



Motion Control

8. Motion Control

Wan Kyun Chung, Li-Chen Fu, Torsten Kröger

This chapter will focus on the motion control of robotic rigid manipulators. In other words, this chapter does not treat the motion control of mobile robots, flexible manipulators, and manipulators with elastic joints. The main challenge in the motion control problem of rigid manipulators is the complexity of their dynamics and uncertainties. The former results from nonlinearity and coupling in the robot manipulators. The latter is twofold: structured and unstructured. Structured uncertainty means imprecise knowledge of the dynamic parameters and will be touched upon in this chapter, whereas unstructured uncertainty results from joint and link flexibility, actuator dynamics, friction, sensor noise, and unknown environment dynamics, and will be treated in other chapters.

In this chapter, we begin with an introduction to motion control of robot manipulators from a fundamental viewpoint, followed by a survey and brief review of the relevant advanced materials. Specifically, the dynamic model and useful properties of robot manipulators are recalled in Sect. 8.1. The joint and operational space control approaches, two different viewpoints on control of robot manipulators, are compared in Sect. 8.2. Independent joint control and proportional–integral–derivative (PID) control, widely adopted in the field of industrial robots, are presented in Sects. 8.3 and 8.4, respectively. Tracking control, based on feedback linearization, is introduced in Sect. 8.5. The computed–torque control and its variants are described in Sect. 8.6. Adaptive control is introduced in Sect. 8.7 to solve the problem of structural uncertainty, whereas the optimality and robustness issues are covered in Sect. 8.8. To compute suitable set point signals as input values for these motion controllers, Sect. 8.9 introduces reference trajectory planning concepts. Since most controllers of robot manipulators are implemented

by using microprocessors, the issues of digital implementation are discussed in Sect. 8.10. Finally, learning control, one popular approach to intelligent control, is illustrated in Sect. 8.11.

8.1	Introduction to Motion Control	164
8.1.1	Dynamical Model	164
8.1.2	Control Tasks	165
8.1.3	Summary	165
8.2	Joint Space Versus Operational Space Control	166
8.2.1	Joint Space Control	166
8.2.2	Operational Space Control	166
8.3	Independent-Joint Control	167
8.3.1	Controller Design Based on the Single-Joint Model	167
8.3.2	Controller Design Based on the Multijoint Model	169
8.3.3	Summary	169
8.4	PID Control	169
8.4.1	PD Control for Regulation	169
8.4.2	PID Control for Regulation	170
8.4.3	PID Gain Tuning	170
8.4.4	Automatic Tuning	171
8.5	Tracking Control	172
8.5.1	Inverse Dynamics Control	172
8.5.2	Feedback Linearization	172
8.5.3	Passivity-Based Control	173
8.5.4	Summary	174
8.6	Computed-Torque Control	174
8.6.1	Computed-Torque Control	174
8.6.2	Computed-Torque-Like Control	175
8.6.3	Summary	177
8.7	Adaptive Control	177
8.7.1	Adaptive Computed-Torque Control	177
8.7.2	Adaptive Inertia-Related Control	178
8.7.3	Adaptive Control Based on Passivity	179

8.7.4	Adaptive Control with Desired Compensation.....	180	8.9.3	Trajectory Representations	184
8.7.5	Summary.....	180	8.9.4	Trajectory Planning Algorithms ...	186
8.8	Optimal and Robust Control	181	8.10	Digital Implementation	187
8.8.1	Quadratic Optimal Control	181	8.10.1	Z-Transform for Motion Control Implementation	188
8.8.2	Nonlinear \mathcal{H}_∞ Control.....	181	8.10.2	Digital Control for Coding	189
8.8.3	Passivity-Based Design of Nonlinear \mathcal{H}_∞ Control.....	182	8.11	Learning Control	190
8.8.4	A Solution to Inverse Nonlinear \mathcal{H}_∞ Control.....	183	8.11.1	Pure P-Type Learning Control	190
8.9	Trajectory Generation and Planning	183	8.11.2	P-Type Learning Control with a Forgetting Factor	190
8.9.1	Geometric Paths and Trajectories	183	8.11.3	Summary	191
8.9.2	Joint Space and Operational Space Trajectories	184	Video-References		191
			References		191

8.1 Introduction to Motion Control

The dynamical model of robot manipulators will be recalled in this section. Furthermore, important properties of this dynamical model, which is useful in controller design, will then be addressed. Finally, different control tasks of the robot manipulators will be defined.

8.1.1 Dynamical Model

For motion control, the dynamical model of rigid robot manipulators is conveniently described by Lagrange dynamics. Let the robot manipulator have n links and let the $(n \times 1)$ -vector \mathbf{q} of joint variables be $\mathbf{q} = [q_1, \dots, q_n]^T$. The dynamic model of the robot manipulator is then described by Lagrange's equation [8.1–6]

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}) = \boldsymbol{\tau}, \quad (8.1)$$

where $\mathbf{H}(\mathbf{q})$ is the $(n \times n)$ inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ is the $(n \times 1)$ -vector of Coriolis and centrifugal forces, $\boldsymbol{\tau}_g(\mathbf{q})$ is the $(n \times 1)$ -vector of gravity force, and $\boldsymbol{\tau}$ is the $(n \times 1)$ -vector of joint control inputs to be designed. Friction and disturbance input have been neglected here.

Remark 8.1

Other contributions to the dynamic description of the robot manipulators may include the dynamics of the actuators, joint and link flexibility, friction, noise, and disturbances. Without loss of generality, the case of the rigid robot manipulators is stressed here.

The control schemes that we will introduce in this chapter are based on some important properties of the dynamical model of robot manipulators. Before giving a detailed introduction to these different schemes, let us first give a list of those properties.

Property 8.1

The inertia matrix is a symmetric positive-definite matrix, which can be expressed

$$\lambda_h \mathbf{I}_n \leq \mathbf{H}(\mathbf{q}) \leq \lambda_H \mathbf{I}_n, \quad (8.2)$$

where λ_h and λ_H denote positive constants.

Property 8.2

The matrix $\mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) = \dot{\mathbf{H}}(\mathbf{q}) - 2\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ is skew-symmetric for a particular choice of $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ (which is always possible), i. e.,

$$\mathbf{z}^T \mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) \mathbf{z} = 0 \quad (8.3)$$

for any $(n \times 1)$ -vector \mathbf{z} .

Property 8.3

The $(n \times n)$ -matrix $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ satisfies

$$\|\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\| \leq c_o \|\dot{\mathbf{q}}\| \quad (8.4)$$

for some bounded constant c_o .

Property 8.4

The gravity force/torque vector satisfies

$$\|\boldsymbol{\tau}_g(\mathbf{q})\| \leq g_o \quad (8.5)$$

for some bounded constant g_o .

Property 8.5

The equation of motion is linear in the inertia parameters. In other words, there is a $(r \times 1)$ constant

vector \mathbf{a} and an $(n \times r)$ regressor matrix $\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ such that

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}) = \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\mathbf{a}. \quad (8.6)$$

The vector \mathbf{a} is comprised of link masses, moments of inertia, and the link, in various combinations.

Property 8.6

The mapping $\boldsymbol{\tau} \rightarrow \dot{\mathbf{q}}$ is passive; i. e., there exists $\alpha \geq 0$ such that

$$\int_0^t \dot{\mathbf{q}}^T(\beta) \boldsymbol{\tau}(\beta) d\beta \geq -\alpha, \quad \forall t < \infty. \quad (8.7)$$

Remarks 8.1

- Properties 8.3 and 8.4 are very useful since they allow us to establish upper bounds on the nonlinear terms in the dynamical model. As we will see further, several control schemes require knowledge of such upper bounds.
- In Property 8.5, the parameter vector \mathbf{a} is comprised of several variables in various combinations. The dimensionality of the parameter space is not unique, and the search over the parameter space is an important problem.
- In this section, we assume that *the robot manipulator is fully actuated* and this indicates that there is an independent control input for each degree of freedom (DOF). In contrast, the robot manipulators with joint or link flexibility are no longer fully actuated and the control problems are more difficult in general.

8.1.2 Control Tasks

It is instructive for comparative purposes to classify control objectives into the following two classes:

- *Trajectory tracking* is aimed at following a time-varying joint reference trajectory specified within

the manipulator workspace. In general, this desired trajectory is assumed to comply with the actuators' capacity. In other words, the joint velocity and acceleration associated with the desired trajectory should not violate, respectively, the velocity and acceleration limit of the manipulator. In practice, the capacity of actuators is set by torque limits, which result in bounds on the acceleration that are complex and state dependent.

- *Regulation* sometimes is also called point-to-point control. A fixed configuration in the joint space is specified; the objective is to bring to and keep the joint variable at the desired position in spite of torque disturbances and independently of the initial conditions. The behavior of transients and overshooting, are in general, not guaranteed.

The selection of the controller may depend on the type of task to be performed. For example, tasks only requiring the manipulator to move from one position to another without requiring significant precision during the motion between these two points can be solved by regulators, whereas such as welding, painting, and so on, require tracking controllers.

Remarks 8.2

- The regulation problem may be seen as a special case of the tracking problem (for which the desired joint velocity and acceleration are zero).
- The task specification above is given in the joint space and results in joint space control, which is the main content of this chapter. Sometimes, the task specification of the robot manipulators in terms of the desired trajectory of the end-effector (e.g., control with eye-in-hand) is carried out in the task space and gives rise to the operational space control, which will be introduced in Sect. 8.2.

8.1.3 Summary

In this section, we introduced the dynamical model of the robot manipulators and important properties of this dynamical model. Finally, we defined different control tasks of the robot manipulators.

8.2 Joint Space Versus Operational Space Control

In a motion control problem, the manipulator moves to a position to pick up an object, transports that object to another location, and deposits it. Such a task is an integral part of any higher-level manipulation tasks such as painting or spot-welding.

Tasks are usually specified in the task space in terms of a desired trajectory of the end-effector, while control actions are performed in the joint space to achieve the desired goals. This fact naturally leads to two kinds of general control methods, namely *joint space control* and *operational space control (task space control)* schemes.

8.2.1 Joint Space Control

The main goal of the joint space control is to design a feedback controller such that the joint coordinates $q(t) \in R^n$ track the desired motion $q_d(t)$ as closely as possible. To this end, consider the equations of motion (8.1) of an n -DOF manipulator expressed in the joint space [8.2, 4]. In this case, the control of robot manipulators is naturally achieved in the joint space, since the control inputs are the joint torques. Nevertheless, the user specifies a motion in terms of end-effector coordinates, and thus it is necessary to understand the following strategy.

Figure 8.1 shows the basic outline of the joint space control methods. Firstly, the desired motion, which is described in terms of end-effector coordinates, is converted to a corresponding joint trajectory using the inverse kinematics of the manipulator. Then the feedback controller determines the joint torque necessary to move the manipulator along the desired trajectory specified in joint coordinates starting from measurements of the current joint states [8.1, 4, 7, 8].

Since it is always assumed that the desired task is given in terms of the time sequence of the joint motion, joint space control schemes are quite adequate in situations where manipulator tasks can be accurately preplanned and little or no online trajectory adjustments are necessary [8.1, 4, 7, 9]. Typically, inverse kinematics is performed for some intermediate task points, and the joint trajectory is interpolated using the intermediate joint solutions. Although the command trajectory consists of straight-line motions in end-effector coordinates

between interpolation points, the resulting joint motion consists of curvilinear segments that match the desired end-effector trajectory at the interpolation points.

In fact, the joint space control includes simple proportional-derivative (PD) control, PID control, inverse dynamic control, Lyapunov-based control, and passivity-based control, as explained in the following sections.

8.2.2 Operational Space Control

In more complicated and less certain environments, end-effector motion may be subject to online modifications in order to accommodate unexpected events or to respond to sensor inputs. There are a variety of tasks in manufacturing where these type of control problem arise. In particular, it is essential when controlling the interaction between the manipulator and environment is of concern.

Since the desired task is often specified in the operational space and requires precise control of the end-effector motion, joint space control schemes are not suitable in these situations. This motivated a different approach, which can develop control schemes directly based on the dynamics expressed in the operational space [8.10, 11].

Let us suppose that the Jacobian matrix, denoted by $J(q) \in R^{n \times n}$, transforms the joint velocity ($\dot{q} \in R^n$) to the task velocity ($\dot{x} \in R^n$) according to

$$\dot{x} = J(q)\dot{q}. \quad (8.8)$$

Furthermore, assume that it is invertible. Then, the operational space dynamics is expressed as follows

$$f_c = \Lambda(q)\ddot{x} + \Gamma(q, \dot{q})\dot{x} + \eta(q), \quad (8.9)$$

where $f_c \in R^n$ denotes the command forces in the operational space; the *pseudo-inertia matrix* is defined by

$$\Lambda(q) = J^{-T}(q)H(q)J^{-1}(q), \quad (8.10)$$

and $\Gamma(q, \dot{q})$ and $\eta(q)$ are given by

$$\begin{aligned} \Gamma(q, \dot{q}) &= J^{-T}(q)C(q, \dot{q})J^{-1}(q) \\ &\quad - \Lambda(q)J(q)J^{-1}(q), \\ \eta(q) &= J^{-T}(q)\tau_g(q). \end{aligned}$$

The task space variables are usually reconstructed from the joint space variables, via the kinematic mappings. In fact, it is quite rare to have sensors to directly measure end-effector positions and velocities. Also, it

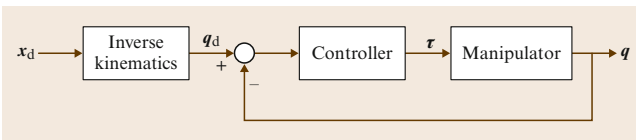


Fig. 8.1 Generic concept of joint space control

is worth remarking that an analytical Jacobian is utilized since the control schemes operate directly on task space quantities, i. e., the end-effector position and orientation.

The main goal of the operational space control is to design a feedback controller that allows execution of an end-effector motion $x(t) \in R^n$ that tracks the desired end-effector motion $x_d(t)$ as closely as possible. To this end, consider the equations of motion (8.9) of the manipulator expressed in the operational space. For this case, Fig. 8.2 shows a schematic diagram of the operational space control methods. There are several advantages to such an approach because operational

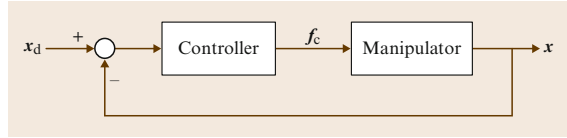


Fig. 8.2 Basic concept of operational space control

space controllers employ a feedback loop that directly minimizes task errors. Inverse kinematics need not be calculated explicitly, since the control algorithm embeds the velocity-level forward kinematics (8.8), as shown in the figure. Now, motion between points can be a straight-line segment in the task space.

8.3 Independent-Joint Control

By independent-joint control (i. e., decentralized control), we mean that the control inputs of each joint only depends on the measurement of the corresponding joint displacement and velocity. Due to its simple structure, this kind of control schemes offers many advantages. For example, by using independent-joint control, communication among different joints is saved. Moreover, since the computational load of controllers may be reduced, only low-cost hardware is required in actual implementations. Finally, independent-joint control has the feature of scalability, since the controllers on all the joints have the same formulation. In this section, two kinds of design of independent-joint control will be introduced: one focused on the dynamical model of each joint (i. e., based on the single-joint model) and the other based on the analysis of the overall dynamical model (i. e., the multijoint model) of robot manipulators.

8.3.1 Controller Design Based on the Single-Joint Model

The simplest independent-joint control strategy is to control each joint axis as a single-input single-output (SISO) system. Coupling effects among joints due to varying configuration during motion are treated as disturbance inputs. Without loss of generality, the actuator is taken as a rotary electric direct-current (DC) motor. Hence, the block diagram of the control scheme of joint i can be represented in the domain of the complex variables as shown in Fig. 8.3. In this scheme, θ is the angular variable of the motor, J is the effective inertia viewed from the motor side, R_a is the armature resistance (auto-inductance being neglected), and k_t and k_v are, respectively, the torque and motor constants. Furthermore, G_v denotes the voltage gain of the power

amplifier so that the reference input is the input voltage V_c of the amplifier instead of the armature voltage V_a . It has also been assumed that $F_m \ll k_v k_t / R_a$, i. e., the mechanical (viscous) friction coefficient has been neglected with respect to the electrical coefficient. Now the input–output transfer function of the motor can be written as

$$M(s) = \frac{k_m}{s(1 + sT_m)}, \quad (8.11)$$

where $k_m = G_v/k_v$ and $T_m = R_a J / k_v k_t$ are, respectively, the voltage-to-velocity gain and the time constant of the motor.

To guide selection of the control structure, start by noticing that an effective rejection of the disturbance d on the output θ is ensured by

1. A large value of the amplifier before the point of intervention of the disturbance
2. The presence of an integral action in the controller so as to cancel the effect of the gravitational component on the output at the steady state (i. e., constant θ).

In this case, as shown in Fig. 8.4, the types of control action with position and velocity feedback are

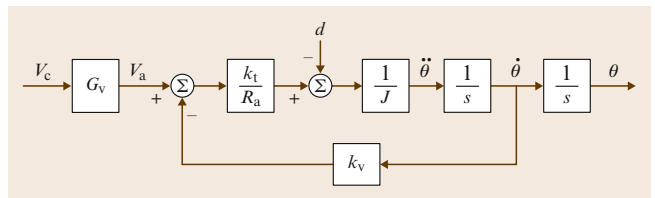


Fig. 8.3 Block scheme of a joint-driven system (after [8.4])

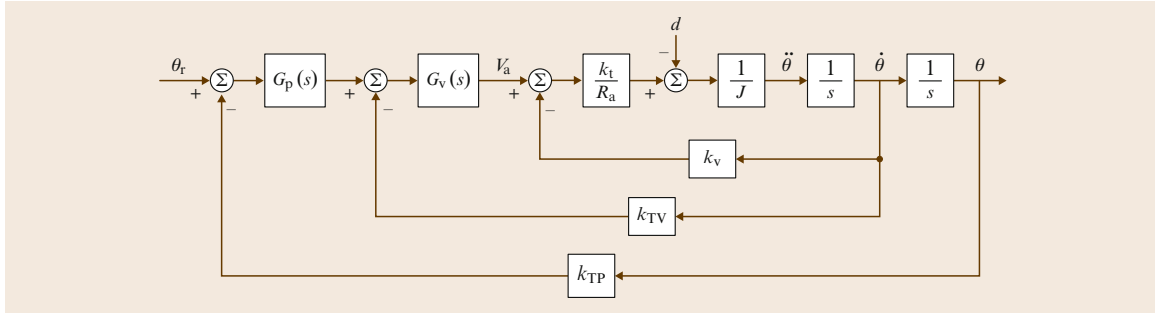


Fig. 8.4 Block scheme of position and velocity feedback (after [8.4])

characterized by [8.4]

$$G_p(s) = K_P, \quad G_v(s) = K_V \frac{1 + sT_V}{s}, \quad (8.12)$$

where $G_p(s)$ and $G_v(s)$ correspond to the position and velocity control actions, respectively. It is worth noting that the inner control action $G_v(s)$ is in a form of propositional–integral (PI) control to yield zero error in the steady state under a constant disturbance d . Furthermore, k_{TP} and k_{TV} are both transducer constants, and the amplifier gain K_V has been embedded in the gain of the inner controller. From the scheme of Fig. 8.4, the transfer function of the forward path is

$$P(s) = \frac{k_m K_P K_V (1 + sT_V)}{s(1 + sT_m)}, \quad (8.13)$$

while that of the return path is

$$H(s) = k_{TP} \left(1 + s \frac{k_{TV}}{K_P k_{TP}} \right). \quad (8.14)$$

The zero of the controller at $s = -1/T_V$ can be chosen so as to cancel the effects of the real pole of the motor at $s = -1/T_m$. Then, by setting $T_V = T_m$, the poles of the closed-loop system move on the root locus as a function of the loop gain, $k_m K_P K_V k_{TV}$. By increasing the position feedback gain K_P , it is possible to confine the closed-loop poles to a region of the complex plane with large absolute real part. Then, the actual location can be established by a suitable choice of K_V .

The closed-loop input–output transfer function is

$$\frac{\Theta(s)}{\Theta_r(s)} = \frac{\frac{1}{k_{TP}}}{1 + \frac{s k_{TP}}{K_P k_{TP}} + \frac{s^2}{k_m K_P k_{TP} K_V}}, \quad (8.15)$$

which can be compared with the typical transfer function of a second-order system

$$W(s) = \frac{\frac{1}{k_{TP}}}{1 + \frac{2\zeta s}{\omega_n} + \frac{s^2}{\omega_n^2}}. \quad (8.16)$$

It can be recognized that, with a suitable choice of the gains, it is possible to obtain any value of natural frequency ω_n and damping ratio ζ . Hence, if ω_n and ζ are given as design specifications, the following relations can be found

$$K_V k_{TV} = \frac{2\zeta \omega_n}{k_m} \quad \text{and} \quad K_P k_{TP} K_V = \frac{\omega_n^2}{k_m}. \quad (8.17)$$

For given transducer constants k_{TP} and k_{TV} , K_V and K_P will be chosen to satisfy the two equations above, respectively. On the other hand, the closed-loop disturbance/output function is

$$\frac{\Theta(s)}{D(s)} = \frac{\frac{s R_a}{k_t K_P K_{TP} K_V (1 + sT_m)}}{1 + \frac{s k_{TV}}{K_P k_{TP}} + \frac{s^2}{k_m K_P k_{TP} K_V}}, \quad (8.18)$$

which shows that the disturbance rejection factor is $X_R(s) = K_P k_{TP} K_V$ and is fixed, provided that K_V and K_P have been chosen via the approach above. Concerning the disturbance dynamics, the zero at the origin introduced by the PI, a real pole at $s = -1/T_m$, and the pair of complex poles with real part $-\zeta \omega_n$ should be kept in mind. In this case, an estimate T_R of the output recovery time needed by the control system to recover from the effect of a disturbance on the joint position can be evaluated by analyzing models of the transfer function above. Such an estimate can reasonably be expressed as $T_R = \max\{T_m, 1/\zeta \omega\}$.

8.3.2 Controller Design Based on the Multijoint Model

In recent years, independent-joint control schemes based on the complete dynamic model of the robot manipulators (i. e., a multijoint model) have been proposed. For example, following the approach of computed-torque-like control, [8.12] dealt with the regulation task for horizontal motions, and [8.13] and [8.14] handled the tracking task for arbitrary smooth trajectories. Since the overall dynamic model is considered, the effects of coupling among joints are handled. These schemes will be introduced in detail in Sect. 8.6.

8.3.3 Summary

In this section, we have presented two independent-joint control schemes: one is based on the single-joint model and the other is based on the multijoint model.

8.4 PID Control

Traditionally, control design in robot manipulators can be understood as the simple fact of tuning of a PD or PID compensator at the level of each motor driving the manipulator joints [8.1]. Fundamentally, a PD controller is a position and velocity feedback that has good closed-loop properties when applied to a double integrator system.

The PID control has a long history since Ziegler and Nichols' PID tuning rules were published in 1942 [8.15]. Actually, the strong point of PID control lies in its *simplicity* and clear physical meaning. Simple control is preferable to complex control, at least in industry, if the performance enhancement obtained by using complex control is not significant enough. The physical meanings of PID control [8.16] are as follows:

- *P-control* means the present effort making a present state into desired state
- *I-control* means the accumulated effort using the experience information of previous states
- *D-control* means the predictive effort reflecting the information about trends in future states.

8.4.1 PD Control for Regulation

A simple design method for manipulator control is to utilize a linear control scheme based on the linearization of the system about an operating point. An example of this method is a PD control with a gravity compensation scheme [8.17, 18]. Gravity compensation acts as

The former focuses on the dynamics of a single joint and regards the interaction among joints as a disturbance. This control scheme is simple but may not be suitable for high-speed tracking. Hence, we introduce the latter, which considers the overall dynamical model of robot manipulators such that the interaction among joints can be handled.

Further Reading

There are different types of feedback applied in the independent-joint control based on the single-joint model (such as pure position feedback or position, velocity, and acceleration feedback). A complete discussion is given in [8.4]. When the joint control servos are required to track reference trajectories with high speeds and accelerations, the tracking capabilities of the above schemes are inevitably degraded. A possible remedy is to adopt decentralized feedforward compensation to reduce tracking errors [8.4, 5].

a bias correction, compensating only for the amount of forces that create overshooting and an asymmetric transient behavior. Formally, it has the following form

$$\tau = \mathbf{K}_P(\mathbf{q}_d - \mathbf{q}) - \mathbf{K}_V\dot{\mathbf{q}} + \tau_g(\mathbf{q}), \quad (8.19)$$

where \mathbf{K}_P and $\mathbf{K}_V \in R^{n \times n}$ are positive-definite gain matrices. This controller is very useful for set-point regulation, i. e., $\mathbf{q}_d = \text{constant}$ [8.7, 18]. When this controller is applied to (8.1), the closed-loop equation becomes

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{K}_V\dot{\mathbf{q}} - \mathbf{K}_P\mathbf{e}_q = \mathbf{0}, \quad (8.20)$$

where $\mathbf{e}_q = \mathbf{q}_d - \mathbf{q}$, and the equilibrium point is $\mathbf{y} = [\mathbf{e}_q^T, \dot{\mathbf{q}}^T]^T = \mathbf{0}$. Now, the stability achieved by PD control with gravity compensation can be analyzed according to the closed-loop dynamic (8.20). Consider the positive-definite function

$$V = \frac{1}{2}\dot{\mathbf{q}}^T \mathbf{H}(\mathbf{q})\dot{\mathbf{q}} + \frac{1}{2}\mathbf{e}_q^T \mathbf{K}_V \mathbf{e}_q.$$

Then, the derivative of function becomes negative semidefinite for any value of $\dot{\mathbf{q}}$ by using Property 8.2 in Sect. 8.1, i. e.,

$$\dot{V} = -\dot{\mathbf{q}}^T \mathbf{K}_V \dot{\mathbf{q}} \leq -\lambda_{\min}(\mathbf{K}_V)\|\dot{\mathbf{q}}\|^2, \quad (8.21)$$

where $\lambda_{\min}(\mathbf{K}_V)$ means the smallest eigenvalue of \mathbf{K}_V . By invoking the Lyapunov stability theory and LaSalle's theorem [8.1], it can be shown that the regulation error will converge asymptotically to zero, while

their high-order derivatives remain bounded. This controller requires knowledge of the gravity components (structure and parameters), though it is simple.

Now, consider simple PD control without gravity compensation

$$\tau = \mathbf{K}_P(q_d - q) - \mathbf{K}_V\dot{q}, \quad (8.22)$$

then the closed-loop dynamic equation becomes

$$\mathbf{H}(q)\ddot{q} + \mathbf{C}(q, \dot{q})\dot{q} + \tau_g(q) + \mathbf{K}_V\dot{q} - \mathbf{K}_P e_q = \mathbf{0}. \quad (8.23)$$

Consider the positive definite function

$$V = \frac{1}{2}\dot{q}^T \mathbf{H}(q)\dot{q} + \frac{1}{2}e_q^T \mathbf{K}_V e_q + U(q) + U_0,$$

where $U(q)$ denotes the potential energy with the relation of $\partial U(q)/\partial q = \tau_g(q)$ and U_0 is a suitable constant. Taking time derivative of V along the closed-loop dynamics (8.23) gives the same result (8.21) with previous one using gravity compensation. In this case, the control system must be stable in the sense of Lyapunov, but it can not conclude that the regulation error will converge to zero by LaSalle's theorem [8.1]. Actually, the system precision (the size of the regulation error vector) will depend on the size of the gain matrix \mathbf{K}_P in the following form

$$\|e_q\| \leq \|\mathbf{K}_P^{-1}\|g_0, \quad (8.24)$$

where g_0 is that in Property 8.4 in Sect. 8.1. Hence, the regulation error can be arbitrarily reduced by increasing \mathbf{K}_P ; nevertheless, measurement noise and other unmodeled dynamics, such as actuator friction, will limit the use of high gains in practice.

8.4.2 PID Control for Regulation

An integral action may be added to the previous PD control in order to deal with gravity forces, which to some extent can be considered as a constant disturbance (from the local point of view). The PID regulation controller can be written in the following general form

$$\tau = \mathbf{K}_P(q_d - q) + \mathbf{K}_I \int f(q_d - q)dt - \mathbf{K}_V\dot{q},$$

where $\mathbf{K}_I \in R^{n \times n}$ is a positive-definite gain matrix, and:

- If $f(q_d - q) = q_d - q$, we have PID control.
- If $\mathbf{K}_I \int (-\dot{q})dt$ is added, we have PI²D control.
- If $f(\cdot) = \tanh(\cdot)$, we have PD + nonlinear integral control.

Global asymptotic stability (GAS) by PID control was proved in [8.12] for robotic motion control system including external disturbances, such as Coulomb

friction. Also, *Tomei* proved the GAS of PD control in [8.19] by using an adaptation for gravity term. On the other hand, *Ortega* et al. showed in [8.20] that PI²D control could yield semiglobal asymptotic stability (SGAS) in the presence of gravity and bounded external disturbances. Also, *Angeli* proved in [8.21] that PD control could achieve the input-output-to-state stability (IOSS) for robotic systems. Also, *Ramirez* et al. proved the SGAS (with some conditions) for PID gains in [8.22]. Also, *Kelly* proved in [8.23] that PD plus nonlinear integral control could achieve GAS under gravity.

Actually, a large integral action in PID control can cause instability of the motion control system. In order to avoid this, the integral gain should be bounded by an upper limit of the following form [8.1]

$$\frac{k_P k_V}{\lambda_H^2} > k_I,$$

where λ_H is that in Property 8.1 in Sect. 8.1, $\mathbf{K}_P = k_P \mathbf{I}$, $\mathbf{K}_I = k_I \mathbf{I}$, and $\mathbf{K}_V = k_V \mathbf{I}$. This relation gives the guidelines for gain selection implicitly. Also, PID control has generated a great variety of PID control plus something, e.g., PID plus friction compensator, PID plus gravity compensator, PID plus disturbance observer.

8.4.3 PID Gain Tuning

The PID control can be utilized for trajectory tracking as well as set-point regulation. True tracking control will be treated after Sect. 8.5. In this section, the simple but useful PID gain tuning method will be introduced for practical use. The general PID controller can be written in the following general form

$$\tau = \mathbf{K}_V \dot{e}_q + \mathbf{K}_P e_q + \mathbf{K}_I \int e_q dt,$$

or, in another form,

$$\tau = \left(\mathbf{K} + \frac{1}{\gamma^2} \mathbf{I} \right) \left(\dot{e}_q + \mathbf{K}_P e_q + \mathbf{K}_I \int e_q dt \right). \quad (8.25)$$

In a fundamental stability analysis of tracking control systems, *Qu* and *Dorsey* proved in [8.24] that PD control could satisfy uniform ultimate boundedness (UUB). Also, *Berghuis* and *Nijmeijer* proposed output feedback PD control, which satisfies semiglobal uniform ultimate boundedness (SGUUB) in [8.25] under gravity and a bounded disturbance. Recently, *Choi* et al. suggested inverse optimal PID control [8.26], assuring extended disturbance input-to-state stability (ISS).

Actually, if a PID controller (8.25) is repeatedly applied to the same set point or desired trajectory, then the maximum error will be proportional to the gains in the following form

$$\max_{0 \leq t \leq t_f} \|e_q(t)\| \propto \frac{\gamma^2}{\sqrt{2k\gamma^2 + 1}}, \quad (8.26)$$

where t_f denotes the final execution time of a given task and $\mathbf{K} = k\mathbf{I}$. This relation can be utilized to tune the gain of a PID controller and is referred to as the compound tuning rule [8.16]. The compound tuning rule implicitly includes simple tuning rules as follows (γ tuning method):

- Square tuning: $\max \|e_q\| \propto \gamma^2$, for a small k
- Linear tuning: $\max \|e_q\| \propto \gamma$, for a large k .

For example, suppose we select positive constant diagonal matrices $\mathbf{K}_p = k_p\mathbf{I}$, $\mathbf{K}_i = k_i\mathbf{I}$, while satisfying $k_p^2 > 2k_i$. For small k , the maximum error will be reduced by $\frac{1}{4}$ according to the square tuning rule, if we reduce the value γ by $\frac{1}{2}$. For large k , the maximum error will be proportionally reduced as γ according to the linear tuning rule. This means that we can tune the PID controller using only one parameter γ when the other gain parameters are fixed [8.16] (VIDEO 25). Although these rules are very useful in tuning the control performance, they can be utilized only for repetitive experiments for the same set point or desired trajectory because the tuning rules consist of proportional relations.

8.4.4 Automatic Tuning

For simplicity, define the composite error to be

$$s(t) = \dot{e}_q + \mathbf{K}_p e_q + \mathbf{K}_i \int e_q dt.$$

Now simple auto-tuning PID control is suggested by choosing one tuning parameter \mathbf{K} as shown in the following control form

$$\tau = \left(\hat{K}(t) + \frac{1}{\gamma^2} \mathbf{I} \right) s(t),$$

and its automatic tuning rule as follow

$$\frac{d\hat{K}_i}{dt} = \Gamma_i s_i^2(t), \quad \text{for } i = 1, \dots, n, \quad (8.27)$$

where Γ_i implies an update gain parameter for i -th control joint.

For practical use of the PID control, a target performance denoted by Ω is specified in advance

$$\sup_{0 \leq t \leq t_f} |s(t)| = \Omega,$$

in order to maintain the composite error within the region Ω . Moreover, since the auto-tuning rule has the decentralized type (8.27), we suggest the decentralized criterion for auto-tuning as follows

$$|s_i| > \frac{\Omega}{\sqrt{2n}}, \quad (8.28)$$

where n is the number of the joint coordinates. As soon as the composite error arrives at the tuning region (8.28), the auto-tuning rule is implemented to assist the achievement of target performance. On the contrary, if the composite error stays in non-tuning region, namely, $|s_i| \leq \Omega/\sqrt{2n}$, then the auto-tuning process stops. For this case, we expect that the gain \hat{K} updated by an auto-tuning rule (8.27) would be larger than the matrix \mathbf{K}_Ω able to achieve the target performance Ω . As a matter of fact, the auto-tuning rule plays a nonlinear damping role in the auto-tuning region.

Matlab Example (Multimedia)

Simple automatic tuning example for one-link manipulator control system is shown in the multimedia source to help readers' understanding.

Further Reading

The PID-type controllers were designed to solve the regulation control problem. They have the advantage of requiring knowledge of neither the model structure nor the model parameters. Also, the stability achieved by PID-type controllers was presented in this section. A range of books and papers [8.1, 15, 16, 22, 27, 28] are available to the robotics audience, detailing the individual tuning methods used in PID control and their concrete proofs.

8.5 Tracking Control

While independent PID controls are adequate in most set-point regulation problems, there are many tasks that require effective trajectory tracking capabilities such as plasma-welding, laser-cutting or high-speed operations in the presence of obstacles. In this case, employing local schemes requires moving slowly through a number of intermediate set points, thus considerably delaying the completion of the task. Therefore, to improve the trajectory tracking performance, controllers should take account of the manipulator dynamic model via a computed-torque-like technique.

The tracking control problem in the joint or task space consists of following a given time-varying trajectory $\mathbf{q}_d(t)$ or $\mathbf{x}_d(t)$ and its successive derivatives $\dot{\mathbf{q}}_d(t)$ or $\dot{\mathbf{x}}_d(t)$ and $\ddot{\mathbf{q}}_d(t)$ or $\ddot{\mathbf{x}}_d(t)$, which describe the desired velocity and acceleration, respectively. To obtain successful performance, significant effort has been devoted to the development of model-based control strategies [8.1, 2, 7]. Among the control approaches reported in the literature, typical methods include the inverse dynamics control, the feedback linearization technique, and the passivity-based control method.

8.5.1 Inverse Dynamics Control

Though the inverse dynamics control has a theoretical background, such as the theory of feedback linearization discussed later, its starting point is mechanical engineering intuition based on cancelling nonlinear terms and decoupling the dynamics of each link. Inverse dynamics control in joint space has the form

$$\boldsymbol{\tau} = \mathbf{H}(\mathbf{q})\mathbf{v} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}), \quad (8.29)$$

which, applied to (8.1), yields a set of n decoupled linear systems, e.g., $\ddot{\mathbf{q}} = \mathbf{v}$, where \mathbf{v} is an auxiliary control input to be designed. Typical choices for \mathbf{v} are

$$\mathbf{v} = \ddot{\mathbf{q}}_d + \mathbf{K}_V(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{K}_P(\mathbf{q}_d - \mathbf{q}), \quad (8.30)$$

or with an integral component

$$\begin{aligned} \mathbf{v} = & \ddot{\mathbf{q}}_d + \mathbf{K}_V(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{K}_P(\mathbf{q}_d - \mathbf{q}) \\ & + \mathbf{K}_I \int (\mathbf{q}_d - \mathbf{q}) dt, \end{aligned} \quad (8.31)$$

leading to the error dynamics equation

$$\ddot{\mathbf{e}}_q + \mathbf{K}_V\dot{\mathbf{e}}_q + \mathbf{K}_P\mathbf{e}_q = \mathbf{0}$$

for an auxiliary control input (8.30), and

$$\mathbf{e}_q^{(3)} + \mathbf{K}_V\ddot{\mathbf{e}}_q + \mathbf{K}_P\dot{\mathbf{e}}_q + \mathbf{K}_I\mathbf{e}_q = \mathbf{0},$$

if an auxiliary control input (8.31) is used. Both error dynamics are exponentially stable by a suitable choice of the gain matrices \mathbf{K}_V and \mathbf{K}_P (and \mathbf{K}_I).

Alternatively, inverse dynamics control can be described in the operational space. Consider the operational space dynamics (8.9). If the following inverse dynamics control is used in the operational space,

$$\mathbf{f}_c = \boldsymbol{\Lambda}(\mathbf{q})(\ddot{\mathbf{x}}_d + \mathbf{K}_V\dot{\mathbf{e}}_x + \mathbf{K}_P\mathbf{e}_x) + \boldsymbol{\Gamma}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \boldsymbol{\eta}(\mathbf{q}),$$

where $\mathbf{e}_x = \mathbf{x}_d - \mathbf{x}$, the resulting error dynamics is

$$\ddot{\mathbf{e}}_x + \mathbf{K}_V\dot{\mathbf{e}}_x + \mathbf{K}_P\mathbf{e}_x = \mathbf{0}, \quad (8.32)$$

and it is also exponentially stable. One apparent advantage of using this controller is that \mathbf{K}_P and \mathbf{K}_V can be selected with a clear physical meaning in operational space. However, as can be seen in (8.10), $\boldsymbol{\Lambda}(\mathbf{q})$ becomes very large when the robot approaches singular configurations [8.8]. This means that large forces in some direction are needed to move the arm.

8.5.2 Feedback Linearization

This approach generalizes the concept of inverse dynamics of rigid manipulators. The basic idea of feedback linearization is to construct a transformation as a so-called *inner-loop control*, which exactly linearizes the nonlinear system after a suitable state space change of coordinates. One can then design a second stage or *outer-loop control* in the new coordinates to satisfy the traditional control design specifications such as tracking, disturbance rejection, etc. [8.5, 29]. The full power of the feedback linearization scheme for manipulator control becomes apparent if one includes in the dynamic description of the manipulator the transmission dynamics, such as the elasticity resulting from shaft windup, gear elasticity, etc. [8.5].

In recent years, an impressive volume of literature has emerged in the area of differential-geometric methods for nonlinear systems. Most of the results in this area are intended to give abstract coordinate-free descriptions of various geometric properties of nonlinear systems and as such are difficult for the non-mathematician to follow. It is our intention here to give only the basic idea of the feedback linearization scheme and to introduce a simple version of this technique that finds an immediate application to the manipulator control problem. The reader is referred to [8.30] for a comprehensive treatment of the feedback linearization technique using differential-geometric methods.

Let us now develop a simple approach to the determination of linear state-space representations of the

manipulator dynamics (8.1) by considering a general sort of output $\xi \in R^p$

$$\xi = h(q) + r(t), \quad (8.33)$$

where $h(q)$ is a general predetermined function of the joint coordinate $q \in R^n$ and $r(t)$ is a general predetermined time function. The control objective will be to select the joint torque inputs τ in order to make the output $\xi(t)$ go to zero.

The choice of $h(q)$ and $r(t)$ is based on the control purpose. For example, if $h(q) = -q$ and $r(t) = q_d(t)$, the desired joint space trajectory we would like the manipulator to follow, then $\xi(t) = q_d(t) - q(t) \equiv e_q(t)$ is the joint space tracking error. Forcing $\xi(t)$ to zero in this case would cause the joint variables $q(t)$ to track their desired values $q_d(t)$, resulting in a manipulator trajectory-following problem. As another example, $\xi(t)$ could represent the operational space tracking error $\xi(t) = x_d(t) - x(t) \equiv e_x(t)$. Then, controlling $\xi(t)$ to zero would result in trajectory following directly in operational space where the desired motion is usually specified.

To determine a linear state-variable model for manipulator controller design, let us simply differentiate the output $\xi(t)$ twice to obtain

$$\dot{\xi} = \frac{\partial h}{\partial q} \dot{q} + \dot{r} = T\dot{q} + \dot{r}, \quad (8.34)$$

$$\ddot{\xi} = T\ddot{q} + \dot{T}\dot{q} + \ddot{r}, \quad (8.35)$$

where we defined a $(p \times n)$ transformation matrix of the form

$$T(q) = \frac{\partial h(q)}{\partial q} = \begin{pmatrix} \frac{\partial h}{\partial q_1} & \frac{\partial h}{\partial q_2} & \cdots & \frac{\partial h}{\partial q_n} \end{pmatrix}. \quad (8.36)$$

Given the output $h(q)$, it is straightforward to compute the transformation $T(q)$ associated with $h(q)$. In the special case where ξ represents the operational space velocity error, then $T(q)$ denotes the Jacobian matrix $J(q)$.

According to (8.1),

$$\ddot{q} = H^{-1}(q)[\tau - n(q, \dot{q})], \quad (8.37)$$

with the nonlinear terms represented by

$$n(q, \dot{q}) = C(q, \dot{q})\dot{q} + \tau_g(q). \quad (8.38)$$

Then (8.35) yields

$$\ddot{\xi} = \ddot{r} + \dot{T}\dot{q} + T(q)H^{-1}(q)[\tau - n(q, \dot{q})]. \quad (8.39)$$

Define the *control input* function

$$u = \ddot{r} + \dot{T}\dot{q} + T(q)H^{-1}(q)[\tau - n(q, \dot{q})]. \quad (8.40)$$

Now we may define a state $y(t) \in R^{2p}$ by $y = (\xi \dot{\xi})$ and write the manipulator dynamics as

$$\dot{y} = \begin{pmatrix} 0 & I_p \\ 0 & 0 \end{pmatrix} y + \begin{pmatrix} 0 \\ I_p \end{pmatrix} u. \quad (8.41)$$

This is a linear state-space system of the form

$$\dot{y} = Ay + Bu, \quad (8.42)$$

driven by the control input u . Due to the special form of A and B , this system is called the *Brunovsky canonical form* and it is always controllable from $u(t)$.

Since (8.40) is said to be a *linearizing transformation* for the manipulator dynamic equation, one may invert this transformation to obtain the joint torque

$$\tau = H(q)T^+(q)(u - \ddot{r} - \dot{T}\dot{q}) + n(q, \dot{q}), \quad (8.43)$$

where T^+ denotes the *Moore–Penrose inverse* of the transformation matrix $T(q)$.

In the special case $\xi = e_q(t)$, and if we select $u(t)$ so that (8.41) is stable by the PD feedback $u = -K_P \xi - K_V \dot{\xi}$, then $T = -I_n$ and the control input torque $\tau(t)$ defined by (8.43) makes the manipulator move in such a way that $y(t)$ goes to zero. In this case, the feedback linearizing control and the inverse dynamics control become the same.

8.5.3 Passivity-Based Control

This method explicitly uses the passivity properties of the Lagrangian system [8.31, 32]. In comparison with the inverse dynamics method, passivity-based controllers are expected to have better robust properties because they do not rely on the exact cancellation of the manipulator nonlinearities. The passivity-based control input is given by

$$\begin{aligned} \dot{q}_r &= \dot{q}_d + \alpha e_q, \quad \alpha > 0, \\ \tau &= H(q)\ddot{q}_r + C(q, \dot{q})\dot{q}_r + \tau_g(q) + K_V \dot{e}_q + K_P e_q. \end{aligned} \quad (8.44)$$

With (8.44), we obtain the following closed-loop system

$$H(q)\dot{s}_q + C(q, \dot{q})s_q + K_V \dot{e}_q + K_P e_q = 0, \quad (8.45)$$

where $s_q = \dot{e}_q + \alpha e_q$. Let us choose a Lyapunov function $V(y, t)$ as follows

$$V = \frac{1}{2} y^T \begin{pmatrix} \alpha K_V + K_P + \alpha^2 H & \alpha H \\ \alpha H & H \end{pmatrix} y = \frac{1}{2} y^T P y. \quad (8.46)$$

Since the above equation is positive definite, it has a unique equilibrium at the origin, i. e., $\mathbf{y} = (\mathbf{e}_q^T, \dot{\mathbf{e}}_q^T)^T = \mathbf{0}$. Moreover, V can be bounded by

$$\sigma_m \|\mathbf{y}\|^2 \leq \mathbf{y}^T \mathbf{P} \mathbf{y} \leq \sigma_M \|\mathbf{y}\|^2, \quad \sigma_M \geq \sigma_m > 0. \quad (8.47)$$

The time derivative of V gives

$$\dot{V} = -\dot{\mathbf{e}}_q^T \mathbf{K}_V \dot{\mathbf{e}}_q - \alpha \mathbf{e}_q^T \mathbf{K}_P \mathbf{e}_q = -\mathbf{y}^T \mathbf{Q} \mathbf{y} < 0, \quad (8.48)$$

where $\mathbf{Q} = \text{diag}[\alpha \mathbf{K}_P, \mathbf{K}_V]$. Since \mathbf{Q} is positive definite and quadratic in \mathbf{y} , it can be also bounded by

$$\kappa_m \|\mathbf{y}\|^2 \leq \mathbf{y}^T \mathbf{Q} \mathbf{y} \leq \kappa_M \|\mathbf{y}\|^2, \quad \kappa_M \geq \kappa_m > 0. \quad (8.49)$$

Then, from the bound of the Lyapunov function V , we get

$$\dot{V} \leq -\kappa_m \|\mathbf{y}\|^2 = -2\eta V, \quad \eta = \frac{\kappa_m}{\sigma_M}, \quad (8.50)$$

which finally yields

$$V(t) \leq V(0)e^{-2\eta t}. \quad (8.51)$$

It has been shown that the value of α affects the tracking result dramatically [8.33]. The manipulator tends to vibrate for small values of α . Larger values of α allow better tracking performance and protect \mathbf{s}_q from being spoiled by the velocity measurement noise when the position error is small. In [8.34], it was suggested that

$$\mathbf{K}_P = \alpha \mathbf{K}_V \quad (8.52)$$

be used for quadratic optimization.

8.5.4 Summary

In this section, we have reviewed some of the model-based motion control methods proposed to date. Under some control approaches, the closed-loop system has either asymptotic stability or globally exponential stability. However, such ideal performance cannot be obtained in practical implementation because factors such as sampling rate, measurement noise, disturbances, and unmodeled dynamics will limit the achievable gain and the performance of the control algorithms [8.33, 35, 36].

8.6 Computed-Torque Control

Through the years many kinds of robot control schemes have been proposed. Most of these can be considered as special cases of the class of computed-torque control (Fig. 8.5) which is the technique of applying feedback linearization to nonlinear systems in general [8.37, 38]. In the section, computed-torque control will be first introduced, and its variant, so-called computed-torque-like control, will be introduced later.

8.6.1 Computed-Torque Control

Consider the control input (8.29)

$$\boldsymbol{\tau} = \mathbf{H}(\mathbf{q}) \mathbf{v} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}),$$

which is also known as computed-torque control; it consists of an inner nonlinear compensation loop and an

outer loop with an exogenous control signal \mathbf{v} . Substituting this control law into the dynamical model of the robot manipulator, it follows that

$$\ddot{\mathbf{q}} = \mathbf{v}. \quad (8.53)$$

It is important to note that this control input converts a complicated nonlinear controller design problem into a simple design problem for a linear system consisting of n decoupled subsystems. One approach to the outer-loop control \mathbf{v} is propositional-derivative (PD) feedback, as in (8.30)

$$\mathbf{v} = \ddot{\mathbf{q}}_d + \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q,$$

in which case the overall control input becomes

$$\boldsymbol{\tau} = \mathbf{H}(\mathbf{q})(\ddot{\mathbf{q}}_d + \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q) + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}),$$

and the resulting linear error dynamics are

$$\ddot{\mathbf{e}}_q + \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q = \mathbf{0}. \quad (8.54)$$

According to linear system theory, convergence of the tracking error to zero is guaranteed [8.29, 39].

Remark 8.2

One usually lets \mathbf{K}_V and \mathbf{K}_P be $(n \times n)$ diagonal positive-definite gain matrices (i. e., $\mathbf{K}_V = \text{diag}(k_{v,1},$

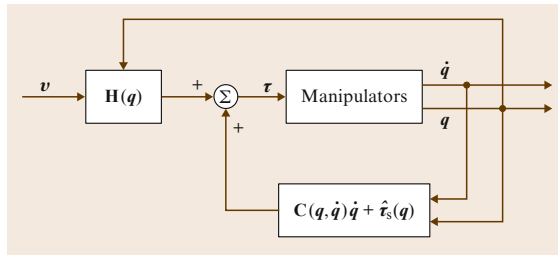


Fig. 8.5 Computed-torque control

$\dots, k_{v,n}) > \mathbf{0}$, $\mathbf{K}_P = \text{diag}(k_{p,1}, \dots, k_{p,n}) > \mathbf{0}$) to guarantee the stability of the error system. However, the format of the foregoing control never leads to *independent joint control* because the outer-loop multiplier $\mathbf{H}(\mathbf{q})$ and the full nonlinear compensation term $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q})$ in the inner loop scramble all joint signals among different control channels.

8.6.2 Computed-Torque-Like Control

It is worth noting that the implementation of computed-torque control requires that parameters of the dynamical model are accurately known and the control input is computed in real time. In order to avoid those problems, several variations of this control scheme have been proposed, for example, computed-torque-like control. An entire class of computed-torque-like controllers can be obtained by modifying the computed-torque control as

$$\boldsymbol{\tau} = \hat{\mathbf{H}}(\mathbf{q})\mathbf{v} + \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \hat{\boldsymbol{\tau}}_g(\mathbf{q}), \quad (8.55)$$

where $\hat{\cdot}$ represents the computed or nominal value and indicates that the theoretically exact feedback linearization cannot be achieved in practice due to the uncertainty in the systems. The overall control scheme is depicted in Fig. 8.6.

Computed-Torque-Like Control with Variable-Structure Compensation

Since there is parametric uncertainty, compensation is required in the outer-loop design to achieve trajectory tracking. The following shows a computed-torque-like control with variable-structure compensation

$$\mathbf{v} = \ddot{\mathbf{q}}_d + \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q + \Delta \mathbf{v}, \quad (8.56)$$

where the variable-structure compensation $\Delta \mathbf{v}$ is devised as

$$\Delta \mathbf{v} = \begin{cases} -\rho(\mathbf{x}, t) \frac{\mathbf{B}^T \mathbf{P} \mathbf{x}}{\|\mathbf{B}^T \mathbf{P} \mathbf{x}\|} & \text{if } \|\mathbf{B}^T \mathbf{P} \mathbf{x}\| \neq 0, \\ 0 & \text{if } \|\mathbf{B}^T \mathbf{P} \mathbf{x}\| = 0, \end{cases} \quad (8.57)$$

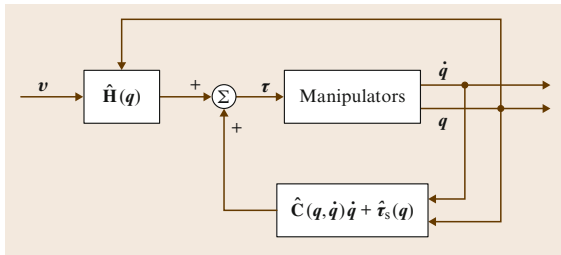


Fig. 8.6 Computed-torque-like control

where $\mathbf{x} = (\mathbf{e}_q^T, \dot{\mathbf{e}}_q^T)^T$, $\mathbf{B} = (\mathbf{0}, \mathbf{I}_n)^T$, \mathbf{P} is a $(2n \times 2n)$ symmetric positive-definite matrix satisfying

$$\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} = -\mathbf{Q}, \quad (8.58)$$

with \mathbf{A} being defined as

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_n \\ -\mathbf{K}_P & -\mathbf{K}_V \end{pmatrix}, \quad (8.59)$$

\mathbf{Q} being any appropriate $(2n \times 2n)$ symmetric positive-definite matrix,

$$\rho(\mathbf{x}, t) = \frac{1}{1 - \alpha} [\alpha\beta + \|\mathbf{K}\| \|\mathbf{x}\| + \bar{\mathbf{H}}\phi(\mathbf{x}, t)], \quad (8.60)$$

where α and β are positive constants such that $\|\mathbf{H}^{-1}(\mathbf{q})\hat{\mathbf{H}}(\mathbf{q}) - \mathbf{I}_n\| \leq \alpha < 1$ for all $\mathbf{q} \in R^n$ and $\sup_{t \in [0, \infty)} \|\ddot{\mathbf{q}}_d(t)\| < \beta$, respectively, \mathbf{K} is the $(n \times 2n)$ matrix defined as $\mathbf{K} = [\mathbf{K}_P, \mathbf{K}_V]$, $\bar{\lambda}_H$ being a positive constant such that $\|\mathbf{H}^{-1}(\mathbf{q})\| \leq \bar{\lambda}_H$ for all $\mathbf{q} \in R^n$, and the function ϕ being defined as

$$\begin{aligned} & \|[\hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})]\dot{\mathbf{q}} + [\hat{\boldsymbol{\tau}}_g(\mathbf{q}) - \boldsymbol{\tau}_g(\mathbf{q})]\| \\ & \leq \phi(\mathbf{x}, t). \end{aligned} \quad (8.61)$$

Convergence of the tracking error to zero can be shown using the Lyapunov function

$$V = \mathbf{x}^T \mathbf{P} \mathbf{x}, \quad (8.62)$$

following the stability analysis in [8.5, 40].

Remarks 8.3

- By Property 8.1 in Sect. 8.1, there exist positive constants $\bar{\lambda}_H$ and $\bar{\lambda}_h$ such that $\bar{\lambda}_h \leq \|\mathbf{H}^{-1}(\mathbf{q})\| \leq \bar{\lambda}_H$ for all $\mathbf{q} \in R^n$. If we choose

$$\hat{\mathbf{H}} = \frac{1}{c} \mathbf{I}_n, \quad \text{where } c = \frac{\bar{\lambda}_H + \bar{\lambda}_h}{2}, \quad (8.63)$$

it can be shown that

$$\|\mathbf{H}^{-1}(\mathbf{q})\hat{\mathbf{H}}(\mathbf{q}) - \mathbf{I}_n\| \leq \frac{\bar{\lambda}_H - \bar{\lambda}_h}{\bar{\lambda}_H + \bar{\lambda}_h} \equiv \alpha < 1, \quad (8.64)$$

which indicates that there is always at least one choice of $\hat{\mathbf{H}}$ for some $\alpha < 1$.

- Due to the discontinuity in $\Delta \mathbf{v}$, chattering phenomenon may occur when the control scheme is applied. It is worth noting that chattering is often undesirable since the high-frequency component in

the control can excite unmodeled dynamic effect (such as joint flexibility) [8.6, 29, 38]. In order to avoid chattering, the variable-structure compensation can be modified to become smooth, i. e.,

$$\Delta \mathbf{v} = \begin{cases} -\rho(\mathbf{x}, t) \frac{\mathbf{B}^T \mathbf{P} \mathbf{x}}{\|\mathbf{B}^T \mathbf{P} \mathbf{x}\|} & \text{if } \|\mathbf{B}^T \mathbf{P} \mathbf{x}\| > \varepsilon, \\ -\frac{\rho(\mathbf{x}, t)}{\varepsilon} \mathbf{B}^T \mathbf{P} \mathbf{x} & \text{if } \|\mathbf{B}^T \mathbf{P} \mathbf{x}\| \leq \varepsilon, \end{cases} \quad (8.65)$$

where ε is a positive constant and is used as the boundary layer. Following this modification, convergence of tracking errors to a residual set can be ensured, and the size of this residual set can be made smaller by use of a smaller value of ε .

Computed-Torque-Like Control with Independent-Joint Compensation

Obviously, the previous compensation scheme is centralized, which implies that the online computation load is heavy and high-cost hardware is required for practical implementation. In order to solve this problem, a scheme with independent-joint compensation is introduced below. In this scheme, a computed-torque-like control is designed with estimates as

$$\hat{\mathbf{H}}(\mathbf{q}) = \mathbf{I}, \quad \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{0}, \quad \hat{\boldsymbol{\tau}}(\mathbf{q}) = \mathbf{0}. \quad (8.66)$$

Then, we use the outer-loop \mathbf{v} as

$$\mathbf{v} = \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q + \Delta \mathbf{v}, \quad (8.67)$$

where the positive constants \mathbf{K}_P and \mathbf{K}_V are selected to be sufficiently large, and the i -th component Δv_i of $\Delta \mathbf{v} = (\mathbf{v}_1, \dots, \mathbf{v}_n)^T$ is

$$\Delta v_i = \begin{cases} -[\boldsymbol{\beta}^T \mathbf{w}(\mathbf{q}_d, \dot{\mathbf{q}}_d)]^2 \frac{s_i}{\varepsilon_i} & \text{if } |s_i| \leq \frac{\varepsilon_i}{\boldsymbol{\beta}^T \mathbf{w}(\mathbf{q}_d, \dot{\mathbf{q}}_d)}, \\ -\boldsymbol{\beta}^T \mathbf{w}(\mathbf{q}_d, \dot{\mathbf{q}}_d) \frac{s_i}{|s_i|} & \text{if } |s_i| > \frac{\varepsilon_i}{\boldsymbol{\beta}^T \mathbf{w}(\mathbf{q}_d, \dot{\mathbf{q}}_d)}. \end{cases} \quad (8.68)$$

In this compensation, $s_i = \dot{e}_{q,i} + \lambda_i e_{q,i}$, $i \in \{1, \dots, n\}$, and $\lambda_i, i \in \{1, \dots, n\}$ are positive constants. Furthermore, following the properties of robot manipulators, we have

$$\begin{aligned} & \|\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_d + \boldsymbol{\tau}_g(\mathbf{q})\| \\ & \leq \beta_1 + \beta_2 \|\mathbf{q}\| + \|\dot{\mathbf{q}}\| = \boldsymbol{\beta}^T \mathbf{w}(\mathbf{q}, \dot{\mathbf{q}}) \end{aligned}$$

for some suitable positive constants β_1 , β_2 , and β_3 , where $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3)^T$ and

$$\mathbf{w}(\mathbf{q}, \dot{\mathbf{q}}) = [1, \|\mathbf{q}\|, \|\dot{\mathbf{q}}\|]^T. \quad (8.69)$$

Finally, ε_i , $i \in \{1, \dots, n\}$, is the variable length of the boundary layer, satisfying

$$\dot{\varepsilon}_i = -g_i \varepsilon_i, \quad \varepsilon(0) > 0, \quad g_i > 0. \quad (8.70)$$

It is worth pointing out that the term \mathbf{w} in this control scheme is devised as the desired compensation rather than feedback. Furthermore, this scheme is in a form of independent-joint control and hence has the advantages introduced before. The convergence of the tracking error to zero can be shown using the Lyapunov function

$$V = \frac{1}{2} (\mathbf{e}_q^T \dot{\mathbf{e}}_q^T) \begin{pmatrix} \lambda \mathbf{K}_P & \mathbf{H} \\ \mathbf{H} & \lambda \mathbf{H} \end{pmatrix} \begin{pmatrix} \mathbf{e}_q \\ \dot{\mathbf{e}}_q \end{pmatrix} + \sum_{i=1}^n g_i^{-1} \varepsilon_i, \quad (8.71)$$

whose time derivative along the trajectory of the closed-loop systems follows

$$\dot{V} = -\alpha \left\| \begin{pmatrix} \mathbf{e}_q \\ \dot{\mathbf{e}}_q \end{pmatrix} \right\|^2, \quad (8.72)$$

with α being some positive constant, if sufficiently large \mathbf{K}_P and γ are used. The detailed analysis of stability, which requires Properties 8.3 and 8.4, can be found in [8.13].

Remark 8.3

Similar to computed-torque-like control with variable-structure compensation, we can consider the nonzero boundary layer as

$$\dot{\varepsilon}_i = -g_i \varepsilon_i, \quad \varepsilon(0) > 0, \quad g_i, \alpha_i > 0. \quad (8.73)$$

Following this modification, tracking errors converge to a residual set. The size of this residual set can be made smaller by the use of a smaller value of ε (i. e., smaller α_i).

For the task of point-to-point control, one PD controller with gravity compensation is designed with the estimates

$$\hat{\mathbf{H}}(\mathbf{q}) = \mathbf{I}, \quad \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{0}, \quad \hat{\boldsymbol{\tau}}(\mathbf{q}) = \boldsymbol{\tau}_g(\mathbf{q}), \quad (8.74)$$

with $\boldsymbol{\tau}_g(\mathbf{q})$ being the gravity term of the dynamical model of the robot manipulators. Then, we use the outer-loop \mathbf{v} as

$$\mathbf{v} = \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q, \quad (8.75)$$

such that the control input becomes

$$\boldsymbol{\tau} = \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q + \boldsymbol{\tau}_g(\mathbf{q}). \quad (8.76)$$

This scheme is much simpler to implement than the exact computed-torque controller. The convergence of the tracking error to zero can be shown using the Lyapunov function

$$V = \frac{1}{2} \dot{e}_q^T \mathbf{H}(q) \dot{e}_q + \frac{1}{2} e_q^T \mathbf{K}_p e_q, \quad (8.77)$$

whose time derivative along the solution trajectories of the closed-loop system is

$$\dot{V} = -\dot{e}_q^T \mathbf{K}_v \dot{e}_q. \quad (8.78)$$

The detailed analysis of stability is given in [8.12]. It is necessary to note that this result is for the case of regulation, rather than for the case of tracking, since the former theoretical base, which relies on LaSalle's lemma, requires the system be autonomous (time invariant) [8.38, 41, 42].

Remark 8.4

If we neglect gravity in the dynamical model of the robot manipulators, then the gravity estimation can be omitted here, i. e., $\hat{\tau}_g(q) = \mathbf{0}$, so that the control law becomes

$$\tau = v = \mathbf{K}_v \dot{e}_q + \mathbf{K}_p e_q, \quad (8.79)$$

8.7 Adaptive Control

An adaptive controller differs from an ordinary controller in that the controller parameters are time varying, and there is a mechanism for adjusting these parameters online based on some signals in the closed-loop systems. By the use of such control scheme, the control goal can be achieved even if there is parametric uncertainty in the plant. In this section, we will introduce several adaptive control schemes to deal with the case of imperfect knowledge of dynamical parameters of the robot manipulators. The control performance of those adaptive control schemes, including adaptive computed-torque control, adaptive inertia-related control, adaptive control based on passivity, and adaptive control with desired compensation, are basically derived from Property 8.5. Finally, the condition of persistent excitation, which is important in parameter convergence, will be addressed.

8.7.1 Adaptive Computed-Torque Control

The computed-torque control scheme is appealing, since it allows the designer to transform a MIMO highly coupled nonlinear system into a very simple decoupled

linear system, whose control design is a well-established problem. However, this method of feedback linearization relies on perfect knowledge of the system parameters, and failure to have this will cause erroneous parametric estimates, resulting in a mismatch term in the closed-loop model of the error system. That term can be interpreted as a nonlinear perturbation acting at the input of the closed-loop system. In order to solve the problem due to parametric uncertainty, we instead consider the inverse dynamics method with parameter estimates of

$$\tau = \hat{\mathbf{H}}(q)(\ddot{q}_d + \mathbf{K}_v \dot{e}_q + \mathbf{K}_p e_q) + \hat{\mathbf{C}}(q, \dot{q})\dot{q} + \hat{\tau}_g(q), \quad (8.80)$$

where $\hat{\mathbf{H}}$, $\hat{\mathbf{C}}$, $\hat{\tau}_g$ have the same functional form as \mathbf{H} , \mathbf{C} , τ_g . From Property 8.5 of the dynamics model, we have

$$\hat{\mathbf{H}}(q)\ddot{q} + \hat{\mathbf{C}}(q, \dot{q})\dot{q} + \hat{\tau}_g(q) = \mathbf{Y}(q, \dot{q}, \ddot{q})\hat{a}, \quad (8.81)$$

where $\mathbf{Y}(q, \dot{q}, \ddot{q})$, called the regressor, is a known $(n \times r)$ function matrix and \hat{a} is the $(r \times 1)$ vector that summarizes all the estimated parameters. Substituting this

8.6.3 Summary

In this section, we have presented two control schemes: the computed-torque and computed-torque-like control. The former transforms a multi-input multi-output (MIMO) nonlinear robotic system into a very simple decoupled linear closed-loop system whose control design is a well-established problem. Since the practical implementation of the former control requires pre-knowledge of all the manipulator parameters and its payload, which may not be realistic, the latter was introduced to relax the foregoing requirement and still to achieve the objective of tracking subject to system's uncertainty.

Further Reading

The PD control with different feedforward compensation for tracking control is investigated in [8.43]. An adaptive control scheme based on PD control is presented in [8.19].

control input τ into the manipulator dynamics gives the following closed-loop error model

$$\hat{\mathbf{H}}(q)(\ddot{e}_q + \mathbf{K}_v \dot{e}_q + \mathbf{K}_p e_q) = \mathbf{Y}(q, \dot{q}, \ddot{q})\tilde{a}, \quad (8.82)$$

where $\tilde{a} = \hat{a} - a$. In order to acquire an appropriate adaptive law, we first assume that the acceleration term \ddot{q} is measurable, and that the estimated inertia matrix $\hat{\mathbf{H}}(q)$ is never singular. Now, for convenience, the error equation is rewritten as

$$\dot{x} = \mathbf{A}x + \mathbf{B}\hat{\mathbf{H}}^{-1}(q)\mathbf{Y}(q, \dot{q}, \ddot{q})\tilde{a}, \quad (8.83)$$

with $x = (e_q^T, \dot{e}_q^T)^T$,

$$\mathbf{A} = \begin{pmatrix} \mathbf{0}_n & \mathbf{I}_n \\ -\mathbf{K}_p & -\mathbf{K}_v \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{0}_n \\ \mathbf{I}_n \end{pmatrix}. \quad (8.84)$$

The adaptive law is considered as

$$\dot{\hat{a}} = -\mathbf{\Gamma}^{-1}\mathbf{Y}^T(q, \dot{q}, \ddot{q})\hat{\mathbf{H}}^{-1}(q)\mathbf{B}^T\mathbf{P}x, \quad (8.85)$$

where $\mathbf{\Gamma}$ is an $(r \times r)$ positive-definite constant matrix, and \mathbf{P} is a $(2n \times 2n)$ symmetric positive-definite constant matrix satisfying

$$\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} = -\mathbf{Q}, \quad (8.86)$$

with \mathbf{Q} being a symmetric positive-definite constant matrix with coherent dimension. In this adaptive law, we made two assumptions:

- The joint acceleration \ddot{q} is measurable, and
- The bounded range of the unknown parameter is available.

The first assumption is to ensure that the regressor $\mathbf{Y}(q, \dot{q}, \ddot{q})$ is known a priori, whereas the second assumption is to allow one to keep the estimate $\hat{\mathbf{H}}(q)$ nonsingular by restricting the estimated parameter \hat{a} to lie within a range about the true parameter value.

Convergence of the tracking error and maintaining boundedness of all internal signals can actually be guaranteed by Lyapunov stability theory with the Lyapunov function

$$\dot{V} = -x^T\mathbf{Q}x. \quad (8.87)$$

Detailed stability analysis is given in [8.2].

Remark 8.5

For practical and theoretical reasons, the first assumption above is hardly acceptable. In most cases, it is not easy to obtain an accurate measure of acceleration; the robustness of the above adaptive control scheme with

respect to such a disturbance has to be established. Moreover, from a pure theoretical viewpoint, measuring q, \dot{q}, \ddot{q} means that not only do we need the whole system state vector, but we also need its derivative.

8.7.2 Adaptive Inertia-Related Control

Another adaptive control scheme is now introduced. This proposed scheme does not require the measurement of the manipulator's acceleration nor does it require inversion of the estimated inertia matrix. Hence, the drawbacks of the adaptive computed-torque control scheme are avoided. Let us consider the control input

$$\tau = \hat{\mathbf{H}}(q)\dot{v} + \hat{\mathbf{C}}(q, \dot{q})v + \hat{\tau}_g(q) + \mathbf{K}_D s, \quad (8.88)$$

where the auxiliary signals v and s are defined as $v = \dot{q}_d + \mathbf{\Lambda}e_q$ and $s = v - \dot{q} = \dot{e}_q + \mathbf{\Lambda}e_q$, with $\mathbf{\Lambda}$ being an $(n \times n)$ positive-definite matrix. Following Property 8.5 of the dynamic model, we have

$$\mathbf{H}(q)\dot{v} + \mathbf{C}(q, \dot{q})v + \tau_g(q) = \bar{\mathbf{Y}}(q, \dot{q}, v, \dot{v})a, \quad (8.89)$$

where $\bar{\mathbf{Y}}(\cdot, \cdot, \cdot, \cdot)$ is an $(n \times r)$ matrix of known time functions. The formulation above is the same type of the parameter separation that was used in the formulation of the adaptive computed-torque control. Note that $\bar{\mathbf{Y}}(q, \dot{q}, v, \dot{v})$ is independent of the joint acceleration. Similar to the formulation above, we also have

$$\hat{\mathbf{H}}(q)\dot{v} + \hat{\mathbf{C}}(q, \dot{q})v + \hat{\tau}_g(q) = \bar{\mathbf{Y}}(q, \dot{q}, v, \dot{v})\hat{a}. \quad (8.90)$$

Substituting the control input into the equation of motion, it follows that

$$\begin{aligned} & \mathbf{H}(q)\ddot{q} + \mathbf{C}(q, \dot{q})\dot{q} + \tau_g(q) \\ &= \hat{\mathbf{H}}(q)\dot{v} + \hat{\mathbf{C}}(q, \dot{q})v + \hat{\tau}_g(q) + \mathbf{K}_D s. \end{aligned}$$

Since $\ddot{q} = \dot{v} - \dot{s}$, $\dot{q} = v - s$, the previous result can be rewritten as

$$\mathbf{H}(q)\dot{s} + \mathbf{C}(q, \dot{q})s + \mathbf{K}_D s = \bar{\mathbf{Y}}(q, \dot{q}, v, \dot{v})\tilde{a}, \quad (8.91)$$

where $\tilde{a} = a - \hat{a}$. The adaptive law is considered as

$$\dot{\hat{a}} = \mathbf{\Gamma}\bar{\mathbf{Y}}^T(q, \dot{q}, v, \dot{v})s. \quad (8.92)$$

The convergence of the tracking error to zero with boundedness on all internal signals can be shown

through Lyapunov stability theory using the following Lyapunov-like function

$$V = \frac{1}{2} \mathbf{s}^T \mathbf{H}(\mathbf{q}) \mathbf{s} + \frac{1}{2} \tilde{\mathbf{a}}^T \mathbf{\Gamma}^{-1} \tilde{\mathbf{a}}, \quad (8.93)$$

whose time derivative along the trajectories of the closed-loop systems can be found to be

$$\dot{V} = -\mathbf{s}^T \mathbf{K}_D \mathbf{s}. \quad (8.94)$$

The detailed stability analysis is given in [8.32].

Remark 8.6

- The restrictions for adaptive computed-torque control formerly seen have been removed here.
- The term $\mathbf{K}_D \mathbf{s}$ introduces a PD-type linear stabilizing control action to the error system model.
- The estimated parameters converge to the true parameters provided the reference trajectory satisfies the condition of persistency of excitation,

$$\begin{aligned} \alpha_1 \mathbf{I}_r &\leq \int_{t_0}^{t_0+t} \mathbf{Y}^T(\mathbf{q}_d, \dot{\mathbf{q}}_d, \mathbf{v}, \dot{\mathbf{v}}) \mathbf{Y}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \mathbf{v}, \dot{\mathbf{v}}) dt \\ &\leq \alpha_2 \mathbf{I}_r, \end{aligned}$$

for all t_0 , where α_1 , α_2 , and t are all positive constants.

8.7.3 Adaptive Control Based on Passivity

Taking a physics viewpoint of control, we see that the concept of passivity has become popular for the development of adaptive control schemes. Here, we illustrate how the concept of passivity can be used to design a class of adaptive control laws for robot manipulators. First, we define an auxiliary filtered tracking error signal \mathbf{r} as

$$\mathbf{r} = \mathbf{F}^{-1}(s) \mathbf{e}_q, \quad (8.95)$$

where

$$\mathbf{F}^{-1}(s) = \left[s \mathbf{I}_n + \frac{1}{s} \mathbf{K}(s) \right], \quad (8.96)$$

and s is the Laplace transform variable. The $(n \times n)$ matrix $\mathbf{K}(s)$ is chosen such that $\mathbf{F}(s)$ is a strictly proper, stable transfer function matrix. As in the preceding schemes, the adaptive control strategies has close ties to the ability to separate the known functions from the

unknown constant parameters. We use the expression given above to define

$$\begin{aligned} \mathbf{Z}\boldsymbol{\varphi} &= \mathbf{H}(\mathbf{q})[\ddot{\mathbf{q}}_d + \mathbf{K}(s)\mathbf{e}_q] \\ &\quad + \mathbf{V}(\mathbf{q}, \dot{\mathbf{q}}) \left[\dot{\mathbf{q}}_n + \frac{1}{s} \mathbf{K}(s)\mathbf{e}_q \right] + \boldsymbol{\tau}_g(\mathbf{q}), \end{aligned}$$

where \mathbf{Z} is a known $(n \times r)$ regression matrix and $\boldsymbol{\varphi}$ is a vector of unknown system parameters in the adaptive context. It is important to note that the above can be arranged such that \mathbf{Z} and \mathbf{r} do not depend on the measurement of the joint acceleration $\ddot{\mathbf{q}}$. The adaptive control scheme given here is called the passivity approach because the mapping of $-\mathbf{r} \rightarrow \mathbf{Z}\tilde{\boldsymbol{\varphi}}$ is constructed to be a passive mapping. That is, we develop an adaptive law such that

$$\int_0^t -\mathbf{r}^T(\sigma) \mathbf{Z}(\sigma) \tilde{\boldsymbol{\varphi}}(\sigma) d\sigma \geq -\beta \quad (8.97)$$

is satisfied for all time and for some positive scalar constant β . For this class of adaptive controllers, the control input is given by

$$\boldsymbol{\tau} = \mathbf{Z}\hat{\boldsymbol{\varphi}} + \mathbf{K}_D \mathbf{r}, \quad (8.98)$$

Detailed analysis of stability is given in [8.44].

Remarks 8.4

- If $\mathbf{K}(s)$ is selected such that $\mathbf{H}(s)$ has a relative degree of one, \mathbf{Z} and \mathbf{r} will not depend on $\ddot{\mathbf{q}}$.
- Many types of control schemes can be generated from the adaptive passive control approach by selected different transfer function matrices $\mathbf{K}(s)$ in the definition of \mathbf{r} .
- Note that, by defining $\mathbf{K}(s) = s\boldsymbol{\Lambda}$ such that $\mathbf{F}(s) = (s\mathbf{I}_n + \boldsymbol{\Lambda})^{-1}$, we have the control input

$$\boldsymbol{\tau} = \mathbf{Z}\hat{\boldsymbol{\varphi}} - \mathbf{K}_D \mathbf{r},$$

with

$$\begin{aligned} \mathbf{Z}\hat{\boldsymbol{\varphi}} &= \hat{\mathbf{H}}(\mathbf{q})(\ddot{\mathbf{q}}_d + \boldsymbol{\Lambda}\dot{\mathbf{e}}_q) + \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})(\dot{\mathbf{q}}_d + \boldsymbol{\Lambda}\mathbf{e}_q) \\ &\quad + \hat{\boldsymbol{\tau}}_g(\mathbf{q}). \end{aligned}$$

The adaptive law may be chosen as

$$\dot{\hat{\boldsymbol{\varphi}}} = \boldsymbol{\Gamma} \mathbf{Z}^T (\dot{\mathbf{e}}_q + \boldsymbol{\Lambda}\mathbf{e}_q)$$

to satisfy the condition of passive mapping. This indicates that adaptive inertia-related control can be viewed as a special case of adaptive passive control.

8.7.4 Adaptive Control with Desired Compensation

In order to implement the adaptive control scheme, one needs to calculate the elements of $\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ in real time. However, this procedure may be excessively time consuming since it involves computations with highly nonlinear functions of joint position and velocity. Consequently, the real-time implementation of such a scheme is rather difficult. To overcome this difficulty, the adaptive control with desired compensation was proposed and is discussed here. In other words, the variables \mathbf{q} , $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ are replaced with the desired ones, namely, \mathbf{q}_d , $\dot{\mathbf{q}}_d$, and $\ddot{\mathbf{q}}_d$. Since the desired quantities are known in advance, all their corresponding calculations can be performed offline, which renders real-time implementation more plausible. Let us consider the control input

$$\boldsymbol{\tau} = \mathbf{Y}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)\hat{\mathbf{a}} + k_a \mathbf{s} + k_p \mathbf{e}_q + k_n \|\mathbf{e}_q\|^2 \mathbf{s}, \quad (8.99)$$

where the positive constants k_a , k_p , and k_n are sufficiently large, and the auxiliary signal \mathbf{s} is defined as $\mathbf{s} = \dot{\mathbf{e}}_q + \mathbf{e}_q$. The adaptive law is considered as

$$\dot{\hat{\mathbf{a}}} = -\boldsymbol{\Gamma} \mathbf{Y}^T(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d) \mathbf{s}. \quad (8.100)$$

It is worth noting that the desired compensation is adopted in both the control and adaptive laws such that the computational load can be drastically reduced. For the sake of this analysis, we note that

$$\begin{aligned} & \|\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\mathbf{a} - \mathbf{Y}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)\hat{\mathbf{a}}\| \\ & \leq \zeta_1 \|\mathbf{e}_q\| + \zeta_2 \|\mathbf{e}_q\|^2 + \zeta_3 \|\mathbf{s}\| + \zeta_4 \|\mathbf{s}\| \|\mathbf{e}_q\|, \end{aligned}$$

where ζ_1 , ζ_2 , ζ_3 , and ζ_4 are positive constants. In order to achieve trajectory tracking, it is required that

$$\begin{aligned} k_a & > \zeta_2 + \zeta_4, \\ k_p & > \frac{\zeta_1}{2} + \frac{\zeta_2}{4}, \\ k_v & > \frac{\zeta_1}{2} + \zeta_3 + \frac{\zeta_2}{4}, \end{aligned}$$

(i.e., the gains k_a , k_p , and k_v must be sufficiently large). The convergence of the tracking error to zero with boundedness on all internal signals can be proved through application of Lyapunov stability theory with the following Lyapunov-like function

$$V = \frac{1}{2} \mathbf{s}^T \mathbf{H}(\mathbf{q}) \mathbf{s} + \frac{1}{2} k_p \mathbf{e}_q^T \mathbf{e}_q + \frac{1}{2} \tilde{\mathbf{a}}^T \boldsymbol{\Gamma}^{-1} \tilde{\mathbf{a}}, \quad (8.101)$$

whose time derivative along the trajectories of the closed-loop system can be derived as

$$\dot{V} \leq -\mathbf{x}^T \mathbf{Q} \mathbf{x}, \quad (8.102)$$

where

$$\mathbf{x} = \begin{pmatrix} \|\mathbf{e}_q\| \\ \|\mathbf{s}\| \end{pmatrix}, \quad \mathbf{Q} = \begin{pmatrix} \frac{k_p - \zeta_2}{4} & \frac{-\zeta_1}{2} \\ \frac{-\zeta_1}{2} & \frac{k_v - \zeta_3 - \zeta_4}{4} \end{pmatrix}.$$

Detailed stability analysis can be found in [8.45].

8.7.5 Summary

Since the computed-torque control suffers from parametric uncertainty, a variety of adaptive control schemes have been proposed. Firstly, we have presented an adaptive control scheme based on computed-torque control. Then, in order to overcome the mentioned drawbacks such as the measurability of the joint acceleration and the invertibility of the estimated inertia matrix, we presented an alternative adaptive control scheme that is free of these drawbacks. Recently, to incorporate a physics viewpoint into control, adaptive passivity-based control has become popular, and hence is introduced and discussed here. Finally, to reduce the computational load of the adaptive schemes, we presented an adaptive control with desired compensation.

Further Reading

A computationally very fast scheme dealing with adaptive control of rigid manipulators was presented in [8.46]. The stability analysis was completed by assuming that the joint dynamics are decoupled, i.e., that each joint is considered as an independent second-order linear system. A decentralized high-order adaptive variable-structure control is discussed in [8.47], the proposed scheme makes both position and velocity tracking errors of robot manipulators globally converge to zero asymptotically while allowing all signals in closed-loop systems to be bounded without the information of manipulator parameters. Other pioneering works in the field can be found, for example, in [8.48, 49]; although none of the fundamental dynamic model properties are used, the complete dynamics are taken into account, but the control input is discontinuous and may lead to chattering. Positive definiteness of the inertia matrix is explicitly used in [8.50], although it was assumed that some time-varying quantities remain constant during the adaptation. It is interesting to note that all of these schemes were based on the concept of model reference adaptive control (MRAC) developed in [8.51] for linear systems. Therefore, they are conceptually very different from the truly nonlinear schemes presented in this section.

A passive-based modified version of the least-squares estimation scheme has been proposed in [8.52] and [8.53], which guaranteed closed-loop stability of the scheme. Other schemes can be found in [8.54], where no use is made of the skew-symmetry property, and in [8.55], where the recursive Newton–Euler formulations is used instead of the Lagrange formulation to derive the manipulator dynamics, and thus computation is simplified to facilitate practical implementation.

Even though adaptive control provides a solution to the problem of parametric uncertainty, the robustness

of adaptive controllers remains a topic of great interest in the field. Indeed, measurement noise or unmodeled dynamics (e.g., flexibility) may result in unbounded closed-loop signals. In particular, the estimated parameters may diverge; this is a well-known phenomenon in adaptive control and is called parameter drift. Solutions inspired from the adaptive control of linear systems have been studied [8.56, 57], where a modified estimation ensures boundedness of the estimates. In [8.58], the controller in [8.32] is modified to enhance robustness.

8.8 Optimal and Robust Control

Given a nonlinear system, such as robotic manipulators, one can develop many stabilizing controls [8.29, 41]. In other words, the stability of the control system cannot determine a unique controller. It is natural that one seeks an optimal controller among the many stable ones. However, the design of an optimal controller is possible provided that a rather exact information on the target system is available, such as an exact system model [8.34, 59]. In the presence of **discrepancy between the real system and its mathematical model**, a designed optimal controller is no longer optimal, and may even end up being instable in the actual system. **Generally speaking, the optimal control design framework is not the best one to deal with system uncertainty.** To handle system uncertainty from the control design stage, a robust control design framework is necessary [8.60]. One of main objectives of robust control is to keep the controlled system stable even in presence of uncertainties in the mathematical model, unmodeled dynamics, and the like.

Let us consider an affine nonlinear system described by nonlinear time-varying differential equation in the state $\mathbf{x} = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t) + \mathbf{G}(\mathbf{x}, t)\mathbf{u} + \mathbf{P}(\mathbf{x}, t)\mathbf{w}, \quad (8.103)$$

where $\mathbf{u} \in \mathbb{R}^m$ is the control input, and $\mathbf{w} \in \mathbb{R}^w$ is the disturbance. Without disturbances or unmodeled dynamics, the system simplifies to

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t) + \mathbf{G}(\mathbf{x}, t)\mathbf{u}. \quad (8.104)$$

Actually, there are many kinds of methods describing the nonlinear system according to the objective of control [8.1, 16, 21, 23, 34, 54].

8.8.1 Quadratic Optimal Control

Every optimal controller is based on its own cost function [8.61, 62]. One can define a cost function as [8.63,

64]

$$z = \mathbf{H}(\mathbf{x}, t)\mathbf{x} + \mathbf{K}(\mathbf{x}, t)\mathbf{u},$$

such that $\mathbf{H}^T(\mathbf{x}, t)\mathbf{K}(\mathbf{x}, t) = \mathbf{0}$, $\mathbf{K}^T(\mathbf{x}, t)\mathbf{K}(\mathbf{x}, t) = \mathbf{R}(\mathbf{x}, t) > \mathbf{0}$, and $\mathbf{H}^T(\mathbf{x}, t)\mathbf{H}(\mathbf{x}, t) = \mathbf{Q}(\mathbf{x}, t) > \mathbf{0}$. Then, we have

$$\frac{1}{2}\mathbf{z}^T\mathbf{z} = \frac{1}{2}\mathbf{x}^T\mathbf{Q}(\mathbf{x}, t)\mathbf{x} + \frac{1}{2}\mathbf{u}^T\mathbf{R}(\mathbf{x}, t)\mathbf{u}.$$

The quadratic optimal control for the system (8.104) is found by solving, for a first order differentiable positive-definite function $V(\mathbf{x}, t)$, the Hamilton–Jacobi–Bellman (HJB) equation [8.34, 59]

$$\begin{aligned} 0 = \text{HJB}(\mathbf{x}, t; V) &= V_t(\mathbf{x}, t) + \mathbf{V}_x(\mathbf{x}, t)\mathbf{f}(\mathbf{x}, t) \\ &\quad - \frac{1}{2}\mathbf{V}_x(\mathbf{x}, t)\mathbf{G}(\mathbf{x}, t)\mathbf{R}^{-1}(\mathbf{x}, t)\mathbf{G}^T(\mathbf{x}, t)\mathbf{V}_x^T(\mathbf{x}, t) \\ &\quad + \frac{1}{2}\mathbf{Q}(\mathbf{x}, t), \end{aligned}$$

where $V_t = \frac{\partial V}{\partial t}$ and $\mathbf{V}_x = \frac{\partial V}{\partial \mathbf{x}^T}$. Then the quadratic optimal control is defined by

$$\mathbf{u} = -\mathbf{R}^{-1}(\mathbf{x}, t)\mathbf{G}^T(\mathbf{x}, t)\mathbf{V}_x^T(\mathbf{x}, t). \quad (8.105)$$

Note that the HJB equation is a nonlinear second-order partial differential equation in $V(\mathbf{x}, t)$.

Unlike the aforementioned optimal control problem, the so-called inverse quadratic optimal control problem is to find a set of $\mathbf{Q}(\mathbf{x}, t)$ and $\mathbf{R}(\mathbf{x}, t)$ for which the HJB equation has a solution $V(\mathbf{x}, t)$. Then the inverse quadratic optimal control is defined by (8.105).

8.8.2 Nonlinear \mathcal{H}_∞ Control

When disturbances are not negligible, one can deal with their effect such that

$$\int_0^t \mathbf{z}^T(\mathbf{x}, \tau)\mathbf{z}(\mathbf{x}, \tau) d\tau \leq \gamma^2 \int_0^t \mathbf{w}^T\mathbf{w} d\tau, \quad (8.106)$$

where $\gamma > 0$ specifies the L_2 -gain of the closed-loop system from the disturbance input w to the cost variable z . This is called the L_2 -gain attenuation requirement [8.63–65]. One systematic way to design an optimal and robust control is given by the nonlinear \mathcal{H}_∞ optimal control. Let $\gamma > 0$ be given, by solving the following equation

$$\begin{aligned} \text{HJI}_\gamma(\mathbf{x}, t; V) &= V_t(\mathbf{x}, t) + \mathbf{V}_x(\mathbf{x}, t)\mathbf{f}(\mathbf{x}, t) \\ &\quad - \frac{1}{2}\mathbf{V}_x(\mathbf{x}, t)\{\mathbf{G}(\mathbf{x}, t)\mathbf{R}^{-1}(\mathbf{x}, t)\mathbf{G}^T(\mathbf{x}, t) \\ &\quad - \gamma^{-2}\mathbf{P}(\mathbf{x}, t)\mathbf{P}^T(\mathbf{x}, t)\}\mathbf{V}_x^T(\mathbf{x}, t) \\ &\quad + \frac{1}{2}\mathbf{Q}(\mathbf{x}, t) \leq 0, \end{aligned} \quad (8.107)$$

then the control is defined by

$$\mathbf{u} = -\mathbf{R}^{-1}(\mathbf{x}, t)\mathbf{G}^T(\mathbf{x}, t)\mathbf{V}_x^T(\mathbf{x}, t). \quad (8.108)$$

The partial differential inequality (8.107) is called the Hamilton–Jacobi–Isaac (HJI) inequality. Then one can define the inverse nonlinear \mathcal{H}_∞ optimal control problem that finds a set of $\mathbf{Q}(\mathbf{x}, t)$ and $\mathbf{R}(\mathbf{x}, t)$ such that the L_2 -gain requirement is achieved for a prescribed L_2 -gain γ [8.66].

Two things deserve further comment. The first is that the L_2 -gain requirement is only valid for disturbance signals w whose L_2 -norm is bounded. The second is that \mathcal{H}_∞ optimal control is not uniquely defined. Hence, one can choose a quadratic optimal among many \mathcal{H}_∞ optimal controllers. Precisely speaking, the control (8.108) should be called \mathcal{H}_∞ suboptimal control, since the desired L_2 -gain is prescribed a priori. A true \mathcal{H}_∞ optimal control is to find a minimal value of γ , such that the L_2 -gain requirement is achieved.

8.8.3 Passivity-Based Design of Nonlinear \mathcal{H}_∞ Control

There are many methodologies for the design of optimal and/or robust controls. Among these, passivity-based controls can take full advantage of the properties described above [8.31]. They consist of two parts: one coming from the reference motion compensation while preserving the passivity of the system, and the other to achieve stability, robustness, and/or optimality [8.66, 67].

Let us suppose that the dynamic parameters are identified as $\hat{\mathbf{H}}(\mathbf{q})$, $\hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})$, and $\hat{\boldsymbol{\tau}}_g(\mathbf{q})$, whose counterparts are $\mathbf{H}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$, and $\boldsymbol{\tau}_g(\mathbf{q})$, respectively. Then, passivity-based control generates the following tracking control laws

$$\boldsymbol{\tau} = \hat{\mathbf{H}}(\mathbf{q})\ddot{\mathbf{q}}_{\text{ref}} + \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_{\text{ref}} + \hat{\boldsymbol{\tau}}_g(\mathbf{q}) - \mathbf{u}, \quad (8.109)$$

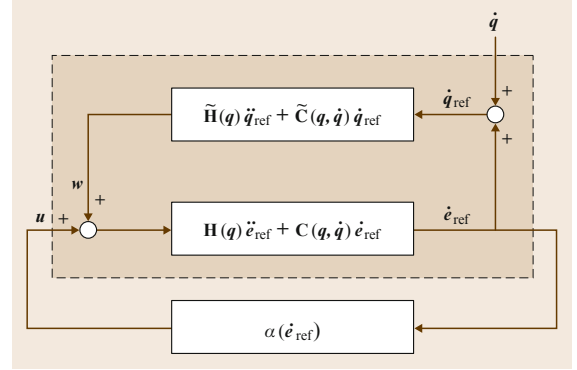


Fig. 8.7 The closed-loop system according to (8.111)

where $\ddot{\mathbf{q}}_{\text{ref}}$ is the reference acceleration defined by

$$\ddot{\mathbf{q}}_{\text{ref}} = \ddot{\mathbf{q}}_d + \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q, \quad (8.110)$$

where $\mathbf{K}_V = \text{diag}\{k_{V,i}\} > 0$ and $\mathbf{K}_P = \text{diag}\{k_{P,i}\} > 0$. Two parameters are involved in generating the reference acceleration. Sometimes the following alternative method can be adopted

$$\ddot{\mathbf{q}}_{\text{ref}} = \ddot{\mathbf{q}}_d + \mathbf{K}_V \dot{\mathbf{e}}_q.$$

This reduces the order of the closed-loop system because the state $\mathbf{x} = (\mathbf{e}_q^T, \dot{\mathbf{e}}_q^T)^T$ is sufficient for the system description, while the definition of (8.110) requires the state $\mathbf{x} = (\int \mathbf{e}_q^T, \mathbf{e}_q^T, \dot{\mathbf{e}}_q^T)^T$.

In Fig. 8.7, the closed-loop dynamics under the control is given by

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{e}}_{\text{ref}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{e}}_{\text{ref}} = \mathbf{u} + \mathbf{w}, \quad (8.111)$$

where

$$\begin{aligned} \ddot{\mathbf{e}}_{\text{ref}} &= \ddot{\mathbf{e}}_q + \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q, \\ \dot{\mathbf{e}}_{\text{ref}} &= \dot{\mathbf{e}}_q + \mathbf{K}_V \mathbf{e}_q + \mathbf{K}_P \int \mathbf{e}_q. \end{aligned}$$

If $\mathbf{d}(t) = 0$ and $\hat{\mathbf{H}} = \mathbf{H}$, $\hat{\mathbf{C}} = \mathbf{C}$, $\hat{\boldsymbol{\tau}}_g = \boldsymbol{\tau}_g$, then $\mathbf{w} = 0$. Otherwise, the disturbance is defined as

$$\mathbf{w} = \tilde{\mathbf{H}}(\mathbf{q})\ddot{\mathbf{q}}_{\text{ref}} + \tilde{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_{\text{ref}} + \tilde{\boldsymbol{\tau}}_g(\mathbf{q}) + \mathbf{d}(t), \quad (8.112)$$

where $\tilde{\mathbf{H}} = \mathbf{H} - \hat{\mathbf{H}}$, $\tilde{\mathbf{C}} = \mathbf{C} - \hat{\mathbf{C}}$, and $\tilde{\boldsymbol{\tau}}_g = \boldsymbol{\tau}_g - \hat{\boldsymbol{\tau}}_g$. It is of particular interest that the system (8.111) defines a passive mapping between $\mathbf{u} + \mathbf{w}$ and $\dot{\mathbf{e}}_{\text{ref}}$.

According to the manner in which the auxiliary control input \mathbf{u} is specified, passivity-based control can achieve stability, robustness, and/or optimality (Fig. 8.7).

8.8.4 A Solution to Inverse Nonlinear \mathcal{H}_∞ Control

Let us define the auxiliary control input by the reference-error feedback

$$\mathbf{u} = -\alpha \mathbf{R}^{-1}(\mathbf{x}, t) \dot{\mathbf{e}}_{\text{ref}}, \quad (8.113)$$

where $\alpha > 1$ is arbitrary. Then, the control provides the inverse nonlinear \mathcal{H}_∞ optimality.

Theorem 8.1 Inverse Nonlinear \mathcal{H}_∞ Optimality [8.66]

Let the reference acceleration generation gain matrices \mathbf{K}_V and \mathbf{K}_P satisfy

$$\mathbf{K}_V^2 > 2\mathbf{K}_P. \quad (8.114)$$

Then for a given $\gamma > 0$, the reference error feedback

$$\mathbf{u} = -\mathbf{K} \dot{\mathbf{e}}_{\text{ref}} = -\mathbf{K} \left(\dot{\mathbf{e}}_q + \mathbf{K}_V \mathbf{e}_q + \mathbf{K}_P \int \mathbf{e}_q \right) \quad (8.115)$$

satisfies the L_2 -gain attenuation requirement for

$$\mathbf{Q} = \begin{pmatrix} \mathbf{K}_P^2 \mathbf{K}_\gamma & 0 & 0 \\ 0 & (\mathbf{K}_V^2 - 2\mathbf{K}_P) \mathbf{K}_\gamma & 0 \\ 0 & 0 & \mathbf{K}_\gamma \end{pmatrix}, \quad (8.116)$$

$$\mathbf{R} = \mathbf{K}^{-1}, \quad (8.117)$$

provided that

$$\mathbf{K}_\gamma = \mathbf{K} - \frac{1}{\gamma^2} \mathbf{I} > 0. \quad (8.118)$$

Given γ , one can set $\mathbf{K} = \alpha \frac{1}{\gamma^2} \mathbf{I}$ for $\alpha > 1$. This yields $\mathbf{K}_\gamma = (\alpha - 1) \frac{1}{\gamma^2} \mathbf{I}$.

When the inertia matrix is identified as a diagonal constant matrix such as $\hat{\mathbf{H}} = \text{diag}\{\hat{m}_i\}$, one should set $\hat{\mathbf{C}} = \mathbf{0}$. In addition, one can set $\hat{\boldsymbol{\tau}}_g = \mathbf{0}$. Then this results in a decoupled PID control of the form

$$\tau_i = \hat{m}_i (\ddot{q}_{d,i} + k_{V,i} \dot{e}_{q,i} + k_{P,i} e_{q,i}) + \alpha \frac{1}{\gamma^2} \left(\dot{e}_{q,i} + k_{V,i} e_{q,i} + k_{P,i} \int e_{q,i} \right)$$

for $\alpha > 1$, which can be rewritten as

$$\tau_i = \hat{m}_i \ddot{q}_{d,i} + \left(\hat{m}_i k_{V,i} + \alpha \frac{1}{\gamma^2} \right) \dot{e}_{q,i} + \left(\hat{m}_i k_{P,i} + \alpha \frac{k_{V,i}}{\gamma^2} \right) e_{q,i} + \alpha \frac{k_{P,i}}{\gamma^2} \int e_{q,i}. \quad (8.119)$$

This leads to a PID control with the desired acceleration feedforward [8.68] given by

$$\tau_i = \hat{m}_i \ddot{q}_{d,i} + k_{V,i}^* \dot{e}_{q,i} + k_{P,i}^* e_{q,i} + k_{I,i}^* \int e_{q,i}, \quad (8.120a)$$

where

$$k_{V,i}^* = \hat{m}_i k_{V,i} + \alpha \frac{1}{\gamma^2}, \quad (8.120b)$$

$$k_{P,i}^* = \hat{m}_i k_{P,i} + \alpha \frac{k_{V,i}}{\gamma^2}, \quad (8.120c)$$

$$k_{I,i}^* = \alpha \frac{k_{P,i}}{\gamma^2}. \quad (8.120d)$$

8.9 Trajectory Generation and Planning

This section deals with the problem of *reference trajectory generation*, that is, the computation of desired position, velocity, acceleration and/or force/torque signals that are used as input values for the robot motion controllers introduced in Sects 8.3–8.8.

8.9.1 Geometric Paths and Trajectories

Path Planning

A path is a geometric representation of a plan to move from a start to a target pose. The task of planning is to find a collision-free path among a collection of

static and dynamic obstacles. Path planning can also include the consideration of dynamic constraints such as workspace boundaries, maximum velocities, maximum accelerations, and maximum jerks. We distinguish between *online* and *offline* path planning algorithms. Offline planned paths are static and calculated prior to execution. Online methods require algorithms that meet real-time constraints (i. e., algorithms that do not exceed a determinable worst-case computation time) to enable path (re-)calculations and/or adaptations during the robot motions in order to react to and interact with dynamic environments. This means that a robot

moves along a path that has not necessarily been computed completely, and which may change during the movement. Details about path planning concepts are described in Chap. 7, in Parts D and E, and specifically in Chap. 47.

Trajectory Planning

A trajectory is more than a path: It also includes velocities, accelerations, and/or jerks along a path (VIDEO 760). A common method is computing trajectories for a priori specified paths, which fulfill a certain criterion (e.g., minimum execution time). We distinguish between *online* and *offline* trajectory planning methods. An offline calculated trajectory cannot be influenced during its execution, while online trajectory planning methods can (re-)calculate and/or adapt robot motions behavior during the movement. The reasons for this (re-)calculation and/or adaptation can vary: improvement of accuracy, better utilization of currently available dynamics, reaction to and interaction with a dynamic environment, or reaction to (sensor) events. Besides the distinction between online and offline methods, we can further distinguish between (1) one-dimensional (1-D) and multi-dimensional trajectories and (2) single-point and multi-point trajectories. Multi-point trajectories typically relate to a path.

8.9.2 Joint Space and Operational Space Trajectories

Depending on the control state space, trajectory generators provide set points for tracking controllers in joint space or in operational space. In either space, a trajectory can be represented in several different ways: cubic splines, quintic splines, higher-order splines, harmonics (sinusoidal functions), exponential functions, Fourier series, and more.

Joint Space Trajectories

Consider the torque control input (8.29)

$$\boldsymbol{\tau} = \mathbf{H}(\mathbf{q})\mathbf{v} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}),$$

and the PD controller (8.30)

$$\mathbf{v} = \ddot{\mathbf{q}}_d + \mathbf{K}_V(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{K}_P(\mathbf{q}_d - \mathbf{q}), \quad (8.121)$$

or PID controller (8.31), respectively, the task of a trajectory generator in joint space coordinates is computing the signals $\mathbf{q}_d(t)$, $\dot{\mathbf{q}}_d(t)$, and $\ddot{\mathbf{q}}_d(t)$. These three signals contain the reference trajectory and are used as input values for the tracking controller.

During nominal operation the joint torques required to execute a trajectory should not exceed joint

force/torque limits $\boldsymbol{\tau}_{\min}(t)$ and $\boldsymbol{\tau}_{\max}(t)$,

$$\boldsymbol{\tau}_{\min}(t) \leq \boldsymbol{\tau}(t) \leq \boldsymbol{\tau}_{\max}(t) \quad \forall t \in \mathbb{R}. \quad (8.122)$$

Operational Space Trajectories

Similarly to (8.29), we can also consider trajectories represented by \mathbf{x}_d , $\dot{\mathbf{x}}_d$, and $\ddot{\mathbf{x}}_d$ for an operational space controller (8.9)

$$\mathbf{f}_c = \boldsymbol{\Lambda}(\mathbf{q})\boldsymbol{\mu} + \boldsymbol{\Gamma}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \boldsymbol{\eta}(\mathbf{q}),$$

with a PD control law of

$$\boldsymbol{\mu} = \ddot{\mathbf{x}}_d + \mathbf{K}_V(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) + \mathbf{K}_P(\mathbf{x}_d - \mathbf{x}). \quad (8.123)$$

With the transformation into joint space using the inverse of (8.8), the operational space trajectory generator must assure that the limits given in (8.122) are not violated. It is the responsibility of the path planner (Chap. 7) that all points along the trajectory are in the robot workspace and that start and goal poses can be reached in the same joint configuration. It is the responsibility of the trajectory planner that joint torque and velocity constraints are not violated even in the presence of kinematic singularities.

8.9.3 Trajectory Representations

Mathematical Representations

Functions for $\ddot{\mathbf{q}}_d(t)$ (8.121) and $\ddot{\mathbf{x}}_d$ (8.123) can be represented in several ways that are described here.

Polynomial Trajectories. One of the simplest ways to represent a robot trajectory is a polynomial function of degree m for each joint $i \in \{1, \dots, n\}$

$$q_i(t) = a_{i,0} + a_{i,1}t + a_{i,2}t^2 + \dots + a_{i,m}t^m, \quad (8.124)$$

so that $\mathbf{q}(t)$ can be composed (or $\mathbf{x}(t)$ in operational space, respectively). In the simplest case, cubic polynomials are used, which, however, leads to non-steady acceleration signals with infinite jerks. Quintic and higher-order polynomials allow for steady acceleration signals as well as arbitrary position, velocity, and acceleration vectors and the beginning and at the end of the trajectory. To determine the coefficients $a_{i,j} \quad \forall (i, j) \in \{1, \dots, n\} \times \{0, \dots, m\}$ of (8.124), the execution time t_{trgt} , at which the target state will be reached needs to be known. The left part of Fig. 8.8a shows a quintic trajectory for three DOFs starting at $t_0 = 0$ s with an execution time of $t_{\text{trgt}} = 2.5$ s. To connect a trajectory segment to preceding and succeeding segments, the fol-

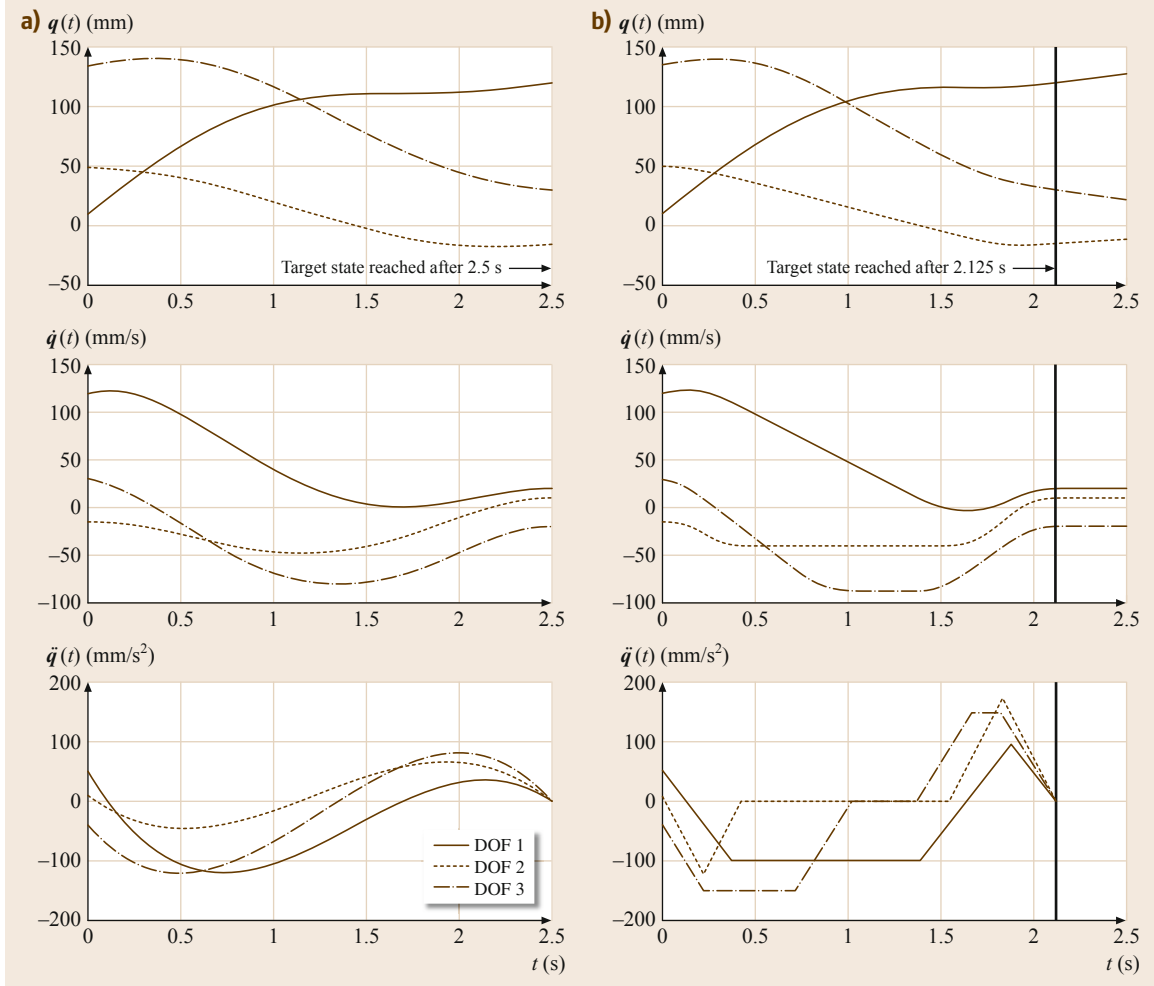


Fig. 8.8a,b Two sample trajectories for a three-DOF robot ($n = 3$). The trajectory in (a) is represented by quintic splines, and the trajectory in (b) by piecewise polynomials

lowing six constraints need to be satisfied for all n joints

$$\begin{aligned}
 q_i(t_0) &= q_{i,\text{start}} = a_{i,0} \\
 q_i(t_{\text{trgt}}) &= q_{i,\text{trgt}} = a_{i,0} + a_{i,1}t_{\text{trgt}} + a_{i,2}t_{\text{trgt}}^2 \\
 &\quad + a_{i,3}t_{\text{trgt}}^3 + a_{i,4}t_{\text{trgt}}^4 + a_{i,5}t_{\text{trgt}}^5 \\
 \dot{q}_i(t_0) &= \dot{q}_{i,0} = a_{i,1} \\
 \dot{q}_i(t_{\text{trgt}}) &= \dot{q}_{i,\text{trgt}} = a_{i,1} + 2a_{i,2}t_{\text{trgt}} + 3a_{i,3}t_{\text{trgt}}^2 \\
 &\quad + 4a_{i,4}t_{\text{trgt}}^3 + 5a_{i,5}t_{\text{trgt}}^4 \\
 \ddot{q}_i(t_0) &= \ddot{q}_{i,0} = 2a_{i,2} \\
 \ddot{q}_i(t_{\text{trgt}}) &= \ddot{q}_{i,\text{trgt}} = 2a_{i,2} + 6a_{i,3}t_{\text{trgt}} + 12a_{i,4}t_{\text{trgt}}^2 \\
 &\quad + 20a_{i,5}t_{\text{trgt}}^3.
 \end{aligned}$$

A unique closed-form solution can be computed, so that all polynomial coefficients $a_{i,j} \forall (i, j) \in$

$\{1, \dots, n\} \times \{0, \dots, 5\}$ can be determined for one trajectory segment.

Piecewise Polynomials. Polynomials of different degrees can be concatenated to represent a trajectory between an initial state $(q_0, \dot{q}_0, \ddot{q}_0)$ and a target state $(q_{\text{trgt}}, \dot{q}_{\text{trgt}}, \ddot{q}_{\text{trgt}})$. For instance, the classical double-S velocity profile [8.69] with trapezoidal acceleration and deceleration profiles and a cruise-velocity segment in the middle consists of seven polynomials of degrees 3-2-3-1-3-2-3 (m in (8.124)). The right part of Fig. 8.8 shows a trajectory represented by piecewise polynomials. To compute time-optimal trajectories under purely kinematic constraints (e.g., $\dot{q}_{\text{max}}, \ddot{q}_{\text{max}}, \ddot{q}_{\text{min}}$, etc.), piecewise polynomials are used, because they allow always using one signal at its kinematic limit (Fig. 8.8b and [8.70]).

Trigonometric Trajectories. Similarly to (8.124), trigonometric functions can be used to represent harmonic, cycloidal, and elliptic trajectories [8.71, 72]. A simple example for a harmonic trajectory for one joint i is

$$q_i(t) = \frac{q_{i,\text{trgt}} - q_{i,0}}{2} \left(1 - \cos \frac{\pi (t - t_0)}{t_{\text{trgt}} - t_0} \right) + q_{i,0} . \quad (8.125)$$

While any order of derivatives of trigonometric functions is continuous, they might be discontinuous at t_0 and t_{trgt} .

Other Representations. *Exponential Trajectories* and *Fourier Series Expansions* [8.72] are particularly suited to minimize natural vibrations on robot mechanisms introduced by reference trajectories.

Trajectories and Paths

To draw the connection to Chap. 7 and Parts D and E, trajectories and paths are often tightly coupled.

Trajectories Along Predefined Paths. A path in joint space can be described by a function $\mathbf{q}(s(t))$ with $s \in [t_0, t_{\text{trgt}}]$, where the start configuration of the path is $\mathbf{q}(t_0)$ and the target configuration is $\mathbf{q}(t_{\text{trgt}})$. To move a robot along the path, an appropriate function $s(t)$ needs to be computed that does not violate any of the kinematic and dynamic constraints [8.73–75]. If a path is given in operational space, $\mathbf{x}(s(t))$ can be mapped to $\mathbf{q}(t)$ (Sect. 8.2).

Multi-Dimensional Trajectories. Instead of using a one-dimensional function $s(t)$ to parameterize a path segment, trajectories can also be described by individual functions for each DOF i to represent $\mathbf{q}(s(t))$ or $\mathbf{x}(s(t))$, respectively. To connect two arbitrary states, the signals for each individual degree of freedom need to be time-synchronized [8.70], so that all DOFs reach their target state of motion at the very same instant. Those trajectories may also be phase-synchronized [8.76], so that the trajectories of all DOFs are derived from a one master DOF and only scaled by a factor to achieve homothety [8.77]. The two trajectories in Fig. 8.8 are time-synchronized but not phase-synchronized.

Multi-Point Trajectories. If instead of an entirely defined geometric path or a motion to a single way point, an entire series of geometric way points is given, the trajectory that connects all way points in a given state space needs to be computed. Trajectory segments between two way points can be represented with any of

the above mentioned representations as long as the position signal and its derivatives up an appropriate order are continuous (at least C^1 continuous). Splines, B-splines, or Bezier splines are used to generate either a reference trajectory or a geometric path, which then can be parameterized with a function $s(t)$.

8.9.4 Trajectory Planning Algorithms

The following part provides an overview of online and offline trajectory planning concepts.

Constraints

Constraints for trajectory planners can manifold:

- **Kinematic:** maximum velocities, accelerations, jerks, etc. and workspace space limits
- **Dynamic:** maximum joint or actuator forces and/or torques
- **Geometric:** no collisions with static and dynamic objects in the workspace
- **Temporal:** reaching a state within a given time interval or at a given time.

These and other constraints may have to be taken into account at the same time. Depending on the robot and the task, additional optimization criteria may be considered (e.g., time-optimality, minimum-jerk, maximum distance to workspace boundaries, minimum energy).

Offline Trajectory Planning

Kahn and Roth [8.78] showed results using optimal, linear control theory to achieve a near-time-optimal solution for linearized manipulators. The resulting trajectories are jerk-limited and lead to smaller trajectory-following errors and to less excitation of structural natural frequencies in the system.

The work of *Brady* [8.79] introduced several techniques of trajectory planning in joint space and *Paul* [8.80] and *Taylor* [8.81] published works about the planning of trajectories in Cartesian space in parallel to *Brady*. *Lin et al.* [8.82] published another purely kinematic approach as did *Castain and Paul* [8.69].

Hollerbach [8.83] first introduced the consideration of the nonlinear inverse robot dynamics for the generation of manipulator trajectories.

During the middle of the 1980s, three groups developed techniques for time-optimal trajectory planning for arbitrarily specified paths: *Bobrow* [8.73], *Shin and McKay* [8.74], and *Pfeiffer and Johanni* [8.75]. Trajectories are composed of three curves: the maximum acceleration curve, the maximum velocity curve, and

the maximum deceleration curve. The proposed algorithms find the intersection points of these three curves.

These algorithms have become the fundament for many follow-up works: *Kyriakopoulos and Sridis* [8.84] added minimum-jerk criteria; *Slotine and Yang* abandoned the computationally expensive calculation of the maximum velocity curve [8.85]; *Shiller and Lu* added methods for handling dynamic singularities [8.86]; *Fiorini and Shiller* extended the algorithm for known dynamic environments with moving obstacles [8.87].

Online Trajectory Planning





An online modification of a planned trajectory may have several reasons: (i) The trajectory becomes adapted in order to improve the accuracy with a path specified beforehand; (ii) The robotic system reacts on sensor signals and/or events that cannot be predicted beforehand, because the robot acts in a (partly) unknown and dynamic environment.

Improving Path Accuracy. All previously described off-line trajectory planning methods assume a dynamic model that describes the behavior of the real robot exactly. In practice, this is often not the case, and some robot parameters are only estimated, some dynamic effects remain unmodeled, and system parameters may change during operation. If this is the case, the resulting robot motion is not time-optimal anymore and/or the maximum actuator forces and/or torques are exceeded, which leads to an undesired difference between the specified and the executed path.

Dahl and Nielsen [8.88] extended [8.73–75] by adapting the acceleration along the path, so that the underlying trajectory-following controller becomes adapted depending on the current state of motion. The

approaches of *Cao et al.* [8.89, 90] use cubic splines to generate smooth paths in joint space with time-optimal trajectories. *Constantinescu and Croft* [8.91] suggest a further improvement to the approach of [8.86] with objective to limit the derivative of actuator forces/torques. *Macfarlane and Croft* extended this approach further by adding jerk limitations to quintic splines (Fig. 8.8) [8.92].

Sensor-Based Trajectory Adaptation

The last paragraph presented an overview of online trajectory generation methods for improving the path accuracy, while this one focuses on the online consideration of sensor signals, for instance, for the purpose of collision avoidance ( VIDEO 757,  VIDEO 758) or switching between controllers or control gains ( VIDEO 759,  VIDEO 761).

In 1988, *Andersson* presented an online trajectory planning for a Ping-Pong-playing PUMA 260 manipulator that computes parameterized quintic polynomials [8.93, 94]. Based on [8.73–75], *Lloyd and Hayward* proposed a technique to transition between two different trajectory segments [8.95] using a transition window [8.81]. *Ahn et al.* introduced a method to connect two arbitrary motion states online, which does not take into account kinematic or dynamic constraints [8.96]. *Broquère et al.*, *Haschke et al.*, and *Kröger* extended this approach for multi-dimensional trajectories, so that kinematic constraints are taken into account [8.70, 97, 98].

Further Reading

Overviews of the domain of robot reference trajectory generation can be found in the textbooks of *Biagiotti and Melchiorri* [8.72], *Craig* [8.99], *Fu et al.* [8.100], and *Spong et al.* [8.101].

8.10 Digital Implementation

Most controllers introduced in the previous sections are digitally implemented on microprocessors. In this section basic but essential practical issues related to their computer implementation are discussed. When the controller is implemented on a computer control system, the analog inputs are read and the outputs are set with a certain sampling period. This is a drawback compared to analog implementations, since sampling introduces time delays into the control loop. Figure 8.9 shows the overall block diagram of control system with a boxed digital implementation part. When a digital computer is used to implement a control law, it is convenient to di-

vide coding sequence in the interrupt routine into four process routines, as shown in Fig. 8.10. Reading the input signal from sensors and writing the control signal to digital-to-analog (D/A) converters synchronized at the correct frequency is very important. Therefore, these processes are located in the first routine. After saving counter values and extracting D/A values, which are already calculated one step before, the next routine produces reference values. Control routines with filters follow and produce scalar or vector control outputs. Finally, the user interface for checking parameter values is made and will be used for tuning and debugging.

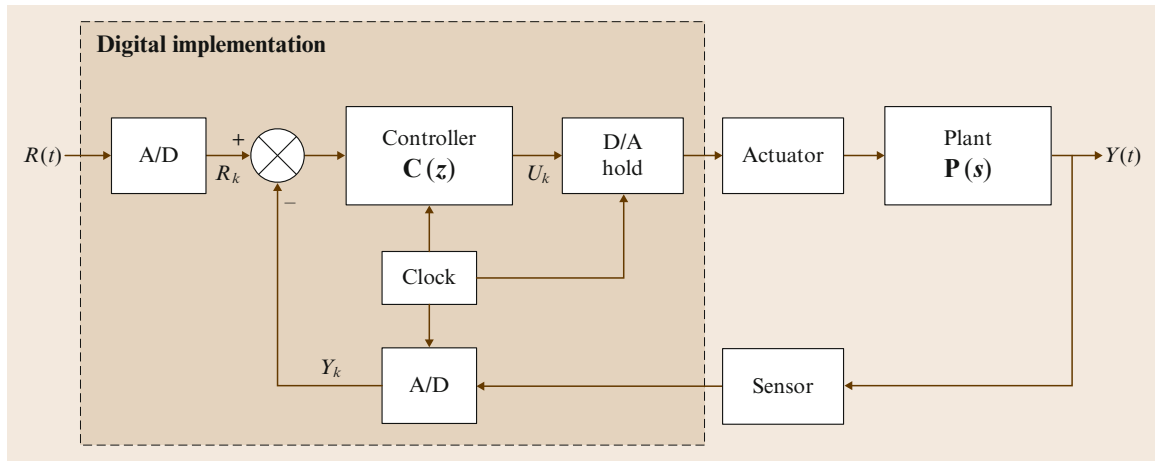


Fig. 8.9 Digital implementation of system control

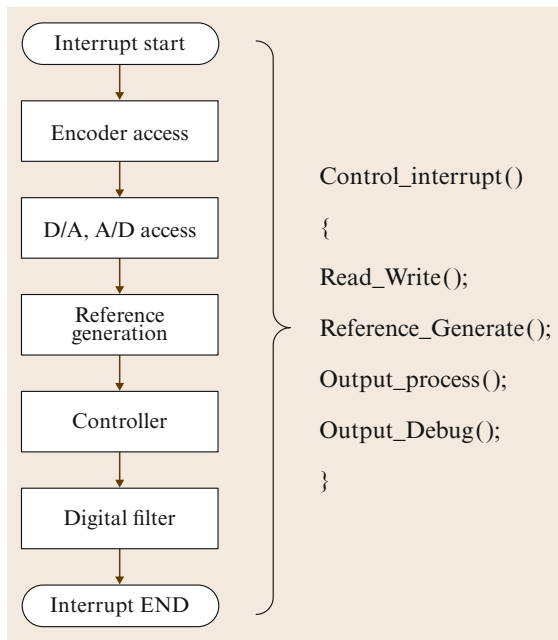


Fig. 8.10 The sequence in the interrupt routine for digital control

8.10.1 Z-Transform for Motion Control Implementation

Continuous-time systems are transformed into discrete-time systems by using the Z-transform. A discrete-time system is used to obtain a mathematical model that gives the behavior of a physical process at the sampling points, although the physical process is still a continuous-time system. A Laplace transform is used for the analysis of control system in the s -domain. In most cases, the design of controllers and filters are done

using tools in the s -domain. In order to realize those results in program code, understanding the Z-transform is essential. All controllers and filters designed in the s -domain can be easily translated to a program code through a Z-transform because it has the form of digitized difference sequences.

A PID controller is used as an example. In transfer function form, this controller has the basic structure

$$\frac{Y(s)}{E(s)} = K_P + \frac{K_I}{s} + sK_V. \quad (8.126)$$

There are several methods for transformation from the frequency domain to the discrete domain. For stability conservation, backward Euler and Tustin algorithms are often used. Though the Tustin algorithm is known as the more exact one, the backward Euler algorithm is utilized in the following procedure.

After substituting the backward Euler equation into (8.126),

$$s \cong \frac{1 - z^{-1}}{T},$$

the following discrete form is produced

$$\frac{Y(z)}{E(z)} = \frac{\alpha + \beta z^{-1} + \gamma z^{-2}}{T(1 - z^{-1})}, \quad (8.127)$$

where

$$\alpha = K_I T^2 + K_P T + K_V,$$

$$\beta = -K_P T - 2K_V,$$

$$\gamma = K_V.$$

Sometimes a differentiator s in the PID controller makes the implementation infeasible when the mea-

surement noise is severe. One can remedy the controller (8.126) by attaching a lowpass filter with filter time constant σ

$$\frac{Y(s)}{E(s)} = K_P + \frac{K_I}{s} + \frac{s}{\sigma s + 1} K_V. \quad (8.128)$$

Again, substituting the backward Euler equation into (8.128) produces

$$\frac{Y(z)}{E(z)} = \frac{\alpha + \beta z^{-1} + \gamma z^{-2}}{1 - \delta z^{-1} - \psi z^{-2}}, \quad (8.129)$$

where

$$\begin{aligned} \alpha &= K_P + K_I T + \frac{K_V}{\sigma + T}, \\ \beta &= -\frac{(2\sigma + T)K_P + \sigma T K_I + 2K_V}{\sigma + T}, \\ \gamma &= \frac{\sigma K_P + K_V}{\sigma + T}, \\ \delta &= \frac{2\sigma + T}{\sigma + T}, \\ \psi &= -\frac{\sigma}{\sigma + T}, \end{aligned}$$

in which the filter time constant σ is determined from a cutoff frequency f_c [Hz] for removing noises such as $\sigma = \frac{1}{2\pi f_c}$.

8.10.2 Digital Control for Coding

Inverse Z-transform produces a difference equation for digital control. Furthermore the difference equation can be directly converted into the control program code. Since the inverse Z-transform of $Y(z)$ is y_k and z^{-1} implies the previous sample time, $z^{-1}Y(z) = y_{k-1}$ and $z^{-2}Y(z) = y_{k-2}$.

Now, the PID controller expressed by (8.127) is rearranged using the difference equation

$$\begin{aligned} T(y_k - y_{k-1}) &= \alpha e_k + \beta e_{k-1} + \gamma e_{k-2}, \\ y_k - y_{k-1} &= \frac{1}{T}(\alpha e_k + \beta e_{k-1} + \gamma e_{k-2}). \end{aligned} \quad (8.130)$$

For practical use, the PID controller can be directly coded in the program as follows

$$y_k = K_{P,c} e_k + K_{V,c} e_k^v + K_{I,c} e_k^i,$$

where

$$\begin{aligned} e_k &= p_{k,\text{desired}} - p_k, \\ e_k^v &= v_{k,\text{desired}} - v_k \\ &= (p_{k,\text{desired}} - p_{k-1,\text{desired}}) - (p_k - p_{k-1}) \end{aligned}$$

$$= e_k - e_{k-1},$$

$$e_k^i = e_{k-1}^i + e_k = \sum_{j=0}^k e_j,$$

in which p_k is the present position, v_k is the present velocity, desired means reference to be followed, and c means coded form for digital control. Now let us obtain the difference between the present control output and the previous one

$$\begin{aligned} y_k - y_{k-1} &= [K_{P,c} e_k - K_{P,c} e_{k-1}] \\ &\quad + [K_{V,c} (e_k - e_{k-1}) - K_{V,c} (e_{k-1} - e_{k-2})] \\ &\quad + [K_{I,c} e_k^i - K_{I,c} e_{k-1}^i] \\ &= (K_{P,c} + K_{V,c} + K_{I,c}) e_k \\ &\quad - (K_{P,c} + 2K_{V,c}) e_{k-1} + K_{V,c} e_{k-2}. \end{aligned} \quad (8.131)$$

Comparing the parameters in (8.130) and (8.131), one obtains

$$\begin{aligned} \frac{\alpha}{T} &= K_P + \frac{K_V}{T} + K_I T = (K_{P,c} + K_{V,c} + K_{I,c}), \\ \frac{\beta}{T} &= -K_P - \frac{2K_V}{T} = -(K_{P,c} + 2K_{V,c}), \\ \frac{\gamma}{T} &= \frac{K_V}{T} = K_{V,c}, \end{aligned}$$

which shows that there is a relation between the designed and coded forms of the gains

$$\begin{aligned} K_{P,c} &= K_P, \\ K_{V,c} &= \frac{K_V}{T}, \\ K_{I,c} &= K_I T. \end{aligned} \quad (8.132)$$

As the sampling frequency is increased in the same system, the coded K_V gain should be increased and the coded K_I gain should be decreased. Using this method, the designed controller can be coded in a digital signal processor (DSP) or microprocessor. However, sufficient analysis and simulation for control algorithms should be performed beforehand to obtain successful control system performance.


In addition, the PID controller with lowpass filter (8.129) can be implemented as

$$\begin{aligned} y_k - \delta y_{k-1} - \psi y_{k-2} &= \alpha e_k + \beta e_{k-1} + \gamma e_{k-2}, \\ y_k &= \delta y_{k-1} + \psi y_{k-2} + \alpha e_k + \beta e_{k-1} + \gamma e_{k-2}. \end{aligned} \quad (8.133)$$

Using the same procedures, one can arrive at the similar control program code for digital control.

PID Control Experiment (Multimedia)

According as the gains change, the performance variations of PID controller implemented in the digital

control system are shown in the multimedia source to help readers' understanding  VIDEO 25.

8.11 Learning Control

Since many robotic applications, such as pick-and-place operations, paintings, and circuit-board assembly, involve repetitive motions, it is natural to consider the use of data gathered in previous cycles to try to improve the performance of the manipulator in subsequent cycles. This is the basic idea of repetitive control or learning control. Consider the robot model given in Sect. 8.1 and suppose that one is given a desired joint trajectory $\mathbf{q}_d(t)$ on a finite time interval $0 \leq t \leq T$. The reference trajectory \mathbf{q}_d is used in repeated trails of the manipulator, assuming either that the trajectory is periodic, $\mathbf{q}_d(T) = \mathbf{q}_d(0)$, (repetitive control) or that the robot is reinitialized to lie on the desired trajectory at the beginning of each trail (learning control). Hereafter, we use the term learning control to mean either repetitive or learning control.

8.11.1 Pure P-Type Learning Control

Let $\boldsymbol{\tau}_k$ be the input torque during the k -th cycle, which produces an output $\mathbf{q}_k(t)$, $0 \leq t \leq T_{\text{bnd}}$. Now, let us consider the following set of assumptions:

- Assumption 1: Every trial ends at a fixed time of duration $T_{\text{bnd}} > 0$.
- Assumption 2: Repetition of the initial setting is satisfied.
- Assumption 3: Invariance of the system dynamics is ensured throughout these repeated trails.
- Assumption 4: Every output \mathbf{q}_k can be measured and thus the error signal $\Delta \mathbf{q}_k = \mathbf{q}_k - \mathbf{q}_d$ can be utilized in the construction of the next input $\boldsymbol{\tau}_{k+1}$.
- Assumption 5: The dynamics of the robot manipulators is invertible.

The learning control problem is to determine a recursive learning law \mathbf{L}

$$\boldsymbol{\tau}_{k+1} = \mathbf{L}[\boldsymbol{\tau}_k(t), \Delta \mathbf{q}_k(t)], \quad 0 \leq t \leq T_{\text{bnd}}, \quad (8.134)$$

where $\Delta \mathbf{q}_k(t) = \mathbf{q}_k(t) - \mathbf{q}_d(t)$, such that $\|\Delta \mathbf{q}_k\| \rightarrow 0$ as $k \rightarrow \infty$ in some suitably defined function norm, $\|\cdot\|$. The initial control input can be any control input that produces a stable output, such as PD control. Such learning control schemes are attractive because accurate models of the dynamics need not be known a priori.

Several approaches have been used to generate a suitable learning law \mathbf{L} and to prove convergence of

the output error. A pure P-type learning law is one of the form

$$\boldsymbol{\tau}_{k+1}(t) = \boldsymbol{\tau}_k(t) - \Phi \Delta \mathbf{q}_k(t), \quad (8.135)$$

and is given this name because the correction term for the input torque at each iteration is proportional to the error $\Delta \mathbf{q}_k$. Now let $\boldsymbol{\tau}_d$ be defined by the computed-torque control, i. e.,

$$\begin{aligned} \boldsymbol{\tau}_d(t) = & \mathbf{H}[\mathbf{q}_d(t)]\ddot{\mathbf{q}}_d(t) + \mathbf{C}[\mathbf{q}_d(t), \dot{\mathbf{q}}_d(t)]\dot{\mathbf{q}}_d(t) \\ & + \boldsymbol{\tau}_g[\mathbf{q}_d(t)]. \end{aligned} \quad (8.136)$$

One should recall that the function $\boldsymbol{\tau}_k$ actually does not need to be computed; it is sufficient to know that it exists. Considering the P-type learning control law, we have

$$\Delta \boldsymbol{\tau}_{k+1}(t) = \Delta \boldsymbol{\tau}_k(t) - \Phi \Delta \mathbf{q}_k(t), \quad (8.137)$$

where $\Delta \boldsymbol{\tau}_k(t) = \boldsymbol{\tau}_k(t) - \boldsymbol{\tau}_d(t)$, so that

$$\|\Delta \boldsymbol{\tau}_{k+1}(t)\|^2 \leq \|\Delta \boldsymbol{\tau}_k(t)\|^2 - \beta \|\Phi \Delta \mathbf{q}_k(t)\|^2 \quad (8.138)$$

provided there exist positive constant λ and β such that

$$\int_0^{T_{\text{bnd}}} e^{-\lambda t} \Delta \mathbf{q}_k^T \Delta \boldsymbol{\tau}_k(t) dt \geq \frac{1+\beta}{2} \|\Phi \Delta \mathbf{q}_k(t)\|^2 \quad (8.139)$$

for all k . It then follows from the inequality above that $\Delta \mathbf{q}_k \rightarrow 0$ in the norm sense as $k \rightarrow \infty$. Detailed stability analysis of this control scheme is given in [8.102, 103].

8.11.2 P-Type Learning Control with a Forgetting Factor

Although pure P-type learning control achieves the desired goal, several strict assumptions may be not valid in actual implementations, for example, there may be an initial setting error. Furthermore, there may be small but

nonrepeatable fluctuations of dynamics. Finally, there may exist a (bounded) measurement noise ξ_k such that

$$\Delta q_k(t) + \xi_k(t) = [q_k(t) + \xi_k(t)] - q_d(t). \quad (8.140)$$

Thus the learning control scheme may fail. In order to enhance the robustness of P-type learning control, a forgetting factor is introduced in the form

$$\begin{aligned} \tau_{k+1}(t) = & (1 - \alpha)\tau_k(t) + \alpha\tau_0(t) \\ & - \Phi[\Delta q_k(t) + \xi_k(t)]. \end{aligned} \quad (8.141)$$

The original idea of using a forgetting factor in learning control originated with [8.104].

It has been rigorously proven that P-type learning control with a forgetting factor guarantees convergence to a neighborhood of the desired one of size $O(\alpha)$. Moreover, if the content of a long-term memory is refreshed after every k trials, where k is of $O(1/\alpha)$, then the trajectories converge to an ε -neighborhood of the

desired control goal. The size of ε is dependent on the magnitude of the initial setting error, the nonrepeatable fluctuations of the dynamics, and the measurement noise. For a detailed stability investigation, please refer to [8.105, 106].







8.11.3 Summary

By applying learning control, the performance of repetitive tasks (such as painting or pick-and-place operation) is improved by utilizing data gathered in the previous cycles. In this section, two learning control schemes were introduced. First, pure P-type learning control and its robustness problem were described. Then P-type learning control with a forgetting factor was presented, enhancing the robustness of learning control.

Further Reading

Rigorous and refined exploration of learning control is first discussed independently in [8.2, 12].

Video-References

-  **VIDEO 25** Gain change of the PID controller
available from <http://handbookofrobotics.org/view-chapter/08/videtails/25>
-  **VIDEO 757** Safe human-robot cooperation
available from <http://handbookofrobotics.org/view-chapter/08/videtails/757>
-  **VIDEO 758** Virtual whiskers – Highly responsive robot collision avoidance
available from <http://handbookofrobotics.org/view-chapter/08/videtails/758>
-  **VIDEO 759** JediBot – Experiments in human-robot sword-fighting
available from <http://handbookofrobotics.org/view-chapter/08/videtails/759>
-  **VIDEO 760** Different jerk limits of robot arm trajectories
available from <http://handbookofrobotics.org/view-chapter/08/videtails/760>
-  **VIDEO 761** Sensor-based online trajectory generation
available from <http://handbookofrobotics.org/view-chapter/08/videtails/761>

References

- | | |
|--|---|
| <p>8.1 C. Canudas de Wit, B. Siciliano, G. Bastin: <i>Theory of Robot Control</i> (Springer, London 1996)</p> <p>8.2 J.J. Craig: <i>Adaptive Control of Mechanical Manipulators</i>, Ph.D. Thesis (UMI Dissertation Information Service, Ann Arbor 1986)</p> <p>8.3 R.J. Schilling: <i>Fundamentals of Robotics: Analysis and Control</i> (Prentice Hall, Englewood Cliffs 1989)</p> <p>8.4 L. Sciavicco, B. Siciliano: <i>Modeling and Control of Robot Manipulator</i> (McGraw-Hill, New York 1996)</p> <p>8.5 M.W. Spong, M. Vidyasagar: <i>Robot Dynamics and Control</i> (Wiley, New York 1989)</p> <p>8.6 M.W. Spong, F.L. Lewis, C.T. Abdallah (Eds.): <i>Robot Control</i> (IEEE, New York 1989)</p> <p>8.7 C.H. An, C.G. Atkeson, J.M. Hollerbach: <i>Model-Based Control of a Robot Manipulator</i> (MIT Press, Cambridge, 1988)</p> | <p>8.8 R.M. Murray, Z. Xi, S.S. Sastry: <i>A Mathematical Introduction to Robotic Manipulation</i> (CRC, Boca Raton 1994)</p> <p>8.9 T. Yoshikawa: <i>Foundations of Robotics</i> (MIT Press, Cambridge 1990)</p> <p>8.10 O. Khatib: A unified approach for motion and force control of robot manipulators: The operational space formulation, <i>IEEE J. Robotics Autom.</i> 3(1), 43–53 (1987)</p> <p>8.11 J.Y.S. Luh, M.W. Walker, R.P.C. Paul: Resolved-acceleration control of mechanical manipulator, <i>IEEE Trans. Autom. Control</i> 25(3), 468–474 (1980)</p> <p>8.12 S. Arimoto, F. Miyazaki: Stability and robustness of PID feedback control for robot manipulators of sensory capability. In: <i>Robotics Research</i>, ed.</p> |
|--|---|

- by M. Brady, R. Paul (MIT Press, Cambridge 1984) pp. 783–799
- 8.13 L.C. Fu: Robust adaptive decentralized control of robot manipulators, *IEEE Trans. Autom. Control* **37**(1), 106–110 (1992)
- 8.14 H. Seraji: Decentralized adaptive control of manipulators: Theory, simulation, and experimentation, *IEEE Trans. Robotics Autom.* **5**(2), 183–201 (1989)
- 8.15 J.G. Ziegler, N.B. Nichols: Optimum settings for automatic controllers, *Trans. ASME* **64**, 759–768 (1942)
- 8.16 Y. Choi, W.K. Chung: *PID Trajectory Tracking Control for Mechanical Systems*, Lecture Notes in Control and Information Sciences, Vol. 289 (Springer, New York 2004)
- 8.17 R. Kelly: PD control with desired gravity compensation of robot manipulators: A review, *Int. J. Robotics Res.* **16**(5), 660–672 (1997)
- 8.18 M. Takegaki, S. Arimoto: A new feedback method for dynamic control of manipulators, *Trans. ASME J. Dyn. Syst. Meas. Control* **103**, 119–125 (1981)
- 8.19 P. Tomei: Adaptive PD controller for robot manipulators, *IEEE Trans. Robotics Autom.* **7**(4), 565–570 (1991)
- 8.20 R. Ortega, A. Loria, R. Kelly: A semi-globally stable output feedback PI^2D regulator for robot manipulators, *IEEE Trans. Autom. Control* **40**(8), 1432–1436 (1995)
- 8.21 D. Angeli: Input-to-State stability of PD-controlled robotic systems, *Automatica* **35**, 1285–1290 (1999)
- 8.22 J.A. Ramirez, I. Cervantes, R. Kelly: PID regulation of robot manipulators: Stability and performance, *Syst. Control Lett.* **41**, 73–83 (2000)
- 8.23 R. Kelly: Global positioning of robot manipulators via PD control plus a class of nonlinear integral actions, *IEEE Trans. Autom. Control* **43**(7), 934–937 (1998)
- 8.24 Z. Qu, J. Dorsey: Robust tracking control of robots by a linear feedback law, *IEEE Trans. Autom. Control* **36**(9), 1081–1084 (1991)
- 8.25 H. Berghuis, H. Nijmeijer: Robust control of robots via linear estimated state feedback, *IEEE Trans. Autom. Control* **39**(10), 2159–2162 (1994)
- 8.26 Y. Choi, W.K. Chung, I.H. Suh: Performance and \mathcal{H}_∞ optimality of PID trajectory tracking controller for Lagrangian systems, *IEEE Trans. Robotics Autom.* **17**(6), 857–869 (2001)
- 8.27 K. Aström, T. Haggglund: *PID Controllers: Theory, Design, and Tuning* (Instrument Society of America, Research Triangle Park 1995)
- 8.28 C.C. Yu: *Autotuning of PID Controllers: Relay Feedback Approach* (Springer, London 1999)
- 8.29 F.L. Lewis, C.T. Abdallah, D.M. Dawson: *Control of Robot Manipulators* (Macmillan, New York 1993)
- 8.30 A. Isidori: *Nonlinear Control Systems: An Introduction*, Lecture Notes in Control and Information Sciences, Vol. 72 (Springer, New York 1985)
- 8.31 H. Berghuis, H. Nijmeijer: A passivity approach to controller–observer design for robots, *IEEE Trans. Robotics Autom.* **9**, 740–754 (1993)
- 8.32 J.J. Slotine, W. Li: On the adaptive control of robot manipulators, *Int. J. Robotics Res.* **6**(3), 49–59 (1987)
- 8.33 G. Liu, A.A. Goldenberg: Comparative study of robust saturation–based control of robot manipulators: analysis and experiments, *Int. J. Robotics Res.* **15**(5), 473–491 (1996)
- 8.34 D.M. Dawson, M. Grabbe, F.L. Lewis: Optimal control of a modified computed–torque controller for a robot manipulator, *Int. J. Robotics Autom.* **6**(3), 161–165 (1991)
- 8.35 D.M. Dawson, Z. Qu, J. Duffie: Robust tracking control for robot manipulators: Theory, simulation and implementation, *Robotica* **11**, 201–208 (1993)
- 8.36 A. Jaritz, M.W. Spong: An experimental comparison of robust control algorithms on a direct drive manipulator, *IEEE Trans. Control Syst. Technol.* **4**(6), 627–640 (1996)
- 8.37 A. Isidori: *Nonlinear Control Systems*, 3rd edn. (Springer, New York 1995)
- 8.38 J.J. Slotine, W. Li: *Applied Nonlinear Control* (Prentice Hall, Englewood Cliffs 1991)
- 8.39 W.J. Rugh: *Linear System Theory*, 2nd edn. (Prentice Hall, Upper Saddle River 1996)
- 8.40 M.W. Spong, M. Vidyasagar: Robust microprocessor control of robot manipulators, *Automatica* **23**(3), 373–379 (1987)
- 8.41 H.K. Khalil: *Nonlinear Systems*, 3rd edn. (Prentice Hall, Upper Saddle River 2002)
- 8.42 M. Vidyasagar: *Nonlinear Systems Analysis*, 2nd edn. (Prentice Hall, Englewood Cliffs 1993)
- 8.43 J.T. Wen: A unified perspective on robot control: The energy Lyapunov function approach, *Int. J. Adapt. Control Signal Process.* **4**, 487–500 (1990)
- 8.44 R. Ortega, M.W. Spong: Adaptive motion control of rigid robots: A tutorial, *Automatica* **25**(6), 877–888 (1989)
- 8.45 N. Sadegh, R. Horowitz: Stability and robustness analysis of a class of adaptive controllers for robotic manipulators, *Int. J. Robotics Res.* **9**(3), 74–92 (1990)
- 8.46 S. Dubowsky, D.T. DesForges: The application of model–reference adaptive control to robotic manipulators, *ASME J. Dyn. Syst. Meas. Control* **37**(1), 106–110 (1992)
- 8.47 S.H. Hsu, L.C. Fu: A fully adaptive decentralized control of robot manipulators, *Automatica* **42**, 1761–1767 (2008)
- 8.48 A. Balestrino, G. de Maria, L. Sciavicco: An adaptive model following control for robotic manipulators, *ASME J. Dyn. Syst. Meas. Control* **105**, 143–151 (1983)
- 8.49 S. Nicosia, P. Tomei: Model reference adaptive control algorithms for industrial robots, *Automatica* **20**, 635–644 (1984)
- 8.50 R. Horowitz, M. Tomizuka: An adaptive control scheme for mechanical manipulators–Compensation of nonlinearity and decoupling control, *ASME J. Dyn. Syst. Meas. Control* **108**, 127–135 (1986)
- 8.51 I.D. Laudau: *Adaptive Control: The Model Reference Approach* (Dekker, New York 1979)
- 8.52 R. Lozano, C. Canudas de Wit: Passivity based adaptive control for mechanical manipulators using LS type estimation, *IEEE Trans. Autom. Control* **35**(12), 1363–1365 (1990)

- 8.53 B. Brogliato, I.D. Laudau, R. Lozano: Passive least squares type estimation algorithm for direct adaptive control, *Int. J. Adapt. Control Signal Process.* **6**, 35–44 (1992)
- 8.54 R. Johansson: Adaptive control of robot manipulator motion, *IEEE Trans. Robotics Autom.* **6**(4), 483–490 (1990)
- 8.55 M.W. Walker: Adaptive control of manipulators containing closed kinematic loops, *IEEE Trans. Robotics Autom.* **6**(1), 10–19 (1990)
- 8.56 J.S. Reed, P.A. Ioannou: Instability analysis and robust adaptive control of robotic manipulators, *IEEE Trans. Autom. Control* **5**(3), 74–92 (1989)
- 8.57 G. Tao: On robust adaptive control of robot manipulators, *Automatica* **28**(4), 803–807 (1992)
- 8.58 H. Berghuis, R. Ogata, H. Nijmeijer: A robust adaptive controller for robot manipulators, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (1992) pp. 1876–1881
- 8.59 R. Johansson: Quadratic optimization of motion coordination and control, *IEEE Trans. Autom. Control* **35**(11), 1197–1208 (1990)
- 8.60 Z. Qu, D.M. Dawson: *Robust Tracking Control of Robot Manipulators* (IEEE, Piscataway 1996)
- 8.61 P. Dorato, C. Abdallah, V. Cerone: *Linear-Quadratic Control* (Prentice Hall, Upper Saddle River 1995)
- 8.62 A. Locatelli: *Optimal Control: An Introduction* (Birkhäuser, Basel 2001)
- 8.63 A. Isidori: Feedback control of nonlinear systems, *Int. J. Robust Nonlin. Control* **2**, 291–311 (1992)
- 8.64 A.J. der van Schaft: Nonlinear state space \mathcal{H}_∞ control theory. In: *Essays on Control: Perspective in Theory and its Applications*, ed. by H.L. Trentelman, J.C. Willems (Birkhäuser, Basel 1993) pp. 153–190
- 8.65 A.J. der van Schaft: L_2 -gain analysis of nonlinear systems and nonlinear state feedback \mathcal{H}_∞ control, *IEEE Trans. Autom. Control* **37**(6), 770–784 (1992)
- 8.66 J. Park, W.K. Chung, Y. Youm: Analytic nonlinear \mathcal{H}_∞ inverse-optimal control for Euler–Lagrange system, *IEEE Trans. Robotics Autom.* **16**(6), 847–854 (2000)
- 8.67 B.S. Chen, T.S. Lee, J.H. Feng: A nonlinear \mathcal{H}_∞ control design in robotics systems under parametric perturbation and external disturbance, *Int. J. Control* **59**(12), 439–461 (1994)
- 8.68 J. Park, W.K. Chung: Design of a robust \mathcal{H}_∞ PID control for industrial manipulators, *ASME J. Dyn. Syst. Meas. Control* **122**(4), 803–812 (2000)
- 8.69 R.H. Castain, R.P. Paul: An on-line dynamic trajectory generator, *Int. J. Robotics Res.* **3**(1), 68–72 (1984)
- 8.70 T. Kröger: *On-Line Trajectory Generation in Robotic Systems*, Springer Tracts in Advanced Robotics, Vol. 58 (Springer, Berlin, Heidelberg 2010)
- 8.71 D. Simon, C. Isik: A trigonometric trajectory generator for robotic arms, *Int. J. Control* **57**(3), 505–517 (1993)
- 8.72 L. Biagiotti, C. Melchiorri: *Trajectory Planning for Automatic Machines and Robots* (Springer, Berlin, Heidelberg 2008)
- 8.73 J.E. Bobrow: Optimal robot path planning using the minimum-time criterion, *IEEE J. Robotics Autom.* **4**(4), 443–450 (1988)
- 8.74 K.G. Shin, N.D. McKay: Minimum-time control of robotic manipulators with geometric path constraints, *IEEE Trans. Autom. Control* **30**(5), 531–541 (1985)
- 8.75 F. Pfeiffer, R. Johanni: A concept for manipulator trajectory planning, *Proc. Int. IEEE Conf. Robotics Autom. (ICRA)* (1986) pp. 1399–1405
- 8.76 W. Khalil, E. Dombre: Trajectory generation. In: *Modeling, Identification and Control of Robots*, ed. by W. Khalil, E. Dombre (Butterworth-Heinemann, Oxford 2004)
- 8.77 A.I. Kostrikin, Y.I. Manin: *Linear Algebra and Geometry* (Gordon and Breach Sci. Publ., Amsterdam 1997)
- 8.78 M.E. Kahn, B. Roth: The near-minimum-time control of open-loop articulated kinematic chains, *ASME J. Dyn. Syst. Meas. Control* **93**, 164–172 (1971)
- 8.79 M. Brady: Trajectory planning. In: *Robot Motion: Planning and Control*, ed. by M. Brady, J.M. Hollerbach, T.L. Johnson, T. Lozano-Pérez, M.T. Mason (MIT Press, Cambridge 1982)
- 8.80 R.P.C. Paul: Manipulator cartesian path control. In: *Robot Motion: Planning and Control*, ed. by M. Brady, J.M. Hollerbach, T.L. Johnson, T. Lozano-Pérez, M.T. Mason (MIT Press, Cambridge 1982)
- 8.81 R.H. Taylor: Planning and execution of straight-line manipulator trajectories. In: *Robot Motion: Planning and Control*, ed. by M. Brady, J.M. Hollerbach, T.L. Johnson, T. Lozano-Pérez, M.T. Mason (MIT Press, Cambridge 1982)
- 8.82 C.-S. Lin, P.-R. Chang, J.Y.S. Luh: Formulation and optimization of cubic polynomial joint trajectories for industrial robots, *IEEE Trans. Autom. Control* **28**(12), 1066–1074 (1983)
- 8.83 J.M. Hollerbach: Dynamic scaling of manipulator trajectories, *ASME J. Dyn. Syst. Meas. Control* **106**(1), 102–106 (1984)
- 8.84 K.J. Kyriakopoulos, G.N. Sridis: Minimum jerk path generation, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (1988) pp. 364–369
- 8.85 J.-J.E. Slotine, H.S. Yang: Improving the efficiency of time-optimal path-following algorithms, *IEEE Trans. Robotics Autom.* **5**(1), 118–124 (1989)
- 8.86 Z. Shiller, H.-H. Lu: Computation of path constrained time optimal motions with dynamic singularities, *ASME J. Dyn. Syst. Meas. Control* **114**(1), 34–40 (1992)
- 8.87 P. Fiorini, Z. Shiller: Time optimal trajectory planning in dynamic environments, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (1996) pp. 1553–1558
- 8.88 O. Dahl, L. Nielsen: Torque limited path following by on-line trajectory time scaling, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (1989) pp. 1122–1128
- 8.89 B. Cao, G.I. Dodds, G.W. Irwin: Time-optimal and smooth constrained path planning for robot manipulators, *Proc. IEEE Int. Conf. Robotics Autom. (ICRA)* (1994) pp. 1853–1858
- 8.90 B. Cao, G.I. Dodds, G.W. Irwin: A practical approach to near time-optimal inspection-task-sequence

- planning for two cooperative industrial robot arms, *Int. J. Robotics Res.* **17**(8), 858–867 (1998)
- 8.91 D. Constantinescu, E.A. Croft: Smooth and time-optimal trajectory planning for industrial manipulators along specified paths, *J. Robotics Syst.* **17**(5), 233–249 (2000)
- 8.92 S. Macfarlane, E.A. Croft: Jerk-bounded manipulator trajectory planning: Design for real-time applications, *IEEE Trans. Robotics Autom.* **19**(1), 42–52 (2003)
- 8.93 R.L. Andersson: *A Robot Ping-Pong Player: Experiment in Real-Time Intelligent Control* (MIT Press, Cambridge 1988)
- 8.94 R.L. Andersson: Aggressive trajectory generator for a robot ping-pong player, *IEEE Control Syst. Mag.* **9**(2), 15–21 (1989)
- 8.95 J. Lloyd, V. Hayward: Trajectory generation for sensor-driven and time-varying tasks, *Int. J. Robotics Res.* **12**(4), 380–393 (1993)
- 8.96 K. Ahn, W.K. Chung, Y. Youn: Arbitrary states polynomial-like trajectory (ASPOT) generation, *Proc. IEEE 30th Annu. Conf. Ind. Electron. Soc.* (2004) pp. 123–128
- 8.97 X. Broquère, D. Sidobre, I. Herrera-Aguilar: Soft motion trajectory planner for service manipulator robot, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2008) pp. 2808–2813
- 8.98 R. Haschke, E. Weitnauer, H. Ritter: On-line planning of time-optimal, jerk-limited trajectories, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2008) pp. 3248–3253
- 8.99 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Prentice Hall, Upper Saddle River 2003)
- 8.100 K.S. Fu, R.C. Gonzalez, C.S.G. Lee: *Robotics: Control, Sensing, Vision and Intelligence* (McGraw-Hill, New York 1988)
- 8.101 M.W. Spong, S.A. Hutchinson, M. Vidyasagar: *Robot Modeling and Control* (Wiley, New York 2006)
- 8.102 S. Arimoto: Mathematical theory of learning with application to robot control. In: *Adaptive and Learning Control*, ed. by K.S. Narendra (Plenum, New York 1986) pp. 379–388
- 8.103 S. Kawamura, F. Miyazaki, S. Arimoto: Realization of robot motion based on a learning method, *IEEE Trans. Syst. Man. Cybern.* **18**(1), 126–134 (1988)
- 8.104 G. Heinzinger, D. Frewick, B. Paden, F. Miyazaki: Robust learning control, *Proc. IEEE Int. Conf. Decis. Control* (1989)
- 8.105 S. Arimoto: Robustness of learning control for robot manipulators, *Proc. IEEE Int. Conf. Decis. Control* (1990) pp. 1523–1528
- 8.106 S. Arimoto, T. Naiwa, H. Suzuki: Selective learning with a forgetting factor for robotic motion control, *Proc. IEEE Int. Conf. Decis. Control* (1991) pp. 728–733