

10. Performance Evaluation and Design Criteria

Jorge Angeles, Frank C. Park

This chapter is devoted to the design of robots, with a focus on serial architectures. In this regard, we start by proposing a stepwise design procedure; then, we recall the main issues in robot design. These issues pertain to workspace geometry, the kinetostatic, the dynamic, the elastostatic, and elastodynamic performance. In doing this, the mathematics behind the concepts addressed is briefly outlined to make the chapter self-contained.

We survey some of the tools and criteria used in the mechanical design and performance evaluation of robots. Our focus is limited to robots that are (a) primarily intended for manipulation tasks and (b) supplied with serial kinematic chains. The kinematics of parallel robots is addressed in detail in Chap. 12. Wheeled robots, walking robots, multifingered hands, and other similar specialized structures are studied in their own chapters.

10.1	The Robot Design Process	229
10.2	Workspace Criteria	231
10.2.1	Reaching a Set of Goal Frames	233
10.2.2	Workspace Volume and Topology ...	233
10.3	Dexterity Indices	235
10.3.1	Local Dexterity for Open Chains	235
10.3.2	Dynamics-Based Local Performance Evaluation	237
10.3.3	Global Dexterity Measures	237
10.3.4	Closed-Chain Dexterity Indices.....	238
10.3.5	Alternative Dexterity-Like Measures	238
10.4	Other Performance Indices	238
10.4.1	Acceleration Radius	238
10.4.2	Elastostatic Performance	239
10.4.3	Elastodynamic Performance.....	241
	References	242

The most obvious application of the criteria and tools described in this chapter is in the mechanical design of a robot. Robot design differs from the design of single-degree-of-freedom machinery in that the latter is intended for one specific task, e.g., picking up a workpiece from a belt conveyor and placing it on a magazine. Moreover, the conveyor is synchronized with the manipulating machine and the magazine is stationary, with well-defined locations where each workpiece is to be placed. Manipulation robots, in contrast, are not in-

tended for one specific task, but rather for a *family of tasks* falling within one class of workpiece motions, e.g., planar, spherical, translational, or motions produced by systems of the Selective Compliance Assembly Robot Arm (SCARA) type, also known as *Schönflies displacements* [10.1]. The challenge that robot designers face is therefore one of *uncertainty* in the specific task that the robot will be required to execute. Design criteria have been devised to help the designer cope with uncertainty, as discussed herein.

10.1 The Robot Design Process

Given a family of tasks that constitute the *functional requirements* in the design process, besides more-detailed *design specifications*, the role of the designer consists in

producing a robot that will meet all the requirements and specifications. The various stages in the robot design job at hand are intended to:

1. determine the *topology* of the kinematic chain underlying the mechanical structure. Under this item we consider first the robot type: serial, parallel or hybrid. Then, a decision is to be made on the layout of the various *subchains* in terms of the type of joints, most commonly, revolute and prismatic. Recently, one additional type has been recognized to be equally useful, the Π -joint, coupling two links under relative translation by means of two other links undergoing identical angular displacements, although about different parallel axes. The four links form a parallelogram four-bar linkage [10.2];
2. determine the *geometric dimensions* of the various links defining the *robotic architecture*, as required to fill a table of *Denavit–Hartenberg parameters* [10.3] so as to satisfy workspace requirements. Although these parameters are usually understood to include the joint variables, these variables do not affect the *robot architecture*; they determine instead the *robot posture*;
3. determine the structural dimensioning of the various links and joints, as needed to meet static load requirements, where load includes both forces and moments – wrenches – under either the most demanding or the most likely operation conditions, depending on the design philosophy adopted at the outset;
4. determine the structural dimensioning of the various links and joints, as needed to meet dynamic load requirements, where loads are inertia effects of links and manipulated object;
5. determine the elastodynamic dimensioning of the overall mechanical structure, including the actuator dynamics, to avoid a specific spectrum of excitation frequencies under either the most demanding or the most likely operation conditions;
6. select the actuators and their mechanical transmissions for the operation conditions adopted at the outset to cope with task uncertainty.

The above stages can be performed sequentially, in the order given above: (i) first, the topology is determined based on the family of tasks specified at the outset and the *shape* of the workspace, as discussed in Sect. 10.2.2; (ii) the link geometry is defined based on the workspace requirements, which include the *maximum reach*, and the topology defined in stage 1; (iii) with the link geometry thus defined, the structural dimensioning of links and joints (unless the robot under design is parallel, which does not fall within the scope of this chapter, all joints are actuated) is undertaken, so as to support the static loads assumed at the outset; (iv)

with the links and joints dimensioned for static-load conditions, the link centers of mass and link inertia matrices are determined for a preliminary evaluation of the motor torque requirements (this evaluation is preliminary in that it does not consider the dynamic load brought about by the actuators; this load can be significant, even in the case of parallel robots, which can have all their motors fixed to the robot base); (v) with the links assumed rigid, joint stiffness is assumed, based on experience or using data from a similar robot, which then leads to an elastodynamic model whose natural modes and frequencies can be determined at a selected set of robot postures (dynamic behavior of the structure is dependent on robot posture) by means of scientific code such as Matlab or computer-aided engineering (CAE) code such as Pro/Engineer or ANSYS; and (vi) if the frequency spectrum of the robot structure is acceptable, the designer can continue to motor selection; otherwise, a redimensioning is required, which means returning to stage 3.

Even though a design cycle can be completed as outlined above, the designer must now incorporate into the elastodynamic model the structural and inertial data provided by the motor manufacturer. This requires a return to stage 5 and a new elastodynamic analysis. It is thus apparent that the robot design process has one element in common with engineering design in general: both are iterative and open-ended [10.4]. Remarkably, however, the various items driving each design stage are, to a large extent, independent of each other, e.g., topology and geometry can be determined independently from motor selection. Obviously, all issues interact in the overall design process, but, within certain design specifications, the various items do not contradict each other, as to warrant a multiobjective design approach. That is, the optimum design of serial robots can be accomplished fairly well by means of a sequence of single-objective optimization jobs. Again, the results of the last stage, motor selection, must be integrated into an overall mathematical model to test the overall performance. One reference addressing practical optimization issues in the conceptual design of industrial robots is [10.5].

Only when the physical limits of components have been exhausted may a radical redesign requiring a return to stage 1 be warranted. This is the case with SCARA systems. Current industrial topologies of these robots are usually of the serial type, with some exceptions, like the *Konig* and *Hartman* RP-AH series robots with parallel architecture [10.6], which feature two serial SCARA systems sharing one common end-effector. The quest for shorter cycle times, as for an industry test cycle (see Sect. 10.2.1), has prompted the industry to look for

alternatives to serial architectures. This is how ABB Robotics is currently marketing a parallel robot, the *FlexPicker*, built upon *Clavel's* Delta robot [10.7], to which a fourth axis has been added in series with the first three. The latter are laid out in a symmetric, parallel architecture that enables Delta to produce pure translations of its moving platform. The shortest cycle time reported by Adept Technology is 420 ms for a payload of 2 kg (with the Adept Cobra s600, a serial robot) but other manufacturers claim even shorter times.

This chapter is organized according to the various stages of the robot design process outlined earlier. Noting that topology selection and geometric dimensioning are tightly coupled in the kinematic design process, we

first begin with an examination of workspace criteria: we review methods for determining the topology of the kinematic chain, followed by the geometric dimensions so as to satisfy workspace requirements. We then review in detail the various criteria developed for characterizing a robot's manipulating capability, focusing on quantitative notions of dexterity based on both kinematic and dynamic models. We then examine methods for structural dimensioning of the links and joints so as to meet both static and dynamic load requirements. Finally, we discuss elastodynamic dimensioning, and actuator and gear sizing, taking into account properties such as the natural frequency of the robot, and force and acceleration capability requirements.

10.2 Workspace Criteria

The most obvious consideration in designing a robot is that its workspace has a set of *required characteristics*. This is a fundamental problem in classical mechanism design, and raises the obvious question of how a user can specify those characteristics.

Issues to consider here pertain, mostly, to what Vijaykumar et al. [10.8] termed the *regional structure* of a manipulator. This applies to manipulators with a *decoupled architecture*, whose last three revolute have concurrent axes, thereby forming a *spherical wrist*, the point of concurrency being the *wrist center*. The manipulation task of architectures of this kind thus allows for a *decoupling* of the positioning and the orientation subtasks: the regional structure, consisting of the first three joints, is first postured so as to locate the center of its wrist at a specified point $C(x, y, z)$; then, the *local structure*, i.e., the wrist, is postured so as to make the end-effector (EE) attain a specified orientation with respect to a frame fixed to the base, given by a rotation matrix.

Most algorithms reported in the literature to determine the workspace of a given robot refer to the workspace of the regional structure. Here, we should distinguish between the workspace of the kinematic chain, regardless of the physical implementation of the chain, and that of the physical robot. In the former, all revolute joints are capable of unlimited rotations about their axes; in the latter, joint limits are needed, for example, to avoid wire entanglement. In the early stages of robot design, joint limits need not be considered, the workspace thus exhibiting symmetries that are proper of the type of joints of the regional structure. If the first joint is a revo-

lute, the workspace has an axis of symmetry, namely, the axis of this revolute joint; if the first joint is prismatic, the workspace has an extrusion symmetry, with the direction of extrusion given by the direction of motion of this joint. As prismatic joints are infinitely extensive, so is the kinematic workspace of a robot with a prismatic joint. The kinematic workspaces of robots with prismatic joints are usually displayed for a finite portion of this workspace.

In the case of parallel robots, to be studied in full detail in Chap. 14, the regional structure is elusive, in general. The usual practice when displaying the workspace for these robots is to assume a constant orientation of the moving plate, the counterpart of the EE of serial robots [10.9]. A common architecture of parallel robots, which arises quite naturally in the design process, entails identical legs symmetrically placed both on the base platform and on the moving platform. Each leg is, in turn, a serial kinematic chain with one or two active joints, all others being passive. The workspace of this kind of robots also exhibits certain symmetries, but no axial symmetry. The symmetries are dictated by the number of legs and the types of actuated joints.

Coming back to serial robots, the workspace can be defined by an envelope that is essentially of one of two types, either a manifold or a surface that is smooth *almost everywhere*, i.e., smooth everywhere except for a set of points of *measure zero* in the Lebesgue sense [10.10]. Broadly speaking, a set of measure zero on a surface is a curve, e.g., a meridian on a sphere, or a set of isolated points on a line, e.g., the set of rational numbers on the real line. A paradigm for this second kind of workspace

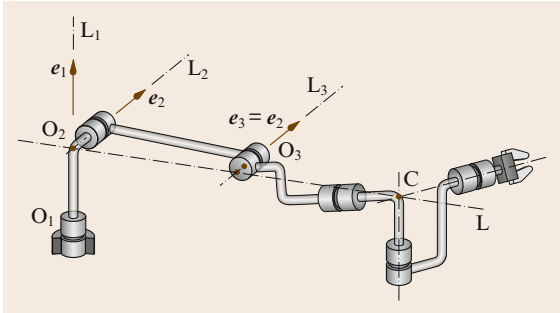


Fig. 10.1 A Puma robot in a fully stretched posture (after [10.11])

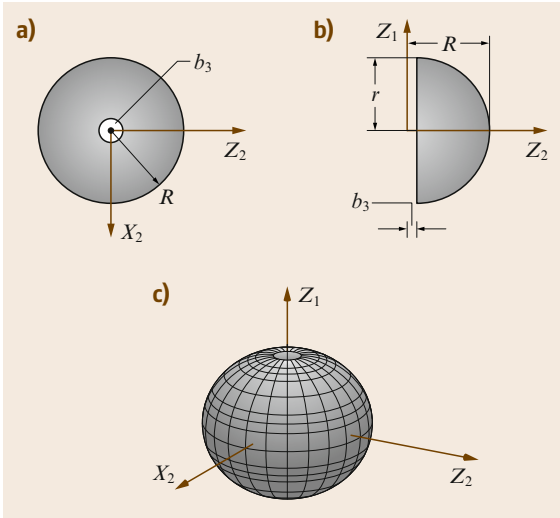


Fig. 10.2a-c The workspace of a Puma robot (after [10.11])

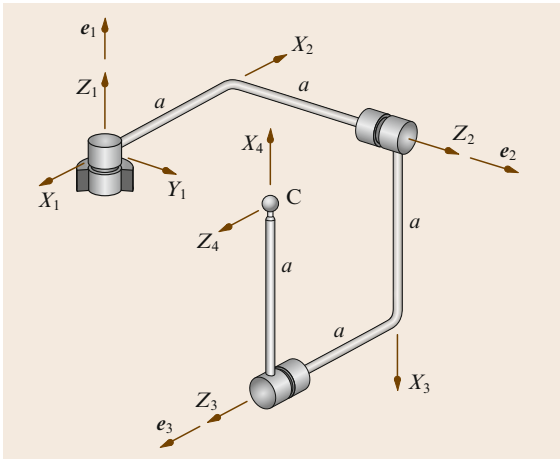


Fig. 10.3 An orthogonal three-revolute robot (after [10.11])

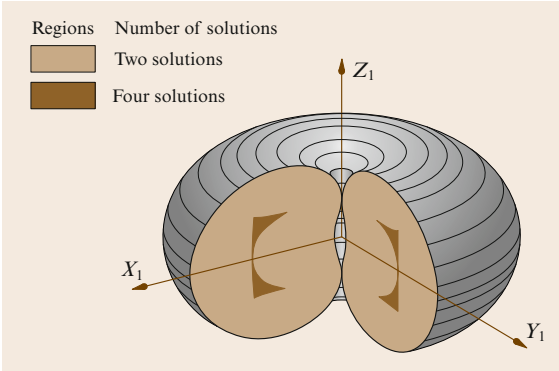


Fig. 10.4 The workspace of the orthogonal robot of Fig. 10.3 (after [10.11])

is that of the Puma robot, whose kinematic chain is displayed in Fig. 10.1. In this figure, the regional and the local structures are clearly distinguished, the former being *fully extended*. The workspace of this robot is obtained upon locking all joints but the second, when the robot is in the posture shown in Fig. 10.1. Then, the second joint is fully rotated about its axis, the center C of the wrist then describing a circle of radius R equal to the distance of C from the line L_2 , the plane of the circle being normal to this line and lying a distance b_3 from the axis L_1 of the first joint. This distance is known as the *shoulder offset*. Now, with all joints locked again, but this time with the first joint unlocked, the robot is turned as a rigid body about L_1 . The result is the toroid of Fig. 10.2. Notice that the solid enclosed by this surface is the result of the Boolean operation $\mathcal{S} - \mathcal{C}$, where \mathcal{S} is the sphere of radius R centered at point O_2 of Fig. 10.1, while \mathcal{C} is the infinite cylinder of radius b_3 and axis L_1 , which appears as Z_1 in Fig. 10.2. It is noteworthy that, although this workspace can be readily generated by a simple Boolean operation, it cannot possibly be generated by an implicit function of the form $f(x, y, z) = 0$ because the surface is not a manifold.

Robots with manifold workspaces are not common in industry. We display in Fig. 10.3 an architecture for the regional structure of a six-axis decoupled robot, with its neighboring axes mutually orthogonal and at the same distance a from each other. The common normals to the two pairs of axes, X_2 and X_3 , also lie at the same distance a , as do X_4 from X_3 and C from Z_3 . Point C is the center of the spherical wrist, the latter not being included in the figure. The workspace of this robot admits a representation of the form $f(x, y, z) = 0$ [10.11], which produces the manifold workspace of Fig. 10.2. The shaded internal region

of the workspace includes all points admitting four real inverse-kinematics solutions, all other points admitting only two.

Given that any point of the workspace boundary represents a positioning singularity – different from an orientation singularity – manipulators with workspace boundaries that are not manifolds exhibit double singularities at the edges of their workspace boundary, which means that at edge points the rank of the robot Jacobian becomes deficient by two. At any other point of the workspace boundary the rank deficiency is by one.

Design rules based on the *shape* of the workspace can now be drawn.

1. If the workspace required is axially symmetric and finite, use a serial robot with a regional structure composed of revolute joints only.
2. If the workspace required is prismatic and *infinite*, use a serial robot with regional structure having one first joint of the prismatic type. Here, *infinite* is used in a restricted sense, meaning much larger in one direction than the others. Moreover,
 - if one direction is required to be much larger than the others, then practical implementations of prismatic joints are available in the form of rails either overhead, thereby giving rise to *gantry* robots, or on the floor.
 - if two directions are required to be much larger than the other, then use a wheeled mobile robot carrying a manipulator on top. A famous realization of this concept is the National Aeronautical and Space Agency's (NASA) *Sojourner* used in the *Pathfinder* mission to Mars in 1997.
3. If axial symmetry is not required, but rather a workspace with various coplanar axes of symmetry, similar to those of regular polygons, use a parallel robot.

10.2.1 Reaching a Set of Goal Frames

Closely related to the problem of workspace specification is that of task specification. In mechanism design it is customary to specify a set of coordinate frames in space, and to design a mechanism with an a priori specified topology that can visit these frames. An order in which the frames must be reached may be given. In the event that not all of the frames are reachable, then one may seek a mechanism that comes *closest*, in some suitable sense, to the specified frames. The lit-

erature on this classical mechanism design problem is vast – see, e.g., [10.1, 12, 13] and the references cited therein. Some further remarks in connection with this goal-frame approach to robot dimensioning are noteworthy.

1. Reaching *exactly* the desired frames may not always be desired or possible: in some cases it is better to use an optimization approach that allows for solutions that will visit the desired poses within a minimum error (provided that an error norm can be suitably engineered, of course).
2. It has been claimed [10.9] that interval analysis allows not only a discrete set of desired poses but also a full six-dimensional (6-D) workspace to be met while taking into account manufacturing errors.
3. The branching problem occurring in single-degree-of-freedom mechanisms may also occur in robot design: a design solution based on via points may indeed visit the prescribed poses, but not all of these may be reachable within the same assembly mode. This problem is exacerbated in the design of serial robots, as a six-degree-of-freedom, revolute-coupled robot may admit up to 16 distinct postures – branches – for one given EE pose [10.14, 15].
4. While a robot designed to visit a set of prescribed poses via its end-effector will be able to visit that set, we should not forget that the purpose of using robots is first and foremost to be able to perform not one single task, but rather a family of tasks. In this light, the set of poses for which a robot is designed might as well be a task that is *representative* of that family.

In connection with remark 4 above, we can cite the design or evaluation of **SCARA** systems. A **SCARA** system is a four-degree-of-freedom serial robot capable of tasks that lie within the *Schönflies* subgroup of the group of rigid-body displacements [10.16, 17], namely the set of three-dimensional displacements augmented with a rotation about an axis of fixed direction. In these systems, the task at hand is given by two vertical segments joined by one horizontal segment. Moreover, the length of the vertical segments is 25.0 mm, that of the horizontal segment being 300.0 mm. While the end-effector is traversing the horizontal segment, moreover, it should rotate about a vertical axis through an angle of 180°. This task specification, which has been adopted by **SCARA** manufacturers, does not indicate how to negotiate the corners, which is left to the imagination of the robotics engineer.

10.2.2 Workspace Volume and Topology

Reachable and Dexterous Workspace

Beginning with the early work of Roth [10.18], there have been many studies on the relationship between manipulator kinematic geometry and its workspace. Most studies have focused on a classification of the workspace into two components, the *reachable* and the *dexterous* workspace [10.19]. Given a reference point P attached to a manipulator end-effector, such as the center of the spherical wrist, or a point on the EE, the reachable workspace is defined to be the set of points in physical space that can be reached by P . The dexterous workspace, on the other hand, is the set of points that can be reached by P with arbitrary EE orientations.

The early literature on workspace focuses on numerical and algebraic methods to characterize these workspaces. Reachable and dexterous workspaces have been analyzed using numerical techniques by Kumar and Waldron [10.19], Yang and Lee [10.20], and Tsai and Soni [10.21], among others. The advantage of these schemes over algebraic approaches is that kinematic constraints can be readily included. More-general design principles or insights, however, are more difficult to come by using these techniques. Among the algebraic approaches to workspace characterization, a topological analysis of robot workspace is given by Gupta and Roth [10.22] and Gupta [10.23]; here the concept of workspace holes and voids is defined, and conditions for their existence are identified. The shape of the reachable and dexterous workspaces is also analyzed as a function of P .

Further studies of workspace analysis were reported by Freudenstein and Primrose [10.24] and by Lin [10.25], where precise relationships between kinematic and workspace parameters are developed, and a class of three-joint manipulators is optimized for workspace volume. A more general analysis of workspace optimization is given in Vijaykumar et al. [10.8]. Performance criteria for manipulators are defined here in terms of the dexterous workspace; given that a manipulator satisfies certain constraints on its Denavit–Hartenberg parameters, it is shown that the optimal 6R design is the elbow manipulator.

A typical design of robot regional architecture is of the *orthogonal type*, consisting of one revolute of the vertical axis and two revolute of the horizontal axes, one of which intersects the vertical axis. Moreover, the most common architecture includes intermediate and distal links of identical lengths. The workspace

of this architecture is thus a sphere of radius equal to twice that common link length. The volume of this workspace is thus determined by the length in question. As shown by Yoshikawa [10.26], the workspace of the two-link planar manipulator defined by the last two links of the foregoing regional architecture is of maximum area for a prescribed reach and equal link lengths. As a result, the volume of the same regional architecture is similarly of maximum volume for a prescribed reach.

Differential-Geometric Workspace Characterization

Workspace can also be approached from a differential-geometric perspective, by regarding the configuration space of a robotic end-effector frame as a subset of the special Euclidean group $SE(3)$. An important physical consideration in defining the workspace volume of spatial mechanisms is that it should not depend on the choice of fixed reference frame. Less obvious but just as important is the requirement that the volume should not depend on which point of the last link the end-effector frame is fixed to. This last condition has the following physical significance: if the EE were enlarged or shrunk, then the robot would have the same workspace volume. The workspace volume of a robot therefore depends only on the joint axes.

The workspace volume of a robot is defined by regarding $SE(3)$ as a Riemannian manifold, so that the workspace volume is simply the volume of the image of the forward kinematic map f with respect to the volume form on $SE(3)$. It is known that $SE(3)$ has a bi-invariant volume form, that is, the notion of volume is invariant with respect to the choice of both the fixed (base) and moving (end-effector) frames – see, e.g., Loncaric [10.27]. Paden and Sastry [10.28] provide the following visualization for this volume form: Suppose an airplane is restricted to move within a cube of airspace of length 1 km on a side. At each point within this cube the airplane can point itself anywhere in a 4π solid angle and roll 2π about the direction it is pointing. The orientation volume at such a point is $4\pi \times 2\pi = 8\pi^2 \text{ rad}^3$. Multiplying by the positional volume one obtains $8\pi^2 \text{ rad}^3 \text{ km}^3$ for the volume of the free configuration space of the aircraft.

This depiction is the notion of workspace volume used for robots; it has the advantage of being able to trade off orientation freedom for positional freedom smoothly, unlike the popular notion of dexterous workspace. Note that the actual numerical value one obtains will depend on the choice of length scale for physical space; this

in itself does not pose a serious problem for workspace volumes, as long as the same length scale is maintained when comparing different workspaces.

In [10.28] *Paden* and *Sastry* show that the optimal 6R manipulator that maximizes the workspace volume subject to a kinematic length constraint is the

elbow manipulator. This result is consistent with the earlier finding of *Vijaykumar* et al. [10.8], but the authors employ the geometric framework outlined above. Moreover, the results are obtained without some of the a priori assumptions on the kinematic structure made by *Vijaykumar*.

10.3 Dexterity Indices

10.3.1 Local Dexterity for Open Chains

Dexterity can be defined as the ability to move and apply forces and torques in arbitrary directions with equal ease, the concept thus belonging to the realm of *kinetostatics*, which is the study of the interplay between *feasible twists* and *constraint wrenches* in multibody mechanical systems under *static conservative conditions*. Here, twist is the six-dimensional array of velocity variables of a rigid body, involving three components of a landmark-point velocity and three of angular velocity; wrench, in turn, is the six-dimensional array of static variables acting on a rigid body, three accounting for the resultant force applied at the same landmark point and three for the concomitant moment acting on the same body.

Salisbury and *Craig* [10.29] introduced the concept of dexterity when working on the design of articulated hands. At issue is the way in which input joint velocity errors propagate to the output velocities of each fingertip. To illustrate this concept, let $\mathbf{J}(\boldsymbol{\theta})$ denote the Jacobian of the forward kinematic map, i. e.,

$$\mathbf{t} = \mathbf{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \quad (10.1)$$

in which $\boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$ denote the vectors of joint variables and joint rates, respectively, while \mathbf{t} is the EE *twist*, defined, in turn, as,

$$\mathbf{t} = \begin{pmatrix} \boldsymbol{\omega} \\ \dot{\mathbf{p}} \end{pmatrix} \quad (10.2)$$

with $\boldsymbol{\omega}$ denoting the angular velocity of the EE and $\dot{\mathbf{p}}$ the velocity of the *operation point* P of the EE, at which the task is specified.

Let n and m be the dimensions of the joint space \mathcal{J} and of the end-effector configuration space \mathcal{G} , respectively, where $n \geq m$. Now, the image under $\mathbf{J}(\boldsymbol{\theta})$ of the unit hypersphere $\{\boldsymbol{\theta} \mid \|\dot{\boldsymbol{\theta}}\| = 1\}$ is an ellipsoid in the space of twists \mathbf{t} . Indeed, if we assume for concreteness that $n = m$ – the more general case $n \neq m$ can also be accommodated, but we refrain from including it here in

order to streamline the discussion that follows – then the *polar decomposition* [10.30] of \mathbf{J} takes the form

$$\mathbf{J} = \mathbf{R}\mathbf{U} = \mathbf{V}\mathbf{R}, \quad (10.3)$$

where \mathbf{R} is an orthogonal matrix, whether proper or improper – if proper, then \mathbf{R} represents a rotation; if improper, then a reflection – while \mathbf{U} and \mathbf{V} are symmetric and at least positive-semidefinite matrices. Moreover, if \mathbf{J} is nonsingular, then \mathbf{U} and \mathbf{V} are both positive-definite and the decomposition is *unique*. In any event, these two matrices are related by a similarity transformation, namely,

$$\mathbf{V} = \mathbf{R}\mathbf{U}\mathbf{R}^\top \quad (10.4)$$

which means that the two matrices have identical real nonnegative eigenvalues. These eigenvalues are nothing but the *singular values* of \mathbf{J} . Indeed, if $\boldsymbol{\Sigma}$ denotes the diagonal representation of \mathbf{U} , with its i -th diagonal entry σ_i denoting the i -th eigenvalue of \mathbf{U} , or of \mathbf{V} for that matter, then the *eigenvalue decomposition* of \mathbf{U} is

$$\mathbf{U} = \mathbf{E}\boldsymbol{\Sigma}\mathbf{E}^\top \quad (10.5)$$

in which \mathbf{E} is the orthogonal matrix whose i -th column \mathbf{e}_i represents the i -th eigenvector of \mathbf{U} , while $\mathbf{R}\mathbf{e}_i$ denotes the i -th eigenvector of \mathbf{V} . Now, if we substitute the above decomposition of \mathbf{U} into the polar decomposition of \mathbf{J} , (10.3), we obtain

$$\mathbf{J} = \mathbf{R}\mathbf{E}\boldsymbol{\Sigma}\mathbf{E}^\top \quad (10.6)$$

which is nothing but the *singular-value decomposition* of \mathbf{J} , the diagonal entries of $\boldsymbol{\Sigma}$ being the *singular values* of \mathbf{J} .

Now we can provide a geometric interpretation of the forward kinematic mapping of (10.1), for we can rewrite this mapping in the form

$$\mathbf{t} = \mathbf{R}\mathbf{U}\dot{\boldsymbol{\theta}}. \quad (10.7)$$

At nonsingular postures, the Jacobian is invertible, and hence so is \mathbf{U} , whence the foregoing expression leads to

$$\dot{\boldsymbol{\theta}} = \mathbf{U}^{-1}\mathbf{R}^\top\mathbf{t}. \quad (10.8)$$

Furthermore, if we assume that all the components of both the twist array \mathbf{t} and the joint-rate array $\dot{\boldsymbol{\theta}}$ have the same physical units, which is the case for purely positioning or purely orienting manipulators with only revolute joints, then we can take the Euclidean norm of both sides of (10.8), thereby obtaining

$$\|\dot{\boldsymbol{\theta}}\|^2 = \mathbf{t}^\top \mathbf{R} \mathbf{U}^{-2} \mathbf{R}^\top \mathbf{t}. \quad (10.9)$$

Moreover, if we substitute \mathbf{U} by its eigenvalue decomposition, (10.5), in the above equation, we obtain

$$\|\dot{\boldsymbol{\theta}}\|^2 = \mathbf{t}^\top \mathbf{R} \mathbf{E} \boldsymbol{\Sigma}^{-2} \mathbf{E}^\top \mathbf{R}^\top \mathbf{t}$$

and, if we let

$$\mathbf{v} = \mathbf{E}^\top \mathbf{R}^\top \mathbf{t} \quad (10.10)$$

then the above expression for $\|\dot{\boldsymbol{\theta}}\|^2$ becomes

$$\mathbf{v}^\top \boldsymbol{\Sigma}^{-2} \mathbf{v} = \|\dot{\boldsymbol{\theta}}\|^2. \quad (10.11)$$

Now, if the i -th component of \mathbf{v} is denoted by v_i , for $i = 1, \dots, n$, and we look at the mapping of the unit ball in \mathcal{J} , $\|\dot{\boldsymbol{\theta}}\|^2 = 1$, (10.11) leads to

$$\frac{v_1^2}{\sigma_1^2} + \frac{v_2^2}{\sigma_2^2} + \dots + \frac{v_n^2}{\sigma_n^2} = 1 \quad (10.12)$$

which is the canonical equation of an ellipsoid of semi-axes $\{\sigma_i\}_1^n$ in the \mathcal{J} -space, i. e., the space of Cartesian velocities, or twists of the EE. Notice that the ellipsoid in question takes its canonical form when represented in a coordinate frame of axes oriented in the directions of the eigenvectors of \mathbf{U} . The equation of the ellipsoid along the original axes in \mathcal{J} -space is, of course,

$$\mathbf{t}^\top \mathbf{R} \mathbf{E} \boldsymbol{\Sigma}^{-2} \mathbf{E}^\top \mathbf{R}^\top \mathbf{t} = 1. \quad (10.13)$$

In summary the unit ball in joint space is mapped by the Jacobian-inverse \mathbf{J}^{-1} into an ellipsoid whose semi-axes are the singular values of \mathbf{J} . That is, \mathbf{J} *distorts* the unit ball in the joint-rate space into an ellipsoid in the EE-twist space. Hence, a measure of the quality of motion and force transmission of the robotic architecture from the joints to the EE is given by the above distortion; *the smaller the distortion, the higher the quality of the transmission*.

A measure of the Jacobian-induced distortion can thus be defined as the ratio of the largest σ_M to the smallest σ_m singular values of \mathbf{J} , which is nothing but the *condition number* κ_2 of \mathbf{J} based on the matrix 2-norm [10.31], i. e.,

$$\kappa_2 = \frac{\sigma_M}{\sigma_m}. \quad (10.14)$$

Actually, (10.14) is only one possibility of computing the condition number of \mathbf{J} , or of any $m \times n$ matrix for that matter, and certainly not the most economical. Notice that this definition requires knowledge of the singular values of the Jacobian. However, computing the singular values of a matrix is as computationally intensive as computing eigenvalues, with the added cost of the polar decomposition; the combined operation is slightly less expensive than the singular-value decomposition [10.32]. The most general definition of condition number, for $n \times n$ matrices, is [10.31]

$$\kappa(\mathbf{A}) = \|\mathbf{A}\| \|\mathbf{A}^{-1}\|. \quad (10.15)$$

As stated above, the expression (10.14) is obtained when the matrix 2-norm is adopted in (10.15). The matrix 2-norm is defined as

$$\|\mathbf{A}\|_2 \equiv \max_i \{ \sigma_i \}. \quad (10.16)$$

If, on the other hand, the *weighted matrix Frobenius norm* is adopted, which is defined as

$$\|\mathbf{A}\|_F \equiv \sqrt{\frac{1}{n} \text{tr}(\mathbf{A} \mathbf{A}^\top)} \equiv \sqrt{\frac{1}{n} \text{tr}(\mathbf{A}^\top \mathbf{A})} \quad (10.17)$$

then, apparently, the computation of the singular values can be obviated. When the weight $1/n$ is omitted in the above definition, the standard Frobenius norm is obtained. Notice, however, that the weighted Frobenius norm is more significant in engineering, for it does not depend on the number of rows and columns of the matrix at hand. The weighted Frobenius norm, in fact, yields the root-mean-square (rms) value of the set of singular values.

The *Frobenius condition number* κ_F of the Jacobian \mathbf{J} , based on the matrix Frobenius norm, is then

$$\begin{aligned} \kappa_F(\mathbf{J}) &= \frac{1}{n} \sqrt{\text{tr}(\mathbf{J} \mathbf{J}^\top)} \sqrt{\text{tr}[(\mathbf{J} \mathbf{J}^\top)^{-1}]} \\ &= \frac{1}{n} \sqrt{\text{tr}(\mathbf{J}^\top \mathbf{J})} \sqrt{\text{tr}[(\mathbf{J}^\top \mathbf{J})^{-1}]}. \end{aligned} \quad (10.18)$$

One more important difference between the two forms of computing the matrix condition number is worth pointing out: $\kappa_2(\cdot)$ is not an *analytic function* of its matrix argument, while $\kappa_F(\cdot)$ is. Hence, the condition number based on the Frobenius norm can be applied to great advantage in robot architecture design, as $\kappa_F(\cdot)$ is differentiable and lends itself to gradient-dependent optimization methods, which are much faster than direct methods based only on function evaluations. In robot control, which requires real-time computations, $\kappa_F(\cdot)$ also offers practical advantages, for its computation

obviates the knowledge of the singular values, only requiring a matrix inversion, which is a rather simple task.

The significance of the condition number in robot design and control can be best understood if we recall that the concept stems from numerical analysis in connection with the solution of the linear system of equations (10.1) for $\dot{\theta}$. Given that \mathbf{J} is a function of both the architecture parameters and the posture variables θ , \mathbf{J} is known only to within the error with which those quantities are known. Further, let the architecture parameters, namely, the constant values in the Denavit–Hartenberg parameter list, be stored in an array \mathbf{p} . Both \mathbf{p} and θ are known up to measurement errors $\delta\mathbf{p}$ and $\delta\theta$, respectively. Moreover, the twist \mathbf{t} is input into the control software of the robot with an unavoidable error $\delta\mathbf{t}$.

In solving (10.1) for $\dot{\theta}$ with floating-point arithmetic, the computed value is contaminated with a roundoff error $\delta\dot{\theta}$. The *relative error* in the computed $\dot{\theta}$ is related to the relative errors in the architecture parameters and posture variables by the relation [10.31]

$$\frac{\|\delta\dot{\theta}\|}{\|\dot{\theta}\|} \leq \kappa(\mathbf{J}) \left(\frac{\|\delta\mathbf{p}\|}{\|\mathbf{p}\|} + \frac{\|\delta\theta\|}{\|\theta\|} + \frac{\|\delta\mathbf{t}\|}{\|\mathbf{t}\|} \right), \quad (10.19)$$

where \mathbf{p} and θ represent the (unknown) actual values of these vectors and \mathbf{t} the nominal value of the twist.

Nevertheless, the foregoing discussion applies to tasks involving either positioning or orientation, but not both. Most frequently, robotic tasks involve both positioning and orientation, which leads to Jacobian matrices with entries bearing disparate physical units, the consequence being that the Jacobian singular values have disparate units as well. Indeed, singular values associated with positioning bear units of length, whereas those associated with orientation are dimensionless. As a consequence, it is impossible to either order all singular values from smallest to largest or to add them all.

To cope with the issue of disparate units in the Jacobian entries, and to allow for the computation of the Jacobian condition number, the concept of *characteristic length* has been proposed [10.11]. The characteristic length L has been defined as the length by which the entries of the Jacobian with units of length are to be divided to render the Jacobian condition number a minimum at an optimum posture. This definition, while sound, lacks an immediate geometric interpretation, which has made its adoption in the robotics community difficult. In order to provide for a geometric interpretation, the concept of *homogeneous space* was recently introduced [10.33]. Using this concept, the robot architecture is designed in a *dimensionless space*, with points whose coordinates are dimensionless real numbers. In this way, the

six Plücker coordinates [10.34] of lines are all dimensionless, and hence the columns of the robot Jacobian matrix, comprising the Plücker coordinates of the revolute axes, are dimensionless as well. As a consequence, the Jacobian singular values are all dimensionless, and the Jacobian condition number can be defined. Once the robotic architecture has been designed for minimum condition number, under geometric constraints on link-length ratios and angles between neighboring revolute axes, for example, the maximum reach of the robot can be calculated. The maximum reach r will thus be a dimensionless quantity. When this quantity is compared with the prescribed maximum reach R , with units of length, the characteristic length is computed as the ratio $L = R/r$.

10.3.2 Dynamics-Based Local Performance Evaluation

Since motions are caused by forces and torques acting on rigid bodies, it seems reasonable to formulate performance indices that take into account the inertial properties of the mechanism. Asada [10.35] defines the *generalized-inertia ellipsoid* (GIE) as the ellipsoid defined by the product $\mathbf{G} = \mathbf{J}^{-\top} \mathbf{M} \mathbf{J}^{-1}$, where \mathbf{M} denotes the inertia matrix of the manipulator. This ellipsoid is that with semiaxes given by the singular values of the foregoing product. Yoshikawa [10.36], in turn, defines a corresponding dynamic manipulability measure as $\det[\mathbf{J} \mathbf{M}^{-1} (\mathbf{J} \mathbf{M}^{-1})^{\top}]$. Physically these concepts represent two distinct phenomena. Suppose the robot is viewed as an input–output device which, given a joint torque, produces an acceleration at the end-effector. Yoshikawa’s index measures the uniformity of this torque–acceleration gain, whereas Asada’s GIE characterizes the inverse of this gain. If a human operator were holding the end-effector and attempting to move it about, the GIE would measure the resistance of the robot to this end-effector motion.

Other measures that attempt to capture robot performance as a function of the dynamics can be cited: Voglewede and Ebert–Uphoff [10.37] propose performance indices based on joint stiffness and link inertia, with the aim of establishing a distance to singularity for any robot posture, while Bowling and Khatib [10.38] propose a general framework for capturing the dynamic capability of a general robot manipulator that includes the velocity and acceleration characteristics of the end-effector, taking into account factors such as torque and the velocity limits of the actuators.

10.3.3 Global Dexterity Measures

The above measures are local in the sense that they characterize the dexterity of a robot at a given posture. Local measures are useful for applications ranging from redundancy resolution to workpiece positioning, but for design applications a global measure may be more desirable. One straightforward way of extending local measures to global ones is to integrate them over the allowable joint space. In [10.39] *Gosselin and Angeles* integrate the Jacobian condition number over the workspace to define a global measure, thereby producing a *global conditioning index*. For the simpler cases of planar positioning and spherical manipulators, the global conditioning index was found to coincide with its local counterpart.

10.3.4 Closed-Chain Dexterity Indices

The formulation of dexterity for closed chains presents a number of subtleties. The first obvious difference is that the joint configuration space of a closed chain will no longer be a flat space; in the general case it will be a curved multidimensional surface embedded in a higher-dimensional (typically flat) space. Also, dual to the open chain case, the forward kinematics for closed chains is generally more difficult to solve than the inverse kinematics, with the possibility of multiple solutions. Another important difference is that only a subset of the joints may be actuated, and that the number of actuated joints may even exceed the mechanism's kinematic degrees of freedom.

Several coordinate-based formulations for closed-chain dexterity have been proposed for specific mechanisms [10.40] and for cooperating robot systems whose joints are all assumed to be actuated [10.41, 42], at least some of which have led to apparently paradoxical results [10.43, 44]. Because of the nonlinear characteristics unique to closed-chain mechanisms discussed above, particular care must be exercised in formulating coordinate-based dexterity measures.

Another recent line of work has attempted a coordinate-invariant differential-geometric formulation of

dexterity for closed chains. In this framework the joint and end-effector configuration spaces are regarded as Riemannian manifolds with an appropriate choice of Riemannian metric, with the choice of joint space metric reflecting the characteristics of the joint actuators. The previous notions of ellipsoid developed for serial chains can be extended to general closed chains, containing both passive and active joints, and possibly redundantly actuated [10.45, 46].

10.3.5 Alternative Dexterity-Like Measures

The various definitions of dexterity discussed above all refer to the same qualitative feature of the ability of a robot to move and apply forces in arbitrary directions. A different viewpoint is taken in the work of *Liégeois* [10.47] and *Klein and Huang* [10.48], where dexterity is quantified in terms of joint-range availability. The driving motivation here lies in that most robots have joint limits; therefore, one should minimize the possibility of a joint reaching a stop.

Hollerbach [10.49] takes an alternative approach to designing a redundant 7R manipulator, by considering: (a) elimination of internal singularities, (b) ability to avoid obstacles in the workspace; (c) the solvability of the kinematic equations, and (d) mechanical constructability. Based on these four criteria, he derived a particular 7R design with the morphology of the human arm, i. e., composed of two spherical joints defining the shoulder and the wrist plus an intermediate revolute playing the role of the elbow. Now, while a redundant architecture should remain fully capable of performing six-DOF tasks upon locking one joint, the architecture reported in this reference may end up by losing this capability if the elbow joint is locked.

From a control perspective, *Spong* [10.50] shows that, if the inertia matrix of a manipulator has a vanishing Riemannian curvature, there exists a set of coordinates in which the equations of motion assume a particularly simple form. The curvature of the inertia matrix also reflects the sensitivity of the dynamics to certain robot parameters. Minimizing the curvature, therefore, is another possible criterion for robot design.

10.4 Other Performance Indices

10.4.1 Acceleration Radius

Another measure that attempts to capture the dynamic capabilities of a robotic manipulator is the *acceler-*

ation radius. Initially proposed by *Graettinger and Krogh* in [10.51], the acceleration radius measures the minimum acceleration capability of the end-effector, in arbitrary directions, for the given torque bounds on the

actuators. Specifically, given the dynamics equations for a serial chain in the form

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} \quad (10.20)$$

in which M is the robot *mass matrix* – also known as the *inertia matrix* – in *joint space*, and $C(\theta, \dot{\theta})$ is the matrix mapping the joint-rate vector to the vector of Coriolis and centrifugal forces in the same space. Moreover, the actuators are assumed to have joint torque limits of the form

$$\tau_{\min} \leq \tau \leq \tau_{\max}, \quad (10.21)$$

where the lower and upper limits $\tau_{\min}, \tau_{\max} \in \mathbb{R}^n$ are constant or functions of the manipulator posture θ . The end-effector *twist rate*, denoted by \dot{i} , can be expressed as

$$\dot{i} = J(\theta)\ddot{\theta} + \dot{J}(\theta, \dot{\theta})\dot{\theta}, \quad (10.22)$$

where $J(\theta, \dot{\theta})$ is the Jacobian time-derivative. The Jacobian J was introduced in (10.1).

Under the assumption that $J(\theta)$ is nonsingular, one can write

$$\ddot{\theta} = J(\theta)^{-1}\dot{i} - \dot{J}(\theta)^{-1}J(\theta, \dot{\theta}). \quad (10.23)$$

Substituting the above expression into the dynamic equations (10.20) leads to

$$\tau(\theta, \dot{\theta}, \ddot{\theta}) = M'(\theta)\dot{i} + C'(\theta, \dot{\theta}), \quad (10.24)$$

where

$$M'(\theta) = M(\theta)J(\theta)^{-1} \\ C'(\theta, \dot{\theta}) = [C(\theta, \dot{\theta}) - M(\theta)J(\theta)^{-1}\dot{J}(\theta, \dot{\theta})]J^{-1}(\theta).$$

For a given state $(\theta, \dot{\theta})$, the linear torque bounds (10.21) now define a polytope in the twist-rate space. *Graettinger* and *Krogh* [10.51] define the acceleration radius to be the largest sphere centered at the origin that is contained in this polytope; the radius reflects the minimal guaranteed EE acceleration in arbitrary directions. This concept is applied to measure the EE acceleration capability of a manipulator, as well as to determine the actuator sizes for achieving a desired acceleration radius. *Bowling* and *Khatib* [10.38] generalize this concept to capture both force and acceleration capabilities of the end-effector, with a view to quantifying the worst-case dynamic performance capability of a manipulator.

10.4.2 Elastostatic Performance

Elastostatic performance refers to the robotic system's response to the applied load – force and moment, i. e.,

wrench – under static equilibrium. This response may be measured in terms of the *stiffness* of the manipulator, which determines the translation and angular deflections when the EE is subjected to an applied wrench.

Robot deflections have two sources: link and joint deflection. In the presence of long links, as in the space robot *Canadarm2*, link compliance is a major source of deflection. However, in the majority of today's serial robots, the main source of deflection occurs at the joints.

In this chapter, we assume that the manipulator links are rigid, and model the joints as linearly elastic torsional springs. The more complex problem of link flexibility is studied in detail in Chap. 13. That is, for the elastostatic model we base our analysis on the assumption that, under a positioning task, the joints are locked at a certain posture θ_0 , while the EE is subject to a perturbation wrench Δw that is balanced by an elastic joint torque $\Delta \tau$. Under these conditions, $\Delta \theta$ and $\Delta \tau$ obey the well-known linear relation

$$K \Delta \theta = \Delta \tau \quad (10.25)$$

in which K is the *stiffness matrix* in joint space at the given posture. Moreover, the matrix K is diagonal, with its entries equal to the torsional stiffnesses of the corresponding joints, K is therefore *posture independent*, i. e., constant throughout the whole robot workspace. Moreover, since all joints exhibit a finite, nonzero stiffness, K is invertible, its inverse C being termed the *compliance matrix*. We can thus express the inverse relation of (10.25) as

$$\Delta \theta = C \Delta \tau. \quad (10.26)$$

It should be apparent that $\Delta \theta$ and $\Delta \tau$ have an *incremental* nature, for both are measured from the equilibrium posture, at which $\Delta \tau$ and $\Delta \theta$ vanish.

On the issue of stiffness matrix, *Griffis* and *Duffy* [10.52] proposed a mapping from an incremental rigid-body displacement Δx into an incremental wrench Δw that turned out to be nonsymmetric. The concept behind this mapping was formalized by *Howard* et al. [10.53] by means of Lie algebra. However, in the foregoing papers, Δx and Δw turn out to be incompatible in the sense that their reciprocal product does not yield incremental work – the point at which Δx is defined is distinct from that at which Δw is applied. For this reason, the array representation of the same mapping cannot be, properly speaking, termed a stiffness matrix.

For a constant magnitude of $\Delta \tau$, the deflection attains its maximum value in the direction of the eigenvector associated with the maximum eigenvalue of C

or, equivalently, with the minimum eigenvalue of \mathbf{K} , denoted by κ_{\min} . In terms of elastostatic performance, we aim to (a) make the maximum deflection a minimum, i. e., we want to maximize κ_{\min} , and (b) make the magnitude of the deflection $\|\Delta\theta\|$ as insensitive as possible to changes in the direction of the applied load $\Delta\tau$. This can be done by rendering κ_{\min} as close as possible to κ_{\max} . The first aim is associated with the stiffness constants, i. e., the higher the constants the lower the deflections. The later, however, is associated with the concept of isotropy, the ideal case being when all the eigenvalues of \mathbf{K} are identical, which means that \mathbf{K} is isotropic. Due to the *pyramidal effect* of serial robots, in which downstream motors carry their upstream counterparts, the joint stiffness is bound to be largest for the proximal joints. Hence an isotropic stiffness matrix is impossible for serial robots.

Notice that (10.25) and (10.26) may also be formulated in the task space, i. e.,

$$\mathbf{K}_C \Delta \mathbf{x} = \Delta \mathbf{w}, \quad (10.27)$$

where $\Delta \mathbf{x} \equiv \mathbf{t} \Delta t$, with Δt denoting a *small* time interval producing a correspondingly *small* change $\Delta \mathbf{x}$ in the pose of the EE. That is,

$$\Delta \mathbf{x} = \mathbf{J} \dot{\theta} \Delta t = \mathbf{J} \Delta \theta \quad (10.28)$$

which is a linear transformation of the joint-vector increment into the pose-increment vector. We show next that *the stiffness matrix is not frame invariant*. This means that, under the linear transformation from joint to Cartesian coordinates, *the stiffness matrix does not obey a similarity transformation*. For quick reference, we recall below the definition of similarity transformation: if $\mathbf{y} = \mathbf{L}\mathbf{x}$ is a linear transformation of \mathbb{R}^n into itself, and we introduce a change of vector basis, $\mathbf{x}' = \mathbf{A}\mathbf{x}$, $\mathbf{y}' = \mathbf{A}\mathbf{y}$, then \mathbf{L} changes to \mathbf{L}' , which is given by

$$\mathbf{L}' = \mathbf{A} \mathbf{L} \mathbf{A}^{-1}. \quad (10.29)$$

The above transformation of any vector of \mathbb{R}^n into another one of the same space, and of a matrix \mathbf{L} into \mathbf{L}' , as given by (10.29), is termed a *similarity transformation*. Notice that \mathbf{A} is guaranteed to be invertible, as it represents a change of coordinate frame.

Now, under the change of coordinates given by (10.28), (10.27) leads to

$$\mathbf{K}_C \mathbf{J} \Delta \theta = \mathbf{J}^{-\top} \Delta \tau, \quad (10.30)$$

where we have used the kinetostatic relation [10.11]

$$\mathbf{J}^\top \Delta \mathbf{w} = \Delta \tau.$$

The exponent $-\top$ denotes the transpose of the inverse or, equivalently, the inverse of the transpose. Upon multiplication of both sides of (10.30) by \mathbf{J}^\top from the left, we end up with

$$\mathbf{J}^\top \mathbf{K}_C \mathbf{J} \Delta \theta = \Delta \tau. \quad (10.31)$$

If we now compare (10.25) with (10.31), we can readily derive the relations between the stiffness matrix \mathbf{K} in joint space and the stiffness matrix \mathbf{K}_C in Cartesian space:

$$\mathbf{K} = \mathbf{J}^\top \mathbf{K}_C \mathbf{J} \quad \text{and} \quad \mathbf{K}_C = \mathbf{J}^{-\top} \mathbf{K} \mathbf{J}^{-1} \quad (10.32)$$

which apparently is not a similarity transformation between \mathbf{K} and \mathbf{K}_C . What this means is that the two matrices do not share the same set of eigenvalues and their eigenvectors are not related by the linear relation (10.28). As a matter of fact, if the robot is revolute-coupled, the entries of its stiffness matrix \mathbf{K} all have units of Nm, i. e., of *torsional stiffness*, while the entries of \mathbf{K}_C have disparate units. To show this, the Jacobian matrix, the inverse Jacobian and the two stiffness matrices are partitioned correspondingly into four 3×3 blocks, i. e., as

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_{11} & \mathbf{J}_{12} \\ \mathbf{J}_{21} & \mathbf{J}_{22} \end{pmatrix}, \quad \mathbf{J}^{-1} = \begin{pmatrix} \mathbf{J}'_{11} & \mathbf{J}'_{12} \\ \mathbf{J}'_{21} & \mathbf{J}'_{22} \end{pmatrix},$$

$$\mathbf{K} = \begin{pmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{12}^\top & \mathbf{K}_{22} \end{pmatrix}, \quad \mathbf{K}_C = \begin{pmatrix} \mathbf{K}_{C11} & \mathbf{K}_{C12} \\ \mathbf{K}_{C12}^\top & \mathbf{K}_{C22} \end{pmatrix}.$$

Furthermore, in light of the definition of the twist, (10.2), the two upper blocks of \mathbf{J} are dimensionless, while its two lower blocks have units of length [10.11]. As a consequence, the two left blocks of \mathbf{J}^{-1} are dimensionless, while its two right blocks have units of inverse length. Now, the blocks of \mathbf{K}_C are computed from the corresponding relation in (10.32), thereby obtaining

$$\begin{aligned} \mathbf{K}_{C11} &= \mathbf{J}_{11}^\top (\mathbf{K}_{11} \mathbf{J}'_{11} + \mathbf{K}_{12} \mathbf{J}'_{21}) \\ &\quad + \mathbf{J}_{21}^\top (\mathbf{K}_{12}^\top \mathbf{J}'_{11} + \mathbf{K}_{22} \mathbf{J}'_{21}) \\ \mathbf{K}_{C12} &= \mathbf{J}_{11}^\top (\mathbf{K}_{11} \mathbf{J}'_{12} + \mathbf{K}_{12} \mathbf{J}'_{22}) \\ &\quad + \mathbf{J}_{21}^\top (\mathbf{K}_{12}^\top \mathbf{J}'_{12} + \mathbf{K}_{22} \mathbf{J}'_{22}) \\ \mathbf{K}_{C21} &= \mathbf{K}_{C12}^\top \\ \mathbf{K}_{C22} &= \mathbf{J}_{12}^\top (\mathbf{K}_{11} \mathbf{J}'_{12} + \mathbf{K}_{12} \mathbf{J}'_{22}) \\ &\quad + \mathbf{J}_{22}^\top (\mathbf{K}_{12}^\top \mathbf{J}'_{12} + \mathbf{K}_{22} \mathbf{J}'_{22}). \end{aligned}$$

It is now apparent that: K_{C11} has entries with units of Nm, i. e., of torsional stiffness; K_{C12} and K_{C21} have entries with units of N; and K_{C22} has entries with units of N/m, i. e., of translational stiffness.

The outcome of the foregoing discussion is that a norm for K is possible, but not one for K_C , unless of course a characteristic length is introduced to render all the entries of K_C dimensionally homogeneous. A norm of a matrix is useful as it indicates *how large* the entries of the matrix are. We would like to characterize how stiff a robot is in both joint and Cartesian spaces. In joint space we could adopt any norm, but notice that the 2-norm, introduced in (10.3), would be misleading, as this norm would assign the value of the strongest joint to the overall robot stiffness. A more suitable norm would be the weighted Frobenius norm, introduced in (10.17), which would assign the rms value of the joint stiffnesses to the overall robot stiffness.

To design a robot optimally, we would therefore aim to maximize the Frobenius norm of its stiffness matrix in joint space, while observing constraints on the robot weight, as stiffer joints lead to heavier joints if the same material is used for all joints.

10.4.3 Elastodynamic Performance

For a general design problem, not only the kinetostatic and elastostatic performances, but also the elastodynamic performance have to be considered. In this regard, we introduce the assumptions of Sect. 10.4.2, with the added condition that inertia forces due to the link masses and moments of inertia are now taken into consideration.

The linearized model of a serial robot at the posture given by θ_0 , if we neglect damping, is

$$M\Delta\ddot{\theta} + K\Delta\theta = \Delta\tau \quad (10.33)$$

in which M is the $n \times n$ positive-definite mass matrix introduced in Sect. 10.4.1, while K was defined in Sect. 10.4.2 as the $n \times n$ positive-definite stiffness matrix in joint space. Both K and M were defined in joint-space coordinates, $\Delta\theta$, representing the vector of joint-variable elastic displacements, as in Sect. 10.4.2. These displacements are produced when, as the joints are all locked at a value θ_0 and thereby become ideal linear springs, the robot is subject to a perturbation $\Delta\tau$, to nonzero initial conditions, or to a combination of both.

Under *free vibration*, i. e., under a motion of the system (10.33) caused by nonzero initial conditions and a zero excitation $\Delta\tau$, the foregoing equation can be

solved for $\Delta\ddot{\theta}$:

$$\Delta\ddot{\theta} = -D\Delta\theta, \quad D \equiv M^{-1}K \quad (10.34)$$

in which the matrix D is known as the *dynamic matrix*. This matrix determines the behavior of the system under consideration, as its eigenvalues are the *natural frequencies* of the system and its eigenvectors the *modal vectors*. Let $\{\omega_i\}_1^n$ and $\{f_i\}_1^n$ denote the sets of eigenvalues and the corresponding eigenvectors of D , respectively. Under the initial conditions $[\Delta\theta(0), \Delta\dot{\theta}(0)]^\top$, in which $\Delta\theta(0)$ is proportional to the i -th eigenvector of D and $\Delta\dot{\theta}(0) = 0$, the ensuing motion is of the form $\Delta\theta(t) = \Delta\theta(0) \cos \omega_i t$ [10.54].

Furthermore, under the change of variables given by (10.28), the model (10.33) changes to

$$MJ^{-1}\Delta\ddot{x} + KJ^{-1}\Delta x = J^\top \Delta w.$$

If we now multiply both sides of the above equation by $J^{-\top}$, we obtain the elastodynamic model (10.33) in Cartesian coordinates, namely

$$J^{-\top}MJ^{-1}\Delta\ddot{x} + J^{-\top}KJ^{-1}\Delta x = \Delta w$$

in which the first matrix coefficient is the mass matrix M_C in Cartesian coordinates, and the second is identified as K_C , which was introduced in (10.32), i. e.,

$$M_C \equiv J^{-\top}MJ^{-1}. \quad (10.35)$$

The elastodynamic model in Cartesian coordinates thus takes the form

$$M_C\Delta\ddot{x} + K_C\Delta x = \Delta w. \quad (10.36)$$

Again, by virtue of the transformation (10.35), it is apparent that the mass matrix, like its stiffness counterpart, is not invariant under a change of coordinates. Moreover, in a revolute-coupled robot, all the entries of M have units of kg m^2 ; however, the entries of M_C have disparate units. An analysis similar to that conducted in Sect. 10.4.2 for the stiffness matrix in Cartesian space reveals that, if M_C is partitioned into four 3×3 blocks, then its upper-left block has units of moment of inertia, its lower-right block has units of mass, while its off-diagonal blocks have units of kg m .

Correspondingly, the dynamic matrix in Cartesian coordinates becomes

$$D_C = M_C^{-1}K_C. \quad (10.37)$$

It is now a simple matter to prove that *the dynamic matrix is invariant under a change of coordinates*. Indeed,

if we substitute transformations (10.32) and (10.35) into (10.37), we obtain

$$\mathbf{D}_C = \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^\top \mathbf{J}^{-\top} \mathbf{K}\mathbf{J}^{-1} = \mathbf{J}\mathbf{M}^{-1}\mathbf{K}\mathbf{J}^{-1},$$

in which the dynamic matrix \mathbf{D} in joint coordinates can be readily identified in the final expression, and hence,

$$\mathbf{D}_C = \mathbf{J}\mathbf{D}\mathbf{J}^{-1}, \quad (10.38)$$

which shows that \mathbf{D}_C is indeed a similarity transformation of \mathbf{D} . As a consequence, the dynamic matrix is indeed invariant under a change of coordinates, the two matrices thus sharing the same sets of eigenvalues, their eigenvectors being related by the same similarity transformation. That is, if the modal vectors in joint space – the eigenvectors of \mathbf{D} – are denoted by $\{\mathbf{f}_i\}_1^n$ and the modal vectors in Cartesian space are denoted by $\{\mathbf{g}_i\}_1^n$, then these two sets are related by

$$\mathbf{g}_i = \mathbf{J}\mathbf{f}_i, \quad i = 1, \dots, n. \quad (10.39)$$

Therefore, the natural frequencies of the elastodynamic model are the same, regardless of the space in which they are calculated, while their natural modes of vibration change under a similarity transformation.

Under an excitation of the form $\Delta\tau = \theta_0 \cos \omega t$ and zero initial conditions, the system is known to respond

with a harmonic motion of frequency ω and magnitude that depends on both ω and the system *frequency spectrum* $\{\omega_i\}_1^n$ [10.54]. When ω equals one of the natural frequencies of the system, the response magnitude grows unbounded, a phenomenon known as *resonance*. For this reason, when designing a robot, it is imperative that its frequency spectrum does not involve any of the expected operation frequencies, which can be achieved by designing the robot with stiffness and mass matrices that yield a frequency spectrum outside of the frequency range of the operation conditions.

This design task is not straightforward. Indeed, while the stiffness matrix in joint space is constant, the mass matrix is posture dependent, i.e., $\mathbf{M} = \mathbf{M}(\boldsymbol{\theta})$. Because of this feature, the elastodynamic design of a robot is bound to be iterative: the design can be conducted for a *straw-man task*, i.e., a typical task in the target applications, thus defining a set of postures that lead in turn to a set of mass-matrix values. Then, the frequency spectrum for all these postures can be designed to lie above the frequency range of the straw-man task. Since the robot will eventually be commanded to execute other tasks than the straw-man task, simulation of alternative tasks is required to ensure that the design is safe from a resonance viewpoint.

References

- 10.1 O. Bottema, B. Roth: *Theoretical Kinematics* (North-Holland, Amsterdam 1979), Also available by Dover Publishing, New York 1990
- 10.2 J. Angeles: The qualitative synthesis of parallel manipulators, *ASME J. Mech. Des.* **126**(4), 617–624 (2004)
- 10.3 J. Denavit, R.S. Hartenberg: A kinematic notation for lower-pair mechanisms based on matrices, *ASME J. Appl. Mech.* **77**, 215–221 (1955)
- 10.4 G. Pahl, W. Beitz: *Engineering Design. A Systematic Approach*, 3rd edn. (Springer, London 2007), Translated from the original Sixth Edition in German
- 10.5 M. Petterson, J. Andersson, P. Krus, X. Feng, D. Wappling: Industrial Robot Design Optimization in the Conceptual Design Phase, *Proc. Mechatron. Robot.*, Vol. 2, ed. by P. Drews (APS-European Centre for Mechatronics, Aachen 2004) pp.125–130
- 10.6 Koning & Hartman, Amsterdam, The Netherlands, http://www.konighartman.com/nl/producten/aandrijven_en_besturen/robots/ir_rp_ah/ (November 23, 2007)
- 10.7 R. Clavel: Device for the movement and positioning of an element in space, Patent 4976582 (1990)
- 10.8 R. Vijaykumar, K.J. Waldron, M.J. Tsai: Geometric optimization of serial chain manipulator structures for working volume and dexterity, *Int. J. Robot. Res.* **5**(2), 91–103 (1986)
- 10.9 J.P. Merlet: *Parallel Robots* (Springer, Dordrecht 2006)
- 10.10 K. Hoffman: *Analysis in Euclidean Space* (Prentice-Hall, Englewood Cliffs 1975)
- 10.11 J. Angeles: *Fundamentals of Robotic Mechanical Systems*, 3rd edn. (Springer, New York 2007)
- 10.12 L. Burmester: *Lehrbuch der Kinematik* (Arthur Felix, Leipzig 1886), in German
- 10.13 J.M. McCarthy: *Geometric Design of Linkages* (Springer, New York 2000)
- 10.14 H. Li: Ein Verfahren zur vollständigen Lösung der Rückwärtstransformation für Industrieroboter mit allgemeiner Geometrie. Ph.D. Thesis (Universität-Gesamthochschule Duisburg, Duisburg 1990)
- 10.15 M. Raghavan, B. Roth: Kinematic analysis of the 6R manipulator of general geometry, *Proc. 5th Int. Symp. Robot. Res.*, ed. by H. Miura, S. Arimoto (MIT Press, Cambridge 1990)
- 10.16 J. Angeles: The degree of freedom of parallel robots: a group-theoretic approach, *Proc. IEEE Int. Conf. Robot. Autom.* (Barcelona 2005) pp. 1017–1024
- 10.17 C.C. Lee, J.M. Hervé: Translational parallel manipulators with doubly planar limbs, *Mechanism Machine Theory* **41**, 433–455 (2006)

- 10.18 B. Roth: Performance evaluation of manipulators from a kinematic viewpoint, National Bureau of Standards – NBS SP **495**, 39–61 (1976)
- 10.19 A. Kumar, K.J. Waldron: The workspaces of a mechanical manipulator, ASME J. Mech. Des. **103**, 665–672 (1981)
- 10.20 D.C.H. Yang, T.W. Lee: On the workspace of mechanical manipulators, ASME J. Mech. Trans. Autom. Des. **105**, 62–69 (1983)
- 10.21 Y.C. Tsai, A.H. Soni: An algorithm for the workspace of a general n-R robot, ASME J. Mech. Trans. Autom. Des. **105**, 52–57 (1985)
- 10.22 K.C. Gupta, B. Roth: Design considerations for manipulator workspace, ASME J. Mech. Des. **104**, 704–711 (1982)
- 10.23 K.C. Gupta: On the nature of robot workspace, Int. J. Robot. Res. **5**(2), 112–121 (1986)
- 10.24 F. Freudenstein, E. Primrose: On the analysis and synthesis of the workspace of a three-link, turning-pair connected robot arm, ASME J. Mech. Trans. Autom. Des. **106**, 365–370 (1984)
- 10.25 C.C. Lin, F. Freudenstein: Optimization of the workspace of a three-link turning-pair connected robot arm, Int. J. Robot. Res. **5**(2), 91–103 (1986)
- 10.26 T. Yoshikawa: Manipulability of robotic mechanisms, Int. J. Robot. Res. **4**(2), 3–9 (1985)
- 10.27 J. Loncaric: Geometric Analysis of Compliant Mechanisms in Robotics. Ph.D. Thesis (Harvard University, Harvard 1985)
- 10.28 B. Paden, S. Sastry: Optimal kinematic design of 6R manipulators, Int. J. Robot. Res. **7**(2), 43–61 (1988)
- 10.29 J.K. Salisbury, J.J. Craig: Articulated hands: force control and kinematic issues, Int. J. Robot. Res. **1**(1), 4–17 (1982)
- 10.30 G. Strang: *Linear Algebra and Its Applications*, 3rd edn. (Harcourt Brace Jovanovich College Publishers, New York 1988)
- 10.31 G.H. Golub, C.F. Van Loan: *Matrix Computations* (The Johns Hopkins Univ. Press, Baltimore 1989)
- 10.32 A. Dubrulle: An optimum iteration for the matrix polar decomposition, Electron. Trans. Numer. Anal. **8**, 21–25 (1999)
- 10.33 W.A. Khan, J. Angeles: The Kinetostatic Optimization of Robotic Manipulators: The Inverse and the Direct Problems, ASME J. Mech. Des. **128**, 168–178 (2006)
- 10.34 H. Pottmann, J. Wallner: *Computational Line Geometry* (Springer, Berlin, Heidelberg, New York 2001)
- 10.35 H. Asada: A geometrical representation of manipulator dynamics and its application to arm design, Trans. ASME J. Dyn. Sys. Meas. Contr. **105**(3), 131–135 (1983)
- 10.36 T. Yoshikawa: Dynamic manipulability of robot manipulators, Proc. IEEE Int. Conf. Robot. Autom. (1985) pp. 1033–1038
- 10.37 P.A. Voglewede, I. Ebert-Uphoff: Measuring closeness to singularities for parallel manipulators, Proc. IEEE Int. Conf. Robot. Autom. (New Orleans 2004) pp. 4539–4544
- 10.38 A. Bowling, O. Khatib: The dynamic capability equations: a new tool for analyzing robotic manipulator performance, IEEE Trans. Robot. **21**(1), 115–123 (2005)
- 10.39 C.M. Gosselin, J. Angeles: A new performance index for the kinematic optimization of robotic manipulators, Proc. 20th ASME Mech. Conf. (Kissimmee 1988) pp. 441–447
- 10.40 C. Gosselin, J. Angeles: The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator, ASME J. Mech. Trans. Autom. Des. **110**, 35–41 (1988)
- 10.41 A. Bicchi, C. Melchiorri, D. Balluchi: On the mobility and manipulability of general multiple limb robots, IEEE Trans. Robot. Autom. **11**(2), 232–235 (1995)
- 10.42 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Global task space manipulability ellipsoids for multiple-arm systems, IEEE Trans. Robot. Autom. **7**, 678–685 (1991)
- 10.43 C. Melchiorri: Comments on Global task space manipulability ellipsoids for multiple-arm systems and further considerations, IEEE Trans. Robot. Autom. **9**, 232–235 (1993)
- 10.44 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Reply to comments on Global task space manipulability ellipsoids for multiple-arm systems' and further considerations, IEEE Trans. Robot. Autom. **9**, 235–236 (1993)
- 10.45 F.C. Park: Optimal robot design and differential geometry, ASME Special 50th Anniv. Design Issue **117**(B), 87–92 (1995)
- 10.46 F.C. Park, J. Kim: Manipulability of closed kinematic chains, ASME J. Mech. Des. **120**(4), 542–548 (1998)
- 10.47 A. Liégeois: Automatic supervisory control for the configuration and behavior of multibody mechanisms, IEEE Trans. Sys. Man. Cyber. **7**(12), 842–868 (1977)
- 10.48 C.A. Klein, C.H. Huang: Review of pseudo-inverse control for use with kinematically redundant manipulators, IEEE Trans. Sys. Man. Cyber. **13**(2), 245–250 (1983)
- 10.49 J.M. Hollerbach: Optimum kinematic design of a seven degree of freedom manipulator. In: *Robotics Research: The Second International Symposium*, ed. by H. Hanafusa, H. Inoue (MIT Press, Cambridge 1985)
- 10.50 M.W. Spong: Remarks on robot dynamics: canonical transformations and riemannian geometry, Proc. IEEE Int. Conf. Robot. Autom. (1992) pp. 454–472
- 10.51 T.J. Graettinger, B.H. Krogh: The acceleration radius: a global performance measure for robotic

manipulators, IEEE J. Robot. Autom. **4**(11), 60–69 (1988)

10.52 M. Griffis, J. Duffy: Global stiffness modeling of a class of simple compliant couplings, Mechanism Machine Theory **28**, 207–224 (1993)

10.53 S. Howard, M.J. Zefran Kumar: On the 6×6 cartesian stiffness matrix for three-dimensional motions, Mechanism and Machine Theory **33**, 389–408 (1998)

10.54 L. Meirovitch: *Fundamentals of vibrations* (McGraw-Hill, Boston–London 2001)

Industrial R

42. Industrial Robotics

Martin Hägele, Klas Nilsson, J. Norberto Pires

Most robots today can trace their origin to early industrial robot designs. Much of the technology that makes robots more human-friendly and adaptable for different applications has emerged from manufacturers of industrial robots. Industrial robots are by far the largest commercial application of robotics technology today. All the important foundations for robot control were initially developed with industrial applications in mind. These applications deserve special attention in order to understand the origin of robotics science and to appreciate many unsolved problems that still prevent the wider use of robots in manufacturing. In this chapter we present a brief history and descriptions of typical industrial robotics applications. We show how robots with different mechanisms fit different applications. Even though robots are well established in large-scale manufacturing, particularly in automobile and related component assembly, there are still many challenging problems to solve. The range of feasible applications could significantly increase if robots were easier to install, to integrate with other manufacturing processes,

and to program, particularly with adaptive sensing and automatic error recovery. We outline some of these remaining challenges for researchers.

Industrial robots are considered as a cornerstone of competitive manufacturing, which aims to combine high productivity, quality, and adaptability at minimal cost. In 2007 more than one million industrial robot installations were reported, with automotive industries as the predominant users with a share of more than 60% [42.1]. However, high-growth industries (in life sciences, electronics, solar cells, food, and logistics) and emerging manufacturing processes (gluing, coating, laser-based processes, precision assembly etc.) will increasingly depend on advanced robot technology. These industries' share of the number of robot installations has been growing steadily.

42.1 A Short History of Industrial Robots	964
42.2 Typical Applications and Robot Configurations	969
42.2.1 Welding	969
42.2.2 Car Body Assembly	969
42.2.3 Painting.....	971
42.2.4 Material Transfer Automation	971
42.2.5 Machining.....	974
42.2.6 Human-Robot Cooperation for Handling Tasks.....	974
42.3 Kinematics and Mechanisms	975
42.4 Task Descriptions – Teaching and Programming	976
42.5 End-Effectors and System Integration	980
42.6 Conclusions and Long-Term Challenges ..	983
References	985

The production of industrial robots on the one hand, and the planning, integration, and operation of robot workcells on the other hand, are largely independent engineering tasks. In order to be produced in sufficiently large quantities, a robot design should meet the requirements for the widest set of potential applications. As this is difficult to achieve in practice, various classes of robot designs regarding payload capacity, number of robot axes, and workspace volume have emerged for application categories such as assembly, palletizing, painting, welding, machining, and general handling tasks.

Generally, a robot workcell consists of one or more robots with controllers and robot peripherals: grippers

or tools, safety devices, sensors, and material transfer components for moving and presenting parts. Typically, the cost of a complete robot workcell is four times the cost of the robots alone.

A robot workcell is usually the result of customized planning, integration, programming, and configuration, requiring significant engineering expertise. Standardized engineering methods, tools, and best-practice examples have become available to reduce costs and provide more predictable performance [42.2].

Today's industrial robots are mainly the result of the requirements of capital-intensive large-volume manufacturing, mainly defined by the automotive, electronics, and electrical goods industries. Future industrial robots will not be a mere extrapolation of today's designs with respect to features and performance data, but will rather follow new design principles addressing a wider range of application areas and industries. At the same time, new technologies, particularly from the information technol-

ogy (IT) world, will have an increasing impact on the design, performance, and cost of future industrial robots.

International and national standards now help to quantify robot performance and define safety precautions, geometry, and media interfaces. Most robots operate behind secure barriers to keep people at a safe distance [42.3]. Recently, improved safety standards have allowed direct human–robot collaboration, permitting robots and human factory workers to share the same workspace [42.4].

We will first present a historical introduction to industrial robotics with a selection of contemporary application examples, then the basic principles that are used in industrial robotics and a review of programming methods will be presented. We will also discuss tools (end-effectors) and system integration requirements. The chapter will be closed with the presentation of selected, unsolved problems that currently inhibit the wider application of industrial robots.

42.1 A Short History of Industrial Robots

The invention of the industrial robot dates back to 1954 when George Devol filed a patent on a *programmed article transfer* (Fig. 42.1). After teaming up with Joseph Engelberger, the first robot company, Unimation, was founded and put the first robot into service at a General Motors plant in 1961 for extracting parts

from a die-casting machine. Most of the hydraulically actuated *Unimates* were sold through the following years for workpiece handling and for spot-welding of car bodies [42.5]. Both applications were successful, which means that the robots worked reliably and ensured uniform quality. Soon, many other companies

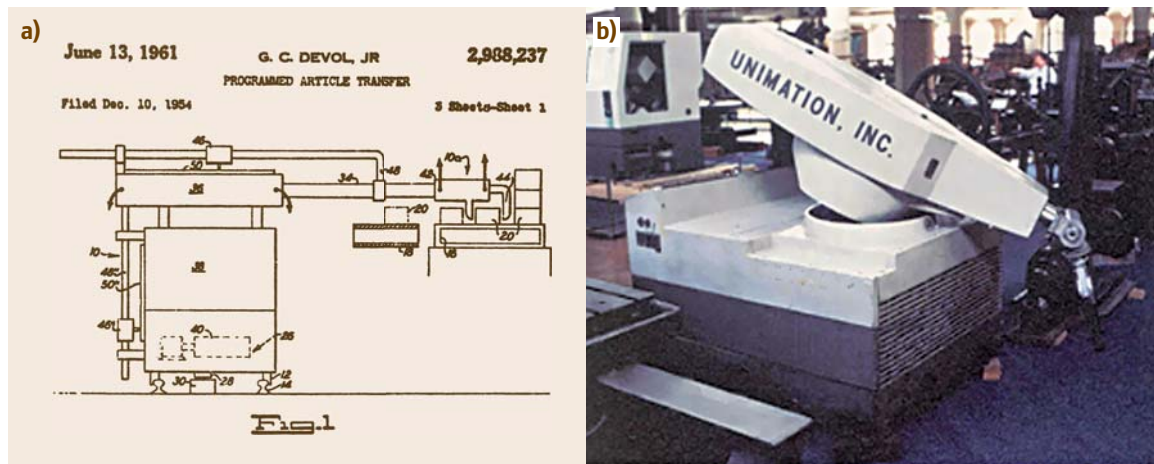


Fig. 42.1a,b The invention of the industrial robot (a) This patent was the start of a joint effort of G. Devol and J. Engelberger to form the first robot company, Unimation, a fusion of the words universal and automation. The company was acquired by Westinghouse in the late 1980s. (b) The first Unimation performed a rather simple handling task in 1961 at a General Motors plant. Other car manufacturers followed. The photo shows the first robot installed at Ford from their Museum in Dearborn

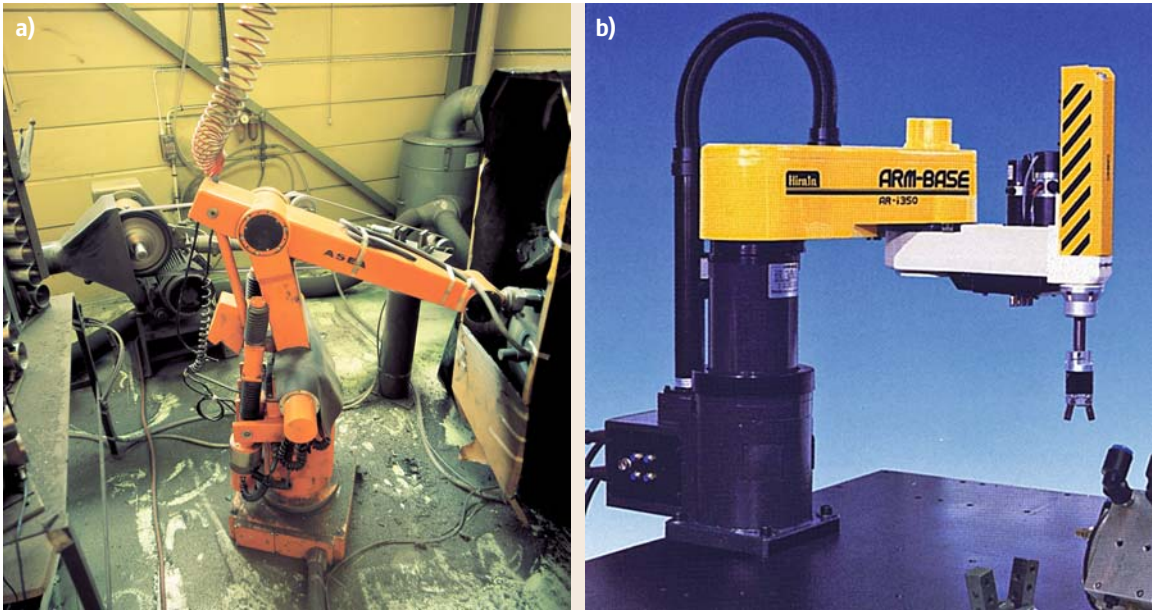


Fig. 42.2a,b The all electric (a) IRB-6 and (b) a SCARA-type kinematic (a) First introduced in 1973, the IRB-6 has been a breakthrough development as it was the first serially-produced robot product, which combined all-electric-drives technology and a microcomputer for programming and motion control. The robot proved very robust. Life-times of more than 20 years in harsh productions were reported (by courtesy ABB Automation, Friedberg) (b) The *selective compliance assembly robot arm* (SCARA) is particularly suited for assembly tasks as it combines rigidity in the vertical axis and compliance in the horizontal axis. In 1978, the first Hirata AR-300 was put together. Depicted is the successor design, the AR-i350. The SCARA design combines three or four rotational and one translational axis (by courtesy HIRATA Robotics, Mainz)

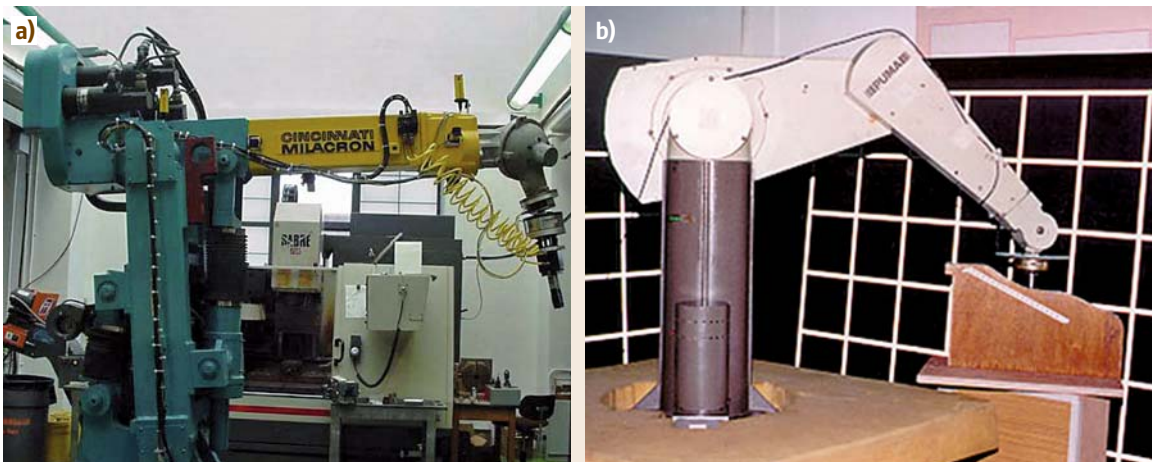


Fig. 42.3a,b Cincinnati Milacron T3 and the Unimation PUMA 560 (a) In 1974, Cincinnati Milacron introduced the first microcomputer controlled robot. The first T3 (*The Tomorrow Tool*) models used hydraulic drives, later they were replaced by electric motors as shown in the photo. The CM robotics division was acquired by ABB in the late 1970s (b) This 6 axis PUMA (programmable universal machine for assembly) came close to the dexterity of a human arm. After its launch in 1979 by Unimation it became one of the most popular arms and was used as a reference in robotics research for many years

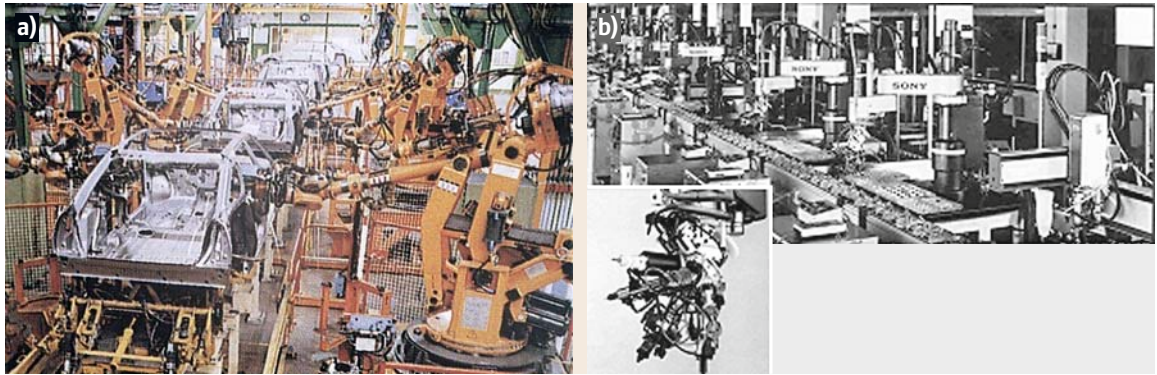


Fig. 42.4a,b Spot-welding line and videocassette recorder (VCR) assembly as pioneering applications for industrial robot applications. (a) Spot welding quickly became a primary application for robots as these jobs were particularly exhausting and hazardous for workers. A typical car body welding line from 1985 is displayed. The car model shown is a French Citroën CX (b) An automated VCR assembly line (ca. 1989) with SCARAs carrying a turret with multi-gripper tools. Typically five parts are added by one robot before the VCR is moved to the next station of the fully automated assembly line

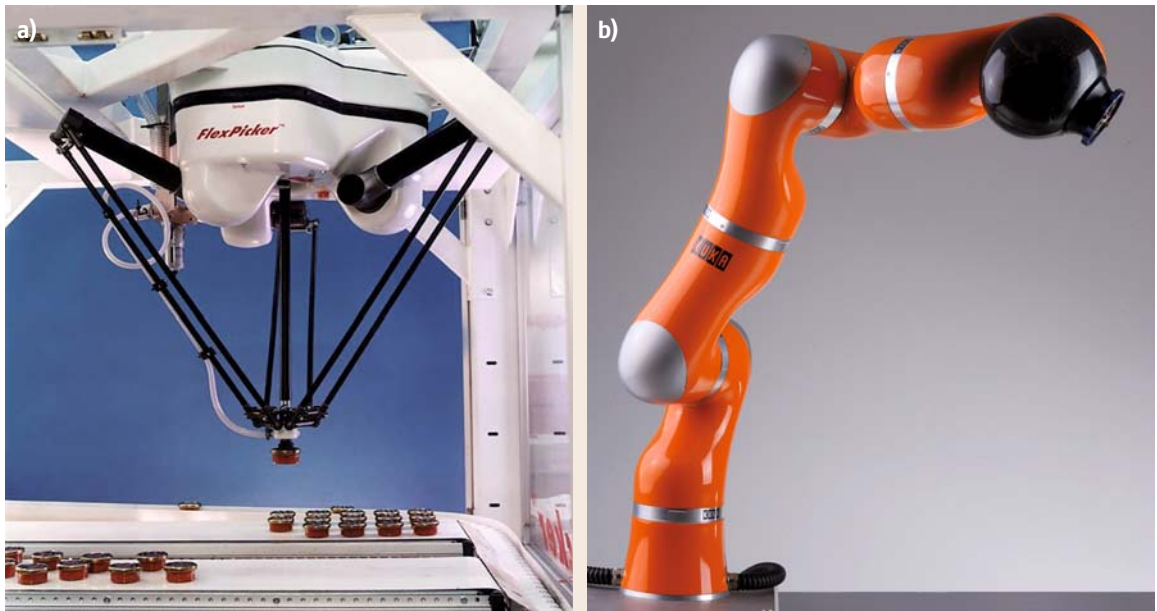


Fig. 42.5a,b The ABB FlexPicker parallel robot for high speed picking and the KUKA light weight robot arm (a) *Parallel kinematic machines (PKM)* represent an interesting approach to achieve high stiffness at low inertia, thus allowing accurate, high speed motions. Initially suggested by Clavel this 4 axis robot (called Delta) is used for high speed pick-and-place tasks. The robot reaches accelerations of up to 10 g (b) The KUKA light-weight robot is the result of a long research and development process by DLR, towards an arm design with a weight-to-payload ratio of 1:1. The 7 axis arm which is suited for human-robot cooperation imitates the dexterity of a human arm

started to develop and manufacture industrial robots. An innovation-driven industry was born. However, it took many years until this industry became profitable.

The breakthrough Stanford Arm was designed as a research prototype in 1969 by Victor Scheinman (see Chap. 3), at that time a mechanical engineering stu-

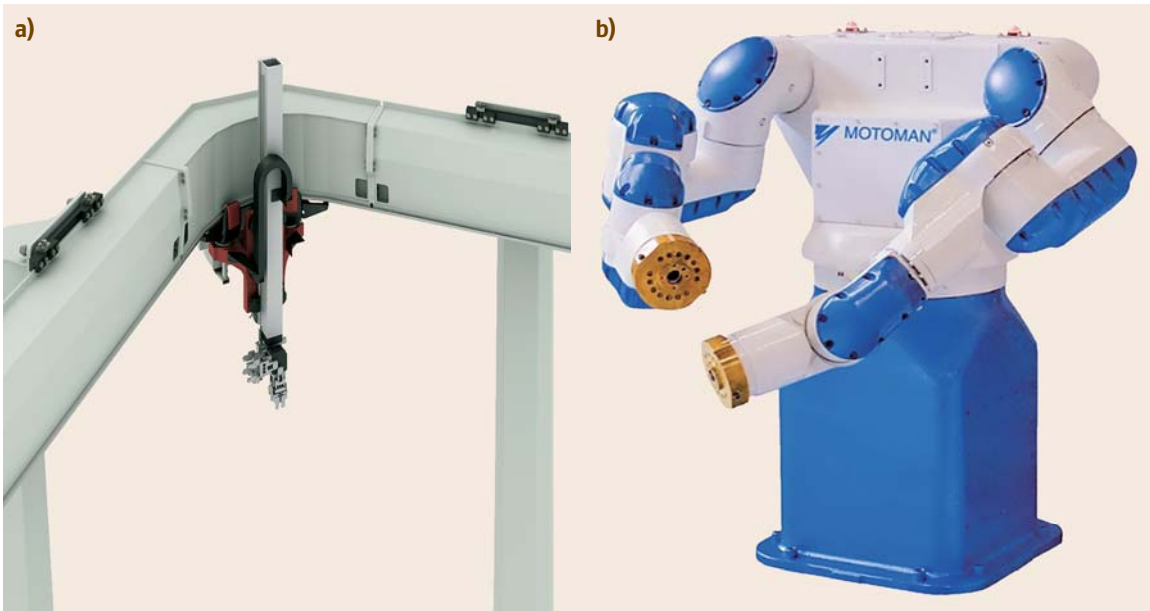


Fig. 42.6a,b Cartesian robot mobility and two-armed robot dexterity (a) The roboLoop of Güdel is a curved-track gantry and transfer system. One or more robot arms circulate as carriers in a closed transfer system. The system can be installed suspended, in gantry configuration, or as a floor-standing system. A signal bus allows the control and coordination of multiple robo-carriers (by courtesy Güdel, Langenthal) (b) Motoman's DA-20 dual-arm robot provides high-speed motion with two six-axis arms that provide enhanced, human-like flexibility of movement. The robot also provides jig-less operation with one robot arm holding a part while the other performs operations on the held part (by courtesy Motoman, Kalmar)

dent working in the Stanford Artificial Intelligence Laboratory (SAIL) [42.6]. The six-degree-of-freedom (6-DOF) all-electric manipulator was controlled by a standard computer, a Digital Equipment PDP-6. The non-anthropomorphic kinematic configuration with one prismatic and five rotational joints was configured such that the equations for solving the robot kinematics were simple enough to speed up computations. Drives consisted of direct-current (DC) electric motors, harmonic drive and spur gear reducers, potentiometers and tachometers for position and velocity feedback. Subsequent robot designs were strongly influenced by Scheinman's concepts (Figs. 42.2 and 42.3).

In 1973, the company ASEA (now ABB) introduced the first microcomputer-controlled all-electric industrial robot, the IRB-6, which allowed continuous path motion, a precondition for arc-welding or machining (Fig. 42.2). The design proved to be very robust and robot lifetimes of more than 20 years were reported [42.7]. In the 1970s intense diffusion of robots into car manufacturing set in mostly for (spot-)welding and handling applications (Fig. 42.4).

In 1978, the *selective compliance assembly robot arm* (SCARA) was invented by Hiroshi Makino of Yamaguchi University, Japan [42.8]. The ground-breaking four-axis low-cost design was perfectly suited for small parts assembly as the kinematic configuration allows fast and compliant arm motions (Fig. 42.2). Flexible assembly systems based on the SCARA robot in conjunction with compatible product designs (design for assembly, DFA) have contributed significantly to creating a worldwide boom in high-volume electronics production and consumer products [42.9].

Requirements regarding a robot's speed and weight have led to novel kinematic and transmission designs. From early on, the reduction of the mass and inertia of robot structures was a primary research target, where the human arm with a weight-to-load ratio of 1:1 was considered the ultimate benchmark. In 2006, this goal was reached in the form of the company KUKA lightweight robot, a compact 7-DOF robot arm with advanced force-control capabilities [42.10]. Another approach towards lightweight and stiff structures has been pursued since the 1980s by developing parallel kinematic machines

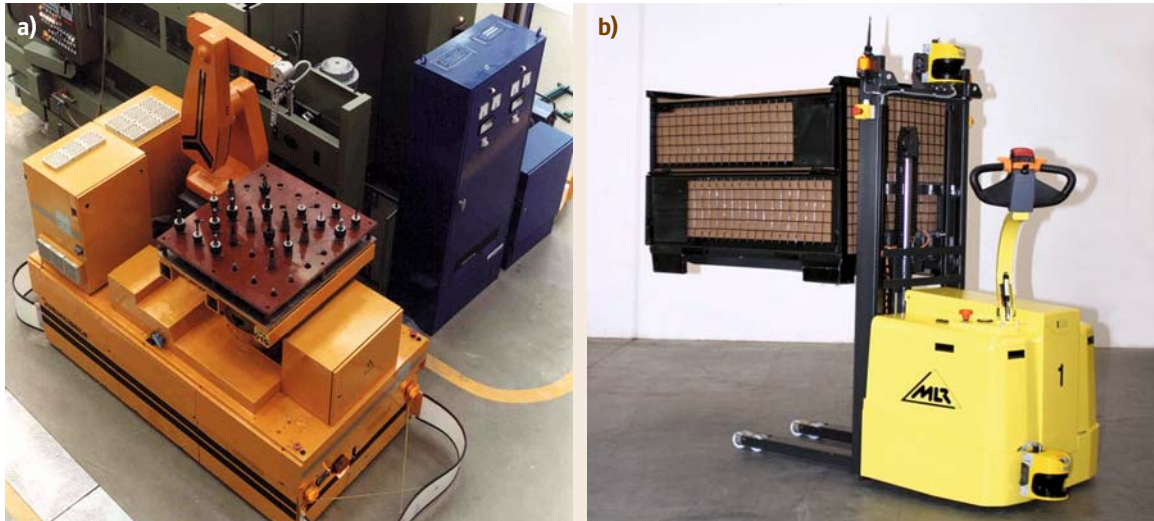


Fig. 42.7a,b A mobile robot arm for machine (un)loading and an autonomous fork lift (a) The MORO (mobile robot, 1984), developed at Fraunhofer IPA, Stuttgart, was one of the first prototypes to combine a robot arm on a wire-bound mobile platform which follows a wire buried in the floor. Mobile arms were particularly used in highly automated clean room manufacturing facilities to automatically load and unload process equipment (b) This automated fork-lift by MLR combines manual operation with fully autonomous navigation and (un-)loading of boxes and pallets. A laser scanner using retro-reflecting tapes or other landmarks (walls, pillars) avoids constraints imposed by simple buried wire guidance technology (by courtesy MLR System, Ludwigsburg)

which connect the machine's basis with its end-effector by three to six parallel struts [42.11]. These so-called parallel robots (see Chaps. 3 and 12) are particularly suited to achieve the highest speeds (e.g., for picking), precision (e.g., for machining), or handling high work loads (Fig. 42.5). However, workspace volumes tend to be smaller than those of serial or open kinematic chain robots which are comparable in size.

Still, Cartesian robots are ideally suited for covering large workspaces. Apart from the traditional design using three orthogonal translational axes, the company Güdel introduced a curved-track gantry in 1998. The concept allows one or more robot arms to track curves and to circulate in a closed transfer system. Thus, the workspace of robot can be immensely increased at high speed and precision. This may be particularly valuable in logistics or machine tending [42.12].

Two-handed dexterous manipulation can be critical for complex assembly tasks, simultaneous handling and processing of workpieces, or the handling of large objects. The first commercial robot for synchronized, two-handed manipulation was introduced by MOTOMAN in 2005 [42.13]. As a dual-arm robot imi-

tates the reach and dexterity of human arms, it can be placed on a site that previously accommodated human workers. Thus, capital expenditure can be reduced. It features 13 axes of motion: six axes per arm, plus a single axis for the base rotation (Fig. 42.6).

In parallel to industrial robots *automated guided vehicles* (AGVs) have emerged. These mobile robots are used for moving workpieces or loading equipment from point to point. Within the concept of automated *flexible manufacturing systems* (FMS) AGVs have become an important part of their routing flexibility. Initially AGVs relied on prepared floors such as embedded wires or magnets for motion guidance. Meanwhile, freely navigating AGVs are used in large-scale manufacturing and logistics. Usually, their navigation is based on laser scanners that provide an accurate two-dimensional map of the actual environment for self-localization and obstacle avoidance [42.14]. Early on, combinations of AGVs and robot arms were realized to automatically load and unload machine tools (Fig. 42.7). Only in some selected environments such as (un-)loading process equipment in the semiconductor industry these mobile arms were economically advantageous.

By 2007, the evolution of industrial robots was marked by the following main trends [42.15]:

- The average robot unit price fell to about one-third of its equivalent price in 1990, which meant that automation is becoming more affordable. At the same time, robot performance parameters such as speed, load capacity, and mean time between failures (MTBF) have dramatically improved. These improvements provide a faster return on investment, particularly for small, short-run batch production.
- Off-the-shelf components from personal computer (PC) technologies, consumer software, and the IT industry have contributed to improved performance-to-cost ratios. Today, most manufacturers integrate PC-based processors in their controllers as well as software for programming, communication, simulation, and maintenance from high-volume IT-markets.
- Multiple robots can be programmed and synchronized in real time by one controller, which allows robots to cooperate precisely on a single workpiece.
- Increasingly, vision systems for object identification, localization, and quality control have become an integral part of the robot controller.
- Robots are networked by fieldbuses or ethernet for control, configuration, and maintenance.
- New financing arrangements allow end-users to rent a robot or even have a robot workcell operated by a specialized company or even the robot supplier in order to reduce risks or to save on investment capital.
- Training and education have become important services to end-users to increase acceptance of robot technology. Specific multimedia material and courses aim at educating industrial engineers and workforce to effectively plan, program, operate, and maintain industrial robot workcells.

42.2 Typical Applications and Robot Configurations

42.2.1 Welding

Welding ranks among the most important joining processes in manufacturing. Manual welding requires highly skilled workers, as small imperfections in the weld can lead to severe consequences. Why is a robot suited to perform this critical job? Modern welding robots have the following characteristics:

- Computer control allows the programming of task sequences, robot motions, external actuators, sensors, and communication with external devices.
- Free definition and parameterization of robot positions/orientations, reference frames, and trajectories.
- High repeatability and positioning accuracy of trajectories. Typically repeatability is some ± 0.1 mm and positioning accuracy is of the order of ± 1.0 mm.
- High speeds of the end-effector of up to 8 m/s.
- Typically, articulated robots have six DOF so that commanded orientations and positions in their workspace can be reached. Workspace extensions by mounting the robot on a linear axis (seventh DOF) are common, especially for welding of large structures.
- Typical payloads of 6–100 kg.
- Advanced programmable logic controller (PLC) capabilities such as fast input/output control and synchronizing actions within the robot workcell.
- Interfacing to high-level factory control through fieldbuses or ethernet.

Metal inert/active gas (MIG/MAG) welding is the predominant application of industrial robotics today. The automatic arc-welding process is based on a consumable wire electrode and a shielding gas that are fed through a welding gun (Fig. 42.8). Electric current sources, torches, and peripheral devices for automatic cleaning and maintaining the torch (anti-splatter, wire-cutting, tool changer etc.) are offered by specialized companies. Often sensors are used to track welding gaps and measure weld seams either before or synchronously to the welding process, thus adapting the robot's trajectory in the presence of workpiece variation and distortion. Also, cooperating robots have been introduced where one robot fixes and moves the workpiece in synchronization with another robot carrying a welding tool so that the weld can be performed with the pool of molten metal horizontal.

Another interesting robot task is multilayer welding. In regular intervals the robot measures the profile of the weld gap and adaptively generates subsequent tool paths to apply successive weld seams until the final required geometry is reached. In the example shown in Fig. 42.8c up to 70 seams can be produced by the robot to weld thick metal plates.

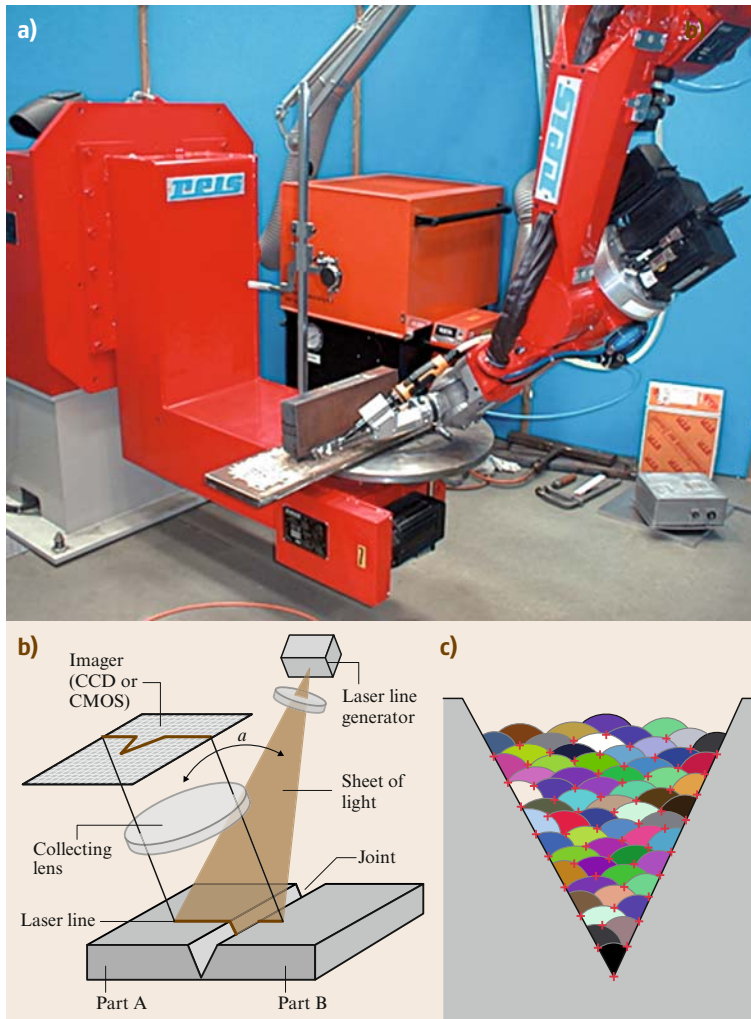


Fig. 42.8a–c Robot welding with sensor guidance. **(a)** The welding robot workcell consists of a 6 DOF robot, a turning table so that the seam is accessible and welded in a flat position, the weld source with a fume extractor and the welding torch. Safety fences and light shields are not shown. The welding torch is attached to the robot flange. A service station (not shown) in the robot workcell is approached regularly to clean the gas nozzle. **(b)** A typical weld seam sensor is based on the so-called laser triangulation principle. The sensor is attached to the robot and a laser projects a sharp stripe on the seam. The stripe is detected by an imager, typically a 2-D CCD camera. From the extracted contour the position of the seam relative to the sensor is calculated. **(c)** Multiple seams can be generated based on this information

42.2.2 Car Body Assembly

Early on, car body assembly became the predominant robot application as the benefit for robot automation was apparent. Handling and positioning the metal sheets, spot welding, and transport of the body frames was either hazardous, physically demanding to the worker, or difficult to realize on fixed automation lines given the desired variety of car body configurations to be assembled on the one production line (Fig. 42.9). In the stamping section metal sheets are cut into plates (or *blanks*) ready for pressing into body panels. In subsequent steps robots load the panels onto a tray that fixes the panels for other robots to be spot-welded. After inspection the finished bodies are transferred by conveyor to the paint shop.

Individual assembly units and components such as motors, transmissions, axles, doors or fenders are pre-mounted in separate areas. The car body is delivered to the body assembly area *just in sequence*, i. e. at the right time and place on the assembly line. The climax of the assembly process (wedding) – when the engine, drive and chassis first meet takes place before the last parts are mounted on the car.

Today's industrial robots, particularly in the workload category of 100–300 kg, are to a large extent the results of the requirements stemming from this application:

- Required repeatability of at least ± 0.5 mm under typical loads of some 100 kg for the spot-welding

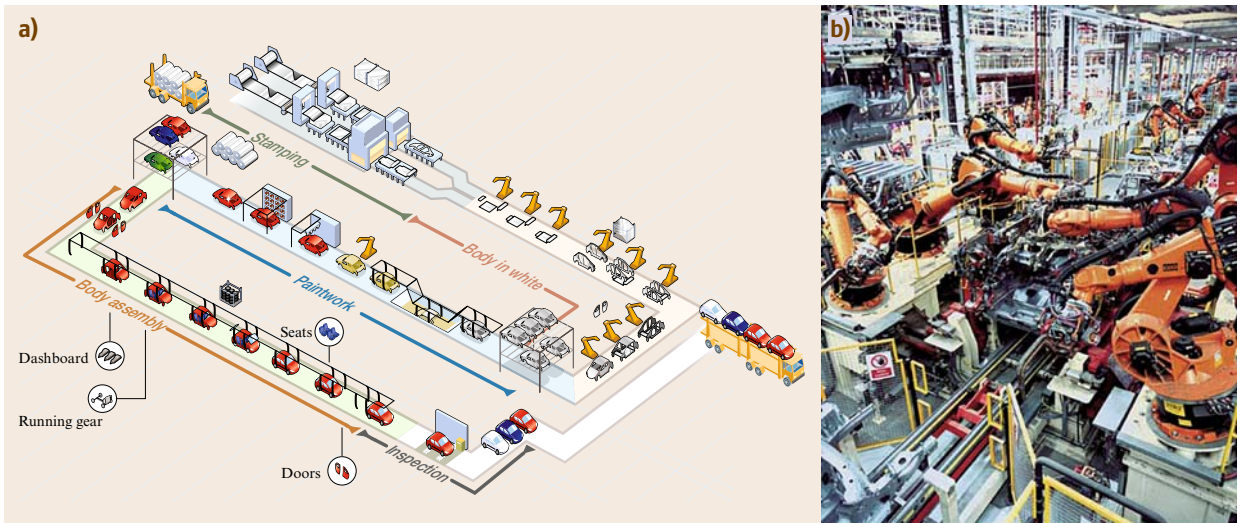


Fig. 42.9a,b Car body assembly. **(a)** A car body assembly usually follows the illustrated steps: Stamping of the metal sheet into plates, fixing and alignment of the plates on trays, spot welding, painting the car body, and final assembly of the car body (doors, dashboard, windcreens, power-train seats, and tires). Car factories can host well over 1000 robots working two to three shifts per day. (Courtesy PSA Peugeot Citroën, Paris and Art Movies, Paris) **(b)** The Mercedes A class assembly in Rastatt Germany is highly automated. The picture shows spot welding robots along the, *body in white* transfer line. Trays carrying car bodies pass through the *robot garden*

tool and cable package leads to stiff and heavy arm structures. A typical robot weight-to-payload ratio is of the order of 1500 kg:150 kg.

- Three-shift continuous operation requires highest reliability of both robot and equipment. Typical mean time between failure (MTBF) is reported to be around than 50 000 h [42.1].
- The line capacity depends on the robot's speed to place spot welds. Thus, the *point-to-point* (PTP) movement time between welding positions should be kept as short as possible. This is mainly achieved by powerful drives so that the mean power consumption of the robot drives is typically 5 kW.
- Most of the trajectories, positions, and orientations are generated using *offline programming* (OLP) systems. Accurate simulations of the robot motion depend on robot models, which incorporate both robot kinematic properties and controller characteristics. The *realistic robot simulation* (RSS) interface is the accepted standard format for robot models of offline programming systems [42.16].

42.2.3 Painting

Hazardous working conditions for human operators motivated Trallfa, a Norwegian company, to develop simple

spray-painting robots in 1969, particularly for spraying bumpers and other plastic parts in the automotive industry. Initially pneumatically driven for antiexplosion reasons, today's robot designs are fully electric with greatly improved spray guns. They also have hooks and grippers to open hoods and doors during the painting task. Hollow wrists that house gas and paint hoses allow fast and agile motions. Spray guns for robots have evolved dramatically for delivering uniform quality using as little paint and solvent as possible and also for switching between different paint colors. Originally spray-painting robots replicated movements copied from human workers. Most of the programming for robot painting today is done offline as state-of-the-art programming systems offer integrated process simulations to optimize paint deposition, thickness, and coverage (Fig. 42.10).

42.2.4 Material Transfer Automation

Generally, industrial practice in robot workcell planning aims at finding a compromise between reducing the variation of the workpiece position and the cost of sensor systems to compensate for residual variation. Nearly all parts arrive at robot workcells in a repeatable manner, either being stored in special magazines, or by being

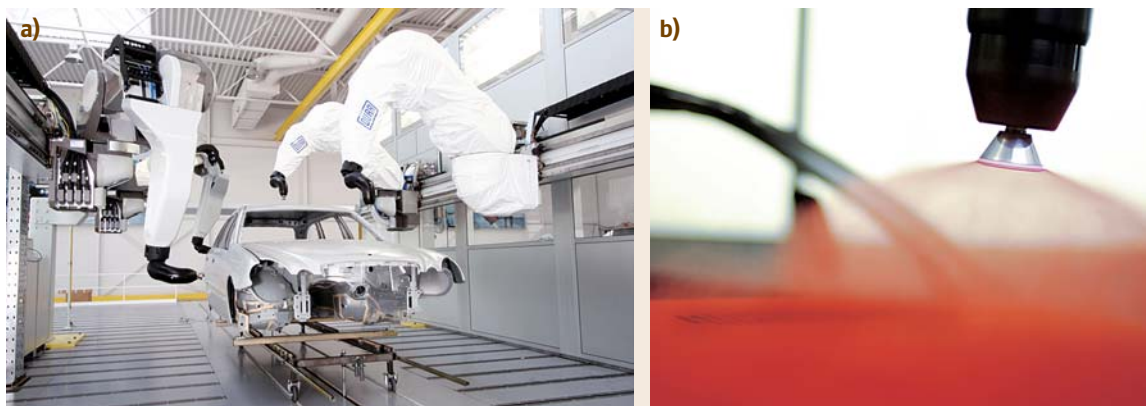


Fig. 42.10a,b Painting robots. (a) A multi-robot workcell for car body painting (b) High-speed rotating atomizer and charge. Painting robots. (a) are used for surface coating of car bodies and other parts. All current paint materials such as solvent-based paints, water-borne paints and powder can be applied. In car production, multiple robots work in parallel for optimal throughput and accessibility of the car body. Most of the programming which includes the synchronisation of the robots is performed offline. Paint guns are critical to the process quality. Figure (b) shows a Dürr EcoBell2 paint gun which atomizes the paint material at the edge of the rotating bell disk by centrifugal forces (by courtesy Dürr, Stuttgart)

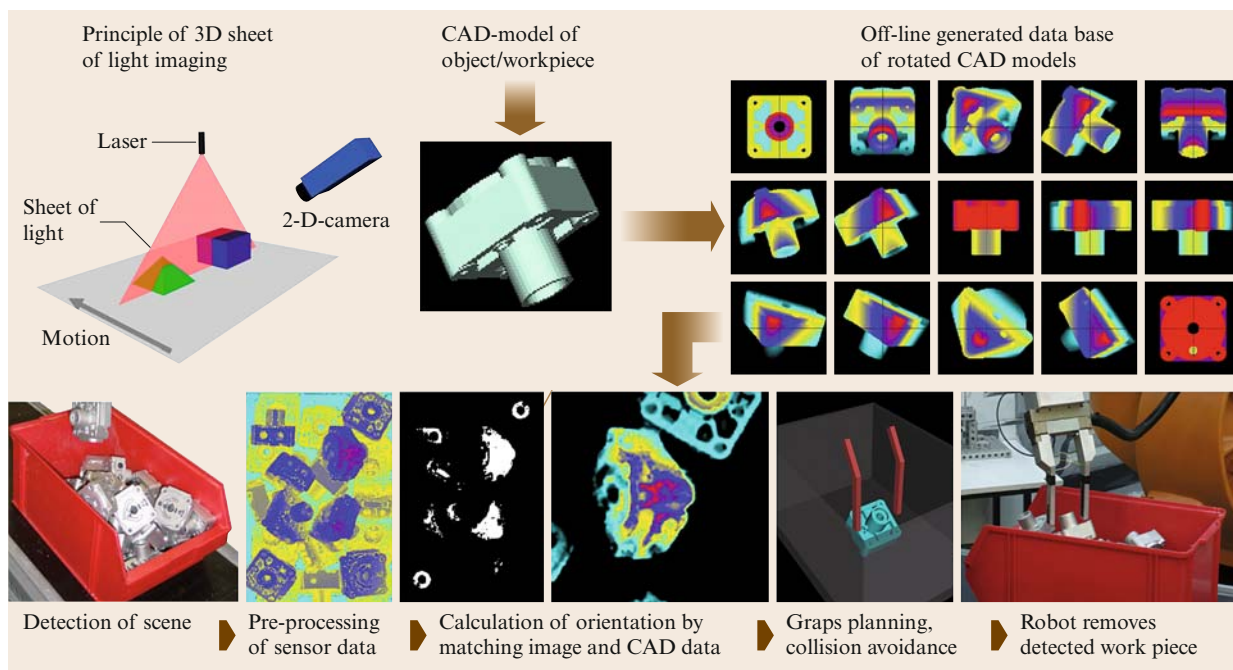


Fig. 42.11 Robot bin picking. A scene containing the objects is acquired by a 3-D sensor e.g. based on the laser triangulation principle. Beforehand this approach turns the CAD object virtually in discrete spatial angles in an off-line calculation. Feature histograms for each view are generated and stored in a database. A best match between actual feature histograms and the simulated sets of histograms determines the location of the object. A grasp has to be selected and a collision-free trajectory is generated. A typical cycle time of a location process is between 1 and 2 s

transported by vibrating devices that allow the parts to settle into a predictable orientation.

If randomly oriented in a carrier or a box, parts have to be properly located so that the robot can grasp them appropriately. This challenge in industrial robotics has been referred to as *bin-picking* and has been investigated by numerous researchers since the mid-1980s [42.17]. Even though an abundance of approaches has been presented a cost-effective standard solution has not been established yet [42.18].

Barely 10% of industrial robot installations in 2006 were sensor-equipped. However, it is expected that many

future robots will have standard embedded force-torque and vision sensors. This is a precondition for advanced vision for object identification and localization in manufacturing.

One method for determining random object locations using the object's computer-aided design (CAD) data starts with point clouds of a scene acquired by a three-dimensional (3-D) sensor (Fig. 42.11). A selection of 3-D sensors is described in Chap. 22. The CAD model has been virtually turned into discrete spatial angles in an offline calculation. Feature histograms for each view are generated and stored in a database with

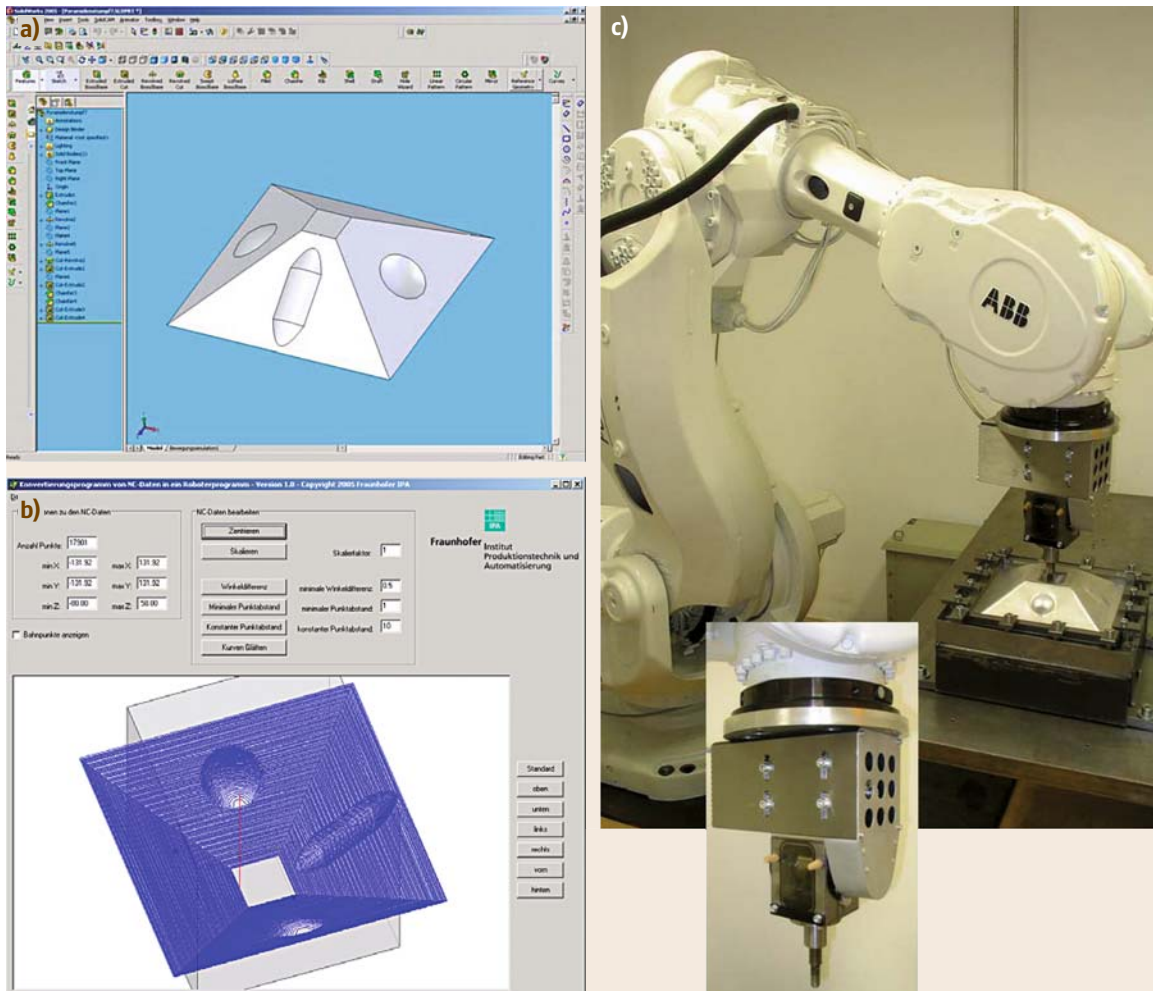


Fig. 42.12a–c Trajectory generation for incremental machining processes. In this example, the forming process of metal sheets is based on an oscillating stamp (amplitude 1 mm, 50 Hz frequency) which locally plastifies the metal in incremental steps. From the CAD model (a), the robot's trajectories are calculated offline on the basis of specific material models (b). Each line represents a part of the tool trajectory. (c) Shows a demonstrator robot workcell with the tool in detail

the given angular information (typically in 10° steps for all spatial angles).

A best-match process compares actual feature histograms with the simulated sets of histograms in the database. For the determined location of the object, a grasp has to be selected and a collision-free trajectory be generated. The latter steps can be quite critical as residual workpieces at the box bottom may constrain the robot's grasp and departing trajectory [42.19].

42.2.5 Machining

Compared to a milling machine or a lathe, standard robots possess much less stiffness (by a factor of 20–50 times), but much greater dexterity. A serial robot's stiffness is usually very anisotropic throughout its working space and may vary for a typical heavy duty model in the range 200–700 N/mm. Therefore, robots can machine workpieces (grinding, fettling, polishing etc.) provided that tool forces can be reduced to acceptable values for a given robot manipulator. This incremental approach to machining, particularly for cutting and forming, can produce good results. However, these sequential robot motions have to be generated automatically, which requires merging the process information with the workpiece geometry.

An example of a novel process that benefits from the robot's versatility in terms of programming, dexterity, and cost is the so-called incremental forming process of metal sheets. Without using any costly form, metal

sheets are clamped in a rigid frame and the robot produces a given 3-D contour by guiding a tool equipped with a high-frequency oscillating stamp (amplitude typically 1 mm at a frequency of 50 Hz) over the metal surface. The robot's trajectories are calculated from the CAD model on the basis of specific material models. The robot's program is generated and passed to the controller. One-off housings for machines, prototype panels or customized car panels can be economically produced using this method [42.20]. Figure 42.12 depicts the sequence of actions for automatically generating and executing the forming program.

42.2.6 Human–Robot Cooperation for Handling Tasks

Robots for human augmentation (force or precision augmentation) stretch from fully automated operation to acting as a servo-controlled balancer (see also Chap. 57). As an example: in a car drive train assembly the heavy gear box is grasped by the robot which balances it softly so the worker can insert it precisely into the housing (Fig. 42.13). Preprogrammed virtual walls give the worker a realistic feeling of constraints.

The central interface for the worker's tactile commands is a handle equipped with safety switches. These switches trigger the force-torque sensor that is attached to the robot's flange. Upon touching both safety switches (two-handed operation) the force-torque sensor is activated and the robot control is set to a safe



Fig. 42.13 Human-robot-cooperation for handling tasks. Inside a regular workcell which is secured by light curtains, the robot handles gear boxes at regular speed in fully automated mode. Upon approaching the light curtain at reduced speed, the worker grasps the safety switch which activates both the reduced-speed mode and the force-torque sensor. The worker guides the robot almost effortlessly by its handle so that the gear-box is balanced with precision into the rear axle frame for final assembly (Fraunhofer IPA, Stuttgart)

reduced end-effector speed (of some 50 mm/s). Thus, the sensor scales the applied force/torque information into a compliant robot motion. Obviously the physical human–robot cooperation has to obey safety precautions as the robot’s and worker’s workspaces overlap.

The ISO 10218-1:2006 standard specifies requirements and guidelines for the inherent safe design, protective measures, and information for use of industrial robots. It describes basic hazards associated with robots, and provides requirements to eliminate or adequately reduce the risks associated with these hazards. A novel element of this revised standard is the regulation of so-called collaborative operation, where the robot works in direct cooperation with a human within a defined workspace. Basically the collaborative operation depends on several criteria, which have to be met by the robot workcell [42.21]:

1. The hand-guiding equipment shall be located in the area of the end-effector.
2. The robot moves with safe reduced speed (less than 250 mm/s) and safe monitored position. This position monitoring shall be according to at least category 3 of ISO 13849, unless a risk assessment is performed and determines that a higher category is required [42.22, 23].
3. The robot must sense and keep a safe distance from the human. The distance relates to the attended

speed. The distance and speed monitoring shall be according to at least category 3 of ISO 13849, unless a risk assessment is performed and determines that another category is required.

If a robot’s total power consumption can safely be limited to 80 W as well as its static impact forces to 150 N no additional sensor-based safety precautions are needed. Again, these conditions have to be secured by a risk assessment or security systems which comply with at least category 3 of ISO 13849 [42.21]. Currently, novel control algorithms, kinematic and actuator designs are being investigated to realize so-called intrinsically safe robots [42.24].

The described workcell complies with the ISO standard in such a way that the presence of the human is detected by activating both safety switches of the handle (first condition). The robot’s built-in safe controller safely measures the end-effector’s position and limits its velocity [42.25] (second condition). Meanwhile most of the robot manufactures provide safe category 3 controllers. If the worker’s presence in the workspace is not known (third condition, not applicable here) a safe sensor system has to detect safely the workers location according to category 3. Three-dimensional sensors meeting category 3 first appeared on the market in 2006, thus opening up a wide field of potential collaborative operations [42.26].

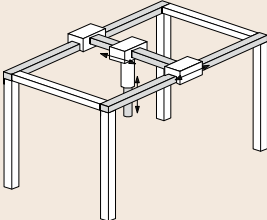
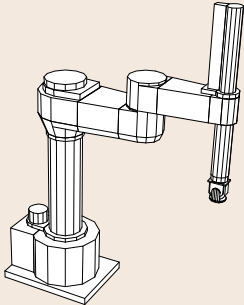
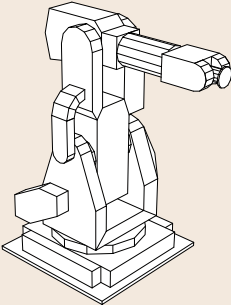
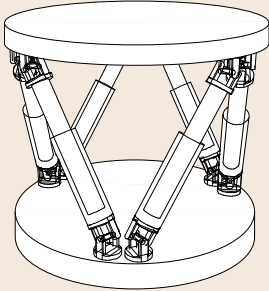
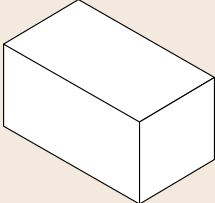
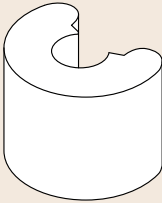
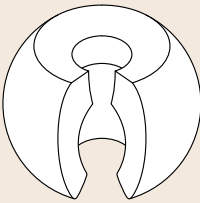




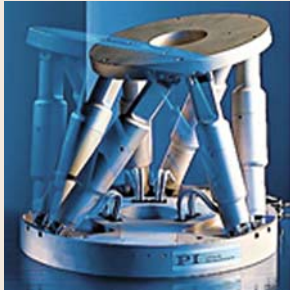
42.3 Kinematics and Mechanisms

The choice of mechanism, its kinematic properties, the computation methods used to determine joint motions, and the intended application of a robot manipulator are all closely related. The diagrams in Table 42.1 show several common types of robot mechanism.

With advances in the state of the art in kinematic algorithms and computer hardware processing capabilities, computation is much less of a constraint on mechanism choice than it was for early robot designers. The choice of mechanical structure of the robot depends mostly on fundamental mechanical requirements such as payload and workspace size. Considering a given level of cost, there is usually a tradeoff between workspace size and stiffness. To enable the robot to reach inside or around obstacles it is clearly advantageous to use an articulated mechanical design.

Considering also the stiffness and accuracy (in a practical sense considering what is reasonable to build), the picture is more complex. Each of the first three types in Table 42.1 we refer to as *serial kinematic machines* (SKMs), while the last is a *parallel kinematic machine* (PKM). To obtain maximum stiffness, again for a certain minimum level of cost, the end-effector is better supported from different directions, and here the PKM has significant advantages. On the other hand, if high stiffness (but not low weight and high dexterity) is the main concern, a typical computerized numerical control (CNC) machine (e.g., for milling) is identical in principle to the gantry mechanism. There are also modular systems with servo-controlled actuators that can be used to build both robots with purpose-designed mechanisms.

Table 42.1 Mechanical types of motions with different kinematic: Gantry is what a Cartesian coordinate robot is typically called, the **SCARA** is a *selective compliant articulated robot* (or *selective compliance assembly robot arm*, since the main application is planar assembly), the *Articulated* (all joints are rotary and placed in series after each other) robot is often referred to as an arm robot, and the *Parallel* is shown with prismatic joints, but can also have revolute joints such as the Delta robot (Fig. 42.5). Combinations are common

Gantry	SCARA	Serial articulated	Parallel
			
Form of workspace			
			
Product example			
			

42.4 Task Descriptions – Teaching and Programming

We cannot instruct a robot in the same way that one would instruct a human worker how to carry out a task. With skilled workers knowing the applications, devices, processes, and the general requirements on the product to be manufactured, we would only need to summarize *what* needs to be done.

In practice, since it is difficult to encode much of the required background knowledge, we aim for programming principles that resemble instructing a (totally) unskilled worker, telling precisely *how* every aspect of the task is to be performed. That requires a much more explicit way of instructing the robot, but one which is

still human-friendly. More specifically, as a wish-list that we will come back to at the end of the chapter, we would like to teach robots by

- manually guiding the robot to the positions of interest, or even along the desired paths or trajectories if human accuracy is enough
- having simple ways to make use of CAD data whenever available
- using different complementary modalities (paths of communication between the human and the robot), such as speech and gestures
- choreographing the task movements, for instance loops and conditions, without requiring extensive programming competencies
- means of describing acceptable variation, e.g., as expected or normal deviations from the nominal path
- specification of how external sensing should be used for new types of motions or for handling unknown variations

Initially, mainly during the 1970s and 1980s, there were some painting robots that could be programmed by manual guidance. This was possible due to the following abilities

- Applications such as painting permit the use of a lightweight arm, including the end-effector, possibly balanced with respect to gravity if needed.
- The accuracy requirements were (compared to today) modest so it was possible to use back-drivable drive trains and actuators, and the definition of the motions could then be done manually by the operator moving the end-effector along the path. The recorded poses, including the timing/speed information, then define the programmed motion.
- Since no inverse kinematics was needed during programming or real-time operation, it was not a problem from a computing point of view to use arm kinematics without singularities in the workspace.
- The requirements of optimality of painting motions were also modest, compared to recent years when environmental conditions (for nature and workers) calls for minimal use of paint.

It is often referred to as an inevitable problem that there are singularities to be handled within the workspace. However, to simplify the kinematics and its inverse from a software point of view (e.g., during the 1980s considering the power of the microprocessors and algorithms available at that time), robots were actually designed to have simple (to compute) inverse kinematics. For instance, the wrist orientation was decoupled

from the translation by the arm by using wrist axes that intersected with the arm axes. The resulting singularities within the workspace could be managed by restrictions on wrist orientations, but an unfortunate implication was that robot arms were no longer back-drivable (close to the singularities) when designs were (due to engineering and repeatability requirements) adopted to standard industrial controllers. Then with the development of microprocessor-based industrial controllers and the definition of motions based on jogging (manual moves by for instance using a joystick) and CAD data, still lacking efficient and robust techniques for human-like instructions (speech, gestures, etc.), the means of robot programming became closer to computer programming (extended with motion primitives).

Robot programming languages and environments have traditionally been separated into online programming (using the actual robot in situ) and offline programming (using software tools without occupying the robot). With the increasing power of offline programming tools, their emerging ability to connect to the physical robot, and the increasing level of software functions that are embedded into the robot control system, online programming is now unusual, except to verify and manually adjust programs generated offline. Still, of course, it is economically important to minimize downtime for robot programming, and advanced sensor-based applications may be too hard to develop without access to the true dynamics of the physical workcell for fine-tuning. Nevertheless, robot languages and software tools must provide for both methods of programming.

Even though robot languages from different manufacturers look similar, there are many semantic differences that have to do with both the meaning when programs are running (the robot performing its operations) and the way the robot is instructed. The need to ensure that existing robot programs can operate with replacement robots and controllers, and also to make use of existing knowledge in robot programmers and incorporated into offline programming software, requires manufacturers to continue to support their original proprietary languages. Features such as backward execution (at least of motion statements) and interactive editing during interpretation by the robot (in combination with restrictions to make the programming simpler) also make robot languages different from conventional computer programming languages.

During the last decade and in current developments, the trend has been to focus on the tool (robot end-effector) and on the process knowledge needed for the manufacturing process, and to let the operator express

Table 42.2 **Example:** Simple pick-and-place operation with a typical industrial robot controller

ABB Rapid (a proprietary language) program: positions obtained from a charge-coupled device (CCD) camera; end-effector tool is pneumatic. MoveJ denotes joint-space motion and MoveL denotes linear/Cartesian motion. Basically there are four arguments for such motions: target position (here relative, by use of the offset function), the maximum motion speed (here using predefined v-constants with the number specifying the desired speed in mm/s along the path), the desired accuracy before continuing (fine referring to no corner blending at all), and the tool being used (including frames, inertia, etc., here in the tool0 data record).	
PROC cam_pick()	
MoveJ app_point, v1500, z25, tool0;	// Preposition near the working table (using a 25 mm tolerance)
MoveJ Offs(camera1 1,x,y,-30), v1500, fine, tool0;	// Approach 30 mm above the object object, position x, y from webcam
temp1:=CRobT();	// Acquire current position
MoveL Offs(temp1,0,0,-38), v400, fine, tool0;	// Move to object: account for suction cup flexibility (8 mm)
Set DO08;	// Vacuum ON, pick object
WaitTime 1.5;	// Wait 1.5 s
MoveL Offs(temp1,0,0,10), v400, fine, tool0;	// Move up 10 mm
MoveJ app_point, v1500, z25, tool0;	// Move “fly-by” to box with 25 mm tolerance
MoveJ place_box, v1500, fine, tool0;	// Move to box position
Reset DO08;	// Vacuum OFF, release object
Set DO07;	// Blow ON
WaitTime 0.8;	// Wait 0.8 s
Reset DO07;	// Blow OFF
ENDPROC	

Table 42.3 Robot-related centricity, ranging from a high-level view of the work to be accomplished in terms of the product to manufacture, to a low-level robot motion view that, in practice, constraints what manufacturing operations can be performed

<div><div>User application</div><div>Robot constraint</div></div>	Product: Description of the final shape and assemblies of the workpieces, in terms of that product; the robot system plans the operations.
	Process: Based on known sequences of specific manufacturing operations, each of these is specified in terms of their processes parameters.
	Tool: The motion of the robot-held tool is specified in terms of programs or manual guidance; the user knows the process it accomplishes.
	Arm: The robot arm and its end-plate for tool mounting is programmed how to move in Cartesian space; the user knows the tool.
	Joint: For each specified location the joint angles are specified, so straight-line motions are difficult; the robot provides coordinated servo control.

the robot task in such terms. This development results in a need for an increased level of abstraction to simplify the programming, reflecting the fact that the so-called robot programmer knows the production process very well, but has quite limited programming skills. To understand why such a high level of abstraction has not come into widespread usage, we may compare this with

the early days of industrial robotics when there was no kinematics software built into the controllers, and hence the robots were programmed via joint-space motions. (Kinematics here deals with the relation between robot motors/joints and the end-effector motions.) Built-in inverse kinematics permits tool motion to be specified in Cartesian coordinates, which is clearly

a great simplification in many applications. That is, the robot user could focus more on the work to be carried out by the robot and less on the robot itself. However, robot properties such as joint limits cannot be neglected. Until the beginning of the 1990s, robots did not follow programmed trajectories very well at high speeds or accelerations, and full accuracy could only be achieved at low speed. Modern robot systems with high-performance model-based motion control perform their tasks with much greater accuracy at high speed due to model based control features, see Chap. 6 [42.27, 28].

There are increasing opportunities to raise the level of abstraction to simplify the use of robots even more by encoding more knowledge about the robot, tools, the process, and workcells into control systems.

Example 42.1:

A machine part consisting of plates and pipes of aluminium is to be manufactured, and there is robot equipment available for production of this (and other) types of workpieces.

- A *product*-centric system would generate configurations and robot programs, and instruct the operator for whatever manual assistance that might be needed (e.g., clamping and fixturing). The system would determine the welding data such as the type of welding and how many passes for each seam.
- A *process*-centric system would instead accept input in terms of the welding parameters to use, including the order of the welds. The system would select input signals to the process equipments (such as what output voltage the robot controller should set such that the welding will be done with a certain current), and robot programs specifying the motions of the weld pistol would be generated.
- A *tool*-centric way of programming would require the operator to set up the process manually by configuring the equipment in terms of their native settings, and appropriate tool data would be configured such that the robot controller can accomplish the programmed motions, which are specified by giving coordinates and motions data referring to the end-effector.
- An *arm*-centric system is similar to the tool-centric approach in that Cartesian and straight-line motions can be (and need to be) explicitly specified/programmed by the programmer, but extra work is needed since the robot does not support a general tool frame.
- A *joint*-centric style would be needed if one of the very early robot systems is being used, requiring

a straight line to be programmed by lots of joint-space poses close to each other.

Thus, the question is, are you programming robot joint servos to make the robot provide a service for you, are you programming/commanding an arm how to move a tool, is it the tool that is made programmable by means of the robot, is it manufacturing services that are ordered by specifying the desired process parameters, or is it an intelligent system that simply can produce your product?

As a final goal related to the product-centric view, so-called task-level programming is desirable. This has been a goal since the 1980s, and implicitly also since the very beginning of robotics. It would mean that the user simply tells the robot what should be done and the robot would know how to do it, but it would require extensive knowledge about the environment and so-called machine intelligence. The need for extensive modeling of the environment of the robot is well known. Sensing of the environment is costly, but in an industrial environment it should only be needed occasionally. Modeling has to encompass full component dynamics and the limitations of the manufacturing process. With these difficulties, task-level programming is not yet achievable in practice.

As indicated in the application examples, it is now common practice to generate robot programs from geometric data in CAD files. That is, the CAD application could be the environment used for specifying how the robot should perform the required operations on the specified parts. This is not quite task-level programming since human operators do the overall planning. CAD software packages are powerful 3-D tools and are now very common among manufacturing companies. Consequently, using those tools for robot programming is desirable since the operator may start the offline programming of the necessary manufacturing operations using the 3-D model of the product. There are two possible approaches:

1. Use the CAD interface to parameterize predefined robot programs, tailoring their behavior by using geometric information [42.16, 29]. This means defining motions, adding process parameters, deciding on what to do with data from devices, etc.
2. Generate the entire robot code from the information extracted from a CAD file, including the interface code to handle cell devices [42.30, 31].

The example presented in Fig. 42.14 was designed to run from the CAD package INVENTOR [42.32],



Fig. 42.14 Human-user interface for the CAD programming environment

from Autodesk. Basically it enables a user to program a workcell composed by a robot and a bending machine. The robot is used to feed and handle the sheets of metal to the bending machine, and to palletize the final product. The depicted pro-

gramming interface is an illustrative example of the possibilities, while there are a number of major vendors of CAD or offline programming system providing similar features in current and upcoming products.

42.5 End-Effectors and System Integration

It is interesting to note that connecting different workcell devices with each other, and integrating them into a working system, is hardly mentioned in the robotics literature. Nevertheless, in actual nontrivial installations, this part typically represents half of the cost of the installation. The automation scenario includes all the problems of integrating computers and their peripherals, plus additional issues that have to do with the variety of (electrically and mechanically incompatible) devices and their interaction with the physical environment (including its inaccuracies, tolerances, and unmodeled physical effects such as backlash and friction). The number of variations is enormous so it is often not possible to create reusable solutions. In total this results in a need for extensive engineering to put a robot to work. This engineering is what we call system integration.

Carrying out system integration according to current practice is not a scientific problem as such (although how to improve the situation is), but the obstacles it comprises

form a barrier to applying advanced sensor-based control for improved flexibility, as is needed in short-series production. In particular, in future types of applications using external sensing and high-performance feedback control within the workcell, system integration will be an even bigger problem since it includes tuning of the feedback too.

For long production runs, the engineering cost of system integration is less of a problem since its cost per manufactured part is small. On the other hand, the trend towards shorter series of customized products, or products with many variants that are not kept in stock, calls for high flexibility and short changeover times. Flexibility in this context refers to variable product variants, batch sizes, and process parameters. In particular this is a problem in small and medium-sized productions, but the trend is similar for larger enterprises as flexibility requirements are continuously on the rise. One may think that simply by using well standards for *in-*

Table 42.4 Stages of system integration, typically carried out in the order listed

Physical	Selecting equipment based on dimensioning for mechanical size, load, and stress
	Mechanical interfacing (locations, adapter plates, etc.)
	Electrical power supply (voltages and currents for robots, effectors, feeders, etc.)
	Connections for analog signals (shielding, scaling, currents, binary levels, etc.)
Communication	Interconnections for single-bit digital I/O
	Byte-wise data communication, including latencies and bit rates
	Transfer of byte sequences
Configuration	Configuration of messages between interacting devices
	Establishment of services
	Tuning for performance and resource utilization
Application	Definition of application-level functions/services
Task	Application programming, using the application-level services

put/output (I/O) and well-defined interfaces, integration should be just a matter of connecting things together and run the system. Let us examine why this is not the case.

Example 42.2 Step 1 of 3:

Simple pick-and-insert (assembly) operation. As a first step let us consider only one robot performing pick-and-place from/to known locations, and assume it is a well-known object such that we can use one specific type of gripper (such as a gripper, in this case based on a vacuum cup). We then only need one digital output from the robot controller. Let us call this output GRIPPER (i. e., the name of the number of the output) and the values GRASP and RELEASE for the high and low signals depending on the sign of the hardware connection. An example of a robot program can be found in the previous section.

Of course the object-oriented software solution would be to have a gripper class with operations grasp

and release. That would be appropriate for a robot simulator in pure software, but for integration of real systems such encapsulation of data (abstract data types) would introduce practical difficulties because:

- values are explicit and accomplished by external (in this case) hardware, and for testing and debugging we need to access and measure them,
- online operator interfaces permit direct manipulation of values, including reading of output values. The use of functions of object methods (so-called set:ers and get:ers) would only complicate the picture; maintaining consistency with the external devices (such as the definitions of the constants GRIPPER, GRASP, and RELEASE) would be no simpler.

Already in this toy example we can see that the user I/O configuration depends on both connected external devices (the gripper), and also on what I/O modules (typically units on some kind of field bus) that are in-

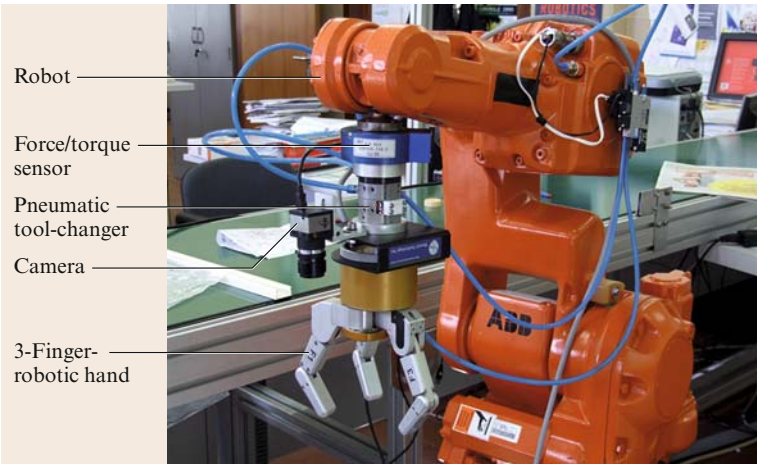


Fig. 42.15 Force controlled robot with multiple tools: force-torque sensor (JR3), robotic hand (Barrett), CCD USB camera (uEye) and tool-changer (ATI). This type of flexible gripper is not (yet) common in industry, where simple 1 DOF grippers with fingers tailored to the product are normally used

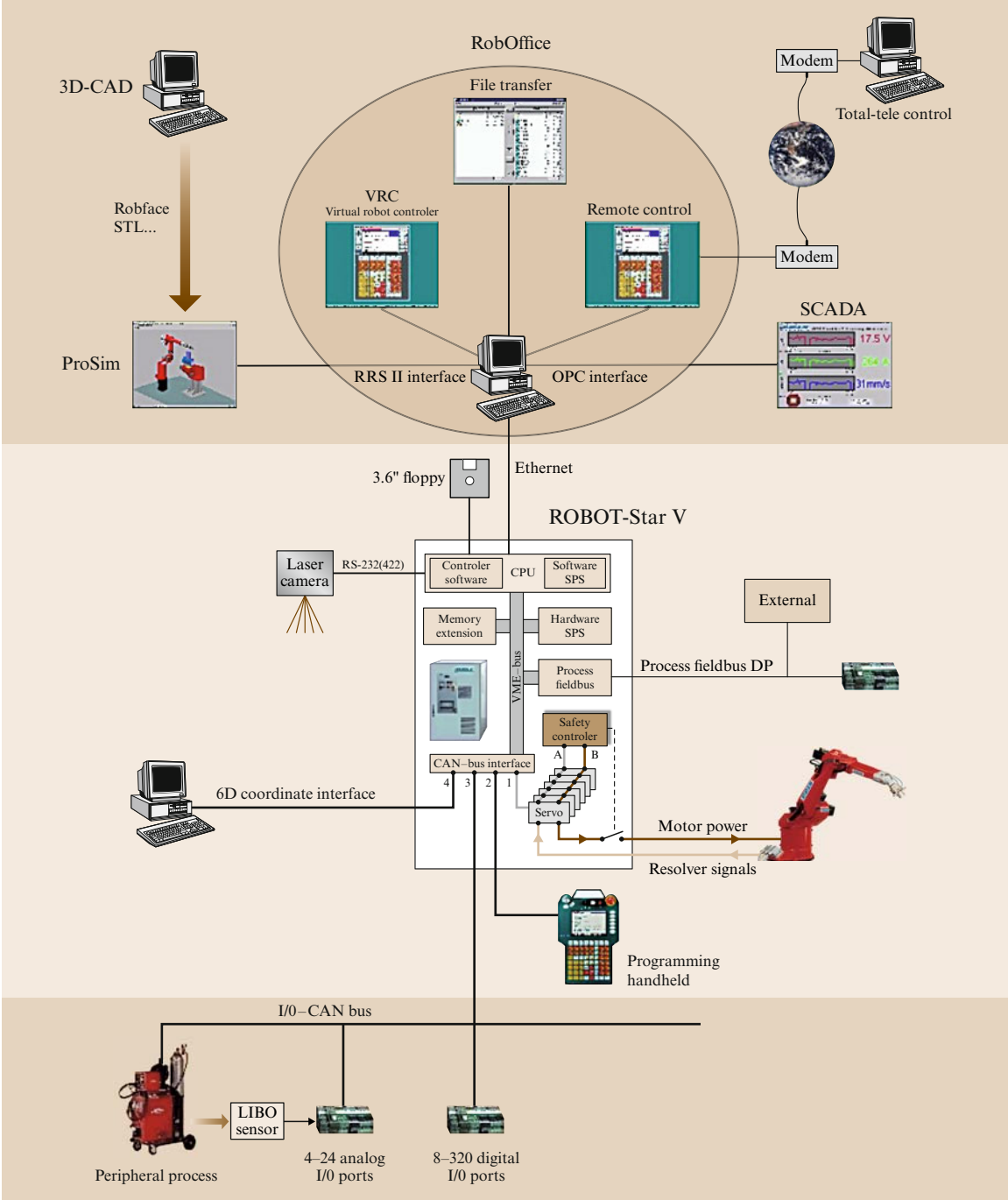


Fig. 42.16 Control modules and interfaces (Reis) on different levels, from low-level peripheral interfaces to high-level factory interfaces. The robot controller contains the middle part. The vertical integration in combination with heterogeneous hardware forms a major challenge (by courtesy Reis Robotics, Obernburg)

stalled in the robot controller (which may be subject to change when another task is given to the robot). Returning to our example, selecting modules and interfaces extends to the items of Table 42.4, where the application and task levels relate to robot programming and the research issue of how to simplify system integration, respectively.

Example 42.2 Step 2 of 3:

Simple pick-and-insert (assembly) operation. Assume we want to use a probe to detect the type and the location of the workpiece, that we need several different types of grippers depending on what object we should handle, and that the place operation needs to be a force-controlled insertion. Additionally, since we need different tools for different operations, we need a tool exchanger.

With these additional requirements we need more mechanical interfaces, the load/weight/performance considerations are more difficult (the weight of the tool exchanger and adaptor plate decreases the net payload for the grasped objects), and several more electrical interfaces need to be included.

Equipment examples are shown in Fig. 42.15, an example that extends the pick-and-insert case to include unknown positions and a gripper that can grab several types of pieces (including a wrist force sensor and force feedback in the robot fingers); details concerning robot programming and device configurations are omitted for brevity. In the presented example a three-finger robotic hand [42.33] was used to illustrate that these advanced and programmable tools are finding their way into industrial applications [42.34], for example, the hand used here has position and force-sensing capabilities. When grasping an object, the finger base and tip links move together. If a base link encounters an object it stops, while

the tip link keeps moving until it makes contact with the object as well. If the tip links encounter an object first, the whole finger assembly stops moving.

The example depicted in Fig. 42.15 shows a setup where the robot can pick the workpiece from an unknown position; the working pieces are identified using a CCD camera that returns the number of pieces and their position. The robot is then commanded to pick one and update its information from the cell [42.30, 35]. This example uses two tasks: one to receive remote commands, and the other to implement the gripper and camera services.

Example 42.2 Step 3 of 3:

Simple pick-and-insert (assembly) operation. As a final requirement, to monitor the quality of the assembly operation, assume that we want to have slip detection embedded into the gripper and that we need logging of the force control signal. The production statistics should be provided on the overall plant level of the factory.

These additional requirements call for integration of low-level devices with high-level factory control system, and is hence called *vertical integration*, whereas integration within (for instance) the workcell level is called *horizontal integration*.

Lack of self-descriptive and self-contained data descriptions that are also useful at the real-time level further increases the integration effort since data interfaces/conversions typically have to be manually written. In some cases, as illustrated in Fig. 42.16, there are powerful software tools available for the integration of the robot user level and the engineering level [42.36, 37]. The fully (on all levels) integrated and nonproprietary system, sometimes referred to as a *digital factory*, still is a challenge, particularly for small and medium-sized productions [42.38, 39].

42.6 Conclusions and Long-Term Challenges

The widespread use of robots in standard, large-scale production such as the automotive industry, where robots (even with impressive performance, quality, and semiautomatic programming) perform repetitive tasks in very well-known environments, for some time resulted in the common opinion that *industrial robotics is a solved problem*. However, these applications comprise only a minor part of the industrial work needed in any wealthy society, especially considering the number of companies

and the variety of applications. The use of robots in small and medium-sized manufacturing is still tiny.

Global prosperity and wealth requires resource-efficient and human-assistive robots. The challenges today are to recognize and overcome the barriers that are currently preventing robots from being more widely used.

Taking a closer look at the scientific and technological barriers, we find the following challenges:

- *Human-friendly task specification*, including intuitive ways of expressing permitted/normal/expected variations. That is, there are many upcoming and promising techniques for user-friendly human–robot interaction (such as speech, gestures, manual guidance, and so on), but the focus is still on specification of the nominal task (see Chaps. 58 and 59). The foreseen variations, and the unforeseen variations experienced during robot work, are more difficult to manage. When instructing a human he/she has an extensive and typically implicit knowledge about the work and the involved processes. To teach a robot, it is an issue both how to realize what the robot does not know, and how to convey the missing information efficiently.
- *Efficient mobile manipulation*. Successful implementations and systems are available for both mobility and for manipulation, but accomplished in different systems and using different types of (typically incompatible) platforms. A first step would be to accomplish mobile manipulation at all, including the *combination* of legged locomotion see Chap. 56 (for stairs and rough terrain), autonomous navigation (with adaptive but predictable understanding of constraints), dexterous manipulation, and robust force/torque interaction with environments (that have unknown stiffness). As a second step, all this needs to be done with decent performance using reasonably priced hardware, and with interfaces according to the previous item. Thus, we are far from useful mobile manipulation.
- *Low-cost components including low-cost actuation*. Actuation of high-performance robots represent about a third of the overall robot cost, and improved modularity often results in a higher total hardware cost (due to less opportunities for mechatronic optimization). On the other hand, cost-optimized (with respect to certain applications) systems result in more-specialized components and smaller volumes, with higher costs for short-series production of those components. Since future robotics and automation solutions might provide the needed cost efficiency for short-series customized components, we can interpret this as a boot-strapping problem, involving both technical and business aspects. The starting point is probably new core components that can fit into many types of systems and applications, calling for more mechatronics research and synergies with other products.
- *Composition of subsystems*. In most successful fields of engineering, the principle of superposition holds,

meaning that problems can be divided into subproblems and that the solutions can then be superimposed (added/combined) onto each other such that the total solution comprises a solution to the overall problem. These principles are of key importance in physics and mathematics, and within engineering some examples are solid-state mechanics, thermal dynamics, civil engineering, and electronics. However, there is no such thing for software, and therefore not for mechatronics (which includes software) or robotics (programmable mechatronics) either. Thus, composition of unencapsulated subsystems is costly in terms of engineering effort.

Even worse, the same applies to encapsulated software modules and subsystems. For efficiency, system interconnections should go directly to known (and hopefully standardized) interfaces, to avoid the indirections and extra load (weight, maintenance, etc.) of intermediate adapters (applying to both mechanics for end-effector mounting and to software). Interfaces can be agreed upon, but the development of new versions typically maintains backward compatibility (newer devices can be connected to old controller), while including the reuse of devices calls for mechanisms for forward compatibility (automatic upgrade based on meta information of new interfaces) to cover the case that a device is connected to a robot that is not equipped with all the legacy or vendor-specific code.

- *Embodiment of engineering and research results*. Use or deployment of new technical solutions today still starts from scratch, including analysis, understanding, implementation, testing, and so on. This is the same as for many other technical areas, but the exceptional wide variety of technologies involved with robotics and the need for flexibility and upgrading makes it especially important in this field. Embodiment into components is one approach, but knowledge can be applicable to engineering, deployment, and operation, so the representation and the principle of usage are two important issues. Improved methods are less useful if they are overly domain specific or if engineering is experienced to be significantly more complicated. Software is imperative, as well as platform and context dependent, while know-how is more declarative and symbolic. Thus, there is still a long way to go for efficient robotics engineering and reuse of know-how.
- *Open dependable systems*. Systems need to be open to permit extensions by third parties, since there is no way for system providers to foresee all upcoming

ing needs in a variety of new application areas. On the other hand, systems need to be closed such that the correctness of certain functions can be ensured. Extensive modularization in terms of hardware and supervisory software makes systems more expensive and less flexible (contrary to the needs of openness). Highly restrictive frameworks and means of programming will not be accepted for widespread use within short-time-to-market development. Most software modules do not come with formal specification, and there is less understanding of such needs. Thus, systems engineering is a key problem.

- *Sustainable manufacturing.* Manufacturing is about transformation of resources into products, and productivity (low cost and high performance) is a must.

For long-term sustainability, however, those resources in terms of materials and the like must be recycled. In most cases this can be achieved by crushing the product and sorting the materials, but in some cases disassembly and automatic sorting of specific parts are needed. There is therefore a need for robots in recycling and demanufacturing. Based on future solutions to the above items, this is then a robot application challenge.

An overall issue is how both industry and academia can combine their efforts such that sound business can be combined with scientific research so that future development overcomes the barriers that are formed by the above challenges.

References

- 42.1 The International Federation of Robotics (IFR): *World Robotics 2007. Statistics, Market Analysis, Forecasts, Case Studies and Profitability of Robot Investment* (IFR Statistical Department, Frankfurt 2007), <http://www.ifrstat.org>
- 42.2 M.P. Groover: *Automation, Production Systems, and Computer-Integrated Manufacturing*, 2nd edn. (Prentice Hall, Upper Saddle River 2000)
- 42.3 B.S. Dhillon, A.R.M. Fashandi, K.L. Liu: Robot systems reliability and safety: a review, *J. Qual. Mainten. Eng.* **8**(3), 170–212 (2002)
- 42.4 R.D. Schraft, S. Schmid, S. Thiernemann: Man-robot cooperation in a flexible assembly cell, *Assemb. Autom.* **22**(2), 136–138 (2002)
- 42.5 S.Y. Nof: *Handbook of Industrial Robotics* (Wiley, New York 1985)
- 42.6 V.D. Scheinman: Design of a Computer Controlled Manipulator. Ph.D. Thesis (Stanford University, Department of Computer Science 1969)
- 42.7 L. Westerlund: *The Extended Arm of Man. A History of the Industrial Robot* (Informations Förlaget, Stockholm 2000)
- 42.8 H. Makino: Assembly Robot, Patent 4341502 (1980)
- 42.9 G. Boothroyd, L. Altling: Design for assembly and disassembly, *Int. Inst. Prod. Eng. Res. Annal. (CIRP Annals)* **41**(2), 625–636 (1992)
- 42.10 G. Hirzinger, N. Sporer, A. Albu-Schaffer, M. Hahnle, R. Krenn, A. Pascucci, M. Schedl: DLR's torque-controlled light weight robot III – are we reaching the technological limits now?, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Vol. 2 (Washington 2002) pp. 170–176
- 42.11 R. Clavel: Device for the movement and positioning of an element in space, Patent 4976582 (1989)
- 42.12 R. Bloss: Innovation at IMTS, *Ind. Robot* **30**(2), 159–161 (2003)
- 42.13 Y. Kusuda: The international robot exhibition 2005 in Tokyo, *Ind. Robot* **33**(5), 342–348 (2006)
- 42.14 Y.R. Siegwart, I.R. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT Press, Cambridge 2004)
- 42.15 R.D. Schraft, M. Hägele, A. Breckweg: *Man and robot without separating systems*. Managers Navigator. World of Automation and Metalworking: 8th Edition for Europe (VDMA-Verlag, Frankfurt am Main 2006) pp.4–5
- 42.16 R. Bernhard, G. Schreck, C. Willnow: Development of virtual robot controllers and future trends, 6th IFAC Symp. Cost oriented Automation (Fraunhofer IPK, Berlin 2001)
- 42.17 K. Ikeuchi, B.K.P. Horn, S. Nagata: Picking up an object from a pile of objects, A.I. Memo 726, Artificial Intelligence Laboratory, Massachusetts Institute of Technology (1983)
- 42.18 J. Kirkegaard, T.B. Moeslund: Bin-picking based on harmonic shape contexts and graph-based matching, *Proc. 18th Int. Conf. Pattern Recogn. (ICPR'06)* (Hong Kong 2006) pp. 581–584
- 42.19 K. Modrich: 3D machine vision solution for bin picking applications, *Proc. Int. Robot. Vision Show (Rosemont 2007)*
- 42.20 T. Schaefer, R.D. Schraft: Incremental sheet metal forming by industrial robots, *Rapid Prototyp. J.* **11**(5), 278–286 (2005)
- 42.21 International Organization for Standardization (ISO): *Robots for Industrial Environments – Safety Requirements* (ISO, Geneva 2007), ISO 10218-1:2006/Cor 1:2007
- 42.22 International Organization for Standardization (ISO): *Safety of Machinery – Safety-Related Parts of Control Systems – Part 1: General Principles for Design* (ISO, Geneva 2006), ISO 13849-1:2006

- 42.23 International Organization for Standardization (ISO): *Safety of Machinery – Safety-Related Parts of Control Systems – Part 2: Validation* (ISO, Geneva 2003), ISO 13849-2:2003
- 42.24 A. Albu-Schäffer, C. Ott, G. Hirzinger: A unified passivity based control framework for position, torque and impedance control of flexible joint robots, *Int. J. Robot. Res.* **26**, 23–39 (2007)
- 42.25 A. Kochan: Robots and operators work hand in hand, *Ind. Robot* **33**(6), 422–424 (2006)
- 42.26 Three-dimensional control and monitoring – The first safe camera system SafetyEYE opens up new horizons for safety and security, Press release Pilz GmbH and Co. KG, Ostfildern, Germany, (2006) <http://www.pilz.de>
- 42.27 B. Siciliano, L. Villani: *Robot Force Control*, Ser. Eng. Comput. Sci. (Springer, Berlin, Heidelberg 2000)
- 42.28 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Prentice Hall, Upper Saddle River 2003)
- 42.29 U. Thomas, F.M. Wahl, J. Maass, J. Hesselbach: Towards a new concept of robot programming in high speed assembly applications, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS 2005)* (2005) pp. 3827–3833
- 42.30 J.N. Pires, A. Loureiro, G. Bolmsjö: *Welding Robots* (Springer, London 2006)
- 42.31 A. Blomdell, G. Böllmsjö, T. Brogardh, P. Cederberg, M. Isaksson, R. Johansson, M. Haage, K. Nilsson, M. Olsson, T.A. Robertsson, J. Wang: Extending an industrial robot controller: implementation and applications of a fast open sensor interface, *IEEE Robot. Autom. Mag.* **12**(3), 85–94 (2005)
- 42.32 Autodesk: *Inventor Application Programming Interface. Manual* (Autodesk Inc, 2007)
- 42.33 W. Townsend: The BarrettHand grasper – programmable flexible part handling and assembly, *Ind. Robot* **27**(3), 181–188 (2000)
- 42.34 A. Wolf, R. Steinmann, H. Schunk: *Grippers in Motion* (Springer, New York, 2005)
- 42.35 J.N. Pires: *Industrial Robot Programming, Building Applications for the Factories of the Future* (Springer, New York 2007)
- 42.36 R. Zurawski: *Integration Technologies for Industrial Automated Systems* (CRC, Boca Raton 2006)
- 42.37 M. Hobday, A. Davies, A. Prencipe: Systems integration: a core capability of the modern corporation, *Ind. Corp. Change* **14**(6), 1109–1143 (2005)
- 42.38 G. Wöhlke, E. Schiller: Digital planning validation in automotive industry, *Comput. Ind.* **56**(4), 393–405 (2005)
- 42.39 E. Westkämper: Strategic development of factories under the influence of emergent technologies, *CIRP Ann. Manuf. Technol.* **56**(1), 419–422 (2007)