14

Control of Production and Assembly Machines

In reprogrammable flexible manufacturing, it is envisaged that individual machines will carry out their assigned tasks with minimal operator intervention upon receipt of an appropriate high-level execution command. Such automatic device control normally means forcing a servomechanism employed by a production or assembly machine to achieve (or yield) a desired output parameter value in the continuous-time domain. In this chapter, our focus will be on the automatic control of two representative classes of production and assembly machines: material removal machine tools and industrial robotic manipulators. In Chap. 15, our attention will shift to the (higher-level) manufacturing system control that is based on discrete event system (DES) control theory, that is, the control of the flow of parts between machines.

14.1 NUMERICAL CONTROL OF MACHINE TOOLS

Material removal is achieved by the relative motion of a cutting tool with respect to a workpiece (Chaps. 8 and 9). In turning operations, the cutting tool can move in two orthogonal directions (feed and depth) and engage a rotating workpiece. The real-time control objective is to move the cutting tool along a prescribed path while controlling its position and velocity—the spindle rate is normally set to a fixed value. In three-axis milling operations,

the workpiece can move in three orthogonal directions and engage a rotating cutting tool. The real-time control objective is to move the workpiece (via the motion of the worktable) along a prescribed path while controlling its position and velocity—the (tool holder's) spindle rate is normally set to a fixed value. In drilling operations, the workpiece can move in two orthogonal directions, in a plane perpendicular to the one-axis motion of the cutting tool. The real-time control objective is to move the workpiece from one point to another and translate the tool vertically according to the specific hole depth requirement while the workpiece is kept stationary—the (tool holder's) spindle rate is normally set to a fixed value.

14.1.1 Development of Machine Tool Control

The term numerical control (NC), synonymous with machine tool control, can be traced back to the development of the pertinent control technology in 1952 at the Massachusetts Institute of Technology (MIT), U.S.A. The Servomechanism Laboratory at MIT was contracted at the time by the Parsons Corporation to develop a universal control technology for machine tools through a US Air Force contract. The preliminary outcome of this research was a retrofitted vertical (tracer) milling machine, whose three motion axes could be simultaneously controlled by a hybrid (digital/analog) controller. A punched tape, coded with the sequence of machining instructions, was utilized to program the controller of this first NC machine tool.

The first commercial NC machine controllers were developed by four separate companies based on US Air Force contracts—Bendix, EMI, General Dynamics, and General Electric. Some claim that this diversification attempt and promotion of competition is the lead cause of still having different formats for NC programs and thus a lack of portability of a NC program from one controller to another.

In 1960s, NC controllers relied on dedicated digital hardware for the execution of simple motion commands (straight line and circular arcs). These machine control units (MCUs) allowed programmers to download a sequence of operations to be executed by the dedicated hardware–based (versus software-based) motion generators (interpolators) and controllers. Many of these controllers are still in use today, in the form of original equipment (older NC machine tools) or as customized controllers retrofitted on originally manual machines.

The mid and late 1960s were marked by the development and widespread use of mainframe computers (especially those by IBM). At the time, several large manufacturers attempted to network their individual NC machines under the umbrella of one (or more) such mainframe computers. The purpose was centralized control, where one computer assigned tasks

and directly downloaded corresponding programs to the individual NC controllers. The term direct numerical control (DNC) was appropriately adopted for such configurations. The practice of DNC, however, was short lived owing to frequent down times of the main computer (not tolerable in manufacturing) and continued use of mass production strategies that did not require frequent changes in the programming of NC machines.

The term DNC has also referred in the past to attempts to control several machines using one centralized computer, where this controller downloaded step-by-step individual instructions to individual machines, as opposed to complete programs. Naturally, this practice had an even shorter life in manufacturing environments owing to frequent computer down times.

The term computerized numerical control (CNC) was introduced in the early 1970s with the development of minicomputer-based controllers for machine tool control. The early use of minicomputers was later replaced with the use of dedicated microprocessor-based NC controllers, as miniaturization rapidly allowed the packaging of CPU and memory devices with servo controllers into small controller units. Such controllers carry out motion planning and control functions in software, as opposed to via very restricted hardware circuits. The primary advantage of CNC machines, however, has been noted as their capability of allowing the adaptive control of machining operations. That is, CNC controllers can be appropriately programmed to vary the (input) process parameters, such as cutting speed and/or feed rate, in direct response to varying cutting conditions, such as tool wear and variable depth of cut that would cause undesirable increases in machining forces.

The factory of the future will be a networked environment, where production plans and control programs will be downloaded to appropriate CNC machines when needed (i.e., just-in-time control) (Fig. 1). Based on this

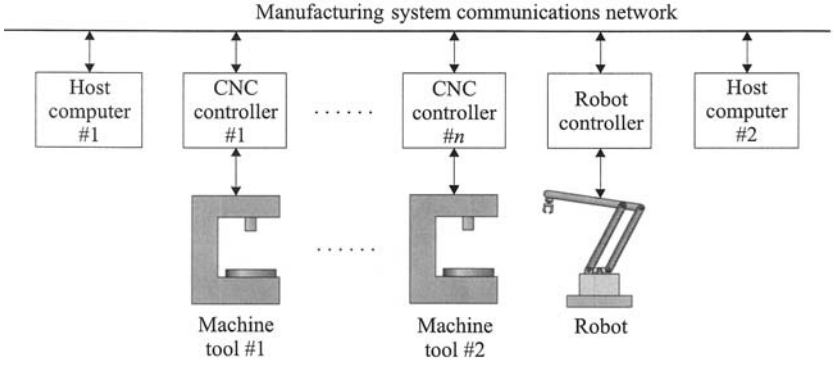

FIGURE 1 Distributed numerical control.

premise, the term distributed numerical control (DNC) has rapidly gained acceptance since the 1990s and was replaced the earlier acronym for direct numerical control. Although the current DNC architectures normally assume direct physical connection of CNC controllers to a centralized computer, in the near future there will be no such apparent connections. As shown in Fig. 1, all CNC controllers will have networking capabilities and receive commands and/or be downloaded programs over the communications network backbone of the factory.

14.1.2 Motion Control

Motion control in NC machines is achieved by issuing coordinated motion commands to the individual drives of the machine tool (Fig. 2). Almost all commercial NC machines employ DC or AC electrical motors that linearly drive stages/tables mounted on ball-bearing leadscrews. These leadscrews provide low-friction (no stick-slip), no-backlash motions with accuracies of 0.001 to 0.005 mm or even better. High-precision machines employ interferometry-based displacement sensors to provide sensory data to the (closed loop) controllers of the individual axes of the machine tool (Chap. 13). Rotational movements (spindle and other feed motions) are normally achieved using high-precision circular bearings (plain, ball, or roller).

Motion Types

Machine tools can be utilized to fabricate workpieces with prismatic and/or rotational geometries. Desired contours are normally achieved through a controlled relative motion of the cutting tool with respect to the workpiece. Holes of desired diameters, on the other hand, are normally achieved by

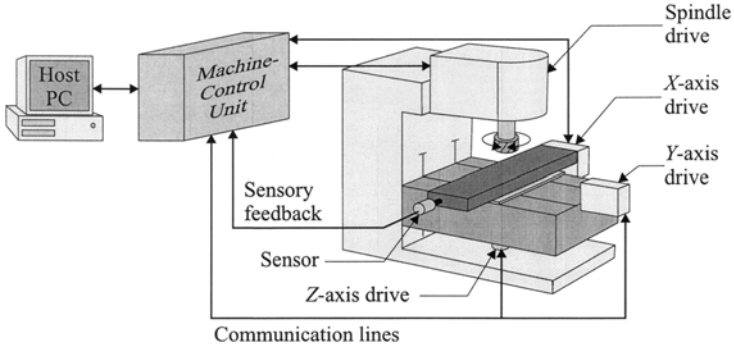

FIGURE 2 Overall NC machine tool control architecture.

holding the workpiece fixed and moving a rotating drill bit into the workpiece vertically. Correspondingly, NC motions have been classified as point-to-point (PTP) motion (e.g., drilling) and contouring, or continuous path (CP), motion (e.g., milling and turning).

In PTP systems, the workpiece is moved from one point to another in the fastest manner without regard to the path followed. The motion is of asynchronous type, where each axis accomplishes its desired movement independent of the others. For example, the $X-Y$ table of the drilling press would follow the path shown in Fig. 3a, where the Y axis continues its motion from Point A to the desired Point B, while the X axis remains stationary after it has already accomplished its necessary incremental motion. Once the table reaches Point B, the drill head is instructed to move in the Z axis, the necessary distance, and cut into the workpiece.

In CP systems, the workpiece (in milling) or the tool (in turning) follows a well-defined path, while the material removal (cutting) process is in progress. All motion axes are controlled individually and move synchronously to achieve the desired workpiece/tool motion (position and speed). For example, the $X-Y$ table of a milling machine would follow the path shown in Figure 3b, when continuously cutting into the workpiece along a two-dimensional path from Point A to Point B.

For both PTP and CP motions, the coordinates of points or paths can be defined with respect to a global (world) coordinate frame or with respect to the last location of the workpiece/tool: absolute versus incremental positioning, respectively. Regardless of the positioning system chosen, the primary problem in contouring is the resolution of the desired path into multiple individual motions of the machine axes, whose combination would yield a cutter motion that is closest possible to the desired path. This motion-planning phase is often called interpolation. In earlier NC machine controllers, interpolation was carried out exclusively in dedicated hardware

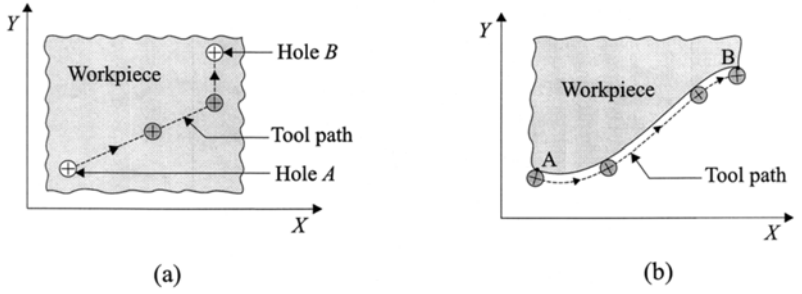

FIGURE 3 (a) Point-to-point; (b) continuous path motion.

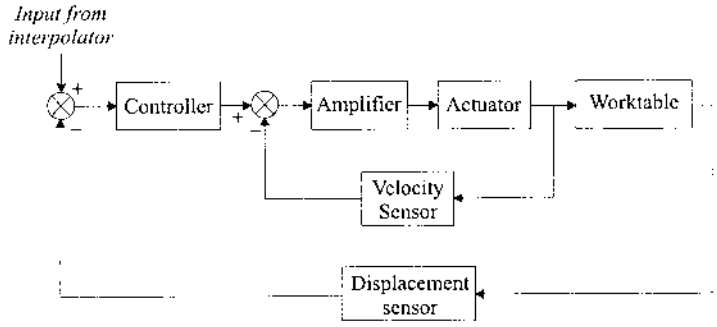

FIGURE 4 Closed-loop NC machine tool CP motion control.

boards, thus limiting the contouring capability of the machine tool to mostly straight-line and circular-path motions. In modern CNC machines, interpolation is carried out in software, thus allowing any desired curvature to be approximated by polynomial or spline-fit equations.

Closed-Loop Control

In PTP motion, individual axes are provided with incremental motion commands executed with no regard to the path followed. Although control can be carried out in an open-loop manner, encoders mounted on the leadscrews allow for closed-loop control of the motion (Chap. 13).

In CP motion, the interpolator provides individual axes with necessary motion commands in order to achieve the desired tool path (Fig. 4). Encoders and tachometers provide the necessary feedback information; interferometry type sensors can be used for high-precision displacement and velocity applications (Chap. 13).

Adaptive Control

Adaptive control of machine tools refers to the automatic adaptation of cutting parameters in response to changes in machining conditions (Fig. 5). A collection of sensors (acoustic, thermal, dynamic, etc.) are utilized to monitor cutting forces/torques, cutting temperatures, mechanical vibrations, acoustic emissions, in order to predict tool wear, the potential for tool breakage, chatter, and so on. A software-based adaptive controller utilizes the collected information in order to change feed rate and cutting velocity in real time and provide this information to the interpolator of the CNC controller for the generation of new motion commands (Fig. 4). Prediction techniques, such as neural networks, fuzzy logic, and heuristic rules, can be used in the calculation of new cutting parameters.

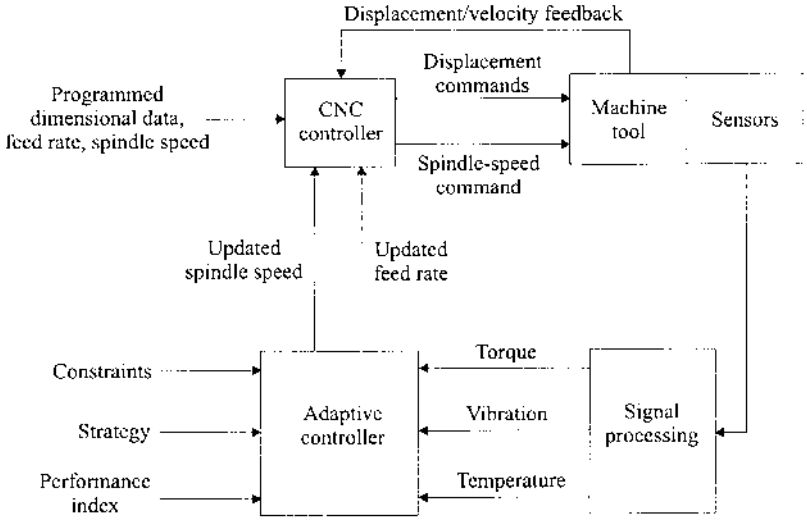

FIGURE 5 Adaptive control for machining.

From a commercial point of view, the primary objective of an adaptive control system should be to optimize a performance index, such as machining time or cost, subject to the capability limits of the machine tool and the dimensional constraints imposed on the workpiece. It would, for example, be desirable to adjust automatically the cutting parameters in real time for maximizing material removal rates.

Adaptive controllers capable of real-time optimization are still in their research phase owing to the high complexity of the machining process. Constraint-based adaptive controllers, however, are considered to be mature enough for commercialization. Such controllers adjust cutting parameters in real time in order to maintain cutting forces/torques, vibrations, temperature, and so on at or below their user-specified limits. For example, a machine tool's feed rate would be reduced in response to cutting-force increases due to tool wear, unexpected variations in workpiece hardness, and raw-material (stock) geometry, and so on.

Adaptive control is discussed below for two metal-cutting applications:

Adaptive control in turning: The cutting tool in NC lathes is mounted onto a stage whose motion is controlled in two orthogonal axes, the feed and depth-of-cut directions. Tool wear in turning is normally a continuous process leading to tool degradation in the form of flank wear and crater wear (Chap. 8). Although flank wear yields a continuous increase in cutting forces, initial crater wear can create favorable cutting conditions and lead to

reduction in cutting forces. Beyond a crater-wear threshold, both wear mechanisms lead to gradual increases in cutting forces.

There exist a variety of force sensors, commercialized since the early 1970s (e.g., Kistler, Prometec, Montronix, and Sandvik), that can be easily mounted on the stage of the lathe, underneath the tool holder. Such instruments utilize piezoelectric or strain gages as the force detection transducers (Chap. 13). Cutting forces can also be evaluated by monitoring torque requirements on the drivers of the cutting tool stage and/or on the spindle motor. Such measurements, however, are only used as complementary information and not as sole indicators of force owing to difficulties in mathematical predictions of force directions and magnitudes.

Acoustic emissions from the cutting zone (low amplitude and high frequency) have also been sensed via piezoelectric detectors (microphones) for estimating tool wear. Continuous signals are generated in the shear zone and at the workpiece–tool and chip–tool interfaces, while discontinuous signals are generated by the breakage of the chips. The frequencies of these signals are much higher than other potential emissions in the surroundings, such as machine tool vibrations. A number of classical statistical pattern-recognition schema to have been developed by academic researchers during the 1980s and 1990s for identifying tool wear via acoustic emissions. However, in practice, acoustic sensors have only been used as early warning systems to indicate imminent failure of the cutting tool and not for continuous feedback to the adaptive controllers.

Adaptive control in milling: The cutting operation in milling is an intermittent process, where a cutting edge engages the workpiece periodically and remains engaged for a portion of the full rotation of the multitooth tool. Thus, besides the gradual tool wear, one must monitor for force and torque overloads, chatter-causing vibrations, and catastrophic tool failure. Force overloads at the engagement of the tool with the workpiece (especially in the case of small-diameter tools) can severely damage the tool and subsequently the workpiece.

As in turning, force sensors placed underneath the workpiece fixtures and torque sensors mounted on the spindle of milling machines can be effectively utilized to detect spindle stalls, cutting-force overloads, and tool wear/breakage. Acoustic sensors have also been used in milling to detect chatter—a self-excited vibration mechanism due to the regeneration of periodical waviness on the machined workpiece—by listening to emissions of increasing amplitude (Chap. 8).

As discussed above, many different sensors can be used to monitor the working condition of a machine tool for its adaptive control. For example, tool wear can be monitored using force sensors mounted under the tool

holder (in turning) or under the workpiece (in milling), torque sensors mounted on the spindle motor, and acoustic sensors placed in close vicinity to the cutting interface. Naturally, each sensor outputs its conclusion based on its received and analyzed signals with an associated uncertainty. This uncertainty consists of components such as (Gaussian) random noise, (systematic) fixed errors due to inaccurate calibration, and limitations of the pattern analysis technique used in manipulating the collected data. The use of multiple sensors (multisensor integration) and the merging of their outputs (data fusion) can benefit the monitoring process by reducing the uncertainty level.

Multisensor integration is the choice of the number and the types of sensors for the task at hand and their optimal placement in the workspace for maximum accuracy. Two possible strategies for multisensor integration are (1) to select and configure a minimum number of sensors and utilize them continuously (for the entire duration of the process monitored), or (2) to select a large number of sensors (more than the minimum) and configure them in real time (i.e., select subsets of sensors) according to a criterion to best suit the needs of the monitoring objective as machining progresses. For the latter strategy, for example, we can use only force transducers at the beginning of cutting but activate and merge additional data received from acoustic sensors toward the end of the expected/predicted tool life.

Multiple sensors can provide a data fusion module with two types of information: (1) data about one feature observed by multiple sensors—redundant information, or (2) data about the subfeatures of one feature, in cases where no one single sensor can perceive the totality of the feature level—complementary information. The data collected can in turn be fused at multiple levels: signal level or feature level. Signal level data fusion is common for sensing configurations, multiple identical (redundant) sensors observing the same feature. A common problem at this level of fusion is the temporal and spatial alignment of data collected from multiple sensors (i.e., ensuring that all sensors observe the same feature at the same time—synchronization). At feature-level fusion, the primary problem is the spatial transformation of information for spatial alignment.

Common methods for signal-level data fusion include weighted averaging of measurements, recursive estimation of current and future measurements using the Kalman filter, hierarchical estimation using a Bayesian estimator for combination of multisensor data according to probability theory, Dempster–Shafer reasoning approach for combining only evident data (i.e., not assigning probabilities to unavailable data), fuzzy-logic reasoning via the assignment of discrete values (between 0 to 1) to different propositions—a multivalued-logic approach, and so on.

14.1.3 Programming of NC Machine Tools

The programming of a NC machine tool is preceded by the determination of a suitable (preferably optimal) process plan. A process plan specifies how a part is to be machined: the sequence of individual operations, the specific machine tools on which these operations are to be carried out, the machining parameters (e.g., feed rate, cutting velocity) for each operation, and so forth.

All NC machine tools are equipped with controllers that can interpret a machine language–based program and convert these instructions into motion commands of the numerically controlled axes. These machine language programs have been commonly referred to as *g*-code. Unfortunately, for historical reasons, different commercial NC controllers use similar but different *g*-codes.

During the period 1955 to 1958, the first high-level programming language for NC machine tools was developed under the coordination of researchers from MIT. This programming language (APT, automatically programmed tool) reached maturity in the early 1960s and served as a guideline for the development of many subsequent NC programming languages, such as EXAPT (extended subset of APT) developed by the Institute of Technology in Aachen, Germany, ADAPT (adaptation of APT) and AUTOSPOT (automatic system for positioning tools), both by IBM, U.S.A., among many others. A program written in one such high-level languages needs to be translated into the specific *g*-code of the NC machine tool to be utilized for the machining of the workpiece at hand.

Since the late 1980s, most commercial CAD software packages allow users to generate cutting tool paths automatically in an interactive manner, bypassing the generation of a high-level language program. The user can simulate the machining operation and, having been satisfied with the outcome, can request the CAD system to generate the corresponding *g*-code program (specific to the NC controller to be utilized) and directly download it to the NC machine tool over the communications network.

g-Code

A *g*-code program consists of a collection of statements/blocks to be executed in a sequential manner. Each statement comprises a number of "words"—a letter followed by an integer number. The first word in a statement is the block number designated by the letter N followed by the number of the block (e.g., N0027, for the 27th line in the *g*-code program). The next word is typically the preparatory function designated by the letter G (hence, the letter "*g*" in *g*-code) followed by a two-digit number. Several examples of G words are given in Table 1.

TABLE 1 Some *G* Words[a]

Code	Function	Code	Function
G00	Point-to-point motion	G20	Imperial units
G01	Linear-interpolation motion	G21	Metric units
G02	Clockwise circular-interpolation motion	G32	Thread cutting
		G98	Per-minute feed rate
G03	Counterclockwise circular-interpolation motion	G99	Per-revolution feed rate

[a] May be different for different NC controllers.

The preparatory function is followed by dimensional words designated by axes' letters X, Y, and Z with corresponding dimensions, normally expressed as multiples of smallest possible incremental displacements (e.g., X3712 Y-47000 Z12000; multiples of 0.01 mm) or in absolute coordinates (e.g., X175.25 Y325.00 Z136.50). The feed rate and spindle speed words are designated by the letters F and S, respectively, followed by the corresponding numerical values in the chosen units. Next come the tool number word designated by the letter T and the miscellaneous function word designated by the letter M (Table 2).

A typical *g*-code program block is

N0027 G90 G01 X175.25 Y325.00 Z136.50 F125 S800 T1712 M03 M08;

the statement Number 27 (N0027) specifies the use of absolute coordinates (G90), a linear interpolation motion (G01) from current location to a position defined by the X, Y, Z coordinates (X175.25 Y325.00 Z136.50), a feed rate of 125 mm/min (F125) along the path, a spindle speed of 800 rev/min (S800), tool number 1712 (T1712), a clockwise turn of the spindle (M03), and coolant on (M08).

TABLE 2 Some M Words[a]

Code	Function	Code	Function
M00	Program stop (during run)	M08	Coolant on
M02	End of program	M11	Tool change
M03	Spindle start clockwise	M98	Call a subprogram
M05	Spindle stop	M99	Return to main program

[a] May be different for different NC controllers.

APT Language

The APT language serves two purposes: it (1) provides the NC controller with the pertinent geometric description of the workpiece, and (2) instructs it to carry out a series of operations for the machining of this workpiece. It achieves these objectives by utilizing about 600 geometric and motion command words.

The typical APT statement comprises two segments separated by a slash. The APT word to the left of the slash is modified by the information provided on the right side of the slash.

Geometric statements: The geometry of the workpiece pertinent to its machining can be described by a collection of points, lines, and surfaces. A few exemplary ways of describing such entities are given here:

Point definition by its coordinates:

Point_Name = POINT/X, Y, Z coordinates

P7 = POINT/200, 315, 793

Point definition by the intersection of two lines:

Point_Name = POINT/INTOF, Line_Name_1, Line_Name_2

P11 = POINT/INTOF, L3, L7

Line definition by two points:

Line_Name = LINE/Point_Name_1, Point_Name_2

L3 = LINE/P9, P21

Line definition by a point and an angle with respect to an axis:

Line_Name = LINE/Point_Name, ATANGL, Angle_Value, Axis_Name

L7 = LINE/P8, ATANGL, -75, YAXIS

Defining a circle by its center and radius:

Circle_Name = CIRCLE/CENTER, Point_Name, RADIUS, Radius_Dimension

C3 = CIRCLE/CENTER, P14, RADIUS, 35

Defining a plane by its equation $ax + by + cz = d$:

Plane_Name = PLANE/a, b, c, d

PL1 = PLANE/7.5, -3.1, 0.3, 6.7

Defining a (circular) cylindrical surface by a tangent plane, along a given line, with a given radius:

Surface_Name = CYLNDR/Side_of_Plane, TANTO, Plane_Name, THRU, Line_Name, RADIUS, Radius_Dimension

CYL3 = CYLNDR/ZLARGE,TANTO, PL1, THRU, L7, RADIUS, 25

Motion statements: The relative movement of the tool with respect to the workpiece can be of PTP or CP (contouring) type. A few exemplary ways of describing motion are given here:

PTP motion commands:

GOTO/Point_Name; Go to Point Point_Name.

GODLTA/ΔX, ΔY, ΔZ; Move incrementally by (ΔX, ΔY, ΔZ).

CP motion commands: In APT programming, motion commands are based on the relative movement of the cutting tool with respect to a stationary workpiece. The tool's motion is restricted by three surfaces: The depth (part) surface, on which the tool-end moves, the tangent (drive) surface, along which the tool slides, and the constraint (check) surface, which defines the end of the motion (Fig. 6). Thus the contouring motion commands on a given part surface are defined by the drive-surface and check-surface planes:

$$\begin{bmatrix} \text{GOFWD} \\ \text{GOBACK} \\ \text{GOLFT} \\ \text{GORGT} \\ \text{GOUP} \\ \text{GODOWN} \end{bmatrix} / \text{ Drive\_Surface, } \begin{bmatrix} \text{TO} \\ \text{ON} \\ \text{PAST} \\ \text{TANTO} \end{bmatrix}, \text{ Check\_Surface}$$

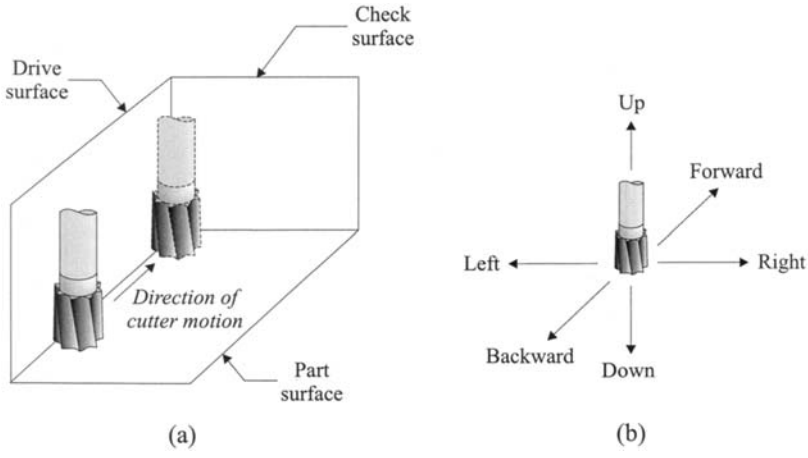

FIGURE 6 (a) Control surfaces; (b) control directions for contouring.

In three-dimensional machining, the contouring motion starts by relocating the tool from its current location to a point defined by the three surfaces constraining the motion of the tool:

FROM/Point_Name

$$\text{GO/}\begin{bmatrix}\text{TO}\\ \text{ON}\\ \text{PAST}\end{bmatrix}\text{, Drive\_Surface, }\begin{bmatrix}\text{TO}\\ \text{ON}\\ \text{PAST}\end{bmatrix}\text{, Part\_Surface, }\begin{bmatrix}\text{TO}\\ \text{ON}\\ \text{PAST}\end{bmatrix}\text{, Check\_Surface}$$

The following example program defines the two-dimensional contouring of the part profile shown in Fig. 7:

```
FROM/P1
GO/TO, L1, ON, PSURF, ON, L2
GORGT/L1, TANTO, C1
GOFRWD/C1, TANTO, L2
GOLFT/L2, PAST, L1
GOTO/P1
```

Other APT Statements:

```
MACHIN/Postprocessor_Name     ; Machine-specific
                                postprocessor
UNITS/MM or UNITS/INCHES      ; Units
FEDRAT/Value_per_minute or
          Value_per_revolution ; Feedrate
SPINDL/Speed, CLW             ; Spindle turns at "Speed"
                                clockwise
COOLNT/ON                     ; Coolant on
```

Once an APT program is obtained, a processor is needed to generate the cutter location data (CLDATA) file. The CLDATA file is then utilized

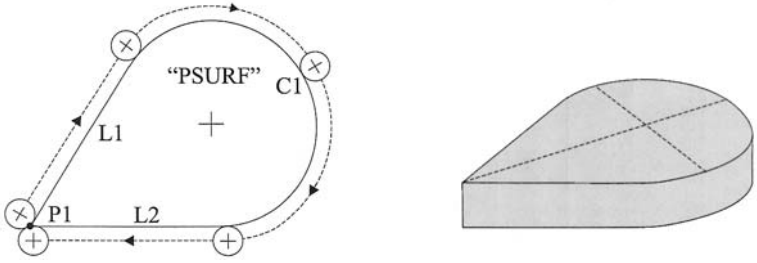

FIGURE 7 Contouring example.

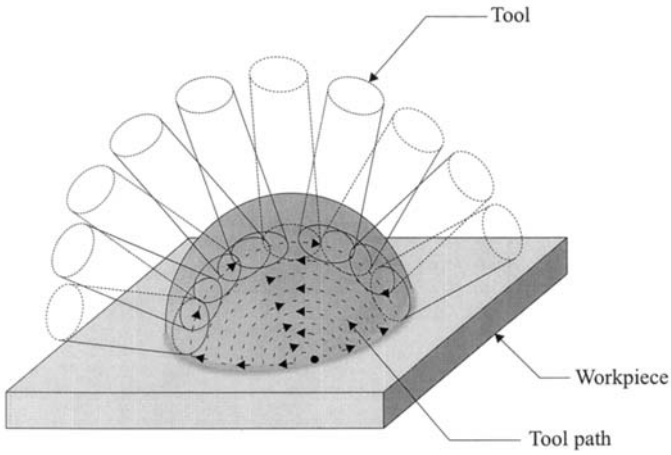

FIGURE 8 Tool path simulation.

as an input to a postprocessor specific for the NC machine tool at hand for the generation of the *g*-code. The first line in the APT program specifies this specific postprocessor (MACHIN/ Post_Processor_Name).

Computer-Aided NC Machine Programming

In the integrated and networked factory of the future, engineers will likely bypass the manual generation of NC part programs and exclusively adopt the rapidly developing computer-aided tools for the automatic generation of *g*-codes. Today an engineer can create the solid model of a stock, define cutting tool paths on this stock by specifying control surfaces corresponding to desired optimal cutting parameters, and prompt the software to generate corresponding a CLDATA file and subsequently to postprocess this file according to the NC controller at hand. This automatic process is then finalized by the downloading of the *g*-code to the controller of the CNC machine over the computer network.

Figure 8 shows a simulation example for a computer-aided generated tool path.

14.2 CONTROL OF ROBOTIC MANIPULATORS

As discussed in Chap. 12, robotic manipulators have been utilized in the manufacturing industry in a variety of applications, ranging from spot welding to spray painting, to electronic component assembly, and so on. The vast majority of these manipulators are open-chain mechanisms, comprising

a set of links attached via revolute (rotary) or prismatic (linear) actuators in series. A number of closed-chain mechanism manipulators, comprising a set of links/actuators configured in parallel versus in series, have also been utilized in the manufacturing industry for high-precision tasks, but our focus herein will be on serial manipulators (Fig. 9).

Serial robotic manipulators have been configured in three distinct geometrical forms (Sec. 12.3.1, Figs. 12.10 to 12.13): rectangular, cylindrical, and spherical. This classification is based on the geometry of the workspace of the manipulator. For example, an articulated robot comprising a sequence of rotary joints can be classified as a spherical-geometry robot since its workspace is spherical in nature. Regardless of their geometric classification, industrial robotic manipulators carry out tasks that require their end-effector (gripper or specialized tool) to move in point-to-point (PTP) or continuous path (CP) mode. Thus, as NC controllers for machine tools, robot motion controllers must ensure specific trajectory following in real time, as defined by the trajectory planning module of the controller.

Unlike in NC motion interpolation for machining, with the exception of five-axis machining, trajectory planning for industrial robots is a complex matter owing to the dynamics of open-chain manipulators moving payloads in three-dimensional Cartesian space subject to gravitational, centrifugal, and inertial forces. Thus in this section, robot motion planning and control will be addressed in the following order: kinematics/dynamics, trajectory

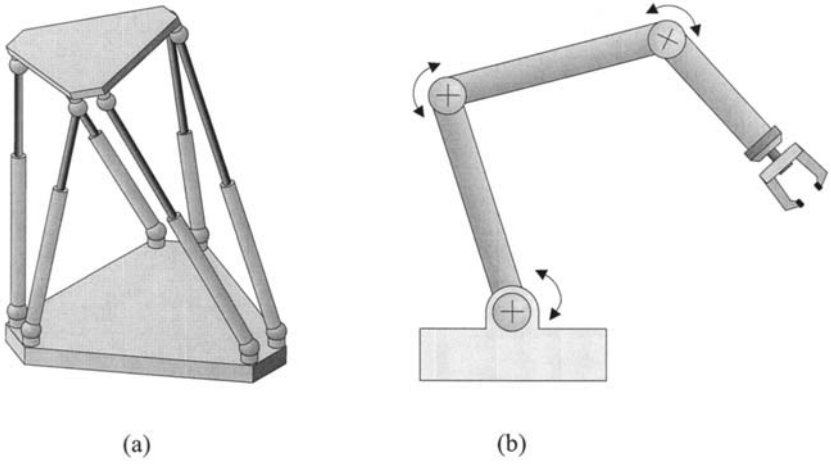

FIGURE 9 (a) A parallel; (b) a serial manipulator.

planning, and control, (Section 14.2.1). Robot programming techniques will be discussed in the subsequent subsection.

14.2.1 Motion Planning and Control

The first challenge in robot motion control is the transformation of a desired task space motion command into corresponding joint space (actuator) motion commands for the individual joints of the manipulator. For example, given the manipulator's latest stand (configuration) and a desired incremental end-effector translational motion of (ΔX, ΔY, ΔZ), while maintaining a constant orientation, the task at hand is to determine corresponding joint motions, $\Delta\theta_i$, $i = 1, n$, where n is the degrees of freedom (dof) of the robot.

A transformation of positional/velocity/acceleration information between task space (normally, Cartesian) and joint space coordinates can only be achieved via the kinematic model of the manipulator. A dynamic model of the manipulator, however, is needed for calculating available joint torques/forces in response to load carrying task space requirements, especially when one attempts to minimize the required effort or motion time along a given end-effector path—trajectory planning.

The majority of industrial robots employ closed loop controllers designed to drive the individual actuators of the manipulator, when executing a desired task space trajectory converted into individual joint commands by a trajectory planning module (Fig. 10).

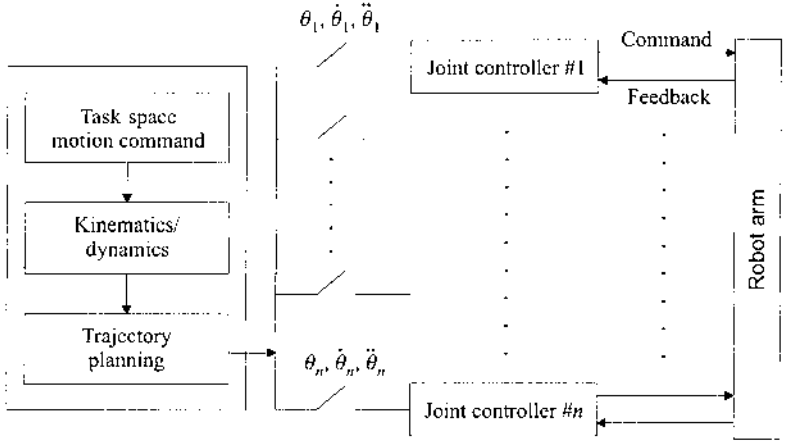

FIGURE 10 Robot motion control.

The objective of a kinematic model is to relate the motion of the robot end-effector in task space coordinates to joint space coordinates: the individual motions of the joints. This is a typical rigid body motion description in three-dimensional space.

Homogeneous transformations (4×4 matrices) have been often used to describe the motion of a rigid body, defined by a Cartesian frame, with respect to a fixed coordinate system. The following four matrices describe a translation of (d_x, d_y, d_z) and a rotation of θ with respect to the X, Y, and Z axes, respectively:

$$\text{Trans } (d_x, d_y, d_z) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14.1a}$$

$$\text{Rot } (x, \theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14.1b}$$

$$\text{Rot } (y, \theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14.1c}$$

$$\text{Rot } (z, \theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14.1d}$$

Any rigid body motion can be described by the above matrices multiplied in the sequence of their application. For example, a translation of the object frame from F_1 to F_2 (d_x, d_y, d_z), followed by an arbitrary rotation, θ, about the new X_2 axis, would yield a new frame location, F_3, which could be followed by an arbitrary rotation, ψ, about the new Z_3 axis to yield F_4. The last object location defined by F_4 with respect to its initial location F_1 would then be defined by

$$^1T_4 = \text{Trans}(d_x, d_y, d_z)\ \text{Rot}(X_2, \theta)\ \text{Rot}(Z_3, \psi)$$

In the case of open chain (serial) manipulators, a (rigid body) frame attached to the end of a link is moved in space by a joint located at the start

of the link. We must sequentially combine the individual motions of every moving link, from the end-effector all the way to the base, in order to obtain the overall kinematic model of the robot. A commonly used notation for this purpose was developed by J. Denavit and R. S. Hartenberg in the early 1950s—now called the D–H transformation.

According to D–H notation, a rotary joint causes the following transformation to a frame attached to the end of the link it is driving (Fig. 11):

$$A_i = \begin{bmatrix} \cos\theta & -\sin\theta\cos\alpha & \sin\theta\sin\alpha & a\cos\theta \\ \sin\theta & \cos\theta\cos\alpha & -\cos\theta\sin\alpha & a\sin\theta \\ 0 & \sin\alpha & \cos\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14.2}$$

where θ is the variable rotation of the ith joint about its Z-axis and (α, a, d) are constant offsets. Similarly, the following transformation matrix describes

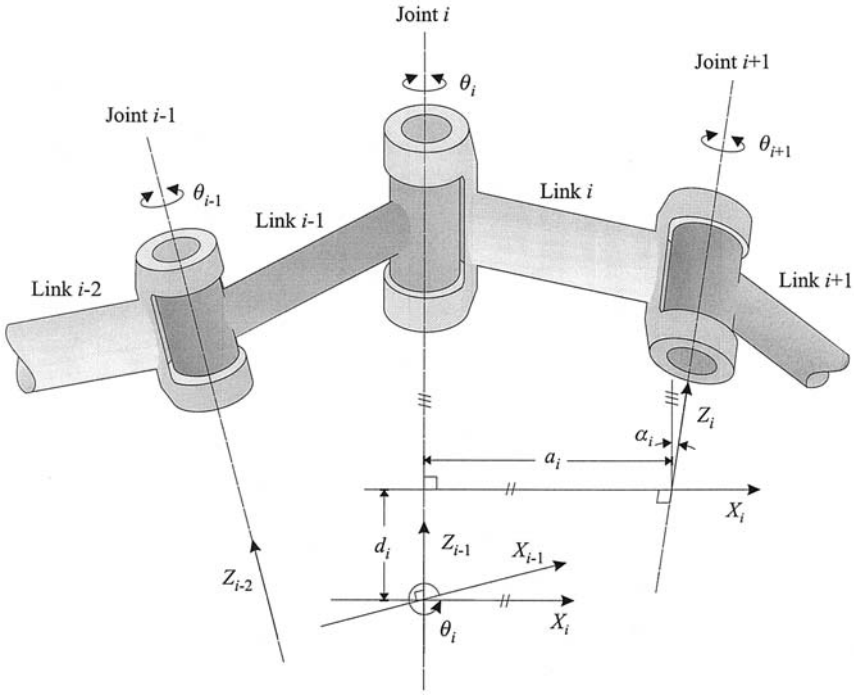

FIGURE 11 D–H notation for a rotary joint.

the displacement of a frame attached to the end of the link driven by a prismatic (linear) joint (Fig. 12):

$$A_i = \begin{bmatrix} \cos\theta & -\sin\theta\cos\alpha & \sin\theta\sin\alpha & 0 \\ \sin\theta & \cos\theta\cos\alpha & -\cos\theta\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14.3}$$

where d is the variable displacement of the ith joint along its Z axis, and θ, α are constant offsets.

For an n dof manipulator, the transformation matrix relating the motion of the end-effector, defined by the frame, F_e, with respect to a fixed

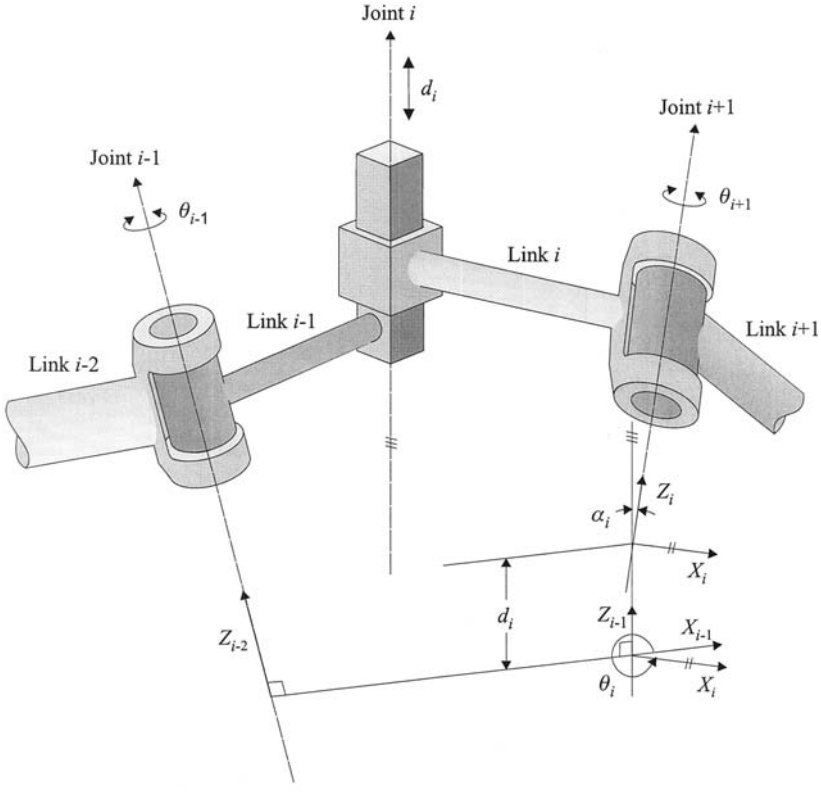

FIGURE 12 D–H notation for a prismatic joint.

frame at the base of the robot, F_0, is obtained by multiplying sequentially all the pertinent A matrices,

$$^0T_e = A_1A_2 \cdots A_{n-1}A_n = f(\theta_i, \alpha_i, a_i, d_i) \tag{14.4}$$

where ^0T_e is a function of the joint variables and constant offsets, $i = 1, n$ (Fig. 13).

The term direct kinematics is used for serial manipulators to describe the process of obtaining the location of the end-effector, F_e, with respect to the robot's fixed base frame, F_0, by solving the Eq. (14.4) for a set of joint variable values. This answers the question, where is the end-effector, having moved the joints of the robot by arbitrary amounts? Correspondingly, the term inverse kinematics is used (for serial manipulators) to describe the process of obtaining the individual joint displacement values for a specific location of the end-effector: this process would require the inversion of Eq. (14.4). Inverse kinematics is a nontrivial process owing to the presence of inverse trigonometric terms in the kinematic models of the majority of industrial robots that employ rotary joints.

In the context of kinematics, PTP motion planning requires us to find the robot configuration, defined by the variable joint displacement values, corresponding to the location to which we want to move the end-effector, and planning appropriate joint trajectories from the current robot configuration to that point (Fig. 14a). CP motion planning, on the other hand, requires us to define a given continuous path in terms of a sufficient number of representative points, carry out inverse kinematics, just as in PTP motion planning, to determine the corresponding robot configurations, and plan

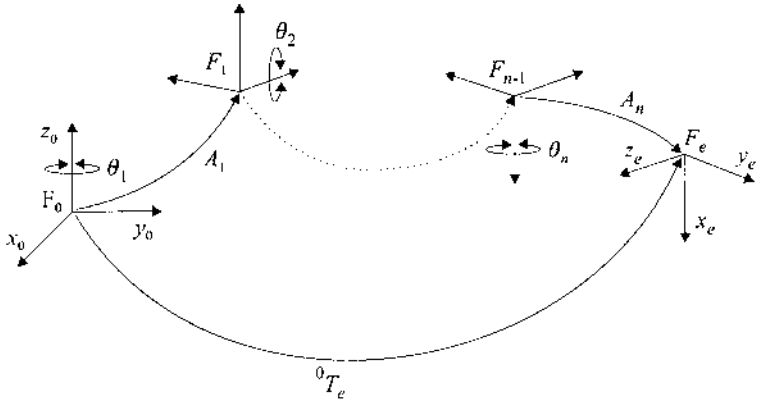

FIGURE 13 An n-dof robot transformation.

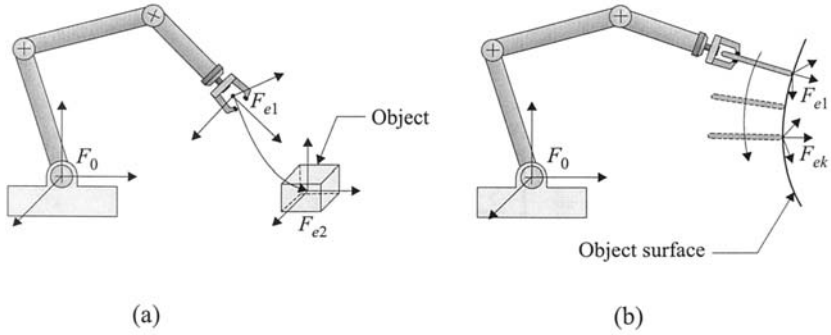

FIGURE 14 (a) PTP motion to grasp object; (b) CP motion to arc weld.

individual synchronized joint trajectories to obtain the desired (smooth) CP motion (Fig. 14b).

The kinematic model of a serial manipulator can be differentiated to yield a relationship between the joint and the Cartesian end-effector velocities. This relationship can be expressed in a matrix form as

$$\mathbf{V} = \mathbf{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \tag{14.5}$$

where $\mathbf{V}$ is the Cartesian end-effector velocity vector comprising both linear and angular velocity components, $\dot{\boldsymbol{\theta}}$ is the (generic) joint velocity vector, and $\mathbf{J}(\boldsymbol{\theta})$ is the Jacobian matrix expressed as a function of the instantaneous robot configuration (i.e., the joint displacement values, $\boldsymbol{\theta}$).

The (serial) robot's Jacobian matrix can also be utilized to express the relationship between joint torques/forces and the static Cartesian end-effector forces/moments:

$$\boldsymbol{\tau} = \mathbf{J}^{\mathbf{T}}\mathbf{F} \tag{14.6}$$

where $\boldsymbol{\tau}$ is the generic joint torque vector, $\mathbf{F}$ is the generic end-effector static force vector, and $\mathbf{J}^{\mathrm{T}}$ is the transpose of the robot's Jacobian matrix.

As an example, the Jacobian matrix of a SCARA robot shown in Fig. 15 is

$$\mathrm{J} = \begin{bmatrix} 0 & a_1\cos\theta_1 & a_1\cos\theta_1 + a_2\cos(\theta_1+\theta_2) & 0 \\ 0 & a_1\sin\theta_1 & a_1\sin\theta_1 + a_2\sin(\theta_1+\theta_2) & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix} \tag{14.7}$$

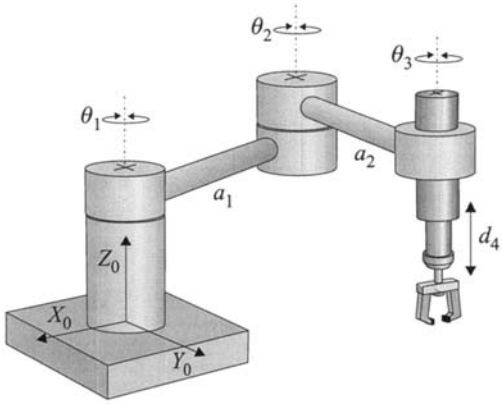

FIGURE 15 A SCARA geometry robot.

The Jacobian in Eq. (14.7) can be expressed as a square matrix by eliminating the fourth and fifth rows, which indicate that the end-effector does not have more than the only angular velocity, ω, with respect to the Z axis.

Robot Dynamics

The dynamic model of a robotic manipulator, that is, its equations of motion, can be obtained from the common laws of physics and Newtonian, Eulerian, and Lagrangian mechanics. The objective at hand is to relate the available joint torques/forces to the motion of the robot end effector. The problem is complicated due to the multibody structure of the (serial) open chain manipulator that is, the motion of the end effector is governed by the individual motions of a series of interconnected (rigid body) links, each driven by an actuator. The pertinent literature includes detailed algorithms for the real-time calculations (solutions) of the dynamic model. Herein, our focus will be only on the overall discussion of the typical robot dynamic model and not on its solution for the calculation of joint torques, for example, for a desired output (i.e., end-effector velocity, acceleration, and force output).

The two common approaches to modeling the dynamic behavior of a serial manipulator are the Lagrange–Euler and the Newton–Euler formulations. The latter method can be adapted to a recursive solution of the robot dynamics using the d'Alembert principle of equilibrium for each link individually.

The overall Newton–Euler dynamic model of the robot in joint-space coordinates can be expressed as

$$\mathbf{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{G}(\boldsymbol{\theta}) = \boldsymbol{\tau} \tag{14.8}$$

where $\boldsymbol{\theta}$, $\dot{\boldsymbol{\theta}}$, $\ddot{\boldsymbol{\theta}}$, and $\boldsymbol{\tau}$ are the (generic) joint displacement, velocity, acceleration, and torque vectors, respectively, $\mathbf{M}$ is the inertia matrix, $\mathbf{C}$ is the Coriolis matrix, and $\mathbf{G}$ is the gravity vector. As is clearly apparent from Eq. (14.8), the torque requirements are a function of the instantaneous robot configuration. For example, when accelerating around its base joint, a robot would require less torque if the arm were retracted (i.e., minimum distance from the shoulder), as opposed to being fully stretched.

Trajectory Planning

As discussed above, a robotic manipulator is required to move either in PTP or in CP motion modes in task space. However, robot motion control necessitates that commands be given in joint space to yield a required task space end-effector path. We must plan individual joint trajectories—joint displacement, velocity, and acceleration as a function of time, to meet this objective. PTP and CP motions are treated separately below:

Point-to-Point Motion: In PTP motion, the robot end-effector is required to move from its current point to another: a point is defined by a Cartesian frame location (position and orientation—"pose") with respect to a global (world) coordinate system (Fig. 14a). Although the path followed is of no significance, except for avoiding collisions, all manipulator joints must start and end their motions synchronously. Such a strategy would minimize acceleration periods and thus minimize joint torque requirements—a phenomenon not considered in NC machining due to the absence of significant inertial forces.

Trajectory planning for PTP motion starts by determining the joint displacement values at both ends of the motion corresponding to the two end-effector Cartesian frames F_{e1} and F_{e2}—inverse kinematics, ${}^{1}\theta_i$ and ${}^{2}\theta_i$, $i = 1, n$, for an n-dof manipulator. The next step involves determining individual joint trajectories: a vast majority of commercial robots only utilize kinematics to determine these trajectories; only a very few utilize the dynamic models of the manipulators. We will address both approaches below.

The dynamic model of the robot, Eq. (14.8), clearly indicates that the availability of joint torques to maximize joint velocities and accelerations is a function of the instantaneous robot configuration and the geometry and mass of the object carried. In the absence of dynamic model utilization during trajectory planning, one must therefore assume some logical limits

for the joint velocities and accelerations. Normally, worst-case values are utilized for these limits, i.e., assuming that the robot configuration is in its most unfavorable configuration and carrying a payload of maximum mass. Based on these limits and user-defined joint trajectory velocity profiles, trapezoidal, or parabolic, we must first determine the slowest joint, #k, i.e., the joint that will take the longest time to accomplish its motion, $\Delta\theta_k = {}^2\theta_k - {}^1\theta_k$. The time required to achieve $\Delta\theta_k$ is defined as the overall robot motion time for the end-effector to move from frame F_{e1} to F_{e2}. All other joints are slowed down to yield a synchronous motion for the robot that ends at time t_f.

Over the past three decades, many joint trajectory velocity profiles have been proposed. The two most common ones are shown in Fig. 16a and 16b for the slowest joint, θ_k.

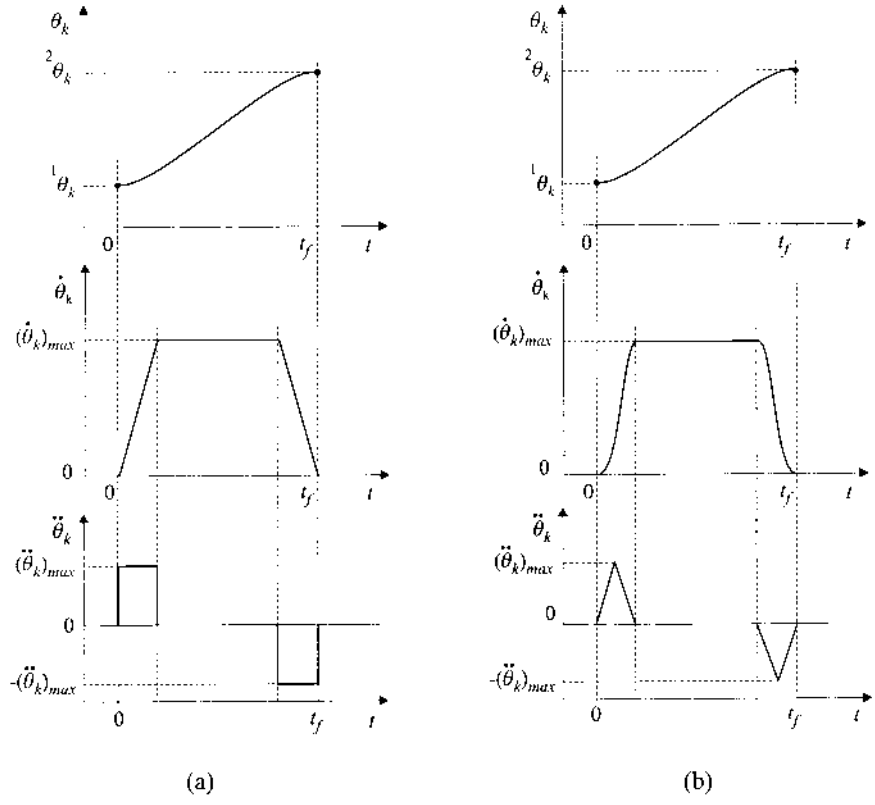

FIGURE 16 (a) Trapezoidal; (b) parabolic joint velocity profiles.

Once all joint trajectories have been planned, one may choose to carry out a computer simulation in order to determine whether the resultant PTP end-effector (Cartesian) path causes a collision. In case of a potential collision, we would need to select intermediate points to go around obstacles without actually stopping at these points. Such continuous PTP (CPTP) motion will be addressed below.

As noted above, PTP motion is normally carried at maximum joint speeds in order to minimize motion times and increase productivity. However, true minimum time PTP motion can only be obtained by considering the robot's dynamic model. Two such trajectory planning algorithms were developed in the early 1980s and have formed the basis of numerous other ones that followed them. The authors of these works were K. G. Shin and N. D. McKay from the University of Michigan at Ann Arbor and J. E. Bobrow, S. Dubowsky, and J. S. Gibson from the University of California at Irvine and Los Angeles and MIT, respectively. Both solution methods simultaneously determine the Cartesian end-effector path and corresponding joint trajectories that maximize joint torque utilization and thus minimize robot motion time subject to all robot kinematic and dynamic constraints.

Continuous Point-to-Point Motion: In CPTP motion, the robot end effector is required to move through all intermediate points (i.e., end-effector frames) without stopping and preferably following continuous joint velocity profiles. A common solution approach to this problem is to achieve velocity continuity at an intermediate point by accelerating/decelerating the joint motion prior to getting to that point and in the process, potentially, not pass through the point itself but only close to it. A preferred alternative strategy, however, would be the employment of spline curves for the individual segments of the joint trajectory.

Through an iterative process, one can fit cubic or quintic splines to all the trajectories of the robot's n joints, while satisfying displacement, velocity, and acceleration continuity at all the knots (Fig. 17). The overall motion time can be minimized through an iterative process or using parametric closed form equations, subject to all the joints' individual kinematic constraints (i.e., $\dot{\theta}_{\max}$ and $\ddot{\theta}_{\max}$).

Continuous Path Motion: In CP motion, the robot end-effector is required to follow a Cartesian path, normally with a constant speed (Fig. 14b). The motion (translation and rotation) of the end-effector frame is defined as a function of time. The solution of the (joint space) trajectory planning problem for CP motion requires discretization of the Cartesian path in terms of a set of points (frames) on this path separated by Cartesian distance and time. The robot can then be required to carry out a CPTP motion through these points, as described above, whose corresponding joint displacement values are determined by inverse kinematics.

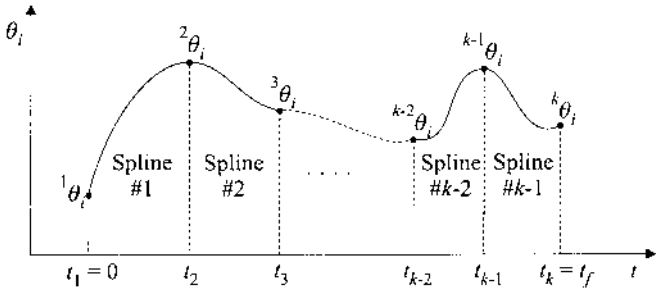

FIGURE 17 Spline fits for CPTP motion.

In CPTP motion, however, the end-effector can be forced to pass only through the selected set of points yielding a Cartesian path that approximates the desired path (e.g., a straight-line motion with constant end-effector orientation with respect to the workpiece). If the resultant path following errors in Cartesian space are greater than an acceptable threshold, we would have to increase the number of points selected to approximate the desired path.

As a generalization of CP motion for real-time implementation, one can simply use the inverse Jacobian matrix of the robot in determining joint velocities corresponding to the instantaneous end-effector velocity requirements (Eq. 14.5). This method, originally proposed by D. E. Whitney in the early 1970s, is commonly known as the resolved motion-rate-control method. If at any instant, the robot's dynamic capabilities cannot match the required end-effector motion requirements, a tracking error results.

Motion Control

Motion commands generated by a robot trajectory planning module are passed onto the individual joint controllers for their real-time implementation. As in NC machine tool control, the closed loop control of these (individual) manipulator joints yields the desired end-effector motion (Fig. 18). The majority of industrial robots employ DC (electric) servomotors for reliable displacement/velocity/acceleration/force control, though hydraulic drives have also been used in large load carrying applications (Chap. 12). The focus of this section is on the motion control of industrial robots that utilize DC servomotors.

The majority of industrial robots utilize PD or PID controllers, without relying on the existence of accurate manipulator dynamic models for trajectory tracking (Chap. 13). These controllers behave reasonably well

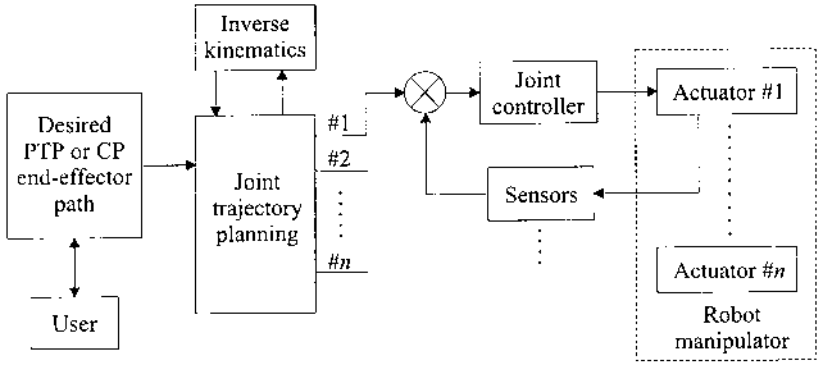

FIGURE 18 Robot motion control architecture.

for the execution of repetitive (constant load carrying and constant speed) motion tasks that do not involve (force-based) interactions with their environment. The joint acceleration commands in a PID controller can be simply calculated in the form

$$\ddot{\boldsymbol{\theta}} = \ddot{\boldsymbol{\theta}}_r + \mathbf{K}_d\dot{\mathbf{e}} + \mathbf{K}_p\mathbf{e} + \mathbf{K}_i \int \mathbf{e}\, dt \tag{14.9}$$

where the error vector term is $\mathbf{e} = \boldsymbol{\theta}_r - \boldsymbol{\theta}$ (i.e., the desired joint displacement value minus the current joint displacement value) and $\mathbf{K}_d$, $\mathbf{K}_p$, and $\mathbf{K}_i$ are the constant, diagonal (derivative, position, and integral) gain matrices, respectively.

Trajectory tracking can be significantly improved if the robot controller is provided with a reliable dynamic model. The computed torque (CT) technique is one of many control laws developed in the past two decades for the control of multilink manipulators using the robot dynamic model:

$$\boldsymbol{\tau} = \mathbf{M}(\boldsymbol{\theta})(\ddot{\boldsymbol{\theta}}_r - \mathbf{K}_d\dot{\mathbf{e}} - \mathbf{K}_p\mathbf{e}) + \mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{G}(\boldsymbol{\theta}) \tag{14.10}$$

The above torque-control equation, based on the joint space dynamic model of the robot [Eq. (14.8)], can be converted into a Cartesian space form using the Jacobian matrix of the robot.

In the case of frequently varying task space operating conditions with which the above-mentioned PID or CT controllers cannot cope, users may choose to employ an adaptive control (AC) scheme. Such AC schema update the feedback gains according to the latest output measurements in joint space and/or task space coordinates.

All the above discussion focused on the position/velocity control of industrial robots carrying out simple motion tasks, such as painting,

machine loading/unloading, or even spot welding. A variety of other industrial tasks, however, require the robot end-effector to exert controlled force on their environment (e.g., insertion, cutting). In order to cope with such phenomena, an industrial robot must be controlled using techniques such as impedance control or hybrid position/force control (Fig. 19). The former regulates the ratio of force to motion (i.e., mechanical impedance), while the latter decomposes the problem into two separate entities (i.e., force versus position control) and merges their solutions.

14.2.1 Robot Programming

The programming of industrial robots must be reviewed in the context of trajectory planning and control, as discussed above in Sec. 14.2.1. For PTP motion, the robot user aims at moving the manipulator from one point to another in the fastest possible manner with little regard to the actual path followed. For CP motion, on the other hand, a Cartesian path must be followed by the robot end-effector with a given velocity profile. Accordingly, one would expect to teach the robot the necessary points for PTP motion and the Cartesian path for CP motion, and instruct it, via a computer program, to execute the desired task.

In this section, robot programming will be addressed in three subsections: The first two subsections review the teaching and programming of robots in an on-line manner, which is valid for the vast majority of commercial robots, while the last subsection reviews the off-line programming of robot motions.

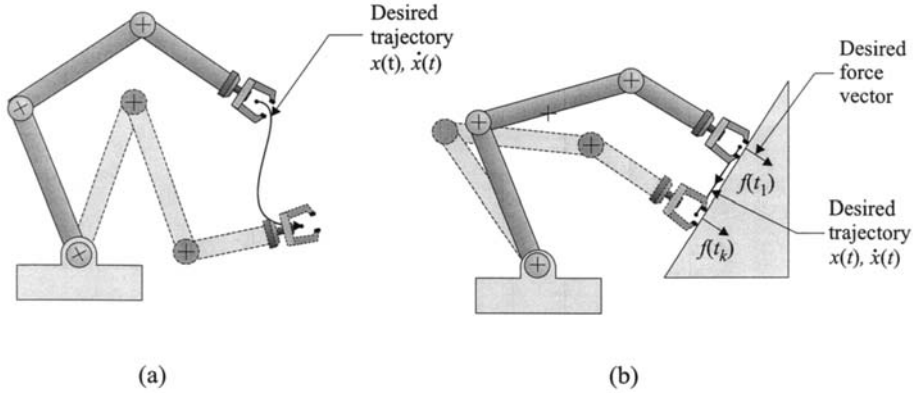

FIGURE 19 (a) Position control; (b) hybrid position/force control.

On-Line Teaching

The vast majority of commercial industrial robots are sold today without an accompanying kinematic model that would allow users to program the manipulator in an off-line manner. Furthermore, most industrial environments do not permit the specification of the exact locations (i.e., the reference coordinate frames) of the objects to be manipulated by the robot with respect to a global (world) coordinate frame. Thus the lack of a priori known precise (Cartesian) spatial relationships between a robot and objects necessitates the adaptation of teaching by showing (also known as lead-through) techniques. Naturally, such techniques are economically viable only for large batch sizes, where the cost of on-line programming (setup) of the industrial robot and, frequently, other production machines, can be divided and absorbed by the large number of (identical) parts. For small batch sizes or one-of-a-kind cost-effective manufacturing, we need to program all manufacturing equipment, including robots, in an off-line manner.

The most common method of teaching a fixed point (a Cartesian frame) is the use of a teach pendant (supplied with all commercial robots) (Fig. 20). A teach pendant allows the user to move the robot end-effector to any location within its workspace by moving the individual joints or by commanding the robot controller to move the end-effector in the Cartesian space. Cartesian space motion can be achieved either with respect to the robot's base frame or to its end-effector frame. Once a satisfactory location is obtained, the robot controller is asked to memorize this point. The majority of controllers memorize the joint displacement values corresponding to this point and not the Cartesian coordinates of the end-effector.

In regard to CP motion, the majority of commercial robots only allow straight line path following (with constant or varying end-effector orientation along the path) between two taught points; no other path geometries are permitted. Alternatively, users can chain link a large number of points for CPTP motion as an approximation of the continuous path.

For special purpose applications, such as spray painting, where one may need hundreds of points to approximate complex paths, some commercial companies provide users with a stripped-down version of the industrial robot for hand held lead-through teaching. The "slave" manipulator would be identical in mechanical configuration to its "master," except for the stripping of the high-ratio transmissions and other nonfunctional heavy components. Once the transmissions are removed, a human operator can physically hold the hand (or tool) of the robot and mimic the desired Cartesian space path, while the joint encoders/tachometers memorize the hundreds of points and time them.

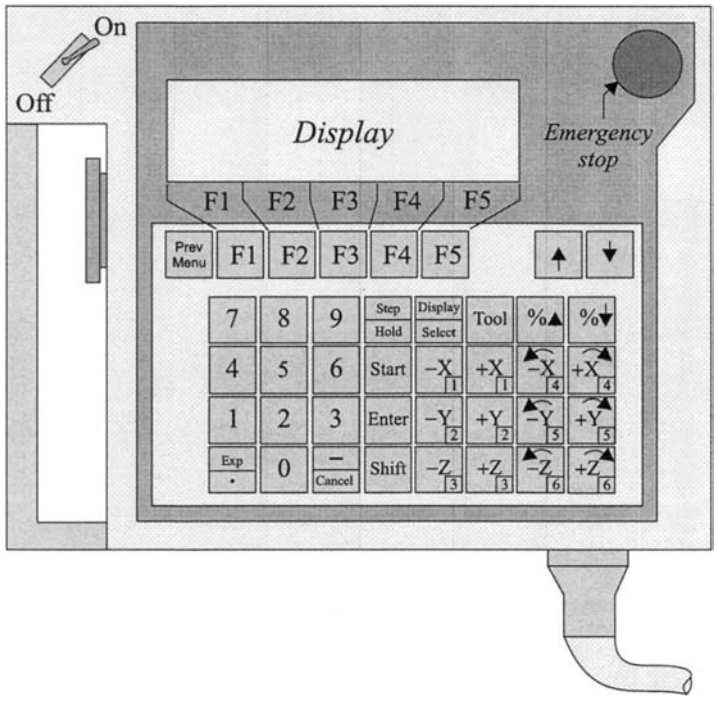

FIGURE 20 A GMF robot teach pendant.

Programming Languages

Industrial robots can be programmed to play back on-line taught tasks (PTP or CP motion). Unfortunately, over the past three decades, no single programming language has emerged and been adopted as a standard. Robot manufacturers today continue to market their hardware with accompanying proprietary software—operating system and programming language. As a consequence, there exist many robot programming languages, and industrial users must learn a variety of them if they own different makes of robots.

The majority of robot languages employ a limited number of commands (such as those of the APT language developed for machine tools): MOVE TO, APPROACH, OPEN GRIPPER, DEPART, etc. They also allow for sensory feedback using IF/THEN statements: for example, a robot end-effector can be instructed to move and grasp an object only after a positive sensory reading is obtained indicating object presence. At a higher level, a robot can also be instructed to follow a given trajectory subject to

potential minor positional variations based on real-time sensory feedback: for example, in arc welding, seam tracking is achieved by receiving continuous distance measurements from a proximity sensor attached to the robot end-effector (Chaps. 12 and 13).

In all the above-mentioned cases, the robot is instructed to move between pretaught points or follow prespecified trajectories with or without sensory feedback. There can be no gross variations from planned Cartesian paths or the specified order of tasks. Computer programs can be directly input into the robot controller (by directly typing on the provided console) or, when available, prepared on an external personal computer and downloaded using a serial communication port.

Off-Line Programming

The challenge of flexible manufacturing, in robotic environments, can only be met through the use of off-line programming, that is, without interrupting the current operation of the manipulator. For highly structured environ-

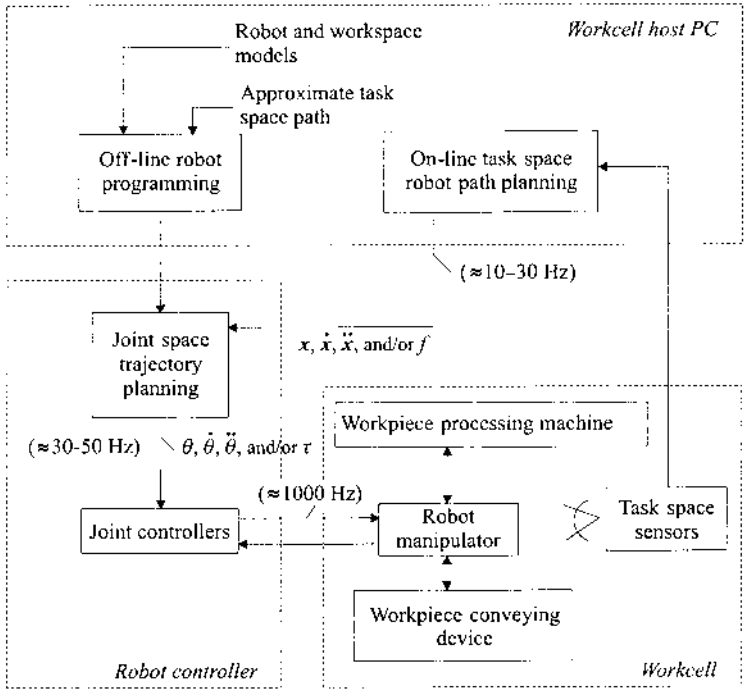

FIGURE 21 Robotic workcell planning/control.

ments, we must know the exact kinematic model of the robot as well as the geometric model of its environment in order to program off-line its motions. The commercial software CimStation Robotics, marketed by SILMA Inc. (a division of Adept Technology Inc.), allows users to create the CAD-based geometric model of their manufacturing environment and place in this environment the model of the industrial robot to be used (from an available database). Once all the motions are planned and graphically simulated, the user can ask the software for the automatic generation of the corresponding robot program (using a corresponding postprocessor, made available by SILMA) and download it to the robot's controller.

For manufacturing environments that are not highly structured (i.e., potential gross variations in object locations can be expected), no available robot programming language can be utilized for on-line or off-line programming of the industrial robot. In such cases, the robot must be programmed at the highest possible task level programming mode—for example, for the grasping of an object placed on a moving conveyor. This robot program must be supported by real-time information received from several sensors that monitor the robot's workspace and provide it with accurate and timely information regarding the object's latest state (position and velocity). Such languages and inference algorithms, although still in the research state, represent the near future of robotic implementation in loosely structured manufacturing environments. These programs/algorithms can also deal with inaccuracies in the robot's kinematic/dynamic models by directly feeding back information on the robot's end effector's (Cartesian-space) state in relation to the object's state (Fig. 21).

REVIEW QUESTIONS

1. Define the following terms as used for machine tools: numerical control (NC), computerized NC (CNC), and distributed NC (DNC).
2. Define point-to-point (PTP) motion and continuous path (CP) (contouring) motion in machining. Give specific examples.
3. Discuss adaptive control for machining. In your discussion make specific references to instrumentation and pattern recognition requirements.
4. Discuss multisensor integration and data fusion for *intelligent machining*.
5. Discuss *g*-code programming versus APT (or any other high-level language) programming of NC controllers.
6. Define the primary "words" used in *g*-code based programming.
7. Define computer-aided NC programming.
8. Discuss the objective in transforming motion commands between robot task space and joint space coordinate systems.

9. Describe the direct and inverse kinematic models for open chain serial robotic manipulators. How can one use the Denavit–Hartenberg (D-H) rigid body transformations in this context?
10. Why would one (time) differentiate the kinematic model of a robotic manipulator?
11. What is robot trajectory planning? Describe the need for using the manipulator's dynamic model in robot trajectory planning.
12. Define PTP motion and CP motion in robotics. How do these motion modes compare to their counterparts in machining? Explain similarities/differences.
13. What is trajectory tracking in robot motion control?
14. Compare robot on-line teaching to off-line programming. Discuss methods, feasibility, advantages/disadvantages, and so on.

DISCUSSION QUESTIONS

1. Discuss strategies for retrofitting an existing manufacturing enterprise with automation tools for material as well as information processing. Among others, consider issues such as buying turn-key solutions versus developing in-house solutions and carrying out consultations in a bottom-up approach, starting on the factory floor, versus a top-to-bottom approach, starting on the executive board of the company and progressing downward to the factory floor.
2. Job shops that produce one-of-a-kind products have been considered as the most difficult environments to automate, where a product can be manufactured within a few minutes or may require several days of fabrication. Discuss the role of computers in facilitating the transformation of such manual, skilled-labor dependent environments to intensive automation-based environments.
3. The factory of the future would be a totally networked enterprise. Information management in this enterprise would be a complex issue. In regards to planning, monitoring, and control, discuss the level of detail of information that the controllers (humans or computers) have to deal with in such environments. For example, some argue that in a hierarchical information management environment, activities are more of the planning type at the higher levels of the hierarchy and of the control type at the lower levels. It has also been argued that the level of detail significantly decreases as you ascend the enterprise ladder.
4. Machining centers with multitool turrets or other tool-changing mechanisms are designed to increase the operational flexibility of machine tools in terms of being able to carry out a variety of material removal

operations. Discuss the utilization of such machining centers as replacements for multiple single-objective machine tools.

5. When presented with a process planning problem for the machining of a nontrivial part, different (expert) machinists formulate different process plans. Naturally, only one of these plans is (time or cost) optimal. Considering this and other issues, compare manual (operator-based) machining versus NC-based machining, as enterprises are moving toward integrated and computerized manufacturing. Formulate at least one scenario where manual machining would be favorable.
6. An important beneficial feature of CNC machine tools is the potential of implementing an adaptive control strategy that would regulate the material removal (i.e., cutting) parameters in real time. Discuss the necessary conditions for such an implementation in terms of monitoring the material removal environment, signal processing, and decision making for making (real time) changes in input parameters.
7. Since most commercial CAD packages can automatically generate (controller specific) *g*-code programs for NC machines, discuss the value of learning a high-level NC programming language, such as APT and its variations, which would need to be subsequently also translated to *g*-code using a postprocessor (i.e., a compiler).
8. Computer-aided remote programming of NC machines is now possible through a factory's communications network. Most commercial CAD packages do allow users to plan cutting tool paths and automatically generate corresponding *g*-codes (for specific controllers) for such remote programming. However, one must note that none of these packages can optimize cutting parameters (e.g., depth of cut, feed rate). In the absence of such planners, discuss the logistics of generating tool paths automatically on a CAD workstation and the added value of this activity to the general computer-aided manufacturing planning process.
9. Tool wear can have a detrimental effect on satisfying the (geometric) dimensional specifications of a machined part, including its surface finish, especially for hard materials and complex three-dimensional surfaces. Discuss possible remedies to this problem in terms of on-line depth-of-cut compensation in turning, milling, and drilling. Address the issues of on-line sensory feedback (i.e., measurement of tool wear or object dimensions) and microscale depth-of-cut compensation using secondary (e.g., piezoceramic based) actuators (e.g., placed under the tool holder in turning).
10. The supervisory control of a manufacturing process relies on timely and accurate sensory feedback. However, owing to the difficult production conditions (high temperatures, high pressures, physical obstructions)

many production output parameters cannot be directly measured. Sensors instead observe and quantify certain physical phenomena (e.g., acoustic emissions) and relate these measurements to production output parameters (e.g., tool wear) that we desire to monitor. Discuss the following and other issues in the context of effective autonomous process control: the availability of effective signal-processing and pattern-recognition techniques, the use of multiple sensors to monitor one phenomenon (i.e., sensor fusion), the decoupling of information obtained by a sensor, whose outputs may have been influenced by several output parameters, the availability of models that could optimally change a machine input parameter in response to changes monitored in one or more output parameters.

11. In machining, tool change (because of wear) may constitute a significant part of setup time. This is especially true in multipoint cutting, such as milling, where all the inserts have to be replaced together. Almost a century-long work in the area of tool wear has yet to yield reliable models of the wear mechanisms, which would allow users to maximize utilization of the tools and thus minimize the number of tool changes. Discuss the use of a variety of sensors and pattern-recognition techniques for on-line intelligent machining in the absence of such models, or in support of approximate models.

12. Process planning in machining (in its limited definition) refers to the optimal selection of cutting parameters: the number of passes and tool paths for each pass, the depths of cut, the feed rates, the cutting velocities, and so on. It has been often advocated that computer algorithms be utilized for the search of the optimal parameter values. Although financially affordable in mass production environments, such (generative) programs may not be feasible in one-of-a-kind or small-batch production environments, where manufacturing times may be comparatively short. Discuss the utilization of group technology (GT)-based process planners in such computational-time limited production environments.

13. Analysis of a production process via computer-aided modeling and simulation can lead to an optimal process plan with significant savings in production time and cost. Discuss the issue of time and resources spent on obtaining an optimal plan and the actual (absolute) savings obtained due to this optimization, for example, spending several hours in planning to reduce production time from 2 minutes to 1 minute. Present your analysis as a comparison of one-of-a-kind production versus mass production.

14. NC machines can be efficiently programmed to execute a prepared process plan in terms of the relative motion of the tool and the

workpiece. Although this level of programmability does provide the users of such machines with automation and flexibility, the setup change requirements (e.g., workpiece fixturing) between products could negate these benefits. In this context, discuss effective ways of using NC machines in automated, flexible manufacturing environments.

15. Machining centers increase the automation/flexibility levels of machine tools by allowing the automatic change of cutting tools via turrets or tool magazines. Some machining centers also allow the off-line fixturing of workpieces onto standard pallets, which would minimize the on-line setup time (i.e., reduce downtime of the machine). While the machine is working on one part fixtured on Pallet 1, the next part can be fixtured on Pallet 2 and loaded onto the machine when it is has completed operating on the first part. Discuss the use of such universal machining centers versus the use of single-tool, single-pallet, unipurpose machine tools.

16. Automation of materials processing or handling equipment has been often associated with increased product quality and reduced production cost. Discuss the specific benefits associated with automating material removal machines (e.g., lathes, milling machines) using NC and CNC technologies in comparison to manual machines, where the operator measures and sets the cutting parameters manually. In your discussion, also address the issues of one-of-a-kind production versus mass production and flexible manufacturing versus automated manufacturing.

17. The load carrying capacity of a spatial mechanism, for example an industrial manipulator, is a function of the path it follows and the velocity and acceleration profile of its end-effector along this path. One must consider the dynamics of the mechanism's motion when planning its end-effector's paths. This problem is complex in nature and thus rarely addressed for industrial robots. Most suppliers simply specify worst-case scenarios when defining industrial robot specifications in regard to achievable speeds and load carrying capacities. How would one deal with such a problem when integrating a robot into a manufacturing system?

18. The absence of accurate robot kinematic models, compounded with the absence of accurate geometric world models of their working environments, has often forced users to define Cartesian locations through manual teaching techniques (teleoperation). The robot end-effectors are moved to their desired destinations through teach pendants, and the controllers are required to memorize joint encoder readings at these locations. The use of such playback-based robot motion techniques thus forces objects always to be at their expected locations with very stringent tolerances. Discuss the potential of using

a variety of task-space sensors in controlling the robot motion that can lead to the relaxing of positioning requirements for objects that are static or those that are in motion.

19. Industrial robots have been often labeled as being deaf and blind operators with, furthermore, no tactile feedback detection capability. Discuss in general terms what would be the benefits of having a variety of visual and nonvisual sensors monitoring the robot's working environment and feeding back accurate and timely information to the motion controller of such manipulators.

20. Bin picking is a term used for the robotic grasping of a single component from an open bin that contains many randomly oriented, identical (and sometimes not identical) components. Discuss difficulties associated with such operations. Propose alternative solutions to robotic bin picking.

21. The necessary programming of robot task space locations by physically moving the robot's end-effector to these positions, while it is taken off the manufacturing line, has severely limited their use to mass production environments. That is, a although industrial robots provide a high level of automation, they cannot be time-efficiently programmed and used for one-of-a-kind or small batch productions. Discuss potential remedies that would allow robots to be programmed for their next task while they are performing their current task.

22. Most industrial robots need to be programmed using proprietary computer languages also developed by the makers of these robots (or their controllers). Discuss the potential negative impact such a diversification of programming languages can have on the decision-making process of purchasers/integrators/users of such machines.

23. Industrial robots have been often designed to replace the human operator in manufacturing settings. The past several decades have shown us, however, that there still exist significant gaps between humans' and robots' abilities, primarily because of the unavailability of artificial perception technologies. Compare humans to pertinent anthropomorphic robots in terms of the following and other issues: mechanical configuration and mobility, power source, workspace, payload capacity, accuracy, communications (wireless!), supervisory control ability, sensory perception, ability to process data, coping with uncertainties, and working in hazardous environments.

24. Human operators have been argued to be intelligent, autonomous, and flexible when compared to industrial robots. Discuss several manufacturing applications in which one would tend to use human operators as opposed to industrial robots (even those supported by a variety of sensors) in the context of these three properties.

25. The primary contributing factors to the achievement of high accuracies in machine tools are (1) the employment of high-precision and rigid linear actuators, (2) their Cartesian configuration, in which the linear stages are stacked on top of each other, thus avoiding significant inertia problems, and (3) the possibility of employing interferometers that can measure linear distances smaller than half a light wavelength. Discuss all three factors as you consider alternatives to the design of machine tools, for example, the use of rotary joint–based industrial robots.

26. Manufacturing systems, supported by computers, can be classified as manual versus automated and flexible (reconfigurable, reprogrammable, etc.) versus rigid. Discuss these classifications and elaborate on the intersections of their domains (e.g., manual and flexible, etc.). Note that each classification may have sublevels and subclassifications (i.e., different "levels of gray").

27. In the factory of the future, no unexpected machine breakdowns will be experienced! Such an environment, however, can only be achieved if a preventive maintenance program is implemented, in which all machines and tools are modeled (mathematically and/or using heuristics). These models would allow manufacturers to schedule maintenance operations as needed. Discuss the feasibility of implementing factory-wide preventive maintenance programs in the absence of our ability to model completely all existing physical phenomena, and furthermore in the lack of a large variety of sensors that can monitor the states of these machines and provide timely feedback to such models.

BIBLIOGRAPHY

Altintas, Yusuf. (2000). *Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Vibrations, and CNC Design*. New York: Cambridge Press.

Amic, Peter J. (1997). *Computer Numerical Control Programming*. Upper Saddle River, NJ: Prentice Hall.

Angeles, Jorge. (1997). *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*. New York: Springer-Verlag.

Armarego, E. J. A., Brown, R. H. (1969). *The Machining of Metals*. Englewood Cliffs, NJ: Prentice-Hall.

Asada, H., Slotine, J-J. E. (1986). *Robot Analysis and Control*. New York: John Wiley.

Bedworth, David D., Henderson, Mark R., Wolfe, Philip M. (1991). *Computer-Integrated Design and Manufacturing*. New York: McGraw-Hill.

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

Boothroyd, Geoffrey, Knight, Winston A. (1989). *Fundamentals of Machining and Machine Tools*. New York: Marcel Dekker.

Brooks, R. R., Iyengar, S. S. (1998). *Multi-Sensor Fusion: Fundamentals and Applications with Software*. Upper Saddle River, NJ: Prentice Hall.

Chang, Chao-Hwa, Melkanoff, Michel A. (1989). *NC Machine Programming and Software Design*. Englewood Cliffs, NJ: Prentice Hall.

Choi, G. S., Wang, Z., Dornfeld, D. A. (1991). Adaptive optimal control of machining process using neural networks. IEEE Proceedings of the International Conference on Robotics and Automation, Sacramento, CA (pp. 1567–1572).

Craig, John J. (1989). *Introduction to Robotics: Mechanics and Control*. Reading, MA: Addison-Wesley.

Croft, Elizabeth A. (1995). On-Line Planning for Robotic Interception of Moving Objects. Ph.D. diss., Department of Mechanical Engineering, University of Toronto, Toronto, Canada.

Croft, E. A., Fenton, R. G., Benhabib, B. (Apr. 1998). Optimal rendezvous-point selection for robotic interception of moving objects. *IEEE Transactions on Systems, Man, and Cybernetics,* Part B 28(2):192–204.

DeGarmo, E. Paul, Black, J. T., Kohser, Ronald A. (1997). *Materials and Processes in Manufacturing*. Upper Saddle River, NJ: Prentice Hall.

DeVries, Warren R. (1992). *Analysis of Material Removal Processes*. New York: Springer-Verlag.

Diei, E. N., Dornfeld, D. A. (August 1987). Acoustic emission sensing of tool wear in face milling. *Transactions of the ASME, Journal of Engineering for Industry* 109(3):234–240.

Dornfeld, D. A. (1991). Monitoring of the machining process by means of acoustic emission sensors. ASTM Symposium on Acoustic Emission: Current Practice and Future Directions, Charlotte, NC (pp. 328–344).

Doyle, Lawrence E., et al (1985). *Manufacturing Processes and Materials for Engineers*. Englewood Cliffs, NJ: Prentice-Hall.

Drozda, Thomas J., Wick Charles, eds. (1998). *Tool and Manufacturing Engineers Handbook*. Dearborn, MI: Society of Manufacturing Engineers.

Fu, K. S., Gonzalez, R. C., Lee, C. S. G. (1987). *Robotics: Control, Sensing, Vision, and Intelligence*. New York: McGraw-Hill.

Groover, Mikell P. (1996). *Fundamentals of Modern Manufacturing: Materials, Processes, and Systems*. Upper Saddle River, NJ: Prentice-Hall.

Hine, Charles R. (1970). *Machine Tools and Processes for Engineers*. New York, NY: McGraw-Hill.

Hsu, P. L., Fann, W. R. (Nov. 1996). Fuzzy adaptive control of machining processes with a self-learning algorithm. *Transactions of the ASME, Journal of Manufacturing Science and Engineering* 118(4):522–530.

Hujic, D., Croft, E. A., Zak, G., Fenton, R. G., Mills, J. K., Benhabib, B. (Sept. 1998). Time-optimal interception of moving objects—an active prediction, planning and execution system. *IEEE/ASME Trans. on Mechatronics* 3(3):225–239.

Joshi, Rajive, Sanderson, Arthur C. (1999). *Multisensor Fusion: A Minimal Representation Framework*. River Edge, NJ: World Scientific.

Kalpakjian, Serope, Schmid, Steven R. (2000). *Manufacturing Engineering and Technology*. Upper Saddle River, NJ: Prentice Hall.
Koivo, Antti J. (1989). *Fundamentals for Control of Robotic Manipulators*. New York: John Wiley.
Koren, Yoram. (1983). *Computer Control of Manufacturing Systems*. New York: McGraw-Hill.
Koren, Yoram. (1985). *Robotics for Engineers*. New York: McGraw-Hill.
Laumond, J.-P. (1998). *Robot Motion Planning and Control*. New York: Springer-Verlag.
Lewis, Frank L., Abdallah, C. T., Dawson, D. M. (1993). *Control of Robot Manipulators*. New York: Maxwell Macmillan International.
Mehrandezh, Mehran. (1999). Navigation-Guidance Based Robotic Interception of Moving Objects. Ph.D. diss., Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada.
Mehrandezh, M., Sela, M. N., Fenton, R. G., Benhabib, B. (1999). Robotic interception of moving objects using ideal proportional navigation guidance technique. *J. of Robotics and Autonomous Systems* 28:295–310.
Mongi, A. Abidi, Gonzalez, Ralph C. (1992). *Data Fusion in Robotics and Machine Intelligence*. Boston: Academic Press.
McMillan, Gregory K., ed. (1999). *Process/Industrial Instruments and Controls Handbook*. New York: McGraw-Hill.
Murray, Richard M., Li, Zexiang, Sastry, S. Shankar (1994). *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL: CRC Press.
Nee, John G., ed. (1998). *Fundamentals of Tool Design*. Dearborn, MI: Society of Manufacturing Engineers.
Nof, Shimon Y., ed. (1999). *Handbook of Industrial Robotics*. New York: John Wiley.
Paul, Richard P. (1981). *Robot Manipulators: Mathematics, Programming, and Control*. Cambridge, MA: MIT Press.
Luo, Ren C., Kay, Michael G. (1995). *Multisensor Integration and Fusion for Intelligent Machines and Systems*. Norwood, NJ: Ablex.
Schey, John A. (1987). *Introduction to Manufacturing Processes*. New York: McGraw-Hill.
Sciavicco, L., Siciliano, Bruno (2000). *Modeling and Control of Robot Manipulators*. New York: Springer-Verlag.
Shiller, Z., Dubowsky, S. (1987). Acceleration map and its use in minimum-time motion planning of robotic manipulators. *ASME Proceedings of the International Computers in Engineering Conference,* New York (pp. 229–234).
Shin, K. C., McKay, N. D. (July 1985). Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Transactions of Automatic Control* 30(6):531–541.
Siciliano, B., Valavanis, K. P., eds. (1998). *Control Problems in Robotics and Automation*. New York: Springer-Verlag.
Spong, Mark W., Vidyasagar, M. (1989). *Robot Dynamics and Control*. New York: John Wiley.

Spong, Mark W., Lewis, Frank L., Abdallah, Chauki T., eds. (1993). *Robot Control: Dynamics, Motion Planning and Analysis*. New York: IEEE Press.
Stephenson, David A., Agapiou, John S. (1997). *Metal Cutting Theory and Practice*. Marcel Dekker, New York.
Szafarczyk, Maciej, ed. (1994). *Automatic Supervision in Manufacturing*. New York: Springer-Verlag.
Tabarah, Edward (1993). Multi-Arm Robot Kinematics/Dynamics. Ph.D. diss., Department of Mechanical Engineering, University of Toronto, Toronto, Canada.
Tabarah, E., Benhabib, B., Fenton, R. G. (Dec. 1994). Motion planning for cooperative robotic systems performing contact operations. *ASME J. of Mechanical Design* 116(4):1177–1180.
Thyer, G. E. (1991). *Computer Numerical Control of Machine Tools*. New York: Industrial Press.
Tlusty, Jiri (2000). *Manufacturing Processes and Equipment*. Upper Saddle River, NJ: Prentice Hall.
Tsai, Lung-Wen (1999). *Robot Analysis: The Mechanics of Serial and Parallel Manipulators*. New York: John Wiley.
Valavanis, K., Saridis, George N. (1992). *Intelligent Robotic Systems: Theory, Design, and Applications*. Boston, MA: Kluwer.
Valentino, James (2000). *Introduction to Computer Numerical Control*. Upper Saddle River, NJ: Prentice Hall.
Vickers, G. W., Ly, M., Oetter, R. G. (1990). *Numerically Controlled Machine Tools*. New York: Ellis Horwood.
Whitney, D. E. (Dec. 1972). The mathematics of coordinate control of prosthetic arms and manipulators. *ASME Transactions, Journal of Dynamic Systems, Measurement, and Control*, 303–309.
Whitney, D. E. (1987). Historic perspective and state-of-the-art in robot force control. *International Journal of Robotics Research* 6(1):3–14.
Wright, P. K., Hansen, F. B., Pavlakos, E. (1990). Tool wear and failure monitoring on an open-architecture machine tool. Winter Annual Meeting of the American Society of Mechanical Engineers, Dallas, TX (pp. 211–228).