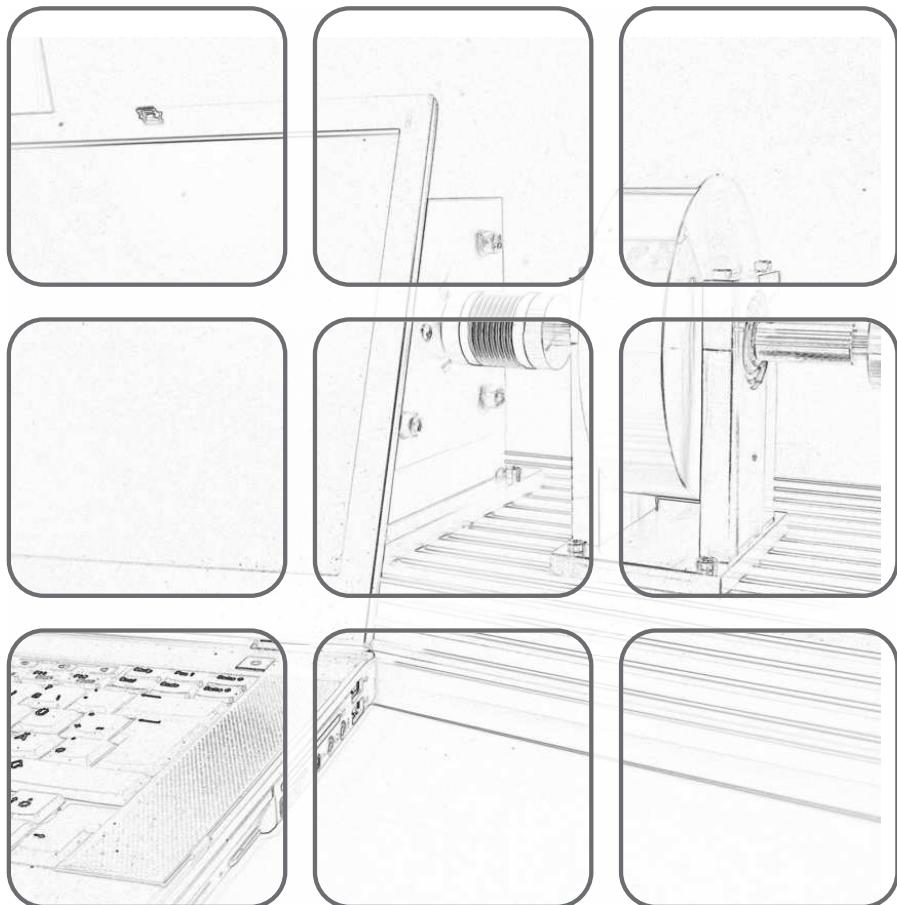


AUTOMATISIERUNGS- & STEUERUNGSSYSTEME

Vorlesung

Wintersemester 2020/2021

Dipl.-Ing. Martin Melik-Merkumians
unter der Leitung von
Univ.-Prof. Dr.sc.techn. Georg Schitter



Automatisierungs- und Steuerungssysteme

Vorlesung

Wintersemester 2020/2021

Dipl.-Ing. Martin Melik-Merkumians

TU Wien
Institut für Automatisierungs- und Regelungstechnik
Gruppe für industrielle Automation

Gusshausstrasse 27-29
1040 Wien
Telefon: +43 1 58801 – 37618
Internet: <http://www.acin.tuwien.ac.at>

Inhaltsverzeichnis

1. Leittechnik	1
1.1. Einführung in die Prozessleittechnik	1
1.1.1. Aufgaben und Begriffe leitechnischer Systeme	2
1.1.2. Automatisierungsspyramide	6
1.1.3. Historische Entwicklung von Prozessleitsystemen	8
1.1.4. Topologien leitechnischer Anlagen	10
1.1.5. Entwicklungstrends	13
1.2. Komponenten der Leittechnik	14
1.2.1. Prozessleitebene	15
1.2.2. Steuerungsebene	19
1.2.3. Feldebene	20
1.3. Beispiele für Prozessleitarchitekturen	28
1.4. Projektierung leitechnischer Strukturen	32
1.4.1. Rohrleitungs- und Instrumentierungsfließbild	32
1.5. Chargenprozesse	35
1.5.1. Rezepturen	35
2. Pneumatische Komponenten	38
2.1. Einführung in die Pneumatik	39
2.2. Drucklufterzeugung	43
2.2.1. Hubkolbenverdichter	44
2.2.2. Membranverdichter	46
2.2.3. Drehkolbenverdichter	46
2.2.4. Schraubenverdichter	47
2.3. Pneumatische Ventile	48
2.3.1. Betätigungsarten	49
2.3.2. Direkte und indirekte Ventile	50
2.3.3. Wechselventile	51
2.3.4. Zweidruckventile	51
2.3.5. Wegeventile	52
2.4. Pneumatische Zylinder und Antriebe	53
2.4.1. Einfachwirkende Zylinder	54
2.4.2. Doppeltwirkende Zylinder	56

2.4.3. Mehrstellungszyylinder	58
2.4.4. Tandemzyylinder	58
2.4.5. Schwenkantriebe	59
2.4.6. Pneumatische Muskel	59
2.5. Pneumatische Sensoren	60
2.5.1. Staudrucksensor	61
2.5.2. Ringstrahlsensor (Reflexdüse)	62
2.5.3. Luftstrahlschranke	63
3. Steuerungstechnik	64
3.1. Einführung	64
3.2. Verbindungsprogrammierte Steuerungen	64
3.3. Speicherprogrammierbare Steuerungen	65
3.3.1. Anforderungen und Auswahlkriterien	65
3.3.2. Kompatibilität	66
3.3.3. Wartbarkeit und Erweiterbarkeit	66
3.3.4. Kommunikationsschnittstellen	67
3.3.5. Abarbeitungsmodell	69
3.3.6. Robustheit	70
3.3.7. Modellierung nach dem Standard IEC 61131-3	71
3.3.7.1. Anweisungsliste	71
3.3.7.2. Strukturierter Text	71
3.3.7.3. Funktionsbausteinsprache	73
3.3.7.4. Kontaktplan	73
3.3.7.5. Ablaufsprache	76
3.3.8. Modellierung nach dem Standard IEC 61499	82
3.3.8.1. Motivation	82
3.3.8.2. Einführung	84
3.3.8.3. Funktionsbausteintypen	89
3.3.8.4. Zusammenfassung	99
4. Zuverlässigkeit und Sicherheit	100
5. Kommunikationssysteme	102
5.1. Einführung	102
5.1.1. Analoge Instrumentierung	103
5.1.2. Digitale Instrumentierung	103
5.1.3. Feldbusse	104

6. Industrieroboter	107
6.1. Einführung	107
6.1.1. Roboterkinematik	108
6.1.2. Antriebe	111
6.1.3. Sensoren	114
6.1.4. Endeffektor	114
6.1.5. Steuerung und Regelung	115
6.1.6. Programmierung	118
6.2. Kinematik	118
6.2.1. Koordinatensysteme	119
6.2.2. Translationen	120
6.2.3. Rotationen	120
6.2.4. Parametrierung von Rotationen	123
6.2.5. Homogene Transformationen	129
6.2.6. Kinematische Ketten	132
6.2.7. Denavit-Hartenberg – Konvention	134
6.2.8. Direktes und inverses kinematisches Problem	139
6.2.9. Manipulator-Jacobi-Matrix	146
6.2.10. Kinematische Redundanzen und Singularitäten	148
6.2.11. Differentielles Modell	150
6.3. Bahnplanung	153
6.3.1. Pfade und Bahnen	153
6.3.2. Arten der Bahnplanung	155
6.3.3. Punkt-zu-Punkt-Bewegungen	158
A. Kennzeichnung von PLT-Stellen	165
Literatur	170

Abbildungsverzeichnis

1.1.	Unterschiedliche Domänen und Prozesse der Leittechnik.	2
1.2.	Strukturierung von Prozessanlagen	3
1.3.	Beispiel des Datenobjekts eines Drucksensors	5
1.4.	Automatisierungspyramide bzw. Ebenenmodell nach NAMUR	5
1.5.	Anforderungen an die Ebenen der Automatisierungspyramide	7
1.6.	Veranschaulichung der drei industriellen Revolutionen	8
1.7.	Feldleitstand um 1950.	9
1.8.	Zentrale Leitwarte mit Mosaiktafel	11
1.9.	Prozessführung mit Bildschirmen.	11
1.10.	Moderne Leitwarte.	11
1.11.	Einordnung der Begriffe zentral, dezentral und verteilt	12
1.12.	Unterschiedliche Prozessleitsystem (PLS)-Topologien.	13
1.13.	Entwicklung von SPSen und PLSen	14
1.14.	Komponenten der Prozessleitebene in einer Client-Server-Architektur mit zwei getrennten Bussystemen.	16
1.15.	Bild einer Leitwarte mit mehreren Operator-Stationen (links) sowie einer grafischen Bedien- und Beobachtungsoberfläche (rechts).	18
1.16.	Client-Server-Architektur für Operator-Stationen (OSen)	18
1.17.	Prinzipieller Aufbau einer Prozessnahe Komponente (PNK).	19
1.18.	Versagenswahrscheinlichkeit einer Sicherheitsfunktion	21
1.19.	Kontinuierlicher Kapazitiver Sensor Liquicap T FM121	23
1.20.	Drehflügelschalter RN 3000 der Firma UWT	23
1.21.	Wirbeldurchflussmesser Prowirl 72F von Endress+Hauser mit Prinzipskizze. Die mittlere Grafik zeigt den prinzipiellen Aufbau. Rechts ist die Skizze einer Kármánschen Wirbelstraße zu sehen.	23
1.22.	Differenzdrucksensor Deltatop DO61W von Endress+Hauser mit Prinzipskizze. Durch die mechanische Verengung entsteht eine Druckdifferenz. Für den Durchfluss gilt $q \propto \sqrt{\Delta p}$	25
1.23.	Unterschiedliche diskrete Wegeventile als Schaltsymbol nach DIN ISO 1219. Von links nach rechts: 2/2-Wegeventil, 3/2-Wegeventil und 4/3-Wegeventil mit sperrender Mittelstellung.	26
1.24.	Stellventil der Firma Samson mit Schnittzeichnung.	26

1.25. Ideale, idealisierte und reale lineare Ventilkennlinie und der dabei verwendete Ventilkegel.	27
1.26. Ideale und reale gleichprozentige Ventilkennlinie und der dabei verwendete Ventilkegel.	27
1.27. Detailaufnahme eines Kavitationsschadens.	28
1.28. Beispielanwendung eines SIMATIC PCS 7.	29
1.29. Das Bild links zeigt die Hardware einer PCS 7 BOX, rechts ist eine Beispielanwendung dargestellt.	30
1.30. Beispielanwendung des Freelance Prozessleitsystems.	31
1.31. Übersicht über das Brewmaxx Prozessleitsystem und die Module <i>Brewmaxx Material</i> , <i>Brewmaxx Integrate</i> und <i>Brewmaxx Connect</i>	32
1.32. Direct iT - der Kern des Brewmaxx Prozessleitsystems.	33
1.33. Beispiel eines Rohrleitungs- und Instrumentierungsfließbilds	34
1.34. Aufbau und Varianten von PLT-Stellen.	34
1.35. Rezeptarten.	36
2.1. Tempeltüröffnung nach Heron.	38
2.2. Einordnung der Pneumatik als Teilgebiet der Fluidtechnik.	40
2.3. Ausgewählte Schaltsymbole der Fluidtechnik.	41
2.4. Beispiel einer vollpneumatischen Steuerung.	43
2.5. Mögliche Arbeitsbereiche der unterschiedlichen Verdichterprinzipien.	44
2.6. Prinzipskizze und p-V-Diagramm eines einfachen Hubkolbenverdichters. .	45
2.7. Prinzipskizze eines Membranverdichters: (1) Membran, (2) mediumfreie Rückseite der Zylinderkammer und (3) Kolben.	46
2.8. Prinzipskizze eines Vielzellen-Rotationsverdichters.	47
2.9. Bild und Skizze zum Schraubenverdichter aus [2].	48
2.10. Beispiele für Kugel- und Tellerventile. Die linke Abbildung zeigt ein einfaches 2/2-Wegeventil (Absperrventil) in Kugelbauform, die rechte Abbildung ein 3/2-Wegeventil in Tellerbauform mit Teller (1), oberem Dichtungskörper (2), Ventilkolben und unterem Dichtungskörper (4).	49
2.11. Typische Betätigungsarten von Ventilen gemäß DIN ISO 1219.	50
2.12. Prinzipskizze eines Wechselventils.	51
2.13. Prinzipskizze eines Zweidruckventils.	52
2.14. Symbol und Skizze eines 5/3-Wegevents.	52
2.15. Linke Abbildung: 5/2-Wegeventil, Durchfluss von 1 nach 2 und von 4 nach 5, linksseitig kombinierte Muskelkraft- und elektrische Betätigung, rechtsseitig federrückgestellt. Rechte Abbildung: 5/3-Wegeventil, in Mittelstellung gesperrt, beidseitig elektrische Betätigung, beidseitig zurückgestellt über Federn.	53

2.16. Einfachwirkender Kolbenstangenzylinder mit Skizze und beispielhafter Ansteuerung durch ein 3/2-Wegeventil. Zur Skizze: Kolben (1), Kolbendichtung (2), Kolbenstange (3), Kolbenstangendichtung (4), Rückstellfeder (5), Zylinderrohr (6), Zylinderboden (7), Druckluftanschluss (8), Zylinderdeckel (9), Buchse (10) und Auslass (11).	55
2.17. Skizze eines einfachwirkenden Membranzylinders mit Membran (1), oberer Membranhalterung (2), unterer Membranhalterung (3), Kolbenstange (4) und Rückstellfeder (5).	56
2.18. Doppeltwirkender Zylinder mit einseitiger Kolbenstange mit Skizze und Schaltung. Zur Skizze: Kolben (1), Kolbendichtung (2), Kolbenstange (3), Kolbenstangendichtung (4), Zylinderrohr (5), Zylinderboden (6), Zylinderdeckel (7), Buchse (8), Dämpfungskolben (9-10), Verstelldrossel (11-12) und Druckluftanschluss (13-14).	57
2.19. Prinzipskizze eines Vierstellungszylinders mit unterschiedlichen Hüben.	58
2.20. Beispiel eines Tandemzylinders.	59
2.21. Skizze eines Schwenkantriebes beispielhaft anhand einem Produkt der Firma Festo.	59
2.22. Skizze eines pneumatischen Muskels der Firma Festo mit Mutter (1), Flansch (2), Hülse (3) und Membran (4).	60
2.23. Schaltsymbol und Kraft-Auslenkungsdiagramm eines pneumatischen Muskels der Firma Festo.	61
2.24. Skizze und Kennlinie eines Staudrucksensors.	62
2.25. Skizze eines Ringstrahlsensors.	62
2.26. Anwendungsbeispiel einer Luftstrahlschranke zur Betätigung eines Zylinders.	63
 3.1. Netzwerkanbindung von Speicherprogrammierbare Steuerungen (SPSen)	68
3.2. Bearbeitungszyklus einer SPS	69
3.3. Kontaktplan Programm für das Schalten einer Lampe	74
3.4. Komponenten und Aufbau eines Sequential Function Charts (laut IEC 61131-3).	77
3.5. Funktionsweise des N-Aktionskennzeichners	78
3.6. Funktionsweise der Aktionskennzeichner S und R	78
3.7. SFC: Serielle Verbindung (Sequentielle Abfolge).	80
3.8. SFC: Parallele Verzweigung.	80
3.9. SFC: Auswahl-Verzweigung.	80
3.10. Trend der Steuerungstechnik von zentralen Strukturen hin zu verteilten Strukturen	83
3.11. Schematische Darstellung des Funktionsbausteins nach IEC 61499 – Function blocks [4]	85

3.12. Schematische Darstellung des Abarbeitungsmodells nach dem Standard IEC 61499	86
3.13. Systemmodell, Gerätemodell und Ressourcenmodell mit Anwendungen und Funktionsbausteinen nach IEC 61499	87
3.14. Anwendungsmodell nach IEC 61499	89
3.15. Beispiel einer Event Execution Control (EEC) welche als Event Control Chart (ECC) implementiert ist	91
3.16. Beispiel eines Composite Function Blocks	92
3.17. Beispiel eines Requesters und Responders	93
3.18. Ablaufdiagramm eines Requesters	95
3.19. Ablaufdiagramm eines Responders	96
3.20. Ablaufdiagramm eines Publisher/Subscribers	97
3.21. Ablaufdiagramm eines Client/Servers	97
3.22. Schematische Darstellung des Adapterkonzepts	98
4.1. Empirischer Zeitverlauf der Versagensrate $\lambda(t)$, auch als „Badewannenkurve“ bekannt.	101
5.1. Beispiel eines Feldmultiplexers.	104
6.1. Unterschiedliche Arten von Robotern: Industrieroboter, mobile, humanoide und androide Roboter.	107
6.2. Schematische Darstellung eines Dreh- und Schubgelenks.	109
6.3. Unterscheidung zwischen serieller und paralleler Kinematik.	110
6.4. Beispiele für parallele Kinematiken. Links: STEWART-GOUGH-Plattform; rechts: „Flexpicker“-Industrieroboter der Firma ABB.	110
6.5. Typische Konfigurationen von Industrierobotern und ihre Arbeitsbereiche.	111
6.6. Beispiel eines SCARA-Roboters mit nierenförmigem Arbeitsbereich.	112
6.7. Schematischer Aufbau und Drehachsen eines Knickarm-Industrieroboters.	112
6.8. Absolutgenauigkeit und Wiederholgenauigkeit	115
6.9. Veränderung der Regelkreisstruktur durch externe Sensoren. Oben: Keine Information über die Umgebung; Mitte: Führung zu externen definierten Orten durch Näherungssensoren; Unten: Verwendung von bildgebenden Sensoren.	117
6.10. Koordinatensystem mit orthonormalen Basisvektoren.	119
6.11. Translation eines Koordinatensystems.	120
6.12. Die Multiplikation von Rotationsmatrizen ist nicht kommutativ.	122
6.13. Illustration der Eulerschen Winkel.	124
6.14. Illustration der Roll-Pitch-Yaw-Winkel.	126
6.15. Definition unterschiedlicher Koordinatensysteme.	134
6.16. Beispielhafte Anwendung der Denavit-Hartenberg-Konventionen.	135

6.17. Der Stanford-Manipulator mit der kinematischen Kette RPRRRR.	138
6.18. Der PUMA-Roboter mit der kinematischen Kette RRRRRR.	140
6.19. Illustration von direkter und inverser Kinematik als Operationen zwischen Konfigurations- und Arbeitsraum.	142
6.20. Planare Darstellung der beiden Lösungen des SCARA-Manipulators. Die linke Darstellung zeigt die Konfiguration \bar{q}_a , die rechte die Konfiguration \bar{q}_b	145
6.21. Beispiel eines redundanten planaren Manipulators mit $n = 5$ und $w = 3$. .	149
6.22. Beispiel einer manuellen Pfadangabe mittels Start-, Durch-, Via- und Endpunkte.	155
6.23. Beispielhafte Bahnen bei Planung im Gelenksraum (links) und im Arbeitsraum (rechts).	156
6.24. Rechentechnische Implementierung von konfigurationsraum- und arbeitsraumorientierter Bahnplanung.	157
6.25. Beispielhafte Bahnplanung per LSPB-Methode für Viapunkte.	162
6.26. Beispielhafte Bahnplanung per LSPB-Methode eines Durchpunkts q_i	163

Tabellenverzeichnis

2.1. Gegenüberstellung von Hydraulik und Pneumatik.	42
3.1. Verwendete Operatoren in ST.	72
5.1. Übersicht über verbreitete Feldbusse und ihre Eigenschaften.	105
6.1. Vergleich der Vor- und Nachteile von elektrischen, hydraulischen und pneumatischen Antrieben für Roboter.	113
6.2. Der Satz von DH-Parametern für den PUMA-Roboter in Abb. 6.18. . . .	140
6.3. Parameterwerte des SCARA-Roboters.	145
A.1. Prozessleittechnische Kategorien nach ISA-88.	165
A.2. Prozessleittechnische Funktion nach ISA-88.	166

Abkürzungsverzeichnis

ABK	Anzeige- und Bedienkomponente
AI	Adapter Interface
AS	Automatisierungsstation
AWL	Anweisungsliste
BFB	Basic Function Block
BNK	Benutzernahe Komponente
BuB	Bedien- und Beobacht-Station
CFB	Composite Function Block
DIN 61512-2	
ECC	Event Control Chart
EEC	Event Execution Control
ERP	Enterprise Resource Planning
ES	Engineering Station
FB	Funktionsbaustein
FB	Function Block
FBD	Function Block Diagram
FBN	Function Block Network
FBS	Funktionsbausteinsprache
IEC 61131	IEC 61131 – Programmable controllers
IEC 61499	IEC 61499 – Function blocks
IPC	Industrie-PC
MES	Manufacturing Execution System
OPC UA	OPC Unified Architecture
OS	Operator-Station
PAA	Prozessabbild der Ausgänge
PAE	Prozessabbild der Eingänge
PEAK	Prozess-Ein-/Ausgabekomponente

PFK	Prozessferne Komponente
PLS	Prozessleitsystem
PNK	Prozessnahe Komponente
PS	Prozessstation
R&I	Rohrleitungs- und Instrumentierungsfließbild
SCADA	Supervisory Control and Data Acquisition
SCARA	Selective Compliance Assembly Robot Arm
SIFB	Service-Interface Function Block
SIL	Sicherheits-Integritätslevel
SPS	Speicherprogrammierbare Steuerung
ST	Structured Text
VPS	Verbindungsprogrammierte Steuerung
XML	eXtensible Markup Language

Vorwort

Die Automatisierungstechnik ist ein interdisziplinäres Forschungs- und Anwendungsgebiet, fungiert als Bindeglied unterschiedlichster Ingenieurwissenschaften und ist aus dem alltäglichen Leben sowie dem industriellen Umfeld nicht mehr wegzudenken. Am Institut für Automatisierungs- und Regelungstechnik (ACIN) werden die neuesten Erkenntnisse und Methoden der unterschiedlichen Wissensgebiete der Automation zur Lösung konkreter praktischer Problemstellungen, sowie zur Entwicklung neuer Produkte und Verfahren angewandt. Besonderer Wert wird dabei in allen Bereichen auf eine sehr enge Verknüpfung von Methodenentwicklung und anwendungsbezogener Forschung gelegt. Das Institut ist Forschungspartner für namhafte nationale und internationale Unternehmen, sowohl im Rahmen von Forschungskooperationen als auch in einer Vielzahl von nationalen und europäischen Forschungsverbundprojekten beteiligt.

Das vorliegende Skriptum begleitend zur Vorlesung *Automatisierungs- und Steuerungssysteme* (376.047, VO) behandelt Architekturen von Steuerungssystemen, deren Hauptkomponenten und Kommunikationsmöglichkeiten, Themen der industriellen Robotik, sowie die Betrachtung der Sicherheit und Zuverlässigkeit eben dieser industrieller Systeme. An den jeweiligen Stellen sind Hinweise auf alternative Literatur bzw. relevante Materialien verzeichnet. Begleitend zu dieser Vorlesung werden die dazugehörigen praktischen Übungen im *Labor Automatisierungs- und Steuerungssysteme* (376.048, LU) abgehalten. Komplementär zur dieser Vorlesung wird von derselben Abteilung am ACIN (Advanced Mechatronic Systems (AMS), Prof. Schitter) die Vorlesung (376.050, VO) und Laborübung (376.051, LU) *Mechatronische Systeme* gehalten, in denen weitere Komponenten und Zusammenhänge mechatronischer Automatisierungssysteme behandelt werden.

Ein herzlicher Dank gilt an dieser Stelle allen Mitautoren und Kollegen für ihre Mitwirkung bei der Erstellung dieses Skriptums, insbesondere jedoch den Herren Andreas Deutschmann und Michael Robin, welche im Rahmen ihrer Anstellung als studentische Mitarbeiter einen großen Beitrag dazu geleistet haben.

Wien im September 2015

Univ.-Prof. Dr.sc.techn. Georg Schitter

Michael Steinegger, M.Sc.

1. Leittechnik

1.1. Einführung in die Prozessleittechnik

Nähern wir uns dem Begriff der Prozessleittechnik durch seine drei Bestandteile: Prozess, Leiten und Technik. Gemäß dem Internationalen Elektrotechnischen Wörterbuch (IEC 60050) gelten die folgenden beiden Definitionen:

„Ein Prozess ist die Gesamtheit von aufeinander einwirkenden Vorgängen in einem System, durch die Materie, Energie oder Information umgeformt, transportiert oder gespeichert wird“

und

„Leiten bezeichnet zweckmäßige Maßnahmen an oder in einem Prozess, um vorgegebene Ziele zu erreichen.“

Der Begriff der Prozessleittechnik umfasst also sämtliche technischen Methoden, Verfahren und Einrichtungen zur Führung von Prozessen, um ein extern vorgegebenes Ziel zu erreichen. Im Allgemeinen soll dies unter Einhaltung verschiedener quantitativer und qualitativer Nebenbedingungen wie z. B. dem Schutz von Mensch und Material erfolgen.

Dieses Konzept lässt sich auf unterschiedlichste Domänen anwenden. Neben Produktionsprozessen besitzen auch Netze – seien es Energienetze, Wassernetze oder Informationsnetze –, Gebäude und Verkehrseinrichtungen leittechnische Aufgabenstellungen (siehe Abb. 1.1). Bei den hier betrachteten Produktionsprozesse kann weiter zwischen verfahrenstechnischen und fertigungstechnischen Prozessen unterschieden werden.

Verfahrenstechnische Prozesse zielen meist auf die Verarbeitung von Fluiden ab. Kontinuierliche Prozesse wie Wasseraufbereitung, Raffinierung von Erdöl oder Glaserzeugung werden dabei als Fließprozesse bezeichnet. Einen typischen kontinuierlichen Prozess der Elektrotechnik stellt beispielsweise die Energieerzeugung in Kraftwerken dar. Wird hingegen nur eine begrenzte Stoffmenge hintereinander verarbeitet, spricht man von Chargen- oder Batchprozessen, welche eine Mischform zwischen kontinuierlichen und diskontinuierlichen Prozessen darstellen. Beispiele dafür sind die Produktion von

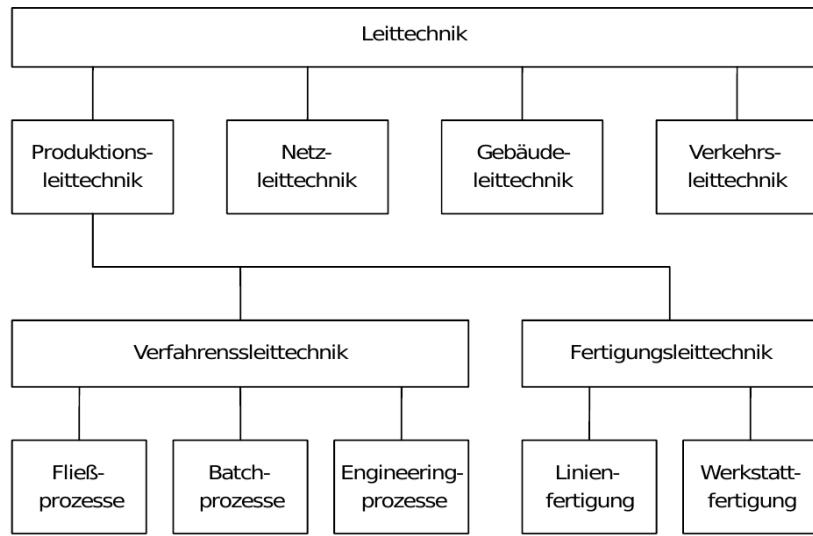


Abbildung 1.1.: Unterschiedliche Domänen und Prozesse der Leittechnik.

Arzneimitteln in der Pharma industrie, die Batchprozesse des Braugewerbes oder die Prozessschritte der Halbleiterindustrie. Auch die Produktion von technischem Wissen kann als Engineeringprozess betrachtet werden.

Fertigungstechnische Prozesse hingegen sind überwiegend diskontinuierliche Prozesse, welche auch als Stückgutprozesse bezeichnet werden. Die Produktion von elektronischen Schaltungen, Autos oder Küchengeräten sind Beispiele für Stückgutprozesse in stark automatisierter Linienfertigung, während z. B. teures technisches Gerät wie Generatoren eher in Werkstättenfertigung produziert werden.

1.1.1. Aufgaben und Begriffe leittechnischer Systeme

Unabhängig von der Domäne erfüllen Leitsysteme eine Reihe von Aufgaben:

Führen: Damit bezeichnet man die selbsttätige Steuerung und Regelung von Anlagen- einheiten und Prozessabläufen. Sie automatisiert die Bedienung des Systems, um externe Vorgaben zu erreichen.

Sichern: Technische Prozesse besitzen im Allgemeinen unzulässige Prozesszustände, die aus Sicherheitsbedenken nicht angenommen werden dürfen. Die Prozesssicherung stellt unabhängig von Vorgaben der Prozessführung sicher, dass sich der Prozess stets in zulässigen Zuständen befindet bzw. im Fehlerfall in einen sicheren Zustand

übergeführ werden kann (z. B. durch Teilabschaltung). Die sicherheitstechnische Relevanz solcher Funktionen verlangt besonders hohe Verfügbarkeitsanforderungen (vgl. Kapitel 4).

Anzeigen: Das Prozessleitsystem selbst soll in der Lage sein, die verteilten Informationen über den Anlagenzustand übersichtlich und konzentriert anzeigen zu können.

Bedienen: Dem Bedienpersonal soll die Möglichkeit gegeben werden, in den Prozess einzugreifen und Vorgaben der Prozessführung zu verändern.

Melden und alarmieren: Beim Eintreten bestimmter Ereignisse soll das Bedienpersonal visuell und/oder akustisch davon in Kenntnis gesetzt werden und mit zusätzlichen Informationen versorgt werden.

Archivieren: Eingetretene Ereignisse, Messwerte und Zustände des Prozesses sollen dauerhaft gespeichert werden, um den Zustandsverlauf des Systems jederzeit rekonstruieren zu können.

Protokollieren und auswerten: Die aktuellen und archivierten Daten sollen automatisiert gelesen, ausgewertet und zu einem Protokoll verarbeitet werden. So können die Herstellungsumstände eines Produkts automatisch dokumentiert werden.

Neben diesen grundlegenden Funktionen können zusätzlich höhere Prozessleitfunktionen zur Verfügung gestellt werden. Beispielsweise können umfassende Rezeptverwaltungen mit parametergesteuerten Rezepturen, Statistische Prozesslenkung, ein statistisches Verfahren, um in Produktionsprozessen ein vordefiniertes Maß an Qualität kostenoptimal einzuhalten, oder optimalitätsbasierten Steuerungen und Regelungen vorhanden sein.

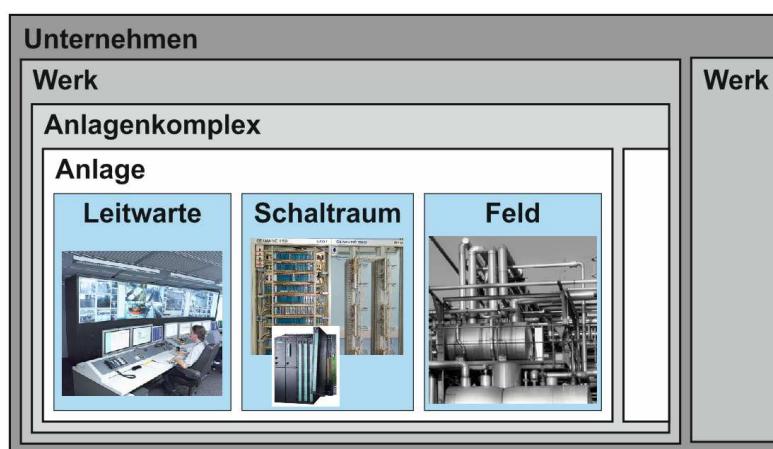


Abbildung 1.2.: Strukturierung von Prozessanlagen gemäß ISA-88 / DIN EN 61512-1.

Um Prozessleitsysteme (PLSe) und deren unterschiedlichen Topologien beschreiben zu können, ist es notwendig, die von Prozessanlagen verwendeten räumlichen Begriffe zu kennen. Abb. 1.2 veranschaulicht die gem. ISA-88 / DIN EN 61512-1 in Prozessanlagen auftretenden Bezeichnungen. Ein Unternehmen besteht darin aus mehreren Werken, welche aus mehreren Anlagenkomplexen bestehen. Diese können wiederum in einzelne Anlagen unterteilt werden. Zu jeder Anlage gehört ein sogenanntes Feld, die Gesamtheit aller Prozesskomponenten. Der Schaltraum bezeichnet einen meist nur Fachpersonal zugänglichen Raum mit elektrotechnischen Einrichtungen wie Schaltanlagen und Sicherheitseinrichtungen. Zur Überwachung und Steuerung der Anlage dient die Leitwarte, in welcher Operator-Stationen die Interaktion mit dem Prozessleitsystem ermöglichen.

Um die oben aufgezählten Aufgaben im Kontext industrieller Fertigung übernehmen zu können, entsteht ein enormer Aufwand an Informationsverarbeitung, der die Interaktion unzähliger Maschinen und Personen erfordert. Das gesamte Tätigkeitsspektrum vom Prozessleitsystem kann so selbst als Prozess zur Gewinnung, Übertragung, Verarbeitung, Nutzung und Darstellung von Information gesehen werden. Die Verwendung eines weitgehend einheitlichen Informationssystems digitaler Informationen ermöglicht es, diesen Prozess in fortschreitendem Umfang und über alle Ebenen hinweg mit Computersystemen zu erledigen.

Im Hinblick auf einheitliche Informationssysteme wird zwischen Prozessleitsystemen erster und zweiter Generation unterschieden. Prozessleitsysteme der ersten Generation besitzen kein einheitliches Informationssystem. Die Repräsentation der Daten hängt von der speziellen Komponente ab, welche deshalb durch ihre Hardware spezifische Funktionen erfüllen und nicht austauschbar ist. Gemeint sind damit z. B. dedizierte Engineering Stationen (ESen) und Bedien- und Beobacht-Stationen (BuBs).

Systeme der zweiten Generation hingegen basieren auf einem systemweit einheitlichen, objektorientierten Informationsmodell. Datenverarbeitung und -darstellung kann dadurch von generischen Komponenten übernommen werden, jede Komponente ist daher in der Lage jede Aufgabe auszuführen. Solche Systeme sind offen hinsichtlich übergeordneten Systemen sowie Komponenten der Feldebene. Abb. 1.3 zeigt ein beispielhaftes Datenobjekt, welches im gegebenen Fall einer physikalisch vorhandenen Prozessleitstelle zur Druckmessung zugeordnet ist.

Das komplexe Wirkungsgefüge aller Komponenten eines Prozessleitsystems wurde in den 1980er und 1990er Jahren in einem Ebenenschema systematisiert und in den Unternehmenskontext eingegliedert. Im Folgenden soll mit der Automatisierungspyramide eine aktuelle Systematisierung und Hierarchisierung der industriellen Fertigung gezeigt werden.

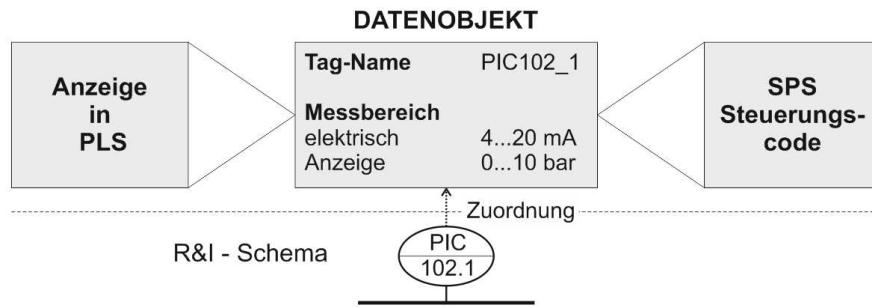


Abbildung 1.3.: Beispiel des Datenobjekts eines Drucksensors in Systemen zweiter Generation.

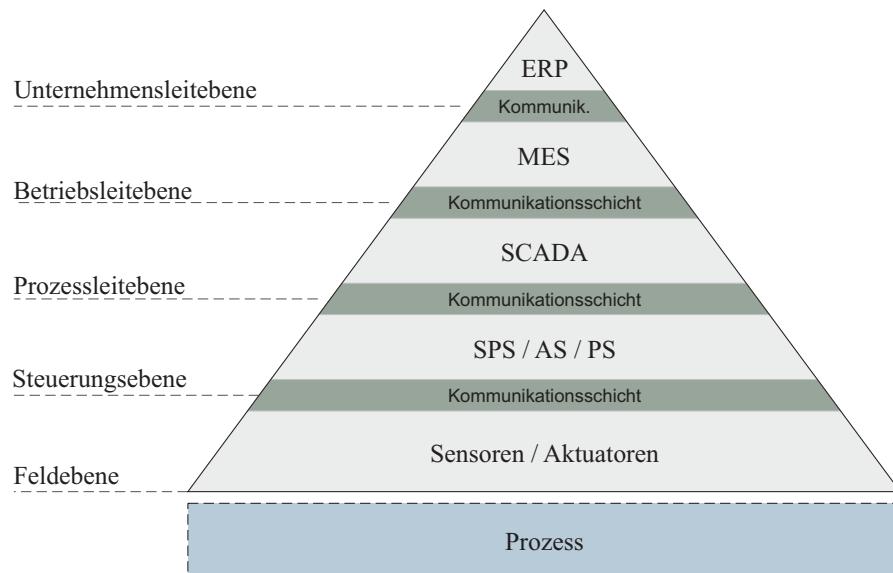


Abbildung 1.4.: Automatisierungspyramide bzw. Ebenenmodell nach NAMUR (Interessengemeinschaft Automatisierungstechnik der Prozessindustrie).

1.1.2. Automatisierungspyramide

Die Automatisierungspyramide (siehe Abb. 1.4) unterscheidet zwischen fünf technischen Ebenen samt dazwischen liegenden Kommunikationsschichten, welche über den Produktionsprozess konstruiert werden, um die Komplexität der Gesamtaufgabe auf einzelne Teilaufgaben herunterzubrechen. Jeder Ebene sind dabei spezifische Aufgaben und Ziele zugeordnet:

Feldebene Die Feldebene umfasst Anlagen und Geräte „im Feld“, also mit unmittelbarem Kontakt zum Prozess. Sie stellt die Schnittstelle zum Prozess dar. Aus leittechnischer Sicht sind hier besonders Sensorik und Aktorik von Interesse, um Informationen über Prozesszustände zu erhalten und auf den Prozess einwirken zu können. Dies wird auch als Instrumentierung bezeichnet.

Steuerungsebene Ziel der Steuerungsebene ist eine autonome Steuerung und Regelung des Prozesses hinsichtlich externer, von der Prozessleitebene bestimmter, Vorgaben. Die darunter liegende Feldebene dient der Steuerungsebene als Mittel, durch welche der Prozess erfasst und beeinflusst werden kann. Typischerweise werden hierzu Speicherprogrammierbare Steuerungen (SPSen), Automatisierungsstationen (ASen) und Prozessstationen (PSen) eingesetzt¹.

Prozessleitebene Sie dient der Prozessführung und Überwachung durch Bedienpersonal, der Ausführung von vorgegebenen Rezepturen und der Erfassung von Produktionsdaten. Computersysteme für diese Aufgabenstellung werden als SCADA-Systeme² bezeichnet.

Betriebsleitebene Zu den Aufgaben der Betriebsleitebene gehören Materialmanagement (wie z. B. Prozess- und Lagerlogistik), Qualitätssicherung, Produktionsdatenerfassung und die Ermittlung von wirtschaftlichen Kennzahlen (wie z. B. Auslastung, Zykluszeit, Ausschussquotient, etc.), aber auch Funktionen wie Rezeptverwaltung und Parameterhandling (Produktionsfeinplanung). Dies wird von sog. Manufacturing Execution Systems (MES) erledigt, welche auf Betriebsebene der ausführende Arm von ERP sind. Diese Software-Pakete werden sowohl als Gesamtpakete als auch als spezialisierte Komponenten angeboten.

Unternehmensleitebene Ihr unterliegt die effiziente Ressourcenallokation von Kapital,

¹ Alle drei Gerätetypen sind Mikrocomputer-basierte Steuerungsgeräte und bezeichnen dieselbe Funktionalität, unterscheiden sich jedoch oft hinsichtlich Leistungsklasse. Meist bezeichnen Automatisierungsstationen (ASen) und Prozessstationen (PSen) leistungsfähige Controller (z. B. Siemens S7-400), während klassische Speicherprogrammierbare Steuerungen (SPSen) etwas leistungsschwächer sind (z. B. Siemens S7-300).

² Supervisory Control and Data Acquisition (SCADA)

Betriebsmittel und Personal. Diese Aufgabe wird auch als Enterprise Resource Planning (ERP) bezeichnet. Dazu wird spezialisierte Software eingesetzt, welche Funktionen für Controlling, Rechnungswesen, Materialmanagement und -logistik etc. zur Verfügung stellen. Im Vergleich zur Betriebsleitebene wird hier langfristiger und abstrahiert von den technischen Details geplant, während MES auch technische Parameter einer lokalen Produktionslinie erfassen. Typische Anbieter von ERP-Software sind z. B. SAP, Oracle, Sage und Infor.

Prozessleitsysteme bezeichnen in diesem Schema ein ebenenübergreifendes System zur Handhabung von Prozessen und implementieren die unteren drei Ebenen der Automatisierungspyramide. Die weiteren Ausführungen werden sich in der Regel auf solche PLSe beziehen.

Während die obersten Ebenen eher dispositiven Charakter haben, wie z. B. die kostenminimale Lenkung von Materialströmen und Warenbeständen unter Einhaltung des Liefertermins, besitzen die darunter liegenden Ebenen zunehmend operativen Charakter. Diese unterschiedlichen Aufgaben führen zu unterschiedlichen Anforderungen hinsichtlich der Reaktionsgeschwindigkeit bzw. Abtastzeit und der anfallenden Datenmenge (siehe Abb. 1.5). So werden beispielsweise auf Feldebene nur wenige Bit verarbeitet, was jedoch innerhalb weniger Millisekunden erfolgen muss. Auf Betriebs- und Unternehmensebene hingegen haben Entscheidung einen wesentlich längeren Zeithorizont, während die benötigten Datenmengen, auch abseits von Wissen und Erfahrung, ganze Datenbank-Server füllen.

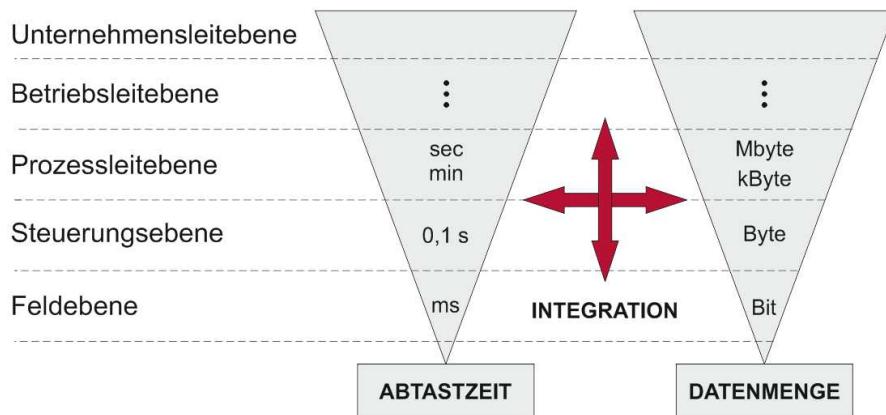


Abbildung 1.5.: Anforderungen an die unterschiedlichen Ebenen der Automatisierungs-pyramide.

Wie oben bereits erwähnt, erfolgt die Datenhaltung und -verarbeitung inzwischen durchgehend mittels Computersystemen. Es liegt daher nahe, unterschiedliche Einzelkompo-

nenten, Software-Tools und Services, von der Feld- bis zu Unternehmensleitebene, in eine durchgängige Automatisierungslösung zu integrieren. Siemens hat dieses Konzept 1996 unter dem Begriff Totally Integrated Automation (TIA) eingeführt.

Prozessleitsysteme besitzen in der Regel eine sehr lange Lebensdauer, da Umbauten zeitintensiv, aufwendig und kostspielig sind. Dementsprechend wird großer Wert auf Abwärtskompatibilität gelegt. Begriffe und Entwicklungen sind entsprechend stark von den eigenen geschichtlichen Veränderungen geprägt, weshalb im Folgenden auf die historische Entwicklung der PLSe eingegangen wird.

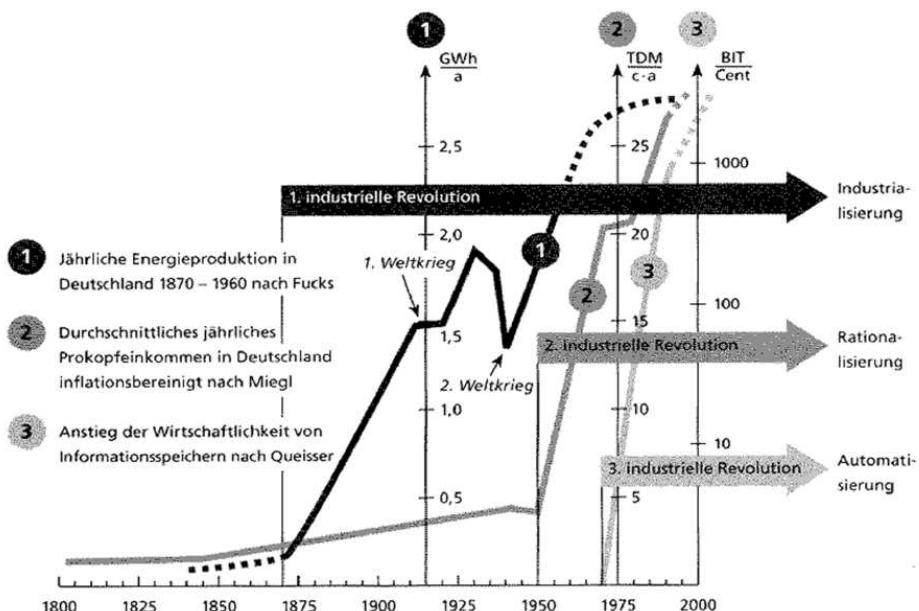


Abbildung 1.6.: Veranschaulichung der drei industriellen Revolutionen anhand der Indikatoren Energiebedarf, Pro-Kopf-Einkommen und Informationsspeicherkosten.

1.1.3. Historische Entwicklung von Prozessleitsystemen

Der Begriff des Prozesses hat sich seit dem 19. Jahrhundert ausgehend von Naturwissenschaften und politischer Philosophie auch im wissenschaftlich-technischen und betriebswirtschaftlichen Denken etabliert. Die hier betrachteten Produktionsprozesse wurden schon damals in ihrer wechselseitigen Abhängigkeit von Mensch und Maschine charak-

terisiert, welche im Zuge der industriellen Revolution einem fundamentalen Wandel unterlag. Dieser Wandel kann als Abfolge dreier Phasen bzw. Revolutionen verstanden werden (siehe Abb. 1.6): das beginnende Maschinenzeitalter mit rapidem Anstieg des Energieverbrauchs und dem Entstehen von Fabriken, die zunehmende Rationalisierung und Massenproduktion³ und die Verwendung von Computersystemen zur Automatisierung der Produktionsanlagen. Das moderne Bild eines autonom und maschinell ablauenden Produktionsprozesses ist ein Produkt fortschreitender Automatisierung, welche im Verlauf des 20. Jahrhunderts zu massiven Veränderungen führte.

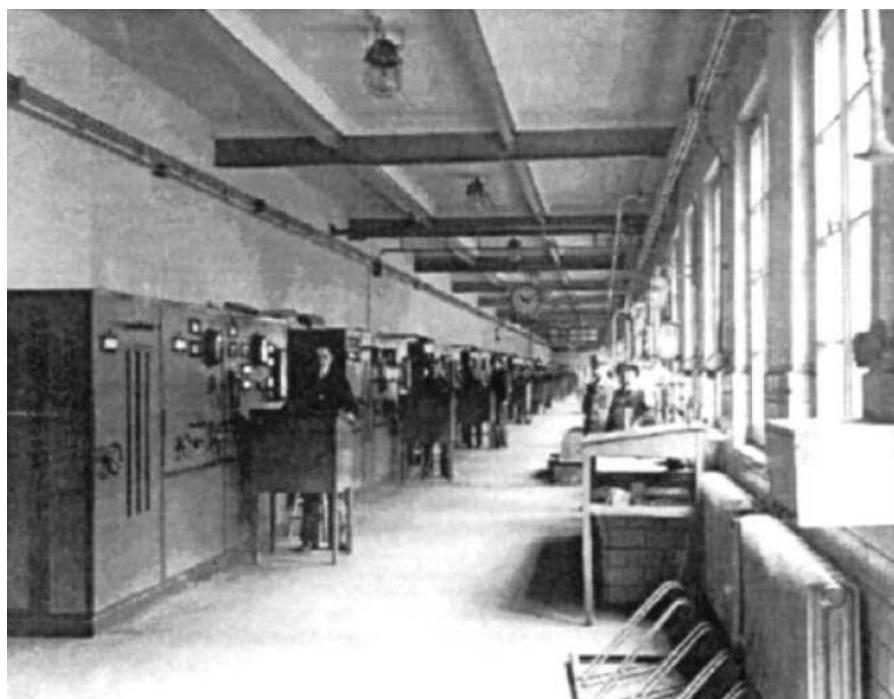


Abbildung 1.7.: Feldleitstand um 1950.

Bezogen auf den Grad der Automatisierung können vier historische Stufen der Prozessleitung unterschieden werden:

Erste Automatisierungsstufe Obwohl in einigen Bereichen bereits einfache mechanische, hydraulische und pneumatische Regler eingesetzt werden und einen automatisierten Eingriff erlauben, erfolgt die Führung des Prozesses überwiegend von Menschen. Sensorik und Aktorik ist bereits an sogenannten Feldleitständen gebündelt verfügbar. Die bedienende Person überwacht von dort aus die Messwerte und

³Der Begriff der zweiten industriellen Revolution ist sowohl an sich umstritten als auch in der zeitlichen Einordnung stark schwankend.

greift manuell in den Prozess ein. Die Menge der kleinen verteilten Feldleitstände führt zu sehr hohem Personalbedarf (siehe Abb. 1.7).

Zweite Automatisierungsstufe Die Weiterentwicklung der Mess- und Regelungstechnik ermöglicht zunehmend komplexe Regelungen. Steuerungen in Relaistechnik können feste sequentielle Abläufe automatisiert abarbeiten, womit einfache Steuerungsaufgaben nicht mehr manuell ausgeführt werden müssen.

Dritte Automatisierungsstufe Bedienung und Beobachtung des Prozesses wird in großen Messwarten zentralisiert. Dadurch sind die meisten Informationen in der Warte verfügbar, die schiere Zahl der Anzeigen und Schalter macht die Übersicht innerhalb dieser Informationsflut jedoch schwierig. Durch riesige R&I-Schaltpläne der Anlage (Mosaiktafeln, siehe Abb. 1.8 rechts) und beginnende Rechnerunterstützung wird versucht, dem entgegenzuwirken. Abb. 1.8 zeigt links eine beispielhafte Leitwarte.

Vierte Automatisierungsstufe Die Verfügbarkeit von erschwinglichen Mikrocomputersystemen ermöglicht es, die Hardware in der Anlage zu verteilen. Das gesamte Prozessleitsystem wird zunehmend digital aufgebaut. Die Visualisierung des Prozesses erfolgt zunehmend über Bildschirme, welche die unflexiblen und teuren Mosaiksysteme teilweise ersetzen. Die Bedienung erfolgt durch Tastatur oder Lichtgriffel wie in Abb. 1.9 dargestellt.

Aktuelle Situation Die Dezentralisierung der Automatisierungssysteme nimmt zu, Funktionalität und Intelligenz verlagert sich von zentralen Leitwarten hin zu SPSen und Feldgeräten. Die Kommunikation zwischen Feldgeräten erfolgt über unterschiedliche Bussysteme und echtzeitfähigem Ethernet. Die eigentlich nicht für industrielle Anwendungen gedachten PCs kehren als universell einsetzbare Industrie-PCs (IPCs) zurück. Die Bedienung dieser Systeme erfolgt über Tastatur und Maus. Zur Visualisierung stehen neben Bildschirmen auch Videowände oder Projektoren zur Verfügung, welche vom PLS vorselektierte Informationen und Alarne anzeigen (siehe Abb. 1.10). So können sich die Anlagenbediener*innen auf die wichtigsten Prozessbilder und Aktivitäten konzentrieren.

1.1.4. Topologien leittechnischer Anlagen

Es zeigten sich im vorherigen Abschnitt zwei aufeinander folgende historische Bewegungen. Anfangs hin zu zentralisierter Organisation, um den Bedarf an Arbeitskräften zu reduzieren bzw. aufgrund der exzessiven Kosten von Computersystemen. Später erlauben die rapide sinkenden Kosten von Mikrocomputersystemen ab den 1980er Jahren eine Dezentralisierung der steuerungstechnischen Komponenten. Die zweite Bewegung ist in

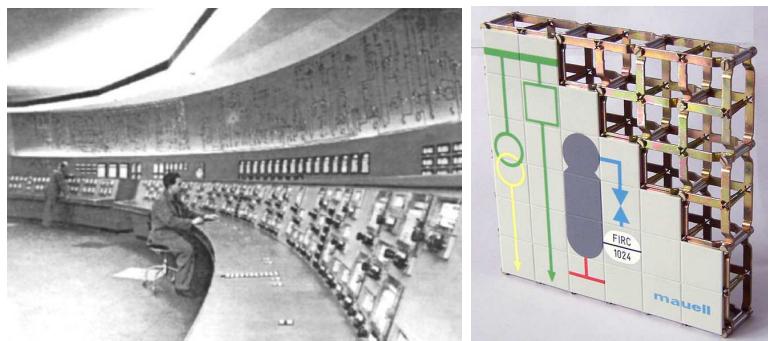


Abbildung 1.8.: Zentrale Leitwarte (links) und Ausschnitt einer Mosaiktafel zur Visualisierung (rechts).

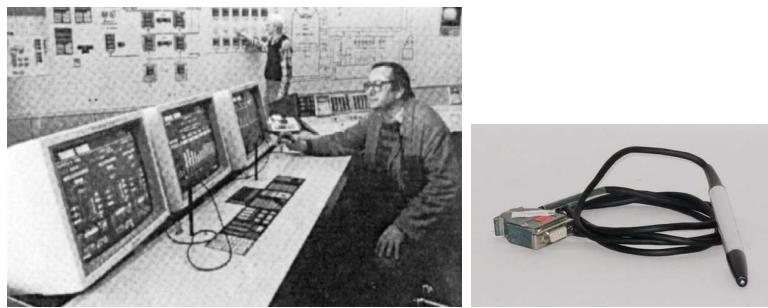


Abbildung 1.9.: Prozessführung mit Bildschirmen.



Abbildung 1.10.: Moderne Leitwarte.

Abb. 1.11 hinsichtlich Hard- und Software dargestellt.

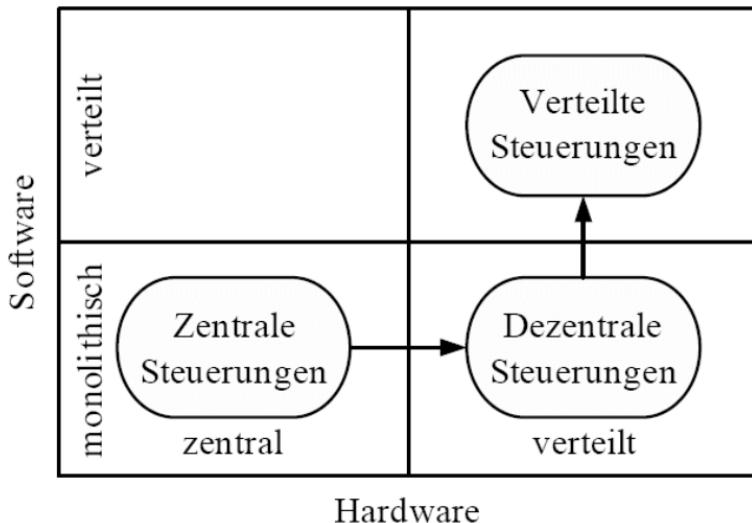


Abbildung 1.11.: Einordnung der Begriffe zentral, dezentral und verteilt. Die Pfeile symbolisieren die anhaltende historische Bewegung hin zu verteilten Systemen.

Zentrale Steuerungen sind durch zentrale Hardware und monolithische Software gekennzeichnet, wie es für Anlagen der dritten Automatisierungsstufe kennzeichnend war. Durch Verteilung der Hardware im Feld erhält man Dezentrale Steuerungen. Die verteilten Komponenten sind dabei zwar durchaus mit Mikrocomputern ausgestattet, beinhalten jedoch keinen Teil der Steuerungsapplikation⁴. Diese ist noch immer monolithisch in einem Prozessrechner ausgeführt. Verteilte Steuerungen hingegen sind auch softwareseitig in Teilsysteme und Komponenten strukturiert, welche auf den verteilten Hardwarekomponenten laufen.

Diese Veränderungen werden auch in den verwendeten Topologien ersichtlich, welche in Abb. 1.12 visualisiert sind. In Einzelgerätetechnik ausgeführte Systeme enthalten einzelne nicht vernetzte Teilsysteme. Darunter fallen Anlagen der ersten und zweiten Automatisierungsstufe mit einzelnen Messgeräten und Ventilen bzw. einfachen Regelkreisen. Bei Verwendung eines zentralen Prozessrechners werden alle Signale von Sensoren und Aktoren zentral mit einem Rechner in der Leitwarte erfasst und gesteuert, das heißt der steuerungstechnische Teil des PLS ist als zentrale Steuerung ausgeführt. Diese Topo-

⁴Beispielsweise sei die Aufnahme von Messwerten genannt. Die IO-Einheiten im Feld übernehmen Messwertaufnahme und -verarbeitung, aber keine Steuerungsaufgaben.

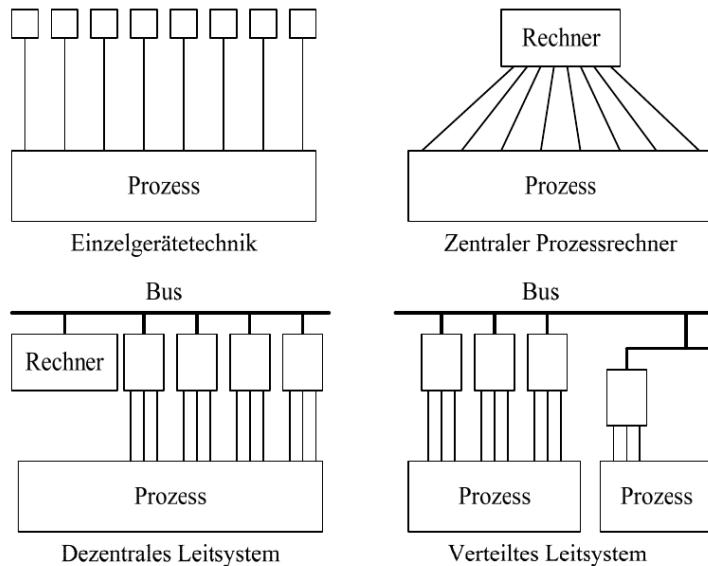


Abbildung 1.12.: Unterschiedliche PLS-Topologien.

logie kennzeichnet die dritte Automatisierungsstufe. Analog zum zentralen Leitsystem ist ein dezentrales, respektive verteiltes Leitsystem durch eine dezentrale bzw. verteilte Steuerung gekennzeichnet. Dezentrale Leitsysteme sind maßgebend für die vierte Automatisierungsstufe, während verteilte Leitsysteme aktuell erst bezogen auf Teilanlagen erprobt werden und Gegenstand der Forschung sind.

1.1.5. Entwicklungstrends

Während für große Anlagen typischerweise komplexe Prozessleitsysteme mit einer Vielzahl an verteilten Prozessnahen Komponenten (PNKs), je nach Hersteller auch als Automatisierungsstationen oder Prozessstationen bezeichnet, verwendet wurden, dominierten SPSEN – kleine, autonome Systeme ohne übergeordnetes Leitsystem. Inzwischen bieten PLSe durch kleine IPCs zunehmend kostengünstige Stand-Alone-Lösungen (vgl. PCS 7 BOX in Abschnitt 1.3), während SPSEN ihre Regelungsfähigkeiten verbessern, sich mit anderen SPSEN und Rechnern koppeln lassen und immer höhere Steuerungsfähigkeiten bieten. Diese Entwicklungen lassen beiden Bereiche, wie in Abb. 1.13 dargestellt, zunehmend verschmelzen.

Moderne kommerzielle Prozessleitsysteme verwenden dezentrale Architekturen und wer-

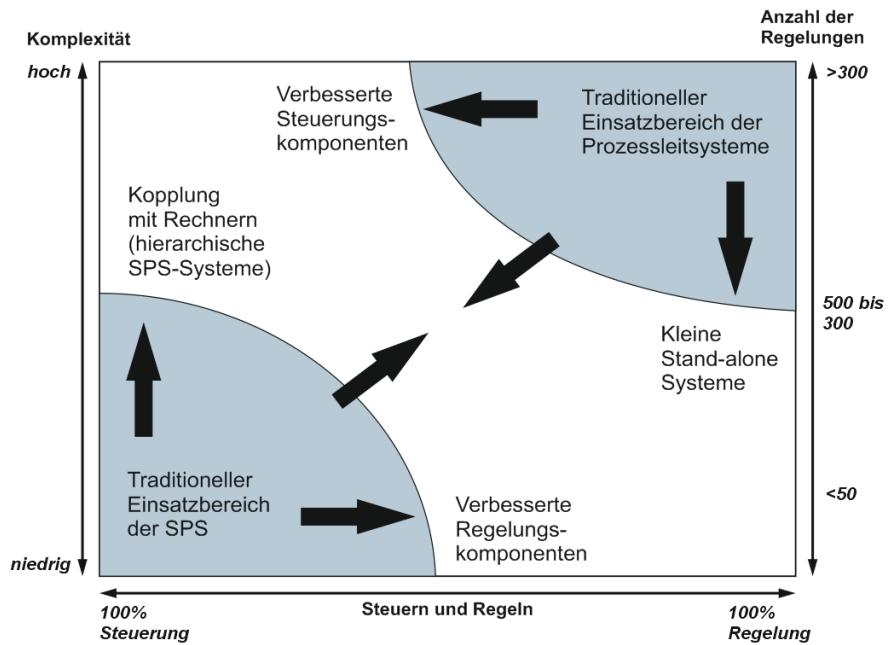


Abbildung 1.13.: Entwicklungstendenzen von SPSEN und PLSEn.

den von Herstellern als komplette Systemlösungen angeboten. Diese vorkonfigurierten Systeme enthalten ein umfangreiches Toolset – beispielsweise Wegesteuerungssoftware für komplexe Feststofftransporte, fertige Archivierungslösungen, Security-Konzepte zur Sicherung der Anlage und systemweite Entwicklungs- und Engineeringsoftware – womit der Kunde die Anwendung spezifiziert. Beispielhaft sei der Simatic PDM (Process Device Manager) genannt, durch welchen über 1200 Geräte verschiedenster Hersteller innerhalb eines PCS7-Systems einheitlich konfiguriert und serviciert werden können.

1.2. Komponenten der Leittechnik

Leittechnische Systeme und die Vielzahl der verwendeten Komponenten (siehe Abb. 1.14) werden üblicherweise je nach Nähe zum zugrundeliegenden Prozess unterschiedlichen Klassen zugerechnet. In einer groben Klassifikation können dabei vier Arten von Komponenten unterschieden werden.

Prozessnahe Komponenten (PNKs) sind Komponenten, welche direkte Schnittstellen zu im Feld verwendeten Sensoren und Aktuatoren besitzen. Darunter fallen sowohl klassische SPSEN, ASes und PSen als auch intelligente Sensoren und Aktoren, wel-

che ihre Teilaufgaben autonom handhaben.

Prozessferne Komponenten (PfKs) nutzen die von prozessnahen Komponenten zur Verfügung gestellte, abstrahierte Ebene, um globale Prozesssteuerung, -visualisierung und -protokollierung zu implementieren. Je nach Literatur werden sie auch vermehrt als Benutzernahe Komponenten (BNKs) bezeichnet.

Kommunikationssysteme verbinden Komponenten innerhalb eines Leitsystems. Neben direkten Einzelverdrahtungen verwenden moderne Prozessleitsysteme zur Steigerung der Flexibilität überwiegend Bustopologien. Je nach Anforderung wird hierbei zwischen Systembussen und Anlagen- oder Feldbussen unterschieden.

Gateways bieten die Anbindung an ein unternehmensinternes Intranet oder das Internet. Die Trennung von Netzbereichen ist schon aus Gründen der Skalierbarkeit und Sicherheit gegeben.

Die Komponenten können den unterschiedlichen Ebenen der Automatisierungspyramide (siehe Abschnitt 1.1.2) zugeordnet werden. Solche Klassifizierungen sind jedoch selten eindeutig und frei von Überschneidungen. Zudem verschwimmen die Grenzen, da etwa SPSEN zunehmend höhere Funktionalitäten zur Verfügung stellen. Sie bilden dennoch eine hilfreiche Abstraktion zur Orientierung innerhalb der in der Praxis vorhandenen Vielfalt.

1.2.1. Prozessleitebene

Innerhalb der Prozessleitebene treten überwiegend PfKs auf. Gemäß den Aufgaben und Funktionen der Prozessleitebene (siehe Abschnitt 1.1.1) sind hier unterschiedliche Gerätklassen tätig. So soll der gesamte Produktionsprozess protokolliert und dokumentiert werden, um später Analysen oder Fehlerbehebungen zu ermöglichen. Weiters muss der Prozess selbst visualisiert werden. Bei Anlagen mit tausenden von Sensoren und Aktoren ist es durchaus herausfordernd, diese Daten dem Bedienpersonal übersichtlich zu präsentieren und die globale Steuerung des Prozesses zu ermöglichen. So sollten z. B. Trend-, Produktions- und Kapazitätsanalysen möglich sein, oder, je nach Anlage, eine Rezeptverwaltung samt Versionierung im System integriert sein, um eine Charge jederzeit rückverfolgen zu können. Letzteres wird in vielen Produktionsprozessen der Pharma-, Lebensmittel- oder auch Fahrzeugindustrie gesetzlich gefordert, um für eine rekonstruierbare Lieferkette zu sorgen. Um die Anlage warten und verändern zu können, sind spezielle Engineering Stationen (ESen) vorgesehen. Beispiele für Komponenten der Prozessleitebene sind in Abb. 1.14 dargestellt.

Operator-Stationen (OSen) stellen eine der prominentesten Komponenten der Prozesslei-

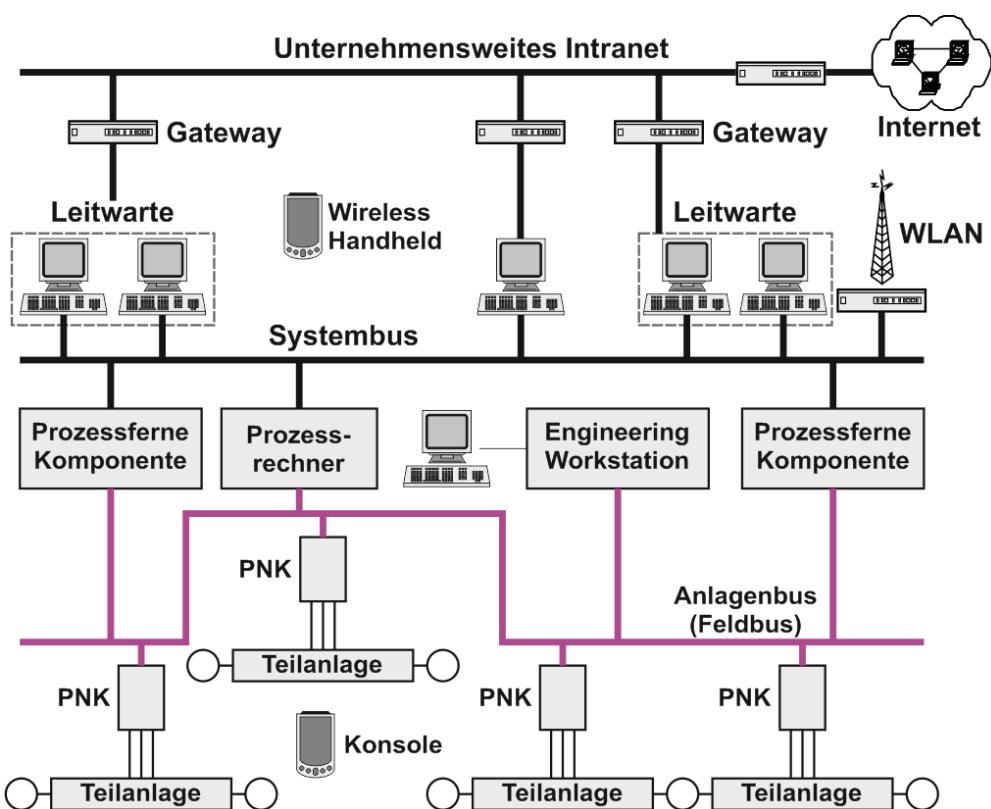


Abbildung 1.14.: Komponenten der Prozessleitebene in einer Client-Server-Architektur mit zwei getrennten Bussystemen.

tebene dar. Sie dienen der Visualisierung von Prozessgrößen sowie der Bedienung des Prozessleitsystems (PLS). Synonym mit OS werden auch die Begriffe Bedien- und Beobachtungsstation (BuB) oder Anzeige- und Bedienkomponente (ABK) verwendet. Abb. 1.15 zeigt mehrere beispielhafte OSen. Durch die Verwendung eines einheitlichen Informationsmodells (vgl. „Systeme 2. Generation“) und aktuellen Computersystemen können OSen hinsichtlich Anlagengröße und Kundenanforderungen einfach adaptiert werden. Während z. B. im Bild der klassischen Leitwarte eines Atomkraftwerks OSen zentral an einem Ort, der Leitwarte, durch spezialisierte Hardware realisiert wurden, können damit einfach OSen und insbesondere sog. Mehrplatzsysteme geschaffen werden. So können OSen sowohl örtlich verteilt als auch mit kontextualisierten Oberflächen betrieben werden.

Dabei stellt sich die Frage, wie Informationen im Prozessleitsystem zur Verfügung gestellt werden. Obwohl es eine Vielzahl an Architekturen gibt, wird meist auf eine Client-Server- oder Einzelbusarchitektur zurückgegriffen.

In Client-Server-Architekturen erstellen Server ein Abbild der Steuerungsebene und erzielen Befehle an diese. Dazu werden von den Servern zyklisch die benötigten Daten aller PNPs am Anlagenbus abgefragt und den Clients am Systembus zur Verfügung gestellt. Dadurch skaliert bei großen Systemen die Buslast besser als bei Einzelbussystemen und den Anwendungen der über-gelagerten Ebenen wird eine einheitliche Schnittstelle zur Verfügung gestellt. Da bei einem Serverausfall die Anlage nicht mehr bedienbar ist, werden redundante Systeme verwendet, was Hardwareaufwand und Kosten erhöht. Ein prominentes Beispiel für eine Client-Server-Architektur ist das SIMATIC PCS 7 Prozessleitsystem von Siemens (siehe Abb. 1.16).

Bei Einzelbus-Architekturen hängen sowohl PNPs als auch PFKs am selben Bus. Damit ist eine Kommunikation aller Busteilnehmer untereinander möglich. So ist im Vergleich zur Server-Client-Architektur eine höhere Verfügbarkeit und eine bessere Verteilung der Intelligenz möglich, doch führt dies zu hohen Busauslastungen in großen Anlagen. Zudem ist hier kein zentrales Abbild des Prozesses verfügbar, was die Anbindung externer Systeme in den übergeordneten Ebenen erschwert. ABBs Freelance Prozessleitsystem verwendet beispielsweise diese Architektur.

Analog zu OSen werden zur Wartung und Konfiguration eines Leitsystems sog. ESen verwendet. Sie erlauben es, anlagenweites Engineering an jeder beliebigen ES durchzuführen, ohne sich physisch zu den unterschiedlichen Komponenten zu begeben. Darunter fällt unter anderem die Programmierung und Konfiguration von PNPs und Kommunikationsnetzwerken, und das Design von OSen und deren Benutzeroberflächen.

Sowohl OSen als auch ESen mit spezialisierter Hardware werden wegen der geringeren Kosten als auch der erhöhten Flexibilität von IPCs abgelöst. Die Funktion von OSen und ESen können dadurch oft auf ein- und demselben PC realisiert werden.



Abbildung 1.15.: Bild einer Leitwarte mit mehreren Operator-Stationen (links) sowie einer grafischen Bedien- und Beobachtungssoberfläche (rechts).

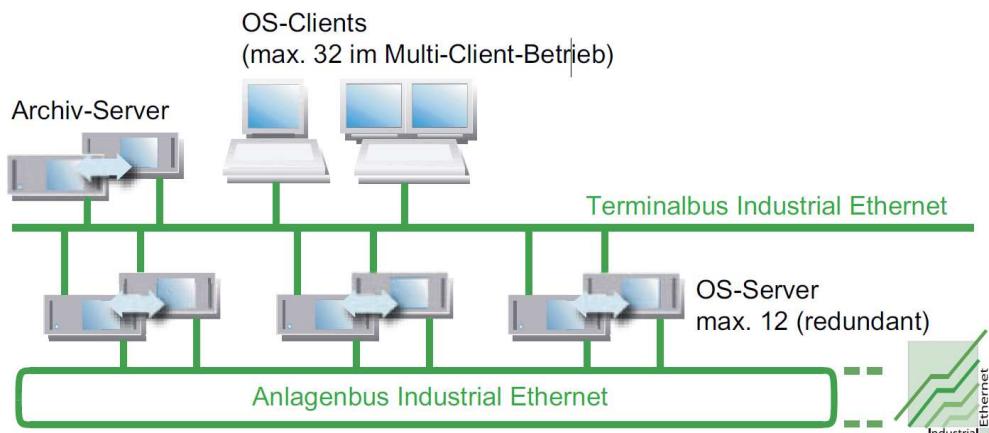


Abbildung 1.16.: Client-Server-Architektur für OSen

1.2.2. Steuerungsebene

Während innerhalb der Prozesseleitebene größtenteils PFKs auftreten, begegnet man innerhalb der Steuerungsebene vornehmlich PNKs. Sie übernehmen die prozessnahe Messsignalaunahme, Steuerung und Regelung und sind oft Anlagenteilen zugeordnet. Die dabei vorgenommene Aufteilung erfolgt nach Maßgabe des zu kontrollierenden Prozesses. Diese einzelnen Anlagenteile werden durch die jeweilige PNK autark und dezentral gehandhabt und derart beeinflusst, dass die Vorgaben der Prozesseleitebene möglichst genau eingehalten werden. Gleichzeitig werden der Prozesseleitebene Prozess- und Statusinformationen (wie z. B. Messdaten oder verarbeitete Kenngrößen des Prozesses) mitgeteilt.

Klassischer Aufbau einer prozessnahen Komponente

Der prinzipielle Aufbau einer PNK ist in Abb. 1.17 dargestellt. Die Hardware-Architektur einer PNK unterscheidet sich prinzipiell nicht von der einer SPS. Die PNK-Zentraleinheit ist dabei im Wesentlichen ein Mikrocomputer-System. Die zusätzliche „Glue Logic“ übernimmt die Anbindung ansonsten inkompatibler Komponenten (z. B. anders getaktete Komponenten). Zur Verbindung mit der Außenwelt werden Busschnittstellen und sog. Prozess-Ein-/Ausgabekomponenten (PEAKs) verwendet, welche meist über einen eigenen Prozessor verfügen. Die PEAK kümmern sich autonom um Messsignalbereitung und A/D- bzw. D/A-Wandlung.

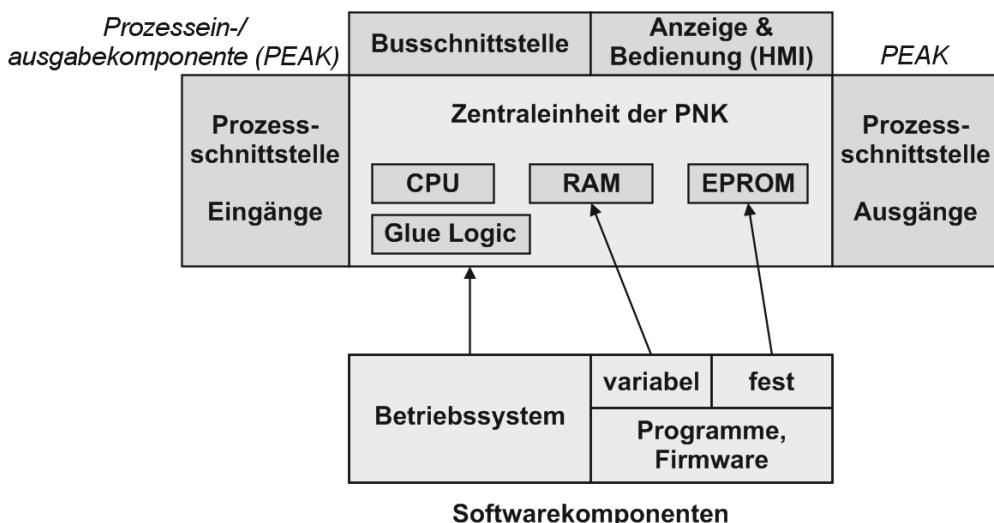


Abbildung 1.17.: Prinzipieller Aufbau einer PNK.

Die Softwarekomponente einer PNK besteht aus einem optimierten Echtzeitbetriebssystem, welches meist vom Hersteller vorgegeben ist. Die Gestaltung der Funktionalität wird dabei durch vorgegebene Funktionsbausteine bewältigt. Einige typische Funktionsgruppen und deren Bausteine sind:

- Mathematische Operationen (Grundrechnungsarten, Vergleichsoperatoren, elementare Funktionen und Integrations- und Differentiationsblöcke)
- Signalanpassung (digitale Filter)
- Steuerung (Logikblöcke wie UND, ODER, NAND, etc., Ablaufsteuerungen)
- Regelung (z. B. PD, PI, PID)
- Überwachung (Grenzwertüberwachung, Status- und Fehlermeldungen)

Vergleich von prozessnahen Komponenten mit klassischen Speicherprogrammierbaren Steuerungen

Die obige Beschreibung von PNKs könnte eins zu eins auf klassische SPSen angewandt werden. Warum werden also unterschiedliche Begrifflichkeiten für dieselbe Sache eingeführt? PNKs sind üblicherweise für den sehr rauen industriellen Betrieb in der Verfahrenstechnik ausgelegt. So erfüllen sie hohe Anforderungen hinsichtlich Betriebstemperaturen, Staub- und Feuchtigkeitsschutz, Vibrationsfestigkeit und Säure- und Korrosionsbeständigkeit von Gehäuse und Anschlusskontakten. Zusätzlich sind Ausführungen mit Explosionsschutz erhältlich. Außerdem sind die elektrischen Betriebs- und Grenzwerte in der Regel besser als bei SPSen. Historisch sind PNKs im Kontext von Prozessleitsystemen entstanden, während SPSen eher zur autonomen Automatisierung kleiner Anlagenteile der Fertigungsindustrie dienten. Trotzdem ist die Grenze zwischen PNKs und SPSen unscharf und wird von aktuellen Entwicklungstrends weiter verwischt (siehe Abschnitt 1.1.5).

1.2.3. Feldebene

Komponenten auf dieser untersten Ebene der Automatisierungspyramide interagieren direkt mit dem Prozess via Sensoren und Aktuatoren.

Ein besonders wichtiger Teilaспект von Automatisierungs- und Steuerungssystemen ist deren Zuverlässigkeit bzw. Versagenswahrscheinlichkeit. Gerade bei der Verwendung elektrischer und computerbasierter Geräte zur Implementierung von Sicherheitsfunk-

tionen ist diese essenziell. Die funktionale Sicherheit definiert dafür sog. Sicherheits-Integritätslevels (SILs), welche je nach Anforderungen und Relevanz der Sicherheitsfunktion unterschiedliche Wertebereiche für die zulässige mittlere Versagenswahrscheinlichkeit festlegen (siehe Abb. 1.18). Es zeigt sich, dass die Versagenswahrscheinlichkeit einer Sicherheitsfunktion zum überwiegenden Teil durch die Zuverlässigkeit von Sensorik und Aktorik bestimmt sind.

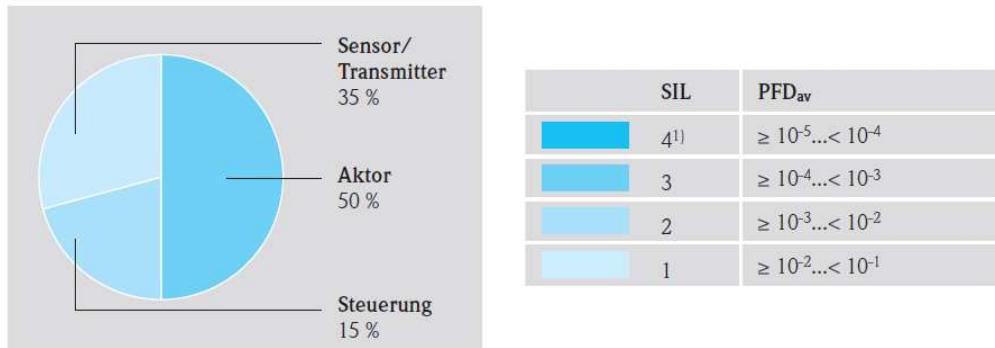


Abbildung 1.18.: Links: Verteilung der Versagenswahrscheinlichkeit einer Sicherheitsfunktion auf die Bereiche Sensorik (inkl. Transmitter), Steuerung und Aktorik. Rechts: Zulässige mittlere Versagenswahrscheinlichkeit PFD_{av} (average probability of failure on demand) je nach SIL (aus [1]).

Im Folgenden sollen einige Sensoren und Aktoren exemplarisch betrachtet werden. Insbesondere werden Bauteile vorgestellt, welche in den einschlägigen Vorlesungen nicht oder kaum dargestellt wurden. Auf pneumatische Aktoren wird später gesondert eingegangen.

Füllstandsmessung

Zur Erfassung des Füllstands von Flüssigkeiten oder Schüttgütern in einem Behältnis gibt es eine Vielzahl von Methoden. Grundsätzlich wird dabei zwischen Grenzschaltern, welche die binären Zustände „Referenzlevel überschritten“ bzw. „Referenzlevel unterschritten“ annehmen, und kontinuierlichen Füllstandssensoren, welche den aktuellen Füllstand als analogen oder digitalen Messwert ausgeben, unterscheiden. Die Messwerte kontinuierlicher Sensoren werden je nach Typ als einheitenlose Prozentangabe des Gesamtbehältnisses oder in Längen-, Volumen- oder Masseneinheiten ausgegeben. Die Wahl eines Messprinzips hängt neben dem Kostenfaktor auch von der konkreten Einsatzumgebung wie Verschmutzungen, Ablagerungen, Neigung zur Schaumbildung und den Eigenschaften des Inhalts ab.

Einige typische Methoden zur Messung wären:

Schwimmer stellen eine der ältesten Messprinzipien dar. Ein Schwimmkörper mit geringerer Dichte als die zu messende Flüssigkeit schwimmt auf dieser. Die Füllstandsmessung wird dadurch auf eine Längenmessung abgebildet. Oft wird der Schwimmkörper in einem separaten Rohr geführt, um die laterale Position festzulegen. Je nach verwendeter Längenmessung kann das Prinzip als kontinuierlicher Sensor oder Grenzschalter (z. B. mittels Reedschalter) verwendet werden.

Drehflügelschalter dienen als reine Grenzwertschalter für Schüttgut. Ein sich frei drehender Flügel an der gewünschten Referenzposition wird dabei durch das Schüttgut blockiert und betätigt damit einen Schalter. Dieses Messprinzip ist relativ robust gegen Staub und Ablagerungen. Abb. 1.20 zeigt einen Drehflügelschalter der Firma UWT.

Kontinuierliche Kapazitive Sensoren verwenden zwei von oben in die Flüssigkeit ragende Elektroden. Durch die Dielektrizitätszahl ε der Flüssigkeit hängt die Kapazität der Anordnung von der Füllhöhe ab. In Abb. 1.19 ist ein Sensor der Firma Endress+Hauser samt Funktionsskizze dargestellt. Probleme können durch schäumende Flüssigkeiten und Ablagerungen an den Elektroden entstehen. Letztere werden durch gleichspannungsbasierte Kapazitätsmessungen verschlimmert, da geladene Partikel von den Elektroden angezogen werden.

Kapazitive Füllstandsgeber detektieren die Änderung der Dielektrizitätszahl ε an einer definierten Behälterhöhe. Bei geeigneten Materialien kann die Dielektrizitätszahl durch die Behälterwände hindurch detektiert werden, wodurch der Schalter nicht der Flüssigkeit ausgesetzt wird. Wie schon beim kontinuierlichen kapazitiven Messprinzip können durch Gleichfelder Ablagerungen im Messbereich die Messung beeinflussen.

Ultraschallsensoren basieren auf der Laufzeitmessung eines akustischen Impulses. Durch die Laufzeit kann der Abstand der Oberfläche des Inhalts zum Sensor ermittelt werden, woraus sich die Füllhöhe ergibt.

Durchflusssensoren

Besonders in verfahrenstechnischen Anlagen der chemischen und biologischen Industrie ist die Erfassung von Zu- und Abflüssen in Prozessen notwendig. Auch hier existiert eine Vielzahl an Messprinzipien: von Ovalradzählern über optischen Verfahren mittels Schwebstoffen hin zu kalorimetrischen Methoden. Viele dieser Verfahren sind Gegenstand der Vorlesung „Sensorik und Sensorsysteme“ des Bachelorstudiums. Zwei besonders wichtige Verfahren sollen hier nochmals kurz umrissen werden.

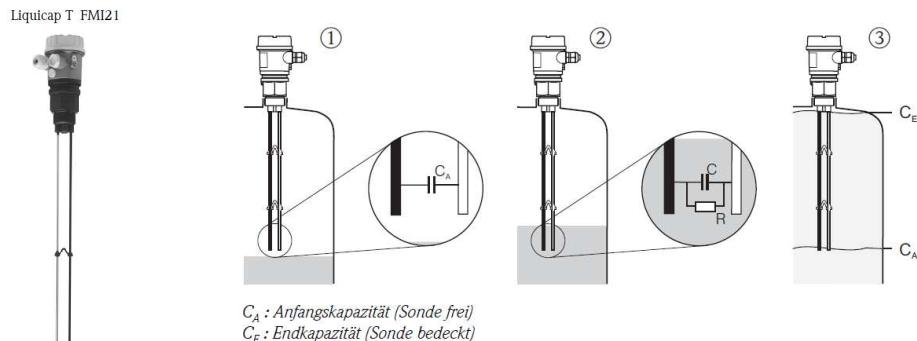


Abbildung 1.19.: Kontinuierlicher Kapazitiver Sensor Liquicap T FM121 von Endress+Hauser. Zwischen den Elektroden befinden sich Abstandshalter, um die Kapazitätsänderung durch Verformungen gering zu halten.



Abbildung 1.20.: Drehflügelschalter RN 3000 der Firma UWT. Ein drehbar gelagerter Synchronmotor treibt den Drehflügel an. Wird dieser blockiert, schaltet das resultierende Moment einen Schalter.

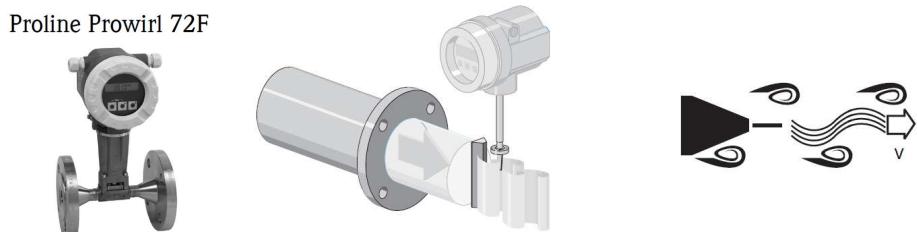


Abbildung 1.21.: Wirbeldurchflussmesser Prowirl 72F von Endress+Hauser mit Prinzipskizze. Die mittlere Grafik zeigt den prinzipiellen Aufbau. Rechts ist die Skizze einer Kármánschen Wirbelstraße zu sehen.

Wirbeldurchflussmesser verwenden die Wirbelfrequenz der sog. Kármánschen Wirbelstraße an einem Störkörper im Flussgebiet. Diese entsteht bei ausreichend großen Reynolds-Zahlen durch Ablösen der Strömung vom Störkörper, woraus sich gegenseitig Wirbel bilden (siehe rechtes Bild von Abb. 1.21). Zwischen der Frequenz der vorbeilaufenden Wirbel f und der Strömungsgeschwindigkeit v besteht der Zusammenhang

$$f = S_r \frac{v}{d},$$

wobei d die sog. charakteristische Länge des Störkörpers darstellt. Der Faktor S_r bezeichnet die Strouhal-Zahl, welche von Geometrie und Reynolds-Zahl des Systems abhängt. Die Wirbel hinter dem Störkörper führen zu lokalen Druckdifferenzen, welche durch unterschiedliche Methoden gemessen werden können. So kann beispielsweise eine Flosse in die Strömung eingebracht werden, welche durch die Druckdifferenzen mit der Wirbelfrequenz zu schwingen anfängt. Durch Messung der Wirbelfrequenz kann, bei bekannter Geometrie und Materialparametern, auf die Strömungsgeschwindigkeit geschlossen werden. Sollte die Reynoldszahl zu gering sein um Wirbel zu bilden, kann sie durch Verringern des Rohrdurchmessers erhöht werden. So ist es möglich, eine große Bandbreite von Fluiden und Strömungsgeschwindigkeiten zu messen.

Differenzdruckmesser erfassen die Druckdifferenz an einer in die Strömung eingebrachten Blende (siehe Abb. 1.22). Aus der Massenerhaltung folgt, dass die Strömungsgeschwindigkeit an der Blende größer als im restlichen Rohr sein muss (d.h. $v_2 > v_1$), weshalb dort laut Satz von Bernoulli der statische Druck abfällt. Für die an der Blende entstehende Druckdifferenz Δp gilt der Zusammenhang

$$q = \frac{\epsilon C}{\sqrt{1 - \beta^4}} A \sqrt{2 \rho \Delta p} \quad (1.1)$$

mit der Blendenfläche $A = \frac{d^2 \pi}{4}$, der Massendichte ρ , der Expansionszahl kompressibler Medien ϵ und dem von der Reynolds-Zahl abhängigen Durchflusskoeffizienten C . Durch Messung der Druckdifferenz Δp kann auf den Durchfluss q geschlossen werden. Wie im Druckdiagramm in Abb. 1.22 gezeigt, steigt der Druck in einiger Entfernung von der Drossel wieder an; die verbleibende Druckdifferenz kommt durch Umwandlung von kinetischer in thermische Energie in Zuge der turbulenten Expansion ⁵.

⁵Bei scharfkantigen Drosseln tritt eine Einschnürung hinter der Blende auf, welche als *vena contracta* bezeichnet wird, wobei weiterhin der Bernoullische Satz angewandt werden kann. Die Beschreibung der turbulenten Expansion danach ist hingegen schwieriger. Blenden zur Durchflussmessung sind nach ISO 5167 genormt.

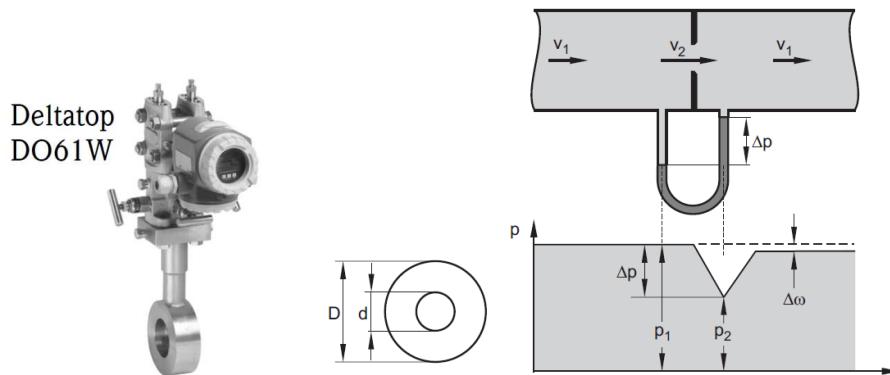


Abbildung 1.22.: Differenzdrucksensor Deltatop DO61W von Endress+Hauser mit Prinzipskizze. Durch die mechanische Verengung entsteht eine Druckdifferenz. Für den Durchfluss gilt $q \propto \sqrt{\Delta p}$.

Ventile

In verfahrenstechnischen Anlagen werden Ventile eingesetzt, um auf den Prozess einwirken zu können. Ein Blick in die DIN ISO 1219, welche die Symbole der Fluidtechnik normiert, zeigt die Vielfalt an Typen und Aufgabengebieten. Dabei kann je nach Perspektive anhand unterschiedlichster Kriterien wie mechanischer Bauform, Wirkungsweise, Betätigungsart oder schlicht Aufgabengebiet unterschieden werden. Beispielsweise seien hier Bezeichnungen wie Kugel-, Proportional-, Hand- oder Rückschlagventil genannt, deren Namen schon auf die unterschiedlichen Aspekte hinweisen. Besonders wichtig für unsere Belange ist dabei eine funktionale Einordnung, also Anzahl der Schaltstellungen und Anschlüsse (Wegeventile) sowie die Art der Wirkung (diskret vs. kontinuierlich). Während für Steuerungsaufgaben oft diskrete Wegeventile genügen, muss in Regelungsanwendungen in der Regel auf kontinuierlich wirkende Proportional- bzw. Servoventile zurückgegriffen werden.

Für Wegeventile (egal ob diskret oder kontinuierlich wirkend) wird folgendes Benennungsschema verwendet: Ein m/n -Wegeventil besitzt m Fluidanschlüsse und n Schaltstellungen. In Abb. 1.23 sind drei einfache Wegeventile nach DIN ISO 1219 dargestellt, wobei Betätigungsart bzw. Wirkung nicht eingezeichnet wurden. Ein konkretes Beispiel für ein kontinuierliches 2/2-Wegeventil, diese Kombination wird auch als Stellventil bezeichnet, inklusive pneumatischer Betätigung ist in Abb. 1.24 dargestellt. Das gezeigte Ventil hat zwei Mediumanschlüsse (Einlass A und Auslass B) und baulich bedingt zwei Schaltstellungen, nämlich Ventil vollständig geschlossen (Hub = 0%) und vollständig geöffnet (Hub = 100%). Durch Einsatz eines pneumatischen Stellungsreglers kann das Ventil jedoch kontinuierlich betrieben und somit beliebige Volumenströme durch das

Ventil geregelt werden. Solche Stellventile werden in der Verfahrenstechnik verwendet, um Fluid- oder Gasströme durch Veränderung des Widerstandes innerhalb der Rohrleitung regeln zu können. Dieser Vorgang kann analog zu elektrischen Schaltkreisen verstanden werden: durch Einbringen eines Widerstandes (hier eine Blende) verringert sich bei gleicher Potenzialdifferenz (hier Druckdifferenz) der fließende elektrische Strom (hier Volumenstrom)⁶.

Eine wichtige Kenngröße für die Dimensionierung von Ventilen ist der sog. K_v -Wert. Er beschreibt den möglichen Durchsatz eines Ventils und besitzt die Einheit m^3/h . Der K_v -Wert entspricht dabei dem Durchfluss von Wasser in m^3/h mit der Temperatur von 5 °C–30 °C bei einer Druckdifferenz von 0,98 bar. Dieser Wert ist abhängig vom Öffnungsgrad des Ventils, weshalb zur Beschreibung eines Ventils der K_v -Wert bei maximaler Öffnung K_{vs} sowie die Abhängigkeit vom Ventilhub H verwendet werden.

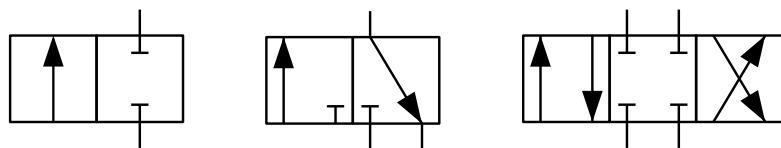


Abbildung 1.23.: Unterschiedliche diskrete Wegeventile als Schaltsymbol nach DIN ISO 1219. Von links nach rechts: 2/2-Wegeventil, 3/2-Wegeventil und 4/3-Wegeventil mit sperrender Mittelstellung.

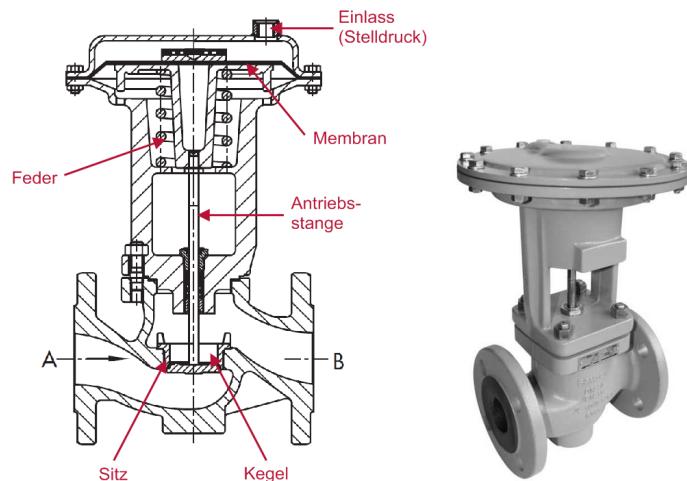


Abbildung 1.24.: Stellventil der Firma Samson mit Schnittzeichnung.

⁶Diese Äquivalenzrelation kann auf verschiedenste physikalische und chemische Domänen ausgeweitet werden, wobei dabei nur noch von sog. Flow- und Effortvariablen gesprochen wird.

Stellventile werden meist mit linearer oder gleichprozentiger Kennlinie angeboten. Eine lineare Kennlinie bedeutet dabei für den Hub H , dass

$$\frac{\Delta K_v}{\Delta H} = \text{const}$$

gilt. Bei der praktischen Realisierung kann dies nicht über den gesamten Arbeitsbereich erfüllt werden (vgl. Abb. 1.25). Etwas weniger intuitiv ist die gleichprozentige Kennlinie. Hier gilt idealisiert entlang der gesamten Kennlinie die Beziehung

$$\frac{\frac{\Delta K_v}{K_{vs}}}{\frac{\Delta H}{H_{100}}} = \text{const}$$

mit dem maximalen Hub H_{100} . Damit ist gesichert, dass für jede beliebige Ventilstellung eine gleiche Hubänderung zu einer prozentual gleichen Änderung des K_v -Wertes führt. Wie schon bei der linearen Kennlinie ergeben sich auch hier im Bereich kleiner Ventilöffnungen Abweichungen von der idealen Charakteristik, wie in Abb. 1.26 dargestellt.

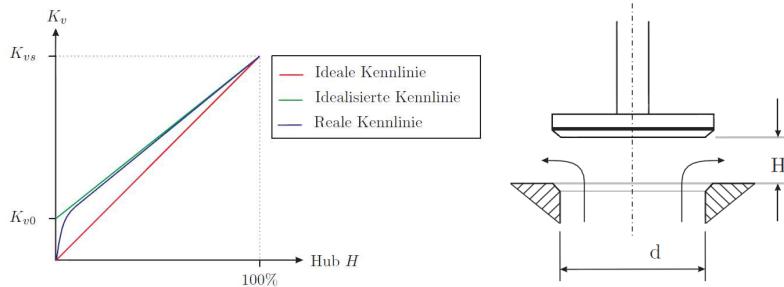


Abbildung 1.25.: Ideale, idealisierte und reale lineare Ventilkennlinie und der dabei verwendete Ventilkegel.

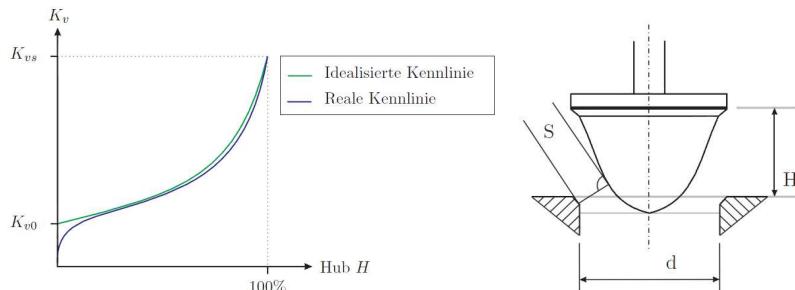


Abbildung 1.26.: Ideale und reale gleichprozentige Ventilkennlinie und der dabei verwendete Ventilkegel.

Abschließend soll hier noch auf einen unerwünschten Effekt bei Stellventilen und allgemein Fluidsystemen eingegangen werden, nämlich das Problem der Kavitation. Unter

Kavitation versteht man die spontane Bildung und Implosion von Dampfbläschen in Flüssigkeiten. Diese entstehen bei Strömungen, wenn der lokale hydrostatische Druck unter den Dampfdruck des Mediums absinkt (z. B. bei gebogenen Rohrleitungen mit hoher Strömungsgeschwindigkeit). Durch die hohen lokalen Druckschwankungen entstehen mechanische Schwingungen und Schall sowie die als Kavitationsfraß bezeichnete mechanische Erosion fester Oberflächen (siehe Abb. 1.27). Nach dem Satz von Bernoulli erhöht sich lokal an Verengungen die Strömungsgeschwindigkeit, während der hydrostatische Druck abnimmt. Dadurch ist vornehmlich der Ventilkegel dem Verschleiß durch Kavitation ausgesetzt.



Abbildung 1.27.: Detailaufnahme eines Kavitationsschadens.

1.3. Beispiele für Prozesseitarchitekturen

Dieses Kapitel gibt einen kurzen Überblick über typische Vertreter von Prozessleitsystemen aus dem Bereich der industriellen Automation sowie dem Brauwesen.

SIMATIC PCS 7 (Siemens)

SIMATIC PCS 7 bezeichnet ein von der Firma Siemens entwickeltes Prozessleitsystem, dass vor allem in Europa einen Quasi-Standard darstellt. Es zählt zu den prominenten

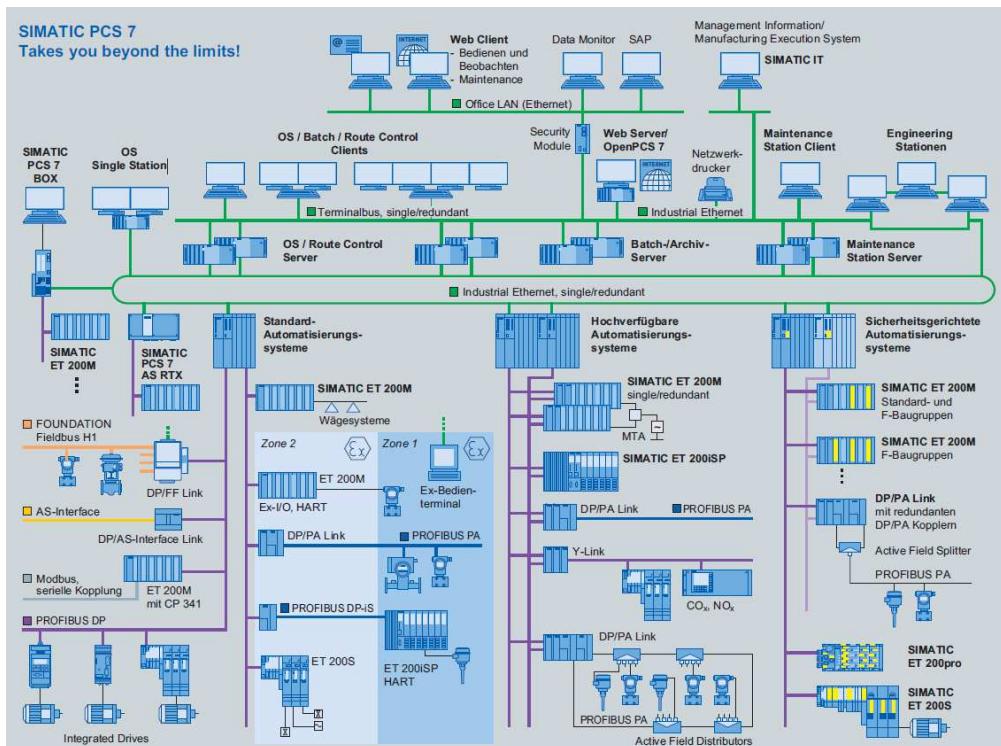


Abbildung 1.28.: Beispielanwendung eines SIMATIC PCS 7.

Vertretern der Server-Client-Architektur, welche zwei getrennte Bussysteme verwenden. Der Systembus verbindet dabei alle PNKs untereinander und mit dem Server. Dieser sammelt zyklisch alle Daten der PNKs ein und stellt dieses konsistente, globale Prozessabbild den OSen und ESEN per Terminalbus zur Verfügung. Abb. 1.28 zeigt ein Beispielsystem. Als Systembus und Terminalbus dienen Industrial Ethernet. Der Terminalbus ist über ein Gateway mit dem Firmen-LAN verbunden. Am Systembus hängen mehrere PNKs, welche die steuerungstechnischen Aufgaben erfüllen. Deren Sensoren und Aktoren sind als Remote-I/O per Profibus mit der PNK verbunden⁷. In Ex-Zonen wird die entsprechende Version SIMATIC ET 200iSP verwendet. Wenn Geräte mit anderen Feldbussen verwendet werden, können diese per Link-Baustein integriert werden.

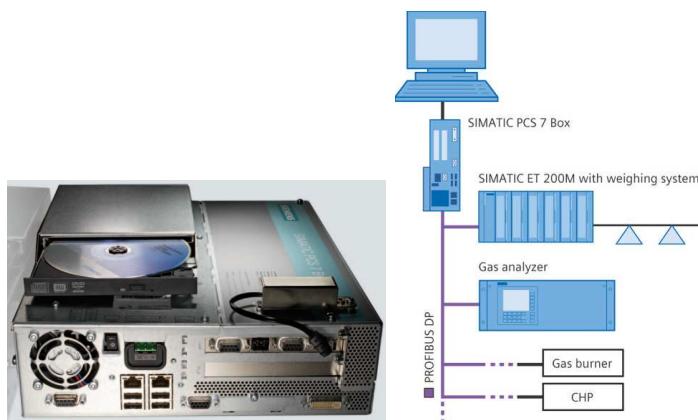


Abbildung 1.29.: Das Bild links zeigt die Hardware einer PCS 7 BOX, rechts ist eine Beispielanwendung dargestellt.

SIMATIC PCS 7 BOX (Siemens)

SIMATIC PCS 7 BOX ist eine IPC-basierte Lösung für kleine Anwendungen wie z. B. für angeschlossene Teilanlagen oder die Automatisierung eines Labors bzw. Technikums, welche nicht den vollen Umfang des PCS 7 Systems benötigen. Dabei sind alle Teile eines Prozesseitensystems, wie OS, ES und PNK, in einer Einheit integriert. In Abb. 1.29 ist die Hardware der PCS 7 BOX dargestellt. Neben dem DVD-Laufwerk und der 24V DC Stromversorgung sind eine Reihe von Kommunikationsschnittstellen wie Ethernet und Profibus vorhanden. Die gezeigte Beispielanwendung verwendet die PCS 7 BOX zur Steuerung und Visualisierung, wobei weitere Komponenten wie der Gasbrenner und -analysator per Profibus angebunden sind. Weitere Sensorik und Aktorik werden über die Remote-I/O-Komponente SIMATIC ET 200M angesteuert.

⁷Ein Beispiel für einen Remote-IO-Baustein wäre die SIMATIC ET 200.

Freelance (ABB)

Im Gegensatz zur SIMATIC PCS 7 ist das Prozesseitsystem Freelance der Firma ABB ein Beispiel für eine Einzelbusarchitektur. Freelance richtet sich dabei an kleinere bis mittlere Anlagen in der Prozessindustrie. Sowohl OSen und ESEN als auch PNKs sind mit demselben Industrial Ethernet vernetzt. Die Sensorik und Aktorik der einzelnen PNKs ist je nach Type entweder direkt, per Remote-I/Os oder durch verschiedene Feldbusse angebunden, wie in Abb. 1.30 dargestellt.

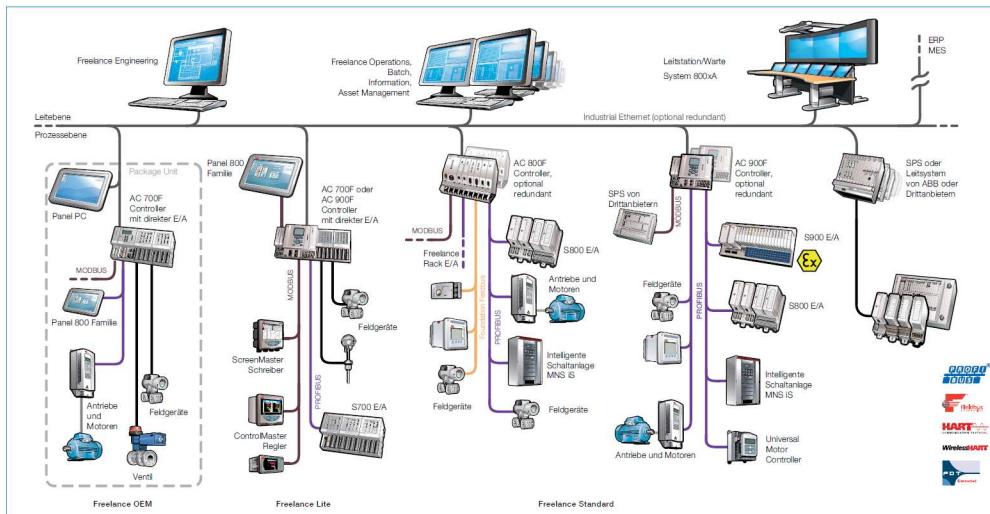


Abbildung 1.30.: Beispieldarstellung des Freelance Prozesseitsystems.

Brewmaxx (ProLeiT)

Während alle vorherigen Prozesseitsysteme branchenübergreifend eingesetzt werden, konzentriert sich die Firma ProLeiT mit Brewmaxx auf den Nischenmarkt der Brauereien. Es verwendet eine Server-Client-Architektur und ist modular in die sog. Basissysteme Direct iT (siehe Abb. 1.32), Liqui iT und Acquis iT aufgeteilt. Das Basissystem Direct iT stellt den Kern des Prozesseitsystems Brewmaxx dar, auf welches die Rezeptursteuerung Liqui iT und die Datenerfassungs- und Visualisierungskomponente Acquis iT aufsetzen. Neben diesen Basissystemen werden noch sog. Module angeboten (siehe Abb. 1.31). So stellt *Brewmaxx Material* prozessnahes Materialmanagement zur Verfügung, mit welchem neben Bestandsinformationen auch die Rückverfolgung einzelner Chargen möglich ist. *Brewmaxx Integrate* bietet anlagenweites Informationsmanagement und ermöglicht durch eine web-basierte Arbeitsweise eine besonders einfache Einbindung von Clients.

ohne spezielle Software. Mittels *Brewmaxx Connect* können auch Fremdsysteme eingebunden werden.

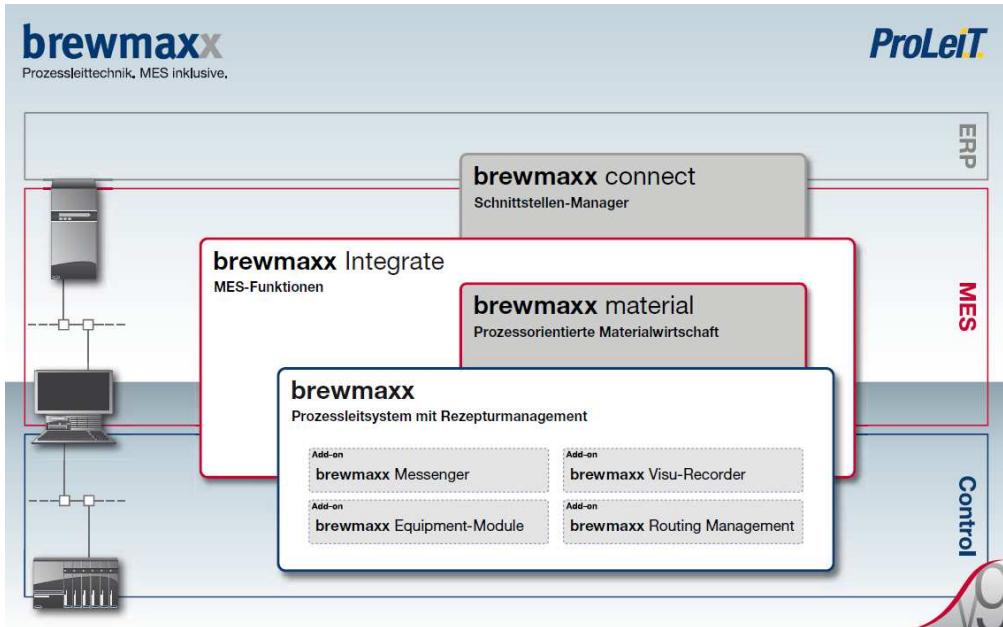


Abbildung 1.31.: Übersicht über das Brewmaxx Prozessleitsystem und die Module *Brewmaxx Material*, *Brewmaxx Integrate* und *Brewmaxx Connect*.

1.4. Projektierung leittechnischer Strukturen

1.4.1. Rohrleitungs- und Instrumentierungsfließbild

Das Rohrleitungs- und Instrumentierungsfließbild (R&I) bildet als gewerkeübergreifendes Planungsdokument der Vorplanung die Schnittstelle zwischen der Verfahrenstechnik und der Automatisierungstechnik. Als Aspekte der Verfahrenstechnik sind im R&I die zur Prozessrealisierung benötigten Komponenten wie Behälter, deren Verbindungen über Rohrleitungen, sowie Aktoren (z. B. Ventile) zum Eingriff in den Prozess und Sensoren (vgl. Abb. 1.33). Die beiden letztgenannten Komponentenklassen wiederum stellen die Schnittstellen zur Automatisierungstechnik dar. Jedem Prozessinterface (Sensor oder Aktor) wird eine sogenannte PLT-Stelle zugeordnet und im R&I verzeichnet. Abbildung 1.34 zeigt ein Beispiel einer PLT-Stelle mit den unterschiedlichen grafischen Repräsentationen.

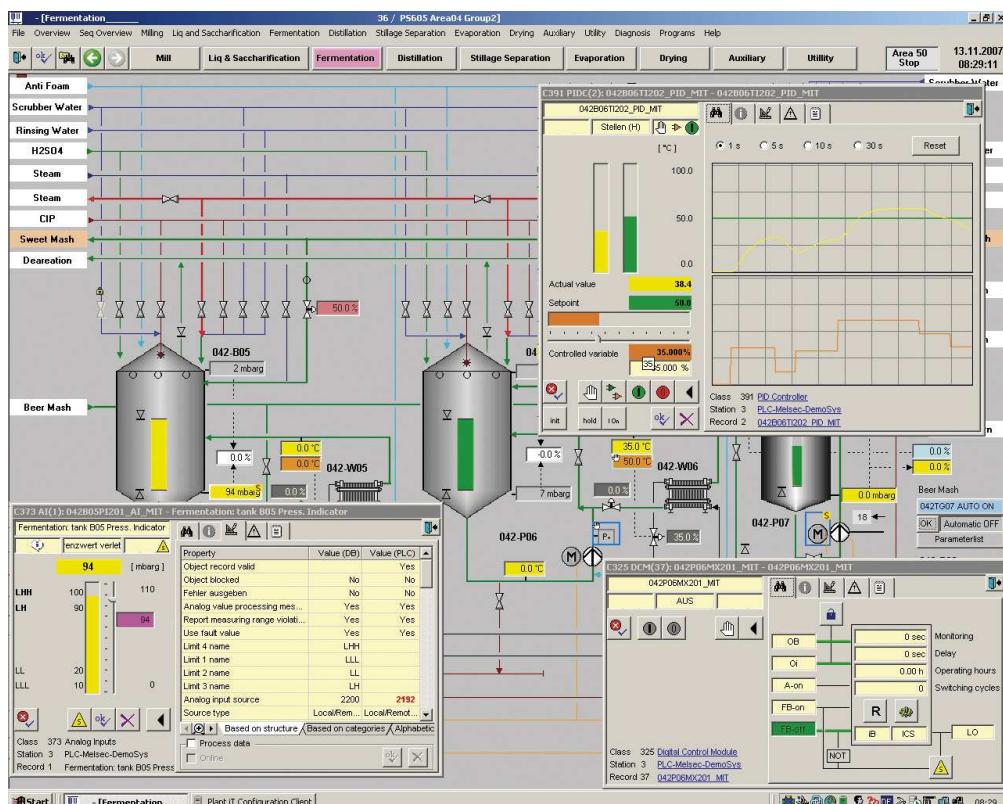


Abbildung 1.32.: Direct iT - der Kern des Brewmaxx Prozessleitsystems.

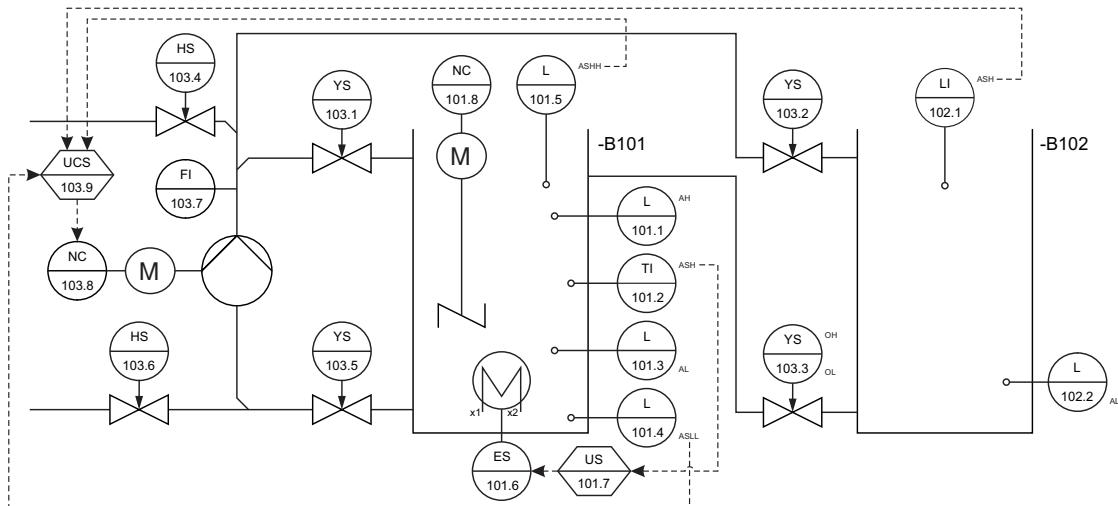


Abbildung 1.33.: Beispiel eines Rohrleitungs- und Instrumentierungsfließbilds (R&I).

Das R&I zeigt die verfahrenstechnische Anlage im Odo-Struger-Labor, welche für die zugehörige Laborübung verwendet wird.

Anhand der grafischen Repräsentation einer PLT-Stelle kann bereits der Ausgabe- bzw. Bedienort des zugrundeliegenden Gerätes erkannt werden. Feldgeräte, welche keine weitere Verbindung zu einem Prozessleitsystem haben (z. B. rein händisch betätigtes Ventile ohne Stellungsrückmeldung), werden ohne Querstrich repräsentiert. Ist beispielsweise der Messwert eines Sensors in einer zentralen Leitwarte visualisiert, so wird dies durch einen einfachen Querstrich gekennzeichnet. Bei einer Visualisierung in einem örtlichen (dezentralen) Leitstand wird dies hingegen mit einem doppelten Querstrich signalisiert.

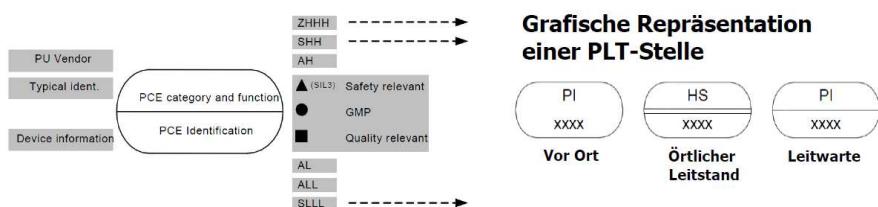


Abbildung 1.34.: Aufbau und Varianten von PLT-Stellen.

Eine PLT-Stelle beinhaltet ferner einen Kategorien- und Funktionsschlüssel sowie eine eindeutige Identifikationsnummer. Die möglichen Kategorien und Funktionen sind im Anhang A in den Tabellen A.1 bzw. A.2 zusammengefasst.

1.5. Chargenprozesse

Wie bereits in Abschnitt 1.1 erwähnt lassen sich technische Prozesse in kontinuierliche, diskontinuierliche sowie chargenorientierte Prozesse einteilen. Nach dem Standard DIN EN 61512-1 bzw. ISA-88 wird ein Chargenprozess folgendermaßen definiert:

„Ein Prozess, der zur Herstellung von endlichen Stoffmengen führt, indem Mengen von Einsatzstoffen unter Nutzung einer oder mehrerer Einrichtungen innerhalb eines endlichen Zeitraums einer geordneten Folge von Verarbeitungsaktivitäten unterzogen werden.“

Ein Chargenprozess ist somit durch zyklisches Abarbeiten von Produktionsschritten wie Füllen, Bearbeiten (Verweilen) oder Entleeren charakterisiert. Dabei treten auch unproduktive Totzeiten auf, der Stofffluss erfolgt schubweise, die Zu- und Abfuhr zu diskreten Zeitpunkten. Die charakteristischen Größen wie Druck, Temperatur und Stoffkonzentration sind im Idealfall in der ganzen Anlage örtlich gleich aber in den Bearbeitungsphasen (Verweilzeit) veränderlich, z. B. durch Heizen, Umwälzen, Kühlen.

Dem gegenüber stehen die kontinuierlichen Prozesse auch Fließprozesse genannt, welche durch dauerhaften Betrieb ohne Totzeiten mit ununterbrochenem Stoff- und Energiefluss gekennzeichnet sind. Bei Idealbetrieb geschieht die Produktion unter gleich bleibenden Bedingungen, d. h. alle charakteristischen Größen wie Druck, Temperatur und Stoffkonzentration sind entlang des Strömungspfads in der Anlage örtlich verschieden aber zeitlich konstant. Im Idealfall werden solche Prozesse nur für Instandhaltung unterbrochen. Ein typisches Beispiel für einen Fließprozess ist die Raffinierung in der Ölindustrie.

1.5.1. Rezepturen

Ein Rezept oder eine Rezeptur ist neben dem R&I eines der grundlegenden verfahrenstechnischen Dokumente bei der Planung von Anlagen für Chargenprozesse (engl. Batch Process). Der Begriff *Rezept* ist vom lateinischen *recipe* für *nimm* abgeleitet und aus historischen Gründen beibehalten worden. Es beschreibt bildhaft den Zusammenhang zwischen Einsatzstoffen und Verfahren, da ein Rezept zur Herstellung eines Stoffes sowohl die Edukte als auch die Verarbeitungsschritte enthält. Ein Rezept beinhaltet somit die Produktionsprozessbeschreibung, die als Vorlage zur Erstellung einer Ablaufsteuerung dient.

Der Standard (DIN 61512-2) beschreibt einerseits die Rezeptformen als Resultat und Abbildung der Rezeptentwicklung und andererseits die verschiedenen Elemente zur Prozessbeschreibung. Weiters definiert die DIN 61512-2 die vier Rezeptarten Verfahrensrezept,

Werksrezept, Grundrezept und Steuerrezept (vgl. Abb. 1.35).

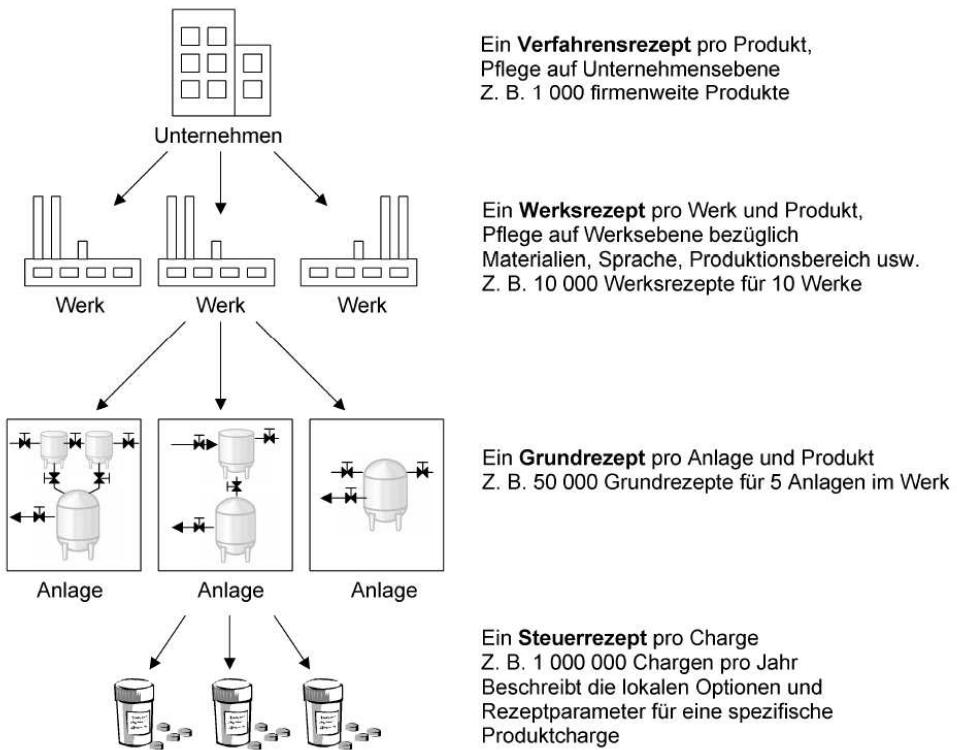


Abbildung 1.35.: Rezeptarten.

Verfahrensrezept Die Grundlage für Rezepte auf den unteren Ebenen bildet das Verfahrensrezept, welches auf Unternehmensebene definiert wird. Das Verfahrensrezept wird ohne spezifische Kenntnis der Anlagenausrüstung, welche zur Herstellung des Produkts benutzt wird, erstellt. Es bestimmt die Rohstoffe, ihre relativen Mengen und die erforderliche Verarbeitung, allerdings ohne Bezug zu einem bestimmten Werk oder der in diesem Werk verfügbaren Ausrüstung. Es wird von Personen mit Kenntnissen der verfahrenstechnischen Details und den Verarbeitungsanforderungen, die für das betreffende Produkt typisch sind, erstellt.

Werksrezept Das Werksrezept ist spezifisch für ein bestimmtes Werk eines Unternehmens. Es ist eine Kombination von werksspezifischer Information und dem Verfahrensrezept. Es wird gewöhnlich aus einem Verfahrensrezept abgeleitet, um die Bedingungen einer bestimmten Produktionsstätte oder gesetzlichen Rahmenbedingungen zu erfüllen.

Es bietet einen für werksbezogene, langfristige Produktionsplanung erforderlichen DetAILlierungsgrad, ist jedoch noch nicht spezifisch genug für eine bestimmte Kombination von Anlagenausführungen. Es kann mehrere Werksrezepte geben, welche vom gleichen Verfahrensrezept abgeleitet wurden. Jedes Teil-Werksrezept bildet hier einen Teil des Verfahrensrezepts für ein bestimmtes Werk ab.

Grundrezept Die verfahrenstechnische Umsetzung der durch das Verfahrensrezept vorgegebenen chemischen Verfahren wird im Grundrezept spezifiziert. Dafür muss das Rezept bezüglich des Prozessablaufs, der Produktionsgröße, der verwendeten Typen von Anlagen und des Automatisierungsgrades präzisiert werden. Es kann mehrere von einem Werksrezept abgeleitete Grundrezepte geben, von denen jedes einen Teil des Werksrezepts abdeckt, welcher in einer Anlage ausgeführt werden kann. Es beinhaltet alle Informationen des Werksrezepts und zusätzlich die Produktionsanweisungen und Verfahrensvorschriften zu den jeweiligen Verfahrensabschnitten. Das Grundrezept ist nicht mehr anlagenneutral, da es verfahrenstechnische und automatisierungstechnische Informationen bezüglich des Prozessablaufs enthält. Anderseits ist ein Grundrezept so konzipiert, dass keine konkreten Anlagenteile wie z. B. Kessel oder Ventile, bezeichnet werden.

Steuerrezept Ein *Steuerrezept* wird aus dem Grundrezept nach Anforderungen der Produktionsdurchführung auf einer konkreten Produktionsanlage generiert. Es beinhaltet unter anderem die auftragsspezifischen Informationen wie Produktionstermine, Produktionsmenge, Materialchargen, Chargennummern sowie die Angaben zum geplanten Starttermin, Laufzeit und Teilanlagenzuweisung. Es beinhaltet auch alle Daten und Werte bezüglich Quantität, Qualität und Prozessverlauf.

2. Pneumatische Komponenten

Der Einsatz von Pneumatik reicht historisch lange zurück und wird meist beginnend mit Heron von Alexandria erzählt. Eine der bekanntesten historischen Anwendungen pneumatischer Prinzipien ist die Tempeltüröffnung nach Heron aus dem ersten Jahrhundert n. Chr., wie sie in Abb. 2.1 dargestellt ist.

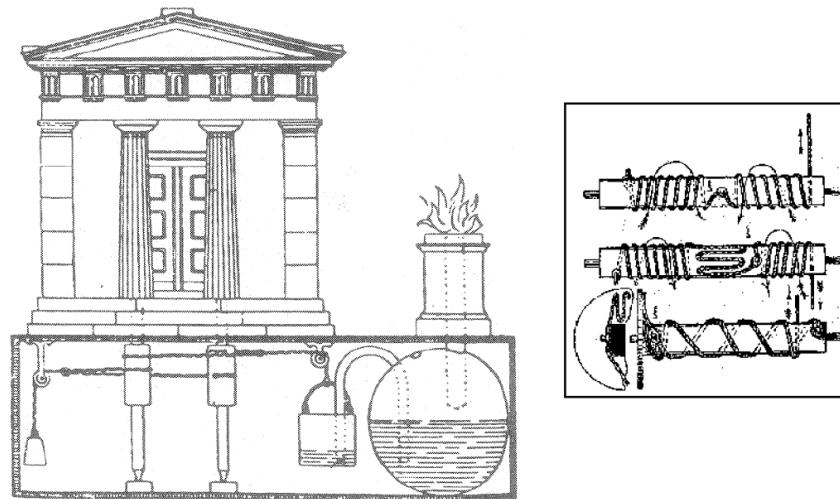


Abbildung 2.1.: Tempeltüröffnung nach Heron.

Erste großtechnische Anlagen pneumatischer Natur entstanden im Zuge der industriellen Revolution Mitte des 19. Jahrhunderts. Seither werden pneumatische Komponenten auch im Kontext industrieller Fertigung verwendet. Ihre breite Verwendung erschließt sich aus den für viele Einsatzzwecke günstigen Eigenschaften. Im Vergleich zu Mechanik, Elektrotechnik und Hydraulik bietet die Pneumatik in einigen Bereichen wesentliche Vorteile. Im Folgenden werden einige Eigenschaften bzw. deren Vor- und Nachteile pneumatischer Systeme kurz erläutert.

Vorteile

- **Menge:** Luft ist praktisch überall in unbegrenzter Menge verfügbar.
- **Transport:** Luft kann sehr einfach in Rohrleitungen über weite Strecken transportiert werden.
- **Speicherfähigkeit:** Druckluft kann in einem (transportablen) Druckbehälter gespeichert und dort entnommen werden.
- **Temperatur:** Druckluft ist nahezu unempfindlich gegen Temperaturschwankungen. Dies garantiert einen zuverlässigen Betrieb selbst unter extremen Bedingungen.
- **Sicherheit:** Druckluft bietet kein Risiko in Bezug auf Feuergefahr.
- **Sauberkeit:** Nichtgeölte Druckluft verursacht keine Verschmutzung.
- **Geschwindigkeit:** Über Druckluft können hohe Kolbengeschwindigkeiten und kurze Schaltzeiten erzielt werden.
- **Überlastsicherung:** Pneumatische Werkzeuge und Arbeitselemente können bis zum Stillstand belastet werden und sind somit überlastsicher.

Nachteile

- **Aufbereitung:** Druckluft muss aufbereitet werden, da sonst die Gefahr erhöhten Verschleißes der Pneumatikkomponenten durch Schmutzpartikel und Kondenswasser besteht.
- **Verdichtung:** Mit Druckluft ist es, vor allem wegen der Kompression, nicht möglich, gleichmäßige und konstante Kolbengeschwindigkeiten zu erzielen. Kontinuierliche Kolbenbewegungen lassen sich nur durch komplizierte nichtlineare Regelungen erreichen.
- **Kraft:** Druckluft ist nur bis zu einem bestimmten Kraftbedarf wirtschaftlich. Bei dem normalerweise verwendeten Betriebsdruck von 6 bis 7 bar und in Abhängigkeit von Hub und Geschwindigkeit liegt diese Grenze zwischen 40 und 50 kN.
- **Abluft:** Das Entweichen der Luft ist mit hoher Geräuschentwicklung verbunden. Dieses Problem kann aber weitgehend durch schallabsorbierende Materialien und Schalldämpfer gelöst werden.

2.1. Einführung in die Pneumatik

Unter dem Begriff Pneumatik versteht man die technische Nutzung von Druckluft, also verdichteter Luft, zur Übertragung und Wandlung von Energie. Gemeinsam mit der Hydraulik, welche flüssige Medien verwendet, ist sie ein Teilgebiet der Fluidtechnik (siehe Abb. 2.2). Die Verwendung von Druckluft für andere Zwecke, wie z. B. die Weiterführung von Stoffen oder zur Trocknung, zählt definitionsgemäß nicht zur Pneumatik. Eine Ausnahme stellt dabei die Informationsverarbeitung mittels pneumatisch arbeitenden

Logikgliedern dar, welche zur Steuerung pneumatischer Geräte verwendet wird.

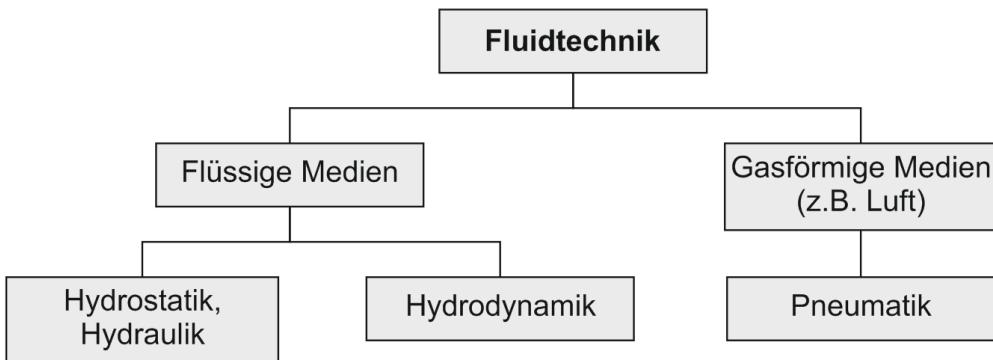


Abbildung 2.2.: Einordnung der Pneumatik als Teilgebiet der Fluidtechnik.

Obwohl die Funktionsprinzipien von Hydraulik und Pneumatik sehr ähnlich sind und sich auch die eingesetzten Bauteile prinzipiell wenig unterscheiden, führen die physikalischen Spezifika der Medien zu verschiedenen Eigenschaften. Moderne Hydrauliksysteme verwenden meist spezielle Hydrauliköle¹ auf Mineralölbasis. Durch den hohen Kompressionsmodul von Hydraulikölen, in etwa $2 \cdot 10^9$ Pa, kann es für die meisten technischen Anwendung als ideal inkompressibel angenommen werden. Umgebungsluft hingegen besitzt einen Kompressionsmodul von etwa $1 \cdot 10^5$ Pa, weshalb pneumatische Systeme eine wesentlich geringere Steifigkeit als hydraulische Systeme aufweisen und die Positionsregelung von pneumatischen Aktoren vergleichsweise aufwendig ist. Eine unvollständige Auflistung von typischen Eigenschaften hydraulischer und pneumatischer Systeme zeigt Tab. 2.1.

Da die Komponenten von Hydraulik und Pneumatik funktional ähnlich sind, werden für beide dieselben Schaltsymbole verwendet. Diese sind, wie bereits in Abschnitt 1.2.3 kurz erwähnt, in der Norm DIN ISO 1219 für die gesamte Fluidtechnik vereinheitlicht. Ohne Anspruch auf Vollständigkeit sind in Abb. 2.3 die Schaltsymbole für die wichtigsten Geräteklassen angeführt.

Wie eingangs erwähnt, können mit pneumatischen Komponenten logische Schaltungen aufgebaut werden. Das in Abb. 2.3 abgebildete Zweidruck- und Wechselventil entspricht dabei einer logischen UND- bzw. ODER-Funktion. Im Vergleich zu elektrischen Steuerungen sind diese sehr kostenintensiv, weshalb oft gemischte Schaltungstypen auftreten, also elektrisch, pneumatisch und/oder hydraulisch. Man unterscheidet deshalb aus Sicht

¹Wichtige Eigenschaften von Fluiden zum Einsatz in hydraulischen Geräten sind beispielsweise Schmierfähigkeit, Alterungsbeständigkeit und ein hohes Benetzungsvermögen. Ferner sollte der Temperaturreinfluss auf die Viskosität gering sein und der Flammpunkt möglichst hoch liegen.

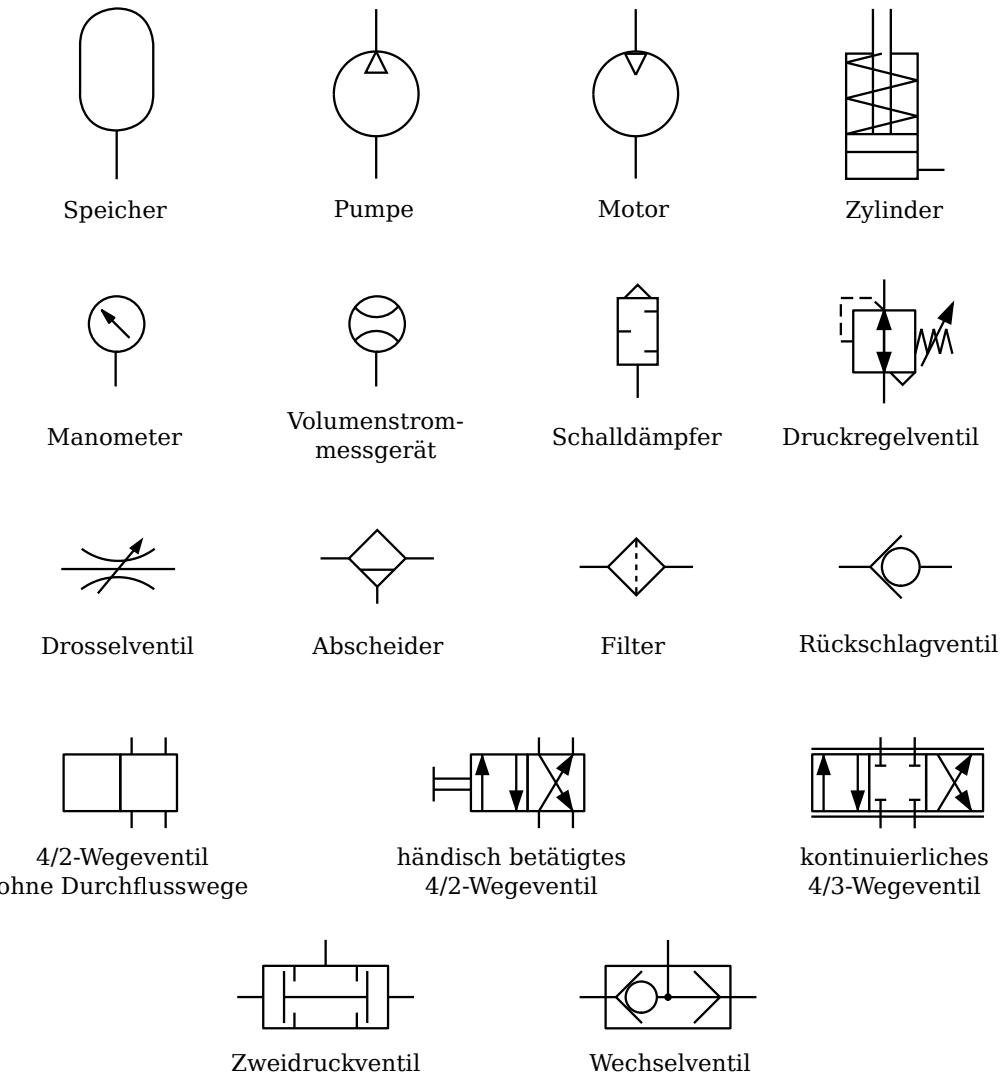


Abbildung 2.3.: Ausgewählte Schalsymbole der Fluidtechnik.

Hydraulik	Pneumatik
Hohe Kräfte	Hohe Geschwindigkeit
Sehr hohe Leistungsdichten	Relativ hohe Leistungsdichte
Einfache Positionsregelung	Meist ungeregelter Punkt-zu-Punkt-Betrieb
Geschlossene Kreise	Offene Kreise möglich
Hohe Stellgenauigkeit	Laut
Leckagen belasten die Umwelt	Keine Brandgefahr
	Keine Umweltgefährdung durch Medium

Tabelle 2.1.: Gegenüberstellung von Hydraulik und Pneumatik.

der Pneumatik zwischen folgenden Schaltungsklassen.

Vollpneumatischen Steuerungen Sie verwenden ausschließlich pneumatische Komponenten zur Signaleingabe, Steuerung und als Arbeitselemente. Letztere werden zum Teil indirekt über Verstärker, d. h. über durch Druckluft betätigtes Ventile, angesteuert, wenn der Steuerkreis nicht in der Lage ist, den Aktor direkt zu betreiben. Vollpneumatische Steuerungen werden bevorzugt in explosionsgefährdeten Umgebungen (EX-Bereichen) eingesetzt.

Elektropneumatischen Steuerungen Im Gegensatz zu vollpneumatischen Steuerungen ist hier die Signalverarbeitung elektrisch ausgeführt, lediglich die Arbeitselemente sind pneumatische Komponenten. Diese werden über Magnetventile vom elektrischen Steuerkreis betätigt. Dies reduziert bei komplexen Steuerungsaufgaben den Kostenaufwand.

Pneumatisch-hydraulischen Steuerungen Während die Signalverarbeitung von pneumatischen Komponenten erledigt wird, sind die Arbeitselemente hydraulisch ausgeführt. Die Hydraulikventile werden entsprechend pneumatisch betätigt.

Ein Beispiel, noch bevor einzelne Komponenten im Detail vorgestellt werden: In Abb. 2.4 ist eine einfache vollpneumatische Steuerung dargestellt. Der Versorgungsdruck wird im 0Z-Block durch Abscheider und Filter gereinigt und auf konstanten Druck geregelt. Durch den Schalter 0S kann die Schaltung komplett deaktiviert werden. Im Grundzustand, also so lange die Schalter 1S1, 1S2 und 1S3 nicht betätigt wurden, ist der Zylinder eingefahren. Um das pneumatisch doppelt betätigtes Ventil 1V2 in die linke Stellung zu

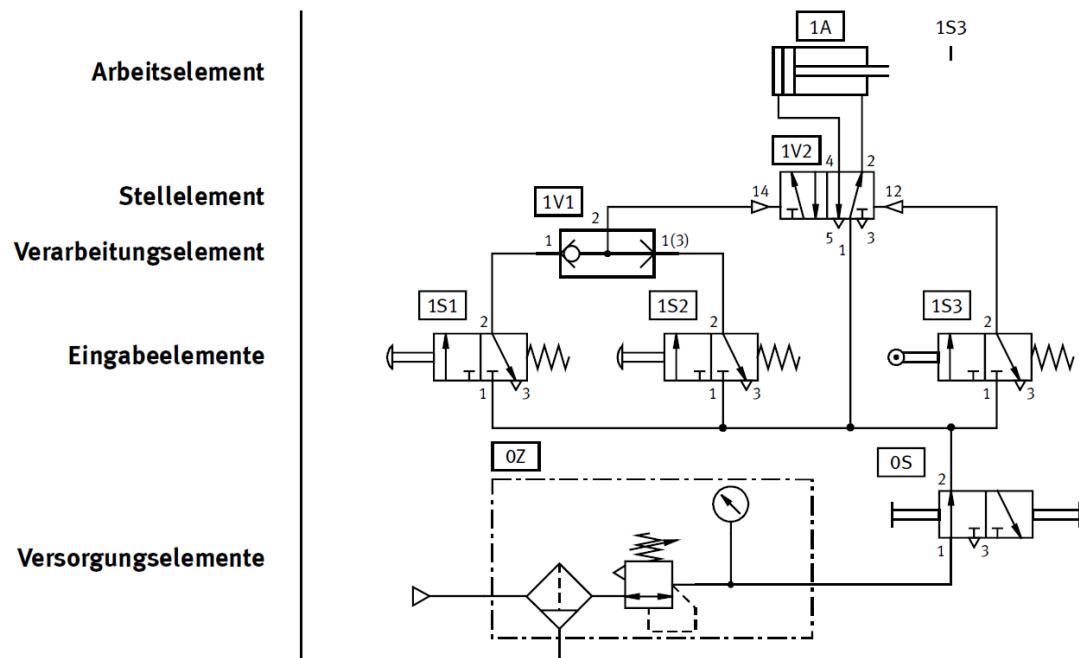


Abbildung 2.4.: Beispiel einer vollpneumatischen Steuerung.

bringen und damit den Zylinder ausfahren zu lassen, muss **1S1** oder **1S2** betätigt sein, während **1S3** nicht geschaltet sein darf. Es gilt also für den logischen Zustand des Zylinders $1A = (1S1 \vee 1S2) \wedge \neg 1S3$. Neben dem schematischen Kolben des Zylinders in Abb. 2.4 ist eine Endlagenbetätigung für **1S3** eingezeichnet. Dadurch wird der Zylinder bei Erreichen der Endlage automatisch eingefahren.

2.2. Drucklufterzeugung

Die Verwendung von Pneumatik in industriellen Anlagen erfordert eine funktionierende Infrastruktur zur Versorgung mit Druckluft. Diese besteht im Wesentlichen aus einem Netz, eventuell Druckluftspeichern, Aufbereitungseinheiten und Verdichtern zur Drucklufterzeugung. Das Schaltsymbol von Verdichtern ist jenes der Pumpe in Abb. 2.3. Während bei inkompressiblen Medien nur von Pumpen gesprochen werden kann, hängt die Bezeichnung bei kompressiblen Medien mitunter von der hauptsächlichen Funktion ab. Soll das Medium befördert werden, spricht man von Pumpen; soll es verdichtet werden, von Verdichtern bzw. Kompressoren.

Die bekannten Bauarten von Verdichtern lassen sich in zwei prinzipbedingte Klassen einordnen: Einerseits die Verdrängungsverdichter, deren Domäne hohe Druckniveaus bei gleichzeitig geringen Fördermengen sind. Komplementär dazu erreichen Turboverdichter bzw. dynamische Verdichter, also mit Schaufeln bestückte Laufräder, hohe Volumenströme bei vergleichsweise geringeren Drücken. Eine Übersicht über die typischen Betriebsbereiche der verschiedenen Verdichterbauarten gibt Abb. 2.5.

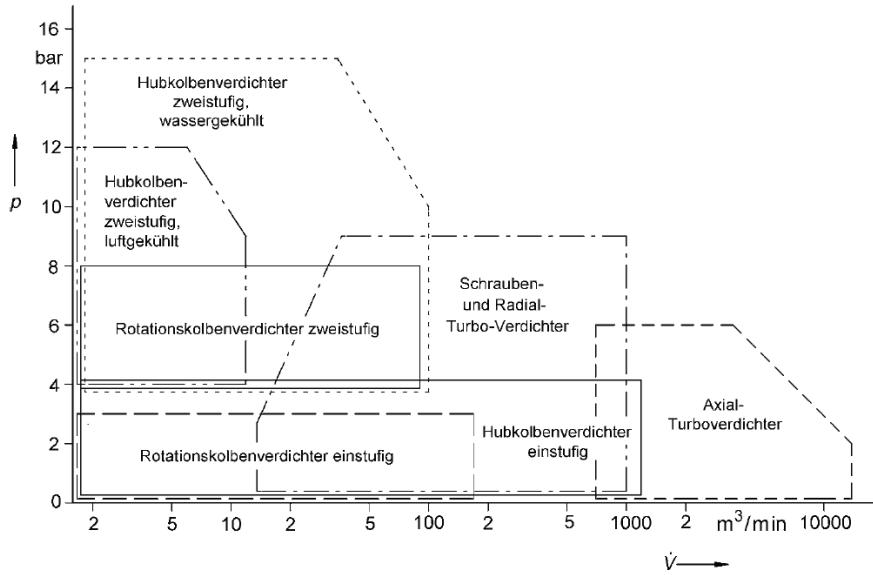


Abbildung 2.5.: Mögliche Arbeitsbereiche der unterschiedlichen Verdichterprinzipien.

2.2.1. Hubkolbenverdichter

In seiner Grundbauweise nach Abb. 2.6 besteht ein Hubkolbenverdichter aus einer zylindrischen Kammer mit Einlassventil (2) und Auslassventil (3) sowie einem Zylinder (1), welcher über eine Pleuelstange mit der Kurbelwelle verbunden ist. Um die beim Kompressionsprozess entstehende Wärme abzuführen, sind an der Außenwand der Zylinderkammer Kühlrippen (5) angebracht. Durch Kurbelwelle und Pleuelstange wird die Drehbewegung in eine lineare Bewegung des Zylinders umgesetzt, welche einen in vier Phasen ablaufenden zyklischen Prozess bewirkt. Während der Ansaugphase (I) ist das Einlassventil geöffnet und der Kolben zieht Luft in die Zylinderkammer. Aufgrund des endlichen Querschnitts der Zuleitung bzw. der Luftfilter und des daraus resultierenden Druckabfalls liegt der Druck in der Kammer p unter dem Umgebungsdruck p_{amb} . Am unteren Umkehrpunkt mit maximalem Hub verschwindet der zufließende Volumenstrom und der Kammerdruck ist gleich dem Umgebungsdruck; damit endet die Ansaugphase und das

Ventil schließt sich. Während der Kompressionsphase (II) ist die Zylinderkammer dicht verschlossen und der Druck steigt, in erster Näherung gilt das Boyle-Mariottesche Gesetz, reziprok zum sinkenden Kammervolumen. Steigt der Kammerdruck über den am Auslass liegenden Enddruck p_2 , öffnet sich das Auslassventil und die Ausschubphase (III) beginnt. Bei annähernd konstantem Druck p_2 wird verdichtete Luft aus der Kammer gedrückt. Am oberen Umkehrpunkt schließt sich das Auslassventil, die Bewegungsrichtung des Kolbens kehrt sich um und die Rückexpansionsphase (IV) beginnt. Die Luft im sog. Schadraum wird expandiert, bis der Kammerdruck p unter den Umgebungsdruck p_{amb} sinkt, dass Ventil geöffnet wird und der Kreisprozess von Neuem beginnt. Qualitativ lässt sich dieser zyklische Prozess auf alle Arten von Verdrängungsverdichtern übertragen.

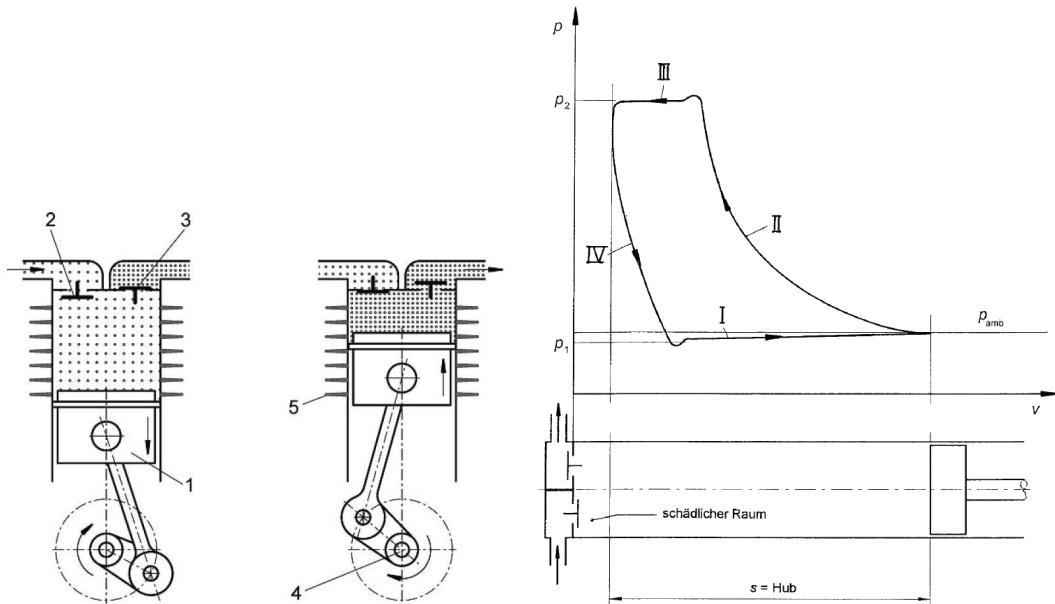


Abbildung 2.6.: Prinzipskizze und p-V-Diagramm eines einfachen Hubkolbenverdichters.

Der in Abb. 2.6 eingezeichnete Schadraum bezeichnet das Restvolumen zwischen Zylinderkopf und oberem Umkehrpunkt des Zylinders. Dieses Volumen ergibt sich aus den Freiräumen in den Ventiltaschen, den Fertigungstoleranzen und einigen konstruktiven Eigenarten. Durch den Schadraum verringert sich der Wirkungsgrad des Verdichters, da in der Rückexpansionsphase (IV) das Volumen des Schadraums expandiert wird, bevor der Verdichter neues Volumen ansaugen kann.

Die Verwendung mehrerer Zylinderkammern führt zu einer homogeneren Belastung an der Welle. Durch Parallelschaltung mehrerer Zylinder kann die Fördermenge, durch Serienschaltung der maximale Druck erhöht werden. So kann sowohl Fördermenge als auch Kompressionsverhältnis über weite Strecken angepasst werden.

2.2.2. Membranverdichter

Sehr ähnlich zum Hubkolbenverdichter funktioniert der Membranverdichter wie in Abb. 2.7 dargestellt. Anstatt die Luft direkt in die Zylinderkammer zu saugen, wird die Deformation einer elastischen Membran genutzt. Im Vergleich zum Hubkolbenverdichter ist die bewegte Fläche groß, während der Hub eher geringer ausfällt. Ein wesentlicher Vorteil von Membranverdichtern ist die Trennung von Kompressionsraum und Kolbenraum. Dadurch wird das größte Problem des Hubkolbenverdichters, die Abdichtung des Zylinderrandes, behoben. Gleichzeitig sind so Kompressionsraum und Kolbenraum vor wechselseitiger Verunreinigung geschützt. Letzteres ist abseits vom Medium Luft z. B. in der Pharma- und Lebensmittelindustrie essenziell.

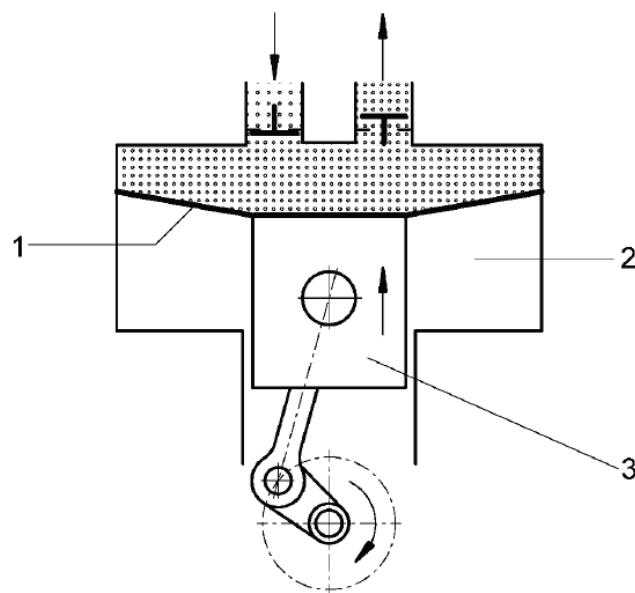


Abbildung 2.7.: Prinzipskizze eines Membranverdichters: (1) Membran, (2) mediumfreie Rückseite der Zylinderkammer und (3) Kolben.

2.2.3. Drehkolbenverdichter

Diese Bauform wird auch Rotations- oder Umlaufverdichter genannt. Kennzeichnend für sie ist die rotierende Ausführung der Komponenten, welche kinetische in pneumatische Energie umsetzen. In Abb. 2.8 ist als Beispiel eine Vielzellen-Rotationsverdichter dargestellt. In einem kreisrunden Gehäuse (4) dreht sich ein exzentrisch montierter, ebenfalls kreisrunder Rotor (1) mit radialen Lamellen (2). Durch im Rotor angebrachte Federn

und bei höheren Drehzahlen durch die Fliehkraft werden die Lamellen gegen die Gehäusewand gedrückt und bilden so abgeschlossene Zellen (5), welche durch die Exzentrizität im Zuge einer Umdrehung zunehmend komprimiert und schließlich ausgestoßen werden. Besonders vorteilhaft am Drehkolbenverdichter sind die stoßarme und gleichmäßige Förderung und der ruhige Lauf, während der Verschleiß der Lamellen und der geringe Wirkungsgrad durch die mangelnde Abdichtung der Zellen als Nachteile gelten.

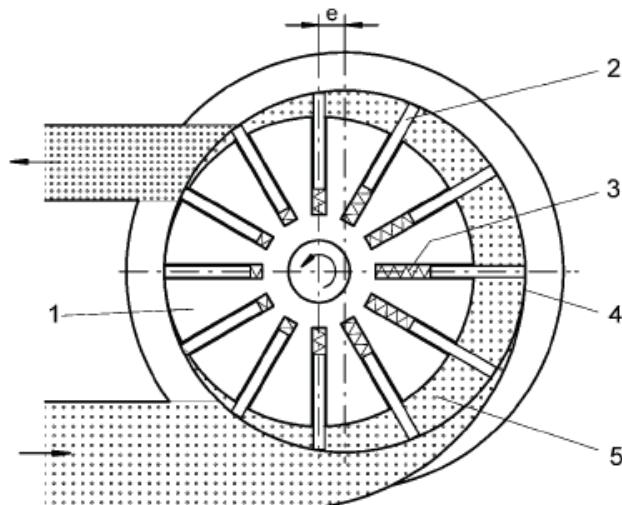


Abbildung 2.8.: Prinzipskizze eines Vielzellen-Rotationsverdichters.

2.2.4. Schraubenverdichter

Schraubenverdichter gehören wie Drehkolbenverdichter zur Gruppe der Rotationsverdichter. Die Idee zur Kompression von Fluiden mittels zwei ineinander geschlungenen Schrauben entstand bereits im 19. Jahrhundert, doch die ersten funktionsfähigen Konstruktionen erschienen aufgrund der komplexen, schwierig zu fertigenden Geometrie der Schrauben erst in den 1940ern. Abb. 2.9 zeigt eine Skizze der Verdichterschrauben und ein Bild von Schrauben und Gehäuse eines Verdichters. Zur Funktionsweise: Die rotierenden Schrauben laufen in einem an der Mantelseite dicht verschlossenem Gehäuse und saugen an den Stirnseiten Luft in den Bereich zwischen Schraube und Gehäusewand an. Durch die Rotationsbewegung werden so einzelne abgeschlossene Kammern in axiale Richtung bewegt, wobei sich aufgrund der Schraubenform das Volumen kontinuierlich verkleinert. An der gegenüberliegenden Stirnseite wird die so komprimierte Luft ausgestoßen. Wie schon Drehkolbenverdichter zeichnen sich auch Schraubenverdichter durch stoßarme und gleichmäßige Förderung aus. Abgesehen von den fertigungstechnischen

Anforderungen an die Verdichterschrauben sind sie einfach und robust gebaut, haben kleine Abmessungen und ein geringes Gewicht.

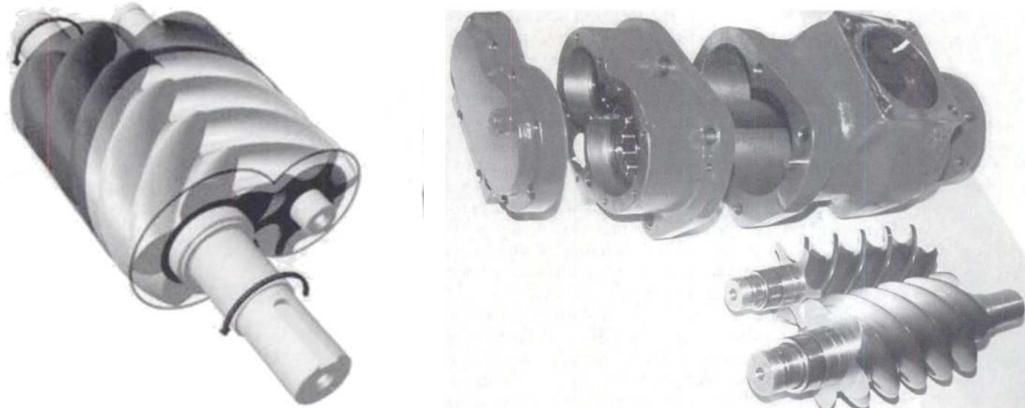


Abbildung 2.9.: Bild und Skizze zum Schraubenverdichter aus [2].

2.3. Pneumatische Ventile

Ganz allgemein werden Ventile verwendet, um Fluidströme sperren, öffnen, drosseln oder lenken zu können. Bereits in Abschnitt 1.2.3 wurde kurz auf die verschiedenen Arten von Ventilen und die unterschiedlichsten Betrachtungs- und Benennungskontexte wie mechanische Bauform, Wirkungsweise, Betätigungsart oder Anwendungsgebiet eingegangen. Hydraulische und pneumatische Ventile unterscheiden sich dabei kaum in ihrer Funktionsweise, doch ergeben sich Unterschiede in den eingesetzten Ventiltypen. Aus der Verwendung von pneumatischen Kreisen zur Verarbeitung von Informationen in der Ansteuerung von pneumatischen oder hydraulischen Aktoren resultieren beispielsweise spezifische Typen von pneumatischen Ventilen wie Wechsel- oder Zweidruckventile.

Ein wichtiges konstruktives Merkmal von Ventilen ist die Bauart des Absperrkörpers. Typische Bauformen sind Kugel- und Tellerventile. Abb. 2.10 zeigt Beispiele zu beiden Bauformen.

Kugelventile bestehen aus einer konischen Verengung, welche durch eine mit Federkraft in die Verengung gedrückte Kugel verschlossen wird. Mittels Kolben kann die Kugel aus der Verengung gedrückt werden und das Ventil öffnet. Ist der Druck an Anschluss 2 ausreichend höher, als jener an Anschluss 1 um die Federkraft zu überwinden, so öffnet das Ventil ebenfalls. So sind einfache Rückschlagventile möglich. Kugelventile sind sehr

einfach im Aufbau, jedoch nicht immer optimal dicht.

Tellerventile hingegen überkommen diesen Nachteil, sind dafür aufwendiger in der Fertigung. Die ersten drei Zeichnungen von rechts in Abb. 2.10 zeigen ein 3/2-Wegeventil in Tellerbauform. In der linken Stellung sind Anschluss 1 und Anschluss 2 miteinander verbunden, während die beiden Federn der Teller gegen den oberen Dichtungskörper drücken. Durch Betätigung des Ventils wird der Teller durch den Kolben nach unten bewegt und gegen den unteren Dichtungskörper gedrückt (zweite Zeichnung von rechts). Durch weitere Bewegung des Kolbens nach unten wird der untere Dichtungskörper zunehmend nach unten verschoben, worauf die obere Feder den oberen Dichtungskörper nach unten zieht und Anschluss 2 mit Anschluss 3 verbindet.

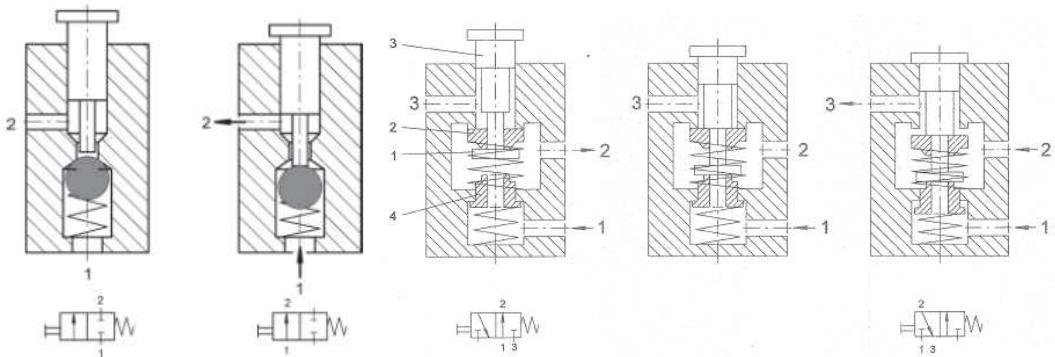


Abbildung 2.10.: Beispiele für Kugel- und Tellerventile. Die linke Abbildung zeigt ein einfaches 2/2-Wegeventil (Absperrventil) in Kugelbauform, die rechte Abbildung ein 3/2-Wegeventil in Tellerbauform mit Teller (1), oberem Dichtungskörper (2), Ventilkolben und unterem Dichtungskörper (4).

2.3.1. Betätigungsarten

Abb. 2.3 zeigt neben zwei Ventilen ohne Betätigungsart auch eines mit händischer Betätigung. Schon die Art der Zeichnung legt nahe, dass durch den symbolischen Mechanismus das Ventil geschaltet, d. h. die blockartigen Schaltstellungen verschoben werden. Je nach Wirkung der Betätigung wird das Symbol links oder rechts vom Ventilsymbol angehängt.

Händische Betätigung ist jedoch bei weitem nicht die einzige Möglichkeit. Abb. 2.11 zeigt eine Auswahl von typischen Betätigungsarten. Die meisten gezeigten Arten sind in mehreren Varianten zu finden. Beispielsweise werden Knöpfe, welche gedrückt und gezogen werden können, als beidseitig abgerundeter Knopf gekennzeichnet. Weiterhin können, wie beim elektromagnetischer Betätigung mit pneumatischer Vorstufe, Betätigungsarten in

Serie geschaltet werden. Komplementär dazu ist es möglich, mehrere parallele Betätigung zu verwenden, indem man die Blöcke übereinander zeichnet (vgl. Abb. 2.14). So kommen beispielsweise oft manuelle oder elektromagnetische Betätigungen in Kombination mit Rückstellfedern vor, um jederzeit eine wohldefinierte Ventilstellung sicherzustellen.

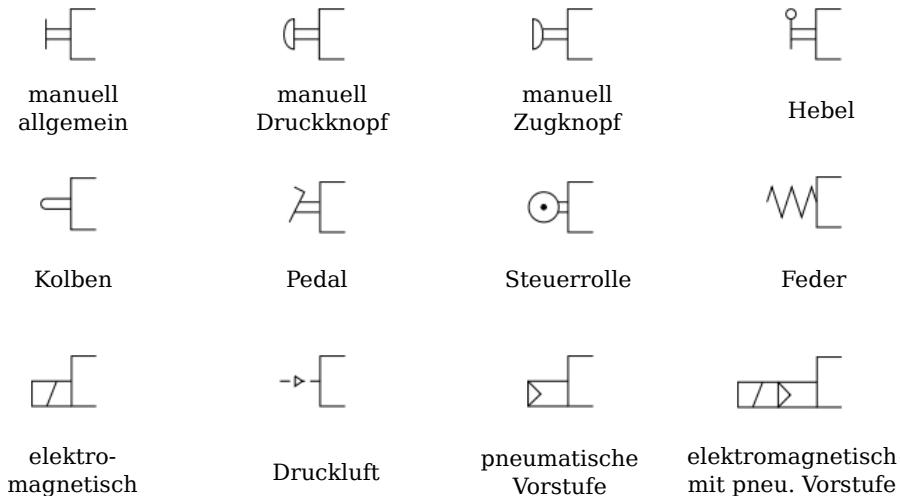


Abbildung 2.11.: Typische Betätigungsarten von Ventilen gemäß DIN ISO 1219.

2.3.2. Direkte und indirekte Ventile

Die auf den Ventilkolben wirkenden Kräfte setzen sich aus zwei verschiedenen Komponenten zusammen: einerseits die Aufgrund des stationären Drucks wirkenden Kräfte und andererseits die durch den Volumenstrom hervorgerufenen Strömungskräfte. Für Stellelemente mit größeren Volumenströmen können diese Kräfte beträchtliche Anforderungen an die Betätigungs elemente darstellen, welche durch eine pneumatische Vorstufe, wie in Abb. 2.11 dargestellt, verringert werden können. Man spricht hierbei von indirekten Ventilen.

Indirekte Ventile, manchmal auch vorgesteuerte Ventile genannt, verwenden das Medium zur Bewegung des Ventilkolbens. Ein wesentlich kleineres, in das Ventil integrierte Vorsteuerungsventil schaltet dabei mit viel geringeren Volumenströmen das eigentliche Hauptventil, wodurch die notwendige Kraft zum Schalten des Steuerventils verringert wird. So können indirekte Ventile bei elektromagnetischer Betätigung mit relativ kleinen und leistungsschwachen Spulen bestückt werden. Die Funktion eines indirekten Ventils benötigt jedoch ausreichend hohe Betriebsdrücke, da ansonsten das Hauptventil nicht mehr schaltet.

Direkte Ventile benötigen entsprechend kraftvolle Betätigungslemente, wodurch beispielsweise die Spulen von elektromagnetisch betätigten Ventilen größer, als das Ventil selbst seien können. Dafür können direkte Ventile unabhängig von den Druckverhältnissen schalten, weshalb sie z. B. im Vakumbereich eingesetzt werden.

2.3.3. Wechselventile

Wechselventile werden zur pneumatischen Informationsverarbeitung verwendet. Abb. 2.12 zeigt eine Prinzipskizze. Sie bestehen üblicherweise aus einem, in diesem Fall kugelförmigen, Schaltkörper in einem Gehäuse mit zwei Eingängen und einem Ausgang. Der Schaltkörper wird durch den höheren der beiden Eingangsdrücke an den anderen Eingang gepresst, weshalb am Ausgang stets der höhere Druck erscheint. Wird an keinen der beiden Eingänge Druckluft angelegt, wird auch am Ausgang kein Druck vorhanden sein; werden hingegen beide Eingänge mit Druckluft versorgt, setzt sich das höhere Druckniveau durch und liegt auch am Ausgang an. Diese Funktion entspricht für binäre Druckluftsignale einem logischen ODER.

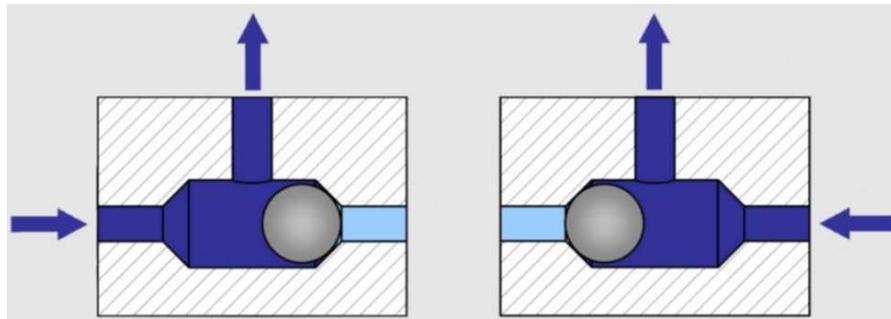


Abbildung 2.12.: Prinzipskizze eines Wechselventils.

2.3.4. Zweidruckventile

Zweidruckventile werden wie Wechselventile zur Informationsverarbeitung genutzt. Wie in Abb. 2.13 dargestellt bestehen Zweidruckventile aus einem hantelförmigen Schaltkörper in einem Gehäuse mit zwei Kammern, zwei Eingängen und einem Ausgang. Auf den hantelförmigen Schaltkörper wirkt die Differenz der beiden Eingangsdrücke, wodurch sich der höhere Druckpegel selbst vom Ausgang trennt und der Anschluss mit niedrigerem Druckpegel mit dem Ausgang verbunden wird. Liegt an keinem der beiden Eingänge Druckluft an, ist auch am Ausgang keine vorhanden. Für binäre Druckluftsignale entspricht das Zweidruckventil einem logischen UND.

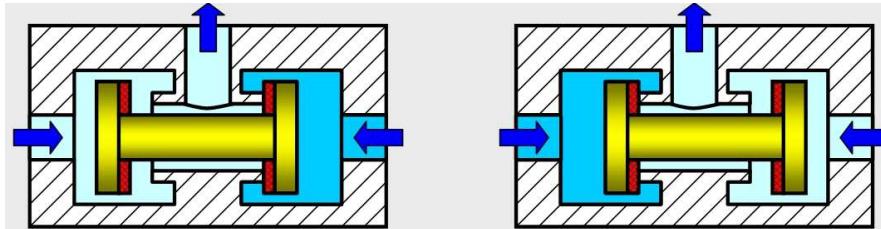


Abbildung 2.13.: Prinzipskizze eines Zweidruckventils.

2.3.5. Wegeventile

Wegeventile können ihre Eingänge abhängig von der Schaltstellung verschiedenen Ausgängen zuordnen, sie schalten, ihrem Namen entsprechend, unterschiedliche Wege. Wie schon in Abschnitt 1.2.3 erwähnt, wird folgendes Benennungsschema verwendet: Ein m/n -Wegeventil besitzt m Fluidanschlüsse und n Schaltstellungen.

In Abb. 2.14 ist beispielhaft ein 5/3-Wegeventil mit elektromagnetischer Betätigung und Rückstellfedern in Schaltsymbol und Skizze dargestellt. Der Anschluss für die Druckluft P wird, abhängig von der Ansteuerung der beiden Magneten A und B, durch Verschiebung des Kolbens auf die beiden Ausgänge A oder B geschalten; der andere Ausgang wird mit TA bzw. TB verbunden. Die Bezifferung der Anschlüsse folgt dabei folgendem Schema: Die Nummer (1) bezeichnet den Druckluftanschluss, gerade Zahlen (2,4,...) werden für Ausgänge (Arbeitsleitungen zu den pneumatischen Aktoren) und ungerade Zahlen (3,5,...) für Auslässe (Entlüftungsleitungen) verwendet. Hierbei ist zu beachten, dass die Nummerierung nur an demjenigen Quadrat gezeichnet wird, welches die Grundstellung des Ventils darstellt. Wie wir später sehen werden, spielen Wegeventile eine wichtige Rolle bei der Ansteuerung von Zylindern.

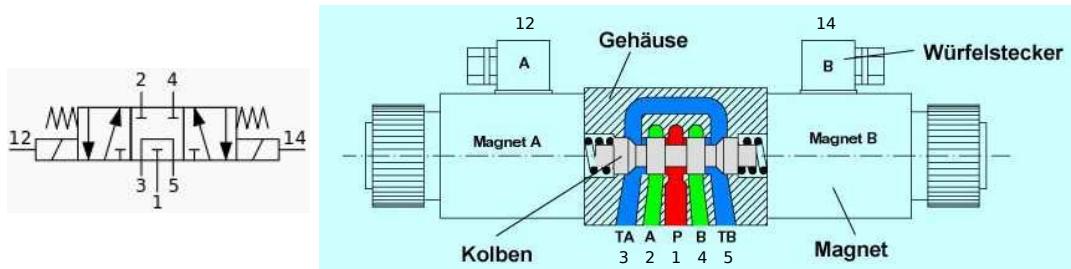


Abbildung 2.14.: Symbol und Skizze eines 5/3-Wegeventils.

Die Abb. 2.15 zeigt zwei Beispiele für die normgerechte Bezeichnung von Wegeventilen.

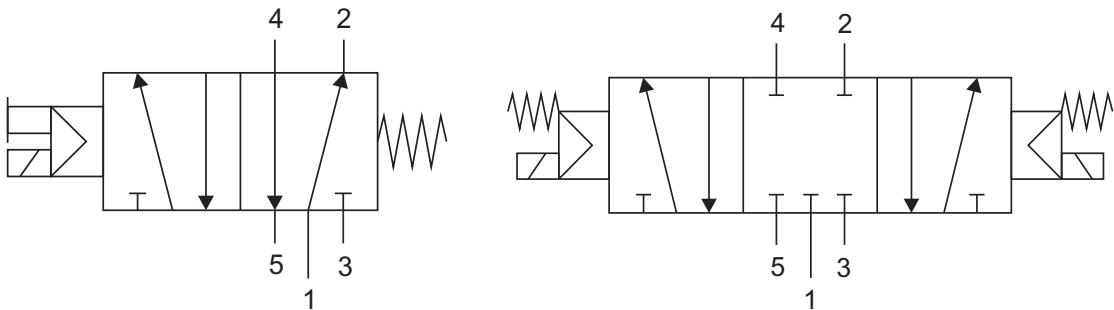


Abbildung 2.15.: Linke Abbildung: 5/2-Wegeventil, Durchfluss von 1 nach 2 und von 4 nach 5, linksseitig kombinierte Muskelkraft- und elektrische Betätigung, rechtsseitig federrückgestellt. Rechte Abbildung: 5/3-Wegeventil, in Mittelstellung gesperrt, beidseitig elektrische Betätigung, beidseitig zurückgestellt über Federn.

2.4. Pneumatische Zylinder und Antriebe

Pneumatische Aktoren wandeln durch thermodynamische Prozesse die in der Druckluft gespeicherte Energie in mechanische Energie um, welche auf den Prozess wirkt. Innerhalb der Fluidtechnik wird auch ganz allgemein von Arbeitsgliedern gesprochen. Wie in Abschnitt 2.1 bei den prinzipiellen Eigenschaften von pneumatischen Systemen erwähnt, ist die Leistungsdichte von pneumatischen Systemen zwar geringer als bei hydraulischen Systemen, aber im Vergleich zu beispielsweise elektrischen Antrieben immer noch deutlich höher. Pneumatische Antriebe lassen sich deshalb sehr kompakt bauen, was gerade bei Handwerkzeugen ein Vorteil ist. Zudem lassen sich kleine und mittlere Massen, typisch für Produktionsanlagen, durch die hohe Dynamik sehr schnell bewegen. Die hohe Überlastsicherheit und das Fehlen von Eigenwärme ist in einigen Anwendungen vorteilhaft.

Aufgrund der Art der ausgeführten Bewegung wird zwischen drei Arten von Antrieben unterschieden: Linearantriebe, Rotationsantriebe und Schwenkantriebe. Erstere führen nur lineare, d. h. geradlinige, Bewegungen aus, wie es z. B. für Druckluftzylinder der Fall ist. Unter Rotationsantriebe versteht man beispielsweise Druckluftmotoren, also Antriebe mit unbegrenzter rotatorischen Bewegungsmöglichkeit. Haben rotatorisch wirkende Antriebe nur einen endlichen Rotationsbereich zwischen zwei definierten Endlagen, bezeichnet man diese als Schwenkantriebe.

Rotationsantriebe auf pneumatischer Basis funktionieren analog zu den in Abschnitt 2.2 dargestellten Verdichtern, kehren den Prozess jedoch um. Die Bauformen sind dementsprechend ähnlich. Ein zum Vielzellen-Rotationsverdichter ähnlicher Antrieb wird bei-

spielsweise als Lamellenmotor bezeichnet, während analog zum Hubkolbenverdichter von Hubkolbenmotoren gesprochen wird. Im Weiteren wird deshalb auf Linear- und Schwenkantriebe eingegangen.

Wie bereits in der Einleitung erwähnt, ist die freie Positionsregelung von pneumatischen Aktoren aufwendig, weshalb diese, anders als hydraulische Aktoren, nur stationär in den Endlagen betrieben werden. Die mathematische Beschreibung dieser Zustände, die Fluidstatik, ist entsprechend einfach.

2.4.1. Einfachwirkende Zylinder

Eine typische Bauform von pneumatischen Zylindern sind einfachwirkende Zylinder, also Zylinder, bei welchen nur eine Zylinderkammer mit Druckluft befüllt werden kann. Dementsprechend lässt sich der Zylinder mittels Druckluft nur in eine Richtung bewegen. Durch eine mechanische Feder wird eine der beiden Endlagen zur Grundstellung, die ohne Druckluftzufuhr angenommen wird. Für die inaktive Zylinderkammer wird in der Regel eine Ent-/Belüftungsbohrung vorgesehen, damit die verdrängte Luft widerstandslos entweichen kann.

Einfachwirkende Zylinder haben konstruktionsbedingt einen geringen Druckluftverbrauch und eine definierte Position im unversorgten Zustand. Zur Steuerung genügt ein einfaches 3/2-Wegeventil. Die im Vergleich zum doppeltwirkenden Zylinder zusätzliche Feder kann jedoch den maximalen Hub des Zylinders einschränken. Zudem kann eine Kraftwirkung nur in eine Richtung aufgebracht werden, da die Rückstellkraft durch die Feder beschränkt ist. Deshalb werden Lastarbeiten nur in dieser Richtung durchgeführt. Zusätzlich ist die wirkende Kraft von der Auslenkung des Zylinders abhängig und die maximale Kraft geringer als bei Bauformen ohne Rückstellfeder.

Die klassische Bauweise einfachwirkender Zylinder sind Kolbenstangenzylinder wie in Abb. 2.16 dargestellt. Der erreichbare Hub ist relativ groß, doch führen die Leckagen an der Zylinderkolbendichtung zu Druckluftströmen im stationären Zustand. Eine alternative Bauweise einfachwirkender Zylinder ist der Membranzylinder wie in Abb. 2.17 dargestellt. Wie schon beim Membranverdichter verschwinden hier die Leckagen und das Aufbringen von konstanten Kräften ist ohne Druckluftverluste möglich. Die Membranbauweise beschränkt den maximalen Hub des Zylinders dramatisch, dafür ist die für den Kraftaufbau zur Verfügung stehende Fläche vergleichsweise groß. Membranzylinder werden dementsprechend in Spann- und Pressanwendungen eingesetzt.

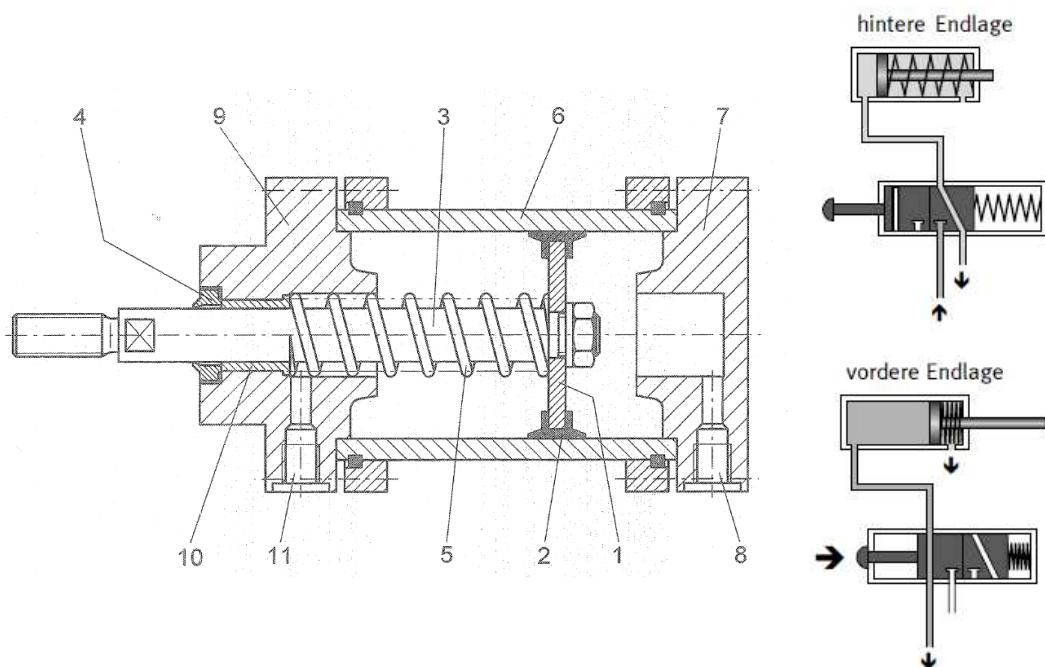


Abbildung 2.16.: Einfachwirkender Kolbenstangenzylinder mit Skizze und beispielhafter Ansteuerung durch ein 3/2-Wegeventil. Zur Skizze: Kolben (1), Kolbendichtung (2), Kolbenstange (3), Kolbenstangendichtung (4), Rückstellfeder (5), Zylinderrohr (6), Zylinderboden (7), Druckluftanschluss (8), Zylinderdeckel (9), Buchse (10) und Auslass (11).

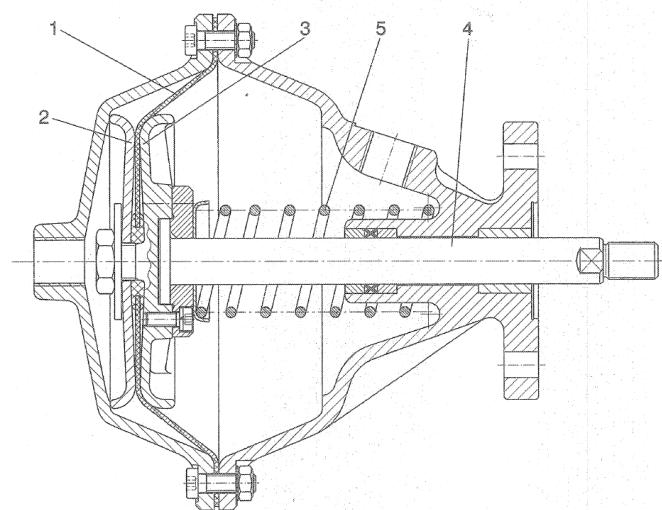


Abbildung 2.17.: Skizze eines einfachwirkenden Membranzylinders mit Membran (1), oberer Membranhalterung (2), unterer Membranhalterung (3), Kolbenstange (4) und Rückstellfeder (5).

2.4.2. Doppeltwirkende Zylinder

Doppeltwirkende (selten auch doppelwirkende) Zylinder, sind noch simpler gebaut als ihre einfachwirkenden Pendants. Die Rückstellfeder entfällt bei dieser Bauform und stellen die verbreitetste Bauform dar. Zwei Druckluftanschlüsse ermöglichen dem doppeltwirkenden Zylinder in beide Richtungen die gewünschte Kraft aufzubringen. Zum Ausfahren wird die hintere Kammer mit Druckluft gefüllt, während die Vordere entlüftet wird. Wird umgekehrt die vordere Kammer be- und die hintere entlüftet, fährt der Zylinder ein.

Die Kraftwirkung von doppeltwirkenden Zylindern ist in beide Richtungen unabhängig von der Kolbenposition und höher als bei einem ansonsten baugleichen einfachwirkenden Zylinder. Da Ein- und Ausfahren Druckluft benötigt, ist der Druckluftbedarf in etwa doppelt so hoch als beim einfachwirkenden Zylinder.

Die häufig verwendete Bauart mit einseitiger Kolbenstange ist in Abb. 2.18 dargestellt. Da die Kolbenstange in dieser Bauart nicht durch beide Zylinderkammern geführt ist, sind die effektiven Zylinderflächen und damit die wirksame Kraft beim Ein- und Ausfahren unterschiedlich. Zur Ansteuerung von doppeltwirkenden Zylindern muss zumindest das etwas komplexere 4/2-Wegeventil verwendet werden, welches die beiden Anschlusspaare auskreuzt.

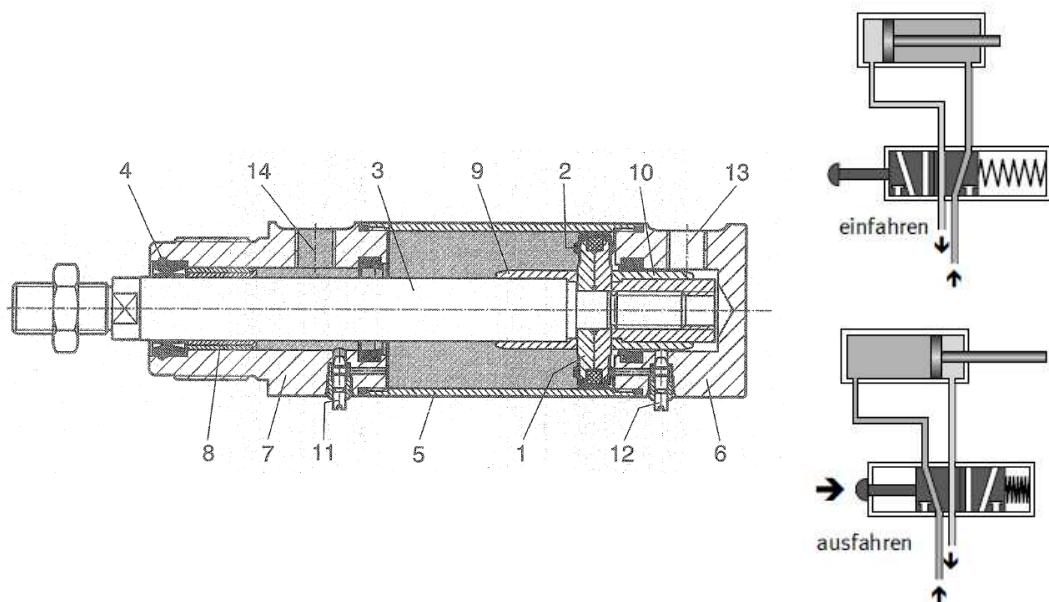


Abbildung 2.18.: Doppeltwirkender Zylinder mit einseitiger Kolbenstange mit Skizze und Schaltung. Zur Skizze: Kolben (1), Kolbendichtung (2), Kolbenstange (3), Kolbenstangendichtung (4), Zylinderrohr (5), Zylinderboden (6), Zylinderdeckel (7), Buchse (8), Dämpfungskolben (9-10), Verstelldrossel (11-12) und Druckluftanschluss (13-14).

2.4.3. Mehrstellungszyylinder

Da pneumatische Zylinder üblicherweise an die Endlagen gefahren werden, ist das Anfahren mehrerer Endpositionen nicht direkt möglich. Durch die Aneinanderreihung mehrerer Zylinder kann dies, wenn auch begrenzt, erreicht werden. Abb. 2.19 zeigt eine Prinzipskizze eines Vierstellungszyinders. Hierfür werden zwei doppeltwirkende Zylinder miteinander verschraubt. Durch die Verwendung von Zylindern mit unterschiedlichen Hüben H_1, H_2 lassen sich so, durch Variation der angefahrenen Endlagen der zwei Zylinder, vier unterschiedliche Stellungen 0, H_1 , H_2 und $H_1 + H_2$ anfahren.

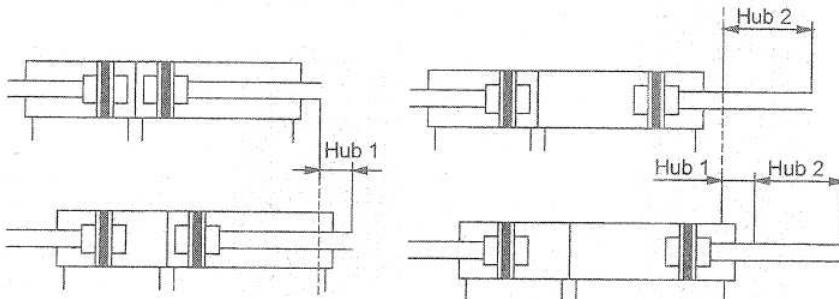


Abbildung 2.19.: Prinzipskizze eines Vierstellungszyinders mit unterschiedlichen Hüben.

2.4.4. Tandemzylinder

Gemäß $F = A_{zyl} \cdot p$ muss zur Erhöhung der Kraft F entweder der Kammerdruck p oder die Zylinderfläche A_{zyl} erhöht werden. Da Ersterer durch den Versorgungsdruck beschränkt ist, bleibt oft nur die Vergrößerung der Zylinderfläche übrig. Will man den Zylinderdurchmesser nicht erhöhen, kann auch durch zwei doppeltwirkende Zylinder hindurch eine einzelne Kolbenstange geführt werden. Dadurch ist die effektive Zylinderfläche $A_{zyl,eff}$ die Summe der beiden einzelnen Zylinderflächen A_{zyl1}, A_{zyl2} oder für beliebig viele zusammengeschaltete Zylinder allgemein:

$$A_{zyl,eff} = \sum_{i=1}^n A_{zyl,i}$$

woraus einfach ersichtlich

$$F = p \sum_{i=1}^n A_{zyl,i}$$

folgt.

Tandemzylinder (fig. 2.20) werden deshalb für hohe benötigte Kräfte eingesetzt, wenn sich aufgrund der Einbausituation der Zylinderdurchmesser nicht erhöhen lässt. Durch Verwendungen mehrerer Stufen kann die maximale Kraft weiter erhöht werden.

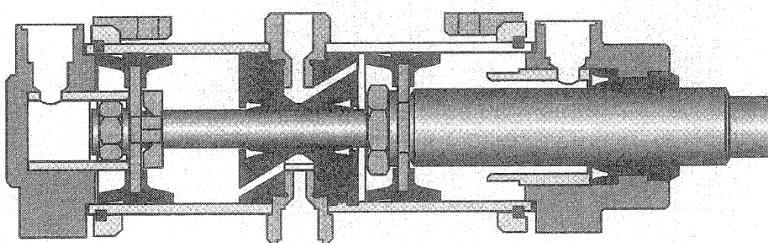


Abbildung 2.20.: Beispiel eines Tandemzylinders.

2.4.5. Schwenkantriebe

Schwenkantriebe dienen dazu, eine Rotation zwischen fest vorgegebenen Endwinkeln auszuführen, eine durchaus typische Aufgabe in Produktionsanlagen. Dazu wird, wie in Abb. 2.21 dargestellt, eine zylinderartige Konstruktion verwendet, welche aus zwei Kolben und einer dazwischen liegenden Zahnstange besteht. Durch ein fest montiertes Ritzel wird diese lineare Bewegung in eine rotatorische Bewegung umgesetzt. Aus der Kombination von Zylindershub und der Zahnanzahl von Zahnstange und -rad ergibt sich der Rotationswinkel. Typische Werte sind 90° , 180° , 270° und 360° , es sind jedoch auch Varianten mit frei einstellbaren Endlagen erhältlich. Das mögliche Drehmoment von Schwenkantrieben bewegt sich zwischen etwa $1 \cdot 10^{-1}$ N m und $1 \cdot 10^3$ N m bei 6 bar.

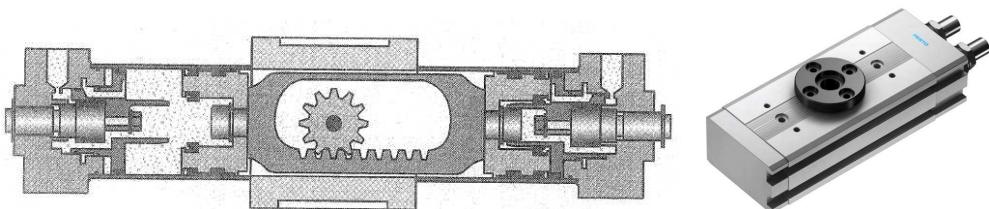


Abbildung 2.21.: Skizze eines Schwenkantriebes beispielhaft anhand einem Produkt der Firma Festo.

2.4.6. Pneumatische Muskel

Eine relativ neue Art von Aktoren sind pneumatische Muskel, deren Membran-Kontraktionsprinzip der Funktionsweise biologischer Muskel nachempfunden ist (siehe Abb. 2.22).

Ein druckfester Schlauch mit eingewobenen Fasern (die Membran) dehnt sich unter Druck in Querrichtung aus, weshalb der Muskel in Längsrichtung kontrahiert. Pneumatische Muskel sind einfachwirkende Aktoren und dürfen auch nur als Zugsystem betrieben werden.

Besonders vorteilhaft sind das hohe Kraft-Gewicht-Verhältnis und die sehr hohen Beschleunigungswerte. Im Unterschied zu Zylindern arbeiten pneumatische Muskeln reibungsfrei, weshalb der sog. Slip-Stick-Effekt nicht auftritt. Dieser bezeichnet die Schwierigkeit, unter Haftreibung kleine Auslenkungen präzise auszuführen. Deshalb wird beispielsweise bei Hydraulikventilen zur Vermeidung von Haftreibung ein kleines Wechselseignal überlagert, um den Ventilkolben in Bewegung zu halten. Dieses Signal wird auch als Dithersignal bezeichnet. Durch die hermetische Abdichtung sind pneumatische Muskeln sowohl für Anwendungen in Lebensmittel- und Pharmaindustrie als auch besonders staub- und schmutzbelastete Umgebungen geeignet. Weiterhin können pneumatische Muskeln kontinuierlich positioniert werden, während Zylinder typischerweise nur an den Endlagen betrieben werden. Nachteilig ist der relativ geringe Hub, welcher sich je nach Kraftanforderungen zwischen 10% und 25% bewegt.

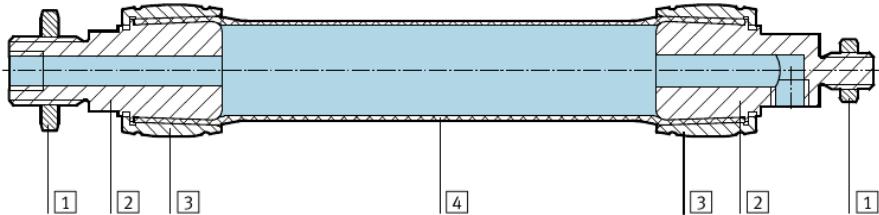


Abbildung 2.22.: Skizze eines pneumatischen Muskels der Firma Festo mit Mutter (1), Flansch (2), Hülse (3) und Membran (4).

Die Steifigkeit des Muskels ist wesentlich geringer als jene von Zylindern, weshalb sie auch als pneumatische Federn mit veränderlicher Federsteifigkeit betrachtet werden können. Abb. 2.23 zeigt einen beispielhaften Zusammenhang zwischen der Kraft F und der prozentualen Auslenkung h für unterschiedliche Betriebsdrücke.

2.5. Pneumatische Sensoren

Zu den weniger bekannten Anwendungen der Pneumatik zählt der Einsatz in Sensoren zur Positionsmessung. Dabei wird meist die Veränderung des freien Fluidstrahls bei Interaktion mit einem Messobjekt zur Erzeugung eines pneumatischen oder elektrischen Messsignals verwendet. Das Sensorsignal ist direkte, also ohne Umwandlung in ein elektrisches Signal, in pneumatischen Steuerungen verwendbar. Überwiegend werden pneu-

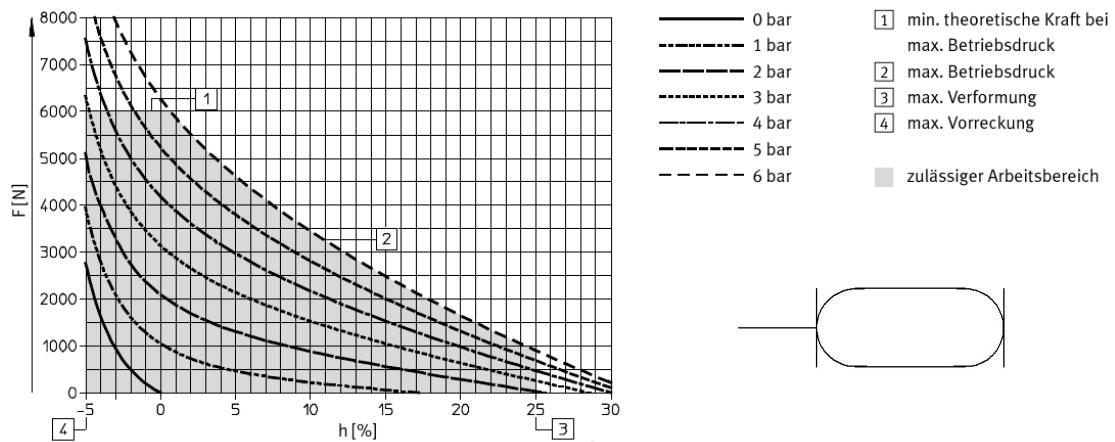


Abbildung 2.23.: Schalsymbol und Kraft-Auslenkungsdiagramm eines pneumatischen Muskels der Firma Festo.

matische Sensoren als berührungslose Näherungsschalter eingesetzt oder sind in Apparaturen zur Werkstückaufnahme integriert. So kann beispielsweise in einem pneumatischen Sauger die Unterbrechung des Volumenstroms als Signal für einen erfolgreichen Ansaugvorgang verwendet werden. Zu den größten Vorteilen pneumatischer Sensoren gehören Berührungsfreiheit, Werkstoffunabhängigkeit, die Unempfindlichkeit gegen magnetische Felder und die rauen Bedingungen, sowie die Anwendungsmöglichkeit in explosionsgefährdeten Umgebungen. Zudem sind pneumatische Sensoren meist sehr kompakt zu bauen und können so auch kleine Objekte erkennen. Weiterhin werden oft große mögliche Tastabstände und große zulässige Temperaturbereiche als Vorteile genannt.

2.5.1. Staudrucksensor

Staudrucksensoren arbeiten nach dem Düse-Prallplatte-Prinzip wie in Abb. 2.24 dargestellt. Eine Strahldüse mit kegelförmiger Spitze wird mit einer in 90° stehenden Bohrung zum Abgreifen des Messdrucks p_2 ausgestattet. Der an der Düse austretende Fluidstrahl wird durch das Messobjekt je nach Abstand zur Düse s zunehmend aufgestaut, wodurch sich der Messdruck p_2 erhöht. Wird die Düse vollständig verschlossen, stellt sich am Ausgang der Versorgungsdruck p_1 ein. Entfernt man das Messobjekt vollständig, entsteht ein Unterdruck am Ausgang. Zum Aufbau eines Staudrucks müssen die Objektabstände kleiner als 20 % des Düsendurchmessers sein, weshalb übliche Messbereiche bei einigen wenigen Millimetern liegen.

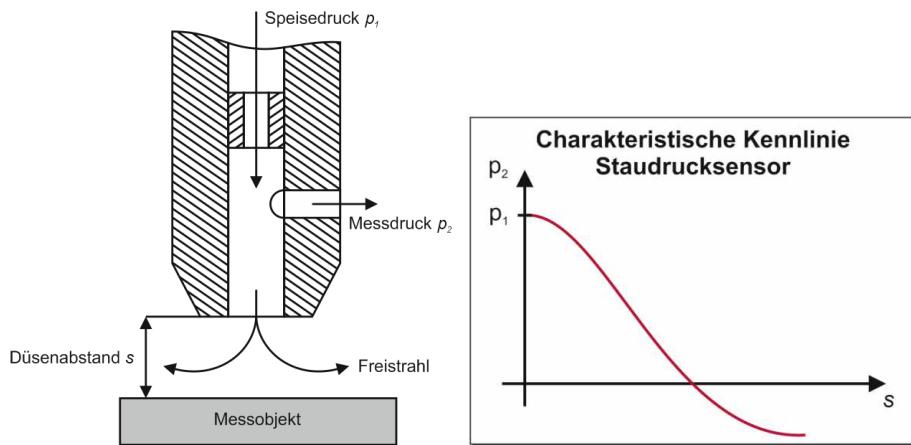


Abbildung 2.24.: Skizze und Kennlinie eines Staudrucksensors.

2.5.2. Ringstrahlsensor (Reflexdüse)

Problematisch am Staudrucksensor ist der geringe Messbereich, da der Staueffekt erst sehr spät einsetzt. Ringstrahlsensoren überwinden dieses Problem durch Verwendung eines Strahlmantels. Wie in Abb. 2.25 dargestellt ist die Messöffnung von einem ringförmigen Auslass eingeschlossen. Nähert sich ein Objekt der Ringstrahldüse, wird ein Teil des an der Ringdüse austretenden Strahls umgelenkt und im Inneren aufgestaut, wodurch sich der Messdruck p_2 im zentralen Ausgangskanal erhöht. Dadurch lassen sich Messbereiche bis 10 mm realisieren.

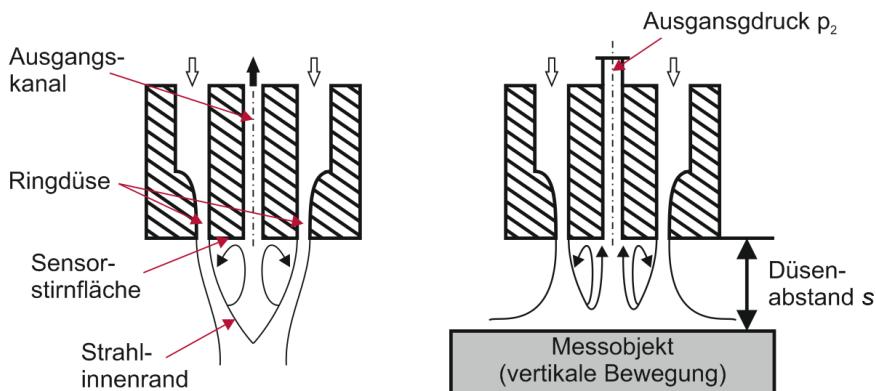


Abbildung 2.25.: Skizze eines Ringstrahlsensors.

2.5.3. Luftstrahlschranke

Luftstrahlschranken erkennen, analog zu den bekannteren Lichtstrahlschranken, die Unterbrechung des erzeugten Luftstroms. Als Messsignal steht der Druck am Empfänger zu Verfügung. Wird der Luftstrom unterbrochen, fällt der Druck zwangsweise ab. Mit der erhaltenen Messinformation kann beispielsweise direkt ein Wegeventil geschaltet werden (siehe Abb. 2.26). Luftschränken werden über Niederdruckregler mit ca. 0,1 bar–2,5 bar versorgt. Der Abstand zwischen Sender und Empfänger ist kleiner 100 mm.

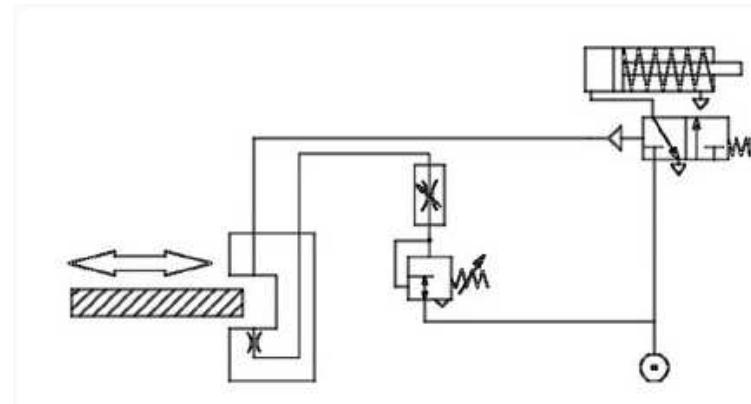


Abbildung 2.26.: Anwendungsbeispiel einer Luftstrahlschranke zur Betätigung eines Zylinders.

3. Steuerungstechnik

Nachdem in den letzten Kapitel ein Überblick über Prozessleitsysteme sowie typische Komponenten zur Realisierung solcher hierarchischer Leittopologien geschaffen wurde, werden in diesem Kapitel die Themen der Steuerungstechnik, wie der Entwurf und Realisierungsmöglichkeiten von industriellen Steuerungen, behandelt.

3.1. Einführung

Das primäre Ziel der Steuerungs- und Regelungstechnik ist die gezielte Einflussnahme auf einen Prozess um sicherzustellen, dass sowohl das statische als auch dynamische Verhalten der jeweils gewünschten Aufgabenstellung entspricht. Aus informationstheoretischer Sicht bedeutet dies also einen direkten oder indirekten Informationsaustausch mit dem Prozess. Zunächst müssen Informationen über den aktuellen Zustand über Sensoren des zu steuernden oder regelnden Systems erfasst und der Steuer- oder Regeleinrichtung zugeführt werden. Aufgrund dieser Prozessinformationen werden über die Steuer- oder Regeleinrichtung Entscheidungen über Aktionen getroffen. Im Falle einer Steuerungseinrichtung (z. B. einer klassischen SPS) könnte dies eine logische Verknüpfung von logischen Eingangssignalen sein. Nach der Verarbeitung der Prozessinformationen werden die getroffenen Entscheidungen über dedizierte Kommunikationskanäle (vgl. Kapitel 5) an die installierten Aktoren weitergeleitet. Die Verarbeitung der Informationen kann dabei zyklisch (z. B. wie bei der klassischen SPS) oder bedarfsgesteuert (z. B. bei verteilten Systemen mit ereignisorientiertem Abarbeitungsprinzip) geschehen.

3.2. Verbindungsprogrammierte Steuerungen

Typische steuerungstechnische Programme bestehen aus Funktionsgliedern und deren Verbindungen. Werden die Funktionsglieder hardwaremäßig ausgeführt und die Verbindung eben dieser Funktionsglieder mechanisch ausgeführt, um ein Steuerungsprogramm zu realisieren, so liegt eine sogenannte Verbindungsprogrammierte Steuerung (VPS) vor. Neben der mechanischen Ausführung solcher VPS können solche Steuerungen auch hy-

draulisch, (elektro-)pneumatisch, elektromechanisch (z. B. kontaktgebend durch Relais) oder elektronisch (Schaltkreise) ausgeführt werden. Ein großer Nachteil von VPS besteht im hohen Aufwand bei gewünschten Programmänderungen, da diese z. B. bei mechanischer Ausführung zur Umverdrahtung führen.

3.3. Speicherprogrammierbare Steuerungen

Betriebsweite Vernetzung, konsistente Datenverwaltung und Echtzeitaspekte spielen in der Planungs- und Ausbauphase moderner Produktionsbetriebe eine zentrale Rolle (Stichwort: Computer-Integrated Manufacturing). Längst ist aus der klassischen Speicherprogrammierbare Steuerung (SPS) ein integraler Bestandteil betriebsinterner Netzwerke geworden, die an verfügbaren Kommunikationsinterfaces, Befehlssatz und Rechenleistung modernen PCs kaum nachstehen. Damit ist auch die Aufgabenteilung zwischen PCs, Industrie-PCs (IPCs) (robuste PCs mit Echtzeitbetriebssystem und Steuerhardware) und SPSen nicht mehr allgemein gültig definierbar.

3.3.1. Anforderungen und Auswahlkriterien

Mit SPSen werden typischerweise Eingangsbedingungen (Digitaleingänge, Analogeingänge, Daten externer Geräte) mit momentanen Betriebszuständen (Bit-Merker, Zählerstände oder Zeitbedingungen) zu Ausgangszuständen (Digitalausgänge, Analogausgänge, Daten externer Geräte) verknüpft. Dabei ist zu beachten, dass vielfach Sensoren, Aktoren und Daten örtlich nicht an den Aufstellungsort einer SPS gebunden sind. Eine weitere wichtige Funktion besteht in der Bereitstellung von Informationen für eine Betriebsdatenerfassung (BDE), die mit einer Vernetzung einhergeht. Für die Auswahl geeigneter SPSen sind aus Sicht der Leistungsfähigkeit folgende technische Rahmenbedingungen zu beachten:

- Anzahl und Art der ansteuerbaren Ein- und Ausgänge
- Speicherbedarf Daten/Programm
- Verarbeitungsgeschwindigkeit
- Kommunikationsschnittstellen und deren Vernetzung
- Betriebsumgebung: Robustheit und Sicherheit

Darüber hinaus sind, abhängig vom jeweiligen Anwendungsfall strategische Rahmenbe-

dingungen zu berücksichtigen, unter anderem die

- Kompatibilität des Systems zu bestehender Hardware/Software
- Bestehendes Know-how (vorhandene Programmierkenntnisse etc.)
- Wartbarkeit (Zugriffsrechte und -möglichkeiten von Personen)

3.3.2. Kompatibilität

Im Gegensatz zur extremen Kurzlebigkeit im Bereich der PC–Technologie, wird bei SPS–Architekturen auf langfristige Unterstützung eingeführter Systemkomponenten und Programmiertechniken gesetzt. Den Vorteilen einer einfachen Wartbarkeit, niedrigen Schulungskosten nach einmaliger Einlernphase und Portabilität bestehender Software stehen dabei oftmals konzeptionelle Schwächen und inkonsistente Systemerweiterungen gegenüber. Ein betriebsweiter Übergang auf neue Gerätegenerationen wäre ohne die Bereitstellung von Interimslösungen vielfach undenkbar. Durch Abwärtskompatibilität (Prozessor oder Emulation) zu älteren Produkten werden teilweise Umrüstzeiten (Produktionsstillstand!) im Stundenzirkel möglich. Teile der Software können anschließend sukzessive durch neue Programmmoduln ergänzt oder ersetzt werden.

3.3.3. Wartbarkeit und Erweiterbarkeit

Vor Ankauf einer SPS ist zunächst zu prüfen, inwieweit mögliche künftige Ergänzungen zu berücksichtigen sind. Gefordert ist also, einen möglichst günstigen Kompromiss zwischen Leistungsfähigkeit (Programm- und Datenspeicher, Zykluszeit, max. I/O digital, analog), Komplexität (Installations- und Einschulungsaufwand!) und Systemkosten zu finden.

Außerdem ist zu untersuchen, ob bei Ausfall einer Komponente eine Abschaltung der kompletten Steuerung zulässig ist, oder ob fehlerhafte Teile im Betrieb tauschbar sind (**hot plugable**). Dazu müssen Mechanismen bereitgestellt werden, die eine Visualisierung bzw. eine direkte Prozessbeeinflussung durch manuellen Eingriff erlauben.

Mit der Methode des **Forcings** können der SPS einerseits gezielt Eingangszustände vorgetäuscht werden (z. B. Tausch eines Sensors während des Betriebes), auf der anderen Seite kann durch Forcing eines Ausgangs ein Ausgangszustand festgelegt werden, der bis zur Deaktivierung dieser Funktion den berechneten Ausgangszustand ersetzt (z. B. Abschalten eines Lüfters während Servicearbeiten).

Eine über das Forcing hinausgehende Funktion ist die **Online-Programmierung**, eine Methode, bei der die Verknüpfungslogik im laufenden Betrieb verändert werden kann. Diese Art zu programmieren steht im Gegensatz zur **Offline-Programmierung**, bei dem zunächst unabhängig von der SPS ein Programm erstellt wird und anschließend im sogenannten **Download** auf die Steuerung übertragen wird. Beim **Upload** erfolgt eine Übertragung des Programms von der SPS zum Programmiergerät.

Im Allgemeinen bestehen also umfangreiche Möglichkeiten, einen gesteuerten Prozess durch Bedienfehler gefährlich zu beeinflussen. Damit ist auch zu prüfen, welche innerbetrieblichen Sicherheitsvorkehrungen zu treffen sind. Moderne Entwicklungssysteme und SPS-Architekturen (**Protected Processors**) erlauben die Verwaltung mehrerer Benutzergruppen mit unterschiedlichen Rechten (z. B. Abteilung Technik: voller Zugriff, Abteilung Service: nur Upload und Offline-Programmierung, ...).

3.3.4. Kommunikationsschnittstellen

Herkömmliche Topologien basieren noch auf einer Unterteilung zwischen

- Leitebene (oder Informationsebene)
- Steuerungsebene (oder Kontroll-/Automatisierungsebene)
- Feldebene (oder Einzelgeräteebene bzw. Device-Ebene)

PCs und Workstations werden dabei der Informationsebene, SPSen der Kontrollebene und die von ihnen angesteuerten Komponenten der Einzelgeräteebene zugeordnet.

Demgegenüber zeigt Abb. 3.1 eine Architektur, die zunehmend an Bedeutung gewinnt. Dabei wird zwischen Informationsebene und Kontrollebene nicht mehr unterschieden, SPSen werden direkt in ein Ethernet-Netzwerk eingebunden. Auf Einzelgeräteebene werden Sensoren, Aktoren, aber auch komplette Ein- und Ausgabebaugruppen und intelligente Subsysteme über einen offenen, nicht herstellerspezifischen Feldbus mit der übergeordneten SPS verbunden. Nach zahlreichen firmenspezifischen Lösungen (Data-Highway+, Remote I/O, usw.) besteht damit erstmals die Möglichkeit einer weitgehend herstellerunabhängigen Kombination von Systemen und Endgeräten. Beispiele für solche sind: Interbus, DeviceNet, Profibus, usw. Der Trend ist sogar dahingehend auch Geräte in der Device-Ebene an das Ethernet zu hängen, wodurch auch die Informations- und Einzelgeräteebene zusammenwächst.

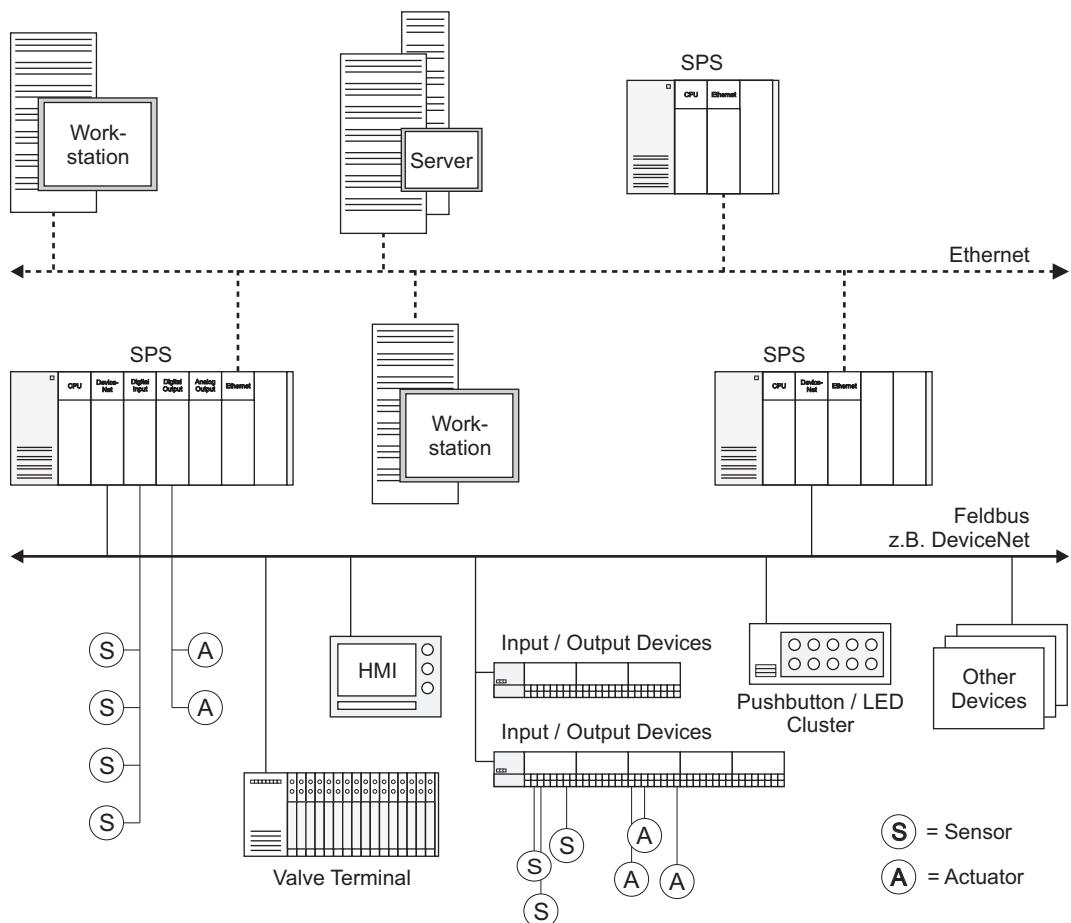


Abbildung 3.1.: Netzwerkanbindung von SPSen

3.3.5. Abarbeitungsmodell

Historisch gesehen ist die SPS aus dem Bedürfnis entstanden, verdrahtete Relaislogik durch eine programmierbare Hardware zu ersetzen. Im Gegensatz zu Relaissteuerungen erfolgt die Abarbeitung der Schaltungslogik im wesentlichen sequentiell bzw. nur quasi parallel, d. h. die Funktion wird auf ein Programm abgebildet und in einem Prozessor schrittweise ausgeführt. Die Verarbeitungsgeschwindigkeit hängt neben weiteren Faktoren also unmittelbar von der Komplexität der Verknüpfungslogik ab und wird typischerweise in „ms/1k bit Instructions“ angegeben. Das Abarbeitungsprinzip erfolgt dabei zyklisch mit den in Abb. 3.2 dargestellten Schritten.

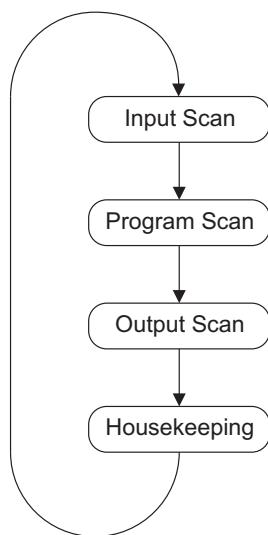


Abbildung 3.2.: Bearbeitungszyklus einer SPS

Der erste Schritt im SPS-Zyklus stellt das Einlesen aller Eingangswerte in das sogenannte Prozessabbild der Eingänge (PAE) dar. Für alle weiteren Schritte in einem Zyklus stehen somit alle Prozesswerte zu einem definierten Zeitpunkt über das PAE zur Verfügung. Durch einen einmaligen Programmdurchlauf werden dann die internen Zustände sowie die Ausgangswerte ermittelt und diese sukzessive in das Prozessabbild der Ausgänge (PAA) geschrieben. Erst nach Erreichen des Programmendes wird das PAA an die Peripheriebaugruppen bzw. integrierten Ein-/Ausgänge übertragen. Zum Schluss eines SPS-Zyklus folgt eine Housekeeping-Phase, in der interne Verwaltungsaufgaben ausgeführt werden. Diese letzte Phase kann z. B. genutzt werden, um mit einem Programmiergerät zu kommunizieren und während des Betriebs Programmänderungen einzuspielen.

Besteht die Notwendigkeit auf bestimmte Ereignisse rascher als der gesamte Programm-

durchlauf es erlaubt zu reagieren, so sind unterschiedliche Lösungsansätze denkbar:

- Bei Programmbeginn werden zeitkritische Zustände abgefragt, bei Bedarf werden nur jene Programmteile ausgeführt, die für die Abarbeitung dieser Zustandsänderungen relevant sind.
- Einsatz von intelligenten Subsystemen (zusätzlicher Hardware-, Kosten- und Konfigurationsaufwand)
- Verwendung von Interrupt-Routinen
- Einsatz multitaskingfähiger Steuerungen, die vom Konzept in Abb. 3.2 abweichen; damit können für unterschiedliche Prozesse verschiedene Prioritäten festgelegt und so die Verarbeitungsgeschwindigkeit optimiert werden.

Alle genannten Punkte sind nur unter bestimmten Voraussetzungen sinnvoll anwendbar. Weitere Konzepte erlauben die Unterteilung des in Abb. 3.2 dargestellten Zyklus in mehrere Teilschritte. Die Abarbeitung lautet dann

- Input Scan #1
- Program Scan #1
- Output Scan #1
- ...
- Input Scan #n
- Program Scan #n
- Output Scan #n
- House Keeping

Eine Verwendung dieses I/O orientierten Zeitscheiben-Verfahren (Time-Slicing) kann für bestimmte Applikationen Vorteile bringen.

3.3.6. Robustheit

Ein wesentliches Merkmal das SPSEN auszeichnet ist deren robuster Aufbau. Auf bewegliche Teile wie Laufwerke, Lüfter etc. wird verzichtet, Module sind formschlüssig arretierbar. Im industriellen Umfeld herrschende Bedingungen wie Verschmutzung durch

Staub, Öl, schwankende Luftfeuchtigkeit und Temperatur, aber auch Vibratoren und Spannungsschwankungen im Versorgungsnetz dürfen die Funktion einer SPS nicht beeinflussen. Dazu werden SPSen den jeweiligen Betriebsbedingungen entsprechend in staubdichte Schaltschränke eingebaut, Störspannungen müssen gegebenenfalls durch Filter abgefangen werden. Für die Steuerung sicherheitskritischer Abläufe kommen entweder der Einsatz redundanter Zusatzsysteme oder spezielle Sicherheits-SPSen in Frage.

3.3.7. Modellierung nach dem Standard IEC 61131-3

Der internationale Standard IEC 61131-3[3] definiert insgesamt fünf verschiedene Sprachen zur Vereinheitlichung der Programmierung von SPSen. Diese lassen sich in textbasierte sowie in grafische Sprachen untergliedern. Die Gruppe der textbasierten Sprachen umfasst die Anweisungsliste (engl. Instruction List) und Strukturierter Text (engl. Structured Text). Die Funktionsbausteinsprache (engl. Function Block Diagram) sowie der Kontaktplan (engl. Ladder Diagram) zählen zu den grafischen Sprachen. Eine Mischform bildet die Ablaufsprache (engl. Sequential Function Chart), da z. B. die Transitionsbedingungen sowohl in einer textuellen als auch in einer grafischen Sprache implementiert werden können.

3.3.7.1. Anweisungsliste

Bei der Anweisungsliste (AWL) handelt es sich um eine textbasierte, zeilenorientierte und maschinennahe Programmiersprache deren Struktur an Assemblersprache erinnert.

3.3.7.2. Strukturierter Text

Der Strukturierte Text (ST) ist den höheren Programmiersprachen wie Pascal oder C ähnlich. Er bietet umfassende syntaktische Elemente, die ideal sind um komplexe rechnerische Aufgaben, Algorithmen sowie entscheidungstreffende Aufgaben (IF-THEN-ELSE) darzustellen. Das Programm besteht aus einer Reihenfolge von Anweisungen, die durch Semikolons getrennt sind. Die Zuweisung erfolgt mit dem Zeichen `:=`, Kommentare stehen zwischen den Zeichen `(*` und `*)`. Die Operatoren der Programmiersprache ST sind in der Tabelle 3.1 aufgelistet.

Der Strukturierte Text verfügt über eine weite Palette von Befehlen und Anweisungen, beginnend mit der Anweisung zum Aufruf von Funktionsbausteinen, dazu kommen bedingte Anweisungen wie

Operator	Beschreibung	Priorität
(...)	Klammer	höchste
Funktionsnahme (...)	Parameterliste einer Funktion	
**	Potenzierung	
-	Negation	
NOT	Boolesches Komplement	
	Vorzeichen	
*	Multiplikation	
/	Division	
MOD	Modulo	
+	Additon	
-	Subtraktion	
<,>,<=,>=	Vergleichsoperator	
=, <>		
AND , &	Boolesches UND	
XOR	Boolesches Exklusiv-OR	
OR	Boolesches OR	niedrigste

Tabelle 3.1.: Verwendete Operatoren in ST.

- IF...ELSE...THEN,
- CASE,

sowie Wiederholungsanweisungen

- FOR...DO,
- WHILE...DO und
- REPEAT...UNTIL.

3.3.7.3. Funktionsbausteinsprache

Neben den bereits vorgestellten textuellen Programmiersprachen standardisiert die IEC 61131-3 auch grafische Sprachen, wie beispielsweise die Funktionsbausteinsprache (FBS). Ihr Hauptprogrammierelement ist der Funktionsbaustein (FB), eine grafisch als Block repräsentierte Softwareeinheit. Ein FB kann einfache digitale Verknüpfungslogik (UND-/ODER-/ XOR-Gatter) repräsentieren, aber auch komplexe Regler, deren Parametrierung meist über die Eingänge des FBs durchgeführt. Prinzipiell sind FB durch den Applikationsingenieur in einer der fünf IEC 61131-3 Sprachen frei programmierbar. Die Ein- und Ausgänge eines FB stellen die Schnittstellen zu anderen FBs oder dem Prozessabbild dar. Die FBS ist besonders für Verknüpfungssteuerungen geeignet und durch die grafische Repräsentation sehr einfach verständlich.

3.3.7.4. Kontaktplan

Wie bereits an früherer Stelle erwähnt, war die Speicherprogrammierbare Steuerung ursprünglich als Ersatz einer verdrahteten Logik gedacht. Daraus hat sich die auch heute noch weltweit (aber besonders im amerikanischen Raum) meistverbreitete Programmiertechnik für Speicherprogrammierbare Steuerungen entwickelt: Der Kontaktplan (KOP, engl.: Ladder Logic). Der Betriebselektriker früherer Relaissteuerungen war mehr mit Begriffen, wie Öffner, Schliesser und Reihen- bzw. Parallelschaltung vertraut, als mit den Begriffen der Digitaltechnik, wie UND, ODER, NICHT. Während der fortschreitenden Entwicklung der SPSen, wurden jedoch immer weitere Methoden höherer Programmiersprachen, wie Schleifen und Funktionen in die SPSen integriert, sodass es spätestens dann zu arger Konfusion kommt, wenn Schleifen und Sprungbefehle zwischen Masse und 24 Volt gelegt werden (den vertikalen Leitern des KOPs). Neben diesen eher unüblichen Auswüchsen weist der Kontaktplan aber auch einige Vorteile gegenüber einer textuellen Schaltungsbeschreibung auf. So können beispielsweise momentane äußere und innere

Zustände der Steuerung bei entsprechend niedriger Systemdynamik am Bildschirm unmittelbar beobachtet werden.

Der Kontaktplan besteht aus zwei vertikalen Leitern – der Energieversorgung (engl. Rails) – und beliebig vielen horizontalen Leitern – den Schaltkreisen – welche als Stränge (engl. Rungs) bezeichnet werden. Die Schaltkreise werden durch unterschiedliche Kontakte, welche auf der linken Strangseite liegen, geschlossen bzw. unterbrochen, wodurch ein oder mehrere Verbraucher, welche auf der rechten Strangseite liegen, „energetisiert“ bzw. „ent-energetisiert“ werden. Horizontal angeordnete Komponenten sind seriell geschaltet, vertikal untereinander angeordnete parallel. Findet sich ein geschlossener Strompfad durch die Kontakte auf der linken Seite, so werden die Verbraucher auf der rechten Seite aktiviert. Die Abb. 3.3 zeigt einen einfachen elektrischen Kontaktplan bestehend aus einem einzigen Strang, um eine Lampe zu schalten. Die Lampe wird beim Betätigen des Tasters 1 oder des Tasters 2 leuchten, wenn der Not-Aus Schalter nicht gedrückt ist. Man beachte, dass der Normalzustand (Ruhezustand) für den Taster 1 und 2 der Nicht-Betätigten ist („Schliesser“), während der Normalzustand des Not-Aus Schalters der Betätigten ist („Öffner“). Eine praktische Bedeutung hat dieses Beispiel jedoch nicht, da die Lampe nur leuchtet so lange Taster 1 oder Taster 2 auch tatsächlich gedrückt gehalten werden und erlischt beim Auslassen der Tasten wieder.

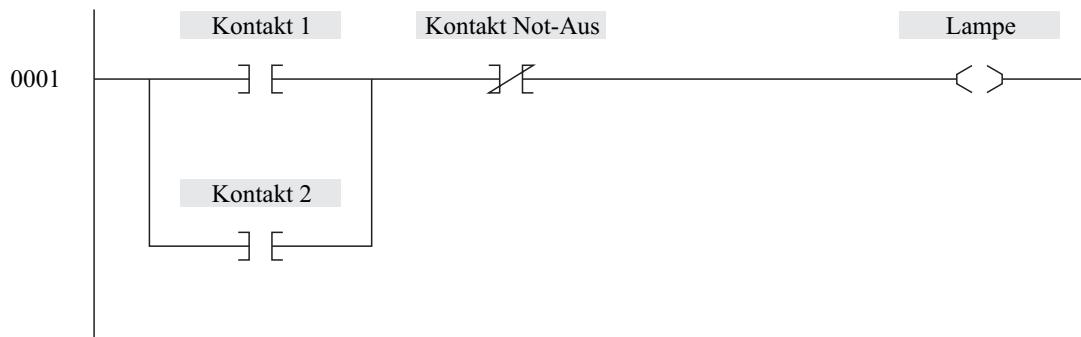


Abbildung 3.3.: Kontaktplan Programm für das Schalten einer Lampe

In SPSEN werden die Ein- und Ausgänge auf Register abgebildet. Für die Implementierung des Kontaktplans als Computerprogrammiersprache gilt: Die Kontakte sind Abfragen nach dem Status von Registern, sogenannte Eingangsanweisungen (engl. Input Instructions) und die Verbraucher, sogenannte Ausgangsanweisungen (engl. Output Instructions) sind im einfachen Fall das Setzen und Rücksetzen von Registern oder komplexe Funktionsblöcke, wie z. B. Zähler, Timer, Unterprogramme und viele mehr, welche im Laufe der Jahre der Programmiersprache hinzugefügt wurden. Die Funktion dieser Blöcke kann nicht mehr grafisch repräsentativ dargestellt werden, wodurch besonders bei häufiger Verwendung dieser Funktionsblöcke die Lesbarkeit der Programme leidet. Es gilt weiterhin, dass die Eingangsanweisungen links und die Ausgangsanweisungen

rechts im Strang stehen.

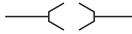
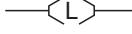
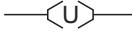
Als Kontaktplan wird im weiteren Verlauf dieses Skriptums – im Gegensatz zu der fest verdrahteten Steuerung – das Softwareprogramm bzw. dessen Quellcode bezeichnet. Der Kontaktplan stellt in der SPS somit lediglich die logische Verknüpfung zwischen den Speicher-, den Eingangs- und den Ausgangsregistern dar. Er lässt keine Rückschlüsse auf die Außenbeschaltung der SPS zu. Die meisten SPSEN unterstützen jedoch die Benennung von Registern und das Kommentieren von Ein- und Ausgangsanweisungen und von Strängen, wodurch das Programm übersichtlich und lesbar bleibt. Auch für die Laborübung empfehlen wir dringend, sich diese Möglichkeit zu Nutzen zu machen.

Sowohl die Eingangs-, als auch die Ausgangsanweisungen erwarten im Allgemeinen einen Operanden, mit welchem die gewünschte Funktion durchzuführen ist. Meistens ist der Operand ein Bitregister, jedoch für die komplexeren Ausgangsanweisungen können es auch Byte- oder Wortregister sein bzw. verlangen manche überhaupt nach mehreren Operanden.

Eingangsanweisungen In diesem Abschnitt werden die wichtigsten Eingangsanweisungen mit Bitoperanden vorgestellt.

Symbol	Beschreibung
	Examine if Closed („Schliesser“): Liefert Wahr, wenn Operand Wahr, ansonsten Falsch.
	Examine if Open („Öffner“): Liefert Wahr, wenn Operand Falsch, ansonsten Falsch.
-[OSR]-	One Shot Rising: Liefert einmalig Wahr, wenn Operand von Falsch auf Wahr wechselt, ansonsten Falsch.
-[OSF]-	One Shot Falling: Liefert einmalig Wahr, wenn Operand von Wahr auf Falsch wechselt, ansonsten Falsch.

Ausgangsanweisungen In diesem Abschnitt werden die wichtigsten Ausgangsanweisungen mit Bitoperanden vorgestellt.

Symbol	Beschreibung
	Output Energize: Setzt den Operanden auf Wahr, wenn der Strang „stromdurchflossen“ ist, ansonsten setzt er ihn auf Falsch. Der Operand zeigt somit direkt an, ob es einen gültigen „Strompfad“ durch die Eingangsanweisungen gibt.
	Output Latch: Setzt den Operanden auf Wahr, wenn der Strang „stromdurchflossen“ ist. Der Operand bleibt Wahr, auch wenn der Strang wieder „stromlos“ wird.
	Output Unlatch: Setzt den Operanden auf Falsch, wenn der Strang „stromdurchflossen“ ist. Der Operand bleibt Falsch, auch wenn der Strang „stromlos“ wird.

3.3.7.5. Ablaufsprache

Die Ablaufsprache basiert auf der gleichen Methode wie die in der Vorlesung behandelten Petri-Netze. Grundidee ist es, ein komplexes Programm in kleinere logische Einheiten aufzuteilen und die Ablaufsteuerung zwischen diesen Einheiten zu beschreiben und darzustellen. Die Ablaufsprache ermöglicht die Darstellung sowohl von sequentiellen als auch von parallelen Prozessen. Sie beschreibt und definiert zeit- und ereignisorientierte Steuerungssequenzen. Es handelt sich dabei um keine vollständige Sprache da sie Instruktionen aus anderen Sprachen erlaubt bzw. benötigt (z. B. zur logischen Verknüpfung von Eingangswerten zu einer Transitionsbedingung).

Die drei grundlegenden Elemente der Ablaufsprache sind Schritte, Aktionen innerhalb der Schritte und Transitionen zwischen den Schritten. Ein Programm, das ein SFC repräsentiert, besteht somit aus einer endlichen Menge an Schritten und Transitionen und ist so aufgebaut, dass jedem Schritt immer eine Transition folgen muss bzw. umgekehrt jeder Transition muss immer ein Schritt folgen. Ein Beispiel eines SFCs ist in Abb. 3.4 dargestellt.

Ein Schritt stellt eine Steuerungseinheit dar. Er beschreibt einen bestimmten Systemzustand, der zu kontrollieren ist. Es wird zwischen normalen Schritten und dem Initialschritt unterschieden. Es kann nur einen Initialschritt in einer SFC-Sequenz geben, wobei dieser immer als erster beim Sequenzstart aktiviert wird. Jedem Schritt sind zwei Variablen zugeordnet, die Schrittsynchronisation und -überwachung ermöglichen.

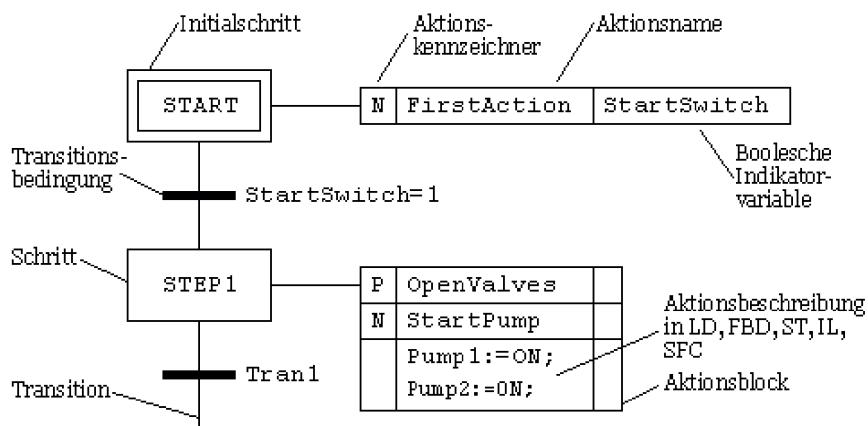


Abbildung 3.4.: Komponenten und Aufbau eines Sequential Function Charts (laut IEC 61131-3).

1. Die erste Variable ist die boolesche Variable *Step Activ Flag*, auch Token genannt (*<SchrittName>.X*), die den aktiven Schritt markiert. Sie hat den Wert *TRUE* solange der Schritt aktiv ist.
2. Die zweite Variable ist die Laufzeitvariable (*<SchrittName>.T*) vom Typ *TI-ME[ms]*. Wenn der Schritt aktiv wird, wird ein Timer *<SchrittName>.T* gestartet und kann jederzeit abgefragt werden (wird ein Schritt wiederholt aktiv, so wird der Zähler jedes Mal auf Null zurückgesetzt).

Eine Transition ist genau definiert durch die Übergangsbedingung, wobei diese einen boolescher Ausdruck repräsentiert, der in IL, ST, LD oder FBD beschrieben werden kann. Ist das Resultat der Übergangsbedingung *TRUE*, so wird der derzeitige Schritt inaktiv und der nächste Schritt wird aktiviert. Ein Schritt kann keine, eine oder mehrere Aktionen haben. Eine Aktion definiert, wie sich das System zu verhalten hat, wenn der zugehörige Schritt aktiv wird (z. B. Pumpe starten, Ventil öffnen). Ist ein Schritt aktiv, so wird die verbundene Aktion auf jeden Fall ausgeführt, auch wenn die nachfolgende Transition bereits aktiv wird. Bei einem Schritt ohne Aktion wartet das System einfach, bis die nächste Übergangsbedingung *TRUE* wird. Damit kann auf bestimmte Ereignisse gewartet werden. Eine Aktion besteht aus Instruktionen, die in IL, ST, LD, FBD oder SFC geschrieben sind. Eine boolesche Indikatorvariable ist wahlweise einem Aktionsblock als Kommentar zugewiesen. Sie zeigt an, dass die Aktion durchgeführt wurde. Die Ausführung einer Aktion wird durch einen ihr zugeordneten sog. Aktionskennzeichner gesteuert (siehe unten). Ein Aktionskennzeichner (action qualifier) ist immer einer Aktion zugeordnet. Er steuert die Aktion so, dass abhängig von dem ausgewählten Aktionskennzeichner der

Ausführungszeitpunkt und/oder die Ausführungszeitspanne der Aktion definiert ist. Die wichtigsten Aktionskennzeichner nach dem Standard IEC 61131-3 sind die folgende:

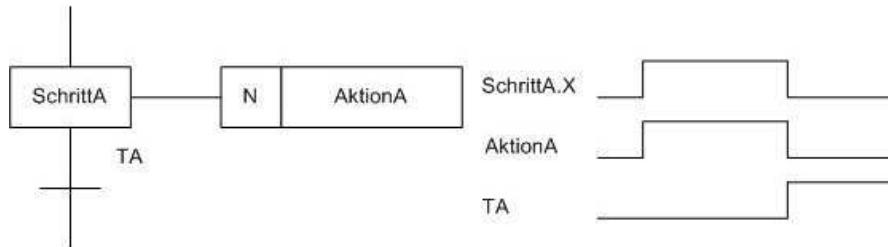


Abbildung 3.5.: Funktionsweise des N-Aktionskennzeichners

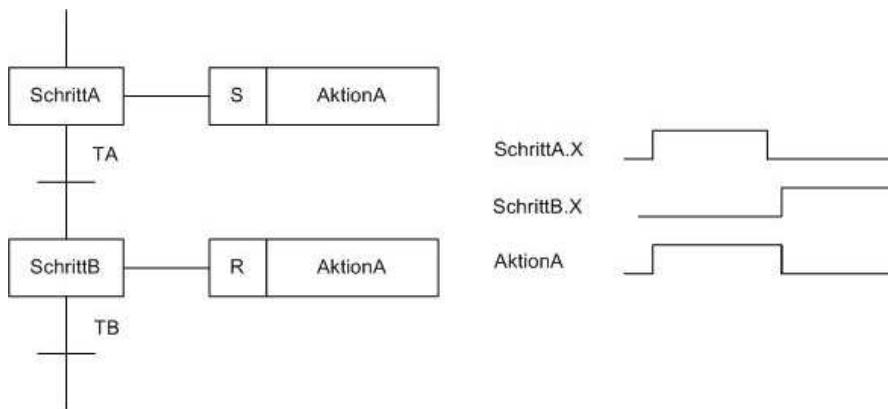


Abbildung 3.6.: Funktionsweise der Aktionskennzeichner S und R

- N (Non-stored): Die Aktion wird ausgeführt so lange der Schritt aktiv ist.
- S (Set): Die Aktion wird ausgeführt, sobald der Schritt aktiv ist und bis der zugehörige Aktionskennzeichner zum Rücksetzen aktiv wird.
- R (Overriding reset): Die mit S, SD, DS und SL Aktionskennzeichner ausgeführten Aktionen werden abgebrochen.
- L (Time limited): Die Aktion wird für die vorgegebene Zeit T ausgeführt so lange der Schritt aktiv ist.
- D (Time delayed): Die Aktion wird erst nach dem Ablauf der vorgegebenen Zeit T ausgeführt, und zwar so lange der Schritt aktiv ist.
- P (Pulse): Die Aktion wird genau zweimal ausgeführt. Einmal sobald der Schritt aktiv wird und einmal, wenn er deaktiviert wird.

- P1 (Pulse: rising edge): Die Aktion wird einmal ausgeführt, sobald der Schritt aktiv wird.
- P0 (Pulse: falling edge): Die Aktion wird einmal ausgeführt, sobald der Schritt deaktiviert wird.
- SD (Stored and time delayed): Nachdem der Schritt aktiviert wurde, wird die Aktion nach dem Ablauf der vorgegebenen Zeit T ausgeführt. Dies passiert selbst, wenn der Schritt nicht mehr aktiv ist. Diese Bedingung bleibt bestehen bis der R Aktionskennzeichner ausgeführt wird.
- DS (Time delayed and stored): Wie beim SD-Kennzeichner wird die Aktion erst nach einer gewissen Wartezeit T ausgeführt. Der Unterschied ist, dass der Schritt während der Wartezeit aktiv sein muss.
- SL (Stored and time limited): Sobald der Schritt aktiv ist, wird die Aktion für die vorgegebene Zeit T ausgeführt, bis der R Aktionskennzeichner aktiv wird.

Als Beispiel ist die Anwendung des Aktionskennzeichners N in Abb. 3.5 beschrieben. Dem *SchrittA* ist die *AktionA* zugeordnet. Sobald der *SchrittA* aktiv ist, wird die *AktionA* wiederholt ausgeführt bis der *SchrittA* inaktiv wird. Dies geschieht, wenn die Übergangsbedingung der Transition TA auf *Wahr* gesetzt ist. Das zugehörige Zeitdiagramm illustriert den Ablauf. *SchrittA.X* ist *Wahr* solange *SchrittA* aktiv ist. Sobald der *SchrittA* inaktiv wird (Sobald die Transation TA *Wahr* ergibt), wird die Variable *SchrittA.X* auf *Falsch* zurückgesetzt. Die *AktionA* wird nur solange ausgeführt als *SchrittA.X* *Wahr* ist. Die Funktionsweise der Aktionskennzeichner S und R ist grafisch anhand eines SFC und eines Zeitdiagramms in Abb. 3.6 dargelegt.

Einige Aktionen können zu komplex sein, um innerhalb eines Aktionsblocks beschrieben zu werden. Es wird empfohlen sie dann in einer anderen SFC-Sequenz oder einer anderen IEC 61131-3 Sprache zu definieren. Im Rahmen dieser Laborübung wird ST zum Beschreiben der Aktionen empfohlen.

Ausführungsmöglichkeiten

SFC als grafische Sprache bietet mehrere Möglichkeiten ein Programm auszuführen, welche im Folgenden kurz beschrieben werden.

Serielle Verbindung Eine serielle Verbindung (engl. Sequence), wie sie in Abb. 3.7 dargestellt ist, stellt eine Folge von Schritten dar, die miteinander durch Transitionen

verbunden sind. Ist die mit einer Transition verbundene Übergangsbedingung *TRUE*, so wird der Schritt vor der Transition deaktiviert und der nächste Schritt aktiviert. Bleibt sie *FALSE* wird der Schritt vor der Transition wiederholt, bis die Übergangsbedingung *TRUE* ist. Die Ablaufsteuerung erfolgt in der Regel von oben nach unten. Der Ablauf kann aber auch mittels eines Sprungs zu einem bestimmten Schritt gesteuert werden.

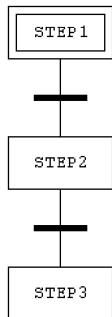


Abbildung 3.7.: SFC: Serielle Verbindung (Sequentielle Abfolge).

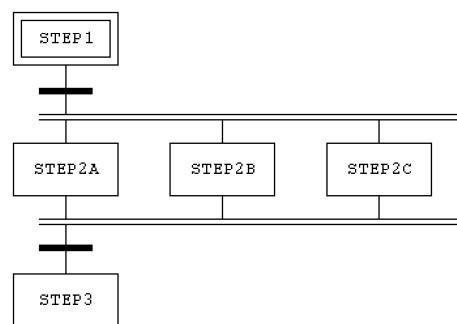


Abbildung 3.8.: SFC: Parallele Verzweigung.

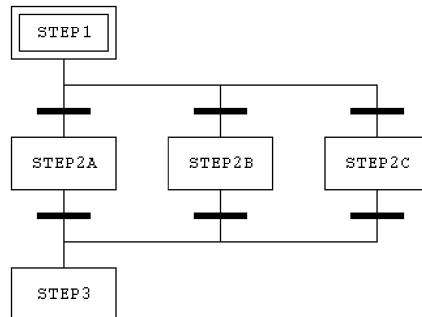


Abbildung 3.9.: SFC: Auswahl-Verzweigung.

Parallele Verzweigung In einer parallelen Verzweigung (engl. *Simultaneous Branch*), wie sie in Abb. 3.8 dargestellt ist, werden alle Zweige gleichzeitig ausgeführt bis die Transition aktiv wird. Die weiteren Schritte der SFC-Sequenz werden erst dann ausgeführt, wenn die Abläufe in allen Zweigen einer parallelen Verzweigung beendet sind.

Auswahlverzweigung Eine Auswahlverzweigung (engl. Selection Branch), wie sie in Abb. 3.9 dargestellt ist, stellt mehrere alternative Zweigsequenzen zur Verfügung. Dabei wird nur ein Zweig ausgeführt, abhängig davon, welche Transition aktiv ist (d. h. wenn ihre Übergangsbedingung auf *TRUE* ist). Jede dieser Zweigsequenzen kann keinen, einen oder mehrere Schritte und zumindest eine Transition beinhalten. Die Übergangsbedingungen der Sequenzen werden, per Definition, von links nach rechts überprüft. Sobald eine Übergangsbedingung auf *TRUE* gesetzt ist, wird die zu ihr gehörende Sequenz ausgeführt.

3.3.8. Modellierung nach dem Standard IEC 61499

Während in den 1960er, 70er und 1980er Jahren noch ein Verkäufermarkt herrschte, genügte es in dieser Zeit der Massenproduktion lediglich „gut genug“ zu sein. Seit den 1990er Jahren hat sich die Situation jedoch zu einem Käufermarkt gewandelt, das Angebot übersteigt die Nachfrage und aufgrund der internationalen Konkurrenz müssen produzierende Betriebe hohe Anforderungen erfüllen um im Markt bestehen zu können. Aufgrund der Marktmacht der Kunden im Käufermarkt muss die produzierende Industrie nicht nur günstige Preise bieten, sondern es werden auch eine hohe Qualität, kurze Lieferzeiten, sowie Individualität der Produkte gefordert. Für den Hersteller bedeutet das die Produktion kleiner Losgrößen und personalisierter Produkte, und damit verbunden schnell ändernde Absatzmärkte. Für die Fabriksebene bedeutet das, dass die Anlage einerseits schnell rekonfiguriert, einfach skalierbar und erweiterbar sein muss. Diese Anforderungen können mit einem hohen Grad an Anlagenflexibilität erfüllt werden. Die IEC 61131 – Programmable controllers [3] hatte jedoch diese Veränderungen nicht vorher gesehen, weshalb Rekonfiguration und Anlagenflexibilität nicht im Fokus der Entwicklungen standen und somit nicht im Standard berücksichtigt wurden. Um mit diesen neuen Anforderungen umgehen zu können, wurde auf Basis der IEC 61131 ein neuer Standard entwickelt, die IEC 61499 – Function blocks [4].

3.3.8.1. Motivation

Die IEC 61499 entstand aus dem Verlangen heraus sich vom Trend der inflexiblen und monolithischen Software zu lösen und den Schritt zu flexibleren Ansätzen (z. B. Schwarm-Intelligenz, Holonik) und verteilten Steuerungen zu wagen. In Abb. 3.10 ist ein Überblick der Entwicklung der Steuerungstechnik dargestellt. Die klassische Steuerungstechnik hatte noch einen Schaltschrank in dem die Software, als auch Hardware lokal vorhanden war. Der Stand der Technik ist die dezentrale Steuerungstechnik, in der die Steuerungssoftware noch lokal in einer SPS vorhanden ist, jedoch intelligente E/A-Komponenten in das Feld ausgelagert werden.

Bei der Idee der IEC 61499 hingegen ist das Ziel sich vom Schaltschrank vollständig zu lösen und dabei die Steuerungssoftware selbst auf intelligente, verteilte Automationskomponenten vollständig ins Feld zu verlagern. Die IEC 61499 verfolgt die Philosophie der Modularität, da jede Komponente einer Steuerungslösung für sich abgeschlossen und gekapselt, als auch im Verband, funktionieren soll. Hierdurch soll es ermöglicht werden eine Automationsanlage komponentenbasiert und modular aufzubauen.

Der Ansatz der Modularität bringt einige Vorteile mit sich. Getestete Module können

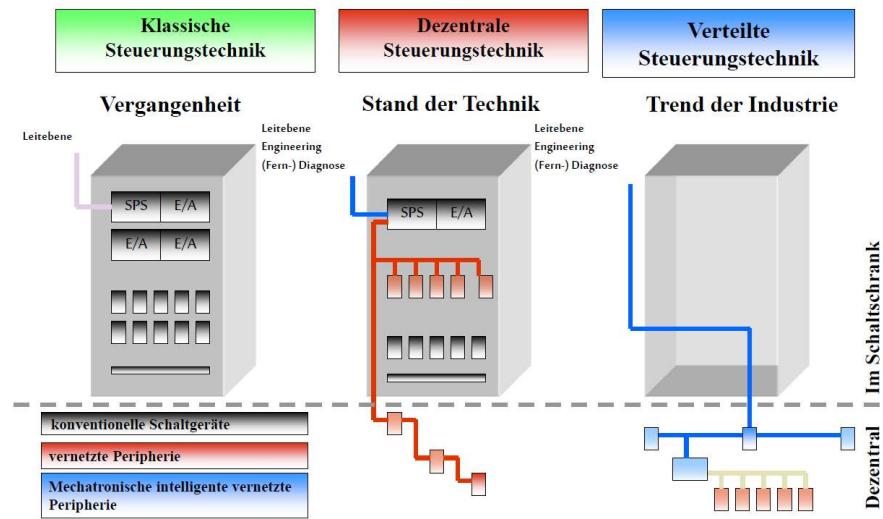


Abbildung 3.10.: Trend der Steuerungstechnik von zentralen Strukturen hin zu verteilten Strukturen

leicht wiederverwendet werden, die Hard- und Software ist schnell erweiterbar. Es kann eine kürzere Inbetriebnahme durch herstellerübergreifendes Engineering sichergestellt werden. Ein weiterer großer Vorteil ergibt sich für den Hersteller, da dieser sein spezifisches Know-how in der Komponente kapseln und als Interface zur Verfügung stellen, ohne dabei sein intellektuelles Eigentum (z. B. Ansteuerung, Regelung) offenlegen zu müssen.

Die Nachteile sind unter anderem die komplexe Interaktion zwischen den Komponenten, ein aufwendigeres Engineering sowie die erhöhte Fehlerwahrscheinlichkeit bei steigender Komplexität. Es darf dabei auch nicht der Übergang zu einer komplett neuen Steuerungsphilosophie unterschätzt werden.

Einer der Grundgedanken der IEC 61499 ist die Interoperabilität von Automationskomponenten verschiedenster Hersteller, weshalb großer Wert auf eine offene Architektur gelegt wird. Eine Architektur wird als offen bezeichnet, wenn deren funktionalen Einheiten die Eigenschaften Portierbarkeit, Interoperabilität und Konfigurierbarkeit erfüllen. Dies erlaubt die Entwicklung von Steuerungscode, welcher anpassungsfähig und wieder verwendbar geschrieben werden kann.

Portierbarkeit Unter Portierbarkeit versteht man, dass Software, die von einem weiteren Implementierungswerkzeug erstellt wurde, trotzdem korrekt interpretiert und verarbeitet werden kann. Seit der Einführung des IEC 61131 Standards gibt es das Verlangen Programme untereinander auszutauschen. Hierfür hat das PLCopen Technical Committee 6 (TC6) ein offenes Format entwickelt, um unabhängig von speziellen Entwicklungsumgebungen zu sein. Das resultierte im sogenannten PLCopen eXtensible Markup Language (XML) welches vollständige IEC 61131 Projekte herstellerunabhängig abbilden kann. Beispiele für Sprachen mit portierbaren Code sind Python, Perl, MATLAB, und die verschiedenen Java Virtual Machine Sprachen (Java, Groovy, Scala, usw.).

Interoperabilität Unter Interoperabilität versteht man, dass Geräte, welche nach demselben System operieren (z. B. IEC 61131 oder IEC 61499 konform), miteinander kommunizieren können. Insbesondere muss dies für Geräte unterschiedlicher Hersteller gegeben sein, wobei die Funktionalität erhalten und somit die verteilte Ausführung gewährleistet ist. Ein Beispiel aus der Automationstechnik ist das Kommunikationsprotokoll OPC Unified Architecture (OPC UA). Dieses Protokoll ermöglicht es nicht nur Maschinendaten herstellerunabhängig zu transportieren, sondern macht diese Daten auch maschinenlesbar semantisch verfügbar.

Konfigurierbarkeit Unter Konfigurierbarkeit versteht man, dass Geräte und deren Software von unterschiedlichen Softwaretools verschiedener Hersteller konfiguriert (ausgewählt, zugewiesen zu Geräten, verbunden und parametriert) werden können.

3.3.8.2. Einführung

Das Grundprogrammierelement der IEC 61499, der Function Block (FB) ist von der Grundidee der IEC 61131 Function Block Diagram (FBD) Sprache entnommen. Um den FB unabhängig von dem in der IEC 61131 genutzten Abarbeitungszyklus zu machen, wird er um Eventein- und ausgänge erweitert, die seine prinzipielle Abarbeitung steuern. Zuerst wird auf das allgemeine Abarbeitungsmodell der IEC 61499 FBs eingegangen, danach werden die FB Typen Basic Function Block (BFB), Composite Function Block (CFB), Service-Interface Function Block (SIFB) und Adapter Interface (AI) besprochen.

Der Funktionsbaustein Das Grundelement der IEC 61499 ist der Function Block (FB), abgebildet in Abb. 3.11. Er besteht aus dem Event Execution Control (EEC), welche im Allgemeinen die eventgesteuerte Abarbeitungslogik besitzt und der sogenannten Encap-

sulated Functionality, welche die auszuführenden Programme (Algorithmen, angebundene Bibliotheken, Kommunikation, usw.) beinhaltet.

Der FB ist prinzipiell passiv und wird durch eingehende Events (links im Bild 3.11) aktiviert und kann ausgehende Events (rechts im Bild 3.11) senden. Eventein- bzw. Ausgänge können mit Datenein- bzw. Ausgängen über das "With-Construct" assoziiert werden, wodurch die assoziierten Daten beim Eintreffen eines Eingangsevents in den internen Speicher des FBs übernommen werden. Bei Ausgangsevents und assoziierten Datenausgängen hingegen werden die internen Daten des FBs für die außenliegenden Teilnehmer zugänglich gemacht.

Ein- und Ausgangsevents sind hierbei instantane Ereignisse welche auftreten können. Sie haben keinen expliziten Wert und werden ähnlich eines Dirac-Pulses mit einer zeitlichen Ausdehnung von null modelliert. Dies hat auch zur Folge, dass im Eventmodell Events nie exakt gleichzeitig auftreten können.

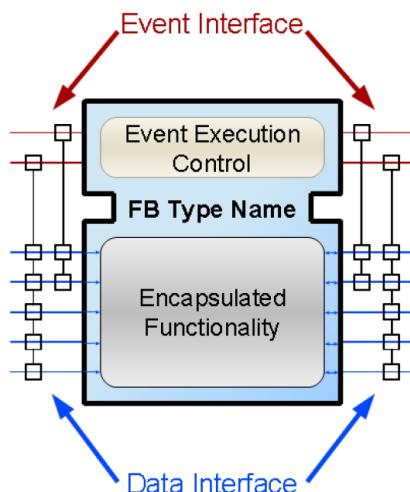


Abbildung 3.11.: Schematische Darstellung des Funktionsbausteins nach IEC 61499

Das Abarbeitungsmodell von FBs ist rein ereignisgesteuert und in Abb. 3.12 dargestellt. Die folgenden Punkte erläutern die für alle FB-Typen gültige Abarbeitungssequenz:

1. Ein Eingangsevent kommt an
2. Die mit dem Eingangsevent assoziierten Dateneingänge werden abgetastet und in das interne Modell des FB übertragen
3. Die EEC wird über das eingehende Event benachrichtigt

4. Die entsprechende interne Funktionalitäten werden abgearbeitet
5. Sobald die internen Funktionalitäten ihre Abarbeitung beendet haben, stellen sie intern neue Ausgangsdaten zur Verfügung
6. Das Ausgangsevent ist bereit gesendet zu werden, assoziierte Ausgangsdaten werden nach Außen übertragen und verfügbar gemacht.
7. Das Ausgangsevent wird gesendet
8. Die Schritte 4–7 können hierbei mehrmals ohne weitere Aktivierung von Außen durchlaufen werden

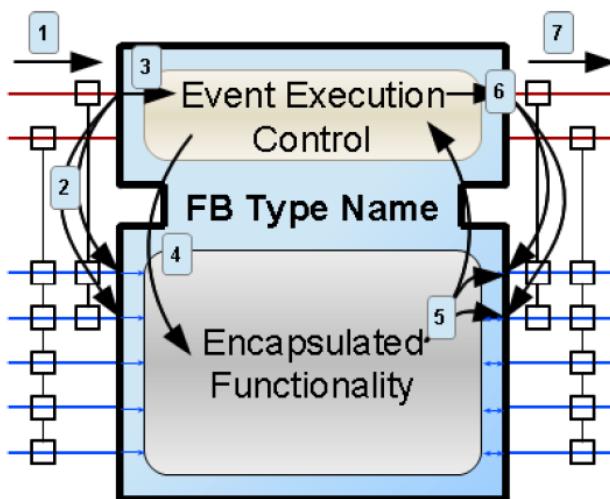


Abbildung 3.12.: Schematische Darstellung des Abarbeitungsmodells nach dem Standard IEC 61499

In der IEC 61499 wird eine hierarchische Struktur mit einem Top-Down-Ansatz definiert. Diese beinhaltet die Ansicht von System-Gerät-Ressource-Anwendung-Funktionsbaustein um den Aufbau und die Abläufe zu beschreiben. Eine schematische Darstellung ist in Abb. 3.13 zu sehen.

Systemmodell (System Model) Von außen betrachtet besteht ein verteiltes System aus einer Reihe von Geräten, welche über ein Kommunikationsnetzwerk verbunden sind. Hierbei können dieselbe oder verschiedene Anwendungen parallel und geräteübergreifend ablaufen. Eine Anwendung beschreibt ein Stück Software, welches aufgrund der immer intelligenteren Sensoren und Aktuatoren über mehrere Geräte verteilt ablaufen kann.

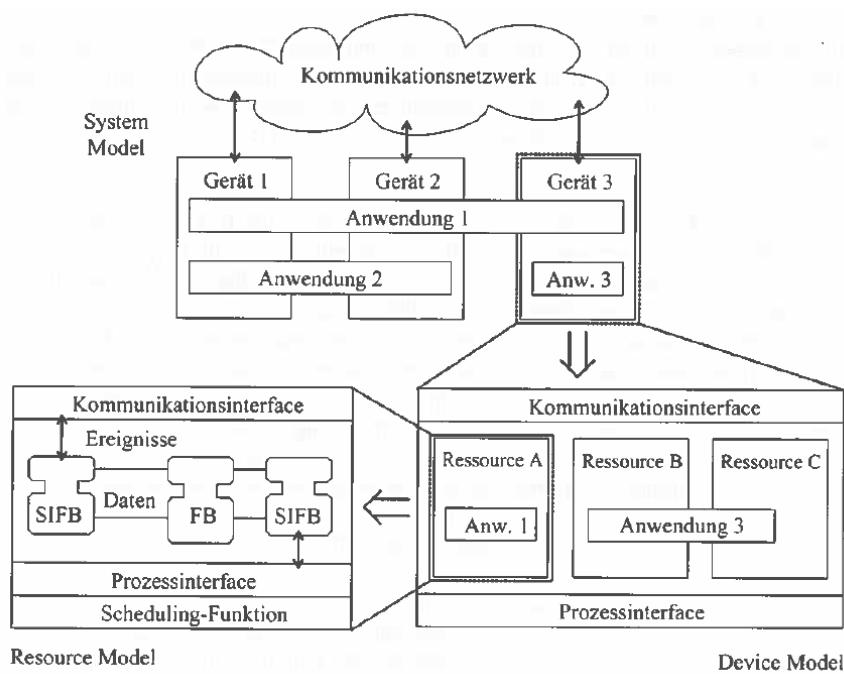


Abbildung 3.13.: Systemmodell, Gerätemodell und Ressourcenmodell mit Anwendungen und Funktionsbausteinen nach IEC 61499

Beispiel hierfür ist die Mehrgrößenregelung, wo die Anwendung die Information von verschiedenen Sensoren benötigt, welche physikalisch verteilt sind.

Gerätemodell (Device Model) Aus dem Blickwinkel des Geräts besteht dieses aus:

- den Anwendungen
- einer Schnittstelle zum Kommunikationsinterface
- einer Schnittstelle zum Prozessinterface
- der Hardware, auf der die Ressource abläuft.

Eine Ressource stellt die Funktionen für die Abarbeitung der Programme bereit und ermöglicht die unabhängige Ausführung und Steuerung von Funktionsbausteinen. Es können auf einem Gerät mehrere Ressourcen vorhanden sein. Ein Gerät hat neben der Schnittstelle zum Kommunikationsinterface auch eine Schnittstelle zum Prozessinterface, welches mit den physischen Prozessen durch Eingänge und Ausgänge miteinander in Verbindung tritt.

Ressourcenmodell (Resource Model) Eine Ressource wiederum ermöglicht das Aufrufen und Abarbeiten mehrerer FBs. Es wird unterschieden zwischen den FBs, welche die eigentliche Anwendung ausführen, und den Service-Interface Function Blocks (SIFBs), welche die Ankopplung an externe Bibliotheken (z. B. MATLAB) und das Kommunikations- bzw. Prozessinterface ermöglichen. Eine Ressource ist in sich eigenständig. Das bedeutet, dass es möglich ist eine Ressource zu erstellen, zu verändern und zu löschen, ohne andere Ressourcen damit zu beeinflussen. Die Ressource stellt die Infrastruktur für die Kommunikation von Daten und Events mit der Umgebung (z. B. Sensoren, Aktoren) dar. Sie ermöglicht es auch Daten zu speichern, Algorithmen auszuführen, Abarbeitung zu planen, etc.

Anwendungsmodell (Application Model) Die eigentliche Verschaltung und Programmierung findet nicht auf Ebene der Ressource, sondern auf der Anwendungsebene, wie in Abb. 3.14 dargestellt, statt. Hier können FBs, unabhängig von der Ressource, miteinander verschaltet werden. Die Events und Daten fließen links in den FB hinein, werden verarbeitet und von der rechten Seite wieder ausgegeben. Eine Anwendung ist ein abstrakter Begriff für das gewünschte Verhalten eines Systems. Sie enthält weder Variablen noch ein Interface zur Kommunikation mit Prozessen und damit kann die semantische Abarbeitung nur erfolgen, wenn die Anwendung mit Ressourcen oder Geräten assoziiert

wird.

Verteilungsmodell (Distribution Model) Eine Haupteigenschaft der IEC 61499 ist die Möglichkeit, Anwendungen und Unteranwendungen auf ein oder mehreren Ressourcen zu verteilen. Ein FB wird als atomare Einheit dargestellt und kann daher nur auf einer Ressource ausgeführt werden, d. h. die Performance eines FBs ist unabhängig von der Verteilung. Im Gegenteil dazu kann bei einem Datenaustausch von verteilten Anwendungen und Unteranwendungen zwischen den Ressourcen, die Laufzeit und Zuverlässigkeit aufgrund des Kommunikationsnetzwerks beeinflusst werden.

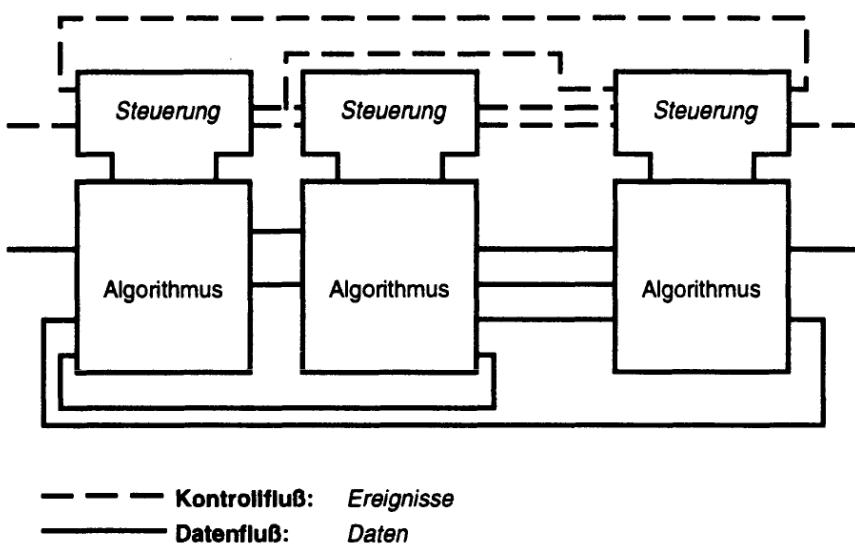


Abbildung 3.14.: Anwendungsmodell nach IEC 61499

3.3.8.3. Funktionsbausteintypen

In der IEC 61499 unterscheidet man zwischen Basic Function Blocks (BFBs), Composite Function Blocks (CFBs), SIFBs und den Adapter Interfaces (AIs). Diese werden im Nachfolgenden näher beleuchtet.

Basic Function Block Ein BFB ist eine abgeschlossene Programmeinheit, in der die EEC als Zustandsautomat beschrieben wird. Die IEC 61499 gibt hierfür das Event Control Chart (ECC) zur Beschreibung des Zustandsautomaten vor. Die ECC definiert hierbei Zustände (Rechtecke in Abb. 3.15), welche einen Namen (z. B. START, INIT, MAIN)

besitzen. Ein Zustand kann mit ein oder mehreren Aktionen assoziiert werden, welche im ECC durch eine horizontale Linie mit dem Zustand verbunden sind. Aktionen können Algorithmen und Ausgangsevents beinhalten. Bei Aktivierung des Zustands werden die Aktionen ausgeführt, d. h. Algorithmen werden abgearbeitet und assoziierte Ausgangsevents werden gesendet. Falls dem Zustand kein Algorithmus zugewiesen ist, wird das assoziierte Event sofort nach Betreten des Zustandes ausgelöst.

Die Zustände sind untereinander über Zustandstransitionen verbunden. Zustandstransitionen haben eine Richtung, vom Pfeilschaft zur Pfeilspitze, und eine Transitionsbedingung welche aus einem einzelnen Event und einer logischen Verknüpfung von Daten bestehen kann, also beispielsweise INIT, wenn die Transition mit dem INIT Event passieren soll, oder INIT & QI=TRUE um die Transition nur dann beim Eintreten des INIT Events durchzuführen, falls der boolesche QI Eingang den Wert TRUE besitzt. Des Weiteren kann das Event wegfallen und die Transition nur durch eine Datenbedingung beschrieben werden, was bedeutet das jedes eintretende Event die Transition auslösen kann so lange der entsprechende Zustand aktiv und die Datenbedingung erfüllt ist. Eine Transition kann auch ohne jegliche Bedingungen durchgeführt werden, dies sind die sogenannten unbedingten Transitionen. Eine unbedingte Transition wird angezeigt, indem die Transitionsbedingung auf 1 (entspricht TRUE) gesetzt wird. Da ein Zustandswechsel jedoch nur dann möglich ist, wenn der Baustein aktiviert ist, also ein Eingangsevent angekommen ist, und die ECC solange aktiv bleibt bis keine weiteren Transitionen mehr aktiviert werden können, sind unbedingte Transitionen nie die erste Transition die geschaltet werden kann. Dies ist einfach darin zu erklären, dass ein aktiver FB eine unbedingte Transition konsumieren und daher einen Zustandswechsel vollführen würde. Daraus ist auch abzuleiten, dass zwei Zustände nicht durch gegengleiche unbedingte Transitionen verbunden werden dürfen, da sich daraus eine Endlosschleife ergeben würde.

Aus diesen Regeln ergibt sich auch, dass es Situationen gibt, in denen mehr als eine Transition erfüllt wäre und damit schalten würde. Um hier eine eindeutige Transition festlegen zu können, haben Transitionen eine Prioritätsreihenfolge mit der eindeutig die zu schaltende Transition ermittelt wird.

Die eingangs erwähnten Algorithmen können laut IEC 61499 in jeder beliebigen Sprache implementiert wie C/C++, Java, Ada¹, Python, etc., es ist jedoch üblich Structured Text (ST) nach der IEC 61131 zu verwenden.

Composite Function Block Ein CFB ist, ähnlich einer Subroutine in C, eine Kombination mehrerer FBs beliebigen Typs, wie in Abb. 3.16 dargestellt. Nach Außen hin ist nur

¹Benannt nach Ada Lovelace, eine britische Mathematikerin des 19. Jahrhunderts und die erste Person die als Programmierer_in bezeichnet werden kann.

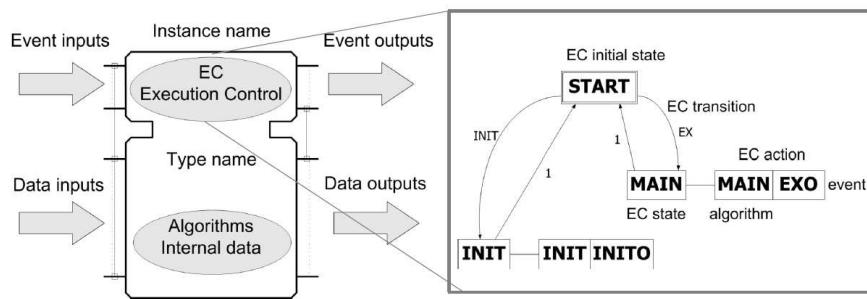


Abbildung 3.15.: Beispiel einer EEC welche als ECC implementiert ist

die Schnittstelle des CFB sichtbar. Die Event- und Datenein- und ausgänge werden an die internen FBs weiter geleitet und durch das interne Function Block Network (FBN) verarbeitet. Die EEC eines CFB ist dadurch nicht explizit wie bei einem BFB durch die ECC definiert, sondern ergibt sich aus dem Verhalten des internen FBN, d. h. dem kombinierten Verhalten der intern verschalteten FBs. Ein Beispiel für die Verwendung eines CFB ist ein Portalroboter mit Sauger, welcher aus mehreren FBs besteht, aber als eine Einheit fungiert.

Unteranwendung (Subapplication) Eine Unteranwendung hat die externe Charakteristik eines FBs (Daten- und Eventinterface) und kann FBNs, bestehend aus FBs, CFBs und SIFBs, enthalten. Sie kann als spezieller CFB angesehen werden, da die Benutzung nur in Anwendungen oder Unteranwendungen möglich ist und die gleichen Verteilungseigenschaften wie bei einer Anwendung gelten. Das bedeutet, eine Unteranwendung kann im Gegenzug zu einem CFB optional auf ein oder mehreren Ressourcen verteilt werden.

Service-Interface Function Block Die bisherigen FBs können nur Algorithmen beschreiben, welche Teil der IEC 61499 sind, also die interne Abarbeitungslogik eines Steuerungsprogramms, ohne auf IEC 61499-externe Ressourcen wie externe Bibliotheken, Kommunikations- oder Prozessinterfaces zuzugreifen. Es ist jedoch in einer industriellen Anwendung unausweichlich die Daten von Sensoren zu lesen, beziehungsweise Daten an Aktuatoren zu schicken. SIFBs stellen das Interface zwischen der Anwendung und der umgebenden Hardware (E/A Komponenten) und Software (Bibliotheken, OS Funktionen) dar. Ein SIFB besteht wie jeder andere FB im IEC 61499 aus Daten und Event Ein- und Ausgängen. Im Allgemeinen unterscheidet man, wie in Abb. 3.17 angedeutet, zwischen SIFBs vom Typ **Requester** und SIFBs vom Typ **Responder**. Der Unterschied zwischen diesen beiden Typen liegt darin, wer den FB aktiviert. Der Requester SIFB wird hierbei durch die Anwendung aktiviert.

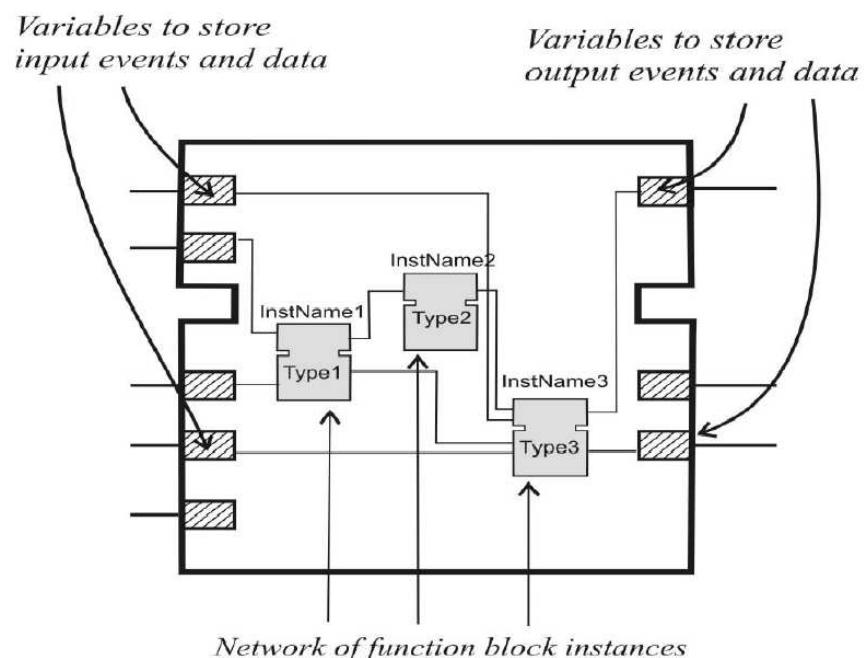


Abbildung 3.16.: Beispiel eines Composite Function Blocks

Der Responder FB jedoch wird durch die Ressource aktiviert. Dadurch verstößt er als einziger FB-Typ gegen die Regel, dass FBs prinzipiell passiv und nur durch den Eingang eines Events aktiv werden dürfen. Der Grund hierfür ist, dass es erstens einen Ursprung für das erste Event geben muss und zweitens, dass Ereignisse des unterlagerten Systems, auf dem eine IEC 61499 Laufzeitumgebung läuft, Ereignisse an die Laufzeitumgebung liefern können muss, beispielsweise für Timer, Änderungen von Eingängen, Hardwarefehlern oder Beendigung des Programms.

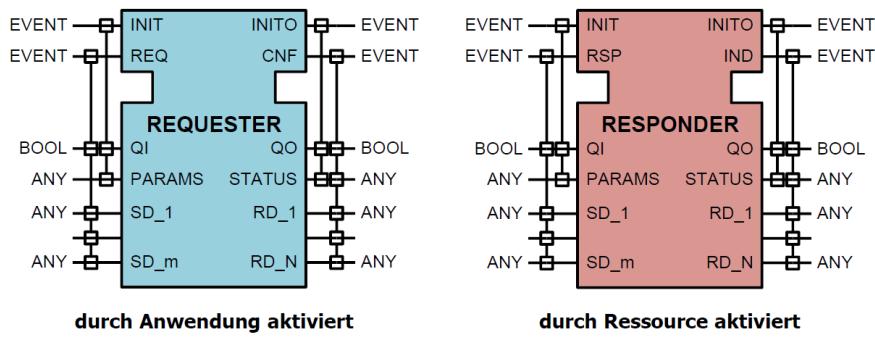


Abbildung 3.17.: Beispiel eines Requesters und Responders

Obwohl SIFBs zumeist nicht vom Anwendungs-Entwickler erstellt werden, sondern typischerweise von Komponentenherstellern (z. B. für jeweilige Feldbusssysteme oder intelligente Sensoren), gibt es Konventionen für typische Event- und Daten- Eingänge und Ausgängen, welche kurz erläutert werden.

INIT: Bei einem INIT-Event wird eine Initialisierung erwünscht. Dieses wird mit dem Daten-Eingang QI sowie PARAMS gemapped. QI hat hierbei die Aufgabe aufzuzeigen, ob der Service des Funktionsbausteins initialisiert oder terminiert werden soll. PARAMS enthält für die Ausführung wichtige Parameter wie zum Beispiel Baudrate oder Netzwerkadressen.

REQ: REQ steht für Request eines Requester-Eingang-Events. Bei Eingang des Events werden Daten von einem externen Gerät (zum Beispiel Sensor) gesendet. Deshalb werden hier die Eingänge der Daten SD gemapped. QI hat hierbei die Aufgabe aufzuzeigen, ob der Funktionsbaustein initialisiert oder terminiert ist.

RSP: RSP steht für Response eines Responder-Eingang-Events. Es wird signalisiert, dass die ausgegebenen Daten verarbeitet wurden.

INITO: Nach durchlaufener Initialisierung des FB wird ein INITO-Event ausgesendet. Der FB muss dabei aber nicht erfolgreich initialisiert worden sein. Das Mappen mit dem Ausgang QO dient dazu aufzuzeigen, ob die Initialisierung erfolgreich war ($QO = \text{TRUE}$) oder fehlgeschlagen ist ($QO = \text{FALSE}$). Des Weiteren kann noch ein Status mitgeteilt werden, welcher zum Beispiel einen Fehlercode, der Information über die fehlgeschlagene Initialisierung enthält.

CNF: CNF steht für Confirmation eines Requester-Ausgang-Events. Dieses Event zeigt an, dass die Eingangsdaten des Requesters nach einem REQ-Event verarbeitet wurden.

IND: IND steht für Indication eines Responder-Ausgang-Events. Dieses wird ausgesendet, wenn neue Ausgangsdaten verfügbar sind, z. B. ein neues Datenpaket empfangen und verarbeitet wurde.

QI : BOOL QI steht für Qualifier-Input. Ein *True* zeigt eine gewünschte Initialisierung und ein *False* eine Terminierung an.

PARAMS : ANY PARAMS steht für Parameters. Diese können von beliebigen Typs sein.

SD : ANY SD steht für Send Data. Dies sind die Eingangsdaten eines SIFB welche an die externe Ressource gesendet werden sollen.

QO : BOOL QO steht für Qualifier-Output. Ein *True* zeigt eine erfolgreiche Initialisierung und ein *False* eine fehlgeschlagene Initialisierung an.

STATUS : ANY STATUS wird verwendet, um die Bereitschaft des Funktionsbausteins wiederzugeben sowie einen Code bei etwaigen Fehlern.

RD : ANY RD steht für Recieve Data. Diese werden von der externen Ressource an die IEC 61499 Laufzeitumgebung gesandt.

In Abb. 3.18 ist das Ablaufdiagramm eines Requesters dargestellt. Generell gibt es hier drei Phasen. Die erste initialisiert den gewünschten Dienst (z. B. Kommunikation). Es wird ein INIT-Event geschickt. Das + bedeutet ein *True* von QI, also der Wunsch einer Initialisierung. Bei erfolgreicher Initialisierung des FB, wird ein INITO(+) gesendet, wobei das + hier ein *True* von QO bedeutet. Nun ist der Requester bereit Daten zur Verarbeitung zu übernehmen, in dem gezeigten Fall ist dies das Senden von Daten. Beim Eingang eines REQ-Events werden die Daten die an SD übernommen und verarbeitet, im gezeigten Fall gesendet. Bei einer erfolgreichen Datenverarbeitung wird dann das Ausgangs-Event CNF(+) gesendet. Wie zu sehen ist, wird das REQ-Event durch die Anwendung ausgelöst. Ein Beispiel hierfür ist das Senden eines Sensorwertes. Wenn der Dienst bzw. die Verbindung nicht mehr benötigt wird, kann diese mit einem INIT(-) beendet werden.

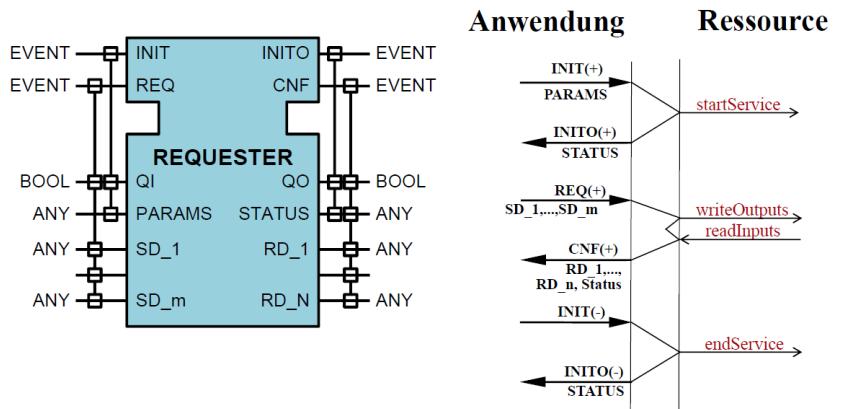


Abbildung 3.18.: Ablaufdiagramm eines Requesters

In Abb. 3.19 ist das Ablaufdiagramm eines Responders dargestellt. Die Initialisierung findet auch hier durch die Anwendung statt. Im Unterschied zum Requester, wird jedoch ein IND-Event durch die Ressource getriggert. Das bedeutet, dass Daten zum Lesen bereit sind, welche an den RD Ausgängen bereitgestellt werden. Wenn die Daten angekommen und verarbeitet sind, erhält der Responder von der Anwendung ein RSP-Event.

Um das Ganze noch zu verdeutlichen, ist in Abb. 3.20 das Ablaufdiagramm bei einem Datentransfer von Publisher und Subscriber dargestellt. Diese Funktionsblöcke sind für die uni-direktionale Kommunikation gedacht, das bedeutet, dass der Publisher die Daten an den Subscriber sendet. Bei Eingang eines REQ-Events an den Publisher, werden die

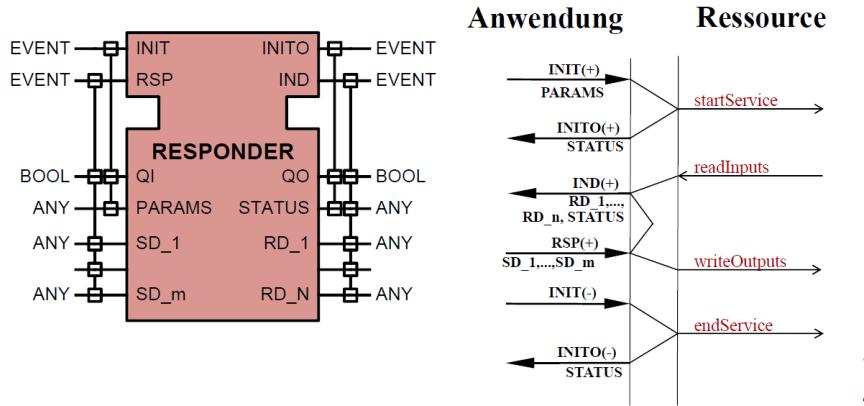


Abbildung 3.19.: Ablaufdiagramm eines Responders

Daten die an SD anliegen, zum Subscriber gesendet. Dies löst ein IND-Event aus welches angezeigt, dass Daten empfangen wurden. Die Daten werden dann an die Anwendung über die RD Ports weitergegeben. Wenn die Daten von der Anwendung fertig verarbeitet wurden, schickt diese an den Subscriber ein RSP-Event. Bei erfolgreichem Datentransfer wird vom Publisher ein CNF-Event gesendet, was die erfolgreiche Datenübertragung andeutet.

Beispielsweise benötigt die Anwendung den Wert eines Sensors, weshalb sie ein Request an den Publisher schickt, der die Sensordaten an den Subscriber sendet. Um aufzuzeigen, dass der Datentransfer erfolgreich war, wird ein CNF-Event vom Publisher gesendet. Wenn die Daten angekommen sind, wird mit einem IND-Event angezeigt, dass die Daten verarbeitet werden können. Wenn dies geschehen ist, wird eine Bestätigung mittels RSP-Event angezeigt.

Neben der uni-direkionalen Kommunikation von Publisher/Subscriber, gibt es auch die bidirektionale Verbindung eines Client/Server-Systems. Diese ist in Abb. 3.21 abgebildet.

Adapter Interface Ein häufiger Kritikpunkt an der IEC 61499 ist, dass Event und Datenleitung vollkommen unabhängig voneinander verbunden werden können. Dies ist üblicherweise von Vorteil, kann jedoch auch zu unübersichtlichen FBNs führen. In manchen Fällen sind aber Event und Datenleitungen Teil einer Softwareschnittstelle. In solchen Fällen kann es leicht zu Fehlern in den Verbindungen der Daten und Eventleitungen kommen, bzw. werden diese Interfaces nicht erkannt und inkompatible FBs angeschlossen.

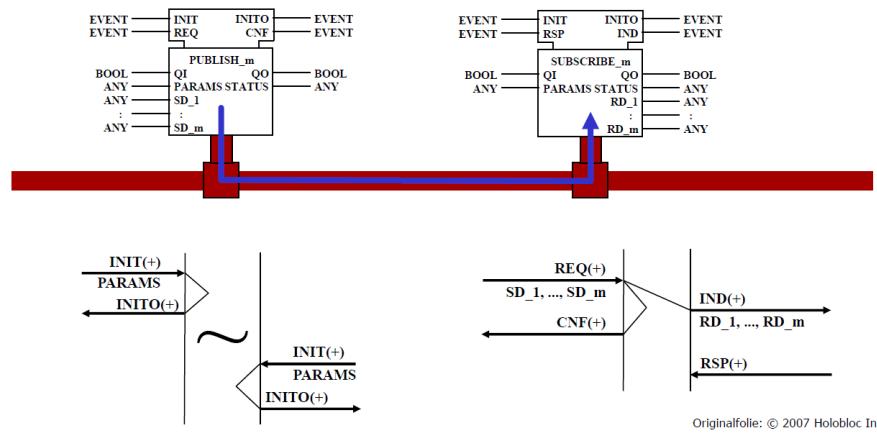


Abbildung 3.20.: Ablaufdiagramm eines Publisher/Subscribers

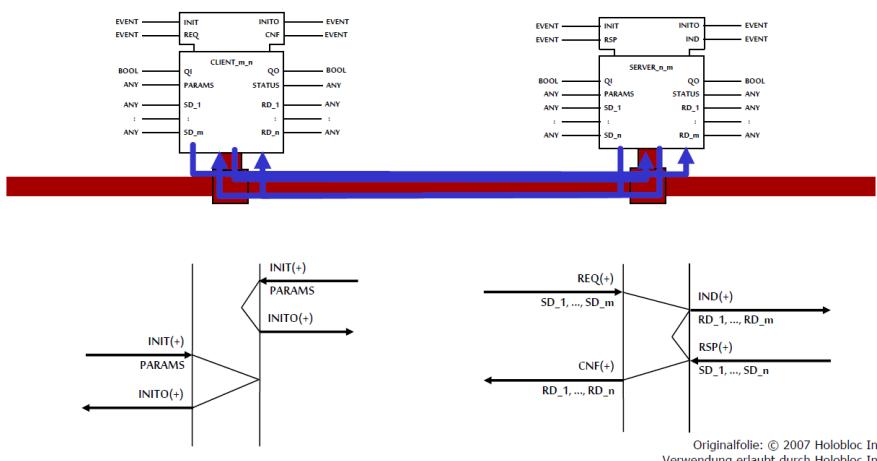


Abbildung 3.21.: Ablaufdiagramm eines Client/Servers

Das AI schafft hier Abhilfe, indem es Event- und Datenleitungen in eine einzelne bidirektionale Verbindung zusammenfasst. AIs werden grafisch ebenfalls als FB dargestellt und besitzen wie die anderen FB-Typen Eingangs- und Ausgangsdaten und -Events, jedoch im Gegensatz zu den anderen FB-Typen keine Algorithmen, Zustände oder Konstanten. Wie bereits erwähnt sind AIs bidirektionale Verbindungen die zwei FBs verbinden und stellen Interfaces dar, wie man sie beispielsweise aus Java oder C# kennt. Die Seite der AI-Verbindung die Daten und Events bereitstellt wird „Plug“ genannt, die Seite die Daten und Events konsumiert wird „Socket“ genannt. Die Namen leiten sich aus der Analogie von elektrischen Steckern und Kupplungen ab, da nur Sockets und Plugs gleichen Typs, also gleiche Anzahl und Datentypen von Daten- und Eventleitungen, verbunden werden können.

Das Prinzip ist schematisch in Abb. 3.22 dargestellt.

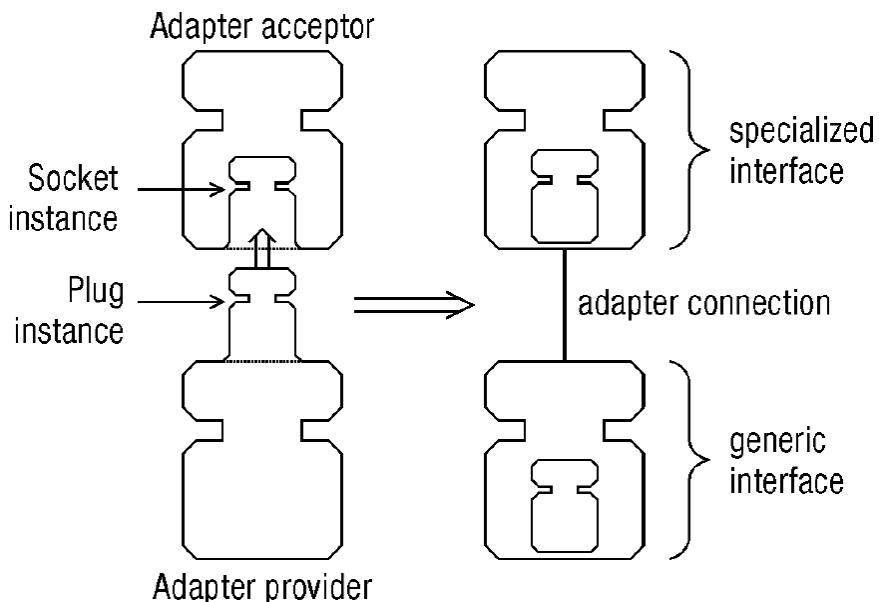


Abbildung 3.22.: Schematische Darstellung des Adapterkonzepts

Die IEC 61499 sieht vor, dass BFBs oder CFBs AIs entweder als Sockets oder Plugs, beinhalten können. Bei CFBs wird die AI als weiterer FB des internen FBN dargestellt, wobei, je nachdem ob es sich um einen Plug oder Socket handelt, der zusätzliche FB einfach oder gespiegelt im FBN vorhanden ist.

Bei BFBs gibt es keine explizite Darstellung, die Daten und Eventleitungen müssen in der ECC und in den Algorithmen mit ihrem vollständig qualifizierten Namen angesprochen werden. Wenn ein fiktiver AI-Typ den Namen **Interact** hätte, würde das Event **INIT**

mit dem Namen `Interact.INIT` im BFB angesprochen werden.

Ein mögliches Beispiel wäre eine Temperatur-Regelstrecke. Die Messwerte werden in den Regelerbaustein eingespielt, welcher nach erfolgter Berechnung der Stellgrößen diese an die Aktoren ausgibt. Die exakte Implementierung des Regelalgorithmus (PID, MPC, o. ä.) die der Regelerbaustein verwendet, kann in Objekt-orientierter Programmierweise (vgl. "Strategy-Pattern"), über einen Implementierungs-FB, der über einen zusätzlichen AI angebunden wird, definiert werden.

3.3.8.4. Zusammenfassung

Die IEC 61499 entstand aus dem Verlangen nach mehr Flexibilität, das durch Kapselung und Modularität erreicht wurde. Hierfür ist eine offene Architektur essenziell. Die elementare Programmeinheit bildet hier der Funktionsbaustein, welcher vom Standard IEC 61131 ausgehend um ein Event-Interface erweitert wurde. Die Abarbeitung erfolgt nicht zyklisch, sondern rein ereignisgesteuert. Bei Eingang eines Events wird über die EEC der jeweilige Algorithmus ausgeführt. BFBs und CFBs beschreiben die Logik eines IEC 61499 Programms, während SIFBs externe Ressourcen, wie Bibliotheken, Kommunikation und Prozessinterfaces an das IEC 61499 Programm anbinden. CFBs sind entsprechend den Subroutinen aus anderen Programmiersprachen und AIs repräsentieren Programmschnittstellen.

4. Zuverlässigkeit und Sicherheit

Anmerkung: Bis zur Fertigstellung dieses Kapitels ist zur Klausurvorbereitung ausschließlich der entsprechende Foliensatz (AuS_09_Zuverlaessigkeit.pdf) verbindlich.

Die heutigen Produktionssysteme, seien es Fertigungszellen oder hochkomplexen chemische Produktionsanlagen, stellen hohe Anforderungen an die Zuverlässigkeit und Sicherheit der Anlage bzw. der Einzelkomponenten. Dabei beschreibt die Zuverlässigkeit das Ausfallverhalten eines Systems auf statistischer Basis und bezieht sich dabei auf die Systemeigenschaften hinsichtlich der Erfüllung von Erfordernissen. Einige wichtige Kenngrößen zur Beurteilung der Zuverlässigkeit eines Systems ist die Zuverlässigkeits- oder Überlebenswahrscheinlichkeit $R(t)$. Diese ist folgendermaßen definiert:

Definition 4.1 (Zuverlässigkeitsfunktion). Die Zuverlässigkeitsfunktion $R(t)$ ist die Wahrscheinlichkeit, dass eine bestimmte Betrachtungseinheit in einem Zeitraum von 0 bis t funktionsfähig ist.

Analog gilt für die Versagenswahrscheinlichkeit:

Definition 4.2 (Versagenswahrscheinlichkeit). Die Versagenswahrscheinlichkeit $Q(t)$ ist die Wahrscheinlichkeit, dass eine bestimmte Betrachtungseinheit bereits innerhalb des Zeitraums 0 bis t versagt. Es gilt, dass $R(t) + Q(t) = 1$.

Für die Zuverlässigkeitsfunktion $R(t)$ ist in Anlehnung an die Exponentialverteilung eine Versagensrate bzw. Ausfallsrate $\lambda(t)$ definiert:

Definition 4.3 (Versagensrate). Die Versagensrate $\lambda(t)$ ist die negative Ableitung der logarithmischen Zuverlässigkeitsfunktion zum Zeitpunkt t , also

$$\lambda(t) = -\frac{d}{dt} \ln R(t) = -\frac{1}{R(t)} \frac{dR(t)}{dt}. \quad (4.1)$$

Die Versagensrate $\lambda(t)$ ist damit trotz gewisser Ähnlichkeiten im Allgemeinen keine

Wahrscheinlichkeitsdichtefunktion¹. Damit lässt sich die Zuverlässigkeitfunktion aufgrund einer bekannten Versagensrate schreiben als

$$R(t) = \exp \left(- \int_0^t \lambda(\tau) d\tau \right). \quad (4.2)$$

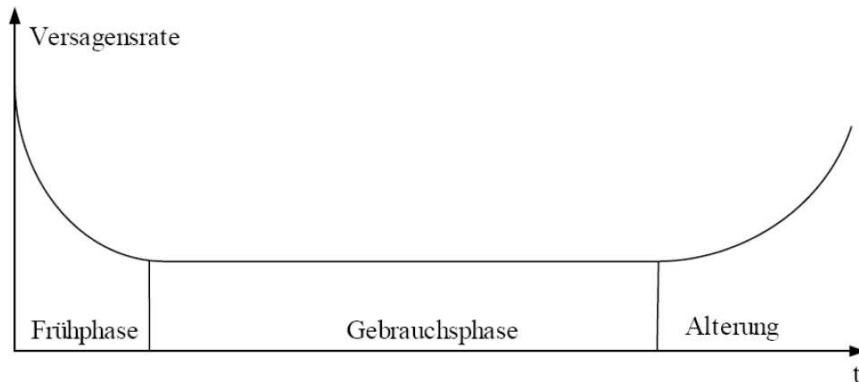


Abbildung 4.1.: Empirischer Zeitverlauf der Versagensrate $\lambda(t)$, auch als „Badewannenkurve“ bekannt.

Der Verlauf der Versagensrate von technischen Systemen zeigt üblicherweise qualitativ sehr ähnliches Verhalten: die sog. „Badewannenkurve“ wie in Abb. 4.1 dargestellt. Die Versagensrate setzt sich demnach aus drei Komponenten zusammen: den Frühausfällen, welche auf Produktionsfehler und defekte Bauelemente zurückzuführen sind, einer langen und konstant niedrigen Phase und einem Anstieg bei hohen Betriebszeiten durch Alterung und Verschleiß.

Anmerkung: Bis zur Fertigstellung dieses Kapitels ist zur Klausurvorbereitung ausschließlich der entsprechende Foliensatz (AuS_09_Zuverlaessigkeit.pdf) verbindlich.

¹Die Versagensrate kann auch als Grenzwert der bedingten Wahrscheinlichkeit hinsichtlich Δt gesehen werden, dass im Intervall $(t, t + \Delta t)$ ein Fehler eintritt, unter der Annahme, es sei von 0 bis t noch keiner eingetreten.

5. Kommunikationssysteme

Anmerkung: Bis zur Fertigstellung dieses Kapitels ist zur Klausurvorbereitung ausschließlich der entsprechende Foliensatz (AuS_10_Kommunikationssysteme.pdf) verbindlich. Dieses Kapitel beleuchtet die Möglichkeiten der Vernetzung der in den Kapiteln 1 und 3 näher erläuterten Komponenten und gibt einen Überblick über die heutige industriell eingesetzten Kommunikationssysteme.

5.1. Einführung

Moderne Leitsysteme benötigen robuste, störungsfreie Kommunikationsmöglichkeiten mit unterschiedlichsten Anforderungen. Netzwerke sollen dabei die Kommunikation zwischen Geräten der gleichen Ebene (horizontale Kommunikation), Geräten unterschiedlicher Ebenen (vertikale Kommunikation), BedienerInnen und System sowie zwischen Feldkomponenten und Prozessnahe Komponenten (PNKs) ermöglichen. Wichtige Merkmale eines Netzwerks sind

- offene, standardisierte Protokollspezifikationen,
- geringer Installationsaufwand, also
 - variable Anbindung aller Komponenten (vgl. Netzwerktopologie) und
 - Zweileitertechnik,
- hohe Datenübertragungsrate,
- flexible Änderungen im laufenden Betrieb und
- Einsatzmöglichkeit in explosionsfähigen Atmosphären.

Wie bereits in Kapitel 1 bemerkt, unterscheidet man in Prozessleitsystemen grob zwischen Systembussen und Anlagen- oder Feldbussen. Während erstere für die Kommunikation innerhalb der Prozessleitebene verwendet werden, sind letztere innerhalb der Steuerungsebene anzutreffen. Die zunehmende Verwendung von echtzeitfähigem Ether-

net für Geräte auf Steuerungsebene (Industrial Ethernet) verwischt diese Abgrenzung, wirft jedoch neue Fragen hinsichtlich Netzwerksicherheit auf.

Dieselbe Entwicklung zeigt sich auch bei der als Instrumentierung bezeichneten Anbindung von Sensoren und Aktoren an PNKs. Während bis in die 1980er Jahre die Einzelverdrahtung jedes einzelnen Sensors Usus war, werden auch hier vielfach Bussysteme vorgezogen. In der Praxis führt dies zum gleichzeitigen Betrieb unterschiedlicher Arten, wie Signale in das Leitsystem integriert werden.

5.1.1. Analoge Instrumentierung

Die klassische Anbindung von Sensoren erfolgte über die in IEC 60381 (Teil 1) normierte analoge Stromschnittstelle. Die Verwendung von Stromschnittstellen hat dabei den Vorteil, dass diese wesentlich weniger störanfällig sind als übliche Spannungsschnittstellen. Gemäß IEC 60381 existieren zwei Varianten welche Wertebereiche zur Darstellung des Signals zulässig sind: 0 mA–20 mA oder 4 mA–20 mA. Letztere ermöglicht es unabhängig vom Signalpegel, den Sensor gleichzeitig mit Energie zu versorgen. Da zusätzlich auch ein Kabelbruch einfach detektiert¹ werden kann, wird überwiegend die zweite Variante eingesetzt.

5.1.2. Digitale Instrumentierung

Um Sensoren ohne Busschnittstelle an den Bus koppeln zu können, muss an einem Punkt zwischen Sensor und SPS eine Digital-Analog-Wandlung stattfinden. Praktischerweise verbindet man dies gleich mit einem Multiplexer, um mehrere Komponenten im Feld ansprechen zu können. Je nach Ort der Wandlung unterscheidet man hier zwei Typen:

Rackbus Die D/A-Umsetzung erfolgt im Schaltraum, von wo aus wieder eine herkömmliche Parallelverdrahtung erfolgt.

Remote-I/O Die D/A-Umsetzung erfolgt in der Nähe der betreffenden Sensoren im Feld selbst.

Beide Varianten stellen einen Mittelweg zwischen analoger Instrumentierung mittels Einzelverdrahtung und einer direkten Anbindung von Sensorik und Aktorik per Feldbus dar. Abb. 5.1 zeigt einen Feldmultiplexer.

¹Im Englischen als „live-zero“ bezeichnet, da am Nullpunkt der Skala immer noch ein Signal vorhanden ist.

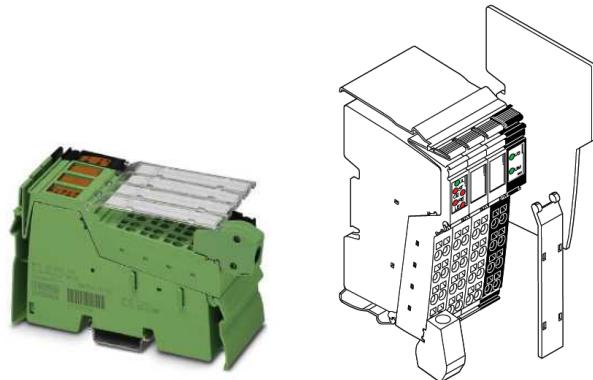


Abbildung 5.1.: Beispiel eines Feldmultiplexers.

5.1.3. Feldbusse

	Profibus	Interbus	AS-Interface	DeviceNet	CANopen	ControlNet
Anwendungsbereich	Feldebene	Feldebene	Sensor-/ Aktorenbene	Feldebene	Feldebene	Zellenebene
Besonderheiten	Mehrere Varianten mit abgestufter Funktionalität: DP, DPV1, DPV2, PA	Sehr schnelle, gute Diagnose	Schnell, kostengünstig, ideal für rein binäre E/A-Signale	Basiert auf CAN überträgt zusätzlich 24 Volt im Buskabel	Ideal für kleine Datenmengen und schnelle Synchronisation	Hohe Übertragungsleistung bei gleichbleibenden Buszykluszeiten
Reaktionszeiten	Mittel	Klein	Klein	Mittel	Mittel	Klein
Topologie	Linie	Ring	Linie, Stern, Baum	Linie	Linie	Linie
Übertragungsmedium	Kabel (2-adrig, geschirmt, verdrillt), Lichtwellenleiter	Kabel (5-adrig, geschirmt, verdrillt), Lichtwellenleiter	Kabel (2-adrig, ungeschirmt)	Kabel (4-adrig, geschirmt, verdrillt)	Kabel (4-adrig, geschirmt, verdrillt)	Koaxialkabel
Übertragungsverfahren (Physical Layer)	RS-485, LWL oder IEC 61158-2 (MBP)	RS-485, LWL	Alternierende Pulsmodulation	CAN	CAN	Modulation
Max. Teilnehmer	126	256	62	64	127	99
Max. Ausdehnung je Segment	100 m bei 12 Mbit/s, 1200 m bei 9,6 kbit/s	Zwischen 2 Geräten max. 400 m	100 m	100 m bei 500 kbit/s, 1 km bei 62,5 kbit/s	100 m bei 500 kbit/s, 1 km bei 62,5 kbit/s	1 km
Einsatz im Ex-Bereich	DP: nein, PA: ja	Nein	Bedingt	Nein	Nein	Nein

Tabelle 5.1.: Übersicht über verbreitete Feldbusse und ihre Eigenschaften.

Anmerkung: Bis zur Fertigstellung dieses Kapitels ist zur Klausurvorbereitung ausschließlich der entsprechende Foliensatz (AuS_10_Kommunikationssysteme.pdf) verbindlich.

6. Industrieroboter

6.1. Einführung

Die Bezeichnung Roboter geht ursprünglich auf das tschechische Wort *robo*ta zurück, welches mit „Arbeit“, „Frondienst“ oder „Zwangarbeit“ übersetzt werden kann. Im Drama „R.U.R - Rossum's Universal Robots“ von Karel Čapek treten in Tanks gezüchtete, menschenähnliche künstliche Wesen als billige Arbeitskräfte auf. Auf Vorschlag seines Bruders Josef Čapek verwendete er die Bezeichnung „robot“, welche in Folge viele Sprachen als Lehnwort in ihren alltäglichen Gebrauch übernahmen. Im österreichischen bzw. bairischen Sprachraum wurde mit Robot bzw. Robath bereits davor die zu leistende Fronarbeit von Leibeigenen bezeichnet.

Während sich die Verwendung des Begriffs Roboter ursprünglich ausschließlich auf humanoide - also in der Gestalt dem Mensch ähnliche - Roboter bezog, weichte sich diese enge Abgrenzung in der zweiten Hälfte des 20. Jahrhunderts auf und inkludierte auch programmierbare Automaten zur Handhabung und Fertigung. Abb. 6.1 zeigt eine exemplarische Aneinanderreihung von Roboterarten.



Abbildung 6.1.: Unterschiedliche Arten von Robotern: Industrieroboter, mobile, humanoide und androide Roboter.

Im Vergleich zu anderen Roboterarten sind Industrieroboter in der Regel fest an einen

Arbeitsplatz gebunden. Es gibt bis heute keine einheitliche Definition von Industrierobotern. In der VDI-Richtlinie VDI 2860 werden sie folgendermaßen definiert:

„Industrieroboter sind universell einsetzbare Bewegungsautomaten mit mehreren Achsen, deren Bewegungen hinsichtlich Bewegungsfolge und Wegen bzw. Winkeln frei (d. h. ohne mechanischen Eingriff) programmierbar und gegebenenfalls sensorgeführt sind. Sie sind mit Greifern, Werkzeugen oder anderen Fertigungsmitteln ausrüstbar und können Handhabungs- und/oder Fertigungsaufgaben ausführen.“

Davon abweichend existieren Definitionen der RIA (Robotic Industries Association) oder JARA (Japan Robot Association), die den Begriff des Roboters enger oder weiter fassen. Unabhängig davon bestehen Roboter stets aus fünf Komponentengruppen - Kinematik, Antriebe, Sensorik, Endeffektor und Steuerung -, welche erst im Zusammenspiel die Funktionsfähigkeit des Roboters herstellen. Diese sollen im Folgenden kurz vorgestellt werden.

6.1.1. Roboterkinematik

Unter der Roboterkinematik versteht man den mechanischen Aufbau des Roboters, welcher ihm die Bewegungsfreiheiten einräumt. Roboter werden dabei meist als Starrkörpersysteme beschrieben, deren Teilkörper durch Gelenke in ihrer Bewegungsfreiheit eingeschränkt sind. Jedem unabhängigen Gelenk wird ein *Bewegungsfreiheitsgrad* zugeordnet. Dabei wird zwischen Schubgelenken, welche translatorische Bewegungen ermöglichen, und Drehgelenken, welche rotatorische Bewegungen ermöglichen, unterschieden¹ (siehe Abb. 6.2). Die Anzahl der Freiheitsgrade ist für die möglichen Einsatzgebiete von Bedeutung: Soll beispielsweise der Endeffektor frei im Raum positioniert werden können, muss der Roboter über mindestens sechs Freiheitsgrade verfügen². Eine höhere Anzahl von Freiheitsgraden führt zu redundanten Posen, die zum Teil vorteilhaft sein können, jedoch den Steuerungsaufwand erhöhen.

Die Kombination von Starrkörpern und Gelenken wird auch als kinematische Kette bezeichnet. Dies geht aus der Beschreibung der Pose der einzelnen Starrkörperelemente durch Transformationen hervor, welche so sequentiell die Elemente miteinander „verketten“. Kinematische Ketten lassen sich so als Graphen darstellen, deren Knoten die einzelnen Starrkörperelemente und deren Kanten die Transformationen dazwischen dar-

¹Tatsächlich gibt es wesentlich mehr Typen von Gelenken. Es wird dabei nach F. Reuleaux zwischen Standardgelenken („niedrige Elementpaare“) und komplexen Gelenken („höhere Elementpaare“) unterschieden.

²Der euklidische, dreidimensionale Raum besitzt sechs Freiheitsgrade: drei translatorische und drei rotatorische Freiheitsgrade.

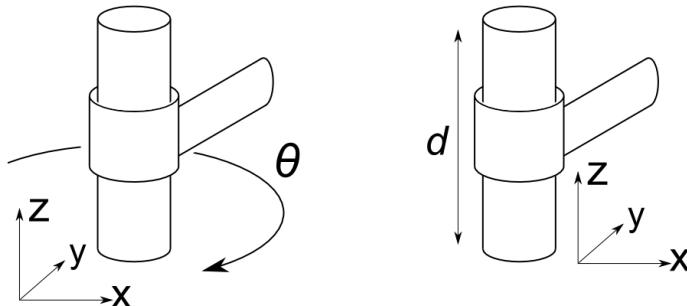


Abbildung 6.2.: Schematische Darstellung eines Dreh- und Schubgelenks.

stellen. Ist die kinematische Kette frei von Schleifen³, wird von einer *offenen* Kette bzw. einer seriellen Kinematik gesprochen. Ist jeder Starrkörper Element einer Schleife, wird von einer *vollständig geschlossenen* Kette bzw. paralleler Kinematik gesprochen. Befinden sich sowohl Schleifen als auch offene Enden in der kinematischen Kette, wird sie entsprechend als *teilweise geschlossen* bezeichnet (siehe Abb. 6.3).

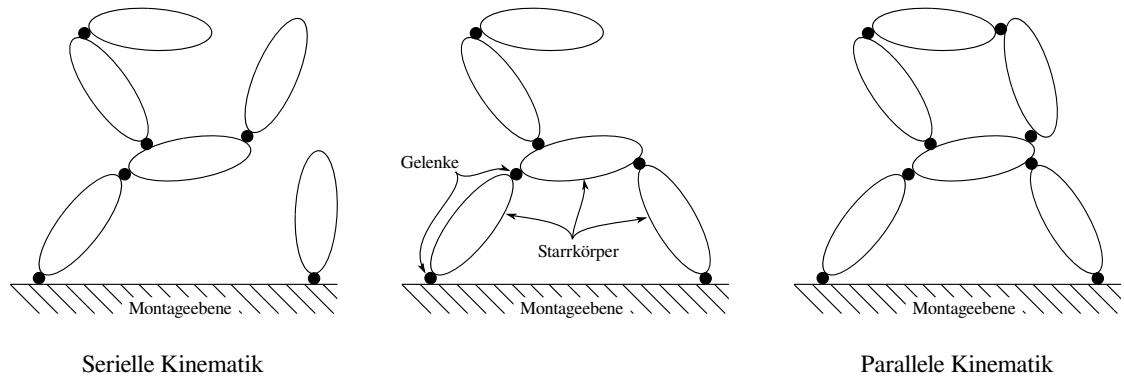
Parallele Kinematiken sind im Vergleich zu seriellen Kinematiken schwieriger zu beschreiben. Die Lösung des sog. vorwärtsskinematischen Problems - also die Berechnung der Pose des Endeffektors aus den Gelenksfreiheitsgraden – für die STEWART-GOUGH-Plattform (siehe Abb. 6.4) ist beispielsweise Gegenstand aktueller Forschung. Motiviert wird dies durch die guten mechanischen Eigenschaften paralleler Kinematiken: Sie sind wesentlich steifer, leichter und deshalb auch schneller als ihre seriellen Pendants. Erkauft werden diese Vorteile durch einen eingeschränkten Arbeitsbereich, d.i. die Menge der erreichbaren Endeffektorpunkte, und den vergleichsweise hohen Steuerungs- und Regelungsaufwand. Deswegen werden Industrieroboter bis heute meist als serielle Kinematik gebaut. Eine typische Ausnahme stellen Pick-and-Place-Roboter mit paralleler Kinematik - sogenannte Delta-Roboter - wie in Abb. 6.4 dar.

Je nach Aufgabenbereich, wie beispielsweise Lackieren, Schweißen, Sortieren, Verpacken oder Montieren, haben sich unterschiedliche Bauformen von Robotern mit unterschiedlichen Spezifikationen hinsichtlich Baugröße, Nutzlast, Arbeitsraum, Genauigkeit und Geschwindigkeit etabliert. Typische Konfigurationen und ihre Arbeitsbereiche sind in Abb. 6.5 dargestellt.

Eine häufig verwendete Bauform von Industrierobotern sind sog. SCARA-Roboter⁴ wie in Abb. 6.6 dargestellt. Sie bestehen aus einer seriellen Kinematik mit vier Freiheits-

³D. h. sie lässt sich in Baumstruktur darstellen.

⁴Selective Compliance Assembly Robot Arm (SCARA)



Serielle Kinematik

Parallele Kinematik

Abbildung 6.3.: Unterscheidung zwischen serieller und paralleler Kinematik.

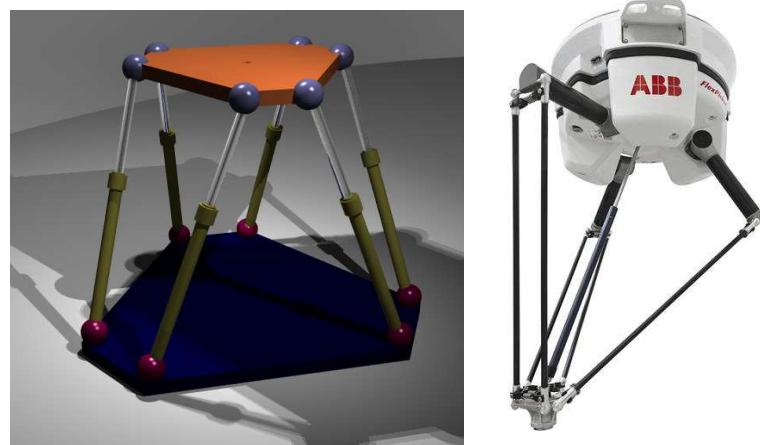


Abbildung 6.4.: Beispiele für parallele Kinematiken. Links: STEWART-GOUGH-Plattform; rechts: „Flexpicker“-Industrieroboter der Firma ABB.

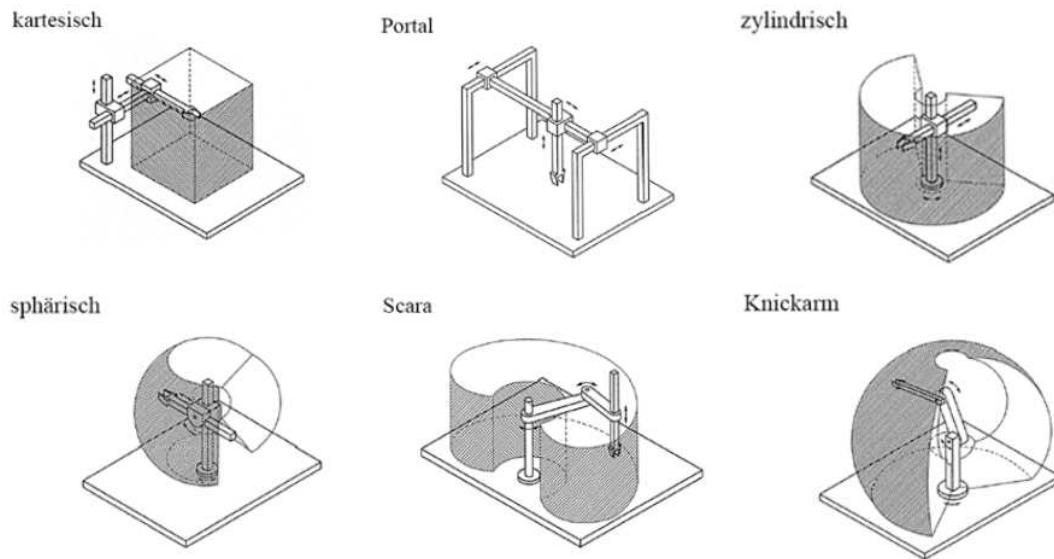


Abbildung 6.5.: Typische Konfigurationen von Industrierobotern und ihre Arbeitsbereiche.

graden - wovon drei rotatorisch und einer linear operieren - und besitzen einen nierenförmigen Arbeitsbereich. SCARA-Roboter werden vor allem zum Fügen von Bauteilen sowie für Pick-and-Place-Aufgaben verwendet. Gerade für Fügearbeiten sind die hohe mechanische Steifigkeit in z-Richtung und die daher möglichen, hohen Fügekräfte von Vorteil.

Eine andere typische Konfiguration von Industrierobotern stellen Knickarm-Roboter dar. Sie besitzen sechs rotatorische Freiheitsgrade und können so den Endeffektor im Inneren des Arbeitsbereiches in jede beliebige, räumliche Pose bewegen, weshalb sie universell einsetzbar sind. Beispielsweise werden sie für Schweiß- und Lackierarbeiten verwendet, bei welchen komplexen räumlichen Trajektorien gefolgt werden muss.

6.1.2. Antriebe

Entscheidend für die mögliche Dynamik und Tragfähigkeit eines Roboters ist die Aktivierung der Gelenke. Durch sie kann auf die Freiheitsgrade der Kinematik eingewirkt und so die Pose des Roboters verändert werden. Dafür werden elektrische, pneumatische und hydraulische Antriebe mit ihren spezifischen Vor- und Nachteilen verwendet. Eine Übersicht liefert Tab. 6.1.

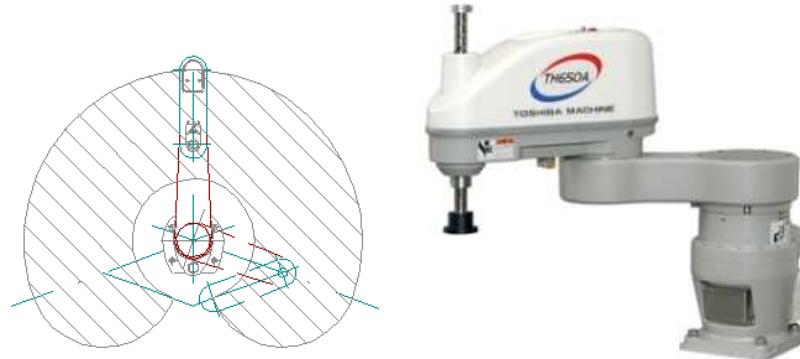


Abbildung 6.6.: Beispiel eines SCARA-Roboters mit nierenförmigem Arbeitsbereich.

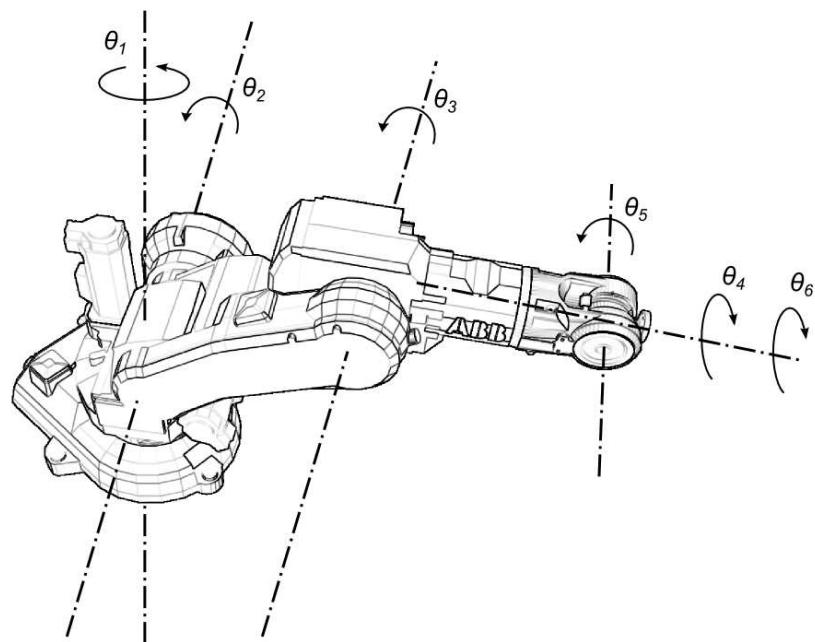


Abbildung 6.7.: Schematischer Aufbau und Drehachsen eines Knickarm-Industrieroboters.

	Vorteile	Nachteile
Elektrisch	Energie überall verfügbar Verzögerungsfreie Reaktion Einfache Konzeption Geräuscharm Betriebssicher	Kleine Momente und Kräfte Getriebe notwendig Hohe Instandhaltungskosten
Hydraulisch	Hohe Kräfte Stufenlose Geschwindigkeitsregelung Wegfallen von Bremseinrichtungen Hohe Steifigkeit	Hydraulikbehälter notwendig Lärmentwicklung Thermische Einflüsse
Pneumatisch	Druckluft oft vorhanden Günstiges Leistungsgewicht Wartungsfreundlich Explosionssicher	Kräftebeschränkt Geräuschentwicklung Bremsverschleiß

Tabelle 6.1.: Vergleich der Vor- und Nachteile von elektrischen, hydraulischen und pneumatischen Antrieben für Roboter.

6.1.3. Sensoren

Roboter benötigen Sensoren, um ihren eigenen Zustand zu erfassen und Informationen über ihre Umgebung zu erhalten. Man unterscheidet dementsprechend zwischen innerer und äußerer Sensorik.

Innere Sensoren erfassen den Zustand des Roboters. Darunter fällt die aktuelle Pose der kinematischen Kette, aber auch dynamische Größen wie Geschwindigkeit, Beschleunigung oder Krafteinwirkung der Gelenke. Erweiterte Informationen beinhalten beispielsweise die Funktionsfähigkeit einzelner Komponenten. Diese Informationen dienen der Positionsregelung und der Einhaltung sicherer Betriebsbereiche. Typische Sensoren sind inkrementelle und absolute Drehwinkelgeber induktiver oder optischer Bauart.

Externe Sensoren erfassen die Umgebung des Roboters. Sie werden benötigt, wenn der Arbeitsbereich des Roboters nicht oder nicht ausreichend genau bekannt ist. So können extern definierte Orte an- und Hindernisse umfahren werden. Einfache externe Sensorik verwendet kontaktierende Sensoren oder Näherungssensoren. Die wachsende Rechtleistung ermöglicht die zunehmende Verwendung von Bildverarbeitungssystemen. Auch die Erfassung von physikalischen Parametern wie Temperatur, Luftfeuchte, etc. erfolgt durch externe Sensoren.

Interne und externe Sensoren erfüllen also Aufgaben mit unterschiedlicher Zielsetzung. Roboter ohne äußere Sensorik sind gewissermaßen blind, weshalb die Planung der auszuführenden Bewegung durch „a priori“-Wissen über den Zustand des Arbeitsraums erfolgen muss. Durch äußere Sensoren lässt sich dieser Nachteil - auf Kosten aufwändiger Steuerungs- und Regelungsfunktionen - überkommen.

6.1.4. Endeffektor

Als Endeffektor bezeichnet man das am Ende des Roboterarms bzw. der kinematischen Kette liegende Glied, welches mit der Umgebung interagiert. Nicht als Endeffektoren gelten Komponenten, die der Mobilität des Roboters dienen.

Industrieroboter besitzen in der Regel einen Montageflansch, an welchem je nach Anwendungsgebiet unterschiedliche Endeffektoren angebracht werden können. Typische Endeffektoren für Produktionsanlagen sind Schweißköpfe, Lackierpistolen, diverse Greifer, Bohrer oder Fräser.

Die korrekte Arbeit des Endeffektors erfordert eine ausreichend hohe Genauigkeit des Roboters. Wesentliche Kenngrößen bei deren Beurteilung sind die Absolutgenauigkeit

sowie die Wiederholgenauigkeit, welche beide in Abb. 6.8 veranschaulicht sind.

Absolutgenauigkeit $\Delta g = |P_{soll} - P_{ist}|$ wobei P_{soll} als die Sollposition eines programmierten Raumpunktes und P_{ist} als die tatsächlich angefahrene Position definiert ist. Diese Abweichung ergibt sich aus zufälligen und systematischen Fehlern. Letztere ergeben sich beispielsweise aus systematischen Fehlern der internen Sensoren, Fertigungstoleranzen der Kinematik oder Vereinfachungen am mathematischen Modell.

Wiederholgenauigkeit $\Delta w = |P_{soll}^* - P_{ist}|$ wobei P_{soll}^* einen geteachten oder geführt angefahrenen Raumpunkt bezeichnet. Systematische Fehler spielen in diesem Fall keine Rolle. Üblicherweise gilt für Industrieroboter, dass $\Delta w \ll \Delta g$.

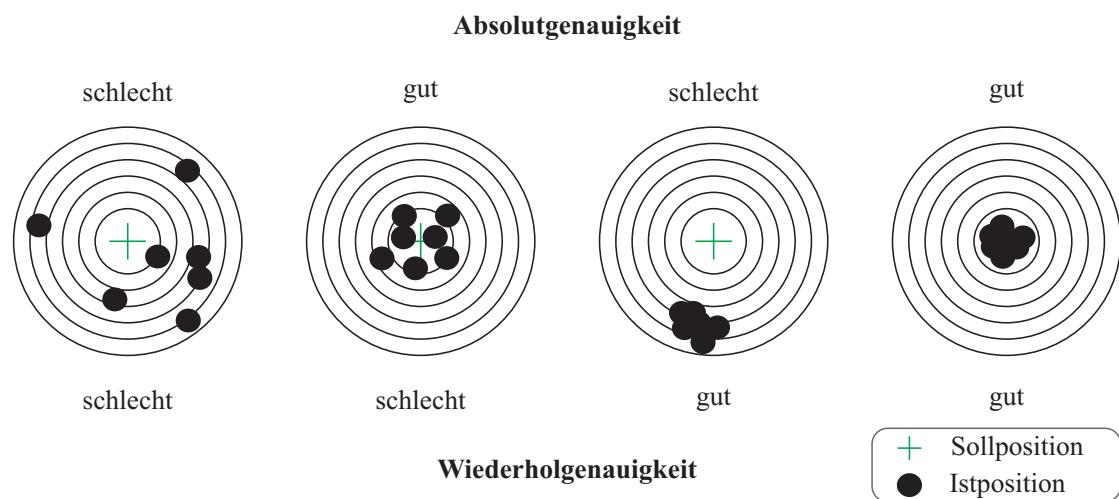


Abbildung 6.8.: Absolutgenauigkeit und Wiederholgenauigkeit

6.1.5. Steuerung und Regelung

Die notwendige Informationsverarbeitung, um einen Industrieroboter einsetzen zu können, wird von Computersystemen übernommen. Die dahinter liegende regelungstechnische Aufgabe kann dabei in zwei Teilprobleme zerlegt werden:

Wege- und Bahnplanung Aus der externen Aufgabenstellung - also der gewünschten Anwendung des Roboters - folgt eine mehr oder minder genaue Beschreibung der Bewegung des Roboters. Die Wahl eines Weges, welche die Aufgabenstellung hinreichend erfüllt, bezeichnet man als Wege- bzw. Bahnplanung⁵. Dieses Problem ist

⁵Wir werden später zwischen Wege- und Bahnplanung unterscheiden.

im Allgemeinen unterbestimmt; beispielsweise wird die Bewegung von einem Punkt P_1 zu einem anderen Punkt P_2 von unendlich vielen Trajektorien erfüllt. Gleichzeitig unterliegt die Bewegung des Industrieroboters physikalischen Beschränkungen wie den endlichen Stellgrößen der Antriebe oder den Limitierungen der Kinematik. Ziel der Bahnplanung ist ein Sollgrößenverlauf für jedes aktuierte Gelenk des Roboters.

Regelung Die Regelung versucht, den von der Bahnplanung vorgegebenen Trajektorien unter Einhaltung der Stellgrößenbeschränkungen bestmöglich zu folgen. Dafür existieren eine Reihe von Methoden wie beispielsweise *PD*- oder *Computed-Torque*-Reglerentwürfe, welche sich an der Struktur der Euler-Lagrange-Gleichungen⁶ orientieren. Hierzu sei auf die Vorlesung *Regelungssysteme 2* verwiesen.

Sowohl für die Bahnplanung als auch die Regelung des Roboters ist die Ausstattung mit Sensoren essenziell. Der klassische, historisch älteste Zugang verwendet nur interne Sensoren (siehe Abb. 6.9 oben). Der Arbeitsraum des Roboters ist frei bzw. nur vom Werkstück und anderen Robotern belegt, deren Positionen jedoch genau bekannt ist. Die auszuführenden Bewegungen können durch dieses implizite Wissen im Vorhinein, man sagt auch *offline*, vorausberechnet werden. Diese Methode ist einfach und kommt mit geringstem algorithmischen und sensorischen Aufwand aus.

Im Gegensatz dazu werden äußere Sensoren verwendet, um auf veränderliche Arbeitsumgebungen und Werkstücke eingehen zu können. Die auszuführenden Bewegungen sind nun von äußeren Messgrößen abhängig. Dabei kann zwischen zwei Fällen unterschieden werden:

Vielfach will man extern definierte Punkte anfahren, deren Lage in etwa bekannt ist. Beispielsweise will man einen definierten Abstand (oder Kontakt) zu einem Werkstück einnehmen, dessen Position einer gewissen Schwankungsbreite unterliegt. Dies kann durch Näherungs- oder Kontaktsensoren erreicht werden, welche in die Regelung einbezogen werden (siehe Abb. 6.9 Mitte). Die Bahnplanung erfolgt immer noch *offline*. Derartige Methoden werden aktuell eingesetzt.

Durch die Verwendung von bildgebenden Sensoren ist man in der Lage, den gesamten Arbeitsraum zu erfassen. Dadurch können sich Fremdobjekte und Werkstücke im Arbeitsraum befinden, deren Lage nicht bekannt ist. Die Bahnplanung muss nun *online*, also während der Ausführung, geschehen, um den gewünschten Zielpunkt ohne Kollision anzufahren (siehe Abb. 6.9 unten). Der Rechenaufwand für diese Methoden ist beträchtlich; sowohl zur Verarbeitung der Bilddaten als auch zur laufenden Berechnung günstiger Trajektorien. Sie sind aktuell Gegenstand der Forschung.

⁶siehe Vorlesung *Modellbildung* aus dem Bachelorstudium Elektrotechnik.

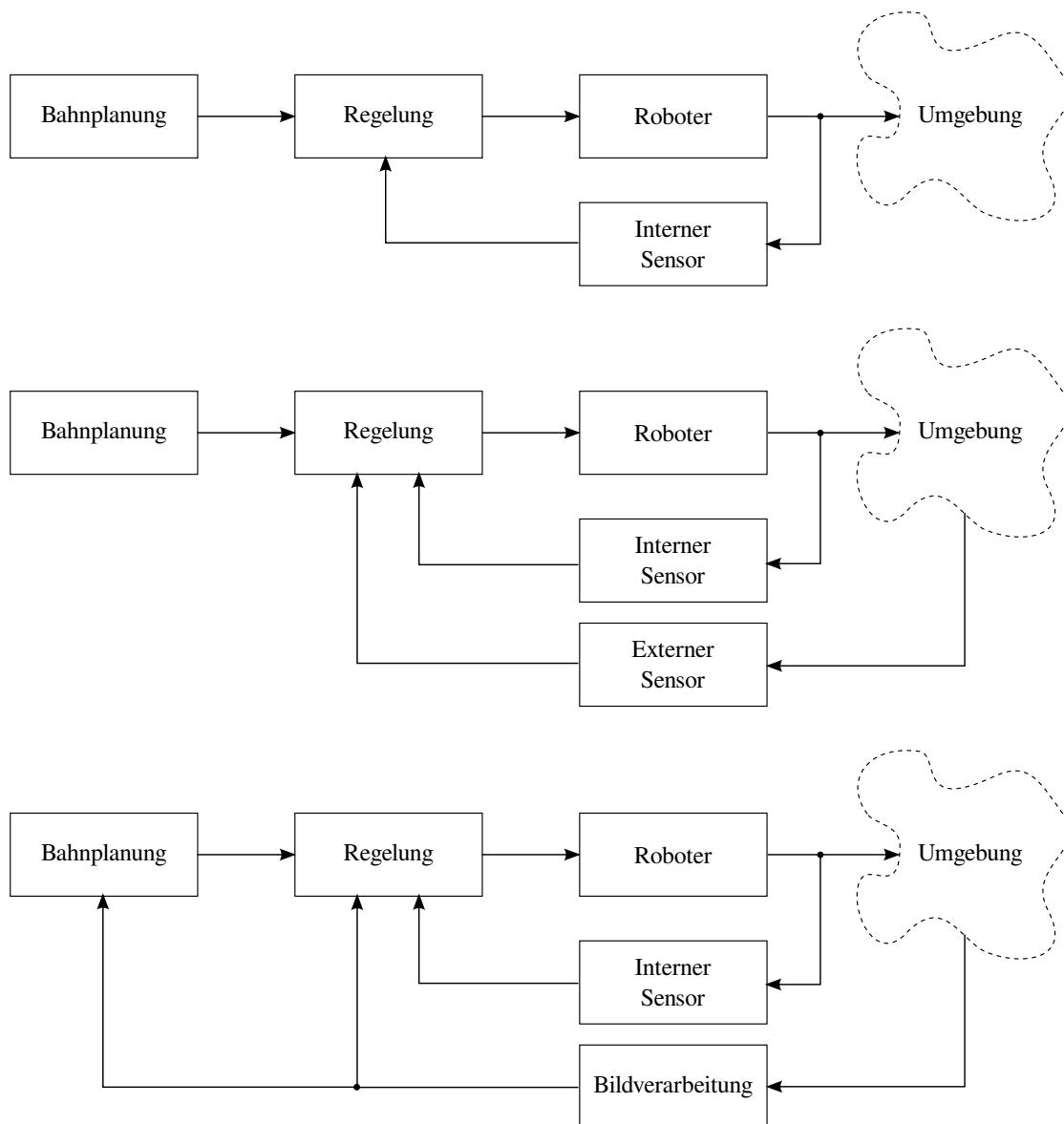


Abbildung 6.9.: Veränderung der Regelkreisstruktur durch externe Sensoren. Oben: Keine Information über die Umgebung; Mitte: Führung zu externen definierten Orten durch Näherungssensoren; Unten: Verwendung von bildgebenden Sensoren.

6.1.6. Programmierung

Bei kommerziell erhältlichen Robotern ist die notwendige Software zur Bahnplanung und Regelung bereits inkludiert. Der Endbenutzer muss die gewünschten Bewegungsabläufe nur noch mittels spezialisierter Programmiersprachen konfigurieren. Beispiele für solche Sprachen sind RAPID der Firma ABB oder KRL (KUKA Robot Language) der Firma KUKA. Zur Erstellung solcher Programme stehen zwei Verfahren zur Verfügung:

Online-Programmierung Der Programmierer befindet sich direkt vor Ort am Roboter und leitet im sog. Teach-In-Modus dem Roboterarm mittels Steuerkonsole Punkt für Punkt die Bewegungssequenzen an. Alternativ kann auch eine kontinuierliche, geführte Bewegung als Sequenz gespeichert werden, was beispielsweise für Lackiervorgänge verwendet wird. Die Wiederholgenauigkeit geteachter Bewegungen ist sehr gut, doch führt ein Teach-In zu Standzeiten der Anlage.

Offline-Programmierung Durch detaillierte CAD-Modelle des Roboters und der Umgebung ist es möglich, auf Basis eines Simulationsmodells Bewegungen am Computer zu planen. Für die Funktionsfähigkeit solcher Programme ist die Genauigkeit der CAD-Modelle essenziell, dafür liefern sie dann gute Absolutgenauigkeit. Da Offline-Programmierung ohne Hardware auskommt, ist sie flexibel und kommt ohne Standzeiten der Anlage aus.

Praktisch werden oft Kombinationen beider Methoden eingesetzt. So kann z. B. der Bewegungsablauf im Vorhinein geplant und programmiert werden und lediglich einige wenige Bezugspunkte werden geteacht.

6.2. Kinematik

Die Kinematik betrachtet als Teilgebiet der Mechanik die Bewegung von Punkten und Körpern. Sie beschreibt den Zusammenhang zwischen der Anordnung eines Körpers und den durch die physische Anordnung gegebenen kinematischen Freiheitsgraden. Im Gegensatz zur Dynamik beschäftigt sich die Kinematik nicht mit den Ursachen dieser Bewegung, also den Kräften und Momenten, sondern stellt vielmehr Zwangsbedingungen für diese dar.

In Anwendungen der Robotik genügt meist die Beschreibung der einzelnen Roboterglieder.

der als bewegte Starrkörper im euklidischen Vektorraum⁷ \mathbb{R}^3 . Dazu wird jedem Starrkörper S_i mit einem ihm zugeordneten Koordinatensystem Σ_i identifiziert. Die Beschreibung der Starrkörperbewegung reduziert sich dadurch auf eine Koordinatentransformation.

6.2.1. Koordinatensysteme

Unter einem Koordinatensystem $\Sigma_a = (0_a x_a y_a z_a)$ im Vektorraum \mathbb{R}^3 verstehen wir einen ausgezeichneten Nullpunkt 0_a sowie eine Menge an den Koordinatenachsen x_a, y_a, z_a ausgerichteten orthonormalen Basisvektoren $e_{x_a}, e_{y_a}, e_{z_a}$ wie in Abb. 6.10 dargestellt.

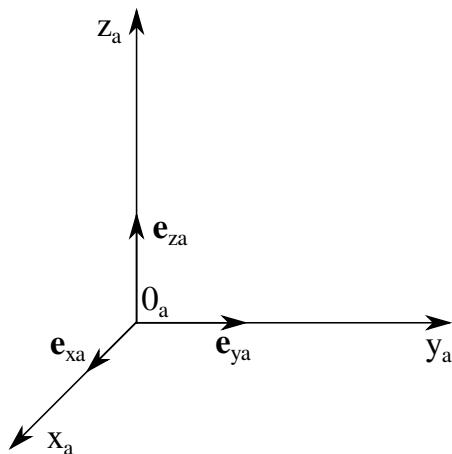


Abbildung 6.10.: Koordinatensystem mit orthonormalen Basisvektoren.

Bewegungen als Abbildungen zwischen zwei Koordinatensystemen müssen aus physikalischen Überlegungen längen- und winkeltreue Bijektion sein, welche die Orientierung des Raums erhalten⁸. Es lässt sich zeigen, dass derartige Abbildungen aus Rotationen und Translationen zusammengesetzt werden können.

⁷Der euklidische Vektorraum \mathbb{R}^n als Modell für den physikalischen (vorrelativistischen) Raum unterscheidet nicht zwischen Punkten und Vektoren. Dieser etwas unbefriedigende Umstand hat zur Schaffung des euklidischen Punktraumes \mathbb{E}^n Anlass gegeben, welcher gelegentlich in kinematischen Betrachtungen verwendet wird.

⁸Solche Transformationen werden auch gleichsinnige Kongruenztransformationen genannt. Da die Hintereinanderausführung mehrerer Bewegungen wieder eine Bewegung ist, bilden Bewegungen eine Gruppe, welche im dreidimensionalen Raum mit $SE(3)$ für *special euclidian group* bezeichnet wird.

6.2.2. Translationen

Angenommen die Nullpunkte zweier gegeneinander nicht verdrehter Koordinatensysteme $\Sigma_0 = (0_0 x_0 y_0 z_0)$ und $\Sigma_1 = (0_1 x_1 y_1 z_1)$ sind durch den Verschiebungsvektor \mathbf{d}_0^1 wie in Abb. 6.11 dargestellt miteinander verbunden. Ein Punkt P besitzt die Ortsvektoren⁹ \mathbf{p}_0 im System Σ_0 und \mathbf{p}_1 im System Σ_1 . Es gilt daher

$$\mathbf{p}_0 = \mathbf{p}_1 + \mathbf{d}_0^1. \quad (6.1)$$

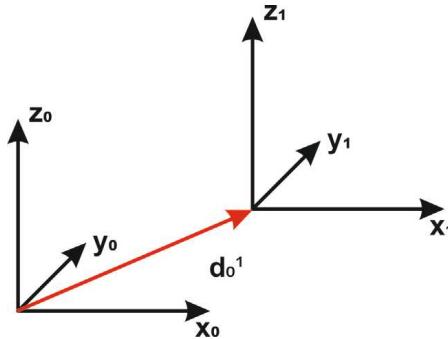


Abbildung 6.11.: Translation eines Koordinatensystems.

6.2.3. Rotationen

Angenommen zwei Koordinatensysteme besitzen denselben Nullpunkt, also $\Sigma_0 = (0x_0 y_0 z_0)$ und $\Sigma_1 = (0x_1 y_1 z_1)$. Der Ortsvektor eines Punktes P lässt sich dann in der Form

$$\mathbf{p}_0 = p_{0x} \mathbf{e}_{x_0} + p_{0y} \mathbf{e}_{y_0} + p_{0z} \mathbf{e}_{z_0} \quad (6.2)$$

bzw.

$$\mathbf{p}_1 = p_{1x} \mathbf{e}_{x_1} + p_{1y} \mathbf{e}_{y_1} + p_{1z} \mathbf{e}_{z_1} \quad (6.3)$$

darstellen. Wegen der Orthogonalität der Basisvektoren lassen sich die Koeffizienten einfach durch die Projektion bestimmen. Für $i \in \{x, y, z\}$ gilt daher

$$p_{0i} = \mathbf{e}_{i_0}^T \mathbf{p}_0 = \mathbf{e}_{i_0}^T \mathbf{p}_1 = p_{1x} \mathbf{e}_{i_0}^T \mathbf{e}_{x_1} + p_{1y} \mathbf{e}_{i_0}^T \mathbf{e}_{y_1} + p_{1z} \mathbf{e}_{i_0}^T \mathbf{e}_{z_1}. \quad (6.4)$$

⁹Zur Nomenklatur: Der tiefgestellte Index kennzeichnet hier und im Folgenden immer das Bezugssystem der Größe. Der Verschiebungsvektor \mathbf{d}_0^1 bezeichnet dementsprechend die Verschiebung von Σ_1 gegenüber Σ_0 dargestellt im System Σ_0 .

Schreibt man (6.4) in Matrixform, ergibt sich¹⁰

$$\underbrace{\begin{bmatrix} p_{0x} \\ p_{0y} \\ p_{0z} \end{bmatrix}}_{\mathbf{p}_0} = \underbrace{\begin{bmatrix} \mathbf{e}_{x_0}^T \mathbf{e}_{x_1} & \mathbf{e}_{x_0}^T \mathbf{e}_{y_1} & \mathbf{e}_{x_0}^T \mathbf{e}_{z_1} \\ \mathbf{e}_{y_0}^T \mathbf{e}_{x_1} & \mathbf{e}_{y_0}^T \mathbf{e}_{y_1} & \mathbf{e}_{y_0}^T \mathbf{e}_{z_1} \\ \mathbf{e}_{z_0}^T \mathbf{e}_{x_1} & \mathbf{e}_{z_0}^T \mathbf{e}_{y_1} & \mathbf{e}_{z_0}^T \mathbf{e}_{z_1} \end{bmatrix}}_{\mathbf{R}_0^1} \underbrace{\begin{bmatrix} p_{1x} \\ p_{1y} \\ p_{1z} \end{bmatrix}}_{\mathbf{p}_1}. \quad (6.5)$$

Diese Bewegung stellt eine Rotation dar. Die (3×3) -Matrix \mathbf{R}_0^1 dreht den Vektor \mathbf{p} von Σ_1 nach Σ_0 . Auf analoge Weise lässt sich die umgekehrte Matrix \mathbf{R}_1^0 darstellen. Aus der Beziehung

$$\mathbf{p}_0 = \mathbf{R}_0^1 \mathbf{p}_1 = \underbrace{\mathbf{R}_0^1 \mathbf{R}_1^0}_{\mathbf{I}_3} \mathbf{p}_0 \quad (6.6)$$

mit der (3×3) -Einheitsmatrix \mathbf{I}_3 und der Darstellung in (6.5) folgt die Orthogonalität

$$(\mathbf{R}_0^1)^{-1} = \mathbf{R}_1^0 = (\mathbf{R}_0^1)^T \quad (6.7)$$

der Matrix \mathbf{R}_0^1 . Allgemein gilt $\det(\mathbf{R}_0^1) = \pm 1$ für Matrizen der Bauform (6.5). Um die Orientierung des Koordinatensystems zu erhalten, muss $\det(\mathbf{R}_0^1) = 1$ gelten. Diese Matrizen werden Dreh- oder Rotationsmatrizen genannt. Da die Multiplikation zweier Rotationsmatrizen wiederum eine Rotationsmatrix ergibt, die Matrixmultiplikation assoziativ ist, ein Einselement (die Einheitsmatrix \mathbf{I}_3) und zu jeder Rotationsmatrix ein inverses Element existiert, bilden die Rotationsmatrizen mit der Matrixmultiplikation eine Gruppe. Diese wird als $\text{SO}(3)$ für *special orthogonal group of order 3* bezeichnet.

Im Allgemeinen drehen Rotationsmatrizen um eine beliebig im Raum ausgerichtete Drehachse. Besonders einfach darstellbar sind die sog. elementaren Rotationsmatrizen, welche um eine der Koordinatenachsen drehen. Eine Drehung um die x-Achse in mathematisch positive Richtung kann durch die Matrix

$$\mathbf{R}_{x,\Phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\Phi) & -\sin(\Phi) \\ 0 & \sin(\Phi) & \cos(\Phi) \end{bmatrix} \quad (6.8)$$

erwirkt werden. Um sich Schreibaufwand zu ersparen, werden die Winkelfunktionen

¹⁰Es wird hier nicht zwischen der Koeffizientendarstellung und der Entwicklung der Vektors \mathbf{p} unterschieden. Dies ist unkritisch, solange man sich bewusst ist, in welchem Koordinatensystem die Koeffizienten dargestellt sind, was durch die Kennzeichnung im Index ersichtlich ist.

$\cos(\Phi)$ und $\sin(\Phi)$ oft abgekürzt als c_Φ und s_Φ geschrieben. Die verbleibenden elementaren Drehmatrizen um die y- und z-Achse lassen sich somit als

$$\mathbf{R}_{y,\theta} = \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \quad \text{und} \quad \mathbf{R}_{z,\Psi} = \begin{bmatrix} c_\Psi & -s_\Psi & 0 \\ s_\Psi & c_\Psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6.9)$$

schreiben.

Zu beachten ist, dass die Gruppe $\text{SO}(3)$ nicht kommutativ ist, also i. A.

$$\mathbf{R}_A \mathbf{R}_B \neq \mathbf{R}_B \mathbf{R}_A \quad (6.10)$$

gilt. Dieser Sachverhalt ist auch anschaulich unmittelbar klar (Abb. 6.12): Angenommen

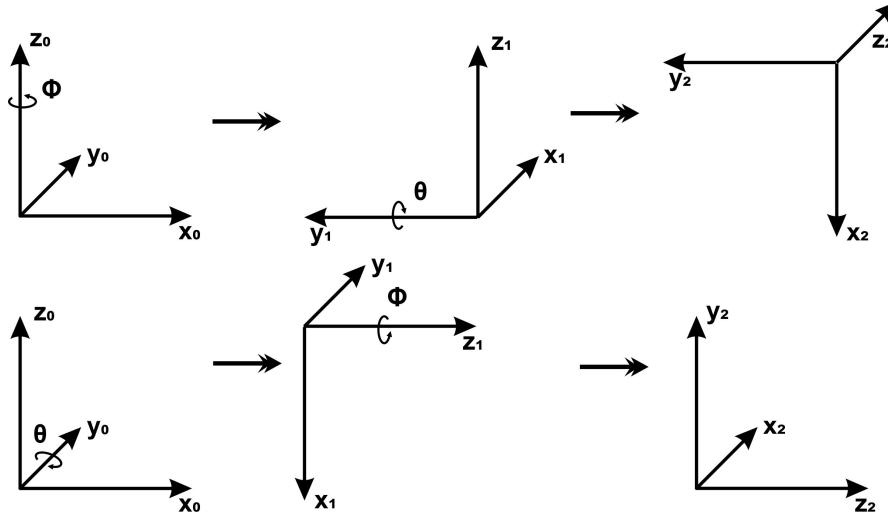


Abbildung 6.12.: Die Multiplikation von Rotationsmatrizen ist nicht kommutativ.

drei Koordinatensysteme $\Sigma_0 = (0x_0y_0z_0)$, $\Sigma_1 = (0x_1y_1z_1)$, $\Sigma_2 = (0x_2y_2z_2)$ mit identischem Nullpunkt werden durch die Drehungen $\mathbf{R}_0^1 = \mathbf{R}_{z,90^\circ}$ und $\mathbf{R}_1^2 = \mathbf{R}_{y,90^\circ}$ verbunden. In der oberen Reihe der Abbildung wird durch

$$\mathbf{p}_0 = \mathbf{R}_0^1 \mathbf{p}_1 = \mathbf{R}_0^1 \mathbf{R}_1^2 \mathbf{p}_2 = \mathbf{R}_0^2 \mathbf{p}_2 \quad (6.11)$$

die übliche Verkettung der Drehungen

$$\mathbf{R}_0^2 = \mathbf{R}_0^1 \mathbf{R}_1^2 \quad (6.12)$$

gebildet. Das Ergebnis weicht zur unteren Reihe, wo in umgekehrter Reihenfolge verfahren wurde, ab. Während im ersten Fall stets um die betreffende Achse des *aktuellen* Koordinatensystems gedreht wurde, scheint es, als wäre im zweiten Fall stets um die festen Achsen des *ursprünglichen* Systems Σ_0 gedreht worden. Dies ist in der Tat immer der Fall, wenn man die Reihenfolge der Multiplikation umkehrt. Angenommen ein Vektor \mathbf{p}_0 soll zuerst um die z-Achse und danach um die y-Achse des ursprünglichen Systems Σ_0 gedreht werden. Dazu wird zuerst um die z-Achse gedreht, also $\mathbf{p}_0 = \mathbf{R}_{z,90^\circ} \mathbf{p}_1$. Danach wird \mathbf{p}_1 durch einen Umweg über Σ_0 um dessen y-Achse gedreht, also

$$\mathbf{p}_1 = (\mathbf{R}_{z,90^\circ})^{-1} \mathbf{R}_{y,90^\circ} \mathbf{R}_{z,90^\circ} \mathbf{p}_2. \quad (6.13)$$

Daraus folgt

$$\mathbf{p}_0 = \mathbf{R}_{z,90^\circ} (\mathbf{R}_{z,90^\circ})^{-1} \mathbf{R}_{y,90^\circ} \mathbf{R}_{z,90^\circ} \mathbf{p}_2 = \mathbf{R}_{y,90^\circ} \mathbf{R}_{z,90^\circ} \mathbf{p}_2. \quad (6.14)$$

6.2.4. Parametrierung von Rotationen

Sind zwei Koordinatensysteme durch eine allgemeine Drehung miteinander verbunden, stellt sich die Frage, wie eine derartige Drehung einfach dargestellt werden kann. Die neun Einträge einer (3×3) -Rotationsmatrix \mathbf{R} sind schon aufgrund der obigen Eigenschaften von Rotationsmatrizen nicht unabhängig voneinander wählbar. Es lässt sich für die Gruppe $\text{SO}(n)$ allgemein zeigen, dass diese $\frac{n(n-1)}{2}$ Freiheitsgrade besitzt. Für den dreidimensionalen Fall folgt daraus, dass sich eine Rotation durch drei Freiheitsgrade eindeutig parametrieren lässt.

Eulersche Winkel

Eine Variante zur Parametrierung sind die sog. Eulerschen Winkel.

Satz 6.1 (Euler-Parametrierung). Eine allgemeine Rotationsmatrix $\mathbf{R}_0^1 \in \text{SO}(3)$ lässt sich durch das Euler-Tripel (Φ, θ, ψ) als Kombination elementarer Rotationen

$$\mathbf{R}_0^1 = \mathbf{R}_{z,\Phi} \mathbf{R}_{y,\theta} \mathbf{R}_{z,\psi} \quad (6.15)$$

darstellen (siehe Abb. 6.13). Dabei wird zuerst um die z-Achse des Ursprungssystems um Φ gedreht, danach wird die aktuelle y-Achse um θ und letztlich um ψ um die nun aktuelle z-Achse gedreht. Setzt man die elementaren Rotationsmatrizen ein, folgt die

parametrierte Rotationsmatrix

$$\mathbf{R}_0^1 = \begin{bmatrix} c_\Phi c_\theta c_\psi - s_\Phi s_\psi & -c_\Phi c_\theta s_\psi - s_\Phi c_\psi & c_\Phi s_\theta \\ s_\Phi c_\theta c_\psi + c_\Phi s_\psi & -s_\Phi c_\theta s_\psi + c_\Phi c_\psi & s_\Phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}. \quad (6.16)$$

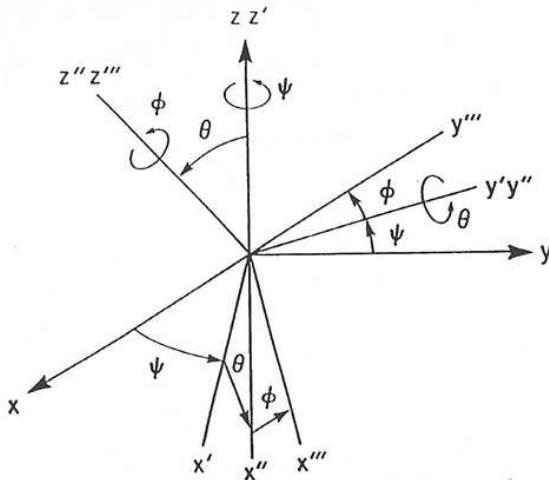


Abbildung 6.13.: Illustration der Eulerschen Winkel.

Angenommen eine bekannte Matrix \mathbf{R} ist eine Rotationsmatrix und besitzt die bekannten Einträgen $\mathbf{R} = (r_{ij})$ mit $i, j \in 1, 2, 3$. Durch Koeffizientenvergleich mit (6.16) lassen sich die Eulerschen Winkel durch

$$\theta = \text{atan2}\left(\sqrt{r_{31}^2 + r_{32}^2}, r_{33}\right), \quad (6.17)$$

$$\Phi = \text{atan2}\left(\frac{r_{23}}{\sin(\theta)}, \frac{r_{13}}{\sin(\theta)}\right) \quad \text{und} \quad (6.18)$$

$$\psi = \text{atan2}\left(\frac{r_{32}}{\sin(\theta)}, -\frac{r_{31}}{\sin(\theta)}\right) \quad (6.19)$$

mit der Funktion

$$\text{atan2}(y, x) = \begin{cases} \arctan\left(\frac{y}{x}\right), & x > 0, \\ \arctan\left(\frac{y}{x}\right) + \pi, & x < 0 \wedge y \geq 0, \\ \arctan\left(\frac{y}{x}\right) - \pi, & x < 0 \wedge y < 0, \\ \frac{\pi}{2}, & x = 0 \wedge y > 0, \\ -\frac{\pi}{2}, & x = 0 \wedge y < 0, \\ \text{undefiniert,} & x = y = 0, \end{cases} \quad (6.20)$$

berechnen, welche durch die Verwendung von zwei Argumenten eine vorzeichenrichtige Erweiterung der gewöhnlichen Arcustangens-Funktion darstellt.

Roll-Pitch-Yaw-Winkel

Eine Alternative zu den Eulerschen Winkeln stellen die sog. Roll-Pitch-Yaw-Winkel¹¹ (RPY-Winkel) dar. Anders als bei den Eulerschen Winkeln beziehen sich diese auf einen festen Referenzrahmen.

Satz 6.2 (Roll-Pitch-Yaw-Parametrisierung). Eine allgemeine Rotationsmatrix $\mathbf{R}_0^1 \in \text{SO}(3)$ lässt sich durch das RPY-Tripel (Φ, θ, ψ) als

$$\mathbf{R}_0^1 = \mathbf{R}_{z,\Phi} \mathbf{R}_{y,\theta} \mathbf{R}_{x,\psi} \quad (6.21)$$

darstellen, deren Winkel sich wie in Abb. 6.14 auf das System Σ_0 beziehen. Φ wird als Roll-, θ als Nick(Pitch)- und ψ als Gier(Yaw)-Winkel bezeichnet. Eine explizite Darstellung ergibt

$$\mathbf{R}_0^1 = \begin{bmatrix} c_\Phi c_\theta & c_\Phi s_\theta s_\psi - s_\Phi c_\psi & c_\Phi s_\theta c_\psi + s_\Phi s_\psi \\ s_\Phi c_\theta & s_\Phi s_\theta s_\psi + c_\Phi c_\psi & s_\Phi s_\theta c_\psi - c_\Phi s_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}. \quad (6.22)$$

Für eine gegebene Rotationsmatrix $\mathbf{R} = (r_{ij})$ mit $i, j \in \{1, 2, 3\}$ lassen sich analog zu den Eulerschen Winkel auch die RPY-Winkel durch

$$\theta = \text{atan2}\left(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}\right) \quad (6.23)$$

$$\Phi = \text{atan2}\left(\frac{r_{21}}{\cos(\theta)}, \frac{r_{11}}{\cos(\theta)}\right) \quad (6.24)$$

$$\psi = \text{atan2}\left(\frac{r_{32}}{\cos(\theta)}, \frac{r_{33}}{\cos(\theta)}\right) \quad (6.25)$$

bestimmen.

¹¹Ins Deutsche übersetzt werde sie auch als Roll-Nick-Gier-Winkel bezeichnet. Schon der Bezeichnung nach erkennt man ihren Bezug zur Luftfahrt.

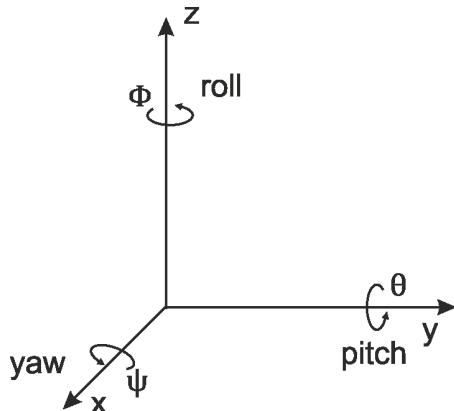


Abbildung 6.14.: Illustration der Roll-Pitch-Yaw-Winkel.

Kritische Punkte

Die Parametrierung einer Drehung durch Euler- oder RPY-Winkel hat den großen Vorteil, recht anschaulich zu sein. Beide Darstellungen erweisen sich jedoch an einzelnen Punkten - sog. *kritischen* Punkten - als problematisch. Dieses Phänomen wird als *Kardanische Blockade* (engl. gimbal lock) bezeichnet.

Betrachten wir beispielsweise den Fall einer eulerschen Parametrierung. Die Rotationsmatrix ist gegeben durch $\mathbf{R} = \mathbf{R}_{z,\Phi}\mathbf{R}_{y,\theta}\mathbf{R}_{z,\psi}$. Für den Fall $\theta = 0$ ergibt sich die Rotationsmatrix

$$\mathbf{R}|_{\theta=0} = \begin{bmatrix} \cos(\Phi + \psi) & -\sin(\Phi + \psi) & 0 \\ \sin(\Phi + \psi) & \cos(\Phi + \psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (6.26)$$

Die Rotationsmatrix \mathbf{R} dreht nur noch mit $\Phi + \Psi$ um die z-Achse, besitzt also nur noch einen Freiheitsgrad, obwohl wir nur einen Freiheitsgrad (durch Festlegen von θ) gesetzt hatten. Während der Winkel zur Drehung um die z-Achse also durch 2 Freiheitsgrade bestimmt sind, weshalb unendlich viele Kombinationen von Φ und Ψ für eine vorgegebene Drehung existieren, fehlt gewissermaßen ein Freiheitsgrad für Drehungen abseits der z-Achse.

Etwas formaler betrachtet folgt dies aus dem Umstand, dass der an einem kritischen Punkt $P_1 = (\Phi_1, \theta_1, \psi_1)$ der Tangentialraum der Abbildung $(\Phi, \theta, \psi) \rightarrow \mathbf{R}$, wie durch (6.15) oder (6.21) gegeben, eine Dimension verloren; d. h. es gilt

$$\dim\left(\frac{\partial \mathbf{R}}{\partial \Phi}\Big|_{P_1}, \frac{\partial \mathbf{R}}{\partial \theta}\Big|_{P_1}, \frac{\partial \mathbf{R}}{\partial \psi}\Big|_{P_1}\right) < 3 \quad (6.27)$$

An den kritischen Punkten verlieren Euler- und RPY-Parametrierung also ihre Fähigkeit, alle Rotationen der Drehgruppe $\text{SO}(3)$ darzustellen. Bei Kardangelenken führt dies zu einer mechanischen Blockade, da die fehlenden Richtungen vom Kardangelenk nicht mehr durchgeführt werden können.

Quaternionen

Als Konsequenz werden oftmals andere mathematische Konstrukte zur Parametrierung der $\text{SO}(3)$ herangezogen - beispielsweise die sog. Quaternionen \mathbb{H} . Sie stellen eine Erweiterung des Prinzips der komplexen Zahlen \mathbb{C} auf *drei* imaginäre Einheiten i, j und k dar. Eine Quaternion $q \in \mathbb{H}$ kann durch diese Einheiten als

$$q = q_1 + iq_2 + jq_3 + kq_4 \quad (6.28)$$

dargestellt werden¹². Wie bei den komplexen Zahlen werden folgende multiplikativen Rechenregeln vereinbart:

$$i^2 = j^2 = k^2 = -1 \quad (6.29)$$

$$ij = +k \quad jk = +i \quad ki = +j \quad (6.30)$$

$$ji = -k \quad kj = -i \quad ik = -j. \quad (6.31)$$

Die damit eingeführte Multiplikation auf \mathbb{H} ist distributiv, assoziativ, aber i. A. nicht kommutativ, weshalb die Quaternionen lediglich einen *Schiefkörper* darstellen. Viele weitere Konstrukte lassen sich - in etwas modifizierter Form - auf natürliche Weise auf Quaternionen erweitern. So gilt

$$\text{Re}\{q\} = q_1, \quad (6.32)$$

$$\text{Im}\{q\} = iq_2 + jq_3 + kq_4, \quad (6.33)$$

$$\bar{q} = q_1 - iq_2 - jq_3 - kq_4 \quad \text{und} \quad (6.34)$$

$$|q| = q\bar{q} = \sqrt{q_1^2 + q_2^2 + q_3^2 + q_4^2}. \quad (6.35)$$

Eine Besonderheit der Quaternionen liegt darin, dass die quadratischen Formen von Einheitsquaternionen (also mit $|q| = 1$) eine Drehung des Imaginärteils - interpretiert als Vektor des \mathbb{R}^3 – bewirken.

¹²Analog zur Betrachtung der komplexen Zahlen \mathbb{C} als isomorph zum \mathbb{R}^2 können die Quaternionen \mathbb{H} als isomorph zum \mathbb{R}^4 gesehen werden.

Satz 6.3 (Drehung durch Quaternionen). Stellt man zwei Vektoren $\mathbf{v}, \mathbf{w} \in \mathbb{R}^3$ mit $\mathbf{v} = [v_x, v_y, v_z]^T$ und $\mathbf{w} = [w_x, w_y, w_z]^T$ als rein imaginäre Quaternion $q_v = iv_x + jv_y + kv_z$ bzw. $q_w = iw_x + jw_y + kw_z$ dar, so existiert eine Quaternion $\rho \in \mathbb{H}$ mit $|\rho| = 1$, dass

$$q_w = \rho q_v \bar{\rho} \quad (6.36)$$

eine Drehung von \mathbf{v} nach \mathbf{w} bewirkt.

Die Bedeutung von ρ erschließt sich dabei folgendermaßen: Ein mathematisch positive Drehung um den Winkel α mit der Rotationsachse in Richtung des Einheitsvektors \mathbf{r} wird durch die Quaternion

$$\rho = \cos\left(\frac{\alpha}{2}\right) + \sin\left(\frac{\alpha}{2}\right)(ir_x + jr_y + kr_z) \quad (6.37)$$

erreicht.

Aus einer gegebenen Einheitsquaternion $\rho = q_1 + iq_2 + jq_3 + kq_4$ ergibt sich die korrespondierende Rotationsmatrix zu

$$\mathbf{R} = \begin{bmatrix} 1 - 2(q_3^2 + q_4^2) & 2(q_2q_3 - q_1q_4) & 2(q_2q_4 + q_1q_3) \\ 2(q_2q_3 + q_1q_4) & 1 - 2(q_4^2 + q_2^2) & 2(q_3q_4 - q_1q_2) \\ 2(q_2q_4 - q_1q_3) & 2(q_3q_4 + q_1q_2) & 1 - 2(q_2^2 + q_3^2) \end{bmatrix}. \quad (6.38)$$

Umgekehrt folgt für eine gegebene Matrix $\mathbf{R} = (r_{mn})$ mit $m, n \in \{1, 2, 3\}$ die Einheitsquaternion q mit den Komponenten

$$q_1 = \frac{\sqrt{r_{11} + r_{22} + r_{33} + 1}}{2} \quad (6.39)$$

$$q_2 = \text{sign}(r_{32} - r_{23}) \frac{\sqrt{r_{11} - r_{22} - r_{33} + 1}}{2} \quad (6.40)$$

$$q_3 = \text{sign}(r_{13} - r_{31}) \frac{\sqrt{-r_{11} + r_{22} - r_{33} + 1}}{2} \quad (6.41)$$

$$q_4 = \text{sign}(r_{21} - r_{12}) \frac{\sqrt{-r_{11} - r_{22} - r_{33} + 1}}{2}. \quad (6.42)$$

Durch die quadratische Form in (6.36) führen zwei Einheitsquaternionen ρ_1, ρ_2 mit $\rho_1 = -\rho_2$ stets dieselbe Drehung aus. Wie man also sieht, ist die Zuordnung von Drehungen der $\text{SO}(3)$ zu Einheitsquaternionen ρ nicht eindeutig - jede Drehung kann durch *genau zwei* Einheitsquaternionen dargestellt werden.

Achse/Winkel-Parametrierung

Bisher wurde immer um die Achsen eines Koordinatensystems gedreht und eine Rotationsmatrix so aus elementaren Rotationsmatrizen zusammengesetzt. Ein anderer Zugang nutzt folgende Erkenntnis: Eine allgemeine Drehung der SO(3) lässt sich immer durch eine Drehung um eine allgemeine Drehachse mit einem Drehwinkel θ darstellen.

Satz 6.4 (Achse/Winkel-Parametrierung). Eine allgemeine Rotationsmatrix $\mathbf{R}_0^1 \in \text{SO}(3)$ lässt sich durch eine Drehachse entlang des Vektors $\mathbf{k} \in \mathbb{R}^3$ mit $\|\mathbf{k}\| = 1$ und den Drehwinkel θ als^a

$$\mathbf{R}_0^1 = \mathbf{I}_3 + \sin(\theta)[\mathbf{k}]_{\times} + (1 - \cos(\theta))(\mathbf{k}^T \mathbf{k} - \mathbf{I}_3) \quad (6.43)$$

darstellen, wobei \mathbf{I}_3 die 3×3 -Einheitsmatrix und $[\mathbf{k}]_{\times}$ den Kreuzproduktoperator $[\cdot]_{\times} : \mathbb{R}^3 \rightarrow \mathbb{R}^{3 \times 3}$ durch

$$[\mathbf{k}]_{\times} = \begin{bmatrix} 0 & -k_3 & k_2 \\ k_3 & 0 & -k_1 \\ -k_2 & k_1 & 0 \end{bmatrix} \quad (6.44)$$

bezeichnet. Die Namensgebung folgt aus dem Umstand, dass sich für alle $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$ das Kreuzprodukt als $\mathbf{x} \times \mathbf{y} = [\mathbf{x}]_{\times} \mathbf{y}$ darstellen lässt.

^aDieser Zusammenhang folgt aus der sog. Rodrigues-Formel.

Alternativ kann auch ein allgemeiner Vektor \mathbf{x} als Rotationsachse angegeben werden, dessen Betrag den Drehwinkel angibt, also $\|\mathbf{x}\| = \theta$ und $\mathbf{k} = \frac{\mathbf{x}}{\|\mathbf{x}\|}$. Diese Darstellung ist besonders für Geschwindigkeiten verbreitet, deren Winkelgeschwindigkeitsvektors ebenfalls in Richtung der Drehachse zeigt. Wir werden dies später noch sehen.

6.2.5. Homogene Transformationen

Da wir bereits wissen, dass eine allgemeine Starrkörperbewegung aus Rotation und Translation zusammengesetzt werden kann, stehen nun alle Werkzeuge zur Verfügung. Angenommen zwei Koordinatensysteme $\Sigma_0 = (0_0 x_0 y_0 z_0)$ und $\Sigma_1 = (0_1 x_1 y_1 z_1)$ sind durch die Rotationsmatrix \mathbf{R}_0^1 und den Verschiebungsvektor \mathbf{d}_0^1 miteinander verbunden. Zur Transformation wird nun zuerst das System Σ_1 gedreht, sodass seine Achsen parallel zu Σ_0 liegen, und anschließend in das System Σ_0 zurückverschoben.

Satz 6.5. Die Ortsvektoren \mathbf{p}_0 und \mathbf{p}_1 eines Punktes P in den Systemen $\Sigma_0 = (0_0x_0y_0z_0)$ und $\Sigma_1 = (0_1x_1y_1z_1)$ transformieren sich nach

$$\mathbf{p}_0 = \mathbf{d}_0^1 + \mathbf{R}_0^1 \mathbf{p}_1, \quad (6.45)$$

wobei \mathbf{d}_0^1 den Verschiebungsvektor zwischen den Nullpunkten dargestellt in Σ_0 und \mathbf{R}_0^1 die Rotationsmatrix zwischen den beiden Systemen bezeichnen.

Betrachtet man ein weiteres System $\Sigma_2 = (0_2x_2y_2z_2)$ mit dem Verschiebungsvektor \mathbf{d}_1^2 und der Rotationsmatrix \mathbf{R}_1^2 relativ zum System Σ_1 , so folgt aus

$$\mathbf{p}_0 = \mathbf{d}_0^1 + \mathbf{R}_0^1 \mathbf{p}_1 \quad \text{und} \quad \mathbf{p}_1 = \mathbf{d}_1^2 + \mathbf{R}_1^2 \mathbf{p}_2 \quad (6.46)$$

der Zusammenhang von Σ_0 und Σ_2 zu

$$\mathbf{p}_0 = \mathbf{d}_0^1 + \mathbf{R}_0^1 \mathbf{d}_1^2 + \mathbf{R}_0^1 \mathbf{R}_1^2 \mathbf{p}_2 = \mathbf{d}_0^1 + \mathbf{R}_0^1 \mathbf{d}_1^2 + \mathbf{R}_0^1 \mathbf{d}_1^2 + \mathbf{R}_0^1 \mathbf{R}_1^2 \mathbf{p}_2. \quad (6.47)$$

Mit der Anzahl an verwendeten Koordinatensystemen wird dieses Kalkül zunehmend unhandlich. Dies folgt daraus, das (6.45) eine affine und keine lineare Abbildung darstellt. Durch die Verwendung von homogenen Koordinaten lässt sich dieses Problem umgehen.

Satz 6.6 (Homogene Transformation). Eine Transformation zwischen Koordinatensystemen gemäß (6.45), lässt sich auch als

$$\underbrace{\begin{bmatrix} \mathbf{p}_0 \\ 1 \end{bmatrix}}_{\mathbf{P}_0} = \underbrace{\begin{bmatrix} \mathbf{R}_0^1 & \mathbf{d}_0^1 \\ \mathbf{0}^T & 1 \end{bmatrix}}_{\mathbf{T}_0^1} \underbrace{\begin{bmatrix} \mathbf{p}_1 \\ 1 \end{bmatrix}}_{\mathbf{P}_1} \quad (6.48)$$

schreiben, wobei \mathbf{T}_0^1 als homogene Transformationsmatrix und $\mathbf{P}_0, \mathbf{P}_1$ als homogene Koordinatenvektoren des Punktes P bezeichnet werden.

Ähnlich wie bei den Rotationsmatrizen gilt auch hier, dass die Menge aller homogenen Transformationsmatrizen der Bauform (6.48) mit der Rotationsmatrix $\mathbf{R}_0^1 \in \text{SO}(3)$ und den Translationsvektoren $\mathbf{d} \in \mathbb{R}^3$ eine Gruppe darstellen - die Bewegungsgruppe des dreidimensionalen Raumes¹³ SE(3).

¹³Wie man bereits aus der homogenen Transformationsmatrix erkennt, besitzt die SE(3) sechs Freiheitsgrade, repräsentiert durch die drei Rotationsparameter der Drehmatrix und den drei Komponenten des Verschiebungsvektors.

Der Zusammenhang von Σ_0 und Σ_2 aus (6.47) lässt sich wegen

$$\begin{bmatrix} \mathbf{p}_0 \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_0^1 \mathbf{R}_1^2 & \mathbf{d}_0^1 + \mathbf{R}_0^1 \mathbf{d}_1^2 \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p}_2 \\ 1 \end{bmatrix} \quad (6.49)$$

als

$$\mathbf{P}_0 = \mathbf{T}_0^1 \mathbf{T}_1^2 \mathbf{P}_2 = \mathbf{T}_0^2 \mathbf{P}_2 \quad (6.50)$$

schreiben.

Durch Umformen von (6.48) lässt sich einfach zeigen, dass für die inverse Transformation gilt

$$(\mathbf{T}_0^1)^{-1} = \mathbf{T}_1^0 = \begin{bmatrix} (\mathbf{R}_0^1)^T & -(\mathbf{R}_0^1)^T \mathbf{d}_0^1 \\ \mathbf{0}^T & 1 \end{bmatrix}, \quad (6.51)$$

womit auch gezeigt ist, dass homogene Transformationsmatrizen im Gegensatz zu Rotationsmatrizen im Allgemeinen (d. h. für $\mathbf{d}_0^1 \neq 0$) *keine* orthogonalen Matrizen darstellen.

Wie schon bei den Rotationsmatrizen gibt es eine Reihe von elementaren homogenen Transformationsmatrizen, nämlich die elementaren Rotationen und Translationen

$$\text{Rot}(i, \gamma) = \begin{bmatrix} \mathbf{R}_{i,\gamma} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \quad \text{Trans}(x, y, z) = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (6.52)$$

wobei $i \in \{x, y, z\}$.

Die Bezeichnungen homogene Transformation und homogene Koordinaten kommen aus der projektiven Geometrie, mit welcher beispielsweise die perspektivische Betrachtung eines Raumes durch einen Beobachter dargestellt werden kann. Allgemeine homogene Transformationsmatrizen besitzen dann die Form

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{d} \\ \mathbf{f}^T & s \end{bmatrix} \quad (6.53)$$

mit der Rotationsmatrix $\mathbf{R} \in \text{SO}(3)$, dem Verschiebungsvektor $\mathbf{d} \in \mathbb{R}^3$, dem Perspektivenvektor $\mathbf{f} \in \mathbb{R}^3$ und der Skalierung s . Solche Transformationen werden beispielsweise in der Computergrafik verwendet. Die oben verwendeten homogenen Transformationsmatrizen stellen also einen Spezialfall dar, in welchem die letzte Zeile von \mathbf{T} die Form $[0, 0, 0, 1]$ besitzt.

6.2.6. Kinematische Ketten

Bereits in Abschnitt 6.1.1 wurden kinematische Ketten als Kombination von Starrkörpern und Gelenken beschrieben. Wird den einzelnen Starrkörpern ein körperfestes Koordinatensystem zugeordnet, kann deren Bewegung - wie wir gerade gesehen haben - durch homogene Transformationen der Bauform (6.48) dargestellt werden.

Im Folgenden werden wir drei häufige Vereinfachungen treffen. Erstens werden ausschließlich Gelenke mit einfacherem Freiheitsgrad betrachtet, also einfache Dreh- oder Schubgelenke. Dies ist keine große Einschränkung der Allgemeinheit, da komplexere Gelenke durch deren Kombinationen dargestellt werden können. Zweitens werden ausschließlich serielle Kinematiken betrachtet. Die Handhabung paralleler Kinematiken ist um Größenordnungen schwieriger und muss für jede Klasse separat diskutiert werden, weshalb hier darauf verzichtet wird. Drittens werden lediglich Roboter mit einem Endeffektor behandelt. Durch diese Vereinfachungen stellt sich ein Roboter folgendermaßen dar:

Definition 6.1 (Serieller Roboter). Ein serieller Roboter \mathcal{R}_n besteht aus einer kinematischen Kette mit $(n + 1)$ den Starrkörperteilen des Roboters zugeordneten Koordinatensystemen $\Sigma_0, \dots, \Sigma_n$ und n Gelenken G_1, \dots, G_n . Ein Gelenk G_i verbindet die Systeme Σ_{i-1} und Σ_i und kann durch einen einzigen Freiheitsgrad q_i als $\mathbf{T}_{i-1}^i(q_i)$ beschrieben werden. Die Systeme Σ_0 und Σ_n werden dabei als *Roboterbasis-* und *Endeffektorsysteme* bezeichnet, weshalb auch die Benennung $\Sigma_n = \Sigma_E$ häufig auftritt.

Wendet man die bisherigen Erkenntnisse zu homogenen Transformationen auf serielle Roboter nach Def. 6.1 an, erhält man folgendes Resultat:

Satz 6.7 (Serieller Roboter). Durch einen verallgemeinerten Gelenkwinkelvektor $\mathbf{q} = [q_1, \dots, q_n]^T$ wird die Pose des Roboters \mathcal{R}_n eindeutig dargestellt; er besitzt also n Freiheitsgrade. Die Pose eines mit dem i -ten Starrkörperteil verbundenen Koordinatensystems Σ_i , $i \in 1, \dots, n$ relativ zum Roboterbasissystem Σ_0 errechnet sich zu

$$\mathbf{T}_0^i(q_1, \dots, q_i) = \mathbf{T}_0^1(q_1) \dots \mathbf{T}_{i-1}^i(q_i). \quad (6.54)$$

Für das Endeffektorsystem gilt entsprechend

$$\mathbf{T}_0^E(\mathbf{q}) = \mathbf{T}_0^n(\mathbf{q}) = \mathbf{T}_0^1(q_1) \dots \mathbf{T}_{n-1}^n(q_n). \quad (6.55)$$

Unter den oben genannten Einschränkungen bestehen kinematische Ketten aus einer Aneinanderreihung einfacher Gelenke, weshalb ihre Struktur durch die Abfolge der Gelenks-

typen - beginnend mit dem Basissystem - festgelegt ist. Ausgehend von den englischen Bezeichnungen werden Drehgelenke (*revolute joints*) mit R und Schubgelenke (*prismatic joints*) mit P bezeichnet. Ein SCARA-Roboter wie in Abb. 6.6 besitzt dementsprechend die kinematische Kette RRRP.

Im Allgemeinen ist der Wertebereich der Gelenkskoordinate q_i durch physikalische Begrenzungen des Gelenks G_i eingeschränkt. Zudem kann es passieren, dass eine Pose des Roboters \mathcal{R}_n seine Selbstdurchdringung fordern würde, was durch eine Beschränkung des Wertebereichs verhindert werden kann.

Definition 6.2 (Konfigurationsraum). Die Menge aller möglichen Gelenkskoordinaten \mathbf{q} eines Roboters \mathcal{R}_n wird als *Konfigurationsraum* $\mathcal{U} \subset \mathbb{R}^n$ bezeichnet.

Die Pose des Endeffektors ist durch die zugehörige homogene Transformationsmatrix \mathbf{T}_0^E bestimmt. Oftmals möchte man jedoch explizit die kartesischen Koordinaten des Endeffektors und seine Orientierungswinkel (d. h. Euler- oder RPY-Winkel) angeben.

Definition 6.3 (Arbeitsraum). Die Menge aller $\mathbf{T}_0^E(\mathbf{q})$ mit $\mathbf{q} \in \mathcal{U}$ heißt Arbeitsraum \mathcal{W} und stellt eine Teilmenge^a der euklidischen Gruppe dar, also $\mathcal{W} \subset \text{SE}(3)$. Aufgrund von Satz 6.6 und den Parametrierungen (6.17), (6.23) bzw. (6.43) kann \mathcal{W} auch als Teilmenge des \mathbb{R}^6 durch den Vektor $\mathbf{w} = [\mathbf{d}^T, \boldsymbol{\Omega}^T]^T$ dargestellt werden, wobei $\boldsymbol{\Omega}$ je nach gewählter Parametrierung aus drei Winkel oder dem Vektor der Rotationsachse besteht.

^aDer Arbeitsraum ist *keine* Untergruppe des $\text{SE}(3)$, da die Kombination zweier solcher Elemente den Arbeitsraum verlassen kann.

Anstelle des Endeffektorsystems werden häufig zwei verwandte Begriffsbildungen verwendet. Einerseits das *Tool-Attachment-Point*-System Σ_{TAP} , welches den Montageflansch der austauschbaren Werkzeugtypen bezeichnet. Der Ursprung des *Tool-Center-Point*-Systems Σ_{TCP} hingegen stellt einen gedachten Referenzpunkt des gewählten Werkzeugs dar. Beispielsweise kann damit die Spitze einer Schweißelektrode oder einen in Arbeitsdistanz einer Sprühpistole vorgelagerter Punkt bezeichnet werden, welche durch den Roboter entlang einer vorgegebenen Bahn bewegt werden soll. Weiters häufig vorkommende Systeme sind ein Weltsystem Σ_W und ein Arbeitssystem Σ_S zur Referenzierung externer Gegenstände sowie ein Objektsystem Σ_O , um bewegliche Objekte zu beschreiben.

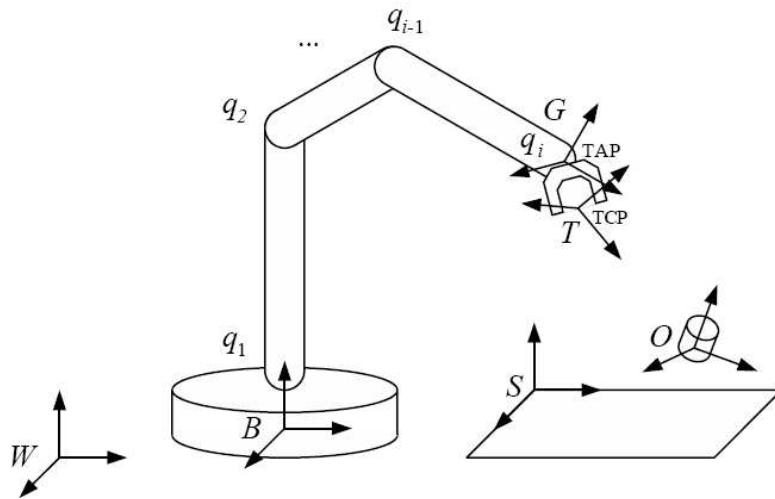


Abbildung 6.15.: Definition unterschiedlicher Koordinatensysteme.

6.2.7. Denavit-Hartenberg – Konvention

Formal ist der Roboter \mathcal{R}_n durch die obige Definition als kinematische Kette vollständig bestimmt, es ist jedoch unklar, wie die Koordinatensysteme und daraus resultierend die homogenen Transformationen einfach festgelegt bzw. bestimmt werden können. Um bei der Wahl der Koordinatensysteme $\Sigma_0, \dots, \Sigma_n$ und den Transformationsmatrizen $\mathbf{T}_0^1, \dots, \mathbf{T}_{n-1}^n$ möglichst systematisch vorzugehen, wird meist ein 1955 von JACQUES DENAVIT und RICHARD HARTENBERG eingeführter Satz von Konventionen verwendet (im Weiteren abgekürzt DH-Konvention). Damit lassen sich rotatorische oder lineare Gelenksverbindungen mit beliebigen Starrkörperverbindungen durch vier sog. DH-Parameter (θ, d, a, α) angeben. Die grundlegende Idee hinter der DH-Konvention bezieht sich auf die sog. *gemeinsame Normale* zweier Geraden, welche beide auf kürzestem Abstand miteinander verbindet.

Definition 6.4 (DH-Koordinatensysteme). Angenommen das System Σ_{i-1} ist bereits nach DH-Konvention festgelegt, dann erhält man das nächste konsistent orientierte System Σ_i durch folgende Schritte:

- Zuerst wird die z_i -Achse entlang der Bewegungsachse des Gelenks G_{i+1} ausgerichtet. Für Drehgelenke ist dies die Drehachse, für lineare Gelenke die Richtung der linearen Bewegung.
- Besitzen die beiden Koordinatenachsen entlang $\mathbf{e}_{z_{i-1}}$ und \mathbf{e}_{z_i} eine eindeutige

gemeinsame Normale, wird der Ursprung 0_i in den Schnittpunkt der gemeinsamen Normalen mit der z_i -Achse gelegt. Sollten die beiden Koordinatenachsen parallel zueinander sein, existiert keine eindeutige gemeinsame Normale und der Ursprung 0_i kann beliebig entlang der z_i -Achse festgelegt werden.

- Anschließend wird \mathbf{e}_{x_i} entlang der gemeinsamen Normalen, in Richtung des neuen Gelenks zeigend, ausgerichtet. Sollten die beiden Nullpunkte zusammenfallen, orientiert man \mathbf{e}_{x_i} normal auf die von $\mathbf{e}_{z_i}, \mathbf{e}_{z_{i-1}}$ aufgespannte Ebene.
- Der fehlende Basisvektor \mathbf{e}_{y_i} wird dem beiden bestimmten Richtungen rechtswendig zugeordnet, also $\mathbf{e}_{y_i} = \mathbf{e}_{z_i} \times \mathbf{e}_{x_i}$.

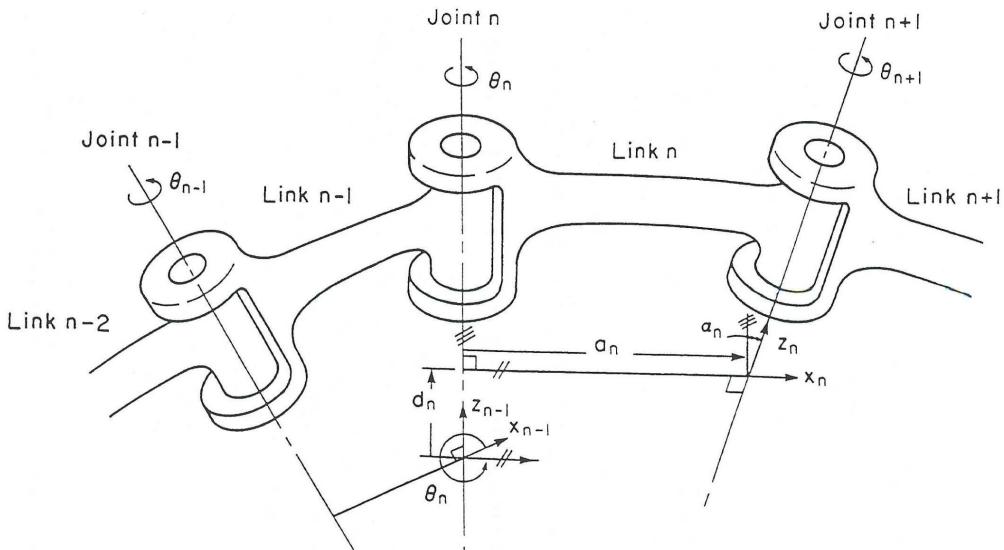


Abbildung 6.16.: Beispielhafte Anwendung der Denavit-Hartenberg-Konventionen.

Def. 6.4 stellt eine Iterationsvorschrift dar, welche von einem bereits korrekt orientierten System ausgehend ein weiteres erzeugt.

Die Wahl des Basissystems Σ_0 wird dadurch nicht abgedeckt. In der Tat ist diese Wahl bis auf eine Ausnahme frei: die Koordinatenachse z_0 muss mit der Bewegungsachse von G_1 zusammenfallen. Die Lage des Ursprungs 0_0 und die Orientierung von \mathbf{e}_{x_0} und \mathbf{e}_{y_0} ist frei. Eine ähnliche Situation ergibt sich beim Endeffektorsystem Σ_n . Hier ist kein weiteres Gelenk G_{n+1} vorhanden, weshalb die \mathbf{e}_{z_n} Richtung frei gewählt werden kann.

Bemerkung 6.1. Es hat sich dabei die Konvention etabliert, die z-Achse des Endeffektorsystems in Richtung des fortschreitenden Endeffektors und die y-Achse in

Richtung der sog. Gleitebene zu orientieren. Darunter fällt beispielsweise die Öffnungsrichtung von Greifern. Betrachtet man an einer allgemeinen Konfiguration \mathbf{q}^* die Transformation des Endeffektorsystems

$$\mathbf{P}_0 = \mathbf{T}_0^E(\mathbf{q}^*) \mathbf{P}_E = \begin{bmatrix} \mathbf{n}(\mathbf{q}^*) & \mathbf{s}(\mathbf{q}^*) & \mathbf{a}(\mathbf{q}^*) & \mathbf{d}(\mathbf{q}^*) \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{P}_E, \quad (6.56)$$

erkennt man die Vektoren \mathbf{n} , \mathbf{s} und \mathbf{a} als die lokale Basis des Endeffektorsystems Σ_E am Punkt \mathbf{q}^* dargestellt im Basissystem Σ_0 . Daher stammen die engl. Bezeichnungen *approach vector* \mathbf{a} , *slide/slip vector* \mathbf{s} und *normal vector* \mathbf{n} .

Mit diesen Festlegungen können die durch das Gelenk G_i verbundenen Koordinatensysteme Σ_{i-1}, Σ_i durch die DH-Parameter $(\theta_i, d_i, a_i, \alpha_i)$ immer auf folgende Weise ineinander übergeführt werden:

- Zuerst wird Σ_{i-1} mit θ_i um die z_{i-1} -Achse gedreht. Somit sind die beiden x -Achsen parallel.
- Dann verschiebt man den Ursprung 0_{i-1} um den Abstand d_i entlang der z_{i-1} -Achse, womit die beiden Ursprünge auf der gemeinsamen Normalen liegen.
- Durch Verschiebung von 0_{i-1} um a_i entlang der x_i -Achse decken sich die beiden Ursprünge.
- Letztlich werden durch eine Drehung von α_i um die x_i -Achse die beiden Koordinatensysteme in Deckung gebracht.

Man erhält daher die Transformationsmatrix

$$\mathbf{T}_{i-1}^i(q_i) = \text{Rot}(z, \theta_i) \text{Trans}(0, 0, d_i) \text{Trans}(a_i, 0, 0) \text{Rot}(x, \alpha_i) \quad (6.57)$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (6.58)$$

$$(6.59)$$

Satz 6.8 (DH-Parameter). Die homogene Transformation zwischen zwei durch das Gelenk G_i verbundenen und nach Def. 6.4 orientierten Koordinatensystemen Σ_{i-1}, Σ_i eines Roboters \mathcal{R}_n ist durch die Denavit-Hartenberg-Parameter $(\theta_i, d_i, a_i, \alpha_i)$ eindeutig bestimmt.

tig in der Form

$$\mathbf{T}_{i-1}^i(q_i) = \text{Rot}(z, \theta_i) \text{ Trans}(0, 0, d_i) \text{ Trans}(a_i, 0, 0) \text{ Rot}(x, \alpha_i) \quad (6.60)$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.61)$$

festgelegt. Abhängig von der Art des Gelenks G_i gilt

$$q_i = \begin{cases} d_i & \text{für Schubgelenke,} \\ \theta_i & \text{für Drehgelenke.} \end{cases} \quad (6.62)$$

Angesichts des Umstandes, dass die euklidische Gruppe SE(3) sechs Freiheitsgrade besitzt, scheint es auf den ersten Blick seltsam, diese mit nur vier Parametern beschreiben zu können. Es lässt sich jedoch zeigen, dass dies durch die beiden Zusatzbedingungen zur Orientierung der Koordinatensysteme eindeutig möglich ist. Die einfache und komplette Definition der DH-Parametrisierung wird durch zwei Nachteile erkauft: einerseits sind einzelne Koordinatensysteme nicht mehr frei wählbar, was manchmal unangenehm ist; andererseits variieren bei quasi-parallelen Achsen die DH-Parameter stark auf kleine Veränderungen, was zu Problemen bei der Kalibrierung führen kann.

Um die Funktionsweise der DH-Konvention zu verstehen ist es besonders hilfreich, Beispiele im Detail nachzuvollziehen. Es ist jedoch zu beachten, dass aufgrund der Wahlfreiheit des Basissystems und eventueller paralleler Koordinatenachsen (siehe Def. 6.4) die DH-Konventionen i. A. keine *eindeutige* Lösung erfordern.

Beispiel 6.1 (Stanford-Manipulator). Ein einfaches Beispiel eines Roboters mit sechs Freiheitsgraden stellt der sog. Stanford-Manipulator mit der kinematischen Kette RRPRRR und dem verallgemeinerten Gelenksvektor $\mathbf{q} = [\theta_1, \theta_2, d_3, \theta_4, \theta_5, \theta_6]^T$ dar. Abb. 6.17 zeigt eine Darstellung mit konsistent nach DH-Konvention orientierten Koordinatensystemen Σ_0 bis Σ_6 . Zur besseren Übersicht wurden die eigentlich zusammenfallenden Ursprünge $0_3, 0_4, 0_5$ nebeneinander gezeichnet.

Beispiel 6.2. Der sog. PUMA-Roboter besitzt sechs Freiheitsgrade und ist ausschließlich aus Drehgelenken zusammengesetzt - d. h. er besitzt die kinematische Kette RRRRRR. Der verallgemeinerte Gelenkswinkelvektor hat also die Bauform $\mathbf{q} = [\theta_1, \dots, \theta_6]^T$. Abb. 6.18 zeigt eine Darstellung eines PUMA-Roboters mit konsistent nach DH-Konvention orientierten Koordinatensystemen Σ_0 bis Σ_6 . Die Wahl

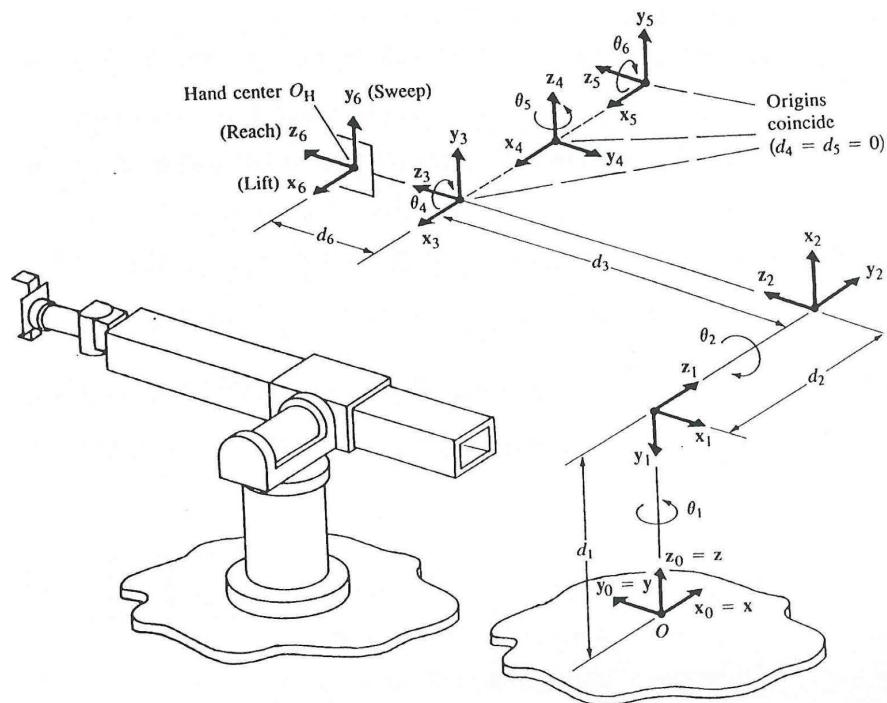


Abbildung 6.17.: Der Stanford-Manipulator mit der kinematischen Kette RRPURR.

des Basissystems Σ_0 mag in diesem Fall etwas überraschen, ist jedoch problemlos möglich. Wie schon in Bsp. 6.1 fallen mehrere Ursprünge (0_0 und 0_1 , 0_2 und 0_3 , 0_4 und 0_5) zusammen.

In Tab. 6.2 sind die DH-Parameter des Roboters angegeben. Die erste Spalte ist dabei gleichzeitig der zur in Abb. 6.18 dargestellten Pose gehörende Gelenkwinkelvektor. Die homogenen Transformationsmatrizen der kinematischen Kette lauten daher

$$\begin{aligned} \mathbf{T}_0^1(q_1) &= \begin{bmatrix} c_{q_1} & 0 & -s_{q_1} & 0 \\ s_{q_1} & 0 & c_{q_1} & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & \mathbf{T}_1^2(q_2) &= \begin{bmatrix} c_{q_2} & -s_{q_2} & 0 & a_2 c_{q_2} \\ s_{q_2} & c_{q_2} & 0 & a_2 s_{q_2} \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\ \mathbf{T}_2^3(q_3) &= \begin{bmatrix} c_{q_3} & 0 & s_{q_3} & a_3 c_{q_3} \\ s_{q_3} & 0 & -c_{q_3} & a_3 s_{q_3} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & \mathbf{T}_3^4(q_4) &= \begin{bmatrix} c_{q_4} & 0 & -s_{q_4} & 0 \\ s_{q_4} & 0 & c_{q_4} & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (6.63) \\ \mathbf{T}_4^5(q_5) &= \begin{bmatrix} c_{q_5} & 0 & s_{q_5} & 0 \\ s_{q_5} & 0 & -c_{q_5} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & \mathbf{T}_5^6(q_6) &= \begin{bmatrix} c_{q_6} & -s_{q_6} & 0 & 0 \\ s_{q_6} & c_{q_6} & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \end{aligned}$$

(6.64)

Damit folgt für die Transformation zwischen Basissystem und Endeffektor

$$\mathbf{T}_0^6(\mathbf{q}) = \mathbf{T}_0^E(\mathbf{q}) = \mathbf{T}_0^1(q_1) \mathbf{T}_1^2(q_2) \mathbf{T}_2^3(q_3) \mathbf{T}_3^4(q_4) \mathbf{T}_4^5(q_5) \mathbf{T}_5^6(q_6). \quad (6.65)$$

6.2.8. Direktes und inverses kinematisches Problem

Nach dieser Vorarbeit lassen sich zwei zentrale Aufgaben der Kinematik formulieren: das *direkte* und das *indirekte* kinematische Problem¹⁴.

Ein gegebener Roboter \mathcal{R}_n wird einerseits durch seine Gelenkskoordinaten \mathbf{q} im Konfigurationsraum \mathcal{U} festgelegt, welche andererseits mit einer Pose des Endeffektors im dreidimensionalen Raum $\mathbf{T}_0^E \in \text{SE}(3)$ verknüpft ist.

¹⁴Die beiden Probleme werden manchmal auch als Vorwärts- und Rückwärtseinematik bezeichnet.

Gelenk G_i	θ_i	d_i	a_i	α_i
1	90	0,00	0,00	-90
2	0	149,09	431,80	0
3	90	0,00	-20,32	90
4	0	433,07	0,00	-90
5	0	0,00	0,00	90
6	0	56,25	0,00	0

Tabelle 6.2.: Der Satz von DH-Parametern für den PUMA-Roboter in Abb. 6.18.

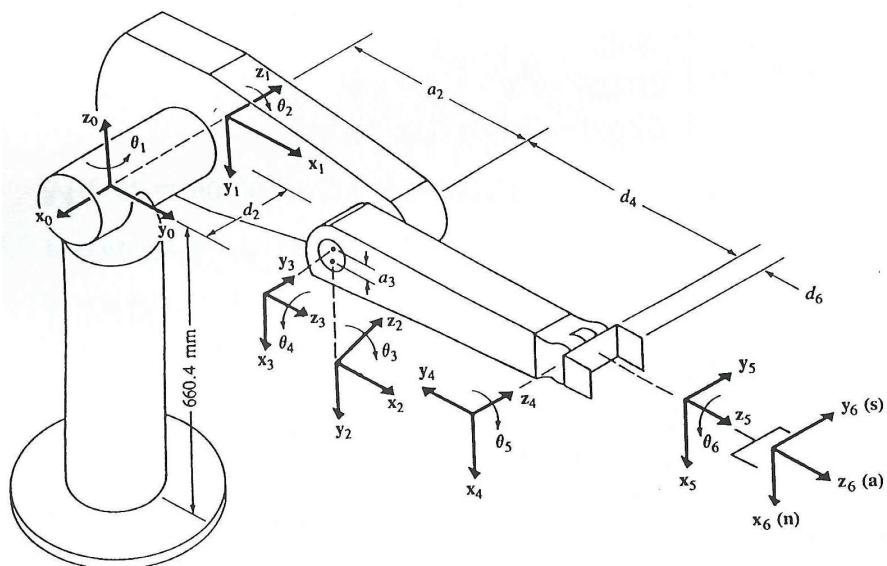


Abbildung 6.18.: Der PUMA-Roboter mit der kinematischen Kette RRRRRR.

Das direkte kinematische Problem stellt sich als:

Definition 6.5 (Direktes kinematisches Problem). Für einen gegebenen verallgemeinerten Gelenkwinkelvektor \mathbf{q} aus dem Konfigurationsraum \mathcal{U} eines seriellen Roboters \mathcal{R}_n ist die zugehörige Endeffektorpose $\mathbf{T}_0^E \in \mathcal{W}$ gesucht.

Über die letzten Abschnitte hinweg wurde im Wesentlichen auf die Lösung dieses Problems hingearbeitet. Das Resultat (6.55) aus Satz 6.7 nahm die Lösung des direkten Problems eigentlich schon vorweg. Es steht damit also sogar eine analytische Lösung für alle $\mathbf{q} \in \mathcal{U}$ zur Verfügung - die Lösung des direkten Problems erfolgt durch Einsetzen.

Wesentlich komplexer zu lösen ist das inverse Problem:

Definition 6.6 (Inverses kinematisches Problem). Zu einer gegebenen Endeffektorpose $\mathbf{T} \in \mathcal{W}$ sind alle möglichen verallgemeinerten Gelenkwinkelvektoren $\bar{\mathbf{q}}$ gesucht, sodass $\mathbf{T}_0^E(\bar{\mathbf{q}}) = \mathbf{T}$.

Bemerkung 6.2. Anstatt einer Pose $\mathbf{T} \in \text{SE}(3)$ wird häufig, wie in Def. 6.3 erwähnt, der 6-dimensionale Vektor $\mathbf{w} = [\mathbf{d}^T, \boldsymbol{\Omega}^T]^T$ angegeben, welcher durch Angabe der verwendeten Parametrierung der $\text{SO}(3)$ mit genau einer Pose korrespondiert. Wir schreiben dafür im Folgenden auch $\mathbf{w} = \mathbf{f}(\mathbf{q})$. Das inverse Problem kann daher auch als Suche einer inversen Funktion $\mathbf{g}(\mathbf{w}) = \mathbf{f}^{-1}(\mathbf{w}) = \mathbf{q}$ verstanden werden. Diese Schreibweise verschleiert jedoch, dass dies i. A. nur lokal und nicht-eindeutig möglich ist (vgl. *Satz von der impliziten Funktion*).

Das inverse Problem stellt ein nichtlineares Gleichungssystem dar, da die Einträge der homogenen Transformationsmatrix \mathbf{T}_0^E i. A. aus trigonometrischen Funktionen zusammengesetzt sind. Dementsprechend kann keine allgemein gültige Lösungsmethode angegeben werden. Für spezielle Typen lassen sich in der Literatur Methoden oder Abschätzungen zur Anzahl der Lösungen finden.

Durch die Forderung, dass $\mathbf{T} \in \mathcal{W}$ liegen soll, muss mindestens eine Lösung existieren. Da in der Praxis oftmals der Arbeitsraum eines Roboters nicht genau bekannt ist, kann für eine allgemeine Pose $\mathbf{T} \in \text{SE}(3)$ auch keine Lösung existieren - die betreffende Pose ist dann im Umkehrschluss nicht Teil des Arbeitsraums.

Da die Gruppe $\text{SE}(3)$ durch sechs Parameter beschrieben wird, können wir i. A. erwarten, dass bei einem Roboter \mathcal{R}_n

- für $n \leq 6$ endlich viele Lösungen auftreten,

- für $n > 6$ unendlich viele Lösungen auftreten (*kinematische Redundanz*).

Abb. 6.19 zeigt eine grafische Darstellung der Zusammenhänge.

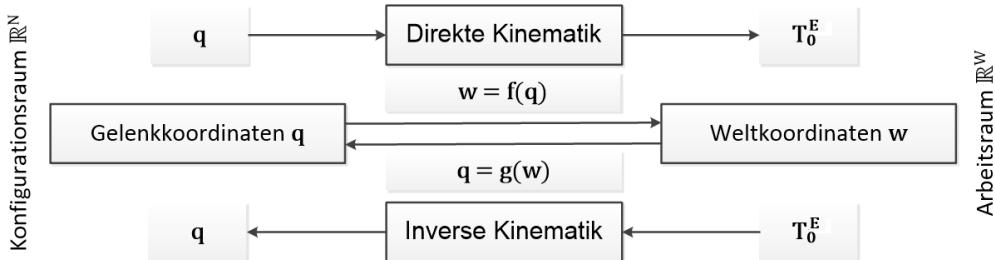


Abbildung 6.19.: Illustration von direkter und inverser Kinematik als Operationen zwischen Konfigurations- und Arbeitsraum.

Da analytische Lösungsmöglichkeiten des inversen Problems oft scheitern, werden vielfach numerische Lösungsmethoden wie das *Newton-Raphson-Verfahren* eingesetzt. Für einen Startpunkt $\mathbf{q}^* = \mathbf{q}_0$ ausreichend nahe an der Lösung $\bar{\mathbf{q}}$ konvergiert die Iteration

$$\mathbf{q}_{i+1} = \mathbf{q}_i - \left(\frac{\partial \mathbf{f}}{\partial \mathbf{q}}(\mathbf{q}_i) \right)^{-1} (\mathbf{f}(\mathbf{q}_i) - \mathbf{w}) \quad (6.66)$$

, sodass $\mathbf{q}_i \rightarrow \bar{\mathbf{q}}$ für $i \rightarrow \infty$.

Problematisch ist dabei, neben ihrer Existenz, die Berechnung der inversen Jacobimatrix $(\frac{\partial \mathbf{f}}{\partial \mathbf{q}}(\mathbf{q}_i))^{-1}$. Kann diese nicht analytisch invertiert werden, muss in jedem Iterationsschritt ein Gleichungssystem gelöst werden, was den Algorithmus sehr aufwendig macht. Zudem ist nicht klar, gegen welche Lösung das Verfahren im Falle von Mehrdeutigkeiten konvergiert, zumal eine Abschätzung der Einzugsbereiche schwierig ist.

Beispiel 6.3 (SCARA-Manipulator). Betrachtet man einen SCARA-Manipulator ähnlich Abb. 6.6 mit der kinematischen Kette RRRP und den Gelenkskoordinaten $\mathbf{q} \in \mathcal{U} \subset \mathbb{R}^4$. Durch Festlegen der Koordinatensysteme $\Sigma_0, \dots, \Sigma_4$ ergeben sich die Denavit-Hartenberg-Parameter

$$\begin{aligned} (\theta_1, d_1, a_1, \alpha_1) &= (q_1, 0, l_1, 0), & (\theta_2, d_2, a_2, \alpha_2) &= (q_2, 0, l_2, 0), \\ (\theta_3, d_3, a_3, \alpha_3) &= (q_3, 0, 0, 0), & (\theta_4, d_4, a_4, \alpha_4) &= (0, 0, -q_4, 0), \end{aligned}$$

womit gemäß Satz 6.8 die Transformationsmatrizen

$$\mathbf{T}_0^1(q_1) = \begin{bmatrix} \cos(q_1) & -\sin(q_1) & 0 & l_1 \cos(q_1) \\ \sin(q_1) & \cos(q_1) & 0 & l_1 \sin(q_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{T}_1^2(q_2) = \begin{bmatrix} \cos(q_2) & -\sin(q_2) & 0 & l_2 \cos(q_2) \\ \sin(q_2) & \cos(q_2) & 0 & l_2 \sin(q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{T}_2^3(q_3) = \begin{bmatrix} \cos(q_3) & -\sin(q_3) & 0 & 0 \\ \sin(q_3) & \cos(q_3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{T}_3^4(q_4) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -q_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

festgelegt sind. Da der Arbeitsraum des Manipulators nicht bekannt ist, soll vorerst eine allgemeine Pose $\mathbf{T} \in \text{SE}(3)$ angefahren werden, welche durch den Verschiebungsvektor $\mathbf{d} = [d_x, d_y, d_z]^T$ und die RPY-Winkel Φ, θ, ψ beschrieben wird. Es ergibt sich die gewünschte Endeffektorpose

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{d} \\ \mathbf{0}^T & 1 \end{bmatrix} \quad (6.67)$$

mit der Drehmatrix $\mathbf{R} = \mathbf{R}_{z,\Phi} \mathbf{R}_{y,\theta} \mathbf{R}_{x,\psi}$. Das inverse kinematische Problem bezeichnet also die Lösungen $\bar{\mathbf{q}}$ des 4×4 – dimensionalen nichtlinearen Gleichungssystems

$$\mathbf{T}_0^E(\bar{\mathbf{q}}) = \mathbf{T} \quad (6.68)$$

mit $\mathbf{T}_0^E(\bar{\mathbf{q}}) = \mathbf{T}_0^1 \mathbf{T}_1^2 \mathbf{T}_2^3 \mathbf{T}_3^4$. Aufgrund der Struktur von homogenen Matrizen sind die vier Gleichungen der letzten Zeile trivial erfüllt. Eine Lösung $\bar{\mathbf{q}}$ muss daher die

folgenden 12 Gleichungen erfüllen:

$$\begin{aligned}
 & \frac{1}{2} [c_{(\phi-\theta)} + c_{(\phi+\theta)}] = c_{(q_3+q_1+q_2)}, \\
 & \frac{1}{2} [s_{(\phi+\psi)} + s_{(\phi-\psi)}] - \frac{1}{4} [c_{(-\psi+\phi+\theta)} - c_{(\psi+\phi+\theta)} - c_{(-\psi+\phi-\theta)} + c_{(\psi+\phi-\theta)}] = s_{(q_3+q_1+q_2)}, \\
 & \frac{1}{2} [c_{(\phi-\psi)} - c_{(\phi+\psi)}] + \frac{1}{4} [s_{(\psi+\phi+\theta)} + s_{(-\psi+\phi+\theta)} - s_{(\psi+\phi-\theta)} - s_{(-\psi+\phi-\theta)}] = 0, \\
 & d_x = l_2 c_{(q_1+q_2)} + l_1 c_{q_1}, \\
 & \frac{1}{2} [s_{(\phi+\theta)} + s_{(\phi-\theta)}] = s_{(q_3+q_1+q_2)}, \\
 & \frac{1}{2} [c_{(\phi-\psi)} + c_{(\phi+\psi)}] + \frac{1}{4} [s_{(\psi+\phi-\theta)} - s_{(-\psi+\phi-\theta)} - s_{(\psi+\phi+\theta)} + s_{(-\psi+\phi+\theta)}] = c_{(q_3+q_1+q_2)}, \\
 & + \frac{1}{2} [s_{(\phi+\psi)} + s_{(\phi-\psi)}] - \frac{1}{4} [c_{(-\psi+\phi-\theta)} + c_{(\psi+\phi-\theta)} - c_{(-\psi+\phi+\theta)} - c_{(\psi+\phi+\theta)}] = 0, \\
 & d_y = l_2 s_{(q_1+q_2)} + l_1 s_{q_1}, \\
 & s_\theta = 0, \\
 & c_\theta s_\psi = 0, \\
 & c_\theta c_\psi = 1, \\
 & q_4 = d_z.
 \end{aligned}$$

Wie man schnell erkennt, sind einige Gleichungen lediglich von den RPY-Winkeln abhängig und schränken daher die möglichen Posen ein. Dies war auch zu erwarten, da der SCARA-Manipulator nur vier Freiheitsgrade besitzt. Wie man aus den letzten Gleichungen erkennt, muss

$$\theta = 0 \quad \text{und} \quad \psi = 0 \tag{6.69}$$

gelten. Es können also nur Drehungen in der xy-Ebene ausgeführt werden, der Arbeitsraum \mathcal{W} ist entsprechend beschränkt. Das Gleichungssystem vereinfacht sich dadurch dramatisch. In der Tat kann mittels Computer-Algebra-Programmen eine explizite Lösung angegeben werden. Für die Sollpose mit den Parameterwerten aus Tab. 6.3 ergeben sich die beiden Lösungen

$$\bar{\mathbf{q}}_a = [1.749, -1.939, 1.749, -0.5]^T \quad \text{und} \quad \bar{\mathbf{q}}_b = [-0.184, 1.939, -0.184, -0.5]^T$$

Die beiden Lösungsmöglichkeiten sind in Abb. 6.20 schematisch dargestellt.

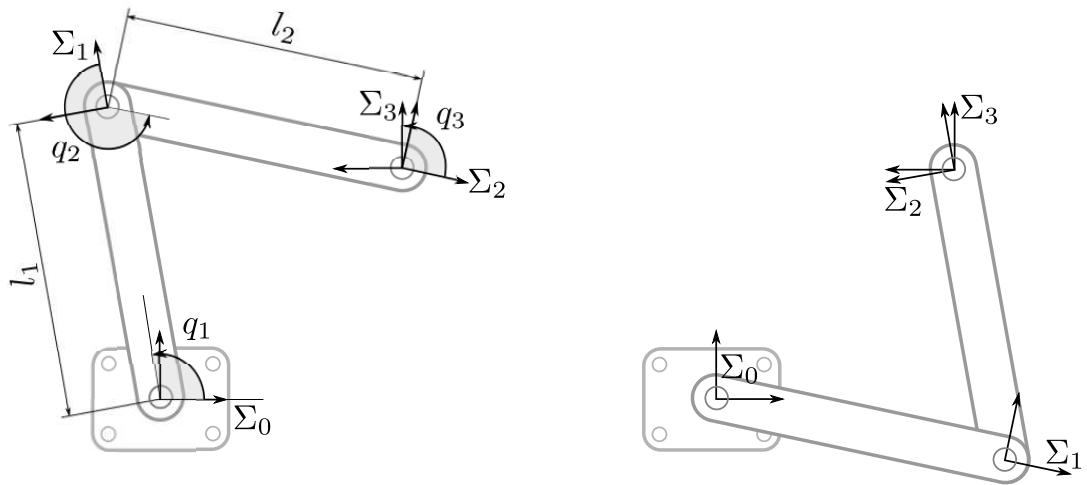


Abbildung 6.20.: Planare Darstellung der beiden Lösungen des SCARA-Manipulators.
Die linke Darstellung zeigt die Konfiguration \bar{q}_a , die rechte die Konfiguration \bar{q}_b .

Parameter	Wert	Parameter	Wert
Φ	$\frac{\pi}{2}$	d_x	0.8
θ	0	d_y	0.8
ψ	0	d_z	-0.5
l_1	1	l_2	1

Tabelle 6.3.: Parameterwerte des SCARA-Roboters.

6.2.9. Manipulator-Jacobi-Matrix

Neben den Positionen sind vielfach die Geschwindigkeiten im Arbeits- und Konfigurationsraum von Interesse. Insbesondere möchte man Geschwindigkeiten in beiden Räumen ineinander umrechnen können. Da sich Gelenkwinkelvektor $\mathbf{q}(t)$ über die Zeit verändert, sind auch die Rotationsmatrizen und Verschiebungsvektoren Funktionen der Zeit, was im Folgenden der Einfachheit wegen nicht explizit notiert wird.

Betrachtet man einen starr mit dem System Σ_i verbundenen Punkt P mit der homogenen Transformation

$$\mathbf{H}_0^i = \begin{bmatrix} \mathbf{R}_0^i & \mathbf{d}_0^i \\ \mathbf{0}^T & 1 \end{bmatrix} \quad \text{mit } \mathbf{R}_0^i \in \text{SO}(3) \quad (6.70)$$

relativ zum Basissystem Σ_0 , folgt für die Zeitableitung

$$\dot{\mathbf{p}}_0 = \dot{\mathbf{R}}_0^i \mathbf{p}_i + \mathbf{R}_0^i \underbrace{\dot{\mathbf{p}}_i}_{=0} + \dot{\mathbf{d}}_0^i. \quad (6.71)$$

Daraus folgt

$$\dot{\mathbf{p}}_0 = \underbrace{\dot{\mathbf{R}}_0^i (\mathbf{R}_0^i)^T}_{=\mathbf{S}_0^i} (\mathbf{p}_0 - \mathbf{d}_0^i) + \dot{\mathbf{d}}_0^i. \quad (6.72)$$

Dieser Zusammenhang beschreibt anschaulich die Bewegung des Punktes P als Kombination eines translatorischen Anteils $\dot{\mathbf{d}}_0^i$ und eines rotatorischen Anteils $\dot{\mathbf{R}}_0^i (\mathbf{R}_0^i)^T (\mathbf{p}_0 - \mathbf{d}_0^i)$.

Aufgrund der Orthogonalität der Rotationsmatrizen gilt $\mathbf{R}_0^i (\mathbf{R}_0^i)^T = \mathbf{I}_3$. Durch Ableiten nach der Zeit erhält man

$$\dot{\mathbf{R}}_0^i (\mathbf{R}_0^i)^T + \mathbf{R}_0^i (\dot{\mathbf{R}}_0^i)^T = \mathbf{0}_3 \quad (6.73)$$

mit der (3×3) -Nullmatrix $\mathbf{0}_3$. Daraus folgt:

$$\mathbf{S}_0^i = \dot{\mathbf{R}}_0^i (\mathbf{R}_0^i)^T = -\mathbf{R}_0^i (\dot{\mathbf{R}}_0^i)^T = -(\mathbf{S}_0^i)^T. \quad (6.74)$$

Satz 6.9. Sei $\mathbf{R} \in \text{SO}(3)$ eine Rotationsmatrix. Die Matrix $\mathbf{S} = \dot{\mathbf{R}} \mathbf{R}^T$ wird als Winkelgeschwindigkeitsmatrix bezeichnet. Sie ist schiefsymmetrisch (d.h. $\mathbf{S}^T = -\mathbf{S}$) und besitzt daher die Darstellung

$$\mathbf{S} = \begin{bmatrix} 0 & -s_3 & s_2 \\ s_3 & 0 & -s_1 \\ -s_2 & s_1 & 0 \end{bmatrix} = [\mathbf{s}]_\times \quad (6.75)$$

wobei $\mathbf{s} \in \mathbb{R}^3$ den üblichen Winkelgeschwindigkeitsvektor bezeichnet, dessen Richtung entlang der Drehachse ausgerichtet ist und dessen Betrag der Winkelgeschwindigkeit

entspricht.

Bemerkung 6.3. \mathbf{s} hängt dabei eng mit der Achse/Winkel-Parametrierung (6.43) von Rotationen zusammen. Erhält man eine Rotationsmatrix \mathbf{R} beispielsweise durch Achse/Winkel-Parametrierung mit \mathbf{k} ($\|\mathbf{k}\| = 1$) und θ , so folgt für deren Winkelgeschwindigkeitsmatrix

$$\mathbf{S} = [\mathbf{k}\dot{\theta}]_{\times} = [\mathbf{s}]_{\times}. \quad (6.76)$$

Die obige Behauptung, dass \mathbf{s} in Richtung der Rotationsachse zeigt und $\|\mathbf{s}\| = |\dot{\theta}|$ gilt, ist damit gezeigt.

Mit den translatorischen Geschwindigkeiten $\mathbf{v}_0^i = \dot{\mathbf{d}}_0^i$ lässt sich (6.72) in der gewohnten Darstellung

$$\dot{\mathbf{p}}_0 = \mathbf{s}_0^i \times (\mathbf{p}_0 - \mathbf{d}_0^i) + \mathbf{v}_0^i \quad (6.77)$$

schreiben.

Durch Anwendung der Kettenregel auf (6.74) ergibt sich

$$\mathbf{S}_0^i = [\mathbf{s}_0^i]_{\times} = \sum_{j=1}^n \frac{\partial \mathbf{R}_0^i}{\partial q_j}(\mathbf{q}) (\mathbf{R}_0^i(\mathbf{q}))^T \dot{q}_j. \quad (6.78)$$

Die Einträge der Winkelgeschwindigkeitsmatrix sind also linear in \dot{q}_j . Durch Herauspinken ihrer drei unabhängigen Einträge¹⁵ ergibt sich der lineare Zusammenhang

$$\mathbf{s}_0^i = (\mathbf{J}_s)_0^i(\mathbf{q}) \dot{\mathbf{q}} \quad (6.79)$$

mit der *rotatorischen Manipulator Jacobi-Matrix* $(\mathbf{J}_s)_0^i(\mathbf{q})$. Analog ergibt sich

$$\mathbf{v}_0^i = \dot{\mathbf{d}}_0^i = \sum_{j=1}^n \frac{\partial \mathbf{d}_0^i}{\partial q_j}(\mathbf{q}) \dot{q}_j \quad (6.80)$$

und damit direkt

$$\mathbf{v} = (\mathbf{J}_v)_0^i(\mathbf{q}) \dot{\mathbf{q}}. \quad (6.81)$$

Die Matrix $(\mathbf{J}_v)_0^i(\mathbf{q})$ wird dabei als *translatorische Manipulator Jacobi-Matrix* bezeichnet. Fasst man die Beziehungen (6.79) und (6.81) zusammen, ergibt sich folgender Satz:

¹⁵Also die inverse Operation zum Kreuzproduktoperator.

Definition 6.7 (Manipulator Jacobi-Matrix). Die $(6 \times n)$ -Matrix $\mathbf{J}_0^i(\mathbf{q})$ definiert durch

$$\dot{\mathbf{w}} = \begin{bmatrix} \mathbf{v} \\ \mathbf{s} \end{bmatrix} = \begin{bmatrix} (\mathbf{J}_v)_0^i(\mathbf{q}) \\ (\mathbf{J}_s)_0^i(\mathbf{q}) \end{bmatrix} \dot{\mathbf{q}} = \mathbf{J}_0^i \dot{\mathbf{q}} \quad (6.82)$$

wird als *Manipulator Jacobi-Matrix* am Punkt \mathbf{q} bezeichnet und bildet die Gelenkgeschwindigkeiten $\dot{\mathbf{q}}$ linear auf die translatorischen und rotatorischen Geschwindigkeiten $\dot{\mathbf{w}}$ ab.^a

^aAnstatt den Winkelgeschwindigkeiten gemäß (6.75) können alternativ auch die Zeitableitungen der Euler- oder RPY-Winkel verwendet werden, welche sich für eine gegebene Rotation aus (6.17) bzw. (6.23) ergeben. Diese Darstellung wird oft als *analytische* Manipulator Jacobi-Matrix bezeichnet.

Die Geschwindigkeiten im Arbeitsraum \mathcal{W} lassen sich also linear aus den Gelenkswinkelgeschwindigkeiten berechnen. Üblicherweise will man jedoch aus gegebenen translatorischen und rotatorischen Geschwindigkeiten auf die dazu benötigten Gelenkswinkelgeschwindigkeiten schließen. Das resultierende lineare Gleichungssystem (6.82) ist i. A. jedoch nicht oder nicht eindeutig lösbar.

6.2.10. Kinematische Redundanzen und Singularitäten

Wie im Bsp. 6.3 gesehen, ergeben sich für Roboter mit weniger Freiheitsgraden als der Raum - also $n < 6$ - Einschränkungen des Arbeitsraums. Die umgekehrte Überlegung führt zum Begriff der kinematischen Redundanz.

Definition 6.8 (Kinematische Redundanz). Besitzt ein Roboter \mathcal{R}_n einen Arbeitsraum \mathcal{W} mit w Freiheitsgraden und gilt

$$n > w, \quad (6.83)$$

so spricht man von *kinematischer Redundanz*.

Kinematische Redundanz unterscheidet sich von der bloßen Existenz mehrerer Lösungen, wie beim SCARA-Manipulator in Bsp. 6.3. Das inverse Problem kinematisch redundanter Roboter besitzt i. A. ein ganzes Kontinuum an Lösungen. Ein Beispiel für solch einen Fall ist in Abb. 6.21 dargestellt. Der planare Manipulator besitzt fünf Freiheitsgrade, die Bewegungen der Ebene $SE(2)$ kommen jedoch mit 3 Freiheitsgraden - den Koordinaten x, y und einem Drehwinkel ϕ - aus. Auch wenn die Endeffektorposition festgehalten wird, kann sich die Konfiguration *kontinuierlich* verändern. Anders gesprochen: Die Menge der

möglichen Konfigurationen $\bar{\mathbf{q}} \in \mathcal{W}$ zu einer Endeffektorpose $\mathbf{T} = \mathbf{T}_0^E(\bar{\mathbf{q}})$ ist eine Untermannigfaltigkeit des Konfigurationsraums \mathcal{U} .

Satz 6.10 (Eigenbewegungen). *Die Menge aller Punkte $\bar{\mathbf{q}} \in \mathcal{U}$ zu einer Pose $\mathbf{T} \in \mathcal{W}$ eines redundanten Roboters \mathcal{R}_n , für welche*

$$\mathbf{T} = \mathbf{T}_0^E(\bar{\mathbf{q}}) \quad (6.84)$$

gilt, werden als Eigenbewegungen bezeichnet. Diese sind glatt aneinander gebunden, können jedoch in mehrere disjunkte Teilmengen zerfallen^a. Bewegungen innerhalb dieser Menge werden auch als interne Bewegungen bezeichnet.

^ad. h. es kann mehrere Arten kontinuierlicher Bewegungen geben, die jedoch nicht durch eine interne Bewegung ineinander übergeführt werden können.

Für Roboter \mathcal{R}_n mit $n \leq 6$ hängt die Frage kinematischer Redundanz also vom konkreten Arbeitsraum \mathcal{W} ab, während für $n > 6$ wegen den sechs Freiheitsgraden des dreidimensionalen Raumes jedenfalls kinematische Redundanz vorliegt.

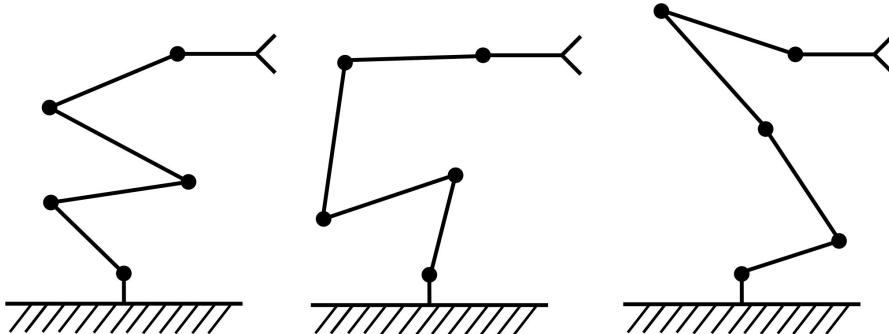


Abbildung 6.21.: Beispiel eines redundanten planaren Manipulators mit $n = 5$ und $w = 3$.

Umgekehrt kann es passieren, dass der Endeffektor eines Roboters \mathcal{R}_n lokal Bewegungsfreiheitsgrade verliert. Durch die Manipulator Jacobi-Matrix $\mathbf{J}_0^E(\bar{\mathbf{q}})$ ist für eine Konfiguration $\bar{\mathbf{q}}$ lokal der Zusammenhang von Gelenkswinkel- und Arbeitsraumgeschwindigkeiten gegeben. Der durch die Spalten der Jacobi-Matrix aufgespannte Raum - ihr Bild $\text{Im}(\mathbf{J}_0^E(\bar{\mathbf{q}}))$ – informiert daher von lokal erreichbaren Posen. Ist die Dimension des Bildes kleiner als jene des Arbeitsraumes, existieren „Richtungen“, welche zumindest lokal nicht angefahren werden können. Dies motiviert folgende Definition:

Definition 6.9 (Kinematische Singularität). Eine Lage des Endeffektors $\mathbf{q}^* \in \mathcal{U}$

eines seriellen Roboters \mathcal{R}_n mit $n \leq 6$ nennt man *singulär*, wenn die Manipulator Jacobi-Matrix $\mathbf{J}_0^i(\mathbf{q}^*)$ nicht-maximalen Rang besitzt. Ansonsten wird diese Lage als *regulär* bezeichnet.

Bemerkung 6.4. Für den besonders wichtigen Fall von Robotern mit sechs Gelenken ist gemäß Def. 6.9 eine Konfiguration \mathbf{q}^* genau dann singulär, wenn $\det(\mathbf{J}_0^i(\mathbf{q}^*)) = 0$ gilt.

Die Bedeutung singulärer Punkte folgt aus ihren zum Teil unangenehmen Eigenschaften:

1. In der Umgebung von Singularitäten besitzt das inverse kinematische Problem keine eindeutige Lösung.
2. Es besteht ein unstetiger Zusammenhang von \mathbf{w} und \mathbf{q} . Für geringe Geschwindigkeiten im Arbeitsraum ergeben sich hohe Gelenksgeschwindigkeiten in der Umgebung von Singularitäten.
3. Beschränkte Kräfte bzw. Momente am Manipulator können zu unbeschränkten Kräften bzw. Momenten in den Gelenken führen.
4. Typischerweise treten Singularitäten an den Rändern der Arbeitsbereiche auf.

Bei der Planung von Bewegungsbahnen sind singuläre Stellungen des Manipulators daher zu beachten.

6.2.11. Differentielles Modell

Während die direkte, analytische Lösung des inversen kinematischen Problems bei seriellen Robotern schon in einfachen Fällen ein nicht-triviales Problem darstellt, erhöhen kinematische Singularitäten und Redundanzen den Aufwand nochmals erheblich. Auch eine numerische Lösung mittels Newton-Raphson-Verfahren scheint für redundante Roboter wenig praktikabel. Durch die Manipulator Jacobi-Matrix besteht ein linearer Zusammenhang zwischen den zeitlichen Änderungsraten im Arbeits- und Konfigurationsraum, welcher auch als differentielles Modell des Manipulators

$$\dot{\mathbf{w}} = \mathbf{J}_0^i(\mathbf{q}) \dot{\mathbf{q}} \quad (6.85)$$

bezeichnet wird. Differentielle Modelle werden dabei aus unterschiedlichen Gründen eingesetzt.

Inverses Problem für Bahnkurven

Das inverse kinematische Problem stellt sich in der Praxis häufig nicht für einzelne Punkte, sondern für vorgegebene Bahnkurven. Unter einer Bahnkurve verstehen wir hier eine durch die Zeit $t \in [t_a, t_b]$ parametrierte Kurve $\mathbf{w}(t) \subset \text{SE}(3)$.

Für den Fall $n = 6$ ist die Manipulator Jacobi-Matrix quadratisch und lässt sich für alle \mathbf{q} mit $\det(\mathbf{J}_0^E(\mathbf{q})) \neq 0$ invertieren. Man erhält

$$\dot{\mathbf{q}}(t) = (\mathbf{J}_0^E)^{-1}(\mathbf{q}(t)) \dot{\mathbf{w}}(t). \quad (6.86)$$

Diskretisiert man das Zeitintervall $[t_a, t_b]$ durch k Stützstellen $t_a = t_0 < t_1 < \dots < t_{k-1} = t_b$, so ergibt sich der Zusammenhang zwischen zwei aufeinander folgenden Stützstellen durch Integration

$$\mathbf{q}(t_{i+1}) - \mathbf{q}(t_i) = \int_{t_i}^{t_{i+1}} (\mathbf{J}_0^E)^{-1}(\mathbf{q}(\tau)) \dot{\mathbf{w}}(\tau) d\tau. \quad (6.87)$$

Durch die Annahme, dass sich $(\mathbf{J}_0^E)^{-1}(\mathbf{q}(\tau))$ für $\tau \in [t_i, t_{i+1}]$ nur wenig verändert, folgt die Approximation

$$\mathbf{q}(t_{i+1}) = \mathbf{q}(t_i) + (\mathbf{J}_0^E)^{-1}(\mathbf{q}(t_i)) \underbrace{(\mathbf{w}(t_{i+1}) - \mathbf{w}(t_i)) (t_{i+1} - t_i)}_{\Delta \mathbf{w}}. \quad (6.88)$$

Durch die Abkürzung $\mathbf{q}(t_i) = \mathbf{q}_i$ folgt damit

$$\mathbf{q}_{i+1} = \mathbf{q}_i + (\mathbf{J}_0^E)^{-1}(\mathbf{q}_i) \Delta \mathbf{w}. \quad (6.89)$$

Diese Iterationsvorschrift berechnet aus einem bekannten \mathbf{q}_0 , welches das inverse Problem $\mathbf{T}_0^E(\mathbf{q}_0) = \mathbf{w}(t_a)$ löst, eine approximative Lösung des inversen Problems für alle folgenden Zeiten $t \in [t_a, t_b]$. Da sich das inverse Problem für spezielle Lagen oft intuitiv geometrisch lösen lässt, kann so für alle folgenden Bahnpunkte eine Lösung gefunden werden.

Vermeidung von Singularitäten

Für eine singuläre Manipulator Jacobi-Matrix ist lediglich eine Bewegung in *beliebige, allgemeine* Richtungen nicht mehr möglich, d. h. für spezielles $\dot{\mathbf{w}}$ kann durchaus eine Lösung existieren. Zudem wissen wir, dass nahe an singulären Punkten die nötige Gelenkgeschwindigkeit sehr groß wird. Betrachtet man die Lösung des Optimierungsproblems

$$\min_{\dot{\mathbf{q}}} \{ \| \dot{\mathbf{w}} - \mathbf{J}_0^E(\mathbf{q}) \dot{\mathbf{q}} \|^2 + \lambda^2 \| \dot{\mathbf{q}} \|^2 \}, \quad (6.90)$$

so stimmt dessen Lösung für $\lambda = 0$ mit der Lösung des inversen differentiellen Problems (6.85) überein. Für $|\lambda| > 0$ werden singuläre Punkte zulasten von Bahnfehlern zunehmend umfahren.

Die Lösung des unbeschränkten Optimierungsproblems¹⁶ (6.90) muss

$$\frac{d}{d\dot{\mathbf{q}}} \left[(\dot{\mathbf{w}} - \mathbf{J}_0^E(\mathbf{q}) \dot{\mathbf{q}})^T (\dot{\mathbf{w}} - \mathbf{J}_0^E(\mathbf{q}) \dot{\mathbf{q}}) + \lambda^2 \dot{\mathbf{q}}^T \dot{\mathbf{q}} \right] = 0 \quad (6.91)$$

erfüllen, woraus sich nach kurzer Rechnung die Lösung

$$\dot{\mathbf{q}}_\lambda = [(\mathbf{J}_0^E)^T (\mathbf{J}_0^E) + \lambda^2 \mathbf{I}]^{-1} (\mathbf{J}_0^E)^T \dot{\mathbf{w}} \quad (6.92)$$

ergibt. Dieser modifizierte Zusammenhang zwischen Arbeits- und Konfigurationsraum kann beispielsweise für Bahnkurven wieder wie oben beschrieben gelöst werden.

Redundanzen

Im Fall von kinematisch redundanten Robotern ist die Manipulator Jacobi-Matrix nicht quadratisch, da gemäß (6.83) der Konfigurationsraum \mathcal{U} mehr Freiheitsgrade als der Arbeitsraum \mathcal{W} aufweist. Aus der möglichen Menge der Eigenbewegungen (6.84) sollte eine eindeutige Lösung ausgewählt werden. Dies kann durch

1. zusätzliche kinematische Zwangsbedingungen, oder
2. Optimalität hinsichtlich eines Gütekriteriums erfolgen.

Ersteres bewirkt also eine Erweiterung der Jacobi-Matrix zu einer regulären, quadratischen Matrix. Letzteres hängt ganz vom gewählten Gütekriterium ab. Werden beispielsweise Bewegungen mit minimaler Norm bevorzugt, führt dies zu einem Optimierungsproblem ähnlich (6.90).

¹⁶Im Allgemeinen sind die gültigen Lösungen, also der Konfigurationsraum \mathcal{U} , nur eine Teilmenge des \mathbb{R}^n , weshalb es sich um ein Optimierungsproblem mit Nebenbedingungen handelt (siehe VO Optimierung).

6.3. Bahnhplanung

Die bisherigen Ausführungen haben sich mit den kinematischen Zusammenhängen zwischen Konfigurations- und Arbeitsraum beschäftigt, welche die kinematischen Strukturen des Roboters generieren und als direktes und inverses Problem bezeichnet werden. Beide hängen nur von dem Roboter intrinsischen kinematischen Eigenschaften ab. Gerade aus ingenieurwissenschaftlicher Perspektive steht die Fähigkeit, mit einem Roboter gezielt Bewegungen zwischen zwei oder mehreren Punkten auszuführen, im Vordergrund. Die möglichen Bewegungen unterliegen aufgrund physischer Hindernisse im Arbeitsraum, der Aufgabenstellung und den Fähigkeiten der Aktorik verschiedenen Beschränkungen.

6.3.1. Pfade und Bahnen

Um eine wichtige Unterscheidung zu treffen, sollten wir uns zuerst ein Bild über zwei Begriffe machen, welche in diesen Bereich verwendet werden.

Definition 6.10 (Pfad). Unter einem Pfad oder Weg zwischen einem Anfangspunkt \mathbf{q}_A und einem Endpunkt \mathbf{q}_E versteht man eine stetige Abbildung $\tau : [0, 1] \mapsto \mathcal{U}$ mit $\tau(0) = \mathbf{q}_A$ und $\tau(1) = \mathbf{q}_E$.

Eng mit dem Begriff des Pfades verknüpft ist jener der Bahn bzw. der Trajektorie.

Definition 6.11 (Bahn). Unter einer Bahn oder Trajektorie zwischen einem Anfangspunkt \mathbf{q}_A und einem Endpunkt \mathbf{q}_E versteht man eine stetige Abbildung $\mathbf{q}(t) : [t_A, t_E] \mapsto \mathcal{U}$ mit den Anfangs- und Endzeiten t_A, t_E , sodass $\mathbf{q}(t_A) = \mathbf{q}_A$ und $\mathbf{q}(t_E) = \mathbf{q}_E$.

Beide Definitionen sind hier im Konfigurationsraum \mathcal{U} formuliert, kann jedoch auch im Arbeitsraum \mathcal{W} formuliert werden. Dabei müsste man jedoch fordern, dass zumindest die Anfangskonfiguration zu einer gegebenen Anfangspose eindeutig festgelegt bzw. gegeben sein muss. Zudem enthält die Angabe der Konfiguration *immer* Information über den gesamten Zustand des Roboters und nicht bloß des Endeffektors¹⁷.

Vergleicht man die beiden Definitionen, so unterscheiden sich die beiden lediglich dadurch, dass eine Bahn durch ein Zeitintervall parametrisiert wird und somit eine kinematische Größe darstellt. Die Unterscheidung knüpft jedoch zwei grundlegend verschiedene Probleme an: Die *Planung eines möglichen Pfades* durch einen mit Hindernissen beleg-

¹⁷z.B. bei redundanten Robotern.

ten Arbeitsraum gegenüber der *Planung einer möglichen Bahn* unter kinematischen und dynamischen Randbedingungen des Roboters. Die beiden Probleme unterscheiden sich also im Wesen ihrer Einschränkungen.

Zur algorithmischen Lösung des Pfadplanungsproblems wurden eine Reihe von globalen und lokalen Methoden entwickelt, deren Lösung im Vergleich zur knappen Aufgabenstellung einiges an rechnerischem Aufwand generieren. Typische Verfahren basieren beispielsweise auf Sichtlinien, stochastischen Suchalgorithmen oder auf fiktiven, konstruierten Potentialfeldern, welche den Endeffektor zum Zielpunkt ziehen und von Hindernissen abstoßen.

Das Ergebnis einer erfolgreichen algorithmischen Pfadplanung ist, letztlich schon aufgrund der Verarbeitung in Computersystemen, eine Folge von Punkten (P_i), $i = 0 \dots N - 1$, welche vom Roboter durchfahren werden sollen. Dabei kann es sich je nach Verfahren um Punkte im Konfigurations- oder Arbeitsraum handeln. Die Aufgabe der Bahnplanung besteht nun darin, einen stetigen Verlauf aus den Punkten (P_i) zu generieren, welcher vom Roboter kollisionsfrei und innerhalb der dynamischen Limitierungen ausgeführt werden kann. Dabei werden in der Regel die einzelnen Teilbewegungen $P_i \rightarrow P_{i+1}, i = 1, \dots, n - 2$ sequentiell abgearbeitet.

Besonders in der Anwendung von Industrierobotern wird anstatt algorithmischer Pfadplanungsmethoden oft auf eine manuelle Planung zurückgegriffen, wie in Abschnitt 6.1.6 erwähnt. Dabei werden nicht alle Punkte gleich behandelt, sondern in folgende Punktekategorien unterschieden:

Start- und Zielpunkte S bzw. Z bilden die Anfangs- und Endpunkte einer durchgängigen Bewegung, welche (meist) mit Geschwindigkeit Null exakt zu erreichen sind.

Durchpunkte D sind exakt mit definierter Geschwindigkeit zur durchfahren.

Viapunkte V sind lediglich möglichst nahe zu passieren. Dies ermöglicht der Planer_in, bekannte Hindernisse im Arbeitsraum zu umfahren.

Abb. 6.22 zeigt ein Beispiel eines durch solche Punkte festgelegten Pfades. Viapunkte geben dem Bahnplanungsalgorithmus dabei zusätzliche Freiheitsgrade. Die Planung kann also nach anderen Gütekriterien wie Bewegungsgenauigkeit oder Zeitoptimalität geführt werden.

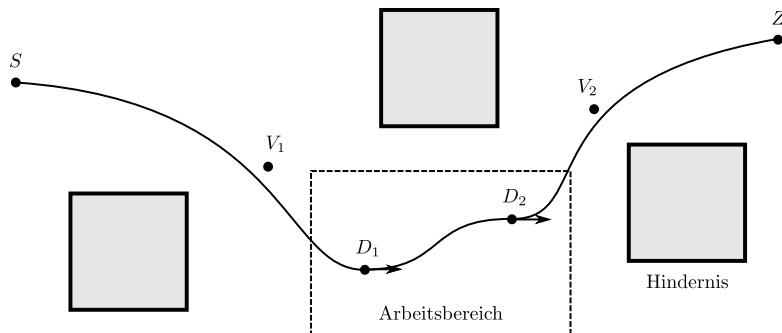


Abbildung 6.22.: Beispiel einer manuellen Pfadangabe mittels Start-, Durch-, Via- und Endpunkte.

6.3.2. Arten der Bahnplanung

Innerhalb dieser Vorlesung werden lediglich kinematische Modelle von Robotern behandelt. Durch die Verwendung des Lagrange-Formalismus¹⁸ lassen sich daraus jedoch mit relativ geringem Aufwand kinetische (d. h. dynamische) Modelle ableiten. Je nachdem, welche Modelle zur Planung verwendet werden, unterscheidet man zwischen

kinematischer Bahnplanung Hierbei wird der Roboter durch ein kinematisches Modell repräsentiert. Die dynamischen Limitierungen werden durch Begrenzungen der Gelenkgeschwindigkeiten $\dot{q}_{i,max}$ und -beschleunigungen $\ddot{q}_{i,max}$ berücksichtigt, ohne die dynamischen Wechselwirkungen mitzurechnen. Gesucht wird also eine Trajektorie $\mathbf{q}(t)$ mit

$$\max\{\dot{q}_i(t)\} \leq \dot{q}_{i,max} \quad \text{und} \quad \max\{\ddot{q}_i(t)\} \leq \ddot{q}_{i,max} \quad (6.93)$$

für $n = 1, \dots, n$ und zu wählender Zeitdauer $t_E - t_A$. Für konservative Bahnverläufe (etwa $< 1 \text{ m/s}$) genügt dieses einfache Verfahren.

kinetischer Bahnplanung Sollen höhere Geschwindigkeiten ((etwa $> 2 \text{ m/s}$) erreicht werden, werden kinetische Modelle benötigt, um die physikalische Realität ausreichend genau zu erfassen.

Eine andere Unterscheidung betrifft den Raum, in welchem die Bahnplanung stattfindet (siehe Abb. 6.23).

Planung im Konfigurationsraum \mathcal{U} Eine Planung im Gelenksraum liefert direkt Sollgrößen für die verallgemeinerten Gelenkwinkel $\mathbf{q}_{soll}(t)$. Zudem wird das Problem

¹⁸siehe Vorlesung *Modellbildung* bzw. Fachvertiefung *Automatisierungs- und Regelungstechnik*

von Singularitäten im Arbeitsraum vermieden. Die Bahn im Arbeitsraum kann einfach über $\mathbf{w}_{soll}(t) = \mathbf{T}_0^E(\mathbf{q}_{soll}(t))$ berechnet werden. Den geringen Rechenaufwand erkauft man sich jedoch mit dem Nachteil komplizierter Bahnen im Arbeitsraum, welche zusätzlich eine Kollisionsüberwachung notwendig machen.

Planung im Arbeitsraum \mathcal{W} Wird die Bahn direkt im Arbeitsraum geplant, ergeben sich deutlich kompliziertere Bahnen im Konfigurationsraum. Durch die Bahnplanung im Arbeitsraum vereinfacht sich die Kollisionsvermeidung, der Rechenaufwand steigt jedoch, da die Lösung des inversen Problems benötigt wird.

Der unterschiedliche Rechenaufwand der beiden Methoden folgt aus der Häufigkeit, mit welcher die algorithmisch aufwendige inverse Lösung berechnet werden muss. Wie in Abb. 6.24 dargestellt, werden bei Planung im Konfigurationsraum die üblicherweise im Arbeitsraum vorgegebenen Pfadpunkte $(T_{soll})_i$ in verallgemeinerte Gelenkwinkel $(q_{soll})_i$ umgerechnet und durch Interpolation als ein (quasi-) kontinuierlicher Pfad $q_{soll}(t)$ dargestellt. Während die Interpolation sehr fein erfolgen muss, um dem geregelten Manipulator möglichst glatte Verläufe vorzugeben, kann die inverse Lösung wesentlich seltener aufgerufen werden, d. h. $\tau_{inv} \gg \tau_{int}$. Die Planung im Arbeitsraum hingegen interpoliert zuerst, weshalb auch die inverse Lösung im Interpolationstakt aufgerufen werden muss. Typische Werte der beiden Zeiten sind $\tau_{int} \approx 2 \text{ ms}$ und $\tau_{inv} \approx 0,1 \text{ s} - 1 \text{ s}$. Die inverse Lösung wird also bei Bahnplanung im Arbeitsraum um einen Faktor 50–500 häufiger aufgerufen als bei Pendant im Konfigurationsraum.

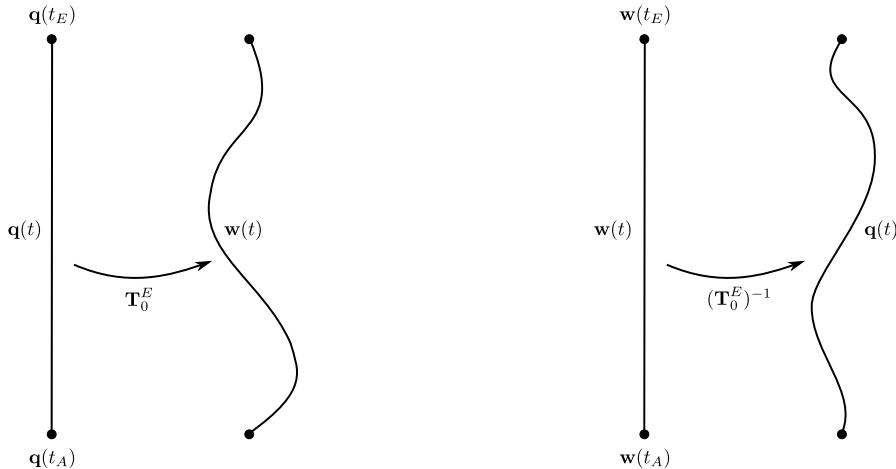


Abbildung 6.23.: Beispielhafte Bahnen bei Planung im Gelenksraum (links) und im Arbeitsraum (rechts).

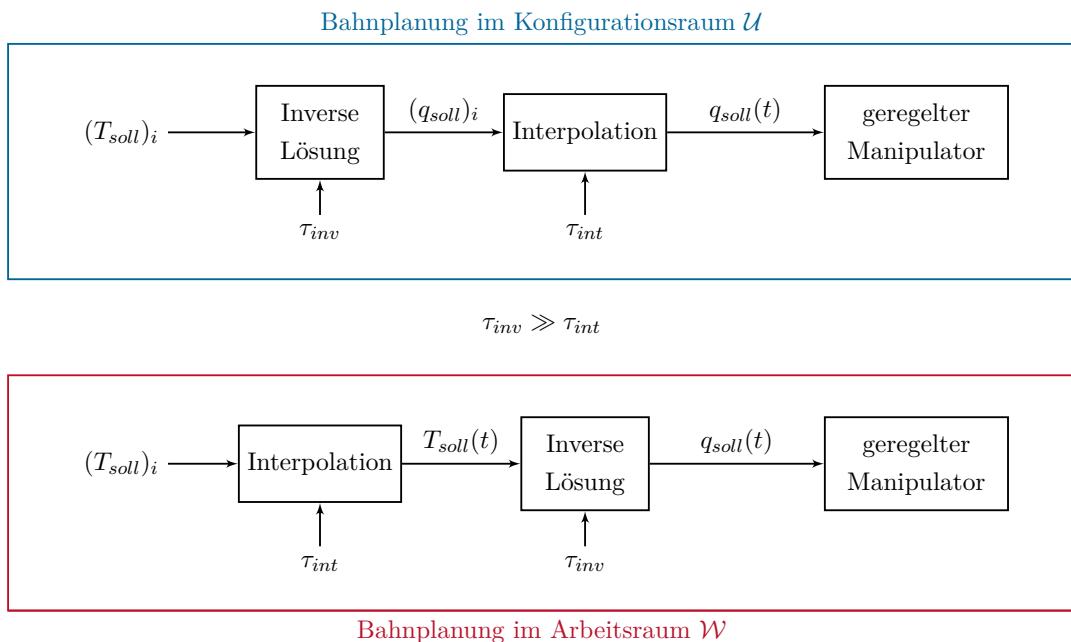


Abbildung 6.24.: Rechentechnische Implementierung von konfigurationsraum- und arbeitsraumorientierter Bahnplanung.

6.3.3. Punkt-zu-Punkt-Bewegungen

Wie oben erwähnt, wird die Planung einer Bahn oft iterativ als Planung von $N - 1$ Punkt-zu-Punkt-Bewegungen gelöst. Grundsätzlich ließe sich das komplette Bahnplanungsproblem für alle N Punkte durch ein Polynom $N - 1$ -ten Grades lösen. Problematisch dabei ist, dass Polynome hoher Ordnung i. A. von einem gewünschten, „glatten“ Verlauf stark abweichen und Oszillationen um die Stützstellen auftreten können¹⁹. Daher wird meist auf *Splines*, also stückweise Polynomfunktionen geringer Ordnung, oder stückweise trigonometrische Funktionen zurückgegriffen.

Für einen Roboter \mathcal{R}_n müssen also n geeignete, stetige Funktionen zwischen den Punkten $\mathbf{q}_A = \mathbf{q}(t_A)$ und $\mathbf{q}_E = \mathbf{q}(t_E)$ unter der Einhaltung spezifischer Randbedingungen (wie Geschwindigkeit und Beschleunigung) gefunden werden²⁰, wozu die Wahl der Zeitspanne $\Delta t = t_E - t_A$ i. A. frei ist. Ohne Beschränkung der Allgemeinheit kann jede Komponente separat geplant werden, weshalb wir im Folgenden nur den eindimensionalen Fall betrachten.

Polynom-Interpolation

Es wird eine Funktion $q(t) : [t_A, t_E] \rightarrow X \subset \mathbb{R}$ gesucht, deren Werte am Rand durch q_A und q_E vorgegeben sind. Zusätzlich sollen die Geschwindigkeit an den Rändern v_A, v_E vorgeben werden können. Eine erste Wahl zur Interpolation sind Polynome geeigneter Ordnung. Um vier Bedingungen erfüllen zu können, wird ein Polynom dritter Ordnung

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (6.94)$$

gewählt. Die vier Randbedingungen

$$q(t_A) = q_A \quad q(t_E) = q_E \quad \dot{q}(t_A) = v_A \quad \dot{q}(t_E) = v_E \quad (6.95)$$

führen mit der Zeitableitung des Polynoms (6.94)

$$\dot{q}(t) = a_1 + 2a_2 t + 3a_3 t^2 \quad (6.96)$$

¹⁹vgl. „RUNGES Phänomen“.

²⁰Die Vorgehensweise im Arbeitsraum ist im Wesentlichen äquivalent. Für die Parametrierung der Drehgruppe $SO(3)$ wird eine sinnvolle Darstellungsform aus Abschnitt 6.2.4 gewählt, wobei ggf. auf singuläre Punkte geachtet werden muss.

zum Gleichungssystem

$$\begin{bmatrix} 1 & t_A & t_A^2 & t_A^3 \\ 1 & t_E & t_E^2 & t_E^3 \\ 0 & 1 & 2t_A & 3t_A^2 \\ 0 & 1 & 2t_E & 3t_E^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} q_A \\ q_E \\ v_A \\ v_E \end{bmatrix}. \quad (6.97)$$

Die Determinante der Koeffizientenmatrix ist gleich $(t_E - t_A)^4$ und damit größer Null für alle nicht-identen Anfangs- und Endzeiten. Das Gleichungssystem (6.97) ist also immer eindeutig lösbar. Die maximale Zeitableitung \dot{q}_{max} dieser Interpolation erhält man durch

$$\dot{q}_{max} = \max_{t \in [t_A, t_E]} \{\dot{q}(t)\} = \max\{\dot{q}(t_A), \dot{q}(t_E), \dot{q}(\bar{t})\} \quad (6.98)$$

mit $\bar{t} = -\frac{a_2}{3a_3}$. Umgekehrt kann jedoch nicht direkt von einer vorgegebenen maximalen Geschwindigkeit und Anfangszeit auf eine passende Endzeit geschlossen werden. Hier muss also iterativ vorgegangen werden.

Werden mehrere Lösungen (6.94) aneinandergefügt, sind diese bei geeigneten Randbedingungen zwar stets stetig differenzierbar, die zweite Ableitung $\ddot{q}(t)$ ist jedoch i. A. unstetig. Dies führt zu ruckartigen Bewegungen, welche Eigenschwingungen des Roboters anregen können. Deshalb wird oft auf Polynome fünfter Ordnung zurückgegriffen, sodass auch die zweiten Ableitungen an den Rändern frei gewählt werden können. Die Vorgehensweise ist analog zu (6.97).

Lineare Segmente mit parabolischen Übergängen (LSPB)

Soll sich der Roboter mit konstanter Geschwindigkeit bewegen, verwendet man oft die Methode der linearen Segmente mit parabolischen Übergängen (*linear segments with parabolic blends* LSPB). Das Verfahren kann dabei sowohl für Durch-, End- oder Viapunkte verwendet werden, wobei wir uns zuerst den Viapunkten zuwenden.

Viapunkte

Die Idee der LSPB-Methode ist, eine Phase mit konstanter Geschwindigkeit durch parabolische Beschleunigungs- oder Verzögerungsphasen zu verbinden. In einem Bereich²¹ $t_i - t_{BE} < t \leq t_i + t_{BE}$ um t_i soll eine quadratische Funktion gesucht werden, welche stetig differenzierbar an eine lineare Funktion zwischen $t_i + t_{BE}$ und $t_{i+1} - t_{BE}$ anschließt.

²¹Die Beschleunigungszeit t_{BE} kann i. A. für jeden Punkt verschieden gewählt werden. Später werden wir kurz darauf zurückkommen.

Der Zeitrahmen *eines* Planungsabschnitts reicht also von $t_i - t_{BE}$ bis $t_{i+1} - t_{BE}$. Für zwei allgemeine Funktionen

$$h_1(t) = a_0 + a_1t + a_2t^2 \quad \text{und} \quad h_2(t) = b_0 + b_1t \quad (6.99)$$

ergeben sich die Bedingungen

$$h_1(t_i - t_{BE}) = q(t_i - t_{BE}), \quad (6.100a)$$

$$\dot{h}_1(t_i - t_{BE}) = \dot{q}(t_i - t_{BE}), \quad (6.100b)$$

$$h_1(t_i + t_{BE}) - h_2(t_i + t_{BE}) = 0, \quad (6.100c)$$

$$\dot{h}_1(t_i + t_{BE}) - \dot{h}_2(t_i + t_{BE}) = 0 \quad \text{und} \quad (6.100d)$$

$$h_2(t_{i+1}) = q_{i+1}. \quad (6.100e)$$

Die Werte $q(t_i - t_{BE})$ und $\dot{q}(t_i - t_{BE})$ sind ein Ergebnis des vorherigen Planungsabschnittes. Die Bedingungen (6.100a)-(6.100d) halten die oben genannten Stetigkeits- und Differenzierbarkeitsanforderungen ein. Da am rechten Rand $t_{i+1} - t_{BE}$ kein Wert angegeben werden kann, wird als fünfte Bedingung für den Zeitpunkt t_{i+1} formuliert (siehe (6.100e)). Mit der Annahme, dass auch im vorherigen Abschnitt mit dem LSPB-Verfahren geplant wurde, also im Bereich $t_{i-1} + t_{BE}$ bis t_i ein Geradenstück eingepasst wurde, folgt für die Ableitung $\dot{q}(t_i - t_{BE}) = \frac{q_i - q(t_i - t_{BE})}{t_{BE}}$, womit sich (6.100a)-(6.100e) als

$$\begin{bmatrix} 1 & t_i - t_{BE} & (t_i - t_{BE})^2 & 0 & 0 \\ 0 & 1 & 2(t_i - t_{BE}) & 0 & 0 \\ 1 & t_i + t_{BE} & (t_i + t_{BE})^2 & -1 & -(t_i - t_{BE}) \\ 0 & 1 & 2(t_i + t_{BE}) & 0 & 0 \\ 0 & 0 & 0 & 1 & t_{i+1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ b_0 \\ b_1 \end{bmatrix} = \begin{bmatrix} q(t_i - t_{BE}) \\ \frac{q_i - q(t_i - t_{BE})}{t_{BE}} \\ 0 \\ 0 \\ q_{i+1} \end{bmatrix} \quad (6.101)$$

darstellen lassen mit der Determinante der Koeffizientenmatrix $-4(t_{i+1} - t_i)t_{BE} < 0$. Abseits von Werten q_i der Pfadplanung und dem Zeitpunkt t_i des vorherigen Planungsabschnitts benötigt dieses Gleichungssystem den Wert $q(t_i - t_{BE})$ des vorherigen Abschnitts. Für mehrere Punkte entsteht also eine Kette an Gleichungssystemen, welche sukzessive gelöst werden müssen. Es kann gezeigt werden, dass für Viapunkte die Geradenabschnitte immer der stückweise linearen Approximation

$$q_{lin}(t) = \begin{cases} \dots \\ q_{i-1} + \frac{q_i - q_{i-1}}{t_i - t_{i-1}}(t - t_{i-1}) & \text{für } t \in [t_{i-1}, t_i] \\ q_i + \frac{q_{i+1} - q_i}{t_{i+1} - t_i}(t - t_i) & \text{für } t \in [t_i, t_{i+1}] \\ \dots \end{cases} \quad (6.102)$$

entsprechen, wie in Abb. 6.25 dargestellt. Daraus kann schließlich der linke Randwert zu

$$q(t_i - t_{BE}) = q_{i-1} + \frac{q_i - q_{i-1}}{t_i - t_{i-1}} (t_i - t_{BE} - t_{i-1}) \quad (6.103)$$

bestimmt werden, womit (6.101) vom vorherigen Abschnitt unabhängig wird und separat für jeden Punkt q_i gelöst werden kann.

Abb. 6.25 zeigt das Resultat für drei Viapunkte $q_{i-1} \dots q_{i+1}$. Durch die LSPB-Methode kann zwar ein stetiger Verlauf der Geschwindigkeit, nicht jedoch der Beschleunigung gesichert werden und führt daher zu einer stark ruckbehafteten Bewegung.

Durchpunkte

Zur Planung eines Durchpunktes q_i mittels LSPB-Verfahrens muss etwas mehr Aufwand betrieben werden. Betrachtet man die Bedingungen (6.100), so fällt auf, dass alle fünf bereits zur Erfüllung der Stetigkeits- und Differenzierbarkeitsanforderungen bzw. des Endpunkts benötigt werden. Die einzigen Freiheitsgrade am linken Rand ($q(t_i - t_{BE})$ und $\dot{q}(t_i - t_{BE})$) sind bereits durch die Lösung des vorhergehenden Abschnitts bestimmt. Durch Hinzufügen eines virtuellen Hilfspunktes kann dies umgangen werden. Dazu wird ein unbestimmter Hilfspunkt q_h an der Stelle $t_i - t_{BE}$ eingeführt, welcher vom vorherigen Abschnitt gerade angefahren wurde. Dadurch folgt für (6.101), dass durch $q(t_i - t_{BE}) \rightarrow q_h$ und $\dot{q}(t_i - t_{BE}) \rightarrow \frac{q_h - q_{i-1}}{t_i - t_{BE} - t_{i-1}}$ ein neuer Freiheitsgrad eingeführt wird. Dieser wird durch die zusätzliche Durchtrittsbedingung

$$h_1(t_i) = q_i \quad (6.104)$$

ergänzt. In Matrixform folgt daher

$$\underbrace{\begin{bmatrix} 1 & t_i - t_{BE} & (t_i - t_{BE})^2 & 0 & 0 & -1 \\ 0 & 1 & 2(t_i - t_{BE}) & 0 & 0 & \frac{-1}{t_i - t_{BE} - t_{i-1}} \\ 1 & t_i + t_{BE} & (t_i + t_{BE})^2 & -1 & -(t_i - t_{BE}) & 0 \\ 0 & 1 & 2(t_i + t_{BE}) & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & t_{i+1} & 0 \\ 1 & t_i & t_i^2 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{K}} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ b_0 \\ b_1 \\ q_h \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{-q_i}{t_i - t_{BE} - t_{i-1}} \\ 0 \\ 0 \\ q_{i+1} \\ q_i \end{bmatrix}. \quad (6.105)$$

Die Determinante der Koeffizientenmatrix

$$\det(\mathbf{K}) = -\frac{-t_{BE}^2 (t_{i+1} - t_{i-1}) + 4 t_{BE} (t_{i+1} - t_i)(t_i - t_{i-1})}{t_i - t_{i-1} - t_{BE}} \quad (6.106)$$

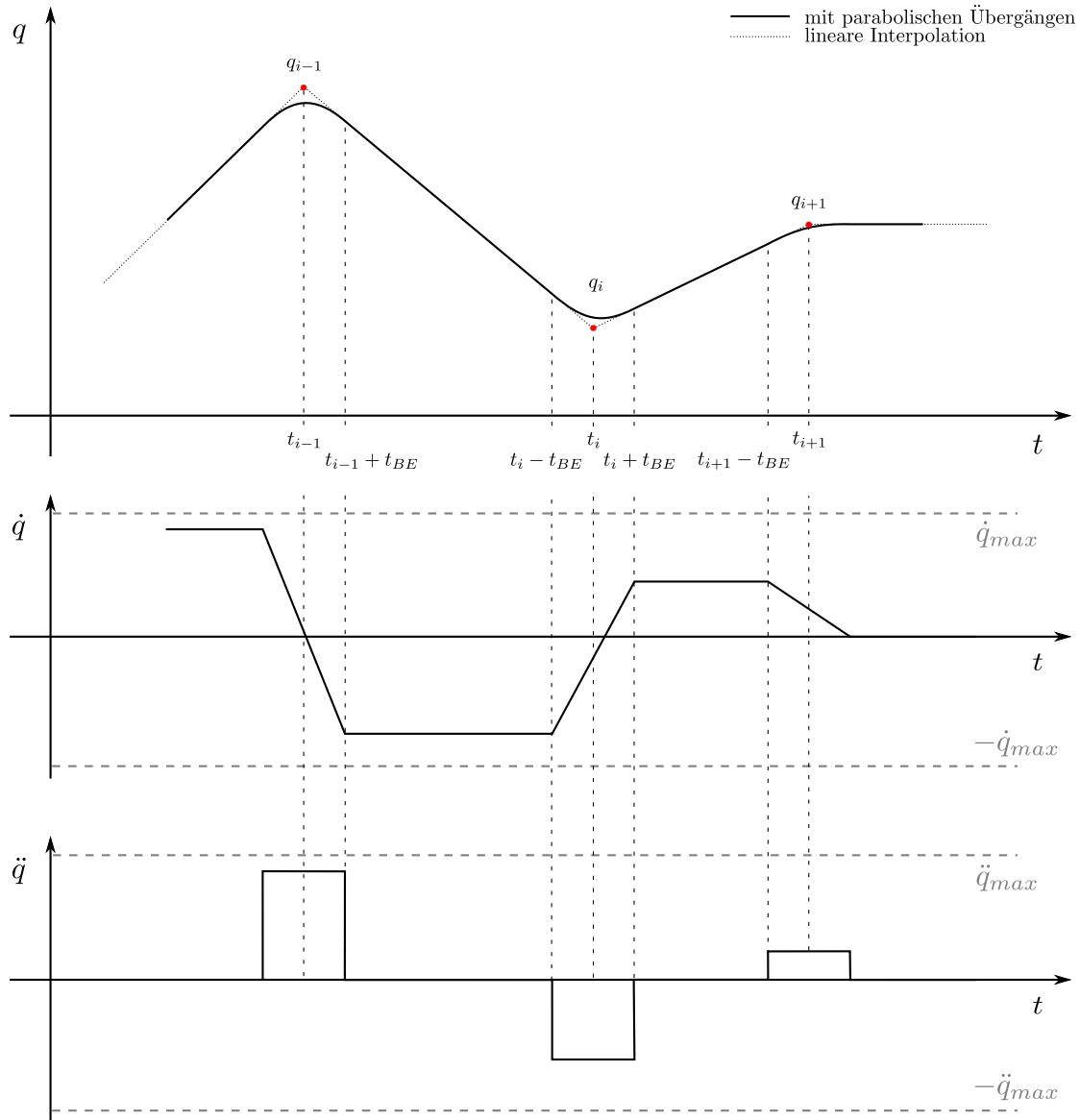


Abbildung 6.25.: Beispielhafte Bahnplanung per LSPB-Methode für Viapunkte.

ist im Allgemeinen nicht ungleich Null und somit kann das Gleichungssystem (6.105) keine Lösung besitzen. Es ist für äquidistante Zeiten $t_i - t_{i-1} = \Delta t$ einfach zu zeigen, dass die Beschleunigungsdauer $t_{BE} \neq k \Delta t$ mit $k \in \{1, 2\}$, was jedoch schon aus praktischen Gründen widersinnig ist, da für $t_{BE} \geq \frac{\Delta t}{2}$ die Lösung unstetig wird.

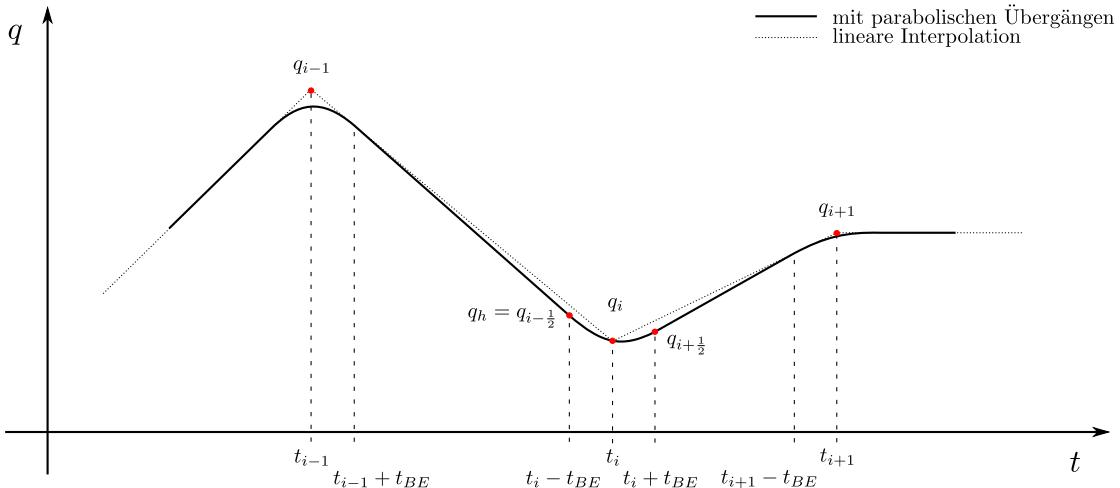


Abbildung 6.26.: Beispielhafte Bahnplanung per LSPB-Methode eines Durchpunkts q_i .

In Abb. 6.26 ist eine beispielhafte Bahn eines Durchpunktes dargestellt. Die Geradenstücke können wegen des Versatzes von q_h nicht mehr mit der linearen Interpolation übereinstimmen. Die Verwendung des LSPB-Verfahrens für Durchpunkte wirkt sich auf das vorherige Segment aus, da dessen Geradenstück nicht mehr gegen q_i sondern gegen q_h einlaufen soll. Dies kann durch Einfügen eines Zwischenpunktes $q_{i-\frac{1}{2}} = q_h$ kompensiert werden. Selbiges gilt für die Gerade rechts von q_i für den Fall eines weiteren Durchpunktes q_{i+1} , weshalb auch hier ein Zwischenpunkt $q_{i+\frac{1}{2}} = h_1(t_i + t_{BE})$ eingefügt wird.

Bestimmung der Zeitpunkte

Die Wahl der Zeitpunkte t_i und der Beschleunigungsdauer t_{BE} müssen bereits vor Berechnung der Interpolation (6.101) bzw. (6.105) festgelegt werden. Aufgrund der a priori-Kenntnisse des Verlaufs folgt für Viapunkte sofort

$$\dot{q}_{i,max} = \frac{q_{i+1} - q_i}{t_{i+1} - t_i} \leq \dot{q}_{max} \quad (6.107)$$

$$\ddot{q}_{i,max} = \frac{\dot{q}_{i,max} - \dot{q}_{i-1,max}}{2 t_{BE}} \leq \ddot{q}_{max} \quad (6.108)$$

Aus (6.107) lässt sich aus dem bekannten aktuellen Zeitpunkt, dieser wurde im vorhergehenden Planungsabschnitt bestimmt und angefahren, ein geeigneter Zeitpunkt t_{i+1} bestimmen. Damit ist auch eine geeignete Beschleunigungsduer t_{BE} über (6.108) festgelegt. Die Wahl der beiden Größen ist jedoch nicht vollkommen unabhängig; es muss geprüft werden, ob die resultierende Beschleunigungszeit des Punktes t_i mit den Beschleunigungsbereichen des vorangehenden oder folgenden Punktes überlappt, also

$$t_{i-1} + t_{BE,i-1} \leq t_i - t_{BE,i} \quad \text{und} \quad t_i + t_{BE,i} \leq t_{i+1} - t_{BE,i+1}, \quad (6.109)$$

wobei der Index der Beschleunigungszeiten $t_{BE,i}$ die Zugehörigkeit zum Punkt q_i bezeichnet. Vereinfachend kann der Zeitbereich zwischen zwei Punkten halbiert werden und den jeweiligen Punkten als Beschleunigungsbereich dienen. Damit folgt die vereinfachte Bedingung

$$t_{BE,i} \leq \min\left(\frac{t_{i+1} - t_i}{2}, \frac{t_i - t_{i-1}}{2}\right). \quad (6.110)$$

A. Kennzeichnung von PLT-Stellen

Buchstabe	Prozessleittechnische Kategorie
A	Analysis
B	Burner or combustion
D	Density
E	Voltage
F	Flow
G	Distance, length, position
H	Hand or manual or manually initiated operation
I	Current
J	Power
K	Time-based function
L	Level
M	Moisture or humidity
N	Actuation setting (motor)
P	Pressure
Q	Quantity or counter
R	Radiation
S	Speed or frequency
T	Temperature
V	Vibration or mechanical analysis
W	Weight, mass, force
Y	Actuation setting (valve)

Tabelle A.1.: Prozessleittechnische Kategorien nach ISA-88.

Buchstabe	Prozessleittechnische Funktion
A	Alarm, message
B	Restriction
C	Control
D	Difference
F	Ratio
H	High limit, on, opened
I	Indication of analogue values
L	Low limit, off, closed
O	Local or PCS status indication of binary signals
Q	Integrating or counting
R	Recorded value
S	Binary control function or switching function (not safety relevant)
Y	Computing function
Z	Binary control function or switching function (safety relevant)

Tabelle A.2.: Prozessleittechnische Funktion nach ISA-88.

Glossar

K_v -Wert Der K_v -Wert ist eine Kenngröße für den erzielbaren Durchsatz eines Ventils und dient zu deren Dimensionierung. Genauer gibt er den Durchfluss von Wasser in m³/h bei einer Wassertemperatur von 5 °C–30 °C und einer Druckdifferenz von 1 bar an.

Dezentrale Steuerungen Bei einer dezentralen Steuerungsarchitektur ist die Steuerungssoftware monolithisch und läuft auf einer zentralen Steuerungseinheit. Die Sensoren und Aktoren sind jedoch an dezentrale Peripheriekomponenten angebunden, die bereit eine Signalaufbereitung bzw. Vorverarbeitung (z. B. Drahtbruchüberwachung) übernehmen. Die Kommunikation mit der zentralen Steuerungseinheit erfolgt über Feldbusse.

DIN ISO 1219 Grafische Symbole und Schaltpläne der Fluidtechnik sind in der Norm DIN ISO 1219 normiert.

Event Control Chart Eine spezielle Transitionsnotation für IEC 61499 Basic Function Blocks

Event Execution Control Ein abstraktes Modell welches die Zuweisung der eingehenden Events zu den auszuführenden Algorithmen beschreibt

eXtensible Markup Language Eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdateien. XML wird unter anderem für den plattform- und implementationsunabhängigen Austausch von Daten zwischen Computersystemen eingesetzt.

horizontale Kommunikation Die Querkommunikation zwischen Komponenten auf einer Ebene der Automatisierungspyramide wird als horizontale Kommunikation bezeichnet. Unter horizontaler Integration wird die Integration unterschiedlicher Systeme und Komponenten der gleichen Ebene zur Verbesserung des Datenaustausches wird als horizontale Integration verstanden.

IEC 60381 normiert analoge Signale in Regel- und Steuerungsanlagen.

Industrial Ethernet bezeichnet die Verwendung von echtzeitfähigem Ethernet-Adaptionen in der industriellen Fertigung.

Industrie-PC Im prinzipiellen Aufbau ident mit einem Standard-PC. Die verbauten Komponenten sind jedoch speziell auf die rauere Umgebung in Industrieanlagen ausgelegt. Beispielsweise sind die Gehäuse oft staubdicht ausgeführt und auf bewegliche Teil (Lüfter) wird bewusst verzichtet.

kinematische Kette ein durch Gelenke verbundenes System von Starrkörpern.

Netzwerktopologie bezeichnet die Struktur der Verbindungen zwischen Geräten eines Netzwerks wie beispielsweise Bus-, Stern- oder Ringstrukturen.

Remote-I/O bezeichnet die Anbindung von herkömmlichen Sensoren über einen im Feld verbauten D/A-Wandler.

Reynolds-Zahl stellt eine dimensionslose Größe aus der Strömungsmechanik dar. Sie beschreibt das Verhältnis von Trägheits- und viskosen Kräften eines Systems und damit deren Einfluss auf das Strömungsverhalten.

Statistische Prozesslenkung Statistische Prozesslenkung bezeichnet ein statistisches Verfahren, um in Produktionsprozessen ein vordefiniertes Maß an Qualität kostenoptimal einzuhalten.

Stellventil sind kontinuierlich wirkende Ventile mit nur einem Fluidweg, also der Grenzfall eines 2/2-Wegevents.

Teach-In manuelles Einlernen der Bewegungsbahnen des Roboters durch einen Programmierer.

Totally Integrated Automation ist ein von Siemens geprägter Begriff für die durchgängige Integration von Automatisierungslösungen über alle Ebenen der Automatisierungspyramide hinweg.

VDI 2860 standardisiert Montage- und Handhabungstechnik und beinhaltet u. a. eine Definition von Industrierobotern.

Verbindungsprogrammierte Steuerung Bei verbindungsprogrammierten Steuerungen wird das Programm durch mechanische, pneumatische, elektro-pneumatische oder hydraulische, aber auch elektromechanische (Relais) oder elektronische (ICs) Baulemente und deren fixe Verbindung realisiert. Allen Umsetzungsmöglichkeiten ist gemein, dass eine Änderung des Programms schwierig und/oder zeitaufwändig ist.

Verteilte Steuerungen sind in Hard- und Software verteilt ausgeführt.

vertikale Kommunikation Die Kommunikation zwischen Komponenten unterschiedlicher Ebenen der Automatisierungspyramide wird als vertikale Kommunikation bezeichnet.

Wegeventile Unter Wegeventilen versteht man in der Fluidtechnik Ventile mit mehreren einstellbaren Fluidwegen. Die Ventile unterscheiden sich dabei nach Bauart, Anzahl der Schaltstellungen und Durchflusswege und Betätigungsart. Die verwendeten Schaltsymbole sind in DIN ISO 1219 geregelt.

Zentrale Steuerungen Eine zentrale Steuerungsarchitektur im klassischen Sinn ist dadurch gekennzeichnet, dass zentral eine Komponente die Steuerung der gesamten Anlage übernimmt. Die Sensoren und Aktoren sind dabei über analoge Schnittstellen (4 mA–20 mA) mit der Steuerungskomponente verbunden.

Literatur

- [1] Endress+Hauser, *DIN EN IEC 61508/IEC 61511 Funktionale Sicherheit in der Prozess-Instrumentierung zur Risikoreduzierung*, [\protect\t1\textdollarFILE/SIL_Broschuere_de.pdf](http://www.endress.com/eh/productDBs/homeDBs/resourceadditional.nsf/imgref/Download_SIL_Broschuerde_de.pdf).
- [2] N. Stosic, I. Smith und A. Kovacevic, *Screw Compressors: Mathematical Modelling and Performance Calculation*. Springer Verlag, 2005.
- [3] IEC TC65/WG6, *IEC 61131: Standard - Programmable controllers – Parts 1 to 8*. International Electrical Commission, 2003.
- [4] ——, *IEC 61499: Function blocks for industrial-process measurement and control systems – Parts 1 to 4*. Geneva: International Electrotechnical Commission (IEC), 2004-2005.
- [5] B. Favre-Bulle, *Automatisierung komplexer Industrieprozesse - Systeme, Verfahren und Informationsmanagement*. Springer-Verlag, 2004.
- [6] M. Felleisen, *Prozessleittechnik für die Verfahrensindustrie*. Oldenbourg Industrieverlag München, 2001.
- [7] D. Abel, U. Epple und G.-U. Spohr, *Integration von Advanced Control in der Prozessindustrie - Rapid Control Prototyping*. Wiley-VCH Verlag, 2008.