

# Software-Werkzeug zur Auslegung von robusten PID-Reglern

am  
Lehrstuhl für  
Steuerungs- und Regelungstechnik  
der Technischen Universität München  
Professor Dr.-Ing./Univ. Tokio Martin Buss  
eingereichte

DIPLOMARBEIT

des  
cand. ing. Thomas Hulin

Betreuer: Professor Dr.-Ing. J. Ackermann, N. Bajcinca,  
Dr. F. Freyberger, Dr. D. Kaesbauer  
Beginn: 17. Februar 2003  
Zwischenbericht: 25. Juni 2003  
Abgabe: 19. September 2003  
Verzeichnis-Nr.:



## **Software-Werkzeug zur Auslegung von robusten PID-Reglern**

Thomas Hulin

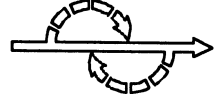
Einstellregeln für die drei Parameter von PID-Reglern sind im Allgemeinen nur auf Strecken bestimmten Typs anwendbar und erlauben zumeist keine exakte Berücksichtigung von Totzeiten oder Parameterunsicherheiten. Neue theoretische Erkenntnisse, basierend auf der Grundlage der Methode der singulären Frequenzen, ermöglichen die Bestimmung eines robust stabilisierenden dreidimensionalen Gebietes im Raum der PID-Reglerparameter. Damit können Systeme mit Parameterunsicherheiten, einschließlich von totzeitbehafteten Systemen untersucht werden. Diese Verfahren lassen neben Hurwitz- und  $\sigma$ -Stabilität auch die Behandlung von der besonders für zeitdiskrete Systeme nützlichen Kreisstabilität zu, ebenso die Feineinstellung von Reglerparametern. Diese Arbeit fasst die neuesten Erkenntnisse dieser Thematik zusammen und stellt ein darauf aufbauendes Software-Werkzeug für den Regelungstechniker vor. Das Tool ermöglicht über eine einfach zu bedienende Benutzeroberfläche die Eingabe von Regelstrecken mit Parameterunsicherheiten, die Auswahl eines Reglers und  $\Gamma$ -Gebietes, die Reglersynthese, die Visualisierung des berechneten, stabilen Gebietes, die Auswahl eines Satzes an Reglerparametern sowie die interaktive Analyse mit den gebräuchlichen Analyseverfahren.

## **Software Tool for Synthesis of Robust PID Controllers**

Thomas Hulin

Generally, tuning rules of PID controllers are limited to systems of certain classes. Moreover, they permit no accurate consideration of time-lag or parameter uncertainties. The latest research results based on the method of singular frequencies approach make possible the determination of the robust stabilizing three-dimensional region(s) in the coordinates of PID controller parameters for dead-time systems with parameter uncertainties. These techniques apply in addition to Hurwitz also for  $\sigma$ - and circle-stability. The latter is of particular importance for time-discrete systems. Three-term controller fine-tuning is feasible, as well. This work summarizes the latest results on this topic and introduces a software tool for control engineers. The tool supports the following functionalities via a graphical user interface: the input systems with dead-time and parameter uncertainties, the selection of a controller and a  $\Gamma$  region, the controller synthesis, the visualization of the computed stable region, the selection of a set of controller parameters, as well as the interactive analysis with the common analysis methods.





DIPLOMARBEIT  
gemeinsam mit  
DLR, Institut für Robotik und Mechatronik, Prof. Dr. J. Ackermann  
für  
Herrn Thomas Hulin  
Matriel-Nr.: 2054770, Studienrichtung D

### Software-Werkzeug zur Auslegung von robusten PID-Reglern

#### Umfeld und Ziel der Arbeit:

In [1] wurde ein neues Verfahren zur Auslegung robuster PID-Regler aus [2] auf Totzeitsysteme erweitert. Die praktische Anwendung dieser Verfahren hängt entscheidend davon ab, dass ein einfach zu benutzendes Software-Werkzeug für den Praktiker zur Verfügung steht. Auf der Basis von Matlab soll ein Prototyp eines interaktiven grafischen Software-Werkzeugs (GUI) zur Auslegung von robusten PID-Reglern erstellt und auf Beispiele aus der Literatur angewendet werden. Der Prototyp soll Interessenten über das Internet kostenlos verfügbar gemacht werden.

#### Anforderungen:

Eingabe der Regelstrecke in Form einer parametrischen Übertragungsfunktion, ggf. mit Totzeit. Eingabe der Minimal- und Maximalwerte der Parameter und der Totzeit. Auswahl von Repräsentanten aus dem Arbeitsbereich.

Alternativ: Nur Eingabe von Repräsentanten.

Eingabe von  $\sigma_{\max}$ , das ist der maximale Realteil aller Eigenwerte des geschlossenen Kreises als Spezifikation der Einstellzeit ( $\sigma$ -Stabilität).

Berechnung der Menge der simultan (d.h. für alle gewählten Repräsentanten)  $\sigma$ -stabilisierenden PID-Regler in Form eines dreidimensionalen  $\sigma$ -Stabilitätsgebiets im Raum der drei Reglerparameter  $K_p$ ,  $K_I$  und  $K_D$ . Benutzerunterstützung bei der zweidimensionalen Betrachtung dieses Bildes.

Alternativ: Festlegung von  $K_p$  durch den Benutzer.

#### Optionen sind

- automatische  $\sigma_{\max}$ -Minimierung, und
- iterative Eingabe von  $\sigma_{\max}$  zur Kompromissfindung mit anderen Spezifikationen (z. B. Phasenreserve, maximales Überschwingen, Stabilitätsreserve im Raum der Streckenparameter). Hierzu sind Schnittstellen zu Analyseprogrammen in Matlab, PARADISE erforderlich.

Das Software-Werkzeug soll insbesondere ein schnelles interaktives Arbeiten ermöglichen und dem Benutzer die automatisierbaren Zwischenschritte des Entwurfs (z. B. Detektion der  $\sigma$ -stabilen Polygone, Bestimmung des  $K_p$ -Intervalls) abnehmen.

#### Literatur:

- [1] N. Hohenbichler, Auslegung robuster PID-Regler für Totzeitsysteme. TU München Lehrstuhl für Steuerungs- und Regelungstechnik, Diplomarbeit, 2002.
- [2] J. Ackermann, P. Blue, T. Bünte, L. Güvenc, D. Kaesbauer, M. Kordt, M. Muhler and D. Odenthal, Robust Control, Springer-Verlag, London, 2002.

#### Betreuung:

Beim DLR: Prof. Dr. J. Ackermann in Zusammenarbeit mit Dr. Kaesbauer und Herrn Bajcinca.  
Am Lehrstuhl: Dr. Freyberger

Beginn: 17. Februar 2003  
1. Zwischenbericht: 16. Mai 2003  
Abgabe: 17. August 2003  
Abschlussvortrag:

(J. Ackermann)  
Professor

(G. Schmidt)  
Professor



# Inhaltsverzeichnis

Symbolverzeichnis	ix
Einleitung	1
<b>1 Methode der singulären Frequenzen</b>	<b>5</b>
1.1 Grundlagen	5
1.2 Vorabtransformation von PID-Regelkreisen	6
1.3 Singuläre Frequenzen	8
1.4 Robustheit der Rangbedingung	9
1.5 Singuläre $\Gamma$ -Gebiete	10
1.6 Berechnung der singulären Frequenzen	14
1.7 Berechnung der singulären Geraden	16
1.8 Innere Polygone	18
1.9 Klassen geeigneter Eigenwertspezifikationen	23
1.10 Totzeit	26
1.11 Algorithmus	28
1.12 Entstehen/ Verschwinden von stabilen Polygonen	29
<b>2 Anleitung zur Bedienung von <i>RobSin</i></b>	<b>33</b>
2.1 Starten von <i>RobSin</i>	33
2.2 Aufbau der Benutzeroberfläche	34
2.3 Behandlung des Beispiels mit <i>RobSin</i>	41
2.4 Das <i>Options</i> -Fenster	56
2.5 Bedienung von <i>RobSin</i> über die Kommandozeile	58
<b>3 Anwendungsbeispiel</b>	<b>64</b>
3.1 Laplace-Bereich	64
3.2 Zeitdiskreter Bereich	71
<b>4 Software-Architektur und Funktionen</b>	<b>76</b>
4.1 Objektorientierte Programmierung	76
4.2 Kommunikation zwischen den Objekten	77
4.3 Ablegen der Objektdaten im Speicher	80

4.4	Die Klassen von <i>RobSin</i> . . . . .	81
4.5	Die Klasse <i>singf</i> . . . . .	81
4.6	Das Verzeichnis <i>util</i> . . . . .	82
<b>5</b>	<b>Zusammenfassung und Ausblick</b>	<b>85</b>
5.1	Schlussfolgerung über die Methode der singulären Frequenzen . .	85
5.2	Das Software-Werkzeug als Ergebnis . . . . .	86
5.3	Ausblick auf mögliche zukünftige Entwicklungen . . . . .	86
<b>A</b>	<b>Fehlerbehandlung</b>	<b>89</b>
A.1	Fehlermeldungen . . . . .	89
A.2	Fehler bei Berechnungen . . . . .	90
	<b>Literaturverzeichnis</b>	<b>92</b>

# Symbolverzeichnis

$\widetilde{(\dots)}$	Ausdruck mit beliebiger Anzahl an Reglerparametern
$\widehat{(\dots)}$	Modifizierter Ausdruck
$(\dots)^*$	Fester Wert eines Ausdrucks
$(\dots)^{-1}$	Inverse einer Matrix
$a, b, c$	Abkürzungen für einen Ausdruck abhängig von $\mathbf{T}$
$(\dots)_c$	Die Kreisstabilität betreffend
$\mathbf{c} = [c_0, c_1, c_2]^T$	Reelle Koeffizienten von $C(z)$
$d(\cdot)$	Polygon im Zähler der Three-Term-Control-Struktur
$e$	Vorzeichenfunktion
$g(\cdot), g_H(\eta)$	Determinanten
$i$	Positive, ganzzahlige Zählvariable
$j$	Imaginäre Einheit
$k$	Federkonstante
$\mathbf{k} = [k_I, k_P, k_D]^T$	Parameter des PID-Reglers
$l$	Positive, ganzzahlige Zählvariable
$m$	Reeller Mittelpunkt des Kreises für Kreisstabilität
$m_1, m_2$	Massen
$n_p$	Anzahl der Pole des offenen Regelkreises außerhalb des $\Gamma$ -Gebietes
$n_q$	Anzahl der Streckenparameter $\mathbf{q}$
$n_0$	Anzahl der Pole des offenen Regelkreises auf dem Rand des $\Gamma$ -Gebietes $\partial\Gamma$
$n(\cdot)$	Polygon im Nenner der Three-Term-Control-Struktur
$\vec{n}$	Gradient von $F$ , der impliziten Darstellung von $\partial\Gamma$
$p(s)$	Polynom in $s$
$\mathbf{p} = [p_0, p_1, p_2]^T$	Reelle Koeffizienten von $p(s)$
$\mathbf{q}$	Vektor der Streckenparameter ohne Totzeit
$r$	Radius des Kreises für Kreisstabilität
$\mathbf{r} = [r_0, r_1, r_2]^T$	Vektor der linear transformierten Reglerparameter

$\mathbf{r}^A(\cdot)$	Anfangspunkt einer Kante
$\mathbf{r}^E(\cdot)$	Endpunkt einer Kante
$s$	Komplexe Variable der Laplace-Transformation
$t_{\dots}$	Elemente der Matrix $\mathbf{T}$
$u_{\dots}$	Abkürzung für einen Ausdruck
$u(s)$	Ausgang des Reglers, Eingang der Regelstrecke
$x$	Positive, ganzzahlige Zählvariable
$y(s)$	Ausgang von Regelkreisen
$w(s)$	Eingang von Regelkreisen
$z$	Komplexe Variable der $z$ -Transformation
$z_s$	Singuläre Frequenz
$z(\alpha)$	Parametrisierungsfunktion von $\partial\Gamma$
$A(z, \mathbf{q})$	Zählerpolynom von $G(z, \mathbf{q})$
$B(z, \mathbf{q})$	Nennerpolynom von $G(z, \mathbf{q})$
$B(z, L, \mathbf{q})$	Nennerquasipolynom von $G(z, L, \mathbf{q})$ mit Totzeit $L$
$C(s)$	Reglerstruktur für zeitkontinuierliche Systeme
$C(z)$	Polynomialer Anteil der Reglers
$D$	Mindestdämpfung
$D(z)$	Reglerstruktur für zeitdiskrete Systeme
$\mathbf{E}$	Einheitsmatrix
$F$	Implizite Darstellung von $\partial\Gamma$
$G(z, \dots)$	Offener Regelkreis reduziert um $C(z)$ mit den Parametern $\mathbf{q}$
$G_i(z)$	Offener Regelkreis reduziert um $C(z)$
$\Im \{.\}$	Imaginärteil eines komplexen Ausdrucks
$I_{\dots}$	Imaginärteil eines Polynoms oder Quasipolynoms
$I_{\text{ges}}$	Intervall, in dem nach kritischen Frequenzen gesucht wird
$I^x$	Intervall zwischen zwei benachbarten $\alpha_E$
$J, J_{z_s}$	Determinanten
$L$	Totzeit
$N$	Anzahl der Kanten eines Polygons
$N(s, \mathbf{q})$	Nennerpolynom von $G(z, \mathbf{q})$ ohne Totzeit
$P_c = P(z, \mathbf{c}, \dots)$	Charakteristisches Polynom bezüglich den Parametern $\mathbf{c}$
$P_k$	Charakteristisches Polynom bezüglich den Parametern $\mathbf{k}$
$P_r$	Charakteristisches Polynom bezüglich den Parametern $\mathbf{r}$
$\mathcal{P}$	Matrix abhängig vom charakteristischen Polynom $P_k$
$Q$	Betriebsbereich
$\Re \{.\}$	Realteil eines komplexen Ausdrucks

$R_{\dots}$	Realteil eines Polynoms oder Quasipolynoms
$(\dots)^T$	Transponierte Matrix
$T_{\text{ein}}$	Einschwingzeit
$T_R$	Verzögerungszeitkonstante des PID-Reglers
$T_S$	Abtastzeit
$\mathbf{T}$	Transformationsmatrix zwischen $\mathbf{r}$ und $\mathbf{c}$
$\mathbf{T}_{a=0}$	Spezialfall von $\mathbf{T}$ für $\sigma$ -Stabilität
$\mathbf{T}_{a \neq 0}$	Spezialfall von $\mathbf{T}$ für Kreisstabilität
$\mathbf{T}_{\text{ges}}$	Transformationsmatrix zwischen $\mathbf{r}$ und $\mathbf{k}$
$\mathbf{T}_{\text{PID}, s}$	Transformationsmatrix zwischen $\mathbf{c}$ und $\mathbf{k}$ für zeitkontinuierliche Systeme
$\mathbf{T}_{\text{PID}, z}$	Transformationsmatrix zwischen $\mathbf{c}$ und $\mathbf{k}$ für zeitdiskrete Systeme
$\mathbf{Z}$	Matrix abhängig von $\tau$ und $\eta$
$\alpha$	Parameter für die Parametrisierung von $\partial\Gamma$
$\alpha_E$	Stelle eines Extremum von $r_1(\alpha)$
$\alpha_K$	Kritische Frequenz
$\alpha_S$	Singuläre Frequenz als Parameter $\alpha$
$\delta r_0, \delta r_2$	Infinitesimal kleine Bewegungen entlang der Achse $r_0$ bzw. $r_2$
$\delta(\cdot)$	Gerichtete Kante, die Teil einer Geraden ist
$\eta$	Imaginärteil von $z$
$\lambda(z_S)$	Singuläre Gerade für die singuläre Frequenz $s_Z$
$\vec{\mu}_0, \vec{\mu}_2$	Bewegungen in der komplexen Ebene
$\sigma$	Realteil von $s$
$\sigma_{\text{max}}$	Verschiebungsparameter des $\Gamma$ -Gebietes für $\sigma$ -Stabilität
$(\dots)_\sigma$	Die $\sigma$ -Stabilität betreffend
$\tau$	Realteil von $z$
$\tau_{\text{max}}$	Verschiebungsparameter des $\Gamma$ -Gebietes für $\sigma$ -Stabilität
$\varphi$	Winkel zwischen reeller Achse und Dämpfungsgeraden
$\omega$	Imaginärteil von $s$
$\Delta_i$	Abstand zweier Punkte entlang einer Achse
$\Delta I^k$	Teilintervall von $I_{\text{ges}}^k$
$\Delta\Phi$	Winkeländerung des Zeigers für das Nyquist-Kriterium
$\Gamma$	Bezeichnung des Gebietes für eine Eigenwertspezifikation in der komplexen Ebene
$\partial\Gamma$	Rand des $\Gamma$ -Gebietes
$\Pi$	Polygon



# Einleitung

Konventionelle PID-Regler sind in der Lage, das Verhalten der meisten Prozesse in der Industrie zu verbessern. Dies ist einer der Gründe dafür, dass PID-Regler seit bereits sechs Jahrzehnten das am häufigsten eingesetzte Reglerkonzept darstellen. Der Hauptgrund für die weite Akzeptanz dieses Reglers liegt jedoch in seiner einfachen Struktur, die sich in vielen praktischen regelungstechnischen Problemen als sehr robust gegenüber Störungen und Nichtlinearitäten bewiesen hat.

Trotz der Beliebtheit, widmen sich Wissenschaftler erst seit dem letzten Jahrzehnt verstärkt diesem Reglerkonzept. In dieser Zeit entstanden zahlreiche Denkansätze für die Bestimmung der drei Parameter von PID-Reglern. Der Autor von [O'D03] zählte dreihundertachtzig Veröffentlichungen über den Gebrauch von PI und PID Reglern für den Einsatz bei totzeitbehafteten Prozessen.

Umfragen in der Industrie ergaben jedoch ein ernüchterndes Ergebnis. Die Firma Techmation Inc. untersuchte tausend Regelkreise in einigen hundert Anlagen mit dem Resultat, dass mehr als 30% der eingesetzten Regler im Handbetrieb arbeiten und mindestens weitere 30% der Regelkreise schlecht eingestellt sind ([End93]). Ein Autor in [Bia96] berichtet von einem ähnlichen Ergebnis bei einer Untersuchung in der kanadischen Papierindustrie, bei der sich herausstellte, dass nur 20% der vorgefundenen Regler eine gute Funktionalität zeigten.

Diese Umfragen stützen die Feststellung des Autors von [DHB00], dass sich im Bereich der Regelungstechnik seit Ende der 50er Jahre ein bedeutender Spalt zwischen Theorie und Praxis aufgetan hat. Der Grund dafür liegt zum einen darin, dass Forschungsergebnisse zu dem Thema schwer zugänglich und weit verstreut in der Fachliteratur vorliegen, zum anderen ist die verwendete Notation in diesen Arbeiten nicht einheitlich ([O'D03]).

Software-Werkzeuge können den Regelungstechniker bei der Einstellung von optimalen Reglerparametern unterstützen und helfen diesen Spalt zu schließen. Es existiert bereits eine Vielzahl solcher Hilfsmittel. Die beiden Internetseiten [Man03] und [MW003] geben eine Übersicht über Software aus dem Bereich der Regelungstechnik, mit der ein guter Satz an Reglerparametern gefunden werden kann.

Vor einigen Jahren begannen einige Wissenschaftler, anstelle nach einem optimalen Regler zu suchen, sich für das Gebiet aller stabilisierenden Reglerparameter zu interessieren. Um einen Satz an Reglerparametern zu finden, wird für

gewöhnlich ein Regelkreis nach bestimmten Kriterien optimiert. Diese Optimierung findet normalerweise auf Kosten anderer Qualitätsmerkmale statt, welche bei der Optimierung nicht beachtet wurden. Aus diesem Grund kann es extrem nützlich sein, alle stabilisierenden PID-Regler zu kennen.

In den Anfängen dieser neuen Richtung konnten die Stabilitätsgrenzen im Raum der Reglerparameter oft nur aufwändig bestimmt werden. Vor ein paar Jahren zeigten die Autoren in [DHB00], dass das Stabilitätsgebiet für PID-Regler mit festem proportionalen Anteil  $k_P$  aus konvexen Polygonen besteht. Dies bringt einen entscheidenden Vorteil gegenüber den früheren Ergebnissen – die Bestimmung von Polygonen ist bedeutend einfacher und schneller. Der Beweis dieses Ansatzes führte über ein modifiziertes Hermite-Biehler Theorem.

Das Buch [ABB<sup>+</sup>02] verwendet zur Herleitung dieses Ergebnisses das Parameterraumverfahren. Es stellte sich heraus, dass die Wurzeln des charakteristischen Polynoms eines geschlossenen Regelkreises, bei einer Veränderung der nicht fixierten Reglerparameter  $k_I$  und  $k_D$ , eine Parallele zur imaginären Achse ausschließlich bei einer singulären Frequenz überqueren können.

In [Baj01], [BKA02] und [Hoh02] wird diese Theorie auf totzeitbehaftete Systeme erweitert. Für die automatische Erkennung von stabilen Polygonen stellt [Baj01] einen Algorithmus vor, der die Entwicklung eines Software-Werkzeuges erst ermöglichte. Die Diplomarbeit [Hoh02] ergänzt die Analyse im Parameterraum für Totzeitsysteme und zeigt die Überlegenheit dieses Verfahrens gegenüber vielen anderen Einstellregeln.

Der Konferenzbeitrag [AKB02] untersucht mit dem Ansatz der singulären Frequenzen die zeitdiskrete Regelung und stellt dazu eine Möglichkeit zur gleichzeitigen Einstellung von Dämpfung und Einschwingzeit vor. Die praktische Anwendung dieses Verfahrens hängt entscheidend davon ab, dass ein einfach zu benutzendes Software-Werkzeug für den Praktiker zur Verfügung steht.

Der Autor von [Baj01] entwickelte, aufbauend auf all diesen theoretischen Fortschritten, ein MATLAB<sup>®</sup>-Programm mit integriertem Maple<sup>®</sup>-Code zur Berechnung des Gebietes aller stabilisierenden Regler, welches per Eingabe über die Kommandozeile mit dem Anwender kommuniziert. Die Darstellung dieses Gebietes erfolgt in einer dreidimensionalen Grafik. Der Name dieses Software-Werkzeuges, *RobSin*, leitet sich ab von „robust design based on singular frequencies“. Diesem Tool fehlte für den Einsatz als nützliches Hilfsmittel eine einfach zu bedienende Benutzeroberfläche, welche im Rahmen dieser Diplomarbeit entstand. Die Internetseite [DLR03] stellt das Software-Werkzeug *RobSin* kostenlos zum Herunterladen bereit.

Die vorliegende Arbeit führt den Leser in die hinter dem Software-Werkzeug stehende Theorie ein und erklärt ihm die Bedienung und Funktionsweise dieses Tools.

Das erste Kapitel stellt die Methode der singulären Frequenzen vor. Sie repräsentiert das Verfahren zur Bestimmung des Gebietes aller stabilisierenden Reglerparameter.

Im zweiten Kapitel wird der Regelungstechniker Schritt für Schritt in die Bedienung von *RobSin* eingeführt. Es erklärt anhand eines Beispiels sowohl die Handhabung der Benutzeroberfläche, als auch die Bedienung über die Kommandozeile von MATLAB®.

Unter Behandlung eines weiteren Beispiels geht das dritte Kapitel auf verschiedene regelungstechnische Aspekte ein. So beschreibt es unter anderem die Feineinstellung von Reglerparametern und die gleichzeitige Einstellung von Dämpfung und Einschwingzeit.

Besonders interessant ist Kapitel vier für den Softwareentwickler. Es geht auf verschiedene programmiertechnische Fragestellungen ein und gibt einen Überblick über verwendete Objekte und wichtige Methoden und Funktionen.

Abschließend befindet sich im fünften Kapitel eine Zusammenfassung dieser Arbeit. Weiterhin legt dieses Kapitel dar, welche Fortschritte in der Theorie nötig sind, um den Funktionsumfang des Software-Werkzeuges zu erweitern.

Der Anhang beinhaltet Lösungsvorschläge für Probleme, die in Verbindung mit *RobSin* auftreten.



# Kapitel 1

## Methode der singulären Frequenzen

Für die Berechnung des Gebietes aller robust stabilisierenden Reglerparameter verwendet das im Rahmen dieser Diplomarbeit weiter entwickelte Software-Werkzeug *RobSin* die *Methode der Singulären Frequenzen* aus den Arbeiten [AK00], [Baj01] und [AKB02]. Dieses Kapitel stellt diese Methode vor und erarbeitet die weiteren theoretischen Grundlagen, die für eine automatische Berechnung des dreidimensionalen Gebietes aller robust stabilisierenden Reglerparameter notwendig sind.

Die in diesem Kapitel vorgestellten Berechnungen sind für zeitkontinuierliche und für zeitdiskrete Systeme größtenteils identisch. Die komplexe Variable wird daher frei gewählt zu  $z$ . Einige Abschnitte benötigen jedoch die Unterscheidung zwischen zeitkontinuierlichen und zeitdiskreten Systemen. An diesen Stellen ersetzt für zeitkontinuierliche Systeme die Laplace-Variable  $s$  die komplexe Variable  $z$ .

### 1.1 Grundlagen

Die Struktur des in diesem Kapitel verwendeten, umstrukturierten Regelkreises stellt Abbildung 1.1 dar. Der betrachtete Anteil des Reglers  $C(z)$  ist polynomial

$$C(z) = c_0 + c_1z + c_2z^2, \quad (1.1)$$

mit den Reglerparametern  $\mathbf{c} = [c_0, c_1, c_2]^T$ . Die angepasste Regelstrecke lautet

$$G(z, \mathbf{q}) = \frac{A(z, \mathbf{q})}{B(z, \mathbf{q})}. \quad (1.2)$$

Dabei sind  $A(z, \mathbf{q})$  und  $B(z, \mathbf{q})$  reelle Polynome in  $z$ , deren Koeffizienten kontinuierlich von den unsicheren Streckenparametern  $\mathbf{q}$  abhängen. Die Parameter

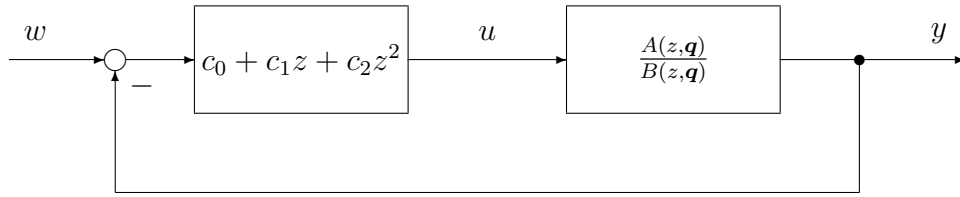


Abbildung 1.1: Umstrukturierter PID-Regelkreis.

$\mathbf{q} = [q_1, \dots, q_{n_q}]^T$  variieren zwischen den unteren und oberen Schranken,  $q_i^-$  und  $q_i^+$ . Der Betriebsbereich  $Q$  beschreibt dann ein Hyperrechteck,

$$Q = \{ \mathbf{q} \mid q_i \in [q_i^-; q_i^+], i = 1, \dots, n_q \}, \quad (1.3)$$

welches auch als *Q-Box* bezeichnet wird ([ABB<sup>+</sup>02]).

Abschnitt 1.4 zeigt, dass auch die Behandlung einer Familie von endlich vielen Strecken  $G(z, \mathbf{q}) = \{G_i(z)\}$ , mit  $i \in \{1, 2, 3, \dots\}$ , möglich ist. Die Regelstrecke  $G(z, \mathbf{q})$  wird zunächst als totzeitlos angenommen. Abschnitt 1.10 erweitert die Berechnungen auf totzeitbehaftete Systeme.

Die in diesem Kapitel vorgestellte Methode hat zwei Anwendungsgebiete. Zum einen eignet sie sich zur Bestimmung des dreidimensionalen Gebietes aller stabilisierenden Reglerparameter  $\mathbf{c}$ . Zum anderen erlaubt sie eine Feineinstellung (engl.: *fine-tuning*) von Parametern des Reglers (siehe Abschnitt 3.1.3). Diese ist jedoch dadurch begrenzt, dass sich die betrachteten Parameter im geschlossenen Regelkreis als Polynom zweiten Grades (1.1) abspalten lassen müssen, so dass sich die in Abbildung 1.1 gezeigte Struktur ergibt.

Die Methode der singulären Frequenzen eignet sich besonders für PID-Regler. Der polynomiale Anteil eines Reglers (1.1) ist gleichwertig mit einem PID-Regler, da sich diese beiden Strukturen durch eine lineare Transformation zusammen mit einer Multiplikation ineinander überführen lassen.

## 1.2 Vorabtransformation von PID-Regelkreisen

Die Vorabtransformation bringt PID-Regelkreise in die Struktur aus Abbildung 1.1. Sie ist eine Kombination aus linearer Transformation der Reglerparameter  $\mathbf{c}$  und Multiplikation der Regelstrecke  $G(z, \mathbf{q})$ . Jedoch unterscheidet sie sich für zeitkontinuierliche und zeitdiskrete Systeme. Liegt der zu untersuchende Anteil eines Regler bereits in der polynomialen Form (1.1) vor, so entfällt diese Transformation.

### 1.2.1 Zeitkontinuierliche Systeme

Für zeitkontinuierliche Systeme betrachte man PID-Regler von der Form

$$k_P + \frac{k_I}{s} + \frac{k_D s}{T_R s + 1}. \quad (1.4)$$

Dabei steht  $T_R$  für die Verzögerungszeitkonstante des PID-Reglers. Die Multiplikation von (1.4) mit dem Faktor  $s(T_R s + 1)$  ergibt

$$\begin{aligned} \left( k_P + \frac{k_I}{s} + \frac{k_D s}{T_R s + 1} \right) \cdot s(T_R s + 1) &= \\ &= \underbrace{k_I}_{c_0} + \underbrace{(k_P + k_I T_R)}_{c_1} s + \underbrace{(k_D + k_P T_R)}_{c_2} s^2 = \\ &= c_0 + c_1 s + c_2 s^2. \end{aligned} \quad (1.5)$$

Daraus folgt die Transformationsmatrix  $\mathbf{T}_{\text{PID},s}^{-1}$  für zeitkontinuierliche Systeme von den Koordinaten  $\mathbf{k} = [k_I, k_P, k_D]^T$  des PID-Reglers zu den Koordinaten  $\mathbf{c}$  des polynomialen Anteils eines Reglers zweiter Ordnung

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ T_R & 1 & 0 \\ 0 & T_R & 1 \end{bmatrix}}_{\mathbf{T}_{\text{PID},s}^{-1}} \begin{bmatrix} k_I \\ k_P \\ k_D \end{bmatrix}. \quad (1.6)$$

Die Inverse dieser Matrix ist die Transformationsmatrix  $\mathbf{T}_{\text{PID},s}$

$$\mathbf{T}_{\text{PID},s} = \left( \mathbf{T}_{\text{PID},s}^{-1} \right)^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -T_R & 1 & 0 \\ T_R^2 & -T_R & 1 \end{bmatrix}. \quad (1.7)$$

Der Übergang zu der in Abbildung 1.1 gezeigten Struktur umfasst demnach die Transformation der Reglerparameter  $\mathbf{k} = \mathbf{T}_{\text{PID},s} \mathbf{c}$  und die Multiplikation der Regelstrecke mit  $1/s(T_R s + 1)$ . Weiterhin wird  $s$  durch  $z$  ohne zusätzliche Umrechnung ersetzt.

### 1.2.2 Zeitdiskrete Systeme

Für zeitdiskrete Systeme besitze der PID-Regler die Struktur

$$k_P + \frac{k_I}{1 - z^{-1}} + k_D(1 - z^{-1}). \quad (1.8)$$

Folgende Berechnungen zeigen die Multiplikation von Gleichung (1.8) mit  $z^2(1 - z^{-1})$ , welche auf die Transformationsmatrix  $\mathbf{T}_{\text{PID},z}$  für zeitdiskrete Systeme von

den Koordinaten  $\mathbf{c}$  des polynomialen Anteils eines Reglers zweiter Ordnung zu den Koordinaten  $\mathbf{k}$  des PID-Reglers führen.

$$\begin{aligned} & \left( k_P + \frac{k_I}{1-z^{-1}} + k_D(1-z^{-1}) \right) \cdot z^2(1-z^{-1}) = \\ & = \underbrace{k_D}_{c_0} + \underbrace{(-k_P - 2k_D)}_{c_1} z + \underbrace{(k_I + k_P + k_D)}_{c_2} z^2 = \end{aligned} \quad (1.9)$$

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & -2 \\ 1 & 1 & 1 \end{bmatrix}}_{\mathbf{T}_{\text{PID},z}^{-1}} \begin{bmatrix} k_I \\ k_P \\ k_D \end{bmatrix} \quad (1.10)$$

$$\mathbf{T}_{\text{PID},z} = \left( \mathbf{T}_{\text{PID},z}^{-1} \right)^{-1} = \begin{bmatrix} 1 & 1 & 1 \\ -2 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix}. \quad (1.11)$$

Analog zum vorherigen Unterabschnitt umfasst der Übergang zu der in Abbildung 1.1 gezeigten Struktur zwei Schritte, die Transformation der Reglerparameter  $\mathbf{k} = \mathbf{T}_{\text{PID},z} \mathbf{c}$  und die Multiplikation der Regelstrecke mit  $1/z(z-1)$ .

### 1.3 Singuläre Frequenzen

Die weiterführenden theoretischen Herleitungen benötigen die Definitionen von *singulären Frequenzen* und von *singulären  $\Gamma$ -Gebieten*, welche in Analogie zu [Baj01] und [ABB<sup>+</sup>02] angegeben werden. Sie gelten auch für Polynome höherer Ordnung als zwei. (Diese sind im Folgenden durch eine Tilde gekennzeichnet.)

Für die Definition von singulären Frequenzen betrachte man die charakteristische Gleichung eines beliebigen Regelkreises

$$\tilde{P}(z, \tilde{\mathbf{c}}) = 0 \quad \triangleq \quad \begin{bmatrix} \Re\{\tilde{P}(z, \tilde{\mathbf{c}})\} \\ \Im\{\tilde{P}(z, \tilde{\mathbf{c}})\} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad (1.12)$$

wobei die Komponenten des Parametervektors  $\tilde{\mathbf{c}}$  linear in die Koeffizienten des Polynoms  $\tilde{P}$  eingehen.  $\tilde{\mathbf{c}}^*$  sei ein fester Regler.

**Definition 1.1** Eine Frequenz  $z = z_S$  ist genau dann *singulär*, falls folgende zwei Bedingungen erfüllt sind:

$$\text{Rangbedingung: } \text{Rg}(\partial \tilde{P} / \partial \tilde{\mathbf{c}}) \Big|_{z_S} = 1 \quad (1.13)$$

$$\text{Wurzelbedingung: } P(z_S, \tilde{\mathbf{c}}^*) = 0. \quad (1.14)$$

Stellt  $z_S$  eine singuläre Frequenz dar, so ist ihre konjugiert komplexe Zahl ebenfalls eine singuläre Frequenz. Man beachte, dass eine singuläre Frequenz auf eine Hyperfläche in  $\tilde{\mathbf{c}}$  abgebildet wird.

## 1.4 Robustheit der Rangbedingung

Ein geschlossener Regelkreis besitze das nachstehende charakteristische Polynom,

$$\tilde{P}(z, \tilde{\mathbf{k}}, \mathbf{q}) = A(z, \mathbf{q}) \tilde{C}(z, \tilde{\mathbf{k}}) + B(z, \mathbf{q}). \quad (1.15)$$

Die Reglerstruktur  $\tilde{C}(z, \tilde{\mathbf{k}})$  sei ein Polynom in  $z$  beliebigen Grades mit den Reglerparametern  $\tilde{\mathbf{k}}$ , welche linear in die Koeffizienten des Polynoms  $\tilde{C}(z, \tilde{\mathbf{k}})$  eingehen.

**Theorem 1.1 (Bajcinca)** *Die Rangbedingung (1.13) für das charakteristische Polynom (1.15) bestimmt sich ausschließlich aus der Reglerstruktur  $\tilde{C}(z, \tilde{\mathbf{k}})$  und ist unabhängig von der Regelstrecke.*

*Beweis:* Die partielle Ableitung von (1.15) nach den Reglerparametern  $\tilde{\mathbf{k}}$  führt zu

$$\frac{\partial \tilde{P}}{\partial \tilde{\mathbf{k}}} = \mathcal{P} \mathbf{Z} \frac{\partial \tilde{\mathbf{c}}}{\partial \tilde{\mathbf{k}}}, \quad (1.16)$$

mit

$$\mathcal{P} = \begin{bmatrix} R_{\tilde{P}_k} & -I_{\tilde{P}_k} \\ I_{\tilde{P}_k} & R_{\tilde{P}_k} \end{bmatrix}, \quad \mathbf{Z} = \begin{bmatrix} 1 & \tau & \tau^2 - \eta^2 & \dots \\ 0 & \eta & 2\tau\eta & \dots \end{bmatrix}, \quad \frac{\partial \tilde{\mathbf{c}}}{\partial \tilde{\mathbf{k}}} = \left[ \frac{\partial \tilde{c}_1}{\partial \tilde{\mathbf{k}}}, \frac{\partial \tilde{c}_2}{\partial \tilde{\mathbf{k}}}, \dots \right]^T.$$

Dabei sind  $R_{\tilde{P}_k}$  und  $I_{\tilde{P}_k}$  Real- bzw. Imaginärteil von  $\tilde{P}(z, \tilde{\mathbf{k}}, \mathbf{q})$ ,  $\tau$  und  $\eta$  Real- bzw. Imaginärteil von  $z$  und  $\tilde{c}_i$  die Koeffizienten von  $\tilde{C}(z, \tilde{\mathbf{k}})$ .

Nach der Sylvesterschen Ungleichung [Gan65] gilt für den Rang der Matrizen

$$\begin{aligned} \text{Rg}(\mathcal{P}) + \text{Rg}\left(\mathbf{Z} \frac{\partial \tilde{\mathbf{c}}}{\partial \tilde{\mathbf{k}}}\right) - 2 &\leq \text{Rg}\left(\mathcal{P} \mathbf{Z} \frac{\partial \tilde{\mathbf{c}}}{\partial \tilde{\mathbf{k}}}\right) \\ &\leq \min\left(\text{Rg}(\mathcal{P}), \text{Rg}\left(\mathbf{Z} \frac{\partial \tilde{\mathbf{c}}}{\partial \tilde{\mathbf{k}}}\right)\right). \end{aligned}$$

Da  $\text{Rg}(\mathcal{P}) = 2$ , folgt direkt

$$\text{Rg}\left(\frac{\partial \tilde{P}}{\partial \tilde{\mathbf{k}}}\right) = \text{Rg}\left(\mathbf{Z} \frac{\partial \tilde{\mathbf{c}}}{\partial \tilde{\mathbf{k}}}\right). \quad (1.17)$$

□

Die Regelstrecke und die Unsicherheiten  $\mathbf{q}$  in (1.15) haben keinen Einfluss auf die Rangbedingung (1.13) und sind somit robust gegenüber einer Parametervariation von  $\mathbf{q}$ . Sie beeinflussen deswegen auch nicht das Aussehen von *singulären  $\Gamma$ -Gebieten*, welche im nächsten Abschnitt beschrieben werden. Die Lage von singulären Frequenzen  $z_S^i$  auf dem Rand eines singulären  $\Gamma$ -Gebietes ist jedoch sehr wohl abhängig von der Regelstrecke bzw. Parametervariation der Regelstrecke.

Die Methode der singulären Frequenzen ist also sehr nützlich für die Untersuchung einer endlichen Familie von Regelstrecken. Zudem gilt das Theorem 1.1 für Parametertransformationen der Reglerparameter  $\tilde{C}(z, \tilde{\mathbf{k}})$ , welche in den Abschnitten 1.2 und 1.5 Verwendung finden.

## 1.5 Singuläre $\Gamma$ -Gebiete

Für die Reglersynthese mit dem Parameterraumverfahren ist es üblich, die Lage der Pole innerhalb eines vorgegebenen  $\Gamma$ -Gebietes in der  $z$ -Ebene zu platzieren. Liegen alle Pole eines geschlossenen Regelkreises im vorgegebenen  $\Gamma$ -Gebiet, dann ist dieser Regelkreis  $\Gamma$ -stabil, andernfalls  $\Gamma$ -instabil.

**Definition 1.2** Ein  $\Gamma$ -Gebiet heißt *singulär*, falls die Rangbedingung für den gesamten Rand des  $\Gamma$ -Gebietes  $\partial\Gamma$  gilt, d.h.

$$Rg(\partial\tilde{P}/\partial\tilde{\mathbf{c}}) = 1 \quad \text{für alle } z \in \partial\Gamma. \quad (1.18)$$

Für jede Frequenz  $z \in \partial\Gamma$  stellt der Real- und der Imaginärteil von  $\tilde{P}$  jeweils eine Hyperfläche in  $\tilde{\mathbf{c}}$  dar. Ausschließlich bei singulären Frequenzen sind diese beiden Hyperflächen identisch und es existieren Lösungen für die charakteristische Gleichung (1.12). Aus diesem Grund können bei Parametervariation die Eigenwerte die Grenze  $\partial\Gamma$  nur bei singulären Frequenzen überschreiten.

Der Rest dieses Abschnittes zeigt, unter Verwendung der Methode der singulären Frequenzen, welche singulären  $\Gamma$ -Gebiete für Regleranteile der Form (1.1) existieren. Die theoretischen Herleitungen dafür folgen den Arbeiten [Baj01] und [AKB02].

Betrachte man das charakteristische Polynom des geschlossenen Regelkreises aus Abbildung 1.1

$$P(z, \mathbf{c}, \mathbf{q}) = A(z, \mathbf{q})(c_0 + c_1z + c_2z^2) + B(z, \mathbf{q}). \quad (1.19)$$

Es besitzt genau dann eine Wurzel bei  $z = \tau + j\eta$ , wenn gleichzeitig der Realteil  $R_{P_c}$  und der Imaginärteil  $I_{P_c}$  von  $P(z, \mathbf{c}, \mathbf{q})$  verschwinden. Diese Wurzelbedingung ergibt mit folgender Zerlegung in die Real- und Imaginärteile

$$\begin{aligned} A(\tau + j\eta) &= R_A + jI_A \\ B(\tau + j\eta) &= R_B + jI_B, \end{aligned} \quad (1.20)$$

und mit der Substitution  $z = \tau + j\eta$  das nachstehende Gleichungssystem,

$$\begin{bmatrix} R_{P_c} \\ I_{P_c} \end{bmatrix} = \begin{bmatrix} R_A & R_A\tau - I_A\eta & R_A(\tau^2 - \eta^2) - 2I_A\tau\eta \\ I_A & I_A\tau + R_A\eta & I_A(\tau^2 - \eta^2) + 2R_A\tau\eta \end{bmatrix} \times \\ \times \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix} + \begin{bmatrix} R_B \\ I_B \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \quad (1.21)$$

### 1.5.1 Hurwitz-Stabilität

Für Hurwitz-Stabilität repräsentiert die linke Halbebene in der  $z$ -Ebene das  $\Gamma$ -Gebiet. Der Rand des  $\Gamma$ -Gebietes  $\partial\Gamma$  ist die imaginäre Achse. Hurwitz-Stabilität folgt aus Gleichung (1.21) für  $\tau = 0$ . Für festes  $c_1$  gilt dann

$$\begin{bmatrix} R_A & -R_A\eta^2 \\ I_A & -I_A\eta^2 \end{bmatrix} \begin{bmatrix} c_0 \\ c_2 \end{bmatrix} + \begin{bmatrix} -c_1I_A\eta + R_B \\ c_1R_A\eta + I_B \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \quad (1.22)$$

Die Matrix links von  $[c_0, c_2]^T$  ist immer singulär. Damit ist auch die Rangbedingung (1.13) bezüglich den Parametern  $c_0$  und  $c_2$  erfüllt, und zwar für jedes  $z = j\eta$ , wobei  $\eta \in \mathbb{R}$ . Nach Definition 1.2 repräsentiert die linke Halbebene somit ein singuläres  $\Gamma$ -Gebiet für Regleranteile der polynomialen Form (1.1) mit festem  $c_1$ .

Gleichung (1.22) stellt zwei parallele Geraden in der  $(c_0, c_2)$ -Ebene dar. Es gibt genau dann Lösungen, wenn diese beiden Geraden identisch sind, also die folgende Gleichung gilt,

$$g_H(\eta) = \det \begin{bmatrix} R_A & -c_1I_A\eta + R_B \\ I_A & c_1R_A\eta + I_B \end{bmatrix} = 0. \quad (1.23)$$

Die Determinante  $g_H(\eta)$  ist ein Polynom in  $\eta$ . Ihre reellen Nullstellen ergeben nach Definition 1.1 die singulären Frequenzen  $z_s^i$ , wobei jede als Stabilitätsgrenze eine Gerade in der  $(c_0, c_2)$ -Ebene darstellt.

### 1.5.2 Allgemeiner Fall

Dieser Unterabschnitt untersucht für den allgemeinen Fall, also für variables  $c_1$ , welche singulären  $\Gamma$ -Gebiete existieren.

Die Parameter  $\mathbf{c}$  werden mit

$$\mathbf{c} = \mathbf{T}\mathbf{r} = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix} \begin{bmatrix} r_0 \\ r_1 \\ r_2 \end{bmatrix}, \quad \det \mathbf{T} \neq 0 \quad (1.24)$$

linear transformiert zu  $\mathbf{r} = [r_0, r_1, r_2]^T$ . Für festes  $r_1$  wird Gleichung (1.21) zu

$$\begin{aligned}
\begin{bmatrix} R_{P_r} \\ I_{P_r} \end{bmatrix} &= \begin{bmatrix} R_A & R_A\tau - I_A\eta & R_A(\tau^2 - \eta^2) - 2I_A\tau\eta \\ I_A & I_A\tau + R_A\eta & I_A(\tau^2 - \eta^2) + 2R_A\tau\eta \end{bmatrix} \begin{bmatrix} t_{11} & t_{13} \\ t_{21} & t_{23} \\ t_{31} & t_{33} \end{bmatrix} \begin{bmatrix} r_0 \\ r_2 \end{bmatrix} + \\
&+ \begin{bmatrix} t_{12}R_A + t_{22}(R_A\tau - I_A\eta) + t_{32}(R_A(\tau^2 - \eta^2) - 2I_A\tau\eta) \\ t_{12}I_A + t_{22}(I_A\tau + R_A\eta) + t_{32}(I_A(\tau^2 - \eta^2) + 2R_A\tau\eta) \end{bmatrix} r_1 + \\
&+ \begin{bmatrix} R_B \\ I_B \end{bmatrix} \\
&= \begin{bmatrix} u_{10} & u_{12} \\ u_{20} & u_{22} \end{bmatrix} \begin{bmatrix} r_0 \\ r_2 \end{bmatrix} + \begin{bmatrix} u_{11} \\ u_{21} \end{bmatrix} r_1 + \begin{bmatrix} R_B \\ I_B \end{bmatrix} \\
&= \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{1.25}
\end{aligned}$$

wobei  $R_{P_r}$  und  $I_{P_r}$  der Real- und Imaginärteil von dem charakteristischen Polynom  $P(z, \mathbf{r}, \mathbf{q})$  sind, welches äquivalent mit  $P(z, \mathbf{c}, \mathbf{q})$  ist.

Analog zum vorangehenden Unterabschnitt wird auch hier die  $2 \times 2$  Matrix links von  $[r_0, r_2]^T$  auf Singularität hin untersucht. Ist sie singular, so ist die Rangbedingung (1.13) bezüglich den Parametern  $r_0$  und  $r_2$  erfüllt. Damit die Rangbedingung (1.13) auch bezüglich den Parametern  $\mathbf{c}$  gilt, darf die Transformationsmatrix  $\mathbf{T}$  nicht singular sein. Die Berechnung der Determinante  $J$  der  $2 \times 2$  Matrix links von  $[r_0, r_2]^T$  führt zu

$$J = \eta (R_A^2 + I_A^2) (a(\tau^2 + \eta^2) + 2b\tau + c), \tag{1.26}$$

mit

$$\begin{aligned}
a &= t_{21}t_{33} - t_{23}t_{31} \\
b &= t_{11}t_{33} - t_{13}t_{31} \\
c &= t_{11}t_{23} - t_{21}t_{13}.
\end{aligned} \tag{1.27}$$

Für beliebiges  $A$  verschwindet  $J$ , wenn

$$\eta = 0 \tag{1.28}$$

oder

$$a(\tau^2 + \eta^2) + 2b\tau + c = 0. \tag{1.29}$$

Für die letzte Gleichung können zwei Fälle unterschieden werden.

Für  $a = 0$  resultiert die zur imaginären Achse parallele Gerade,

$$\tau = \tau_{\max} := -c/2b. \tag{1.30}$$

$c = 0$  entspricht dem Sonderfall der Hurwitz-Stabilität aus Unterabschnitt 1.5.1. Man beachte, dass die beiden Gleichungen

$$\begin{aligned}
a &= t_{21}t_{33} - t_{23}t_{31} = 0 \\
\tau_{\max} &= -c/2b = -(t_{11}t_{23} - t_{21}t_{13})/2(t_{11}t_{33} - t_{13}t_{31})
\end{aligned} \tag{1.31}$$

eine unendliche Anzahl an Lösungen für die Transformationsmatrix  $\mathbf{T}_{a=0}$  zulassen. Eine allgemeine Form für  $\mathbf{T}_{a=0}$  lässt sich aus (1.31) bestimmen zu

$$\mathbf{T}_{a=0} = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ -2\tau_{\max}t_{31} & t_{22} & -2\tau_{\max}t_{33} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}, \quad \det(\mathbf{T}_{a=0}) \neq 0. \quad (1.32)$$

Für  $a \neq 0$  beschreibt (1.29) einen Kreis in der komplexen  $z$ -Ebene mit Mittelpunkt  $m$  und Radius  $r$ ,

$$(\tau - m)^2 + \eta^2 = r^2, \quad (1.33)$$

wobei  $m$  und  $r$  den Gleichungen

$$\begin{aligned} m &= -b/a \\ r &= \sqrt{(b/a)^2 - c/a} \end{aligned} \quad (1.34)$$

genügen. Auch in diesem Fall gibt es eine unendliche Anzahl an Lösungen für die Transformationsmatrix  $\mathbf{T}_{a \neq 0}$ . Eine allgemeine Form für  $\mathbf{T}_{a \neq 0}$  folgt aus (1.33) und (1.34),

$$\mathbf{T}_{a \neq 0} = \begin{bmatrix} -t_{21}m + t_{31}(r^2 - m^2) & t_{12} & -t_{23}m + t_{33}(r^2 - m^2) \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}, \quad \det(\mathbf{T}_{a \neq 0}) \neq 0. \quad (1.35)$$

Die Matrix aus Gleichung (1.25) links von  $[c_0, c_2]^T$  ist für  $\mathbf{T}_{a=0}$  und für  $\mathbf{T}_{a \neq 0}$  immer singulär. Zusammengefasst existieren für Regleranteile der Form (1.1) ausschließlich singuläre  $\Gamma$ -Gebiete deren Rand  $\partial\Gamma$  entweder einen Kreis mit Mittelpunkt auf der reellen Achse beschreibt, oder identisch einer Parallelen zur imaginären Achse ist.

Daraus ergeben sich zwei sinnvolle Möglichkeiten für die Wahl von singulären  $\Gamma$ -Gebieten, zum einen die linke Halbebene, begrenzt durch eine Parallele zur imaginären Achse bei  $\tau = \tau_{\max}$  ( $\sigma$ -Stabilität), zum anderen eine Kreisscheibe mit Mittelpunkt  $m$  auf der reellen Achse und Radius  $r$  (Kreisstabilität).

Analog zu 1.5.1 besitzen die beiden Zeilen in Gleichung (1.25) als Lösung jeweils eine Gerade in der  $(r_0, r_2)$ -Ebene. Lösungen für das gesamte Gleichungssystem (1.25) treten genau dann auf, wenn diese beiden Geraden aufeinander fallen, also für die singulären Frequenzen  $z_S^i$ . Dazu muss gelten

$$g(\tau, \eta) = u_{10}(u_{21}r_1 + I_B) - u_{20}(u_{11}r_1 + R_B) = 0. \quad (1.36)$$

Als Ergebnis von (1.36) entsteht genau eine Gerade  $\lambda(z_S^i)$  für jede singuläre Frequenz  $z_S^i$ . Aus diesem Grund werden diese Geraden nach [Baj01] *singuläre Geraden* genannt.

**Bemerkung 1.1** Die Funktion  $g(\tau, \eta)$  ergibt sich aus der Determinante aus einer Matrix, welche gebildet wird aus der ersten Spalte der  $2 \times 2$  Matrix links von  $[r_0, r_2]^T$  aus Gleichung (1.25) und den beiden Summanden aus Gleichung (1.25) rechts von  $[r_0, r_2]^T$ . Eine äquivalente Matrix resultiert, wenn anstelle der ersten Spalte die zweite Spalte der  $2 \times 2$  Matrix links von  $[r_0, r_2]^T$  aus Gleichung (1.25) verwendet wird.

**Bemerkung 1.2** Die Arbeiten [Hoh02] und [ABB<sup>+</sup>02] geben den singulären Geraden verschiedene Bezeichnungen in Abhängigkeit von der Lage der zugehörigen singulären Frequenz. So heißen die Geraden für reelle singuläre Frequenzen reelle Wurzelgrenzen, für unendliche singuläre Frequenzen infinite Wurzelgrenzen und für die restlichen singuläre Frequenzen komplexe Wurzelgrenzen.

## 1.6 Berechnung der singulären Frequenzen

Der vorangegangene Abschnitt zeigt, welche singulären  $\Gamma$ -Gebiete für polynomiale Regleranteile zweiten Grades mit der Struktur aus Gleichung (1.1) existieren. Die singulären Frequenzen sind im Gegensatz zu den singulären  $\Gamma$ -Gebieten abhängig von der Regelstrecke. Wie sie berechnet werden erörtert dieser Abschnitt.

### 1.6.1 Parametrisierung

Die singulären Frequenzen  $z_S^i$  liegen bei  $\Gamma$ -Stabilität für singuläre  $\Gamma$ -Gebiete immer auf dem Rand dieses  $\Gamma$ -Gebietes  $\partial\Gamma$ . Es ist also ausreichend für  $z \in \partial\Gamma$  nach singulären Frequenzen zu suchen. Somit kann durch Parametrisierung von  $\partial\Gamma$  die Parameterabhängigkeit der Gleichung (1.36) von den beiden Parametern  $\tau$  und  $\eta$  auf eins reduziert werden.

Da die Wurzeln des geschlossenen Regelkreises konjugiert komplex und die singulären  $\Gamma$ -Gebiete symmetrisch bezüglich der  $\tau$ -Achse sind, können die Untersuchungen auf die obere Halbebene der  $z$ -Ebene weiter eingeschränkt werden. Eine einfache Parametrisierung für  $\sigma$ -Stabilität lautet dann

$$z_\sigma(\alpha) = \tau_{\max} + j\alpha, \quad \alpha \in [0; \infty). \quad (1.37)$$

Mit  $\tau_{\max} = 0$  entspricht sie der Parametrisierung für Hurwitz-Stabilität. Für Kreisstabilität kann  $\partial\Gamma$  parametrisiert werden durch

$$z_c(\alpha) = m + r \cdot e^{j\alpha}, \quad \alpha \in [0; \pi]. \quad (1.38)$$

Mit diesen beiden Formeln lässt sich Gleichung (1.36) umformen zu  $g(\alpha) = 0$ . Aufgelöst nach  $r_1$  folgt daraus die Funktion

$$r_1(\alpha) = \frac{u_{20}R_B - u_{10}I_B}{u_{10}u_{21} - u_{20}u_{11}}. \quad (1.39)$$

Die Suche nach den singulären Frequenzen  $z_S^i$  hat sich durch die Parametrisierung auf die Suche nach den zugehörigen  $\alpha_S^i$  vereinfacht. Die verschiedenen  $\alpha_S^i$  ergeben sich für ein festes  $r_1^*$  durch Schnitt der Kurve von  $r_1(\alpha)$  mit der Geraden  $r_1 = r_1^*$ . Die zugehörigen singulären Frequenzen  $z_S^i = z(\alpha_S^i)$  werden mit (1.37) bzw. (1.38) berechnet. Zu beachten ist, dass nach (1.28) die Schnitte des singulären  $\Gamma$ -Gebietes mit der reellen Achse unabhängig von der Regelstrecke immer singuläre Frequenzen ergeben.

**Bemerkung 1.3** Die hier eingeführte Funktion  $r_1(\alpha)$  entspricht im Parameterraumverfahren mit PID-Reglern für Hurwitz- und  $\sigma$ -Stabilität der Funktion  $k_P(\eta)$ , bzw. für zeitkontinuierliche Systeme  $k_P(\omega)$ , mit  $\omega = \Im(s)$  (vgl. [ABB<sup>+</sup>02] und [Hoh02]).

### 1.6.2 Algorithmus zur Berechnung singulärer Frequenzen

Die explizite Berechnung der singulären Frequenzen  $z_S^i$  bzw.  $\alpha_S^i$  ist für Systeme höherer Ordnung nichttrivial. Aus diesem Grund stellt dieser Unterabschnitt einen Algorithmus zur Berechnung der singulären Frequenzen vor, der das Problem auf eine Nullstellensuche überträgt.

Die gegenüber  $r_1(\alpha)$  vertikal verschobene Funktion  $\hat{r}_1(\alpha) = r_1(\alpha) - r_1^*$  ist stetig und monoton steigend oder monoton fallend im offenen Intervall  $I^x = (\alpha_E^x; \alpha_E^{x+1})$ , welches begrenzt wird von zwei benachbarten, lokalen Extrema von  $\hat{r}_1(\alpha)$ ,  $\alpha_E^x$  und  $\alpha_E^{x+1}$ . Die Nullstellen von  $\hat{r}_1(\alpha)$  sind die singulären Frequenzen  $\alpha_S^i$ .

Eine Möglichkeit, die singulären Frequenzen  $\alpha_S^i$  zu berechnen, ist also die Suche nach den Nullstellen von  $\hat{r}_1(\alpha)$ . Dazu teile man die  $\alpha$ -Achse in kleine Intervalle  $\Delta I^l$  der Breite  $\Delta\alpha$  auf und untersuche jedes dieser Intervalle  $\Delta I^l$  nach jeweils einer Nullstelle (siehe Abbildung 1.2, oben).

Selbstverständlich können nicht unendlich viele Intervalle  $\Delta I^l$  untersucht werden, so dass eine Einschränkung auf ein Intervall  $I_{\text{ges}}$  erforderlich ist, in dem die Nullstellensuche stattfindet. Wie Abschnitt 1.10 zeigt, gehen durch eine geeignete Begrenzung keine entscheidenden Informationen verloren.

Die Intervallbreite  $\Delta\alpha$  muss klein genug sein, damit der Algorithmus auch zwei benachbarte singuläre Frequenzen unterscheiden kann. Allerdings ist die Intervallbreite  $\Delta\alpha$  indirekt proportional zur Anzahl an Intervallen  $\Delta I^l$ , so dass für schmale Intervalle  $\Delta I^l$  die Rechenzeit stark zunimmt. Ein Kompromiss ist erforderlich.

Für die Bestimmung von singulären Frequenzen für verschiedene Werte von  $r_1^*$  existiert ein schnellerer Algorithmus, welcher die Lage der *kritischen Frequenzen*  $\alpha_K^x$  von  $r_1(\alpha)$  verwendet. Kritische Frequenzen  $\alpha_K^x$  bezeichnen in *RobSin* die Frequenzen, an denen die Ableitung oder der Nenner von  $r_1(\alpha)$  verschwindet, und die Frequenzen an den Intervallgrenzen von  $I_{\text{ges}}$ . Ihre Bedeutung klären die folgenden Überlegungen.

Die lokalen Extrema  $\alpha_E^x$  von  $\hat{r}_1(\alpha)$  sind unabhängig von  $r_1^*$  und gleich den lokalen Extrema von  $r_1(\alpha)$ . Ihre zeitaufwendige Bestimmung muss daher ausschließlich ein einziges Mal zu Beginn des Algorithmus durchgeführt werden. Sie geschieht analog zu der oben beschriebenen Bestimmung der Nullstellen von  $\hat{r}_1(\alpha)$ , nur dass hier an Stelle von  $\hat{r}_1(\alpha)$  die Nullstellensuche auf die Ableitung und den Nenner von  $r_1(\alpha)$  angewandt wird (siehe Abbildung 1.2, Mitte und unten).

In dem Intervall  $I^x$ , begrenzt durch zwei benachbarte, lokale Extrema  $\alpha_E^x$  und  $\alpha_E^{x+1}$ , existiert wegen dem Monotonieverhalten genau dann eine singuläre Frequenz  $\alpha_S^i$ , wenn die beiden Werte von  $\hat{r}_1(\alpha)$  an den Intervallgrenzen von  $I^x$  unterschiedliche Vorzeichen haben,

$$\text{sign}(\hat{r}_1(\alpha_E^x)) \neq \text{sign}(\hat{r}_1(\alpha_E^{x+1})). \quad (1.40)$$

Ist eine Intervallgrenze unendlich, so bestimmt sich das Vorzeichen durch Bildung des Grenzwertes an dieser Stelle.

Mit diesem Wissen ist klar, dass auch maximal eine singuläre Frequenz  $\alpha_S^i$  zwischen zwei benachbarten kritischen Frequenzen  $\alpha_K^x$  und  $\alpha_K^{x+1}$  existieren kann, und zwar genau dann, wenn die Vorzeichenbedingung (1.40) für diese beiden Frequenzen erfüllt ist. Dadurch muss für jedes  $r_1^*$  nur noch für die kleine Zahl an Intervallen  $I^x$ , die sich bestimmen durch die beiden benachbarten kritischen Frequenzen  $\alpha_K^x$  und  $\alpha_K^{x+1}$ , die Nullstellensuche durchgeführt werden.

Der Algorithmus zur Berechnung der singulären Frequenzen lautet damit:

1. Berechne die kritischen Frequenzen  $\alpha_K^x$  im Intervall  $I_{\text{ges}}$  unter Verwendung der Breite  $\Delta\alpha$  der Teilintervalle  $\Delta I^l$ .
2. Berechne für jedes  $r_1^*$  die singulären Frequenzen  $\alpha_S^i$  unter Zuhilfenahme der kritischen Frequenzen  $\alpha_K^x$ .

Eine wichtige Bedeutung der lokalen Extrema  $\alpha_E^x$  von  $r_1(\alpha)$  wurde bisher noch nicht erwähnt. Bewegt sich die horizontale Gerade  $r_1 = r_1^*$  in Richtung der  $r_1$ -Achse über ein lokales Extremum  $r_1(\alpha_E^x)$ , so entstehen oder verschwinden zwei singuläre Frequenzen  $\alpha_S^i$  bzw.  $z_S^i$  und mit ihnen zwei singuläre Geraden  $\lambda(z_S^i)$  in der  $(r_0, r_2)$ -Ebene. Die daraus resultierenden Auswirkungen auf das stabile, dreidimensionale Gebiet erörtert Abschnitt 1.12.

## 1.7 Berechnung der singulären Geraden

Mit den singulären Frequenzen  $z_S^i$  können die singulären Geraden berechnet werden. Ihre Berechnung leitet dieser Abschnitt her.

Wie in Abschnitt 1.5.2 beschrieben, entsprechen in gegenüber  $\mathbf{c}$  linear transformierten Koordinaten  $\mathbf{r}$  singuläre Frequenzen  $z_S^i$  jeweils einer singulären Geraden  $\lambda(z_S^i)$  in der  $(r_0, r_2)$ -Ebene. Ihre Geradengleichung resultiert durch Auflösen

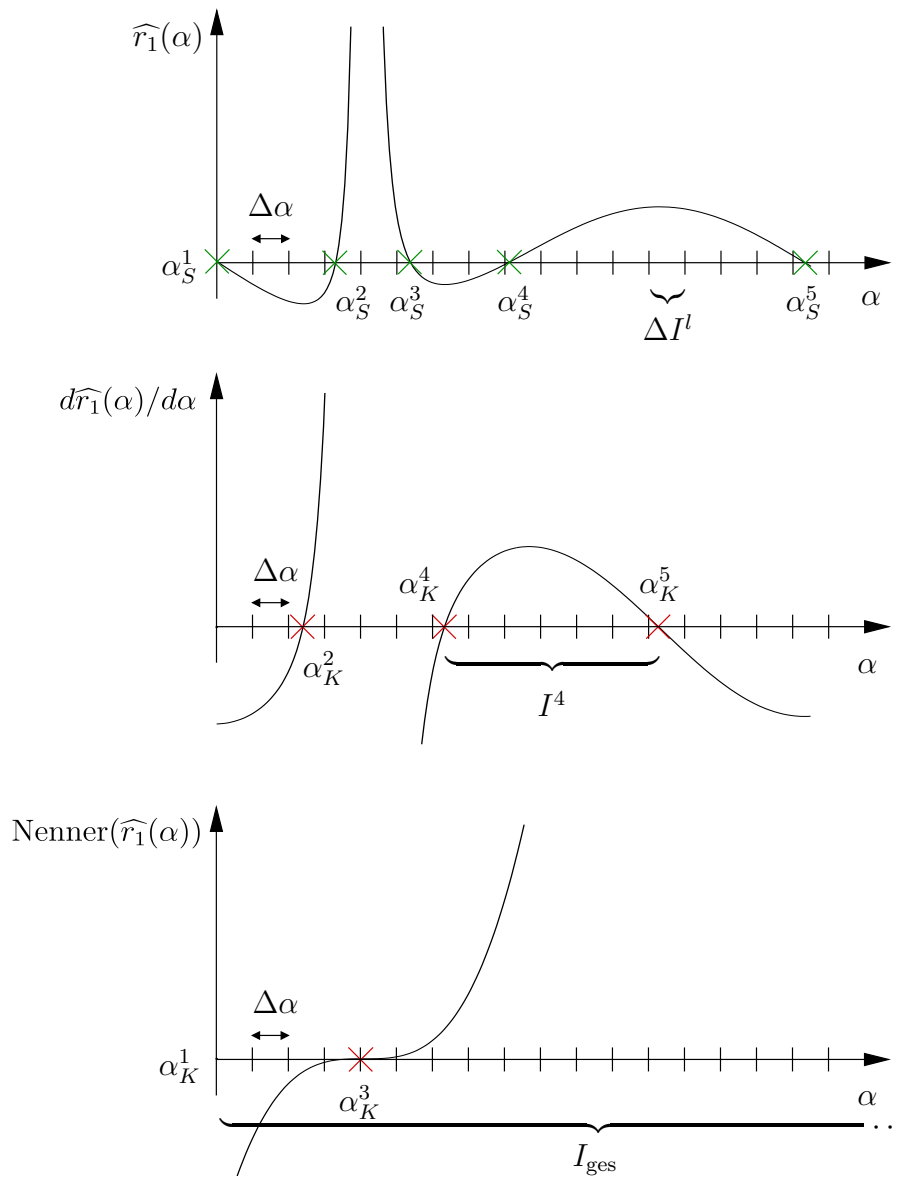


Abbildung 1.2: Schematisch dargestellte Funktion  $\widehat{r}_1(\alpha)$ , ihre Ableitung und ihr Nenner. Die Kreuze markieren die Nullstellen der jeweiligen Funktionen.

von Gleichung (1.25) nach  $r_2$ . Die erste Zeile ergibt

$$r_2 = -\frac{u_{10}}{u_{12}}r_0 - \frac{u_{11}r_1 + R_B}{u_{12}}, \quad (1.41)$$

die zweite

$$r_2 = -\frac{u_{20}}{u_{22}}r_0 - \frac{u_{21}r_1 + I_B}{u_{22}}. \quad (1.42)$$

Für  $u_{12} \neq 0$  und  $u_{22} \neq 0$  sind diese beiden Gleichungen identisch. Für  $u_{12} = u_{22} = 0$  besitzt die singuläre Gerade eine unendliche Steigung. Um die Gerade dennoch zu bestimmen, muss Gleichung (1.25) nach  $r_0$  aufgelöst werden. Die beiden Zeilen ergeben in diesem Fall die identischen Geradengleichungen

$$r_0 = -\frac{u_{11}r_1 + R_B}{u_{10}} \quad (1.43)$$

und

$$r_0 = -\frac{u_{21}r_1 + I_B}{u_{20}}. \quad (1.44)$$

Das Ergebnis dieses Verfahrens ist eine Menge an singulären Geraden  $\lambda(z_S^i)$  in der  $(r_0, r_2)$ -Ebene. Diese Geraden bilden konvexe Polygone. Da der untersuchte Regelkreis nur bei Überschreiten einer singulären Geraden die  $\Gamma$ -Stabilität ändern kann, sind alle Werte  $(r_0, r_2)$  innerhalb eines solchen Polygons bereits dann  $\Gamma$ -stabilisierend, wenn für einen einzigen Punkt innerhalb dieses Polygons das System  $\Gamma$ -stabil ist. Das Polygon heißt dann *stabiles Polygon*. Die Grenzen eines stabilen Polygons heißen *aktive Grenzen*.

Man beachte, dass alle singulären Geraden und somit auch alle Polygone für festes  $r_1 = r_1^*$  in derselben Ebene im  $\mathbf{r}$ -Raum liegen. In den gegenüber  $\mathbf{r}$  linear transformierten Koordinaten  $\mathbf{c}$  befinden sie sich ebenfalls alle auf einer einzigen gedrehten Ebene. Entsprechendes gilt für die Koordinaten  $\mathbf{k}$ . Für verschiedene Werte von  $r_1$  liegen die singulären Geraden und die konvexen Polygone auf jeweils zueinander parallelen Ebenen.

Die Ebenen sind wegen Theorem 1.1 unabhängig von der Regelstrecke. Dies bedeutet, dass für festes  $r_1 = r_1^*$  alle singulären Geraden, auch die von anderen Regelstrecken, jeweils in derselben Ebene liegen. Diese Tatsache ermöglicht die elegante Berücksichtigung von Parameterunsicherheiten, indem die stabilen Polygone der verschiedenen Regelstrecken, die sich aus den einzelnen Betriebspunkten ergeben, zum Schnitt gebracht werden. Falls die Schnittmenge nicht leer ist, so sind die darin enthaltenen Regler gleichzeitig für alle untersuchten Regelstrecken  $\Gamma$ -stabilisierend.

## 1.8 Innere Polygone

Eine verbreitete Methode nach aktiven Grenzen im Parameterraum zu suchen besteht darin, die zu der aktiven Grenze gehörenden Eigenwerte aus dem cha-

rakteristischen Polynom herauszunehmen, und den Rest an Eigenwerten nach  $\Gamma$ -Stabilität zu untersuchen.

Bei den in Abschnitt 1.10 totzeitbehafteten, zeitkontinuierlichen Systemen treten jedoch unendlich viele Eigenwerte auf, deren Lage nicht berechnet werden kann, so dass diese Methode nicht anwendbar ist. Aus diesem Grund stellt dieser Unterabschnitt die in [Baj01] eingeführte Methode der *inneren Polygone* vor. Sie ist auch anwendbar auf totzeitbehaftete Systemen.

**Definition 1.3** *Ein Polygon, das durch eine Menge an singulären Geraden  $\{\lambda(z_S^i)\}$ , mit den zugehörigen singulären Frequenzen  $\{z_S^i\}$ , definiert wird, heißt inneres Polygon, wenn bei der Überquerung  $[\delta r_0, \delta r_2]$  über jede singuläre Gerade  $\lambda(z_S^i)$  von außen nach innen bezüglich des Polygons ein Eigenwert das  $\Gamma$ -Gebiet bei  $z_S^i$  betritt.*

**Lemma 1.1** *Stabile Polygone sind innere Polygone.*

Dieses Lemma beschreibt eine notwendige, jedoch nicht hinreichende Bedingung für stabile Polygone. Eine Aussage über  $\Gamma$ -Stabilität lässt sich erst nach einer Stabilitätsüberprüfung treffen. Falls die inneren Polygone nicht  $\Gamma$ -stabil sind, so existiert kein stabilisierender Regler der verwendeten Struktur.

### 1.8.1 Überquerung von singulären Geraden

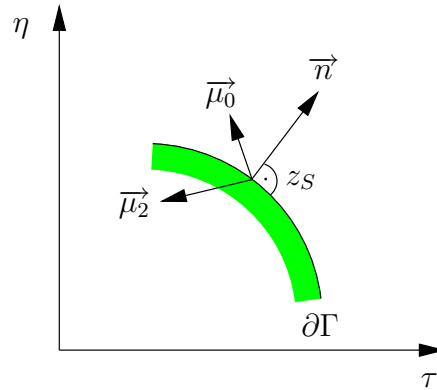
Um innere Polygone zu finden, ist es erforderlich zu wissen, welche Seite einer singulären Gerade  $\lambda(z_S)$  stabiler ist, also auf welcher Seite mehr Eigenwerte im  $\Gamma$ -Gebiet liegen. Dazu wird jeder singulären Geraden eine Vorzeichenfunktion  $e$  zugeordnet, welche abhängt von einer Bewegung  $[\delta r_0, \delta r_2]$  über die singuläre Gerade  $\lambda(z_S)$ . Es gelte die folgende Festlegung:  $e > 0$ , falls mindestens ein Eigenwert das  $\Gamma$ -Gebiet verlässt, und  $e < 0$ , falls mindestens ein Eigenwert es betritt.

Betrachte man zunächst Abbildung 1.3, die schematisch einen Teil vom Rand eines singulären  $\Gamma$ -Gebietes  $\partial\Gamma$  darstellt.  $\vec{n}$  sei ein Vektor senkrecht zu  $\partial\Gamma$ , bei der singulären Frequenz  $z_S$ , mit Richtung auf die instabile Seite des  $\Gamma$ -Gebietes. Die Schattierung zeigt die  $\Gamma$ -stabile Seite an. Die Bedeutung von  $\vec{\mu}_0$  und  $\vec{\mu}_2$  klärt sich im Laufe dieses Abschnittes.  $\partial\Gamma$  lässt sich als implizite Funktion  $F(\tau, \eta) = 0$  angeben, so dass sich  $\vec{n}$  als Gradient von  $F(\tau, \eta)$  berechnet zu

$$\vec{n} := \text{grad}(F)|_{z_S} = \begin{bmatrix} \partial F / \partial \tau \\ \partial F / \partial \eta \end{bmatrix}_{z_S}. \quad (1.45)$$

Für  $\sigma$ -Stabilität lässt sich eine einfache implizite Funktion  $F_\sigma(\tau, \eta)$  und ihr Gradient  $\vec{n}_\sigma$  angeben durch

$$\partial\Gamma : \quad F_\sigma(\tau, \eta) = \tau - \tau_{\max} = 0 \quad (1.46)$$

Abbildung 1.3: Ausschnitt aus  $\partial\Gamma$  mit einer singulären Frequenz  $z_S$ .

und

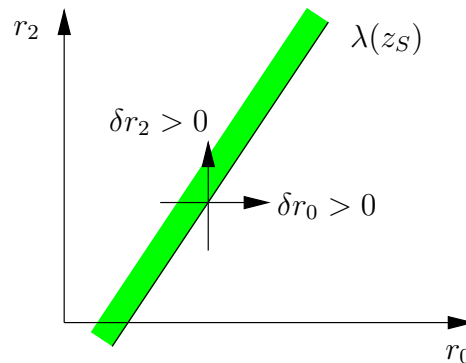
$$\vec{n}_\sigma := \text{grad}(F_\sigma)|_{z_S} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}_{z_S}. \quad (1.47)$$

Für Kreisstabilität repräsentieren die folgenden beiden Gleichungen eine implizite Funktion  $F_c(\tau, \eta)$  und ihren Gradienten  $\vec{n}_c$ ,

$$\partial\Gamma : F_c(\tau, \eta) = \sqrt{(\tau - m)^2 + \eta^2} - r = 0 \quad (1.48)$$

$$\vec{n}_c := \text{grad}(F_c)|_{z_S} = \begin{bmatrix} \frac{\tau - m}{\sqrt{(\tau - m)^2 + \eta^2}} \\ \frac{\eta}{\sqrt{(\tau - m)^2 + \eta^2}} \end{bmatrix}_{z_S}. \quad (1.49)$$

Abbildung 1.4 zeigt in der  $(r_0, r_2)$ -Ebene die singuläre Gerade  $\lambda(z_S)$ , welche der singulären Frequenz  $z_S$  zugeordnet ist.

Abbildung 1.4: Achsenparallele Bewegungen über die singuläre Gerade  $\lambda(z_S)$ .

Die achsparallelen Bewegungen  $\delta r_0 > 0$  und  $\delta r_2 > 0$  über die singuläre Gerade  $\lambda(z_S)$  seien infinitesimal klein. Sie bewirken, dass der zugehörige Eigenwert und der dazu konjugiert komplexe Eigenwert das  $\Gamma$ -Gebiet bei  $z_S$  bzw.  $z_S^*$  betreten oder verlassen. Bei reellen singulären Frequenzen  $z_S = \tau_S$  überquert nur ein Eigenwert die Berandung des  $\Gamma$ -Gebietes  $\partial\Gamma$  bei  $\tau_S$ . Die Richtungen dieser Bewegungen in der  $z$ -Ebene berechnen sich zu

$$\vec{\mu}_0 = \begin{bmatrix} \partial\tau/\partial r_0 \\ \partial\eta/\partial r_0 \end{bmatrix}_{z_S} \quad \text{und} \quad \vec{\mu}_2 = \begin{bmatrix} \partial\tau/\partial r_2 \\ \partial\eta/\partial r_2 \end{bmatrix}_{z_S}, \quad (1.50)$$

wobei  $\vec{\mu}_0$  der Transition  $\delta r_0$  und  $\vec{\mu}_2$  der Transition  $\delta r_2$  entspricht (vgl. Abbildung 1.3). Die Vorzeichenfunktion  $e$  lässt sich dann mit dem Skalarprodukt von  $\vec{n}$  und  $\vec{\mu}$  bestimmen zu

$$e_{0/2} = \text{sign}(\vec{n}^T \cdot \vec{\mu}_{0/2}). \quad (1.51)$$

Sie benötigt die Ausdrücke für die Vektoren  $\vec{\mu}_0$  und  $\vec{\mu}_2$ , die in den folgenden Berechnungen abgeleitet werden. Ohne Beschränkung der Allgemeinheit wird zunächst die Überquerung  $\delta r_0$  untersucht. Die Ableitung des Gleichungssystems (1.25) nach  $r_0$  an der Stelle  $z_S$  ergibt

$$\begin{bmatrix} dR_{P_r}/dr_0 \\ dI_{P_r}/dr_0 \end{bmatrix}_{z_S} = \begin{bmatrix} u_{10} \\ u_{20} \end{bmatrix}_{z_S} + \underbrace{\begin{bmatrix} \partial R_{P_r}/\partial\tau & \partial R_{P_r}/\partial\eta \\ \partial I_{P_r}/\partial\tau & \partial I_{P_r}/\partial\eta \end{bmatrix}_{z_S}}_{\left[ \frac{\partial(R_{P_r}, I_{P_r})}{\partial(\tau, \eta)} \right]_{z_S}} \cdot \begin{bmatrix} \partial\tau/\partial r_0 \\ \partial\eta/\partial r_0 \end{bmatrix}_{z_S} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad (1.52)$$

wobei der zweite Summand durch Abhängigkeit der komplexen Variablen  $z$  von den Parametern  $\mathbf{r}$  entsteht. Dieses Gleichungssystem kann nun nach den Elementen von  $\vec{\mu}_0$  aufgelöst werden zu

$$\begin{aligned} \vec{\mu}_0 = \begin{bmatrix} \partial\tau/\partial r_0 \\ \partial\eta/\partial r_0 \end{bmatrix}_{z_S} &= - \left[ \frac{\partial(R_{P_r}, I_{P_r})}{\partial(\tau, \eta)} \right]_{z_S}^{-1} \cdot \begin{bmatrix} u_{10} \\ u_{20} \end{bmatrix}_{z_S} \\ &= \frac{1}{J_{z_S}} \cdot \begin{bmatrix} -u_{10} \frac{\partial I_{P_r}}{\partial\eta} + u_{20} \frac{\partial R_{P_r}}{\partial\eta} \\ u_{10} \frac{\partial I_{P_r}}{\partial\tau} - u_{20} \frac{\partial R_{P_r}}{\partial\tau} \end{bmatrix}_{z_S} \\ &= \frac{1}{J_{z_S}} \cdot \left[ \det \left[ \frac{\partial(R_{P_r}, I_{P_r})}{\partial(\eta, r_0)} \right] \quad \det \left[ \frac{\partial(R_{P_r}, I_{P_r})}{\partial(r_0, \tau)} \right] \right]_{z_S}^T, \end{aligned} \quad (1.53)$$

mit

$$J_{z_S} = \det \left[ \frac{\partial(R_{P_r}, I_{P_r})}{\partial(\tau, \eta)} \right]_{z_S}. \quad (1.54)$$

Die Cauchy-Riemann Bedingungen (siehe [Wei99])

$$\begin{aligned} \partial R_{P_r}/\partial\tau &= \partial I_{P_r}/\partial\eta \\ \partial R_{P_r}/\partial\eta &= -\partial I_{P_r}/\partial\tau \end{aligned} \quad (1.55)$$

garantieren, dass für einfache singuläre Frequenzen  $z_S$  (mit der Multiplizität eins) die Determinante  $J_{z_S}$  größer als null ist. Der Faktor  $1/J_{z_S}$  in (1.53) beeinflusst also nicht die Richtung und das Vorzeichen von  $\vec{\mu}_0$ . Ihm muss keine weitere Aufmerksamkeit geschenkt werden. Entsprechende Überlegungen, wie sie für  $\delta r_0$  angestellt wurden, gelten auch für die Transition  $\delta r_2$ . Damit lässt sich folgendes Theorem formulieren.

**Theorem 1.2 (Bajcinca)** *Sei  $z_S$  eine singuläre Frequenz. Die Überquerung über die singuläre Gerade  $\lambda(z_S)$  lässt einen Eigenwert das  $\Gamma$ -Gebiet bei  $z_S$  betreten oder verlassen, abhängig von dem Vorzeichen des Ausdrucks*

$$e_{0/2} := \text{sign} \left( \frac{\partial F}{\partial \tau} \cdot \det \left[ \frac{\partial(R_{P_r}, I_{P_r})}{\partial(\eta, r_{0/2})} \right] + \frac{\partial F}{\partial \eta} \cdot \det \left[ \frac{\partial(R_{P_r}, I_{P_r})}{\partial(r_{0/2}, \tau)} \right] \right)_{z_S}. \quad (1.56)$$

*Falls der Ausdruck positiv ist, verlässt mindestens ein Eigenwert das  $\Gamma$ -Gebiet, falls er negativ ist, betritt mindestens ein Eigenwert das  $\Gamma$ -Gebiet.*

Der Ausdruck  $e_0$  bzw.  $e_2$  verschwindet, wenn die singuläre Gerade parallel zur  $r_0$ - bzw.  $r_2$ -Achse verläuft. Es können offensichtlich nicht beide Ausdrücke  $e_0$  und  $e_2$  gleichzeitig zu null werden. Verschwindet einer von ihnen, so muss der andere für die Untersuchung zur Überquerung von singulären Geraden herangezogen werden.

Da  $\Gamma$ -Gebiete symmetrisch zur  $\tau$ -Achse sind, verschwindet für reelle singuläre Frequenzen  $z_S = \tau_S$  die partielle Ableitung  $(\partial F / \partial \eta) \big|_{\tau_S} = 0$ . Damit vereinfacht sich (1.56) zu

$$e_{0/2} := \text{sign} \left( \frac{\partial F}{\partial \tau} \cdot \det \left[ \frac{\partial(R_{P_r}, I_{P_r})}{\partial(\eta, r_{0/2})} \right] \right)_{\tau_S}. \quad (1.57)$$

Das Theorem 1.2 ist sehr allgemein. Es gilt auch für nichtlineare Parameterabhängigkeiten. Es bleibt jedoch noch die Frage offen, wie der Ausdruck (1.56) von den Koordinaten  $[r_0, r_2] \in \lambda(z_S)$  abhängt, bei welchen die singuläre Gerade  $\lambda(z_S)$  überquert wird.

Der Realteil  $R_{P_r}$  und der Imaginärteil  $I_{P_r}$  des charakteristischen Polynoms  $P(z, \mathbf{r}, \mathbf{q})$  ändern sich bei Bewegung entlang der singulären Geraden  $\lambda(z_S)$  nicht. Somit bleiben auch die partiellen Ableitungen von  $P(z, \mathbf{r}, \mathbf{q})$  bezüglich  $\tau$  und  $\eta$  unberührt von solchen Bewegungen. Da die Parameter  $\mathbf{r}$  ohnehin keinen Einfluss auf die Faktoren  $u_{10}$ ,  $u_{20}$ ,  $u_{12}$  und  $u_{22}$  haben und  $\vec{n}$  ausschließlich von der auf der singulären Geraden  $\lambda(z_S)$  konstanten, singulären Frequenz  $z_S$  abhängt, ist  $e_{0/2}$  konstant für alle  $[r_0, r_2] \in \lambda(z_S)$ . Es gilt also das folgende Theorem.

**Theorem 1.3 (Bajcinca)** *Die Vorzeichenfunktion  $e_{0/2}$  ist unabhängig von der Stelle, an der eine singuläre Gerade  $\lambda(z_S)$  überquert wird.*

Dies ist ein sehr wichtiges Ergebnis, da es erlaubt, die singuläre Gerade  $\lambda(z_S)$  an beliebiger Stelle zu kreuzen.

### 1.8.2 Algorithmus zur Bestimmung innerer Polygone

Auf der Grundlage der Überquerung von singulären Geraden lässt sich ein Algorithmus angeben, der die Bestimmung der inneren Polygone automatisiert. Dazu gelte die Annahme, dass ein Polygon  $\Pi$  durch  $N$  Kanten  $\delta(\lambda_i)$  im Uhrzeigersinn umschlossen ist,

$$\Pi = \{\delta(\lambda_1), \delta(\lambda_2), \dots, \delta(\lambda_N)\}, \quad (1.58)$$

so dass der Endpunkt der Kante  $\delta(\lambda_x)$  mit dem Anfangspunkt der nächsten Kante  $\delta(\lambda_{x+1 \bmod N})$  zusammenfällt. Jede Kante  $\delta(\lambda_i)$  ist ein Teil einer singulären Geraden  $\lambda_i$  und ist durch den Anfangspunkt  $\mathbf{r}^A(\lambda_i)$  und den Endpunkt  $\mathbf{r}^E(\lambda_i)$  definiert durch

$$\delta(\lambda_i) = [\mathbf{r}^A(\lambda_i) \quad \mathbf{r}^E(\lambda_i)] = \begin{bmatrix} r_0^A(\lambda_i) & r_0^E(\lambda_i) \\ r_2^A(\lambda_i) & r_2^E(\lambda_i) \end{bmatrix}. \quad (1.59)$$

Ein Algorithmus, der entscheidet, ob ein Polygon  $\Pi$  ein inneres Polygon ist oder nicht, lautet wie folgt.

```

for i = 1 to N
  if abs( $r_0^A(\lambda_i) - r_0^E(\lambda_i)$ ) > abs( $r_2^A(\lambda_i) - r_2^E(\lambda_i)$ ) then
     $e_i = e_0$ 
     $\Delta_i = r_0^E(\lambda_i) - r_0^A(\lambda_i)$ 
  else
     $e_i = e_2$ 
     $\Delta_i = r_2^E(\lambda_i) - r_2^A(\lambda_i)$ 
  end
end
if any(sign( $e_i$ )  $\neq$  sign( $\Delta_i$ ))
   $\Pi$  ist kein inneres Polygon
else
   $\Pi$  ist ein inneres Polygon
end

```

Er muss auf alle Polygone angewendet werden. An dieser Stelle sei nochmals angemerkt, dass innere Polygone nicht zwingend stabile Polygone sein müssen. Eine Aussage über Stabilität eines Polygons kann erst nach einem Stabilitätstest erfolgen.

## 1.9 Klassen geeigneter Eigenwertspezifikationen

Abschnitt 1.5 zeigt, für polynomiale Regleranteile zweiten Grades (1.1) existieren ausschließlich singuläre  $\Gamma$ -Gebiete, die entweder durch eine Parallele zur imaginären Achse ( $\sigma$ -Stabilität), oder durch einen Kreis mit reellem Mittelpunkt

begrenzt werden (Kreisstabilität). Dadurch lassen sich einige Eigenwertspezifikationen von Regelkreisen erfüllen. Die Hurwitz-Stabilität eines Systems garantiert keine ausreichenden Systemeigenschaften. Oft muss ein System in einer vorgegebenen Zeit einschwingen oder eine Mindestdämpfung vorweisen.

Für zeitkontinuierliche Systeme mit der komplexen Variablen  $s = \sigma + j\omega$  kann die Einschwingzeit  $T_{\text{ein}}$  durch Verschieben von  $\sigma_{\text{max}}$ , dem Maximum der Realteile aller Pole, eingestellt werden. Nach [Sch00] berechnet sie sich zu

$$T_{\text{ein}} \leq \frac{3}{|\sigma_{\text{max}}|}. \quad (1.60)$$

Ein schnelleres Einschwingen bei zeitdiskreten Systemen mit der Abtastzeit  $T_S$  resultiert aus einer Beschränkung der Pole auf das Innere eines Kreises mit Mittelpunkt gleich null und Radius kleiner eins.

Die Parallele zur imaginären Achse  $s = \sigma_{\text{max}}$  eingesetzt in die Umrechnungsformel zwischen Laplace-Transformation und  $z$ -Transformation ([LW02])

$$z = e^{s \cdot T_S} \quad (1.61)$$

ergibt den Radius  $r$  des Kreises in der  $z$ -Ebene (siehe Abbildung 1.5),

$$r = e^{\sigma_{\text{max}} \cdot T_S}. \quad (1.62)$$

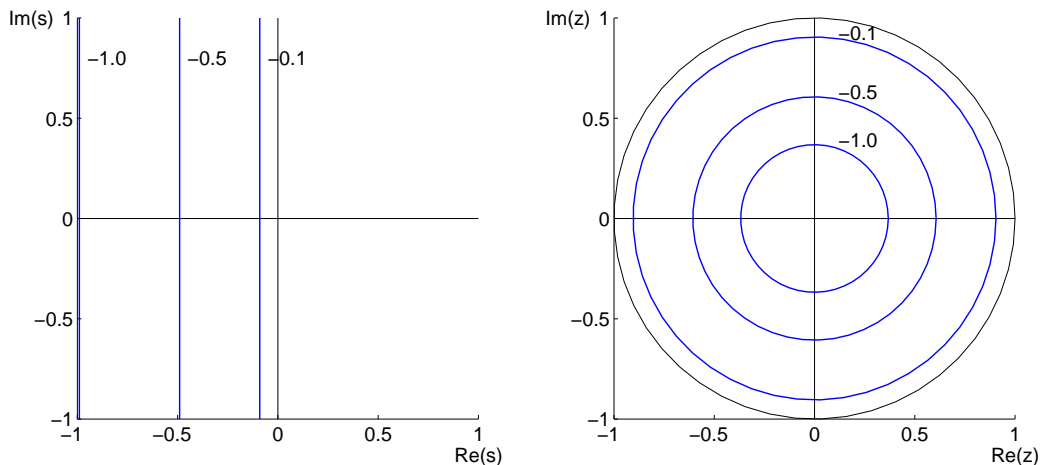


Abbildung 1.5: Links: Einschwingzeit in der  $s$ -Ebene; die Zahlen entsprechen  $\sigma_{\text{max}}$ . Rechts: Einschwingzeit in der  $z$ -Ebene; die Zahlen entsprechen  $\sigma_{\text{max}} \cdot T_S$ .

Da für diesen Fall nur Parallelen zur imaginären Achse und Kreise mit Mittelpunkt null als Rand des  $\Gamma$ -Gebietes  $\partial\Gamma$  entstehen, ist die Methode der singulären Frequenzen mit polynomialen Regleranteilen zweiter Ordnung für die Begrenzung

der Einschwingzeit  $T_{\text{ein}}$  sowohl für zeitkontinuierliche, als auch für zeitdiskrete Systeme einsetzbar.

Die Einstellung einer Mindestdämpfung  $D$  erfordert andere  $\Gamma$ -Gebiete. Im Laplace-Bereich geben zwei Halbgeraden das  $\Gamma$ -Gebiet vor, das alle Pole eines Systems mit garantierter Mindestdämpfung  $D$  enthält. Abbildung 1.6, links, zeigt exemplarisch für einige Werte von  $D$  diese Halbgeraden. Das zugehörige  $\Gamma$ -Gebiet liegt jeweils auf der linken Seite der beiden Halbgeraden. Ihre Parametrierung lautet

$$s_{\pm}(\alpha) = -\alpha \cdot \cos(\varphi) \pm j\alpha \cdot \sin(\varphi), \quad \alpha \in [0; \infty), \quad (1.63)$$

wobei  $\varphi = \arccos(D)$  der Winkel zwischen der Halbgeraden und der reellen Achse ist. Die Transformation (1.61) dieser Halbgeraden in die  $z$ -Ebene führt zu

$$\begin{aligned} z_{\pm}(\alpha) &= e^{T_S \cdot s_{\pm}(\alpha)} \\ &= e^{-\alpha T_S \cdot \cos(\varphi)} e^{\pm j\alpha T_S \cdot \sin(\varphi)}. \end{aligned} \quad (1.64)$$

Die Betrachtung dieses Ergebnisses in Polarkoordinaten zusammen mit der Substitution  $\hat{\alpha} = \alpha T_S$  ergibt für Betrag und Winkel von  $z_{\pm}(\hat{\alpha})$

$$\begin{aligned} |z_{\pm}(\hat{\alpha})| &= \hat{\alpha} \cdot \cos(\varphi) \\ \angle z_{\pm}(\hat{\alpha}) &= \pm j\hat{\alpha} \cdot \sin(\varphi) \end{aligned} \quad \hat{\alpha} \in [0; \infty), \quad (1.65)$$

welches unabhängig von der Abtastzeit  $T_S$  ist. Die exakten Grenzen von  $\Gamma$ -Gebieten für die Einstellung einer Mindestdämpfung beschreiben also zwei, zur reellen Achse symmetrische, logarithmische Spiralen (siehe Abbildung 1.6, rechts).

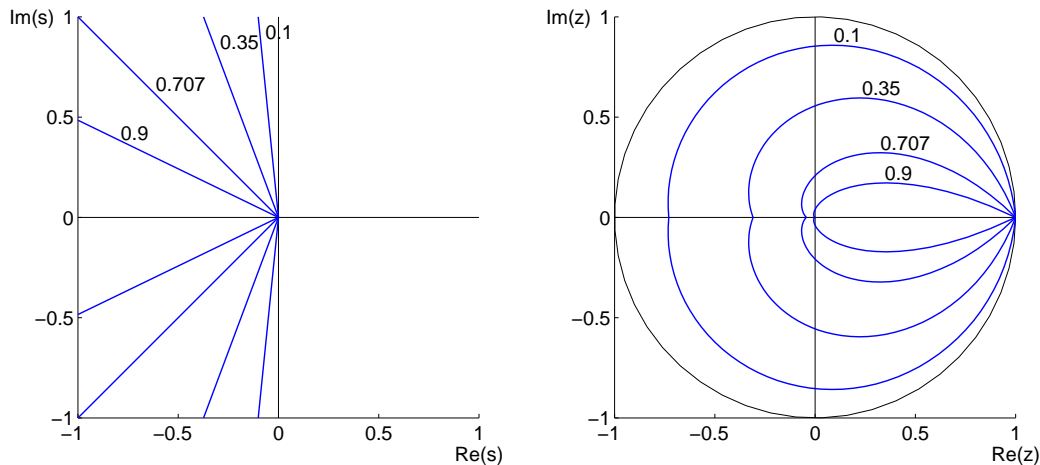


Abbildung 1.6: Links: Dämpfung in der  $s$ -Ebene. Rechts: Dämpfung in der  $z$ -Ebene. Die Zahlen entsprechen jeweils der Dämpfung  $D$ .

Bei gleichzeitiger Begrenzung der Einschwingzeit  $T_{\text{ein}}$  und der Dämpfung  $D$  resultiert als  $\Gamma$ -Gebiet die Schnittmenge der einzelnen  $\Gamma$ -Gebiete. Für den zeitkontinuierlichen Fall entstehen dadurch  $\Gamma$ -Gebiete, die mit der Methode der singulären Frequenzen für polynomiale Regleranteile zweiter Ordnung nicht behandelt werden können. Die Form des  $\Gamma$ -Gebietes, das für zeitdiskrete Systeme beide Kriterien erfüllt, zeigt schematisch Abbildung 1.7, links. Ein Kreis innerhalb des exakten  $\Gamma$ -Gebietes repräsentiert eine restriktive Näherung für diese Kurve (Abbildung 1.7, rechts).

Der Konferenzbeitrag [AKB02] stellt eine einfache Möglichkeit vor, den größten restriktiven Kreis zu finden. Ausgegangen wird von einem Kreis, der die Berandung des  $\Gamma$ -Gebietes  $\partial\Gamma$  für ausschließliche Begrenzung der Einschwingzeit  $T_{\text{ein}}$  beschreibt. Bei Fixierung des rechten Schnittpunktes dieses Kreises mit der reellen Achse muss der Radius des Kreises solange verringert werden, bis der Kreis vollständig im  $\Gamma$ -Gebiet Platz findet.

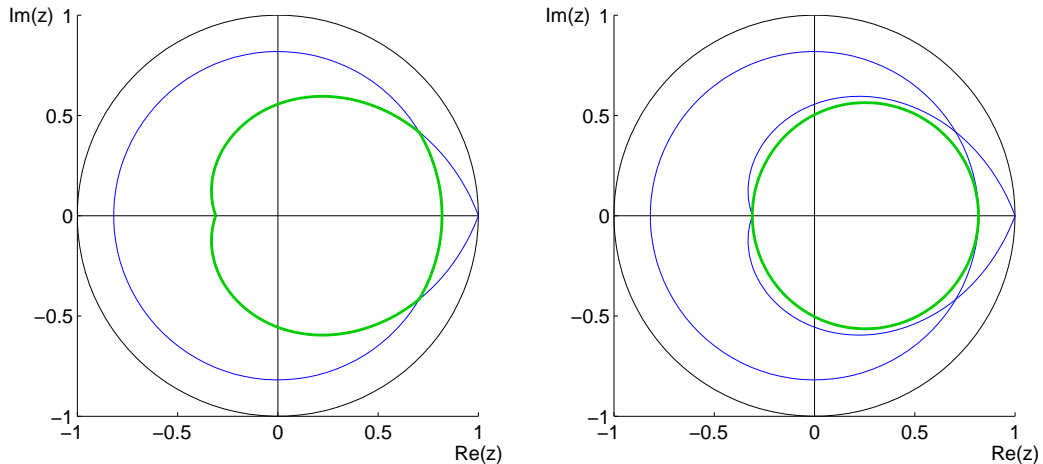


Abbildung 1.7:  $\Gamma$ -Gebiet für gleichzeitiges Einstellen von Einschwingzeit  $T_{\text{ein}}$  und Dämpfung  $D$ . Links: Exaktes  $\Gamma$ -Gebiet. Rechts: Näherung durch einen Kreis.

Die sich in Unterabschnitt 1.3 ergebenden  $\Gamma$ -Gebiete erlauben also für zeitkontinuierliche und zeitdiskrete System eine Festlegung der maximalen Einschwingzeit  $T_{\text{ein}}$ . Eine gleichzeitige Behandlung der Dämpfung  $D$  ist mit den hier behandelten Regleranteilen zweiter Ordnung nur bei zeitdiskreten Systemen möglich.

## 1.10 Totzeit

Eine wesentliche Eigenheit der Methode der singulären Frequenzen ist, dass sie die Behandlung von Systemen mit unbestimmter Totzeit zulässt. Dieser Abschnitt beschreibt kurz die Auswirkungen einer Totzeit. Man kann die Totzeit der Klas-

se der unbestimmten Parameter  $\mathbf{q}$  zuordnen. In den Berechnungen wird sie zur Verdeutlichung jedoch explizit angegeben.

Nach [LW02] wirkt sich bei zeitdiskreten Systemen, mit der Abtastzeit  $T_S$ , eine Totzeit  $L = nT_S$  durch einen zusätzlichen Faktor  $z^{-n}$  im Zähler, bzw.  $z^n$  im Nenner der Regelstrecke aus. Im Rahmen dieser Diplomarbeit werden die Berechnungen eingeschränkt auf Totzeiten, die ein ganzzahliges Vielfaches von der Abtastzeit  $T_S$  sind. So erfolgen die Berechnungen für das Gebiet aller stabilisierenden Regler für diesen Fall analog zu Systemen ohne Totzeit.

Die Auslegung von Reglern für zeitkontinuierliche Totzeitsysteme behandeln die Arbeiten [Baj01] und [Hoh02] ausführlich. Hier soll eine Zusammenfassung der darin aufgeführten Ergebnisse gegeben werden, welche die vorliegende Arbeit betreffen.

Die Übertragungsfunktion von zeitkontinuierlichen, totzeitbehafteten Systemen mit der komplexen Variablen  $s$  lässt sich als Produkt einer Übertragungsfunktion ohne Totzeit und einer Exponentialfunktion darstellen

$$G(s, L, \mathbf{q}) = \frac{A(s, \mathbf{q})}{N(s, \mathbf{q})} \exp(-sL). \quad (1.66)$$

Damit ändert sich das charakteristische Polynom (1.19) zu

$$P(s, \mathbf{c}, L, \mathbf{q}) = A(s, \mathbf{q})(c_0 + c_1s + c_2s^2) + \underbrace{N(s, \mathbf{q}) \exp(sL)}_{B(s, L, \mathbf{q})}. \quad (1.67)$$

Es gehört nun der Klasse der Quasipolynome an. Die Auswirkung der Exponentialfunktion sind trigonometrische Funktionen im Realteil  $R_{P_c}$  und im Imaginärteil  $I_{P_c}$  des charakteristischen Polynoms (1.67). Sie bewirken, dass Gleichung (1.36) eine unendliche Anzahl an Lösungen besitzt, die identisch mit den singulären Frequenzen  $s_S^i$  sind. Dadurch existiert auch eine unendliche Anzahl an singulären Geraden  $\lambda(s_S^i)$ .

Der Betrag der singulären Frequenzen  $s_S^i$  überschreitet alle Grenzen, so dass die Behandlung von Kreisstabilität für diesen Fall keinen Sinn macht. Die Untersuchungen müssen auf den Fall der  $\sigma$ -Stabilität eingeschränkt werden.

Die Berechnung des Gebietes aller stabilisierenden Regler kann nur eine endliche Anzahl an singulären Geraden  $\lambda(s_S^i)$  berücksichtigen. Glücklicherweise nähern sich für große Frequenzen die singulären Geraden den sogenannten infiniten Wurzelgrenzen an und stellen keine aktiven Grenzen im Parameterraum dar. Eine infinite Wurzelgrenze ist eine singuläre Gerade, die zu der singulären Frequenz  $s_S^i = \infty$  gehört. Für Totzeitsysteme existieren entweder keine oder zwei infinite Wurzelgrenzen.

Die Einschränkung auf eine endliche Anzahl von singulären Frequenzen wird erreicht, durch Begrenzung nach oben von  $\alpha$  aus Gleichung (1.37) bzw. (1.39). Die im Unendlichen liegende singuläre Frequenz muss allerdings trotz Begrenzung von  $\alpha$  berücksichtigt werden. Für die Bestimmung einer geeigneten oberen Grenze für  $\alpha$  kann folgende Daumenregel verwendet werden:

Die obere Schranke für  $\alpha$  sollte größer gewählt werden, als der größte Imaginärteil der Wurzeln des Polynoms  $N(s, \mathbf{q})$  aus (1.67).

Die Begründung ist, dass der Eigenwert des geschlossenen Regelkreises, der  $\partial\Gamma$  an der größten Frequenz kreuzt, sehr wahrscheinlich auch der Eigenwert des offenen Regelkreises mit dem größten Imaginärteil ist.

## 1.11 Algorithmus

Die in diesem Kapitel aufgeführte Theorie lässt sich in einem Algorithmus zusammenfassen. Die grundlegenden Schritte eines Algorithmus zur Berechnung aller stabilen Polygone für einen Betriebspunkt aus der *Q-Box* für ein festes  $r_1 = r_1^*$  lauten:

1. Falls ein PID-Regler verwendet wird, transformiere den Regelkreis durch die Vorabtransformation  $\mathbf{T}_{\text{PID}, s/z}$  in die Koordinaten  $\mathbf{c}$ .
2. Transformationsmatrix  $\mathbf{T}_{a=0}$  bzw.  $\mathbf{T}_{a \neq 0}$  festlegen und den Regelkreis in die Koordinaten  $\mathbf{r}$  transformieren.
3. Für  $r_1 = r_1^*$  die singulären Frequenzen  $\{z_S^i\}$  innerhalb des vorgegebenen Intervalls  $I_{\text{ges}}$  für  $\alpha$  bestimmen.
4. Die zugehörigen singulären Geraden  $\{\lambda(z_S^i)\}$  berechnen.
5. Alle Polygone in der  $(r_0, r_2)$ -Ebene finden.
6. Jedes Polygon überprüfen, ob es ein inneres Polygon ist.
7. Jedes innere Polygon nach  $\Gamma$ -Stabilität überprüfen.
8. Rücktransformation der stabilen Polygone in die Ausgangsparameter, also in die Reglerparameter  $\mathbf{c}$  bzw.  $\mathbf{k}$ .

Liegt der Regler bereits in einer Form vor, die ein Polynom zweiten Grades enthält, entfällt der erste Schritt, die Vorabtransformation. Das gesamte dreidimensionale Gebiet aller stabilisierenden Parameter ergibt sich durch Rastern des Parameters  $r_1$ . Dazu müssen die Schritte 3–7 für jeden Rasterpunkt ausgeführt werden. Wie ein geeignetes Raster für  $r_1$  gefunden werden kann, beschreibt der nächste Abschnitt.

Die in Schritt 7 erwähnte Überprüfung der  $\Gamma$ -Stabilität gestaltet sich für totzeitbehaftete, zeitkontinuierliche Systeme schwierig, da die Lage der Eigenwerte nicht berechnet werden kann. Aus diesem Grund wurde dieser Schritt entfernt und der Algorithmus zudem modifiziert. Der modifizierte Algorithmus für ein festes  $r_1 = r_1^*$  für einen Betriebspunkt lautet:

1. Falls ein PID-Regler verwendet wird, transformiere den Regelkreis durch die Vorabtransformation  $\mathbf{T}_{\text{PID}, s/z}$  in die Koordinaten  $\mathbf{c}$ .
2. Transformationsmatrix  $\mathbf{T}_{a=0}$  bzw.  $\mathbf{T}_{a \neq 0}$  festlegen und den Regelkreis in die Koordinaten  $\mathbf{r}$  transformieren.
3. Für  $r_1 = r_1^*$  die singulären Frequenzen  $\{z_S^i\}$  innerhalb des vorgegebenen Intervalls  $I_{\text{ges}}$  für  $\alpha$  bestimmen.
4. Die zugehörigen singulären Geraden  $\{\lambda(z_S^i)\}$  berechnen.
5. Alle Polygone  $\{\Pi^l\}$  in der  $(r_0, r_2)$ -Ebene finden.
6. Beliebiges Polygon als Referenzpolygon  $\Pi_R \in \{\Pi^l\}$  festlegen.
7. Schleife für alle Polygone  $\Pi \in \{\Pi^l\}$  durchlaufen:  
Ordne dem Polygon  $\Pi$  eine Stabilitätszahl  $\gamma$  zu, welche die Differenz an Eigenwerten angibt, die sich, im Vergleich zum Referenzpolygon  $\Pi_R$ , zusätzlich innerhalb des  $\Gamma$ -Gebietes befinden.<sup>1</sup>
8. Rücktransformation der Polygone mit maximalem  $\gamma$  in die Ausgangsparameter, also in die Reglerparameter  $\mathbf{c}$  bzw.  $\mathbf{k}$ .

Das Ergebnis sind alle inneren Polygone für  $r_1 = r_1^*$  mit maximaler Anzahl an stabilen Eigenwerten. Da die Ebene, in der die Polygone für ein festes  $r_1$  liegen, unabhängig von der Regelstrecke ist (Theorem 1.1), kann die Schnittmenge der inneren Polygone für alle Betriebspunkte berechnet werden. Das Resultat sind konvexe Polygone, die für jeden Betriebspunkt eine Teilmenge seiner inneren Polygone darstellen. Ist für einen Punkt innerhalb dieser Polygone der geschlossene Regelkreis robust  $\Gamma$ -stabil, so sind diese Polygone robust  $\Gamma$ -stabil. Für totzeitbehaftete, zeitkontinuierliche Systeme muss der Anwender entscheiden, ob das Gebiet die Stabilitätskriterien erfüllt (siehe Unterabschnitt 2.3.10).

## 1.12 Entstehen/ Verschwinden von stabilen Polygonen

Die Frage nach der Wahl des Rasterparameters  $r_1$  ist bisher noch nicht geklärt. Für die Festlegung eines geeigneten Rasters wird die Menge aller *stabilen*  $r_1$  benötigt, also die Menge aller  $r_1$ , für die robust  $\Gamma$ -stabile Polygone existieren.

Die Bestimmung dieser Menge wurde in der Fachliteratur bislang jedoch kaum behandelt. Für PT1-Systeme mit zusätzlicher Totzeit konnte die Menge aller hurwitzstabilen  $r_1$  zwar bestimmt werden ([SDB02]), eine universelle Lösung für diese Problematik existiert aber nicht.

<sup>1</sup>Dieser Schritt benötigt die Überquerung von singulären Geraden, welche Unterabschnitt 1.8.1 herleitet.

Dieser Abschnitt gibt eine Anleitung zur Bestimmung aller stabilen  $r_1$ . Zunächst wird allerdings die geklärt, für welche Konstellationen das stabile Gebiet in Richtung der  $r_1$ -Achse enden kann.

### 1.12.1 1. Möglichkeit: Lokalen Extrema von $r_1$

Wie in Unterabschnitt 1.6.2 bereits erwähnt, spielen die lokalen Extrema  $\alpha_E^x$  von  $r_1$  eine besondere Rolle für die Begrenzung des stabilen dreidimensionalen Gebietes. Wird eines von ihnen in  $r_1$ -Richtung überquert, so entstehen oder verschwinden zwei singuläre Frequenzen,  $\alpha_S^i$  und  $\alpha_S^{i+1}$ , und mit ihnen die beiden nahezu parallelen singulären Geraden  $\lambda(\alpha_S^i)$  und  $\lambda(\alpha_S^{i+1})$  (vgl. Abbildung 1.8).

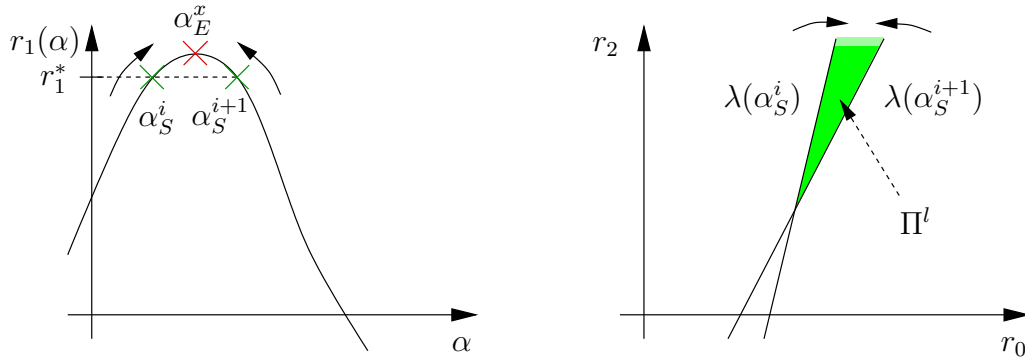


Abbildung 1.8: Links: Zwei aufeinander zulaufende singuläre Frequenzen  $\alpha_S^i$  und  $\alpha_S^{i+1}$ . Rechts: Die daraus resultierende Bewegung der dazugehörigen singulären Geraden  $\lambda(\alpha_S^i)$  und  $\lambda(\alpha_S^{i+1})$ .  $\Pi^l$  ist ein stabiles Polygon.

Falls diese singulären Geraden ein Teil der Begrenzung des einzigen stabilen Polygons  $\Pi^l$  darstellen, beginnt oder endet an dieser Stelle von  $r_1$  das stabile dreidimensionale Gebiet.

In seltenen Fällen begrenzen die beiden singulären Geraden  $\lambda(\alpha_S^i)$  und  $\lambda(\alpha_S^{i+1})$  zwei unterschiedliche stabile Polygone  $\Pi^l$  und  $\Pi^{l+1}$ . Für diese Situation vereinigen oder trennen sich die beiden stabilen Polygone bei Überschreitung des lokalen Extremum  $\alpha_E^x$  in  $r_1$ -Richtung (vgl. Abbildung 1.9). Für eine solche Anordnung stellt das lokale Extremum  $\alpha_E^x$  also keine Begrenzung des stabilen Gebietes in  $r_1$ -Richtung dar.

Wenn zwei singuläre Geraden entstehen oder verschwinden, die kein stabiles Polygon begrenzen, hat das betreffende lokale Extremum  $\alpha_E^x$  keine Bedeutung für das stabile Gebiet.

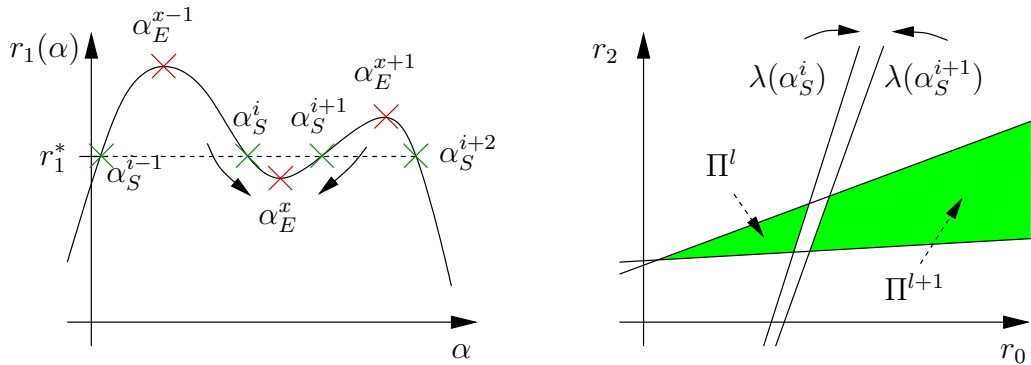


Abbildung 1.9: Links: Vier sich nach unten bewegendes singuläre Frequenzen. Zwei von ihnen,  $\alpha_S^i$  und  $\alpha_S^{i+1}$ , laufen aufeinander zu. Rechts: Die daraus resultierende Bewegung der dazugehörigen singulären Geraden  $\lambda(\alpha_S^i)$  und  $\lambda(\alpha_S^{i+1})$ .  $\Pi^l$  und  $\Pi^{l+1}$  sind zwei stabile Polygone.

### 1.12.2 2. Möglichkeit: Schnittpunkt von mindestens drei Geraden

Neben der Begrenzung des stabilen Gebietes an lokalen Extrema in Richtung der  $r_1$ -Achse, existiert noch eine alternative Begrenzungsmöglichkeit.

Sei  $\Pi^l$  ein stabiles Polygon, welches durch drei singuläre Geraden gebildet wird. Wenn eine dieser singulären Geraden ( $\lambda(\alpha_S^i)$ ) den Schnittpunkt der beiden anderen singulären Geraden überquert, verschwindet die Fläche von  $\Pi^l$  (siehe Abbildung 1.10). Das stabile Gebiet kann demnach auch im Bereich zwischen zwei lokalen Extrema beginnen oder enden.

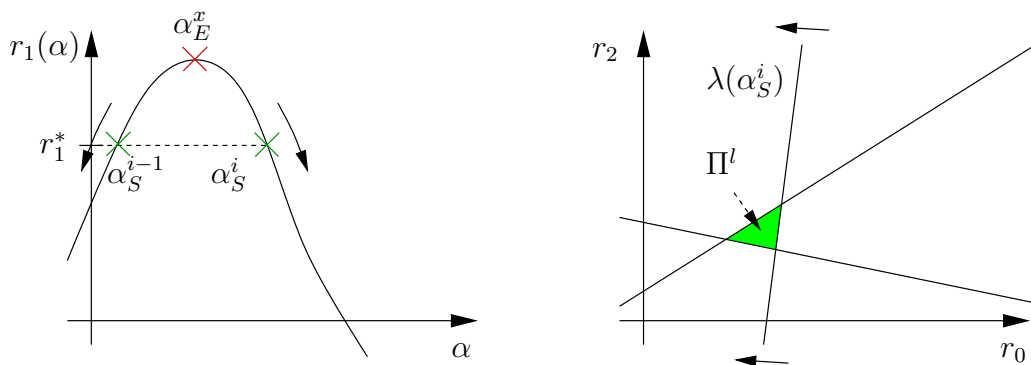


Abbildung 1.10: Drei singuläre Geraden schneiden sich in einem Punkt. Links: Zwei sich entlang der  $r_1(\alpha)$ -Kurve bewegendes singuläre Frequenzen  $\alpha_S^{i-1}$  und  $\alpha_S^i$ . Rechts: Die daraus resultierende Bewegung der singulären Geraden  $\lambda(\alpha_S^i)$ .  $\Pi^l$  ist ein stabiles Polygon.

### 1.12.3 Vorgehen zur Bestimmung des Rasters für $r_1$

Aufgrund der in den beiden vorangehenden Unterabschnitten getroffenen Feststellungen kann man darauf schließen, dass stabile  $r_1$  am wahrscheinlichsten in dem Bereich zu finden sind, in welchem eine Parallele zur  $r_1$ -Achse die Kurve  $r_1(\alpha)$  am häufigsten schneidet. Somit liegt folgendes Vorgehen zur Bestimmung aller stabilen  $r_1$  nahe:

1. Definiere ein Raster für den Bereich, an dem eine Parallele zur  $r_1$ -Achse die meisten Schnitte mit  $r_1(\alpha)$  besitzt.
2. Durchlaufe für jeden Rasterpunkt den Algorithmus zur Bestimmung aller inneren Polygone (Abschnitt 1.11) und untersuche die inneren Polygone nach Stabilität.
3. Falls an einem Ende des Rasters die Fläche der stabilen Polygone nicht verschwindet, erweitere das Raster an diesem Ende um den  $r_1$ -Bereich, in dem zwei singuläre Frequenzen weniger auftreten (also in  $r_1$ -Richtung bis zum nächsten lokalen Extremum).
4. Falls das stabile Gebiet vor einem Ende des Rasters in einem Punkt zusammenläuft, reduziere das Raster um die Rasterpunkte, an denen keine stabilen Polygone existieren.
5. Falls überhaupt keine stabilen Polygone gefunden wurden, erweitere das Raster, so dass auch der  $r_1$ -Bereich eingeschlossen ist, in dem zwei singuläre Frequenzen weniger auftreten (in  $r_1$ -Richtung bis zum nächsten lokalen Extremum). Umfasst das Raster bereits den  $r_1$ -Bereich, in dem nur eine einzige singuläre Frequenz auftritt, so breche die Suche ab, da der Regelkreis nicht stabilisierbar ist.
6. Falls das Raster verändert wurde, gehe zu Schritt 2. Ansonsten umfasst das Raster alle stabilen  $r_1$ .

# Kapitel 2

## Anleitung zur Bedienung von *RobSin*

Bei der Entwicklung des Software-Werkzeuges *RobSin* wurde großer Wert auf eine intuitive Bedienung gelegt. Dennoch benötigt dieses Tool einige Erklärungen. Dieses Kapitel führt den Anwender anhand eines Beispiels in die vollständige Bedienung von *RobSin* ein.

Man betrachte dazu den Regelkreis, bestehend aus der Regelstrecke

$$G(s) = \frac{1}{(s+1)^8} \quad (2.1)$$

zusammen mit dem PID-Regler

$$C(s) = k_P + \frac{k_I}{s} + k_D s. \quad (2.2)$$

Gesucht ist das dreidimensionale Gebiet aller stabilisierenden Reglerparameter  $(k_I, k_P, k_D)$  des PID-Reglers.

### 2.1 Starten von *RobSin*

Das Programm *RobSin* benötigt MATLAB<sup>®</sup> ab Version 6.5 (R13) mit den beiden Toolboxen: *Control System Toolbox* und *Extended Symbolic Math Toolbox*. Zum Starten von *RobSin* muss MATLAB<sup>®</sup> bereits laufen und dort der aktive Pfad gleich dem *RobSin*-Pfad sein. Die Eingabe des Befehls

```
robsin
```

im *Command Window* von MATLAB<sup>®</sup> startet das Software-Werkzeug.<sup>1</sup> Es erscheint nun eine Ausgabe, welche dem Anwender eine Erweiterung in der Liste der Suchpfade (*MATLAB search path*) und die aktuelle Uhrzeit mitteilt:

---

<sup>1</sup>Bei einigen Versionen von MATLAB<sup>®</sup> funktioniert die Darstellung transparenter Flächen nicht. In diesem Fall muss *RobSin* mit einem Parameter aufgerufen werden, um auf die Verwendung von transparenten Flächen zu verzichten: `robsin('opaque')`.

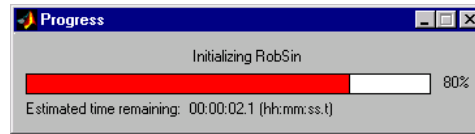


Abbildung 2.1: Die Anzeige des Fortschritts beim Initialisierungsvorgang.

```
'C:\Programme\MATLAB\robsin' added to search path.
'C:\Programme\MATLAB\robsin\util' added to search path.
RobSin started: 24-Dec-2003 17:30:00
```

Anschließend öffnet sich ein Fenster, welches den Fortschritt des Initialisierungsvorgangs von *RobSin* anzeigt (siehe Abbildung 2.1). Dieses Fenster schließt sich nach erfolgreicher Initialisierung selbständig. Es öffnet sich nun die grafische Benutzeroberfläche (engl.: *graphical user interface* oder *GUI*) von *RobSin*, wie sie Abbildung 2.2 zeigt.

## 2.2 Aufbau der Benutzeroberfläche

Das GUI von *RobSin* strukturiert sich analog zu anderen gängigen Software-Werkzeugen. Es enthält eine Menüleiste am oberen Ende und darunter vier Rahmen, welche die Informationen anschaulich visualisieren. Dieser Abschnitt gibt einen Überblick über die Menüpunkte und die Aufgaben der Rahmen.

### 2.2.1 Menüleiste

Die Menüleiste besteht aus den sieben Menüspalten *File*, *View*, *Design*, *Analysis*, *Tools*, *Window* und *Help*. Die einzelnen Menüpunkte, die dazugehörigen Tastenkombinationen und ihre Beschreibungen listet Tabelle 2.1 auf. Die genaue Bedeutung der darin aufgelisteten Beschreibungen klärt sich im Laufe dieses Kapitels.

Menüspalte	Menüpunkt	Tastenkombination	Beschreibung
File	New	<Strg>-N	<i>RobSin</i> in den Ausgangszustand bringen.
	Load...	<Strg>-L	Frühere Session laden.
	Save...	<Strg>-S	Aktuelle Session speichern ohne Eingabe eines neuen Dateinamens.

(Fortsetzung auf der nächsten Seite.)

Menüspalte	Menüpunkt	Tastenkombination	Beschreibung
	Save As...	<Strg>-A	Aktuelle Session speichern mit Eingabe eines Dateinamens.
	Export		Eine Variable aus <i>RobSin</i> in den <i>Workspace</i> exportieren unter Eingabe eines beliebigen Variablennamens
	Options...		<i>Optionsfenster</i> zeigen.
	Close	<Strg>-Q	<i>RobSin</i> beenden.
View	Design: Controller and Gamma Region	<Strg>-1	Ansicht wechseln, so dass die beiden Rahmen <i>Controller Specification</i> und <i>Gamma Region</i> erscheinen.
	Analysis: 2D Controller Parameter Selection and Analysis Plots	<Strg>-2	Ansicht wechseln, so dass die beiden Rahmen <i>Controller Parameter Selection</i> und <i>Analysis Plots</i> erscheinen.
	3D Plot of stabilizing controller parameters	<Strg>-3	<i>Stabilizing-3D-Region</i> -Fenster öffnen bzw. schließen.
	Singular Frequencies Generator - $r_1(\alpha)$	<Strg>-4	<i>Singular-Frequencies-Generator</i> -Fenster öffnen bzw. schließen.
Design	Calculate 3D Controller Region	<Strg>-R	Für das definierte Intervall für $r_1$ das gesamte dreidimensionale Gebiet aller stabilisierenden Reglerparameter berechnen und darstellen.
	Calculate Singular Frequencies Generator		Die Kurve $r_1(\alpha)$ berechnen und darstellen.
	Calculate Single Slice		Innere Polygone für den ausgewählten Wert von $r_1$ berechnen und darstellen.
	Generate Singular Frequencies		Alle singulären Frequenzen für den ausgewählten Wert von $r_1$ berechnen und im <i>Echo</i> -Rahmen ausgeben.

(Fortsetzung auf der nächsten Seite.)

Menüspalte	Menüpunkt	Tastenkombination	Beschreibung
	Generate Singular Lines		Alle singulären Geraden für den ausgewählten Wert von $r_1$ berechnen, in Dateien abspeichern und die Dateinamen im <i>Echo</i> -Rahmen ausgeben.
Analysis	Recalculate Analysis Data		Daten für die Analyse berechnen, so dass ein Regelkreis auch ohne Initialisierung des <i>singf</i> -Objektes oder Berechnung der Polygone analysiert werden kann.
	<none>		Kein Analyseverfahren anwenden.
	Step		Sprungantwort des geschlossenen Regelkreises für die Analyse verwenden.
	Impulse		Impulsantwort des geschlossenen Regelkreises für die Analyse verwenden.
	Bode		Bode-Diagramm des offenen Regelkreises für die Analyse verwenden.
	Bode Magnitude		Betrag des Bode-Diagramms des offenen Regelkreises für die Analyse verwenden.
	Nyquist		Nyquist-Diagramm des offenen Regelkreises für die Analyse verwenden.
	Sigma-Nyquist		Nyquist-Diagramm des offenen Regelkreises, bei dem $s$ durch $s + \sigma_{\max}$ ersetzt wurde, für die Analyse verwenden. Dieses Stabilitätskriterium eignet sich für die Untersuchung von totzeitbehafteten Systemen nach $\sigma$ -Stabilität (siehe Unterabschnitt 2.3.10).
(Fortsetzung auf der nächsten Seite.)			

Menüspalte	Menüpunkt	Tastenkombination	Beschreibung
	Nichols		Nichols-Diagramm des offenen Regelkreises für die Analyse verwenden.
	Pole/Zero		Pol-Nullstellen-Diagramm des geschlossenen Regelkreises für die Analyse verwenden.
Tools	Zoom In		Vergrößerungsmodus für alle Diagramme innerhalb des <i>RobSin</i> -Fensters ein- bzw. ausschalten. <sup>2</sup>
Window	(Mehrere Menüpunkte mit den Namen der in MATLAB <sup>®</sup> geöffneten Fenster)		Das ausgewählte Fenster in den Vordergrund bringen.
Help	German PDF Documentation	<Strg>-H	Dieses Dokument im Adobe <sup>®</sup> Acrobat <sup>®</sup> Reader öffnen.
	(Weitere Dokumente, die für die Arbeit interessant sind)		Ausgewähltes Dokument im Adobe <sup>®</sup> Acrobat <sup>®</sup> Reader öffnen.
	About RobSin		Informationen über <i>RobSin</i> zeigen.

Tabelle 2.1: Das Menü von *RobSin*.

### 2.2.2 Rahmen

Die nach dem Start sichtbaren vier Rahmen lässt Abbildung 2.2 erkennen. Dies sind der *Plant-Specification*-Rahmen, der *Controller-Specification*-Rahmen, der *Gamma-Region*-Rahmen und der *Echo*-Rahmen. Es existieren noch zwei weitere Rahmen, der *Controller-Parameter-Selection*-Rahmen und der *Analysis-Plots*-Rahmen. Der Menüpunkt *Analysis: 2D Controller Parameter Selection and Analysis Plots* in der Menüspalte *View* bringt sie in den Vordergrund. Die einzelnen

<sup>2</sup>MATLAB<sup>®</sup> Version 6.5 (R13) hat Probleme bei der interaktiven Vergrößerung von hintereinanderliegenden Diagrammen. Dies äußert sich bei *RobSin* beispielsweise dadurch, dass die Vergrößerung des  $\Gamma$ -Gebietes im *Gamma-Region*-Rahmen (Unterabschnitt 2.2.2) mit der Maus nicht möglich ist.

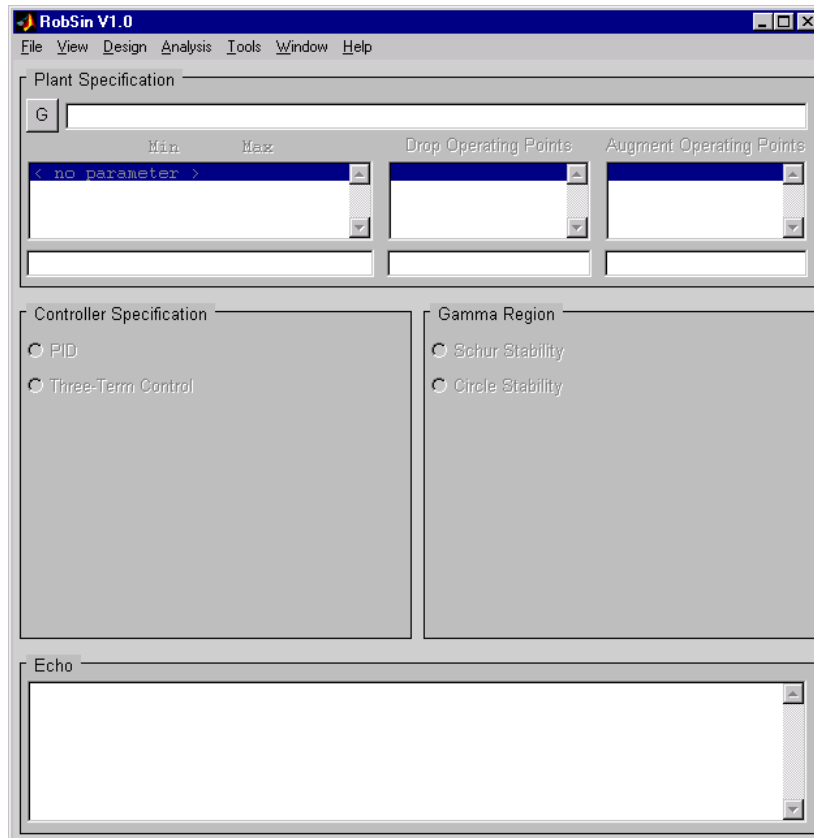


Abbildung 2.2: Das GUI von *RobSin* unmittelbar nach dem Start.

Rahmen erfüllen unterschiedliche Aufgaben.

### Der *Plant-Specification*-Rahmen

Der *Plant-Specification*-Rahmen erlaubt dem Anwender die Eingabe einer Regelstrecke. Dabei gibt es zwei verschiedene Möglichkeiten,

1. die Eingabe einer Regelstrecke mit Parameterunsicherheiten  $\mathbf{q}$  und mit jeweils der oberen und unteren Grenze der Parameter,
2. die explizite Eingabe mehrerer Regelstrecken ohne Parameter.

Aus der ersten dieser beiden Möglichkeiten ergibt sich als Betriebsbereich  $Q$  ein Hyperrechteck im Raum der Parameter  $\mathbf{q}$ , welches auch als *Q-Box* bezeichnet wird ([ABB<sup>+</sup>02]). Dabei ermöglicht es dieser Rahmen, einzelne Arbeitspunkte herauszunehmen und zusätzliche Punkte mit in die Berechnungen einzubeziehen. Diese Form der Eingabe erlaubt eine schnelle und übersichtliche Festlegung der Regelstrecke, jedoch mit der Einschränkung, nur jeweils eine Struktur der Regelstrecke zu untersuchen. Die Listbox mit der Überschrift „Drop Operating Points“ besitzt ein Kontextmenü, welches das Herausnehmen einzelner Arbeitspunkte erleichtert.<sup>3</sup>

Dagegen lässt die zweite Eingabemöglichkeit die Behandlung verschiedener Strukturen zu, allerdings zum Preis einer weniger übersichtlicheren Eingabe der Regelstrecke. Die Reihenfolge der einzelnen Einträge kann bei dieser Eingabemöglichkeit über ein Kontextmenü weiteres angepasst werden.

Eine Schaltfläche bewirkt einen Wechsel zwischen diesen beiden Zuständen. Ihre Aufschrift gibt an, zu welcher Eingabemöglichkeit das Programm bei Aktivierung der Schaltfläche wechselt.  $G$  steht für die explizite Eingabe der Regelstrecken  $G_i$ ,  $Q$  für die Eingabe einer Regelstrecke mit *Q-Box*.

Der *Plant-Specification*-Rahmen legt außerdem die komplexe Variable fest, entweder  $s$ , als Laplace-Variable für zeitkontinuierliche Systeme, oder  $z$ , als komplexe Variable der  $z$ -Transformation für zeitdiskrete Systeme. Für zeitkontinuierliche Systeme erscheint ein Eingabefeld zur Festlegung der Abtastzeit  $T_S$ .

Das Software-Werkzeug überprüft jeden Eintrag des Anwenders in einem Eingabefeld auf Fehlerfreiheit und erzeugt im Falle einer falschen Eingabe eine Warnung. Es versteht auch verschiedene MATLAB<sup>®</sup> Funktionen, wie beispielsweise `sqrt()`, `sin()` oder `exp()` und wertet sie unmittelbar nach der Eingabe aus.

Die Stärke dieses Werkzeuges besteht unter anderem in der Berücksichtigung einer konstanten, aber unsicheren Totzeit  $L$  in der Regelstrecke. Wie eine Regelstrecke um eine Totzeit  $L$  erweitert wird beschreibt Abschnitt 1.10.

---

<sup>3</sup>Kontextmenüs öffnen sich bei Betätigung der rechten Maustaste auf dem entsprechenden Objekt.

### Der *Controller-Specification*-Rahmen

Die Spezifikation des Reglers findet im *Controller-Specification*-Rahmen statt. Die Strukturen der möglichen Regler, aufgeteilt nach zeitkontinuierlichen und zeitdiskreten Systeme, listet Tabelle 2.2 auf. Dabei steht  $T_R$  für die Verzögerungszeitkonstante des PID- bzw. PD-Reglers. Die Funktionen  $n(s)$  und  $d(s)$

Bezeichnung des Reglers	$C(s)$	$D(z)$
PID	$k_P + \frac{k_I}{s} + \frac{k_D s}{T_R s + 1}$	$k_P + \frac{k_I}{1 - z^{-1}} + k_D(1 - z^{-1})$
Three-Term Control	$\frac{n(s) \cdot (c_0 + c_1 s + c_2 s^2)}{d(s)}$	$\frac{n(z) \cdot (c_0 + c_1 z + c_2 z^2)}{d(z)}$

Tabelle 2.2: Mögliche Reglerstrukturen.

bzw.  $n(z)$  und  $d(z)$  sind Polynome in  $s$  bzw.  $z$ . Die Three-Term-Control-Struktur eignet sich besonders für die in Abschnitt 1.1 bereits erwähnte Feineinstellung von Parametern des Reglers.

### Der *Gamma-Region*-Rahmen

Das  $\Gamma$ -Gebiet ist das erlaubte Gebiet aller Pole der Übertragungsfunktion des geschlossenen Regelkreises. Der *Gamma-Region*-Rahmen lässt die Spezifikation des  $\Gamma$ -Gebietes zu und zeigt es in der komplexen  $s$ - bzw.  $z$ -Ebene.

Abhängig davon, ob zeitkontinuierliche oder zeitdiskrete Systeme betrachtet werden, sind verschiedene  $\Gamma$ -Gebiete erlaubt. Für zeitkontinuierliche Systeme lässt das Software-Werkzeug die Einstellungen „*Hurwitz Stability*“ (Hurwitz-Stabilität), „*Sigma Stability*“ ( $\sigma$ -Stabilität) und „*Circle Stability*“ (Kreisstabilität) zu, für zeitdiskrete „*Schur Stability*“ (Schur-Stabilität) und „*Circle Stability*“.

Nach Abschnitt 1.10 ist für zeitkontinuierliche Systeme mit Totzeit die Untersuchung nach Kreisstabilität jedoch sinnlos, so dass *RobSin* in diesem Fall diese Einstellmöglichkeit deaktiviert.

Nach der Auswahl der Form des  $\Gamma$ -Gebietes kann die Einstellung der dazugehörigen Parameter, sofern vorhanden, über zwei Arten erfolgen, nämlich per Eingabe mit der Tastatur oder interaktiv über Positionierung mit der Maus.

### Der *Echo*-Rahmen

Ein weiterer Rahmen mit dem Titel *Echo* gibt die Antworten des Programms auf Benutzereingaben und den Status von Berechnungen aus. Dieser *Echo*-Rahmen zeigt die letzten 200 erfolgten Aktionen der aktiven Sitzung in einer Listbox.

### Der *Controller-Parameter-Selection*-Rahmen

Im Ausgangszustand von *RobSin* sind zwei Rahmen nicht sichtbar. Der *Controller-Parameter-Selection*-Rahmen ist einer von ihnen (siehe Abbildung 2.8 und 2.12). Er zeigt das berechnete Gebiet in der  $(r_0, r_2)$ -Ebene und ermöglicht für die Analyse des Regelkreises die Auswahl der drei, in Abschnitt 1.5 vorgestellten Parameter  $\mathbf{r} = [r_0, r_1, r_2]$ .

Die Festlegung des Rasterparameters  $r_1$  erfolgt entweder über den Schieber, oder per Eingabe in das dafür vorgesehene Eingabefeld. Für die Auswahl der beiden anderen Parameter  $r_0$  und  $r_2$  lässt dieser Rahmen ebenfalls zwei Möglichkeiten zu, interaktiv durch Auswahl mit der Maus innerhalb des  $[r_0, r_2]$ -Koordinatensystems, und explizit durch Eingabe in das zugehörige Eingabefeld.

Durch Drücken der rechten Maustaste öffnet sich ein selbsterklärendes Kontextmenü, welches die Vergrößerung eines Ausschnittes des Koordinatensystems ermöglicht.

### Der *Analysis-Plots*-Rahmen

Der *Analysis-Plots*-Rahmen ist genauso wie der *Controller-Parameter-Selection*-Rahmen zu Beginn nicht sichtbar. Er zeigt die Diagramme für die Analyse des Regelkreises mit den zuvor ausgewählten Parametern  $\mathbf{r}$  (siehe Abbildung 2.8 und 2.12).

Dieser Rahmen unterstützt die gebräuchlichsten Analyseverfahren. Ihre Auswahl findet über ein *Popup*-Menü statt, oder alternativ über die *Analysis*-Menüspalte. Tabelle 2.1 listet die unterstützten Analyseverfahren auf.

Bei Analyse mehrerer Regelstrecken, erscheinen die einzelnen Kurven in unterschiedlichen Farben. Wird eine dieser Kurven mit der linken Maustaste angewählt, öffnet sich ein Text mit Informationen über die dazugehörige Regelstrecke.

Wie im *Controller-Parameter-Selection*-Rahmen öffnet sich auch hier, durch Drücken der rechten Maustaste, ein selbsterklärendes Kontextmenü, welches die Vergrößerung eines Ausschnittes aus dem Koordinatensystem ermöglicht.

Die Schaltfläche mit der Aufschrift „*Open Figure*“ öffnet ein extra Fenster, das die im *Analysis-Plots*-Rahmen dargestellten Kurven anzeigt. Der Inhalt dieses Fensters wird mit der *Control System Toolbox* erzeugt und stellt deswegen die darin implementierte Funktionalität bereit.

## 2.3 Behandlung des Beispiels mit *RobSin*

Dieser Abschnitt beschreibt die gesamte Funktionalität von *RobSin*, anhand des am Anfang dieses Kapitels vorgestellten Beispiels. Zunächst betrachte man den Fall der Hurwitz-Stabilität.

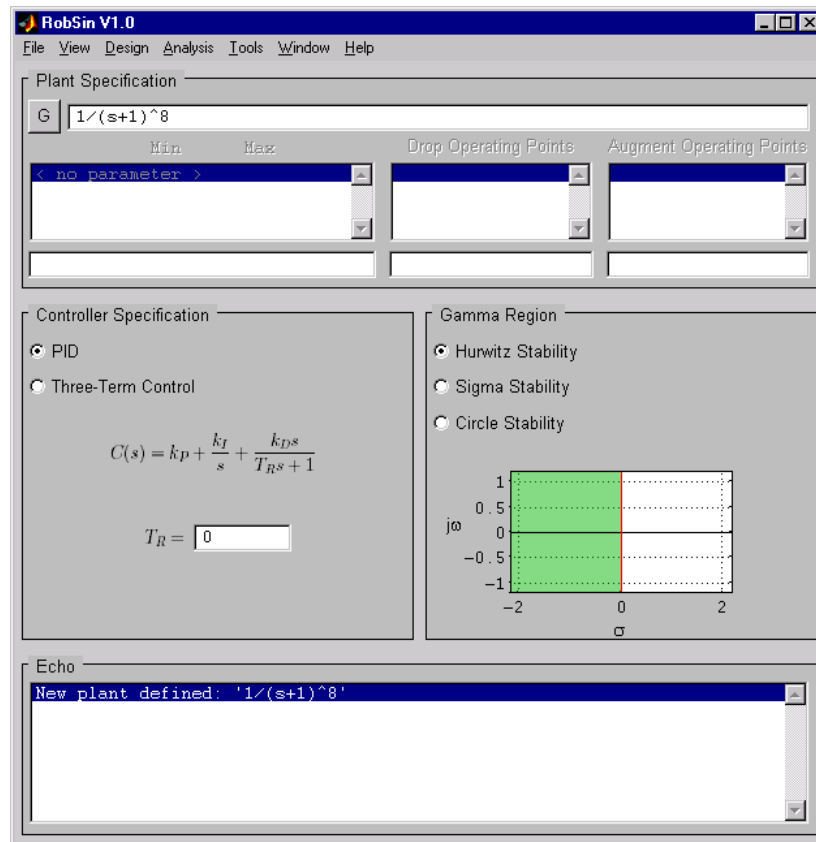


Abbildung 2.3: Die Oberfläche von *RobSin* nach Eingabe des Regelkreises.

### 2.3.1 Eingabe des Regelkreises

Die Formel der Regelstrecke  $G(s)$  wird im *Plant-Specification*-Rahmen im obersten Eingabefeld eingegeben (siehe Abbildung 2.3) mit

$$1/(s+1)^8$$

Daraufhin aktivieren sich die beiden Rahmen, der *Plant-Specification*-Rahmen und der *Gamma-Region*-Rahmen, und die Bestätigungsmeldung

New plant defined: '1/(s+1)^8'

erscheint im *Echo*-Rahmen. Der *Plant-Specification*-Rahmen zeigt die gewünschte Reglerstruktur des PID-Reglers bereits an. Es sind also keine Änderungen innerhalb dieses Rahmens nötig.

Im *Gamma-Region*-Rahmen ist die Hurwitz-Stabilität ausgewählt. Innerhalb dieses Rahmens müssen also ebenfalls keine Änderungen vorgenommen werden. Das  $\Gamma$ -Gebiet erscheint grün mit roter Begrenzung.

### 2.3.2 Generator der singulären Frequenzen

Das Software-Werkzeug *RobSin* bietet die Möglichkeit die Funktion  $r_1(\alpha)$  aus Gleichung (1.39) in einem separaten Fenster anzuzeigen. Die Schnittpunkte der Kurve  $r_1(\alpha)$  mit einer Parallelen zur  $\alpha$ -Achse bei festem  $r_1 = r_1^*$  ergeben nach Abschnitt 1.6 die singulären Frequenzen  $z_S^i = z(\alpha_S^i)$ . Aus diesem Grund heißt die Kurve  $r_1(\alpha)$  *Singular Frequencies Generator* (dt.: Singulärer-Frequenz-Generator).

Der Menüpunkt *Calculate Singular Frequencies Generator* in der Menüspalte *Design* startet die Berechnung dieser Kurve. Dazu initialisiert *RobSin* sein internes Objekt *singf*, welches eine Instanz der in Abschnitt 2.5 beschriebenen, gleichnamigen Klasse ist. Der *Echo*-Rahmen zeigt den Fortschritt der Berechnung. Die gesamte Ausgabe für den Initialisierungsvorgang des *singf*-Objektes lautet:

```
Please wait. Initializing singf object ...
  Initializing fields ... done
  Computing symbolical expressions ... done
  Creating function files ... done
  Generating critical frequencies ... done
The singf object constructed in 3.145 secs4
```

Die einzelnen Aktionen von *RobSin* waren also die Initialisierung der Felder des *singf*-Objektes, die Berechnung verschiedener symbolischer Ausdrücke, das Speichern dieser Ausdrücke als ausführbare Dateien und die Erzeugung der kritischen Frequenzen. Einige symbolische Ausdrücke sind die Funktion  $r_1(\alpha)$ , ihre Ableitung und ihr Nenner. Sie werden für die Berechnung der kritischen Frequenzen benötigt (siehe Unterabschnitt 1.6.2). Die Berechnung dieser Funktionen sind symbolisch und benötigen die *Extended Symbolic Math Toolbox*.

Nach vollendeter Berechnung von  $r_1(\alpha)$  liest sich die letzte Zeile im *Echo*-Rahmen:

```
Calculate r1(alpha) ... done
```

und ein Fenster mit dem Titel „*RobSin V1.0 - Singular Frequencies Generator*“ erscheint, welches die berechneten Daten anzeigt. Um Informationen über die Kurven zu erhalten, besteht die Möglichkeit eine Legende über den Menüpunkt *Legend* in der Menüspalte *Insert* zu erzeugen (siehe Abbildung 2.10). Weiterhin kann, wie beim *Analysis-Plots*-Rahmen, durch Auswahl mit der Maus ein Informationstext für die einzelnen Graphen angezeigt werden.

Eine geeignete Vergrößerung eines Ausschnittes dieser Kurve gibt zu erkennen, dass die meisten singulären Frequenzen für einen Wert von  $r_1$  innerhalb des

<sup>4</sup>Das Beispiel wurde an einem Intel® Pentium® II Rechner mit 400 MHz Taktfrequenz und 128 MB Speicher unter dem Betriebssystem Microsoft® Windows NT Version 4.0 mit Service Pack 6 bearbeitet.

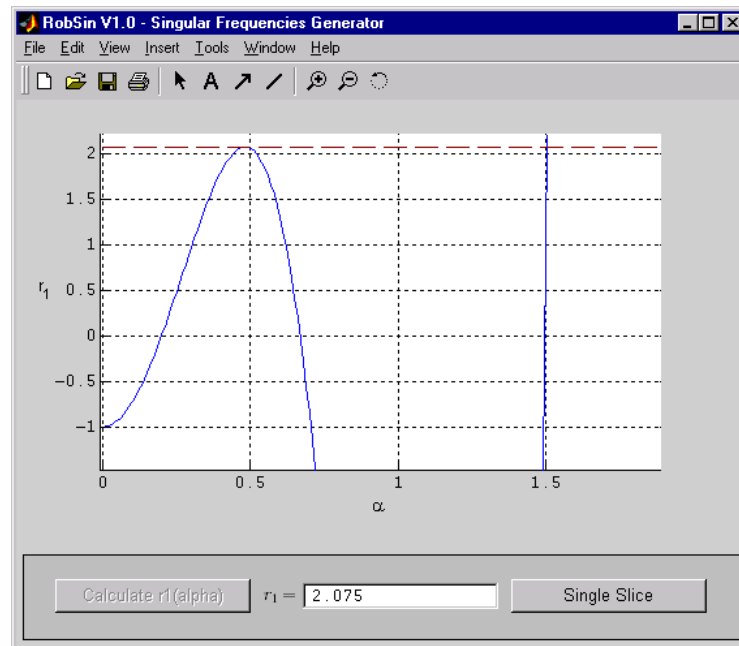


Abbildung 2.4: Das *Singular-Frequencies-Generator*-Fenster mit der Kurve  $r_1(\alpha)$  bei gewähltem  $r_1 = 2.075$ .

Intervalls  $[-1; 2.075]$  entstehen (siehe Abbildung 2.4). Nach den Herleitungen aus Abschnitt 1.12 existieren wahrscheinlich innerhalb dieses Intervalls die stabilen Polygone.

Zusätzlich zu der von MATLAB<sup>®</sup> integrierten Lupenfunktion kann der Anwender die Grenzen der Koordinatenachsen explizit festlegen. Durch Aktivierung der rechten Maustaste innerhalb des Koordinatensystems des *Singular-Frequencies-Generator*-Fensters öffnet sich ein selbsterklärendes Kontextmenü, welches diese Funktion bereitstellt.

Die Auswahl eines festen Wertes für  $r_1$  erfolgt entweder über das dafür vorgesehene Eingabefeld, oder interaktiv durch Auswahl eines Punktes innerhalb des Koordinatensystems mit der linken Maustaste.

**Bemerkung 2.1** RobSin speichert verschiedene Funktionen in ausführbaren Dateien mit der Endung „.m“ im Verzeichnis „project“ ab. Natürlich können Funktionen in MATLAB<sup>®</sup> auch anders definiert werden, z.B. als Inline-Funktion oder symbolisch mit der Extended Symbolic Math Toolbox. Es hat sich jedoch herausgestellt, dass diese Realisierungsmethode mit Abstand die schnellsten Ergebnisse erzielt.

### 2.3.3 Berechnung der singulären Frequenzen

Die singulären Frequenzen  $\alpha_S^i$  für den gewählten Wert von  $r_1$  berechnet *RobSin* durch Auswahl des Menüpunktes *Generate Singular Frequencies* in der Menüspalte *Design* und gibt diese im *Echo*-Rahmen aus. Für den Wert  $r_1 = 2.075$  resultiert die Ausgabe:

```
Generating the singular frequencies for r1=2.075 ... done (0.07 _
secs)
Plant 1: [0; 0.480408152838; 0.482738967057; 1.50405203356; 5.027_
32519164]
```

Die erste Zeile dieser Ausgabe zeigt Informationen über die Berechnung der singulären Frequenzen. Die zweite Zeile listet alle gefundenen singulären Frequenzen  $\alpha_S^i$  für die angezeigte Regelstrecke auf. Für das verwendete Beispiel existiert nur eine Regelstrecke. Bei Verwendung einer Q-Box, also von Parameterunsicherheiten, kann die Nummer einer Regelstrecke durch Auswahl einer  $r_1(\alpha)$ -Kurve mit der Maus festgestellt werden oder mit Hilfe der Legende im *Singular-Frequencies-Generator*-Fenster.

Aus den angezeigten singulären Frequenzen ist ersichtlich, dass die Möglichkeit bestünde, den  $r_1$ -Wert noch ein kleines Stück weiter zu erhöhen, bevor die zweite und die dritte Frequenz zusammenfallen.

Die singulären Frequenzen  $z_S^i$  in der komplexen Ebene können mit den Gleichungen (1.37) für Hurwitz- und  $\sigma$ -Stabilität bzw. (1.38) für Kreisstabilität berechnet werden.

### 2.3.4 Berechnung der singulären Geraden

Die Berechnung der singulären Geraden, wie sie Abschnitt 1.7 beschreibt, wird über den Menüpunkt *Generate Singular Lines* in der Menüspalte *Design* gestartet. Nach erfolgreicher Berechnung der singulären Geraden für den Wert  $r_1 = 2.075$  zeigt der *Echo*-Rahmen folgende Zeilen zusätzlich:

```
Generating the singular lines for r1=2.075 ...
  Generating singular frequencies ... done
  Generating singular lines ... done
  Creating function files and executing singular line crossing ...
..... done
Singular lines generated in 0.832 secs
'sl1.m' pertains to plant 1, alpha=0
'sl2.m' pertains to plant 1, alpha=0.480408152838
'sl3.m' pertains to plant 1, alpha=0.482738967057
'sl4.m' pertains to plant 1, alpha=1.50405203356
'sl5.m' pertains to plant 1, alpha=5.02732519164
```

Es wurden die singulären Frequenzen und Geraden erzeugt, letztere in ausführbaren Dateien gespeichert und die Überquerung von singulären Geraden, wie sie Unterabschnitt 1.8.1 beschreibt, durchgeführt. Die letzten Zeilen dieser Meldung teilen dem Anwender die Namen dieser Dateien mit, die zu den jeweils angezeigten singulären Frequenzen der nummerierten Regelstrecke gehören. Die Dateien befinden sich im *RobSin*-Pfad im Verzeichnis „project“.

### 2.3.5 Berechnung der inneren Polygone für einen festen Wert für $r_1$

Für einen festgelegten Wert von  $r_1$  ermöglicht *RobSin* die Berechnung der inneren Polygone, wie sie Abschnitt 1.8 herleitet. Diese Polygone liegen senkrecht zur  $r_1$ -Achse in der  $(r_0, r_2)$ -Ebene. Der Algorithmus dafür kann über zwei Wege gestartet werden, entweder über die Schaltfläche mit der Bezeichnung „*Single Slice*“ im *Singular-Frequencies-Generator*-Fenster, oder über den Menüpunkt *Calculate Single Slice* in der *Design*-Menüspalte.

Die Ausführung dieses Algorithmus für  $r_1 = 0.5$  führt zu folgender Ausgabe im *Echo*-Rahmen:

```
Generating slice for r1=0.5 ...
  Generating singular frequencies ... done
  Generating singular lines ... done
  Creating function files and executing singular line crossing ...
  ..... done
  Generating segments of singular lines ... done
  Generating all polygons ... done
  Detecting inner polygons ..... done
Slice generated in 2.273 secs
```

Die einzelnen Rechenschritte sind die Berechnung der singulären Frequenzen und Geraden, die Erzeugung der Geradengleichungen in ausführbaren Dateien, die Überquerung von singulären Geraden, wie sie Unterabschnitt 1.8.1 beschreibt, die Erzeugung der Geradenstücke und der sich daraus ergebenden Polygone und schließlich die Bestimmung der inneren Polygone nach Unterabschnitt 1.8.2.

Im Anschluss daran öffnet sich ein Fenster mit dem Titel „*RobSin V1.0 - Stabilizing 3D Region*“, welches dreidimensional die berechneten Polygone in den Koordinaten  $\mathbf{k}$  des Reglers anzeigt (siehe Abbildung 2.5). In dem behandelten Beispiel ergibt sich nur ein inneres Polygon, welches senkrecht zur  $k_P$ -Achse liegt.

Der Grund für die Lage des Polygons senkrecht zur  $k_P$ -Achse ist die Identität der beiden Koordinaten  $\mathbf{k} = [k_I, k_P, k_D]^T$  und  $\mathbf{r} = [r_0, r_1, r_2]^T$ .

Die Transformationsmatrix  $\mathbf{T}_{\text{PID},s}$  (Unterabschnitt 1.2.1) ist für den hier verwendeten PID-Regler mit Verzögerungszeitkonstante  $T_R = 0$  gleich der Einheitsmatrix  $\mathbf{E}$ . Die beiden Transformationsmatrizen  $\mathbf{T}_{a=0}$  und  $\mathbf{T}_{a \neq 0}$  aus Unterab-

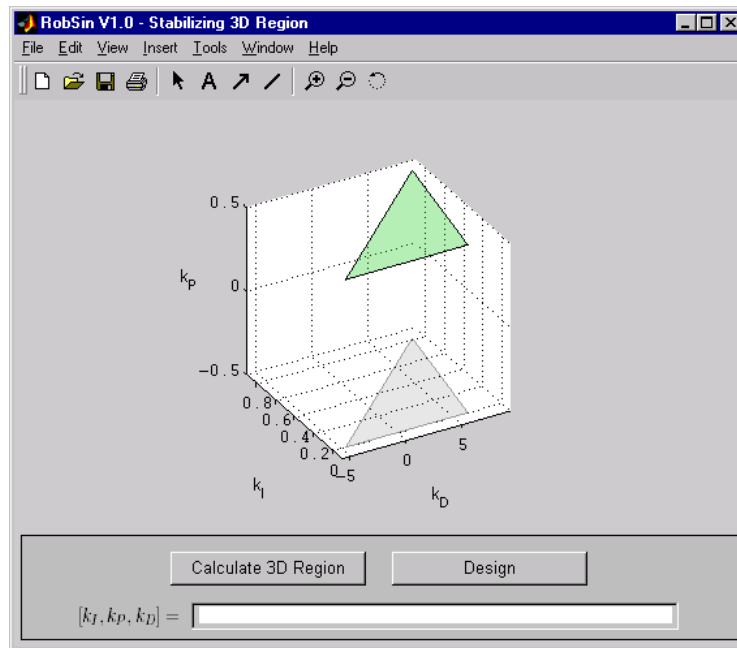


Abbildung 2.5: Das *Stabilizing-3D-Region*-Fenster mit dem inneren und stabilen Polygon für  $r_1 = 0.5$ .

schnitt 1.5.2 wurden bei *RobSin* willkürlich festgelegt zu

$$\mathbf{T}_{a=0} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -2\sigma_{\max} \\ 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$

bzw.

$$\mathbf{T}_{a \neq 0} = \begin{bmatrix} r^2 - m^2 & 1 & -m \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}. \quad (2.4)$$

So resultiert für den betrachteten Fall der Hurwitz-Stabilität ( $\sigma_{\max} = 0$ ) ebenfalls die Einheitsmatrix  $\mathbf{T}_{a=0} = \mathbf{E}$ . Die Transformation  $\mathbf{T}_{\text{ges}}$  zwischen den beiden Koordinaten  $\mathbf{k}$  und  $\mathbf{r}$  berechnet sich durch das Matrixprodukt

$$\mathbf{T}_{\text{ges}} = \mathbf{T}_{\text{PID},s} \cdot \mathbf{T}_{a=0}. \quad (2.5)$$

Die Umrechnung zwischen den beiden Koordinatensystemen lautet dann

$$\mathbf{k} = \mathbf{T}_{\text{ges}} \mathbf{r}. \quad (2.6)$$

Die Matrix  $\mathbf{T}_{\text{ges}}$  ergibt für diesen Fall die Einheitsmatrix, so dass die beiden Koordinaten  $\mathbf{k}$  und  $\mathbf{r}$  übereinstimmen.

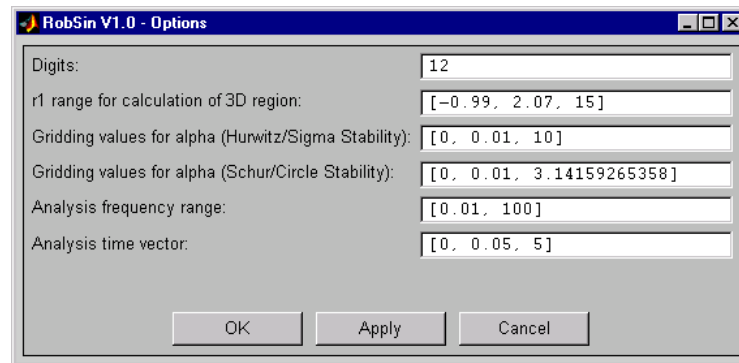


Abbildung 2.6: Das *Options*-Fenster zum Ändern der Einstellungen von *RobSin*.

Das im Fenster enthaltene Eingabefeld erlaubt für die Analyse die Festlegung eines Reglers in den Reglerparametern. Durch Aktivierung der Schaltfläche mit der Aufschrift „*Analysis*“ im *Stabilizing-3D-Region*-Fenster, oder durch Auswahl des Menüpunktes *Analysis: 2D Controller Parameter Selection and Analysis Plots* in der *Analysis*-Menüspalte erscheinen die beiden Rahmen für die Analyse.

### 2.3.6 Berechnung des dreidimensionalen Gebietes aller stabilisierenden Reglerparameter

Der verwendete PID-Regler (2.2) besitzt die drei Parameter ( $k_I, k_P, k_D$ ). Im Raum dieser drei Parameter existiert für die Regelstrecke (2.1) ein dreidimensionales Gebiet aller stabilisierenden Reglerparameter. *RobSin* unterstützt die Berechnung und Visualisierung dieses Gebietes.

Der Algorithmus für diese Berechnung wird über den Menüpunkt *Calculate 3D Controller Region* gestartet, oder alternativ über die Schaltfläche mit der Aufschrift „*Calculate 3D Region*“ im *Stabilizing-3D-Region*-Fenster.

Das dreidimensionale Gebiet setzt sich zusammen aus der Berechnung der inneren Polygone für verschiedene Werte von  $r_1$ . Diese Werte können im *Options*-Fenster festgelegt werden. Die Auswahl des Menüpunktes *Options...* öffnet dieses Fenster. Die Eingabe von

$[-0.99, 2.07, 15]$

in das Eingabefeld mit der Bezeichnung „*r1 range for calculation of 3D region:*“ innerhalb des *Options*-Fensters veranlasst *RobSin* dazu, bei der Berechnung der dreidimensionalen Gebietes, 15 Polygone im gleichen Abstand zueinander, innerhalb des Intervalls  $r_1 \in [-0.99; 2.07]$  zu erzeugen (siehe Abbildung 2.6). Die genaue Beschreibung der Einstellmöglichkeiten im *Options*-Fenster befindet sich in Abschnitt 2.4.

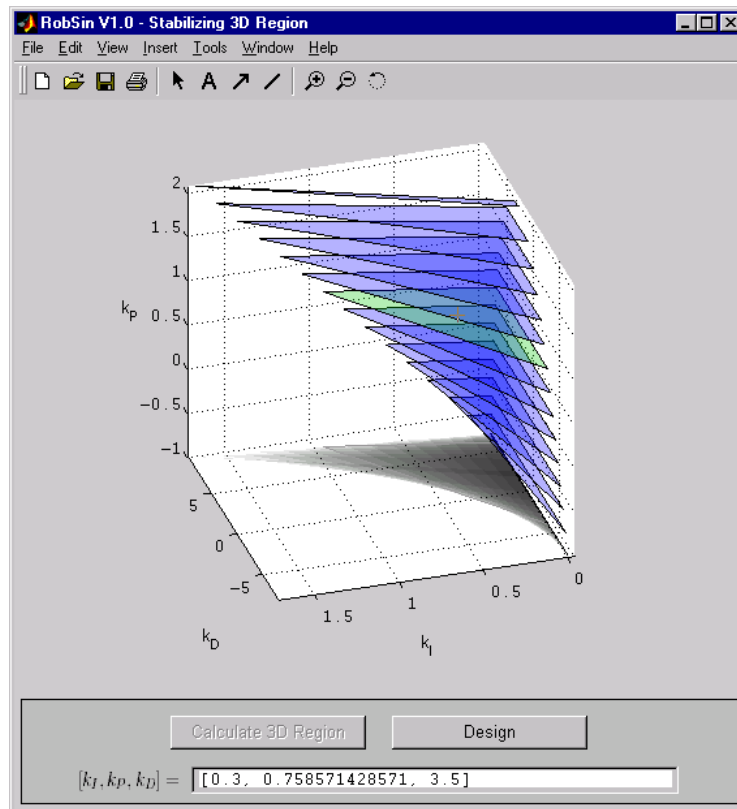


Abbildung 2.7: Das *Stabilizing 3D Region*-Fenster mit den inneren und stabilen Polygonen für  $r_1 \in [-0.99; 2.07]$  mit 15 Rasterwerten.

Die Berechnung des dreidimensionalen Gebietes verursacht folgende Ausgabe im *Echo*-Rahmen

Generating 15 slices for r1 in [-0.99, 2.07]:

- 1.: Generating slice for r1=-0.99 ..... done (2.033 secs)
- 2.: Generating slice for r1=-0.77143 ..... done (2.053 secs)

...

- 15.: Generating slice for r1=2.07 ..... done (2.103 secs)

All slices generated in 31.566 secs

Die einzelnen Zeilen der Ausgabe sind selbsterklärend. Im Anschluss an die Ausgabe öffnet sich das *Stabilizing 3D Region*-Fenster und zeigt die berechneten Polygone (siehe Abbildung 2.7). Sie liegen, aus dem im vorangehenden Unterabschnitt beschriebenen Grund, wieder senkrecht zur  $k_P$ -Achse.

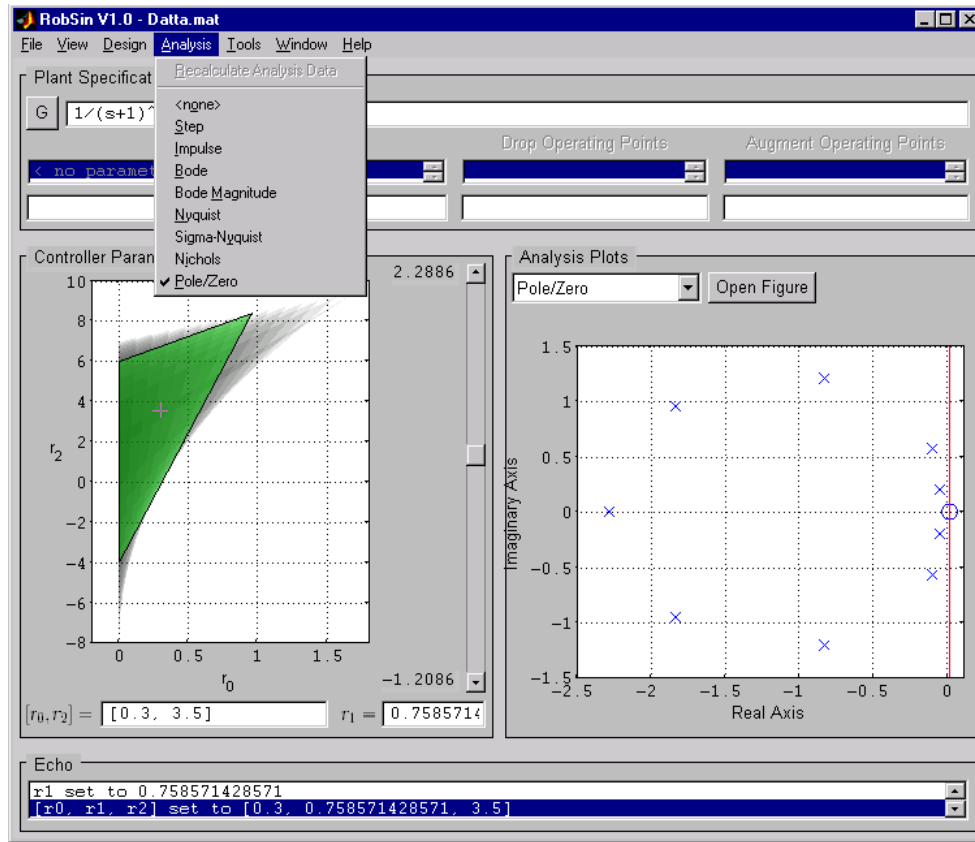


Abbildung 2.8: Die Oberfläche von *RobSin* für die Auswahl eines Reglers und die Analyse des sich dadurch ergebenden Regelkreises. Die Größe der einzelnen Rahmen wurde mit der Maus verändert.

### 2.3.7 Die Analyse eines ausgewählten Reglers

Dieser Unterabschnitt beschreibt anhand des betrachteten Beispiels die Analysemöglichkeiten eines Regelkreises in *RobSin*. Das Ziel ist es, einen im Auge des Anwenders „möglichst guten“ Regler zu finden.

Entscheidend ist dabei die einfache Auswahl der Reglerparameter  $\mathbf{r}$ , bzw.  $\mathbf{k}$ . *RobSin* ermöglicht, neben der Eingabe per Tastatur, die Festlegung des Reglers mit der Maus. Diese gliedert sich in zwei Teile, in die Auswahl des Rasterparameters  $r_1$  und in die Bestimmung der beiden Parameter  $r_0$  und  $r_2$ .

Der Rasterparameter  $r_1$  kann mit der Maus entweder durch Auswahl eines Polygons im *Stabilizing-3D-Region*-Fenster oder durch Verwendung des Schiebers im *Controller-Parameter-Selection*-Rahmen festgelegt werden.

Die Wahl der beiden anderen Parameter  $r_0$  und  $r_2$  erfolgt interaktiv durch Drücken der linken Maustaste innerhalb der zweidimensionalen Grafik im *Controller-Parameter-Selection*-Rahmen. Die Analyse funktioniert auch, wenn die

Maus bei gedrückter Maustaste bewegt wird. Der *Analysis-Plots*-Rahmen berechnet und aktualisiert dabei die Analysediagramme unmittelbar nach Veränderung der Parameter  $\mathbf{r}$  bzw.  $\mathbf{k}$ .

Die Eingabe der Reglerparameter  $\mathbf{r}$  bzw.  $\mathbf{k}$  in die dafür vorgesehenen Eingabefenster per Tastatur ermöglicht die exakte Festlegung dieser Parameter. Der Rasterparameter  $r_1$  muss nicht zwingend auf der Ebene eines Polygons liegen. Die Analyse funktioniert auch für Werte zwischen zwei Polygonen oder außerhalb des dreidimensionalen Gebietes.

Die Änderung des Parameters  $r_1$  verursacht eine Meldung im *Echo*-Rahmen, wie z.B.

```
r1 set to 0.758571428571
```

Hingegen erzeugt die Festlegung der anderen Parameter beispielsweise die Ausgabe

```
[r0, r1, r2] set to [0.3, 0.758571428571, 3.5]
```

Für die Analyse muss der Anwender nun ein Analyseverfahren auswählen, wie in Unterabschnitt 2.2.2 beschrieben. Beispielsweise zeigt Abbildung 2.8 das Pol-Nullstellen-Diagramm für den Regler  $\mathbf{r} = [0.3, 0.758571428571, 3.5]$ . Der Regelkreis ist für diesen Regler innerhalb des dreidimensionalen Gebietes stabil.

### 2.3.8 Speichern und Laden von Sitzungen

Um auf die eingegebenen und berechneten Daten zu einem späteren Zeitpunkt zugreifen zu können, ermöglicht *RobSin* das Speichern und Laden von Sitzungen. Durch Auswahl der Menüpunkte *Save...*, *Load...* bzw. *Save As...* in der *File*-Menüspalte werden diese Aktionen ausgeführt. Das Standardverzeichnis für das Laden und Speichern von Sitzungen ist das Verzeichnis „examples“ im *RobSin*-Pfad.

Die Dateien von *RobSin* haben die Endung „.mat“ und können auch außerhalb dieses Software-Werkzeuges in MATLAB<sup>®</sup> geladen werden. Dieser Vorgang erzeugt im *Workspace* (dt.: Arbeitsspeicher) von MATLAB<sup>®</sup> die Variable „RobSinData“, die alle Daten einer *RobSin*-Sitzung enthält.

Das behandelte Beispiel, für den Fall der Hurwitz-Stabilität, wurde in der Datei „Datta.mat“ abgelegt.

### 2.3.9 Berechnungen für $\sigma$ -Stabilität

Wie bereits erwähnt, lässt *RobSin* neben der Behandlung der Hurwitz-Stabilität auch andere Eigenwertspezifikationen zu. Bei der  $\sigma$ -Stabilität existiert für ein willkürlich festgelegtes  $\sigma_{\max} = -0.1$  bei dem untersuchten Regelkreis noch ein dreidimensionales Gebiet, in dem der Regelkreis die Eigenwertspezifikationen erfüllt.

Die Einstellung auf die  $\sigma$ -Stabilität mit  $\sigma_{\max} = -0.1$  erfolgt im *Gamma-Region*-Rahmen. Die Transformationsmatrix  $\mathbf{T}_{a=0}$  wird zu

$$\mathbf{T}_{a=0} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0.2 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.7)$$

Da  $\mathbf{T}_{\text{PID},s}$  wieder gleich der Einheitsmatrix  $\mathbf{E}$  ist, gilt nach (2.5) für die Transformationsmatrix  $\mathbf{T}_{\text{ges}}$

$$\mathbf{T}_{\text{ges}} = \mathbf{T}_{a=0}. \quad (2.8)$$

Die Beziehung zwischen den Koordinaten  $\mathbf{k}$  und  $\mathbf{r}$  beschreibt Gleichung (2.6). Die sich ergebenden, stabilen Polygone liegen also nicht mehr in der  $(k_I, k_D)$ -Ebene (siehe Abbildung 2.9).

Dieses Beispiel befindet sich in der Datei „Datta.sigma.mat“.

### 2.3.10 Behandlung einer unsicheren Totzeit

Eine Stärke von *RobSin* besteht in der Behandlung von totzeitbehafteten Systemen. Um dies zu demonstrieren, soll zusätzlich zu der untersuchten, die folgende Regelstrecke mit der Totzeit  $L = 1.5$  behandelt werden,

$$G_L(s) = \frac{1}{(s+1)^8} \cdot e^{-1.5s}. \quad (2.9)$$

Für die zusätzliche Eingabe von  $G_L(s)$  modifiziere man die Formel für die Regelstrecke im *Plant-Specification*-Rahmen beispielsweise zu

```
1/(s+1)^8*exp(-L*s)
```

Alle verwendeten Parameter werden automatisch erkannt und angezeigt. Die Definition der Grenzen für die unsicheren Parameter findet im selben Rahmen statt. Dazu wähle man einen Parameter mit der Maus aus und schreibe in das darunter liegende Eingabefeld seine untere und obere Schranke. Im Beispiel müssen die Grenzen für die Totzeit  $L$  festgelegt werden, durch

```
[0, 1.5]
```

Als Bestätigungsmeldung für die beiden Eingaben erscheint im *Echo*-Rahmen

```
New plant defined: '1/(s+1)^8*exp(-L*s)' with dead time: 'L'
Bounds of parameter 'L' set to [0, 1.5]
```

Als Regler verende man unverändert den PID-Regler. Für das  $\Gamma$ -Gebiet betrachte man den Fall der  $\sigma$ -Stabilität aus dem vorangegangenen Unterabschnitt mit  $\sigma_{\max} = -0.1$ .

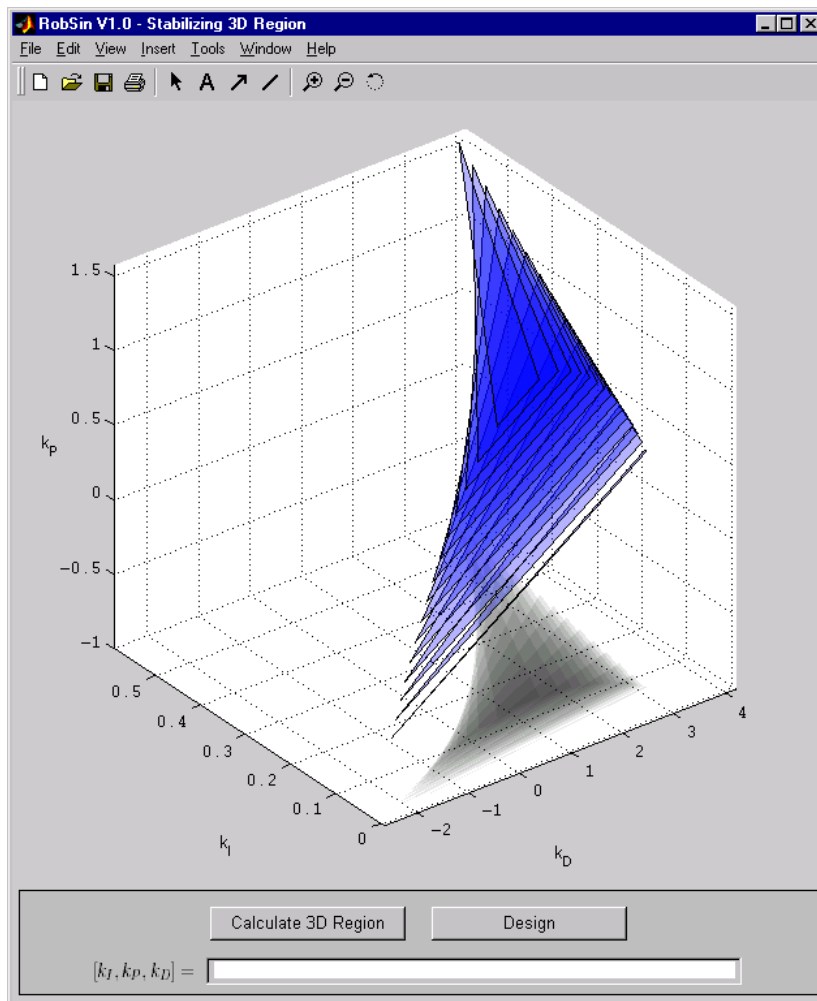


Abbildung 2.9: Das *Stabilizing 3D Region*-Fenster mit den inneren und stabilen Polygonen für  $\sigma_{\max} = -0.1$  mit  $r_1 \in [-0.04; 0.8]$  mit 15 Rasterwerten.

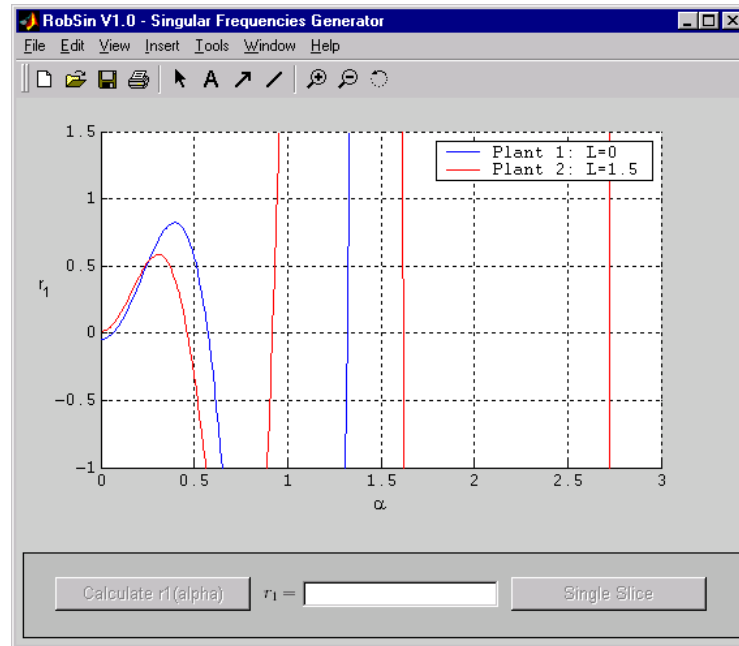


Abbildung 2.10: Das *Singular-Frequencies-Generator*-Fenster mit der Kurve  $r_1(\alpha)$  für die beiden Regelstrecken (2.1) und (2.9) für  $\sigma_{\max} = -0.1$ .

Aus dem sich daraus ergebenden Generator der singulären Frequenzen, also der Funktion  $r_1(\alpha)$ , ist ersichtlich, dass ausschließlich für Werte von  $r_1$  innerhalb des Intervalls  $[0.04; 0.583]$  Polygone existieren, in denen der geschlossene Regelkreis robust stabil ist (siehe Abbildung 2.10).

Abbildung 2.11 zeigt das von *RobSin* berechnete, dreidimensionale Gebiet aller robust stabilisierenden Reglerparameter für diesen Fall. Die Transformationsmatrix  $\mathbf{T}_{\text{ges}}$  ist die selbe wie im letzten Unterabschnitt, da auch das  $\Gamma$ -Gebiet und die Reglerstruktur gleich sind.

Ein fester Regler aus dem stabilen Gebiet kann weder mit Hilfe des Pol-Nullstellen-Diagramms, noch unter Verwendung des Nyquist-Diagramms nach  $\sigma$ -Stabilität untersucht werden, da zum einen für Totzeitsysteme die explizite Berechnung aller Pole unmöglich ist und zum anderen das Nyquist-Kriterium nur eine Aussage über Hurwitz-Stabilität zulässt.

Aus diesem Grund stellt *RobSin* das Analyseverfahren „Sigma-Nyquist“ zur Verfügung. Es ersetzt in dem zu analysierenden offenen Regelkreis die komplexe Variable  $s$  durch  $s + \sigma_{\max}$  und zeichnet anschließend das Nyquist-Diagramm dieses modifizierten Regelkreises. Auf diesen modifizierten Regelkreis lässt sich das *angepasste Nyquist-Kriterium* anwenden:

Ein Regelkreis ist stabil, wenn der Zeiger vom kritischen Punkt  $(-1, j0)$  zum Ortskurvenpunkt des offenen Regelkreises  $C(s) \cdot G(s)$

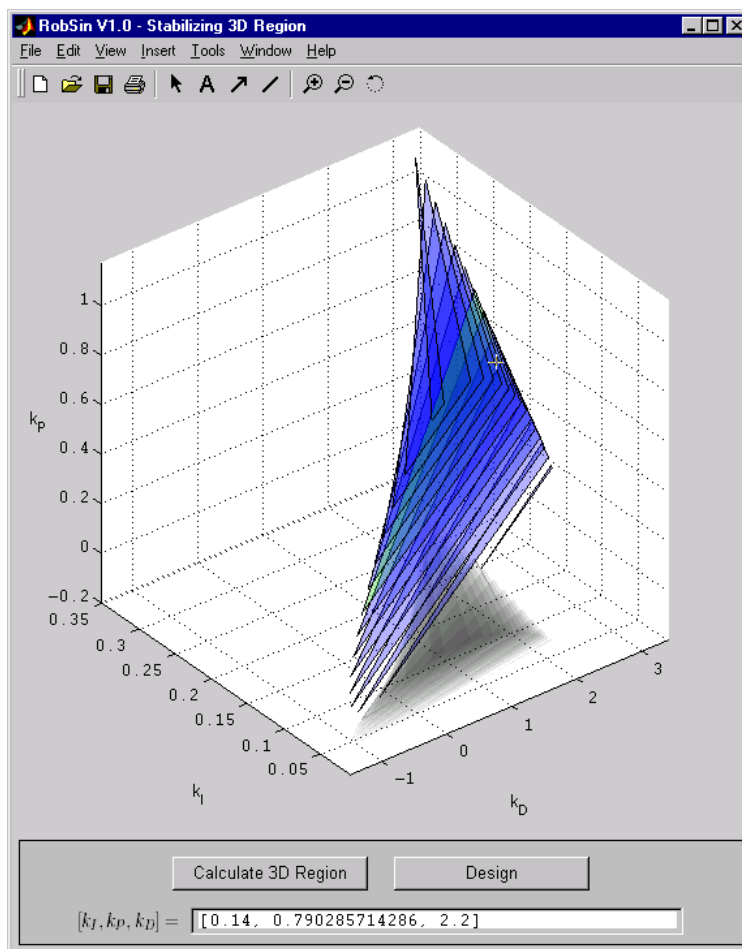


Abbildung 2.11: Das *Stabilizing 3D Region*-Fenster mit den inneren und robust-stabilen Polygonen für die beiden Regelstrecken (2.1) und (2.9) für  $\sigma_{\max} = -0.1$  mit  $r_1 \in [0.04; 0.583]$  mit 15 Rasterwerten.

beim Durchlaufen der Kreisfrequenz von  $0 \leq \omega < \infty$  die stetige Winkeländerung

$$\Delta\Phi = (n_p + n_0/2) \cdot \pi \quad (2.10)$$

erfährt. Dabei sind  $n_p$  die Pole von  $C(s) \cdot G(s)$  außerhalb des  $\Gamma$ -Gebietes, und  $n_0$  diejenigen auf der Berandung des  $\Gamma$ -Gebietes  $\partial\Gamma$ .

Ist das Kriterium erfüllt, so ist der geschlossene Regelkreis  $\sigma$ -stabil (siehe Abbildung 2.12).

## 2.4 Das *Options*-Fenster

Die im letzten Abschnitt beschriebenen Berechnungen mit *RobSin* hängen von verschiedenen Parametern ab. Diese Parameter lassen sich im *Options*-Fenster einstellen. Der Menüpunkt *Options...* in der Menüspalte *File* öffnet dieses Fenster (siehe Abbildung 2.6). Es enthält Schaltflächen und Eingabefelder.

### 2.4.1 Die Schaltflächen

Die drei Schaltflächen im *Options*-Fenster erfüllen die folgenden Funktionen:

**OK** übernimmt die sichtbaren Einstellungen und schließt das *Options*-Fenster,

**Apply** übernimmt die sichtbaren Einstellungen, ohne das *Options*-Fenster zu schließen,

**Cancel** schließt das *Options*-Fenster, ohne die Einstellungen zu aktualisieren.

### 2.4.2 Die Eingabefelder

Die Eingabefelder im *Options*-Fenster erlauben die Einstellung von Parametern. Die Veränderungen der Parameter haben folgende Bedeutungen und Auswirkungen:

**Digits** ist ganzzahlig und legt die Rechengenauigkeit für *RobSin* fest.

**r1 range for calculation of 3D region** ist entweder ein Vektor mit drei Einträgen, oder eine ganze Zahl.

Als Vektor definiert dieses Eingabefeld den Bereich für  $r_1$ , in dem die Berechnung des dreidimensionalen Gebietes aus Unterabschnitt 2.3.6 durchgeführt werden, sowie die Anzahl der Rasterpunkte für  $r_1$ . Der Vektor ist von der Form  $[r_{1,\min}, r_{1,\max}, n_{r_1}]$ , mit  $r_{1,\min}$  gleich der unteren Grenze von  $r_1$ ,  $r_{1,\max}$  gleich der oberen Grenze von  $r_1$  und  $n_{r_1}$  gleich der Anzahl der Rasterpunkte.

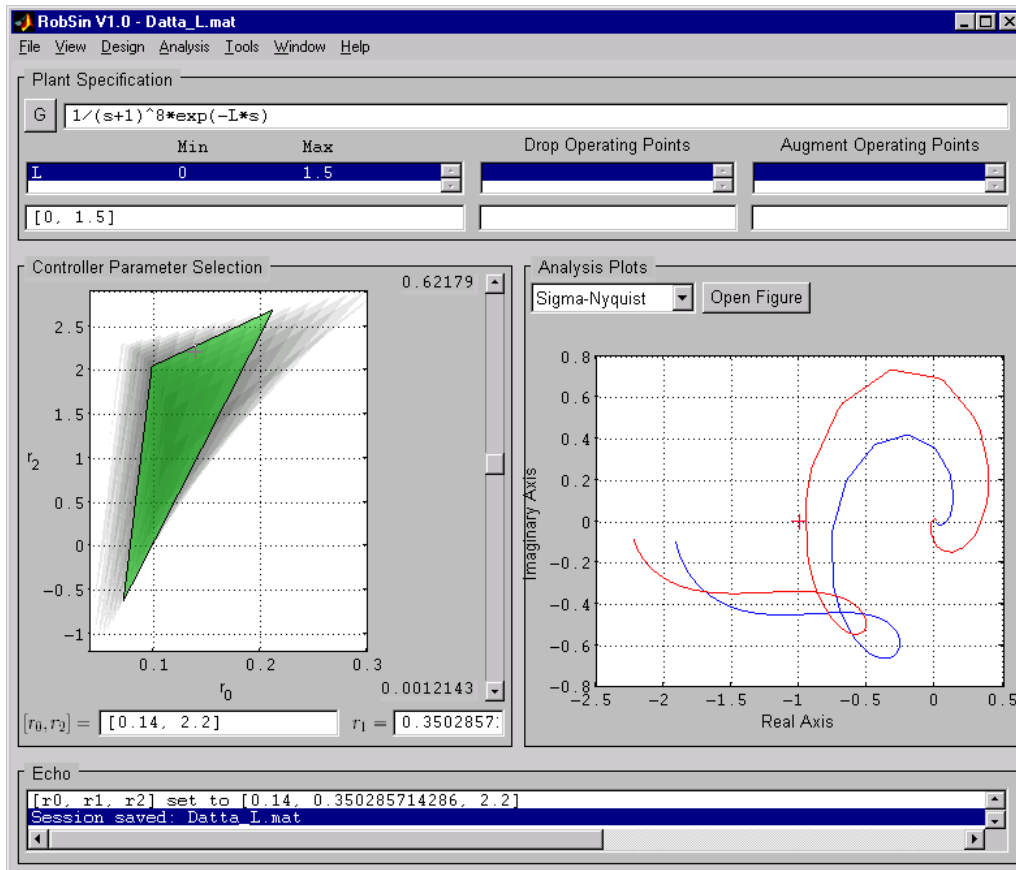


Abbildung 2.12: Die Oberfläche von *RobSin* für die gleichzeitige Analyse des Regelkreises mit den beiden Regelstrecken (2.1) und (2.9) für  $\sigma_{\max} = -0.1$ . Der geschlossene Regelkreis mit dem ausgewählten PID-Regler ist robust  $\sigma$ -stabil. (Ein Pol des *offenen* Regelkreises, und zwar des PID-Reglers, liegt im Ursprung und somit außerhalb des  $\Gamma$ -Gebietes, so dass um den Punkt  $s = -1$  eine Winkeländerung von  $+\pi/2$  vollzogen wird.)

**Gridding values for alpha (Hurwitz/Sigma Stability)** legt für die Hurwitz- und die  $\sigma$ -Stabilität die Rasterpunkte für  $\alpha$  fest, zwischen denen nach jeweils einer kritischen Frequenz  $\alpha_K^i$  gesucht wird (siehe Unterabschnitt 1.6.2). Ein Vektor der Form  $[\alpha_{\min}, \Delta\alpha, \alpha_{\max}]$ , mit  $\alpha_{\min}$  gleich der unteren Grenze von  $\alpha$ ,  $\alpha_{\max}$  gleich der oberen Grenze von  $\alpha$  und  $\Delta\alpha$  gleich der Schrittweite, definiert die Rasterpunkte. Es sind ausschließlich positive Frequenzen erlaubt.

**Gridding values for alpha (Schur/Circle Stability)** legt für die Schur- und die Kreisstabilität die Rasterpunkte für  $\alpha$  fest, zwischen denen nach jeweils einer kritischen Frequenz  $\alpha_K^i$  gesucht wird. Ein Vektor, analog zum vorherigen Eingabefeld, definiert die Rasterpunkte. Die Frequenz  $\alpha$  ist hier jedoch nach oben hin beschränkt:  $\alpha \leq \pi$ .

**Analysis frequency range** ist ein Vektor mit zwei Parametern, der den Frequenzbereich für  $\omega$  für das Bode-, das Nyquist- und das Nichols-Diagramm im *Analysis-Plots*-Rahmen setzt. Er ist von der Form  $[\omega_{\min}, \omega_{\max}]$ , mit der unteren und oberen Grenze  $\omega_{\min} > 0$  bzw.  $\omega_{\max} > \omega_{\min}$ .

**Analysis time vector** ist ein Vektor mit drei Parametern, der den Zeitbereich für die Sprung- und die Impulsantwort im *Analysis-Plots*-Rahmen festlegt. Er ist von der Form  $[t_{\min}, \Delta t, t_{\max}]$ , mit der unteren und oberen Grenze des Zeitbereichs  $t_{\min}$  bzw.  $t_{\max}$  und der Schrittweite  $\Delta t$ .

## 2.5 Bedienung von *RobSin* über die Kommandozeile

Die in Abschnitt 2.3 dargestellte Funktionalität von *RobSin* ist zum großen Teil auch über die Kommandozeile abrufbar. Dieser Abschnitt beschreibt, wie der Anwender ohne die grafische Oberfläche von *RobSin* die Algorithmen zur Berechnung des Gebietes aller stabilisierenden Reglerparameter startet.

Die Klasse *singf* beinhaltet diese Algorithmen, welche über Funktionsaufrufe von öffentlichen Methoden dieser Klasse ausgeführt werden. Nach der Beschreibung des Konstruktors *singf* sind diese Methoden in alphabetischer Reihenfolge aufgeführt. Am Ende jeder Beschreibung steht der Funktionsaufruf der jeweiligen Funktion für die gleichzeitige Behandlung der beiden Regelstrecken (2.1) und (2.9).

### 2.5.1 Der Konstruktor *singf*

Jede Klasse benötigt einen Konstruktor, der eine Instanz seiner Klasse initialisiert. Bei der Klasse *singf* lautet der Aufruf des gleichnamigen Konstruktors beispielsweise wie folgt:

```
sq = singf(argin,dispObj)
```

Der Konstruktor besitzt die beiden Eingangsparameter `argin` und `dispObj` und den Ausgangsparameter `sq`, dem die neue Instanz der Klasse `singf` zugewiesen wird.

Die Berechnung des dreidimensionalen Gebietes erfordert Informationen über das  $\Gamma$ -Gebiet und den Regelkreis des zu untersuchenden Systems. Der Eingangsparameter `argin` enthält diese Informationen. *RobSin* bietet die Möglichkeit diesen Parameter in den *Workspace* (dt.: Arbeitsspeicher) von MATLAB<sup>®</sup> zu exportieren (siehe Tabelle 2.1).

Für das aus Unterabschnitt 2.3.10 verwendete System erzeugt *RobSin* folgende Werte für den Parameter `argin`:

```
argin.gamma.region = 'sigma';
argin.gamma.params = '[-0.1]';
argin.gamma.frequency.param = 'alpha';
argin.gamma.frequency.range = '[0,0.01,10]';

argin.sys.complexvariable = 's';
argin.sys.SampleTime = '-1';
argin.sys.input = 'a,b';

argin.sys.plant(1).Ld = '0';
argin.sys.plant(1).A = '1';
argin.sys.plant(1).B = 's^9+8*s^8+28*s^7+56*s^6+70*s^5+56*s^4+28_
*s^3+8*s^2+s';

argin.sys.plant(2).Ld = '.1';
argin.sys.plant(2).A = '1';
argin.sys.plant(2).B = 's^9+8*s^8+28*s^7+56*s^6+70*s^5+56*s^4+28_
*s^3+8*s^2+s';
```

Es ist erkennbar, dass `argin` als Struktur mit den beiden Feldern `gamma` und `sys` definiert wurde.

Das Feld `gamma` beschreibt das ausgewählte  $\Gamma$ -Gebiet (das  $\Gamma$ -Gebiet für  $\sigma$ -Stabilität mit  $\sigma_{\max} = -0.1$ ) und beinhaltet ein weiteres Feld `frequency`, in dem das Raster für den Parameter  $\alpha$  der Funktion  $r_1(\alpha)$  enthalten ist. Dieses Raster definiert die Teilintervalle, in denen nach jeweils einer kritischen Frequenz  $\alpha_K^i$  gesucht wird (siehe Unterabschnitt 1.6.2). In dem behandelten Fall haben die Teilintervalle  $\Delta I^i$  die Breite  $\Delta\alpha = 0.01$  und liegen selber im Intervall  $I_{\text{ges}} = [0; 10]$ .

Weitere Möglichkeiten für das Feld `gamma.region` sind `hurwitz` für Hurwitz-Stabilität und `circle` für Kreisstabilität. Das Feld `gamma.params` muss dann zu `'[0]'` bzw. `'[<Kreismittelpunkt>,<Radius>]'` gesetzt werden.

Das zweite Feld von `argin`, `sys`, beinhaltet die Daten für die Regelstrecken. Alle gleichzeitig untersuchten Regelstrecken müssen die gleiche komplexe Variable, die gleiche Abtastzeit und die gleiche Eingangsstruktur besitzen. Das untersuchte System ist zeitkontinuierlich mit der komplexen Variablen  $s$ . Für zeitkontinuierliche Systeme gibt es keine Abtastzeit, so dass diese zu  $-1$  gesetzt werden muss. Für zeitdiskrete Systeme mit der komplexen Variablen  $z$  muss die Abtastzeit dagegen angegeben werden.

Als Eingangsstruktur erlaubt die Klasse `singf` entweder die explizite Angabe von Zähler `A` und Nenner `B` der Regelstrecke (`'a,b'`), oder die Angabe als Bruch (`'g'`). In beiden Fällen muss eine Totzeit `Ld` immer explizit angegeben werden. Wie die Auflistung von `argin` zu erkennen gibt, verwendet `RobSin` die explizite Angabe von Zähler und Nenner. Bei der Angabe als Bruch muss die Regelstrecke im Feld `argin.sys.plant(i).G` definiert sein.

Der zweite Eingangsparameter `dispObj` ist optional. Wird er weggelassen, so erzeugt das Objekt `sq` seine Ausgaben im *Command Window* von MATLAB®. Andernfalls versucht es, eine Methode des Objektes `dispObj` mit dem Namen `message` mit der Ausgabenachricht als Eingangsparameter aufzurufen. Diese zweite Möglichkeit verwendet `RobSin` für die Anzeige des Rechenfortschritts im *Echo*-Rahmen. Für die Berechnungen im *Command Window* von MATLAB® sollte daher der Eingangsparameter `dispObj` weggelassen werden.

### Beispiel:

```
>> sq = singf(argin);
Please wait. Initializing singf object ...
  Initializing fields ... done
  Computing symbolical expressions ... done
  Creating functions files ... done
  Generating critical frequencies ... done
The singf object constructed in 5.297 secs
```

## 2.5.2 Die Methode *genall*

Die Methode *genall* berechnet alle inneren Polygone für den vorgegebenen Bereich von  $r_1$ . Der Funktionsaufruf lautet:

```
[sq,polygons] = genall(sq,gpar_beg,gpar_end,gpar_n)5
```

Dabei ist `sq` eine Instanz der Klasse `singf`, `polygons` die berechneten, inneren Polygone, `gpar_beg` die untere, `gpar_end` die obere Grenze des Bereiches von  $r_1$  und `gpar_n` die Anzahl der Rasterpunkte. Der Ausgangsparameter `polygons`

---

<sup>5</sup>Zusätzlich zu den Methodenaufrufen aus MATLAB® wurde für die öffentlichen Methoden der Klasse `singf` die gleichbedeutende, in anderen objektorientierten Programmiersprachen weit verbreitete Syntax ermöglicht: `sq.genall(gpar_beg,gpar_end,gpar_n)`

ist optional. Wird er weggelassen, so zeigt die Methode *genall* die berechneten, inneren Polygone in einem extra Fenster an.

**Beispiel:**

```
>> sq.genall(0.04,0.583,15);
Generating 15 slices for r1 in [0.04, 0.583]:
  1.: Generating slice for r1=0.04 ..... done (8.562 secs)
  2.: Generating slice for r1=0.078786 ..... done (7.02 secs)
  ...
 15.: Generating slice for r1=0.583 ..... done (6.88 secs)
All slices generated in 103.759 secs
```

### 2.5.3 Die Methode *gensfs*

Die Methode *gensfs* berechnet die singulären Frequenzen  $\alpha_s^i$  für den vorgegebenen Wert von  $r_1$  (*gpar*) und speichert sie in der Objektvariablen *SingularFrequencies*:

```
[sq] = gensfs(sq,gpar)
```

**Beispiel:**

```
>> sq.gensfs(0.3);
Generating the singular frequencies for r1=0.3 ..... done (0.281
s_
ecs)
```

### 2.5.4 Die Methode *genslice*

Die Methode *genslice* erzeugt für einen festen Wert von  $r_1$  (*gpar*) die inneren Polygone und zeigt sie in einem Fenster:

```
[sq,polygon] = genslice(sq,gpar)
```

Der Ausgangsparameter *polygon* ist auch hier wieder optional. Fehlt er, so erscheinen die Polygone in einem extra Fenster.

**Beispiel:**

```
>> sq.genslice(0.3);
Generating slice for r1=0.3 ...
  Generating singular frequencies ..... done
  Generating singular lines ... done
  Creating function files and executing singular line crossing ...
  ..... done
```

```

Generating segments of singular lines ... done
Generating all polygons ... done
Detecting inner polygons ..... done
Calculate r1(alpha) .... done
Slice generated in 8.953 secs

```

### 2.5.5 Die Methode *gensls*

Die Methode *gensls* berechnet für einen übergebenen Wert von  $r_1$  (*gpar*) die singulären Linien und speichert sie im *RobSin*-Pfad im Verzeichnis „project“ unter den Dateinamen „slx.m“, mit der Nummerierung  $x$ :

```
[sq] = gensls(sq,gpar)
```

**Beispiel:**

```

>> sq.gensls(0.3);
Generating the singular lines for r1=0.3 ...
  Generating singular frequencies ..... done
  Generating singular lines ... done
  Creating function files and executing singular line crossing ...
  ..... done
Singular lines generated in 2.814 secs

```

### 2.5.6 Die Methode *getandanalyze*

Die Methode *getandanalyze* kann aufgerufen nachdem mit der Methode *genslice* innerer Polygone erzeugt wurden. Sie ermöglicht die Auswahl eines Reglers in der Abbildung der inneren Polygone mit der Maus und erzeugt daraufhin das Nyquist-Diagramm und die Wurzelortskurve in eigenen Fenstern:

```
getandanalyze(sq)
```

**Beispiel:**

```
>> sq.getandanalyze;
```

### 2.5.7 Die Methode *plotsfgens*

Die Methode *plotsfgens* berechnet die Kurve  $r_1(\alpha)$  und zeigt sie in einem Fenster:

```
plotsfgens(sq)
```

**Beispiel:**

```

>> sq.plotsfgens;
Calculate r1(alpha) .... done

```

### 2.5.8 Die Methode *plotstabpolgs*

Die Methode *plotstabpolgs* zeigt die zuletzt, für einen Wert von  $r_1$ , berechneten Polygone in einem Fenster:

```
plotstabpolgs(sq)
```

**Beispiel:**

```
>> sq.plotstabpolgs;
```

# Kapitel 3

## Anwendungsbeispiel

Die Algorithmen zur Berechnung des stabilen Gebietes im Raum der Reglerparameter basieren im Software-Werkzeug *RobSin* auf der Methode der singulären Frequenzen. Im Gegensatz zum zweiten Kapitel, welches die Bedienung von *RobSin* erklärt, beschreibt dieses Kapitel die regelungstechnischen Aspekte des Verfahrens anhand eines Beispiels aus der Fachliteratur. So behandelt es unter anderem die Feineinstellung von Reglerparametern für die robuste Stabilisierung und die gleichzeitige Einstellung von Dämpfung und Einschwingzeit.

Das Beispiel stammt aus dem Konferenzbeitrag [WB90], der ein einfaches mechanisches System als Referenzproblem für den robusten Reglerentwurf vorgeschlägt. Die Anordnung dieses Systems zeigt Abbildung 3.1. Das System wird zunächst im Laplace- und anschließend im zeitdiskreten Bereich untersucht.

### 3.1 Laplace-Bereich

Das System besteht aus zwei Massen,  $m_1$  und  $m_2$ , die über eine Feder  $k$  verbunden sind. Die Übertragungsfunktion von der Kraft  $u$ , welche an der ersten Masse  $m_1$  angreift, zur Position  $y$  der zweiten Masse  $m_2$  lässt sich im Laplace-Bereich leicht

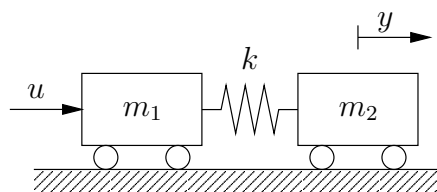


Abbildung 3.1: Schematische Darstellung des Referenzproblems.

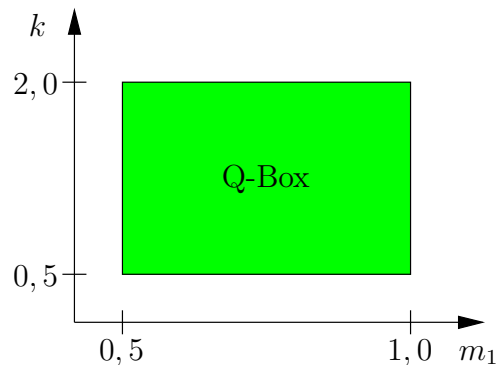


Abbildung 3.2: Parameterunsicherheiten als Q-Box.

bestimmen zu

$$G(s, k, m_1, m_2) = \frac{y(s)}{u(s)} = \frac{k / m_1 m_2}{s^2 \left( s^2 + k \frac{m_1 + m_2}{m_1 m_2} \right)}. \quad (3.1)$$

Die beiden Parameter  $k$  und  $m_1$  seien unsicher innerhalb ihrer Schranken,<sup>1</sup>

$$\begin{aligned} 0,5 < k < 2 \\ 0,5 < m_1 < 1. \end{aligned} \quad (3.2)$$

Die Masse  $m_2$  sei konstant,

$$m_2 = 1. \quad (3.3)$$

Aus diesen Parameterunsicherheiten resultiert eine sogenannte Q-Box in der  $(m_1, k)$ -Ebene (Abbildung 3.2). Das Ziel ist die gleichzeitige robuste Stabilisierung dieses Systems für alle Betriebspunkte mit Hilfe des Software-Werkzeuges *RobSin*. Den Regelkreis zeigt Abbildung 3.3. In den meisten Fällen ist ein Regelkreis genau dann robust stabil, wenn er für alle Eckpunkte des Hyperrechtecks  $Q$  (auch *Q-Box* genannt) Stabilität aufweist (vgl. [ABB<sup>+</sup>02]). Es reicht dann aus, nur die für diese Extremwerte die Stabilitätsüberprüfung durchzuführen.

### 3.1.1 Reglerstruktur

Der Versuch, die Regelstrecke mit Hilfe eines PID-Reglers im Hurwitzschen Sinne zu stabilisieren, zeigt, dass die robuste Stabilisierung nicht möglich ist. Der Grund dafür ist die Lage der vier Pole der Regelstrecke auf der imaginären Achse (siehe Abbildung 3.4).

<sup>1</sup>In diesem Kapitel entsprechen die Einheiten physikalischer Größen den SI-Einheiten.

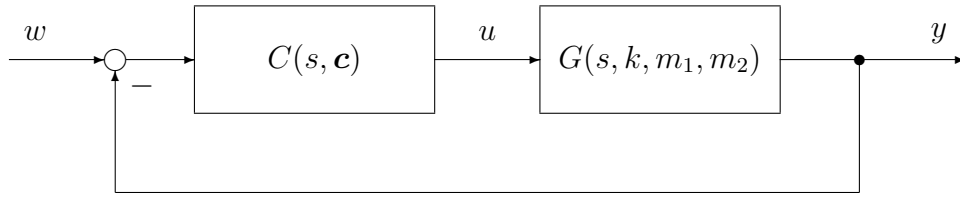


Abbildung 3.3: Regler  $C(s, \mathbf{c})$  und Regelstrecke  $G(s, k, m_1, m_2)$  im Standardregelkreis.

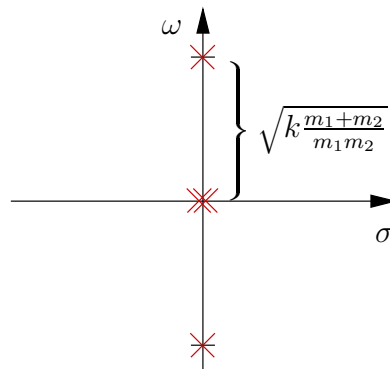


Abbildung 3.4: Die vier Pole der Regelstrecke  $G(s, k, m_1, m_2)$ .

Der Regler benötigt vier Nullstellen, um diese Pole in die linke Hälfte der  $s$ -Ebene zu verschieben. Ein geeigneter Regler ist beispielsweise

$$C(s, \mathbf{c}) = \frac{1000 (6 + 4.5s + s^2) (c_0 + c_1s + c_2s^2)}{(s + 10)(s + 14.05)(s + 12.16)(s + 5.07)}. \quad (3.4)$$

Das System ist mit diesem Regler robust stabilisierbar.

### 3.1.2 Bestimmung des stabilen Gebietes im Raum der Reglerparameter

Die Vorgabe einer garantierten Einschwingzeit  $T_{\text{ein}} = 30$  durch ein  $\sigma_{\text{max}} = -0.1$  ergibt die in Abbildung 3.5 gezeigte Funktion  $r_1(\alpha)$ . Aus ihr geht hervor, dass die stabilen Polygone aller Voraussicht nach im Intervall  $r_1 \in [-0.397; 0.455]$  zu finden sind, da hier die meisten Schnitte mit einer Parallelen zur  $\alpha$ -Achse entstehen (siehe Abschnitt 1.12).

Die Berechnung der inneren Polygone für Werte von  $r_1$  innerhalb dieses Intervalls führt jedoch zu dem Ergebnis, dass die obere Grenze des Intervalls nicht das stabile, dreidimensionale Gebiet beschränkt. Die Suche nach stabilen Polygonen wird daher um das Intervall  $[0.455; 2.710]$  ausgeweitet, da hier die zweitgrößte

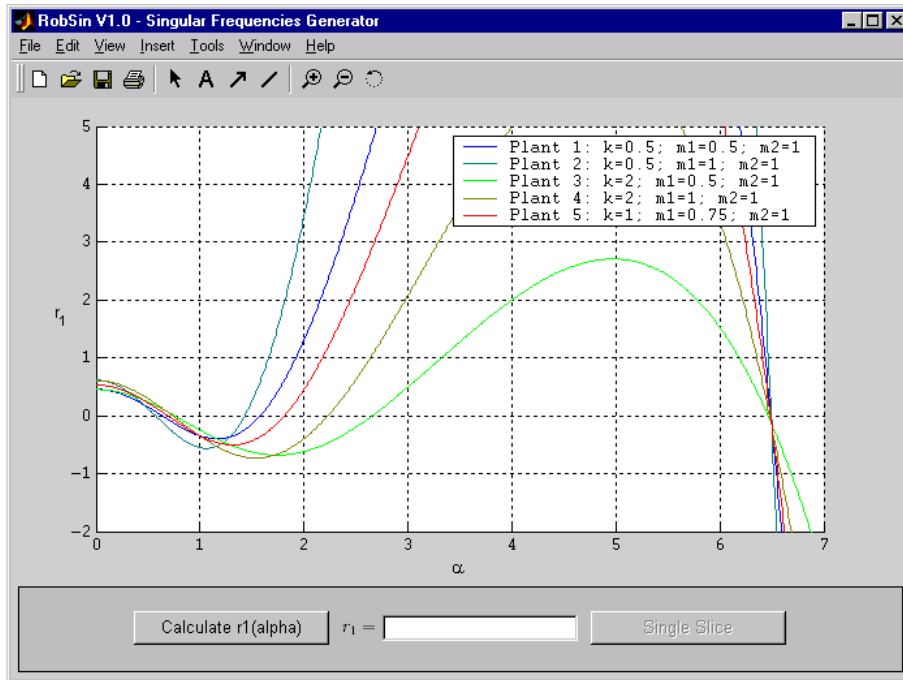


Abbildung 3.5: Das robust  $\Gamma$ -stabile Gebiet, mit  $\sigma_{\max} = -0.1$ , für den geschlossenen Regelkreis aus  $G(s, k, m_1, m_2)$  und  $C(s, \mathbf{c})$ .

Anzahl an Schnitten von  $r_1(\alpha)$  mit einer Parallelen zur  $\alpha$ -Achse resultieren. Das Suchintervall ist nun  $r_1 \in [-0.397; 2.710]$ .

Das Ergebnis der Berechnung der stabilen Polygone innerhalb dieses Suchintervalls zeigt Abbildung 3.6 in den Reglerparametern  $\mathbf{c}$ . Aus diesen Polygonen, transformiert in die Koordinaten  $\mathbf{r}$ , geht hervor, dass ausschließlich für  $r_1 \in [0.34; 0.69]$  stabile Polygone existieren.

### 3.1.3 Feineinstellung der Reglerparameter

Auf den Regler (3.4) kann die Feineinstellung (engl.: *fine-tuning*) der Parameter angewendet werden, da er im Zähler ein weiteres Polynom zweiten Grades besitzt,

$$p(s) = 6 + 4.5s + s^2. \quad (3.5)$$

Die drei Parameter  $\mathbf{p} = [p_0, p_1, p_2]^T$  dieses Polynoms können optimiert werden. Dazu muss vorerst ein Parametersatz aus dem Inneren des stabilen Gebietes von Abbildung 3.6 ausgewählt werden, der das System möglichst *gut* stabilisiert, beispielsweise

$$\mathbf{c}^* := [0.43, 1.316, 3.78]^T. \quad (3.6)$$

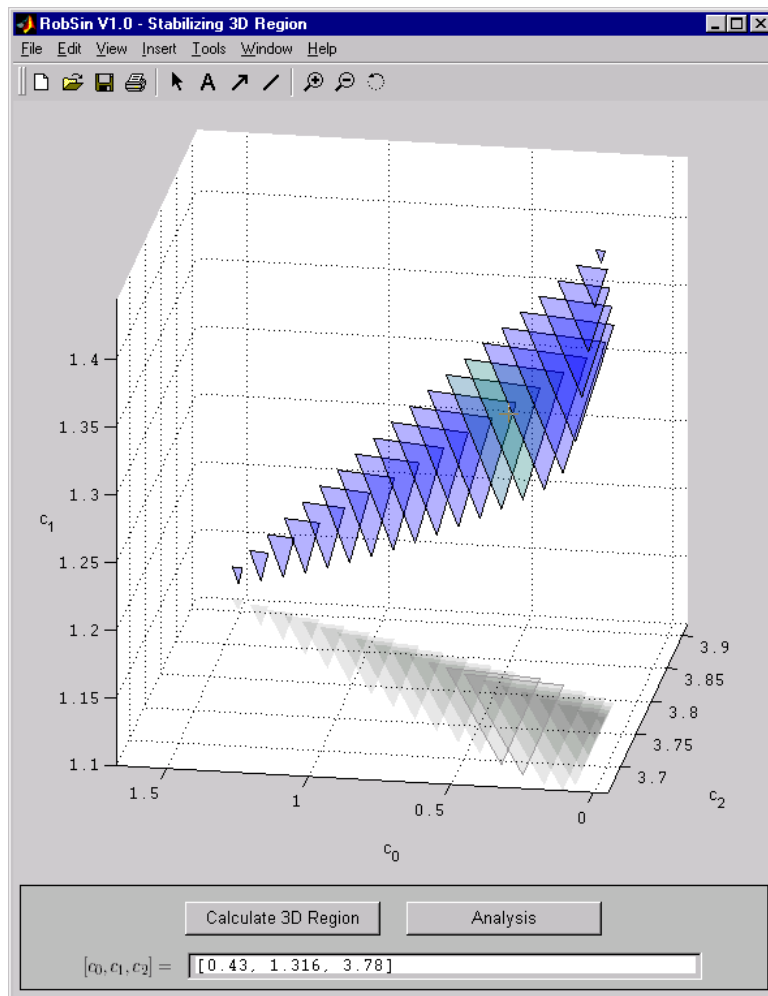


Abbildung 3.6: Das robuste  $\Gamma$ -stabile Gebiet, mit  $\sigma_{\max} = -0.1$ , für den geschlossenen Regelkreis aus  $G(s, k, m_1, m_2)$  und  $C(s, \mathbf{c})$ .

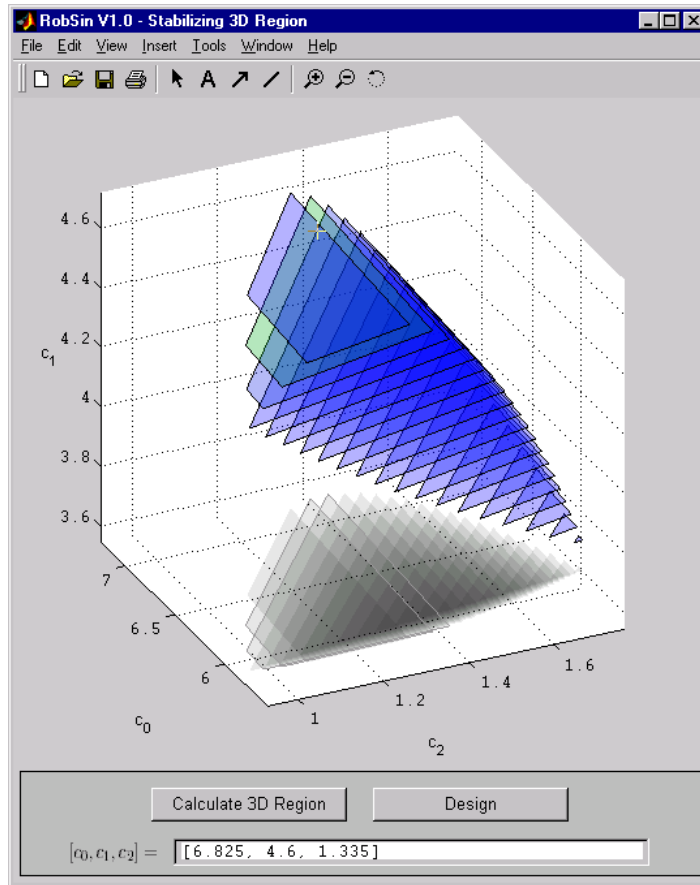


Abbildung 3.7: Das robust  $\Gamma$ -stabile Gebiet, mit  $\sigma_{\max} = -0.1$ , für den geschlossenen Regelkreis aus  $G(s, k, m_1, m_2)$  und  $\hat{C}(s, \mathbf{p})$  im Raum der Reglerparameter  $\mathbf{p}$ . (Bei *RobSin* heißen die Reglerparameter für andere Regler als PID-Regler immer  $\mathbf{c}$ .)

Aus  $C(s, \mathbf{c}^*)$  ergibt sich so der zu optimierende Regleransatz  $\hat{C}(s, \mathbf{p})$ , abhängig von den Reglerparametern  $\mathbf{p}$ ,

$$\hat{C}(s, \mathbf{p}) = \frac{1000 (p_0 + p_1 s + p_2 s^2) (0.43 + 1.316s + 3.78s^2)}{(s + 10)(s + 14.05)(s + 12.16)(s + 5.07)}. \quad (3.7)$$

Bei gleichem  $\sigma_{\max}$  resultiert für die Reglerstruktur  $\hat{C}(s, \mathbf{p})$  ein stabiles Gebiet, welches selbstverständlich auch den festen Regler  $\hat{C}(s, \mathbf{p}_1^*)$ , mit

$$\mathbf{p}_1^* = [6, 4.5, 1]^T, \quad (3.8)$$

enthält (siehe Abbildung 3.7). Der geschlossene Regelkreis mit dem Regler  $\hat{C}(s, \mathbf{p}_1^*)$  weist einen maximalen Realteil der Pole  $\sigma_{\max} \approx -0.1016$  auf. Innerhalb des berechneten Gebietes existieren jedoch Parametersätze mit etwas besserem

Einschwingverhalten. Beispielsweise besitzt der geschlossene Regelkreis mit den Reglerparametern

$$\mathbf{p}_2^* := [6.825, 4.6, 1.335]^T \quad (3.9)$$

den maximalen Realteil seiner Pole  $\sigma_{\max} \approx -0.1062$  (vgl. Abbildung 3.8) und somit nach (1.60) die garantierte maximale Einschwingzeit  $T_{\text{ein}} \leq 28.25$ .

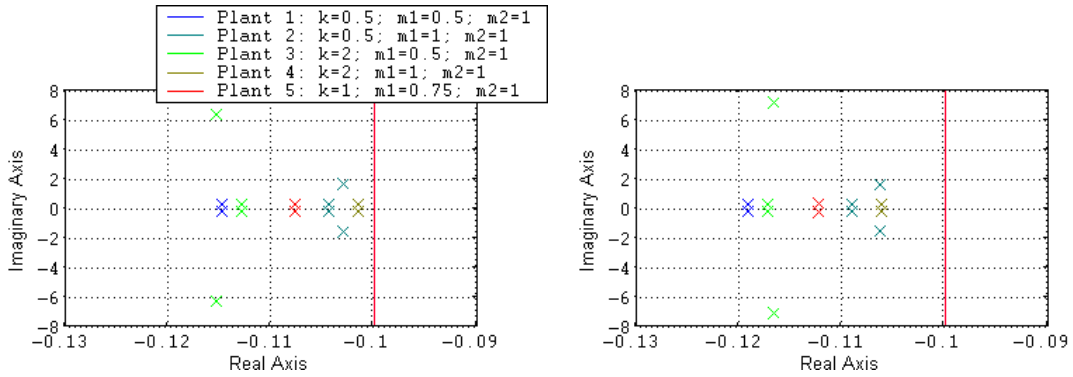


Abbildung 3.8: Ausschnitte aus dem *Analysis-Plots*-Rahmen von *RobSin*. Die Legende wurde im *Singular-Frequencies-Generator*-Fenster erzeugt. Links: Vergrößerter Ausschnitt aus dem Pol-Nullstellen-Diagramm des geschlossenen Regelkreises aus  $G(s, k, m_1, m_2)$  und  $\hat{C}(s, \mathbf{p}_1^*)$ . Rechts: Vergrößerter Ausschnitt aus dem Pol-Nullstellen-Diagramm des geschlossenen Regelkreises aus  $G(s, k, m_1, m_2)$  und  $\hat{C}(s, \mathbf{p}_2^*)$ . Die vertikale, rote Linie stellt jeweils den Rand des  $\Gamma$ -Gebietes  $\partial\Gamma$  für  $\sigma$ -Stabilität mit  $\sigma_{\max} = -0.1$  dar.

### 3.1.4 Totzeit

Das System aus Abbildung 3.1 besitze eine zusätzliche Totzeit  $L = 0.005$ , beispielsweise durch eine Verzögerung der Stellkraft  $u$  (siehe Abbildung 3.9),

$$G_L(s, k, m_1, m_2) = \frac{y(s)}{u(s)} = \frac{k / m_1 m_2}{s^2 \left( s^2 + k \frac{m_1 + m_2}{m_1 m_2} \right)} e^{-0.005s}. \quad (3.10)$$

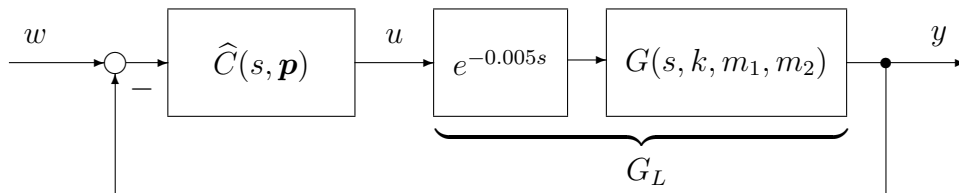


Abbildung 3.9: Regler  $\hat{C}(s, \mathbf{p})$ , Totzeitglied und Regelstrecke  $G(s, k, m_1, m_2)$  im Standardregelkreis.

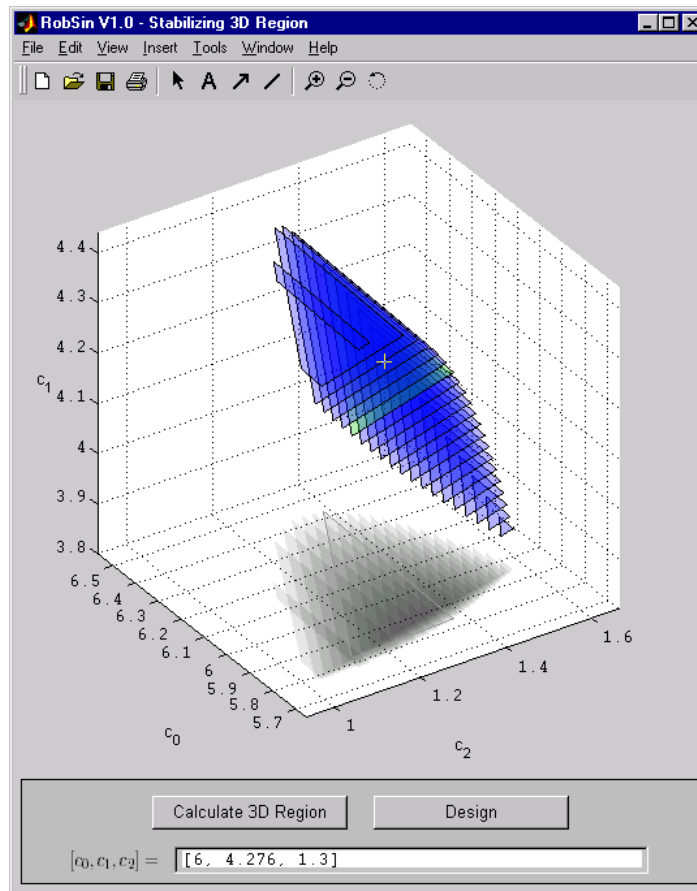


Abbildung 3.10: Das robust  $\Gamma$ -stabile Gebiet, mit  $\sigma_{\max} = -0.1$ , für den geschlossenen Regelkreis aus  $G_L(s, k, m_1, m_2)$  und  $\hat{C}(s, \mathbf{p})$  im Raum der Reglerparameter  $\mathbf{p}$ .

Gesucht sei wieder ein Parametersatz  $\mathbf{p}$  für den Regler  $\hat{C}(s, \mathbf{p})$ , der den geschlossenen Regelkreis robust  $\sigma$ -stabilisiert, mit  $\sigma_{\max} = -0.1$ .

Das Ergebnis stellt Abbildung 3.10 dar. Es ist eine Teilmenge des von Abbildung 3.7 gezeigten Gebietes. Den Stabilitätsnachweis liefert das in Abschnitt 2.3.10 vorgestellte Analyseverfahren „Sigma-Nyquist“ (vgl. Abbildung 3.11).

## 3.2 Zeitdiskreter Bereich

Die im vorigen Abschnitt durchgeführten Berechnungen lassen sich auch in den zeitdiskreten Bereich übertragen. Ein Vorteil der Methode der singulären Frequenzen für zeitdiskrete Systeme ist die gleichzeitige Behandlung von garantierter Einschwingzeit und Dämpfung (siehe Abschnitt 1.9).

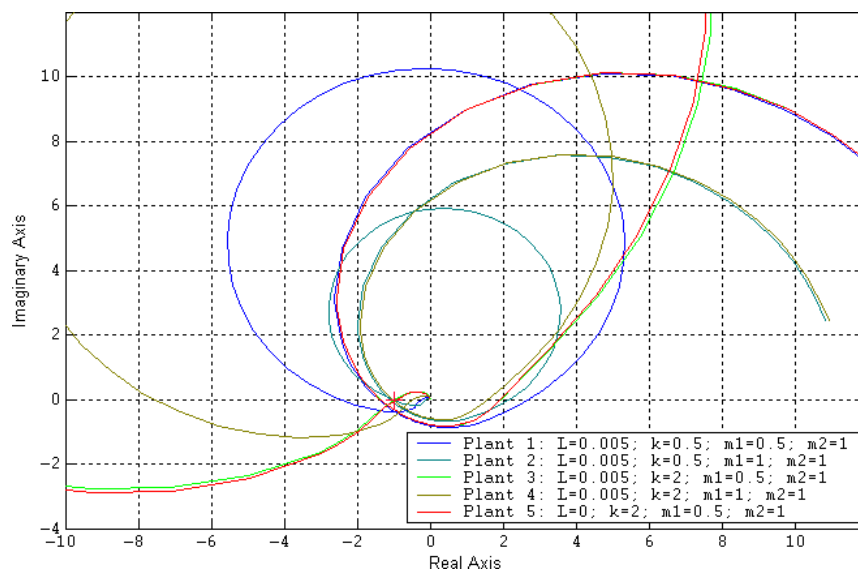


Abbildung 3.11: Das Sigma-Nyquist-Diagramm für den geschlossenen Regelkreis aus  $G_L(s, k, m_1, m_2)$  und  $\hat{C}(s, [6, 4.276, 1.3]^T)$ . Er ist robust  $\Gamma$ -stabil. Zusätzlich wird das Diagramm, welches sich aus der Regelstrecke  $G(s, 2, 0.5, 1)$  ergibt, angezeigt. Jede Kurve umschließt den Punkt  $s = -1$  zwei mal im mathematisch positiven Sinn, da  $G_L(s, k, m_1, m_2)$  und  $G(s, k, m_1, m_2)$  jeweils vier Pole auf der imaginären Achse, also außerhalb des  $\Gamma$ -Gebietes besitzen (vgl. Abschnitt 2.3.10).

Die exakte Formel für die Umrechnung von zeitkontinuierlichen Übertragungsfunktionen in den zeitdiskreten Bereich lautet

$$z = e^{sT_S}, \quad (3.11)$$

mit  $T_S$  gleich der Abtastzeit. Da die Methode der singulären Frequenzen nur auf Polynome anwendbar ist, können Regelstrecken, die mit dieser Formel berechnet wurden, nicht mit *RobSin* behandelt werden.

Um dennoch mit zeitdiskreten Systemen arbeiten zu können, muss die Umrechnungsformel (3.11) approximiert werden. Eine einfache Approximation stellt die Näherung für das Halteglied nullter Ordnung dar, auch bekannt als *Rückwärts-Rechteck-Approximation*,

$$s \triangleq \frac{z - 1}{T_S}. \quad (3.12)$$

Aus (3.1) folgt damit für eine Abtastzeit  $T_S = 0.05$  die zeitdiskrete Regelstrecke

$$G(z, k, m_1, m_2) = \frac{y(z)}{u(z)} = \frac{k / m_1 m_2}{400 (z - 1)^2 \left( 400 (z - 1)^2 + k \frac{m_1 + m_2}{m_1 m_2} \right)}. \quad (3.13)$$

Analog zum zeitkontinuierlichen Bereich benötigt der Regler vier Nullstellen, um die Eigenwerte in das gewünschte  $\Gamma$ -Gebiet zu verschieben. Transformiert man den zeitkontinuierlichen Regler  $\widehat{C}(s, \mathbf{p}_2^*)$  mit (3.12) in den zeitkontinuierlichen Bereich, so ist der sich damit ergebende Regelkreis instabil, da zum einen die Diskretisierung einer zusätzlichen Totzeit entspricht, zum anderen die Approximation ungenügend genau ist. Die Stabilisierung des Systems benötigt daher einen anderen Regler, beispielsweise

$$C(z, \mathbf{c}) = \frac{10\,000 (0.9726 - 1.952z + 0.98z^2) (c_0 + c_1z + c_2z^2)}{(2z - 1)(2z - 0.595)(2z - 0.784)(2z - 1.493)}. \quad (3.14)$$

Die Suche nach einem Parametersatz  $\mathbf{c}$ , der die Pole des geschlossenen Regelkreises in einen Kreis mit Mittelpunkt  $m = 0.005$  und Radius  $r = 0.99275$  verschiebt, ergab das in Abbildung 3.12 gezeigte Gebiet.

Dieses  $\Gamma$ -Gebiet entspricht einem  $\sigma_{\max} \approx -0.04505$  und somit einer Einschwingzeit  $T_{\text{ein}} \leq 66.59$ . Die Mindestdämpfung, die durch das  $\Gamma$ -Gebiet erreicht wird, beträgt  $D \leq 0.00392$ , da für diesen Wert der Kreis des  $\Gamma$ -Gebietes innerhalb der sich durch die Dämpfung ergebenden, logarithmischen Spiralen liegt (siehe Abschnitt 1.9).

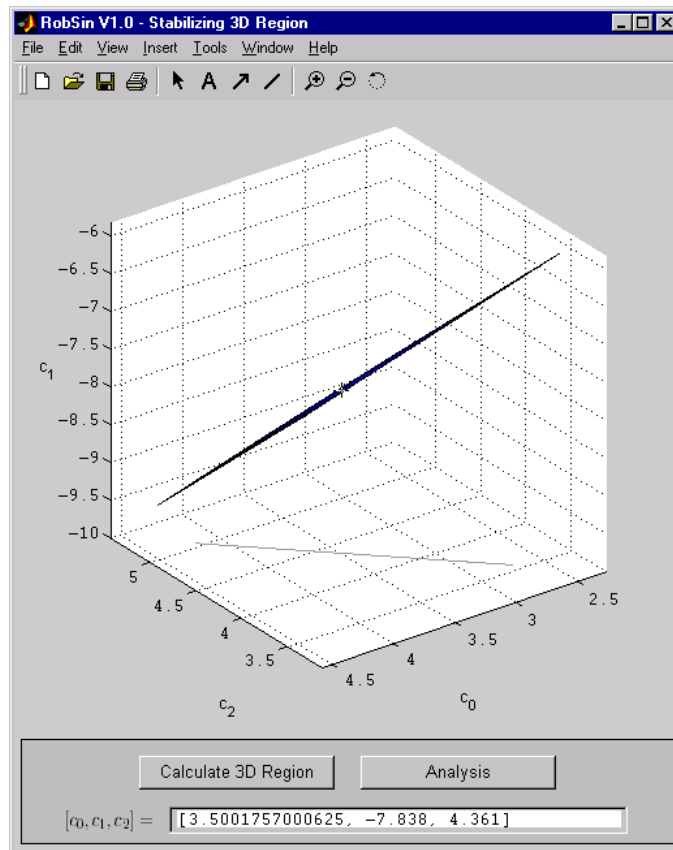


Abbildung 3.12: Das robust  $\Gamma$ -stabile Gebiet für den geschlossenen Regelkreis aus  $G(z, k, m_1, m_2)$  und  $C(z, \mathbf{c})$  für Kreisstabilität (Mittelpunkt  $m = 0.005$ , Radius  $r = 0.99275$ ).

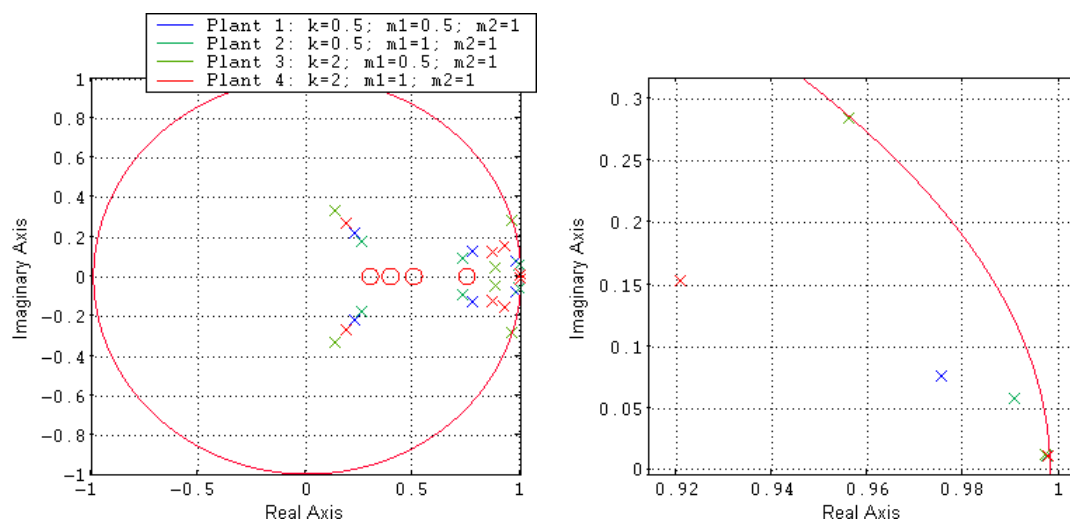


Abbildung 3.13: Ausschnitte aus dem *Analysis-Plots*-Rahmen von *RobSin*. Links: Pol-Nullstellen-Diagramm des robust stabilen, geschlossenen Regelkreises aus  $G(z, k, m_1, m_2)$  und  $C(z, [3.500, -7.838, 4.361])$ , welcher mit der Rückwärts-Rechteck-Approximation berechnet wurde. Rechts: Vergrößerter Ausschnitt aus der linken Abbildung. Der rote Kreis stellt jeweils die Berandung des  $\Gamma$ -Gebietes dar.

# Kapitel 4

## Software-Architektur und Funktionen

Die vorangegangenen Kapitel leiteten die theoretischen Hintergründe des Software-Werkzeuges *RobSin* her und bereiteten den Leser auf seine Anwendung vor. Dieses Kapitel erarbeitet dagegen die für den Softwareentwickler interessanten, programmiertechnischen Hintergründe von *RobSin*. Es setzt gewisse Grundkenntnisse in der objektorientierten Programmierung voraus.

### 4.1 Objektorientierte Programmierung

Moderne Programmiersprachen stellen die Verwendung von Datenstrukturen und objektorientierter Programmierung bereit. Datenstrukturen gruppieren logisch zusammenhängende Informationen. Die objektorientierte Programmierung erlaubt es, Funktionen zu definieren, die eine transparente Bearbeitung von in einer Datenstruktur gruppierten Information ermöglichen. Datenstrukturen mit zugehörigen Funktionen, sogenannten *Methoden*, heißen *Objekte*. Die Definitionen der Merkmale von Objekten heißen *Klassen*.

Das Lexikon [Obj01] nennt die folgenden, wesentlichen Vorteile der objektorientierten Programmierung gegenüber der prozeduralen Programmierung,

- Aufteilung von komplexen Programmen in überschaubare Einheiten,
- einfache, fest definierte Schnittstellen zwischen den einzelnen Objekten,
- Vermeidung von Programmierfehlern beim Informationsaustausch zwischen den Objekten,
- Reduzierung des Programmieraufwandes durch *Vererbung*.<sup>1</sup>

---

<sup>1</sup>Vererbung (engl.: inheritance) ist ein Fachbegriff aus der objektorientierten Programmierung. Sie ermöglicht die Wiederverwendung von Klassen durch Erzeugung einer hierarchi-

Objektorientierte Programmierung vereinfacht also nicht nur die Programmierung, sondern vor allem die Wartung und Erweiterung von Software enorm. Softwareentwickler, die mit einer Software nicht vertraut sind, können sich wesentlich schneller in den Programmcode einarbeiten als bei der prozeduralen Programmierung.

Die erste ernst zu nehmende objektorientierte Programmiersprache war C++. Ihr folgte die moderne Programmiersprache Java<sup>TM</sup>, welche auf eine plattform-unabhängige Gestaltung von Software-Werkzeugen spezialisiert ist. MATLAB<sup>®</sup> unterstützt ab Version 5 die Erzeugung von Datenstrukturen und Objekten. Das Software-Werkzeug *RobSin*, welches auf der MATLAB<sup>®</sup> Version 6.5 (R13) basiert, nutzt die Möglichkeit der objektorientierten Programmierung.

MATLAB<sup>®</sup> unterscheidet zwischen drei verschiedenen Gruppierungen von Funktionen und Methoden innerhalb von Objekten,

**öffentliche Methoden** können von allen Orten aufgerufen werden,

**private Methoden** sind nur sichtbar für die eigenen Methoden und Funktionen eines Objektes und können nur von diesen aufgerufen werden,

**lokale Funktionen** stehen in der Datei einer Methode und können nur innerhalb dieser Datei aufgerufen werden.

Um die Lesbarkeit des Quellcodes zu verbessern, weisen die Namen der Methoden und Funktionen der in *RobSin* verwendeten Objekte Unterscheidungsmerkmale auf, je nachdem zu welcher Gruppe sie gehören: private Methoden beginnen mit „private“, lokale Funktionen mit „local“ und öffentliche Methoden ohne gemeinsames Merkmal im Namen.

## 4.2 Kommunikation zwischen den Objekten

Das vorangehende Kapitel stellt die einzelnen, sichtbaren Elemente von *RobSin* vor. Jedes Fenster (*Figure*) und jeder Rahmen (*Frame*) sind als Objekte (*Fensterobjekte* bzw. *Rahmenobjekte*), also Instanzen von Klassen realisiert. Das Fensterobjekt des eigentlichen *RobSin* Fensters ist das Hauptobjekt (*RobSinGUI*), welches die anderen Fensterobjekte und die Rahmenobjekte beinhaltet. Jedes einzelne dieser Objekte verwaltet die Daten selbst, die es für die Visualisierung benötigt.

Diese Eigenständigkeit erfordert einen Informationsaustausch zwischen den Fenster- und Rahmenobjekten. Für diese Aufgabe besitzt jedes dieser Objekte eine Kommunikationsschnittstelle, die das Senden und Empfangen von Daten

---

schen Anordnung. Eine ausführliche Erklärung dieses Begriffes findet sich in der Literatur (z.B. [UII02]).

ermöglicht. Sie funktioniert analog zu der in Java<sup>TM</sup> weit verbreiteten Methode der *ActionEvents* (siehe [Ull02]).

Für die Kommunikation in eine Richtung (*unidirektional*), vom Senderobjekt zum Empfängerobjekt (*ActionListener*), benötigen die beiden beteiligten Objekte bestimmte Methoden. Das Senderobjekt besitzt die Methoden

**addActionListener(ActionListener)** Nimmt das Objekt *ActionListener* in der Liste aller Empfängerobjekte dieses Senderobjektes auf.

**removeActionListener(ActionListener)** Entfernt das Objekt *ActionListener* aus der Liste aller Empfängerobjekte dieses Senderobjektes.

**privateFireEvents(event,eventData)** Ruft die Methode *actionPerformed* der Empfängerobjekte aus der Liste aller Empfängerobjekte dieses Senderobjektes mit den Parametern *event* und *eventData* auf.

Die Methode *privateFireEvents* ist privat, und kann nur von Methoden des dazugehörigen Objektes gesehen werden. Das Empfängerobjekt besitzt die Methoden

**actionPerformed(event,eventData)** Überprüft, ob das Ereignis mit dem Namen *event* dem Empfängerobjekt bekannt ist. Wenn dies der Fall ist, wird das Ereignis durch Ausführen einer Befehlsfolge bearbeitet. Der Parameter *eventData* enthält die zum Ereignis gehörenden Daten.

**equals(object)** Gibt den Wert *wahr* zurück, falls das Objekt *object* das Empfängerobjekt ist, andernfalls *falsch*.<sup>2</sup>

**getIDData** Liefert eine eindeutige Zeichenkette für dieses Empfängerobjekt.

Die unidirektionale Kommunikation erlaubt auch die Übertragung eines Ereignisses an mehrere Empfängerobjekte.

Ein einfaches Beispiel für eine unidirektionale Kommunikation ist die Ausgabe von Aktionen oder Rechenergebnissen im *Echo*-Rahmen (siehe Kapitel 2). Der *Echo*-Rahmen übernimmt dafür die Rolle eines Empfängerobjektes. Bevor eine Kommunikation möglich ist, muss der Aufbau einer Kommunikationsschnittstelle mit *addActionListener* erfolgen. Dieser wird generell im Initialisierungsvorgang beim Programmstart vorgenommen. Anschließend übermittelt die Methode *privateFireEvents* das Ereignis an den *Echo*-Rahmen. Dazu ruft sie die Methode *actionPerformed* des *Echo*-Rahmens auf, welche die Befehlsfolge zur Ausgabe der Daten am Bildschirm durchläuft.

<sup>2</sup>MATLAB<sup>®</sup> bietet nicht die Möglichkeit zwei Objekte auf Identität hin zu überprüfen. Die Funktionsfähigkeit der Methode *removeActionListener* von Senderobjekten erfordert jedoch die Identitätsprüfung zweier Objekte. Zusammen mit der Methode *getIDData* konnte eine solche Überprüfung realisiert werden.

Die Kommunikation in beide Richtungen zwischen zwei Objekten (*bidirektional*) basiert auf der unidirektionalen Kommunikation. Beide Objekte übernehmen gleichzeitig die Rolle eines Empfängerobjektes und eines Senderobjektes. Zum Aufbau der bidirektionalen Kommunikation müssen sich beide beim jeweils anderen Objekt mit *addActionListener* anmelden.

Die Struktur der Kommunikation bei *RobSin* ist sternförmig, wobei das *RobSinGUI*-Objekt als zentraler Knoten empfangene Daten verarbeitet oder weiterleitet, sowie Daten aussendet (siehe Abbildung 4.1). Die sternförmige Anordnung hat gegenüber einem kreisförmigen Datenaustausch den Vorteil, dass die visuelle Struktur der Kommunikationsstruktur entspricht.

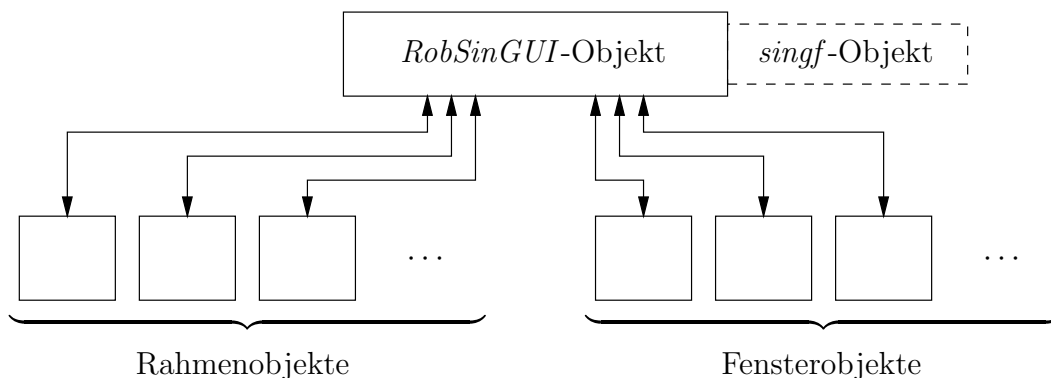


Abbildung 4.1: Kommunikation zwischen den Objekten. Das Objekt *singf* führt die Berechnungen des Gebietes aller stabilisierenden Regler durch.

Das *RobSinGUI*-Objekt leitet grundsätzlich die von ihm empfangenen Ereignisse an alle Empfängerobjekte weiter.<sup>3</sup> Die einzelnen Empfängerobjekte entscheiden selber, ob die Daten für sie relevant sind. Diese Objekte sind deswegen in gewisser Weise intelligent. Über den Umweg des zentralen Knotens können die einzelnen Fenster- und Rahmenobjekte miteinander kommunizieren.

Eine weitere besondere Eigenschaft dieser zentralen Kommunikation ist, dass die einzelnen Fenster- und Rahmenobjekte nichts über die gegenseitige Existenz wissen müssen, da das zentrale Objekt die Vermittlung der Informationen übernimmt. Dieses Objekt ist auch das einzige Objekt, das alle Fenster- und Rahmenobjekte kennt. Alle anderen Objekte kennen nur das zentrale *RobSinGUI*-Objekt.

Der Algorithmus für die Berechnungen des Gebietes aller stabilisierenden Regler ist Teil einer eigenen Klasse (*singf*). Eine Instanz dieser Klasse ist im *RobSinGUI*-Objekt enthalten. Ihre Funktionalität erläutert Abschnitt 2.5.

<sup>3</sup>Eine Übersicht über alle Ereignisse, die in *RobSin* implementiert sind, befindet sich in der Datei „allEvents.txt“ im Homeverzeichnis von *RobSin*.

### 4.3 Ablegen der Objektdaten im Speicher

Objektorientierte Programmiersprachen erlauben ihren Objekten Daten in Objektvariablen abzuspeichern. Dieses Speichern lässt auch MATLAB<sup>®</sup> zu. Jedoch existieren bei dieser Programmiersprache keine Referenzen auf Variablen oder Objekte, und somit gibt es nicht die Möglichkeit des Referenzaufrufs (engl.: *call by reference*). Aus diesem Grund müssen Objekte an der Stelle des Aufrufs einer Methode mit dem veränderten Objekt überschrieben werden.

Beispielsweise lautet in Java<sup>™</sup> der Aufruf einer Methode `setData(newData)`, welche die Daten in einer Objektvariablen eines Objektes `X` mit den neuen Daten `newData` überschreibt, wie folgt,

```
X.setData(newData);
```

In MATLAB<sup>®</sup> entspricht dieser Aufruf folgendem Befehl,

```
X = setData(X,newData);
```

Die bidirektionale Kommunikation zwischen zwei Objekten kann in Java<sup>™</sup> dadurch realisiert werden, dass jedes Objekt während des Kommunikationsaufbaus jeweils eine Referenz pro Kommunikationspartner speichert. Da diese Möglichkeit in MATLAB<sup>®</sup> nicht besteht, bedient sich *RobSin* der beiden Funktionen `setappdata(h,name,value)` und `getappdata(h,name,value)`. Diese können Daten in einem Fenster mit der Identifikationsnummer `h` (*handle*) von MATLAB<sup>®</sup> speichern bzw. aus diesem Fenster herauslesen. `name` ist die Bezeichnung für die eigentlichen Daten `value`.

Jedes Objekt speichert in der Initialisierungsphase von *RobSin* die beiden unveränderlichen Einträge, die Identifikationsnummer `h` des Fensters und Bezeichnung `name` für seine Daten `value`, jeweils als Objektvariable. Damit ist es jederzeit in der Lage seine Daten auszulesen und zu manipulieren.

Um auf das obige Beispiel zurückzukommen, betrachte man den nun vereinfachten Befehl zum Aufruf der Methode `setData`,

```
setData(X,newData);
```

Die Zuweisung `X=` kann weggelassen werden, da sich die Objektvariablen während der Laufzeit nicht ändern. Eine Änderungen in den Daten ist jedoch möglich. Ein Befehlsfolge in der Methode `setData`, die dies bewirkt, lautet,

```
value = getappdata(X.h,X.name);
value = newData;
setappdata(X.h,X.name,value);
```

Somit ist eine bidirektionale Kommunikation in MATLAB<sup>®</sup> möglich, trotz dem Fehlen von Referenzen.

## 4.4 Die Klassen von *RobSin*

Das Software-Werkzeug *RobSin* beinhaltet mehrere Klassen. Tabelle 4.1 listet alle Klassen zusammen mit einer kurzen Beschreibung auf.

Klasse	Beschreibung
AnalysisController-SelectionFrame	Rahmen zur interaktiven Auswahl eines Reglers in den $r$ -Koordinaten
AnalysisPlotsFrame	Rahmen für die Darstellung von Analysediagrammen
Controller	Reglerstruktur
ControllerFrame	Rahmen zur Auswahl der Reglerstruktur
EchoFrame	Rahmen für die Anzeige der zuletzt ausgeführten Aktionen
GammaRegionFrame	Rahmen für die interaktive Auswahl des $\Gamma$ -Gebietes
OptionsFigure	Fenster für die Anzeige und Eingabe der Einstellungen
Plant	Regelstrecke mit Q-Box
PlantsFrame	Rahmen für die Eingabe der Regelstrecke
RobSinGUI	<i>RobSin</i> -Fenster, beinhaltet alle anderen Fenster- und Rahmenobjekte, und das Objekt <i>singf</i>
singf	Algorithmen der Methode der singulären Frequenzen
SingularFrequencies-GeneratorFigure	Fenster zur Darstellung der Kurve $r_1(\alpha)$
StablePolygons-3DFigure	Fenster zur Darstellung des dreidimensionalen Gebietes aller stabilisierenden Reglerparameter

Tabelle 4.1: Die Klassen von *RobSin*.

## 4.5 Die Klasse *singf*

Die Algorithmen zur Methode der singulären Frequenzen beinhaltet die Klasse *singf*. Im Gegensatz zu Abschnitt 2.5, der eine Anleitung zur Bedienung von *RobSin* ohne die Benutzeroberfläche darstellt, fasst dieser Abschnitt die öffentlichen Methoden dieser Klasse in einer Tabelle zusammen:

Methode	Beschreibung
display	Zeigt Informationen zum Objekt im Command Window von MATLAB <sup>®</sup> .
genall	Erzeugt für ein Intervall für $r_1$ die inneren Polygone.
gensfs	Berechnet für einen Wert von $r_1$ die singulären Frequenzen.
genslice	Erzeugt für einen Wert von $r_1$ die inneren Polygone.
gensls	Berechnet und speichert die singulären Geraden.
get	Gibt eine Objektvariable zurück.
getandanalyse	Ermöglicht die Auswahl eines Reglers und zeigt für den Regelkreis mit diesem Regler das Nyquist-Diagramm und die Wurzelortskurve.
plotsfgens	Berechnet und zeigt die Kurve $r_1(\alpha)$ .
plotstabpolgs	Zeigt die zuletzt berechneten inneren Polygone für einen Wert von $r_1$ .
set	Weist einer Objektvariablen einen Wert zu.
singf	Ist der Konstruktor der Klasse. Er initialisiert das Objekt und führt einige Berechnungen aus.
subsasgn	Ermöglicht die Zuweisung von Objektvariablen mit c++-Syntax (z.B. <code>sq.DispObject=[];</code> ).
suboref	Ermöglicht den Aufruf von Methoden und den Zugriff auf Objektvariablen mit c++-Syntax (z.B. <code>sq.display</code> ).

Tabelle 4.2: Öffentliche Methoden der Klasse *singf*.

## 4.6 Das Verzeichnis *util*

Die Funktionalität der Klassen befindet sich hauptsächlich in ihren Methoden. Einige Programmteile finden jedoch in mehreren Klassen Verwendung oder sind nicht speziell auf eine Klasse zugeschnitten. Diese Programmteile wurden in einzelne Funktionen in das Verzeichnis „util“ (Abk.: *utilities*; dt.: Hilfsmittel) ausgliedert. Tabelle 4.3 gibt einen Überblick über diese Funktionen.<sup>4</sup>

<sup>4</sup>Die Eingabe `help <Name der Funktion>` in das Command Window von MATLAB<sup>®</sup> erzeugt eine detaillierte Beschreibung der jeweiligen Funktion.

Funktion	Beschreibung
<code>arg2str</code>	Bestimmt den Befehl, der die übergebene Variable erzeugen würde, als Zeichenkette (Umkehrfunktion zu „eval“).
<code>arg2strAccuracy</code>	Gleiche Funktion wie <code>arg2str</code> mit zusätzlicher Angabe der Zahlengenauigkeit. <sup>5</sup>
<code>bool2onoff</code>	Wandelt einen booleschen Ausdruck in eine Zeichenkette ( <code>on</code> oder <code>off</code> ) um.
<code>centerFigure</code>	Zentriert ein Fenster am Bildschirm, bzw. positioniert es unter Angabe einer prozentualen Positionsangabe.
<code>cutzeros</code>	Entfernt die von Maple <sup>®</sup> erzeugten Nullen und überflüssigen Kommata am Ende jeder reellen Zahl einer Formel.
<code>fast_ismember</code>	Gibt die Position eines Elementes in einem Array zurück. Ist im Vergleich zu <code>ismember</code> schneller, jedoch mit eingeschränkter Funktionalität.
<code>floorAccuracy</code>	Rundet eine Zahl nach der <code>n</code> -ten Stelle hinter dem Komma ab, mit <code>n</code> als Eingabeparameter.
<code>getFontSize</code>	Gibt die Höhe und Breite eines Buchstabens der angegebenen Schriftart zurück.
<code>getRobSinPath</code>	Gibt den <i>RobSin</i> -Pfad zurück.
<code>isdigit</code>	Überprüft, ob das übergebene Zeichen eine Ziffer ist.
<code>maple2tf</code>	Wandelt eine Zeichenkette in eine Übertragungsfunktion ( <i>tf</i> -Objekt) um.
<code>mathEval</code>	Wandelt eine Zeichenkette in eine Zahl oder Matrix um und lässt im Gegensatz zu <code>str2num</code> nur mathematische Funktionen zu.
<code>nyquistsigma</code>	Substituiert die komplexe Variable einer Übertragungsfunktion und führt anschließend mit der modifizierten Übertragungsfunktion die Funktion <code>nyquist</code> aus (siehe Abschnitt 2.3.10).
<code>roundAccuracy</code>	Rundet eine Zahl nach der <code>n</code> -ten Stelle hinter dem Komma, mit <code>n</code> als Eingabeparameter.
<code>setForAll</code>	Führt den Befehl <code>set</code> für alle übergebenen Elemente aus.
<code>setImageToAxes</code>	Lädt ein Bild aus einer Datei und zeigt es in einem Koordinatensystem.
<code>tf2maple</code>	Wandelt eine Übertragungsfunktion ( <i>tf</i> -Objekt) in eine Zeichenkette um.
(Fortsetzung auf der nächsten Seite.)	

<sup>5</sup>MATLAB<sup>®</sup> speichert reelle Zahlen mit einer Genauigkeit von maximal etwa 17 Ziffern.

Funktion	Beschreibung
<code>timebar</code>	Ähnlich der Funktion <code>waitbar</code> mit zusätzlicher Anzeige der noch benötigten Zeit und der Prozentangabe der bereits durchgeführten Arbeitsschritte in Verhältnis zum Gesamtaufwand.

Tabelle 4.3: Funktionen im Verzeichnis „util“.

# Kapitel 5

## Zusammenfassung und Ausblick

Die vorliegende Arbeit fasst die neuesten theoretischen Fortschritte, denen das Parameterraumverfahren zugrunde liegt, zusammen und stellt ein Software-Werkzeug vor, welches auf diesen Erkenntnissen aufbaut.

### 5.1 Schlussfolgerung über die Methode der singulären Frequenzen

Die Methode der singulären Frequenzen ist ein universelles Verfahren zur Bestimmung des Gebietes aller robust stabilisierenden Parameter von Reglern, welche ein Polynom im Nenner besitzen. Es ist anwendbar auf totzeitbehaftete Regelstrecken unterschiedlichsten Typs.

Dieses neue Syntheseverfahren lässt neben Hurwitz- und  $\sigma$ -Stabilität auch Kreisstabilität zu, wodurch für zeitdiskrete Systeme die gleichzeitige Einstellung von Einschwingzeit und Dämpfung ermöglicht wird.

Eine besondere Eigenheit dieses Verfahrens ist, dass sich in einer gedrehten Ebene im Raum der Reglerparameter konvexe Polygone als stabile Gebiete ergeben. Da diese Polygone durch singuläre Geraden begrenzt werden, ist eine überaus schnelle Berechnung des stabilen Gebietes möglich. Das gesamte dreidimensionale Gebiet ergibt sich durch Zusammensetzen einzelner paralleler Schnitte für verschiedene Werte des Rasterparameters  $r_1$ .

Die Behandlung von Totzeitsystemen bringt jedoch Schwierigkeiten mit sich, weil der Totzeiterm eine unendliche Anzahl an singulären Frequenzen und somit auch an singulären Geraden verursacht. Eine Begrenzung auf eine endliche Anzahl ist allerdings ohne den Verlust von aktiven Grenzen möglich.

## 5.2 Das Software-Werkzeug als Ergebnis

Im Rahmen dieser Arbeit wurde das Software-Werkzeug *RobSin* weiterentwickelt. Dieses Tool ermöglicht die Berechnung des Gebietes aller stabilisierenden Reglerparameter unter Verwendung der Methode der singulären Frequenzen.

Ausgehend von einem Programm, welches über die Kommandozeile bedient wird, konnte ein professionelles Software-Werkzeug zur Reglersynthese und -analyse geschaffen werden. Es vereint die komplette Funktionalität des Kommandozeilenprogrammes mit zusätzlichen Funktionen, wie beispielsweise der Regelkreisanalyse, in einer übersichtlichen und einfach zu bedienenden Benutzeroberfläche. Die ursprüngliche Möglichkeit der Kommandozeileneingabe ist weiterhin gegeben.

Das Programm erlaubt eine intuitive und computergestützte Festlegung von Regelstrecke mit Parameterunsicherheiten, Regler und  $\Gamma$ -Gebiet. Ein wesentlicher Bestandteil der Arbeit stellt die Möglichkeit zur interaktiven Analyse von Regelkreisen dar. Die berechneten Polygone werden in zwei Koordinatensystemen gezeigt, im Koordinatensystem der Reglerparameter und in den Koordinaten der Polygone. Dadurch ist die Auswahl eines Parametersatzes für die Analyse, neben der Tastatureingabe, auch mit der Maus möglich.

Dem Anwender stehen alle gängigen Analyseverfahren zur Verfügung. Für die Untersuchung von Totzeitsystemen nach  $\sigma$ -Stabilität existiert zudem das angepasste Nyquist-Kriterium.

Zudem besteht die Möglichkeit des Ladens und Speicherns von Sitzungen, so dass zu einem späteren Zeitpunkt mit bereits berechneten Ergebnissen weitergearbeitet werden kann. Eine weitere implementierte Funktion ist die Vergrößerung von Ausschnitten dargestellter Diagramme, die sowohl über Kontextmenüs, als auch über einen Menüpunkt im Menü des *RobSin*-Fensters erreichbar ist. In einem eigenen Fenster ist die Veränderung zahlreicher Einstellungen (engl.: *options*) durchführbar.

Das Programm zeichnet sich durch eine hohe Zuverlässigkeit aus, bei keinem der vierzehn getesteten Beispiele wurde ein Gebiet falsch berechnet.<sup>1</sup> Nur ein Beispiel konnte wegen seiner hohen Komplexität nicht behandelt werden.<sup>2</sup>

## 5.3 Ausblick auf mögliche zukünftige Entwicklungen

*RobSin* repräsentiert die Umsetzung des aktuellen Entwicklungsstandes der Methode der singulären Frequenzen für Regler zweiten Grades als Software. Bei diesem Verfahren fehlen allerdings noch einige wenige Erkenntnisse. So gibt es

---

<sup>1</sup>Die getesteten Beispiele befinden sich im Verzeichnis „examples“.

<sup>2</sup>Der Dateiname dieses Beispiels lautet „mass\_spring.mass.z.tustin.BUG\_fzero.mat“.

zur Zeit keine Möglichkeit für beliebige Strecken die Menge aller stabilen  $r_1$  zu bestimmen.

Mit einer automatischen Erkennung des stabilen Bereichs für  $r_1$  könnte der Schritt der manuellen Festlegung des  $r_1$ -Intervalls wegfallen. Weiterhin wäre eine schnelle automatische Minimierung der Einschwingzeit realisierbar.

Selbstverständlich sind, wie bei den meisten umfangreichen Computerprogrammen, unzählig viele Erweiterungen und Verbesserungen denkbar. So ist eine Erweiterung auf Regler höherer Ordnung, oder die Optimierung der verwendeten Algorithmen nach Rechenzeit und Speicherbedarf vorstellbar.

Da das Software-Werkzeug auf die Verwendung der symbolischen Berechnungen, und somit der teureren *Extended Symbolic Math Toolbox* nicht verzichten kann, beschränkt sich seine Verwendung stark. Eine Lösung, damit das Software-Werkzeug auch im Lehrbereich zum Einsatz kommen kann, wäre beispielsweise eine ferngesteuerte Bedienung über das Internet.



# Anhang A

## Fehlerbehandlung

Dieses Kapitel erklärt die Bedeutung von Fehlern und gibt Hinweise zu ihrer Behebung. Zunächst werden Fehler beschrieben, die von *RobSin* abgefangen werden und eine Fehlermeldung erzeugen. Der daran anschließende Abschnitt behandelt Fehler, die bei der Berechnungen zur Methode der singulären Frequenzen auftreten.

### A.1 Fehlermeldungen

Die meisten Fehlermeldungen haben ihre Ursache in einer falschen Benutzereingabe. Wie die Eingabe in die jeweiligen Eingabefelder aussehen muss, beschreibt Kapitel 2. An dieser Stelle werden ausschließlich Fehlermeldungen behandelt, die nicht durch falsche Eingaben verursacht wurden.

Fehlermeldung	Mögliche Ursache und Behebung
Could not calculate polygons. Error using ==> fzero Function values at interval endpoints must be finite and real.	Die Funktion <i>fzero</i> erzeugt einen Fehler. Die Analyse dieses Fehlers ergab für den untersuchten Fall, dass die Funktion <i>rehash</i> , welche nach Veränderungen in MATLAB® sucht, nicht korrekt arbeitet. Eine Lösung kann sein, den Algorithmus erneut mit gleichen Parametern zu starten. Falls der Fehler dann immer noch auftritt, sollte das Raster von $\alpha$ modifiziert werden.
(Fortsetzung auf der nächsten Seite.)	

Fehlermeldung	Mögliche Ursache und Behebung
Could not calculate polygons. Error using ==> fzero The function values at the interval endpoints must differ in sing.	Die Funktion <i>fzero</i> erzeugt einen Fehler. Der Algorithmus hat wahrscheinlich nicht alle kritischen Frequenzen gefunden, da das Raster für $\alpha$ falsch eingestellt war. Zur Lösung des Fehlers muss das Raster für $\alpha$ modifiziert werden.
Could not initialize singf object. Error using ==> maple Error, (in genAllSymbExps) KO...Jacobian is not ZERO!!!	Die Rangbedingung (1.18) ist nicht erfüllt. Die Ursache liegt in der Ungenauigkeit der Zahlen bei MATLAB <sup>®</sup> . Zur Umgehung dieses Fehlers wird empfohlen, die Genauigkeit der Zahlen zu verändern. Alternativ kann eine Modifikation des $\Gamma$ -Gebietes hilfreich sein.
Could not initialize singf object. Error using ==> maple Error, (in Voltages) invalid subscript selector	Eine Variable in Maple <sup>®</sup> (bzw. in der Extended Symbolic Math Toolbox) wurde falsch initialisiert. Die Ursache ist ein Fehler in der Klasse <i>singf</i> von <i>RobSin</i> . Zur Umgehung dieses Fehlers wird empfohlen, die Genauigkeit der Zahlen zu verändern. Alternativ kann eine Modifikation des $\Gamma$ -Gebietes hilfreich sein.

Tabelle A.1: Richtlinien zur Fehlerbehebung für Fehler mit Fehlermeldungen.

## A.2 Fehler bei Berechnungen zur Methode der singulären Frequenzen

Dieser Abschnitt untersucht Fehler, die bei Berechnungen zur Methode der singulären Frequenzen auftreten können.

Fehler	Mögliche Ursache und Behebung
Eine oder mehrere singuläre Frequenzen wurden nicht gefunden.	Das Raster für $\alpha$ muss angepasst werden (siehe Abschnitt 2.4). Der Algorithmus zur Bestimmung der singulären Frequenzen sucht zwischen jeweils zwei benachbarten Rasterpunkten nach den Nullstellen verschiedener Funktionen (siehe Unterabschnitt 1.6.2). Ist der Abstand zwischen zwei Rasterpunkten zu groß, kann es vorkommen, dass zwei oder mehrere singuläre Frequenzen in dem selben Bereich liegen. Der verwendete Algorithmus findet jedoch nur jeweils maximal eine Nullstelle innerhalb eines vorgegebenen Intervalls, so dass alle übrigen singulären Frequenzen nicht entdeckt werden. Ebenso besteht die Möglichkeit, dass das Suchintervall $I_{\text{ges}}$ zu klein gewählt wurde, so dass einige wichtige, singuläre Frequenzen außerhalb dieses Intervalls liegen und nicht erkannt werden können.
<i>RobSin</i> hat MATLAB <sup>®</sup> zum Absturz gebracht.	Die eingegebene Regelstrecke ist zu komplex. Für die Berechnungen der singulären Frequenzen müssen die Zahlen exakt gespeichert werden. Dazu wandelt der integrierte Algorithmus alle Zahlen in rationale Zahlen um. So kann es leicht passieren, dass die Variable, welche die oft sehr grosse Anzahl an Segmenten der singulären Geraden enthält, schnell mehr als hundert Kilo-byte umfasst.
Bei der Überquerung einer in der $(r_0, r_2)$ -Ebene horizontalen oder vertikalen Kante des Polygons ändert sich die Stabilität des Regelkreises nicht.	Dies ist kein Fehler. Es kann passieren, dass Polygone eine Kante im Unendlichen besitzen. Damit diese Polygone trotzdem dargestellt werden können, wurde ihnen eine künstliche Grenze hinzugefügt.
Die gefundenen Polygone sind nicht stabil.	Dies ist kein Fehler von <i>RobSin</i> . Der Algorithmus bestimmt ausschließlich die <i>inneren Polygone</i> (siehe Abschnitt 1.8). Existiert für ein festes $r_1$ kein stabiles Polygon, so heißt das nicht, dass es keine inneren Polygone gibt.

Tabelle A.2: Richtlinien zur Fehlerbehebung für Fehler ohne Fehlermeldungen.

# Literaturverzeichnis

- [ABB<sup>+</sup>02] ACKERMANN, J. ; BLUE, P. ; BÜNTE, T. ; GÜVENC, L. ; KAESBAUER, D. ; KORDT, M. ; MUHLER, M. ; ODENTHAL, D.: *Robust Control, The Parameter Space Approach*. 2. Auflage. London : Springer-Verlag, 2002. – ISBN 1–85233–514–9
- [AK00] ACKERMANN, J. ; KAESBAUER, D.: Entwurf robuster PID-Regler. Interlaken, Sept. 2000. – Proc. GMAWorkshop „Theoretische Verfahren der Regelungstechnik“ (Ed. K. Schlacher), S. 4–22
- [AKB02] ACKERMANN, J. ; KAESBAUER, D. ; BAJCINCA, N.: Discrete-time robust PID and three-term control. In: *15th Triennial World Congress of the International Federation of Automatic Control*. Barcelona, Spain, Juli 2002
- [Baj01] BAJCINCA, N.: The Method of Singular Frequencies for Robust Control Design in an Affine Parameter Space. In: *9th Mediterranean Conference on Control and Automation*. Dubrovnik, Juni 2001
- [Bia96] BIALKOWSKI, W.L. ; LEVINE, W.S. (Hrsg.): in *The Control Handbook*. CRC Press, 1996. – Kap. 72. – ISBN 0–8493–8570–9
- [BKA02] BAJCINCA, N. ; KOEPPE, R. ; ACKERMANN, J.: Design of Robust Stable Master-Slave Systems with Uncertain Dynamics and Time-Delay. In: *15th Triennial World Congress of the International Federation of Automatic Control*. Barcelona, Spain, Juli 2002
- [DHB00] DATTA, A. ; HO, M. T. ; BHATTACHARYYA, S. P.: *Structure and Synthesis of PID Controllers*. London : Springer-Verlag, 2000. – ISBN 1–85233–614–5
- [DLR03] *DRL: Institute of Robotics and Mechatronics - Tools*. <http://www.robotic.dlr.de/control/tools.html>. 2003
- [End93] ENDER, D. B.: *Process Control Performance: Not as Good as You Think*. Control Engineering, September 1993, S. 180–190

- [Gan65] GANTMACHER, F. R.: *Matrizenrechnung I*. zweite, berichtigte Auflage. VEB Deutscher Verlag der Wissenschaften, 1965
- [Hoh02] HOHENBICHLER, N.: *Auslegung robuster PID-Regler für Totzeitsysteme*. München, TU München, Lehrstuhl für Steuerungs- und Regelungstechnik, Diplomarbeit, 2002
- [LW02] LUTZ, H. ; WENDT, W.: *Taschenbuch der Regelungstechnik*. 4., korrigierte Auflage. Frankfurt am Main : Verlag Harri Deutsch, 2002. – ISBN 3–8171–1668–3
- [Man03] *Loop Tuning Fundamentals*. <http://www.manufacturing.net/ctl/index.asp?layout=article&articleid=CA307745>. 2003. – Abruf am 11.09.2003
- [MW003] *The MathWorks; Third-Party Products*. [http://www.mathworks.com/products/connections/application\\_area.shtml?app\\_id=9](http://www.mathworks.com/products/connections/application_area.shtml?app_id=9). 2003. – Abruf am 11.09.2003
- [Obj01] *Hyper-Lexikon: Objektorientierte Programmierung*. [http://www.hyperkommunikation.ch/lexikon/objektorientierte\\_programmierung.htm](http://www.hyperkommunikation.ch/lexikon/objektorientierte_programmierung.htm). 2001. – Abruf am 29.07.2003
- [O'D03] O'DWYER, A.: *Handbook of PI and PID Controller Tuning Rules*. London : Imperial College Press, 2003. – ISBN 1–86094–342–X
- [Sch00] SCHMIDT, G.: *Regelungs- und Steuerungstechnik 1*. Lehrstuhl für Steuerungs- und Regelungstechnik, Technische Universität München, 2000. – Skriptum zur Vorlesung
- [SDB02] SILVA, G. J. ; DATTA, A. ; BHATTACHARYYA, S. P.: New Results on the Synthesis of PID Controllers. In: *IEEE Transactions on Automatic Control* Bd. 47, 2002, S. 241–252
- [Ull02] ULLENBOOM, C.: *Java ist auch eine Insel*. Galileo Computing, 2002. – ISBN 3–89842–174–0
- [WB90] WIE, B. ; BERNSTEIN, D. S.: A Benchmark Problem for Robust Control Design. San Diego, CA, Mai 1990. – American Control Conference, S. 961–962
- [Wei99] WEISSTEIN, E. W.: *Cauchy-Riemann Equations*. <http://mathworld.wolfram.com/Cauchy-RiemannEquations.html>. 1999. – Abruf am 22.07.2003