Mathematical Optimization in Robotics: Towards Automated High Speed Motion Planning*

Marc C. Steinbach, H. Georg Bock, Heidelberg,
Georgii V. Kostin, Moscow,
Richard W. Longman, New York

Summary. Industrial robots have greatly enhanced the performance of automated manufacturing processes during the last decades. International competition, however, creates an increasing demand to further improve both the accuracy of off-line programming and the resulting cycle times on production lines. To meet these objectives, model based optimization is required. We describe in detail the development of a generic dynamic robot model, specialize it to an actual industrial robot KUKA IR 761, and discuss the problem of dynamic calibration. Efficient and robust trajectory optimization algorithms are then presented which, when integrated into a CAD system, are suitable for routine application in an industrial environment. Our computational results for the KUKA IR 761 robot performing a real life transport maneuver show that considerable gains in productivity can be achieved by minimizing the cycle time.

AMS Subject Classification: 49M37, 65K10, 70B15, 70E15, 70Q05, 73C50, 90C30, 90C90.

Key words: Robot dynamics; Modeling; Calibration; Trajectory optimization; Collision avoidance; Large scale constrained optimization; Sparse SQP.

1 Introduction

Robot manipulators play an important role in modern industrial manufacturing processes; nowadays they are particularly common on automated production lines in the automobile industry. Typical jobs performed by robots range from welding, gluing, or spray-painting to transport and assembly tasks. However, perpetually increasing quality standards and international competition as well as economic reasons impose high demands on precision and reliability, and specifically on the speed of industrial robots, thus calling for sophisticated motion planning techniques.

Today the classical on-site teaching is still common practice. Relying on the knowledge and the intuition of experienced personnel, this method is useful

*Jan. 12, 1997; revised April 15, 1997. To appear in Surveys on Mathematics for Industry.

to implement accurate, collision-free trajectories in a comparatively easy way. More advanced CAD based motion planning systems offer the advantage of designing robot maneuvers off line, thus cutting down production losses during the implementation phase. However, while this works well in relatively slow assembly tasks, the resulting trajectories are less accurate in high speed gluing or transport maneuvers. The reason is that commercial CAD systems use *kinematic* models which include only the robot geometry and worst case restrictions on velocities and accelerations of individual joints and of the tool center point (TCP). Such models are inadequate for controlling the complex nonlinear dynamics of very fast maneuvers. This leads to tracking errors and hence requires time consuming manual corrections when implementing a new manufacturing process; furthermore, predicted cycle times are often exceeded.

To enable reliable off-line programming of fast maneuvers, validated *dynamic* robot models are needed which include centrifugal, gravitational and Coriolis forces, and possibly joint elasticities, friction, motor dynamics, etc.

As soon as reliable dynamic models are available, mathematical optimization algorithms can be applied to minimize the cycle time of certain maneuvers. Scientific investigations on robot trajectory optimization began already in the late sixties; among the earliest is the work of Kahn [28] and Kahn and Roth [29]. During the last decade the topic has received great interest in the academic community, and various approaches have been proposed based on different problem formulations and different types of robot models. Since a rigorous treatment of realistic problems turns out to be very hard, especially if geometric constraints are specified for collision avoidance, many approaches treat greatly simplified problems or apply heuristic optimization strategies. For a comprehensive survey of the literature (until 1990) the reader is referred to [14].

One of the most important types of trajectory optimization problems is known as the *prescribed path* problem. The TCP is required to move along a given (parameterized) curve in cartesian space, with prescribed gripper orientation in each point. It is assumed that these data define the joint positions uniquely, so that only the velocity profile along the path remains to be optimized. This reduces the problem to a one-dimensional optimal control problem which is very well understood; furthermore, very general constraints on actuator torques, joint speeds, etc. can easily be treated. A highly efficient solution algorithm tailored to the minimum time case was first proposed by Bobrow, Dubowsky and Gibson [5, 6], and further developed by Pfeiffer and Johanni [39]; in addition Johanni proposes Dynamic Programming to handle other performance criteria, and subjects the path itself to an outer optimization to find optimal trajectories [27]. The prescribed path problem is an appropriate formulation for many machining tasks such as, e.g., grinding or applying varnish.

The second major type of trajectory optimization problems is known as the *point-to-point (PTP)* problem: only the initial and final points, say A, B, are given, but the shape of the trajectory is subject to optimization. This problem formulation is appropriate, e.g., for transport maneuvers and for unloaded robot motion to start a new task at B after finishing a task at A. Our notion of the PTP problem, however, is more general. We allow restrictions that fix some degrees of freedom along the trajectory but that do not necessarily determine all joint angles uniquely. A typical example is a gluing maneuver where the adhesive emanating from a spray gun is deposited along a prescribed curve, but the TCP path and orientation may vary in certain ranges. Although the prescribed path problem is in principle included in our generalized class of PTP problems, we distinguish this case because of its very special properties.

A precise mathematical statement of the PTP problem is given in the form of a rather general trajectory optimization problem (TOP). The aim is to determine a state function $x = (x_1, x_2)$ and a control function u on time interval $[0, T]$ such that a performance criterion ϕ is minimized subject to path constraints g, multipoint boundary conditions r_i, and differential-algebraic equations (DAE) describing the robot dynamics:

$$\begin{aligned} \phi(T, x(T)) &= \min \\ \dot{x}_1(t) - f_1(t, x(t), u(t)) &= 0 \\ f_2(t, x(t), u(t)) &= 0 \\ g(t, x(t), u(t)) &\in [g_{\min}(t), g_{\max}(t)] \\ r_1(t_1, x(t_1)) + \cdots + r_k(t_k, x(t_k)) &= 0 \end{aligned} \tag{1}$$

Algebraic equations f_2 may result from the problem under consideration (if the TCP path is prescribed, for instance) or from a descriptor form model of the robot's multibody dynamics. If necessary, higher index DAE are reduced to index 1 and treated numerically by invariant projection [46]; in this case the invariants are also contained in f_2. State-of-the-art algorithms for numerical integration of multibody DAE can be found, e.g., in [48, 13, 58, 47]. In the context of robot trajectory *optimization*, however, these are of minor importance.

Previous work by some of the authors and co-workers aimed at developing physical insight in the dynamic interactions of a robot and finding out how much can be gained in PTP optimization; investigations along these lines study optimal basic maneuvers for basic robot types with two or three axes [30, 31, 52, 53]. The physical potential for optimization is given on all robots with revolute joints; these cause nonlinear dynamic interactions that can be exploited to have all motors support the one with the hardest task. This simple principle of cooperation yields considerable savings, often produced by surprisingly esthetic movements. Although the behavior of such solutions can be physically explained, however, it is impossible to find near-optimal trajectories through intuition and experience. Instead, one has to solve the trajectory optimization problem, and hence sophisticated mathematical algorithms are required.

The direct boundary value problem (BVP) approach due to Bock has proven very successful for this purpose; its first implementation in the multiple shooting code MUSCOD [11] was used in most of the investigations mentioned in the previous paragraph. Recent algorithmic developments [50, 45] based on the direct BVP approach allow an efficient and robust treatment of large scale problems with many inequality constraints (cf. section 4); this is crucial for solving real life optimization problems in an industrial environment.

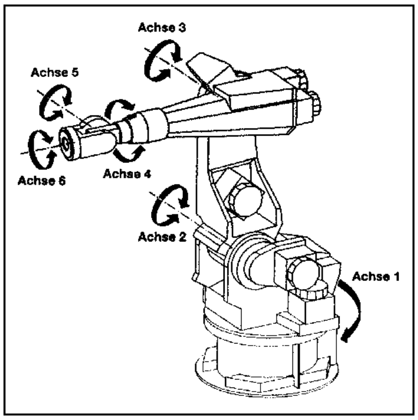

Figure 1: The 6-joint industrial robot KUKA IR 761/125/150.0

The paper is organized as follows. In section 2, dynamic robot modeling is discussed thoroughly, including a detailed presentation of multibody dynamics for a general kinematic chain and of the specific components for a model of the robot KUKA IR 761/125/150.0 shown in Fig. 1. Section 3 provides introductory information and some references on the issue of model calibration. The direct boundary value problem approach is presented in section 4 as the basic means to discretize constrained trajectory optimization problems, and recent algorithmic developments that allow an efficient treatment by SQP methods are described. In section 5, we give numerical optimization results and perform a sensitivity analysis for the KUKA IR 761 robot executing a real life transport maneuver. Finally, we offer an assessment of the practicability of the approach in section 6.

2 Dynamic modeling

The dynamic robot model is certainly a central constituent of any advanced off-line motion planning system. A good model has to satisfy two conflicting objectives. It must include enough detail to represent the real behavior of the robot with sufficient accuracy, and it should permit an efficient, stable evaluation not only of the model equations but also of their derivatives that are needed in optimization. However, the necessary degree of detail may depend on the actual application and on the required accuracy. Therefore we suggest a modular, hierarchical model structure which can be adapted to specific requirements by switching individual components on or off. In the following we develop such a generic robot model. For each component we discuss important aspects concerning model accuracy and the optimization context, and present the specific form for the robot KUKA IR 761. In most cases the practical model is assessed in the framework of more general physical considerations to justify simplifications. For an extensive presentation of general modeling issues in optimization see [54].

2.1 General modeling assumptions

In this paper we consider industrial robots with electric drives. The robot links are assumed to be rigid bodies connected by revolute or prismatic joints with a single degree of freedom each, forming a multibody system with the topological structure of a kinematic chain. A tool or load may be mounted on the last link. The robot is actuated by servo motors through cycloidal or harmonic drives with large gear ratios (~ 100).

2.2 Multibody kinematics

The multibody model plays the role of a skeleton in every robot model; it comprises the global mechanical coupling of the whole system, and requires by far the largest effort in the numerical evaluation of robot dynamics. We begin with the kinematic part of the multibody model.

2.2.1 Kinematic chain

We consider the fixed robot base as link 0; the remaining links are numbered 1 through N from base to tip where link k is connected to link $k-1$ via joint k. On each link we choose a reference point O_k with inertial coordinates $r_k \in \mathbb{R}^3$ and a frame based at O_k with inertial coordinates $R_k \in SO(3)$. The fixed base frame $(R_0, r_0) = (I, 0)$ will be our global reference frame. Relative orientation and position (B_k, l_k) of frame k with respect to frame $k-1$ are given by

$$R_k = R_{k-1} B_k, \qquad r_k = R_{k-1} l_k + r_{k-1}.$$

A combined representation of the rotational and translational components for link frames and joint transformations is achieved by 4×4 *homogeneous matrices*

$$A_k := \begin{pmatrix} R_k & r_k \\ 0 & 1 \end{pmatrix}, \qquad T_k := \begin{pmatrix} B_k & l_k \\ 0 & 1 \end{pmatrix}.$$

The relative orientation of adjacent links is now written $A_k = A_{k-1} T_k$, which by recursion yields the simple matrix product representation $A_k = T_1 \cdots T_k$.

On a moving robot, the joint transformations $T_k(\theta_k)$ and therefore the frames $A_k(\theta_1, \dots, \theta_k)$ depend on the joint variables θ_k and hence on time.

2.2.2 Denavit-Hartenberg representation

The Denavit-Hartenberg representation [17] is commonly used in industry to relate a transformation matrix T_k to its scalar joint variable and three more scalar parameters describing the joint geometry. It requires the following convention for placement of the link frames:

- The X axes of all frames are aligned in the same direction.
- Revolute joints rotate about their Z axes.
- Prismatic joints travel along their Z axes.

joint	θ [deg]	α [deg]	a [m]	d [m]
1	0	90	-0.99	0.25
2	-90	0	0	1.15
3	0	-90	0	0
4	0	90	-1.15	0
5	0	90	0	0
6	0	0	0	0

Table 1: Nominal Denavit-Hartenberg parameters of KUKA IR 761

With these conventions we can describe the general joint transformation as composition of four elementary transformations:

- Rotate an angle θ_k about the Z_{k-1} axis.
- Translate a distance d_k along the Z_{k-1} axis.
- Translate a distance a_k along the X_{k-1} axis.
- Rotate an angle α_k about the X_k axis.

The resulting homogeneous transformation matrix is

$$\begin{aligned} T_k &= \mathrm{Rot}(Z,\theta_k)\,\mathrm{Trans}(Z,d_k)\,\mathrm{Trans}(X,a_k)\,\mathrm{Rot}(X,\alpha_k) \\ &= \begin{pmatrix} \cos\theta_k & -\sin\theta_k\cos\alpha_k & \sin\theta_k\sin\alpha_k & a_k\cos\theta_k \\ \sin\theta_k & \cos\theta_k\cos\alpha_k & -\cos\theta_k\sin\alpha_k & a_k\sin\theta_k \\ 0 & \sin\alpha_k & \cos\alpha_k & d_k \\ 0 & 0 & 0 & 1 \end{pmatrix}. \end{aligned}$$

Except for special robot arm constructions, the joint variable is θ_k in revolute joints and d_k in prismatic joints; the three remaining parameters are constant. Without loss of generality, we will only address robots with revolute joints in the following, such as the KUKA IR 761. Its Denavit-Hartenberg parameters (with all joints in home position) are given in Table 1.

2.3 Multibody dynamics

In multibody *kinematics*, the representation of joint transformations by homogeneous Denavit-Hartenberg matrices is not only mathematically elegant but also computationally efficient. In multibody *dynamics* the situation is more difficult: we need first and second time derivatives to represent velocities and accelerations, but matrix derivatives and multiplications, though mathematically elegant, are computationally inefficient. It is common practice in rigid body mechanics, however, to use vectors for both linear and angular velocities and accelerations; this turns out to be a convenient and efficient formulation for our purpose. In the following we give a brief description of the precise mathematical relation between orientation matrix derivatives and angular velocities.

The linear velocity vectors are simply derivatives of the positions, but the angular velocity vectors are not derivatives of any meaningful physical quantity. Abstractly, the set of orientation matrices $SO(3) \subset \mathbb{R}^{3\times 3}$ is a compact 3-dimensional $\mathcal{C}^\infty$-submanifold which has no global 3-parameter representation without singularities.

Consider a point p fixed on a moving body. Its inertial position and velocity are $p_0(t) = r(t) + R(t)p$ and $\dot{p}_0(t) = \dot{r}(t) + \dot{R}(t)p$, respectively. The rotation part can be written

$$\dot{R}p = \omega_0 \times (Rp) = (R\omega) \times (Rp) = R(\omega \times p),$$

where ω and ω_0 are the angular velocities in the body frame and inertial frame, respectively. On the other hand, we have $\omega \times p = \tilde{\omega} p$ where

$$\tilde{\omega} := \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} = -\tilde{\omega}^* \in A(3)$$

is an antisymmetric matrix; the corresponding inertial angular velocity matrix is $\tilde{\omega}_0 = R\tilde{\omega}R^*$. (Asterisk superscripts denote transposition throughout the paper.)

The above relations define a sequence of canonical linear isomorphisms between $\mathbb{R}^3$ and the tangent space of $SO(3)$ at R (the matrix velocity space),

$$\begin{array}{ccccccc} T_R SO(3) & = & RA(3) & \cong & A(3) & \cong & \mathbb{R}^3 \\ \Cup & & \Cup & & \Cup & & \Cup \\ \dot{R} & = & R\tilde{\omega} & \leftrightarrow & \tilde{\omega} & \leftrightarrow & \omega \end{array}$$

2.3.1 Spatial notation

The isomorphism $A(3) \cong \mathbb{R}^3$ is now used to combine linear and angular velocities and accelerations to 6-dimensional *spatial vectors*. We follow the exposition of Jain [26] where more details can be found. All the quantities used below are *inertial* quantities unless otherwise noted. Let ω and v be the angular and linear velocity of a rigid body with respect to a point O (not necessarily on the body), and N and F be the moment and force about and at O. The spatial velocity V, spatial acceleration α, and spatial force f are defined as

$$V(O) := \begin{pmatrix} \omega \\ v \end{pmatrix}, \quad \alpha(O) := \dot{V}(O), \qquad f(O) := \begin{pmatrix} N \\ F \end{pmatrix}.$$

The central spatial object is the *rigid body transformation operator* for two points O_i and $O_j = O_i + l(i,j)$,

$$\phi(i,j) \equiv \phi(l(i,j)) := \begin{pmatrix} I & \tilde{l}(i,j) \\ 0 & I \end{pmatrix} \in \mathbb{R}^{6\times 6}.$$

Depending on the spatial offset $l(i,j)$ only, $\phi^*(i,j)$ relates the spatial velocities and accelerations at O_i and O_j according to

$$V(j) = \phi^*(i,j)V(i) = \begin{pmatrix} \omega(i) \\ v(i) + \omega(i) \times l(i,j) \end{pmatrix}, \quad \alpha(j) = \phi^*(i,j)\alpha(i), \qquad (2)$$

while $\phi(i,j)$ describes the dual relation of spatial forces,

$$f(i) = \phi(i,j)f(j) = \begin{pmatrix} N(j) + l(i,j) \times F(j) \\ F(j) \end{pmatrix}. \tag{3}$$

Finally we need the spatial inertia of a rigid body at the point O which is given (in terms of the spatial inertia at the center of mass C) by the symmetric matrix

$$M(O) := \phi(p)M(C)\phi^*(p) = \begin{pmatrix} J(O) & m\tilde{p} \\ m\tilde{p}^* & mI \end{pmatrix}, \qquad M(C) := \begin{pmatrix} J(C) & 0 \\ 0 & mI \end{pmatrix}.$$

Here $J(O) = J(C) + m\tilde{p}^*\tilde{p}$ is the body's moment of inertia about O, m is its mass, and $p = l(OC)$ is the vector from O to C.

2.3.2 Recursive Newton-Euler dynamics

Let us now return to the chain-structured robot. We have to connect the base and N robot links through N joints. Let β_k and T_k denote the scalar joint velocities and joint torques associated with joint positions θ_k. The (position-dependent) geometry of each joint is represented by a 1×6 joint matrix $H(k)$ that describes both the relation between joint torque and spatial force across the joint, and between joint velocity and relative spatial velocity $\Delta V(k)$ across the joint,

$$T_k = H(k)f(k), \tag{4}$$

$$\Delta V(k) = H^*(k)\beta_k. \tag{5}$$

(Abstractly speaking, $H^*(k)$ is a spatial tangent vector to the one-dimensional manifold of joint motion.) From base to tip, equations (2) and (5) yield outward transition equations for the link velocities and accelerations,

$$\begin{aligned} V(k) &= \phi^*(k-1,k)V(k-1) + H^*(k)\beta_k, \\ \alpha(k) &= \phi^*(k-1,k)\alpha(k-1) + H^*(k)\dot{\beta}_k + a(k), \\ a(k) &= \dot{\phi}^*(k-1,k)V(k-1) + \dot{H}^*(k)\beta_k, \end{aligned} \tag{6}$$

where $a(k)$ is the Coriolis and centrifugal spatial acceleration of O_k. Conversely, adding (3) to the equations of motion for a single rigid body (see [26]), one gets inward transition equations for the spatial forces,

$$\begin{aligned} f(k) &= \phi(k,k+1)f(k+1) + M(k)\alpha(k) + b(k), \\ b(k) &= \dot{M}(k)V(k) - \dot{\phi}(p(k))M(k)V(k) + \phi(p(k))\gamma(k), \end{aligned} \tag{7}$$

where $\gamma(k)$ is the gravitational spatial force at the center of mass of body k and $b(k)$ is the sum of gyroscopic and gravitational spatial forces at O_k. Assuming a fixed robot base and contact-free motion, we have $V(0) = 0$, $\alpha(0) = 0$ and $f(N+1) = 0$, and (6,7) together with (4) give a complete recursive formulation of the Newton-Euler dynamics for the multibody chain.

In the sequel we are only interested in the relation of joint torques T_k and accelerations $\dot{\beta}_k$, so we concentrate on the essential part of the Newton-Euler recursion which is given here in algorithmic form.

$$
\begin{array}{rl}
 & \text{for } k = 1(1)N \\
\alpha(0) = 0 & \alpha(k) = \phi^*(k-1,k)\alpha(k-1) + H^*(k)\dot{\beta}_k + a(k) \\
 & \text{for } k = N(-1)1 \\
f(N+1) = 0 & f(k) = \phi(k,k+1)f(k+1) + M(k)\alpha(k) + b(k) \\
 & T_k = H(k)f(k)
\end{array}
\tag{8}
$$

For the model extensions discussed below we need two other forms of these equations. A global reformulation is obtained if we combine the joint quantities to vectors $\theta, \beta, \dot{\beta}, T \in \mathbb{R}^N$ and $V, \alpha, f, a, b \in \mathbb{R}^{6N}$ (called "stacked notation" in [26]). Including the definitions of V, a, b, this yields

$$
\begin{array}{ll}
V = \mathcal{E}_\phi^* V + H^*\beta, & \alpha = \mathcal{E}_\phi^*\alpha + H^*\dot{\beta} + a, \\
a = \dot{\mathcal{E}}_\phi^* V + \dot{H}^*\beta, & f = \mathcal{E}_\phi f + M\alpha + b, \\
b = \dot{M}V - \dot{\mathcal{E}}_\phi(p)MV + \mathcal{E}_\phi(p)\gamma, & T = Hf,
\end{array}
$$

where $H := \mathrm{diag}\{H(k)\}$, $M := \mathrm{diag}\{M(k)\}$, and $\mathcal{E}_\phi(p) := \mathrm{diag}\{\phi(p(k))\}$. The recursive structure of (8) is now contained in the global transformation operator

$$
\mathcal{E}_\phi := \begin{pmatrix}
0 & \phi(1,2) & & & \\
 & 0 & \phi(2,3) & & \\
 & & \ddots & \ddots & \\
 & & & 0 & \phi(N-1,N) \\
 & & & & 0
\end{pmatrix}.
$$

Using the matrix $\phi := (I - \mathcal{E}_\phi)^{-1}$, we arrive at the *global spatial* representation of robot dynamics,

$$
\begin{aligned}
\alpha &= \phi^*(H^*\dot{\beta} + a), \\
f &= \phi(M\alpha + b), \\
T &= Hf.
\end{aligned}
\tag{9}
$$

Upon substitution of α into the f equation,

$$
f = \phi M \phi^* H^* \dot{\beta} + \phi M \phi^* a + \phi b,
$$

the joint forces T are finally obtained as

$$
T = \mathcal{M}\dot{\beta} + Q \tag{10}
$$

with

$$
\mathcal{M} = H\phi M \phi^* H^* = \mathcal{M}^*, \qquad Q = H\phi(M\phi^* a + b).
$$

This is a condensed representation of the robot dynamics in $\mathbb{R}^N$, the *state space* representation. It gives a direct description of the well-known linear relation of joint torques T and joint accelerations $\dot{\beta}$, where the positive definite inertia matrix $\mathcal{M}$ and generalized forces Q depend on the robot's dynamic state (θ, β).

2.3.3 Inverse dynamics

The problem of *inverse dynamics* is the calculation of joint torques T_k for given joint accelerations $\dot{\beta}_k$ (and given positions and velocities θ_k, β_k). This can be achieved either by the recursion (8) or through the state space equation (10).

In numerical computations we actually use the recursive formulation since it is more efficient. For the same reason we represent all physical quantities in link coordinates (denoted by an index subscript) rather than inertial coordinates (denoted by an index argument). This simplifies the calculation of a_k, b_k, H_k and makes M_k constant; only the transformation operator $\phi(k-1,k)$ is replaced by a slightly more complicated form involving the relative link orientation,

$$\phi_k := \begin{pmatrix} B_k & \tilde{l}_k B_k \\ 0 & B_k \end{pmatrix} = \begin{pmatrix} I & \tilde{l}_k \\ 0 & I \end{pmatrix} \begin{pmatrix} B_k & \\ & B_k \end{pmatrix}.$$

The recursive Newton-Euler equations (8) and consequently (9) and (10) remain formally unchanged upon substituting these quantities.

For a thourough investigation of recursive dynamics algorithms and a detailed discussion of efficiency considerations we refer the reader to [26] and to the book by Featherstone [20].

2.3.4 Forward dynamics

The problem of *forward dynamics* is the calculation of joint accelerations $\dot{\beta}_k$ for given joint torques T_k (and given positions and velocities θ_k, β_k). This task occurs in the integration of the robot's equations of motion and thus in every nonlinear iteration during optimization. It is more involved than inverse dynamics, requiring the solution of (10) or any equivalent system of linear equations. We choose a reordering of equations (8) that contains explicitly all the spatial quantities involved in robot dynamics:

$$\left(\begin{array}{cccccccc|cccc}
0 & & & & & & & & H_1 & & & \\
& M_1 & 0 & & & & & & -I & \phi_2 & & \\
& 0 & 0 & & & & & & & H_2 & & \\
& & & \ddots & & & & & & -I & \ddots & \\
& & & & M_{N-1} & 0 & & & & & & \phi_N \\
& & & & 0 & 0 & & & & & & H_N \\
& & & & & & & M_N & & & & -I \\
\hline
H_1^* & -I & & & & & & & & & & \\
& \phi_2^* & H_2^* & -I & & & & & & & & \\
& & & \ddots & & & & & & & & \\
& & & & \phi_N^* & & H_N^* & -I & & & &
\end{array}\right)
\left(\begin{array}{c} \dot{\beta}_1 \\ \alpha_1 \\ \dot{\beta}_2 \\ \vdots \\ \alpha_{N-1} \\ \dot{\beta}_N \\ \alpha_N \\ \hline f_1 \\ f_2 \\ \vdots \\ f_N \end{array}\right)
=
\left(\begin{array}{c} T_1 \\ -b_1 \\ T_2 \\ \vdots \\ -b_{N-1} \\ T_N \\ -b_N \\ \hline -a_1 \\ -a_2 \\ \vdots \\ -a_N \end{array}\right)
\tag{11}$$

This form reflects the full recursive structure as well as the symmetric indefinite structure caused by the dual behavior of accelerations and forces. Furthermore,

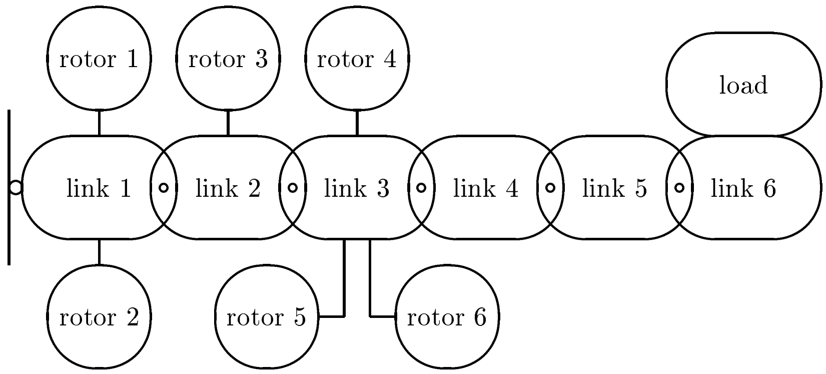

Figure 2: Tree topology of KUKA IR 761 with rotors

it does not imply *how* the system is solved, and it leaves open whether the robot is modeled in ODE form $\dot{\beta} = f(\theta, \beta; T)$ or in index 1 descriptor form with algebraic variables f_k and additional differential variables V_k. Below we use the ODE form, but for certain applications (such as cooperating robots with kinematic loops, for instance) a DAE model may be preferable.

According to [20] the most efficient algorithm to solve (11) for up to eight bodies is the *composite rigid body method* [60] with complexity $O(N^3)$; for more than eight bodies the $O(N)$ *articulated body method* (cf. [56, 2, 19]) is faster. However, the latter turns out to be a special case of our *recursive multistage KKT algorithm* [50] which solves similarly structured systems representing the Karush-Kuhn-Tucker (KKT) optimality conditions in the trajectory optimization context, cf. section 4.2. For this purpose the solver MSKKT is at hand anyway, so we apply it to (11) even though our robot has only six links.

2.4 Rotor inertia

Commercial industrial robots often have cycloidal or harmonic drives with typical gear ratios between 50 and 200. Hence, the motors may perform several thousand rotations per minute which makes rotor inertia a significant factor in robot dynamics. In the following we consider the rotors as additional rigid bodies joined with the links on which they are located. Slowly rotating parts of the gear train (lying behind the drives) are treated as if fixed on their respective links.

2.4.1 Extended multibody system

By j_k we denote the index of the link on which motor k is located. In case of the robot KUKA IR 761 all motors are placed on one of the first three links,

$$j_1 = j_2 = 1, \quad j_3 = 2, \quad j_4 = j_5 = j_6 = 3.$$

The resulting tree-structured system is depicted in Fig. 2.

In the extended multibody system the rotor and rotor shaft of motor k are counted as body and joint $N+k$, respectively. By $\theta = (\theta^0, \theta^1) \in \mathbb{R}^{2N}$ we denote now the augmented variable vector consisting of joint variables $\theta^0 \in \mathbb{R}^N$ (previously θ) and new rotor variables $\theta^1 \in \mathbb{R}^N$. The remaining state space vectors $\beta, \dot{\beta}, T \in \mathbb{R}^{2N}$, global spatial vectors $V, \alpha, f, a, b \in \mathbb{R}^{12N}$ and block-diagonal matrices $M \in \mathbb{R}^{12N\times 12N}$, $H \in \mathbb{R}^{2N\times 12N}$ are partitioned accordingly. Finally we redefine matrices

$$\mathcal{E}_\phi := \begin{pmatrix} \mathcal{E}_\phi^0 & \mathcal{E}_\phi^1 \\ 0 & 0 \end{pmatrix}, \qquad \phi := (I - \mathcal{E}_\phi)^{-1} = \begin{pmatrix} \phi^0 & \phi^0 \mathcal{E}_\phi^1 \\ & I \end{pmatrix},$$

where the link-to-link transformation operator $\mathcal{E}_\phi^0$ (previously $\mathcal{E}_\phi$) contains the chain structure, and the rotor-to-link operator $\mathcal{E}_\phi^1$ represents the coupling of rotors to their respective parent links. In case of the KUKA IR 761, the latter operator is

$$\mathcal{E}_\phi^1 = \begin{pmatrix} \phi(1,7) & \phi(1,8) & 0 & 0 & 0 & 0 \\ & 0 & \phi(2,9) & 0 & 0 & 0 \\ & & 0 & \phi(3,10) & \phi(3,11) & \phi(3,12) \\ & & & 0 & 0 & 0 \\ & & & & 0 & 0 \\ & & & & & 0 \end{pmatrix}.$$

With these quantities, the global spatial formulation (9) and the condensed state space formulation (10) of the dynamic equations remain valid for the augmented system.

For the further analysis we split the state space equation (10) into link and rotor components,

$$\begin{pmatrix} T^0 \\ T^1 \end{pmatrix} = \begin{pmatrix} \mathcal{M}^0 & \mathcal{M}^{1*} \\ \mathcal{M}^1 & J_r \end{pmatrix} \begin{pmatrix} \dot{\beta}^0 \\ \dot{\beta}^1 \end{pmatrix} + \begin{pmatrix} Q^0 \\ Q^1 \end{pmatrix}, \tag{12}$$

where $J_r = H^1 M^1 H^{1*} \in \mathbb{R}^{6\times 6}$ is the diagonal matrix of rotor inertias, and $\mathcal{M}^0, Q^0$ are rather lengthy expressions. However, since the rotors are symmetric with respect to their axes, the combined inertia $M^\oplus$ of links and rotors and the gravitational force $\gamma^\oplus$ at their combined center of mass $p^\oplus$ are independent of the rotor motion. This leads to simplifications in $\mathcal{M}^0$ and Q^0, yielding

$$\begin{aligned} \mathcal{M}^0 &= H^0 \phi^0 M^\oplus \phi^{0*} H^{0*}, & Q^0 &= Q^\oplus + H^0 \phi^0 \mathcal{E}_\phi^1 (M^1 H^{1*}) \dot{\beta}^1, \\ \mathcal{M}^1 &= H^1 M^1 \mathcal{E}_\phi^{1*} \phi^{0*} H^{0*}, & Q^1 &= H^1 (M^1 \mathcal{E}_\phi^{1*} \phi^{0*} a^0 + M^1 a^1 + b^1), \end{aligned}$$

with

$$\begin{aligned} M^\oplus &= M^0 + \mathcal{E}_\phi^1 M^1 \mathcal{E}_\phi^{1*}, \\ Q^\oplus &= H^0 \phi^0 (M^\oplus \phi^{0*} a^0 + b^\oplus), \\ b^\oplus &= \dot{M}^\oplus V^0 - \dot{\mathcal{E}}_\phi(p^\oplus) M^\oplus V^0 + \mathcal{E}_\phi(p^\oplus) \gamma^\oplus. \end{aligned}$$

Equation (12) for the extended multibody system does not include the coupling of rotors and joints through the gear trains; thus it provides a common framework for different types of connection. In the following we consider the idealized case of a rigid coupling; elastic joints are discussed in section 2.5 below.

2.4.2 Inelastic joints

When elasticity in the gear trains can be neglected, the coupling between joint and rotor coordinates θ^0, θ^1 and between joint and rotor torques T^0, T^1 is given by

$$\theta^1 = \mathcal{N}^* \theta^0, \qquad T^0 = \mathcal{N}(\mu - T^1) \tag{13}$$

where μ is the vector of electromagnetic torques and $\mathcal{N}$ is the gear ratio matrix. The matrix $\mathcal{N}$ is often non-diagonal since the robot's hand axes usually interact through a gear box in the wrist. In case of the robot KUKA IR 761 the gear ratio matrix has the form

$$\mathcal{N} = \begin{pmatrix} n_1 & & & & & \\ & n_2 & & & & \\ & & n_3 & & & \\ & & & n_4 & n_5^4 & n_6^4 \\ & & & & n_5 & n_6^5 \\ & & & & & n_6 \end{pmatrix}. \tag{14}$$

Using (13), all the torques T^0, T^1 and rotor variables θ^1 can be eliminated from (12), yielding

$$\mathcal{N}\mu = [\mathcal{M}^0 + (\mathcal{N}\mathcal{M}^1 + \mathcal{M}^{1*}\mathcal{N}^*) + \mathcal{N} J_r \mathcal{N}^*]\dot{\beta}^0 + (Q^0 + \mathcal{N} Q^1).$$

Thus, we have again a state space equation in the joint variables alone, but with modified inertia matrix and force vector, and with joint torques replaced by the (scaled) electromagnetic rotor torques.

2.4.3 Approximation

In the typical case of large gear ratios $n_k \gg 1$ and comparatively small rotor inertias $\|M^1\| \ll \|M^0\|$, we can neglect the off-diagonal inertia matrix $\mathcal{M}^1$ and gyroscopic and Coriolis forces of the relative rotor motion. Thus, we obtain the simplified robot model

$$\mathcal{N}\mu = (\mathcal{M}^0 + \mathcal{N} J_r \mathcal{N}^*)\dot{\beta}^0 + Q^{\oplus} \tag{15}$$

which is similar to (10) but includes the rotor inertia $\mathcal{N} J_r \mathcal{N}^*$ as seen by the motors, and the spatial inertia of the rotors as parts of their respective host links. The equivalent modification of (9) is

$$\begin{aligned} \alpha^0 &= \phi^{0*}(H^{0*}\dot{\beta}^0 + a^0), \\ f^0 &= \phi^0(M^{\oplus}\alpha^0 + b^{\oplus}), \\ \mathcal{N}\mu &= H^0 f^0 + \mathcal{N} J_r \mathcal{N}^* \dot{\beta}^0. \end{aligned}$$

The extra term $\mathcal{N} J_r \mathcal{N}^*$ does not pose any difficulties in the recursive calculation of inverse dynamics. In forward dynamics the situation is again more involved. Diagonal entries n_k in $\mathcal{N}$ create a nonzero mass matrix entry $n_k^2 J_{rk}$ immediately preceding M_k on the diagonal in (11), but off-diagonal entries in $\mathcal{N}$ create nonzero entries outside the diagonal blocks, thus destroying the recursive structure. One can always eliminate the extra coupling by dummy variables, and, fortunately, only three scalar variables are needed for typical industrial robots having a gear ratio matrix like (14). This restores the structure, so we can still apply a recursive algorithm in forward dynamics.

2.5 Joint elasticity

Since harmonic and cycloidal drives exhibit vibration, compliant behavior in the gear trains should also be considered in the overall dynamic model of the robot.

2.5.1 Elasticity model

The coupled equations of motion for a robot with flexible joints are given as

$$\begin{aligned} \mathcal{M}^0 \dot{\beta}^0 + \mathcal{M}^{1*} \dot{\beta}^1 &= -\mathcal{N} T_c(\Delta\theta) - Q^0, \\ \mathcal{M}^1 \dot{\beta}^0 + J_r \dot{\beta}^1 &= \mu + T_c(\Delta\theta) - Q^1, \end{aligned} \tag{16}$$

where the vector of drive stiffness torques T_c depends on the torsion angles $\Delta\theta := \mathcal{N}^* \theta^0 - \theta^1$. Using the approximate model of rotor inertia, (16) simplifies to

$$\begin{aligned} \mathcal{M}^0 \dot{\beta}^0 &= -\mathcal{N} T_c(\Delta\theta) - Q^{\oplus}, \\ J_r \dot{\beta}^1 &= \mu + T_c(\Delta\theta). \end{aligned} \tag{17}$$

These equations or (16) have to be complemented by an elasticity model $T_c(\Delta\theta)$. Linear models for flexible robot joints are very well studied in the literature (for references see [16], e.g.), but real harmonic and cycloidal drives exhibit nonlinear compliant behavior caused by complex deformations of the gear teeth. Some authors approximate experimental measurements of the stiffness curve by a cubic function

$$T_{c,k}(\Delta\theta_k) = K_{1k} \Delta\theta_k + K_{2k} \Delta\theta_k^3$$

with constant coefficients K_{ik} [57, 23]. Alternatively, as in [15], approximations by *piece-wise linear* funtions are typically given in the technical specifications provided by drive manufacturers. We wish to use the technical data, but prefer smooth functions to avoid unnecessary monitoring of discontinuities. Therefore we fit a simple analytical function to the two or three linear segments (or to the measured data if available). As an example, consider segment i of a piece-wise linear model T_c^0 approximating the compliance function $T_{c,k}$ of drive k,

$$T_{c,i}^0(\Delta\theta) = k_i(\Delta\theta - \alpha_i) + T_i, \quad T_1 = 0, \quad \alpha_1 = 0, \quad i = 1, 2, 3. \tag{18}$$

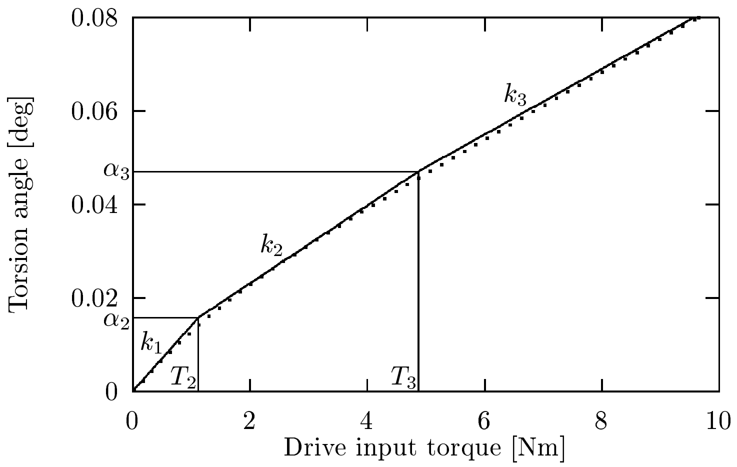

Figure 3: Torsion stiffness of the harmonic drive HDUR-50

Here the slopes k_i and segment-delimiting torques T_i are experimental data, and the angles α_i are obtained as

$$\alpha_2 = T_2/k_1, \quad \alpha_3 = \alpha_2 + (T_3 - T_2)/k_2.$$

Our smooth model function used in this example is given by

$$T_c^1(\Delta\theta) = c_1 \Delta\theta + \text{sign}(\Delta\theta) c_2 [(1 + c_3 |\Delta\theta|^p)^{1/p} - 1], \tag{19}$$

where $p > n + 1$ yields a smoothness level $T_c^1 \in \mathcal{C}^n$ $(n \in \mathbb{N})$, and

$$c_1 = k_1, \quad c_2 = k_3 \alpha_3 - T_3, \quad c_3 = [(k_3 - k_1)/c_2]^p.$$

For $\Delta\theta \ll \alpha_2$ or $\Delta\theta \gg \alpha_3$ the smooth function T_c^1 approaches the piece-wise linear model asymptotically with $|T_c^1| \geq |T_c^0|$. The difference $|T_c^1 - T_c^0|$ decreases (for all angles) if p is increased. Fig. 3 shows the piece-wise linear approximation (solid line) and our smooth fit with a small parameter $p = 3$ (dotted line) for one of the harmonic drives on the KUKA IR 761. Obviously the two compliance models are in good accordance. Note that a numerical integration with order n requires $p > n + 1$; otherwise jumps in higher derivatives will destroy the order.

2.5.2 Discussion

Optimal trajectories for robots with flexible joints have been numerically investigated in [37] based on model (17). The results show that the minimum time control problem has usually multiple local solutions, which are generally slower than the corresponding optimal maneuver in the rigid joints case. Two major classes of local solutions can be distinguished. One class is characterized by frequently switching control torques, especially if the control discretization is comparatively fine. This behavior tends to *damp* oscillations. Maneuvers in the second class are usually a bit slower; they have only a few control impulses that tend to *excite* oscillations.

Neither behavior is acceptable in practice, and in addition, typical first mode frequencies of 5–15 Hz make the trajectories very sensitive to unavoidable model inaccuracies. Besides, the different time scales involved in oscillatory disturbances versus global joint motion introduce artificial stiffness into the dynamic equations, thus increasing the effort for numerical integration considerably.

For these reasons we do not include flexibility in the robot model that is used in the optimization runs below. Instead, we use the flexible model to check feasibility of the optimal trajectories obtained from the rigid model. Alternatively, one might impose more restrictive smoothness conditions on the control torques to reduce the high frequency content of optimal solutions for the flexible model, or one might penalize fast torque changes by adding appropriate terms to the minimum time objective, thus creating a similar smoothing effect. However, it is not yet clear how joint flexibility should be treated in robot trajectory optimization, and additional investigations are being conducted.

2.6 Motor dynamics

Industrial robots are typically powered by electric actuators. In comparison to hydraulic or pneumatic actuators, electric motors are compact, easy to control, and do not require additional complex equipment. The adequate dynamic model for an electric actuator is a linear first order system, see, e.g., [32, 33].

2.6.1 Model of servo motor

Dropping index k, the voltage balance and the relation between electromagnetic torque μ and current I in the armature circuit of servo motor k are given by

$$U = L\dot{I} + RI + \Phi\beta, \tag{20}$$

$$\mu = \nu I, \tag{21}$$

where L and R are the inductance and resistance of the armature winding, Φ is its magnetic flux, U the voltage, $\beta \equiv \beta_{N+k}$ the rotor speed, and ν is a constant factor. For the robot motors these equations have to be combined with (15) for a rigid drive model or with (16) or (17) for an elastic model. In both cases the voltage U replaces the torque μ as control input.

If the electromagnetic response time $\tau_e = L/R$ of the servo motor is sufficiently small we can model the limit case $L = 0$ in (20). The order of the drive model is then reduced and the expression for the electromagnetic torque has the form

$$\mu = \nu I = \nu(U - \Phi\beta)/R. \tag{22}$$

If the voltage limit is large enough, then it does not restrict the permissible rotor torques and velocities (see next section), and equation (22) enables us

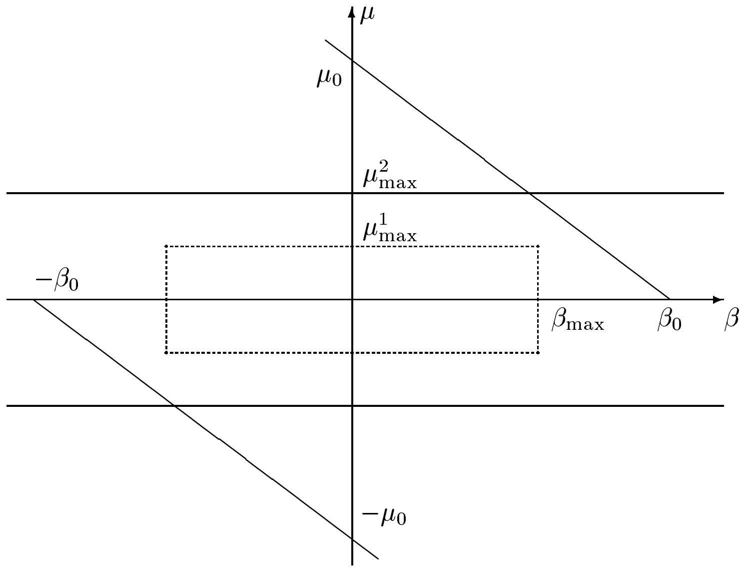

Figure 4: Torque and speed limits for an electric drive

to consider μ as the control input again. In what follows we will assume that $L_k = 0$ for all motors of the robot KUKA IR 761.

Note that the results in [35] indicate that motor dynamics should be considered as part of the robot's feedback controller anyway. Thus we can treat it in a postprocessing rather than in the optimization problem, cf. section 6.

2.6.2 Torque and speed constraints

The rotor torque and velocity of an electric drive are limited by physical and technical characteristics of the motor and the reduction gear. Some of the limits ensure a certain lifetime of the drive under normal operation, others prevent the motor and gear from destruction. Exceeding the latter ones can cause breaking of gear teeth or shafts, for instance, or burnout of the armature winding. In [38] the drive constraints are considered in detail, including the bias of certain limits due to friction. Here we give an overview of the relevant limits, but friction is treated separately in section 2.7 below.

The admissible angular velocities and rotor torques for a drive with servo motor are shown in Fig. 4. Both the motor and the attached gear have a speed limit $|\beta| \le \beta_{\max}$; their minimum corresponds to the dotted vertical lines. For the rotor torque there are actually three constraints. The smallest of them restricts a certain "average" torque during a working cycle,

$$\left[\frac{1}{T}\int_0^T |\mu(t)|^p |\beta(t)|\, dt \Big/ \beta_{\text{av}}\right]^{1/p} \le \mu^0_{\max},$$

where $p \approx 3$ is an empirical parameter depending on the gear construction, and

$$\beta_{\text{av}} := \frac{1}{T} \int_0^T |\beta(t)|\, dt$$

is the average rotor speed. This limit is needed to ensure the specified lifetime of the gear. For the same reason the absolute torque in acceleration and deceleration phases has a limit $\mu_{\max}^1$, the *repeated peak torque limit*, which in addition restricts the current in the armature winding to prevent it from burning out. The admissible area $|\mu| \leq \mu_{\max}^1$ lies between the dotted horizontal lines in Fig. 4. Finally, the solid horizontal lines represent limits $\mu_{\max}^2$ that prevent breaking of the gear. The breaking limit $\mu_{\max}^2$, the *momentary peak torque limit*, is typically two to three times as large as $\mu_{\max}^1$; it may only be reached for a very short time, as in emergency stops (or collisions), and only a few times during the gear's life.

The sloping lines correspond to box constraints on the motor voltage, which according to (22) result in mixed speed and torque restrictions. In case of the KUKA IR 761 all drives are designed such that the voltage constraints do not intersect the area enclosed by the dotted lines, so we use the limits $|\beta| \leq \beta_{\max}$ and $|\mu| \leq \mu_{\max}^1$ in the optimization below.

2.7 Friction

Cycloidal and harmonic drives with high gear ratios cause significant reductions of the effective joint torques due to friction. Experimental measurements indicate that friction in these drives has three components: velocity-independent friction, velocity-dependent friction, and friction from resonant vibration. If we neglect the influence on the gear transmission from resonant friction [15], the total drive friction torque T_f according to the model in [1] is given by

$$T_f(\beta, T) = \begin{cases} \text{sign}(\beta)\kappa(\beta) & \text{if } \beta \neq 0, \\ \text{sign}(T)\min(\kappa_0, |T|) & \text{if } \beta = 0, \end{cases} \tag{23}$$

where β is the rotor velocity, T is the rotor torque, $\kappa(\beta)$ models the velocity dependence of friction, and $\kappa_0 = \kappa(0)$.

The complicated nature of friction introduces state-dependent discontinuities in the dynamic equations, requiring proper numerical treatment by switching functions [18, 58, 61]. Furthermore, dry friction may cause rank deficiencies in the optimization problem. Numerical investigations of optimal robot trajectories under the influence of friction have been performed in [22], where a Coulomb friction model is used, that is, $\kappa(\beta) \equiv \kappa_0$ in (23). More generally, the velocity dependence of friction may be approximated by a cubic function [55]

$$\kappa(\beta) = \kappa_0 + \kappa_1|\beta| + \kappa_3|\beta|^3.$$

However, the creation of a comprehensive model seems rather complicated. Friction coefficients are very difficult to measure. In addition, they depend on the temperature, lubrication and wear-out of gears and hence change with time.

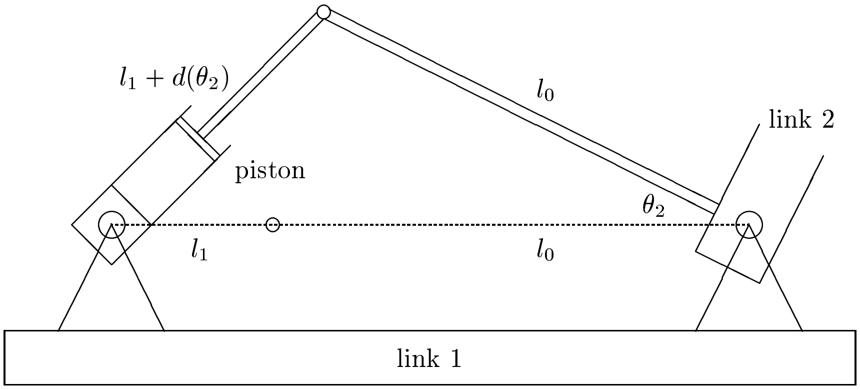

Figure 5: Geometry of pneumatic weight compensation system on KUKA IR 761

Therefore we cannot advocate the use of presently available physical friction models in trajectory optimization. Instead, we employ an empirical model that is typically used in practice. Each drive is assumed to have a certain efficiency, so that a reduction of the nominal joint torque by a constant percentage yields the effective torque

$$T_{k,\mathrm{eff}} = (1 - c_k)T_k.$$

The *loss coefficients* c_k in this equation can be regarded as safety margins for the motor torques so that the feedback controller can compensate for friction. A more precise model might take into account that friction actually *increases* the effective torque during deceleration phases. Anyway, the sensitivity analysis in section 5.4.3 shows that the optimal maneuver time in our application example is only mildly affected by a reduction of torque limits.

2.8 Pneumatic weight compensation

Special robot constructions or certain tools may require dynamic modeling of additional components that are not covered by the "generic" set described in previous sections.

In case of the robot KUKA IR 761/125/150.0 which is designed to handle heavy loads up to 125 kg, a passive pneumatic weight compensation system on axis 2 (the shoulder axis) supports the upper arm motor when the arm is inclined. This system is placed on the first link; it consists of two lever arms with a piston (see Fig. 5) from which the pressure is transmitted by oil to a gas bubble in a pressure container. Its additional torque T_p depends only on the joint angle θ_2; the functional dependence

$$T_p(\theta_2) = T_0 \frac{1}{1 - q[L(\theta_2) - (\lambda - 1)]} \frac{\sin\theta_2}{L(\theta_2)},$$

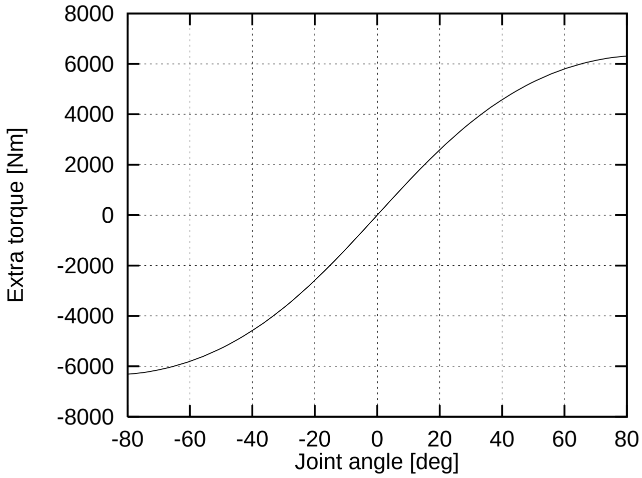

Figure 6: Pneumatic extra torque on the second axis of KUKA IR 761

is plotted in Fig. 6. Here the first factor T_0 is a constant torque, the second factor measures the change of pressure due to deviations from zero position, and the third factor is purely geometric with $L(\theta_2) = \sqrt{\lambda^2 - 2\lambda\cos\theta_2 + 1}$. The remaining parameters are constant,

$$T_0 = Sp_0(l_0 + l_1), \qquad q = Sl_0/V_0, \qquad \lambda = (l_0 + l_1)/l_0,$$

where S, p_0, V_0 denote the effective area of the piston, the adjustable minimal gas pressure, and the maximal gas volume, respectively.

3 Model calibration

Accurate off-line programming and trajectory optimization methods for robots require, of course, quantitatively correct dynamic robot models. Model calibration is the task of adapting a model so that simulation runs reproduce the real system behavior with sufficient accuracy. This process involves decisions about the structure of the model, i.e., how certain subsystems should be modeled, and, given a specific model structure, the estimation of unknown (or inaccurate) parameters from measurements. In our case the structural decisions are concerned with subsystem models for the multibody dynamics, elasticity, friction, etc. Parameters to be estimated include kinematic parameters such as link lengths and joint locations, and dynamic parameters such as link masses, inertia tensors, elastic compliances, friction coefficients, etc.

Calibration is not our main subject in this paper; it is an important research field of its own, so we will not go into too much detail here. However, we will give some general information on the type of parameter estimation problems that arise in dynamic robot calibration, and indicate how the dynamic measurements may be performed. For a collection of recent contributions to robot calibration from both science and industry the reader is referred to [3].

3.1 Parameter estimation

For the parameter estimation we compare measurements $\tilde{y}_{ij}$ of a given robot maneuver with the values $y_i(x(t_j), p)$ obtained from model-based simulation. Here t_j are the sample times and y_i are suitable functions of the joint variables such as inertial coordinates of reference points on the robot, for instance. The measurement errors $\epsilon_{ij} = y_i(x(t_j), p) - \tilde{y}_{ij}$ are assumed to be independent and normally distributed with mean value zero and standard deviation σ_{ij}. Now a Maximum Likelihood estimation is obtained by minimizing the least-squares cost function

$$\|\mu(x(t_1), \ldots, x(t_k), p)\|_2^2 := \sum_{i,j} \sigma_{ij}^{-2} [y_i(x(t_j), p) - \tilde{y}_{ij}]^2$$

subject to

$$\begin{aligned} \dot{x}_1(t) - f_1(x(t), p) &= 0 \\ f_2(x(t), p) &= 0 \\ r(x(t_1), \ldots, x(t_k), p) &= 0 \quad \text{or} \geq 0. \end{aligned}$$

To simplify notation we do not distinguish between measurement times and times at which boundary conditions are evaluated. The latter include typically (parameter-dependent) initial conditions, parameter restrictions, and terminal conditions. If some measurements before and after the maneuver are performed with much higher accuracy than the measurements obtained during robot motion, then the results of these measurements at rest should also be formulated as boundary conditions rather than least-squares conditions.

Suitable algorithms are available for the numerical solution of the (often ill-conditioned) parameter estimation problem [7, 8, 9, 41, 45]. The algorithms are implemented as multiple shooting and collocation codes PARFIT and COLFIT; an additional multiple shooting variant MULTEX takes advantage of the special sparse structure of the Jacobian in the important case of multiple experiments.

3.2 Dynamic measurements

Obtaining dynamic data of high speed robot maneuvers is difficult and costly. In addition to speeds and angles of the motor axes (which are available through internal sensors on the robot) one needs highly accurate measurements of spatial link positions and orientations in very short sample intervals. An initial calibration is needed for each individual robot when it is assembled. Furtermore, (periodic) recalibrations are required due to material ageing or after repairs. To prevent production losses, these recalibrations should be performed on the shop floor, and in the ideal case even during the running production process.

Therefore the measuring system must not directly interfere with the robot, and it should be mobile, robust, easy to use, and inexpensive.

Theodolite triangulation methods are successfully used in *kinematic* robot calibration systems (see Schröer [42]), and have already been applied in practice by KUKA. While theodolite triangulation is extremely accurate, it is also very expensive and too slow for dynamic measurements.

A promising new approach was developed by Hilsebecher and Schletz [25, 40] who combine a system of at least two digital cameras with advanced techniques for image sequence processing to determine the motion of special large area reference patterns on the robot; these motion data are then used as input for the parameter estimation. Experiments using a KUKA IR 161/15 show that the approach works in practice; it can be expected that it will meet all the criteria listed above after further development.

To further cut down calibration costs, optimal experimental design techniques as developed by Hilf [24], e.g., should be used to specify test trajectories that give the necessary dynamic data with minimal measuring effort.

4 Trajectory optimization

For real life trajectory optimization problems one needs robust numerical algorithms that can efficiently handle large numbers of variables and restrictions. In this section we describe a general approach for the discretization of trajectory optimization problems and present two new *sequential quadratic programming* (SQP) methods that handle the resulting sparse structure particularly well.

4.1 Direct BVP discretization

The direct boundary value problem approach to optimal control problem (1) combines a piece-wise parameterization of the control function on a certain grid with a piece-wise state representation via collocation or multiple shooting on a second grid. To avoid technical ballast we assume that these grids are identical, and restrict ourselves to an autonomous ODE control problem rather than the DAE control problem of section 1. On the other hand, we emphasize the presence of inequality restrictions in the trajectory optimization problem (TOP) because of their practical significance. Simple state and control bounds are treated separately from general, usually nonlinear path constraints g:

$$\phi(x(T)) = \min \tag{24}$$
$$\dot{x}(t) - f(x(t), u(t)) = 0 \tag{25}$$
$$g(x(t), u(t)) \in [g_{\min}(t), g_{\max}(t)] \tag{26}$$
$$x(t) \in [x_{\min}(t), x_{\max}(t)] \tag{27}$$
$$u(t) \in [u_{\min}(t), u_{\max}(t)] \tag{28}$$
$$r_1(x(t_1)) + \cdots + r_k(x(t_k)) = 0 \tag{29}$$

For the discretization of (24–29) we choose a grid $\Gamma\colon 0 = \tau_1 < \cdots < \tau_m = T$ with $m \geq k$ nodes as a refinement of the grid $0 = t_1 < \cdots < t_k = T$ on which boundary and interior point conditions are specified. We denote subintervals by $I_j = (\tau_j, \tau_{j+1})$ and the grid inclusion by $\sigma\colon t_i = \tau_{\sigma(i)}$.

Next, control functions are specified piece-wise via free control parameters u_j and fixed base functions v_j with local support, $u(t) = v_j(t, u_j)$ on I_j. That is, u is restricted to some finite-dimensional space of admissible controls independently on each subinterval. Control jumps are permitted at the nodes τ_j.

A state discretization by *collocation* uses polynomials p_j of degree κ_j as local representations of the trajectory x on I_j. The polynomial $p_j(t, x_j, z_j)$ is uniquely parameterized by the local initial value $x_j \equiv p_j(\tau_j, x_j, z_j)$ and by its time derivatives $z_j^i \equiv \dot{p}_j(\tau_j^i, x_j, z_j)$ at *collocation points* $\tau_j^i = \tau_j + \rho_j^i(\tau_{j+1} - \tau_j)$, where $0 \leq \rho_j^1 < \cdots < \rho_j^{\kappa_j} \leq 1$. Each polynomial p_j is required to satisfy the differential equation at all collocation points, specified by *collocation conditions*

$$c_j^i(x_j, z_j, u_j) := z_j^i - f(p_j(\tau_j^i, x_j, z_j), v_j(\tau_j^i, u_j)) = 0, \qquad i = 1(1)\kappa_j, \qquad (30)$$

while global continuity of the piece-wise trajectory representation is ensured through *connection conditions*

$$h_j(x_j, z_j, x_{j+1}) := p_j(\tau_{j+1}, x_j, z_j) - x_{j+1} = 0, \qquad j = 1(1)m-1. \qquad (31)$$

In the case of multiple shooting there are no collocation variables and conditions; the connection conditions appear in the same form as above, but z_j is replaced by u_j in (31), and $p_j(t, x_j, u_j)$ is a numerical solution of the local initial value problem $x_j \equiv p_j(\tau_j, x_j, u_j)$, $\dot{p}_j(t, x_j, u_j) = f(t, p_j(t, x_j, u_j), v_j(t, u_j))$ on I_j, obtained by some suitable integration procedure.

To formulate the discrete control problem we define vectors $y_j := (x_j, z_j, u_j)$, $y := (y_1, \ldots, y_m)$, and the objective function $F_1(y) := \phi(x_m)$. Equality and inequality constraints are collected as $c_j := (c_j^1, \ldots, c_j^{\kappa_j})$ and

$$F_2(y) := \begin{pmatrix} \{c_j(y_j)\}_{j=1}^m \\ \{h_j(y_j, x_{j+1})\}_{j=1}^{m-1} \\ r_1(x_1) + \cdots + r_m(x_m) \end{pmatrix}, \quad F_3(y) := \left(\{g_j(y_j)\}_{j=1}^m \right),$$

where path constraints $g_j(y_j) := g(x_j, v_j(\tau_j, u_j))$ are specified at the nodes only, and r_j vanishes unless $j = \sigma(i)$ for some $i \leq k$. The discrete control problem is now obtained as a large scale nonlinear optimization problem (NLP) in the general form

$$F_1(y) = \min_y \quad \text{subject to} \quad \left\{ \begin{aligned} F_2(y) &= 0 \\ F_3(y) &\in [r_l, r_u] \\ y &\in [b_l, b_u] \end{aligned} \right\}. \qquad (32)$$

Here lower and upper *ranges* r_l, r_u and *bounds* b_l, b_u represent the limits given by path constraints (26–28); in case of unrestricted components the values $\pm\infty$ are formally used.

4.2 Structure-exploiting SQP methods

To solve NLP (32) numerically we apply an SQP iteration, $y^{k+1} = y^k + \alpha^k \Delta y^k$, $\alpha^k \in (0,1]$, where each search direction Δy^k is obtained as solution of a linear-quadratic subproblem (QP):

$$\frac{1}{2}\Delta y^* H^k \Delta y + J_1^k \Delta y = \min_{\Delta y} \quad \text{subject to} \quad \left\{ \begin{array}{r} J_2^k \Delta y + F_2^k = 0 \\ J_3^k \Delta y + F_3^k \in [r_l, r_u] \\ \Delta y + y^k \in [b_l, b_u] \end{array} \right\}. \tag{33}$$

Here $F_i^k := F_i(y^k)$ and $J_i^k := F_i'(y^k)$ are current function and Jacobian values, respectively, and H^k approximates the Hessian of the Lagrangian.

From the definitions of F_1, F_2, F_3 it is apparent that the QP has a specific structure which we call *m-stage block-sparse* [50]: The (exact) Hessian H^k and Jacobians J_2^k, J_3^k are block-diagonal, except for superdiagonal blocks $-I$ and a full row of blocks in J_2^k. These off-diagonal blocks are produced by the linearly coupled connection conditions and boundary conditions h_j and r_j, respectively; the remaining component functions ϕ, c_j, g_j are all completely separated.

More specifically, one obtains block partitionings $H^k = \operatorname{diag}(H_1^k, \ldots, H_m^k)$, $J_3^k = \operatorname{diag}(Q_1^k, \ldots, Q_m^k)$, and

$$J_2^k = \begin{pmatrix} G_1^k & P & & & \\ & G_2^k & P & & \\ & & \ddots & \ddots & \\ & & & G_{m-1}^k & P \\ R_1^k & \multicolumn{3}{c}{\dots\dots\dots\dots\dots\dots} & R_m^k \end{pmatrix}.$$

As in $y_j = (x_j, z_j, u_j)$, the individual blocks are further subdivided as

$$H_j^k = \begin{pmatrix} H_j^{xx} & H_j^{xz} & H_j^{xu} \\ H_j^{zx} & H_j^{zz} & H_j^{zu} \\ H_j^{ux} & H_j^{uz} & H_j^{uu} \end{pmatrix} = (H_j^k)^*, \qquad Q_j^k = \begin{pmatrix} Q_j^x & 0 & Q_j^u \end{pmatrix},$$

and

$$G_j^k = \begin{pmatrix} C_j^x & C_j^z & C_j^u \\ G_j^x & G_j^z & G_j^u \end{pmatrix}, \qquad P = \begin{pmatrix} 0 & 0 & 0 \\ -I & 0 & 0 \end{pmatrix}, \qquad R_j^k = \begin{pmatrix} R_j^x & 0 & 0 \end{pmatrix}.$$

All C_j blocks and all derivatives with respect to z_j are absent in the multiple shooting case, whereas $G_j^x = I$ and $G_j^u = 0$ in collocation.

Due to the bound and range constraints one cannot solve QP (33) directly. Instead, either an *active set strategy* (ASS) or an *interior point method* (IPM) perform a minor iteration treating the inequalities. Both alternatives lead to a sequence of linear, symmetric indefinite equation systems of the form

$$\begin{pmatrix} H^{kl} & (J_2^{kl})^* \\ J_2^{kl} & \end{pmatrix} \begin{pmatrix} \Delta y^{kl} \\ -\Delta\lambda^{kl} \end{pmatrix} = \begin{pmatrix} J_1^{kl} + (J_2^{kl})^* \lambda^{kl} \\ F_2^{kl} \end{pmatrix},$$

each of which represents the Karush-Kuhn-Tucker (KKT) optimality conditions for a purely equality-constrained QP. Moreover, the structure of H^k, J_2^k is preserved in the modified matrices H^{kl}, J_2^{kl} for both approaches, yielding m-stage block-sparse KKT systems. (Details are given in [50].)

As we have seen, the QP (and KKT system) structure results from second-order decoupling of all component functions of F_1, F_2, F_3. While this decoupling property is natural for boundary conditions of trajectory optimization problems and for connection conditions in multiple shooting, it is an extra requirement in collocation. Here the connection conditions are often combined with collocation conditions in certain Hermite-Lobatto schemes, cf. [21, 4, 59]. Although such a formulation reduces the number of NLP variables, the resulting nonlinear coupling across stages destroys the m-stage block-sparse QP structure and, even worse, deteriorates the SQP convergence as compared to our separate formulation (30, 31), cf. [45, Sec. 2.1.3].

Once given, the multistage structure can be exploited on several levels as already described in [11]. First, it obviously permits independent computation (even in parallel) and memory-efficient storage of functions, gradients, and Hessian blocks. Next, one can efficiently approximate the Hessian by high-rank block updates. We use rank-2 BFGS type updates on every block, yielding a global rank-$2m$ update. This leads to local one-step superlinear convergence of the SQP method, with an asymptotic convergence rate that depends only mildly on the grid size. Finally, specially tailored algorithms can make use of the block structure in linear algebra calculations for QP and KKT systems solution.

In the following section we will describe an extremely efficient algorithm that was specifically developed for solving multistage optimization problems: the *recursive multistage SQP method*. This approach combines all the techniques of structure exploitation mentioned above. Its performance is demonstrated in the numerical computations below. We also outline the *partially reduced SQP method* which provides another efficient approach to structure exploitation that is suitable for trajectory optimization.

4.2.1 Recursive multistage SQP method

The recursive multistage SQP approach developed in [49, 50] is based on a general *decoupling strategy* for the numerical treatment of difficult nonlinear problems. Our principal goal in setting up the discrete problem is a *reduction of nonlinear coupling* rather than finding a compact formulation with as few variables as possible. Although the resulting discrete problem will usually be much larger, it also receives a clearer structure that can be exploited in the linearized system. Furthermore, the decoupling enlarges the domain of convergence for the nonlinear iteration, thus making it more robust—and even more efficient except for easy problems.

On the TOP level the nonlinear decoupling is achieved through the direct BVP approach as described above. A "compact" discretization might combine the direct control parameterization with *single shooting* instead, or apply the previously mentioned Hermite-Lobatto collocation scheme.

On the NLP level we introduce slacks $s = (s_l, s_u, t_l, t_u)$ to reformulate bound and range inequalities $y \in [b_l, b_u]$, $F_3(y) \in [r_l, r_u]$ as *equality constraints* and simple *nonnegativity constraints*

$$\hat{F}_3(y) - s = 0, \qquad s \geq 0,$$

where

$$\hat{F}_3(y) = \begin{pmatrix} F_3(y) - r_l \\ r_u - F_3(y) \\ y - b_l \\ b_u - y \end{pmatrix}.$$

Although this is a standard transformation in *linear* programming, the reformulation has important consequences in our nonlinear context. The symmetric, independent treatment of lower and upper bounds and ranges achieves a nonlinear decoupling in accordance with our general strategy. Furthermore, it makes the SQP method an *infeasible point* method: y^k need not satisfy the bound and range restrictions before the final iteration. Hence a suitable initial estimate y^0 is found more easily and, in particular, one can expect fast convergence from a coarse grid solution even if it violates some inequality restrictions after a grid refinement. The efficiency of the SQP method is further increased by the use of an interior point QP solver. In contrast to active set strategies, the interior point approach enables an adaptive accuracy control for the increments $(\Delta y^k, \Delta s^k)$, which saves considerable effort in early SQP iterations.

On the QP level we also introduce slack variables explicitly. Dropping the iteration index k, this yields

$$\hat{J}_3 \Delta y + \hat{F}_3 - s = 0, \qquad s \geq 0$$

instead of the bound and range restrictions. (Since s appears linearly in the NLP, we do not linearize with respect to the slacks, and the increment is obtained as $\Delta s^k = s - s^k$ from the QP solution s.) The decoupling via slack variables has the same positive effects as on the NLP level. In particular, efficient QP solution by a robust primal-dual infeasible interior point method is greatly enhanced by a *natural warm start strategy* in the SQP context: the initial estimate for the SQP increment is simply $\Delta y^0 = 0$, and QP slacks and dual variables are started with the respective NLP variables of the previous SQP iteration. Together with a precise accuracy control via duality theory this leads to rapid convergence of the interior point method as observed below.

On the KKT level, finally, we employ a highly efficient linear indefinite solver which is specifically tailored to the multistage block-sparse structure. Based on the theory of Dynamic Programming, this algorithm generates a symmetric factorization of the KKT matrix using a fixed block elimination scheme. In a backward recursion, the factorization alternates local hierarchical projections with minimizations over the remaining local degrees of freedom in each stage. This yields a true projected Hessian method in the absence of coupled multipoint boundary conditions. Including additional backward and forward recursions for the right hand side, the solver achieves optimal complexity $O(m)$ on the m-stage KKT system. Besides its efficiency the algorithm offers two important advantages compared to general sparse solvers. First, there is no need for an (expensive) structure analysis, and the fill-in is exactly known a priori. Second, in case of a rank deficiency in the KKT system, the defect is exactly located and permits a control-theoretical interpretation on all higher levels. This allows to detect modeling errors, for instance. The multistage KKT algorithm can be used within both active set and interior point QP solvers, and due to its linear complexity it is particularly suited for very fine discretizations. Finally we would like to recall from section 2.3.4 that our code MSKKT is also used to solve the 6-stage KKT system (11) occurring in the forward dynamics problem for the robot KUKA IR 761, i.e., in each evaluation of the right hand side of the robot ODE.

The whole method is implemented in the multistage trajectory optimization package MSTOP consisting of four structure-specific modules and two generic ones. The core module MSKKT supplies the multistage KKT solver and a set of utility operations for the multistage structure. MSIPM and MSSQP on the next two levels implement the nested nonlinear iterations based on generic interior point and SQP modules GENIPM and GENSQP, respectively, and MSTOP finally handles the direct BVP discretization including function and gradient evaluation. The two top levels have recently been implemented, and first computational results with the complete package are reported in section 5 below for the press connection maneuver. Details of the KKT and IPM algorithms and benchmark tests for MSKKT are described in [50], and computational results for MSIPM can be found in [51].

4.2.2 Partially reduced SQP method

The partially reduced SQP approach developed by Schulz [45] is designed for an efficient treatment of problems where *dependent* and *independent* NLP variables can be distinguished, $y = (y_d, y_i) \in \mathbb{R}^{n_d+n_i}$. That is, a certain subset $c(y_d, y_i)$ of the equality constraints F_2 has a full rank partial derivative $\partial c/\partial y_d$ so that y_d is implicitly or explicitly given as a nonlinear function of y_i. (In NLP (32), e.g., this is true for the collocation variables and conditions.) The basic idea is to view the problem as depending on independent variables y_i only, but instead of computing y_d in each SQP step by a full nonlinear iteration, only one Newton step is performed. The approach may reduce the problem size substantially; only a reduced Hessian and gradient of dimension n_i are needed in the QP subproblems. This makes the partially reduced SQP method fast and storage-efficient; it is particularly well-suited for very large control problems with a comparatively small number of control (and other independent) variables, such as typical PDE control problems for instance. In contrast to the older fully reduced approach which reduces the problem with respect to *all* NLP constraints (including active inequalities!), the partially reduced approach permits a convenient and robust treatment of inequalities.

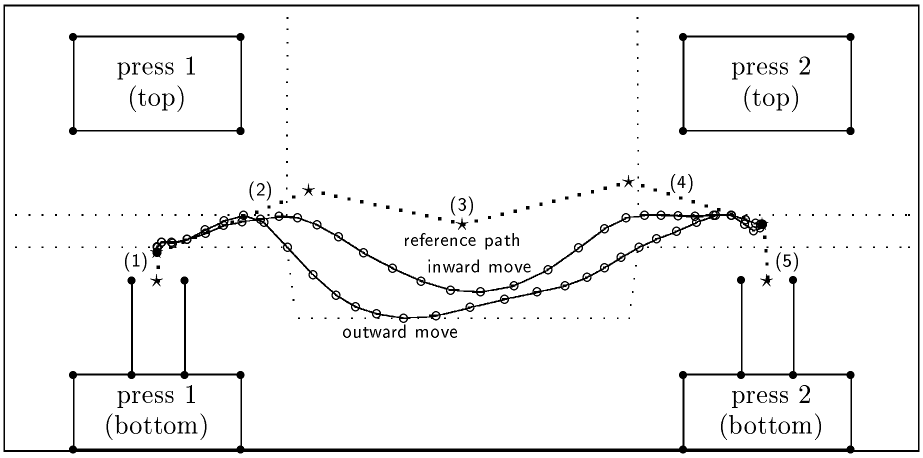

Figure 7: Front view of the press connection workcell (not to scale)

A partially reduced SQP method based on a collocation discretization is available in the trajectory optimization code OCPRSQP [45]. This implementation selects initial states x_1 and all control variables u_j as independent variables; the remaining states and all collocation variables are implicitly eliminated via connection and collocation conditions. The elimination leads to dense QP subproblems which are treated by the active set solver E04NAF from the commercial NAG Fortran library. Alternatively, state variables and continuity conditions could be left in the problem to preserve the m-stage block-sparse structure, and MSIPM could be used as QP solver.

The code OCPRSQP has been applied successfully to various robot optimization problems including trajectory optimization for satellite mounted robots [45, 43, 44, 36, 37]; results for the press connection are reported in [12].

5 Computational results

5.1 A real life transport maneuver

In the following we will consider a typical time-critical transport maneuver as an application example. At Mercedes-Benz, car body parts such as doors are made on production lines consisting of about a dozen hydraulic presses. Raw metal sheets are fed into the line and pressed at every station until they receive their final shape. The transport of partially processed sheets between the presses is accomplished by robots. Since the distance between presses is approximately 7 meters, each robot is equipped with an arm extension on which a pneumatic gripper is mounted. We refer to the transport maneuver as *press connection*; its cycle time depends on the type of object being transported and varies around

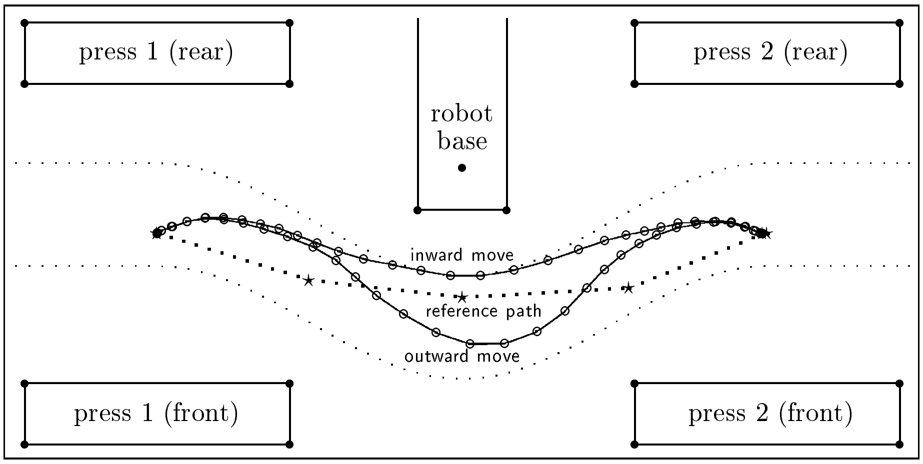

Figure 8: Top view of the press connection workcell (not to scale)

6–7 seconds including the unloaded return trip. During several years great effort has been spent on gradually increasing the throughput of these production lines, but further improvements of productivity are still desired. Therefore Mercedes-Benz has a great interest in mathematical optimization methods as a means to reduce the cycle time of the press connection to a minimum.

5.2 Model of the transport maneuver

In this project we use the commercial CAD system ROBCAD which provides direct user access to the internal database through its Application Programming Interface. The CAD system and a ROBCAD model of the press connection are supplied by our industry partners Tecnomatix GmbH and KUKA GmbH, respectively. The model includes the geometry and kinematics of a workcell containing two presses and the robot. Figures 7 and 8 show a vertical and a horizontal cross section of the workcell, where the distance between presses is scaled down in both cases. The ROBCAD model also includes a reference path consisting of five segments (see Fig. 7): (1) raising the load in the left press, (2) leaving the press, (3) transport to the right press, (4) entering the press, (5) lowering the load. (Note that the third segment is actually divided into two subsections.) The reference path is roughly specified by a small number of *locations* (marked by stars), each defining a position and orientation for the gripper or, more precisely, for the TCP frame. These locations are “interpolated” by a programmable trajectory generator according to the settings of certain motion parameters which influence the resulting shape and velocity profile of the tool center point’s trajectory. One possibility is a simple linear connection of the reference locations as shown in the figures.

joint	$\theta_{\min}$ [deg]	$\theta_{\max}$ [deg]	$\beta_{\max}$ [deg/s]	$T_{\max}$ [Nm]
1	-160	160	95	12860
2	-55	75	95	12860
3	-105	55	95	9507
4	-305	305	150	3683
5	-120	120	126	4376
6	-350	350	214	1547

Table 2: Limits on joint angles, speeds and torques for KUKA IR 761

Actually there are two structurally different basic maneuvers that move the load from the left press to the right press (see Fig. 8). In the first case, joint 6 turns in the negative direction and the load passes under the arm of the robot and close to its base. The reference trajectory is of this type, which we call an *inward move*. In the second case, joint 6 turns in the positive direction and the load travels far from the robot base. We call this an *outward move*. Optimal solutions of both types must be computed to find the fastest maneuver by direct comparison.

So far we consider in our optimization only the roughly horizontal press-to-press motion consisting of segments 2–4; its duration in the ROBCAD simulation is 2.06 seconds. On the vertical segments 1 and 5, which take about 0.55 seconds each, collisions are very likely since the load leaves or approaches its mounting inside one of the presses. Collision avoidance in these zones requires either straight, almost vertical TCP movement as on the reference path, or precise geometric data plus information on locally required safety margins; therefore we defer optimization of the complete maneuver until later.

Collision avoidance on segments 2–4 is achieved as follows. We specify a feasible region for the tool center point, and restrict the gripper orientation inside the presses by tight joint angle limits on the hand axes. Furthermore, when the load passes the robot during the outward move, a tighter joint angle limit on one hand axis prevents a part of the arm extension from hitting the robot arm. As shown by the fine dotted lines in Fig. 8, the horizontal TCP limits leave a curved area reaching from press to press around the robot (so that a safety distance of at least 10 cm between load and robot base is maintained), while the vertical limits shown in Fig. 7 leave narrow tunnels inside the presses and an upwardly open feasible region outside.

In addition to these task-specific geometric restrictions, the standard box constraints given in Table 2 have to be imposed on joint angles, velocities, and torques. The robot model is the one presented in section 2; it includes the multibody dynamics of the six links and the gripper with load, rotor inertias as seen by the motors, the pneumatic weight compensation on the second axis, and friction loss coefficients supplied by KUKA. For the reasons discussed in section 2.5.2 above, joint flexibility is excluded from the model in optimization calculations.

grid size	solution accuracy	optimal time	iterations SQP	IPM	CPU time [s] total	IPM	motion type
20	10^{-3}	1.767	10	81	62.1	5.09	inward
32	10^{-3}	1.770	10	90	100.6	8.69	inward
40	10^{-3}	1.767	13	121	165.9	14.47	inward
20	10^{-4}	1.7656	17	140	106.2	8.57	inward
32	10^{-4}	1.7639	21	196	214.7	19.17	inward
40	10^{-4}	1.7634	24	227	307.7	27.74	inward
20	10^{-3}	1.768	11	102	68.9	6.37	outward
32	10^{-3}	1.769	12	115	120.3	11.09	outward
40	10^{-3}	1.768	14	155	179.4	18.70	outward
20	10^{-4}	1.7650	20	197	125.8	11.81	outward
32	10^{-4}	1.7630	31	299	311.6	28.77	outward
40	10^{-4}	1.7636	28	296	357.9	35.93	outward

Table 3: Optimization results for the press connection

5.3 Optimizing the transport maneuver

Numerical optimization runs are performed according to the recursive multistage SQP approach using the multistage trajectory optimization package MSTOP. For the press connection maneuver we parameterize the control by piecewise constant functions on a uniform grid which is also used for the multiple shooting discretization. One step of the classical order four Runge-Kutta method is performed on each interval. Our control variables are the joint torques; the state variables are joint angles, joint velocities, and the unknown maneuver time T as a parameter. Both the inward move and the outward move are optimized with termination accuracies ranging from 10^{-3} to 10^{-6} and on different grids consisting of 20, 32, and 40 intervals. The discretization yields optimization problems with up to 853 variables, 624 equality constraints and 1598 inequality constraints, leaving up to 229 degrees of freedom. A (discrete) initial trajectory for the SQP iteration is generated by linear interpolation of initial and final positions in joint space, with a constant rate of acceleration during the first half of the maneuver and constant deceleration afterwards, resulting in a triangular velocity profile. The maneuver time is determined such that the torque limits are satisfied, but joint speed limits and geometric constraints may be violated. Numerical computations are performed on an Iris Indigo workstation with a 100 MHz R4000/R4010 processor reaching about 10 MFlops.

The optimization results for accuracies of three and four digits are listed in Table 3. Here the SQP iteration count is actually the number of QP subproblems solved: the termination criterion, measuring the expected objective decrease plus a weighted sum of constraint violations, needs a search direction to decide whether the current iterate is acceptable or not; hence at least one QP solution is always required, and no step is performed in the final SQP "iteration".

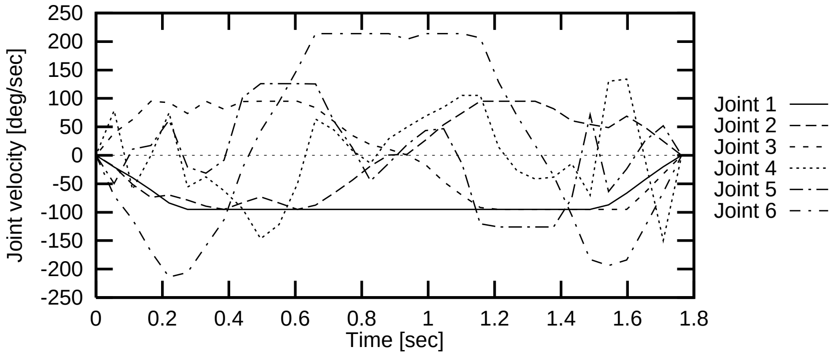

Figure 9: Optimal velocity profiles for outward move

Obviously, the inward move and outward move can be performed with comparable speed, both taking slightly less than 1.77 seconds. Hence the engineer may choose according to other criteria, such as sensitivity with respect to disturbances, safety considerations, or stress on the robot joints. Compared to the reference solution one gains 0.29 seconds in both cases, or 14 % on the (optimized) horizontal part of the maneuver and 9 % in the complete maneuver.

We observe that for all different grids and accuracies the values of the optimal transport time are very close together: they differ by at most 0.01 seconds. This remains true if we include the higher accuracy results; optimal values for the inward move range between 1.760 and 1.770 seconds, while values for the outward move range only between 1.762 and 1.769 seconds. These data indicate that near-optimal solutions can already be obtained on relatively coarse grids with low accuracy. In view of the practical application, we consider a discretization on about 20 intervals and an accuracy of two or three digits as appropriate for CAD based motion planning of maneuvers like the press connection, especially so since the computation time for a collision-free minimum time trajectory is rather low with only 1 to 1.5 minutes. If necessary, the final off-line optimization before down-loading the trajectory may still be performed on a finer grid and with higher accuracy, taking significantly less than 10 minutes for all cases considered above.

Table 3 also shows the good convergence behavior of our method on this problem class. All instances are solved after a relatively small number of SQP iterations: 10 to 14 iterations for three digits, and 17 to 31 iterations for four digits. Furthermore, the average number of interior iterations per SQP iteration is almost constant as a consequence of the warm start strategy. For finer grids and higher accuracies we observe a slight increase, but the average number varies between 8.1 and 11.1 only.

Examination of relative computation times reveals that the time spent on QP solution by the interior point method is never more than 10 % of the total time. The remaining time less at most 0.3 % is in all cases consumed by function and gradient generation for setting up the QP and during the line search, i.e., in evaluating and differentiating the robot's forward dynamics equations. The constant percentage of the two times results from the fact that SQP step reductions are very rare, so the line search needs only one function evaluation to accept the (full) step. The small percentage for the interior point method demonstrates the efficiency of the multistage SQP approach: about 90 % of computation time is actually spent to set up the problem, and only 10 % is needed for its solution. We expect that this high ratio can be reduced considerably if the differentiation of forward dynamics is implemented more efficiently, but this will only cut down the total effort.

In the following we take a closer look at the optimal solutions on 32 intervals with an accuracy of 10^{-4}. Vertical and horizontal projections of the trajectories for both the inward move and the outward move are included in Figs. 7 and 8; Fig. 9 shows the time histories of optimal joint velocities for the outward move. We observe that the tunnel constraints inside the presses are active in both cases, the lower TCP limit between presses is reached during the outward move, and the rear TCP limit near the robot base is touched during the inward move. The front TCP limit does not restrict the motion in any case.

To conclude this section, we note that the shape of optimal trajectories varies significantly among different optimization runs, even for the same maneuver type and even if optimal times agree up to the prescribed tolerance. Althought all solutions exhibit a “swinging” motion (which is a universal characteristic feature of optimal PTP trajectories), their shape is almost undetermined between the presses, especially in the vertical direction. Moreover, we note that for both maneuver types and without any regularity the lower TCP limit is active in some cases and inactive in others. Thus, the geometric constraints merely keep the TCP inside the feasible region, but they do not reduce its freedom of motion very much. This suggests that other restrictions may have a more significant influence in the optimization, and indeed we see in Fig. 9 that the joint speed limits of Table 2 are active during large portions of the maneuver. Every limit is reached at least once and remains active on a nondegenerate interval, except for joint four. Similar observations apply to the inward move.

5.4 Sensitivity analysis

In the previous section we suspected that different types of constraints do not have the same influence on possible reductions of the maneuver time. To clarify this, we perform a sensitivity analysis with respect to some of the restrictions, namely the joint speed limit on the first axis, the geometric constraints, and the maximal joint torques.

5.4.1 Joint speed limits

Velocity restrictions are apparently active during large portions of the maneuver. In particular, the base joint 1 which has to travel by far the greatest distance in

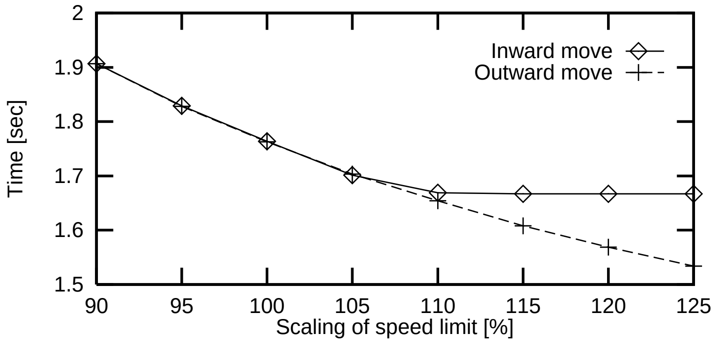

Figure 10: Optimal maneuver time vs. speed limit on first joint

the outward move, is at its speed limit for about 70 % of the maneuver time. In this period it rotates by 125 degrees (or 87 %) of the total 144 degrees. These observations suggest that the first joint speed might actually be the limiting factor in the optimization.

Figure 10 shows the optimal maneuver times for the inward move and the outward move if the first speed limit is varied between 90 % and 125 % of its nominal value. These data confirm our assumption. For both maneuver types, the final time drops considerably if the first speed limit is increased. However, in case of the inward move no improvement can be achieved if joint 1 is allowed to rotate at more than 110 % of its nominal speed. At this point the velocity bound on joint 6 becomes the limiting factor in the optimization.

5.4.2 Geometric constraints

For the geometric restrictions we compare only two cases: the constrained maneuver described above, and the completely unconstrained maneuver. The unconstrained optimal times are 1.7591 seconds (vs. 1.7639) for the inward move, and 1.7559 seconds (vs. 1.7630) for the outward move. In other words, collision avoidance increases the optimal maneuver times by only 0.3 % and 0.4 %! This drastically demonstrates the ambivalent role of geometric constraints: although they are essential in practice and difficult to handle in the optimization, their influence on the final time can be practically negligible—as in the case of this press connection maneuver.

5.4.3 Maximal motor torques

Finally we investigate the influence of joint torque limits on the maneuver time. All the limits are multiplied by a common factor in the range from 0.9 to 1.25; the resulting maneuver times are shown in Fig. 11. We see that the influence is not negligible but far smaller than the influence of the first joint speed limit. Even if the maximal torques are all increased to 125 % of their nominal values,

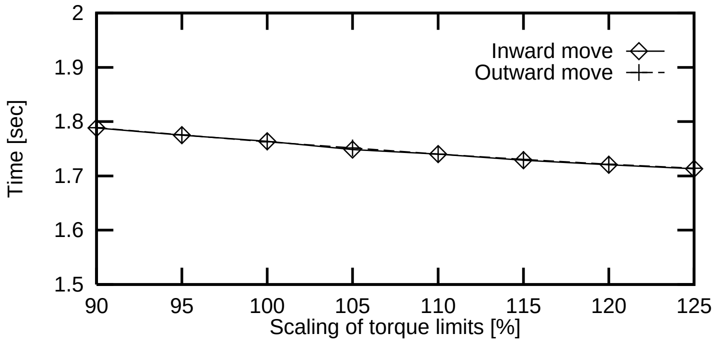

Figure 11: Optimal maneuver time vs. maximal torques

the optimal maneuver times drop only to 1.7132 seconds (or 97.1 %) for the inward move, and to 1.7143 seconds (or 97.2 %) for the outward move. Hence, reducing the torque limits does not much degrade performance. This justifies the introduction of safety margins to compensate for friction and possibly other disturbances that cannot be included in the model with sufficient accuracy.

6 Practical considerations

In view of a practical implementation of optimal trajectories in current robot hardware, we note that the results above are idealized in several ways, even if a detailed, calibrated robot model is used. Some of the issues that must be addressed are:

1) *The need for feedback:* It is necessary that there be feedback control operating that can correct for off nominal situations, for example to account for inaccuracies in the dynamic robot model. Ideally, the feedback would be optimal with respect to any disturbed state, at least in a linearized sense as developed in [10, 34]. Practically, one must make use of the feedback controllers in the robot hardware, since anything else would require substantial hardware modifications, and severely limit the applications of the approach. Of course, it is important to use reduced torque limits in the optimization, as we did by introducing friction loss coefficients. Then there is a margin for use of the feedback controller to make corrective action without exceeding the torque limits.

2) *Vibrations:* As already discussed in section 2.5, vibrations from joint flexibility can not yet be handled satisfactorily, and further research is being conducted on various possible approaches.

3) *Wearout and lifetime:* Time optimal control is aggressive, asking at least some actuators to work as hard as they can. Some of the restrictions discussed in section 2.6.2 protect the motors and gears against premature wearout, but in certain situations it may be important to include additional constraints, or choose a different cost function, to ensure a desired lifetime of other components.

4) *Getting the feedback controller to produce the optimal torque input:* The time optimal control problem as stated above develops an optimal torque history $u(t)$, but the hardware only allows to give position commands $\theta_c(t)$ to the feedback controllers for each joint. In [35] this issue is addressed in detail; here we give a brief summary of the results. If we simply give the optimal trajectory as the command, then a typical controller will always be behind in executing the trajectory, and hence, take longer to complete the maneuver. The problem is that the control law of the feedback controller uses its own logic to decide how much torque to apply, and in such controllers that are essentially proportional control with rate feedback, one does not get a large torque unless there is a large error. In addition, the rate feedback can retard sudden large changes in output. Hence, we must be smarter, and find a way to make the feedback controller generate the torque history we want. This requires that we back calculate the command that one would have to give the controller in order that it produce the desired torque output. It is natural in this context to include the motor dynamics in the model of the feedback controller, since the motor's back electromagnetic force acts as a feedback loop. Now, back calculation involves the inversion of a certain transfer function $G(s)$ associated with the controller, and the numbers n and m of the poles and zeros of $G(s)$ and the smoothness level of the torque history $u(t)$ determine whether this is possible or not. Assuming that we require a continuous command, $\theta_c \in \mathcal{C}^0$, the main result of [35] tells us that we must have $u \in \mathcal{C}^{n-m}$. If we ask only for a piece-wise continuous command with possible jumps at the grid points, $\theta_c \in \mathcal{C}^{-1}$, we must have $u \in \mathcal{C}^{n-m-1}$. These conditions can always be satisfied by choosing an appropriate class of admissible control functions for the trajectory optimization. For instance, in the typical cases $n - m - 1 \in \{0, 1\}$ one may use continuous functions with piece-wise constant slopes, or piece-wise quadratic functions with matching values and slopes at the grid points. If the time constants of all servo motors are negligible, then the necessary smoothness level is reduced by one and we may even be able to back calculate the command from the piece-wise constant parameterization of the optimal torque that was chosen above.

7 Conclusions

We have discussed the issue of dynamic robot modeling in the optimization context, and presented the specific components of a modular, generic model for the commercial robot KUKA IR 761. Using the nominal technical robot data, the model is sufficiently accurate and detailed to conduct realistic studies of off-line motion planning and trajectory optimization. For actual application in the production process, of course, a dynamic calibration will be required, and additional modeling work may become necessary.

We have further described recent numerical algorithms for the robust and efficient treatment of large optimization problems with many inequality constraints. Computational results for the new multistage trajectory optimization package MSTOP document that it performs excellently in state and control constrained point-to-point robot trajectory optimization. Fast optimization codes are thus available; it remains to make them accessible in the engineer's working environment and simplify their application by integrating the numerical software in a CAD system. To be specific, CAD based tools for the formulation of geometric constraints have to be developed, so that subroutines for the evaluation of task-specific restrictions and their derivatives can be generated automatically.

Finally, we have demonstrated in our example problem that collision-free high speed trajectories can be computed automatically. Although the time savings of nine percent may not appear very large, they represent a significant improvement by industrial standards. One must keep in mind that the press connection is a time critical maneuver that runs already very efficiently, so even a five percent reduction in time and hence cost would be considered a substantial gain in the production process.

8 Acknowledgements

This work was supported by the federal ministery of education, science, research and technology (BMBF) under grant 03-BO7HEI-6/3.0M750. G. V. Kostin was also supported by the Alexander von Humboldt Foundation. The authors would further like to thank their industry partners KUKA GmbH, Augsburg, and Tecnomatix GmbH, Dietzenbach, for providing technical robot and problem data and the CAD system ROBCAD, respectively, and for numerous valuable discussions. Finally, we are indebted to an unknown referee and to J. P. Schlöder for helpful suggestions and clarifications.

References

[1] L. D. Akulenko, S. K. Kaushinis, and G. V. Kostin. Influence of the dry friction upon controlled motion of electromechanical systems. *Comp. and Syst. Sci. Int.*, (1), 1994.

[2] W. W. Armstrong. Recursive solution to the equations of motion of an n link manipulator. In *Proc. 5th World Congr. Theory of Machines and Mechanisms*, volume 2, pages 1343–1346, Montreal, 1979.

[3] R. Bernhardt and S. L. Albright, editors. *Robot Calibration*. Chapman & Hall, 1993.

[4] J. T. Betts and W. P. Huffman. Path constrained trajectory optimization using sparse sequential quadratic programming. *AIAA J. Guidance*, 16(1):59–68, 1993.

[5] J. E. Bobrow. *Optimal Control of Robotic Manipulators*. Ph. D. dissertation, University of California, Los Angeles, 1982.

[6] J. E. Bobrow, S. Dubowsky, and J. S. Gibson. Time-optimal control of robotic manipulators along specified paths. *Int. J. Robotics Research*, 4(3):3–17, 1985.

[7] H. G. Bock. Numerical treatment of inverse problems in chemical reaction systems. In Ebert, Deuflhard, and Jäger, editors, *Modeling of Chemical Reaction Systems*, volume 18 of *Series Chemical Physics*, pages 102–125. Springer Verlag, 1981.

[8] H. G. Bock. *Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen*. Ph. D. dissertation, Bonner Mathematische Schriften 183, University of Bonn, 1987.

[9] H. G. Bock, E. Eich, and J. P. Schlöder. Numerical solution of constrained least squares boundary value problems in differential-algebraic equations. Preprint 440, SFB 123 „Stochastische Mathematische Modelle“, University of Heidelberg, 1987.

[10] H. G. Bock and P. Krämer-Eis. A multiple shooting method for numerical computation of open and closed loop controls in nonlinear systems. In *Proc. 9th IFAC World Congress*, volume IX, pages 179–183, Budapest, Hungary, 1984. Pergamon Press.

[11] H. G. Bock and K. J. Plitt. A multiple shooting algorithm for direct solution of constrained optimal control problems. In *Proc. 9th IFAC World Congress*, volume IX, pages 242–247, Budapest, Hungary, 1984. Pergamon Press.

[12] H. G. Bock, J. P. Schlöder, M. C. Steinbach, H. Wörn, V. H. Schulz, and R. W. Longman. Schnelle Roboter am Fließband: Mathematische Bahnoptimierung in der Praxis. In K.-H. Hoffmann, W. Jäger, T. Lohmann, and H. Schunck, editors, *Mathematik – Schlüsseltechnologie für die Zukunft*, pages 539–550. Springer Verlag, 1997.

[13] H. G. Bock and R. von Schwerin. An inverse dynamics ADAMS-method for constrained multibody systems. Preprint 93-27, IWR, 1993.

[14] N. N. Bolotnik and F. L. Chernousko. Optimization of manipulation robot control. *Sov. J. Comp. and Syst. Sci.*, 28(5):127–169, 1991.

[15] P. Chedmail and J.-P. Martineau. Characterization of the friction parameters of harmonic drive actuators. In C. L. Kirk and D. J. Inman, editors, *Dynamics and Control of Structures in Space III*, pages 567–581. Computational Mechanics Publications, Southampton, Boston, 1996.

[16] F. L. Chernousko, N. N. Bolotnik, and V. G. Gradetsky. *Manipulation Robots: Dynamics, Control, and Optimization*. CRC Press, Inc., Boca Raton, 1993.

[17] J. Denavit and R. S. Hartenberg. A kinematic notation for lower pair mechanisms based on matrices. *Trans. ASME J. Appl. Mech.*, 22(1):215–221, 1955.

[18] E. Eich. *Projizierende Mehrschrittverfahren zur numerischen Lösung von Bewegungsgleichungen technischer Mehrkörpersysteme mit Zwangsbedingungen und Unstetigkeiten.* Number 109 in Fortschr.-Ber. VDI, Reihe 18: Mechanik/Bruchmechanik. VDI Verlag, Düsseldorf, 1991.

[19] R. Featherstone. The calculation of robot dynamics using articulated-body inertias. *Int. J. Robotics Research*, 2(1):13–30, 1983.

[20] R. Featherstone. *Robot dynamics algorithms.* Kluwer Academic Publishers, Boston, Dordrecht, Lancaster, 1987.

[21] C. R. Hargraves and S. W. Paris. Direct trajectory optimization using nonlinear programming and collocation. *AIAA J. Guidance*, 10(4):338–342, 1987.

[22] B. Hartel, M. C. Steinbach, H. G. Bock, and R. W. Longman. The influence of friction on time optimal robot trajectories. Submitted for publication, Nov. 1996.

[23] T. Hidaka, T. Ishida, Y. Zhang, M. Sasahara, and Y. Tanioka. Vibration of a strain-wave gearing in an industrial robot. In *Proc. 1990 Int. Power Transm. and Gearing Conf.—New Techn. Power Transm.*, pages 789–794, New York, 1990. ASME.

[24] K.-D. Hilf. *Optimale Versuchsplanung zur dynamischen Roboterkalibrierung.* Number 590 in Fortschr.-Ber. VDI, Reihe 8: Meß-, Steuerungs- und Regelungstechnik. VDI Verlag, Düsseldorf, 1996.

[25] J. Hilsebecher. Vermessung von räumlichen Trajektorien zur kinematischen Kalibration von Robotern. Diploma thesis, University of Heidelberg, 1995.

[26] A. Jain. Unified formulation of dynamics for serial rigid multibody systems. *AIAA J. Guidance*, 14(3):531–542, 1991.

[27] R. Johanni. *Optimale Bahnplanung bei Industrierobotern.* Number 51 in Fortschr.-Ber. VDI, Reihe 18: Mechanik/Bruchmechanik. VDI Verlag, Düsseldorf, 1988.

[28] M. E. Kahn. *The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains.* Ph. D. dissertation, Stanford University, 1970.

[29] M. E. Kahn and B. Roth. The near-minimum-time control of open-loop articulated kinematic chains. *ASME J. Dyn. Syst., Meas., and Contr.*, 93:164–172, Sept. 1971.

[30] J. Konzelmann, H. G. Bock, and R. W. Longman. Time optimal trajectories of elbow robots by direct methods. In *Proc. 1989 AIAA Guid., Nav., and Contr. Conf.*, pages 883–894, Boston, MA.

[31] J. Konzelmann, H. G. Bock, and R. W. Longman. Time optimal trajectories of polar robot manipulators by direct methods. *Modeling and Simulation*, 20(5):1933–1939, 1989.

[32] G. V. Kostin. Dynamics of controlled rotations of loaded elastic link in a manipulational system with an electrical drive. *Sov. J. Comp. and Syst. Sci.*, (3), 1990.

[33] G. V. Kostin. Modeling the control motions of an electromechanical manipulation robot with elastic links. *Sov. J. Comp. and Syst. Sci.*, (5), 1992.

[34] P. Krämer-Eis. *Ein Mehrzielverfahren zur numerischen Berechnung optimaler Feedback-Steuerungen bei beschränkten nichtlinearen Steuerungsproblemen.* Ph. D. dissertation, Bonner Mathematische Schriften 183, University of Bonn, 1985.

[35] R. W. Longman, J. Li, M. C. Steinbach, and H. G. Bock. Issues in the implementation of time-optimal robot path planning. In *Proc. AIAA Nonlinear Dynamical Systems Symposium*, Reno, NV, Jan. 1997. Submitted to AIAA J. Guidance.

[36] R. W. Longman, V. H. Schulz, and H. G. Bock. Path planning for satellite mounted robots. In C. L. Kirk and D. J. Inman, editors, *Dynamics and Control of Structures in Space III*, pages 17–32. Computational Mechanics Publications, Southampton, Boston, 1996.

[37] M. Mössner-Beigel, M. C. Steinbach, H. G. Bock, and R. W. Longman. Time optimal path planning in polar robots with joint flexibility. Submitted for publication, Nov. 1996.

[38] F. Pfeiffer, F. L. Chernousko, N. N. Bolotnik, G. V. Kostin, and T. Rossmann. Tube-crawling robot: Modelling and optimization. *IEEE Trans. Robotics and Autom.* To appear.

[39] F. Pfeiffer and R. Johanni. A concept for manipulator trajectory planning. *IEEE J. Robotics and Autom.*, 3(2):115–123, 1987.

[40] B. Schletz. Hochgenaue bildverarbeitende Meßverfahren in der dynamischen Roboterkalibrierung. Diploma thesis, University of Heidelberg, 1995.

[41] J. P. Schlöder. *Numerische Methoden zur Behandlung hochdimensionaler Aufgaben der Parameteridentifizierung.* Ph. D. dissertation, Bonner Mathematische Schriften 187, University of Bonn, 1988.

[42] K. Schröer. *Identifikation von Kalibrationsparametern kinematischer Ketten.* Hanser, München, 1993.

[43] V. H. Schulz. A direct PRSQP method for path planning of satellite mounted robots. *ZAMM*, 76(S3):17–20, 1996.

[44] V. H. Schulz. Optimal paths for satellite mounted robots. *ZAMM*, 76(S3):291–294, 1996.

[45] V. H. Schulz. *Reduced SQP Methods for Large-Scale Optimal Control Problems in DAE with Application to Path Planning Problems for Satellite Mounted Robots.* Ph. D. dissertation, University of Heidelberg, 1996.

[46] V. H. Schulz, H. G. Bock, and M. C. Steinbach. Exploiting invariants in the numerical solution of multipoint boundary value problems for DAE. *SIAM J. Sci. Comput.*, 18(6), 1997. To appear. Also IWR Preprint 93-69, 1993.

[47] B. Simeon. MBSPACK—numerical integration software for constrained mechanical motion. *Surv. Math. Ind.*, 5(3):169–202, 1995.

[48] B. Simeon, C. Führer, and P. Rentrop. Differential-algebraic equations in vehicle system dynamics. *Surv. Math. Ind.*, 1(1):1–38, 1991.

[49] M. C. Steinbach. A structured interior point SQP method for nonlinear optimal control problems. In R. Bulirsch and D. Kraft, editors, *Computational Optimal Control*, volume 115 of *Int. Series Numer. Math.*, pages 213–222. Birkhäuser Verlag, Basel, Switzerland, 1994.

[50] M. C. Steinbach. *Fast Recursive SQP Methods for Large-Scale Optimal Control Problems.* Ph. D. dissertation, University of Heidelberg, 1995.

[51] M. C. Steinbach. Structured interior point SQP methods in optimal control. *ZAMM*, 76(S3):59–62, 1996.

[52] M. C. Steinbach, H. G. Bock, and R. W. Longman. Time optimal control of SCARA robots. In *Proc. 1990 AIAA Guid., Nav., and Contr. Conf.*, pages 707–716, Portland, OR.

[53] M. C. Steinbach, H. G. Bock, and R. W. Longman. Time-optimal extension and retraction of robots: Numerical analysis of the switching structure. *J. Optim. Theory and Appl.*, 84(3):589–616, 1995.

[54] I. Troch. Modelling for optimal control of systems. *Surv. Math. Ind.*, 5(4):203–292, 1995.

[55] T. D. Tuttle and W. Seering. Modeling a harmonic drive gear transmission. In *Proc. IEEE Int. Conf. Robotics and Autom.*, 1993.

[56] A. F. Vereshagin. Computer simulation of the dynamics of complicated mechanisms of robot manipulators. *Eng. Cyber.*, 6:65–70, 1974.

[57] D. P. Volkov and Y. N. Zubkov. Vibrations in drive with a harmonic gear transmission. *Russ. Eng. J.*, 58(5):11–15, 1978.

[58] R. von Schwerin and M. Winckler. MBSSIM—a guide to an integrator library for multibody systems simulation, version 1.00. Technical Report 94-75, IWR, University of Heidelberg, 1994.

[59] O. von Stryk. *Numerische Lösung optimaler Steuerungsprobleme: Diskretisierung, Parameteroptimierung und Berechnung der adjungierten Variablen.* Number 441 in Fortschr.-Ber. VDI, Reihe 8: Meß-, Steuerungs- und Regelungstechnik. VDI Verlag, Düsseldorf, 1995.

[60] M. W. Walker and D. E. Orin. Efficient dynamic computer simulation of robotic mechanisms. *ASME J. Dyn. Syst., Meas., and Contr.*, 104:205–211, 1982.

[61] M. Winckler and R. von Schwerin. Simulating vehicle systems with discontinuous effects using MBSSIM. *ZAMM*, 76(S1):587–588, 1996.

Authors' addresses: M. C. Steinbach and H. G. Bock, Interdisciplinary Center for Scientific Computing (IWR), University of Heidelberg, Im Neuenheimer Feld 368, 69120 Heidelberg, Germany—G. V. Kostin, Institute for Problems in Mechanics, Russian Academy of Sciences, prospekt Vernadskogo 101, Moscow 117526, Russia; presently guest researcher at IWR—R. W. Longman, Department of Mechanical Engineering, Columbia University, New York, New York 10027, USA. The first author is presently moving to the Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), Takustraße 7, 14195 Berlin, Germany.