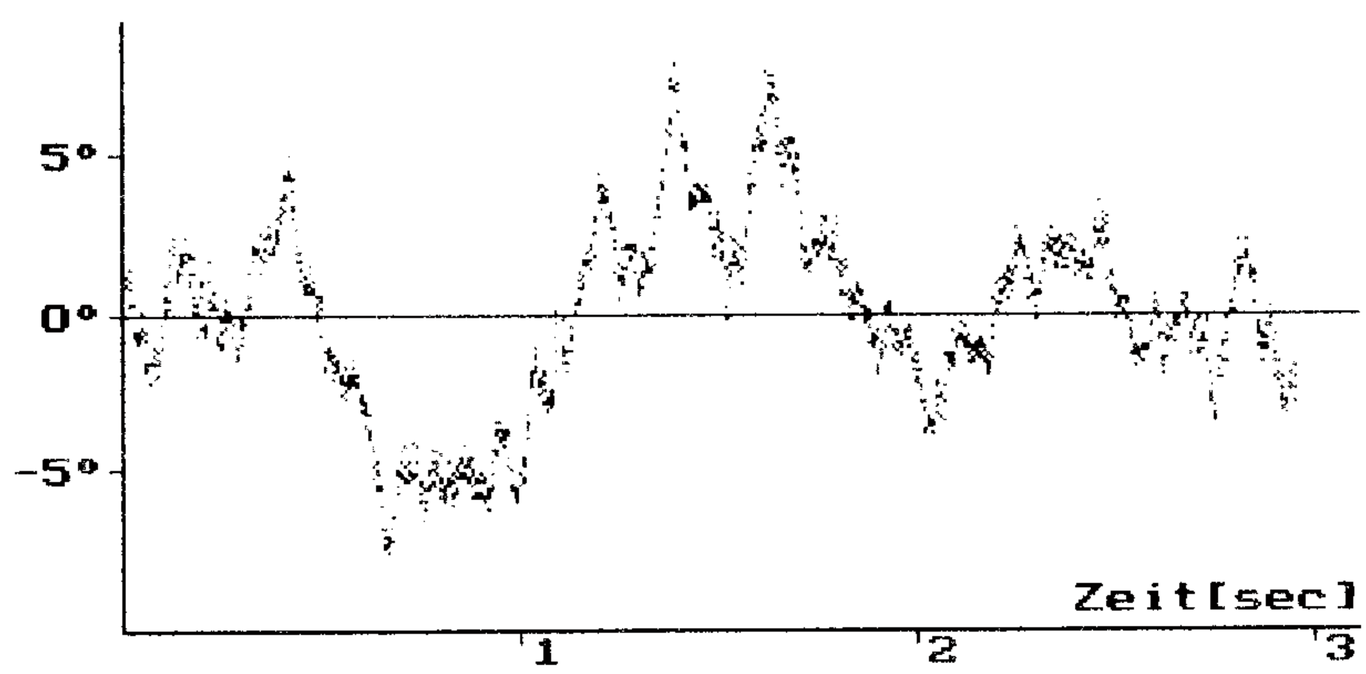

Abbildung 4.9: Zeitverlauf des Winkels vor Optimierung

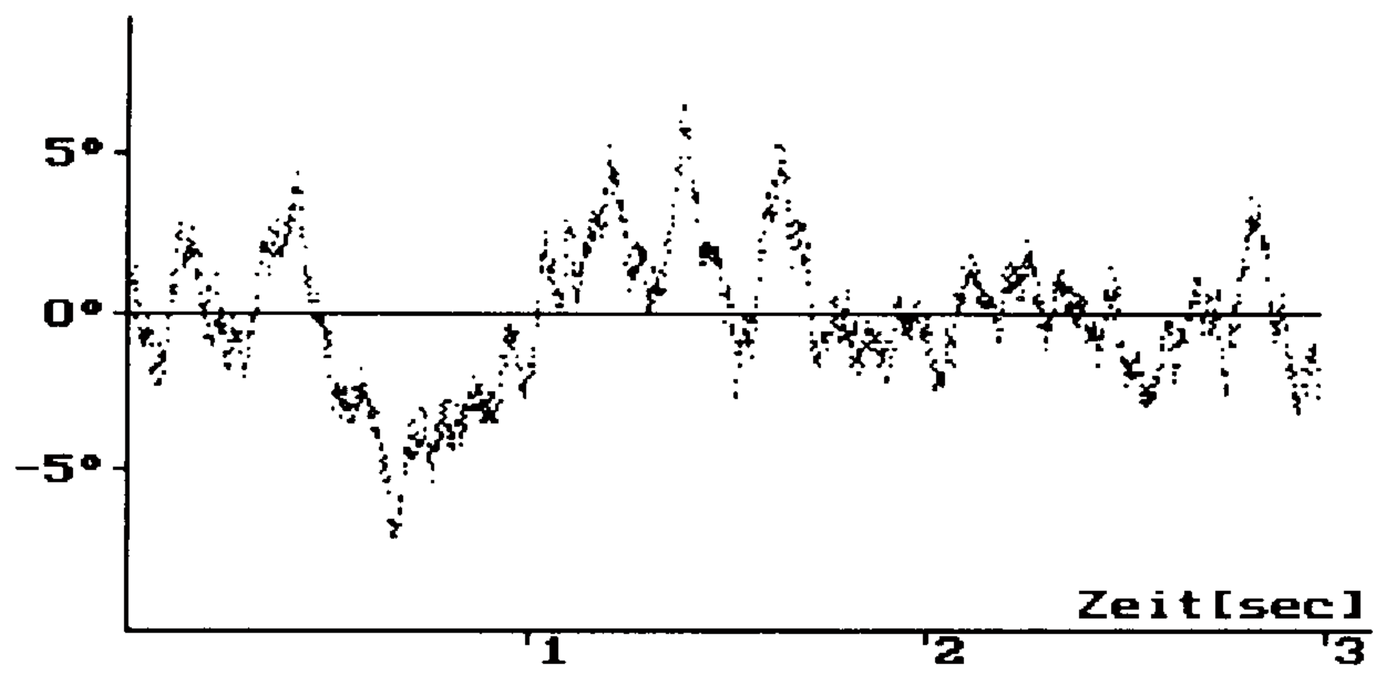

Abbildung 4.10: Zeitverlauf des Winkels nach Optimierung

aufgeschalteten Störungen gleich sind. Dennoch ist erkennbar, daß im zweiten Fall die Regelabweichung kleiner ist.

Die Verbesserungen des Regelfehlers erfolgen relativ gleichmäßig verteilt über 55 Generationen.Da aber in diesem Fall eine einfache Form der Evolutionsstrategie implementiert ist, kann auch bei dem Optimierungsverlauf kein bestes Ergebnis erwartet werden.

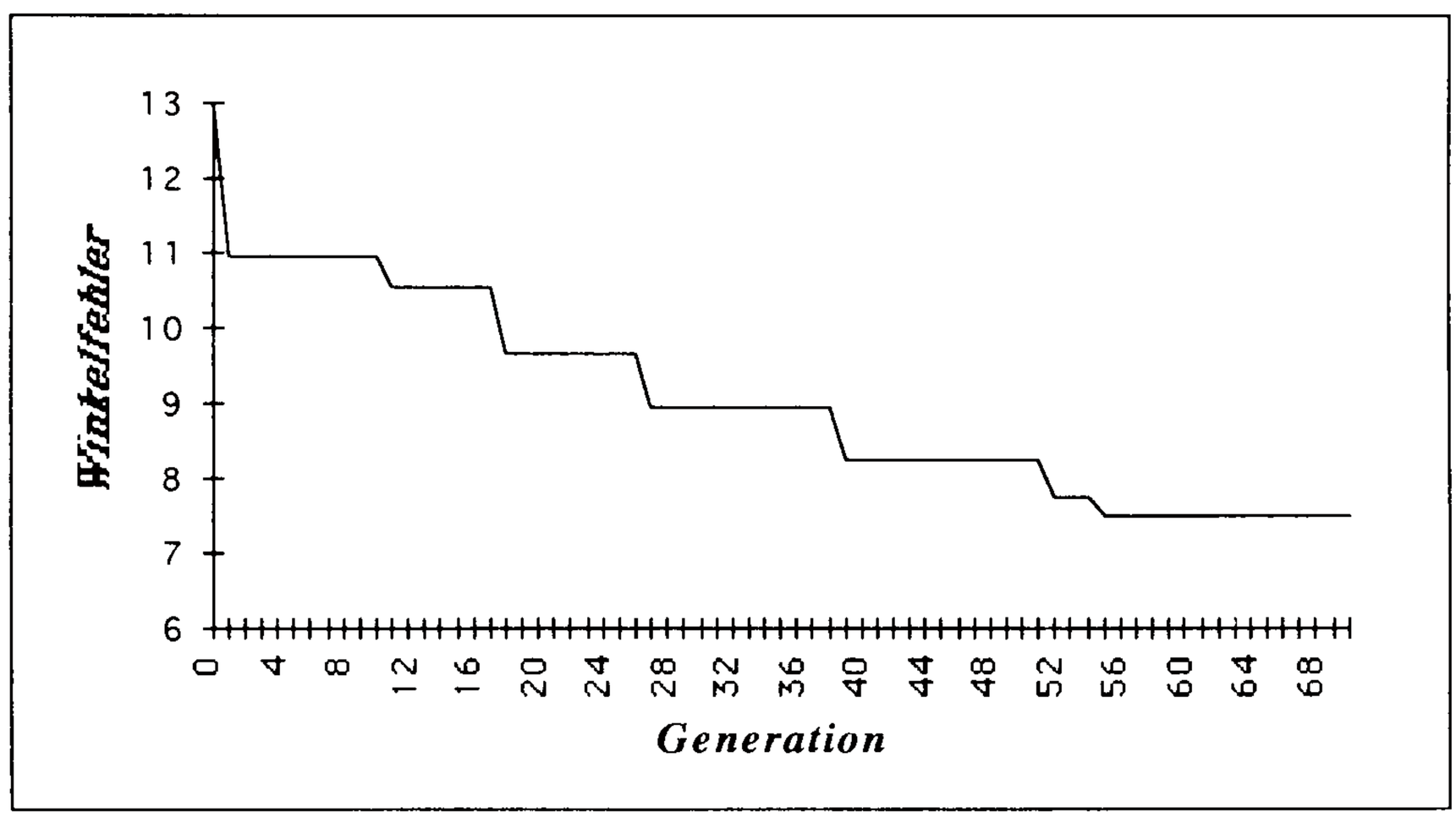

Abbildung 4.11: Optimierungsverlauf

4.5 Kennlinienfelder

Um das Regelverhalten vor und nach der Optimierung darzustellen, ist die Stellgröße in einem 3–dimensionalen Kennlinienfeld dargestellt. Die beiden Eingangsgrößen „Win-

kel"und „Omega"bilden die xy Ebene. Auf dieser Ebene steht senkrecht die z Achse, die von der Stellgröße gebildet wird.

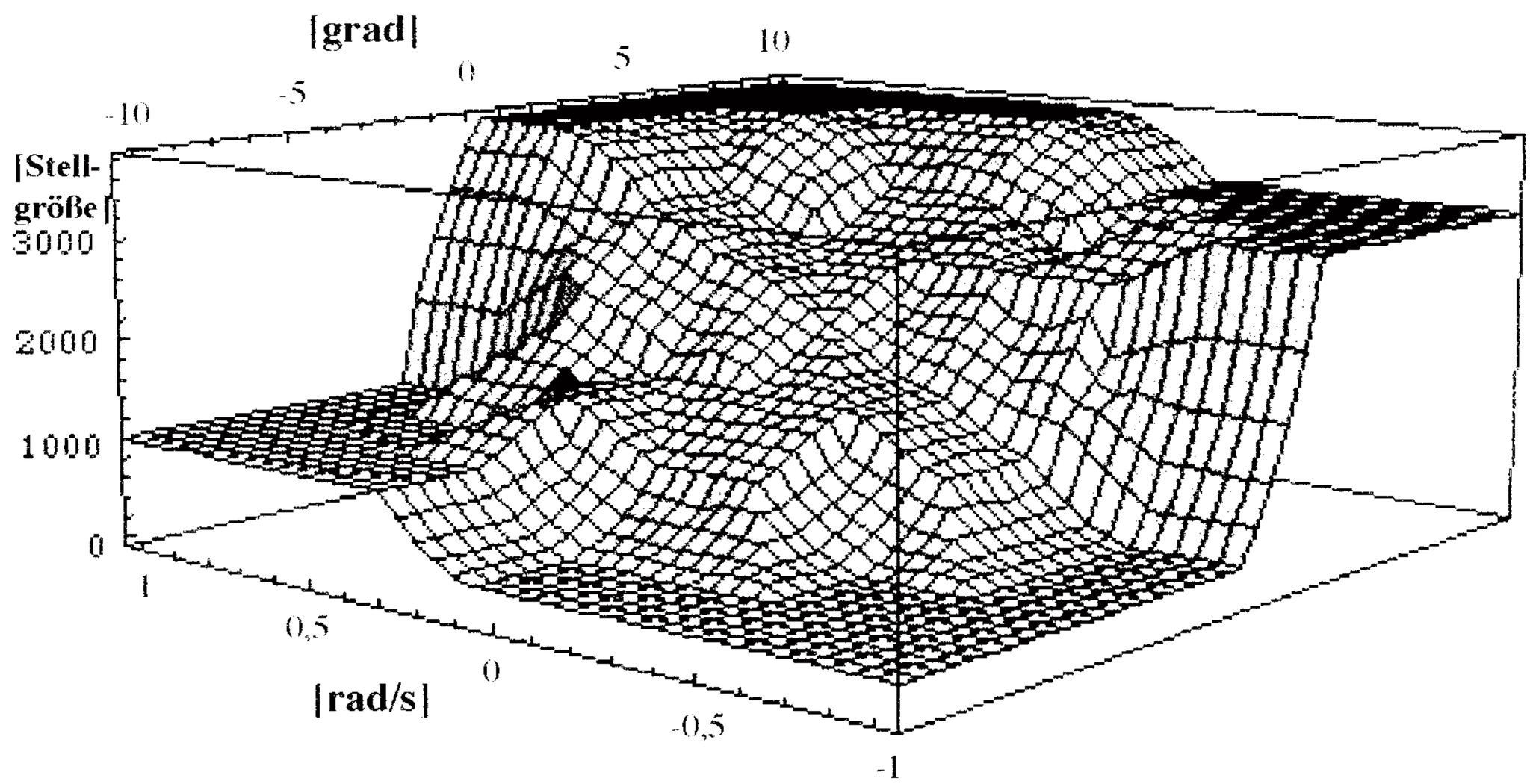

Abbildung 4.12: Kennlinienfeld vor Optimierung

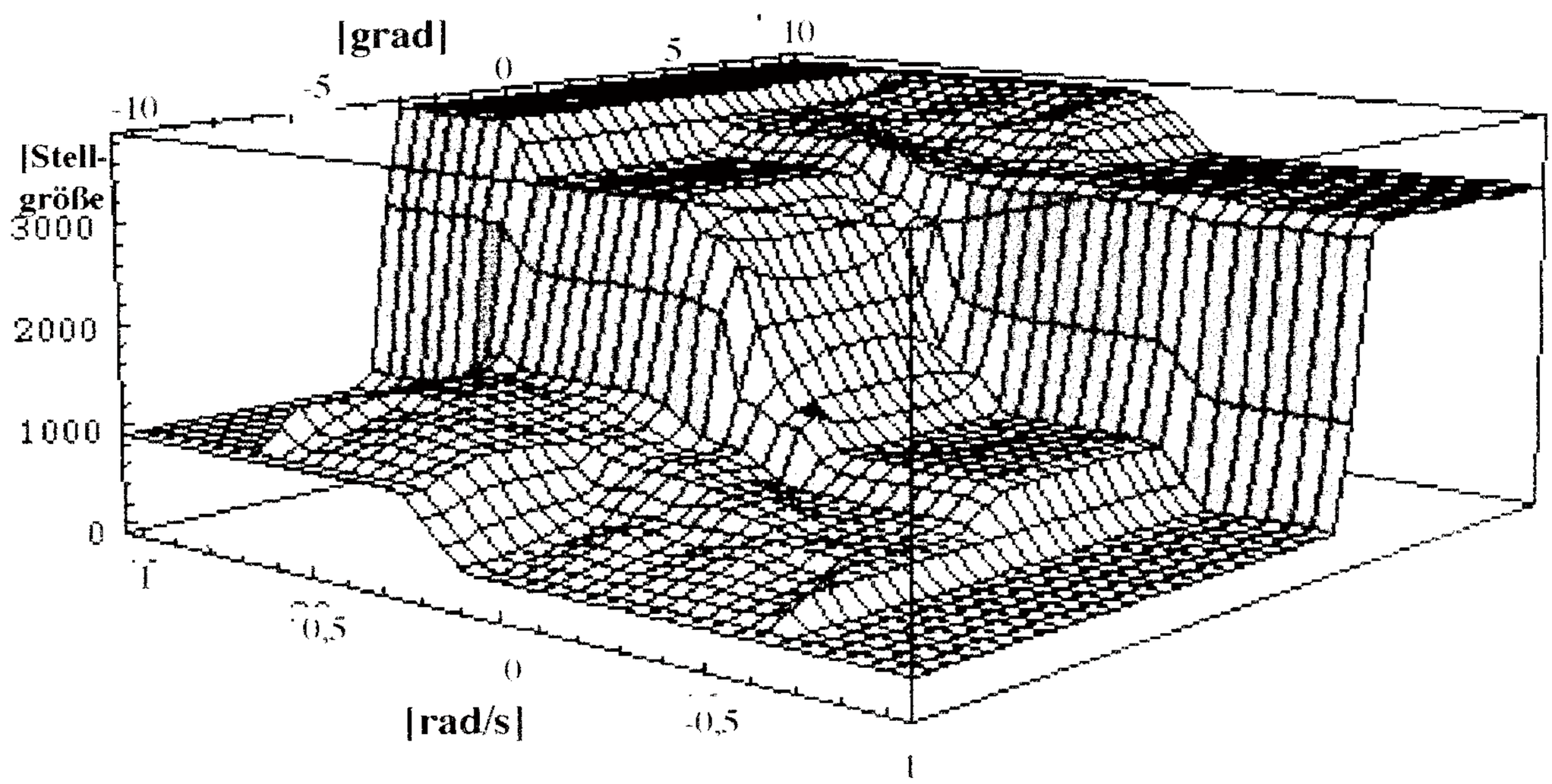

Abbildung 4.13: Kennlinienfeld nach Optimierung

Das Kennlinienfeld 4.12 hat die Form, die auf Grund der vorgegebenen Regelbasis zu erwarten ist. Betrachtet man das Regelverhalten in Bild 4.13, fällt auf, daß viele Flächen parallel zur x oder y Achse verlaufen. In diesen Bereichen ist die Stellgröße

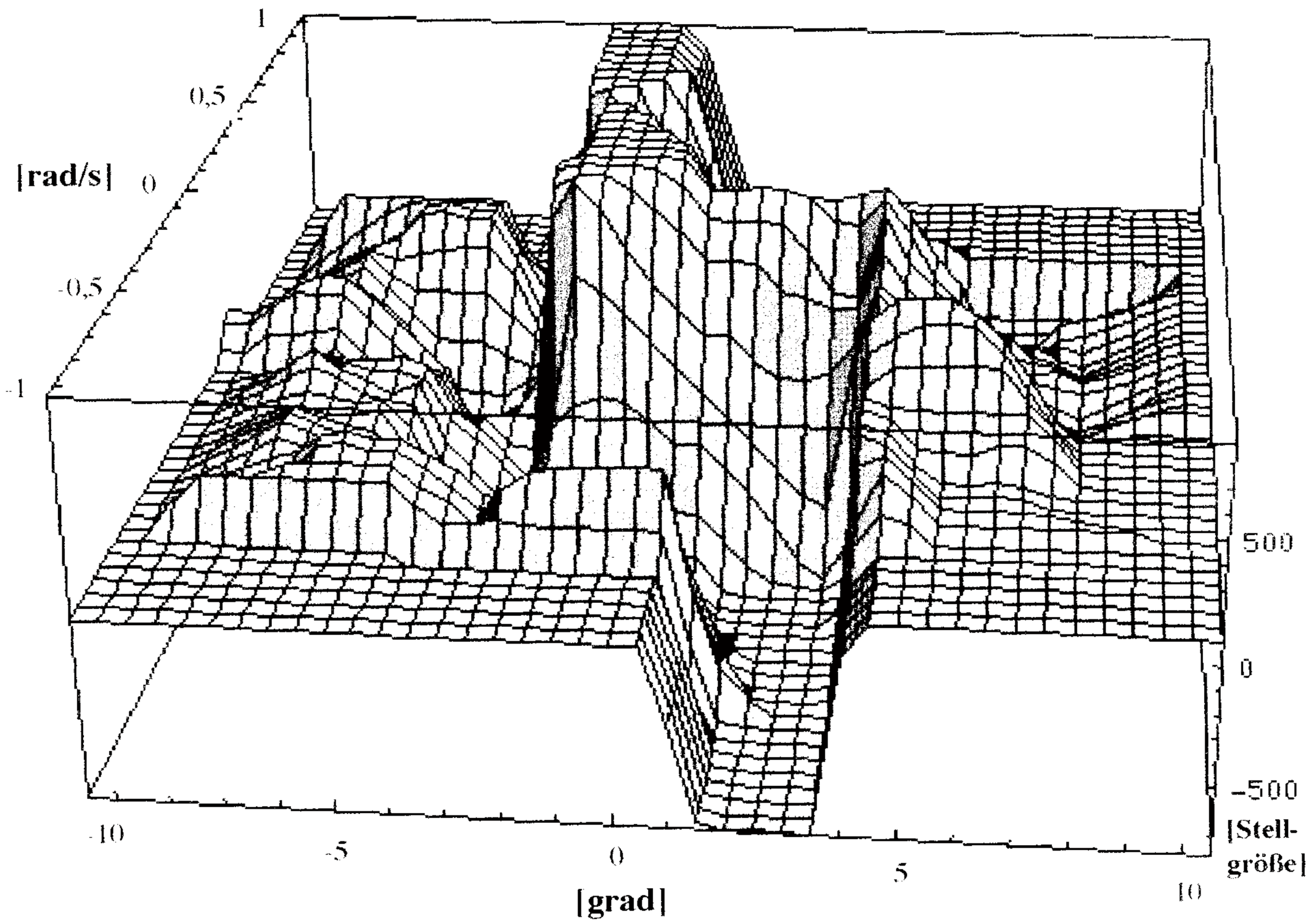

Abbildung 4.14: Differenz zwischen den Kennlinien (3D)

jeweils nur vom Winkel oder von der Winkelgeschwindigkeit abhängig. So wirkt bei einem Winkel zwischen $-2,5[°]$ bis $+2,5[°]$ fast nur die Winkelgeschwindigkeit auf die Stellgröße ein. Bild 4.14 zeigt die Differenz zwischen den Kennlinien 4.12 und 4.13 . Es ist erkennbar, daß sich der stufenförmige Bereich verschoben und vergrößert hat. Da in der 3-dimensionalen Darstellung einige Bereiche verdeckt werden, ist das Differenzfeld noch einmal als Density-Grafik dargestellt. In diesem Bild kann man nun erkennen, daß in zwei Bereichen eine extrem starke Änderung erfolgte.

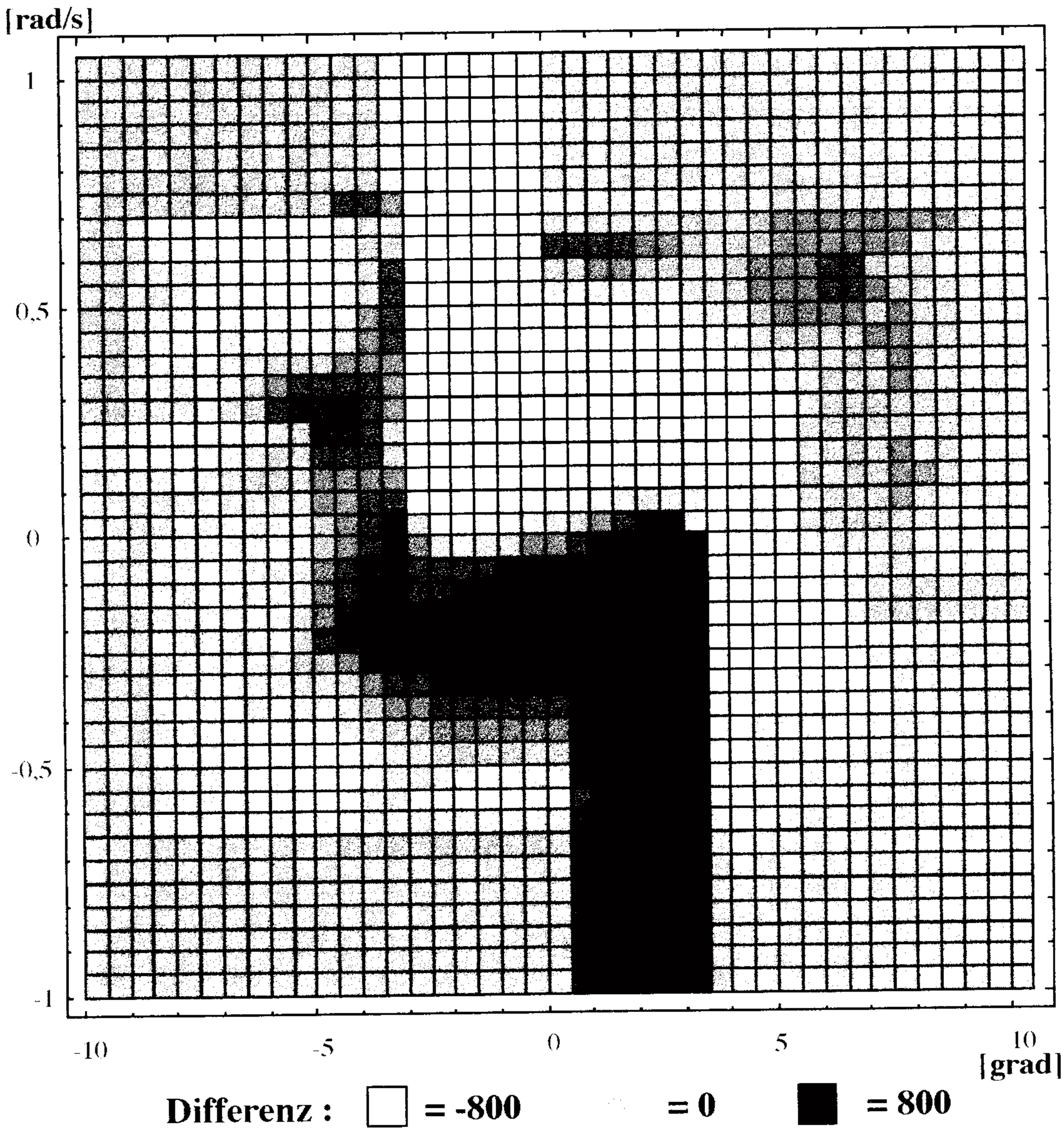

Abbildung 4.15: Differenz zwischen den Kennlinien (Density)

Das Bild 4.15 wird nun mit der Regelmatrix 4.1 verglichen. Werden dabei die nicht optimierten linguistischen Variablen zugrunde gelegt, fällt auf, daß einige Regeln nicht das gewünschte Regelverhalten beschreiben. In den schwarzen Bereichen ist die

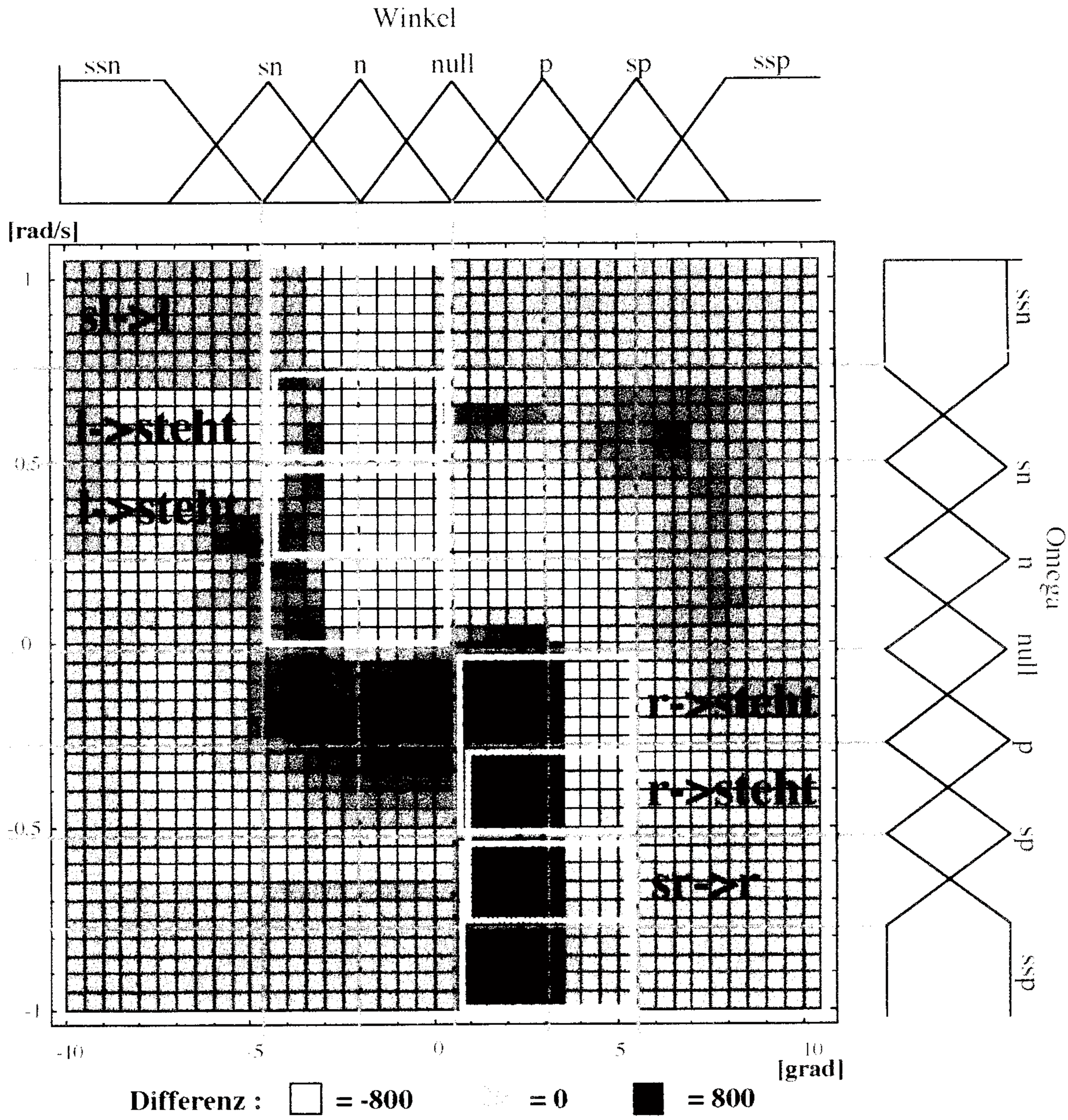

Abbildung 4.16: Auswahl der falschen Regeln

Stellgröße um jeweils einen Wert (bezogen auf eine unscharfe Menge) zu groß. In den weißen Bereichen ist die Stellgröße zu klein. Die Regelbasis wird nun entsprechend nachkorrigiert. In Tabelle 4.2 sind die korrigierten Regeln in Fettschrift hervorgehoben. Die Auswahl der nachzukorrigierenden Regeln ist in Bild 4.16 grafisch dargestellt. Die Bereiche, die jeweils von einer falschen Regel beschrieben werden, sind durch eine farbige Umrandung markiert.

Wie zu erkennen ist, überschneiden die weißen und schwarzen Bereiche sich nicht genau mit den Wertebereichen der entsprechenden Fuzzy-Sets. Um die Regelkorrektur optimal durchzuführen, müßte entweder die Regelanzahl erhöht werden, was aber zu einer sehr großen Regelbasis führt, oder die Sets müßten in die entsprechenden Bereiche verschoben werden. Für einen erneuten Optimierungslauf sind aber die gleichmäßig verteilten Sets beibehalten worden,da alle Routinen des Programms darauf abgestimmt sind. Diese Einschränkung liegt darin begründet, daß bei Erstellung des Programms die Möglichkeit der Regelkorrektur nicht absehbar war.

		φ						
		ssn	*sn*	*n*	*mitte*	*p*	*sp*	*ssp*
$\dot{\varphi}$	*ssn*	ssl	ssl	ssl	ssl	**r**	sr	sr
	sn	ssl	ssl	sl	sl	**steht**	sr	sr
	n	ssl	sl	l	l	**steht**	r	sr
	null	ssl	sl	l	steht	r	sr	ssr
	p	sl	l	**steht**	r	r	sr	ssr
	sp	sl	sl	**steht**	sr	sr	ssr	ssr
	ssp	sl	sl	**l**	ssr	ssr	ssr	ssr

Tabelle 4.2: Regelbasis nach Korrektur

Das Regelverhalten dieser Regelbasis und den nicht optimierten linguistischen Variablen zeigt Bild 4.17. Wenn man das Kennlinienfeld 4.17 betrachtet, sind keine großen Unterschiede zu Bild 4.12 zu erkennen. Deshalb ist in Bild 4.18 die Differenz zwischen 4.12 und 4.17 grafisch dargestellt. Wird dieses Bild mit Bild 4.15 verglichen, zeigt sich, daß die Änderung durch die Regelkorrektur ähnlich der Optimierung der Variablen ist. Die weißen und schwarzen Bereiche sind ungefähr gleich.

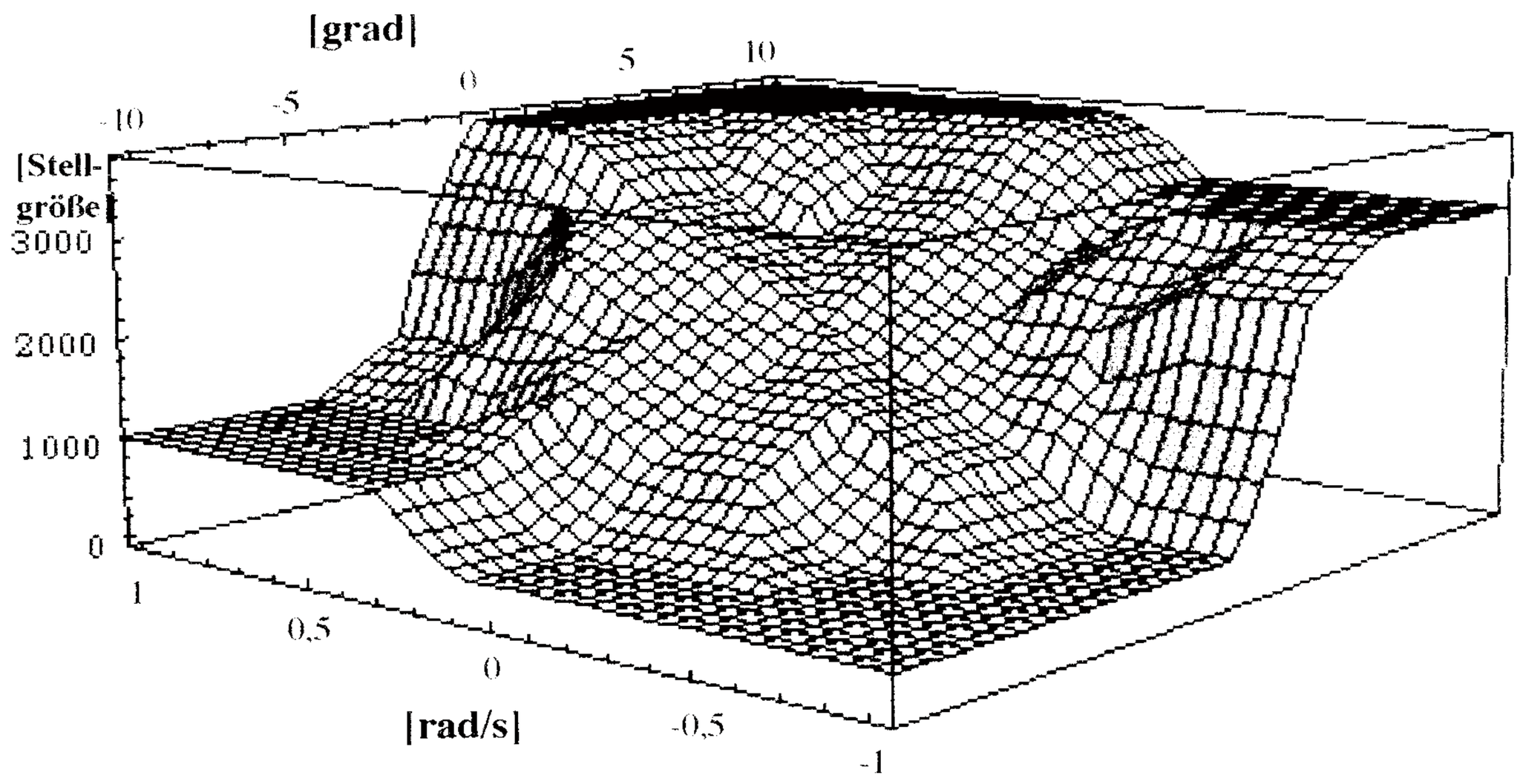

Abbildung 4.17: Kennlinienfeld nach Regelkorrektur (vor Optimierung)

Bild 4.19 zeigt die Differenz zwischen Bild 4.17 und 4.13. Also den Unterschied zwischen dem Regelverhalten nach der ersten Optimierung mit **ES** und dem Regelverhalten nach der Regelkorrektur. Der Z-förmige Bereich entsteht durch die nicht ganz übereinstimmenden schwarzen und weißen Bereiche von Bild 4.16, mit den Sets der korrigierten Regeln. Für eine exakte Regelkorrektur müßten die ausgewählten Bereiche vollständig weiß oder schwarz sein.

Mit der neuen Regelbasis wird eine weitere Optimierung der linguistischen Variablen durchgeführt. Nach 33 Generationen erfolgt keine weitere Verbesserung. In den Bildern 4.20, 4.21 und 4.22 sind die linguistischen Variablen nach dem zweiten Optimierungslauf dargestellt.

Bei der Variablen „Winkel" fällt auf, daß zwischen dem Fuzzy-Set „null" und den angrenzenden Sets eine Lücke ist. Für die Winkel in diesen Bereichen entsteht dadurch eine Lücke im Regelverhalten. Diese Lücke ist ein Zeichen dafür, daß das erreichte Suboptimum nicht besonders gut ist. Um dieses zu verhindern muß entweder ein besseres ES-Verfahren (siehe 4.6), oder eine Korrektur von Hand erfolgen.

Um die Lücken zu schließen, wird das Set „null" derart verbreitet, daß es die angrenzenden Sets überschneidet.

Bild 4.23 zeigt das Regelverhalten mit den Lücken. Diese Lücken bilden einen Einschnitt oder eine Erhebung mit dreieckigem Profil im Diagramm. Nachdem das Set „null" verbreitert wurde, verschwinden auch die Lücken im Regelverhalten (Bild

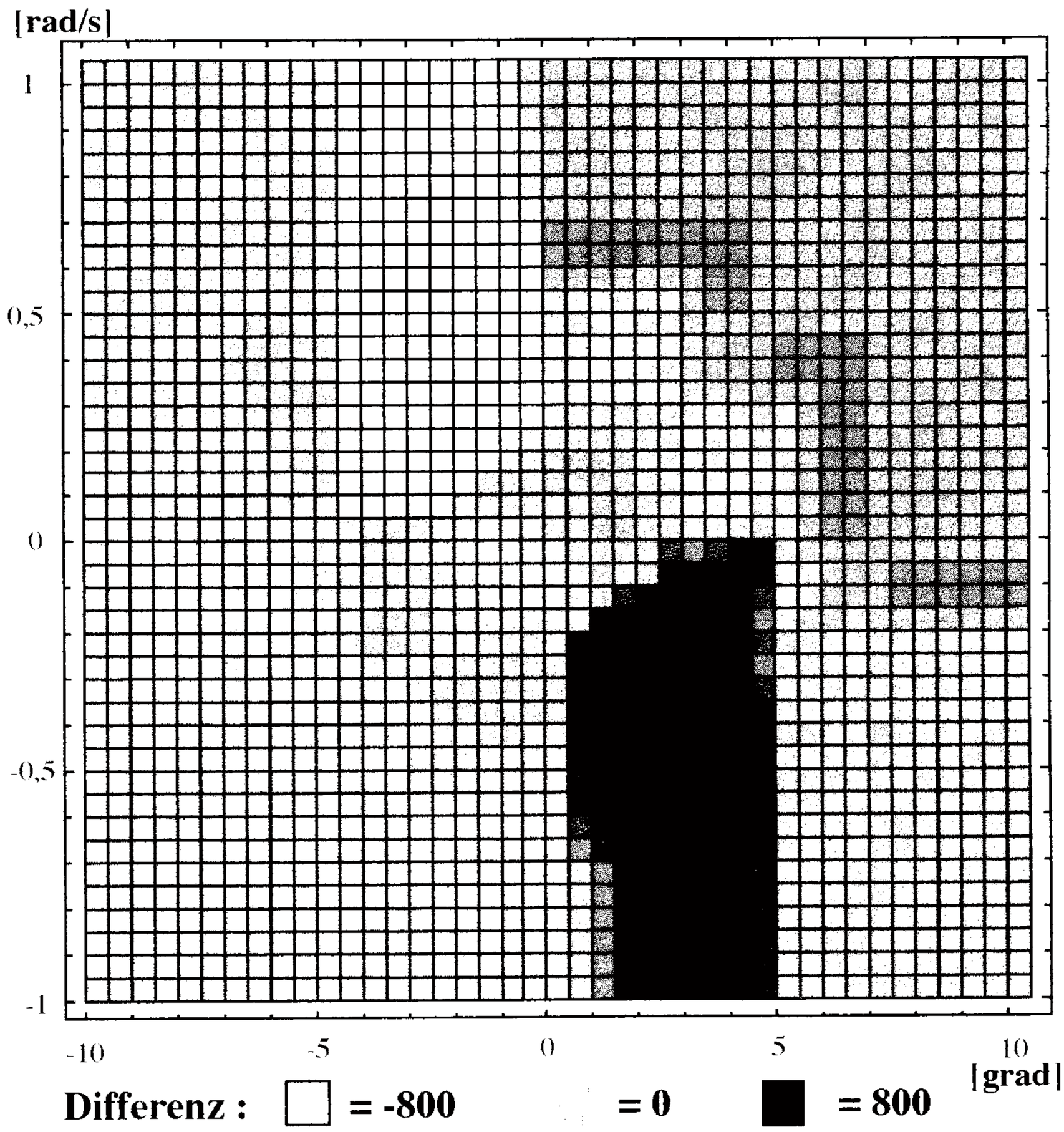

Abbildung 4.18: Differenzfeld 2

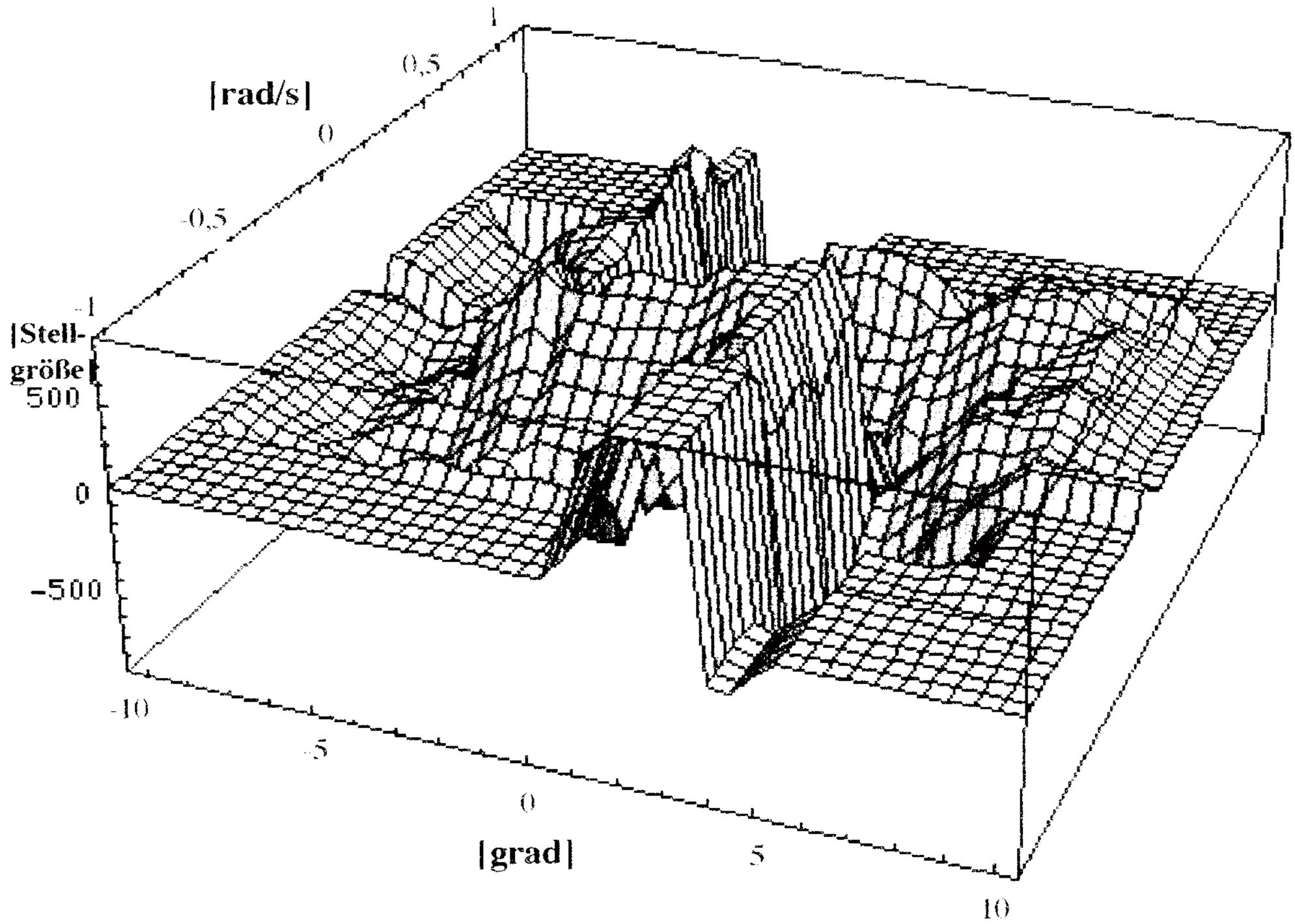

Abbildung 4.19: Differenzfeld 3

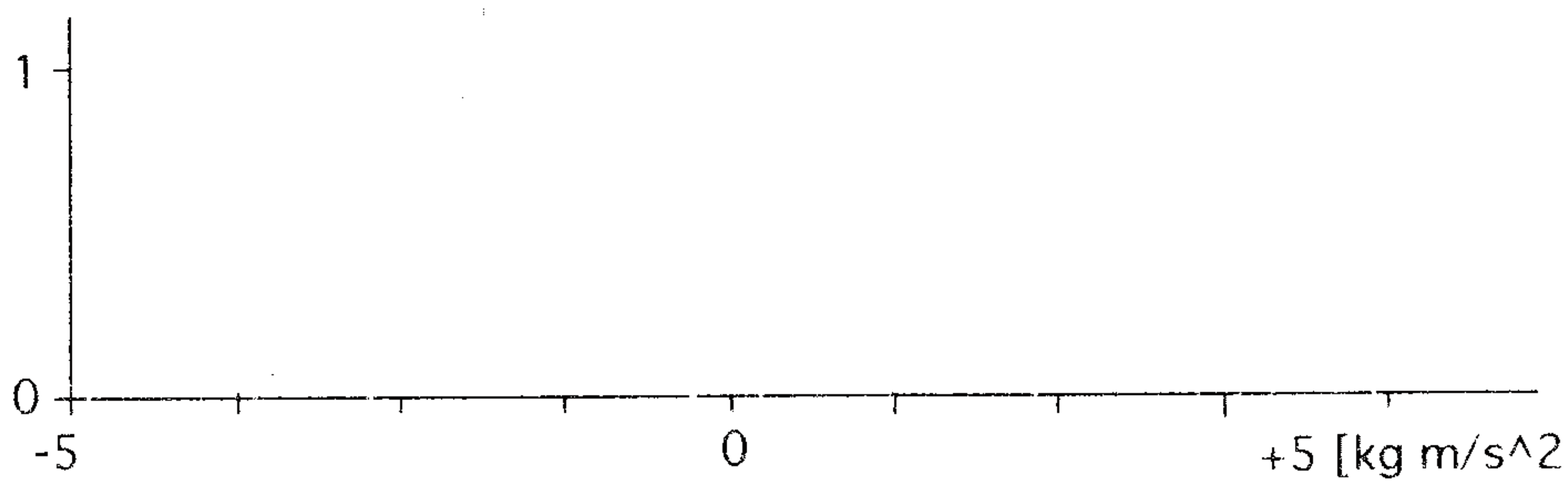

Abbildung 4.20: „Wagen" nach Optimierung und Regelkorrektur

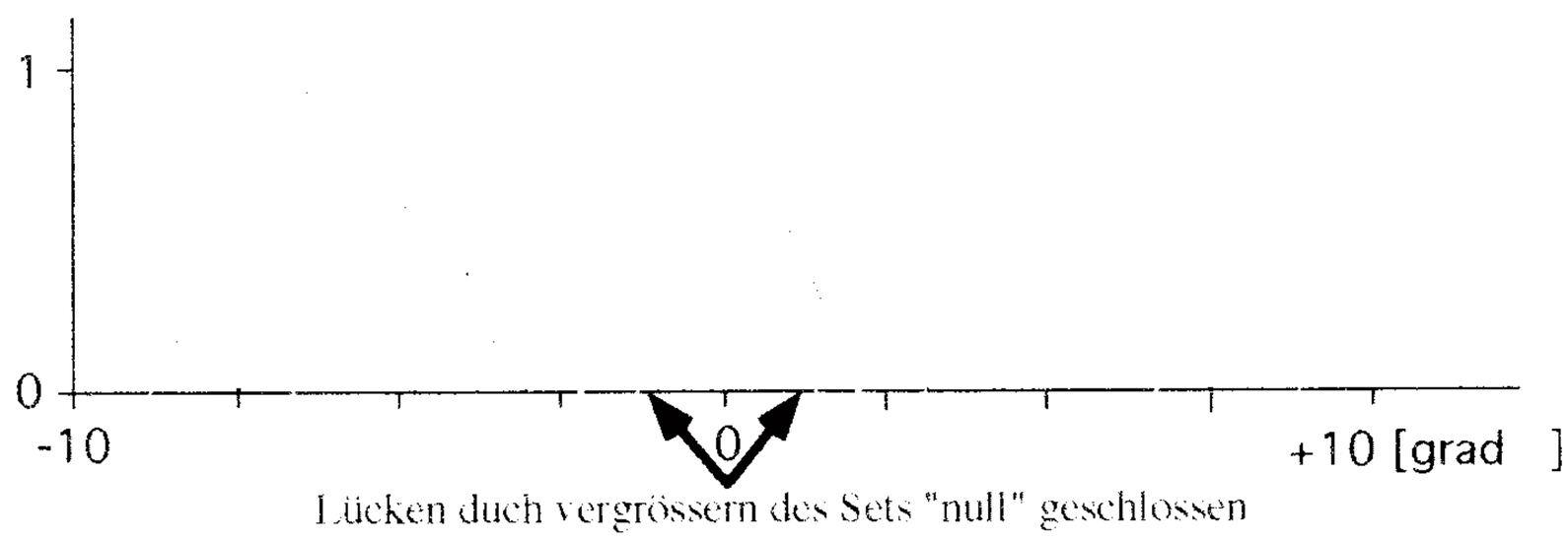

Abbildung 4.21: „Winkel"nach Optimierung und Regelkorrektur

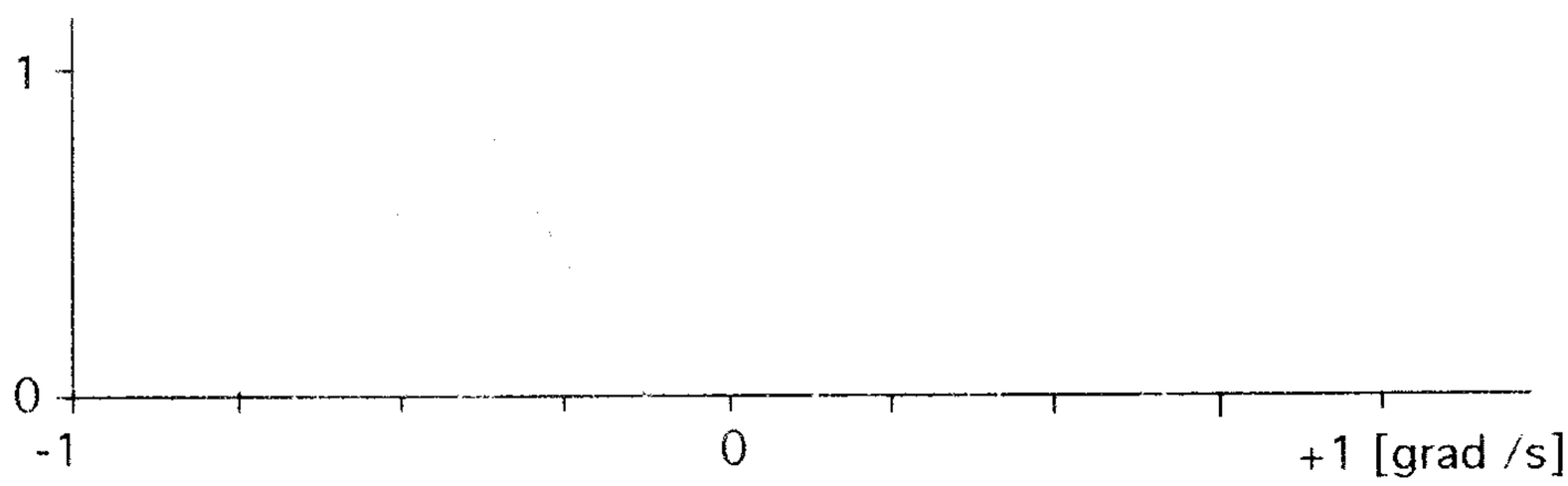

Abbildung 4.22: „Omega"nach Optimierung und Regelkorrektur

4.24).

Die Differenz zwischen Bild 4.23 und 4.24 zeigt Bild 4.25. Es ist zu erkennen, daß durch die Set-Änderung nur die Regellücken verschwunden sind. Der Rest des Regelverhaltens wurde nicht verändert.

Wie in Bild 4.28 zu sehen ist, ist der Optimierungsverlauf nach der Regelkorrektur etwas besser geworden. Der Regelfehler wird schneller kleiner und ist auch beim Abbruch der Optimierung kleiner. Nach dem 1.Optimierungslauf ist der Regelfehler um 42.4% besser geworden. Um 45.3% hat er sich nach zwei Optimierungsläufen und einer Regelkorrektur verbessert. Wird der Zeitverlauf jeweils vor und nach der Optimierung verglichen, dann fällt auf, daß die durch Störungen erzeugten Spitzen schneller ausgeregelt werden. Dadurch schwingt der Stab aber auch stärker über den Nullpunkt hinaus.

4.6 Zusammenfassung und Ausblick

Bei dem implementierten Verfahren der Evolutionsstrategie, wird ein relativ schlechtes Suboptimum erreicht. Ein Grund dafür ist die kurze Simulation über 3 Sekunden. Innerhalb dieser Zeit kommen nicht alle möglichen Kombinationen zwischen Winkel und Winkelgeschwindigkeit vor. Wird die simulierte Zeit verdoppelt, um die Simu-

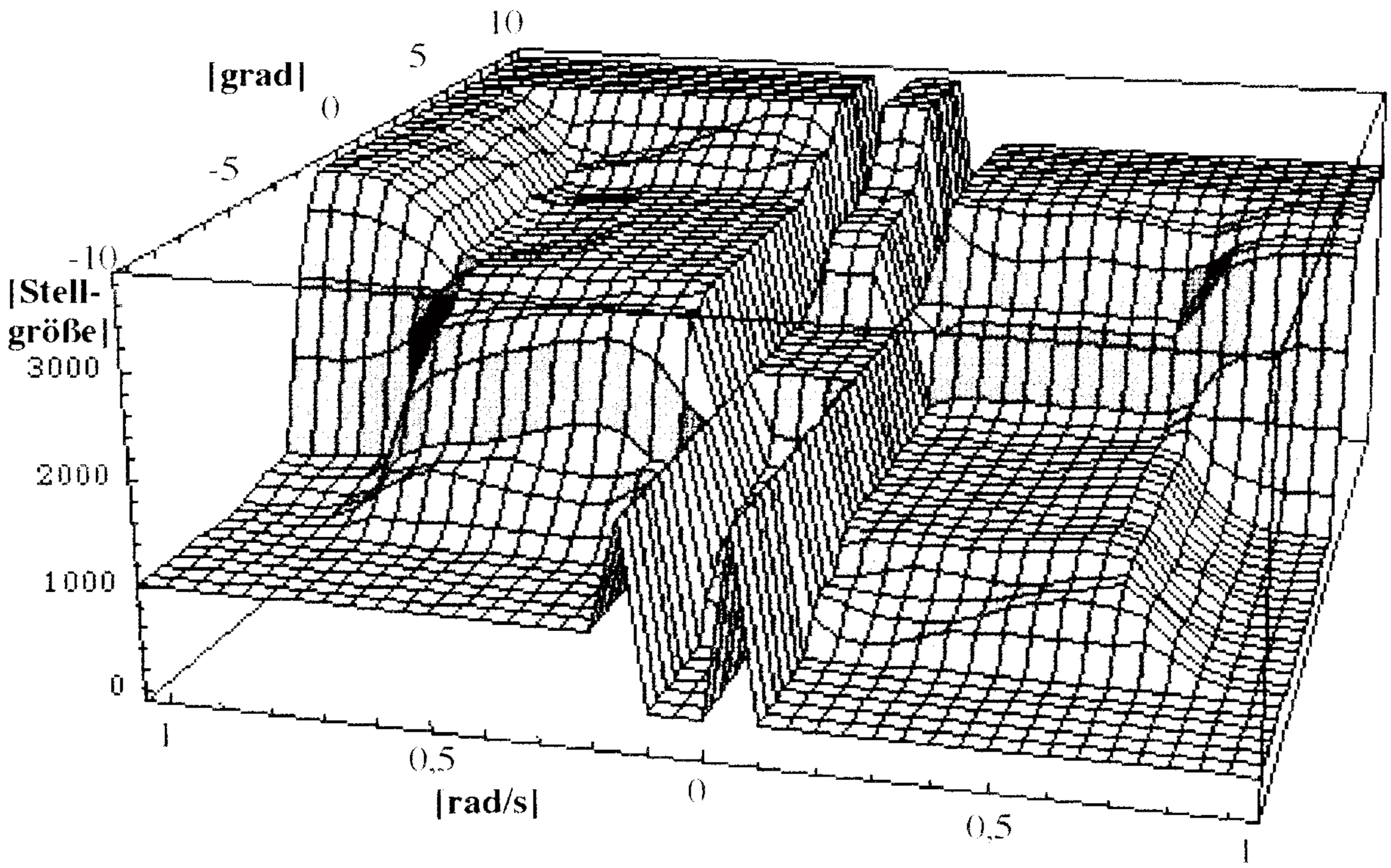

Abbildung 4.23: Kennlinienfeld nach Regelkorrektur (nach Optimierung)

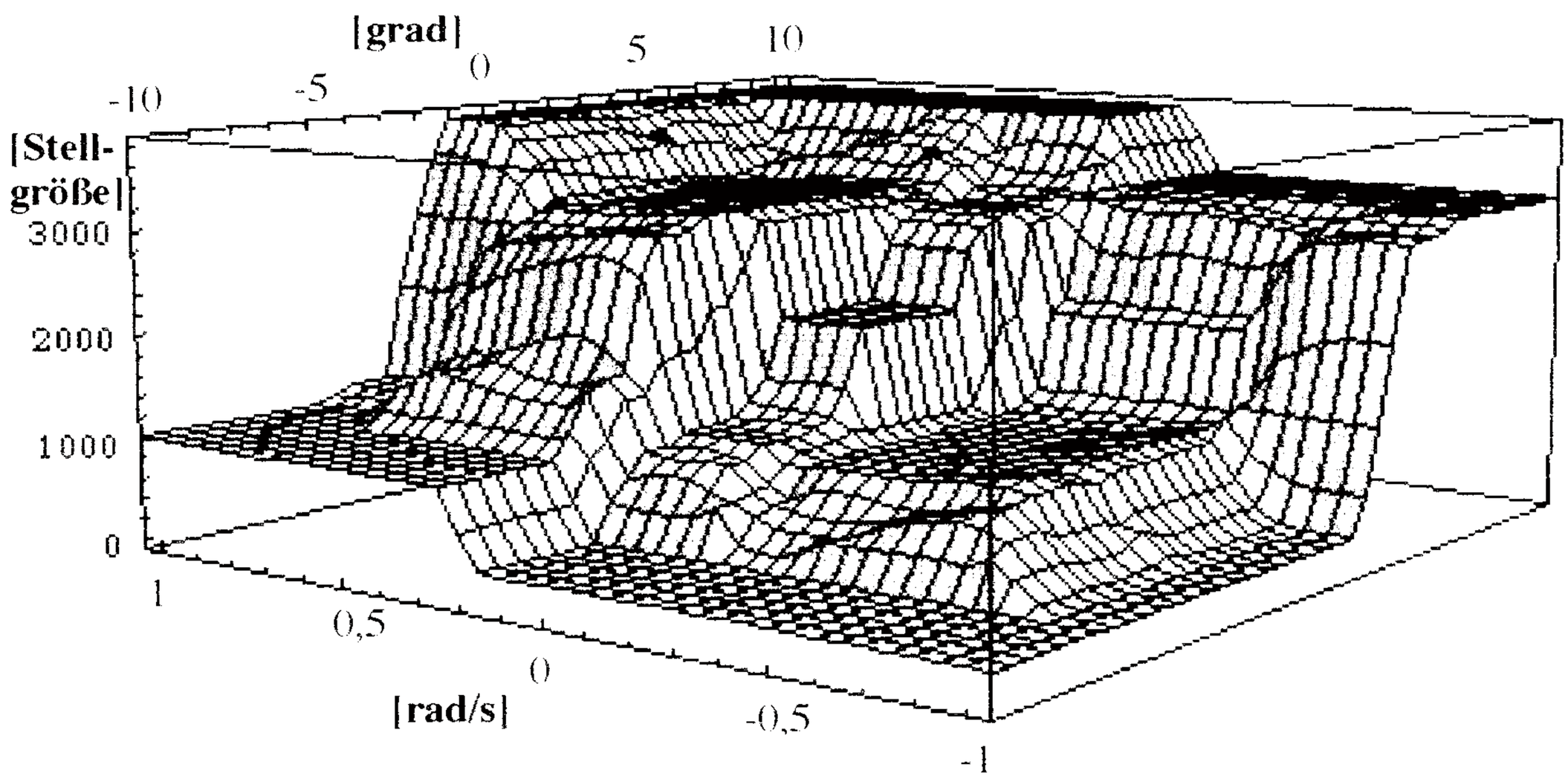

Abbildung 4.24: Kennlinienfeld ohne Lücke

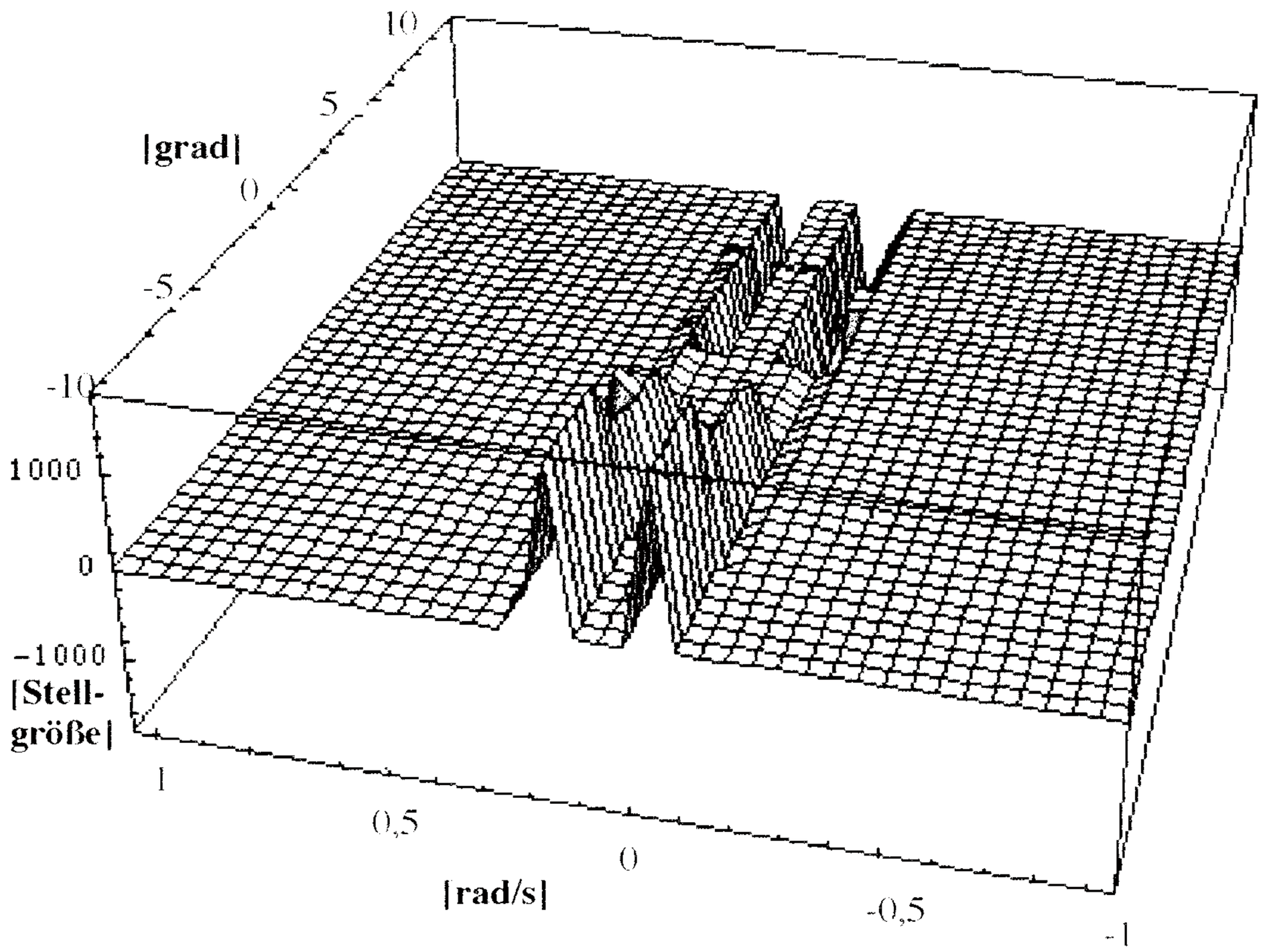

Abbildung 4.25: Differenzfeld 4

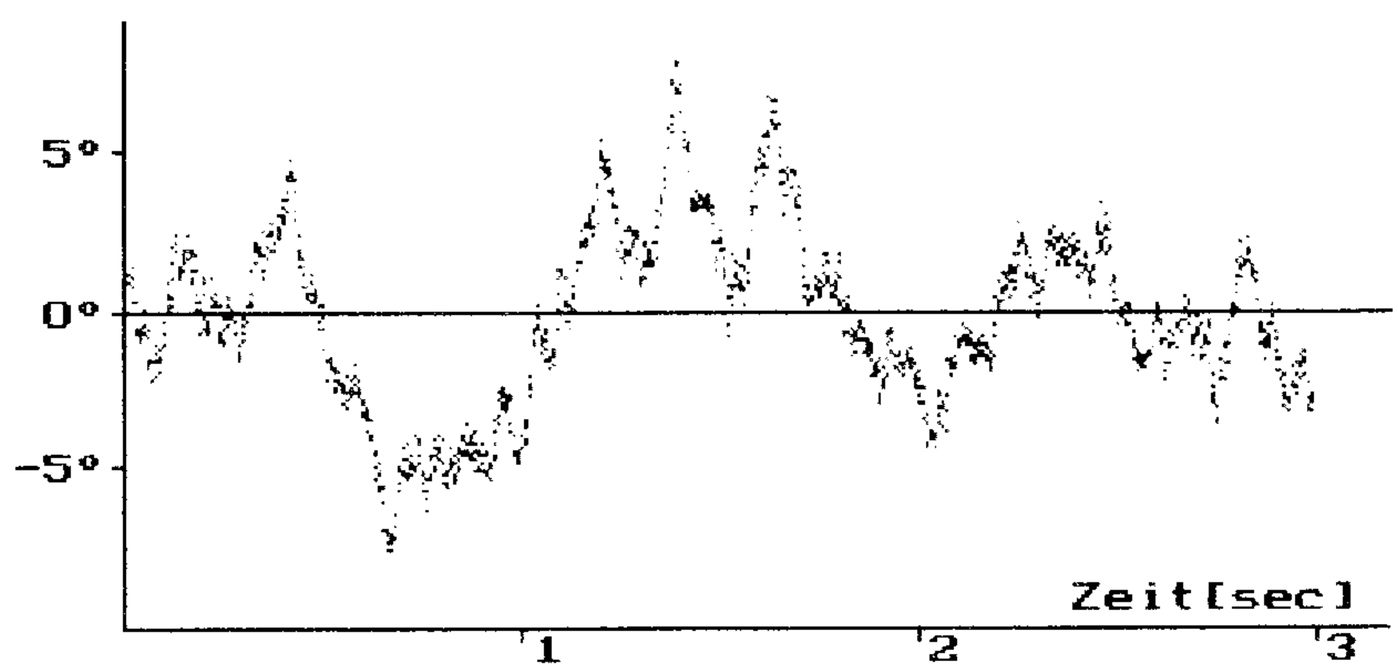

Abbildung 4.26: Zeitverlauf des Winkels nach Regelkorrektur (vor ES)

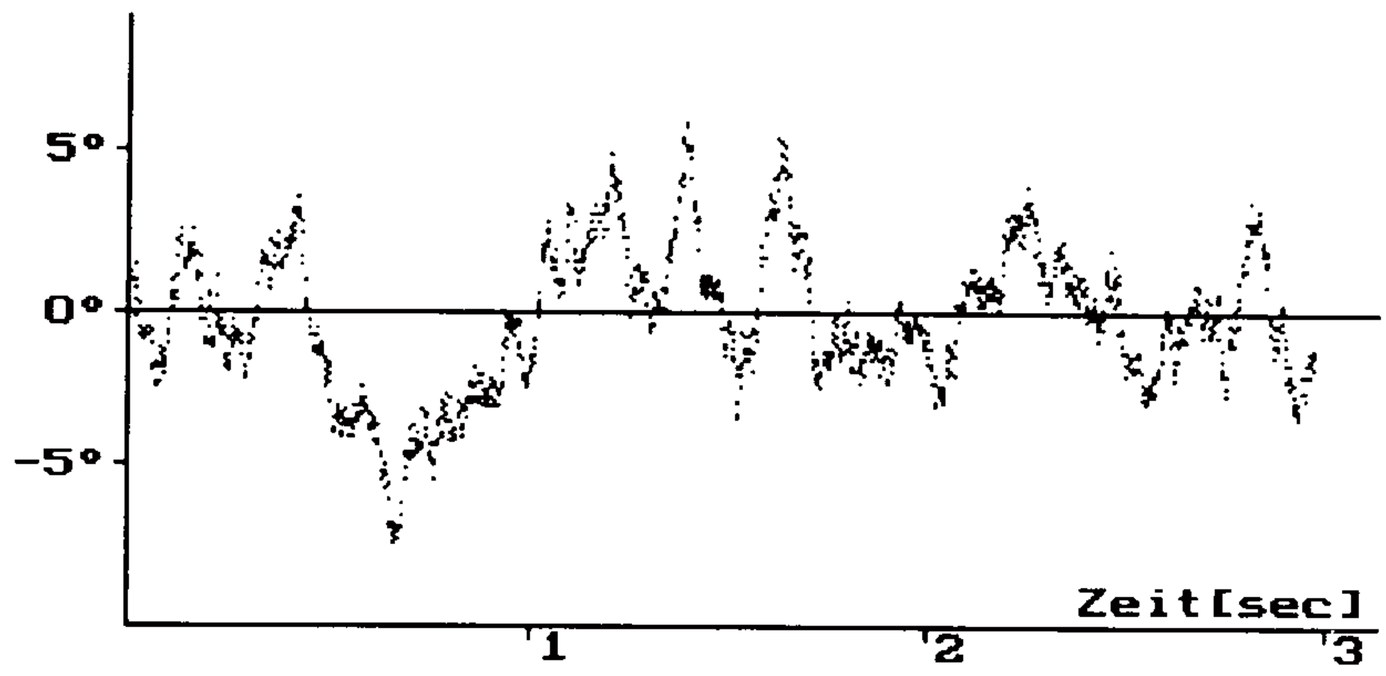

Abbildung 4.27: Zeitverlauf des Winkel nach Regelkorrektur (nach ES)

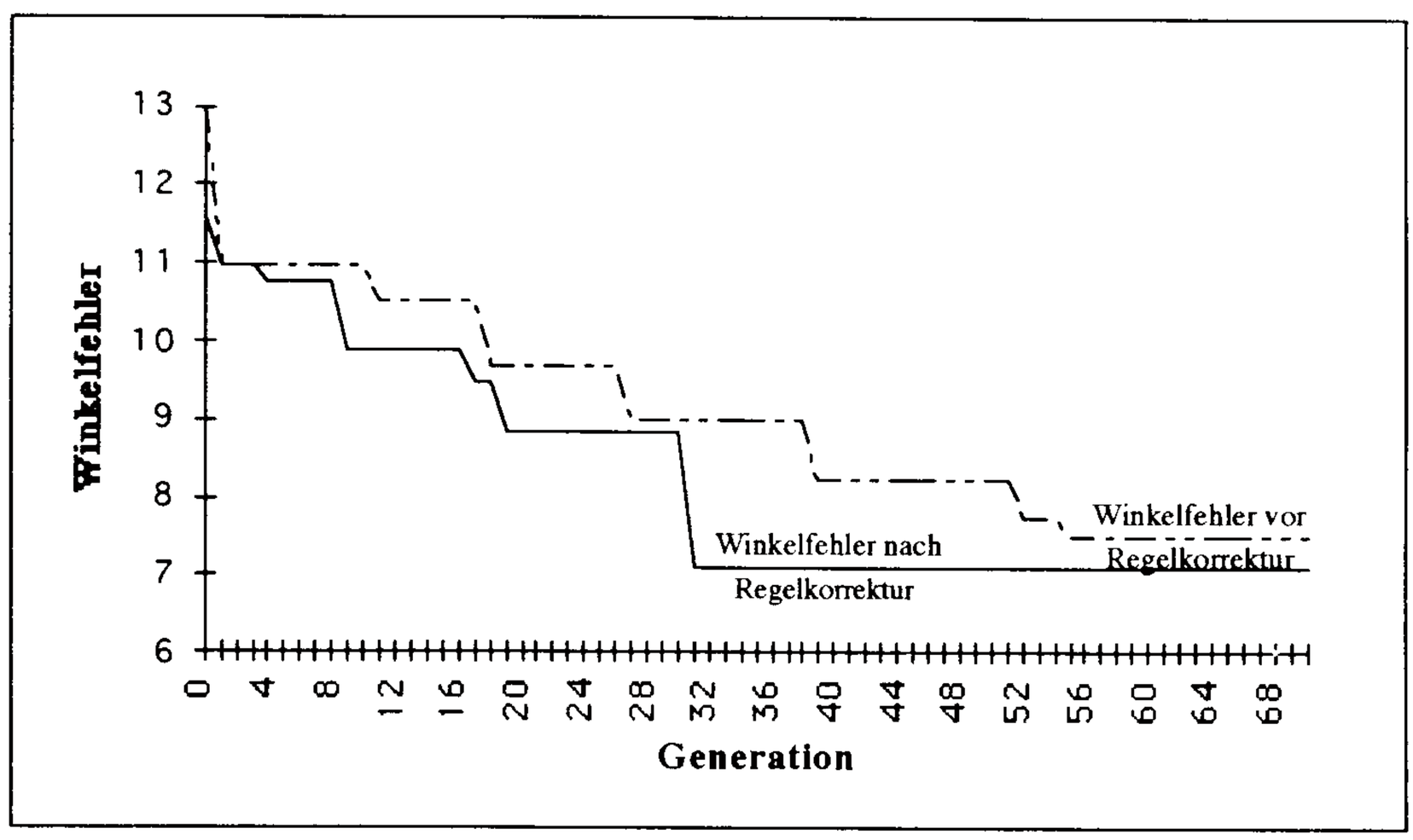

Abbildung 4.28: Optimierungsverlauf 2

lation mehr der Realität anzugleichen, verdoppelt sich aber auch die Gesamtzeit der Optimierung. Bei zwei Durchläufen, mit einer Regelkorrektur dazwischen, würde die Gesamtzeit etwa 12 Tage betragen.

Ein weiterer Grund ist die hier verwendete (1+5)-gliedrige Evolutionsstrategie. Sie ist der einfachste Algorithmus der **ES**. Werden mehrere Eltern verwendet und durch Rekombination die Nachkommen erzeugt, oder bei Erreichen eines Suboptimums ein mehrere Generationen zurückliegender Elter verwendet, so kann dadurch ein Festhängen in einem Suboptimum verhindert werden. Da aber die Datenstrukturen der linguistischen Variablen sehr groß und komliziert sind, bedeutet die Anwendung dieser ES-Algorithmen einen erheblichen Mehraufwand im Programm. Die Programmierung wird dadurch sehr fehleranfällig und die Simulationszeit für jede Generation steigt um ein vielfaches an. Die in Abb. 4.29 gezeigte Optimierung kann also mit

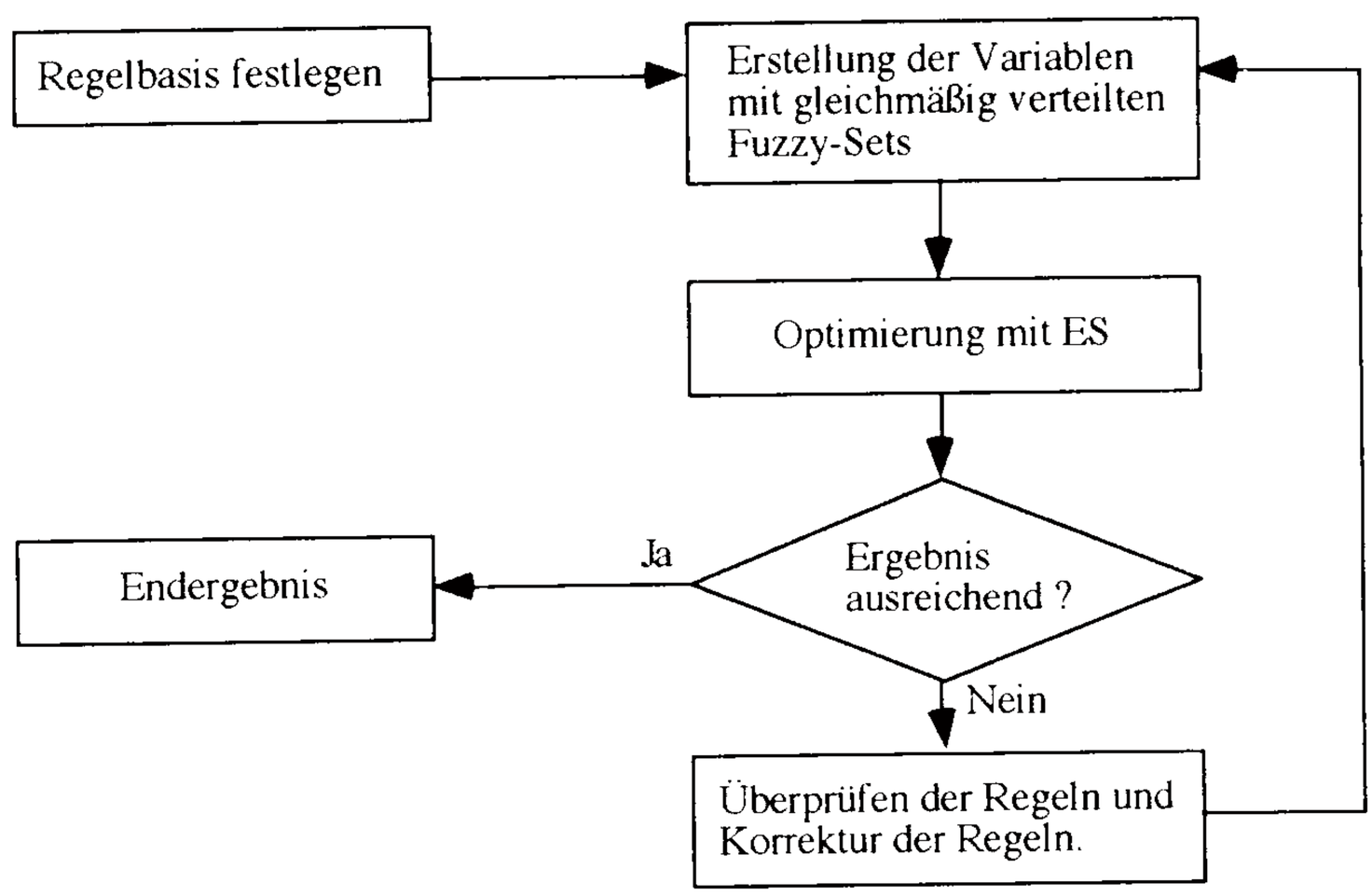

Abbildung 4.29: Flußdiagramm der Optimierung

folgenden Vorschlägen verbessert werden:

- Erhöhen der Simulationszeit.
- Anwendung eines besseren ES-Algorithmus.
- Anpassung von Form und Lage der Sets bei der Regelkorrektur.

Kapitel 5

Das reale System

Ein großes Problem bei der Regelung des Stab–Wagen–Systems mit einem **PC** ist dessen Rechengeschwindigkeit. Je größer die Regelbasis wird, desto schwieriger ist eine Fuzzy–Regelung in Echtzeit durchzuführen.

Wird nur eine Simulation durchgeführt, dann ist der Zeitfaktor nicht weiter schlimm. Bei dem Stab–Wagen–System ist es nun aber nicht möglich die Fuzzyfizierung und Defuzzyfizierung in Echtzeit durchzuführen. Ein möglicher Ansatz für die Lösung dieses Problems ist die in [6] beschriebene Darstellung der unscharfen Mengen durch diskrete Werte. Wenn man davon ausgeht das viele AD/DA–Karten nur mit 8–Bit Wandlern arbeiten,dann kann ein linguistische Variable in maximal 256 diskrete Werte eingeteilt werden.

Von diesem Ansatz ausgehend werden nun nicht die Fuzzy–Sets in diskrete Werte umgewandelt, sondern für eine feste Anzahl verschiedener Eingangswerte wird die Fuzzyfizierung und Defuzzyfizierung durchgeführt. Diese Werte sind gleichmäßig über den ganzen Bereich der linguistische Variable verteilt.Anschließend werden die berechneten diskreten Stellgrößen in eine Regel–Matrix eingetragen. Sind alle Felder der Matrix berechnet, wird sie in der Datei „`Regeltab.dat` "abgespeichert.

Bei dem Stab–Wagen–System sind die linguistischen Variablen „Winkel "und „Omega "jeweils in 101 diskrete Werte aufgeteilt. Damit besteht die Regel–Matrix also aus 10201 vorberechneten Stellgrößen. Da es sich bei der verwendeten AD/DA–Karte um 12–Bit Wandler handelt, wäre es zwar möglich für den Winkel und Omega jeweils 4095 Werte zu nehmen, aber die Regel–Matrix darf die Größe von 65kByte nicht überschreiten. Bei den 3D–Kennlinien im Kapitel Evolutionsstrategie handelt es sich um die grafische Darstellung einer $(41 * 41)$ Matrix.

Da im Regelprogramm nun nur noch die Stellgröße für einen bestimmten Winkel und eine bestimmte Winkelgeschwindigkeit aus der Regel–Matrix gelesen werden muß, ist eine sehr schnelle Regelung möglich. Für den Fall daß eine weitere Eingangsvariable eingeführt werden soll, zum Beispiel für die Position des Wagens, so muß für jeden diskreten Positionswert eine extra $(\varphi * \dot{\varphi})$ Matrix erzeugt werden. Eine 3–dimensionale

Matrix kann nicht verwendet werden, weil diese die 65kByte Grenze sehr schnell überschreitet.

5.1 Anbindung der Hardware

Der Hardwareaufbau wurde weitgehend von [11] und [12] übernommen. Da der Wagen aber nicht vom vorhandenen Regler angesteuert werden sollte, wurde dieser abgekoppelt. Dies geschah durch das Entfernen des Verstärkers zwischen Regler und Motor des Systems.

Die Ansteuerung des Antriebsmotors, der über einen Zahnriemen den Wagen bewegt, erfolgt über einen bipolaren Leistungsverstärker (**Kepco BOP 36-6M**) dessen Eingangssignal direkt von der AD/DA–Karte kommt.

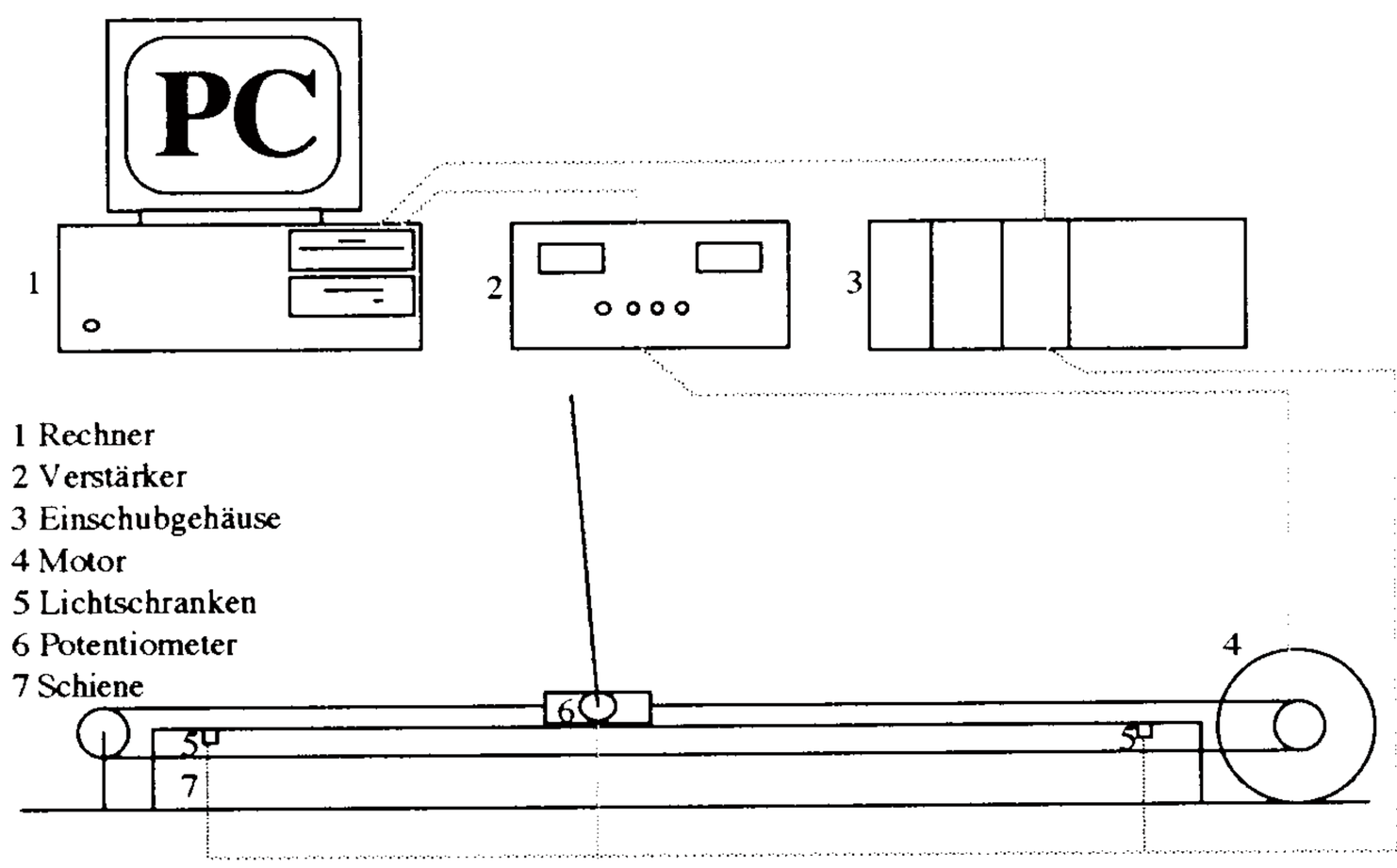

Abbildung 5.1: Übersicht des Hardwareaufbaus

5.1.1 Die analogen und digitalen Signale

Von der Fuzzy–Logik wird für die Stellgröße ein Wert zwischen 0 und 4095 geliefert. Dieser wird vom DA–Wandler in eine Spannung von 0 bis $10V$ umgewandelt. Der bipolare Verstärker ist so eingestellt, daß die Ausgangsspannung von $-20V$ bis $+20V$ reicht und somit einen Rechts– und Linkslauf des Motors ermöglicht.

Der Stabwinkel wird mit einem Präzisions–Potentiometer gemessen und mit Hilfe einer Widerstandsmeßbrücke in eine Spannung umgewandelt. Diese Spannung wird zum einen an Pin 24 von Reglerplatine 3 (Bild A.2) abgegriffen und zum AD–Wandler geführt. Außerdem wird hinter dem differenzierenden Teil der Schaltung an Pin 6 von IC4 die Spannung abgegriffen die $\dot{\varphi}$ entspricht.

Da das Programm auch reagieren soll wenn der Wagen ein Ende der Schiene erreicht, oder die Winkelbegrenzung auslöst, müssen auch die Signale der Lichtschranken und der Winkelbegrenzung dem Rechner zugeführt werden. Die TTL–Signale der beiden Lichtschranken werden an Pin 20 und Pin 24 von Platine 2 (Bild A.1) abgegriffen . Das Signal für die Winkelbegrenzung wird durch die über ein UND–Gatter geschalteten Signale von Pin 12 und Pin 13 der Platine 2 erzeugt.

In der Diplomarbeit [11] sind die Schaltungen ausführlich dokumentiert. Da in dem entsprechenden Anhang auch die Schaltpläne und Bestückungspläne enthalten sind, wird in dieser Arbeit nicht weiter darauf eingegangen.

5.1.2 Ansteuerung der AD/DA Karte

Die Ansteuerung der AD/DA Karte erfolgt mit der externen `procedure pcl718`. Dieser wird eine Funktionsnummer und eine entsprechende Anzahl Parameter übergeben.

Zuerst wird `pcl718` mit der Funktionsnummer 0 aufgerufen, mit den Parametern zur Initialisierung der Karte. So wird dabei z.B. der Adressbereich eingestellt, indem die Karte alle Werte einliest bzw. ausgibt. Die meisten wichtigen Parameter wurden jedoch direkt mit Jumpern auf der Karte eingestellt. Als Beispiel sei hier die Referenzspannung für den DA–Wandler genannt, die von einer externen Kalibrierquelle stammt. Diese beträgt $10V$, wogegen die interne Referenzspannung nur $5V$ beträgt. Eine Referenzspannung von $5V$ hätte eine maximale Ausgangsspannung des DA–Wandlers von ebenfalls $5V$ bedeutet, die aber als Eingangsspannung des Leistungsverstärkers nicht ausreichend wäre. Denn dieser muß wiederrum für die größte Stellgröße des Reglers eine Ausgangsspannung von $ca. \pm 20V$ erzeugen.

Nach jedem Aufruf von `pcl718` wird überprüft ob ein Fehler aufgetreten ist. Sollte das der Fall sein wird eine entsprechende Meldung ausgegeben.

5.2 Das Regelungs–Programm

Das Regelungs–Programm ist sehr kurz, da keine Rechnungen zur Regelung notwendig sind. Es wird hauptsächlich die `procedure pcl718` aufgerufen, die eingelesenen Werte mit einigen Programmschritten ausgewertet und die aus der Regelmatrix ausgelesene Stellgröße an den DA–Wandler weitergegeben.

Bevor dies geschieht,muß die Stellgröße an das Anlaufverhalten des Motors angepaßt werden. In Bild(5.2 a) ist die Kennlinie des Motors dargestellt. Um den Bereich um den Nullpunkt auszugleichen, wird die Stellgröße **u** korrigiert. Dazu wird bei $u\rangle 2048$ dieser um 500 erhöht. Bei $u\langle 2048$ werden 500 abgezogen. Soll der Wagen stehen,also bei $u = 2048$, so ändert der Wert sich nicht. Das Anlaufverhalten des Motors entspricht dann etwa Bild(5.2 b). Die Korrektur verschiebt also den linearen Teil der Kennlinie näher an den Nullpunkt.

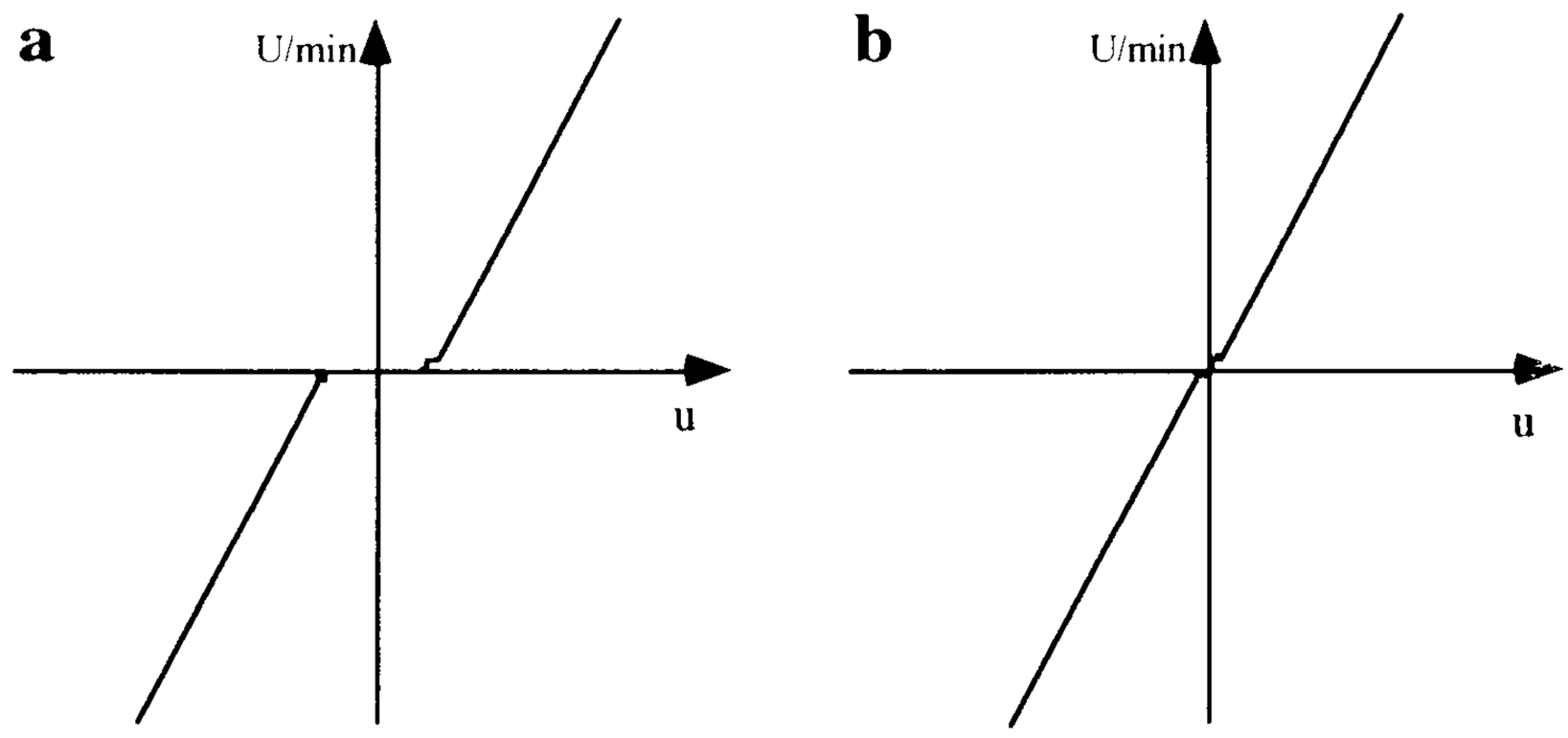

Abbildung 5.2: Motor–Kennlinie

Tritt beim Aufruf von `pcl718` ein Fehler auf, so wird an einigen Stellen dafür gesorgt, daß die Stellgröße auf 2048 gesetzt wird. Dadurch wird erreicht, daß bei einem Fehler der Motor abgeschaltet wird.

Nach dem Start des Programms muß die Winkelabweichung eingegeben werden. Unter Winkelabweichung ist der Wert zu verstehen, der entsteht wenn der gemessene Winkelwert von seinem Sollwert abweicht. Steht der Stab senkrecht, so sollte der Winkel gleich null sein. Wird vom AD–Wandler jedoch z.B. eine 197 (Intergerwert des Wandlers und nicht grad) erzeugt, so beträgt die Winkelabweichung 197. Um nun diesen Wert zu erhalten, muß der Stab von Hand in eine senkrechte Position gebracht werden. Danach wird daß Hilfsprogramm `Offset` gestartet. Am Bildschirm wird nun der Winkelwert und der Omegawert angezeigt. Da in einer Schleife fortlaufend gemessen wird, ändern sich durch Meßfehler auch die angezeigten Werte geringfügig. Der jeweils am häufigsten auftretende Wert wird dann im anschließend gestartetem Regelprogramm eingegeben.

Die einzige Rechnung, die in diesem Programm auftaucht , ist die Umrechnung der gemessenen Werte vom Winkel und Omega in den jeweiligen Matrix–Index. Diese Umrechnung erfolgt nach folgenden Gleichungen:

$$Winkelindex = round(\frac{Winkelmeß}{(\frac{winkelgrenze}{(\frac{zwischenwerte-1}{2})})} + \frac{zwischenwerte - 1}{2} + 1)$$

$$Omegaindex = round(\frac{Omegameß}{(\frac{omegagrenze}{(\frac{zwischenwerte-1}{2})})} + \frac{zwischenwerte - 1}{2} + 1)$$

Dabei ist Winkelgrenze (Omega-) der im Programm `Stab-0x` festgelegte Bereich (± 2048) für den die Stellgröße berechnet wurde.Zwischenwerte ist die Konstante die die Größe der Regelmatrix (z.B. $41 * 41$) angibt.

Wurde die Stabregelung gestartet, macht sich sofort die Drift des Wagens zu einer Seite der Schiene bemerkbar. Dieses ist dadurch zu erklären, daß nur der Stabwinkel

und nicht die Wagenposition ausgeregelt wird.

Wird dem Stab ein kleiner Stoß in Richtung Schienenmitte versetzt, bevor der Wagen das Schienenende erreicht, dann fährt dieser wieder in Richtung Schienenmitte.

Stabregelung — Hauptprogramm des Regelprogramms

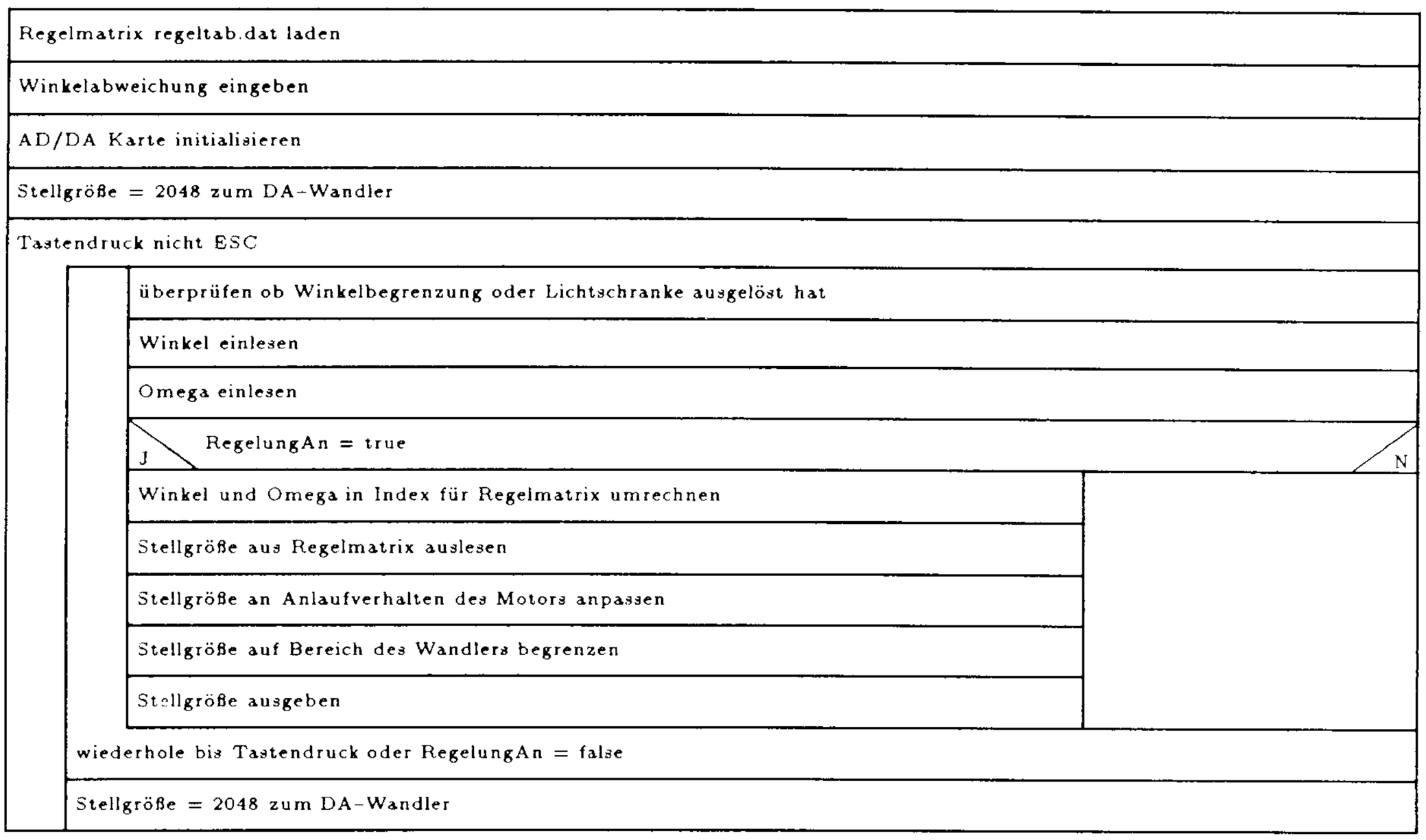

Abbildung 5.3: Struktogramm Regelprogramm

Abbildungsverzeichnis

2.1 Stab-Wagen-System 5
2.2 Kräfteverteilung 6
2.3 Struktogramm Procedure Y-Werte 10
2.4 Struktogramm Procedure Initmatrix 11
2.5 Struktogramm Procedure mul1 12
2.6 Struktogramm Procedure mul2 12
2.7 Struktogramm Procedure mul3 12
2.8 Struktogramm Procedure mul4 13
2.9 Struktogramm Procedure mul5 13
2.10 Struktogramm Procedure add1 13
2.11 Struktogramm Procedure add4 13
2.12 Struktogramm Procedure R-matrix 14
2.13 Struktogramm Procedure Stabzeichnen 16
2.14 Darstellung der Simulation auf dem Monitor 18

3.1 Grundprinzip eines Fuzzy–Reglers 19
3.2 Beispiele für Mitgliedschaftsfunktionen 20
3.3 Überlappungen 21
3.4 Linguistische Variable 'Wagen' 22
3.5 Linguistische Variable 'Winkel' 22
3.6 Linguistische Variable 'Omega' 22
3.7 UND Operator 23
3.8 ODER Operator 24
3.9 GAMMA Operator 24
3.10 Prämissenberechnung 25

3.11 Max-Prod Inferenz 27
3.12 Defuzzifizierung 27
3.13 Struktogramm Regelprüfen 28
3.14 Struktogramm Inferenz 29
3.15 Struktogramm MaxProd 29
3.16 Struktogramm Welchezugehörigkeit 30

4.1 Mutation eines Fuzzy-Sets 32
4.2 Struktogramm Optimieren 33
4.3 Struktogramm Naechsteskind 33
4.4 Struktogramm zufallaendern 35
4.5 Linguistische Variablen vor Optimierung 36
4.6 „Wagen“nach Optimierung 36
4.7 „Winkel“nach Optimierung 37
4.8 „Omega“nach Optimierung 37
4.9 Zeitverlauf des Winkels vor Optimierung 37
4.10 Zeitverlauf des Winkels nach Optimierung 38
4.11 Optimierungsverlauf 38
4.12 Kennlinienfeld vor Optimierung 39
4.13 Kennlinienfeld nach Optimierung 39
4.14 Differenz zwischen den Kennlinien (3D) 40
4.15 Differenz zwischen den Kennlinien (Density) 41
4.16 Auswahl der falschen Regeln 42
4.17 Kennlinienfeld nach Regelkorrektur (vor Optimierung) 44
4.18 Differenzfeld 2 45
4.19 Differenzfeld 3 46
4.20 „Wagen“nach Optimierung und Regelkorrektur 46
4.21 „Winkel“nach Optimierung und Regelkorrektur 47
4.22 „Omega“nach Optimierung und Regelkorrektur 47
4.23 Kennlinienfeld nach Regelkorrektur (nach Optimierung) 48
4.24 Kennlinienfeld ohne Lücke 48
4.25 Differenzfeld 4 49
4.26 Zeitverlauf des Winkels nach Regelkorrektur (vor ES) 49

4.27 Zeitverlauf des Winkel nach Regelkorrektur (nach ES) 50
4.28 Optimierungsverlauf 2 50
4.29 Flußdiagramm der Optimierung 51

5.1 Übersicht des Hardwareaufbaus 53
5.2 Motor–Kennlinie 55
5.3 Struktogramm Regelprogramm 56

A.1 Reglerplatine 2 94
A.2 Reglerplatine 3 95

Tabellenverzeichnis

2.1 Vereinbarung der Setbezeichnungen 17

3.1 Regeln des Stab-Wagen-Systems 26

4.1 Regelbasis der Stellgröße für 7 Sets pro Variable 34
4.2 Regelbasis nach Korrektur 43

Literaturverzeichnis

[1] Jürgen Ackermann. Abtastregelung. Springer-Verlag 1972

[2] Bandemer,Gottwald. Einführung in Fuzzy–Methoden. Akademie Verlag. 1992

[3] Helmut Kopka. LaTeX. Eine Einführung. Addison-Wesley GmbH 1990

[4] Helmut Kopka. LaTeX. Erweiterungsmöglichkeiten. Addison-Wesley GmbH 1992

[5] Föllinger,Franke. Einführung in die Zustandsbeschreibung dynamischer Systeme. Oldenburg Verlag 1982

[6] Thomas Tilli. Fuzzy-Logik. Francis-Verlag 1991

[7] Constantin von Altrock. Über den Daumen gepeilt. ct 3/1991

[8] Ingo Rechenberg. Evolutionäre Wege in die Zukunft. Evolutionsstrategie–Optimierung nach Prinzipien der biologischen Evolution. Beltz Verlag

[9] Goser,Surmann. Clevere Regler schnell entworfen. Elektronik 6/1992

[10] Thomas Blümecke. Wunder der Evolution. ct 12/1991

[11] Schlingmann,Nießer. Diplomarbeit. Aufbau und Realisierung einer Regelung für einen balancierten Stab. 1990

[12] Diplomarbeit. Goerke,Schumacher. Diplomarbeit. Untersuchung zum Aufbau eines Beobachters für das Modell des balancierten Stabes. 1990

[13] Kandel,Li,Cao. Fuzzy Inference and its applicability to control systems. Fuzzy Sets and Systems 48. 1992

[14] Bossel. Simulation dynamischer Systeme. Vieweg Verlag.

[15] Günther Trautzl. Mit Fuzzy–Logik näher zur Natur ?. Elektronik 9,10,16/1991

[16] Edwin Bühler. Studienarbeit. Programmierung eines Fuzzy–Editors. 1992

[17] Daniel Gariglio. Fuzzy in der Praxis. Elektronik 20/1991

[18] Zadeh. Fuzzy Sets. Information And Control. 8/1965.