Diss. ETH No. 10385

10. März 1994

Hierarchical Control of Rigid Multi-Body Systems

A dissertation submitted to the

SWISS FEDERAL INSTITUTE OF TECHNOLOGY
ZURICH

for the degree of
Doctor of Technical Sciences

presented by

MARK ANDREAS BUSENHART

Dipl. Masch.-Ing. ETH
born January 24, 1961
citizen of Zurich, ZH

accepted on the recommendation of
Prof. Dr. H.P. Geering, examiner
Prof. Dr. G. Schweitzer, co-examiner

1994

H. P. Geering

Leer - Vide - Empty

To my wife Esther

and to my parents

Leer - Vide - Empty

Acknowledgments

I am obliged to Professor Hans Peter Geering for giving me the opportunity to prepare this thesis and for its supervision. He provided me with an ideal working environment, supported me very generously, and gave me the opportunity to participate in numerous conferences.

I am indebted to Professor Gerhard Schweitzer for accepting the co-examination and for his careful reading of the whole manuscript. His numerous comments gave me valuable feedback.

Brigitte Rohrbach edited the entire manuscript thoroughly several times, mostly in here free time. I am greatly indebted to her for this tremendous effort, her great enthusiasm, and all her constructive criticisms, which helped me improve this document and the standard of my English.

Christoph Keller proved to be a very competent person with whom to discuss my work, because of his quick understanding, and he helped me improve both the quality of this thesis and its language.

I would like to thank my father for all the numerous hours he spent editing the complete manuscript. His very punctual weekly telephone calls during the last year developed into an appreciated and well-liked habit.

Urs Christen made some corrections to Chapters 2 and 3, and we had some very interesting discussions comparing my proposed method to conventional linear control theory.

I am very grateful to all those whose contributions to this work have not been explicitly mentioned above, in particular all the members of the IMRT, who made my stay there a very pleasant one.

Finally, I would like to express my gratitude to my wife Esther, without whose patience and understanding this work would never have been completed. It was thanks to her support that I could concentrate fully on my thesis, which was especially important in the last year of the work. Also, with her well-trained eye, she proved to be an expert defining the best layout and style for the manuscript.

Leer - Vide - Empty

Abstract

A new method for modelling and controlling complex nonlinear systems is developed. It is elaborated for arbitrary, rigid multi-body systems in particular.

First, a formalism is worked out for the description of the kinematics and dynamics of rigid multi-body systems. It can cope with open and closed kinematical chains as well as with unilateral constraints.

The controller possesses a hierarchical structure. The key idea is to define the overall control task by a set of subtasks to be solved separately rather than by specifying a demanded trajectory and applying an explicit control law for guiding the system along that trajectory. These subtasks form the elements of the hierarchical structure. Their interdependencies are given by the interconnections. The main advantage of this controller structure is the resulting flexibility, as there is no upper limit to the number of elements that can be added. Therefore, the controller can be improved step by step until the given control task is solved with the desired accuracy and minimum effort.

Every subtask is defined by a set of goals or rules, i.e., implicit or explicit linear equations, and weighting functions specifying their importance or precision. The solution of a subtask is the decision complying with the rules as closely as possible, weighing their importance. Thus, instead of defining the solution of a subtask explicitly the solution is specified by a non-negative performance index to be minimized. With this procedure, a solution can be found for determined, overdetermined, and underdetermined problems. This is especially useful in connection with singularities and redundancies. Also, the goals can be activated and deactivated smoothly by the corresponding weighting functions. This allows the introduction of a certain "fuzziness" in the modelling or in the control rules in a rather straightforward manner.

The solution of the minimization problem is achieved numerically. The algorithm applied is very robust and well-suited for parallel computation. The fact that it can be optimized off-line for an arbitrary number of processors allows for a substantial reduction of the computational costs.

The evaluation of the performance indices defining the kinematics and dynamics of arbitrary rigid multi-body systems and the off-line minimization of the numerical effort to compute the minima of the performance indices is computerized using a symbol manipulation program.

Finally, two simulation examples for the modelling and control of a cart pole system and a walking robot are given, illustrating the application of the proposed method.

Zusammenfassung

Vorgestellt wird eine neue Methode zur Modellierung und Regelung komplexer, nichtlinearer Systeme, welche speziell für starre Mehrkörpersysteme ausgearbeitet wird.

Zunächst wird ein Formalismus zur Beschreibung der Kinematik und Dynamik starrer Mehrkörpersysteme hergeleitet. Dieser eignet sich sowohl für Systeme mit offenen und geschlossenen kinematischen Ketten als auch für Systeme mit einseitigen Bindungen.

Der Regler besitzt eine hierarchische Struktur. Die Grundidee besteht darin, das Regelungsproblem durch Teilaufgaben zu definieren, die separat gelöst werden können, statt eine Solltrajektorie vorzugeben und das System mittels eines geeigneten, expliziten Regelgesetzes entlang dieser Trajektorie zu führen. Die Teilaufgaben bilden die Elemente der hierarchischen Struktur. Ihre Abhängigkeiten werden durch die Struktur beschrieben. Der Vorteil einer solchen Reglerstruktur liegt in ihrer großen Flexibilität. Diese erlaubt es, den Regler durch Hinzufügen von Elementen schrittweise zu erweitern, bis das Regelungsproblem mit der geforderten Genauigkeit und minimalem Aufwand gelöst wird.

Jede Teilaufgabe wird durch einen Satz von Zielen oder Regeln in Form von expliziten und impliziten linearen Gleichungen und den zugehörigen Gewichtungsfunktionen festgelegt. Die Gewichtungsfunktionen sind ein Maß für die Wichtigkeit oder die Präzision der einzelnen Regeln. Die Lösung einer Teilaufgabe ist die Aktion, welche alle Regeln unter Berücksichtigung ihrer Gewichtungsfunktionen möglichst schwach verletzt. Anstelle einer expliziten Definition werden die Lösungen der Teilaufgaben damit als Minima der zugehörigen Gütekriterien definiert. Auf diese Weise läßt sich sowohl bei bestimmten als auch bei unter- oder überbestimmten Problemstellungen eine Lösung finden. Dies ist ein großer Vorteil im Zusammenhang mit Singularitäten und Redundanzen. Die Gewichtungsfunktionen erlauben ein kontinuierliches An- und Abschalten der einzelnen

Regeln. Damit ist es auf einfache Weise möglich, eine gewisse Unschärfe bei der Modellbildung und bei den Regelgesetzen einzubringen.

Die momentane Lösung einer Teilaufgabe, d.h. das Minimum des zugehörigen Gütekriteriums, wird numerisch berechnet. Der verwendete Algorithmus ist sehr robust und gut geeignet zur parallelen Verarbeitung. Er kann off-line optimiert werden, was zu einer erheblichen Verringerung der numerischen Kosten führt.

Die Ermittlung der Gütekriterien, welche die Kinematik und die Dynamik beliebiger starrer Mehrkörpersysteme beschreiben, sowie die Off-line-Minimierung des numerischen Aufwandes zur Ermittlung der Minima der Gütekriterien wird mit Hilfe eines Computer-Algebra Systems automatisiert.

Die Anwendung der Methode wird anhand von zwei Simulationsbeispielen näher erläutert. Im ersten Beispiel handelt es sich bei der Regelstrecke um ein auf einem Wagen befestigtes Pendel und im zweiten um einen zweibeinigen, gehenden Roboter.

Table of Contents

Leer - Vide - Empty

Preface

There are numerous formulations for the problem of robot control. One was given by [Corby85]:

"Robot should understand a problem, resolve it and apply an appropriate solution starting with incomplete information; in addition robot can use on-line information derived from the environment to modify behavior to attain a set of objectives."

Except for executing the most simple working tasks, today's robot manipulators still are far from being competitive with man. The prime reasons for this are that their sensor equipment is much inferior to man's perceptive capabilities and that they possess a relatively small number of degrees of freedom. Additionally, they depend on a solution trajectory. They are not yet capable of "understanding" a problem and finding a reasonable solution or of reacting to unforeseen events. There are mainly two ways to deal with these problems. One is to enhance the system's versatility by introducing redundancies of all kinds, i.e., a more comprehensive sensor equipment, extra degrees of freedom, or additional actuators. The resulting redundancies are rather natural and highly desirable. Redundant sensor information, for example, provides improved reliability of the input data [Hager90]. Structural redundancies can be applied for a more specific use of the single degrees of freedom [Egelan87], while redundancies in the actuator space may be introduced for safety reasons or to keep away from force and torque limits. The second way of improving a robot's flexibility is to supply the controller with the necessary degrees of freedom by not completely specifying the demanded trajectory.

The main topic of this work is the development of a controller capable of taking advantage of all kinds of redundancies in a well-coordinated way. Additionally, this controller is to be able to find an acceptable solution to a problem, even where this solution is not fully prescribed. This controller is especially devised for arbitrary, rigid multi-body systems.

Figure I gives an overview of the interdependencies of the various chapters.

Chapter 1 begins with a short review of existing control concepts for robot manipulators and lists the main difficulties in robotic control. The specific objective of this work is then described in more detail.

The main problems in the modelling of robot systems stem from the intricacy of the kinematical equations, especially the difficulties in describing rotations in 3-dimensional space. Any redundancies enhance a system's complexity and thereby further increase these difficulties. The problem is not so much the derivation of the dynamical equations, but an appropriate formulation of these equations leading to numerically efficient algorithms. Chapter 2 introduces a mathematical tool called "Geometric Algebra." With this algebra it is possible to give a global one-to-one representation of the kinematics of arbitrary, rigid multi-body systems. The derivation of the kinematical and dynamical equations using this algebra is the content of Chapter 3. These equations are valid for multi-body systems with open and closed kinematical chains and for systems with boundary constraints.

The best controller is always the one that solves a given task with the desired accuracy and the least effort possible. The control algorithm therefore has to offer the necessary flexibility to perfectly "tailor" the controller to the given plant and task. The structure of the controller proposed in this work, its specification, and the numerical evaluation of the control signals are explained in Chapter 4. The application of this concept to mechanical systems is the content of Chapter 5.

The usage of the proposed method is explained for a cart pole system and a biped robot walking in a vertical plane. The simulation results and the numerical costs to evaluate the control vector are given in Chapter 6.

Finally, Chapter 7 completes this work with a summary and some concluding remarks on the proposed method.

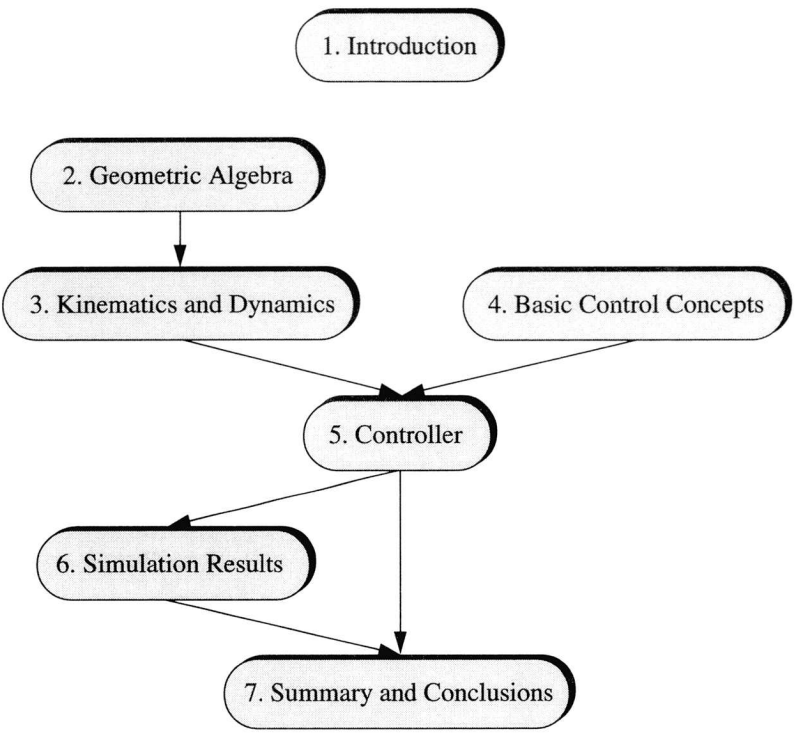


Figure I Overview

Leer - Vide - Empty

List of Symbols

A symbol index can be found in Appendix A

Symbol	*Significance*
α, β	integers specifying body reference frames
θ'_{aj}	rotational velocity vector of the unitary spinor S'_{aj}
τ_a	hinge torque vector (hinge a)
ω	rotational velocity vector of the unitary spinor R
a	integer specifying hinges
a	real number, scalar part of R, or magnitude of $\mathbf{a}$
$\mathbf{a}$	vector in the unprimed reference frame $\{\mathbf{e}_k\}$
$\mathbf{a}'$	vector in the primed reference frame $\{\mathbf{e}'_k\}$
$\hat{\mathbf{a}}$	unit vector of vector $\mathbf{a}$
$\dot{\mathbf{a}}$	first time derivative of vector $\mathbf{a}$
${}^{i}\mathbf{a}$	resultant vector of vectors $\mathbf{a}'_i$ to $\mathbf{a}'_n$
$\underline{\mathbf{a}} = [a_k]$	matrix representation of vector $\mathbf{a}$
$A = \sum_r \mathbf{A}_r$	multivector
$\langle A \rangle_r, \mathbf{A}_r$	r-blade or r-vector part of multivector A
$A^\dagger$	reverse of multivector A
$\lvert A \rvert$	magnitude of multivector A
A^{-1}	multiplicative inverse of multivector A
AB	geometric product of multivectors A and B
$A \cdot B$	inner product of multivectors A and B
$A \wedge B$	outer or wedge product of multivectors A and B
$\mathsf{A} = [a_{jk}]$	matrix

Symbol	*Significance*
A^{+}	Moore-Penrose-Inverse of matrix A
$\underline{\mathsf{A}}^{*}$	sparse representation of matrix A
$c_{\alpha a}, c^{+}_{\alpha a}, c^{-}_{\alpha a}$	connectivity coefficients
C	connectivity matrix
D.O.F.	degrees of freedom
$\{\mathbf{e}_k\}$	orthonormal reference frame, unprimed frame
$\{\mathbf{e}'_k\}$	orthonormal reference frame, primed frame
$\{\mathbf{e}'_{\alpha_k}\}$	orthonormal frame rigidly attached to body α
$\mathcal{E}_3$	3-dimensional Euclidean space
E	unit matrix
f	linear function, linear operator, or linear transformation
$\bar{f}$	transpose or adjoint of f
$\mathbf{f}_a$	hinge force (hinge a)
$\mathbf{F}_\alpha$	resultant force exerted on body α
$\mathbf{F}_{g\alpha}$	gravitational force acting on body α
$\mathcal{G}$	geometric algebra
$\mathcal{G}_3$	geometric algebra of 3-dimensional Euclidean space
i	dextral unit pseudoscalar
I_{α_k}	principal value of the inertia tensor $\mathcal{J}_\alpha$ body α in direction $\mathbf{e}'_{\alpha_k}$
J	Jacobian matrix
$\mathcal{J}_\alpha$	inertia tensor of body α
$J(\mathsf{z}, \underline{\mathsf{y}})$	performance index (node)
$J(\mathrm{p}, \mathrm{r})$	performance index (dyadic reduction)
$\mathbf{l}_\alpha$	angular momentum of body α
m_α	mass of body α

Symbol	*Significance*
M	mass matrix
n	number of hinges
n_a	number of links in hinge a
n_{raj}	number of rotational D.O.F. in link j of hinge a
n_{taj}	number of translational D.O.F. in link j of hinge a
N	number of rigid bodies
(p, r)	dyadic reduction step
$\mathbf{p}$	position vector
$\{\mathbf{p}\}$	position space
$\mathbf{P}_\alpha$	momentum of body α
$\mathcal{R}$	rotation operator of the unitary spinor R
$R = a + i\mathbf{b}$	unitary spinor or quaternion
R_α	quaternion (orientation of $\{\mathbf{e}'_{\alpha_k}\}$ relative to $\{\mathbf{e}_k\}$)
${}^{i}R$	product of spinors R'_i to R'_n
$\underline{\mathsf{R}}$	matrix representation of the unitary spinor R
$\mathsf{R} = [r_{jk}]$	matrix representation of the rotation operator $\mathcal{R}$
$\{R \mid \mathbf{a}\}$	rigid displacement
$\{R_\alpha \mid \mathbf{X}_\alpha\}$	rigid displacement specifying position and orientation of reference frame $\{\mathbf{e}'_{\alpha_k}\}$ with respect to the inertial frame $\{\mathbf{e}_k\}$
$\mathbf{s}'_a$	vector defining the translation from the body reference frame $\{\mathbf{e}'_{\beta_k}\}$ to the body reference frame $\{\mathbf{e}'_{\alpha_k}\}$ of the two bodies connected by hinge a
$\mathbf{s}'_{aj}$	vector defining the translation from reference frame $\{\mathbf{e}'_{aj_k}\}$ to reference frame $\{\mathbf{e}'_{aj+1_k}\}$ of hinge a

Symbol	*Significance*
S'_a	unitary spinor defining the rotation from the body reference frame $\{\mathbf{e}'_{\beta_k}\}$ to the body reference frame $\{\mathbf{e}'_{\alpha_k}\}$ of the two bodies connected by hinge a
S'_{aj}	unitary spinor defining the rotation from reference frame $\{\mathbf{e}'_{aj_k}\}$ to reference frame $\{\mathbf{e}'_{aj+1_k}\}$ of hinge a
$\{S'_a \mid \mathbf{s}'_a\}$	rigid displacement relating the position spaces of the two rigid bodies connected by hinge a
$\{S'_{aj} \mid \mathbf{s}'_{aj}\}$	rigid displacement specifying link j of hinge a
$\mathbf{u}_p$	p*th* control force or torque vector
$\underline{u}$	control vector
U	monic, upper triangular matrix
$\mathbf{X}_\alpha$	position vector of reference frame $\{\mathbf{e}'_{\alpha_k}\}$ with respect to the inertial frame $\{\mathbf{e}_k\}$.
$\underline{y}$	node input vector
$\tilde{\underline{y}}$	node output vector
$\underline{z}$	vector of internal node variables
$\underline{z}_d$	demanded values for vector $\underline{z}$

Chapter 1

Introduction

1.1 Brief Review

Most of the robot manipulators in use today have up to six degrees of freedom. Every joint is actuated by a separate drive and is equipped with sensors measuring the joint positions and velocities.

The most straightforward and wide-spread control concept nowadays is the joint motion control. The demanded trajectory in joint coordinates is computed off-line and the system is guided along this trajectory with independent PID-type controllers for each joint. This results in simple control laws and therefore high sampling rates. This type of controller is well-suited for working tasks like spraying, painting, and welding, as well as simple pick-and-place tasks. However, this concept has several shortcomings.

As the joints are controlled independently, they disturb each other, i.e., the resulting error dynamics are coupled. Moreover, the error dynamics of the manipulator are nonlinear due to the nonlinearities of the manipulator dynamics. This leads to a performance deterioration of the control system, especially for fast movements. Another shortcoming is that the trajectory has to be determined in advance. The controller is thus precluded from reacting to unforeseen events, e.g., to avoid collision with unknown objects.

To improve the dynamical behavior of the controlled system, it is therefore necessary 1) to compensate the nonlinear terms, 2) to find an appropriate linear feedback law, and 3) to calculate the demanded trajectory on-line. With an actuator in each joint, the compensation of the nonlinearities of the plant is always possible. This results in linear error dynamics, which are defined by the linear feedback control law. They can be formulated in joint coordinates or in task coordinates [Markie73], [LuWaPa80b], [Freund82]. Standard approaches to the determination of the linear feedback law are, for example, decoupling and pole placement, or multivariable robust servomechanism theory [DesRot85].

These approaches theoretically solve the motion control problem of robot manipulators. Unfortunately, they presume that the precision of the dynamic model is rather high and that the feedback law and the demanded trajectory are computed sufficiently fast, i.e., with a bandwidth about 10 to 50 times higher than the bandwidth of this system. In reality these are the real annoyances in robot control. There are always effects such as payload variations, friction, or actuator dynamics that cannot be modeled exactly. At the same time, using such complex feedback laws it is, generally, not yet possible to attain the necessary sampling frequencies with the existing hardware.

Several approaches have been in use to bring the on-line computational costs back to a feasible level. While one method compensates only part of the nonlinear dynamics, another one updates the terms for the dynamics compensation at lower rates than the linear feedback terms, as proposed by [Leahy89]. A third approach is to off-line determine the demanded trajectory, linearize the dynamics for an arbitrary number of states, and evaluate the corresponding feedback laws [Seraji87], [Sperli86]. This method is especially suited for repetitive tasks. For arbitrary tasks, however, it quickly ceases to be feasible as the storage requirements for the feedback matrices covering the complete state space tend to become huge.

The drawback of pure motion control is that it is not adequate for contact tasks with stiff contact. The first of the two main reasons is that motion and contact forces depend on each other. It is therefore not possible to realize an arbitrary motion and an arbitrary force in the same direction at the same time, i.e., a compromise is always necessary between the demands from motion control and force control in the same direction. The second reason is that the contact forces depend on the robot's dynamics and on the dynamics of the contact surface. Additionally, the force mechanism is usually relatively stiff, i.e., small changes in the relative motion between the robot manipulator and the contact surface lead to massive changes for the contact forces. This makes it extremely difficult to predict the contact forces with the needed accuracy.

The easiest way to solve these problems is through the introduction of special-purpose mechanical devices ensuring selective compliance. If they are chosen properly, they permit the solution of simple contact tasks with pure motion control. Unfortunately, this is at the price of higher elasticities in the robot manipulator structure.

The alternative to this is to supply the robot manipulator with force sensors and to additionally control these forces. The difficulty arising then is the interdependency of motion and forces. There are mainly two approaches to deal with this:

In the first approach, motion and force are controlled simultaneously in all directions. This can be achieved in numerous ways. The force error can be transformed into a motion error signal in Cartesian coordinates. This error signal is then added to the corresponding error signal from the motion control loop (cf. stiffness control, damping control, impedance control in [Whitne87]). Another technique is to transform the motion error into a force error, i.e., to define the forces necessary to act on the robot system to compensate the actual motional errors, and to add these forces to the measured force error (cf. implicit force feedback in [Whitne87]).

The second approach distinguishes between purely motion controlled directions and purely force controlled directions [Khatib87]. This strategy is especially suitable for tasks with well-defined contact surfaces.

Applying concurrent motion and force control allows even more complex contact tasks like grinding, deburring, or assembly tasks to be executed. However, since the force measurements are given in task coordinates, time-consuming on-line computations for the coordinate transformations cannot be avoided.

For the motion control cycle, this means that we have to evaluate the inverse of the Jacobian (due to inverse kinematics). The matrix inversion may cause additional problems when the mapping between Cartesian space and joint space is not one-to-one, as is the case if we have redundancies or singularities. In the force control loop, however, there is no need for a matrix inversion. As the transformation of the force error in Cartesian space into joint space is given by multiplication with the Jacobian transpose, motion control via the force control loop (implicit force feedback) then becomes a rather attractive solution.

1.2 Main Difficulties

Summing up, a main source of problems in robotic control is the complexity of the kinematical equations. The solution of the forward or inverse kinematics already represents a considerable computational burden. This becomes even worse in the presence of singularities or redundancies where the inverse kinematics, i.e., the mapping between Cartesian space and joint space, is no longer one-to-one. In addition, numerical problems close to singularities may arise.

The dynamics of robot manipulators are even more intricate than the kinematics. Additional problems often occur in connection with the model-

ing of friction, with elasticities in the joints, or changes in the system structure. Such changes in the system structure take place due to boundary constraints, e.g., as the robot gets in contact with the environment.

The trouble with the standard approach to control robotic systems described in Section 1.1, besides the very complex control laws, is its lack of flexibility, due to the predetermined trajectory. This restricts the possibilities of the controller for reacting to unforeseen events, thus making the application of more sophisticated sensory equipment such as vision sensors unavailing.

Another drawback is the somewhat rigid controller structure. The feedback law is given explicitly. Thus, for systems with a changing structure a feedback law has to be found for every possible configuration. For more constrained systems moving in complex environments this is not a feasible approach as, in general, all possible situations for such systems are impossible to be foreseen.

1.3 Objectives

The purpose of this work is the development of a control algorithm suitable for the control of nonlinear systems. This algorithm is especially elaborated for robotic systems, i.e., for arbitrary, rigid multi-body systems with redundancies, open and closed kinematical chains, and boundary constraints. The objective is to obtain a controller capable of solving the given control task with the desired accuracy and minimal effort.

The main elements of the approach selected for the achieving of this objective are

- a hierarchical controller structure,
- a suitable subtask formulation, and
- an efficient and robust computation of the control vector.

One of the major problems in control, generally, is to properly weigh the inefficiency produced by inexact modelling against the inefficiency due to time losses incurred in the computation of more complex models and feedback laws. Hence, it is rather important to have a concept allowing to easily improve the controller complexity. The hierarchical controller structure is chosen to attain this flexibility. Its nature being more general than that of typical controllers allows the definition of the overall task by an arbitrary number of dependent subtasks.

Obviously, a lucid representation of these subtasks and, as a consequence, of the whole control task is of major importance in robot control, as the complexity of the kinematical and dynamical equations describing these systems renders their utilization very difficult.

However, the subtask representation is only one aspect of the approach selected. At the same time it is a prerequisite for the efficient numerical processing of the sensor input to evaluate the control vector. As a result, the minimization of these numerical costs needs to be given serious attention as well.

Chapter 2

Geometric Algebra

This chapter gives a short introduction to the mathematical tool called geometric algebra. Geometric algebra is a unified mathematical language integrating other algebraic systems such as vector algebra, complex numbers, quaternions, and matrix algebra in a coherent way. Also, it is a very efficient and versatile computational tool. The basic motivation for the use of geometric algebra emanates from its great efficiency in representing geometrical relations in 3-dimensional space, both algebraically and numerically. This is of great importance in the modeling of robot kinematics and dynamics. Of course, it would be far beyond the scope of this work to describe the geometric algebra as a whole. The interested reader is therefore referred to [HesSob84] and [Hesten86], where the new ideas and concepts of geometric algebra are developed and explained in a most exciting and fascinating way. Thus, the emphasis here will be on transformations in 3-dimensional Euclidean space.

The axioms of geometric algebra are defined in the beginning of Section 2.1, whereas the end of this section gives some geometrical interpretations for the elements and operations introduced. Since the physical space is a 3-dimensional Euclidean space, Section 2.2 specifies the subalgebra for this space. This subalgebra will be the mathematical tool used. Additionally, some important identities are established. Linear operators and transformations are introduced in Section 2.3. The emphasis is on rotations and rigid displacements. Both are of major importance for the modeling of

kinematical and dynamical relations for arbitrary, rigid multi-body systems. Finally, Section 2.4 covers the matrix representation of the elements of geometric algebra to be used for computations.

2.1 Axioms of Geometric Algebra

An algebra is an abstract mathematical system consisting of a vector space, a multiplication, and some axioms relating this multiplication to vector addition and scalar multiplication. The geometric algebra G uses the same axioms as elementary scalar algebra except for the commutativity of the multiplication. But, in contrast to scalar algebra, the elements of geometric algebra contain much more information. They are called multivectors.

- A multivector A is composed of elements of mixed grade

$$A = \sum_r \langle A \rangle_r = \langle A \rangle_0 + \langle A \rangle_1 + \langle A \rangle_2 + \langle A \rangle_3 + \ldots \tag{2.1}$$

$\langle A \rangle_r$ or $\mathbf{A}_r$ are elements of grade r, also referred to as r-vector part or r-blade of the multivector A. r-blades are directed numbers, representing a magnitude, a direction, and an orientation. They are symbolized with upper case bold letters. The 0-blade or scalar $\langle A \rangle_0 = \mathbf{A}_0 = a$ is the only exception to this, possessing no directional property. Therefore, scalars can be interpreted as real numbers. They will usually be denoted by lower case, italic letters. A 1-blade $\langle A \rangle_1 = \mathbf{A}_1 = \mathbf{a}$ is what we conventionally call a vector, describing length and direction of a line. It will normally be represented by lower case, bold letters. A bivector $\langle A \rangle_2 = \mathbf{A}_2$ and a trivector $\langle A \rangle_3 = \mathbf{A}_3$ characterize magnitude and directional properties of a plane and of 3-dimensional space, respectively.

Geometric algebra is defined by the subsequent set of axioms:

- The algebra is closed, i.e., the sum and the product of two multivectors yield multivectors.
- Addition is commutative

$$A + B = B + A. \tag{2.2}$$

- Addition and multiplication are associative, i.e.,

$$(A + B) + C = A + (B + C) \text{ and} \tag{2.3}$$

$$(AB)\,C = A\,(BC)\,. \tag{2.4}$$

- Multiplication is left and right distributive with respect to addition

$$A\,(B + C) = AB + AC \text{ and} \tag{2.5}$$

$$(B + C)A = BA + CA. \tag{2.6}$$

- There exist unique additive and multiplicative identities

$$A + 0 = A \text{ and} \tag{2.7}$$

$$1A = A. \tag{2.8}$$

- There is a unique additive inverse:

$$A + (-A) = 0. \tag{2.9}$$

- All r-blades form a linear subspace, i.e.,

$$\langle A + B \rangle_r = \langle A \rangle_r + \langle B \rangle_r \text{ and} \tag{2.10}$$

$$\langle aA \rangle_r = a\langle A \rangle_r = \langle A \rangle_r a \qquad \text{if } a = \langle a \rangle_0 \tag{2.11}$$

- The square of a non-zero vector **a** is a scalar value (Euclidean Axiom)

$$\mathbf{aa} = \mathbf{a}^2 = \langle \mathbf{a}^2 \rangle_0 = |\mathbf{a}|^2 > 0. \tag{2.12}$$

- r-blades can always be factored in the following manner

$$\mathbf{A}_r = \mathbf{a}_1 \mathbf{a}_2 \ldots \mathbf{a}_r \qquad \text{for } r > 0, \text{ with} \tag{2.13}$$

$$\mathbf{a}_j \mathbf{a}_k = -\mathbf{a}_k \mathbf{a}_j = -(\mathbf{a}_j \mathbf{a}_k)^\dagger. \tag{2.14}$$

In equation (2.14) the operator of *reversion*, †, is introduced. This operator is a very helpful tool to reorder factors in algebraic manipulations. It has the following properties:

$$(AB)^\dagger = B^\dagger A^\dagger, \tag{2.15}$$

$$(A+B)^\dagger = A^\dagger + B^\dagger, \tag{2.16}$$

$$\langle A^\dagger \rangle_0 = \langle A \rangle_0, \text{ and} \tag{2.17}$$

$$\langle A^\dagger \rangle_1 = \langle A \rangle_1. \tag{2.18}$$

There are three types of multiplications, called inner, outer, and geometric product.

- The inner product is defined by

$$A \cdot B = \sum_r \sum_s \mathbf{A}_r \cdot \mathbf{B}_s, \text{ with} \tag{2.19}$$

$$\mathbf{A}_r \cdot \mathbf{B}_s = \begin{cases} \langle \mathbf{A}_r \mathbf{B}_s \rangle_{|r-s|} & \text{if } r, s > 0, \\ 0 & \text{if } r = 0 \text{ or } s = 0. \end{cases} \tag{2.20}$$

- The outer, or wedge product, is given as

$$A \wedge B = \sum_r \sum_s \mathbf{A}_r \wedge \mathbf{B}_s, \quad \text{with} \tag{2.21}$$

$$\mathbf{A}_r \wedge \mathbf{B}_s = \langle \mathbf{A}_r \mathbf{B}_s \rangle_{r+s}. \tag{2.22}$$

The inner and the outer products both describe relative directional properties of their factors. The inner product is a grade-lowering operation, whereas the outer product raises the grade. Inner and outer products complement each other, specifying independent directional properties. This will be explained in more detail at the end of this section. Thus, the complete description of the directional relations between different spaces is only given by the geometric product of the corresponding blades, because the geometric product merges the inner and the outer product.

- The geometric product is defined by

$$AB = \sum_r \sum_s \mathbf{A}_r \mathbf{B}_s, \quad \text{with} \tag{2.23}$$

$$\mathbf{A}_r \mathbf{B}_s = \sum_{k=0}^{m} \langle \mathbf{A}_r \mathbf{B}_s \rangle_{|r-s|+2k} \qquad m = \frac{r+s-|r-s|}{2}. \tag{2.24}$$

- The inner and the outer products are united in the geometric product. This is evident for the product of a vector with an r-blade (cf. Appendix B4):

$$\mathbf{a}\mathbf{B}_r = \mathbf{a} \cdot \mathbf{B}_r + \mathbf{a} \wedge \mathbf{B}_r, \quad \text{with} \tag{2.25}$$

$$\mathbf{a} \cdot \mathbf{B}_r = \langle \mathbf{a}\mathbf{B}_r \rangle_{r-1} = \tfrac{1}{2}(\mathbf{a}\mathbf{B}_r - (-1)^r \mathbf{B}_r \mathbf{a}) \quad \text{and} \tag{2.26}$$

$$\mathbf{a} \wedge \mathbf{B}_r = \langle \mathbf{a}\mathbf{B}_r \rangle_{r+1} = \tfrac{1}{2}(\mathbf{a}\mathbf{B}_r + (-1)^r \mathbf{B}_r \mathbf{a}). \tag{2.27}$$

To avoid too many parentheses, it is convenient to introduce the following precedence conventions:

- inner and outer products before geometrical products and
- outer products before inner products

The *magnitude* $|A|$ of an arbitrary multivector A is now defined as

$$|A| = \langle A^\dagger A \rangle_0^{1/2}. \tag{2.28}$$

The *multiplicative inverse* A^{-1} of a multivector A satisfies the following equation:

$$A^{-1}A = 1. \tag{2.29}$$

The inverse of an r-blade can *always* be evaluated. In Appendix B3 the following relation is established:

$$\langle \mathbf{A}_r^\dagger \mathbf{A}_r \rangle_0 = \mathbf{A}_r^\dagger \mathbf{A}_r. \tag{2.30}$$

Additionally, using equations (2.28) and (2.29) the subsequent transformations can be performed:

$$|\mathbf{A}_r|^2 \mathbf{A}_r^{-1} = \langle \mathbf{A}_r^\dagger \mathbf{A}_r \rangle_0 \mathbf{A}_r^{-1} = (\mathbf{A}_r^\dagger \mathbf{A}_r)\, \mathbf{A}_r^{-1} = \mathbf{A}_r^\dagger (\mathbf{A}_r \mathbf{A}_r^{-1}) = \mathbf{A}_r^\dagger. \tag{2.31}$$

Thus, the inverse of the r-blade $\mathbf{A}_r$ is always given by

$$\mathbf{A}_r^{-1} = \frac{\mathbf{A}_r}{|\mathbf{A}_r|^2}. \tag{2.32}$$

So far, geometric algebra has been established purely mathematically. As already mentioned in the introduction, it is a mathematical tool to describe geometrical relations in an efficient manner. But, to use this tool appropriately, it is necessary to understand its elements and their geometrical meanings.

We start with the relative directional relations of two vectors described by the inner and the outer products, as mentioned in Section 2.1. The equation $\mathbf{a} \cdot \mathbf{b} = 0$ is a simple way of expressing that two vectors are orthogonal. The inner product (2.19) is a grade-lowering operation. Thus, the inner product of two vectors $\mathbf{a}$ and $\mathbf{b}$ is a scalar quantity. Geometrically, $\mathbf{a} \cdot \mathbf{b}$ can be interpreted as the perpendicular projection of $\mathbf{a}$ on the unit vector $\hat{\mathbf{b}}$ of $\mathbf{b}$, dilated by the magnitude $|\mathbf{a}|$ of $\mathbf{a}$ (cf. Figure 2.1). As the perpendicular projection

$$\mathbf{a} \cdot \hat{\mathbf{b}} = |\mathbf{a}| \cos\varphi \tag{2.33}$$

depends on the angle φ between both vectors, the inner product does so as well. But, in contrast to the perpendicular projection, it depends on the magnitudes of both vectors, having the following important symmetric property as a consequence

$$\begin{aligned} \mathbf{a} \cdot \mathbf{b} &= \mathbf{a} \cdot \hat{\mathbf{b}}\, |\mathbf{b}| = |\mathbf{a}| \cos\varphi\, |\mathbf{b}| \\ &= |\mathbf{b}| \cos\varphi\, |\mathbf{a}| = \mathbf{b} \cdot \hat{\mathbf{a}}\, |\mathbf{a}| = \mathbf{b} \cdot \mathbf{a} \,. \end{aligned} \tag{2.34}$$

Summarizing we can say that the inner product relates scalar quantities as angles and vector magnitudes to vectors. But the inner product fails to specify the plane defined by two vectors if they are not parallel.

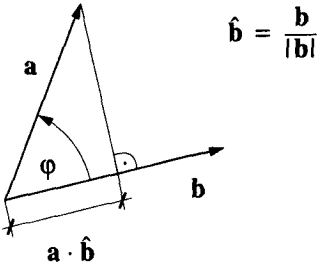


Figure 2.1 Inner Product of two Vectors

The outer product (2.21), as a grade-raising operation, bridges this gap, relating vectors to bivectors. The bivector $\mathbf{B} = \langle B \rangle_2 = \mathbf{a} \wedge \mathbf{b}$ represents the plane defined by the two vectors $\mathbf{a}$ and $\mathbf{b}$, its magnitude, corresponding to the area of the parallelogram specified by these vectors, and its orientation (cf. Figure 2.2). Thus, the statement $\mathbf{a} \wedge \mathbf{b} = 0$ implies that the two vectors are parallel. The magnitude of this bivector can be given as $|\mathbf{B}| = |\mathbf{a}|\,|\mathbf{b}| \sin\varphi$. Additionally, it can be seen that a change in the order of succession of the vectors results in a plane with the same magnitude and direction, but with opposite orientation. The bivector $\mathbf{b} \wedge \mathbf{a} = -(\mathbf{a} \wedge \mathbf{b}) = -\mathbf{B}$, e.g., is oriented in opposite direction to $\mathbf{B}$, i.e., clockwise (cf. Figure 2.2.)

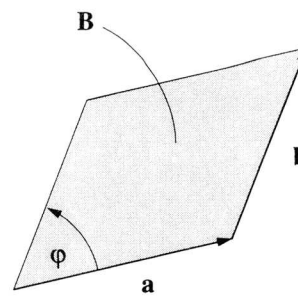


Figure 2.2 Outer Product of two Vectors

Example 2.1

The components of vector $\mathbf{a}$ parallel and orthogonal to vector $\hat{\mathbf{b}}$, $\mathbf{a}_\parallel$ and $\mathbf{a}_\perp$, can be expressed most simply with the inner and outer products:

$$\begin{aligned} \mathbf{a} &= \hat{\mathbf{b}}\hat{\mathbf{b}}\mathbf{a} = \hat{\mathbf{b}}\,(\hat{\mathbf{b}}\mathbf{a}) = \hat{\mathbf{b}}\,(\hat{\mathbf{b}} \cdot \mathbf{a} + \hat{\mathbf{b}} \wedge \mathbf{a}) \\ &= \hat{\mathbf{b}}\,(\hat{\mathbf{b}} \cdot \mathbf{a}) + \hat{\mathbf{b}}\,(\hat{\mathbf{b}} \wedge \mathbf{a}) = \mathbf{a}_\parallel + \mathbf{a}_\perp . \end{aligned} \tag{2.35}$$

The outer product of a vector $\mathbf{a}$ and a bivector $\mathbf{B} = \mathbf{b} \wedge \mathbf{c}$ is a trivector $\mathbf{T} = \langle T \rangle_3$, with

$$\mathbf{T} = \mathbf{a} \wedge \mathbf{B} = \mathbf{a} \wedge (\mathbf{b} \wedge \mathbf{c}) = (\mathbf{a} \wedge \mathbf{b}) \wedge \mathbf{c} = \mathbf{a} \wedge \mathbf{b} \wedge \mathbf{c} . \tag{2.36}$$

Geometrically, a trivector characterizes a parallelepiped (cf. Figure 2.3), the magnitude $|\mathbf{T}|$ now standing for the volume of this parallelepiped. Therefore, the condition that the three vectors $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ lie on the same plane is most simply formulated with $\mathbf{a} \wedge \mathbf{b} \wedge \mathbf{c} = 0$. This concludes the geometrical interpretation of the elements to be used in 3-dimensional Euclidean space.

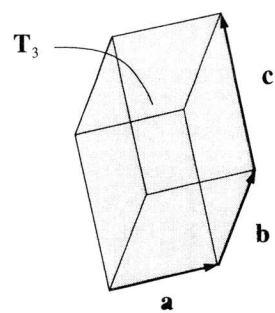


Figure 2.3 Outer Product of Vector and Bivector

2.2 Algebra of Euclidean 3-Space

The physical space is a 3-dimensional Euclidean space. Therefore, it is reasonable to specialize and restrict the geometric algebra $\mathcal{G}$ to the geometric algebra of the 3-dimensional Euclidean space $\mathcal{E}_3$, denoted with $\mathcal{G}_3$.

We first introduce an *orthonormal reference frame* or basis $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ defined by

$$\mathbf{e}_j\mathbf{e}_k = \begin{cases} \mathbf{e}_j \wedge \mathbf{e}_k = -\mathbf{e}_k\mathbf{e}_j & \text{for } j \neq k, \\ 1 & \text{for } j = k. \end{cases} \tag{2.37}$$

In short notation we will write $\{\mathbf{e}_k\}$ for $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$. To describe the set of all elements in $\mathcal{E}_3$ and their orientation, the *dextral unit pseudoscalar* i of $\mathcal{E}_3$ is defined, additionally,

$$i = \mathbf{e}_1\mathbf{e}_2\mathbf{e}_3 = \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3, \tag{2.38}$$

dextral meaning that $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$ form a right-handed reference frame.

The 3-dimensional Euclidean space can now be defined simply as the set of all vectors $\mathbf{x}$ satisfying the equation

$$\mathbf{x} \wedge i = 0. \tag{2.39}$$

The consequence of this is that

$$\mathbf{x}i = \mathbf{x} \cdot i + \mathbf{x} \wedge i = \mathbf{x} \cdot i = i \cdot \mathbf{x} = i\mathbf{x}. \tag{2.40}$$

The properties of i can be given as follows:

$$i^\dagger = -i, \tag{2.41}$$

$$i^2 = -1, \tag{2.42}$$

$$iA = Ai \qquad \text{for every } A \text{ in } \mathcal{G}_3, \text{ and} \tag{2.43}$$

$$\mathbf{a} \wedge \mathbf{b} \wedge \mathbf{c} = \lambda i \qquad \text{for } \mathbf{a}, \mathbf{b}, \mathbf{c} \text{ in } \mathcal{G}_3. \tag{2.44}$$

The proofs for these equations are given in Appendix C. From equation (2.43) we can see that i behaves like a scalar in $\mathcal{G}_3$ (cf. (2.11)), although it is a trivector, which justifies the name pseudoscalar.

The product iA of the unit pseudoscalar i and an arbitrary multivector A is denoted as the *dual* of A. The relation between vector algebra and geometric algebra is defined by the duality relation between the cross product and the outer product

$$\mathbf{a} \wedge \mathbf{b} = i\mathbf{a} \times \mathbf{b}. \tag{2.45}$$

Example 2.2

Geometric product of two multivectors composed of a scalar and a bivector part. Axioms and equations employed: (2.5), (2.11), (2.42), (2.25), and (2.45). Note that the result is again a multivector with the same structure (scalar and bivector part).

$$\begin{aligned}(a_1 + i\mathbf{b}_1)\,(a_2 + i\mathbf{b}_2) &= a_1 a_2 + i\,(a_1\mathbf{b}_2 + a_2\mathbf{b}_1) + i^2\mathbf{b}_1\mathbf{b}_2 \\ &= a_1 a_2 + i\,(a_1\mathbf{b}_2 + a_2\mathbf{b}_1) - (\mathbf{b}_1 \cdot \mathbf{b}_2 + \mathbf{b}_1 \wedge \mathbf{b}_2) \\ &= a_1 a_2 - \mathbf{b}_1 \cdot \mathbf{b}_2 + i\,(a_1\mathbf{b}_2 + a_2\mathbf{b}_1 - \mathbf{b}_1 \times \mathbf{b}_2)\,.\end{aligned} \tag{2.46}$$

Example 2.3

Proof of the relation $(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c} = \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})$. Axioms and equations employed: (2.4), (2.45), (2.27), (2.43),and (2.26)

$$\begin{aligned}\langle (\mathbf{ab})\,\mathbf{c} \rangle_3 &= \langle \mathbf{a}\,(\mathbf{bc}) \rangle_3 \\ (\mathbf{a} \wedge \mathbf{b}) \wedge \mathbf{c} &= \mathbf{a} \wedge (\mathbf{b} \wedge \mathbf{c}) \\ (i\mathbf{a} \times \mathbf{b}) \wedge \mathbf{c} &= \mathbf{a} \wedge (i\mathbf{b} \times \mathbf{c}) \\ \tfrac{1}{2}\,((i\mathbf{a} \times \mathbf{b})\,\mathbf{c} + \mathbf{c}\,(i\mathbf{a} \times \mathbf{b})) &= \tfrac{1}{2}\,(\mathbf{a}\,(i\mathbf{b} \times \mathbf{c}) + (i\mathbf{b} \times \mathbf{c})\,\mathbf{a}) \\ \tfrac{1}{2}\,i\,((\mathbf{a} \times \mathbf{b})\,\mathbf{c} + \mathbf{c}\,(\mathbf{a} \times \mathbf{b})) &= \tfrac{1}{2}\,i(\mathbf{a}\,(\mathbf{b} \times \mathbf{c}) + (\mathbf{b} \times \mathbf{c})\,\mathbf{a}) \\ i\,(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c} &= i\,\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})\,.\end{aligned} \tag{2.47}$$

Example 2.4

Proof of the relation $\mathbf{a} \cdot (\mathbf{b} \wedge \mathbf{c}) = \mathbf{a} \cdot \mathbf{bc} - \mathbf{a} \cdot \mathbf{cb}$. Axioms and equations employed: (2.25), (2.26), and (2.27)

$$\begin{aligned}\mathbf{abc} &= (\mathbf{ab})\,\mathbf{c} = (2\mathbf{a} \cdot \mathbf{b} - \mathbf{ba})\,\mathbf{c} = 2\mathbf{a} \cdot \mathbf{bc} - \mathbf{b}\,(\mathbf{ac}) \\ &= 2\mathbf{a} \cdot \mathbf{bc} - \mathbf{b}\,(2\mathbf{a} \cdot \mathbf{c} - \mathbf{ca}) = 2\,(\mathbf{a} \cdot \mathbf{bc} - \mathbf{a} \cdot \mathbf{cb}) + \mathbf{bca}.\end{aligned} \tag{2.48}$$

Reordering of (2.48) yields

$$\tfrac{1}{2}(\mathbf{abc}-\mathbf{bca}) = \mathbf{a}\cdot\mathbf{bc}-\mathbf{a}\cdot\mathbf{cb}. \tag{2.49}$$

On the other hand we can write

$$\begin{aligned}\tfrac{1}{2}(\mathbf{abc}-\mathbf{bca}) &= \tfrac{1}{2}(\mathbf{a}(\mathbf{bc})-(\mathbf{bc})\mathbf{a})\\ &= \tfrac{1}{2}(\mathbf{a}(\mathbf{b}\cdot\mathbf{c}+\mathbf{b}\wedge\mathbf{c})-(\mathbf{b}\cdot\mathbf{c}+\mathbf{b}\wedge\mathbf{c})\mathbf{a})\\ &= \tfrac{1}{2}(\mathbf{ab}\wedge\mathbf{c}-\mathbf{b}\wedge\mathbf{ca}) = \mathbf{a}\cdot(\mathbf{b}\wedge\mathbf{c}).\end{aligned} \tag{2.50}$$

Finally, equating (2.49) and (2.50) the proof is established.

2.3 Linear Operators and Transformations

A *linear function* f, also named *linear operator* or *linear transformation*, is defined by

$$f(\mathbf{x}+\mathbf{y}) = f(\mathbf{x})+f(\mathbf{y}) \tag{2.51}$$

$$f(a\mathbf{x}) = af(\mathbf{x}) \tag{2.52}$$

Linear functions will be written in script type.

Example 2.5

Inner and outer product are both linear functions. This can be seen from (2.5) and (2.11), with

$$\begin{aligned}f(\mathbf{x}+\mathbf{y}) &= \mathbf{a}(\mathbf{x}+\mathbf{y}) = \mathbf{a}\cdot(\mathbf{x}+\mathbf{y})+\mathbf{a}\wedge(\mathbf{x}+\mathbf{y})\\ &= \mathbf{ax}+\mathbf{ay} = \mathbf{a}\cdot\mathbf{x}+\mathbf{a}\cdot\mathbf{y}+\mathbf{a}\wedge\mathbf{x}+\mathbf{a}\wedge\mathbf{y}\end{aligned} \tag{2.53}$$

and

$$\begin{aligned} f(b\mathbf{x}) &= \mathbf{a}(b\mathbf{x}) = \mathbf{a}\cdot(b\mathbf{x}) + \mathbf{a}\wedge(b\mathbf{x}) \\ &= ba\mathbf{x} = b\mathbf{a}\cdot\mathbf{x} + b\mathbf{a}\wedge\mathbf{x}. \end{aligned} \tag{2.54}$$

Properties of linear operators:

- product and sum are again linear operators,
- linear operators are associative

$$f(gh) = (fg)h, \tag{2.55}$$

- linear operators are distributive with respect to addition

$$f(g+h) = fg+fh, \text{ and} \tag{2.56}$$

- scalar multiplication is commutative

$$af = fa. \tag{2.57}$$

Additionally, the *transpose* or *adjoint* $\bar{f}$ of a linear operator f in $\mathcal{E}_3$ is defined as the unique linear operator in $\mathcal{E}_3$ satisfying the relation

$$\mathbf{x}\cdot(f\mathbf{y}) = (\bar{f}\mathbf{x})\cdot\mathbf{y} \tag{2.58}$$

for every $\mathbf{x}, \mathbf{y}$ in $\mathcal{E}_3$.

2.3.1 Rotations

Rotations are linear, orthogonal transformations. For orthogonal transformations the inner product is invariant. Introducing the rotation operator $\mathcal{R}$ we can write

$$(\mathcal{R}\mathbf{x})\cdot(\mathcal{R}\mathbf{y}) = \mathbf{x}\cdot\mathbf{y} = \mathbf{x}\cdot(\bar{\mathcal{R}}\mathcal{R}\mathbf{y}). \tag{2.59}$$

It is obvious from (2.59) that the inverse is equal to the adjoint for the rotation operator, implying that this inverse is a *non-singular* operator and, therefore, always exists. Rotations in $\mathcal{E}_3$ are represented in the following canonical form:

$$\mathcal{R}\mathbf{r}' = R^\dagger \mathbf{r}' R. \tag{2.60}$$

R is said to be a spinor or quaternion. A spinor is a multivector possessing a scalar and a bivector part. The magnitude of spinor R is 1, i.e.,

$$R^\dagger R = 1. \tag{2.61}$$

Therefore, R called a *unitary spinor.* It is usually written as

$$R = a + i\mathbf{b} = a + ib\hat{\mathbf{b}} \tag{2.62}$$

or

$$R = \cos\tfrac{1}{2}\varphi + i\sin\tfrac{1}{2}\varphi\,\hat{\mathbf{b}}. \tag{2.63}$$

To explain the meaning of these parameters, the transformation

$$\mathbf{r} = R^\dagger \mathbf{r}' R = \mathcal{R}\mathbf{r}' \tag{2.64}$$

is depicted in Figure 2.4. It can be seen that the unit vector $\hat{\mathbf{b}}$ defines the rotation axis, while the dual of $\hat{\mathbf{b}}$, $i\hat{\mathbf{b}}$, characterizes the plane of rotation. The rotation angle is given by φ (right-hand rule!) or the parameters a, b. The relation between a, b, and φ is defined by (2.62) and (2.63). The reason for having two parameters a, b in one case and only one parameter φ in the other case is that the parameters a and b have to fulfill (2.61), additionally. Note that (2.61) is always true for (2.63). Thus, there is only one independent parameter in either case. $\mathbf{r}'$ can be interpreted as a vector rigidly attached to a rotating reference frame called *primed system* $\{\mathbf{e}'_k\}$, i.e.,

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{r}' = \dot{\mathbf{r}}' = 0. \tag{2.65}$$

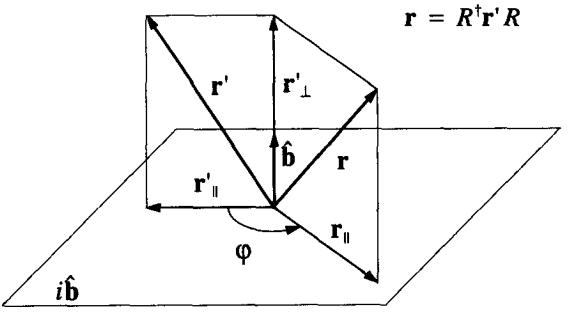


Figure 2.4 Rotation of Vector $\mathbf{r}'$

$\mathbf{r}$ represents the corresponding vector after the rotation and in the non-moving or unprimed system $\{\mathbf{e}_k\}$. The relation between the primed and the unprimed reference frame can now be evaluated as follows. Assume that the primed and unprimed reference frames are identical before the transformation, i.e.,

$$\mathbf{r}' = \sum_k r'_k \mathbf{e}_k . \tag{2.66}$$

Next, assume that the primed system is rotated, resulting in a new vector $\mathbf{r}$ or, in mathematical terms,

$$\begin{aligned} \mathbf{r} &= \sum_k r_k \mathbf{e}_k = \mathcal{R}\mathbf{r}' = \mathcal{R}\sum_k r'_k \mathbf{e}_k \\ &= \sum_k r'_k \mathcal{R}\mathbf{e}_k = \sum_k r'_k \mathbf{e}'_k \end{aligned} . \tag{2.67}$$

Thus, the primed system is defined as

$$\mathbf{e}'_k = \mathcal{R}\mathbf{e}_k \tag{2.68}$$

with respect to the unprimed system. From now on, the term primed system will always stand for a moving reference frame, whereas the unprimed system denotes the non-moving system.

In Example 2.2 it was shown that the product of two quaternions is again a quaternion. Therefore, two arbitrary, consecutive rotations can always be replaced by a single rotation, i.e.,

$$\mathcal{R}\mathbf{r}' = \mathcal{R}_2\mathcal{R}_1\,\mathbf{r}' = R_2^\dagger R_1^\dagger\,\mathbf{r}'R_1R_2 = R^\dagger\mathbf{r}'R \tag{2.69}$$

with

$$\begin{aligned} R &= R_1R_2 = \langle R_1R_2\rangle_0 + \langle R_1R_2\rangle_2 \\ &= a_1a_2 - \mathbf{b}_1\cdot\mathbf{b}_2 + i\,(a_1\mathbf{b}_2 + a_2\mathbf{b}_1 - \mathbf{b}_1\times\mathbf{b}_2)\,. \end{aligned} \tag{2.70}$$

The first time derivative of $\mathbf{r}$ can be specified in the following manner: With (2.64) and (2.65) it can be written as

$$\begin{aligned} \dot{\mathbf{r}} = \frac{\mathrm{d}}{\mathrm{d}t}(\mathcal{R}\mathbf{r}') &= \dot{\mathcal{R}}\mathbf{r}' + \mathcal{R}\dot{\mathbf{r}}' = \frac{\mathrm{d}}{\mathrm{d}t}(R^\dagger\mathbf{r}'R) + 0 \\ &= \dot{R}^\dagger\mathbf{r}'R + R^\dagger\mathbf{r}'\dot{R}. \end{aligned} \tag{2.71}$$

The magnitude r of vector $\mathbf{r}$ can also be expressed as

$$\mathbf{r}^\dagger\mathbf{r} = \mathbf{r}\mathbf{r} = \mathbf{r}\cdot\mathbf{r} = r^2. \tag{2.72}$$

Therefore,

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{r}\mathbf{r}) = \dot{\mathbf{r}}\mathbf{r} + \mathbf{r}\dot{\mathbf{r}} = 0. \tag{2.73}$$

This implies that $\mathbf{r}\dot{\mathbf{r}}$ is a bivector, i.e.,

$$\mathbf{r}\dot{\mathbf{r}} = -\dot{\mathbf{r}}\mathbf{r} = \mathbf{r}\wedge\dot{\mathbf{r}}. \tag{2.74}$$

Left multiplication with $\mathbf{r}^{-1}$ and substitution of $\mathbf{r}^{-1}\wedge\dot{\mathbf{r}}$ by $i\omega$ result in

$$\begin{aligned}\dot{\mathbf{r}} &= \mathbf{r}^{-1}(\mathbf{r}\wedge\dot{\mathbf{r}}) = \mathbf{r}(\mathbf{r}^{-1}\wedge\dot{\mathbf{r}}) = \mathbf{r}\cdot(\mathbf{r}^{-1}\wedge\dot{\mathbf{r}})\\ &= \mathbf{r}\cdot(i\omega) = i(\mathbf{r}\wedge\omega) = \omega\times\mathbf{r}.\end{aligned} \tag{2.75}$$

The vector ω is called *rotational velocity* of the unitary spinor R. Equation (2.75) can also be written as

$$\begin{aligned}\dot{\mathbf{r}} &= \mathbf{r}\cdot(i\omega) = \tfrac{1}{2}(\mathbf{r}i\omega - i\omega\mathbf{r})\\ &= \tfrac{1}{2}(R^{\dagger}\mathbf{r}'Ri\omega - i\omega R^{\dagger}\mathbf{r}'R).\end{aligned} \tag{2.76}$$

Thus, by combination of (2.71), (2.75), and (2.76) the following two important relations are deduced:

$$\dot{\mathcal{R}}\mathbf{r}' = \omega\times\mathcal{R}\mathbf{r}' \tag{2.77}$$

and

$$\dot{R} = \tfrac{1}{2}Ri\omega. \tag{2.78}$$

Example 2.6

Evaluation of ω in component form. From (2.61) we know that

$$\frac{\mathrm{d}}{\mathrm{d}t}(R^{\dagger}R) = 0. \tag{2.79}$$

Using (2.62) we can thus write

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}(a^2+\mathbf{b}^2) &= 2a\dot{a}+\dot{\mathbf{b}}\mathbf{b}+\mathbf{b}\dot{\mathbf{b}}\\ &= 2a\dot{a}+\dot{\mathbf{b}}\cdot\mathbf{b}+\dot{\mathbf{b}}\wedge\mathbf{b}+\mathbf{b}\cdot\dot{\mathbf{b}}+\mathbf{b}\wedge\dot{\mathbf{b}}\\ &= 2a\dot{a}+2\mathbf{b}\cdot\dot{\mathbf{b}}+\dot{\mathbf{b}}\wedge\mathbf{b}-\dot{\mathbf{b}}\wedge\mathbf{b}\\ &= 2a\dot{a}+2\mathbf{b}\cdot\dot{\mathbf{b}} = 0.\end{aligned} \tag{2.80}$$

Left multiplication of (2.78) with $-2i\,R^{\dagger}$ combined with relation (2.80) then result in the following expression for ω:

$$\begin{aligned}\omega &= -2iR^{\dagger}\dot{R} \\ &= -2i\,(a+i\mathbf{b})^{\dagger}\,(\dot{a}+i\dot{\mathbf{b}}) \\ &= -2i\,(\,(a\dot{a}+\mathbf{b}\cdot\dot{\mathbf{b}})+i\,(a\dot{\mathbf{b}}-\dot{a}\mathbf{b}+\mathbf{b}\times\dot{\mathbf{b}})\,) \\ &= 2\,(a\dot{\mathbf{b}}-\dot{a}\mathbf{b}+\mathbf{b}\times\dot{\mathbf{b}})\,.\end{aligned} \tag{2.81}$$

2.3.2 Rigid Displacements

The position of a particle P is given by its *position vector* $\mathbf{p}$. This vector describes the position relative to some rigid body or reference frame. The set of all possible positions forms the *position space* $\{\mathbf{p}\}$. The position spaces of two reference frames are related by a transformation called *rigid displacement*. A rigid displacement is an *isometry*, i.e., a point-to-point transformation leaving the distances between two points unchanged. Thus, it is well suited to describe the motions of all particles of a rigid body. A rigid displacement is the superposition of a rotation and a translation and is denoted with $\{R \mid \mathbf{a}\}$. The unitary spinor R characterizes the rotation, whereas vector $\mathbf{a}$ defines the translational motion. The relation between $\mathbf{p}$ and $\mathbf{p}'$ (cf. Figure 2.5), i.e., the position of the same particle in the unprimed and the primed reference frames, is given by

$$\mathbf{p} = \{R \mid \mathbf{a}\}\,\mathbf{p}' = R^{\dagger}\mathbf{p}'R+\mathbf{a} = \mathcal{R}\mathbf{p}'+\mathbf{a}. \tag{2.82}$$

Symbolically, the rigid displacement in (2.82) is represented as shown in Figure 2.6. In case of more than one primed reference frame, the different primed systems are going to be distinguished by indices, the indices on the baseline specifying the reference frames and the subscript indices being used for numbering.

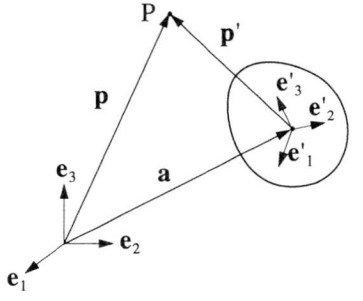


Figure 2.5 Position Spaces $\{\mathbf{p}\}$ *and* $\{\mathbf{p}'\}$

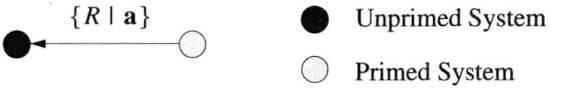


Figure 2.6 Rigid Displacement

Example 2.7

Vector $\mathbf{x}'_1$ denotes a vector from the primed system $\{\mathbf{e}'_{1_k}\}$, and $\mathbf{x}'_{aj_i}$ characterizes the *ith* vector given in $\{\mathbf{e}'_{aj_k}\}$, with

$$\mathbf{e}'_{aj_k} = \mathcal{R}_{aj}\mathbf{e}_k \qquad \text{for } k = 1, 2, 3 . \tag{2.83}$$

The rigid displacement caused by two consecutive rigid displacements is defined as follows:

$$\begin{aligned}\mathbf{p} &= \{R'_2 \mid \mathbf{a}'_2\}\,\{R'_1 \mid \mathbf{a}'_1\}\,\mathbf{p}' \\ &= \{R'_2 \mid \mathbf{a}'_2\}\,R'^{\dagger}_1\mathbf{p}'R'_1 + \mathbf{a}'_1 \\ &= R'^{\dagger}_2\,(R'^{\dagger}_1\mathbf{p}'R'_1 + \mathbf{a}'_1)\,R'_2 + \mathbf{a}'_2 \\ &= \{R'_1R'_2 \mid R'^{\dagger}_2\mathbf{a}'_1R'_2 + \mathbf{a}'_2\}\,\mathbf{p}'.\end{aligned} \tag{2.84}$$

Symbolically, this transformation is represented by the following graph (cf. Figure 2.7). Thus, the resulting rigid displacement is defined as

$$\{R \mid \mathbf{a}\} = \{R'_1R'_2 \mid R'^{\dagger}_2\mathbf{a}'_1R'_2 + \mathbf{a}'_2\}\,. \tag{2.85}$$

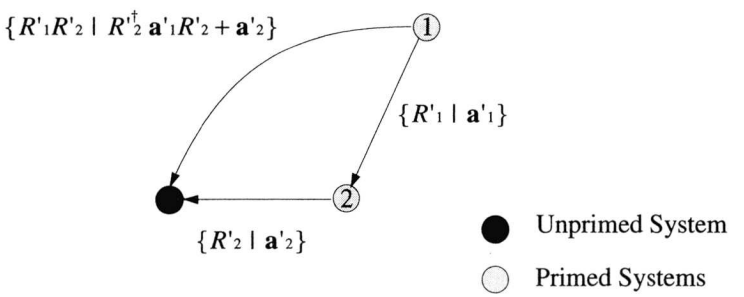


Figure 2.7 Two Consecutive Rigid Displacements

In analogy to this, the rigid displacement caused by n consecutive rigid displacements (cf. Figure 2.8) can now be defined as

$$\begin{aligned}\{R \mid \mathbf{a}\} &= \{R'_n \mid \mathbf{a}'_n\}\ldots\{R'_2 \mid \mathbf{a}'_2\}\,\{R'_1 \mid \mathbf{a}'_1\} \\ &= \{R'_1R'_2\ldots R'_n \mid (R'_2R'_3\ldots R'_n)^{\dagger}\mathbf{a}'_1\,(R'_2R'_3\ldots R'_n) \\ &\quad + (R'_3\ldots R'_n)^{\dagger}\mathbf{a}'_2\,(R'_3\ldots R'_n) + \ldots + \mathbf{a}'_n\,\}.\end{aligned} \tag{2.86}$$

To simplify expression (2.86), we introduce the following definitions:

$$
{}^{i}R = \begin{cases} R'_i R'_{i+1} \ldots R'_n = \prod_{j=i}^{n} R'_j & 1 \le i \le n, \\ 1 & \text{else}, \end{cases} \tag{2.87}
$$

and

$$
{}^{i}\mathbf{a} = \begin{cases} \sum_{j=i}^{n} {}^{j+1}R^{\dagger}\, \mathbf{a}'_j\, {}^{j+1}R = \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\mathbf{a}'_j & 1 \le i \le n, \\ 0 & \text{else}. \end{cases} \tag{2.88}
$$

Using these definitions, (2.86) can now be expressed in the simple form

$$
\{R \mid \mathbf{a}\} = \{{}^{1}R \mid {}^{1}\mathbf{a}\}. \tag{2.89}
$$

Remark 2.1:

Equations (2.87) and (2.88) are very suitable for recursive evaluation. We can write

$$
\begin{aligned} {}^{i}R &= R'_i\, {}^{i+1}R, \\ {}^{i}\mathbf{a} &= {}^{i+1}\mathbf{a} + {}^{i+1}\mathcal{R}\mathbf{a}'_i. \end{aligned} \qquad 1 \le i \le n \tag{2.90}
$$

Figure 2.8 shows the difference between the rigid displacements $\{R'_i \mid \mathbf{a}'_i\}$ and $\{{}^{i}R \mid {}^{i}\mathbf{a}\}$. The first one defines the relation between the position space of the reference frame $\{\mathbf{e}'_{i_k}\}$ with respect to the reference frame $\{\mathbf{e}'_{i+1_k}\}$, while the second one specifies the relation of the position spaces of the reference frame $\{\mathbf{e}'_{i_k}\}$ and the unprimed reference frame $\{\mathbf{e}_k\}$. We can also see that the two rigid displacements $\{R'_n \mid \mathbf{a}'_n\}$ and $\{{}^{n}R \mid {}^{n}\mathbf{a}\}$ are equivalent.

The first time derivative of a rigid displacement is

$$
\{\dot{R} \mid \dot{\mathbf{a}}\} = \{{}^{1}\dot{R} \mid {}^{1}\dot{\mathbf{a}}\} = \{\tfrac{1}{2}\, {}^{1}R i\, {}^{1}\omega \mid {}^{1}\dot{\mathbf{a}}\}. \tag{2.91}
$$

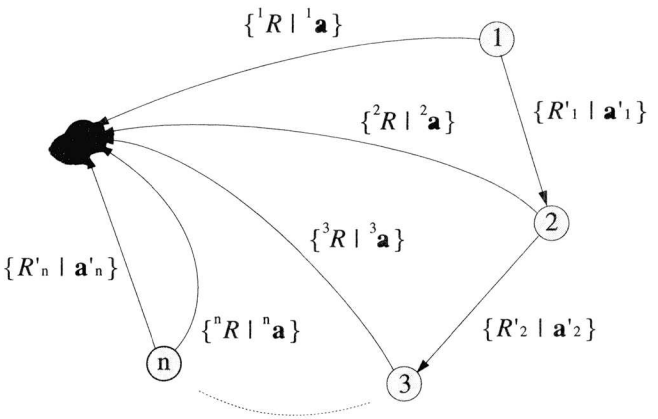


Figure 2.8 n Consecutive Rigid Displacements

For ${}^{i}\omega$ and ${}^{i}\dot{\mathbf{a}}$, in relation of ω'_j and $\dot{\mathbf{a}}'_j$ we can write (cf. Appendices D1 and D2)

$$
\begin{aligned}
{}^{i}\omega &= \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\,\omega'_j\,, \\
{}^{i}\dot{\mathbf{a}} &= \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\,\dot{\mathbf{a}}'_j + \sum_{j=i+1}^{n} {}^{j+1}\mathcal{R}\,\omega'_j \times ({}^{i}\mathbf{a} - {}^{j}\mathbf{a})\,.
\end{aligned}
\qquad 1 \le i \le n \qquad (2.92)
$$

In recursive formulation, (2.92) is expressed as (cf. Appendices D1 and D2.1)

$$
\begin{aligned}
{}^{i}\omega &= {}^{i+1}\omega + {}^{i+1}\mathcal{R}\,\omega'_i \\
{}^{i}\dot{\mathbf{a}} &= {}^{i+1}\dot{\mathbf{a}} + {}^{i+1}\omega \times {}^{i+1}\mathcal{R}\,\mathbf{a}'_i + {}^{i+1}\mathcal{R}\,\dot{\mathbf{a}}'_i\,.
\end{aligned}
\qquad 1 \le i \le n \qquad (2.93)
$$

Finally, the second time derivative of the rigid displacement $\{{}^1R \mid {}^1\mathbf{a}\}$ introduces the new variables ${}^1\dot{\omega}$ and ${}^1\ddot{\mathbf{a}}$. They are defined as (cf. Appendices D3 and D4)

$$
\begin{aligned}
{}^i\dot{\omega} &= \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\omega}'_j + \sum_{j=i}^{n-1} {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\omega'_j , \\
{}^i\ddot{\mathbf{a}} &= \sum_{j=i+1}^{n} {}^{j+1}\mathcal{R}\dot{\omega}'_j \times ({}^i\mathbf{a} - {}^j\mathbf{a}) + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\ddot{\mathbf{a}}'_j + \\
&\quad \sum_{j=i}^{n-1} \Big(\sum_{j^*=j+1}^{n-1} ({}^{j^*+1}\omega \times {}^{j^*+1}\mathcal{R}\omega'_{j^*}) \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + \\
&\qquad {}^{j+1}\omega \times ({}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + 2\,{}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j) \Big) .
\end{aligned}
\qquad 1 \le i \le n \quad (2.94)
$$

To simplify these equations we additionally define the vectors ${}^j\mathbf{g}$ and ${}^j\mathbf{h}$ by

$$
{}^i\mathbf{g} = \sum_{j=i}^{n-1} {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\omega'_j , \qquad (2.95)
$$

$$
\begin{aligned}
{}^i\mathbf{h} &= \sum_{j=i}^{n-1} \Big(\sum_{j^*=j+1}^{n-1} ({}^{j^*+1}\omega \times {}^{j^*+1}\mathcal{R}\omega'_{j^*}) \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + \\
&\qquad {}^{j+1}\omega \times ({}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + 2\,{}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j) \Big) \\
&= \sum_{j=i}^{n-1} ({}^{j+1}\mathbf{g} \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + {}^{j+1}\omega \times ({}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + 2\,{}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j)) .
\end{aligned}
\qquad (2.96)
$$

Remark 2.2:

Both definitions are again very appropriate for recursive evaluation. We write

$$
\begin{aligned}
{}^i\mathbf{g} &= {}^{i+1}\mathbf{g} + {}^{i+1}\omega \times {}^{i+1}\mathcal{R}\omega'_i , \\
{}^i\mathbf{h} &= {}^{i+1}\mathbf{h} + {}^{i+1}\mathbf{g} \times {}^{i+1}\mathcal{R}\mathbf{a}'_i + {}^{i+1}\omega \times ({}^{i+1}\omega \times {}^{i+1}\mathcal{R}\mathbf{a}'_i + 2\,{}^{i+1}\mathcal{R}\dot{\mathbf{a}}'_i)
\end{aligned}
\qquad (2.97)
$$

for $1 \le i < n$ with the initial conditions ${}^n\mathbf{g} = 0$ and ${}^n\mathbf{h} = 0$.

The simplified form of (2.98) can, therefore, be written as

$$\begin{aligned} {}^{i}\dot{\omega} &= \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\omega}'_j + {}^{i}\mathbf{g}\,, \\ {}^{i}\ddot{\mathbf{a}} &= \sum_{j=i+1}^{n} {}^{j+1}\mathcal{R}\dot{\omega}'_j \times ({}^{i}\mathbf{a} - {}^{j}\mathbf{a}) + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\ddot{\mathbf{a}}'_j + {}^{i}\mathbf{h}\,. \end{aligned} \tag{2.98}$$

2.4 Matrix Notation

This section introduces the matrix notation of vectors, spinors, and vector valued linear transformations. The matrix representation is very useful for computational evaluations. Note that the matrix representation is signified by a new typeface.

The matrix representation $\underline{\mathsf{a}}$ of vector $\mathbf{a}$ is given as

$$\underline{\mathsf{a}} = [a_k] = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}^{\mathrm{T}}, \tag{2.99}$$

with

$$\mathbf{a} = \sum_k \mathbf{a} \cdot \mathbf{e}_k \mathbf{e}_k = (a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + a_3\mathbf{e}_3)\,. \tag{2.100}$$

Thus, in case of one subscript index, this index designates the row number. In the same way the matrix representation $\underline{\mathsf{R}}$ of the unitary spinor R, with

$$R = a + i\mathbf{b} = a + i\,(b_1\mathbf{e}_1 + b_2\mathbf{e}_2 + b_3\mathbf{e}_3)\,, \tag{2.101}$$

can be represented as

$$\underline{R} = \begin{bmatrix} a & b_1 & b_2 & b_3 \end{bmatrix}^T = \begin{bmatrix} a \\ \underline{b} \end{bmatrix}. \tag{2.102}$$

The underscore emphasizes that $\underline{R}$, although with capital letter, stands for a vector.

The matrix representation of arbitrary, vector valued linear functions f of vector variables, $\mathbf{y} = f\mathbf{x}$, is derived as follows:

$$\mathbf{y} = f\mathbf{x} = f(\sum_k x_k \mathbf{e}_k) = \sum_k (f\mathbf{e}_k)\, x_k. \tag{2.103}$$

Since $f\mathbf{e}_k$ must be a vector again, it can be decomposed into its components, i.e.,

$$f\mathbf{e}_k = \sum_j \mathbf{e}_j\, (\mathbf{e}_j \cdot f\mathbf{e}_k) = \sum_j \mathbf{e}_j f_{jk}, \tag{2.104}$$

the components being defined as

$$f_{jk} = \mathbf{e}_j \cdot f\mathbf{e}_k. \tag{2.105}$$

With this the initial equation (2.103) may be rewritten in the following manner:

$$\mathbf{y} = \sum_j \mathbf{e}_j y_j = \sum_k f\mathbf{e}_k x_k = \sum_j \sum_k \mathbf{e}_j f_{jk} x_k. \tag{2.106}$$

In matrix notation, equation (2.106) is represented as

$$[y_j] = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = [f_{jk}]\, [x_k]. \tag{2.107}$$

Note that in case of two subscript indices, the first one always specifies the row index and the second one the column index. The matrix representation R of the rotation operator $\mathcal{R}$ can now be given as

$$\mathsf{R} = [r_{jk}], \tag{2.108}$$

with

$$r_{jk} = (2a^2 - 1)\,\mathbf{e}_j \cdot \mathbf{e}_k - 2\sum_i ab_i \mathbf{e}_i \cdot (\mathbf{e}_j \times \mathbf{e}_k) + 2b_j b_k. \tag{2.109}$$

The derivation of (2.109) is given in Appendix C6. Thus, the matrix describing an *arbitrary* orientation in the 3-dimensional Euclidean space is defined by

$$\mathsf{R} = \begin{bmatrix} (2a^2-1)+2b_1^2 & 2(b_1b_2-ab_3) & 2(b_1b_3+ab_2) \\ 2(b_2b_1+ab_3) & (2a^2-1)+2b_2^2 & 2(b_2b_3-ab_1) \\ 2(b_3b_1-ab_2) & 2(b_3b_2+ab_1) & (2a^2-1)+2b_3^2 \end{bmatrix}. \tag{2.110}$$

Remark 2.3:

Since the inverse of R is equivalent to its transpose (cf. (2.59)), it always exists. We can, therefore, write in matrix notation

$$\mathsf{R}^{-1} = \mathsf{R}^{\mathrm{T}}. \tag{2.111}$$

To represent the product of two spinors $R = R_1 R_2$ in matrix notation we additionally introduce the matrix $\overline{\mathsf{R}}$, with

$$\overline{\mathsf{R}} = \begin{bmatrix} -b_1 & -b_2 & -b_3 \\ a & b_3 & -b_2 \\ -b_3 & a & b_1 \\ b_2 & -b_1 & a \end{bmatrix}. \tag{2.112}$$

The spinor product can then be written as

$$\underline{R} = \left[\underline{R}_1\ \bar{\underline{R}}_1\right]\underline{R}_2. \tag{2.113}$$

To prove this relation, we simply expand the spinor product (cf. Example 2.2),

$$\begin{aligned} R = R_1 R_2 &= (a_1 + i\mathbf{b}_1)\ (a_2 + i\mathbf{b}_2) \\ &= a_1 a_2 - \mathbf{b}_1 \cdot \mathbf{b}_2 + i\,(a_1 \mathbf{b}_2 + a_2 \mathbf{b}_1 - \mathbf{b}_1 \times \mathbf{b}_2)\,. \end{aligned} \tag{2.114}$$

and transform it into matrix notation

$$\begin{aligned} \underline{R} = \begin{bmatrix} a \\ \underline{b} \end{bmatrix} &= \begin{bmatrix} a_1 a_2 - \underline{b}_1^{\mathrm{T}} \underline{b}_2 \\ a_1 \underline{b}_2 + a_2 \underline{b}_1 - \underline{b}_1 \times \underline{b}_2 \end{bmatrix} \\ &= \begin{bmatrix} a_1 & -b_{1_1} & -b_{1_2} & -b_{1_3} \\ b_{1_1} & a_1 & b_{1_3} & -b_{1_2} \\ b_{1_2} & -b_{1_3} & a_1 & b_{1_1} \\ b_{1_3} & b_{1_2} & -b_{1_1} & a_1 \end{bmatrix} \begin{bmatrix} a_2 \\ b_{2_1} \\ b_{2_2} \\ b_{2_3} \end{bmatrix} = \left[\underline{R}_1\ \bar{\underline{R}}_1\right]\underline{R}_2. \end{aligned} \tag{2.115}$$

Chapter 3

Kinematics and Dynamics

The main problem in the control of robots arises from the great complexity of the kinematical and dynamical equations describing these systems. Therefore, it is of great importance to use a numerically efficient and suitable representation of these equations for the controller.

As long as the main interest lies in the modeling of the gross motions, it is appropriate to neglect vibrations and to model the parts of the robot as rigid bodies. Elasticities can be included in the connections between these bodies. A lot of work has been done in the field of modeling the dynamics of rigid multi-body systems. The bases for the different methods are the *Newton-Euler laws*, the *Euler-Lagrange equations*, or *d'Alembert's principle*. Using the Newton-Euler equations, the dynamics of a rigid multi-body system are described in redundant form [Witten77]. The state of the system is both specified in Cartesian and in joint coordinates, and, as intermediate variables, there are the joint forces and torques. The dynamics are given by the Newton-Euler laws, relating the Cartesian coordinates to the forces and torques, whereas the redundancies can be resolved using a supplementary set of constraint equations. In the Euler-Lagrange approach, the dynamical equations are directly derived from two scalar quantities, the kinetic and the potential energy of the system. This results in a minimal state representation of the dynamics, i.e., there are no redundancies or intermediate variables anymore. Thus we obtain a more compact form for the dynamics. The connection between both approaches is given by d'Alem-

bert's principle which allows to transform the Newton-Euler representation of the dynamics into the Euler-Lagrange representation (cf. [KanLev80], [KanFäs84], [KanLev85], and [WaBuSh85]).

The numerical efficiency of a model not only depends on the physical laws or principles chosen to derive the equations but, to a large extent, on the variables selected and on the representation of the resulting equations. One possibility to improve the numerical efficiency for the evaluation of the dynamical equations consists in using recursive formulations. This was done, for example, by [LuWaPa80b] for Newton-Euler equations or by [Holler80] for Euler-Lagrange equations. [BalPat91] give a more detailed comparison of some other recursive formulations and introduce their own improved recursive representation.

In this chapter the dynamical equations are given in terms of Newton-Euler equations. While this approach induces a larger number of variables and equations, these equations are simpler, less coupled and more suitable for parallel computations than the minimal state representations of the dynamics usually derived with Euler-Lagrange equations. Since the redundant variables used to model the system represent physical quantities of interest for the control of the system, cumbersome evaluations of these unknowns can be avoided. Additionally, this approach permits to specify systems with closed kinematical chains quite easily and is much more appropriate for the definition of systems with variable structure, i.e., for systems with a changing number of degrees of freedom. Such changes in the system structure usually occur in connection with kinematic singularities or unilateral constraints. Working with minimal state representations, we have to change the dynamical model with every shift in the system structure. A system possessing n unilateral constraints with two states each, for example, has to be defined by 2^n models in minimal state representation. Using the Newton-Euler approach, such a unilateral constraint is specified by a constraint that is valid only if certain boundary constraints are satisfied.

The following sections derive the equations describing the kinematical and dynamical properties of arbitrary rigid, open-chain or closed-chain, multi-body systems. These equations form the basis for the kinematic and dynamic models to be used by the controller described in Chapter 5. The kinematical and dynamical equations are developed using the geometric algebra introduced in Chapter 2. This algebra permits to express the complex kinematical relations in rather simple form. The directional properties are described with quaternions or spinors. The advantage is twofold. First, there is no need to introduce angles if they are not required. Thus, the numerical effort to calculate the trigonometric functions can be avoided. Of course, it is always possible to evaluate an angle needed from the appropriate spinor. Second, singularities, as they can appear in connection with artificially introduced rotation axes, e.g., with Euler or Cardan angles for ball joints, are prevented.

An arbitrary, rigid multi-body system is completely characterized by

- its interconnection structure,
- the number of rigid bodies and their masses and inertias, and
- the geometrical and physical properties of its interconnections.

The interconnection structure specifies which bodies affect each other. It is represented by directed graphs as proposed in [Witten77]. This visualization of the interconnection structure using directed graphs is explained in Section 3.1. The coupling between two connected rigid bodies determines their possible relative rotational and translational motion as well as the forces and torques they exert on each other. Section 3.2 is concerned with the definition of these geometrical and physical properties of a connection, whereas Section 3.3 formulates the resulting motion and force constraints. Finally, Section 3.4 specifies the dynamics in terms of Newton-Euler equations. The constraints (geometric, force, and torque) and the dynamical equations are all given in a form suitable for recursive evaluation. This decreases the numerical effort substantially.

3.1 Interconnection Structure

The interconnection structure can be described rather easily using directed graphs. Rigid bodies are depicted by circles, whereas the connections between the bodies are symbolized with arrows. A connection always links exactly two rigid bodies. The interpretation of the direction of the arrows will be explained in the next section. The unprimed reference frame is symbolized by a black circle (cf. Figure 3.2).

Example 3.1

Definition of the interconnection structure of the system depicted in Figure 3.1 using a directed graph. There are two rigid bodies related to each other and to the reference frame by a total of four connections. This is expressed in the graph in Figure 3.2.

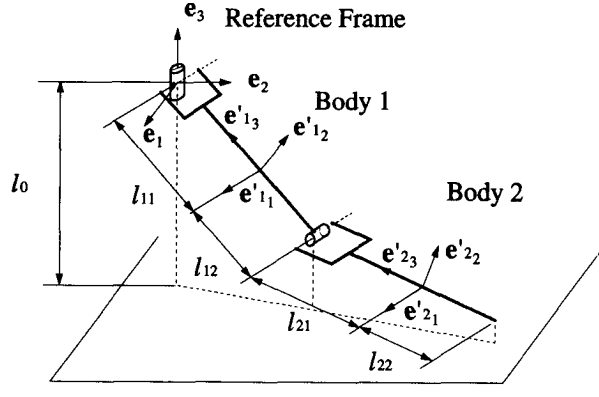


Figure 3.1

Connection 1 links body 1 to the reference frame and connection 2 ties body 2 to body 1. Connections 3 and 4, relating body 1 and 2, respectively, to the reference frame are only "active" if the joint between both bodies or the tip of the second body touch the plane. This type of

connection is called a *unilateral connection,* in contrast to *bilateral connections*, such as 1 and 2. Graphically they are differentiated by different line styles.

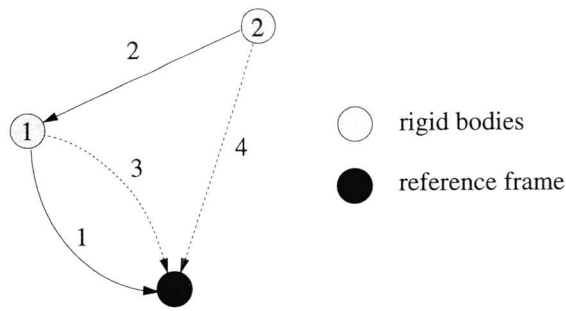


Figure 3.2

Next, this graphical representation of the interconnection structure is translated into mathematical notation. Let us assume that we have an arbitrary, rigid multi-body system composed of N rigid bodies linked with n connections specified by a directed graph.

Remark 3.1:

To avoid confusion, the indices "α" and "β" are always used to specify rigid bodies, while the index "a" refers to the connections. Thus, these indices are defined as

$$\begin{aligned} \{\alpha, \beta\} &\in [1, N], \\ a &\in [1, n]. \end{aligned} \tag{3.1}$$

From now on, we will also refer to a body indexed with "α" or a constraint indexed with "a" as body "α" and constraint "a", respectively.

The complete directed graph is represented by the *connectivity matrix* $[c_{\alpha a}]$ with the following definition for the matrix elements:

$$c_{\alpha a} = \begin{cases} -1 & \text{if arrow a is pointing away from body } \alpha \\ 1 & \text{if arrow a is pointing towards body } \alpha \\ 0 & \text{else} \end{cases} \tag{3.2}$$

Example 3.2

The connectivity matrix representing the directed graph (cf. Figure 3.2) in Example 3.1 is:

$$[c_{\alpha a}] = \begin{bmatrix} -1 & 1 & -1 & 0 \\ 0 & -1 & 0 & -1 \end{bmatrix}. \tag{3.3}$$

In addition, the following definitions are introduced for later use:

$$\begin{aligned} c^{+}_{\alpha a} &= \tfrac{1}{2} c_{\alpha a} (c_{\alpha a} + 1), \\ c^{-}_{\alpha a} &= \tfrac{1}{2} c_{\alpha a} (c_{\alpha a} - 1), \end{aligned} \tag{3.4}$$

both integers, $c^{+}_{\alpha a}$ and $c^{-}_{\alpha a}$, being equal to one if body "α" represents the end or the beginning of constraint "a", respectively, and zero otherwise.

3.2 Hinge Definition

In the previous section, the interconnection structure of arbitrary, multibody systems was defined by directed graphs and the corresponding connectivity matrix $[c_{\alpha a}]$. Next, the coupling caused by the connections has to be specified. We adopt the term *hinge* from [Witten77], standing for this coupling between two bodies. Such a hinge possesses geometrical and physical properties. The geometrical properties define the possible relative motion between the two connected bodies, whereas the physical properties determine the forces and torques that can be transmitted by the hinge. Both

properties depend on the rigid displacement relating the position spaces of the two connected bodies.

Assume that the hinge with index "a" is composed of n_a massless links, each one again represented by a rigid displacement $\{S'_{aj} | \mathbf{s}'_{aj}\}$. The resultant rigid displacement $\{S'_a | \mathbf{s}'_a\}$ is defined as

$$\{S'_a | \mathbf{s}'_a\} = \{S'_{an_a} | \mathbf{s}'_{an_a}\} \ldots \{S'_{a2} | \mathbf{s}'_{a2}\} \{S'_{a1} | \mathbf{s}'_{a1}\}. \tag{3.5}$$

It can be computed recursively from by (2.87), (2.88), and (2.89). The same is true for the resultant rotational and translational velocities θ'_a and $\dot{\mathbf{s}}'_a$ as well as for their first time derivatives (cf. (2.92), (2.97), and (2.98)). What remains to be done now is to specify the geometrical and the physical properties of the separate links of such a hinge.

The geometrical properties of a link consisting of n_{raj} rotational and n_{taj} translational degrees of freedom are fully determined by

$$S'_{aj} = a'_{aj} + i\mathbf{b}'_{aj} = a'_{aj} + i\sum_{k=0}^{n_{raj}} b'_{aj_k} \hat{\mathbf{b}}'_{aj_k}, \tag{3.6}$$

$$\mathbf{s}'_{aj} = \sum_{k=0}^{n_{taj}} \mathbf{s}'_{aj_k} = \sum_{k=0}^{n_{taj}} s'_{aj_k} \hat{\mathbf{s}}'_{aj_k}, \tag{3.7}$$

and the boundary constraints of the components b'_{aj_k} and s'_{aj_k}.

Remark 3.2:

The index "j" denotes a link of a hinge, while "k" specifies its degrees of freedom. Thus,

$$j \in [1, n_a] \tag{3.8}$$

and

$$k \in \begin{cases} [1, n_{raj}] & \text{for } S'_{aj}, \\ [1, n_{taj}] & \text{for } \mathbf{s}'_{aj}. \end{cases} \tag{3.9}$$

The components b'_{aj_0} and s'_{aj_0} are constant and represent the hinge parameters. S'_{aj} either represents a fixed rotational joint ($n_{raj} = 0$), a pin joint ($n_{raj} = 1$), or a ball joint ($n_{raj} = 3$). A Cardan joint, for example, with two rotational degrees of freedom is specified by two consecutive pin joints. $\mathbf{s}'_{aj}$, on the other hand, characterizes the translational joints.

Example 3.3

The rigid displacement specifying a translational joint with one degree of freedom in direction $\hat{\mathbf{s}}'_{aj_1}$ is defined as

$$\{S'_{a1} \mid \mathbf{s}'_{a1}\} = \{1 \mid \mathbf{s}'_{a1_0} + s'_{a1_1}\hat{\mathbf{s}}'_{a1_1}\}. \tag{3.10}$$

Example 3.4

The rigid displacement describing a ball joint is described as

$$\{S'_{a1} \mid \mathbf{s}'_{a1}\} = \{a'_{a1} + i(b'_{a1_1}\mathbf{e}_1 + b'_{a1_2}\mathbf{e}_2 + b'_{a1_3}\mathbf{e}_3) \mid 0\}. \tag{3.11}$$

Example 3.5

The *jth* link of hinge "a", composed of a pin joint and two translational joints (cf. Figure 3.3), is represented by the following rigid displacement:

$$\{S'_{aj} \mid \mathbf{s}'_{aj}\} = \{a'_{aj} + b'_{aj}\hat{\mathbf{b}}'_{aj} \mid \mathbf{s}'_{aj_0} + s'_{aj_1}\hat{\mathbf{s}}'_{aj_1} + s'_{aj_2}\hat{\mathbf{s}}'_{aj_2}\}. \tag{3.12}$$

Example 3.6

Specification of the hinges of the system depicted in Figure 3.1 (cf. Figure 3.2): The first hinge, between body 1 and the unprimed reference frame, possesses two rotational degrees of freedom. It is characterized by the following three rigid displacements:

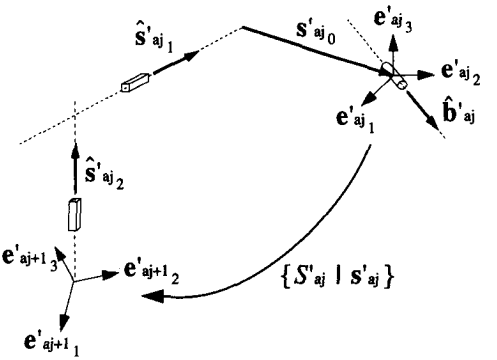


Figure 3.3 Rotational and Translational Joints

$$
\begin{aligned}
\{S'_{11} \mid \mathbf{s}'_{11}\} &= \{1 \mid -\tfrac{1}{2} l_{11}\, \mathbf{e}_3\}, \\
\{S'_{12} \mid \mathbf{s}'_{12}\} &= \{a'_{12} + i b'_{12_1} \mathbf{e}_1 \mid 0\}, \\
\{S'_{13} \mid \mathbf{s}'_{13}\} &= \{a'_{13} + i b'_{13_1} \mathbf{e}_3 \mid 0\}.
\end{aligned}
\tag{3.13}
$$

Hinge 2, between bodies 2 and 1 and having one rotational degree of freedom, is defined by the following two rigid displacements:

$$
\begin{aligned}
\{S'_{21} \mid \mathbf{s}'_{21}\} &= \{1 \mid -l_{21}\, \mathbf{e}_3\}, \\
\{S'_{22} \mid \mathbf{s}'_{22}\} &= \{a'_{22} + i b'_{22_1} \mathbf{e}_1 \mid -l_{12}\, \mathbf{e}_3\}.
\end{aligned}
\tag{3.14}
$$

The hinges 3 and 4, relating bodies 1 and 2 to the unprimed reference frame, are both described by two rigid displacements possessing three rotational and three translational degrees of freedom each. Additionally, the plane represents a boundary constraint. Thus, both hinges have to be defined by a rigid displacement and a boundary constraint:

$$
\begin{aligned}
\{S'_{31} \mid \mathbf{s}'_{31}\} &= \{1 \mid \tfrac{1}{2} l_{12}\, \mathbf{e}_3\}, \\
\{S'_{32} \mid \mathbf{s}'_{32}\} &= \{a'_{32} + i\,(b'_{32_1} \mathbf{e}_1 + b'_{32_2} \mathbf{e}_2 + b'_{32_3} \mathbf{e}_3) \mid \ldots \\
&\qquad s'_{32_1} \mathbf{e}_1 + s'_{32_2} \mathbf{e}_2 + s'_{32_3} \mathbf{e}_3\ \},
\end{aligned}
\tag{3.15}
$$

and

$$\begin{aligned}\{S'_{41} \mid \mathbf{s}'_{41}\} &= \{1 \mid \tfrac{1}{2} l_{22} \mathbf{e}_3\},\\ \{S'_{42} \mid \mathbf{s}'_{42}\} &= \{a'_{42} + ib'_{42_1}\mathbf{e}_1 + b'_{42_2}\mathbf{e}_2 + b'_{42_3}\mathbf{e}_3 \mid \ldots\\ &\qquad s'_{42_1}\mathbf{e}_1 + s'_{42_2}\mathbf{e}_2 + s'_{42_3}\mathbf{e}_3\}.\end{aligned} \tag{3.16}$$

Additionally, hinges 2 and 3 both possess one boundary constraint with

$$s'_{a2_3} \geq -l_0 \qquad a \in \{3, 4\}. \tag{3.17}$$

The rotational velocity vector θ'_{aj} of spinor $\dot{S}'_{aj}$ and the translational velocity $\dot{\mathbf{s}}'_{aj}$ can be expressed in component form (cf. (2.81) and (3.6)) as follows:

$$\begin{aligned}\theta'_{aj} &= -2iS'^{\dagger}_{aj}\dot{S}'_{aj} = 2\,(a'_{aj}\dot{\mathbf{b}}'_{aj} - \dot{a}'_{aj}\mathbf{b}'_{aj} + \mathbf{b}'_{aj} \times \dot{\mathbf{b}}'_{aj})\\ &= 2\left(-\dot{a}'_{aj}\mathbf{b}'_{aj} + \sum_{k=1}^{n_{raj}} \dot{b}'_{aj_k}\,(a'_{aj}\hat{\mathbf{b}}'_{aj_k} + \mathbf{b}'_{aj} \times \hat{\mathbf{b}}'_{aj_k})\right)\end{aligned} \tag{3.18}$$

and

$$\dot{\mathbf{s}}'_{aj} = \sum_{k=1}^{n_{taj}} \dot{\mathbf{s}}'_{aj_k} = \sum_{k=1}^{n_{taj}} \dot{s}'_{aj_k}\hat{\mathbf{s}}'_{aj_k}. \tag{3.19}$$

Finally, $\dot{\theta}'_{aj}$ and $\ddot{\mathbf{s}}'_{aj}$ are given as

$$\begin{aligned}\dot{\theta}'_{aj} &= 2\frac{\mathrm{d}}{\mathrm{d}t}\,(a'_{aj}\dot{\mathbf{b}}'_{aj} - \dot{a}'_{aj}\mathbf{b}'_{aj} + \mathbf{b}'_{aj} \times \dot{\mathbf{b}}'_{aj})\\ &= 2\,(\dot{a}'_{aj}\dot{\mathbf{b}}'_{aj} + a'_{aj}\ddot{\mathbf{b}}'_{aj} - \ddot{a}'_{aj}\mathbf{b}'_{aj} - \dot{a}'_{aj}\dot{\mathbf{b}}'_{aj}\\ &\qquad \dot{\mathbf{b}}'_{aj} \times \dot{\mathbf{b}}'_{aj} + \mathbf{b}'_{aj} \times \ddot{\mathbf{b}}'_{aj})\\ &= 2\left(-\ddot{a}'_{aj}\mathbf{b}'_{aj} + \sum_{k=1}^{n_{raj}} \ddot{b}'_{aj_k}\,(a'_{aj}\hat{\mathbf{b}}'_{aj_k} + \mathbf{b}'_{aj} \times \hat{\mathbf{b}}'_{aj_k})\right)\end{aligned} \tag{3.20}$$

and

$$\ddot{\mathbf{s}}'_{aj} = \sum_{k=1}^{n_{taj}} \ddot{\mathbf{s}}'_{aj_k} = \sum_{k=1}^{n_{taj}} \ddot{s}'_{aj_k} \hat{\mathbf{s}}'_{aj_k}. \tag{3.21}$$

Example 3.7

Evaluation of the rotational velocity vector θ'_{aj} for the pin joint specified by the rigid displacement

$$\{S'_{aj} \mid \mathbf{s}'_{aj}\} = \{a'_{aj} + ib'_{aj_1} \hat{\mathbf{b}}'_{aj_1} \mid 0\}. \tag{3.22}$$

From (3.18) and (2.63) we obtain

$$\begin{aligned} \theta'_{aj} &= 2\dot{a}'_{aj}(-b'_{aj}\hat{\mathbf{b}}'_{aj}) + \sum_{k=1}^{1} \dot{b}'_{aj_k}(a'_{aj}\hat{\mathbf{b}}'_{aj_k} + \mathbf{b}'_{aj} \times \hat{\mathbf{b}}'_{aj_k}) \\ &= 2(-\dot{a}'_{aj}b'_{aj_1} + a'_{aj}\dot{b}'_{aj_1})\hat{\mathbf{b}}'_{aj_1} \\ &= 2(-(-\sin\tfrac{1}{2}\varphi'_{aj})\tfrac{1}{2}\dot{\varphi}'_{aj}\sin\tfrac{1}{2}\varphi'_{aj} + \cos\tfrac{1}{2}\varphi'_{aj}\cos\tfrac{1}{2}\varphi'_{aj}(\tfrac{1}{2}\dot{\varphi}'_{aj}))\hat{\mathbf{b}}'_{aj_1} \\ &= \dot{\varphi}'_{aj}\hat{\mathbf{b}}'_{aj}. \end{aligned} \tag{3.23}$$

Example 3.8

Evaluation of the rotational velocity vector θ'_{aj} for the ball joint specified by the rigid displacement

$$\{S'_{aj} \mid \mathbf{s}'_{aj}\} = \{a'_{aj} + i\sum_{k=1}^{3} b'_{aj_k}\mathbf{e}_k \mid 0\}. \tag{3.24}$$

Thus, (3.18) becomes

$$\begin{aligned} \theta'_{aj} = {} & 2(-\dot{a}'_{aj}b_{aj_1} + a'_{aj}\dot{b}'_{aj_1} - b_{aj_3}\dot{b}'_{aj_2} + b_{aj_2}\dot{b}'_{aj_3})\mathbf{e}_1 + \\ & 2(-\dot{a}'_{aj}b_{aj_2} + b_{aj_3}\dot{b}'_{aj_1} + a'_{aj}\dot{b}'_{aj_2} - b_{aj_1}\dot{b}'_{aj_3})\mathbf{e}_2 + \\ & 2(-\dot{a}'_{aj}b_{aj_3} - b_{aj_2}\dot{b}'_{aj_1} + b_{aj_1}\dot{b}'_{aj_2} + a'_{aj}\dot{b}'_{aj_3})\mathbf{e}_3. \end{aligned} \tag{3.25}$$

Hitherto we have only defined the geometrical properties of an arbitrary link with the corresponding rigid displacement specifying the direction and type (rotational, translational or both) of its degrees of freedom. Next, we

determine the resulting physical properties, i.e., the forces and torques transmitted by the links. They are denoted by f'_{aj_k} and τ'_{aj_k}. We differentiate between active and passive forces and torques. The *active link forces and torques* are those which are used to control the system, i.e., the actuator forces and torques. They are denoted with $\mathbf{u}_p$, index "p" specifying the actuator. All other forces and torques, as for example, friction forces and torques, forces and torques caused by spring damping elements, etc., are referred to as *passive link forces and torques*. They are designated as f'_{Paj_k} and τ'_{Paj_k}, respectively. For the torque and the force τ'_{aj_k}, f'_{aj_k}, transmitted by link "aj" in directions $\hat{\mathbf{b}}_{aj_k}$ and $\hat{\mathbf{s}}_{aj_k}$, respectively, we write

$$\tau'_{aj_k} = \sum_{p=1}^{n_u} b_{1u_{i_r p}} u_p + \tau'_{Paj_k} \tag{3.26}$$

and

$$f'_{aj_k} = \sum_{p=1}^{n_u} b_{2u_{i_t p}} u_p + f'_{Paj_k}, \tag{3.27}$$

with

$$b_{1u_{i_r p}} = \begin{cases} \hat{\mathbf{u}}_p \cdot \hat{\mathbf{b}}_{aj_k} & \text{if the torque } \mathbf{u}_p \text{ acts on joint } \hat{\mathbf{b}}_{aj_k} \\ 0 & \text{else} \end{cases} \tag{3.28}$$

$$b_{2u_{i_t p}} = \begin{cases} \hat{\mathbf{u}}_p \cdot \hat{\mathbf{s}}_{aj_k} & \text{if the force } \mathbf{u}_p \text{ acts on joint } \hat{\mathbf{s}}_{aj_k} \\ 0 & \text{else} \end{cases} \tag{3.29}$$

$$\begin{aligned} i_r &= i_r(a,j,k) = \sum_{a'=1}^{a-1} \sum_{j'=1}^{n_a} n_{raj} + \sum_{j'=1}^{j-1} n_{raj} + k \qquad k \in [1, n_{raj}] \\ i_t &= i_t(a,j,k) = \sum_{a'=1}^{a-1} \sum_{j'=1}^{n_a} n_{taj} + \sum_{j'=1}^{j-1} n_{taj} + k \qquad k \in [1, n_{taj}] \end{aligned} \tag{3.30}$$

Remark 3.3:

Ideal rotational or translational joints which are not actuated are characterized by $\tau'_{aj_k} = 0$ or $f'_{aj_k} = 0$, respectively.

3.3 Constraints

So far, we have only specified the geometrical and physical properties of arbitrary hinges. Next we define their consequences, i.e., the resulting constraint equations. Assume that the position and orientation of *each* body reference frame $\{\mathbf{e}'_{\alpha_k}\}$ are given by the rigid displacement $\{R_\alpha \mid \mathbf{X}_\alpha\}$ relating $\{\mathbf{e}'_{\alpha_k}\}$ to the primed reference frame $\{\mathbf{e}_k\}$. These rigid displacements describe the configuration of every body and, therefore, of the complete system without taking the connections into account.

An arbitrary hinge "a" between bodies "β" and "α" (cf. Figure 3.4) causes the following constraints

$$\begin{aligned} \{R_\alpha \mid \mathbf{X}_\alpha\}\ \{S'_a \mid \mathbf{s}'_a\} &= \{R_\beta \mid \mathbf{X}_\beta\} \\ \{S'_a R_\alpha \mid \mathcal{R}_\alpha \mathbf{s}'_a + \mathbf{X}_\alpha\} &= \{R_\beta \mid \mathbf{X}_\beta\} \\ \{S_a \mid \mathbf{s}_a\} &= \{R_\beta \mid \mathbf{X}_\beta\}\,. \end{aligned} \tag{3.31}$$

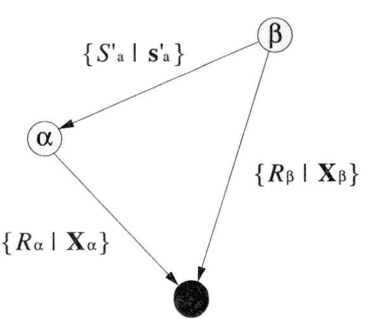


Figure 3.4

This is the basic relation for the derivation of the motional constraints and the force and torque constraints.

Remark 3.4:

$\{{}^{i}S_a \mid {}^{i}\mathbf{s}_a\}$ (cf. (2.87) and (2.88)) is the rigid displacement specifying the link frame $\{\mathbf{e}'_{a_k}\}$ with respect to the base frame $\{\mathbf{e}_k\}$, whereas $\{{}^{i}S'_a \mid {}^{i}\mathbf{s}'_a\}$ does so with respect to the body frame $\{\mathbf{e}'_{\alpha_k}\}$. To apply (2.87), (2.88), (2.92), and (2.98) on $\{{}^{i}S_a \mid {}^{i}\mathbf{s}_a\}$ we must always keep in mind that $\{S_a \mid \mathbf{s}_a\}$ is now composed of $n_a + 1$ rigid displacements and that

$$\begin{aligned} S'_{a\,n_a+1} &= R_\alpha = a_\alpha + i\mathbf{b}_\alpha, \\ \mathbf{s}'_{a\,n_a+1} &= \mathbf{X}_\alpha. \end{aligned} \tag{3.32}$$

3.3.1 Positional Constraints

With equations (3.31), (3.32), (3.5), (2.87), and (2.88) we can write the positional constraints

$$S_a = {}^{1}S_a = \prod_{j=1}^{n_a+1} S'_{aj} = \left(\prod_{j=1}^{n_a} S'_{aj}\right) R_\alpha = R_\beta \tag{3.33}$$

and

$$\mathbf{s}_a = {}^{1}\mathbf{s}_a = \sum_{j=1}^{n_a+1} {}^{j+1}S_a \mathbf{s}'_{aj} = \sum_{j=1}^{n_a} {}^{j+1}S_a \mathbf{s}'_{aj} + \mathbf{X}_\alpha = \mathbf{X}_\beta. \tag{3.34}$$

Equations (3.33) and (3.34) are both nonlinear. Thus, they can in general be solved iteratively only. For that purpose it is a prerequisite to know their linear approximation. The first order Taylor expansion of $S_a(t + \Delta t)$ and $\mathbf{s}_a(t + \Delta t)$, where t is an arbitrary scalar variable, is defined as

$$S_a(t + \Delta t) = S_a(t) + \Delta S_a(t) = S_a(t) + \Delta t \dot{S}_a(t) \tag{3.35}$$

and

$$\mathbf{s}_a(t+\Delta t) = \mathbf{s}_a(t) + \Delta\mathbf{s}_a(t) = \mathbf{s}_a(t) + \Delta t\dot{\mathbf{s}}_a(t) \tag{3.36}$$

The linearized form of the constraint equations (3.33) and (3.34) can then be written as

$$S_a + \Delta S_a - R_\beta - \Delta R_\beta = 0 \tag{3.37}$$

and

$$\mathbf{s}_a + \Delta\mathbf{s}_a - \mathbf{X}_\beta - \Delta\mathbf{X}_\beta = 0. \tag{3.38}$$

ΔS_a and $\Delta\mathbf{s}_a$ are given as (cf. (2.91), (2.92), (3.18))

$$\begin{aligned}\Delta S_a &= \Delta t\,{}^1\dot{S}_a = \tfrac{1}{2}\,{}^1S_a i\Delta t\sum_{j=1}^{n_a+1} {}^{j+1}S_a\theta'_{aj} \\ &= \tfrac{1}{2}\,{}^1S_a i\Delta t\sum_{j=1}^{n_a}({}^{j+1}S_a\theta'_{aj} + \omega_\alpha) \\ &= \tfrac{1}{2}\,{}^1S_a i\sum_{j=1}^{n_a}({}^{j+1}S_a\Delta\varphi'_{aj} + \Delta\vartheta_\alpha)\end{aligned} \tag{3.39}$$

and

$$\begin{aligned}\Delta\mathbf{s}_a &= \Delta t\,{}^1\dot{\mathbf{s}}_a = \Delta t\left(\sum_{j=2}^{n_a+1}({}^{j+1}S_a\theta'_{aj} \times ({}^1\mathbf{s}_a - {}^j\mathbf{s}_a)) + \sum_{j=1}^{n_a+1} {}^{j+1}S_a\dot{\mathbf{s}}'_{aj}\right) \\ &= \Delta t\left(\sum_{j=2}^{n_a} {}^{j+1}S_a\theta'_{aj} \times ({}^1\mathbf{s}_a - {}^j\mathbf{s}_a) + \omega_\alpha \times ({}^1\mathbf{s}_a - \mathbf{X}_\alpha) + \right. \\ &\qquad \left. \sum_{j=1}^{n_a} {}^{j+1}S_a\dot{\mathbf{s}}'_{aj} + \dot{\mathbf{X}}_\alpha\right) \\ &= \sum_{j=2}^{n_a} {}^{j+1}S_a\Delta\varphi'_{aj} \times ({}^1\mathbf{s}_a - {}^j\mathbf{s}_a) + \Delta\vartheta_\alpha \times ({}^1\mathbf{s}_a - \mathbf{X}_\alpha) + \\ &\qquad \sum_{j=1}^{n_a} {}^{j+1}S_a\Delta\mathbf{s}'_{aj} + \Delta\mathbf{X}_\alpha,\end{aligned} \tag{3.40}$$

with

$$ {}^{j+1}\mathcal{S}_a \Delta\varphi'_{aj} = \sum_{k=1}^{n_{raj}} \Delta\varphi'_{aj_k} {}^{j+1}\mathcal{S}_a \hat{\mathbf{b}}'_{aj_k} = \begin{cases} \Delta\varphi'_{aj_1} \hat{\mathbf{b}}_{aj_1} & \text{for pin joints} \\ \sum_{k=1}^{3} \Delta\varphi'_{aj_k} \mathbf{e}'_{aj_k} & \text{for ball joints} \end{cases} \tag{3.41} $$

$$ {}^{j+1}\mathcal{S}_a \Delta\mathbf{s}'_{aj} = \sum_{k=1}^{n_{taj}} \Delta s'_{aj_k} {}^{j+1}\mathcal{S}_a \hat{\mathbf{s}}'_{aj_k} = \sum_{k=1}^{n_{taj}} \Delta s'_{aj_k} \hat{\mathbf{s}}_{aj_k}. \tag{3.42} $$

Substituting $\Delta \mathcal{S}_a$ and $\Delta \mathbf{s}_a$ by (3.39) and (3.40) in (3.37), (3.38) and using (3.2) we get the following linear approximations for the constraint equations:

$$ \begin{aligned} i\mathcal{S}_a \sum_{j=1}^{n_a} \sum_{k=1}^{n_{raj}} \Delta\varphi'_{aj_k} \hat{\mathbf{b}}_{aj_k} + i \sum_{\alpha=1}^{N} (c^+_{\alpha a}\, \mathcal{S}_a - c^-_{\alpha a} R\alpha)\, \Delta\vartheta_\alpha + \\ 2 \sum_{\alpha=1}^{N} (c^+_{\alpha a} \mathcal{S}_a - c^-_{\alpha a} R\alpha) = 0, \end{aligned} \tag{3.43} $$

and

$$ \begin{aligned} \sum_{j=2}^{n_a} \sum_{k=1}^{n_{raj}} \Delta\varphi'_{aj_k} \hat{\mathbf{b}}_{aj_k} \times (\mathbf{s}_a - {}^j\mathbf{s}_a) + \sum_{j=1}^{n_a} \sum_{k=1}^{n_{taj}} \Delta s'_{aj_k} \hat{\mathbf{s}}_{aj_k} + \\ \sum_{\alpha=1}^{N} (c^+_{\alpha a} \Delta\vartheta_\alpha \times (\mathbf{s}_a - \mathbf{X}_\alpha) + c_{\alpha a} \Delta\mathbf{X}_\alpha + c^+_{\alpha a}\, \mathbf{s}_a - c^-_{\alpha a} \mathbf{X}_\alpha) = 0. \end{aligned} \tag{3.44} $$

Remark 3.5:

Care has to be taken in the update step of a spinor, i.e., evaluating the new spinor $R(t+\Delta t)$ from $R(t)$ and $\Delta\vartheta(t)$ if we use the first order approximation

$$ R(t+\Delta t) = R(t) + \Delta t \dot{R}(t) = R(t)\,(1 + \tfrac{1}{2} i \Delta\vartheta(t)) \tag{3.45} $$

to do so. This is due to the fact that the new spinor $R(t+\Delta t)$ is no longer unitary because

$$\begin{aligned} |R(t+\Delta t)|^2 &= (R(t+\Delta t))^{\dagger}(R(t+\Delta t)) \\ &= (1+\tfrac{1}{2}i\Delta\vartheta(t))^{\dagger}R(t)^{\dagger}R(t)(1+\tfrac{1}{2}i\Delta\vartheta(t)) \\ &= (1-\tfrac{1}{2}i\Delta\vartheta(t))(1+\tfrac{1}{2}i\Delta\vartheta(t)) \\ &= 1+\tfrac{1}{4}\Delta\vartheta^2(t). \end{aligned} \tag{3.46}$$

Therefore, it additionally has to be "normalized", i.e., we have to write

$$R(t+\Delta t) = \frac{R(t)(1+\frac{1}{2}i\Delta\vartheta(t))}{(1+\frac{1}{4}i\Delta\vartheta^2(t))^{1/2}}. \tag{3.47}$$

The boundary constraints are specified with the following supplementary positional constraints:

$$\Delta\varphi'_{aj_k} = 0 \qquad \text{if } \varphi'_{aj_k} \le \varphi'_{\min aj_k} \text{ or } \varphi'_{aj_k} \ge \varphi'_{\max aj_k}, \tag{3.48}$$

$$\Delta s'_{aj_k} = 0 \qquad \text{if } s'_{aj_k} \le s'_{\min aj_k} \text{ or } s'_{aj_k} \ge s'_{\max aj_k}. \tag{3.49}$$

Of course, they are only valid if the upper or lower boundaries of the corresponding variables are reached.

3.3.2 Velocity Constraints

Evaluating the first time derivative of (3.31) we can write

$$\begin{aligned} \{\dot{S}_a \mid \dot{\mathbf{s}}_a\} &= \{\dot{R}_\beta \mid \dot{\mathbf{X}}_\beta\}, \\ \{\tfrac{1}{2}S_a i\theta_a \mid \dot{\mathbf{s}}_a\} &= \{\tfrac{1}{2}R_\beta i\omega_\beta \mid \dot{\mathbf{X}}_\beta\}. \end{aligned} \tag{3.50}$$

Equations (3.31), (3.32), and (3.50) are only satisfied for

$$\theta_a = {}^1\theta_a = \sum_{j=1}^{n_a+1} {}^{j+1}\mathcal{S}_a\,\theta'_{aj} = \sum_{j=1}^{n_a} {}^{j+1}\mathcal{S}_a\,\theta'_{aj} + \omega_\alpha = \omega_\beta \tag{3.51}$$

and

$$\begin{aligned}\dot{\mathbf{s}}_a = {}^1\dot{\mathbf{s}}_a &= \sum_{j=2}^{n_a+1} \left({}^{j+1}\mathcal{S}_a\,\theta'_{aj} \times ({}^1\mathbf{s}_a - {}^j\mathbf{s}_a) \right) + \sum_{j=1}^{n_a+1} {}^{j+1}\mathcal{S}_a\,\dot{\mathbf{s}}'_{aj} \\ &= \sum_{j=2}^{n_a} {}^{j+1}\mathcal{S}_a\,\theta'_{aj} \times ({}^1\mathbf{s}_a - {}^j\mathbf{s}_a) + \omega_\alpha \times ({}^1\mathbf{s}_a - \mathbf{X}_\alpha) + \\ &\quad \sum_{j=1}^{n_a} {}^{j+1}\mathcal{S}_a\,\dot{\mathbf{s}}'_{aj} + \dot{\mathbf{X}}_\alpha \\ &= \dot{\mathbf{X}}_\beta\,,\end{aligned} \tag{3.52}$$

with

$${}^{j+1}\mathcal{S}_a\,\theta'_{aj} = \sum_{k=1}^{n_{raj}} \theta'_{aj_k}\, {}^{j+1}\mathcal{S}_a\,\hat{\mathbf{b}}'_{aj_k} = \begin{cases} \theta'_{aj_1}\,\hat{\mathbf{b}}_{aj_1} & \text{for pin joints} \\ \sum_{k=1}^{3} \theta'_{aj_k}\,\mathbf{e}'_{aj_k} & \text{for ball joints} \end{cases} \tag{3.53}$$

$${}^{j+1}\mathcal{S}_a\,\dot{\mathbf{s}}'_{aj} = \sum_{k=1}^{n_{taj}} \dot{s}'_{aj_k}\, {}^{j+1}\mathcal{S}_a\,\hat{\mathbf{s}}'_{aj_k} = \sum_{k=1}^{n_{taj}} \dot{s}'_{aj_k}\,\hat{\mathbf{s}}_{aj_k}\,. \tag{3.54}$$

Thus, additionally incorporating the connectivity coefficients we write for the velocity constraints:

$$\sum_{j=1}^{n_a} \sum_{k=1}^{n_{raj}} \theta'_{aj_k}\,\hat{\mathbf{b}}_{aj_k} + \sum_{\alpha=1}^{N} c_{\alpha a}\,\omega_\alpha = 0 \tag{3.55}$$

and

$$\sum_{j=2}^{n_a}\sum_{k=1}^{n_{taj}}\theta'_{aj_k}\hat{\mathbf{b}}_{aj_k}\times(\mathbf{s}_a-{}^j\mathbf{s}_a)+\sum_{j=1}^{n_a}\sum_{k=1}^{n_{taj}}\dot{s}'_{aj_k}\hat{\mathbf{s}}_{aj_k}+\sum_{\alpha=1}^{N}(c^{+}_{\alpha a}\omega_\alpha\times(\mathbf{s}_a-\mathbf{X}_\alpha)+c_{\alpha a}\dot{\mathbf{X}}_\alpha)=0. \tag{3.56}$$

Furthermore, the boundary constraints yield

$$\theta'_{aj_k}=0 \qquad \text{if } \varphi'_{aj_k}\le\varphi'_{\min aj_k} \text{ or } \varphi'_{aj_k}\ge\varphi'_{\max aj_k}, \tag{3.57}$$

$$\dot{s}'_{aj_k}=0 \qquad \text{if } s'_{aj_k}\le s'_{\min aj_k} \text{ or } s'_{aj_k}\ge s'_{\max aj_k}. \tag{3.58}$$

3.3.3 Acceleration Constraints

For the determination of the acceleration constraints we make use of (2.97), (2.98), substituting ${}^iR$ and ${}^i\mathbf{a}$ by ${}^jS_a$ and ${}^j\mathbf{s}_a$, respectively (cf. (3.32), (3.33), and (3.34)). We get the following expressions for $\dot{\theta}_a$ and $\ddot{\mathbf{s}}_a$:

$$\begin{aligned}\dot{\theta}_a={}^1\dot{\theta}_a&=\sum_{j=1}^{n_a+1}{}^{j+1}\mathcal{S}_a\dot{\theta}'_{aj}+{}^1\mathbf{g}_a=\sum_{j=1}^{n_a}{}^{j+1}\mathcal{S}_a\dot{\theta}'_{aj}+\dot{\omega}_\alpha+{}^1\mathbf{g}_a\\&=\dot{\omega}_\beta\end{aligned} \tag{3.59}$$

and

$$\begin{aligned}\ddot{\mathbf{s}}_a={}^1\ddot{\mathbf{s}}_a&=\sum_{j=2}^{n_a+1}{}^{j+1}\mathcal{S}_a\dot{\theta}'_{aj}\times({}^1\mathbf{s}_a-{}^j\mathbf{s}_a)+\sum_{j=1}^{n_a+1}{}^{j+1}\mathcal{S}_a\ddot{\mathbf{s}}'_{aj}+{}^1\mathbf{h}_a\\&=\sum_{j=2}^{n_a}{}^{j+1}\mathcal{S}_a\dot{\theta}'_{aj}\times({}^1\mathbf{s}_a-{}^j\mathbf{s}_a)+\dot{\omega}_\alpha\times({}^1\mathbf{s}_a-\mathbf{X}_\alpha)+\\&\quad\sum_{j=1}^{n_a}{}^{j+1}\mathcal{S}_a\ddot{\mathbf{s}}'_{aj}+\ddot{\mathbf{X}}_\alpha+{}^1\mathbf{h}_a=\ddot{\mathbf{X}}_\beta\end{aligned} \tag{3.60}$$

with

$$^{j+1}\mathcal{S}_a\dot{\theta}'_{aj} = \sum_{k=1}^{n_{raj}} \dot{\theta}'_{aj_k}\, {}^{j+1}\mathcal{S}_a\hat{\mathbf{b}}'_{aj_k} = \begin{cases} \dot{\theta}'_{aj_1}\hat{\mathbf{b}}_{aj_1} & \text{for pin joints} \\ \sum_{k=1}^{3} \dot{\theta}'_{aj_k}\mathbf{e}'_{aj_k} & \text{for ball joints} \end{cases} \tag{3.61}$$

$$^{j+1}\mathcal{S}_a\ddot{\mathbf{s}}'_{aj} = \sum_{k=1}^{n_{taj}} \ddot{s}'_{aj_k}\, {}^{j+1}\mathcal{S}_a\hat{\mathbf{s}}'_{aj_k} = \sum_{k=1}^{n_{taj}} \ddot{s}'_{aj_k}\hat{\mathbf{s}}_{aj_k}, \tag{3.62}$$

whereas the definitions for $^1\mathbf{g}_a$ and $^1\mathbf{h}_a$ can be found in (2.97). The acceleration constraints in dependence of the connectivity coefficients are then:

$$\sum_{j=1}^{n_a}\sum_{k=1}^{n_{raj}} \dot{\theta}'_{aj_k}\hat{\mathbf{b}}_{aj_k} + \sum_{\alpha=1}^{N} c_{\alpha a}\dot{\omega}_\alpha + {}^1\mathbf{g}_a = 0 \tag{3.63}$$

and

$$\begin{aligned} &\sum_{j=2}^{n_a}\sum_{k=1}^{n_{raj}} \dot{\theta}'_{aj_k}\hat{\mathbf{b}}_{aj_k} \times (\mathbf{s}_a - {}^j\mathbf{s}_a) + \sum_{j=1}^{n_a}\sum_{k=1}^{n_{taj}} \ddot{s}'_{aj_k}\hat{\mathbf{s}}_{aj_k} + \\ &\sum_{\alpha=1}^{N} (c^+_{\alpha a}\,\dot{\omega}_\alpha \times (\mathbf{s}_a - \mathbf{X}_\alpha) + c_{\alpha a}\ddot{\mathbf{X}}_\alpha) - {}^1\mathbf{h}_a = 0. \end{aligned} \tag{3.64}$$

Finally, the boundary constraints are given by

$$\dot{\theta}'_{aj_k} = 0 \qquad \text{if } \beta'_{aj_k} \le \beta'_{\min aj_k} \text{ or } \beta'_{aj_k} \ge \beta'_{\max aj_k}, \tag{3.65}$$

$$\ddot{s}'_{aj_k} = 0 \qquad \text{if } s'_{aj_k} \le s'_{\min aj_k} \text{ or } s'_{aj_k} \ge s'_{\max aj_k}. \tag{3.66}$$

3.3.4 Force and Torque Constraints

A hinge "a" does not only specify the admissible relative movements of two bodies "α" and "β", but also the forces and torques they exert on each other. From now on, they will be designated as *interaction variables*,

symbolized by $\tau_a, \mathbf{f}_a$. Let us assume that hinge "a" is defined from body "β" to body "α".

Remark 3.6:

The cut between bodies "β" and "α" is performed at the origin of the body frame $\{\mathbf{e}'_{\beta_k}\}$, and the hinge moments and forces $\tau_a, \mathbf{f}_a$ are defined as the torque and force, respectively, exerted by body "α" on body "β".

Two sets of forces and torques acting on the same body induce the same motion if their resultant forces and torques are equivalent (Newton-Euler laws, (3.86), (3.87), (3.88), and (3.89)). This knowledge is utilized to establish the force and torque constraints. The resultant forces and torques of $\tau_1, \mathbf{f}_1$ and $\tau_2, \mathbf{f}_2$, both acting on the same body at positions $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively, are specified by (3.88) and (3.89). To be equivalent their resultant moments and forces $\Gamma, \mathbf{F}$ have to be identical. Hence,

$$\Gamma = \tau_1 + \mathbf{r}_1 \times \mathbf{f}_1 = \tau_2 + \mathbf{r}_2 \times \mathbf{f}_2 \tag{3.67}$$

and

$$\mathbf{F} = \mathbf{f}_1 = \mathbf{f}_2. \tag{3.68}$$

Thus, the moments and forces τ_1 and $\mathbf{f}_1$ are equivalent to τ_2 and $\mathbf{f}_2$ if

$$\tau_1 = \tau_2 + (\mathbf{r}_2 - \mathbf{r}_1) \times \mathbf{f}_2 \tag{3.69}$$

and

$$\mathbf{f}_1 = \mathbf{f}_2. \tag{3.70}$$

Therefore, the relation between the joint moments and forces τ'_{aj_k}, f'_{aj_k} and hinge torques and forces $\tau_a, \mathbf{f}_a$ can now be established as

$$\tau'_{aj_k} = \tau_{aj} \cdot \hat{\mathbf{b}}_{aj_k} = (\tau_a + ({}^{l}\mathbf{s}_a - {}^{j}\mathbf{s}_a) \times \mathbf{f}_a) \cdot \hat{\mathbf{b}}_{aj_k}, \tag{3.71}$$

$$f'_{aj_k} = \mathbf{f}_{aj} \cdot \hat{\mathbf{s}}_{aj_k} = \mathbf{f}_a \cdot \hat{\mathbf{s}}_{aj_k}. \tag{3.72}$$

In final form, also making use of (3.26), (3.27) and the relation proved in Example 2.3., we can, therefore, write for the force and torque constraints:

$$\boldsymbol{\tau}_a \cdot \hat{\mathbf{b}}_{aj_k} + \mathbf{f}_a \cdot (\hat{\mathbf{b}}_{aj_k} \times ({}^{1}\mathbf{s}_a - {}^{j}\mathbf{s}_a)) - \sum_{p=1}^{n_u} b_{1u_{ir_p}} u_p - \boldsymbol{\tau}'_{Paj_k} = 0, \tag{3.73}$$

$$\mathbf{f}_a \cdot \hat{\mathbf{s}}_{aj_k} - \sum_{p=1}^{n_u} b_{2u_{it_p}} u_p - f'_{Paj_k} = 0. \tag{3.74}$$

3.4 Dynamics

A body consists of many particles. If the relative position of these particles is fixed we refer to such a body as a *rigid body.* Hereafter, we shall only be concerned with the dynamics of rigid bodies.

An arbitrary rigid body "α" (cf. Figure 3.5) is characterized by its object and state variables. The *object variables* determine the intrinsic, structural properties of the body. These are mass m_α, inertia tensor $\mathcal{J}_\alpha$, position of the center of mass in the body frame, and size or shape of the body. The body mass and the inertia tensor of body $\mathcal{B}_\alpha$ are defined by the following integrals:

$$m_\alpha = \int_{\mathcal{B}_\alpha} dm \tag{3.75}$$

and

$$\mathcal{J}_\alpha \mathbf{a} = \int_{\mathcal{B}_\alpha} \mathbf{r}_\alpha \mathbf{r}_\alpha \wedge \mathbf{a} dm = \int_{\mathcal{B}_\alpha} (\mathbf{r}_\alpha^2 \mathbf{a} - \mathbf{r}_\alpha \mathbf{r}_\alpha \cdot \mathbf{a})\, dm. \tag{3.76}$$

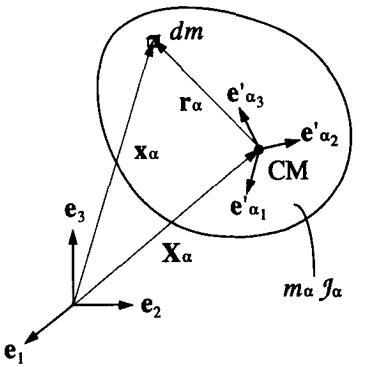


Figure 3.5 Rigid Body

The inertia tensor is linear and symmetric. It has three orthonormal principal vectors $\mathbf{e'}_{\alpha_k}$ (k=1,2,3) satisfying the equation

$$\mathcal{J}_\alpha \mathbf{e'}_{\alpha_k} = I_{\alpha_k} \mathbf{e'}_{\alpha_k} \tag{3.77}$$

for the principal values I_{α_k} (k=1,2,3). The principal vectors specify directions in a fixed relation to the rigid body "α".

Remark 3.7:

To obtain the simplest expressions for the dynamical equations, we assume that the body reference frame $\mathbf{e'}_{\alpha_k}$ (k=1,2,3) *always* has its origin at the center of mass of body "α" pointing in the directions of the three principal axes of the inertia tensor.

Equation (3.76) can be written in terms of I_{α_k} as

$$\mathcal{J}_\alpha \mathbf{a} = \sum_{k=1}^{3} \mathcal{J}_\alpha \mathbf{e'}_{\alpha_k} (\mathbf{e'}_{\alpha_k} \cdot \mathbf{a}) = \sum_{k=1}^{3} I_{\alpha_k} \mathbf{e'}_{\alpha_k} \mathbf{e'}_{\alpha_k} \cdot \mathbf{a} \,. \tag{3.78}$$

In case of axially symmetric bodies the numerical effort to evaluate $\mathcal{J}_\alpha \mathbf{a}$ can further be reduced. Assume that I_{α_1} is the principal moment of inertia in the direction of the symmetry axis $\mathbf{e}'_{\alpha_1}$. The principal moment for all directions perpendicular to $\mathbf{e}'_{\alpha_1}$ is I_{α_2}. Depending on the symmetry axis $\mathbf{e}'_{\alpha_1}$ the expression $\mathcal{J}_\alpha \mathbf{a}$ can be written as

$$\mathcal{J}_\alpha \mathbf{a} = \mathcal{J}_\alpha \mathbf{e}'_{\alpha_1} \mathbf{e}'_{\alpha_1} \mathbf{a} = \mathcal{J}_\alpha \mathbf{e}'_{\alpha_1} (\mathbf{e}'_{\alpha_1} \cdot \mathbf{a} + \mathbf{e}'_{\alpha_1} \wedge \mathbf{a}) . \tag{3.79}$$

The term $\mathbf{e}'_{\alpha_1} (\mathbf{e}'_{\alpha_1} \wedge \mathbf{a})$ represents the projection of $\mathbf{a}$ onto the plane orthogonal to $\mathbf{e}'_{\alpha_1}$ (Example 2.1). Hence,

$$\mathcal{J}_\alpha \mathbf{e}'_{\alpha_1} (\mathbf{e}'_{\alpha_1} \wedge \mathbf{a}) = I_{\alpha_2} \mathbf{e}'_{\alpha_1} (\mathbf{e}'_{\alpha_1} \wedge \mathbf{a}) . \tag{3.80}$$

With (3.77), (3.80), and the relation established in Example 2.4 equation (3.79) can therefore be transformed into the following simplified form for axially symmetric bodies:

$$\begin{aligned} \mathcal{J}_\alpha \mathbf{a} &= I_{\alpha_1} \mathbf{e}'_{\alpha_1} \mathbf{e}'_{\alpha_1} \cdot \mathbf{a} + I_{\alpha_2} \mathbf{e}'_{\alpha_1} \mathbf{e}'_{\alpha_1} \wedge \mathbf{a} \\ &= I_{\alpha_1} \mathbf{e}'_{\alpha_1} \mathbf{e}'_{\alpha_1} \cdot \mathbf{a} + I_{\alpha_2} (\mathbf{e}'_{\alpha_1} \cdot (\mathbf{e}'_{\alpha_1} \wedge \mathbf{a}) + \mathbf{e}'_{\alpha_1} \wedge (\mathbf{e}'_{\alpha_1} \wedge \mathbf{a})) \\ &= I_{\alpha_1} \mathbf{e}'_{\alpha_1} \mathbf{e}'_{\alpha_1} \cdot \mathbf{a} + I_{\alpha_2} (\mathbf{a} - \mathbf{e}'_{\alpha_1} \cdot \mathbf{a}\, \mathbf{e}'_{\alpha_1}) \\ &= I_{\alpha_2} \mathbf{a} + (I_{\alpha_1} - I_{\alpha_2}) \mathbf{e}'_{\alpha_1} \mathbf{e}'_{\alpha_1} \cdot \mathbf{a} . \end{aligned} \tag{3.81}$$

The *state variables* specify the translational and rotational motion of the body relative to the inertial frame $\{\mathbf{e}_k\}$. The translational motion of body "α" is given by the position $\mathbf{X}_\alpha$ of the center of mass and its velocity $\dot{\mathbf{X}}_\alpha$. On the other hand, the rotational motion is defined by the unitary spinor R_α describing the attitude of the body and by the rotational velocity ω_α. These variables completely characterize the motion of every particle of the rigid body if we know its relative position $\mathbf{r}'_\alpha$ in the body fixed, primed reference frame $\{\mathbf{e}'_{\alpha_k}\}$ (cf. Figure 3.5):

$$\mathbf{x}_\alpha = \mathbf{X}_\alpha + \mathcal{R}_\alpha \mathbf{r}'_\alpha = \mathbf{X}_\alpha + \mathbf{r}_\alpha , \tag{3.82}$$

$$\dot{\mathbf{x}}_\alpha = \dot{\mathbf{X}}_\alpha + \dot{\mathcal{R}}_\alpha \mathbf{r}'_\alpha + \mathcal{R}_\alpha \dot{\mathbf{r}}'_\alpha = \dot{\mathbf{X}}_\alpha + \omega_\alpha \times \mathbf{r}_\alpha . \tag{3.83}$$

The dynamical equations describing the changes of the momentum $\mathbf{P}$ and the angular momentum $\mathbf{l}$ in dependence of the forces and torques acting on a body are now rather simple to establish. For a body "α", momentum $\mathbf{P}$ and angular momentum $\mathbf{l}$ are defined as

$$\mathbf{P}_\alpha = m_\alpha \dot{\mathbf{X}}_\alpha \tag{3.84}$$

and

$$\mathbf{l}_\alpha = J_\alpha \omega_\alpha . \tag{3.85}$$

The *Newton and Euler laws* state that the first time derivative of the momentum and the angular momentum of a body "α" are equivalent to the resultant torque Γ_α and force $\mathbf{F}_\alpha$ acting on this body. Thus

$$J_\alpha \dot{\omega}_\alpha + \omega_\alpha \times J_\alpha \omega_\alpha = \Gamma_\alpha \tag{3.86}$$

and

$$m_\alpha \ddot{\mathbf{X}}_\alpha = \mathbf{F}_\alpha , \tag{3.87}$$

where the resultant torque and force acting on body "α" are specified as

$$\Gamma_\alpha = \sum_a \tau_a + \sum_a \mathbf{r}_{\alpha a} \times \mathbf{f}_a \tag{3.88}$$

and

$$\mathbf{F}_\alpha = \sum_a \mathbf{f}_a + \mathbf{F}_{g\alpha} . \tag{3.89}$$

$\mathbf{F}_{g\alpha}$ stands for the gravitational force acting on body "α". All interaction variables are related to a hinge, and vector $\mathbf{r}_{\alpha a}$ defines the point of application of hinge force $\mathbf{f}_a$ on body "α" (relative to the body frame $\{\mathbf{e}'_{\alpha_k}\}$). Thus, the resulting forces and torques acting on body "α" can be given as

$$\Gamma_\alpha = -\sum_{a=1}^{n} \left(c_{\alpha a} \tau_a + c^{+}_{\alpha a} \left(\mathbf{s}_a - \mathbf{X}_\alpha \right) \times \mathbf{f}_a \right) \tag{3.90}$$

and

$$\mathbf{F}_\alpha = -\sum_{a=1}^{n} c_{\alpha a}\mathbf{f}_a + \mathbf{F}_{g\alpha} . \tag{3.91}$$

Hence, the dynamical constraints can now be expressed in their final form as

$$\mathcal{J}_\alpha \dot{\boldsymbol{\omega}}_\alpha + \boldsymbol{\omega}_\alpha \times \mathcal{J}_\alpha \boldsymbol{\omega}_\alpha + \sum_{a=1}^{n} (c_{\alpha a}\boldsymbol{\tau}_a + c_{\alpha a}^{+}(\mathbf{s}_a - \mathbf{X}_\alpha) \times \mathbf{f}_a) = 0 \tag{3.92}$$

and

$$m_\alpha \ddot{\mathbf{X}}_\alpha + \sum_{a=1}^{n} c_{\alpha a}\mathbf{f}_a - \mathbf{F}_{g\alpha} = 0 . \tag{3.93}$$

The constraint equations (3.43), (3.44), (3.48), (3.49), (3.55), (3.56), (3.57), (3.58), (3.63), (3.64), (3.65), (3.66), (3.71), (3.74), (3.92), and (3.93) will be the basis for the modeling and control of robot manipulators to be described in Chapter 5.

Chapter 4

Control Concepts

A controller needs to make a fast and correct assessment of a situation and take an appropriate decision. Appropriate means that it should take the control system closer to the solution of the given problem. To do so the controller is in need of some information. Precision and amount of the required information are determined by the problem.

A problem consists of a result to be achieved and the corresponding desired accuracy. The most effective action to be taken is always the action that solves the given problem with as little effort as possible. In determining what the least possible effort is, we need to distinguish between off-line costs, e.g., for modeling, controller design, or evaluation of the demanded trajectory, and on-line costs to evaluate the control signal based on the actual sensor information.

In general, controllers relying more strongly on off-line computations require simpler control laws. But such controllers can only be as good as the assumptions upon which they rely. Usually they are not able to adapt to the momentary situation or react to unforeseen events. If the demanded trajectory is completely specified, the robot system is deprived of the necessary degrees of freedom to, say, avoid collisions with unexpected objects. Thus, care has to be taken not to over-specify a control problem. The other approach consists in referring more strongly to the given sensor information having more flexible controllers as a consequence. Of course, this procedure

results in higher on-line computational costs to process the sensor information in a sophisticated way.

To simplify the problem definition and thus the solution, it is useful to divide the main problem into subproblems which are easier to formulate and to solve. This may lead to a less than optimal solution of the overall problem, since every subproblem is solved separately, without taking the other subproblems into account. On the other hand, considerable time gains in the solution of these simpler subproblems may be expected. Therefore, in trying to find the best compromise, the inefficiency caused by inexact problem solving must be carefully balanced against the inefficiency produced by time losses due to the evaluation of more complex problems.

The decomposition of the main problem into subproblems which depend on each other leads to a hierarchical structure of the controller. There are two main advantages of such a controller structure. The first is its adaptability resulting in more cost effective controllers. The hierarchical structure permits the addition of as many extra elements as desired, introducing other subproblems to be solved. Thus, it is possible to increase the complexity of the controller step by step until the overall problem can be resolved with the desired accuracy and minimal effort. The second advantage is that this hierarchical structure clearly shows the interdependencies of the different subproblems and designates those subproblems that can be evaluated in parallel. The proposed hierarchical structure is introduced in Section 4.1.

A question of major importance is how such a general subproblem or task can be defined in the most suitable and simple manner. The answer is that this can be achieved best by using a quadratic performance index; its minimum defines the solution of the task. This approach allows the task to be circumscribed by an arbitrary number of constraints to be kept and by their relative importance or accuracy. It will be shown that this is a significant advantage, e.g., in connection with singularities, redundancies, or unilateral constraints. This is explained in Section 4.2.

Finally, Section 4.3 introduces an efficient way to evaluate the minimum of such a performance index. For this purpose, a numerically very robust algorithm, the so-called *Dyadic Reduction*, is used. Since it is capable of finding a solution for determined as well as over- and underdetermined problems, there is no need to further differentiate between these three cases. Furthermore, this algorithm is very suitable for parallel execution, off-line optimization, and sparse representation. The off-line optimization and the sparse representation are described in Sections 4.3.3 and 4.3.4, respectively.

4.1 Structure

Many technical problems are too complicated to be solved directly. They have to be decomposed into simpler subproblems with known interrelations. This decomposition of the overall problem into subproblems which are solved separately but depend on each other induces a hierarchical structure. Various controllers with hierarchical structure have been introduced. G. N. Saridis [Saridi83], [Saridi84], [Saridi85] describes such controllers in the context of hierarchical intelligent control. The proposed controller consists of three basic levels. All levels are measured by their entropies and their entropy rates. Entropy is defined as a measure of the uncertainty of information transmission [Shanno48]. The control problem is reformulated as finding a control sequence such that the total entropy is minimized. J. S. Albus et al. [Albus81], [AlMLBa85] suggest a hierarchical controller with a tree structure consisting of an arbitrary number of levels. The goals and plans of every level are always passed on to the next lower level where they are again decomposed into subgoals and subtasks. The decisions at each level are based on the sensor data processed, information about the state of the lower hierarchical levels, and some local knowledge.

The proposed controller uses elements of both approaches. It is a discrete element possessing a rather general structure as depicted in Figure 4.1. A control cycle operates as follows: at the sampling time t_k, the actual sensor

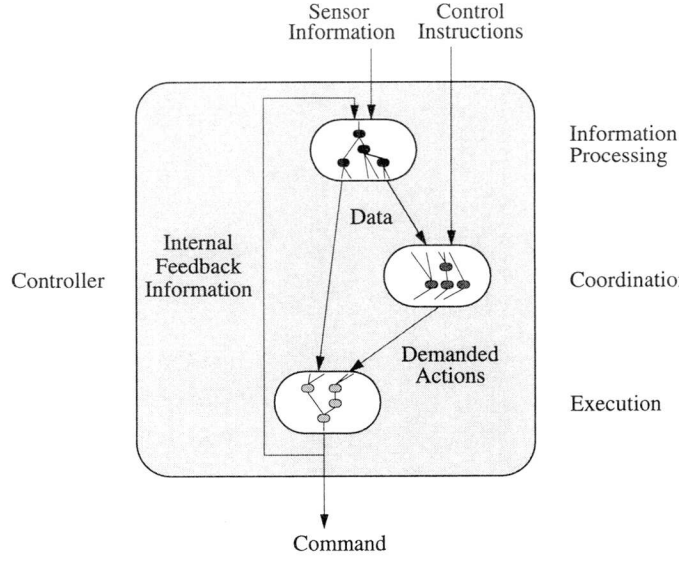


Figure 4.1 Controller Structure

information and the control instructions are sent to the controller, while the output signals for the actuators, denoted with *Command* in Figure 4.1, are emitted. For the evaluation of the next command there exist three information sources designated with *Sensor Information*, *Control Instructions,* and *Feedback Information*. The sensor information includes all measurements of the control system, i.e., all "external" feedback information, whereas the control instructions represent the feedforward information. In addition, there is an internal feedback path containing data such as the momentarily valid command and its estimated consequences. Based on this information the controller computes the command for the next sampling time t_{k+1}. The information flow in this evaluation process is strictly top down. It can be decomposed into three major processing levels referred to as *Information Processing*, *Coordination*, and *Execution*.

The first level, *Information Processing*, is responsible for the extrapolation and the improvement of the estimates of the actual control system state, utilizing the sensor and internal feedback information available. Thus, its output *Data* represents the complete knowledge about the system state accessible to the rest of the controller. The elements of the *Coordination* level evaluate the currently most appropriate strategy, or the *Demanded Actions*, based on the control instructions and the data vector from the information processing level. The *Execution* level, finally, tries to accomplish the demanded actions by proper choice of the *Command* vector making use of the data vector. Moreover, this level evaluates an estimate of the expected consequences of this command vector.

In the simplest case, when complete and sufficiently accurate sensor information and all necessary feedforward information are available, the hierarchic levels of information processing and coordination can be dispensed with. If, however, the errors in the feedback information exceed certain tolerance limits, or if there is a need for the extrapolation of some system states, these errors have to be reduced by the module information processing. The coordination level has to be introduced whenever the control instructions do not describe the task sufficiently or if they are inaccurate.

As illustrated in Figure 4.1, all three levels are composed of an arbitrary number of nodes symbolizing the subproblems or tasks. It is assumed that their interdependencies are constant and, therefore, that the hierarchical structure is rigid.

As a next step, an efficient way needs to be found to define the input/output behavior of such a node or, in other words, to characterize the task of a single node.

4.2 Task Definition

Every node (cf. Figure 4.2) in the hierarchical structure performs a mapping from the input vector $\underline{y}$ onto the output vector $\underline{\tilde{y}}$. From now on, this mapping will also be referred to as node task or simply *task*. Of course, there are numerous ways to specify such an input/output mapping. In the course of this section the different possibilities for the task specification are reviewed. It is our aim to develop a method to define the relation between the node input and output in a general, simple, and efficient way.

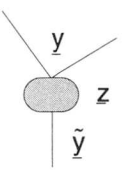


Figure 4.2 Node

A task is specified by the internal knowledge or world model relating input and output data in some way, e.g., goals to be attained, constraints to be kept, or rules to be followed. Based on this knowledge and the actual input $\underline{y}$ the output $\underline{\tilde{y}}$ is evaluated. This mapping is not necessarily one-to-one, i.e., more than one input vector $\underline{y}$ resulting in the same output vector $\underline{\tilde{y}}$ and vice versa is possible. This is the case, for example, if the internal knowledge does not suffice to determine an unequivocal output. On the other hand it is conceivable that the world model is contradictory. Thus, it is very important to find a way for the task definition guaranteeing the input/output mapping in any case.

The evaluation of the output is subdivided into two steps. In the first step the *internal variables* $\underline{z}$ are determined based on the internal knowledge and the input information. The second step is only responsible for the mapping

of these internal variables onto the output vector, thus selecting the internal variables of interest or performing additional simple transformations.

4.2.1 Explicit Task Definition

In the simplest case the internal knowledge is expressed in explicit form, i.e., the internal variables are given explicitly in terms of the input vector. Thus we can write

$$\begin{aligned} \underline{z} &= \underline{f}(\underline{y}) \\ \tilde{\underline{y}} &= \underline{g}(\underline{z}) . \end{aligned} \tag{4.1}$$

Example 4.1

Formulation of the forward kinematical problem for the cart pole system depicted in Figures 5.1 and 5.2. In this case internal knowledge about the relation between input information $\underline{y} = [\varphi'_{22_3}\ \dot{s}'_{11_1}\ \theta'_{22_3}]^T$ and output information $\tilde{\underline{y}} = [\dot{X}_{2_1}\ \dot{X}_{2_2}]^T$ is represented by the explicit equations

$$\underline{z} = \begin{bmatrix} \dot{X}_{2_1} \\ \dot{X}_{2_2} \end{bmatrix} = \underline{f}(\underline{y}) = \begin{bmatrix} 1 & -l\cos\varphi'_{22_3} \\ 0 & -l\sin\varphi'_{22_3} \end{bmatrix} \begin{bmatrix} \dot{s}'_{11_1} \\ \theta'_{22_3} \end{bmatrix}, \tag{4.2}$$

$$\tilde{\underline{y}} = \underline{z}.$$

4.2.2 Implicit Task Definition

Quite often it is much more straightforward to specify the internal knowledge in linear, implicit form

$$\begin{aligned} \underline{A}(\underline{y})\ \underline{z} + \underline{b}(\underline{y}) &= 0 \\ \tilde{\underline{y}} &= \underline{g}(\underline{z}) \end{aligned} \tag{4.3}$$

where $\mathbf{A}$ is an $(m \times n)$ matrix. Thus, the internal knowledge is represented with m linear, implicit equations in n variables.

Example 4.2

Formulation of the inverse kinematical problem for the system depicted in Figure 5.1, i.e., evaluation of $\tilde{\underline{y}} = [\dot{s}'_{11_1}\ \theta'_{22_3}]^T$ in dependence of $\underline{y} = [\varphi'_{22_3}\ \dot{X}_{2_1}\ \dot{X}_{2_2}]^T$. The relation between $\tilde{\underline{y}}$ and $\underline{y}$ can be given as

$$\mathbf{A}(\underline{y})\,\underline{z} + \underline{b}(\underline{y}) = \begin{bmatrix} 1 & -l\cos\varphi'_{22_3} \\ 0 & -l\sin\varphi'_{22_3} \end{bmatrix} \begin{bmatrix} \dot{s}'_{11_1} \\ \theta'_{22_3} \end{bmatrix} + \begin{bmatrix} -\dot{X}_{2_1} \\ -\dot{X}_{2_2} \end{bmatrix} = 0 \tag{4.4}$$

$$\tilde{\underline{y}} = \underline{z}.$$

Example 4.3

The goal consists of moving mass 2 of the plant depicted in Figure 5.1 on a circle with radius r and arbitrary velocity. This can be expressed rather simply by the restriction

$$X_{2_1}^2 + X_{2_2}^2 = r^2. \tag{4.5}$$

This nonlinear constraint can be approximated by the following linear, implicit equation

$$2\begin{bmatrix} X_{2_1} & X_{2_2} \end{bmatrix} \begin{bmatrix} \Delta X_{2_1} \\ \Delta X_{2_2} \end{bmatrix} + X_{2_1}^2 + X_{2_2}^2 - r^2 = 0. \tag{4.6}$$

Of course, (4.3) can be transformed into the explicit task formulation, if the matrix $\mathbf{A}$ is invertible. Thus, we can write

$$\underline{z} = \underline{f}(\underline{y}) = -\mathbf{A}^{-1}(\underline{y})\,\underline{b}(\underline{y}). \tag{4.7}$$

This is the case if $\mathbf{A}$ is non-singular. For $\sin\varphi'_{22_3} = 0$, for example, this is not true in (4.4). In this singular position the problem is either overdetermined ($\dot{X}_{2_2} \neq 0$) or underdetermined ($\dot{X}_{2_2} = 0$). In the first case, it is no

longer possible to find a solution for the equations, whereas in the second case, there exist an infinite number of such solutions. In contrast to this, equation (4.6) from Example 4.3 is always underdetermined for $\Delta X_{2_1}, \Delta X_{2_2}$.

Solutions have to be found now which represent an acceptable compromise in the overdetermined case and in a sophisticated manner take advantage of the extra degrees of freedom in the underdetermined case.

One way to solve the linear, implicit equation (4.3) in the general case (i.e., for $m \neq n$) is through the use of the generalized inverse [BenGre80]. This approach is based upon the work done by E. H. Moore ([Moore20], [Moore35]) and independently by R. Penrose ([Penros55], [Penros56]). That is why the generalized inverse is also referred to as *Moore-Penrose-Inverse*. It uses a more general definition for the inverse A^+ of a matrix A postulating that the following equation must be fulfilled:

$$\mathsf{A}\mathsf{A}^+\mathsf{A} = \mathsf{A}. \tag{4.8}$$

For a square, non-singular matrix A, it is obvious that $\mathsf{A}^+ = \mathsf{A}^{-1}$. The evaluation of the generalized inverse depends on the rank of the matrix to be inverted. Usually this rank needs to be evaluated numerically. This is indeed the weakest point of this method [Zielke83]. First, this results in additional computational costs. Second, there may be severe numerical problems in deciding upon the rank if the matrix is close to changing its rank. Furthermore, supplementary transformations are necessary in cases where the matrix to be inverted does not possess full column or row rank. That is why this approach is mainly feasible for problems with constant and clearly determined matrix rank.

Of course the examples given above are rather simple. But this type of linear, implicit equations changing from overdetermined to underdetermined and vice versa is encountered quite frequently in robotics. It mainly occurs in connection with redundancies, singularities, and unilateral

constraints. Therefore, the pseudo-inverse approach is unsuitable for this kind of problems.

4.2.3 Task Definition Using a Quadratic Performance Index

Another possibility for the representation of the internal knowledge is the use of a quadratic performance index transforming (4.4) into a minimization problem. Instead of postulating $\underline{z}$ to be the exact solution of a linear problem

$$\mathbf{A}(\underline{y})\,\underline{z}+\underline{b}\,(\underline{y}) = \left[\mathbf{A}(\underline{y})\;\; \underline{b}\,(\underline{y})\right]\begin{bmatrix}\underline{z}\\ 1\end{bmatrix} = \mathbf{B}\,(\underline{y})\begin{bmatrix}\underline{z}\\ 1\end{bmatrix} = 0., \tag{4.9}$$

$\underline{z}$ is chosen such that a quadratic performance index $J\,(\underline{z}, \underline{y})$ is minimized:

$$\begin{aligned} J\,(\underline{z}, \underline{y}) &= \tfrac{1}{2}\,(\mathbf{A}(\underline{y})\,\underline{z}+\underline{b}\,(\underline{y}))^{\mathrm{T}}\,\mathbf{D}\,(\underline{y})\,(\mathbf{A}(\underline{y})\,\underline{z}+\underline{b}\,(\underline{y})) \\ &= \tfrac{1}{2}\left[\underline{z}^{\mathrm{T}}\;1\right]\mathbf{B}^{\mathrm{T}}\,(\underline{y})\,\mathbf{D}\,(\underline{y})\,\mathbf{B}\,(\underline{y})\begin{bmatrix}\underline{z}\\ 1\end{bmatrix}. \end{aligned} \tag{4.10}$$

$\mathbf{D}$ stands for a diagonal, positive-definite $(\mathrm{m} \times \mathrm{m})$-weighting matrix

$$\mathbf{D}\,(\underline{y}) = \mathbf{D}\,(\underline{y})^{\mathrm{T}} > 0, \tag{4.11}$$

i.e., a matrix with positive diagonal elements. The matrix $\mathbf{B}\,(\underline{y})$ contains the complete internal knowledge of the task, every row of $\mathbf{B}\,(\underline{y})$ standing for a rule, a constraint, or a goal. However, besides through the implicit world representation in (4.3), the importance or reliability of such a rule can also be specified with the corresponding element in the diagonal matrix $\mathbf{D}\,(\underline{y})$. Thus it is a simple step to introduce redundant rules into the world

model, governing their domain of influence with the corresponding elements of the weighting matrix $\mathsf{D}(\underline{y})$. This can be done rather smoothly as the weighting factors are updated for every control cycle. It was found that this redundant world modeling improves the robustness of the model. Besides, it allows to further decompose the internal knowledge into locally valid, simpler models.

Example 4.4

Solution of the linear problem from Example 4.2:

$$\begin{bmatrix} 1 & -l\cos\varphi'_{22_3} \\ 0 & -l\sin\varphi'_{22_3} \end{bmatrix} \begin{bmatrix} \dot{s}'_{11_1} \\ \theta'_{22_3} \end{bmatrix} + \begin{bmatrix} -\dot{X}_{2_1} \\ -\dot{X}_{2_2} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{z}}_1 & \hat{\mathbf{z}}_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = 0. \tag{4.12}$$

For a constant value of φ'_{22_3} this problem can be visualized by the substitute mechanical system depicted in Figure 4.3 possessing two translational joints in directions $\hat{\mathbf{z}}_1$ and $\hat{\mathbf{z}}_2$, z_1 and z_2 now representing positional values instead of velocities.

The problem consists of finding the values for z_1 and z_2 such that the tip P of the vector $z_1\hat{\mathbf{z}}_1 + z_2\hat{\mathbf{z}}_2 + \mathbf{b}$ points to the origin of the inertial frame. This is equivalent to the request to minimize the potential energy of the two springs shown in Figure 4.3. The second formulation corresponds to the task definition using a performance index, the performance index representing the potential energy and the spring constants standing for the weighting factors. Obviously, both problem formulations lead to the same result.

Remark 4.1:

From now on it is assumed without explicitly writing it anymore that A, B, $\underline{\mathsf{b}}$, and D depend on the input data $\underline{y}$.

Since $J(\underline{z}, \underline{y})$ is a positive, quadratic function in $\underline{z}$, it possesses one extremum only. This global minimum is defined by

$$\frac{\partial}{\partial \underline{z}} J(\underline{z}, \underline{y}) = 0^T \tag{4.13}$$

$$\begin{aligned} \frac{\partial}{\partial \underline{z}} J(\underline{z}, \underline{y}) &= \frac{\partial}{\partial \underline{z}} \left(\frac{1}{2} \begin{bmatrix} \underline{z}^T & 1 \end{bmatrix} B^T D B \begin{bmatrix} \underline{z} \\ 1 \end{bmatrix} \right) \\ &= \begin{bmatrix} \underline{z}^T & 1 \end{bmatrix} \begin{bmatrix} A & \underline{b} \end{bmatrix}^T D A = \underline{z}^T A^T D A + \underline{b}^T D A = 0^T, \end{aligned} \tag{4.14}$$

leading to the solution

$$\underline{z} = -(A^T D A)^{-1} A^T D \underline{b}. \tag{4.15}$$

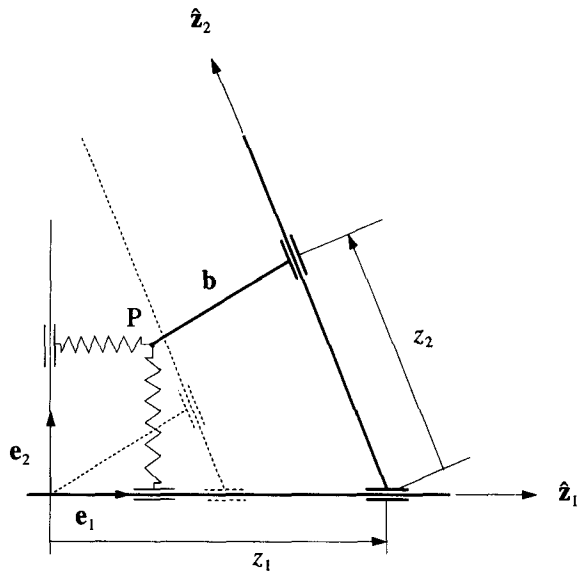


Figure 4.3 Arbitrary Position

Of course, this only holds if the matrix A possesses full column rank. Otherwise matrix $A^T D A$ is not invertible. Thus, we still have to evaluate the

rank of A to decide whether a solution to the problem exists. Furthermore, in accordance with assumption (4.11) a constraint may not be deactivated in that the corresponding weighting factor is set to zero.

To avoid this we add n additional explicit equations embodied in the matrix B_0. This leads to an expansion of the matrices B and D which are now of dimension $(m+n) \times (n+1)$ and $(m+n) \times (m+n)$, respectively. They have the following structures

$$\mathsf{B} = \begin{bmatrix} \mathsf{B}_0 \\ \mathsf{B}_1 \end{bmatrix} = \begin{bmatrix} \mathsf{E} & -\underline{z}_d \\ \mathsf{A} & \underline{b} \end{bmatrix}, \tag{4.16}$$

and

$$\mathsf{D} = \begin{bmatrix} \mathsf{D}_0 & 0 \\ 0 & \mathsf{D}_1 \end{bmatrix}, \tag{4.17}$$

where E stands for the unit matrix and $\underline{z}_d$ represents the demanded values for $\underline{z}$. Thus we get the following extended performance index

$$J(\underline{z}, \underline{y}) = \frac{1}{2} \begin{bmatrix} \underline{z}^T & 1 \end{bmatrix} \mathsf{Q} \begin{bmatrix} \underline{z} \\ 1 \end{bmatrix}, \tag{4.18}$$

with

$$\mathsf{Q} = \mathsf{B}^T \mathsf{D} \mathsf{B} = \begin{bmatrix} \mathsf{E} & -\underline{z}_d \\ \mathsf{A} & \underline{b} \end{bmatrix}^T \begin{bmatrix} \mathsf{D}_0 & 0 \\ 0 & \mathsf{D}_1 \end{bmatrix} \begin{bmatrix} \mathsf{E} & -\underline{z}_d \\ \mathsf{A} & \underline{b} \end{bmatrix}, \tag{4.19}$$

$$\begin{aligned} &\mathsf{D}_0 = \mathsf{D}_0{}^T > 0, \text{ and} \\ &\mathsf{D}_1 = \mathsf{D}_1{}^T \geq 0. \end{aligned} \tag{4.20}$$

With this reformulation the problem can no longer become underdetermined. The internal knowledge is composed of an explicit and an implicit part. Even in the worst case, where $\mathsf{D}_1 = 0$, i.e., when the implicit world

model is absolutely unreliable, the solution is still given as $\underline{z} = \underline{z}_d$. On the other hand, choosing the elements of D_1 large enough in relation to the elements of D_0, the constraint equations can always be satisfied with the desired accuracy.

Example 4.5

Expanded B matrix for Example 4.3:

$$B = \begin{bmatrix} E & -\underline{z}_d \\ A & \underline{b} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2X_{2_1} & 2X_{2_2} & X_{2_1}^2 + X_{2_2}^2 - r^2 \end{bmatrix} \qquad \underline{z} = \begin{bmatrix} \Delta X_{2_1} \\ \Delta X_{2_2} \end{bmatrix}. \tag{4.21}$$

If $D_0 = E$, $\underline{z}$ is chosen in the direction of the closest point on the circle, as it is postulated to minimize the length of $\underline{z}$ and to satisfy the equation specifying the circle at the same time.

Example 4.6

Expanded B matrix for Example 4.2. As mentioned earlier, difficulties may arise close to singularities or in singularities. Figure 4.4 depicts such a situation ($\varphi'_{22_3} \approx \pi$):

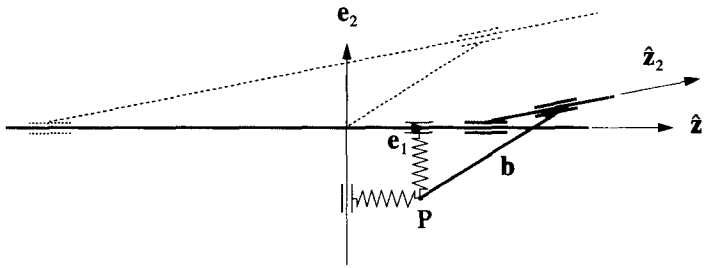


Figure 4.4 Close to Singular Position

It is evident that the values of z_1 and z_2 approach infinity if the two vectors $\hat{\mathbf{z}}_1$ and $\hat{\mathbf{z}}_2$ become parallel. Using the following performance index

$$B = \begin{bmatrix} E & -z_d \\ A & b \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & -z_{\min_1} \\ 1 & -l\cos\varphi'_{22_3} & -\dot{X}_{2_1} \\ 0 & -l\sin\varphi'_{22_3} & -\dot{X}_{2_2} \end{bmatrix}, \qquad z = \begin{bmatrix} \dot{s}'_{11_1} \\ \theta'_{22_3} \end{bmatrix}, \tag{4.22}$$

this can be avoided. The extended performance index defines the potential energy of the system in Figure 4.5, the additional constraints being represented by supplementary springs. Of course, equation (4.12) can no longer be satisfied completely since the problem is over-determined now. Hence, the solution can be regarded as the best compromise to the extended problem only. On the other hand, the solution is now numerically very robust and well-defined, even in the singular position. Notice that one of the springs defines a unilateral constraint being active for z_1 less than $z_{\min_1}$ only. Thus, this is a straightforward approach to model time and state dependent constraints as well.

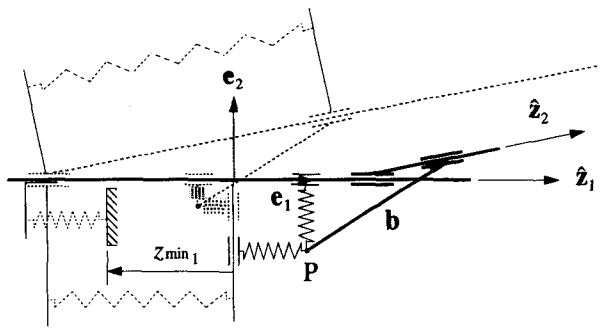


Figure 4.5 Close to Singular Position with Additional Constraints

The minimum, defined by (4.13), now has to satisfy the following equation

$$\begin{aligned}\frac{\partial}{\partial \underline{z}} J(\underline{z}, \underline{y}) &= \frac{\partial}{\partial \underline{z}}\left(\frac{1}{2}\begin{bmatrix}\underline{z}^T & 1\end{bmatrix}\begin{bmatrix}E & -\underline{z}_d\\ A & \underline{b}\end{bmatrix}^T\begin{bmatrix}D_0 & 0\\ 0 & D_1\end{bmatrix}\begin{bmatrix}E & -\underline{z}_d\\ A & \underline{b}\end{bmatrix}\begin{bmatrix}\underline{z}\\ 1\end{bmatrix}\right)\\ &= \begin{bmatrix}\underline{z}^T & 1\end{bmatrix}\begin{bmatrix}E & -\underline{z}_d\\ A & \underline{b}\end{bmatrix}^T\begin{bmatrix}D_0 & 0\\ 0 & D_1\end{bmatrix}\begin{bmatrix}E\\ A\end{bmatrix}\\ &= \underline{z}^T(D_0 + A^T D_1 A) + (-\underline{z}_d^T D_0 + \underline{b}^T D_1 A) = 0^T\end{aligned} \tag{4.24}$$

or, in explicit notation,

$$\underline{z} = (D_0 + A^T D_1 A)^{-1}(D_0 \underline{z}_d - A^T D_1 \underline{b}). \tag{4.25}$$

If the matrix D_0 is positive-definite, matrix $D_0 + A^T D_1 A$ will always be positive-definite as well. Thus, it is always possible to invert this matrix. This can be done either analytically or numerically. Experience shows that it is only feasible to work analytically for $n \leq 4$. For those cases in which $n > 4$ it is recommended to find the solution to (4.24) numerically.

The approach presented above allows the solution of arbitrary linear, implicit problems with the same algorithm using an extended quadratic performance index. It is no longer necessary to find out whether the implicit problem is determined, overdetermined or underdetermined. Thus, costly numerical evaluations of the rank of the matrix A can be avoided. The explicit equations improve the robustness of the algorithm as they roughly outline where the solution should be located. They thus prevent the solution from grossly deviating from $\underline{z}_d$ which might happen close to singular positions. An example for the control of a walking robot using such an extended performance index for the task definition can be found in [BusGee91].

A final problem using this approach is the numerically efficient evaluation of the matrix $(D_0 + A^T D_1 A)^{-1}$. This is especially true in conjunction with large, sparse A matrices. Therefore, the next section presents an algorithm particularly adapted to problems with such matrices.

4.3 Dyadic Reduction

This section introduces a numeric algorithm well-suited for finding the minimum of an extended quadratic performance index of type (4.18). It was found in [Peterk86], where it is introduced as *Dyadic Reduction*. In contrast to the former approach (cf. (4.18)), the matrix D_0 ought to be positive semi-definite only, i.e,

$$D_0 = D_0^T \geq 0 \tag{4.26}$$

or, in other words, this algorithm is also capable of finding a solution for underdetermined problems. With (4.26) it is theoretically possible to completely deactivate the explicit equations in the performance index and still guarantee that the algorithm finds a solution. However, this has to be done with great care in order to avoid misfits of the solution close to or in singularities.

While the algorithm is numerically very robust, it also is very suitable for large linear and sparse problems. This is due to the fact that it can be optimized off-line by using the position of the non-zero elements of matrix B as the only information. This is explained in Section 4.3.3. After this optimization the complete algorithm can be specified in sparse representation, i.e., it is possible to avoid mathematical operations with zero elements and to store only the matrix elements used in the evaluation of the solution. This is described in Section 4.3.4.

4.3.1 Basic Idea

It is assumed that matrices B and D forming matrix Q (cf. (4.18) and (4.19)) can be transformed in the following manner

$$
\begin{aligned}
B^T D B &= \begin{bmatrix} B_0 \\ B_1 \end{bmatrix}^T \begin{bmatrix} D_0 & 0 \\ 0 & D_1 \end{bmatrix} \begin{bmatrix} B_0 \\ B_1 \end{bmatrix} = \begin{bmatrix} E & -\underline{z}_d \\ A & \underline{b} \end{bmatrix}^T \begin{bmatrix} D_0 & 0 \\ 0 & D_1 \end{bmatrix} \begin{bmatrix} E & -\underline{z}_d \\ A & \underline{b} \end{bmatrix} \\
&= \begin{bmatrix} U & -\tilde{\underline{z}}_d \\ 0 & \tilde{\underline{b}} \end{bmatrix}^T \begin{bmatrix} \tilde{D}_0 & 0 \\ 0 & \tilde{D}_1 \end{bmatrix} \begin{bmatrix} U & -\tilde{\underline{z}}_d \\ 0 & \tilde{\underline{b}} \end{bmatrix} = \begin{bmatrix} \tilde{B}_0 \\ \tilde{B}_1 \end{bmatrix}^T \begin{bmatrix} \tilde{D}_0 & 0 \\ 0 & \tilde{D}_1 \end{bmatrix} \begin{bmatrix} \tilde{B}_0 \\ \tilde{B}_1 \end{bmatrix} \\
&= \tilde{B}^T \tilde{D} \tilde{B}.
\end{aligned} \tag{4.27}
$$

U is a $n \times n$ monic, upper triangular matrix. Monic means that all diagonal elements equal one. Thus, the performance index can now be written as

$$
\begin{aligned}
J(\underline{z}, \underline{y}) &= \frac{1}{2} \begin{bmatrix} \underline{z}^T & 1 \end{bmatrix} B^T D B \begin{bmatrix} \underline{z} \\ 1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \underline{z}^T & 1 \end{bmatrix} \tilde{B}^T \tilde{D} \tilde{B} \begin{bmatrix} \underline{z} \\ 1 \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} \underline{z}^T & 1 \end{bmatrix} \begin{bmatrix} U & -\tilde{\underline{z}}_d \\ 0 & \tilde{\underline{b}} \end{bmatrix}^T \begin{bmatrix} \tilde{D}_0 & 0 \\ 0 & \tilde{D}_1 \end{bmatrix} \begin{bmatrix} U & -\tilde{\underline{z}}_d \\ 0 & \tilde{\underline{b}} \end{bmatrix} \begin{bmatrix} \underline{z} \\ 1 \end{bmatrix} \\
&= \frac{1}{2} \left((U\underline{z} - \tilde{\underline{z}}_d)^T \tilde{D}_0 (U\underline{z} - \tilde{\underline{z}}_d) + \tilde{\underline{b}}^T \tilde{D}_1 \tilde{\underline{b}} \right).
\end{aligned} \tag{4.28}
$$

Therefore, the minimum as well as the corresponding solution vector $\underline{z}$ are defined by the two equations

$$
\min_{\underline{z}} \; J(\underline{z}, \underline{y}) = \frac{1}{2} (\tilde{\underline{b}}^T \tilde{D}_1 \tilde{\underline{b}}) \tag{4.29}
$$

and

$$
U\underline{z} - \tilde{\underline{z}}_d = 0. \tag{4.30}
$$

Equation (4.30) always possesses a solution. This solution is easily found by backward substitution, thus eliminating the need for a matrix inversion.

4.3.2 Algorithm

What needs to be found now is the transformation specified by (4.27). The arbitrary matrices Q and B can be written in component form as

$$\mathsf{B} = [b_{ij}] = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1n+1} \\ \dots & \dots & \dots & \dots \\ b_{N1} & b_{N2} & \dots & b_{Nn+1} \end{bmatrix} = \begin{bmatrix} \underline{b}_1^T \\ \dots \\ \underline{b}_N^T \end{bmatrix} \tag{4.31}$$

and

$$\begin{aligned} \mathsf{B}^T\mathsf{D}\mathsf{B} &= \begin{bmatrix} \underline{b}_1 & \dots & \underline{b}_N \end{bmatrix} \begin{bmatrix} d_1 & \dots & \dots \\ \dots & \dots & \dots \\ \dots & \dots & d_N \end{bmatrix} \begin{bmatrix} \underline{b}_1^T \\ \dots \\ \underline{b}_N^T \end{bmatrix} = d_1\underline{b}_1\underline{b}_1^T + \dots + d_N\underline{b}_N\underline{b}_N^T \\ &= \sum_i^N d_i\underline{b}_i\underline{b}_i^T = \sum_i \mathsf{Q}_i = \sum_i [q_{i_{jk}}] = \mathsf{Q}. \end{aligned} \tag{4.32}$$

The proposed numerical algorithm proceeds as follows: It selects two matrices Q_p and Q_r. The vectors $\underline{b}_p^T$ and $\underline{b}_r^T$, also referred to as p and r rows of matrix B, have to be chosen such that they possess their first non-zero element in the same column. Additionally it is postulated that this element equals one for the $\underline{b}_p$ vector, i.e.,

$$\begin{aligned} &b_{pj} = b_{rj} = 0, && \text{for } j = 1, \dots, p-1 \\ &b_{pp} = 1, \\ &b_{rp} \neq 0. \end{aligned} \tag{4.33}$$

The aim is to find a transformation modifying the vectors $\underline{b}_p$ and $\underline{b}_r$ in such a way that the matrix sum $\mathsf{Q}_p + \mathsf{Q}_r$ remains unchanged and that the first non-zero element of row r vanishes. The matrix sum $\mathsf{Q}_p + \mathsf{Q}_r$ has to remain unchanged so that matrix Q and, therefore, the performance index are not changed. Mathematically this can be expressed as

$$
\begin{aligned}
[q_{p_{jk}}] + [q_{r_{jk}}] &= [\tilde{q}_{p_{jk}}] + [\tilde{q}_{r_{jk}}] \\
= d_p \underline{b}_p \underline{b}_p^T + d_r \underline{b}_r \underline{b}_r^T &= \tilde{d}_p \tilde{\underline{b}}_p \tilde{\underline{b}}_p^T + \tilde{d}_r \tilde{\underline{b}}_r \tilde{\underline{b}}_r^T \\
= [d_p b_{pj} b_{pk}] + [d_r b_{rj} b_{rk}] &= [\tilde{d}_p \tilde{b}_{pj} \tilde{b}_{pk}] + [\tilde{d}_r \tilde{b}_{rj} \tilde{b}_{rk}]
\end{aligned}
\tag{4.34}
$$

with

$$
\begin{aligned}
&\tilde{b}_{pj} = \tilde{b}_{rj} = 0 \qquad \text{for } j = 1, \ldots, p-1, \\
&\tilde{b}_{pp} = 1, \\
&\tilde{b}_{rp} = 0.
\end{aligned}
\tag{4.35}
$$

From (4.27) and (4.33) we can see that $p \in [1, n]$ and that $r \in [n+1, n+m]$. A reduction step with rows p and r will be symbolized with (p, r), the first index always designating the p row and the second one the r row. The transformation, preserving (4.34) and (4.35) at the same time, is defined as

$$
\begin{aligned}
&\tilde{d}_p = d_p + d_r b_{rp}^2, \\
&\tilde{d}_r = \begin{cases} (d_p / \tilde{d}_p)\, d_r & \text{if } \tilde{d}_p > \varepsilon, \\ d_r & \text{else}, \end{cases} \\
&k_r = \begin{cases} (d_r / \tilde{d}_p)\, b_{rp} & \text{if } \tilde{d}_p > \varepsilon, \\ 0 & \text{else}, \end{cases}
\end{aligned}
\tag{4.36}
$$

$$
\tilde{b}_{rj} = b_{rj} - b_{rp} b_{pj} \qquad \text{for } j = p+1, \ldots, n+1, \tag{4.37}
$$

$$
\tilde{b}_{pj} = b_{pj} + k_r \tilde{b}_{rj} \qquad \text{for } j = p+1, \ldots, n+1. \tag{4.38}
$$

The scalar ε has to be chosen in accordance with the accuracy of the computing device. The derivation of these equations can be found in Appendix E.

Remark 4.2:

Notice that the reduction step can be skipped if the element b_{rp} to be reduced turns out to be zero as this converts (4.36), (4.37), and (4.38) into $\tilde{d}_p = d_p$, $\tilde{d}_r = d_r$, $k_r = 0$, $\tilde{b}_{rj} = b_{rj}$, and $\tilde{b}_{pj} = b_{pj}$, and thus has no effect on the transformation.

Remark 4.3:

If $b_{pj} = 0$, equation (4.37) does not have to be evaluated as we have $\tilde{b}_{rj} = b_{rj}$ in this case. The same is true for equation (4.38) if $\tilde{b}_{rj} = 0$.

It is evident that this algorithm is very suitable for parallel execution as it allows the simultaneous processing of all admissible pairs $[q_{p_{jk}}]$, $[q_{r_{jk}}]$ simultaneously, i.e., of all pairs meeting condition (4.35). In a simpler form, a reduction step can be described as follows: The first non-zero element of the rows p and r must be in the same column. This element needs to be unity for the p row. Therefore, a non-zero element in column j of the p row, i.e., $b_{pj} \neq 0$, paired with a zero element $b_{rj} = 0$ in the r row always results in a new non-zero element in the r row and vice versa. Zeroes in both rows cause zero elements in these rows again. The same is true for non-zero elements. Thus, the structure of the $\tilde{\mathbf{B}}$ matrix, i.e., the position of all non-zero elements of this matrix, after one reduction step, can be determined without the exact numerical value of the matrix elements being known. The only information necessary is whether or not an element equals zero. Therefore, the complete transformation depends only on the structure of the $\mathbf{B}$ matrix and the step sequence. This allows us to optimize the complete transformation off-line. This is a major advantage over transformations that depend on the numerical values of the matrix elements.

4.3.3 Off-Line Optimization

Symbolically the reduction step (p, r) can be described as follows, x designating an arbitrary non-zero element.

$$\begin{array}{ll} \text{p row} & [0\,0\,1\,x\,x\,0\,0] \\ \text{r row} & [0\,0\,x\,0\,x\,0\,x] \end{array} \xrightarrow{(p,r)} \begin{array}{l} [0\,0\,1\,x\,x\,0\,x] \\ [0\,0\,0\,x\,x\,0\,x] \end{array} \tag{4.39}$$

It is obvious that such a transformation allows the conversion described in (4.27) from an arbitrary B matrix into $\tilde{\mathsf{B}}$ zeroing the non-zero elements of the matrix A to be achieved step by step.

Example 4.7

Transformation of matrix B from Example 4.5 is achieved in two steps. Note that by supposition (4.33) we need to start with $p = 1$.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x & x & x \end{bmatrix} \xrightarrow{(1,3)} \begin{bmatrix} 1 & x & x \\ 0 & 1 & 0 \\ 0 & x & x \end{bmatrix} \xrightarrow{(2,3)} \begin{bmatrix} 1 & x & x \\ 0 & 1 & x \\ 0 & 0 & x \end{bmatrix} \tag{4.40}$$

Example 4.8

The transformation of the subsequent B matrix (Example 4.6 without unilateral constraint) can be performed in two transformation steps with two processors instead of three transformation steps with one processor.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{(1,3)} \begin{bmatrix} 1 & x & x \\ 0 & 1 & 0 \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{(2,3)} \begin{bmatrix} 1 & x & x \\ 0 & 1 & x \\ 0 & 0 & x \\ 0 & x & x \end{bmatrix} \xrightarrow{(2,4)} \begin{bmatrix} 1 & x & x \\ 0 & 1 & x \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix} \tag{4.41}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow[(2,4)]{(1,3)} \begin{bmatrix} 1 & x & x \\ 0 & 1 & x \\ 0 & x & x \\ 0 & 0 & x \end{bmatrix} \xrightarrow{(2,3)} \begin{bmatrix} 1 & x & x \\ 0 & 1 & x \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix} \tag{4.42}$$

It follows from Remarks 4.2 and 4.3 that the numerical costs can be reduced considerably if we know the structure of the matrix B, i.e., the posi-

tion of the zero and non-zero elements. Assume that this structure is specified for $\mathsf{B} = [b_{ij}]$ by

$$\nu_{ij} = \begin{cases} 1 & \text{if } b_{ij} \neq 0 \\ 0 & \text{else} \end{cases} \qquad \begin{array}{l} i \in [1, n+m] \\ j \in [1, n+1] \end{array} \tag{4.43}$$

The structure of matrix B after the reduction step, i.e., $\tilde{\mathsf{B}}$, can then be given by

$$\tilde{\nu}_{ij} = \nu_{ij} + \Delta\nu_{ij} \qquad \begin{array}{l} i \in [1, n+m] \\ j \in [1, n+1] \end{array} \tag{4.44}$$

$$\Delta\nu_{ij} = \begin{cases} (1-\nu_{ij})\,\nu_{rj} & i = p \\ (1-\nu_{ij})\,\nu_{pj} & i = r \end{cases} \tag{4.45}$$

Additionally we introduce the following definitions:

$$\begin{aligned} \nu_i &= \sum_{j=p+1}^{n+1} \nu_{ij}, \\ \tilde{\nu}_i &= \sum_{j=p+1}^{n+1} \tilde{\nu}_{ij}. \end{aligned} \tag{4.46}$$

The total numerical costs for the reduction step (p, r) can now be given as (cf. (4.36), (4.37), (4.38), (4.43), (4.43), and (4.46)):

Additions	Subtractions	Multiplications	Divisions
$1 + \tilde{\nu}_r$	ν_p	$3 + \tilde{\nu}_r + \nu_p$	2

Table 4.1 Numerical Costs for one Reduction Step

Remark 4.4:

If the structure of B is unknown we have to calculate (4.37) and (4.38) for every column, i.e., instead of $\tilde{v}_r$ and v_p times, these equations both have to be evaluated $n - p$ times.

Example 4.9

Comparison of two different step sequences for the dyadic reduction of the same matrix using one and two processors.

$$\begin{bmatrix} 1\,0\,0 \\ 0\,1\,0 \\ x\,x\,x \\ x\,0\,x \end{bmatrix} \xrightarrow{(1,4)} \begin{bmatrix} 1\,0\,x \\ 0\,1\,0 \\ x\,x\,x \\ 0\,0\,x \end{bmatrix} \xrightarrow{(1,3)} \begin{bmatrix} 1\,x\,x \\ 0\,1\,x \\ 0\,x\,x \\ 0\,0\,x \end{bmatrix} \xrightarrow{(2,3)} \begin{bmatrix} 1\,x\,x \\ 0\,1\,x \\ 0\,0\,x \\ 0\,0\,x \end{bmatrix} \tag{4.47}$$

$$\begin{bmatrix} 1\,0\,0 \\ 0\,1\,0 \\ x\,x\,x \\ x\,0\,x \end{bmatrix} \xrightarrow{(1,3)} \begin{bmatrix} 1\,x\,x \\ 0\,1\,0 \\ 0\,x\,x \\ x\,0\,x \end{bmatrix} \xrightarrow[(2,3)]{(1,4)} \begin{bmatrix} 1\,x\,x \\ 0\,1\,x \\ 0\,0\,x \\ 0\,x\,x \end{bmatrix} \xrightarrow{(2,4)} \begin{bmatrix} 1\,x\,x \\ 0\,1\,x \\ 0\,0\,x \\ 0\,0\,x \end{bmatrix} \tag{4.48}$$

The numerical costs for the step sequences (4.47) and (4.48) are given in Table 4.2 (cf. Table 4.1). The third step sequence represents the "longer" path if (4.47) is evaluated by two processors, i.e., if steps $(1,4)$ and $(2,3)$ are carried out simultaneously. Obviously, the overall computational costs to carry out the desired transformation strongly depend on the step sequence. For step sequence (4.47) we only have about half of the computational costs of step sequence (4.47). Thus, it is worth investing in a proper selection of this step sequence.

To define the optimal step sequence, i.e., the step sequence resulting in minimal computational costs, a straightforward approach is again to use a performance index. The optimization of the complete step sequence for matrices exceeding a certain size is too expensive. Therefore, a performance

Step Sequence	+	-	*	/
(1, 4), (1, 3), (2, 3)	6	1	13	6
(1, 3), (2, 3), (1, 4), (2, 4)	10	3	21	8
(1, 3), (1, 4), (2, 4)	8	3	17	6

Table 4.2 Numerical Costs

index defining the momentarily optimal selection for the p and r rows will be formulated.

The issue now is to select the p and r rows for the next reduction step such that the supplementary numerical costs to be expected are kept as low as possible. The additional costs depend on the new non-zero elements Δv_{ij} in row i and column j, caused by the reduction step (p, r). The new non-zero elements in rows p and r are defined by (4.45). However, the new non-zero elements in the p row, Δv_{pj}, will additionally yield non-zero elements in the other rows to be reduced by this p row. Of course, a non-zero element Δv_{pj} will only result in a new non-zero element in row i and column j if this row possesses a non-zero element in column p and a zero element in column j. Henceforth, we can write for Δv_{ij} with index $j \in [p+1, n+1]$

$$\Delta v_{ij} = \begin{cases} (1 - v_{pj})\, v_{rj} & i = p \\ (1 - v_{rj})\, v_{pj} & i = r \\ (1 - v_{pj})\, v_{rj}\, v_{ip}\, (1 - v_{ij}) & i \neq r \text{ and } i \in [n+1, n+m] \end{cases} \tag{4.49}$$

Remark 4.5:

For $i \neq r$ and $i \neq p$ Δv_{ij} is only an estimate of the supplementary non-zero elements from reduction steps (p, i) to follow the actual step (p, r) as it is evaluated from the actual values for v_{ij}.

Next, we have to quantify the numerical costs resulting from these additional non-zero elements. The non-zero elements Δv_{ij} cause additional

numerical costs in the reduction step (p, r), directly, as well as in the ensuing reduction steps (p, i), with $i \in [n+1, n+m]$, except for $i = r$. Additional costs yet follow from the supplementary reduction steps reducing the new non-zero elements ν_{ij} for $i \in [n+1, n+m]$ and $j \in [p+1, n+1]$.

We start with the extra costs from $\Delta\nu_{ij}$ in the reduction step (p, r). These are

$$\sum_{j=p+1}^{n+1} (2\Delta\nu_{rj} + \Delta\nu_{pj}) = 2\Delta\nu_r + \Delta\nu_p, \tag{4.50}$$

with

$$\Delta\nu_i = \sum_{j=p+1}^{n+1} \Delta\nu_{ij}. \tag{4.51}$$

$\Delta\nu_r$ is multiplied by 2 as it leads to supplementary evaluations in (4.37) and (4.38), while $\Delta\nu_p$ causes additional computations in (4.38) only.

The extra numerical costs from $\Delta\nu_p$ in reduction step (p, i) are

$$\sum_{j=p+1}^{n+1} (\nu_{ip}\Delta\nu_{pj} + \Delta\nu_{ij}) = \nu_{ip}\Delta\nu_p + \Delta\nu_i \qquad i \neq r. \tag{4.52}$$

$\Delta\nu_{pj}$ will result in an additional evaluation of (4.37) in column j of row i if this row is reduced by row p, i.e., if $\nu_{ip} = 1$. The evaluations of equation (4.38) can be judged as extra costs only if they have to be performed for new non-zero elements, i.e., for $\Delta\nu_{ij} = 1$.

Finally, we have to estimate the extra numerical costs from $\Delta\nu_{ij}$ for the rest of the transformation. This is achieved with the following expression:

$$\sum_{k=p+1}^{n+1} \sum_{j=p+1}^{k-1} \nu_{ij}\Delta\nu_{ik} + \sum_{k=p+1}^{n} (5 + \sum_{j=k+1}^{n+1} (\nu_{ij} + \Delta\nu_{ij}))\,\Delta\nu_{ik} \tag{4.53}$$

The first term in (4.53) represents the supplementary numerical costs from Δv_{ik} for the reduction of the non-zero elements v_{ij} from columns $p+1$ to $k-1$, while the second term stands for the costs to reduce element Δv_{ij} itself, i.e., for the evaluation of (4.36) and (4.38) of the reduction step (i,k). The index k in the second term runs from $p+1$ to n only, as a non-zero element in the last column of matrix $\tilde{B}$ does not have to be reduced. The first part of the second term characterizes the costs to evaluate (4.36), which are about five times higher than the costs for one evaluation of (4.37) or (4.38). The second part of the second term gives an estimate of the numerical costs in the evaluation of (4.38). Equation (4.53) can then be simplified to the following expression (cf. Appendix E5):

$$v_i \Delta v_i + (\Delta v_i - \Delta v_{i\,n+1}) \left(5 + \frac{\Delta v_i - 1 + \Delta v_{i\,n+1}}{2}\right). \tag{4.54}$$

Hence, adding up the extra costs defined in (4.50), (4.52), and (4.54) (cf. Appendix E6), the following performance index gives a rough estimate for the total extra costs to be expected from reduction step (p,r) to the end of the transformation:

$$\begin{aligned} J(p,r) = {} & \Delta v_r + \sum_{i=n+1}^{n+m} (v_{ip} \Delta v_p + (1+v_i)\Delta v_i) + \\ & \sum_{i=n+1}^{n+m} (\Delta v_i - \Delta v_{i\,n+1}) \left(5 + \frac{\Delta v_i - 1 + \Delta v_{i\,n+1}}{2}\right) \end{aligned} \tag{4.55}$$

It therefore, should be minimized for every transformation step. Using this performance index the whole transformation can be optimized off-line based on the knowledge about the structure of the B matrix only. Examples for the preservation of numerical costs using this performance index to optimize the step sequence are given in Chapter 6.

Example 4.10

Evaluation of the performance indices for the two possibilities in the first reduction step of Example 4.9. The structure of matrix B is specified by (cf. (4.43) and (4.46))

$$[\nu_{ij}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \quad \text{and} \quad [\nu_i] = \begin{bmatrix} 0 \\ 0 \\ 2 \\ 0 \end{bmatrix}. \tag{4.56}$$

The resulting values for $\Delta\nu_{ij}$, $\Delta\nu_i$, and the corresponding performance indices (cf. (4.49), (4.51), and (4.55)) for both cases are given in the table below:

Reduction Step	$[\Delta\nu_{ij}]$	$[\Delta\nu_i]$	$J(r, p)$
$(1, 3)$	$\begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 2 \\ 0 \\ 0 \\ 2 \end{bmatrix}$	12
$(1, 4)$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	0

Table 4.3 Performance Indices

Clearly, step $(1, 4)$ is by far the better choice.

4.3.4 Sparse Representation

After determining the step sequence, the dyadic reduction is completely fixed, i.e., the structure of the $\tilde{B}$ matrix after every single modification step is known. Hence it is possible to avoid calculations on zero elements in both the dyadic reduction and the backward recursion and to specify the complete transformation with index arrays. These index arrays and the matrix $\tilde{B}$ can be stored in sparse representation to avoid the storing of zero

elements. For the sparse representation we apply a *sparse row-wise format* (cf. [Pissan84]) as the dyadic reduction and backward recursion both operate on rows.

An arbitrary matrix

$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \dots & a_{ij} & \dots \\ a_{m1} & \dots & a_{mn} \end{bmatrix} \tag{4.57}$$

can be given in sparse representation by the vectors $\underline{A}^*$, $\underline{\jmath}_A$ and $\underline{J}_A$. Vector $\underline{A}^*$ contains all non-zero elements of A in row-wise order

$$\underline{A}^* = [\dots a_{1j} \dots a_{2j} \dots], \tag{4.58}$$

while the vectors $\underline{\jmath}_A$ and $\underline{J}_A$ hold the row and column information needed to reconstruct matrix A from $\underline{A}^*$. The row pointer vector $\underline{\jmath}_A = [\imath_{A_i}]$ specifies the last elements of every row in $\underline{A}^*$, i.e., index $\imath_{A_i} + 1$ points to the first element of row i. Thus, the j*th* non-zero element of row i is placed at position $k = \imath_{A_i} + j$ in vector $\underline{A}^*$ and the number of non-zero elements of row i can be given as $\imath_{A_{i+1}} - \imath_{A_i}$. The column indices belonging to the elements of $\underline{A}^*$ are given by the vector $\underline{J}_A = [\jmath_{A_k}]$, where $\jmath_{A_k}$ represents the column index of the k*th* element of vector $\underline{A}^*$.

Example 4.11

Sparse representation of the following matrix:

$$\begin{bmatrix} 0 & a_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ a_{31} & 0 & a_{33} & a_{34} \end{bmatrix}. \tag{4.59}$$

The corresponding vectors $\underline{A}^*$, $\underline{J}_A$ and $\underline{J}_A$ are

$$\begin{aligned} \underline{A}^* &= \left[a_{12}\ a_{31}\ a_{33}\ a_{34}\right]^T, \\ \underline{J}_A &= \left[0\ 1\ 1\ 4\right]^T, \text{ and} \\ \underline{J}_A &= \left[2\ 1\ 3\ 4\right]^T. \end{aligned} \tag{4.60}$$

The main problem in the sparse representation of the matrix $\tilde{B}$ is that the structure of this matrix changes with every reduction step. Therefore, the vector

$$\underline{\tilde{B}}^* = \begin{bmatrix} \underline{\tilde{B}}_0^* \\ \underline{\tilde{B}}_1^* \end{bmatrix} \tag{4.61}$$

must contain all elements being non-zero for at least one transformation step.

Dyadic Reduction

After the optimization of the dyadic reduction, the step sequence and the structure of the matrix $\tilde{B}$ are known for every step. Now the issue is to represent this information in sparse form. Thus, a way has to be found to express both the $\tilde{B}$ matrix and the algorithm in sparse notation.

First, the following index vectors are defined: $\underline{J}_p = [i_{p_i}]$ and $\underline{J}_r = [i_{r_i}]$ are the arrays specifying the p and r rows of the reduction step i, while the elements of vector $\underline{J}_{rp} = [i_{rp_i}]$ point to the elements b_{rp}, i.e., the elements to be reduced in the reduction steps. Thus, instead of the indices p, r, and rp we have i_{p_i}, i_{r_i}, and i_{rp_i}. The first part of the transformation (4.36) is thus determined.

Second, the rest of the dyadic reduction, i.e., (4.37) and (4.38), has to be specified. This is achieved with the supplementary vectors $\underline{j}_D$, $\underline{j}_{Dp}$, $\underline{j}_{Dr}$, and $\underline{j}_{Dn}$. Assume that the non-zero elements in the p and r rows are ordered in groups of consecutive non-zero elements for the evaluation of (4.37) and (4.38). The elements of the index vectors are now defined as follows: Element $i_{D_{2i-1}} + 1$ designates the elements in $\underline{j}_{Dp}$, $\underline{j}_{Dr}$ and $\underline{j}_{Dn}$ characterizing the first group of non-zero elements for transformation (4.37) of step i, while $i_{D_{2i}} + 1$ does the same for transformation (4.38). The number of groups for both transformations is given by $i_{D_{2i}} - i_{D_{2i-1}}$ and $i_{D_{2i+1}} - i_{D_{2i}}$, respectively. For instance the j'*th* group of non-zero elements for transformation (4.37) is defined by the elements j_{Dp_j}, j_{Dr_j} and j_{Dn_j} with $j = i_{D_{2i-1}} + j'$. Finally, the elements $j_{Dp_j} + 1$ and $j_{Dr_j} + 1$ point to the first element of such a group in the p and r row, while j_{Dn_j} specifies the number of non-zero elements of this group.

Thus the dyadic reduction is completely defined by $\tilde{B}^*$ and the index vectors $\underline{j}_p$, $\underline{j}_r$, $\underline{j}_{rp}$, $\underline{j}_D$, $\underline{j}_{Dp}$, $\underline{j}_{Dr}$, and $\underline{j}_{Dn}$. The transformation rules are still given by (4.36), (4.37), and (4.38) except for the indices. For step i the indices r, p and rp are replaced by

$$\begin{aligned} r &\rightarrow i_{r_i}, \\ p &\rightarrow i_{p_i}, \\ rp &\rightarrow i_{rp_i}, \end{aligned} \tag{4.62}$$

in (4.36), (4.37) and (4.38), while rj and pj are substituted by

$$\left.\begin{aligned} pj &\rightarrow j_{Dp_j} + k \\ rj &\rightarrow j_{Dr_j} + k \end{aligned}\right\} \quad \begin{aligned} j &= i_{D_{2i-1}}, \ldots, i_{D_{2i}} - 1 \\ k &= 1, \ldots, j_{Dn_j} \end{aligned} \tag{4.63}$$

in (4.37) and by

$$\left.\begin{array}{l} pj \rightarrow j_{Dp_j} + k \\ rj \rightarrow j_{Dr_j} + k \end{array}\right\} \quad \begin{array}{l} j = i_{D_{2i}}, \ldots, i_{D_{2i+1}} - 1 \\ k = 1, \ldots, j_{Dn_j} \end{array} \tag{4.64}$$

in (4.38).

Example 4.12

Dyadic reduction step in sparse notation. Assume that we have the following p and r rows in matrix $\mathbf{B}$ for step 3 (i.e. i = 3), with p = 2 and r = 20.

$$\begin{array}{ll} \text{row 2} & [0\ 1\ 0\ x\ x\ x\ 0\ x\ 0\ x] \\ \text{row 20} & [0\ x\ 0\ x\ 0\ 0\ x\ x\ 0\ x] \end{array} \xrightarrow{(2,20)} \begin{array}{l} [0\ 1\ 0\ x\ x\ x\ x\ x\ 0\ x] \\ [0\ 0\ 0\ x\ x\ x\ x\ x\ 0\ x] \end{array} \tag{4.65}$$

The sparse representation of matrix $\tilde{\mathbf{B}}$ after the second transformation step could look as follows:

$$\begin{aligned} \tilde{\mathbf{B}}^* = [& \ldots, 1, 0, x, x, x, 0, x, x, \ldots \\ & \ldots, x, 0, x, 0, 0, x, x, x, \ldots]^T, \end{aligned} \tag{4.66}$$

with

$$\begin{aligned} \mathbf{J}_p &= [x\ x\ 2\ \ldots], \\ \mathbf{J}_r &= [x\ x\ 20\ \ldots], \text{ and} \\ \mathbf{J}_{rp} &= [x\ x\ 30\ \ldots]. \end{aligned} \tag{4.67}$$

The arrows symbolize pointers, i.e., element 1 of the p row (p = 20) is located in position 5 of vector $\tilde{\mathbf{B}}^*$. Notice that $\tilde{\mathbf{B}}^*$ may still contain zero elements if they are becoming non-zero in a later step (e.g., the elements of the third column in rows 2 and 20). If not they are no longer included (e.g., the elements of the 9*th* column in rows 2 and 20).

For the evaluation of the new elements in the r row (cf. (4.37)), the non-zero elements can be ordered into two groups of three and two elements, respectively. The appertaining index vectors could look like this

$$
\begin{aligned}
\underline{J}_D &= [x\ x\ x\ 9\ 11\ \ldots]^T, \\
\underline{J}_{Dp} &= [\ldots\ \overset{9}{\downarrow}\ 6\ 10\ \ldots]^T, \\
\underline{J}_{Dr} &= [\ldots\ \overset{9}{\downarrow}\ 31\ 35\ \ldots]^T, \quad \text{and} \\
\underline{J}_{Dn} &= [\ldots\ \overset{9}{\downarrow}\ 3\ 2\ \ldots]^T.
\end{aligned}
\tag{4.68}
$$

The new $\tilde{\underline{B}}^*$ vector has the following structure after the transformation (4.37):

$$
\begin{aligned}
\tilde{\underline{B}}^* = [\ldots, 1, 0, x, x, x, 0, x, x, \ldots \\
\overset{30}{\downarrow} \\
\ldots, x, 0, x, x, x, x, x, x, \ldots]^T\ ,
\end{aligned}
\tag{4.69}
$$

For the calculation of the p row elements (cf. (4.38)) all non-zero elements of row r can be combined into one group. Thus, the index vectors are extended by one element only.

$$
\begin{aligned}
\underline{J}_D &= [x\ x\ x\ 9\ 11\ 12\ \ldots]^T, \\
\underline{J}_{Dp} &= [\ldots\ \overset{9}{\downarrow}\ 6\ 10\ 6\ \ldots]^T, \\
\underline{J}_{Dr} &= [\ldots\ \overset{9}{\downarrow}\ 31\ 35\ 31\ \ldots]^T, \quad \text{and} \\
\underline{J}_{Dn} &= [\ldots\ \overset{9}{\downarrow}\ 3\ 2\ 6\ \ldots]^T.
\end{aligned}
\tag{4.70}
$$

Backward Substitution

To following linear equations (cf. (4.30)) can be solved for $\underline{z}$ quite easily by backward substitution:

$$\mathsf{U}\underline{z} - \tilde{\underline{z}}_d = \begin{bmatrix} \mathsf{U} & -\tilde{\underline{z}}_d \end{bmatrix} \begin{bmatrix} \underline{z} \\ 1 \end{bmatrix} = \tilde{\mathsf{B}}_0 \begin{bmatrix} \underline{z} \\ 1 \end{bmatrix} = 0. \tag{4.71}$$

Matrix $\tilde{\mathsf{B}}_0$ stands for the upper part of the matrix B after completion of the dyadic reduction (cf. (4.27)). In sparse notation $\tilde{\mathsf{B}}_0$ is represented by $\underline{\tilde{B}}_0^*$ and the pointer vectors $\underline{J}_{B0}$ and $\underline{J}_{B0}$.

$$\underline{\tilde{B}}_0^* = \begin{bmatrix} \dots \tilde{b}_{ij} \dots & -z_{d_1} \dots \tilde{b}_{ij} \dots & -z_{d_i} \dots & -z_{d_n} \end{bmatrix} \qquad i < j. \tag{4.72}$$

Remark 4.6:

The diagonal elements of U need not be stored as their value always equals one. Additionally we can see from (4.71) that $\tilde{\mathsf{B}}_0$ always possesses non-zero elements in the last column. Therefore, these elements do not have to be referenced in the vector $\underline{J}_{B0}$ either.

Example 4.13

Sparse representation of the following problem for backward substitution:

$$\tilde{\mathsf{B}}_0 \begin{bmatrix} \underline{z} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & a_{12} & 0 & a_{14} & -z_{d_1} \\ 0 & 1 & a_{23} & a_{24} & -z_{d_2} \\ 0 & 0 & 1 & 0 & -z_{d_3} \\ 0 & 0 & 0 & 1 & -z_{d_4} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ 1 \end{bmatrix} = 0. \tag{4.73}$$

In sparse representation (4.73) is defined as

$$\begin{aligned}
\underline{\tilde{B}}_0^* &= \left[a_{12}\ a_{14}\ -z_{d_1}\ a_{23}\ a_{24}\ -z_{d_2}\ -z_{d_3}\ -z_{d_4}\right]^T, \\
\underline{J}_{B0} &= \left[0\,2\,4\,4\,4\right]^T, \text{ and} \\
\underline{J}_{B0} &= \left[2\,4\,3\,4\right]^T.
\end{aligned} \tag{4.74}$$

With this information, the backward recursion is completely specified. If n_B is the number of non-zero elements of $\underline{\tilde{B}}_0^*$ the solution of (4.71) can be computed with the following simple Fortran program:

```
      k = n_B
      DO 10, i = n,1,-1
         dummy = - B_sparse(k)
         k = k-1
         DO 20, j = I_B(i+1), I_B(i)+1,-1
            dummy = dummy - B_sparse(k)*z(J_B(j))
            k =k - 1
20       CONTINUE
         z(i) = dummy
10    CONTINUE
```

Chapter 5

The Controller

This chapter elaborates on a controller which is specialized in the control of robotic systems. The control concepts of Chapter 4 are applied to the arbitrary, rigid multi-body systems treated in Chapter 3. The constraint equations from Chapter 3 are valid for any type of robot systems, i.e., they are applicable to rigid multi-body systems with open and closed kinematic chains, redundancies, joint friction, boundary constraints, and changing structures. For computation they only have to be rewritten in matrix notation. Yet, the controller additionally depends on system properties, such as available sensor equipment, control objectives, etc. Thus, to be more specific, the controller presented in this chapter is developed for a rather simple mechanical system. However, as Chapter 6 describes, with only minimal changes this type of controller is suitable for much more complex systems as well.

The chapter is organized as follows: Section 5.1 briefly describes the cart-pole system to be controlled, its sensor equipment, and the goals to be reached. The information processing level is characterized in Section 5.2, whereas the levels coordination and execution are explained in Sections 5.3 and 5.4, respectively. Finally, Section 5.5 outlines the development of the controller through use of the symbol manipulation language Maple (cf. [ChGeGo88]). Some simulations of the control of this system are presented in Chapter 6.

5.1 Plant

The plant is composed of a pole (body 2) hinged to a cart (body 1) which is driven by a horizontal force **u** (cf. Figure 5.1). The task is to balance the pole moving its tip into the demanded position for an arbitrary initial position and velocity of both bodies. The sensor information consists of the horizontal position of the cart, s'_{11_1}, the orientation of the pole, φ'_{22_3} (cf. Figure 5.2), and the corresponding first time derivatives $\dot{s}'_{11_1}$ and θ'_{22_3}. In addition, there is a boundary constraint for the horizontal movement of the cart (cf. Figure 5.2).

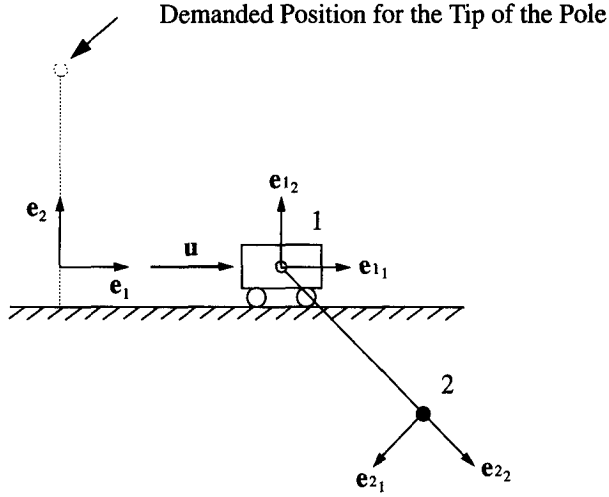


Figure 5.1 Plant

This is a problem frequently posed to students of control engineering as the nonlinear dynamics of the plant are rather easy to model. The typical approach to solve this problem is to linearize the model for the position and velocity demanded and to design a linear controller. The disadvantage of such a controller is that its performance will dramatically decrease if the

system deviates too far from the demanded values. It can be shown that it is not possible to stabilize this system with one linear, time-invariant controller. This can best be seen from Figure 5.2. Consider the pole in the two positions depicted, one in the upper and one in the lower half plane, with zero angular velocity. For both cases we have the same sign in the angular position error φ'_{22_3}. However, to decrease the positional error we have to push in the first case and to pull in the second case. This is not possible for a linear time-invariant controller.

The controller developed below, however, is capable of stabilizing this system in the complete state space.

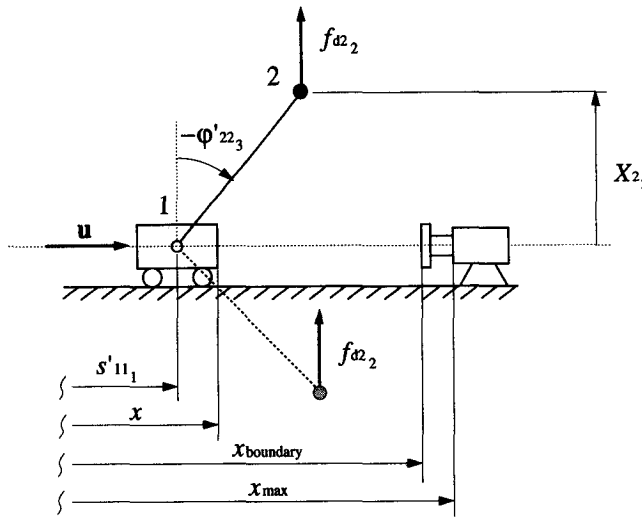


Figure 5.2 Boundary Constraint

5.2 Information Processing

The information processing level (cf. Figure 4.1) is responsible for the best possible update of the internal data to be used by the coordination and execution levels. These data consist of all parameters, state variables, interaction variables, and other intermediate variables. They are related to the sensor data, on the one hand, and depend on each other due to constraint equations, on the other hand. Therefore, the internal data first have to be corrected to improve the consistency of the complete data set (internal and sensor data). This update is subdivided into three steps:

- Improvement of the consistency of the internal data and the latest sensor information
- Extrapolation of the system state at the next sampling time
- Improvement of this estimated system state using the kinematic constraints

The first step, the improvement of the internal data based on the sensor information, strongly depends on the kind of sensor data. Therefore, it cannot be described too specifically for general systems. The mechanism is the following: The point of departure consists of the sensor data and the estimates for these sensor data, or variables related to these data, from the internal data set. This sensor information and these estimates result in a set of linear, explicit equations for the corrected values of the internal data. The interrelations can be defined by an additional set of linear, implicit equations. Finally, the precision or reliability of the different equations is characterized by the corresponding weighting factors. The set of all equations and weighting factors form a performance index, which is minimized numerically.

Example 5.1

Improvement of the internal variable φ of the system depicted in Figure 5.3: $\hat{\varphi}$ stands for an estimate of the angle φ. Assume that we

only have a sensor measuring the torque τ_s caused by a rotational spring between the rod and the horizontal. The characteristics of this spring are modeled by $\tau_s = c\varphi$. The improvement step is now specified by the following B matrix:

$$\mathsf{B} = \begin{bmatrix} 1 & -\hat{\varphi} \\ c & -\tau_s \end{bmatrix} \tag{5.1}$$

for the unknown $\underline{z} = [\varphi]$. The weighting factors determine the reliability of the variables $\hat{\varphi}$ and τ_s, respectively. Thus, if the torque measurement is much more precise than the estimation of the angle, the weight has to be chosen larger for the second row of B than for the first. In case of a second sensor measuring the angle directly as φ_s, the B matrix is simply extended, yielding

$$\mathsf{B} = \begin{bmatrix} 1 & -\hat{\varphi} \\ c & -\tau_s \\ 1 & -\varphi_s \end{bmatrix}, \tag{5.2}$$

and the dimension of the weighting matrix D increases by one.

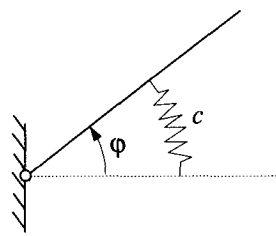


Figure 5.3

The second step, i.e., the extrapolation of the system state at time $t + \Delta t$, is rather simple, if we assume that the first time derivative of the system state is constant for the next time interval Δt. Of course, this assumption is

only valid if Δt is chosen small enough. Additionally, Remark 3.5 concerning the update of unitary spinors has to be taken into account.

Example 5.2

Extrapolation of the hinge position s'_{aj_k}:

$$s'_{aj_k}(t+\Delta t) = s'_{aj_k}(t) + \Delta t\, \dot{s}'_{aj_k}(t)\,. \tag{5.3}$$

If we do not have an estimate for $\dot{s}'_{aj_k}(t)$, it will typically be chosen to be $\dot{s}'_{aj_k}(t) = 0$.

In the third step, the estimates of the system state at time $t+\Delta t$ are matched with the kinematical constraints. Therefore, the intermediate variables ${}^{j}\underline{S}_a$, ${}^{j}\underline{s}_a$, ${}^{j}\underline{\theta}_a$, ${}^{j}\underline{g}_a$, ${}^{j}\underline{h}_a$, and the directions of the rotational and translational hinge coordinates $\hat{\underline{b}}_{aj_k}$, $\hat{\underline{s}}_{aj_k}$ at time $t+\Delta t$ need to be evaluated first. This can be done recursively, to be numerically more efficient. In matrix notation we write for these quantities (cf. (3.31), (3.32), (3.33), (3.34), (3.51), (3.34), (2.90), (2.92), (2.64)):

$${}^{j}\underline{S}_a = \begin{cases} \underline{R}_\alpha & j = n_a + 1, \\ \left[\underline{S}'_{aj}\ \overline{S}'_{aj}\right]\, {}^{j+1}\underline{S}_a & j \in [1, n_a], \end{cases} \tag{5.4}$$

$${}^{j}\underline{s}_a = \begin{cases} \underline{X}_\alpha & j = n_a + 1, \\ {}^{j+1}\underline{s}_a + {}^{j+1}S_a \underline{s}'_{aj} & j \in [1, n_a], \end{cases} \tag{5.5}$$

$${}^{j}\underline{\theta}_a = \begin{cases} \underline{\omega}_\alpha & j = n_a + 1, \\ {}^{j+1}\underline{\theta}_a + {}^{j+1}S_a \underline{\theta}'_{aj} & j \in [1, n_a], \end{cases} \tag{5.6}$$

$$ {}^{j}\underline{g}_a = \begin{cases} 0 & j = n_a + 1, \\ {}^{j+1}\underline{g}_a + {}^{j+1}\underline{\theta}_a \times {}^{j+1}S_a\underline{\theta}'_{aj} & j \in [1, n_a], \end{cases} \tag{5.7} $$

$$ {}^{j}\underline{h}_a = \begin{cases} 0 & j = n_a + 1, \\ {}^{j+1}\underline{h}_a + {}^{j+1}\underline{g}_a \times {}^{j+1}S_a\underline{s}'_{aj} + & \\ {}^{j+1}\underline{\theta}_a \times ({}^{j+1}\underline{\theta}_a \times {}^{j+1}S_a\underline{s}'_{aj} + 2\,{}^{j+1}S_a\dot{\underline{s}}'_{aj}) & j \in [1, n_a], \end{cases} \tag{5.8} $$

$$ \hat{\underline{b}}_{aj_k} = {}^{j}S_a\hat{\underline{b}}'_{aj_k}, \tag{5.9} $$

and

$$ \hat{\underline{s}}_{aj_k} = {}^{j}S_a\hat{\underline{s}}'_{aj_k}. \tag{5.10} $$

Remark 5.1:

Although the term ${}^{i+1}S_a\underline{s}'_{ai}$ appears in more than one equation, it is computed only once.

The conversion of the constraint equations from Chapter 3 into matrix notation is somewhat more complex. It is described fully in Appendix F. For the positional constraints (3.43) and (3.44) we obtain (cf. Appendices F1, F2, and F3)

$$ J_{1S}\Delta\underline{\varphi}' + C_{1S}{}^{T}\Delta\underline{\vartheta} + \underline{b}_{1pos} = 0, \tag{5.11} $$

$$ J_2\Delta\underline{\varphi}' + J_3\Delta\underline{s}' + C_2{}^{T}\Delta\underline{\vartheta} + C_3{}^{T}\Delta\underline{X} + \underline{b}_{2pos} = 0. \tag{5.12} $$

The elements of the matrices C_i, J_i, and the vectors $\underline{b}_{ipos}$ can be expressed in terms of ${}^{j}\underline{S}_a$, ${}^{j}\underline{s}_a$, $\hat{\underline{b}}_{aj_k}$, $\hat{\underline{s}}_{aj_k}$, and the connectivity coefficients. The corresponding performance index is defined by the following matrices B, and D, and the vector $\underline{z}$,

$$\mathbf{B}_{pos} = \begin{bmatrix} \mathbf{E} & 0 \\ \mathbf{A}_{pos} & \underline{\mathbf{b}}_{pos} \end{bmatrix},$$

$$\mathbf{D}_{pos} = \begin{bmatrix} \mathbf{D}_{0pos} & 0 \\ 0 & \mathbf{D}_{1pos} \end{bmatrix}, \tag{5.13}$$

$$\underline{z}_{pos}^{T} = \left[\Delta\underline{\vartheta}^{T}\ \Delta\underline{X}^{T}\ \Delta\underline{\varphi}'^{T}\ \Delta\underline{s}'^{T}\right],$$

with

$$\mathbf{A}_{pos} = \begin{bmatrix} \mathbf{C}_{1S}^{T} & 0 & \mathbf{J}_{1S} & 0 \\ \mathbf{C}_{2}^{T} & \mathbf{C}_{3}^{T} & \mathbf{J}_{2} & \mathbf{J}_{3} \end{bmatrix} \tag{5.14}$$

and

$$\underline{\mathbf{b}}_{pos}^{T} = \left[\underline{\mathbf{b}}_{1pos}^{T}\ \ \underline{\mathbf{b}}_{2pos}^{T}\right]. \tag{5.15}$$

$\mathbf{D}_{0pos}$ is the weight matrix for the explicit equations, while matrix $\mathbf{D}_{1pos}$ belongs to the implicit constraints. The elements of $\mathbf{D}_{0pos}$ specify the precision of the positional estimates, while $\mathbf{D}_{1pos}$ characterizes the "elasticity" of the constraints. Thus, the corresponding weight in $\mathbf{D}_{pos}$ must be chosen larger if a position estimate is more accurate or if a constraint is more rigid.

Example 5.3

Consider the boundary constraint for the cart in the horizontal direction (cf. Figure 5.2). The constraint equation is simply $\Delta x = 0$, if $x \geq x_{max}$. The characteristics of a constraint are specified by the weighting factor. Figure 5.4 shows some examples for the modeling of boundary constraints. The weighting factors are depicted for an absolutely rigid or "ideal" behavior and for linear and nonlinear elastic characteristics of the constraint in the interval $[x_{boundary}, x_{max}]$. Evidently, the weighting factors allow to specify non-ideal constraints in a rather simple manner. This improves the model quality, on the one hand, and

increases its numerical robustness, on the other hand, since the changes in the system structure do not have to be defined so sharply.

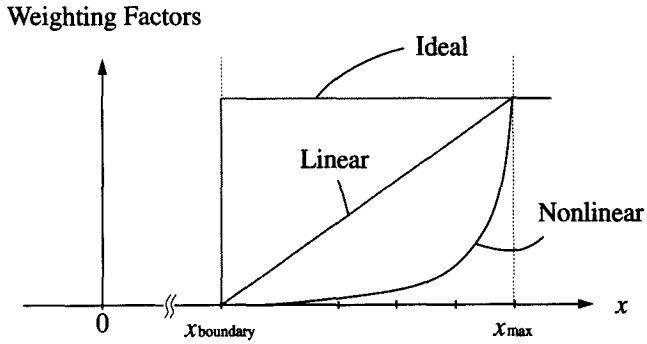


Figure 5.4 Weighting Factors

Remark 5.2:

Because unilateral constraints can produce forces or torques in one direction only, the corresponding weighting factors typically depend on the constraint forces and torques as well.

For the update of the velocity variables we can proceed in the same manner. The velocity constraints (3.55) and (3.56) are written as

$$J_1 \underline{\theta}' + C_1^T \underline{\omega} = 0 \tag{5.16}$$

and

$$J_2 \underline{\theta}' + J_3 \dot{\underline{s}}' + C_2^T \underline{\omega} + C_3^T \dot{\underline{X}} = 0 \tag{5.17}$$

in matrix notation (cf. Appendices F1, F2, and F4). The corresponding performance index is given by the following matrices and vectors:

$$
\begin{aligned}
B_{vel} &= \begin{bmatrix} E & -\underline{z}_{d\,vel} \\ A_{vel} & 0 \end{bmatrix}, \\
D_{vel} &= \begin{bmatrix} D_{0\,vel} & 0 \\ 0 & D_{1\,vel} \end{bmatrix}, \\
\underline{z}_{vel}^{T} &= \begin{bmatrix} \underline{\omega}^{T} & \dot{\underline{X}}^{T} & \underline{\theta}'^{T} & \dot{\underline{s}}'^{T} \end{bmatrix},
\end{aligned}
\tag{5.18}
$$

with

$$
A_{vel} = \begin{bmatrix} C^{T} & J \end{bmatrix} = \begin{bmatrix} C_1^{T} & 0 & J_1 & 0 \\ C_2^{T} & C_3^{T} & J_2 & J_3 \end{bmatrix}, \tag{5.19}
$$

where $\underline{z}_{d\,vel}$ is the vector containing the estimates for $\underline{z}_{vel}$.

5.3 Coordination

The coordination level is responsible for the decision-making (cf. Figure 4.1). If the control instructions are precise enough, there is, of course, no need for additional decision-making. However, such a complete specification of a task comprises the following disadvantages. First, it is rather costly, in many cases even impossible, to determine these control instructions. (Consider, for example, the pole balancing task described in Section 5.1.) Therefore, in most cases, it is only feasible for repetitive tasks. Second, the controller is deprived of the ability to react to unforeseen events if the demanded trajectory is fully specified. Thus, it is best to specify only as much as needed and leave the rest to the controller.

The coordination level evaluates the unknown "demanded values" required to solve the given task and the corresponding weights. For robotic systems the demanded values are either forces, torques, or accelerations.

They represent the goals or the desired actions to be taken and may be contradictory. The fine tuning is achieved with the weighting factors. They are used to specify the relevance or importance of the different demands, on the one hand, and they define the dependency of these demands on the system state or upon time, on the other hand.

Remark 5.3:

Usually the goals and their corresponding weights are defined explicitly. Thus, they can be evaluated directly without any necessity to minimize a performance index.

Example 5.4

To move the tip of the pole (cf. Figure 5.1) into the desired position, we can demand the controller to act on the system in the same manner as the force f_{d2_2} (cf. Figure 5.2), with

$$f_{d2_2} = p_{11}\,(l - X_{2_2}) + p_{12}\,(0 - \dot{X}_{2_2}) \tag{5.20}$$

(l being the length of the pole). Of course, we can also demand a vertical acceleration if the execution level possesses a dynamical model of the plant, i.e., the request could then be

$$\ddot{X}_{d2_2} = p_{21}\,(l - X_{2_2}) + p_{22}\,(0 - \dot{X}_{2_2})\,. \tag{5.21}$$

The second request prescribes the desired motion more precisely. However, a controller with (5.20) or (5.21) as sole demands is not able to guide the system out of the state $\varphi_{2_3} = \pm\pi$ and $\omega_{2_3} = 0$. It is impossible to generate a vertical acceleration or force with the horizontal input force $\mathbf{u}$ under these conditions. The way out is to define another request such as

$$\dot{\omega}_{d2_3} = p_{31}\varphi_{2_3} + p_{32}\,(0 - \omega_{2_3})\,. \tag{5.22}$$

In this state the controller will then additionally try to produce a rotational acceleration which is in fact attainable.

Example 5.5

The weighting factors can be utilized, for example, to avoid unnecessarily high control forces and torques close to or in singular positions. Assume that the pole (cf. Figures 5.1 and 5.2) is in the horizontal position. In this state it is impossible to generate either a vertical or a rotational acceleration with the horizontal force $\mathbf{u}$. Close to this position such an acceleration can theoretically be achieved, but only with very large forces. This is avoided for $\dot{\omega}_{d2_3}$ with the weighting factor depicted in Figure 5.5.

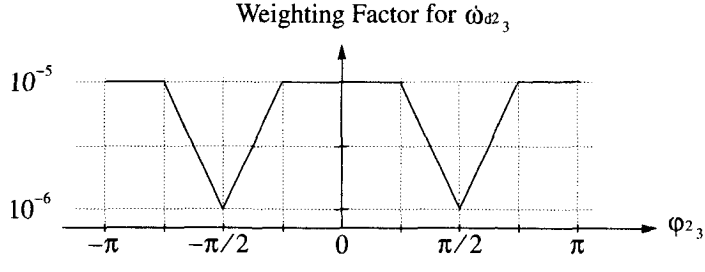


Figure 5.5 Weighting Factor

5.4 Execution

The main task of this level consists in determining a control vector $\mathbf{u}$ such that the desired goals, determined by the coordination level, are attained (cf. Figure 4.1). This is achieved by using the estimates of the actual state of the system from the information processing level. If the dynamics are modeled, the accelerations and the constraint forces are computed automatically. Additionally, the state of the unilateral constraints may be determined based on the latest information of the hinge states and the hinge forces and torques. The relationship between all these quantities

and the control vector **u** is given by the dynamics, the acceleration constraints, and the force and torque constraints.

From the dynamics (cf. (3.92) and (3.93)) the following constraint equations result (cf. Appendices F1, F2, and F7):

$$M_1\dot{\underline{\omega}} + C_1\underline{\tau} + C_2\underline{f} + \underline{b}_{1dyn} = 0, \tag{5.23}$$

$$M_m\ddot{\underline{X}} + C_3\underline{f} + \underline{b}_{2dyn} = 0. \tag{5.24}$$

In matrix notation, the acceleration constraints (3.55) and (3.56) are written as (cf. Appendices F1, F2, and F5)

$$J_1\dot{\underline{\theta}}' + C_1^T\underline{\omega} + \underline{g} = 0 \tag{5.25}$$

and

$$J_2\dot{\underline{\theta}}' + J_3\ddot{\underline{s}}' + C_2^T\underline{\omega} + C_3^T\ddot{\underline{X}} + \underline{h} = 0. \tag{5.26}$$

Finally, the force and torque constraints (3.71) and (3.74) are given as (cf. Appendices F1, F2, and F6)

$$J_1^T\underline{\tau} + J_2^T\underline{f} - B_{1u}\underline{u} - \underline{\tau}'_P = 0, \tag{5.27}$$

$$J_3^T\underline{f} - B_{2u}\underline{u} - \underline{f}'_P = 0. \tag{5.28}$$

These constraints are all united in the matrix A_{dyn} and the vector $\underline{b}_{ddyn}$. The performance index of this subtask is defined by the following matrices and vectors:

$$
\begin{aligned}
\mathbf{B}_{dyn} &= \begin{bmatrix} \mathbf{E} & -\underline{\mathbf{z}}_{d\,dyn} \\ \mathbf{A}_{dyn} & \underline{\mathbf{b}}_{d\,dyn} \end{bmatrix}, \\
\mathbf{D}_{dyn} &= \begin{bmatrix} \mathbf{D}_{0dyn} & 0 \\ 0 & \mathbf{D}_{1dyn} \end{bmatrix}, \\
\underline{\mathbf{z}}_{dyn}^{T} &= \begin{bmatrix} \underline{\mathbf{u}}^{T} & \underline{\dot{\omega}}^{T} & \ddot{\underline{\mathbf{X}}}^{T} & \underline{\tau}^{T} & \underline{\mathbf{f}}^{T} & \dot{\underline{\theta}}'^{T} & \ddot{\underline{\mathbf{s}}}'^{T} \end{bmatrix},
\end{aligned}
\tag{5.29}
$$

$$
\mathbf{A}_{dyn} = \begin{bmatrix} 0 & \mathbf{M} & \mathbf{C} & 0 \\ 0 & \mathbf{C}^{T} & 0 & \mathbf{J} \\ -\mathbf{B}_{u} & 0 & \mathbf{J}^{T} & 0 \end{bmatrix} = \begin{bmatrix} 0 & \mathbf{M}_{I} & 0 & \mathbf{C}_{1} & \mathbf{C}_{2} & 0 & 0 \\ 0 & 0 & \mathbf{M}_{m} & 0 & \mathbf{C}_{3} & 0 & 0 \\ 0 & \mathbf{C}_{1}^{T} & 0 & 0 & 0 & \mathbf{J}_{1} & 0 \\ 0 & \mathbf{C}_{2}^{T} & \mathbf{C}_{3}^{T} & 0 & 0 & \mathbf{J}_{2} & \mathbf{J}_{3} \\ -\mathbf{B}_{1u} & 0 & 0 & \mathbf{J}_{1}^{T} & \mathbf{J}_{2}^{T} & 0 & 0 \\ -\mathbf{B}_{2u} & 0 & 0 & 0 & \mathbf{J}_{3}^{T} & 0 & 0 \end{bmatrix}, \tag{5.30}
$$

$$
\underline{\mathbf{b}}_{dyn}^{T} = \begin{bmatrix} \underline{\mathbf{b}}_{1dyn}^{T} & \underline{\mathbf{b}}_{2dyn}^{T} & \underline{\mathbf{g}}^{T} & \underline{\mathbf{h}}^{T} & -\underline{\tau}'_{P}{}^{T} & -\underline{\mathbf{f}}'_{P}{}^{T} \end{bmatrix}. \tag{5.31}
$$

Remark 5.4:

This performance index specifies the dynamics of arbitrary robotic systems. It can either be used to evaluate the *inverse dynamics*, i.e., the control forces in relation to the actual state and the desired accelerations, or, for the *forward dynamics*, where the momentary accelerations are computed from the control forces and torques [BalPat91]. The only difference is that for the inverse dynamics the weighting factors in $\mathbf{D}_{0dyn}$ belonging to the control forces are chosen much smaller than those belonging to the accelerations and vice-versa for the forward dynamics. Thus, the accelerations are bound to their demanded values in case of the inverse dynamical problem, whereas the values of the control forces are prescribed in case of the forward dynamical problem.

A change in the system structure (i.e., a change in the number of degrees of freedom) due to a singularity does not have to be treated specifically since the dyadic reduction algorithm solves the minimization problem for all cases (i.e., determined, overdetermined, or underdetermined). Thus, it is not necessary to check for such singularities. Changes in the system structure caused by unilateral constraints are simply defined by the weighting factors in D_{dyn}.

Example 5.6

For the cart-pole system with a horizontal boundary constraint (cf. Figures 5.1 and 5.2) there are the explicit boundary constraints $\dot{x} = 0$ and the force constraint $f_c = 0$. If the weighting factor corresponding to the first equation is much larger than the one belonging to $f_c = 0$ we model the "active" constraint. On the other hand, the "free" system is characterized by the first weighting factor being much smaller than the second. This change from the free system to the constrained system and vice-versa can be modeled smoothly. This increases the numerical robustness of the model.

Another advantage of this kind of modeling a robotic system is that the model can be enlarged in small steps, allowing to model the relevant effects only, thereby, reducing the modeling effort to a minimum.

Example 5.7

Assume that we can enforce a certain movement on the cart in Figure 5.1 or maybe have a good guess about this movement, but we do not know its consequences on the motion of the pole. In this case, we simply specify the acceleration of the cart in the explicit part of the performance index and neglect the dynamic equations of the cart. Moreover, we can ignore the acceleration, the force and the torque

constraints if we are not interested in the accelerations and the forces and torques in the hinge between the inertial frame and the cart.

Example 5.8

If we consider the dynamics to be irrelevant, maybe due to small accelerations or small masses, we can simply set the mass matrix to zero and neglect the acceleration constraints. Of course, this can also be done for individual parts of the system.

5.5 Computerized Controller Development

The methodology presented in Chapters 2, 3, 4, and 5 is very suitable for being automated. The procedure of the controller development is briefly outlined in this section. The definition of such a controller takes place in four steps. The first three steps are performed in Maple, an interpreter for symbolic manipulations, while the last step consists in fine-tuning the controller with simulations (cf. Figure 5.6).

In a first step, the mechanical system to be controlled is defined. Thus, we have to specify:

- the object variables m_α, and I_{α_k},
- the interconnection structure $c_{\alpha a}$, and
- the hinge specification: $\hat{\underline{b}}'_{aj_k}$, $\hat{\underline{s}}'_{aj_k}$, $b_{\max aj_k}$, $b_{\min aj_k}$, $s_{\max aj_k}$, $s_{\min aj_k}$, τ'_{aj_k}, and f'_{aj_k}

This information is stored in a text file which, processed by a Maple program, results in the matrices B_{pos}, B_{vel}, and B_{dyn} (cf. (5.13), (5.18), and (5.29)) and the boundary constraints. These matrices fully characterize the kinematical and dynamical properties of the system. In the second step, they are modified and additional B matrices, specifying other tasks of the

controller, are generated manually if necessary. Third, the minimization (i.e., the dyadic reduction) is optimized and represented in sparse form (cf. Section 4.3.3 and 4.3.4) for all performance criteria, characterized by these B matrices. This is done by a second Maple program which additionally produces the corresponding Fortran code, including the code for the boundary constraints. The controller is, thus, fully determined except for the weights of the diagonal matrices and other parameters introduced to tune the controller. The determination and improvement of these parameters is then achieved with simulations in the fourth and last step. Chapter 6 will explain this in more detail.

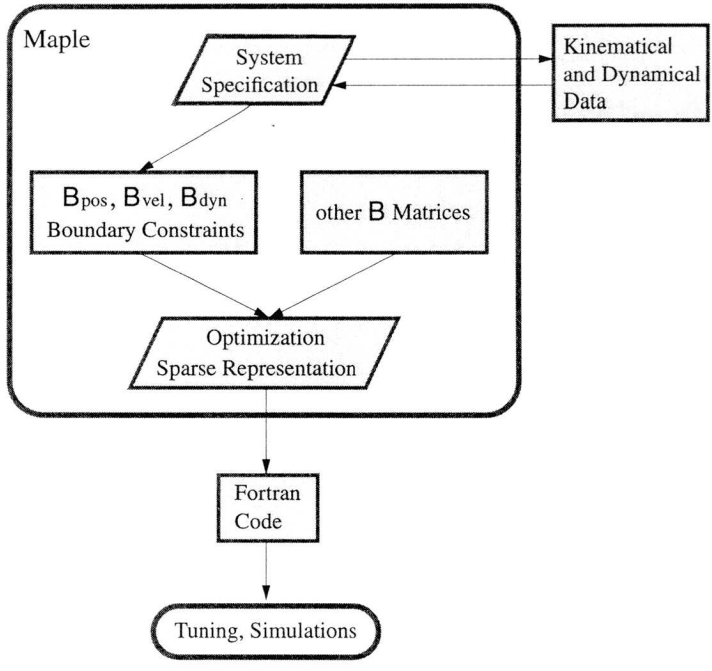


Figure 5.6 Controller Development

Chapter 6

Simulation Results

This chapter describes the application of the proposed methodology on two mechanical systems. The behavior of the controlled systems is illustrated with simulations. The first benchmark treats the control of the cart-pole system introduced in Chapter 5. This is explained in Section 6.1. A much more difficult control problem is outlined in Section 6.2. The matter at issue is the control of a human-like robot walking in the sagittal plane. Finally, Section 6.3 discusses the numerical costs of the most expensive subtasks in the computation of the control vectors for both benchmarks.

6.1 Cart-Pole System

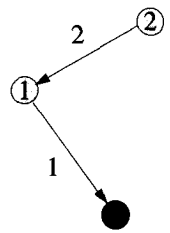
6.1.1 Problem Statement

As outlined in Section 5.1, the plant is a cart-pole system (cf. Figure 5.1), and the control task consists of balancing the pole moving its tip into the demanded position for an arbitrary initial motion. As sensor information we have the hinge coordinates $\varphi'_{22_3}, s'_{11_1}$ (cf. Figure 5.2) and their first time derivatives. The cart is driven by a horizontal force $\mathbf{u}$ acting on the cart, with the sampling rate of the controller fixed at 100 Hz.

From the mechanical point of view, this is a rather simple system both to understand and to model. In contrast, it is a rather difficult system to cope with from the control point of view. This is due to the nonlinearities of the plant and the boundary constraints. Another difficulty arises from having one actuator for two degrees of freedom. (Robot manipulators usually have a separate drive in every joint.) The two degrees of freedom therefore cannot be controlled independently, i.e., a complete compensation of the nonlinear terms, as described in Section 1.1, is not possible.

We use the procedure proposed in Chapter 3 to describe the kinematics and dynamics of the plant.

The first step is to identify the interconnection structure or the corresponding connectivity matrix $[c_{\alpha a}]$ (cf. Figure 6.1). There are two rigid bodies (body 1: cart, body 2: pole) and two hinges.



$$[c_{\alpha a}] = \begin{bmatrix} -1 & 1 \\ 0 & -1 \end{bmatrix}$$

Figure 6.1 Interconnection Structure of Cart-Pole System

Next, we need to define the object variables of the two bodies. They are given in Table 6.1. The principal moment of inertia of the cart does not have to be taken into account as body 1 does not perform a rotation and the pole is modeled as a point mass with the entire mass in its tip.

α	m_α [kg]	I_{α_3} [kg m^2]
1	1	—
2	0.1	0

Table 6.1 Object Variables

Finally, the two hinges need to be characterized. They are described by the following two rigid displacements

$$\{S'_1 \mid \mathbf{s}'_1\} = \{1 \mid s'_{11_1}\mathbf{e}_1\}, \tag{6.1}$$

$$\{S'_2 \mid \mathbf{s}'_2\} = \{a'_{22} + ib'_{22_3}\mathbf{e}_3 \mid 0\}\,\{1 \mid 1\mathbf{e}_1\}, \tag{6.2}$$

the positional boundary constraint

$$s'_{11_1} \in [-5, 5], \tag{6.3}$$

and the physical properties of the two joints

$$f'_{11_1} = u_1, \tag{6.4}$$

$$\tau'_{22_3} = 0. \tag{6.5}$$

From (6.4) and (6.5) we see that the translational joint of the first hinge is actuated and that the rotational joint of the second hinge is assumed to be ideal.

With the connectivity matrix $[c_{\alpha a}]$, the object variables of Table 6.1, and equations (6.1) to (6.5), the kinematics and dynamics of the system are fully determined (cf. Section 5.5).

6.1.2 Solution of the Control Task

The control task is decomposed into six subtasks. Three of these subtasks are defined with a performance index whereas the other three are formulated explicitly. The overall structure of the controller is depicted in Figure 6.2.

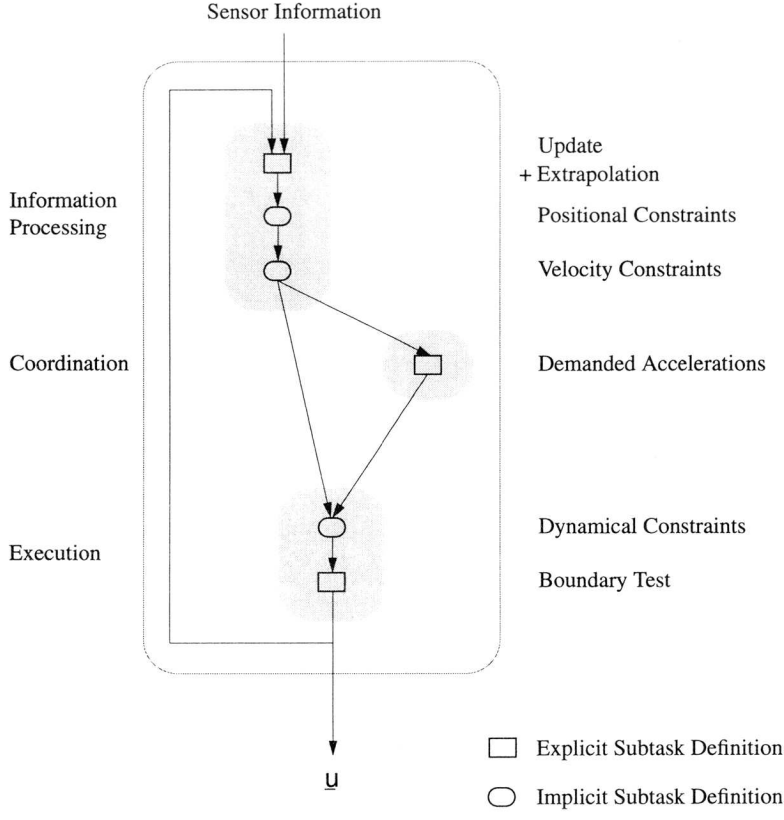


Figure 6.2 Controller Structure

Information Processing Level

Starting with the information processing level, we have three subtasks. The first subtask, *Update and Extrapolation,* calculates a first estimate for the motion of the system at the next sampling time $t + \Delta t$ utilizing the given sensory information. This is done without taking the couplings between the motion variables into account, i.e., each motion variable is updated and extrapolated independently. The estimate $\hat{x}$ of an arbitrary motion variable x at the next sampling time of this variable is computed by the following equation

$$\hat{x}(t+\Delta t) = \hat{x}(t) + p_x(x_s(t) - \hat{x}(t)) + \Delta t\, \dot{\hat{x}}(t), \tag{6.6}$$

where x_s stands for the measured value of x. The parameter p_x specifies the precision of the measurement in relation to the precision of the estimated value, $p_x = 0$ being chosen for an ideal estimate or, if there is no sensor information, and $p_x = 1$ being selected when the measurement of a variable is absolutely reliable with respect to its estimate.

The estimates of the motion variables at the next sampling time are improved by the subtasks *Positional Constraints* and *Velocity Constraints* performing a least square fit for the position and the velocity variables, respectively, taking their interdependencies into account. The performance indices defining these subtasks were introduced in (5.13) and (5.18). The only parameters left to determine are the elements of the weight matrices. The weights of the constraint equations, i.e., the elements of the diagonal matrices D_{1pos} and D_{1vel} are chosen to be one. With the remaining elements of the weight matrices we can specify how "good" the estimates of the position and velocity variables, $\underline{z}_{pos}$ and $\underline{z}_{vel}$, are (cf. (6.8) and (6.9)). We know that the cart can only move in the horizontal direction, and we have the sensor information for the hinge coordinates φ'_{22_3}, s'_{11_1}, plus their first time derivatives. Therefore, the diagonal elements of D_{0pos} and D_{0vel}, united in the two vectors $\underline{d}_{0pos}$ and $\underline{d}_{0vel}$, are selected to be

$$\underline{d}_{0pos}^{T} = \underline{d}_{0vel}^{T} = \left[1\ 10^{-8}\ 10^{-8}\ 1\ 10^{-8}\ 10^{-8}\ 1\ 1\right], \tag{6.7}$$

for

$$\underline{z}_{pos}^{T} = \left[\Delta\vartheta_{1_3}\ \Delta\vartheta_{2_3}\ \Delta X_{1_1}\ \Delta X_{1_2}\ \Delta X_{2_1}\ \Delta X_{2_2}\ \Delta\varphi'_{22_3}\ \Delta s'_{11_1}\right], \tag{6.8}$$

$$\underline{z}_{vel}^{T} = \left[\omega_{1_3}\ \omega_{2_3}\ \dot{X}_{1_1}\ \dot{X}_{1_2}\ \dot{X}_{2_1}\ \dot{X}_{2_2}\ \theta'_{22_3}\ \dot{s}'_{11_1}\right]. \tag{6.9}$$

With these weights the kinematical constraints are fulfilled by adjusting the variables R_2, X_{1_1}, X_{2_1}, X_{2_2}, ω_{2_3}, $\dot{X}_{1_1}$, $\dot{X}_{2_1}$, and $\dot{X}_{2_2}$ appropriately, performing very small changes only for the rest of the variables. (The update of the spinor R_2 from $\Delta\vartheta_{2_3}$ is given by (3.47).)

Coordination Level

The only subtask of the coordination level is the determination of the necessary changes in the motion of the system and of their importance based on the updated and extrapolated information from the information processing level. This is achieved with the following, explicit laws for the demanded accelerations (cf. Figure 6.3):

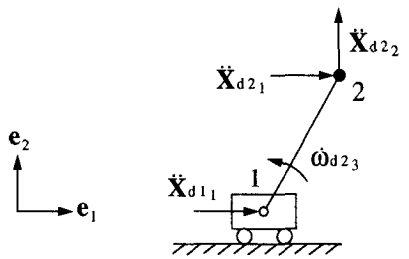


Figure 6.3 Demanded Accelerations

$$\ddot{X}_{d1_1} = p_{11}(X_{2_1} - X_{1_1}) + p_{12}(\dot{X}_{2_1} - \dot{X}_{1_1}) + (1 + p_{13}X_{2_2})\ddot{X}_{d2_1}, \quad (6.10)$$

$$\dot{\omega}_{d2_3} = p_{21}(0 - 2\,\mathrm{atan}(b_{2_3}/a_2)) + p_{22}(0 - \omega_{2_3}), \quad (6.11)$$

$$\ddot{X}_{d2_1} = p_{31}(0 - X_{2_1}) + p_{32}(0 - \dot{X}_{2_1}), \quad (6.12)$$

$$\ddot{X}_{d2_2} = p_{41}(1 - X_{2_2}) + p_{42}(0 - \dot{X}_{2_2}). \quad (6.13)$$

These are simple PD-type control laws in Cartesian space, except for the term $p_{13}X_{2_2}\ddot{X}_{d2_1}$ in (6.10) introducing a coupling between the demanded horizontal cart movement and the demanded horizontal pole movement. It can be interpreted as follows: Assume that $X_{2_1} - X_{1_1} = 0$, $\dot{X}_{2_1} - \dot{X}_{1_1} = 0$, $\ddot{X}_{2_1} = 0$, and that the demanded horizontal acceleration for the tip of the pole, $\ddot{X}_{d2_1}$, is positive (cf. positions A and B in Figure 6.4). If we choose $p_{13} = -0.1$, for example, we achieve that in situation A ($X_{2_2} = 1$) the cart is decelerated first to produce the desired horizontal acceleration of the tip of the pole. In situation B ($X_{2_2} = -1$), on the other hand, the same goal is attained with an acceleration of the cart rather than a deceleration. The values for the other parameters of (6.10), (6.11), (6.12), and (6.13) will be given in Section 6.1.3.

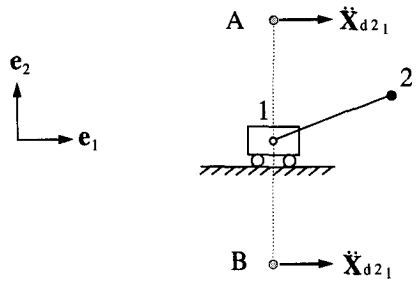


Figure 6.4

The weighting factor for $\ddot{X}_{d2_1}$ is determined to be $d\ddot{x}2_1 = 10^{-5}$. The other weighting factors are defined by the graphs in Figure 6.5. With these weighting factors we mainly prescribe three things. First, the balancing of the pole has higher priority than the position control of the cart close to the demanded position ($X_{1_1} = 0$, $\varphi'22_3 = 0$, $X2_1 = 0$ $X2_2 = 1$). Second, the demanded accelerations $\dot{\omega}_{d2_3}$ and $\ddot{X}_{d2_2}$ have low priority if the pole is close to the horizontal position ($\varphi'22_3 = \pm\frac{\pi}{2}$ and $X2_2 = 0$). The reason for this is that the control force to produce an arbitrarily small acceleration $\dot{\omega}_{d2_3}$ or $\ddot{X}_{d2_2}$ goes towards infinity when the rod comes close to the horizontal position (cf. Example 5.5). This is prevented by the proposed weighting factors. Finally, the values of $d\dot{\omega}2_3$ and $d\ddot{x}2_2$ are reduced to 10^{-6} in the lower vertical position of the rod. This is to reduce the influence of the sign change of $\dot{\omega}_{d2_3}$ and to take into account that both demanded accelerations typically reach their maximum value in that position.

Execution Level

The execution level is concerned with the calculation of the control force vector **u** and with the estimates for the accelerations and the contact forces. This is achieved by the subtask *Dynamical Constraints*. Additionally, the subtask *Boundary Test* evaluates whether or not the positional boundary for the hinge coordinate $s'11_1$ is active (expressed by $d\ddot{s}'11_1 = 1$).

The performance index for the subtask *Dynamical Constraints* was introduced in (5.29). Again, we choose the elements of D_{2dyn} to be equal to one, except for the weight of the force constraint (6.4), which is only valid if the boundary constraint is not active. Henceforth, the corresponding weight is $1 - d\ddot{s}'11_1$. The diagonal elements of D_{0dyn} are determined to be

$$\underline{d}_{0dyn}^{T} = \tag{6.14}$$

$$\left[10^{-6}\ 1\ d\dot{\omega}2_3\ d\ddot{x}1_1\ 1\ 10^{-5}\ d\ddot{x}2_2\ 10^{-7}\ 10^{-7}\ 10^{-7}\ 10^{-7}\ 10^{-7}\ 10^{-7}\ 10^{-7}\ d\ddot{s}'11_1\right],$$

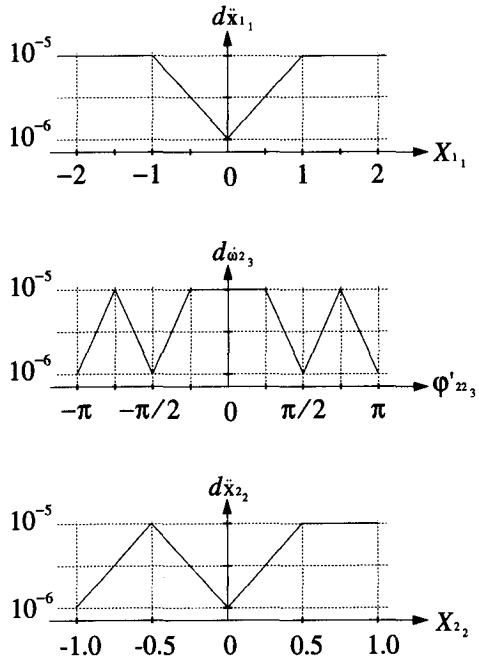


Figure 6.5 Weighting Factors

for

$$\mathbf{z}_{dyn}^{T} = \left[u\ \dot{\omega}_{1_3}\ \dot{\omega}_{2_3}\ \ddot{X}_{1_1}\ \ddot{X}_{1_2}\ \ddot{X}_{2_1}\ \ddot{X}_{2_2}\ \tau_{1_3}\ \tau_{2_3} f_{1_1} f_{1_2} f_{2_1} f_{2_2}\ \dot{\theta}'_{22_3}\ \ddot{s}'_{11_1}\right]. \tag{6.15}$$

The demanded values for all variables except for those determined in the coordination level, i.e., $\dot{\omega}_{2_3}$, $\ddot{X}_{1_1}$, $\ddot{X}_{2_1}$, and $\ddot{X}_{2_2}$, are set to zero. (For $d\dot{\omega}_{2_3}$, $d\ddot{x}_{1_1}$, and $d\ddot{x}_{2_2}$ cf. Figure 6.5.)

This means that the controller evaluates a control signal, keeping the positional constraints ($\dot{\omega}_{1_3} = 0$, $\ddot{X}_{1_2} = 0$) and $\ddot{s}'_{11_1}$, if necessary, in first priority. Second, the controller tries to produce the demanded accelerations

$\dot{\omega}_{d2_3}$, $\ddot{X}_{d1_1}$, $\ddot{X}_{d2_1}$, and $\ddot{X}_{d2_2}$. Third, too large a control force u is avoided and, last, the hinge forces and torques τ_{1_3}, τ_{2_3}, f_{1_1}, f_{1_2}, f_{2_1}, f_{2_2} and the rotational acceleration $\dot{\theta}'_{22_3}$ are held as low as possible.

The evaluation of the demanded acceleration $\ddot{s}'_{d11_1}$ and weighting factor $d_{\ddot{s}'11_1}$ for the next control step are computed in the last subtask, designated as *Boundary Test*. With these two variables, we define the physical properties of the boundary constraint. In this benchmark they are specified as listed in Table 6.2:

	New Value	Condition
$d_{\ddot{s}'11_1}$	1	$d_{\ddot{s}'11_1} = 0$ and $\delta s'_{\min 11_1} > 0$ and $\delta\ddot{s}'_{11_1} > 0$
	1	$d_{\ddot{s}'11_1} = 0$ and $\delta s'_{\max 11_1} < 0$ and $\delta\ddot{s}'_{11_1} < 0$
	0	$d_{\ddot{s}'11_1} = 1$ and $(\delta s'_{\min 11_1} \le 0$ or $\delta\ddot{s}'_{11_1} \le -\varepsilon$ or $f'_{11_1} < 0)$
	0	$d_{\ddot{s}'11_1} = 1$ and $(\delta s'_{\max 11_1} \ge 0$ or $\delta\ddot{s}'_{11_1} \ge \varepsilon$ or $f'_{11_1} > 0)$
	$d_{\ddot{s}'11_1}$	else

Table 6.2

with

$$\begin{aligned} \delta s'_{\min 11_1} &= s'_{\min 11_1} - s'_{11_1} = -5 - s'_{11_1}, \\ \delta s'_{\max 11_1} &= s'_{\max 11_1} - s'_{11_1} = 5 - s'_{11_1}, \\ \delta\ddot{s}'_{11_1} &= \ddot{s}'_{d11_1} - \ddot{s}'_{11_1} = -100\ \dot{s}'_{11_1} - \ddot{s}'_{11_1}. \end{aligned} \tag{6.16}$$

Thus, the boundary constraint is modeled "ideal" (cf. Figure 5.4) with purely plastic characteristics, i.e., $d_{\ddot{s}'11_1}$ is either 1 or 0, and $\ddot{s}'_{d11_1}$ depends on the velocity $\dot{s}'_{11_1}$ only.

6.1.3 Simulation Results

All the simulations are performed using a variable step Kutta-Merson integrator. In a first step, the controller is tuned for a starting position A (i.e., pole hanging down vertically, 1 m to the right of the origin of the reference frame). Additionally, if not mentioned otherwise, an actuator force limit of $u \in [-20\,\mathrm{N}, 20\,\mathrm{N}]$ is introduced for the simulations, and the measurements of the hinge coordinates φ'_{22_3}, s'_{11_1} and their first time derivatives are assumed to be exact. Henceforth, the parameters of the sensor fusion, $p_{\varphi'22_3}$, $p_{s'11_1}$, $p_{\theta'22_3}$, and $p_{\dot{s}'11_1}$, are chosen to equal one (cf. (6.6)). The testing is done for a new starting position B (i.e., pole hanging down vertically in the origin of the reference frame). The aim is to study the robustness of the controller versus parameter errors, measurement noise, and an increase of the sampling period.

Tuning

Good results were achieved by selecting the parameters in (6.10) to (6.13) for critical damping. The "stiffness" to be chosen (parameters p_{i1}) is directly proportional to the response time desired.

For this benchmark, a fast response of the horizontal cart movement (i = 1) is considered to be of prime importance, followed by 2) the response time of the rotation (i = 2) and the vertical movement of the pole (i = 4), and 3) by the horizontal pole movement (i = 3). Additionally, the horizontal pole movement is demanded to be overcritically damped to avoid strong oscillations. The corresponding parameters are given in Table 6.3.

As an illustration of the dependency of the system behavior on these parameters we present some additional simulation results for three different cases. Compared with the nominal parameters of Table 6.3, only the parameters for the rotation and the vertical movement of the pole, as well as the parameter p_{13} are being varied.

i	1	2	3	4
p_{i1}	30	20	10	20
p_{i2}	$2\sqrt{30}$	$2\sqrt{20}$	$10\sqrt{10}$	$2\sqrt{20}$
p_{i3}	-0.5	—	—	—

Table 6.3 Parameters from the Coordination Level

In the first case, we choose a slower response time ($p_{21} = p_{41} = 10$, $p_{13} = -0.5$) and again critical damping. For the second case, we select a faster response time ($p_{21} = p_{41} = 120$, $p_{13} = -1$), critical damping, and neglect the force limits. Finally, in the third case we use the parameters from the second case but take the force limits into account again. Figures 6.6, 6.7, and 6.8 show the behavior of the controlled system for all three cases, respectively.

In case one (Figure 6.6), the pole is brought into the upper vertical position in the third try, whereas the controller succeeds in doing so in the first try in case two (Figure 6.7). This is at the cost of a maximum force which is about five times as high as in the first case (approximately 100 N). If the demanded force above or below the force limit is simply clipped, we get the behavior depicted in Figure 6.8. We can see that the pole falls down one more time before being stabilized in the demanded position. The maximum demanded horizontal control force is about ten times as high as in the first case (approximately 200 N). Figure 6.9 depicts the movement of the tip of the pole for all three cases, whereas Figures 6.10 and 6.11 show the time histories of the horizontal and vertical movement of the tip of the pole.

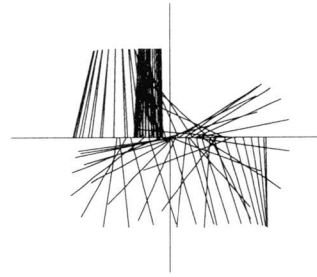


Figure 6.6 Case 1

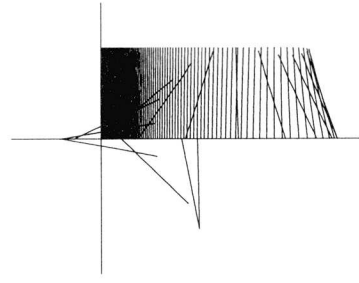


Figure 6.7 Case 2

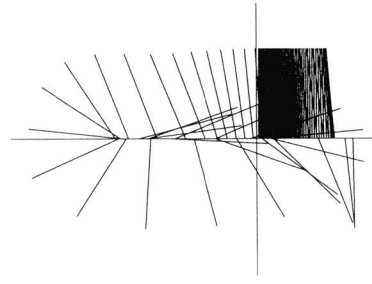


Figure 6.8 Case 3

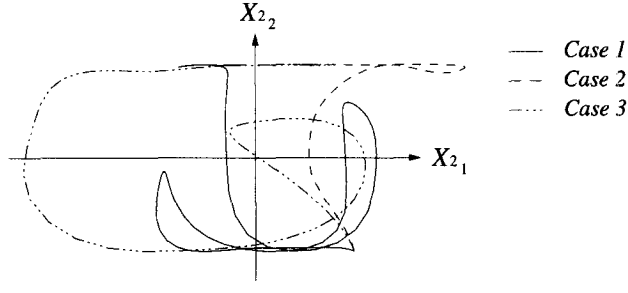


Figure 6.9 Movement of the Tip of the Pole

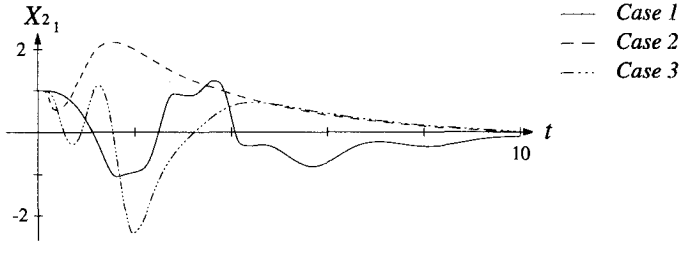


Figure 6.10 Horizontal Movement of the Tip of the Pole

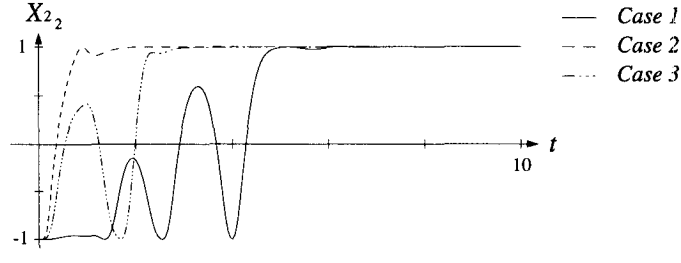


Figure 6.11 Vertical Movement of the Tip of the Pole

Testing

First of all, the controller developed is tested for the initial position B (case 4), yielding the simulation results shown in Figure 6.12.

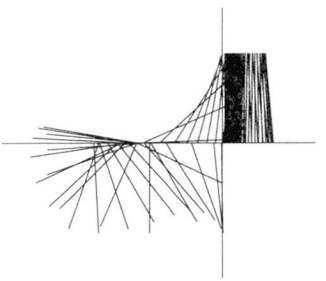


Figure 6.12 Case 4

Second, we examine the robustness of this controller with respect to the estimation errors of the system parameters m_1, m_2, (i.e., masses of both bodies) and l (rod length). For the simulations, these variables are increased or decreased by 20% and 45%, respectively, in the worst direction. This occurs when the true inertia of the cart is higher than the estimated one while the true inertia of the pole is lower. In this case, the controller overestimates its capability to act on the pole through the cart and underestimates the pole's sensitivity. Thus, the estimates for the object variables are chosen as $m_1 = 0.8$, $m_2 = 0.12$, and $l = 1.2$ for case 5, and as $m_1 = 0.55$, $m_2 = 0.145$, and $l = 1.45$ in case 6. The movement of the pole in these two cases compared with case 4 is depicted in Figures 6.13, 6.14, and 6.15. The system is stable in both cases, but the damping of the oscillations decreases drastically from case 5 to case 6. When the parameter errors reach 50%, the proposed controller can no longer stabilize the system.

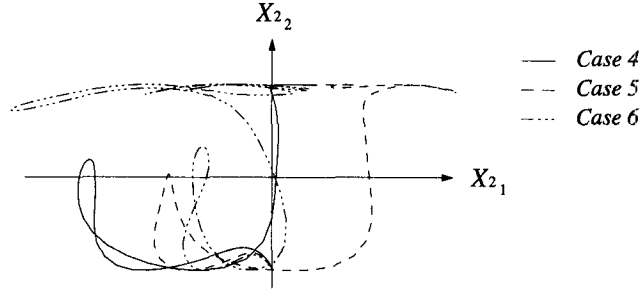


Figure 6.13 Movement of the Tip of the Pole

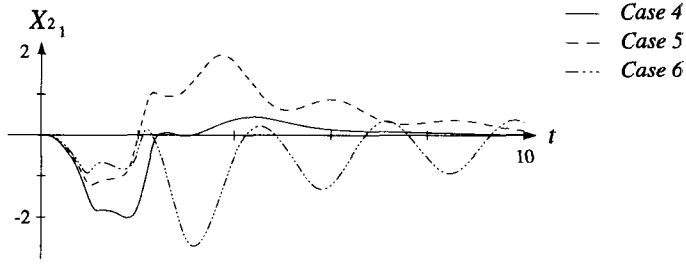


Figure 6.14 Horizontal Movement of the Tip of the Pole

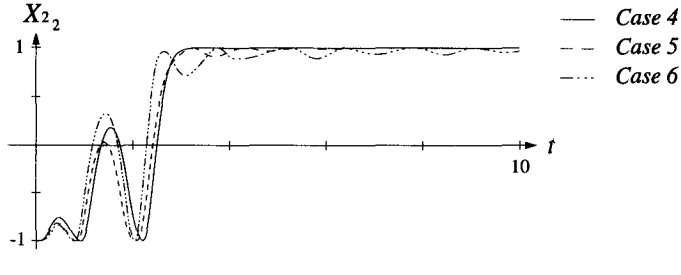


Figure 6.15 Vertical Movement of the Tip of the Pole

Third, we consider measurement noise. Let us assume that we have a measurement error with a Gaussian distribution and standard deviations of

$$\begin{aligned} \sigma_{\varphi'22_3} &= 0.1\pi \text{ [rad]}, & \sigma_{\theta'22_3} &= 0.1\pi \text{ [rad/s]}, \\ \sigma_{s'11_1} &= 0.1 \text{ [m]}, & \sigma_{\dot{s}'11_1} &= 0.1 \text{ [m/s]}. \end{aligned} \tag{6.17}$$

For case 7, we still select the parameters $p_{\varphi'22_3}$, $p_{s'11_1}$, $p_{\theta'22_3}$, and $p_{\dot{s}'11_1}$ of the sensor fusion to be one (cf. (6.6)), i.e., there is no sensor fusion. These parameters are chosen to equal 0.4 in case 8. Again, the system can be stabilized for both cases, but in case 8 the behavior of the system is much closer to that of the undisturbed system (cf. Figures 6.16, 6.17, and 6.18).

Finally, we examine the robustness of the controller with respect to the sampling interval (100 Hz is the nominal case). This is done for a sampling interval of 0.03 (case 9) and 0.06 (case 10) and then compared with case 4 (cf. Figures 6.19, 6.20, and 6.21). For a sampling interval of 0.07, the suggested controller can no longer solve the task.

The simulations show that the proposed hierarchical controller can stabilize the given nonlinear plant in the complete workspace with good performance and very robust behavior even if the controller parameters are chosen in a rather intuitive and simple way.

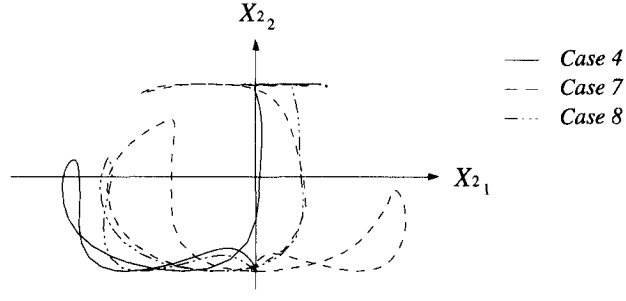


Figure 6.16 Movement of the Tip of the Pole

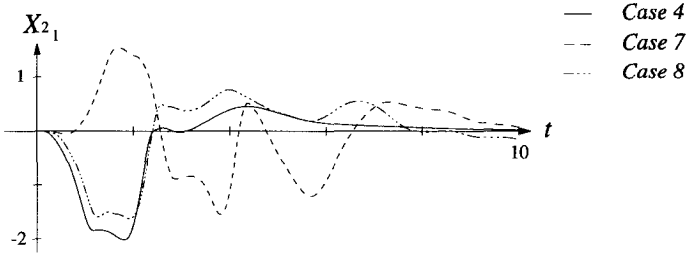


Figure 6.17 Horizontal Movement of the Tip of the Pole

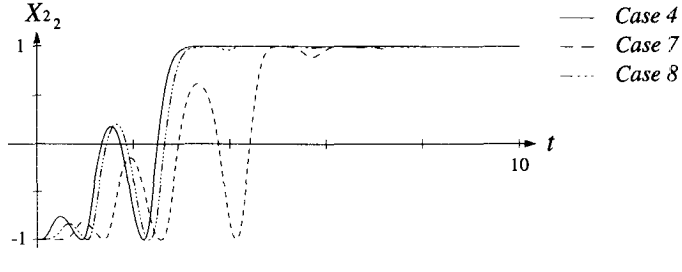


Figure 6.18 Vertical Movement of the Tip of the Pole

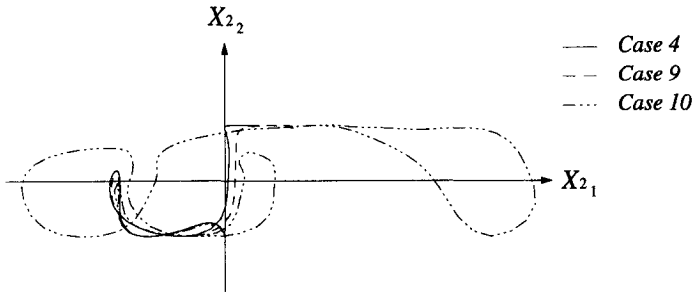


Figure 6.19 Movement of the Tip of the Pole

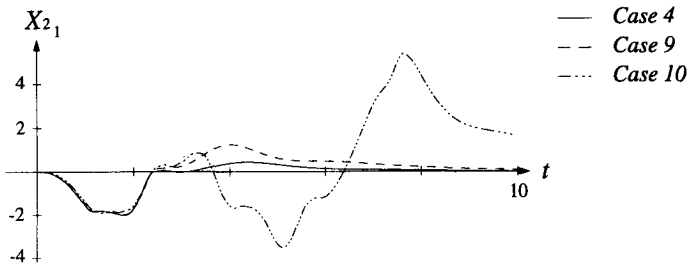


Figure 6.20 Horizontal Movement of the Tip of the Pole

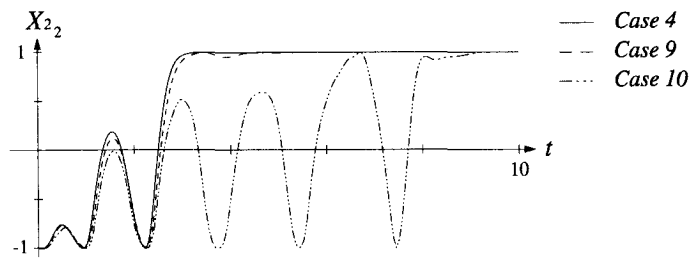


Figure 6.21 Vertical Movement of the Tip of the Pole

6.2 Walking Robot

6.2.1 Problem Statement

The second benchmark describes the modelling and control of a biped robot walking in a vertical plane (cf. Figure 6.22). The plant consists of 11 rigid bodies connected by 14 hinges (cf. Figure 6.26) having a total of 14 rotational and 8 translational degrees of freedom. Ten of the rotational joints are actuated (hips, knees, ankles, shoulders, and elbows). All actuated rotational joints have upper and lower boundaries. In addition, the translational movement of both feet is restricted by the ground. The friction between the heels and the balls of the feet and the ground can either be static or sliding. As sensor information there are the joint angles of all actuated joints and their first time derivatives. Additionally, we have four binary signals indicating whether or not the heels and the balls of the feet touch the ground.

The task is the development of a controller enabling the robot to stably walk with the demanded horizontal velocity of the upper body as the only feedforward information. All other decisions have to be taken on-line by the controller. The sampling rate is again fixed at 100 Hz.

This benchmark may seem to be a little artificial at first sight. The walking robot was mainly chosen for two reasons. It possesses all the properties for which the control algorithm was developed, such as

- complex kinematics and dynamics,
- redundancies, and
- unilateral constraints,

on the one hand. On the other hand we can intuitively understand the dynamic behavior of this walking robot due to our own human experience.

This simplifies the judgement of the simulation results and of the controller behavior.

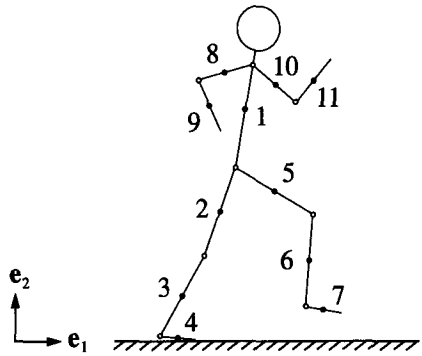


Figure 6.22 Walking Robot

The main source of troubles lies in the unilateral constraints, especially the constraints between the feet and the ground. Hinges 1, 2, 3, 6, 7, 8, 11, 12, 13, and 14 (cf. Figures 6.22 and 6.26) each possess two different states (static contact and free), while the hinges 4, 5, 9, and 10 change between three states (static contact, sliding contact, and free). Summing up, this gives us a total of $2^{10}3^4 = 82'944$ different states for the overall system, i.e., working with explicit models we would have to switch between more than 80'000 models as the behavior of the plant in these different states changes significantly. There are even states where the plant is not completely observable (fewer than two of the hinges 4, 5, 9, and 10 in static contact with the ground) or not completely controllable (fewer than two feet in static contact with the ground). The controller has to 1) determine the actual state of the system, 2) decide if this state has to be changed to generate the desired action, and 3) prevent control actions leading to unwanted changes of the actual state of the plant. The following three examples illustrate this:

Example 6.1

Assume that the robot is in the position depicted in Figure 6.23, i.e., immediately before setting the ball of the left foot (body 7) down on the ground.

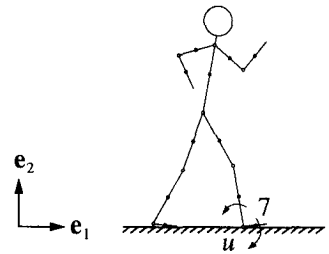


Figure 6.23 Example 6.1

In this situation the actuator torque typically changes from approximately -1 [Nm] to 30 [Nm] immediately after touching the ground. Assume that the controller considers the ball of the foot to be on the ground while it is actually still in the air. This misjudgment of the momentary situation would result in the enormous rotational acceleration of the foot of about -5000 $[\mathrm{rad}/\mathrm{s}^2]$ ($I = 0.006$ $[\mathrm{kg\,m}^2]$ and $I\dot{\omega} = u$) instead of the expected acceleration of about 200 $[\mathrm{rad}/\mathrm{s}^2]$. Thus, the system behavior is extremely sensitive to the correct estimation of the boundary constraints between the feet and the ground. This is why it is necessary to have sensors detecting this contact.

Example 6.2

Suppose that the robot is positioned as shown in Figure 6.24 and that it should move in the horizontal direction without bending the upper body forward. This is only possible if the heel, and later on the ball of the right foot (body 4), are lifted off the ground. Thus, the controller has to decide that the corresponding unilateral constraints have to be

deactivated even if they are still active for the plant, i.e., if they still carry weight.

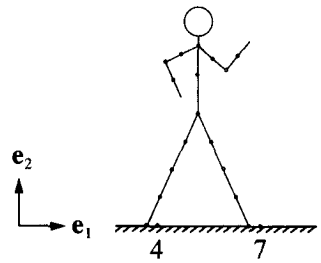


Figure 6.24 Example 6.2

Example 6.3

Another critical situation is shown in Figure 6.25 where the robot touches the ground with one foot and should be accelerated in a positive horizontal direction with $\ddot{\mathbf{X}}_{d1_1}$. The problem with unilateral constraints is that from the controller's point of view they are bilateral constraints which are switched on and off when the constraint forces point in the wrong direction. Unfortunately, the constraint forces depend on the control vector. The decision of whether or not a unilateral constraint is still valid is, therefore, only possible after evaluation of the control vector. In the situation of Figure 6.25 the controller assumes that the vertical acceleration of the left foot (body 7) equals zero. The only possibility for generating the desired horizontal acceleration $\ddot{\mathbf{X}}_{d1_1}$ in this situation is to pull the upper body forward with the foot resulting in a negative contact force $\mathbf{f}$. As the "real" constraint, i.e., the constraint of the plant, cannot generate a negative constraint force, the constraint between the ball of the foot and the floor will be deactivated. For a negative constraint force $\mathbf{f}$ of only -10 [N], for example, this results in a rotational acceleration of approximately 90 $[\mathrm{rad}/\mathrm{s}^2]$ (the foot length being 0.2 m, this results in a torque error $\Delta\tau$

of 2 [Nm]; I = 0.022 [kg m^2], and $I\dot{\omega} = \Delta\tau$). As the controller estimates this acceleration to be equal to zero, this leads to an error for the next extrapolated rotational velocity of about 0.9 [rad/s] ($\Delta t\,(\dot{\omega} - \hat{\dot{\omega}})$). Due to the fact that we can measure the relative rotational velocities only, this error in the absolute rotational velocity of the foot will propagate through the whole structure and completely deteriorate the estimates for the positions and velocities of the whole plant. Therefore, it is of great importance to keep the system from making any unwanted structure changes.

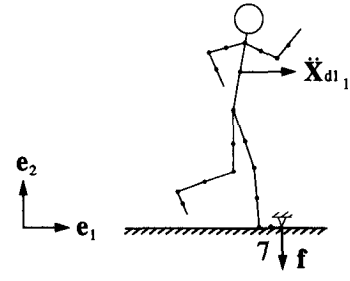


Figure 6.25 Example 6.3

The following paragraphs briefly describes the kinematical and dynamical properties of the plant. The interconnection structure is depicted in Figure 6.26, while the object variables are given in Table 6.4.

The robot possesses rotational joints in the hips, the knees, the ankles, the shoulders and the elbows (hinges 1, 2, 3, 6, 7, 8, 10, 11, 12, 13, 14). Hinges 4, 5, 9, and 10 each possess one rotational and two translational degrees of freedom. The translational degrees of freedom describe the position of the heels and the balls of the feet (cf. Figures 6.22 and 6.26). All hinges of the walking robot are of the following type:

$$\{S'_a \mid \mathbf{s}'_a\} = \{a'_{a2} + i\mathbf{b}'_{a2} \mid \mathbf{s}'_{a2}\}\,\{1 \mid \mathbf{s}'_{a1}\}. \tag{6.18}$$

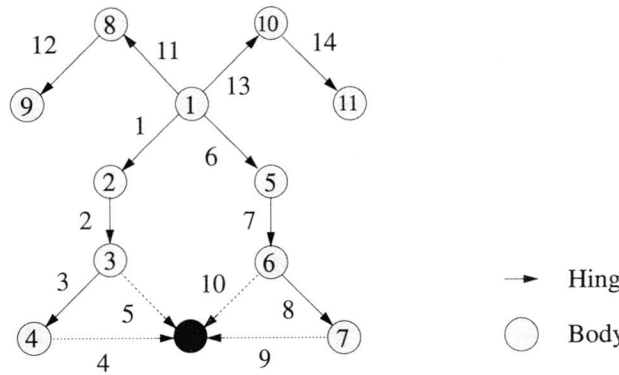


Figure 6.26 Interconnection Structure of Walking Robot

α	m_α [kg]	I_{α_3} [kg m^2]
1	47.0	2.25
2, 5	9.0	0.167
3, 6	5.0	0.09
4, 7	1.6	0.006
8, 9, 10, 11	2.35	0.019

Table 6.4 Object Variables

The values for $\mathbf{s}'_{a1}$, $\mathbf{b}'_{a2}$, and $\mathbf{s}'_{a2}$, the kinematical constraints, and the physical properties of the hinges are given in Tables 6.5, 6.6, and 6.7, respectively. Notice that in contrast with the former benchmark we additionally have velocity constraints in the horizontal direction (cf. Table 6.6). They are only valid for static contact, i.e., if $|f'_{a2\,1}| < \mu_0 |f'_{a2\,2}|$. On the other hand, we have force constraints in the horizontal direction for sliding contact (cf. Table 6.7)

a	$\mathbf{s}'_{a1}$	$\mathbf{b}'_{a2}$	$\mathbf{s}'_{a2}$
1, 6	$0.32\mathbf{e}_2$	$b'_{a2_3}\mathbf{e}_3$	$0.225\mathbf{e}_2$
2, 7	$0.225\mathbf{e}_2$	$b'_{a2_3}\mathbf{e}_3$	$0.225\mathbf{e}_2$
3, 8	$0.225\mathbf{e}_2$	$b'_{a2_3}\mathbf{e}_3$	$-0.1\mathbf{e}_1$
4, 9	$-0.1\mathbf{e}_1$	$b'_{a2_3}\mathbf{e}_3$	$(s'_{a2_1} - 0.1)\,\mathbf{e}_1 + s'_{a2_2}\mathbf{e}_2$
5, 10	$0.1\mathbf{e}_1$	$b'_{a2_3}\mathbf{e}_3$	$(s'_{a2_1} - 0.1)\,\mathbf{e}_1 + s'_{a2_2}\mathbf{e}_2$
11, 13	$-0.18\mathbf{e}_2$	$b'_{a2_3}\mathbf{e}_3$	$0.15\mathbf{e}_2$
12, 14	$0.15\mathbf{e}_2$	$b'_{a2_3}\mathbf{e}_3$	$0.15\mathbf{e}_2$

Table 6.5 Hinge Geometry

a	$b'_{\min a2_3}$	$b'_{\max a2_3}$	$s'_{\min a2_2}$	$\dot{s}'_{a2_1}$
1, 6	$\sin(-\frac{1}{2}\frac{1}{2}\pi)$	$\sin(\frac{1}{2}\frac{1}{2}\pi)$	—	—
2, 7	$\sin(-\frac{1}{2}\frac{1}{72}\pi)$	$\sin(\frac{1}{2}\pi)$	—	—
3, 8	$\sin(-\frac{1}{2}\frac{1}{4}\pi)$	$\sin(\frac{1}{2}\frac{1}{2}\pi)$	—	—
4, 5, 9, 10	—	—	0	0
11, 13	$\sin(-\frac{1}{2}\pi)$	$\sin(\frac{1}{2}\frac{1}{2}\pi)$	—	—
12, 14	$\sin(-\frac{1}{2}\pi)$	0	—	—

Table 6.6 Kinematic Constraints

a	τ'_{a2_3}	f'_{a2_1}	f'_{a2_2}
1, 2, 3	u_a	—	—
4, 5, 9, 10	0	$-\mu_0 \mathrm{sgn}(\dot{s}'_{a2_1})\, f'_{a2_2}$	0
6, 7, 8	$u_{a\text{-}2}$	—	—
11, 12, 13, 14	$u_{a\text{-}4}$	—	—

Table 6.7 Physical Properties of the Hinges

6.2.2 Solution

The controller used to solve this problem possesses nearly the same structure as the controller of the cart-pole system. Therefore, in this section, we will only describe the differences as compared to the controller of the first benchmark. The main changes can be found in the *Coordination Level* as it is, obviously, much more difficult to define the demanded actions for stable walking than for the balancing of a pole.

Information Processing Level

The only changes at the information processing level, compared with the first benchmark, are the additional binary sensor signals c_{a_2} indicating whether the heels and the balls of both feet touch the ground. They are defined as follows:

$$c_{a_2} = \begin{cases} 1 & \text{if } s'_{a2_2} \leq 0 \\ 0 & \text{else} \end{cases} \qquad a = \{4, 5, 9, 10\} \tag{6.19}$$

In the previous examples we referred to bodies 2, 3, and 4 as the right leg and to bodies 5, 6, and 7 as the left leg. Thus, $c_{9_2} = 1$ indicates that the ball of the left foot of the robot is touching the ground (cf. Figures 6.22 and 6.26).

Coordination Level

Before describing the coordination level in detail we introduce the two additional operators *limit* and *interp*. The clipping operator *limit* and the operator for the linear interpolation, *interp*, are defined as follows:

$$limit_{[a,b]}(x) = \begin{cases} a & \text{if } x \le a \\ x & \text{if } a < x < b \\ b & \text{if } x \ge b \end{cases} \tag{6.20}$$

and

$$interp\{(a_1, b_1), \ldots, (a_n, b_n)\}(x) = \begin{cases} b_1 & \text{if } x < a_1 \\ \ldots & \\ b_i + \dfrac{b_{i+1} - b_i}{a_{i+1} - a_i}(x - a_i) & \text{if } a_i \le x < a_i \\ \ldots & \\ b_n & \text{if } x \ge a_n \end{cases} \tag{6.21}$$

Example 6.4

Using the operator *interp*, the weighting factor $d\ddot{x}_{1_1}$ from Figure 6.5 would be defined as follows:

$$d\ddot{x}_{1_1} = interp\{(-2, 10^{-5}), (-1, 10^{-5}), (0, 10^{-6}), (1, 10^{-5}), (2, 10^{-5})\}(X_{1_1}). \tag{6.22}$$

The main difficulties to be solved at the coordination level are the correct estimation of the momentary situation and the decision regarding the appropriate actions to be taken. Estimation of the momentary situation has to be understood in a somewhat broader sense than usual in control. Rather than the estimation of the physical state variables, which is achieved at the information processing level, it is the estimation of the variables describing the actual configuration of the plant, e.g., whether the robot should perform a step or not. This kind of information is represented by "fuzzy" variables possessing a value between 0 and 1. They are calculated by the subtask *State Detection*. The resulting necessary actions are then determined by the three subtasks *Feet Coordination*, *Demanded Contact Forces*, and *Demanded Accelerations* (cf. Figure 6.27). Notice that in contrast with the first benchmark there is one feedforward signal ($\dot{X}_{d1_1}$).

Demanded Accelerations (cf. Figure 6.27). Notice that in contrast with the first benchmark there is one feedforward signal ($\dot{X}_{d1_1}$).

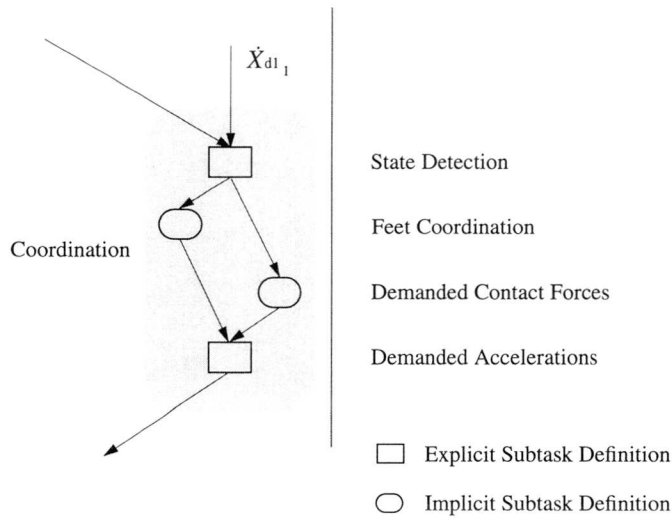


Figure 6.27 Structure of the Coordination Level

The subtask *State Detection* computes all binary and "fuzzy" variables needed to appropriately judge the actual situation. In a first step we introduce the variables IS, RL, ER, ES, CR, CL, and ST. The variable IS ("Initialize Step") is a measure for the stability of the momentary state. Obviously, the most stable situation occurs when the upper body is exactly over the center of both feet in the horizontal direction. This is when IS = 0. On the other hand, IS = 1 signifies that the momentary state is so unstable that a step has to be performed to regain stability. Variable RL ("Right" or "Left" foot) holds the information, as to whether or not the right foot, again in horizontal direction, is farther away from the upper body than the left foot. With ER ("End" step with the "Right" foot) and EL ("End" step with the "Left" foot), respectively, we define when the step phase has to be termi-

nated. The binary variables CR ("Contact" with the "Right" foot) and CL ("Contact" with the "Left" foot) are defined by the force sensor signals. These two variables simply hold the information as to whether either foot touches the ground. Finally, the variable ST ("Stumbling") detects if and to what degree the robot is stumbling. The definitions of all these variables are given in Table 6.8.

Variable	Definition
IS	$interp\{(0,0),(0.02,0),(0.02+10^{-20},1),(0.03,1)\}(x)$ $x = 2(X1_1 + 0.18\dot{X}1_1) - (X4_1 + X7_1) - \mathrm{sgn}(\dot{X}1_1)\lvert X4_1 - X7_1\rvert$
RL	$interp\{(-0.1,0),(-0.05,0),(0.05,1),(0.1,1)\}(x)$ $x = \lvert X1_1 - X4_1\rvert - \lvert X1_1 - X7_1\rvert$
ER	$interp\{(0,0),(0.4,0),(0.6,1),(1,1)\}(x)$ $x = \mathrm{sgn}(\dot{X}1_1)(X4_1 + 0.05\dot{X}4_1 - (X7_1 + 0.05\dot{X}7_1))$
EL	$interp\{(0,0),(0.4,0),(0.6,1),(1,1)\}(x)$ $x = \mathrm{sgn}(\dot{X}1_1)(X7_1 + 0.05\dot{X}7_1 - (X4_1 + 0.05\dot{X}4_1))$
CR	$limit_{[0,1]}(c4_2 + c5_2)$
CL	$limit_{[0,1]}(c9_2 + c10_2)$
ST	$interp\{(-1.5,1),(-1,1),(-0.7,0),(0,0)\}(\dot{X}1_2)$

Table 6.8 Binary and Fuzzy Variables I

Example 6.5

For RL = 1 the right foot is at least 0.05 m farther away from the upper body than the left foot, while RL = −0.25 indicates that the left foot is 0.025 m farther away from the upper body than the right foot.

Next we introduce the variables BP, SR_d, SL_d, SR, and SL. They all depend on the variables of Table 6.8; their definitions are given in Table 6.9. BP ("Balance Phase") is a binary variable. It indicates whether or not the

robot has at least one foot off the ground (BP = 1). The variables SR_d (demand for a "Step" with the "Right" foot) and SL_d (demand for a "Step" with the "Left" foot) stand for the decision to take a step with the right and left foot, respectively, while SR ("Step" with the "Right" foot) and SL ("Step" with the "Left" foot) hold the information as to whether a step should be executed or is presently being executed.

Variable	Definition
BP	$limit_{[0,1]}\,(2 - CR - CL)$
SR_d	$limit_{[0,1]}\,((IS\ RL + 1 - CR)\ (1 - ER)\ CL)$
SL_d	$limit_{[0,1]}\,((IS\ (1 - RL) + 1 - CL)\ (1 - EL)\ CR)$
SR	$limit_{[0,1]}\,(SR_d + 1 - CR)$
SL	$limit_{[0,1]}\,(SL_d + 1 - CL)$

Table 6.9 Binary and Fuzzy Variables II

Example 6.6

The definition of SR_d in Table 6.9 can be explained as follows. A step has to be performed with the right foot, i.e., ($SR_d = 1$), if the following three conditions are valid at the same time:

- the momentary state is becoming unstable and the right foot is farther away from the upper body than the left foot (IS RL > IS (1 − RL)), or the right foot does not touch the ground (1 − CR = 1)
- the step with the right foot does not have to be terminated (1 − ER > 0)
- the left foot touches the ground (CL = 1)

The variables TB_{da_2}, finally, represent the demanded values for the "Translational Boundary" flags TB_{a_2} computed by the subtask boundary

constraints ($TB_{a_2} = 1$ if the boundary constraint for variable s'_{a2_2} is active). They are defined as follows:

Variable	Definition
TB_{d4_2}	$limit_{[0,1]}((1-SR)\,TB_{4_2} + 10TB_{d4_2}ER\,(1-TB_{4_2}) + c4_2c5_2)$
TB_{d5_2}	$limit_{[0,1]}((1-SR)\,TB_{5_2} + 10TB_{d5_2}ER\,(1-TB_{5_2}))$
TB_{d9_2}	$limit_{[0,1]}((1-SL)\,TB_{9_2} + 10TB_{d9_2}EL\,(1-TB_{9_2}) + c9_2c10_2)$
TB_{d10_2}	$limit_{[0,1]}((1-SL)\,TB_{10_2} + 10TB_{d10_2}EL\,(1-TB_{10_2}))$

Table 6.10 Binary and Fuzzy Variables III

Example 6.7

Assume that the heel of the right foot is on the ground, i.e., $TB_{5_2} = 1$. In this case only the first term of the definition for TB_{d5_2}, $(1-SR)\,TB_{5_2}$, is relevant. This term states that TB_{d5_2} is equivalent to TB_{5_2} as long as we do not want to perform a step with the right foot, i.e., for $SR = 0$. As soon as variable SR becomes greater than zero, variable TB_{d5_2} continuously goes towards zero, stating that the constraint should be deactivated. The second term is only relevant at the end of a step with the right foot ($ER > 0$). Its task is to avoid the dissolution of contact with the right heel ($TB_{5_2} = 0$, which may be propagated by subtask *Boundary Test*), e.g., due to impact phenomena. Thus, TB_{d5_2} will still be equal to one even for $TB_{5_2} = 0$ as long as $ER > 0.1$.

The subtask *Feet Coordination* determines the demanded values for the foot motion. This is an implicit subtask, i.e., a subtask defined by a performance index to be minimized. There are 6 variables to be determined:

$$\underline{z}_{Feet}^{T} = \left[X_{d4_1}\; X_{d4_2}\; X_{d7_1}\; X_{d7_2}\; \dot{X}_{d4_1}\; \dot{X}_{d7_1}\right]. \tag{6.23}$$

The 10 constraint equations or rules and the corresponding weights are given in Table 6.11. Rules 1, 2, and 5 define the state of the right foot at rest. On the other hand, rules 7, 8, and 9 characterize the demanded motion for the right foot in the step phase. The same is true for the left foot for the rules 3, 4, 6, and 7, 8, 10, respectively. The actual validity of these rules is given by the weights.

Rule	Constraints	Weights d_{step_i}
1	$X_{d4_1} - X_{4_1} = 0$	$0.01\,(1 - \text{BP}\ \text{SR}_d)$
2	$X_{d4_2} + 0.02 = 0$	$15\,(1 - \text{SR}_d + \text{ST})$
3	$X_{d7_1} - X_{7_1} = 0$	$0.01\,(1 - \text{BP}\ \text{SL}_d)$
4	$X_{d7_2} + 0.02 = 0$	$15\,(1 - \text{SL}_d + \text{ST})$
5	$\dot{X}_{d4_1} = 0$	$0.01\,(1 - \text{BP}\ \text{SR}_d)$
6	$\dot{X}_{d7_1} = 0$	$0.01\,(1 - \text{BP}\ \text{SL}_d)$
7	$2\,(X_{1_1} + 0.8\dot{X}_{1_1}) - X_{d4_1} - X_{d7_1} = 0$	$\text{BP}\,(\text{SR}_d + \text{SL}_d)$
8	$2\dot{X}_{1_1} - \dot{X}_{d4_1} - \dot{X}_{d7_1} = 0$	$\text{BP}\,(\text{SR}_d + \text{SL}_d)$
9	$X_{d4_2} - 0.14 = 0$	$\text{SR}_d\,(1 - \text{ST})$
10	$X_{d7_2} - 0.14 = 0$	$\text{SL}_d\,(1 - \text{ST})$

Table 6.11 Constraints and Weighting Factors of Subtask "Feet Coordination"

Example 6.8

Assume that both feet are on the ground (BP = 0), that a step should be performed with the right foot (SR_d: $0 \rightarrow 1$, $\text{SL}_d = 0$), and that the robot is not stumbling (ST = 0). With this constellation rule 2 is deactivated while rule 9 is activated. All other weighting factors

remain unchanged. Thus, the demanded value X_{d4_2} continuously goes from -0.02 to 0.14. As soon as the right foot loses contact with the ground (BP: $0 \rightarrow 1$), rules 1 and 5 are deactivated and rules 7 and 8 are activated. These rules define the step phase. They guide the foot in such a way that the upper body is always more or less in-between both feet in the horizontal direction.

Remark 6.1:

For walking in arbitrary terrain we only have to change rules 9 and 10 as the step height in this case is no longer constant. The new constraints could, for example, look like this:

$$\begin{aligned} X_{d4_2} - (\hat{X}_{0_2} + 0.14) &= 0, \\ X_{d7_2} - (\hat{X}_{0_2} + 0.14) &= 0, \end{aligned} \tag{6.24}$$

where $\hat{X}_{0_2}$ is a guess for the terrain height.

The subtask *Demanded Contact Forces* is a rather important task to avoid the problems described in Example 6.3. It computes a rough estimate of the contact forces for the feasible demanded accelerations of the upper body. Rough estimate means that all the dynamics except for those of the upper body are disregarded, that all masses are assumed to have zero velocity, and that we neglect sliding contact. For this case the relations between the accelerations of the upper body and the contact forces of the feet with the ground can be expressed by the following three equations:

$$\begin{aligned} &\hat{f}_{d4_2}(s'_{42_1} - X_{1_1}) + \hat{f}_{d5_2}(s'_{52_1} - X_{1_1}) + \hat{f}_{d9_2}(s'_{92_1} - X_{1_1}) + \\ &\hat{f}_{d10_2}(s'_{102_1} - X_{1_1}) + (\hat{f}_{d4_1} + \hat{f}_{d5_1} + \hat{f}_{d9_1} + \hat{f}_{d10_1}) X_{1_2} + \\ &m_2 g (X_{1_1} - X_{2_1}) + m_3 g (X_{1_1} - X_{3_1}) + m_4 g (X_{1_1} - X_{4_1}) + \\ &m_5 g (X_{1_1} - X_{5_1}) + m_6 g (X_{1_1} - X_{6_1}) + m_7 g (X_{1_1} - X_{7_1}) + \\ &m_8 g (X_{1_1} - X_{8_1}) + m_9 g (X_{1_1} - X_{9_1}) + m_{10} g (X_{1_1} - X_{10_1}) + \\ &m_{11} g (X_{1_1} - X_{11_1}) - I_{1_3} \dot{\omega}_{d1_1} = 0 \end{aligned} \tag{6.25}$$

$$\hat{f}_{d4_1} + \hat{f}_{d5_1} + \hat{f}_{d9_1} + \hat{f}_{d10_1} - m_1 \ddot{X}_{d1_1} = 0 \tag{6.26}$$

$$\begin{aligned} &\hat{f}_{d4_2} + \hat{f}_{d5_2} + \hat{f}_{d9_2} + \hat{f}_{d10_2} - (m_1 + m_2 + m_3 + m_4 + m_5 + \\ &m_6 + m_7 + m_8 + m_9 + m_{10} + m_{11}) g - m_1 \ddot{X}_{d1_2} = 0 \end{aligned} \tag{6.27}$$

Starting from the desired accelerations of the upper body

$$\dot{\omega}_{d1_3} = 20\,(0 - 2\,\mathrm{atan}\,(b_{1_3}/a_1)) + 2\sqrt{20}\,(0 - \omega_{1_3}), \tag{6.28}$$

$$\ddot{\hat{X}}_{d1_1} = limit_{[-2,2]}\,(2\sqrt{5}\,(\dot{X}_{d1_1} - \dot{X}_{1_1})), \tag{6.29}$$

$$\ddot{\hat{X}}_{d1_2} = 10\,(X_{d1_2} - X_{1_2}) + 2\sqrt{10}\,(0 - \dot{X}_{1_2}), \tag{6.30}$$

this subtask computes the realizable accelerations $\dot{\omega}_{d1}$, $\ddot{X}_{d1_1}$, $\ddot{X}_{d1_1}$ and the corresponding contact forces. The constraints of this subtask are given in Table 6.12. There are 14 constraints for the following 11 variables:

$$\mathbf{z}_{Foces}^{T} = \left[\dot{\omega}_{d1_3}\ \ddot{X}_{d1_1}\ \ddot{X}_{d1_1}\ \hat{f}_{d4_1}\ \hat{f}_{d4_2}\ \hat{f}_{d5_1}\ \hat{f}_{d5_2}\ \hat{f}_{d9_1}\ \hat{f}_{d9_2}\ \hat{f}_{d10_1}\ \hat{f}_{d10_2}\right]. \tag{6.31}$$

In a second step the demanded forces are postprocessed. This is to guarantee that forces in the horizontal direction, f_{da_1}, can be transmitted to the ground by the friction forces $\mu\, f_{da_2}$ (6.32), and that the forces in the vertical direction, f_{d4_2}, are always greater than 2 N (6.33).

$$f_{da_1} = limit_{[-fd4_2, fd4_2]}\,(\hat{f}_{a_1}) \qquad a \in \{4, 5, 9, 10\} \tag{6.32}$$

$$\begin{aligned} f_{d4_2} &= limit_{[2,1600]}\,(limit_{[2,1600]}\,(\hat{f}_{d4_2}) + limit_{[-1600,2]}\,(\hat{f}_{d5_2})) \\ f_{d5_2} &= limit_{[2,1600]}\,(limit_{[2,1600]}\,(\hat{f}_{d5_2}) + limit_{[-1600,2]}\,(\hat{f}_{d4_2})) \\ f_{d9_2} &= limit_{[2,1600]}\,(limit_{[2,1600]}\,(\hat{f}_{d9_2}) + limit_{[-1600,2]}\,(\hat{f}_{d10_2})) \\ f_{d10_2} &= limit_{[2,1600]}\,(limit_{[2,1600]}\,(\hat{f}_{d10_2}) + limit_{[-1600,2]}\,(\hat{f}_{d9_2})) \end{aligned} \tag{6.33}$$

i	Constraints	Weights d_{state_i}
1	$\dot{\omega}_{d1_3} - \dot{\hat{\omega}}_{d1_3} = 0$	10^{-4}
2	$\ddot{X}_{d1_1} - \ddot{\hat{X}}_{d1_1} = 0$	10^{-4}
3	$\ddot{X}_{d1_2} - \ddot{\hat{X}}_{d1_1} = 0$	10^{-4}
4	$\hat{f}_{d4_1} = 0$	$10^{-4}(1 - \text{TB}_{d4_2}) + 10^{-10}$
5	$\hat{f}_{d4_2} = 0$	$10^{-4}(1 - \text{TB}_{d4_2}) + 10^{-10}$
6	$\hat{f}_{d5_1} = 0$	$10^{-4}(1 - \text{TB}_{d5_2}) + 10^{-10}$
7	$\hat{f}_{d5_2} = 0$	$10^{-4}(1 - \text{TB}_{d5_2}) + 10^{-10}$
8	$\hat{f}_{d9_1} = 0$	$10^{-4}(1 - \text{TB}_{d9_2}) + 10^{-10}$
9	$\hat{f}_{d9_2} = 0$	$10^{-4}(1 - \text{TB}_{d9_2}) + 10^{-10}$
10	$\hat{f}_{d10_1} = 0$	$10^{-4}(1 - \text{TB}_{d10_2}) + 10^{-10}$
11	$\hat{f}_{d10_2} = 0$	$10^{-4}(1 - \text{TB}_{d10_2}) + 10^{-10}$
12	Equation (6.25)	1
13	Equation (6.26)	1
14	Equation (6.27)	1

Table 6.12 Constraints and Weighting Factors of Subtask "Demanded Contact Forces"

Additionally, the corresponding weighting factors are computed as defined in Table 6.13.

With these weighting factors we prevent control actions leading to sliding or an unwanted lifting up of the feet (cf. Example 6.3) since the demanded forces in the horizontal direction are complied with more

$d\omega 1_3$	10^{-4} (CR CL (1 − SR) (1 − SL) + 1)	—
$d\ddot{x}1_1$	2 10^{-5} (CR CL (1 − SR) (1 − SL) + 1)	—
$d\ddot{x}1_2$	10^{-5} (CR + CL + 1)	—
d_{fa_1}	*interp* { $(-100, 10^{-10})$, $(-50, 10^{-10})$, $(0, 10^{-4})$, $(50, 10^{-4})$ } (x) $x = \lvert\hat{f}_{da_1}\rvert - \mu_0 \lvert\hat{f}_{da_2}\rvert$	a ∈ {4, 5, 9, 10 }
d_{fa_2}	*interp* { $(0, 10^{-4})$, $(1, 10^{-4})$, $(100, 10^{-8})$, $(200, 10^{-10})$, $(300, 10^{-10})$ } (f_{da_2})	

Table 6.13 Weighting Factors

strongly as their values tend towards their upper limit, whereas the demanded vertical forces are kept more accurately as they reach their lower boundaries.

The last subtask of the coordination level, designated with *Demanded Accelerations*, computes the demanded values of the accelerations and the corresponding weighting factors. Their definitions are given in Tables 6.14 and 6.15, respectively. The demanded accelerations are again chosen to attain critical damping. Notice that the accelerations of both knees, i.e., $\dot{\theta}'_{d22_3}$ and $\dot{\theta}'_{d72_3}$, are each defined by two control laws which are activated and deactivated by the fuzzy variables SR and SL, respectively.

These demanded contact forces and the demanded accelerations, together with the corresponding weighting factors, defining their momentary importance, represent the decision of the coordination level regarding the behavior of the system in the actual situation to stably move with the desired horizontal velocity.

$\dot{\omega}_{d\alpha}$	$-\omega_{\alpha_3}$	$\alpha \in \{2, 3, 5, 6, 8, 9, 10, 11\}$
$\dot{\omega}_{d\alpha_3}$	$(0 - \varphi_{\alpha_3}) - \omega_{\alpha_3}$	$\alpha \in \{4, 7\}$
$\ddot{X}_{d\alpha_2}$	$0.3\ \ddot{X}_{d\alpha+2_2}$	$\alpha \in \{2, 5\}$
$\ddot{X}_{d\alpha_1}$	$limit_{[-50,50]}\,(20\,(X_{d\alpha_1} - X_{\alpha_1}) + 2\sqrt{20}\,(0 - \dot{X}_{\alpha_1}))$	$\alpha \in \{4, 7\}$
$\ddot{X}_{d\alpha_2}$	$200\,(X_{d\alpha_2} - X_{\alpha_2}) + 2\sqrt{200}\,(0 - \dot{X}_{\alpha_2})$	$\alpha \in \{4, 7\}$
$\dot{\theta}'_{da2_3}$	$(0 - \varphi'_{a2_3}) + 2\,(0 - \theta'_{a2_3})$	$a \in \{1, 6, 11, 13\}$
$\dot{\theta}'_{d22_3}$	$(200\,(\frac{\pi}{18} - \varphi'_{22_3}) + 2\sqrt{200}\,(0 - \theta'_{22_3}))\,SR\ +$ $(50\,(\frac{\pi}{18} - \varphi'_{22_3}) + 2\sqrt{50}\,(0 - \theta'_{22_3}))\,(1 - SR)$	—
$\dot{\theta}'_{da2_3}$	$50\,(-\frac{\pi}{36} - \varphi'_{a2_3}) + 2\sqrt{50}\,(0 - \theta'_{a2_3})$	$a \in \{3, 8\}$
$\dot{\theta}'_{d72_3}$	$(200\,(\frac{\pi}{18} - \varphi'_{72_3}) + 2\sqrt{200}\,(0 - \theta'_{72_3}))\,SL\ +$ $(50\,(\frac{\pi}{18} - \varphi'_{72_3}) + 2\sqrt{50}\,(0 - \theta'_{72_3}))\,(1 - SL)$	—
$\dot{\theta}'_{da2_3}$	$(\frac{\pi}{18} - \varphi'_{a2_3}) + 2\,(0 - \theta'_{a2_3})$	$a \in \{12, 14\}$

Table 6.14 Demanded Accelerations

Execution Level

The execution level has exactly the same structure as that described in the first benchmark (cf. Figure 6.2). The demanded values of vector $\underline{z}_{dyn}$ (cf. (5.29)) of the subtask *Dynamical Constraints* are set to zero if they are not specified by the subtasks demanded forces, demanded accelerations, or boundary test. The corresponding weighting factors can be found in Table G.3 of Appendix G.

The main differences as compared to the execution level of the first benchmark can be found in the subtask *Boundary Test*. This is due to the fact that we additionally have boundaries possessing three different states. The state of the boundary constraints is characterized by the flags RB_a, TB_{a_1}, and TB_{a_2}, $RB_a = 1$ indicating that the rotational joint "a" is in

$d\ddot{x}2_1$	$d\ddot{x}5_1 X$ $X = interp\{(-0.2, 10^{-10}), (0, 10^{-10}),$ $(0.02, 1), (0.07, 2), (0.1, 2)\}(x)$ $x = \text{sgn}(\dot{X}1_1)(X1_1 - X2_1)$	—
$d\ddot{x}4_1$	$10^{-5}(1 - limit_{[0,1]}(c4_2 + c5_2))$	—
$d\ddot{x}4_2$	$10^{-4}(1 - limit_{[0,1]}(c4_2 + c5_2))$	—
$d\ddot{x}5_1$	$d\ddot{x}7_1 X$ $X = interp\{(-0.2, 10^{-10}), (0, 10^{-10}),$ $(0.02, 1), (0.07, 2), (0.1, 2)\}(x)$ $x = \text{sgn}(\dot{X}1_1)(X1_1 - X5_1)$	—
$d\ddot{x}7_1$	$10^{-5}(1 - limit_{[0,1]}(c9_2 + c10_2))$	—
$d\ddot{x}7_2$	$10^{-4}(1 - limit_{[0,1]}(c9_2 + c10_2))$	—
$d\dot{\theta}'a2_3$	$\text{SR}\ \text{X}_a + 10^{-6}(1 - \text{SR})$ $\text{X}_a = interp\{(0, 10^{-10}), (5, 10^{-10}),$ $(10, 10^{-6}), (15, 10^{-8}),$ $(20, 10^{-4}), (50, 10^{-4})\}(\dot{\theta}'da2_3)$	$a \in \{2, 7\}$
$d\dot{\theta}'a2_3$	$10^{-5}\text{SR} + 10^{-10}(1 - \text{SR})$	$a \in \{3, 8\}$
$d\ddot{s}'a2_3$	$interp\{(0, 0.1), (2, 0.1), (3, 1),$ $(400, 1), (600, 100), (1000, 100)\}(f_{da_2})$	$a \in \{4, 5, 9, 10\}$

Table 6.15 Weighting Factors

contact with either the upper or the lower boundary. TB_{a_2} specifies whether $s'a2_2$ has reached the positional boundary, while $TB_{a_1} = 1$ symbolizes static contact, i.e., $\dot{s}'a2_1 = 0$. The boundary flags are defined in Table 6.16:

(6.34)

	Value	Conditions
RB_a	1	$RB_a = 0$ and $((\delta b'_{\min a2_3} > 0$ and $\delta\dot{\theta}'_{a2_3} > 0)$ or $\tau'_{a2_3} > 0)$
	1	$RB_a = 0$ and $((\delta b'_{\max a2_3} < 0$ and $\delta\dot{\theta}'_{a2_3} < 0)$ or $\tau'_{a2_3} < 0)$
	0	$RB_a = 1$ and $(\delta b'_{\min a2_3} \le 0$ or $\delta\dot{\theta}'_{a2_3} \le -\varepsilon$ or $\tau'_{a2_3} < 0)$
	0	$RB_a = 1$ and $(\delta b'_{\max a2_3} \ge 0$ or $\delta\dot{\theta}'_{a2_3} \ge \varepsilon$ or $\tau'_{a2_3} > 0)$
	RB_a	else
TB_{a_1}	1	$\lvert f'_{a_1}\rvert - \mu_0 \lvert f'_{a_2}\rvert \le 0$ and $\lvert \dot{s}'_{a2_1}\rvert < 0.001$
	0	else
TB_{a_2}	1	$TB_{a_2} = 0$ and $((\delta s'_{\min a2_2} > 0$ and $\delta\ddot{s}'_{a2_2} > 0)$ or $f'_{a2_2} > 0)$
	0	$TB_{a_2} = 1$ and $(\delta s'_{\min a2_2} \le 0$ or $\delta\ddot{s}'_{a2_2} \le -\varepsilon$ or $f'_{a2_2} < 0)$
	TB_{a_2}	else

Table 6.16 Boundary Flags

for

$$\begin{aligned} \delta b'_{\min a2_3} &= b'_{\min a2_3} - b'_{a2_3}, \\ \delta b'_{\max a2_3} &= b'_{\max a2_3} - b'_{a2_3}, \\ \delta s'_{\min a2_2} &= s'_{\min a2_2} - s'_{a2_2}, \end{aligned} \tag{6.35}$$

and

$$\begin{aligned} \delta\dot{\theta}'_{a2_3} &= \dot{\theta}'_{d\,a2_3} - \dot{\theta}'_{a2_3} = 0 - \dot{\theta}'_{a2_3}, \\ \delta\ddot{s}'_{a2_1} &= 0, \\ \delta\ddot{s}'_{a2_2} &= \ddot{s}'_{d\,a2_2} - \ddot{s}'_{a2_2} = 0 - \ddot{s}'_{a2_2}. \end{aligned} \tag{6.36}$$

They are valid for $a \in \{1, 2, 3, 6, 7, 8, 11, 12, 13, 14\}$ for rotational boundaries and for $a \in \{4, 5, 9, 10\}$ for translational boundaries.

Remark 6.2:

Notice that the boundary constraint is dissolved if the contact force becomes negative or if $\delta\ddot{s}'_{a2_2} \leq \varepsilon$. This second condition makes it possible that the boundary constraint is deactivated even if the contact force is positive. This is what happens with the constraint for the heel of the right foot in the situation described in Example 6.2.

6.2.3 Simulation Results

The controller was tuned for a demanded horizontal velocity of 0.5 m/s. The resulting behavior of the plant is shown in Figure 6.28. Figure 6.29 depicts the movement of the robot during the third step with a resolution of 0.02 seconds per image. Finally, Figure 6.30 shows the horizontal velocity of the upper body versus time. We can see that after the second step the system is in a steady state.

To test the robustness of the controller, the demanded velocity is increased and decreased. For a demanded velocity of 0.1 m/s the controller is still able to stabilize the plant (cf. Figure 6.31). Smaller demanded horizontal velocities were not tested as the simulation times became excessively long. The simulation results for a demanded horizontal velocity of 1.0 m/s are given in Figure 6.32. Again, the controller succeeds in stabilizing the plant. However, for a demanded horizontal velocity of 1.2 m/s and more this is no longer true.

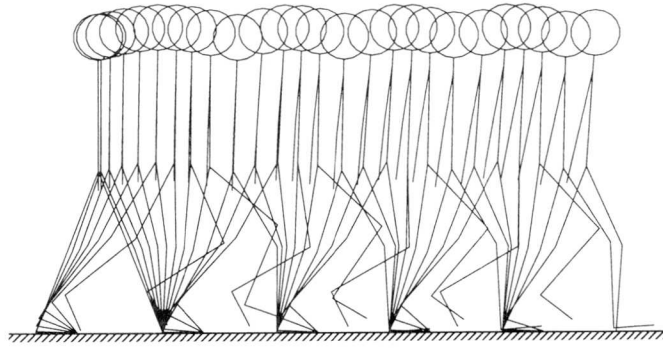


Figure 6.28 Walking with 0.5 m/s Horizontal Velocity

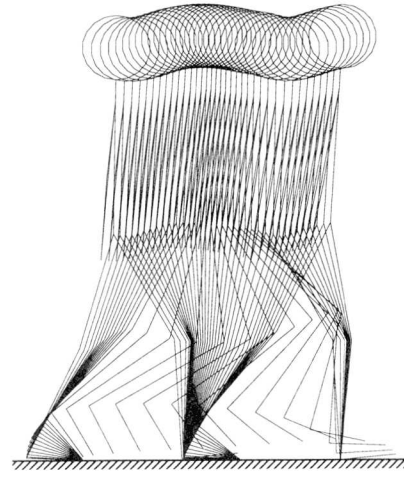


Figure 6.29 Third Step

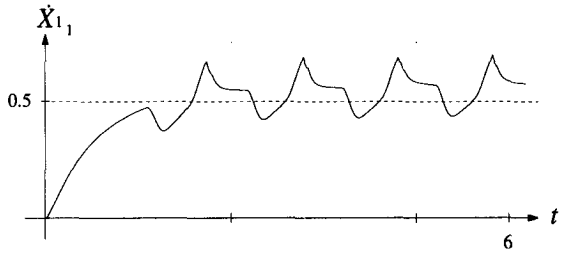


Figure 6.30 Horizontal Velocity of the Upper Body

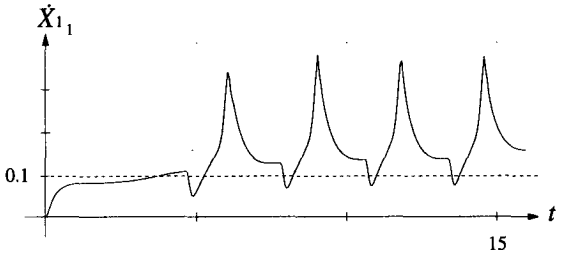


Figure 6.31 Horizontal Velocity of the Upper Body

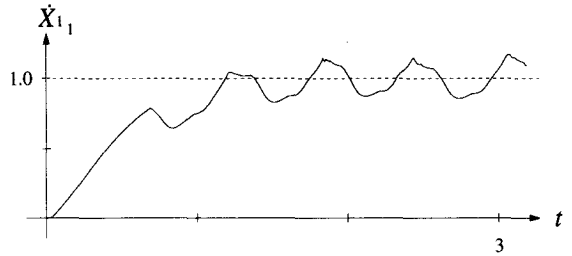


Figure 6.32 Horizontal Velocity of the Upper Body

6.3 Numerical Effort

This section briefly discusses the numerical effort to compute the control vector for the proposed controllers. According to our experience the subtasks *Positional Constraints*, *Velocity Constraints*, and *Dynamics* are the numerically most expensive tasks.

Tables 6.17 and 6.18 give the number of floating point operations to calculate the solution of the subtasks for the first and second benchmark, respectively, with one processor. The computational costs are evaluated for the solution of the minimization problem with Gaussian elimination, with dyadic reduction, and with optimized dyadic reduction. The backward substitution needed in all three cases is taken into account as well.

Utilizing the dyadic reduction approach rather than applying Gaussian elimination we were able to realize savings in the computational costs of between 50% (positional constraints and velocity constraints) and 80% (dynamical constraints) in the first benchmark and around 80% for all three subtasks in the second benchmark. This reduction of floating point operations is mainly due to the avoidance of zero operations which, using dyadic reduction, can be determined off-line. By optimizing the dyadic reduction algorithm the computational costs were decreased by another 20% for the first benchmark and 50% in the second benchmark.

Mode	Performance Index	+	-	*	/
Gauss	J_{pos}	512		512	
	J_{vel}	512		512	
	J_{dyn}	3375		3375	
normal	J_{pos}	99	65	216	52
	J_{vel}	88	66	198	44
	J_{dyn}	270	205	605	130
optimized	J_{pos}	99	65	216	52
	J_{vel}	88	66	198	44
	J_{dyn}	225	171	504	108

Table 6.17 Numerical Costs in the First Benchmark

Mode	Performance Index	+	-	*	/
Gauss	J_{pos}	166 375		166 375	
	J_{vel}	166 375		166 375	
	J_{dyn}	1 225 043		1 225 043	
normal	J_{pos}	17 769	16 519	36 678	2 390
	J_{vel}	16 648	15 601	34 343	2 094
	J_{dyn}	112 623	108 489	229 380	8 268
optimized	J_{pos}	9 545	8 702	19 823	1 576
	J_{vel}	6 451	5 929	13 424	1 044
	J_{dyn}	52 567	50 452	107 249	4 230

Table 6.18 Numerical Costs in the Second Benchmark

Chapter 7

Summary and Conclusions

The proposed method was successfully applied to the modelling and control of rather complex nonlinear systems.

The hierarchical structure proved to be very valuable as it permitted small step increases in the controller's complexity until the given problems could be solved with the desired accuracy and minimal effort.

The use of performance indices to specify the subtasks and the numerical minimization of these indices proved to be an efficient approach. This kind of subtask definition has several advantages. There are solutions for the determined case as well as for the over- and underdetermined cases. Thus, there is no need to pay special attention to redundancies of any kind and singularities as they do not have to be treated differently. Also, there is no necessity to completely specify the demanded trajectory. Therefore, the controller is not superfluously deprived of the degrees of freedom required for reacting to unforeseen events. It is also possible to on-line "deactivate" specific constraints by setting the appropriate weighting factors to zero. This allows to completely change the structure of a subtask or of the whole control task, even though the hierarchical structure of the controller is rigid. This approach thus lends itself to being used to model the dynamics of systems with changing structures in a rather straightforward manner.

The expected high costs to compute the control vector and, therefore, the sampling interval were found to be reducible in several ways. First, through the avoidance of costly trigonometric computations due to the use of quaternions in the kinematical and dynamical equations. Second, by formulating the constraint equations with redundant coordinates (joint coordinates and Cartesian coordinates) and by using recursive definitions of subexpressions wherever possible. Finally, by optimizing the algorithm of dyadic reduction. This optimization is done off-line. In the case of the walking robot described in Chapter 6, for example, this off-line optimization reduced the computational costs by about 50%. Compared with Gaussian elimination, which also could be used to solve the minimization problem, the reduction of the computational costs is even greater (more than 90%).

This off-line optimization makes it possible to determine an upper limit to the calculation time allotted to find the solution of a subtask. This feature is rather important when it comes to deciding upon the hardware needs for guaranteeing the necessary sampling rates. Additionally, it allows to work with sparse matrices, as the whole algorithm is fully predetermined after the off-line optimization, i.e., the position of every nonzero element is known in advance.

The evaluation of the constraint equations for the kinematics and dynamics of arbitrary, rigid multi-body systems and the off-line optimization of the dyadic reduction algorithm for an arbitrary number of processors have been computerized using a symbol manipulation program [ChGeGo88]. This symbol manipulation program proved to be a very valuable and proficient tool for this purpose.

Among the ways to take full advantage of the proposed method would be the asynchronous and parallel computation of the solutions to the subtasks. The asynchronous processing would permit the differentiation between the simple and fast subtasks, on the one hand, and the more comprehensive subtasks, which are less time critical on the other hand. In connection with parallel processing considerable decreases of the sampling interval are yet

possible to attain. Thus, the implementation of the proposed controller on a multiprocessor system is feasible. The parallelization of the algorithm of dyadic reduction is straightforward. The major problem to be resolved is the proper organization of the data flow and an efficient coordination of the different computation steps that can be executed simultaneously.

Appendix

A Symbol Index

Symbol	*Significance*	*Page Introduced*
α, β	integers specifying body reference frames	39
$\Delta\vartheta_\alpha = \Delta t\, \omega_\alpha$	angle (body α)	49
θ'_{aj}	rotational velocity vector of unitary spinor $\dot{S}'_{aj}$	44
θ'_{aj_k}	component of vector θ'_{aj} in direction $\hat{\mathbf{b}}'_{aj_k}$	44
μ_0	coefficient of friction	139
ν_i	number of non-zero elements of row i of matrix B	83
ν_{ij}	integer specifying the structure of matrix B	83
$\tilde{\nu}_i$	number of non-zero elements of row i of matrix $\tilde{\mathsf{B}}$	83
$\tilde{\nu}_{ij}$	integer specifying the structure of matrix $\tilde{\mathsf{B}}$	83
$\Delta\nu_i$	number of new non-zero elements in row i of $\tilde{\mathsf{B}}$	86
$\Delta\nu_{ij}$	new non-zero elements in matrix $\tilde{\mathsf{B}}$	83
$\underline{\tau}$	matrix representation of hinge torques	109
τ_a	hinge torque vector (hinge a)	54
τ_{a_k}	component of τ_a in direction $\mathbf{e}_k$)	54
τ'_{aj_k}	passive link torque (hinge a, link j, direction $\hat{\mathbf{b}}_{aj_k}$)	46
$\underline{\tau}'_P$	matrix representation of passive hinge torques	109
φ	angle	13
$\Delta\varphi'_{aj}$	angle (hinge a, link j)	50
φ'_{aj_k}	rotation angle (hinge a, link j, rotation axis $\hat{\mathbf{b}}'_{aj_k}$)	51

Symbol	*Significance*	*Page Introduced*
$\Delta\varphi'_{aj_k}=\Delta t\ \theta'_{aj_k}$	angle (hinge a, link j, rotation axis $\hat{\mathbf{b}}'_{aj_k}$)	50
$\varphi'_{\min aj_k}$	lower bound of variable φ'_{aj_k}	51
$\varphi'_{\max aj_k}$	upper bound of variable φ'_{aj_k}	51
ω	rotational velocity vector of unitary spinor R	23
${}^{i}\omega$	rotational velocity vector of unitary spinor ${}^{i}R$	28
a	integer specifying hinges	39
a	real number, scalar part of R, or magnitude of **a**	8
a_α	scalar part of unitary spinor R_α	48
a_k	k*th* element of vector $\underline{\mathbf{a}}$	30
a'_{aj}	scalar part of unitary spinor S'_{aj}	41
a	vector in the unprimed reference frame $\{\mathbf{e}_k\}$	20
$\hat{\mathbf{a}}$	unit vector of vector **a**	13
${}^{i}\mathbf{a}$	resultant vector of vectors $\mathbf{a}'_i$ to $\mathbf{a}'_n$	27
$\dot{\mathbf{a}}$	first time derivative of vector **a**	22
$\mathbf{a}'$	vector in the primed reference frame $\{\mathbf{e}'_k\}$	20
$\underline{\mathbf{a}}$	matrix representation of vector **a**	30
A	multivector	8
$\langle A\rangle_r$, $\mathbf{A}_r$	r-blade or r-vector part of multivector A	8
$A^\dagger$	reverse of multivector A	10
$\lvert A\rvert$	magnitude of multivector A	12
A^{-1}	multiplicative inverse of multivector$A$	12
AB	geometric product of multivectors A and B	11
$A\cdot B$	inner product of multivectors AB	10
$A\wedge B$	outer or wedge product of multivectors AB	11
A	matrix	68

Symbol	*Significance*	*Page Introduced*
A^{+}	Moore-Penrose-Inverse of matrix A	69
$\underline{\mathsf{A}}^{*}$	sparse representation of matrix A	89
A_{pos}	matrix (positional constraints)	104
A_{vel}	matrix (velocity constraints)	106
A_{dyn}	matrix (dynamical constraints)	110
b_{ij}	*j*th element of row i of matrix B	79
$\tilde{b}_{ij}$	*j*th element of row i of matrix $\tilde{\mathsf{B}}$	80
$b_{1u_{i_r p}}$	*p*th element of row i_r of matrix B_{1u}	46
$b_{2u_{i_t p}}$	*p*th element of row i_t of matrix B_{2u}	46
b'_{aj_k}	component of $\mathbf{b}'_{aj}$ in direction $\hat{\mathbf{b}}'_{aj_k}$	41
$\mathbf{b}$	vector, cf. R	20
$\mathbf{b}_\alpha$	vector, cf. $R\alpha$	48
$\hat{\mathbf{b}}_{aj_k}$	vector $\hat{\mathbf{b}}'_{aj_k}$ in the unprimed reference frame	49
$\mathbf{b}'_{aj}$	vector, cf. S'_{aj}	41
$\hat{\mathbf{b}}'_{aj_k}$	unit vector of the k*th* component of $\mathbf{b}'_{aj}$	41
$\underline{\hat{\mathsf{b}}}_{aj_k}$	matrix representation of $\hat{\mathbf{b}}_{aj_k}$	103
$\underline{\mathsf{b}}(\underline{y})$	vector (implicit constraints)	67
$\underline{\mathsf{b}}_i$	*i*th column of matrix B	79
$\underline{\mathsf{b}}_{pos}$	vector (positional constraints)	104
$\underline{\mathsf{b}}_{dyn}$	vector (dynamical constraints)	110
$\mathsf{B}(\underline{y})$	matrix (performance index)	70
$\tilde{\mathsf{B}}$	matrix B after one reduction step (dyadic reduction)	78
B_0	matrix	73
B_1	matrix	73
$\mathsf{B}_{1u} = [b_{1u_{i_r p}}]$	matrix	109

Symbol	*Significance*	*Page Introduced*
$\mathbf{B}_{2u} = [b_{2u_{i,p}}]$	matrix	109
$\mathbf{B}_{pos}$	matrix (positional constraints)	104
$\mathbf{B}_{vel}$	matrix (velocity constraints)	106
$\mathbf{B}_{dyn}$	matrix (dynamical constraints)	110
$\underset{\sim}{\tilde{\mathbf{B}}}^{*}$	sparse representation of matrix $\tilde{\mathbf{B}}$	89
$\underset{\sim}{\tilde{\mathbf{B}}}_0^{*}$	sparse representation of matrix $\mathbf{B}_0$	90
$\underset{\sim}{\tilde{\mathbf{B}}}_1^{*}$	sparse representation of matrix $\mathbf{B}_1$	90
BP	variable "Balance Phase"	144
$c_{\alpha a}$	connectivity coefficient	40
$c_{\alpha a}^{+}$	connectivity coefficient	40
$c_{\alpha a}^{-}$	connectivity coefficient	40
c_{a_2}	binary sensor signal	141
$\mathbf{C}$	connectivity matrix	106
$\mathbf{C}_{1S}$	submatrix of matrix $\mathbf{A}_{pos}$	104
$\mathbf{C}_1$	submatrix of matrix $\mathbf{C}$	105
$\mathbf{C}_2$	submatrix of matrix $\mathbf{C}$	103
$\mathbf{C}_3$	submatrix of matrix $\mathbf{C}$	103
CL	variable "Contact" with the "Left" foot	143
CR	variable "Contact" with the "Right" foot	143
d_i	*ith* diagonal element of matrix $\mathbf{D}$	79
$\underset{\sim}{\mathbf{d}}_{0pos}$	vector of diagonal elements of matrix $\mathbf{D}_{0\,pos}$	120
$\underset{\sim}{\mathbf{d}}_{0vel}$	vector of diagonal elements of matrix $\mathbf{D}_{0\,vel}$	120
$\underset{\sim}{\mathbf{d}}_{0dyn}$	vector of diagonal elements of matrix $\mathbf{D}_{0\,dyn}$	122
$\mathbf{D}$	diagonal matrix, weight matrix	70
$\tilde{\mathbf{D}}$	matrix $\mathbf{D}$ after one reduction step (dyadic reduction)	78

Symbol	*Significance*	*Page Introduced*
D_0	diagonal matrix	73
D_1	diagonal matrix	73
D_{pos}	diagonal matrix (positional constraints)	104
D_{vel}	diagonal matrix (velocity constraints)	106
D_{dyn}	diagonal matrix (dynamical constraints)	110
$\mathsf{D}_{0\,pos}$	submatrix of matrix D_{pos}	104
$\mathsf{D}_{0\,vel}$	submatrix of matrix D_{vel}	106
$\mathsf{D}_{0\,dyn}$	submatrix of matrix D_{dyn}	110
D.O.F.	degrees of freedom	xx
$\{\mathbf{e}_k\}$	orthonormal reference frame, unprimed frame	16
$\{\mathbf{e}'_k\}$	orthonormal reference frame, primed frame	20
$\{\mathbf{e}'_{\alpha_k}\}$	orthonormal frame rigidly attached to body α	47
$\mathcal{E}_3$	3-dimensional Euclidean space	15
E	unit matrix	73
EL	variable "End" step with the "Left" foot	143
ER	variable "End" step with the "Right" foot	143
f	linear function, linear operator, linear transformation	18
$\bar{f}$	transpose or adjoint of f	19
f_{jk}	element of matrix representation of linear function f	31
f'_{aj_k}	passive link force (hinge a, link j, direction $\hat{\mathbf{s}}_{aj_k}$)	46
$\mathbf{f}_a$	hinge force (hinge a)	54
$\underline{f}$	matrix representation of hinge forces	109
$\underline{f}'_P$	matrix representation of passive hinge forces	109
$\mathbf{F}_\alpha$	resultant force exerted on body α	59
$\mathbf{F}_{g\alpha}$	gravitational force on body α	59

Symbol	*Significance*	*Page Introduced*
${}^{i}\mathbf{g}$	vector	29
${}^{1}\mathbf{g}_{a}$	vector	54
$\underline{\mathbf{g}}$	vector	109
$\mathcal{G}$	geometric algebra	8
$\mathcal{G}_3$	geometric algebra of 3-dimensional Euclidean space	15
${}^{i}\mathbf{h}$	vector	29
${}^{1}\mathbf{h}_{a}$	vector	54
$\underline{\mathbf{h}}$	vector	109
i	dextral unit pseudoscalar	16
i_{A_i}	index (component of J_A)	89
i_{D_i}	index (component of J_D)	91
i_{P_i}	index (componentJ_P)	90
i_{r_i}	index (component J_r)	90
i_{rp_i}	index (component J_{rp})	90
i_{r}	index, cf. $b_{1u_{ir\,p}}$	46
i_{t}	index, cf. $b_{2u_{it\,p}}$	46
I_{α_k}	principal value of $\mathcal{J}_\alpha$ (body α, direction $\mathbf{e}'_{\alpha_k}$)	57
J_A	index vector	89
J_{B0}	index vector	94
J_D	index vector	91
J_P	index vectors	90
J_r	index vector	90
J_{rp}	index vector	90
interp	operator of linear interpolation	141
IS	variable "Initialize Step"	143

Symbol	*Significance*	*Page Introduced*
j	integer, index specifying the link of a hinge	41
j_{A_k}	index (component of $\underline{J}_A$)	89
j_{Dp_j}	index (component of $\underline{J}_{Dp}$)	91
j_{Dr_j}	index (component of $\underline{J}_{Dr}$)	91
j_{Dn_j}	index (component of $\underline{J}_{Dn}$)	91
J	jacobian matrix	106
J_{1S}	submatrix of matrix A_{pos}	104
J_1	submatrix of matrix J	105
J_2	submatrix of matrix J	103
J_3	submatrix of matrix J	103
$\underline{J}_A$	index vector	89
$\underline{J}_{B0}$	index vector	94
$\underline{J}_{Dp}$	index vector	91
$\underline{J}_{Dr}$	index vector	91
$\underline{J}_{Dn}$	index vector	91
$\mathcal{J}_\alpha$	inertia tensor of body α	56
$J(\underline{z}, \underline{y})$	performance index (node)	70
$J(p, r)$	performance index (dyadic reduction)	87
k	integer, index (components of vectors)	42
l	integer, index (actuators)	46
$\underline{l}$	angular momentum	59
$\underline{l}_\alpha$	angular momentum of body α	59
limit	clipping operator	141
m	number of implicit constraints	68
m_α	mass of body α	56

Symbol	*Significance*	*Page Introduced*
M	mass matrix	110
M_{l}	submatrix of matrix M	109
M_{m}	submatrix of matrix M	109
n	number of hinges	39
n_{a}	number of links in hinge a	41
$\mathrm{n}_{\mathrm{raj}}$	number of rotational D.O.F. in link j of hinge a	41
$\mathrm{n}_{\mathrm{taj}}$	number of translational D.O.F. in link j of hinge a	41
N	number of rigid bodies	39
p_{ij}	parameters	107
(p, r)	dyadic reduction step	80
$\mathbf{p}$	position vector	24
$\{\mathbf{p}\}$	position space	24
$\mathbf{P}$	momentum	59
$\mathbf{P}_{\alpha}$	momentum of body α	59
$q_{\mathrm{i}_{\mathrm{jk}}}$	matrix element of matrix Q_{i}	79
Q	matrix	73
Q_{i}	matrix	79
r_{jk}	matrix element of matrix R	32
$\mathbf{r}_{\alpha}$	position vector of dm_{α} with respect to $\{\mathbf{e}'_{\alpha_{\mathrm{k}}}\}$	57
$\mathbf{r}_{\alpha\mathrm{a}}$	point of application of hinge force $\mathbf{f}_{\mathrm{a}}$ on body α	59
$\mathcal{R}$	rotation operator of unitary spinor R	19
$R = a + i\mathbf{b}$	unitary spinor or quaternion	20
$R_{\alpha} = a_{\alpha} + i\mathbf{b}_{\alpha}$	quaternion (orientation of $\{\mathbf{e}'_{\alpha_{\mathrm{k}}}\}$ relative to $\{\mathbf{e}_{\mathrm{k}}\}$	47
${}^{\mathrm{i}}R$	product of spinors R'_{i} to R'_{n}	27
$\mathsf{R} = [r_{\mathrm{jk}}]$	matrix representation of rotation operator $\mathcal{R}$	32

Symbol	*Significance*	*Page Introduced*
$\underline{R}$	matrix representation of unitary spinor R	31
$\overline{\underline{R}}$	submatrix of matrix $\underline{R}$	32
$\{R \mid \mathbf{a}\}$	rigid displacement	24
$\{R\alpha \mid \mathbf{X}\alpha\}$	rigid displacement	47
RB_{a}	"Rotational Boundary" flag	152
RL	variable "Right" or "Left" foot	143
$s'_{\mathrm{aj_k}}$	component of vector $\mathbf{s}'_{\mathrm{aj}}$ in direction $\hat{\mathbf{s}}'_{\mathrm{aj_k}}$	41
$\Delta s'_{\mathrm{aj_k}} = \Delta t\, \dot{s}'_{\mathrm{aj_k}}$	translation (hinge a, link j, translation axis $\hat{\mathbf{s}}'_{\mathrm{aj_k}}$)	50
$s'_{\mathrm{min\,aj_k}}$	lower bound of variable $s'_{\mathrm{aj_k}}$	51
$s'_{\mathrm{max\,aj_k}}$	upper bound of variable $s'_{\mathrm{aj_k}}$	51
$\delta s'_{\mathrm{min\,aj_k}}$	distance of variable $s'_{\mathrm{aj_k}}$ from the lower boundary	124
$\delta s'_{\mathrm{max\,aj_k}}$	distance of variable $s'_{\mathrm{aj_k}}$ from the upper boundary	124
$\delta \ddot{s}'_{\mathrm{aj_k}}$	acceleration error of variable $s'_{\mathrm{aj_k}}$	124
$\mathbf{s}_{\mathrm{a}}$	vector	47
${}^{\mathrm{i}}\mathbf{s}_{\mathrm{a}}$	vector	48
$\mathbf{s}'_{\mathrm{a}}$	translation vector (hinge a)	41
$\mathbf{s}'_{\mathrm{aj}}$	translation vector (hinge a, link j)	41
$\hat{\mathbf{s}}'_{\mathrm{aj_k}}$	unit vector of the k*th* component of vector $\mathbf{s}'_{\mathrm{aj}}$	41
$\hat{\mathbf{s}}_{\mathrm{aj_k}}$	vector $\hat{\mathbf{s}}'_{\mathrm{aj_k}}$ in the unprimed frame	50
${}^{\mathrm{j}}\underline{\mathbf{s}}_{\mathrm{a}}$	matrix representation of vector ${}^{\mathrm{i}}\mathbf{s}_{\mathrm{a}}$	102
$\underline{\mathbf{s}}'_{\mathrm{aj}}$	matrix representation of vector $\mathbf{s}'_{\mathrm{a}}$	102
$\hat{\underline{\mathbf{s}}}_{\mathrm{aj_k}}$	matrix representation of vector $\hat{\mathbf{s}}_{\mathrm{aj_k}}$	103
S_{a}	unitary spinor	48
${}^{\mathrm{i}}S_{\mathrm{a}}$	unitary spinor	48
$S'_{\mathrm{a}} = a'_{\mathrm{a}} + i\mathbf{b}'_{\mathrm{a}}$	unitary spinor (hinge a)	41

Symbol	*Significance*	*Page Introduced*
$S'_{aj} = a'_{aj} + i\mathbf{b}'_{aj}$	unitary spinor (hinge a, link j)	41
$\{S'_a \mid \mathbf{s}'_a\}$	rigid displacement (hinge a)	41
$\{S'_{aj} \mid \mathbf{s}'_{aj}\}$	rigid displacement (hinge a, link j)	41
${}^{i}\mathcal{S}_a$	rotation operator of unitary spinor ${}^{i}S_a$	48
${}^{j}\mathbf{S}_a$	unitary spinor (hinge a)	102
${}^{j}\underline{\mathbf{S}}_a$	matrix representation of unitary spinor ${}^{i}S_a$	102
$\overline{\mathbf{S}}'_{aj}$	matrix, cf. matrix $\overline{\mathbf{R}}$	102
SL_d	variable demand for a "Step" with the "Left" foot	144
SR_d	variable demand for a "Step" with the "Right" foot	144
SL	variable "Step" with the "Left" foot	144
SR	variable "Step" with the "Right" foot	144
ST	variable "Stumbling"	143
t	scalar variable, generally used as time variable	20
TB_{da_2}	demanded value for TB_{a_2}	145
TB_{a_k}	"Translational Boundary" flag	152
u_p	magnitude of vector $\mathbf{u}_p$	46
$\mathbf{u}_p$	*pth* control force or torque	45
$\underline{\mathbf{u}}$	control vector	109
$\mathbf{U}$	monic, upper triangular matrix	78
$\mathbf{x}_\alpha$	position vector of dm_α with respect to $\{\mathbf{e}_k\}$	57
$\mathbf{X}_\alpha$	position of $\{\mathbf{e}'_{\alpha_k}\}$ relative to $\{\mathbf{e}_k\}$	47
$\underline{\mathbf{X}}_\alpha$	matrix representation of vector $\mathbf{X}_\alpha$	102
$\underline{\mathbf{y}}$	node input vector	66
$\underline{\tilde{\mathbf{y}}}$	node output vector	66
$\mathbf{z}$	vector of internal variables	66

Symbol	*Significance*	*Page Introduced*
$\mathbf{z}_{Feet}$	vector	120
$\mathbf{z}_{Foces}$	vector	120
$\mathbf{z}_{pos}$	vector	104
$\mathbf{z}_{vel}$	vector	106
$\mathbf{z}_{dyn}$	vector	110
$\mathbf{z}_d$	demanded values for vector $\mathbf{z}$	73

B Geometric Algebra

B1: $\langle A^\dagger \rangle_r = (-1)^{(r(r-1))/2} \mathbf{A}_r$

- (2.1), (2.13), (2.14),(2.15), and (2.18):

$$\begin{aligned} \langle A^\dagger \rangle_r &= \langle (\sum_s \mathbf{A}_s)^\dagger \rangle_r = \langle \sum_s \mathbf{A}_s \rangle_r^\dagger = \langle A \rangle_r^\dagger = \mathbf{A}_r^\dagger = (\mathbf{a}_1 \mathbf{a}_2 \ldots \mathbf{a}_r)^\dagger \\ &= (\mathbf{a}_r \ldots \mathbf{a}_2 \mathbf{a}_1) = -(\mathbf{a}_r \ldots \mathbf{a}_1 \mathbf{a}_2) = (-1)^{r-1} (\mathbf{a}_1 \mathbf{a}_r \ldots \mathbf{a}_2) \\ &= (-1)^{(r(r-1))/2} (\mathbf{a}_1 \mathbf{a}_2 \ldots \mathbf{a}_r) = (-1)^{(r(r-1))/2} \mathbf{A}_r \end{aligned} \tag{B1}$$

B2: $\langle A_r B_s \rangle_t = (-1)^{(r(r-1)+s(s-1)+t(t-1))/2} \langle B_s A_r \rangle_t$

- (2.15) and (B1):

$$\begin{aligned} \langle A_r B_s \rangle_t &= \langle B_s^\dagger A_r^\dagger \rangle_t^\dagger = (-1)^{(t(t-1))/2} \langle B_s^\dagger A_r^\dagger \rangle_t \\ &= (-1)^{(t(t-1))/2} (-1)^{(s(s-1))/2} (-1)^{(r(r-1))/2} \langle B_s A_r \rangle_t \\ &= (-1)^{(r(r-1)+s(s-1)+t(t-1))/2} \langle B_s A_r \rangle_t \end{aligned} \tag{B2}$$

B3: $\mathbf{A}_r^\dagger \mathbf{A}_r = \langle \mathbf{A}_r^\dagger \mathbf{A}_r \rangle_0$

- (2.15), (2.18), (2.4), (2.12), and (2.11):

$$\begin{aligned} \mathbf{A}_r^\dagger \mathbf{A}_r &= (\mathbf{a}_r \ldots \mathbf{a}_2 \mathbf{a}_1)(\mathbf{a}_1 \mathbf{a}_2 \ldots \mathbf{a}_r) = (\mathbf{a}_r \ldots \mathbf{a}_2)\, \mathbf{a}_1 \mathbf{a}_1 (\mathbf{a}_2 \ldots \mathbf{a}_r) \\ &= (\mathbf{a}_r \ldots \mathbf{a}_2) |\mathbf{a}_1|^2 (\mathbf{a}_2 \ldots \mathbf{a}_r) = |\mathbf{a}_1|^2 (\mathbf{a}_r \ldots \mathbf{a}_2)(\mathbf{a}_2 \ldots \mathbf{a}_r) \\ &= |\mathbf{a}_1|^2 |\mathbf{a}_2|^2 \ldots |\mathbf{a}_r|^2 \end{aligned} \tag{B3}$$

B4: $\mathbf{a}\mathbf{B}_r = \mathbf{a} \cdot \mathbf{B}_r + \mathbf{a} \wedge \mathbf{B}_r$

- (2.13), (2.23), and (2.24):

$$\mathbf{a}\mathbf{B}_r = \mathbf{a}(\mathbf{b}_1\mathbf{b}_2\ldots\mathbf{b}_r) = \langle\mathbf{a}\mathbf{B}_r\rangle_{r-1} + \langle\mathbf{a}\mathbf{B}_r\rangle_{r+1} \tag{B4.1}$$

- (2.23), (2.24), and (B2):

$$\begin{aligned}\mathbf{B}_r\mathbf{a} &= \langle\mathbf{B}_r\mathbf{a}\rangle_{r-1} + \langle\mathbf{B}_r\mathbf{a}\rangle_{r+1} = \langle\mathbf{a}^\dagger\mathbf{B}_r^\dagger\rangle_{r-1}^\dagger + \langle\mathbf{a}^\dagger\mathbf{B}_r^\dagger\rangle_{r+1}^\dagger \\ &= (-1)^{(r(r-1)+(r-1)(r-2))/2}\langle\mathbf{a}\mathbf{B}_r\rangle_{r-1} + \\ &\quad (-1)^{(r(r-1)+(r+1)r)/2}\langle\mathbf{a}\mathbf{B}_r\rangle_{r+1} \\ &= (-1)^{(r(r-1)-(r-1))}\langle\mathbf{a}\mathbf{B}_r\rangle_{r-1} + (-1)^{(r(r-1)+r)}\langle\mathbf{a}\mathbf{B}_r\rangle_{r+1} \\ &= (-1)^{(r-1)}\langle\mathbf{a}\mathbf{B}_r\rangle_{r-1} + (-1)^r\langle\mathbf{a}\mathbf{B}_r\rangle_{r+1}\end{aligned} \tag{B4.2}$$

- (B4.2):

$$(-1)^{-r}\mathbf{B}_r\mathbf{a} = -\langle\mathbf{a}\mathbf{B}_r\rangle_{r-1} + \langle\mathbf{a}\mathbf{B}_r\rangle_{r+1} = (-1)^r\mathbf{B}_r\mathbf{a} \tag{B4.3}$$

- (B4.1) - (B4.3) solved for $\langle\mathbf{a}\mathbf{B}_r\rangle_{r-1}$:

$$\langle\mathbf{a}\mathbf{B}_r\rangle_{r-1} = \tfrac{1}{2}(\mathbf{a}\mathbf{B}_r - (-1)^r\mathbf{B}_r\mathbf{a}) = \mathbf{a}\cdot\mathbf{B}_r \tag{B4.4}$$

- (B4.1) + (B4.3) solved for $\langle\mathbf{a}\mathbf{B}_r\rangle_{r+1}$:

$$\langle\mathbf{a}\mathbf{B}_r\rangle_{r+1} = \tfrac{1}{2}(\mathbf{a}\mathbf{B}_r + (-1)^r\mathbf{B}_r\mathbf{a}) = \mathbf{a}\wedge\mathbf{B}_r \tag{B4.5}$$

- (B4.1), (B4.4) and (B4.5):

$$\mathbf{a}\mathbf{B}_r = \langle\mathbf{a}\mathbf{B}_r\rangle_{r-1} + \langle\mathbf{a}\mathbf{B}_r\rangle_{r+1} = \mathbf{a}\cdot\mathbf{B}_r + \mathbf{a}\wedge\mathbf{B}_r \tag{B4}$$

C Algebra of Euclidean 3-Space

C1: $i^\dagger = -i$

- (2.38), (2.15), and (B1):

$$i^\dagger = (\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3)^\dagger = \mathbf{e}_3\mathbf{e}_2\mathbf{e}_1 = -(\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3) = -i \tag{C1}$$

C2: $i^2 = -1$

- (2.38), (B1), (2.4), and (2.37):

$$\begin{aligned} i^2 &= (\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3)\,(\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3) = -(\mathbf{e}_3\mathbf{e}_2\mathbf{e}_1)\,(\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3) \\ &= -\mathbf{e}_3\mathbf{e}_2\,(\mathbf{e}_1\mathbf{e}_1)\,\mathbf{e}_2\mathbf{e}_3 = -\mathbf{e}_3\,(\mathbf{e}_2\mathbf{e}_2)\,\mathbf{e}_3 = -\mathbf{e}_3\mathbf{e}_3 = -1 \end{aligned} \tag{C2}$$

C3: $\mathbf{x}i = i\mathbf{x}$

- (2.25), (2.39), (2.26), (2.15), and (B2):

$$\begin{aligned} \mathbf{x}i &= \mathbf{x}\cdot i + \mathbf{x}\wedge i = \mathbf{x}\cdot i = \langle \mathbf{x}i\rangle_2 \\ &= (-1)^{(3\cdot 2+1\cdot 0+2\cdot 1)/2}\langle i\mathbf{x}\rangle_2 = i\mathbf{x} \end{aligned} \tag{C3}$$

C4: $iA = Ai$

- (C3):

$$\begin{aligned} iA &= i\sum_{r=0}^{3}\mathbf{A}_r = i\,(a+\mathbf{b}+i\mathbf{c}+id) = i\,(a+\mathbf{b}+\mathbf{c}i+di) \\ &= ia+i\mathbf{b}+i\mathbf{c}i+idi = ai+\mathbf{b}i+i\mathbf{c}i+idi \\ &= (a+\mathbf{b}+i\mathbf{c}+id)\,i = Ai \end{aligned} \tag{C4}$$

C5: $\mathbf{a}\wedge\mathbf{b}\wedge\mathbf{c} = \lambda i$

- (2.100), Example 2.3:

$$\begin{aligned}\mathbf{a}\wedge\mathbf{b}\wedge\mathbf{c} &= \sum_{\mathrm{i}} a_{\mathrm{i}}\mathbf{e}_{\mathrm{i}} \wedge \sum_{\mathrm{j}} b_{\mathrm{j}}\mathbf{e}_{\mathrm{j}} \wedge \sum_{\mathrm{k}} c_{\mathrm{k}}\mathbf{e}_{\mathrm{k}} \\ &= \sum_{\mathrm{i,j,k}} a_{\mathrm{i}} b_{\mathrm{j}} c_{\mathrm{k}}\mathbf{e}_{\mathrm{i}}\wedge\mathbf{e}_{\mathrm{j}}\wedge\mathbf{e}_{\mathrm{k}} = \sum_{\mathrm{i,j,k}} a_{\mathrm{i}} b_{\mathrm{j}} c_{\mathrm{k}} i\mathbf{e}_{\mathrm{i}}\cdot(\mathbf{e}_{\mathrm{j}}\times\mathbf{e}_{\mathrm{k}}) \\ &= \lambda i\end{aligned} \tag{C5}$$

C6: $r_{\mathrm{jk}} = (2a^2-1)\,\mathbf{e}_{\mathrm{j}}\cdot\mathbf{e}_{\mathrm{k}} - 2\sum_{\mathrm{i}} ab_{\mathrm{i}}\mathbf{e}_{\mathrm{i}}\cdot(\mathbf{e}_{\mathrm{j}}\times\mathbf{e}_{\mathrm{k}}) + 2b_{\mathrm{j}}b_{\mathrm{k}}$

- (2.105), (2.60), (2.62), (B1), (2.43), (2.25), and (2.42):

$$\begin{aligned}r_{\mathrm{jk}} &= \mathbf{e}_{\mathrm{j}}\cdot\mathcal{R}\mathbf{e}_{\mathrm{k}} = \mathbf{e}_{\mathrm{j}}\cdot(R^{\dagger}\mathbf{e}_{\mathrm{k}}R) \\ &= \mathbf{e}_{\mathrm{j}}\cdot((a-i\mathbf{b})\,\mathbf{e}_{\mathrm{k}}\,(a+i\mathbf{b})) \\ &= \mathbf{e}_{\mathrm{j}}\cdot(a\mathbf{e}_{\mathrm{k}}a + a\mathbf{e}_{\mathrm{k}}i\mathbf{b} - i\mathbf{b}\mathbf{e}_{\mathrm{k}}a - i\mathbf{b}\mathbf{e}_{\mathrm{k}}i\mathbf{b}) \\ &= \mathbf{e}_{\mathrm{j}}\cdot(a\mathbf{e}_{\mathrm{k}}a + ia(\mathbf{e}_{\mathrm{k}}\mathbf{b} - \mathbf{b}\mathbf{e}_{\mathrm{k}}) - i^2\mathbf{b}\mathbf{e}_{\mathrm{k}}\mathbf{b}) \\ &= \mathbf{e}_{\mathrm{j}}\cdot(a^2\mathbf{e}_{\mathrm{k}} + 2ia(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b}) + \mathbf{b}\mathbf{e}_{\mathrm{k}}\mathbf{b})\end{aligned} \tag{C6.1}$$

- (2.26), (2.11), (2.43), and (2.27):

$$\begin{aligned}\mathbf{e}_{\mathrm{j}}\cdot(2ia(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b})) &= \tfrac{1}{2}(\mathbf{e}_{\mathrm{j}}(2ia(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b})) - \\ &\qquad -(-1)^{1}(2ia(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b}))\,\mathbf{e}_{\mathrm{j}}) \\ &= ia(\mathbf{e}_{\mathrm{j}}(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b}) + (\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b})\,\mathbf{e}_{\mathrm{j}}) \\ &= 2ia(\mathbf{e}_{\mathrm{j}}\wedge(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b})) \\ &= 2ia\mathbf{e}_{\mathrm{j}}\wedge\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b}\end{aligned} \tag{C6.2}$$

- (2.25) and Example 2.4:

$$\begin{aligned}\mathbf{e}_{\mathrm{j}}\cdot(\mathbf{b}\mathbf{e}_{\mathrm{k}}\mathbf{b}) &= \mathbf{e}_{\mathrm{j}}\cdot(\mathbf{b}(\mathbf{e}_{\mathrm{k}}\cdot\mathbf{b} + \mathbf{e}_{\mathrm{k}}\wedge\mathbf{b})) \\ &= \mathbf{e}_{\mathrm{j}}\cdot(\mathbf{b}b_{\mathrm{k}} + \mathbf{b}\cdot(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b}) + \mathbf{b}\wedge(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b})) \\ &= \mathbf{e}_{\mathrm{j}}\cdot(\mathbf{b}b_{\mathrm{k}} + \mathbf{b}\cdot\mathbf{e}_{\mathrm{k}}\mathbf{b} - \mathbf{b}\cdot\mathbf{b}\mathbf{e}_{\mathrm{k}} + \mathbf{b}\wedge(\mathbf{e}_{\mathrm{k}}\wedge\mathbf{b})) \\ &= \mathbf{e}_{\mathrm{j}}\cdot\mathbf{b}b_{\mathrm{k}} + b_{\mathrm{k}}\mathbf{e}_{\mathrm{j}}\cdot\mathbf{b} - \mathbf{b}^2\mathbf{e}_{\mathrm{j}}\cdot\mathbf{e}_{\mathrm{k}} \\ &= 2b_{\mathrm{j}}b_{\mathrm{k}} - \mathbf{b}^2\mathbf{e}_{\mathrm{j}}\cdot\mathbf{e}_{\mathrm{k}}\end{aligned} \tag{C6.3}$$

- (C6.1), (C6.2), (C6.3), (2.61), Example 2.3, and (2.42):

$$\begin{aligned} r_{jk} &= (a^2 - \mathbf{b}^2)\,\mathbf{e}_j \cdot \mathbf{e}_k + \sum_i 2iab_i \mathbf{e}_i \wedge \mathbf{e}_j \wedge \mathbf{e}_k + 2b_j b_k \\ &= (2a^2 - 1)\,\mathbf{e}_j \cdot \mathbf{e}_k - \sum_i 2ab_i \mathbf{e}_i \cdot (\mathbf{e}_j \times \mathbf{e}_k) + 2b_j b_k \end{aligned} \tag{C6}$$

D Kinematics

D1: ${}^{i}\dot{R} = \frac{1}{2}\, i\, {}^{i}R \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\omega'_{j}$

- (2.87), (2.78), (2.15), (2.61), and (2.60):

$$
\begin{aligned}
{}^{i}\dot{R} &= \frac{\mathrm{d}}{\mathrm{d}t}\,(R'_{i}R'_{i+1}\ldots R'_{n}) \\
&= \dot{R}'_{i}R'_{i+1}\ldots R'_{n} + R'_{i}\dot{R}'_{i+1}\ldots R'_{n} + \ldots + R'_{i}R'_{i+1}\ldots\dot{R}'_{n} \\
&= (\tfrac{1}{2}R'_{i}\,i\omega')\,{}_{i}R'_{i+1} + \ldots R'_{n} + R'_{i}\,(\tfrac{1}{2}R'_{i+1}\,i\omega'_{i+1})\ldots R'_{n} + \ldots \\
&\quad R'_{i}R'_{i+1}\ldots(\tfrac{1}{2}R'_{n}\,i\omega'_{n}) \qquad \text{(D1)} \\
&= \tfrac{1}{2}\,i\,(R'_{i}R'_{i+1}\ldots R'_{n})\;((R'_{i+1}\ldots R'_{n})^{\dagger}\,\omega'_{i}\,(R'_{i+1}\ldots R'_{n}) + \ldots + \omega'_{n}) \\
&= \tfrac{1}{2}\,i\,{}^{i}R\,({}^{i+1}R^{\dagger}\,\omega'_{i}\,{}^{i+1}R + \ldots + \omega'_{n}) \\
&= \tfrac{1}{2}\,i\,{}^{i}R \sum_{j=i}^{n} {}^{j+1}R^{\dagger}\,\omega'_{j}\,{}^{j+1}R = \tfrac{1}{2}\,i\,{}^{i}R \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\omega'_{j}
\end{aligned}
$$

D2: $${}^{i}\dot{\mathbf{a}} = \sum_{j=i+1}^{n} {}^{j+1}\mathcal{R}\omega'_{j} \times ({}^{i}\mathbf{a} - {}^{j}\mathbf{a}) + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_{j}$$

- (2.88) and (2.77):

$$
\begin{aligned}
{}^{i}\dot{\mathbf{a}} &= \frac{\mathrm{d}}{\mathrm{d}t}\,(\sum_{j=i}^{n} {}^{j+1}\mathcal{R}\mathbf{a}'_{j}) = \sum_{j=i}^{n} {}^{j+1}\dot{\mathcal{R}}\mathbf{a}'_{j} + {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_{j} \\
&= \sum_{j=i}^{n} ({}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_{j} + {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_{j}) \qquad \text{(D2.1)} \\
&= \sum_{j=i}^{n-1} {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_{j} + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_{j}
\end{aligned}
$$

- (2.92):

$$\begin{aligned}\sum_{j=i}^{n-1} {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_j &= \sum_{j=i}^{n-1} \Big(\sum_{j^*=j+1}^{n} {}^{j^*+1}\mathcal{R}\omega'_{j^*} \Big) \times {}^{j+1}\mathcal{R}\mathbf{a}'_j \\ &= \sum_{j^*=i+1}^{n} \Big({}^{j^*+1}\mathcal{R}\omega'_{j^*} \times \sum_{j=i}^{j^*-1} {}^{j+1}\mathcal{R}\mathbf{a}'_j \Big) \\ &= \sum_{j^*=i+1}^{n} \Big({}^{j^*+1}\mathcal{R}\omega'_{j^*} \times \Big(\sum_{j=i}^{n} {}^{j+1}\mathcal{R}\mathbf{a}'_j - \sum_{j=j^*}^{n} {}^{j+1}\mathcal{R}\mathbf{a}'_j \Big) \Big) \\ &= \sum_{j^*=i+1}^{n} \Big({}^{j^*+1}\mathcal{R}\omega'_{j^*} \times ({}^{i}\mathbf{a} - {}^{j^*}\mathbf{a}) \Big)\end{aligned} \tag{D2.2}$$

- (D2.1) and (D2.2):

$$\begin{aligned}\sum_{j=i}^{n-1} {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j = \sum_{j^*=i+1}^{n} \Big({}^{j^*+1}\mathcal{R}\omega'_{j^*} \times ({}^{i}\mathbf{a} - {}^{j^*}\mathbf{a}) \Big) + \\ \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j\end{aligned} \tag{D2}$$

D3: $${}^{i}\dot{\omega} = \sum_{j=i}^{n-1} {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\omega'_j + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\omega}'_j$$

- (2.77):

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t} \Big(\sum_{j=i}^{n} {}^{j+1}\mathcal{R}\omega'_j \Big) &= \sum_{j=i}^{n} \Big({}^{j+1}\dot{\mathcal{R}}\omega'_j + {}^{j+1}\mathcal{R}\dot{\omega}'_j \Big) \\ &= \sum_{j=i}^{n-1} {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\omega'_j + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\omega}'_j\end{aligned} \tag{D3}$$

D4: $$\begin{aligned}{}^{i}\ddot{\mathbf{a}} = \sum_{j=i+1}^{n} {}^{j+1}\mathcal{R}\dot{\omega}'_j \times ({}^{i}\mathbf{a} - {}^{j}\mathbf{a}) + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\ddot{\mathbf{a}}'_j + \\ \sum_{j=i}^{n-1} \Big(\sum_{j^*=j+1}^{n-1} ({}^{j^*+1}\omega \times {}^{j^*+1}\mathcal{R}\omega'_{j^*}) \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + \\ {}^{j+1}\omega \times ({}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + 2\,{}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j) \Big)\end{aligned}$$

- (D2.1):

$$
\begin{aligned}
{}^{i}\ddot{\mathbf{a}} &= \frac{d}{dt}({}^{i}\dot{\mathbf{a}}) = \frac{d}{dt}\Big(\sum_{j=i}^{n-1} {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j\Big) \\
&= \sum_{j=i}^{n-1} \big({}^{j+1}\dot{\omega} \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + {}^{j+1}\omega \times {}^{j+1}\dot{\mathcal{R}}\mathbf{a}'_j + {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j\big) + \\
&\quad \sum_{j=i}^{n} \big({}^{j+1}\dot{\mathcal{R}}\dot{\mathbf{a}}'_j + {}^{j+1}\mathcal{R}\ddot{\mathbf{a}}'_j\big) \qquad \text{(D4.1)} \\
&= \sum_{j=i}^{n-1} \big({}^{j+1}\dot{\omega} \times {}^{j+1}\mathcal{R}\mathbf{a}'_j + {}^{j+1}\omega \times ({}^{j+1}\omega \times {}^{j+1}\mathcal{R}\mathbf{a}'_j) + \\
&\qquad {}^{j+1}\omega \times {}^{j+1}\mathcal{R}\dot{\mathbf{a}}'_j\big) + \sum_{j=i}^{n} \big({}^{j+1}\omega \times {}^{j+1}\dot{\mathcal{R}}\dot{\mathbf{a}}'_j + {}^{j+1}\mathcal{R}\ddot{\mathbf{a}}'_j\big)
\end{aligned}
$$

- first term from (D4.1) and (D3):

$$
\begin{aligned}
&\sum_{j=i}^{n-1} {}^{j+1}\dot{\omega} \times {}^{j+1}\mathcal{R}\mathbf{a}'_j \\
&\qquad = \sum_{j=i}^{n-1} \Big(\sum_{j^*=j+1}^{n-1} {}^{j^*+1}\omega \times {}^{j^*+1}\mathcal{R}\omega'_{j^*} + \sum_{j^*=j+1}^{n} {}^{j^*+1}\mathcal{R}\dot{\omega}'_{j^*}\Big) \times {}^{j+1}\mathcal{R}\mathbf{a}'_j
\end{aligned}
\qquad \text{(D4.2)}
$$

- second term from (D4.2):

$$
\begin{aligned}
\sum_{j=i}^{n-1} \Big(\sum_{j^*=j+1}^{n} {}^{j^*+1}\mathcal{R}\dot{\omega}'_{j^*}\Big) \times {}^{j+1}\mathcal{R}\mathbf{a}'_j &= \sum_{j^*=i+1}^{n} \Big({}^{j^*+1}\mathcal{R}\dot{\omega}'_{j^*} \times \sum_{j=i}^{j^*-1} {}^{j+1}\mathcal{R}\mathbf{a}'_j\Big) \\
&= \sum_{j^*=i+1}^{n} \Big({}^{j^*+1}\mathcal{R}\dot{\omega}'_{j^*} \times \Big(\sum_{j=i}^{n} {}^{j+1}\mathcal{R}\mathbf{a}'_j - \sum_{j=j^*}^{n} {}^{j+1}\mathcal{R}\mathbf{a}'_j\Big)\Big) \qquad \text{(D4.3)} \\
&= \sum_{j^*=i+1}^{n} {}^{j^*+1}\mathcal{R}\dot{\omega}'_{j^*} \times ({}^{i}\mathbf{a} - {}^{j^*}\mathbf{a})
\end{aligned}
$$

- (D4.1), (D4.2), and (D4.3):

$$\sum_{j=i}^{n-1}\Big(\Big(\sum_{j^*=j+1}^{n-1} {}^{j^*+1}\omega \times {}^{j^*+1}\mathcal{R}\,{}^{\prime}\omega_{j^*} + \sum_{j^*=j+1}^{n} {}^{j^*+1}\mathcal{R}\,{}^{\prime}\dot{\omega}_{j^*}\Big) \times {}^{j+1}\mathcal{R}\,{}^{\prime}\mathbf{a}_j + {}^{j+1}\omega \times \big({}^{j+1}\omega \times {}^{j+1}\mathcal{R}\,{}^{\prime}\mathbf{a}_j\big) + 2\,{}^{j+1}\omega \times {}^{j+1}\mathcal{R}\,{}^{\prime}\dot{\mathbf{a}}_j\Big) + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\,{}^{\prime}\ddot{\mathbf{a}}_j$$

$$= \sum_{j=i+1}^{n} {}^{j+1}\mathcal{R}\,{}^{\prime}\dot{\omega}_j \times \big({}^{i}\mathbf{a} - {}^{j}\mathbf{a}\big) + \sum_{j=i}^{n} {}^{j+1}\mathcal{R}\,{}^{\prime}\ddot{\mathbf{a}}_j + \sum_{j=i}^{n-1}\Big(\sum_{j^*=j+1}^{n-1}\big({}^{j^*+1}\omega \times {}^{j^*+1}\mathcal{R}\,{}^{\prime}\omega_{j^*}\big) \times {}^{j+1}\mathcal{R}\,{}^{\prime}\mathbf{a}_j + {}^{j+1}\omega \times \big({}^{j+1}\omega \times {}^{j+1}\mathcal{R}\,{}^{\prime}\mathbf{a}_j + 2\,{}^{j+1}\mathcal{R}\,{}^{\prime}\dot{\mathbf{a}}_j\big)\Big) \qquad \text{(D4)}$$

E Dyadic Reduction

E1: $\tilde{d}_p = d_p + d_r b_{rp}^2$

- (4.33), (4.34), (4.35) for j = k = p:

$$\begin{aligned} & q_{p\,jk} + q_{r\,jk} = \tilde{q}_{p\,jk} + \tilde{q}_{r\,jk} \\ = \; & d_p b_{pp} b_{pp} + d_r b_{rp} b_{rp} = \tilde{d}_p \tilde{b}_{pp} \tilde{b}_{pp} + \tilde{d}_r \tilde{b}_{rp} \tilde{b}_{rp} \\ = \; & d_p + d_r b_{rp}^2 = \tilde{d}_p + 0 \end{aligned} \tag{E1}$$

E2: $\tilde{b}_{pj} = \dfrac{d_p b_{pj} + d_r b_{rp} b_{rj}}{\tilde{d}_p}$

- for j > p and k = p:

$$\begin{aligned} & q_{p\,jp} + q_{r\,jp} = \tilde{q}_{p\,jp} + \tilde{q}_{r\,jp} \\ = \; & d_p b_{pj} b_{pp} + d_r b_{rj} b_{rp} = \tilde{d}_p \tilde{b}_{pj} \tilde{b}_{pp} + \tilde{d}_r \tilde{b}_{rj} \tilde{b}_{rp} \\ = \; & d_p b_{pj} + d_r b_{rp} b_{rj} = \tilde{d}_p \tilde{b}_{pj} + 0 \end{aligned} \tag{E2}$$

E3: $\tilde{d}_r = \dfrac{d_p d_r}{\tilde{d}_p}$ and $\tilde{b}_{rj} = b_{rj} - b_{rp} b_{pj}$

- for j > 0 and k > 0:

$$\begin{aligned} & q_{p\,jk} + q_{r\,jk} = \tilde{q}_{p\,jk} + \tilde{q}_{r\,jk} \\ = \; & d_p b_{pj} b_{pk} + d_r b_{rj} b_{rk} = \tilde{d}_p \tilde{b}_{pj} \tilde{b}_{pk} + \tilde{d}_r \tilde{b}_{rj} \tilde{b}_{rk} \end{aligned} \tag{E3.1}$$

- substitution of (E2) in (E3.1):

$$\begin{aligned} & d_p b_{pj} b_{pk} + d_r b_{rj} b_{rk} \\ = \; & \tilde{d}_p \left(\frac{d_p b_{pj} + d_r b_{rp} b_{rj}}{\tilde{d}_p} \right) \left(\frac{d_p b_{pk} + d_r b_{rp} b_{rk}}{\tilde{d}_p} \right) + \tilde{d}_r \tilde{b}_{rj} \tilde{b}_{rk} \end{aligned} \tag{E3.2}$$

- transformation of (E3.2):

$$\begin{aligned}\tilde{d}_r \tilde{b}_{rj} \tilde{b}_{rk} &= d_p b_{pj} b_{pk} + d_r b_{rj} b_{rk} - \\ &\quad \tilde{d}_p \left(\frac{d_p b_{pj} + d_r b_{rp} b_{rj}}{\tilde{d}_p}\right) \left(\frac{d_p b_{pk} + d_r b_{rp} b_{rk}}{\tilde{d}_p}\right) \\ &= d_p \left(1 - \frac{d_p}{\tilde{d}_p}\right) b_{pj} b_{pk} + d_r \left(1 - \frac{d_r b_{rp}^2}{\tilde{d}_p}\right) b_{rj} b_{rk} - \\ &\quad \frac{d_p d_r}{\tilde{d}_p} \left(b_{rp} b_{pj} b_{rk} + b_{rp} b_{rj} b_{pk}\right)\end{aligned} \tag{E3.3}$$

- (E1):

$$1 - \frac{d_p}{\tilde{d}_p} = \frac{\tilde{d}_p - d_p}{\tilde{d}_p} = \frac{d_r b_{rp}^2}{\tilde{d}_p} \tag{E3.4}$$

- (E1):

$$1 - \frac{d_r b_{rp}^2}{\tilde{d}_p} = \frac{\tilde{d}_p - d_r b_{rp}^2}{\tilde{d}_p} = \frac{d_p}{\tilde{d}_p} \tag{E3.5}$$

- (E3.3), (E3.4), (E3.5):

$$\begin{aligned}\tilde{d}_r \tilde{b}_{rj} \tilde{b}_{rk} &= d_p \frac{d_r b_{rp}^2}{\tilde{d}_p} b_{pj} b_{pk} + d_r \frac{d_p}{\tilde{d}_p} b_{rj} b_{rk} - \\ &\quad \frac{d_p d_r}{\tilde{d}_p} \left(b_{rp} b_{pj} b_{rk} + b_{rp} b_{rj} b_{pk}\right) \\ &= \frac{d_p d_r}{\tilde{d}_p} \left(b_{rp}^2 b_{pj} b_{pk} + b_{rj} b_{rk} - b_{rp} b_{pj} b_{rk} - b_{rp} b_{rj} b_{pk}\right) \\ &= \frac{d_p d_r}{\tilde{d}_p} \left(b_{rj} - b_{rp} b_{pj}\right) \left(b_{rk} - b_{rp} b_{pk}\right)\end{aligned} \tag{E3}$$

E4: $\tilde{b}_{pk} = b_{pk} + k_r \tilde{b}_{rk}$ and $k_r = \frac{d_r b_{rp}}{\tilde{d}_p}$

- (E2), (E3) and (E1):

$$\begin{aligned}\tilde{b}_{pj} &= \frac{d_p b_{pj} + d_r b_{rp} b_{rj}}{\tilde{d}_p} \\ &= \frac{d_p b_{pj} + d_r b_{rp} (\tilde{b}_{rj} + b_{rp} b_{pj})}{\tilde{d}_p} \\ &= \frac{(d_p + d_r b_{rp}^2)\, b_{pj} + d_r b_{rp} \tilde{b}_{rj}}{\tilde{d}_p} \\ &= \frac{\tilde{d}_p b_{pj} + d_r b_{rp} \tilde{b}_{rj}}{\tilde{d}_p} = b_{pj} + \frac{d_r b_{rp} \tilde{b}_{rj}}{\tilde{d}_p}\end{aligned} \tag{E4}$$

E5:
$$\sum_{k=p+1}^{n+1} \sum_{j=p+1}^{k-1} \nu_{ij} \Delta\nu_{ik} + \sum_{k=p+1}^{n} (5 + \sum_{j=k+1}^{n+1} (\nu_{ij} + \Delta\nu_{ij}))\, \Delta\nu_{ik}$$
$$= \nu_i \Delta\nu_i + (\Delta\nu_i - \Delta\nu_{i\,n+1})\, (5 + \frac{\Delta\nu_i - 1 + \Delta\nu_{i\,n+1}}{2})$$

- (4.53):

$$\begin{aligned}&\sum_{k=p+1}^{n+1} \sum_{j=p+1}^{k-1} \nu_{ij} \Delta\nu_{ik} + \sum_{k=p+1}^{n} (5 + \sum_{j=k+1}^{n+1} (\nu_{ij} + \Delta\nu_{ij}))\, \Delta\nu_{ik} \\ &= \sum_{k=p+1}^{n+1} (\sum_{j=p+1}^{k-1} \nu_{ij} + \sum_{j=k+1}^{n+1} \nu_{ij})\, \Delta\nu_{ik} - \sum_{j=n+2}^{n+1} \Delta\nu_{ij} \Delta\nu_{i\,n+1} + \\ &\quad \sum_{k=p+1}^{n} 5\Delta\nu_{ik} + \sum_{k=p+1}^{n} \sum_{j=k+1}^{n+1} \Delta\nu_{ij} \Delta\nu_{ik} \\ &= \sum_{k=p+1}^{n+1} \sum_{j=p+1}^{n+1} \nu_{ij} \Delta\nu_{ik} + \sum_{k=p+1}^{n} 5\Delta\nu_{ik} + \sum_{k=p+1}^{n} \sum_{j=k+1}^{n+1} \Delta\nu_{ij} \Delta\nu_{ik}\end{aligned} \tag{E5.1}$$

- (4.51), sum of finite series:

$$\begin{aligned}\sum_{k=p+1}^{n} \sum_{j=k+1}^{n+1} \Delta\nu_{ij} \Delta\nu_{ik} &= (\Delta\nu_i - 1) + (\Delta\nu_i - 2) + \ldots + \Delta\nu_{i\,n+1} \\ &= (\Delta\nu_i - \Delta\nu_{i\,n+1}) \frac{\Delta\nu_i - 1 + \Delta\nu_{i\,n+1}}{2}\end{aligned} \tag{E5.2}$$

- (E5.1), (4.46), (4.51), and (E5.2):

$$\begin{aligned}
&\sum_{k=p+1}^{n+1}\sum_{j=p+1}^{n+1} \nu_{ij}\Delta\nu_{ik} + \sum_{k=p+1}^{n} 5\Delta\nu_{ik} + \sum_{k=p+1}^{n}\sum_{j=k+1}^{n+1} \Delta\nu_{ij}\Delta\nu_{ik} \\
&= \nu_i\Delta\nu_i + 5\,(\Delta\nu_i - \Delta\nu_{i\,n+1}) + (\Delta\nu_i - \Delta\nu_{i\,n+1})\,\frac{\Delta\nu_i - 1 + \Delta\nu_{i\,n+1}}{2} \qquad \text{(E5)} \\
&= \nu_i\Delta\nu_i + (\Delta\nu_i - \Delta\nu_{i\,n+1})\,(5 + \frac{\Delta\nu_i - 1 + \Delta\nu_{i\,n+1}}{2})
\end{aligned}$$

E6:
$$\begin{aligned}
J(p,r) &= \Delta\nu_r + \sum_{i=n+1}^{n+m} (\nu_{ip}\Delta\nu_p + (1+\nu_i)\Delta\nu_i) + \\
&\quad \sum_{i=n+1}^{n+m} (\Delta\nu_i - \Delta\nu_{i\,n+1})\,(5 + \frac{\Delta\nu_i - 1 + \Delta\nu_{i\,n+1}}{2})
\end{aligned}$$

- (4.50), (4.52), (4.53), and (E5):

$$\begin{aligned}
J(p,r) &= 2\Delta\nu_r + \Delta\nu_p + \sum_{\substack{i=n+1 \\ i\neq r}}^{n+m} (\nu_{ip}\Delta\nu_p + \Delta\nu_i) + \\
&\qquad \sum_{i=n+1}^{n+m} (\nu_i\Delta\nu_i + (\Delta\nu_i - \Delta\nu_{i\,n+1})\,(5 + \frac{\Delta\nu_i - 1 + \Delta\nu_{i\,n+1}}{2}))) \\
&= \Delta\nu_r + \sum_{i=n+1}^{n+m} (\nu_{ip}\Delta\nu_p + \Delta\nu_i) \quad + \qquad \text{(E6)} \\
&\qquad \sum_{i=n+1}^{n+m} (\nu_i\Delta\nu_i + (\Delta\nu_i - \Delta\nu_{i\,n+1})\,(5 + \frac{\Delta\nu_i - 1 + \Delta\nu_{i\,n+1}}{2}))) \\
&= \Delta\nu_r + \sum_{i=n+1}^{n+m} (\nu_{ip}\Delta\nu_p + (1+\nu_i)\Delta\nu_i) + \\
&\qquad \sum_{i=n+1}^{n+m} (\Delta\nu_i - \Delta\nu_{i\,n+1})\,(5 + \frac{\Delta\nu_i - 1 + \Delta\nu_{i\,n+1}}{2})
\end{aligned}$$

F Matrix Representation of Constraint Equations

F1: Notational Definitions and Conventions

- The ranges of the different indices are specified as:

$$\begin{aligned} \alpha, \beta &\in [1, N] \\ a &\in [1, n] \\ j &\in [1, n_a] \\ p &\in [1, n_u] \end{aligned} \tag{F1.1}$$

column-

$$\text{wise.k} \in \begin{cases} [1, n_{rot}] & \text{for } \Delta\vartheta_{\alpha_k}, \omega_{\alpha_k}, \dot{\omega}_{\alpha_k}, \tau_{a_k} \\ [1, n_{trans}] & \text{for } \Delta X_{\alpha_k}, \dot{X}_{\alpha_k}, \ddot{X}_{\alpha_k}, f_{a_k} \\ [1, n_{raj}] & \text{for } \hat{\underline{D}}_{aj_k}, \Delta\varphi'_{aj_k}, \theta'_{aj_k}, \dot{\theta}'_{aj_k} \\ [1, n_{taj}] & \text{for } \hat{\underline{s}}_{aj_k}, \Delta s'_{aj_k}, \dot{s}'_{aj_k}, \ddot{s}'_{aj_k} \end{cases} \tag{F1.2}$$

- The *col* operator determines that the indexed elements are aligned column-wise.
- The *row* operator symbolizes that the indexed elements are aligned row-wise.
- The *diag* operator indicates that the indexed elements are aligned in the diagonal of a matrix with all other elements equal to zero.
- An index is said to be *free* if it does not appear on the left side of the equation. Free indices always run over their entire range (cf. (F1.1) and (F1.2)).
- For elements with more than one index the indices are incremented from right to left, i.e., the rightmost index runs from one to its upper bound before the second index from the right is incremented by one

and the rightmost index begins to run from one to the upper bound again.

Example E1.1:

In the following two examples, the index k is a free index in contrast to the indices α, a, and j. The index range of k depends on the variable and is specified in (F1.2)

$$\begin{aligned} \underline{\omega}_\alpha &= col(\omega_{\alpha_k}) = \left[\omega_{\alpha_1}\ \omega_{\alpha_2}\ \omega_{\alpha_3}\right]^T \\ \underline{\theta}'_{aj} &= col(\theta'_{aj_k}) = \left[\theta'_{aj_1} \ldots \theta'_{aj_{n_{raj}}}\right]^T. \end{aligned} \tag{F1.3}$$

Example E1.2:

Assume that $n = 1$, $n_1 = 2$, $n_{r11} = 3$, and $n_{r12} = 1$. In this case $col(b_{aj_k})$ is the short form for

$$\left[b_{11_1}\, b_{11_2}\, b_{11_3}\, b_{12_1}\right]^T. \tag{F1.4}$$

Example E1.3:

In $J_{1a} = row(\hat{\underline{b}}_{aj_k})$, again with $k \in [1, n_{raj}]$ (cf. (F1.2)) and $n = 5$, the matrix J_{1a} stands for

$$\left[\hat{\underline{b}}_{a1_1}\, \hat{\underline{b}}_{a1_2} \ldots \hat{\underline{b}}_{11_{n_{r11}}}\, \hat{\underline{b}}_{a2_1} \ldots \hat{\underline{b}}_{a2_{n_{r12}}} \ldots \hat{\underline{b}}_{a5_{n_{r15}}}\right]. \tag{F1.5}$$

F2: General Definitions

- Unification of vectors with n_{rot} components:

$$\begin{aligned} \Delta\underline{\vartheta} &= col(\Delta\vartheta_{\alpha_k}) \\ \underline{\omega} &= col(\omega_{\alpha_k}) \\ \underline{\dot{\omega}} &= col(\dot{\omega}_{\alpha_k}) \\ \underline{\tau} &= col(\tau_{a_k}) \end{aligned} \tag{F2.1}$$

- Unification of vectors with n_{trans} components:

$$\begin{aligned} \Delta\underline{X} &= col\,(\Delta X_{\alpha_k}) \\ \dot{\underline{X}} &= col\,(\dot{X}_{\alpha_k}) \\ \ddot{\underline{X}} &= col\,(\ddot{X}_{\alpha_k}) \\ \underline{f} &= col\,(f_{a_k}) \end{aligned} \qquad \text{(F2.2)}$$

- Unification of vectors with n_{raj} components:

$$\begin{aligned} \Delta\underline{\varphi}' &= col\,(\Delta\varphi'_{aj_k}) \\ \underline{\theta}' &= col\,(\theta'_{aj_k}) \\ \dot{\underline{\theta}}' &= col\,(\dot{\theta}'_{aj_k}) \end{aligned} \qquad \text{(F2.3)}$$

- Unification of vectors with n_{taj} components:

$$\begin{aligned} \Delta\underline{s}' &= col\,(\Delta s'_{aj_k}) \\ \dot{\underline{s}}' &= col\,(\dot{s}'_{aj_k}) \\ \ddot{\underline{s}}' &= col\,(\ddot{s}'_{aj_k}) \end{aligned} \qquad \text{(F2.4)}$$

- Connectivity matrices (cf. (5.4) and (5.5)):

$$C_i = \left[C_{i\alpha a}\right] \qquad \begin{aligned} C_{1\alpha a} &= c_{\alpha a}\left[\mathbf{e}_j \cdot \mathbf{e}_k\right] \qquad j,k \in [1, n_{rot}] \\ C_{1S\alpha a} &= c^{+}_{\alpha a}\bar{\bar{S}}_a^{\,T} - c^{-}_{\alpha a}\bar{\bar{R}}_\alpha^{\,T} \\ C_{2\alpha a} &= c^{+}_{\alpha a}\, row\,((\underline{s}_a - \underline{X}_\alpha) \times \underline{e}_k) \\ C_{3\alpha a} &= c_{\alpha a}\left[\mathbf{e}_j \cdot \mathbf{e}_k\right] \qquad j,k \in [1, n_{trans}] \end{aligned} \qquad \text{(F2.5)}$$

- Jacobian matrices (cf. (5.4), (5.5), (5.9), and (5.10)):

$$J_i = diag(J_{ia}) \qquad \begin{aligned} J_{1a} &= row(\hat{\underline{b}}_{aj_k}) \\ J_{1Sa} &= row(\tilde{\underline{S}}_a \hat{\underline{b}}_{aj_k}) \\ J_{2a} &= row(\hat{\underline{b}}_{aj_k} \times (\underline{s}_a - {}^j\underline{s}_a)) \\ J_{3a} &= row(\hat{\underline{s}}_{aj_k}) \end{aligned} \qquad \text{(F2.6)}$$

F3: Positional Constraints in Matrix Notation

- Specification of intermediate variables $\underline{b}_{ipos}$ (cf. (5.4) and (5.5)):

$$\underline{b}_{ipos} = col(\underline{b}_{ipos_a}) \qquad \begin{aligned} \underline{b}_{1pos_a} &= 2(c_{\alpha a}^{+} \underline{S}_a - c_{\alpha a}^{-} \underline{R}_\alpha) \\ \underline{b}_{2pos_a} &= c_{\alpha a}^{+} \underline{s}_a - c_{\alpha a}^{-} \underline{X}_\alpha \end{aligned} \qquad \text{(F3.1)}$$

- Representation of (3.43) and (3.44) in matrix notation (cf. (F2.1), (F2.2), (F2.3), (F2.4), (F2.5), (F2.6), and (F3.1)):

$$J_{1S} \Delta\underline{\varphi}' + C_1^T \Delta\underline{\vartheta} + \underline{b}_{1pos} = 0 \qquad \text{(F3.2)}$$

$$J_2 \Delta\underline{\varphi}' + J_3 \Delta\underline{s}' + C_2^T \Delta\underline{\vartheta} + C_3^T \Delta\underline{X} + \underline{b}_{2pos} = 0 \qquad \text{(F3.3)}$$

F4: Velocity Constraints in Matrix Notation

- Representation of (3.55) and (3.56) in matrix notation (cf. (F2.1), (F2.2), (F2.3), (F2.4), (F2.5), and (F2.6)):

$$J_1 \underline{\theta}' + C_1^T \underline{\omega} = 0 \qquad \text{(F4.1)}$$

$$J_2 \underline{\theta}' + J_3 \dot{\underline{s}}' + C_2^T \underline{\omega} + C_3^T \dot{\underline{X}} = 0 \qquad \text{(F4.2)}$$

F5: Acceleration Constraints in Matrix Notation

- Specification of intermediate variables $\underline{g}$ and $\underline{h}$ (cf. (2.95) and (2.96), (5.6), (5.7), and (5.8)):

$$\begin{aligned} \underline{g} &= col({}^{I}\underline{g}_a) \\ \underline{h} &= col({}^{I}\underline{h}_a) \end{aligned} \tag{F5.1}$$

- Representation of (3.63) and (3.64) in matrix notation (cf. (F2.1), (F2.2), (F2.3), (F2.4), (F2.5), (F2.6), and (F5.1)):

$$J_1\dot{\underline{\theta}}' + C_1{}^{T}\dot{\underline{\omega}} + \underline{g} = 0 \tag{F5.2}$$

$$J_2\dot{\underline{\theta}}' + J_3\ddot{\underline{s}}' + C_2{}^{T}\dot{\underline{\omega}} + C_3{}^{T}\ddot{\underline{X}} + \underline{h} = 0 \tag{F5.3}$$

F6: Force and Torque Constraints in Matrix Notation

- Specification of intermediate variables $\underline{\tau}'_P$, $\underline{f}'_P$:

$$\begin{aligned} \underline{\tau}'_P &= col(\tau'_{P\,aj}) \\ \underline{f}'_P &= col(f'_{P\,aj_k}) \end{aligned} \tag{F6.1}$$

- Representation of control vector $\underline{u}$ and the matrices B_{iu} (cf. (3.28), (3.29), and (3.30) for the definition of $b_{1u_{ir\,p}}$ and $b_{2u_{it\,p}}$:

$$\begin{aligned} \underline{u} &= col(u_p) \\ B_{1u} &= \left[b_{1u_{ir\,p}}\right] \\ B_{2u} &= \left[b_{2u_{it\,p}}\right] \end{aligned} \tag{F6.2}$$

- Representation of (3.71) and (3.74) in matrix notation (cf. (F2.1), (F2.2), (F2.6), (F6.1) and (F6.2)):

$$\mathsf{J}_1^{\mathrm{T}}\underline{\tau} + \mathsf{J}_2^{\mathrm{T}}\mathsf{f} - \mathsf{B}_{1u}\underline{\mathsf{u}} - \underline{\tau}'_P = 0 \tag{F6.3}$$

$$\mathsf{J}_3^{\mathrm{T}}\mathsf{f} - \mathsf{B}_{2u}\underline{\mathsf{u}} - \underline{\mathsf{f}}'_P = 0 \tag{F6.4}$$

F7: Dynamical Constraints in Matrix Notation

- Matrix representation of the inertia tensor for the arbitrary rigid body α, (2.105), (3.77), and (3.78):

$$\begin{aligned} I_{\alpha jk} &= \mathbf{e}_j \cdot (\mathcal{J}_\alpha \mathbf{e}_k) = \mathbf{e}_j \cdot \Big(\sum_{k=1}^{3} I_{\alpha_k} \mathbf{e}'_{\alpha_k} \mathbf{e}'_{\alpha_k} \cdot \mathbf{e}_k\Big) \\ &= \sum_{k=1}^{3} I_{\alpha_k} (\mathbf{e}_j \cdot \mathbf{e}'_{\alpha_k}) (\mathbf{e}'_{\alpha_k} \cdot \mathbf{e}_k) \end{aligned} \tag{F7.1}$$

- Matrix representation of the inertia tensor for the symmetric rigid body α with symmetry axis $\mathbf{e}'_{\alpha_1}$, (3.81):

$$\begin{aligned} I_{\alpha jk} &= \mathbf{e}_j \cdot (\mathcal{J}_\alpha \mathbf{e}_k) = \mathbf{e}_j \cdot (I_{\alpha_2} \mathbf{e}_k + (I_{\alpha_1} - I_{\alpha_2}) \mathbf{e}'_{\alpha_1} \mathbf{e}'_{\alpha_1} \cdot \mathbf{e}_k) \\ &= I_{\alpha_2} \mathbf{e}_j \cdot \mathbf{e}_k + (I_{\alpha_1} - I_{\alpha_2}) (\mathbf{e}_j \cdot \mathbf{e}'_{\alpha_1} \mathbf{e}'_{\alpha_1} \cdot \mathbf{e}_k) \end{aligned} \tag{F7.2}$$

- Matrix representation of the mass matrix of the rigid body α (cf. (F7.1) or (F7.2)):

$$\begin{aligned} \mathsf{I}_\alpha &= [I_{\alpha jk}] && j,k \in [1, n_{rot}] \\ \mathsf{m}_\alpha &= m_\alpha \left[\mathbf{e}_j \cdot \mathbf{e}_k\right] && j,k \in [1, n_{trans}] \\ \mathsf{M}_I &= diag\,(\mathsf{I}_\alpha) \\ \mathsf{M}_m &= diag\,(\mathsf{m}_\alpha) \end{aligned} \tag{F7.3}$$

- Specification of intermediate variables $\underline{\mathsf{b}}_{i\,dyn}$:

$$\underline{\mathsf{b}}_{i\,dyn} = col\,(\underline{\mathsf{b}}_{i\,dyn_\alpha}) \qquad \begin{aligned} \underline{\mathsf{b}}_{1\,dyn} &= col\,(\underline{\omega}_\alpha \times \mathsf{I}_\alpha \underline{\omega}_\alpha) \\ \underline{\mathsf{b}}_{2\,dyn} &= col\,(-\underline{F}_{g\alpha}) \end{aligned} \tag{F7.4}$$

- Representation of (3.92) and (3.93) in matrix notation (cf. (F2.1), (F2.2), (F2.5), (F7.3) and (F7.4)):

$$M_1 \dot{\underline{\omega}} + C_1 \underline{\tau} + C_2 f + \underline{b}_{1dyn} = 0 \qquad \text{(F7.5)}$$

$$M_m \ddot{\underline{X}} + C_3 f + \underline{b}_{2dyn} = 0 \qquad \text{(F7.6)}$$

G Weighting Factors

G1: Information Processing Level

- Diagonal elements of weighting matrix for positional constraints

$$\begin{aligned} \underline{d}_{0pos}^{T} &= \left[\underline{d}_{\vartheta}^{T}\ \underline{d}_{X}^{T}\ \underline{d}_{\varphi'}^{T}\ \underline{d}_{s'}^{T}\right], \\ \underline{d}_{1pos} &= col(1_i) \qquad i \in [1, 56] \end{aligned} \tag{G1.1}$$

$$\begin{aligned} \underline{d}_{\vartheta} &= col(d_{\vartheta\alpha_3}) \\ \underline{d}_{X} &= col(d_{X\alpha_k}) \\ \underline{d}_{\varphi'} &= col(d_{\varphi' a2_3}) \\ \underline{d}_{s'} &= col(d_{s' a2_k}) \qquad a \in \{4, 5, 9, 10\} \end{aligned} \tag{G1.2}$$

$d_{\vartheta\alpha_3}, d_{X\alpha_k}$	10^{-4}	$\alpha \in \{1, 2, 3, 5, 6, 8, 9, 10, 11\}$
$d_{\vartheta\alpha_3}, d_{X\alpha_k}$	10^{-8}	$\alpha \in \{4, 7\}$
$d_{\varphi' a2_3}$	10^{-2}	$a \in \{1, 2, 3, 6, 7, 8, 11, 12, 13, 14\}$
$d_{\varphi' a2_3}$	0	$a \in \{4, 5, 9, 10\}$
$d_{s' a2_1}$	1	
$d_{s' a2_2}$	10^{-2}	

Table G.1 Weights for Subtask "Positional Constraints"

- Diagonal elements of weighting matrix for velocity constraints

$$\begin{aligned} \underline{d}_{0vel}^{T} &= \left[\underline{d}_{\omega}^{T}\ \underline{d}_{\dot{X}}^{T}\ \underline{d}_{\theta'}^{T}\ \underline{d}_{\dot{s}'}^{T}\right], \\ \underline{d}_{1vel} &= col(1_i) \qquad i \in [1, 42] \end{aligned} \tag{G1.3}$$

$$
\begin{aligned}
&\underline{d}_{\omega} = col\,(d_{\omega\alpha_3}) \\
&\underline{d}_{\dot{x}} = col\,(d_{\dot{x}\alpha_k}) \\
&\underline{d}_{\theta'} = col\,(d_{\theta' a2_3}) \\
&\underline{d}_{\ddot{s}'} = col\,(d_{\ddot{s}' a2_k}) \qquad a \in \{4, 5, 9, 10\}
\end{aligned}
\tag{G1.4}
$$

$d_{\omega\alpha_3}, d_{\dot{x}\alpha_k}$	10^{-8}	$\alpha = 1$
$d_{\omega\alpha_3}, d_{\dot{x}\alpha_k}$	10^{-10}	$\alpha \in [2, 11]$
$d_{\theta' a2_3}$	10^{-8}	$a \in \{1, 2, 3, 6, 7, 8, 11, 12, 13, 14\}$
$d_{\theta' a2_3}$	10^{-10}	$a \in \{4, 5, 9, 10\}$
$d_{\ddot{s}' a2_1}$	1	
$d_{\ddot{s}' a2_2}$	10^{-8}	

Table G.2 Weights for Subtask "Velocity Constraints"

G2: Execution Level

- Diagonal elements of weighting matrix for dynamical constraints

$$
\underline{d}_{0dyn}^{T} = \left[\underline{d}_{u}^{T}\ \underline{d}_{\dot{\omega}}^{T}\ \underline{d}_{\ddot{x}}^{T}\ \underline{d}_{\tau}^{T}\ \underline{d}_{f}^{T}\ \underline{d}_{\dot{\theta}'}^{T}\ \underline{d}_{\ddot{s}'}^{T}\right], \tag{G2.1}
$$

$$
\underline{d}_{1dyn} = col\,(1_i) \qquad i \in [1, 97]
$$

$$
\begin{aligned}
&\underline{d}_{u} = col\,(d_{u_p}) \\
&\underline{d}_{\dot{\omega}} = col\,(d_{\dot{\omega}\alpha_3}) \\
&\underline{d}_{\ddot{x}} = col\,(d_{\ddot{x}\alpha_k}) \\
&\underline{d}_{\tau} = col\,(d_{\tau a_k}) \\
&\underline{d}_{f} = col\,(d_{f a_k}) \\
&\underline{d}_{\dot{\theta}'} = col\,(d_{\dot{\theta}' a2_3}) \\
&\underline{d}_{\ddot{s}'} = col\,(d_{\ddot{s}' a2_k}) \qquad a \in \{4, 5, 9, 10\}
\end{aligned}
\tag{G2.2}
$$

d_{u_p}	10^{-19}	$p \in [1, 10]$
$d_{\dot{\omega}1_3}$	from subtask demanded contact forces	—
$d_{\dot{\omega}\alpha_3}$	10^{-8}	$\alpha \in [2, 7]$
$d_{\dot{\omega}\alpha_3}$	10^{-6}	$\alpha \in [8, 11]$
$d_{\ddot{x}1_k}$	from subtask demanded contact forces	—
$d_{\ddot{x}\alpha_k}$	from subtask demanded accelerations	$\alpha \in \{2, 4, 5, 7\}$
$d_{\ddot{x}\alpha_k}$	10^{-19}	$\alpha \in \{3, 6, 8, 9, 10, 11\}$
$d_{\tau a_3}$	0	$a \in [1, 14]$
d_{fa_k}	0	$a \in \{1, 2, 3, 6, 7, 8, 11, 12, 13, 14\}$
d_{fa_k}	from subtask demanded contact forces	$a \in \{4, 5, 9, 10\}$
$d_{\ddot{\theta}'a2_3}$	10^{-19}	$a \in \{1, 4, 5, 6, 9, 10\}$
$d_{\ddot{\theta}'a2_3}$	from subtask demanded accelerations	$a \in \{2, 3, 7, 8\}$
$d_{\ddot{\theta}'a2_3}$	10^{-9}	$a \in \{11, 12, 13, 14\}$
$d_{\ddot{s}'a2_k}$	from subtask demanded accelerations	$a \in \{4, 5, 9, 10\}$

Table G.3 Weights for Subtask "Dynamical Constraints"

References

[Albus81] J. S. Albus, *'Brains, Behavior and Robotics,'* BYTE Books, Peterborough, N.H., 1981.

[AlMLBa85] J. S. Albus, C. R. McLean, A. J. Barbera, M. L. Fitzgerald, *'Hierarchical Control for Robots and Teleoperators,'* IEEE Workshop on Intelligent Control, Rensselaer Polytechnical Institute, Troy, N. Y., August, 1985, pp. 39-49.

[BaMiPa86] C. A. Balafoutis, P. Misra, R. V. Patel, *'Recursive Evaluation of Linearized Dynamic Robot Models,'* IEEE Journal of Robotics and Automation, Vol. RA-2, No. 3, September, 1986, pp. 146-155.

[BaPaMi88] C. A. Balafoutis, R. V. Patel, P. Misra, *'Efficient Modeling and Computation of Manipulator Dynamics Using Orthogonal Cartesian Tensors,'* IEEE Journal of Robotics and Automation, Vol. 4, No. 6, December, 1988, pp. 665-676.

[BalPat91] C. A. Balafoutis, R. V. Patel, *'Dynamic Analysis of Robot Manipulators. A Cartesian Approach,'* Kluwer-Academic Publishers, Boston/Dordrecht/London, 1991.

[BenGre80] A. Ben-Israel and T. N. Greville, *'Generalized Inverse: Theory and Applications,'* New York, Krieger, 1980.

[BusGee91] M. A. Busenhart, H. P. Geering, *'An Algorithm for the Motion Control of Complex Redundant Robot Manipulators,'* IEEE Int. Symposium on Intelligent Control, Arlington, VA, August, 1991, pp. 430-435.

[ChGeGo88] B. W. Char, K. O. Geddes, G. H. Gonnet, M. B. Monagan, S. M. Watt, *'Maple: Reference Manual,'* WATCOM Publications Limited, Waterloo, Ont., Canada, 1988.

[Corby85] N. Corby, *'Research Issue in Multi-Sensory Adaptive Control,'* Proceedings of the Workshop on Intelligent Robots, Palo Alto, CA, 1985.

[DesRot85] S. Desa, B. Roth, *'Synthesis of Control Systems for Manipulators Using Multivariable Robust Servomechanism Theory,'* Int. Journal of Robotics Research, Vol. 4, No. 3, Fall, 1985, pp. 18-34.

[Egelan87] O. Egeland, *'Task-Space Tracking with Redundant Manipulators,'* IEEE Journal of Robotics and Automation, Vol.. RA-3, No. 5, October, 1987, pp. 471-475.

[Freund82] E. Freund, *'Fast Nonlinear Control with Arbitrary Pole-Placement for Industrial Robots and Manipulators,'* Int. Journal of Robotics Research, Vol. 1, No. 1, Spring, 1982, pp. 65-78.

[Hager90] G. D. Hager, *'Task Directed Sensor Fusion and Planning,'* Kluwer-Academic Publishers, Boston/Dordrecht/London, 1990.

[HesSob84] D. Hestenes, G. Sobczyk, *'Clifford Algebra to Geometric Calculus,'* D. Reidel Publishing Company, Dordrecht, Holland, 1984.

[Hesten86] D. Hestenes, *'New Foundations for Classical Mechanics,'* D. Reidel Publishing Company, Dordrecht, Holland, 1986.

[Hiller83] M. Hiller, *'Mechanische Systeme,'* Springer Verlag, Berlin/Heidelberg/New York/Tokyo, 1983.

[Holler80] J. M. Hollerbach, *'A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity,'* IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-10, No. 11, November, 1980, pp. 730-736.

[KanFäs84] T. R. Kane, H. P. Fässler, *'Dynamics of Robot Manipulators Involving Closed Loops,'* 5. CISM-IFToMM Symposium on Theory and Practice of Robots and Manipulators, Udine, June, 1984, pp. 97-106.

[KanLev80] T. R. Kane, D. A. Levinson, *'Formulation of Equations of Motion for Complex Spacecraft,'* Journal of Guidance and Control, Vol. 3, No. 2, March/April, 1980, pp. 99-112.

[KanLev85] T. R. Kane, D. A. Levinson, *'Dynamics: Theory and Applications,'* McGraw-Hill Book Company, New York/ London/Tokyo, 1985.

[Khatib87] O. Khatib, *'A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation,'* IEEE Journal of Robotics and Automation, Vol. RA-3, No. 1, February, 1987, pp. 43-53.

[Leahy89] M. B. Leahy Jr., *'Experimental Analysis of Robot Control: A Performance Standard for the Puma-560,'* IEEE Int. Symposium on Intelligent Control, Albany, NY, September, 1989, pp. 257-264.

[LeeCha88] C. S. G. Lee, P. R. Chang, *'Efficient Parallel Algorithms for Robot Forward Dynamics Computation,'* IEEE Transactions on System, Man, and Cybernetics, Vol. 18, No. 2, March/April, 1988, pp. 238-251.

[Li88] C. J. Li, *'A New Method of Dynamics for Robot Manipulators,'* IEEE Transactions on Systems, Man, and Cybernetics, Vol. 18, No. 1, January/February, 1988, pp. 105-114.

[LuWaPa80a] J. Y. S. Luh, M. W. Walker, R. P. C. Paul, *'Resolved Acceleration Control of Mechanical Manipulators,'* IEEE Transactions on Automatic Control, Vol. 25, No. 3, March, 1980, pp. 468-474.

[LuWaPa80b] J. Y. S. Luh, M. W. Walker, R. P. C. Paul, *'On-Line Computational Scheme for Mechanical Manipulators,'* Journal of Dynamic Systems, Measurement, and Control, Vol. 102, June, 1980, pp. 69-76.

[Markie73] B. R. Markiewicz, *'Analysis of the Computer Torque Drive Method and Comparison with Conventional Position Servo for a Computer-Controlled Manipulator,'* Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, TM 33-601, March, 1973.

[Moore20] E. H. Moore, *'On the Reciprocal of the General Algebraic Matrix (Abstract),'* Bulletin of the American Philosophical Society, Vol. 26, 1920, pp. 394-395.

[Moore35] E. H. Moore, *'General Analysis,'* Memoirs of the American Philosophical Society, Vol. 1, 1935, pp. 197-209.

[Penros55] R. Penrose, *'A Generalized Inverse for Matrices,'* Proceedings of the Cambridge Philosophical Society, Vol. 51, 1955, pp. 406-413.

[Penros56] R. Penrose, *'On Best Approximate Solutions of Linear Matrix Equations,'* Proceedings of the Cambridge Philosophical Society, Vol. 52, 1956, pp. 17-19.

[Peterk86] V. Peterka, *'Control of Uncertain Processes: Applied Theory and Algorithms,'* Kybernetika, ACADEMIA, Praha, Vol. 22, 1986.

[Pissan84] S. Pissanetzky, *'Sparse Matrix Technology,'* Academic Press, London, 1984.

[Saridi83] G. N. Saridis, *'Intelligent Robotic Control,'* IEEE Transactions on Automatic Control, Vol. AC-28, No. 5, May, 1983, pp. 547-557.

[Saridi84] G. N. Saridis, *'Control Performance as an Entropy: An Integrated Theory for Intelligent Machines,'* International Conference on Robotics, Atlanta, GA, March, 1984, pp. 594-599.

[Saridi85] G. N. Saridis, *'Foundations of the Theory of Intelligent Control,'* IEEE Workshop on Intelligent Control, Rensselaer Polytechnical Institute, Troy, NY, August, 1985, pp. 23-28.

[Seraji87] H. Seraji, *'An Approach to Multivariable Control of Manipulators,'* Transactions of the ASME, Vol. 109, June, 1987, pp. 146-154.

[Shanno48] C. E. Shannon, *'A Mathematical Theory of Communication,'* The Bell Systems Technical Journal, Vol. 27, July, 1948, pp. 379-656.

[Slotin85] J. E. Slotine, *'The Robust Control of Robot Manipulators,'* Int. Journal of Robotics Research, Vol. 4, No. 2, Summer, 1985, pp. 49-64.

[Sperli86] F. B. Sperling, *'Piecewise Linear Robot Control,'* Int. Symposium on Robot Manipulators: Modeling, Control, and Education, Albuquerque, NM, 1986, pp. 109-115.

[WaBuSh85] C. Wampler, K. Buffinton, J. Shu-hui, *'Formulation of Equations of Motion for Systems Subject to Constraints,'* Journal of Applied Mechanics, Vol. 52, June, 1985, pp. 465-470.

[Whitne87] D. E. Whitney, *'Historical Perspective and State of the Art in Robot Force Control,'* Int. Journal of Robotics Research, Vol. 6, No. 1, Spring, 1987, pp. 3-14.

[Witten77] J. Wittenburg, *'Dynamics of Systems of Rigid Bodies,'* B.G Teubner, Stuttgart, 1977.

[Zielke83] G. Zielke, *'Verallgemeinerte inverse Matrizen,'* Jahrbuch Überblicke Mathematik, Bibliographische Institut AG, 1983, pp. 95-116.

Curriculum Vitae

January 24, 1961	Born in Zurich.
1967-74	Primary school in Zollikerberg, Zurich.
1974-80	Attendance at the Gymnasium Rämibühl, Zurich, Federal Maturity Exam (Matura), type B.
1980-85	Studies in Mechanical Engineering at the Swiss Federal Institute of Technology, Zurich (ETH Zurich).
1986	Diploma in Mechanical Engineering from ETH Zurich, Majors: Mechanics and Control, Diploma Thesis: "Motion and Force Control of a Two Link Robot Manipulator."
1986-88	Teaching assistant and research associate at the Measurement and Control Lab, Department of Mechanical Engineering, ETH Zurich.
1988	Assigned as lecturer for one semester, during Prof. Geering's sabbatical, for teaching the course "Robot Dynamics."
1988-93	Ph.D. student at the Measurement and Control Lab, Department of Mechanical Engineering, ETH Zurich, research in control of rigid multi-body systems.