Robotics

Frank L. Lewis
University of Texas at Arlington

John M. Fitzgerald
University of Texas at Arlington

Ian D. Walker
Rice University

Mark R. Cutkosky
Stanford University

Kok-Meng Lee
Georgia Tech

Ron Bailey
University of Texas at Arlington

Frank L. Lewis
University of Texas at Arlington

Chen Zhou
Georgia Tech

John W. Priest
University of Texas at Arlington

G. T. Stevens, Jr.
University of Texas at Arlington

John M. Fitzgerald
University of Texas at Arlington

Kai Liu
University of Texas at Arlington

14.1 Introduction

The word "robot" was introduced by the Czech playright Karel Čapek in his 1920 play *Rossum's Universal Robots.* The word "robota" in Czech means simply "work." In spite of such practical beginnings, science fiction writers and early Hollywood movies have given us a romantic notion of robots. Thus, in the 1960s robots held out great promises for miraculously revolutionizing industry overnight. In fact, many of the more far-fetched expectations from robots have failed to materialize. For instance, in underwater assembly and oil mining, teleoperated robots are very difficult to manipulate and have largely been replaced or augmented by "smart" quick-fit couplings that simplify the assembly task. However, through good design practices and painstaking attention to detail, engineers have succeeded in applying robotic systems to a wide variety of industrial and manufacturing situations where the environment is structured or predictable. Today, through developments in computers and artificial intelligence techniques and often motivated by the space program, we are on the verge of another breakthrough in robotics that will afford some levels of autonomy in unstructured environments.

On a practical level, robots are distinguished from other electromechanical motion equipment by their dexterous manipulation capability in that robots can work, position, and move tools and other objects with far greater dexterity than other machines found in the factory. Process robot systems are functional components with grippers, end effectors, sensors, and process equipment organized to perform a controlled sequence of tasks to execute a process — they require sophisticated control systems.

The first successful commercial implementation of process robotics was in the U.S. automobile industry. The word "automation" was coined in the 1940s at Ford Motor Company, as a contraction of "automatic motivation." By 1985 thousands of spot welding, machine loading, and material handling applications were working reliably. It is no longer possible to mass produce automobiles while meeting currently accepted quality and cost levels without using robots. By the beginning of 1995 there were over 25,000 robots in use in the U.S. automobile industry. More are applied to spot welding than any other process. For all applications and industries, the world's stock of robots is expected to exceed 1,000,000 units by 1999.

The single most important factor in robot technology development to date has been the use of microprocessor-based control. By 1975 microprocessor controllers for robots made programming and executing coordinated motion of complex multiple degrees-of-freedom (DOF) robots practical and reliable. The robot industry experienced rapid growth and humans were replaced in several manufacturing processes requiring tool and/or workpiece manipulation. As a result the immediate and cumulative dangers of exposure of workers to manipulation-related hazards once accepted as necessary costs have been removed.

A distinguishing feature of robotics is its multidisciplinary nature — to successfully design robotic systems one must have a grasp of electrical, mechanical, industrial, and computer engineering, as well as economics and business practices. The purpose of this chapter is to provide a background in all these areas so that design for robotic applications may be confronted from a position of insight and confidence. The material covered here falls into two broad areas: function and analysis of the single robot, and design and analysis of robot-based systems and workcells.

Section 14.2 presents the available configurations of commercial robot manipulators, with Section 14.3 providing a follow-on in mathematical terms of basic robot geometric issues. The next four sections provide particulars in end-effectors and tooling, sensors and actuators, robot programming languages, and dynamics and real-time control. Section 14.8 deals with planning and intelligent control. The next three sections cover the design of robotic systems for manufacturing and material handling. Specifically, Section 14.9 covers workcell layout and part feeding, Section 14.10 covers product design and economic analysis, and Section 14.11 deals with manufacturing and industrial processes. The final section deals with some special classes of robots including mobile robots, lightweight flexible arms, and the versatile parallel-link arms including the Stewart platform.

14.2 Commercial Robot Manipulators

John M. Fitzgerald

In the most active segments of the robot market, some end-users now buy robots in such large quantities (occasionally a single customer will order hundreds of robots at a time) that market prices are determined primarily by configuration and size category, not by brand. The robot has in this way become like an economic commodity. In just 30 years, the core industrial robotics industry has reached an important level of maturity, which is evidenced by consolidation and recent growth of robot companies. Robots are highly reliable, dependable, and technologically advanced factory equipment. There is a sound body of practical knowledge derived from a large and successful installed base. A strong foundation of theoretical robotics engineering knowledge promises to support continued technical growth.

The majority of the world's robots are supplied by established stable companies using well-established off-the-shelf component technologies. All commercial industrial robots have two physically separate basic elements: the manipulator arm and the controller. The basic architecture of all commercial robots is fundamentally the same. Among the major suppliers the vast majority of industrial robots uses digital servo-controlled electrical motor drives. All are serial link kinematic machines with no more than six axes (degrees of freedom). All are supplied with a proprietary controller. Virtually all robot applications require significant effort of trained skilled engineers and technicians to design and implement them. What makes each robot unique is how the components are put together to achieve performance that yields a competitive product. Clever design refinements compete for applications by pushing existing performance envelopes, or sometimes creating new ones. The most important considerations in the application of an industrial robot center on two issues: Manipulation and Integration.

Commercial Robot Manipulators

Manipulator Performance Characteristics

The combined effects of kinematic structure, axis drive mechanism design, and real-time motion control determine the major manipulation performance characteristics: reach and dexterity, payload, quickness, and precision. Caution must be used when making decisions and comparisons based on manufacturers' published performance specifications because the methods for measuring and reporting them are not standardized across the industry. Published performance specifications provide a reasonable comparison of robots of similar kinematic configuration and size, but more detailed analysis and testing will insure that a particular robot model can reach all of the poses and make all of the moves with the required payload and precision for a specific application.

Reach is characterized by measuring the extents of the space described by the robot motion and *dexterity* by the angular displacement of the individual joints. Horizontal reach, measured radially out from the center of rotation of the base axis to the furthest point of reach in the horizontal plane, is usually specified in robot technical descriptions. For Cartesian robots the range of motion of the first three axes describes the reachable workspace. Some robots will have unusable spaces such as dead zones, singular poses, and wrist-wrap poses inside of the boundaries of their reach. Usually motion test, simulations, or other analysis are used to verify reach and dexterity for each application.

Payload weight is specified by the manufacturer for all industrial robots. Some manufacturers also specify inertial loading for rotational wrist axes. It is common for the payload to be given for extreme velocity and reach conditions. Load limits should be verified for each application, since many robots can lift and move larger-than-specified loads if reach and speed are reduced. Weight and inertia of all tooling, workpieces, cables, and hoses must be included as part of the payload.

Quickness is critical in determining throughput but difficult to determine from published robot specifications. Most manufacturers will specify a maximum speed of either individual joints or for a specific kinematic tool point. Maximum speed ratings can give some indication of the robot's quickness but may be more confusing and misleading than useful. Average speed in a working cycle is the quickness

characteristic of interest. Some manufacturers give cycle times for well-described motion cycles. These motion profiles give a much better representation of quickness. Most robot manufacturers address the issue by conducting application-specific feasibility tests for customer applications.

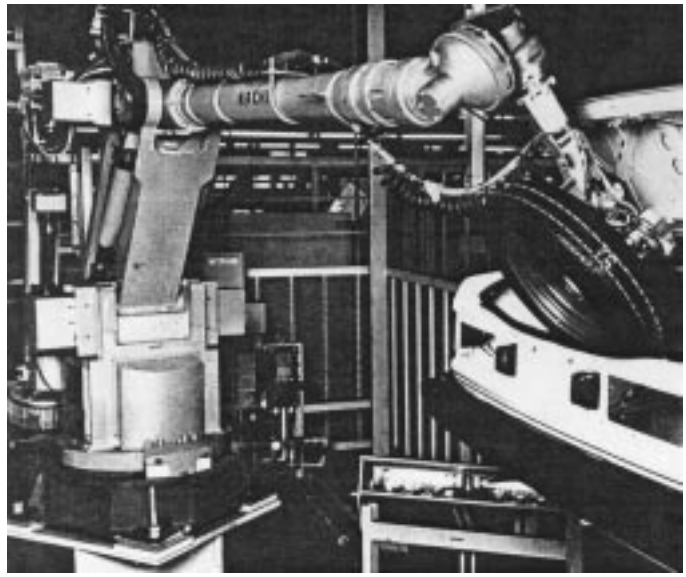
Precision is usually characterized by measuring repeatability. Virtually all robot manufacturers specify static position repeatability. Usually, tool point repeatability is given, but occasionally repeatability will be quoted for each individual axis. *Accuracy* is rarely specified, but it is likely to be at least four times larger than repeatability. Dynamic precision, or the repeatability and accuracy in tracking position, velocity, and acceleration on a continuous path, is not usually specified.

Common Kinematic Configurations

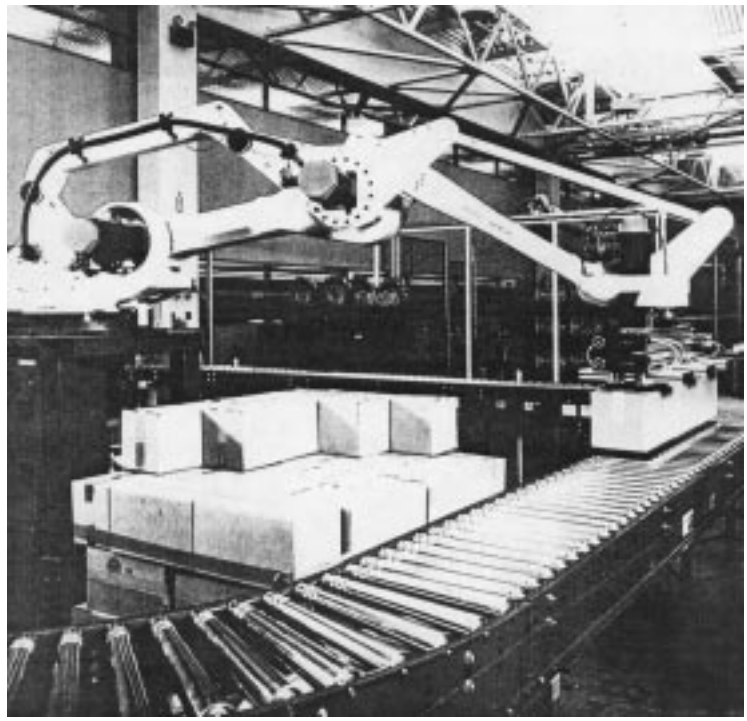
All common commercial industrial robots are serial link manipulators with no more than six kinematically coupled axes of motion. By convention, the axes of motion are numbered in sequence as they are encountered from the base on out to the wrist. The first three axes account for the spatial positioning motion of the robot; their configuration determines the shape of the space through which the robot can be positioned. Any subsequent axes in the kinematic chain provide rotational motions to orient the end of the robot arm and are referred to as wrist axes. There are, in principle, two primary types of motion that a *robot axis* can produce in its driven link: either *revolute* or *prismatic*. It is often useful to classify robots according to the orientation and type of their first three axes. There are four very common commercial robot configurations: Articulated, Type 1 *SCARA,* Type 2 SCARA, and Cartesian. Two other configurations, Cylindrical and Spherical, are now much less common.

Articulated Arms. The variety of commercial articulated arms, most of which have six axes, is very large. All of these robots' axes are revolute. The second and third axes are parallel and work together to produce motion in a vertical plane. The first axis in the base is vertical and revolves the arm sweeping out a large work volume. The need for improved reach, quickness, and payload have continually motivated refinements and improvements of articulated arm designs for decades. Many different types of drive mechanisms have been devised to allow wrist and forearm drive motors and gearboxes to be mounted close in to the first and second axis rotation to minimize the extended mass of the arm. Arm structural designs have been refined to maximize stiffness and strength while reducing weight and inertia. Special designs have been developed to match the performance requirements of nearly all industrial applications and processes. The workspace efficiency of well-designed articulated arms, which is the degree of quick dexterous reach with respect to arm size, is unsurpassed by other arm configurations when five or more degrees of freedom are needed. Some have wide ranges of angular displacement for both the second and third axis, expanding the amount of overhead workspace and allowing the arm to reach behind itself without making a 180° base rotation. Some can be inverted and mounted overhead on moving gantries for transportation over large work areas. A major limiting factor in articulated arm performance is that the second axis has to work to lift both the subsequent arm structure and payload. Springs, pneumatic struts, and counterweights are often used to extend useful reach. Historically, articulated arms have not been capable of achieving accuracy as well as other arm configurations. All axes have joint angle position errors which are multiplied by link radius and accumulated for the entire arm. However, new articulated arm designs continue to demonstrate improved repeatability, and with practical calibration methods they can yield accuracy within two to three times the repeatability. An example of extreme precision in articulated arms is the Staubli Unimation RX arm (see Figure 14.2.1).

Type I SCARA. The Type I SCARA (selectively compliant assembly robot arm) arm uses two parallel revolute joints to produce motion in the horizontal plane. The arm structure is weight-bearing but the first and second axes do no lifting. The third axis of the Type 1 SCARA provides work volume by adding a vertical or Z axis. A fourth revolute axis will add rotation about the Z axis to control orientation in the horizontal plane. This type of robot is rarely found with more than four axes. The Type 1 SCARA is used extensively in the assembly of electronic components and devices, and it is used broadly for the assembly of small- to medium-sized mechanical assemblies. Competition for robot sales in high speed electronics assembly has driven designers to optimize for quickness and precision of motion. A

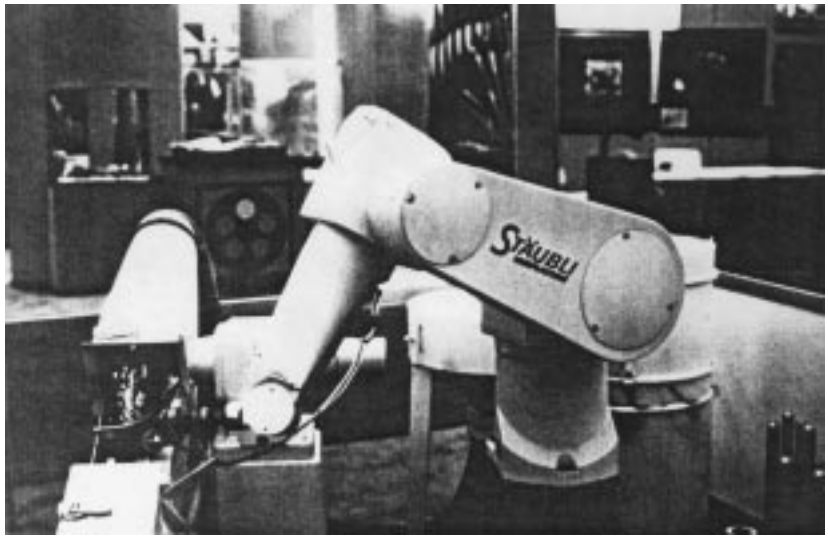


(a)

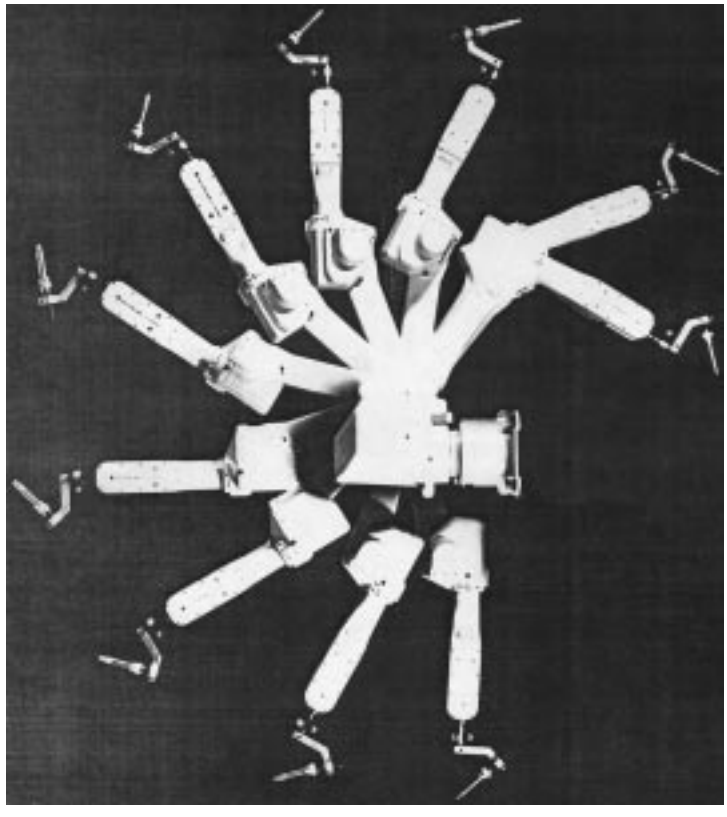


(b)

FIGURE 14.2.1 Articulated arms. (a) Six axes are required to manipulate spare wheel into place (courtesy Nachi, Ltd.); (b) four-axis robot unloading a shipping pallet (courtesy Fanuc Robotics, N.A.); (c) six-axis arm grinding from a casting (courtesy of Staubli Unimation, Inc.); (d) multiple exposure sideview of five-axis arc welding robot (courtesy of Fanuc Robotics, N.A.).



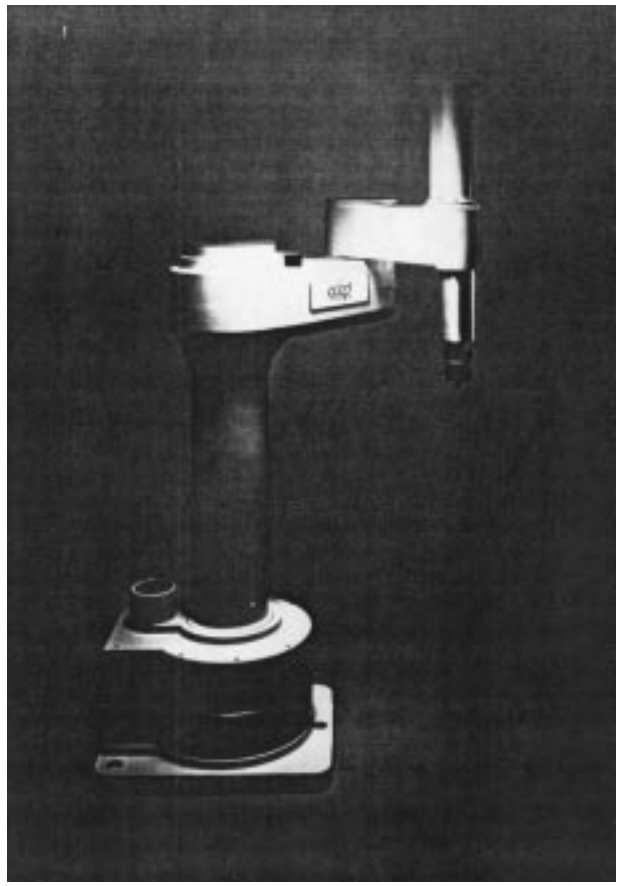
(c)



(d)

FIGURE 14.2.1 continued

well-known optimal SCARA design is the AdeptOne robot shown in Figure 14.2.2a. It can move a 20-lb payload from point "A" up 1 in. over 12 in. and down 1 in. to point "B" and return through the same path back to point "A" in less than 0.8 sec (see Figure 14.2.2).

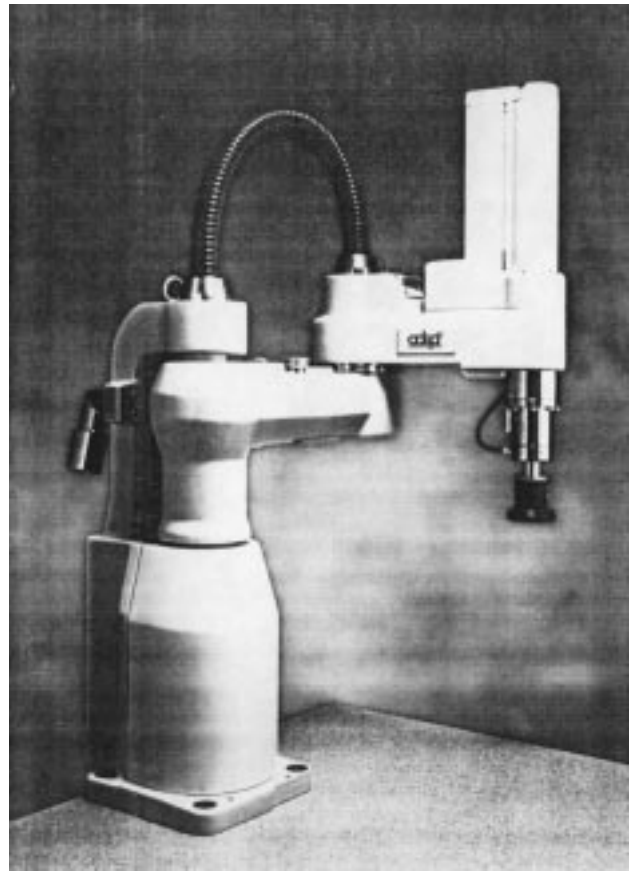


(a)

FIGURE 14.2.2 Type 1 SCARA arms (courtesy of Adept Technologies, Inc.). (a) High precision, high speed midsized SCARA; (b) table top SCARA used for small assemblies.

Type II SCARA. The Type 2 SCARA, also a four-axis configuration, differs from Type 1 in that the first axis is a long, vertical, prismatic Z stroke which lifts the two parallel revolute axes and their links. For quickly moving heavier loads (over approximately 75 lb) over longer distances (over about 3 ft), the Type 2 SCARA configuration is more efficient than the Type 1. The trade-off of weight vs. inertia vs. quickness favors placement of the massive vertical lift mechanism at the base. This configuration is well suited to large mechanical assembly and is most frequently applied to palletizing, packaging, and other heavy material handling applications (see Figure 14.2.3).

Cartesian Coordinate Robots. Cartesian coordinate robots use orthogonal prismatic axes, usually referred to as X, Y, and Z, to translate their end-effector or payload through their rectangular workspace. One, two, or three revolute wrist axes may be added for orientation. Commercial robot companies supply several types of Cartesian coordinate robots with workspace sizes ranging from a few cubic inches to tens of thousands of cubic feet, and payloads ranging to several hundred pounds. Gantry robots are the most common Cartesian style. They have an elevated bridge structure which translates in one horizontal direction on a pair of runway bearings (usually referred to as the X direction), and a carriage which



(b)

FIGURE 14.2.2 continued

moves along the bridge in the horizontal "Y" direction also usually on linear bearings. The third orthogonal axis, which moves in the Z direction, is suspended from the carriage. More than one robot can be operated on a gantry structure by using multiple bridges and carriages. Gantry robots are usually supplied as semicustom designs in size ranges rather than set sizes. Gantry robots have the unique capacity for huge accurate work spaces through the use of rigid structures, precision drives, and work-space calibration. They are well suited to material handling applications where large areas and/or large loads must be serviced. As process robots they are particularly useful in applications such as arc welding, waterjet cutting, and inspection of large, complex, precision parts.

Modular Cartesian robots are also commonly available from several commercial sources. Each module is a self-contained completely functional single axis actuator. Standard liner axis modules which contain all the drive and feedback mechanisms in one complete structural/functional element are coupled to perform coordinated three-axis motion. These modular Cartesian robots have work volumes usually on the order of 10 to 30 in. in X and Y with shorter Z strokes, and payloads under 40 lb. They are typically used in many electronic and small mechanical assembly applications where lower performance than Type 1 SCARA robots is suitable (see Figure 14.2.4).

Spherical and Cylindrical Coordinate Robots. The first two axes of the spherical coordinate robot are revolute and orthogonal to one another, and the third axis provides prismatic radial extension. The result is a natural spherical coordinate system and a work volume that is spherical. The first axis of cylindrical coordinate robots is a revolute base rotation. The second and third are prismatic, resulting in a natural cylindrical motion.

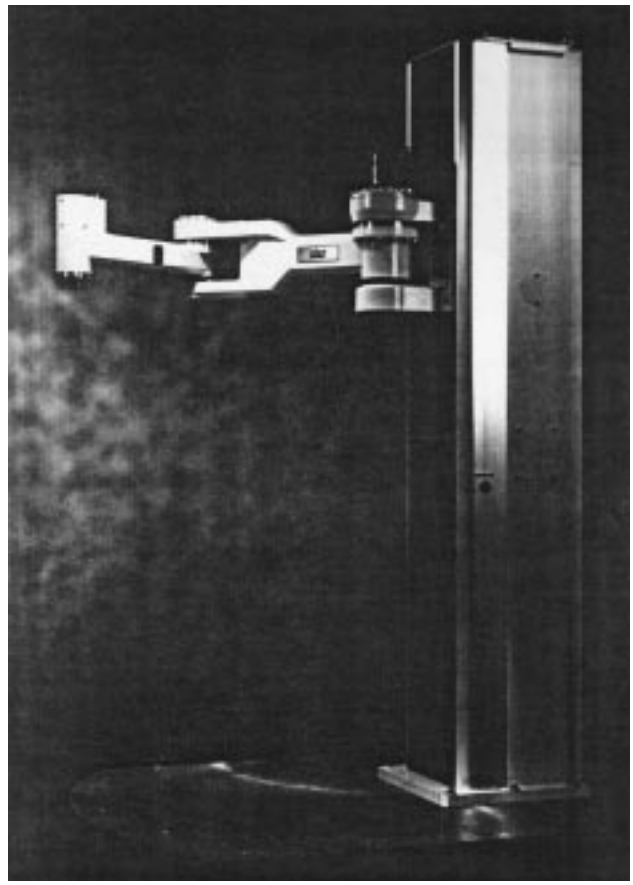
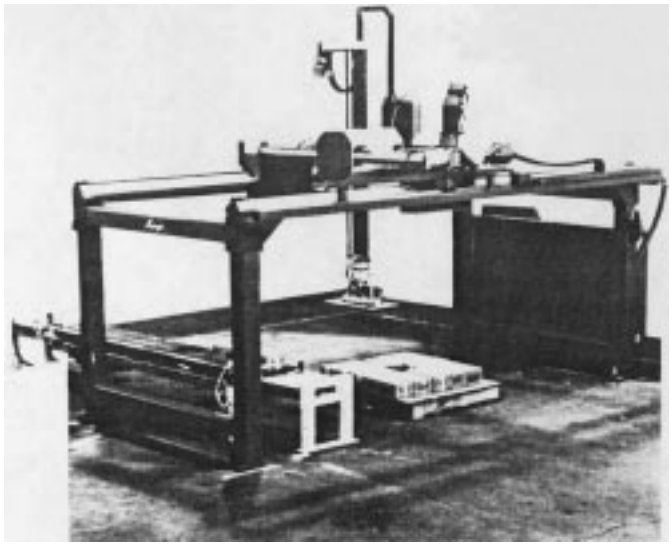


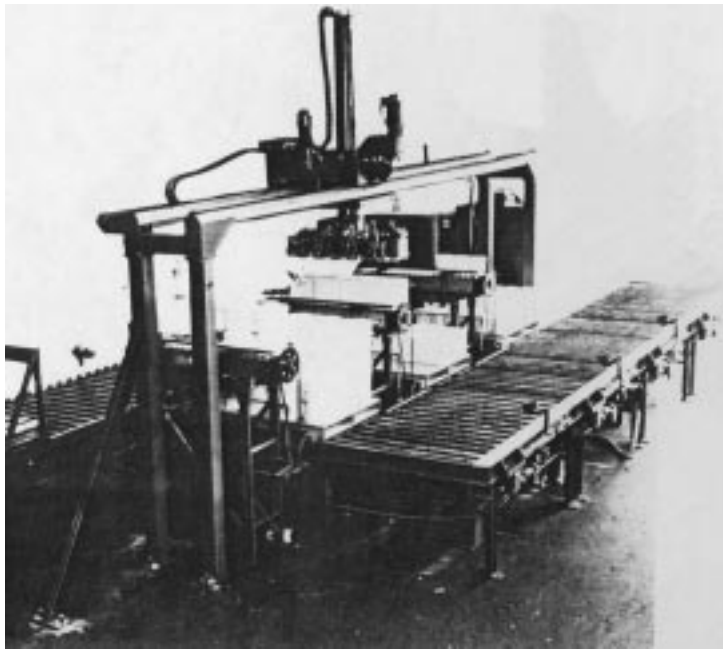
FIGURE 14.2.3 Type 2 SCARA (courtesy of Adept Technologies, Inc.).

Commerical models of spherical and cylindrical robots were originally very common and popular in machine tending and material handling applications. Hundreds are still in use but now there are only a few commercially available models. The Unimate model 2000, a hydraulic-powered spherical coordinate robot, was at one time the most popular robot model in the world. Several models of cylindrical coordinate robots were also available, including a standard model with the largest payload of any robot, the Prab model FC, with a payload of over 600 kg. The decline in use of these two configuations is attributed to problems arising from use of the prismatic link for radial extension/retraction motion. A solid boom requires clearance to fully retract. Hydraulic cylinders used for the same function can retract to less than half of their fully extended length. Type 2 SCARA arms and other revolute jointed arms have displaced most of the cylindrical and spherical coordinate robots (see Figure 14.2.5).

Basic Performance Specifications. Figure 14.2.6 sumarizes the kinematic configurations just described. Table 14.2.1 is a table of basic performance specifications of selected robot models that illustrates the broad spectrum of manipulator performance available from commercial sources. The information contained in the table has been supplied by the respective robot manufacturers. This is not an endorsement by the author or publisher of the robot brands selected, nor is it a verification or validation of the performance values. For more detailed and specific information on the availability of robots, the reader is advised to contact the Robotic Industries Association, 900 Victors Way, P.O. Box 3724, Ann Arbor, MI 48106, or a robot industry trade association in your country for a listing of commercial robot suppliers and system integrators.



(a)

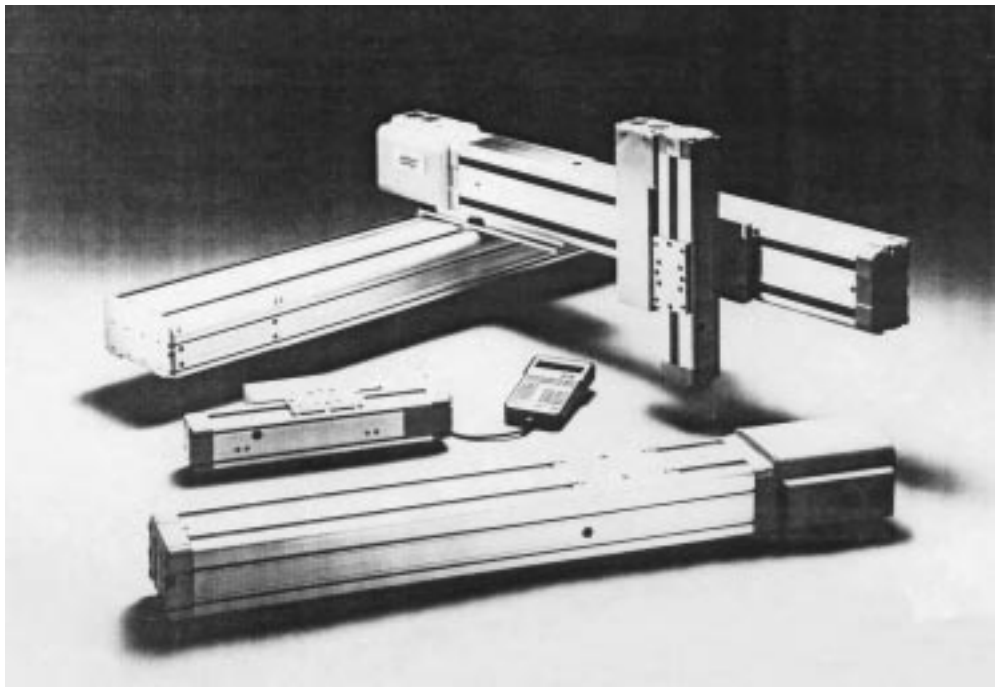


(b)

FIGURE 14.2.4 Cartesian robots. (a) Four-axis gantry robot used for palletizing boxes (courtesy of C&D Robotics, Inc.); (b) three-axis gantry for palletizing (courtesy of C&D Robotics, Inc.); (c) three-axis robot constructed from modular single-axis motion modules (courtesy of Adept Technologies, Inc.).

Drive Types of Commerical Robots

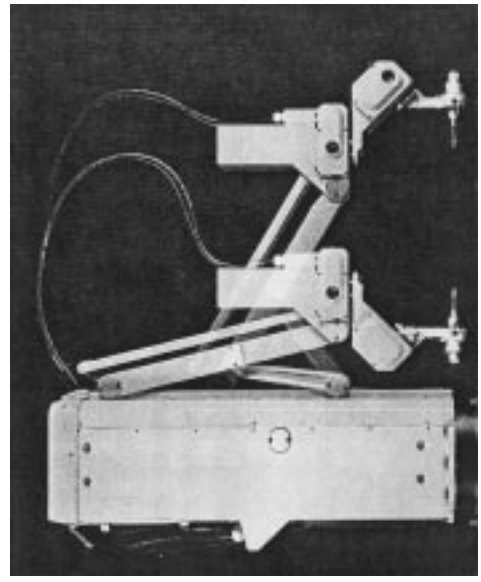
The vast majority of commerical industrial robots uses electric servo motor drives with speed-reducting transmissions. Both AC and DC motors are popular. Some servo hydraulic articulated arm robots are available now for painting applications. It is rare to find robots with servo pneumatic drive axes. All types of mechanical transmissions are used, but the tendency is toward low and zero backlash-type drives. Some robots use direct drive methods to eliminate the amplification of inertia and mechanical backlash associated with other drives. The first axis of the AdeptOne and AdeptThree Type I SCARA



(c)

FIGURE 14.2.4 continued

(a)



(b)

FIGURE 14.2.5 Spherical and cylindrical robots. (a) Hydraulic-powered spherical robot (courtesy Kohol Systems, Inc.); (b) cylindrical arm using scissor mechanism for radial prismatic motion (courtesy of Yamaha Robotics).

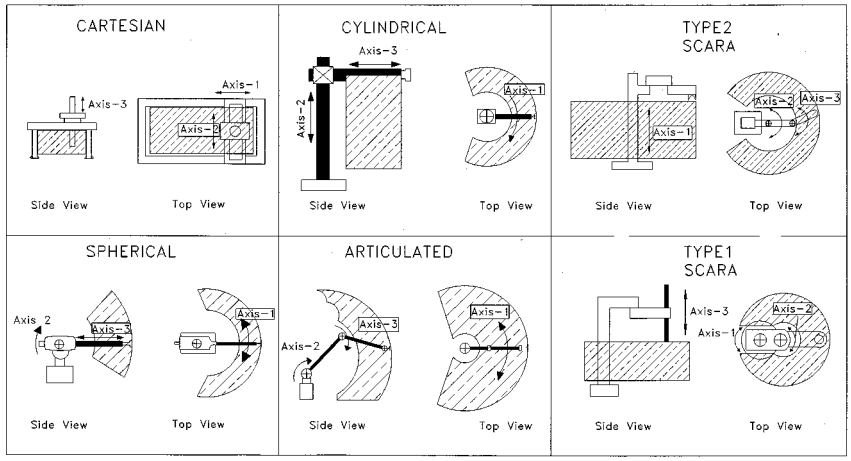


FIGURE 14.2.6 Common kinematic configurations for robots.

TABLE 14.2.1Basic Performance Specifications of Selected Commercial Robots

Configuration	Model	Axes	Payload (kg)	Reach (mm)	Repeatability (mm)	Speed
Articulated	Fanuc M-410i	4	155	3139	+/–0.5	axis 1, 85 deg/sec axis 2, 90 deg/sec axis 3, 100 deg/sec axis 4, 190 deg/sec
	Nachi 8683	6	200	2510	+/–0.5	N/A
	Nachi 7603	6	5	1405	+/–0.1	axis 1, 115 deg/sec axis 2, 115 deg/sec axis 3, 115 deg/sec
	Staubli RX90	6	6	985	+/–0.02	axis 1, 240 deg/sec axis 2, 200 deg/sec axis 3, 286 deg/sec
Type 1 SCARA	AdeptOne	4	9.1	800	+/–0.025	(est.) 1700 mm/sec
	Fanuc A-510	4	20	950	+/–0.065	N/A
Type 2 SCARA	Adept 1850	4	70	1850	X,Y +/–0.3 Z +/–0.3	axis 1, 1500 mm/sec axis 2, 120 deg/sec axis 3, 140 deg/sec axis 4, 225 deg/sec
	Staubli RS 184	4	60	1800	+/–0.15	N/A
Cartesian	PaR Systems XR225	5	190	X 18000 Y 5500 Z 2000	+/–0.125	N/A
	AdeptModules	3	15	X 500 Y 450	+/–0.02	axis 1, 1200 mm/sec axis 2, 1200 mm/sec axis 3, 600 mm/sec
Cylindrical	Kohol K45	4	34	1930	+/–0.2	axis 1, 90 deg/sec axis 2, 500 mm/sec axis 3, 1000 mm/sec
Spherical	Unimation 2000 (Hydraulic, not in production)	5	135		+/–1.25	axis 1, 35 deg/sec axis 2, 35 deg/sec axis 3, 1000 mm/sec

robots is a direct drive motor with the motor stator integrated into the robot base and its armature rotor integral with the first link. Other more common speed-reducing low backlash drive transmissions include toothed belts, roller chains, roller drives, and harmonic drives.

Joint angle position and velocity feedback devices are generally considered an important part of the drive axis. Real-time control performance for tracking position and velocity commands and precision is often affected by the fidelity of feedback. Resolution, signal-to-noise, and innate sampling frequency are important motion control factors ultimately limited by the type of feedback device used.

Given a good robot design, the quality of fabrication and assembly of the drive components must be high to yield good performance. Because of their precision requirements, the drive components are sensitive to manufacturing errors which can readily translate to less than specified manipulator performance.

Commercial Robot Controllers

Commercial robot controllers are specialized multiprocessor computing systems that provide four basic processes allowing integration of the robot into an automation system. These functions which must be factored and weighed for each specific application are Motion Generation, Motion/Process Integration, Human Integration, and Information Integration.

Motion Generation

There are two important controller-related aspects of industrial robot motion generation. One is the extent of manipulation that can be programmed; the other is the ability to execute controlled programmed motion. The unique aspect of each robot system is its real-time kinematic motion control. The details of real-time control are typically not revealed to the user due to safety and proprietary information secrecy reasons. Each robot controller, through its operating system programs, converts digital data into coordinated motion through precise coordination and high speed distribution and communication of the individual axis motion commands which are executed by individual joint controllers. The higher level programming accessed by the end user is a reflection of the sophistication of the real-time controller.

Of greatest importance to the robot user is the motion programming. Each robot manufacturer has its own proprietary programming language. The variety of motion and position command types in a programming language is usually a good indication of the robot's motion generation capability. Program commands which produce complex motion should be available to support the manipulation needs of the application. If palletizing is the application, then simple methods of creating position commands for arrays of positions are essential. If continuous path motion is needed, an associated set of continuous motion commands should be available. The range of motion generation capabilities of commercial industrial robots is wide. Suitability for a particular application can be determined by writing test code.

Motion/Process Integration

Motion/process integration involves methods available to coordinate manipulator motion with process sensor or process controller devices. The most primitive process integration is through discrete digital I/O. For example, an external (to the robot controller) machine controller might send a one-bit signal indicating whether it is ready to be loaded by the robot. The robot control must have the ability to read the signal and to perform logical operations (if then, wait until, do until, etc.) using the signal. At the extreme of process integration, the robot controller can access and operate on large amounts of data in real time during the execution of motion-related processes. For example, in arc welding, sensor data are used to correct tool point positions as the robot is executing a weld path. This requires continuous communication between the welding process sensor and the robot motion generation functions so that there are both a data interface with the controller and motion generation code structure to act on it. Vision-guided high precision pick and place and assembly are major applications in the electronics and semiconductor industries. Experience has shown that the best integrated vision/robot performance has come from running both the robot and the vision system internal to the same computing platform. The

reasons are that data communication is much more efficient due to data bus access, and computing operations are coordinated by one operating system.

Human Integration

Operator integration is critical to the expeditious setup, programming, and maintenance of the robot system. Three controller elements most important for effective human integration are the human *I/O devices*, the information available to the operator in graphic form, and the modes of operation available for human interaction. Position and path teaching effort is dramatically influenced by the type of manual I/O devices available. A teach pendant is needed if the teacher must have access to several vantage points for posing the robot. Some robots have teleoperator-style input devices which allow coordinated manual motion command inputs. These are extremely useful for teaching multiple complex poses. Graphical interfaces, available on some industrial robots, are very effective for conveying information to the operator quickly and efficiently. A graphical interface is most important for applications which require frequent reprogramming and setup changes. Several very useful off-line programming software systems are available from third-party suppliers. These systems use computer models of commercially available robots to simulate path motion and provide rapid programming functions.

Information Integration

Information integration is becoming more important as the trend toward increasing flexibility and agility impacts robotics. Automatic and computer-aided robot task planning and process control functions will require both access to data and the ability to resolve relevant information from CAD systems, process plans and schedules, upstream inspections, and other sources of complex data and information. Many robot controllers now support information integration functions by employing integrated PC interfaces through the communications ports, or in some through direct connections to the robot controller data bus.

14.3 Robot Configurations

Ian D. Walker

Fundamentals and Design Issues

A robot manipulator is fundamentally a collection of *links* connected to each other by *joints*, typically with an *end effector* (designed to contact the environment in some useful fashion) connected to the mechanism. A typical arrangement is to have the links connected serially by the joints in an open-chain fashion. Each joint provides one or more degree of freedom to the mechanism.

Manipulator designs are typically characterized by the number of independent degrees of freedom in the mechanism, the types of joints providing the degrees of freedom, and the geometry of the links connecting the joints. The degrees of freedom can be revolute (relative rotational motion θ between joints) or prismatic (relative linear motion d between joints). A joint may have more than one degree of freedom. Most industrial robots have a total of six independent degrees of freedom. In addition, most current robots have essentially rigid links (we will focus on rigid-link robots throughout this section).

Robots are also characterized by the type of actuators employed. Typically manipulators have hydraulic or electric actuation. In some cases where high precision is not important, pneumatic actuators are used.

A number of successful manipulator designs have emerged, each with a different arrangement of joints and links. Some "elbow" designs, such as the PUMA robots and the SPAR Remote Manipulator System, have a fairly anthropomorphic structure, with revolute joints arranged into "shoulder," "elbow," and "wrist" sections. A mix of revolute and prismatic joints has been adopted in the Stanford Manipulator and the SCARA types of arms. Other arms, such as those produced by IBM, feature prismatic joints for the "shoulder," with a spherical wrist attached. In this case, the prismatic joints are essentially used as positioning devices, with the wrist used for fine motions.

The above designs have six or fewer degrees of freedom. More recent manipulators, such as those of the Robotics Research Corporation series of arms, feature seven or more degrees of freedom. These arms are termed kinematically redundant, which is a useful feature as we will see later.

Key factors that influence the design of a manipulator are the tractability of its geometric (kinematic) analysis and the size and location of its workspace. The workspace of a manipulator can be defined as the set of points that are reachable by the manipulator (with fixed base). Both shape and total volume are important. Manipulator designs such as the SCARA are useful for manufacturing since they have a simple semicylindrical connected volume for their workspace (Spong and Vidyasagar, 1989), which facilitates workcell design. Elbow manipulators tend to have a wider volume of workspace, however the workspace is often more difficult to characterize. The kinematic design of a manipulator can tailor the workspace to some extent to the operational requirements of the robot.

In addition, if a manipulator can be designed so that it has a simplified kinematic analysis, many planning and control functions will in turn be greatly simplified. For example, robots with spherical wrists tend to have much simpler inverse kinematics than those without this feature. Simplification of the kinematic analysis required for a robot can significantly enhance the real-time motion planning and control performance of the robot system. For the rest of this section, we will concentrate on the kinematics of manipulators.

For the purposes of analysis, a set of *joint variables* (which may contain both revolute and prismatic variables), are augmented into a vector q, which uniquely defines the geometric state, or configuration of the robot. However, task description for manipulators is most naturally expressed in terms of a different set of *task coordinates*. These can be the position and orientation of the robot end effector, or of a special task frame, and are denoted here by Y. Thus Y most naturally represents the performance of a task, and q most naturally represents the mechanism used to perform the task. Each of the coordinate systems q and Y contains information critical to the understanding of the overall status of the manipulator. Much of the kinematic analysis of robots therefore centers on transformations between the various sets of coordinates of interest.

Manipulator Kinematics

The study of manipulator kinematics at the position (geometric) level separates naturally into two subproblems: (1) finding the position/orientation of the end effector, or task, frame, given the angles and/or displacements of the joints *(Forward Kinematics);* and (2) finding possible angles/displacements of the joints given the position/orientation of the end effector, or task, frame *(Inverse Kinematics)*. At the velocity level, the *Manipulator Jacobian* relates joint velocities to end effector velocities and is important in motion planning and for identifying *Singularities*. In the case of *Redundant Manipulators*, the Jacobian is particularly crucial in planning and controlling robot motions. We will explore each of these issues in turn in the following subsections.

Example 14.3.1

Figure 14.3.1 shows a planar three-degrees-of-freedom manipulator. The first two joints are revolute, and the third is prismatic. The end effector position (x, y) is expressed with respect to the (fixed) world coordinate frame (x_0, y_0), and the orientation of the end effector is defined as the angle of the second link ϕ measured from the x_0 axis as shown. The link length l_1 is constant. The joint variables are given by the angles θ_1 and θ_2 and the displacement d_3, and are defined as shown. The example will be used throughout this section to demonstrate the ideas behind the various kinematic problems of interest.

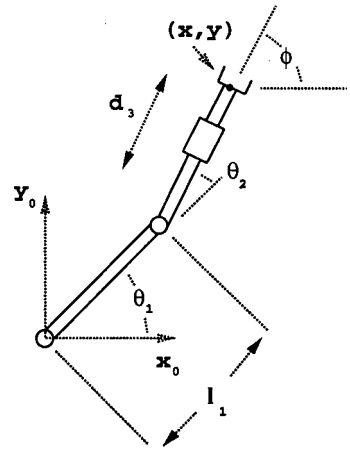


FIGURE 14.3.1 Planar RRP manipulator.

Forward (Direct) Kinematics

Since robots typically have sensors at their joints, making available measurements of the joint configurations, and we are interested in performing tasks at the robot end effector, a natural issue is that of determining the end effector position/orientation Y given a joint configuration q. This problem is the forward kinematics problem and may be expressed symbolically as

$$Y = f(q) \tag{14.3.1}$$

The forward kinematic problem yields a unique solution for Y given q. In some simple cases (such as the example below) the forward kinematics can be derived by inspection. In general, however, the relationship f can be quite complex. A systematic method for determining the function f for any manipulator geometry was proposed by Denavit and Hartenberg (Denavit and Hartenberg, 1955).

The Denavit/Hartenberg (or D-H) technique has become the standard method in robotics for describing the forward kinematics of a manipulator. Essentially, by careful placement of a series of coordinate

frames fixed in each link, the D-H technique reduces the forward kinematics problem to that of combining a series of straightforward consecutive link-to-link transformations from the base to the end effector frame. Using this method, the forward kinematics for any manipulator is summarized in a table of parameters (the D-H parameters). A maximum of three nonzero parameters per link are sufficient to uniquely specify the map *f*. Lack of space prevents us from detailing the method further. The interested reader is referred to Denavit and Hartenberg (1955) and Spong and Vidyasagar (1989).

To summarize, forward kinematics is an extremely important problem in robotics which is also well understood, and for which there is a standard solution technique

Example 14.3.2

In our example, we consider the task space to be the position and orientation of the end effector, i.e., $Y = [x, y, \phi]^T$ as shown. We choose joint coordinates (one for each degree of freedom) by $q = [\theta_1, \theta_2, d_3]^T$. From Figure 14.3.1, with the values as given it may be seen by inspection that

$$x = l_1 \cos(\theta_1) + d_3 \cos(\theta_1 + \theta_2) \tag{14.3.2}$$

$$y = l_1 \sin(\theta_1) + d_3 \sin(\theta_1 + \theta_2) \tag{14.3.3}$$

$$\phi = \theta_1 + \theta_2 \tag{14.3.4}$$

Equations (14.3.2) to (14.3.4) form the forward kinematics for the example robot. Notice that the solution for $Y = [x, y, \phi]^T$ is unique given $q = [\theta_1, \theta_2, d_3]^T$.

Inverse Kinematics

The *inverse kinematics* problem consists of finding possible joint configurations *q* corresponding to a given end effector position/orientation *Y*. This transformation is essential for planning joint positions of the manipulator which will result in desired end effector positions (note that task requirements will specify *Y*, and a corresponding *q* must be planned to perform the task). Conceptually the problem is stated as

$$q = f^{-1}(Y) \tag{14.3.5}$$

In contrast to the forward kinematics problem, the inverse kinematics cannot be solved for arbitrary manipulators by a systematic technique such as the Denavit-Hartenberg method. The relationship (1) does not, in general, invert to a unique solution for *q*, and, indeed, for many manipulators, expressions for *q* cannot even be found in closed form!

For some important types of manipulator design (particularly those mechanisms featuring spherical wrists), closed-form solutions for the inverse kinematics can be found. However, even in these cases, there are at best multiple solutions for *q* (corresponding to "elbow-up," "elbow-down" possibilities for the arm to achieve the end effector configuration in multiple ways). For some designs, there may be an infinite number of solutions for *q* given *Y*, such as in the case of kinematically redundant manipulators discussed shortly.

Extensive investigations of manipulator kinematics have been performed for wide classes of robot designs (Bottema and Roth, 1979; Duffy, 1980). A significant body of work has been built up in the area of inverse kinematics. Solution techniques are often determined by the geometry of a given manipulator design. A number of elegant techniques have been developed for special classes of manipulator designs, and the area continues to be the focus of active research. In cases where closed-form solutions cannot be found, a number of iterative numerical techniques have been developed.

Example 14.3.3

For our planar manipulator, the inverse kinematics requires the solution for $q = [\theta_1, \theta_2, d_3]^T$ given $Y = [x, y, \phi]^T$. Figure 14.3.2 illustrates the situation, with $[x, y, \phi]^T$ given as shown. Notice that for the Y specified in Figure 14.3.2, there are two solutions, corresponding two distinct configurations q.

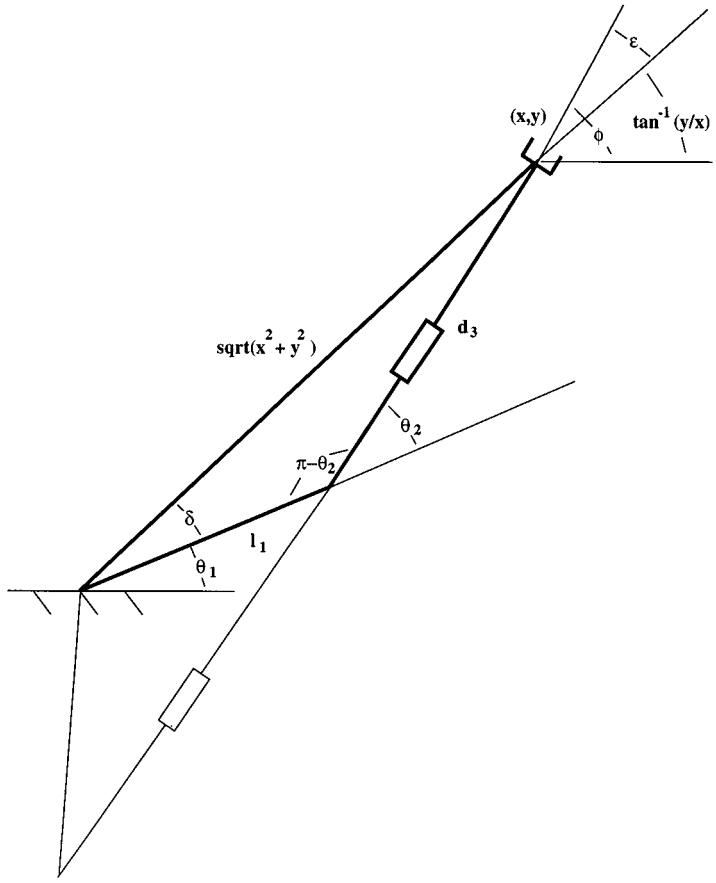


FIGURE 14.3.2 Planar RRP arm inverse kinematics.

The two solutions are sketched in Figure 14.3.2, with the solution for the configuration in bold the focus of the analysis below. The solutions may be found in a number of ways, one of which is outlined here. Consider the triangle formed by the two links of the manipulator and the vector (x, y) in Figure 14.3.2. We see that the angle ε can be found as

$$\varepsilon = \phi - \tan^{-1}(y/x)$$

Now, using the sine rule, we have that

$$l_1/\sin(\varepsilon) = \left(\sqrt{x^2 + y^2}\right)\Big/\sin(\pi - \theta_2) = \left(\sqrt{x^2 + y^2}\right)\Big/\sin(\theta_2)$$

and thus

$$\sin(\theta_2) = \left(\sqrt{x^2 + y^2}\right)\sin(\varepsilon)/l_1$$

The above equation could be used to solve for θ_2. Alternatively, we can find θ_2 as follows. Defining D to be $(\sqrt{x^2 + y^2})\sin(\varepsilon)/l_1$ we have that $\cos(\theta_2) = \pm\sqrt{1 - D^2}$. Then θ_2 can be found as

$$\theta_2 = \tan^{-1}\left[D/\pm\sqrt{1 - D^2}\right] \tag{14.3.6}$$

Notice that this method picks out both possible values of θ_2, corresponding to the two possible inverse kinematic solutions. We now take the solution for θ_2 corresponding to the positive root of $\pm(\sqrt{1 - D^2})$ (i.e., the bold robot configuration in the figure).

Using this solution for θ_2, we can now solve for θ_1 and d_3 as follows. Summing the angles inside the triangle in Figure 14.3.2, we obtain $\pi - [(\pi - \theta_2) + \varepsilon + \delta] = 0$ or

$$\delta = \theta_2 - \varepsilon$$

From Figure 14.3.2 we see that

$$\theta_1 = \tan^{-1}(y/x) - \delta \tag{14.3.7}$$

Finally, use of the cosine rule leads us to a solution for d_3:

$$d_3^2 = l_1^2 + \left(x^2 + y^2\right) - 2l_1\left(\sqrt{x^2 + y^2}\right)\cos(\delta)$$

or

$$d_3 = \sqrt{l_1^2 + \left(x^2 + y^2\right) - 2l_1\left(\sqrt{x^2 + y^2}\right)\cos(\delta)} \tag{14.3.8}$$

Equations (14.3.6) to (14.3.8) comprise an inverse kinematics solution for the manipulator.

Velocity Kinematics: The Manipulator Jacobian

The previous techniques, while extremely important, have been limited to positional analysis. For motion planning purposes, we are also interested in the relationship between joint velocities and task (end effector) velocities. The (linearized) relationship between the joint velocities $\dot{q}$ and the end effector velocities $\dot{Y}$ can be expressed (from Equation (14.3.1)) as

$$\dot{Y} = \left[J(q)\right]\dot{q} \tag{14.3.9}$$

where J is the *manipulator Jacobian* and is given by $\partial f/\partial q$. The manipulator Jacobian is an extremely important quantity in robot analysis, planning, and control. The Jacobian is particularly useful in determining singular configurations, as we shall see shortly.

Given the forward kinematic function f, the Jacobian can be obtained by direct differentiation (as in the example below). Alternatively, the Jacobian can be obtained column by column in a straightforward fashion from quantities in the Denavit-Hartenberg formulation referred to earlier. Since the Denavit-Hartenberg technique is almost always used in the forward kinematics, this is often an efficient and preferred method. For more details of this approach, see Spong and Vidyasagar (1989).

The Jacobian can be used to perform inverse kinematics at the velocity level as follows. If we define $[J^{-1}]$ to be the inverse of the Jacobian (assuming J is square and nonsingular), then

$$\dot{q} = \left[J^{-1}(q)\right]\dot{Y} \tag{14.3.10}$$

and the above expression can be solved iteratively for $\dot{q}$ (and hence q by numerical integration) given a desired end effector trajectory $\dot{Y}$ and the current state q of the manipulator. This method for determining joint trajectories given desired end effector trajectories is known as Resolved Rate Control and has become increasingly popular. The technique is particularly useful when the positional inverse kinematics is difficult or intractable for a given manipulator.

Notice, however, that the above expression requires that *J is* both nonsingular and square. Violation of the nonsingularity assumption means that the robot is in a singular configuration, and if J has more columns than rows, then the robot is kinematically redundant. These two issues will be discussed in the following subsections.

Example 14.3.4

By direct differentiation of the forward kinematics derived earlier for our example,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} -l_1 \sin(\theta_1) - d_3 \sin(\theta_1 + \theta_2) & -d_3 \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) \\ l_1 \cos(\theta_1) + d_3 \cos(\theta_1 + \theta_2) & d_3 \cos(\theta_1 + \theta_2) & \sin(\theta_1 + \theta_2) \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{d}_3 \end{bmatrix} \tag{14.3.11}$$

Notice that each column of the Jacobian represents the (instantaneous) effect of the corresponding joint on the end effector motions. Thus, considering the third column of the Jacobian, we confirm that the third joint (with variable d_3) cannot cause any change in the orientation (ϕ) of the end effector.

Singularities

A significant issue in kinematic analysis surrounds so-called singular configurations. These are defined to be configurations q_s at which $J(q_s)$ has less than full rank (Spong and Vidyasagar, 1989). Physically, these configurations correspond to situations where the *robot joints* have been aligned in such a way that there is at least one direction of motion (the singular direction[s]) for the end effector that physically cannot be achieved by the mechanism. This occurs at workspace boundaries, and when the axes of two (or more) joints line up and are redundantly contributing to an end effector motion, at the cost of another end effector degree of freedom being lost. It is straightforward to show that the singular direction is orthogonal to the column space of $J(q_s)$.

It can also be shown that every manipulator must have singular configurations, i.e., the existence of singularities cannot be eliminated, even by careful design. Singularities are a serious cause of difficulties in robotic analysis and control. Motions have to be carefully planned in the region of singularities. This is not only because at the singularities themselves there will be an unobtainable motion at the end effector, but also because many real-time motion planning and control algorithms make use of the (inverse of the) manipulator Jacobian. In the region surrounding a *singularity*, the Jacobian will become ill-conditioned, leading to the generation of joint velocities in Equation (14.3.10) which are extremely high, even for relatively small end effector velocities. This can lead to numerical instability, and unexpected wild motions of the arm for small, desired end effector motions (this type of behavior characterizes motion near a singularity).

For the above reasons, the analysis of singularities is an important issue in robotics and continues to be the subject of active research.

Example 14.3.5

For our example manipulator, we can find the singular configurations by taking the determinant of its Jacobian found in the previous section and evaluating the joint configurations that cause this determinant to become zero. A straightforward calculation yields

$$\det(J) = l_1 \cos(\theta_1) \tag{14.3.12}$$

and we note that this determinant is zero exactly when θ_1 is a multiple of $\pi/2$. One such configuration ($\theta_1 = \pi/2$, $\theta_2 = -\pi/2$) is shown in Figure 14.3.3. For this configuration, with $l_1 = 1 = d_3$, the Jacobian is given by

$$\begin{bmatrix} -1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

and by inspection, the columns of J are orthogonal to $[0, -1, 1]^T$, which is therefore a singular direction of the manipulator in this configuration. This implies that from the (singular) configuration shown in Figure 14.3.3, the direction $\dot{Y} = [0, -1, 1]^T$ cannot be physically achieved. This can be confirmed by considering the physical device (motion in the negative y direction cannot be achieved while simultaneously increasing the orientation angle ϕ).

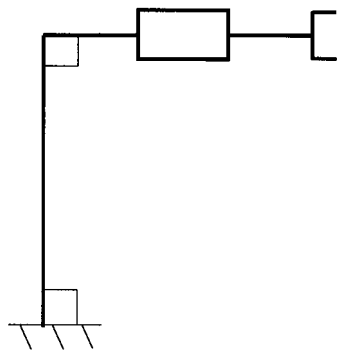


FIGURE 14.3.3 Singular configuration of planar RRP arm.

Redundant Manipulator Kinematics

If the dimension of q is n, the dimension of Y is m, and n is larger than m, then a manipulator is said to be *kinematically redundant* for the task described by Y. This situation occurs for a manipulator with seven or more degrees of freedom when Y is a six-dimensional position/orientation task, or, for example, when a six-degrees-of-freedom manipulator is performing a position task and orientation is not specified.

In this case, the robot mechanism has more degrees of freedom than required by the task. This gives rise to extra complexity in the kinematic analysis due to the extra joints. However, the existence of these extra joints gives rise to the extremely useful *self-motion* property inherent in redundant arms. A self-motion occurs when, with the end effector location held constant, the joints of the manipulator can move (creating an "orbit" of the joints). This allows a much wider variety of configurations (typically an infinite number) for a given end effector location. This added maneuverability is the key feature and advantage of kinematically redundant arms. Note that the human hand/arm has this property. The key question for redundant arms is how to best utilize the self-motion property while still performing specified

end effector motions $\dot{Y}$. A number of motion-planning algorithms have been developed in the last few years for redundant arms (Siciliano, 1990). Most of them center on the Jacobian pseudoinverse as follows.

For kinematically redundant arms, the Jacobian has more columns than rows. If J is of full rank, and we choose $[J^+]$ to be a pseudoinverse of the Jacobian such that $JJ^+ = I$ [for example $J^+ = J^T(JJ^T)^{-1}$], where I is the $m \times m$ identity matrix, then from Equation (14.3.9) a solution for $\dot{q}$ which satisfies end effector velocity of $\dot{Y}$ is given by

$$\dot{q} = \left[J^+(q)\right]\dot{Y} + \left[I - J^+(q)\,J(q)\right]\varepsilon \tag{14.3.13}$$

where ε *is* an $(n \times 1)$ column vector whose values may be arbitrarily selected. Note that conventional nonredundant manipulators have $m = n$, in which case the pseudoinverse becomes J^{-1} and the problem reduces to the resolved rate approach (Equation 14.3.10).

The above solution for $\dot{q}$ has two components. The first component, $[J^+(q)]\,\dot{Y}$, are joint velocities that produce the desired end effector motion $\dot{Y}$ (this can be easily seen by substitution into Equation (14.3.9)). The second term, $[I - J^+(q)J(q)]\varepsilon$, comprises joint velocities which produce no end effector velocities (again, this can be seen by substitution of this term into Equation (14.3.9)). Therefore, the second term produces a self-motion of the arm, which can be tuned by appropriately altering ε. Thus different choices of ε correspond to different choices of the self-motion and various algorithms have been developed to exploit this choice to perform useful subtasks (Siciliano, 1990).

Redundant manipulator analysis has been an active research area in the past few years. A number of arms, such as those recently produced by Robotics Research Corporation, have been designed with seven degrees of freedom to exploit kinematic redundancy. The self-motion in redundant arms can be used to configure the arm to evade obstacles, avoid singularities, minimize effort, and a great many more subtasks in addition to performing the desired main task described by $\dot{Y}$. For a good review of the area, the reader is referred to Siciliano (1990).

Example 14.3.6

If, for our example, we are only concerned with the position of the end effector in the plane, then the arm becomes kinematically redundant. Figure 14.3.4 shows several different (from an infinite number of) configurations for the arm given one end effector position. In this case, J becomes the 2×3 matrix formed by the top two rows of the Jacobian in Equation (14.3.11). The pseudoinverse J^+ will therefore be a 3×2 matrix. Formation of the pseudoinverse is left to the reader as an exercise.

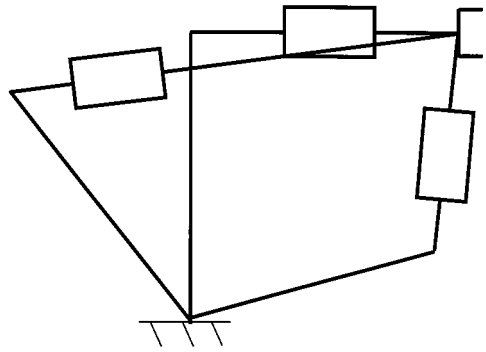


FIGURE 14.3.4 Multiple configurations for RRP arm for specified end effector position only.

Summary

Kinematic analysis is an interesting and important area, a solid understanding of which is required for robot motion planning and control. A number of techniques have been developed and are available to the robotics engineer. For positional analysis, the Denavit-Hartenberg technique provides a systematic approach for forward kinematics. Inverse kinematic solutions typically have been developed on a manipulator (or class of manipulator)-specific basis. However, a number of insightful effective techniques exist for positional inverse kinematic analysis. The manipulator Jacobian is a key tool for analyzing singularities and motion planning at the velocity level. Its use is particularly critical for the emerging generation of kinematically redundant arms

14.4 End Effectors and Tooling

Mark R. Cutkosky and Peter McCormick

End effectors or end-of-arm tools are the devices through which a robot interacts with the world around it, grasping and manipulating parts, inspecting surfaces, and working on them. As such, end effectors are among the most important elements of a robotic application — not "accessories" but an integral component of the overall tooling, fixturing, and sensing strategy. As robots grow more sophisticated and begin to work in more demanding applications, end effector design is becoming increasingly important.

The purpose of this chapter is to introduce some of the main types of end effectors and tooling and to cover issues associated with their design and selection. References are provided for the reader who wishes to go into greater depth on each topic. For those interested in designing their own end effectors, a number of texts including Wright and Cutkosky (1985) provide additional examples.

A Taxonomy of Common End Effectors

Robotic end effectors today include everything from simple two-fingered grippers and vacuum attachments to elaborate multifingered hands. Perhaps the best way to become familiar with end effector design issues is to first review the main end effector types.

Figure 14.4.1 is a taxonomy of common end effectors. It is inspired by an analogous taxonomy of grasps that humans adopt when working with different kinds of objects and in tasks requiring different amounts of precision and strength (Wright and Cutkosky, 1985). The left side includes "passive" grippers that can hold parts, but cannot manipulate them or actively control the grasp force. The right-hand side includes active servo grippers and *dextrous* robot hands found in research laboratories and teleoperated applications.

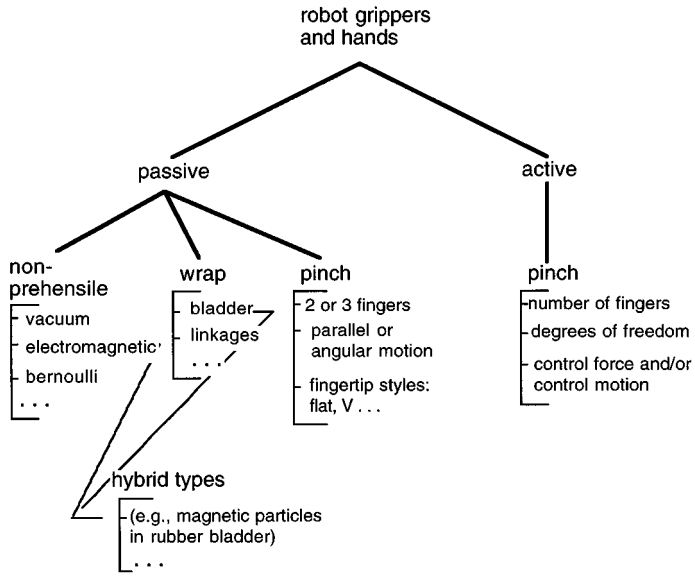


FIGURE 14.4.1 A taxonomy of the basic end effector types.

Passive End Effectors

Most end effectors in use today are passive; they emulate the grasps that people use for holding a heavy object or tool, without manipulating it in the fingers. However, a passive end effector may (and generally should) be equipped with sensors, and the information from these sensors may be used in controlling the robot arm.

The left-most branch of the "passive" side of the taxonomy includes vacuum, electromagnetic, and Bernoulli-effect end effectors. Vacuum grippers, either singly or in combination, are perhaps the most commonly used gripping device in industry today. They are easily adapted to a wide variety of parts — from surface mount microprocessor chips and other small items that require precise placement to large, bulky items such as automobile windshields and aircraft panels. These end effectors are classified as "nonprehensile" because they neither enclose parts nor apply grasp forces across them. Consequently, they are ideal for handling large and delicate items such as glass panels. Unlike grippers with fingers, vacuum grippers to not tend to "center" or relocate parts as they pick them up. As discussed in Table 14.4.1, this feature can be useful when initial part placement is accurate.

TABLE 14.4.1 Task Considerations in End Effector Design

Initial Accuracy. Is the initial accuracy of the part high (as when retrieving a part from a fixture or lathe chuck) or low (as when picking unfixtured components off a conveyor)? In the former case, design the gripper so that it will conform to the part position and orientation (as do the grippers in Figures 14.4.5 and 14.4.6. In the latter case, make the gripper center the part (as will most parallel-jaw grippers).
Final Accuracy. Is the final accuracy of the part high or low? In the former case (as when putting a precisely machined peg into a chamfered hole) the gripper and/or robot arm will need *compliance*. In the latter case, use an end effector that centers the part.
Anticipated Forces. What are the magnitudes of the expected task forces and from what directions will they come? Are these forces resisted directly by the gripper jaws, or indirectly through friction? High forces may lead to the adoption of a "wrap"-type end effector that effectively encircles the part or contacts it at many points.
Other Tasks. Is it useful to add sensing or other tooling at the end effector to reduce cycle time? Is it desirable for the robot to carry multiple parts to minimize cycle time? In such cases consider *compound end effectors*.
Speed and Cycle Time. Are speeds and accelerations large enough that inertial forces and moments should be considered in computing the required grip force?

If difficulties are encountered with a vacuum gripper, it is helpful to remember that problem can be addressed in several ways, including increasing the suction cup area through larger cups or multiple cups, redesigning the parts to be grasped so that they present a smoother surface (perhaps by affixing smooth tape to a surface), and augmenting suction with grasping as discussed below. Figure 14.4.2 shows a large gripper with multiple suction cups for handling thermoplastic auto body panels. This end effector also has pneumatic actuators for providing local left/right and up/down motions.

An interesting noncontact variation on the vacuum end effector is illustrated in Figure 14.4.3. This end effector is designed to lift and transport delicate silicon wafers. It lifts the wafers by blowing gently on them from above so that aerodynamic lift is created via the Bernoulli effect. Thin guides around the periphery of the wafers keep them centered beneath the air source.

The second branch of end effector taxonomy includes "wrap" grippers that hold a part in the same way that a person might hold a heavy hammer or a grapefruit. In such applications, humans use *wrap grasps* in which the fingers envelop a part, and maintain a nearly uniform pressure so that friction is used to maximum advantage. Figures 14.4.4 and 14.4.5 show two kinds of end effectors that achieve a similar effect.

Another approach to handling irregular or soft objects is to augment a vacuum or magnetic gripper with a bladder containing particles or a fluid. When handling ferrous parts, one can employ an electromagnet and iron particles underneath a membrane. Still another approach is to use fingertips filled with an electrorheological fluid that stiffens under the application of an electrostatic field.

The middle branch of the end effector taxonomy includes common two-fingered grippers. These grippers employ a strong "pinch" force between two fingers, in the same way that a person might grasp a key when opening a lock. Most such grippers are sold without fingertips since they are the most product-specific part of the design. The fingertips are designed to match the size of components, the

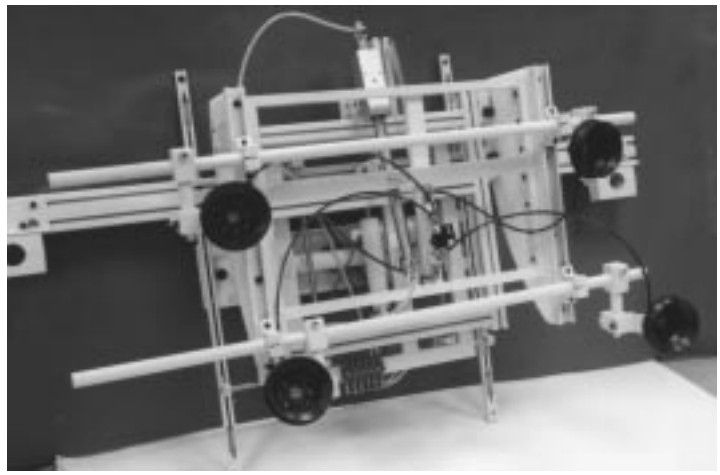


FIGURE 14.4.2 A large end effector for handling autobody panels with actuators for local motions. (Photo courtesy of EOA Systems Inc., Dallas, TX.)

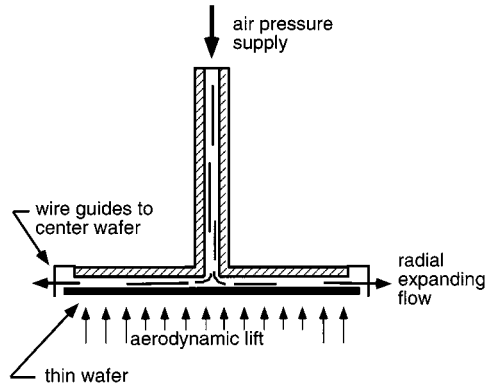


FIGURE 14.4.3 A noncontact end effector for acquiring and transporting delicate wafers.

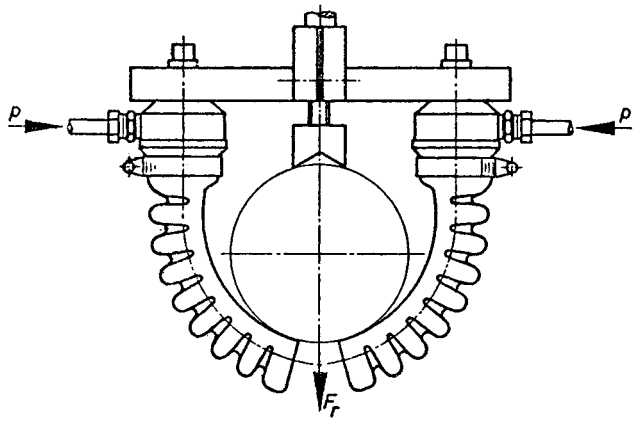


FIGURE 14.4.4 A compliant pneumatic gripper that executes a gentle wrap grasp. (From U.S. Patent No. 3981528, Simrit Corp., Arlington Hts., IL, 1984.)

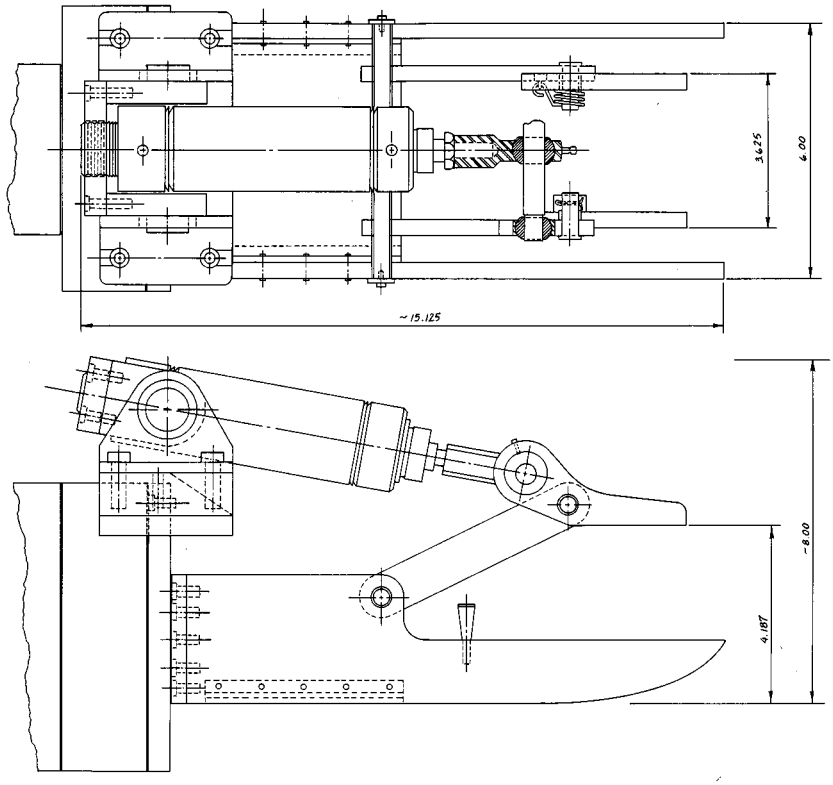


FIGURE 14.4.5 A gripper with pivoted fingers designed to conform to the position and orientation of heavy, irregular parts and to hold them securely. (From U.S. Patent No. 4,545,722, Cutkosky and Kurokawa, 1985.)

shape of components (e.g., flat or V-grooved for cylindrical parts), and the material (e.g., rubber or plastic to avoid damaging fragile objects).

Note that since two-fingered end effectors typically use a single air cylinder or motor that operates both fingers in unison, they will tend to center parts that they grasp. This means that when they grasp constrained parts (e.g., pegs that have been set in holes or parts held in fixtures) some compliance must be added, perhaps with a compliant wrist as discussed in "Wrists and Other End-of-Arm Tooling" below.

Active End Effectors and Hands

The right-hand branch of the taxonomy includes servo grippers and dextrous multifingered hands. Here the distinctions depend largely on the number of fingers and the number of joints or degrees of freedom per finger. For example, the comparatively simple two-fingered servo gripper of Figure 14.4.6 is confined to *"pinch" grasps*, like commercial two-fingered grippers.

Servo-controlled end effectors provide advantages for fine-motion tasks. In comparison to a robot arm, the fingertips are small and light, which means that they can move quickly and precisely. The total range of motion is also small, which permits fine-resolution position and velocity measurements. When equipped with force sensors such as strain gages, the fingers can provide force sensing and control, typically with better accuracy than can be obtained with robot wrist- or joint-mounted sensors. A servo gripper can also be programmed either to control the position of an unconstrained part or to accommodate to the position of a constrained part as discussed in Table 14.4.1.

The sensors of a servo-controlled end effector also provide useful information for robot programming. For example, position sensors can be used to measure the width of a grasped component, thereby providing a check that the correct component has been grasped. Similarly, force sensors are useful for weighing grasped objects and monitoring task-related forces.

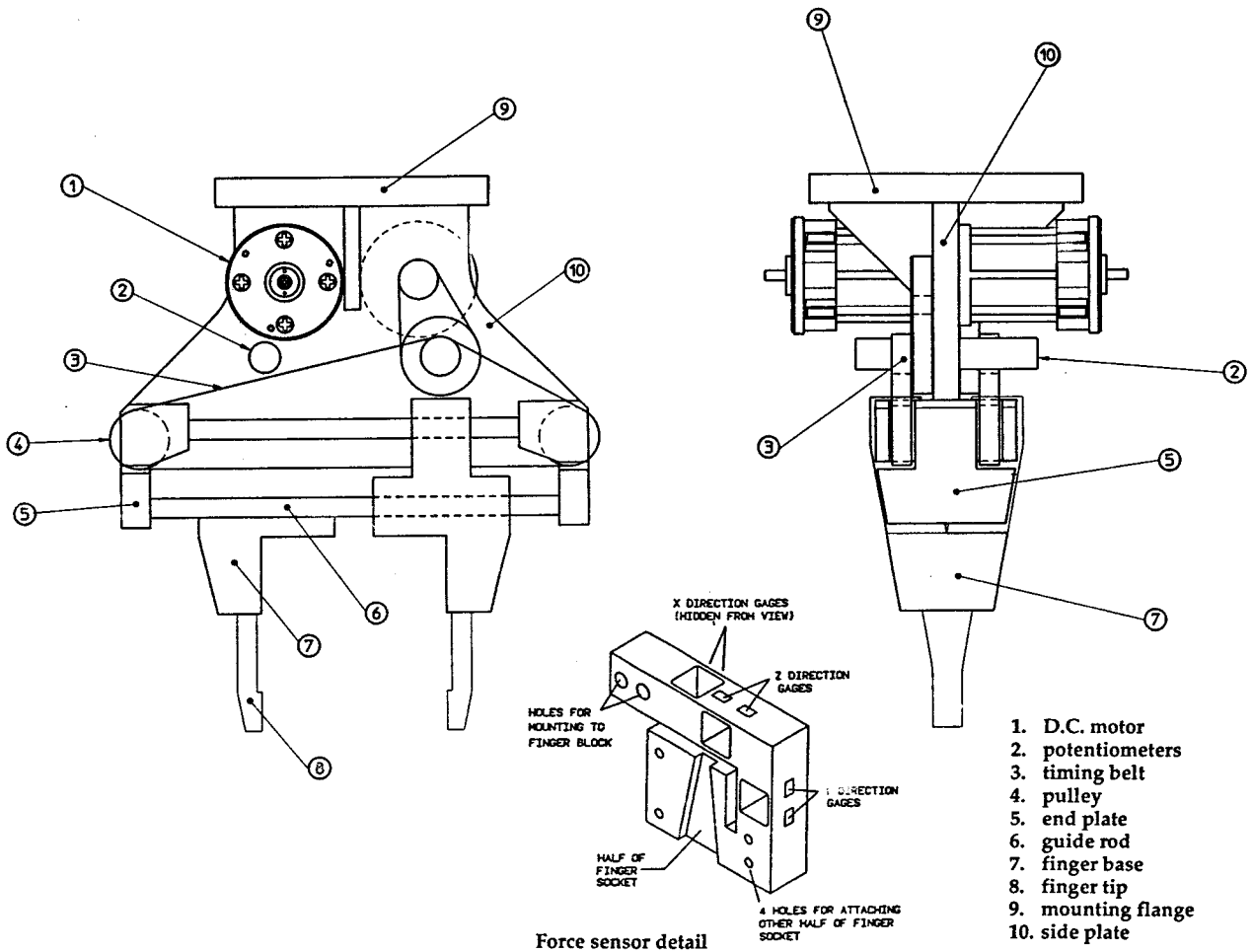


FIGURE 14.4.6 A two-finger servo gripper with force sensing and changeable fingertips. (From E. Pearce et. al, ME210 Report, Stanford University, 1987.)

For applications requiring a combination of dexterity and versatility for grasping a wide range of objects, a dextrous multifingered hand is the ultimate solution. A number of multifingered hands have been described in the literature (see, for example, Jacobsen et al. [1984]) and commercial versions are available. Most of these hands are frankly anthropomorphic, although kinematic criteria such as workspace and *grasp isotropy* (basically a measure of how accurately motions and forces can be controlled in different directions) have also been used.

Despite their practical advantages, dextrous hands have thus far been confined to a few research laboratories. One reason is that the design and control of such hands present numerous difficult tradeoffs among cost, size, power, flexibility and ease of control. For example, the desire to reduce the dimensions of the hand, while providing adequate power, leads to the use of cables that run through the wrist to drive the fingers. These cables bring attendant control problems due to elasticity and friction (Jacobsen et al., 1984).

A second reason for slow progress in applying dextrous hands to manipulation tasks is the formidable challenge of programming and controlling them. The equations associated with several fingertips sliding and rolling on a grasped object are complex — the problem amounts to coordinating several little robots at the end of a robot. In addition, the mechanics of the hand/object system are sensitive to variations in the contact conditions between the fingertips and object (e.g., variations in the object profile and local coefficient of friction). Moreover, during manipulation the fingers are continually making and breaking contact with the object, starting and stopping sliding, etc., with attendant changes in the dynamic and kinematic equations which must be accounted for in controlling the hand. A survey of the dextrous manipulation literature can be found in Pertin-Trocaz (1989).

Wrists and Other End-of-Arm Tooling

In many applications, an active servo gripper is undesirably complicated, fragile, and expensive, and yet it is desirable to obtain some of the compliant force/motion characteristics that an actively controlled gripper can provide. For example, when assembling close-fitting parts, compliance at the end effector can prevent large contact forces from arising due to minor position errors of the robot or manufacturing tolerances in the parts themselves. For such applications a compliant wrist, mounted between the gripper and the robot arm, may be the solution. In particular, *remote center of compliance (RCC)* wrists allow the force/deflection properties of the end effector to be tailored to suit a task. Active wrists have also been developed for use with end effectors for precise, high-bandwidth control of forces and fine motions (Hollis et al., 1988).

Force sensing and quick-change wrists are also commercially available. The former measure the interaction forces between the end effector and the environment and typically come with a dedicated microprocessor for filtering the signals, computing calibration matrices, and communicating with the robot controller. The latter permit end effectors to be automatically engaged or disengaged by the robot and typically include provisions for routing air or hydraulic power as well as electrical signals. They may also contain provisions for overload sensing.

End Effector Design Issues

Good end effector design is in many ways the same as good design of any mechanical device. Foremost, it requires:

- A formal understanding of the functional specifications and relevant constraints. In the authors, experience, most design “failures” occurred not through faulty engineering, but through incompletely articulated requirements and constraints. In other words, the end effector solved the wrong problem.
- A “concurrent engineering” approach in which such issues as ease of maintenance, as well as related problems in fixturing, robot programming, etc., are addressed in parallel with end effector design.

- An attention to details in which issues such as power requirements, impact resistance, and sensor signal routing are not left as an afterthought.

Some of the main considerations are briefly discussed below.

Sensing

Sensors are vital for some manufacturing applications and useful in many others for detecting error conditions. Virtually every end effector design can benefit from the addition of limit switches, proximity sensors, and force overload switches for detecting improperly grasped parts, dropped parts, excessive assembly forces, etc. These binary sensors are inexpensive and easy to connect to most industrial controllers. The next level of sophistication includes analog sensors such as strain gages and thermocouples. For these sensors, a dedicated microprocessor as well as analog instrumentation is typically required to interpret the signals and communicate with the robot controller. The most complex class of sensors includes cameras and tactile arrays. A number of commercial solutions for visual and tactile imaging are available, and may include dedicated microprocessors and software. Although vision systems are usually thought of as separate from end effector design, it is sometimes desirable to build a camera into the end effector; this approach can reduce cycle times because the robot does not have to deposit parts under a separate station for inspecting them.

Actuation

The actuation of industrial end effectors is most commonly pneumatic, due to the availability of compressed air in most applications and the high power-to-weight ratio that can be obtained. The grasp force is controlled by regulating air pressure. The chief drawbacks of pneumatic actuation are the difficulties in achieving precise position control for active hands (due primarily to the compressibility of air) and the need to run air lines down what is otherwise an all-electric robot arm. Electric motors are also common. In these, the grasp force is regulated via the motor current. A variety of drive mechanisms can be employed between the motor or cylinder and the gripper jaws, including worm gears, rack and pinion, toggle linkages, and cams to achieve either uniform grasping forces or a self-locking effect. For a comparison of different actuation technologies, with emphasis on servo-controlled applications, see Hollerbach et al. (1992).

Versatility

Figure 14.4.7 shows a how/why diagram for a hypothetical design problem in which the designer has been asked to redesign an end effector so that it can grasp a wide range of part shapes or types. Designing a versatile end effector or hand might be the most obvious solution, but it is rarely the most economical. A good starting point in such an exercise is to examine the end effector taxonomy in conjunction with the guidelines in Tables 14.4.1 and 14.4.2 to identify promising classes of solutions for the desired range of parts and tasks. The next step is to consider how best to provide the desired range of solutions. Some combination of the following approaches is likely to be effective.

Interchangeable End Effectors. These are perhaps the most common solution for grasping a wider array of part sizes and shapes. The usual approach is to provide a magazine of end effectors and a quick-change wrist so the robot can easily mount and dismount them as required. A similar strategy, and a simpler one if sensory information is to be routed from the end effector down the robot arm, is to provide changeable fingertips for a single end effector.

Compound End Effectors. These are a "Swiss army knife" approach that consists of putting a combination of end effectors on a single arm, or a combination of fingertips on a single end effector. As long as the end effectors or fingertips do not interfere with each other and the ensemble does not weigh too much for the robot arm, this solution combines the advantage of not having to pause to change end effectors with the advantages of custom-designed tooling. Figure 14.4.8 shows a compound end effector with tools for feeding, measuring, cutting, and laying down wires in a cable harness.

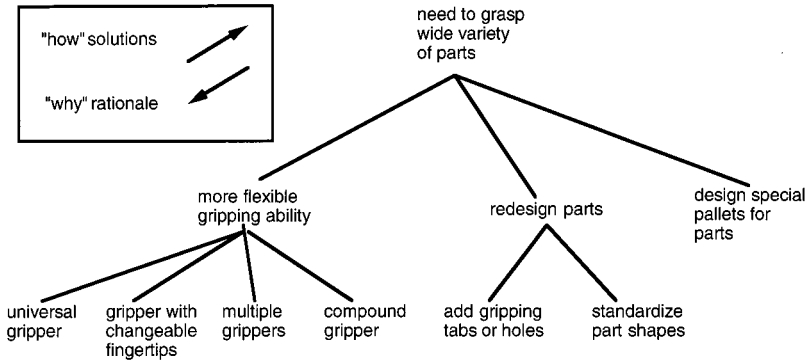


FIGURE 14.4.7 A "how/why" diagram of solutions and rationale for a design problem involving a need to grasp a wide range of parts.

TABLE 14.4.2 Part Characteristics and Associated End Effector Solutions

Size, weight	
Large, heavy	Grippers using wrap grips, taking advantage of friction or *vacuum* or electromagnetic holding
Small, light	Two-fingered gripper; vacuum cup if smooth surface, electromagnet if ferrous alloy
Shape	
Prismatic	Two-fingered parallel-jaw gripper; angular motion if all parts have approximately same dimensions
Cylindrical	Parallel or angular motion two-finger gripper with V-jaw fingertips if light; wrap gripper if heavy; consider gripping on end with three-finger gripper if task or fixtures permit
Flat	Parallel or angular motion gripper or vacuum attachment
Irregular	Wrap grasp using linkages or bladder; consider augmenting grasp with vacuum or electromagnetic holding for heavy parts
Surface	
Smooth	Good for vacuum attachments, simple electromagnets, two-fingered grippers with flat fingertips
Rough	Compliant material (e.g., low durometer rubber) on fingertips or compliant membrane filled with powder or magnetic particles; grippers that use a wrap grasp are less sensitive to variations in surface quality
Slippery	Consider electromagnet or vacuum to help hold onto slippery material; grippers that use a wrap grasp are less sensitive to variations in friction
Material	
Ferrous	Electromagnet (provided that other concerns do not rule out the presence of strong magnetic fields)
Soft	Consider vacuum or soft gripping materials
Very delicate	Soft wrap grippers and vacuum grippers such as those in Figure 14.4.4 can grip very gently; compliant fingertips with foam rubber, or a membrane covering a powder, can also be used to distribute the contact pressure; if the part is very light and fragile consider lifting it using the Bernoulli effect

Redesigned Parts and Fixtures. Stepping back from the end effector, it is useful to recall that the design of the end effector is coupled with the design of fixtures, parts, and the robot. Perhaps we can design special pallets or adapters for the parts that make them simpler to grasp. Another solution is to standardize the design of the parts, using Group Technology principles to reduce the variability in sizes and geometries. When it is difficult to reduce the range of parts to a few standard families (or when the parts are simply hard to grip), consider adding special nonfunctional features such as tabs or handles so that a simple end effector can work with them.

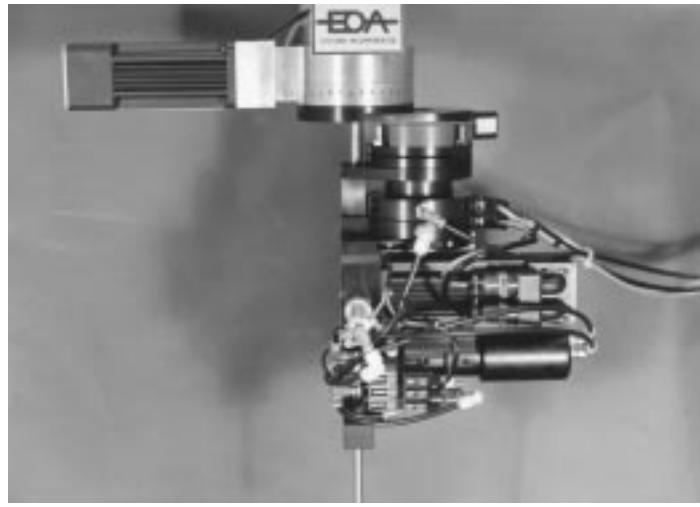


FIGURE 14.4.8 A "compound" end effector with tools for feeding, measuring, cutting, and laying down wires in a cable harness. (Photo courtesy of EOA Systems Inc., Dallas, TX.)

Summary

In summary, we observe that end effector design and selection are inextricably coupled with the design of parts, robots, fixtures, and tooling. While this interdependence complicates end effector design, it also provides opportunities because difficult problems involving geometry, sensing, or task-related forces can be tackled on all of these fronts.

14.5 Sensors and Actuators

Kok-Meng Lee

Sensors and actuators play an important role in robotic manipulation and its applications. They must operate precisely and function reliably as they directly influence the performance of the robot operation. A transducer, a sensor or actuator, like most devices, is described by a number of characteristics and distinctive features. In this section, we describe in detail the different sensing and actuation methods for robotic applications, the operating principle describing the energy conversion, and various significant designs that incorporate these methods. This section is divided into four subsections, namely, tactile and proximity sensors, force sensors, vision, and actuators.

By definition, tactile sensing is the continuously variable sensing of forces and force gradients over an area. This task is usually performed by an $m \times n$ array of industrial sensors called forcels. By considering the outputs from all of the individual forcels, it is possible to construct a tactile image of the targeted object. This ability is a form of sensory feedback which is important in development of robots. These robots will incorporate tactile sensing pads in their end effectors. By using the tactile image of the grasped object, it will be possible to determine such factors as the presence, size, shape, texture, and thermal conductivity of the grasped object. The location and orientation of the object as well as reaction forces and moments could also be detected. Finally, the tactile image could be used to detect the onset of part slipping. Much of the tactile sensor data processing is parallel with that of the vision sensing. Recognition of contacting objects by extracting and classifying features in the tactile image has been a primary goal. Thus, the description of tactile sensor in the following subsection will be focused on transduction methods and their relative advantages and disadvantages.

Proximity sensing, on the other hand, is the detection of approach to a workplace or obstacle prior to touching. Proximity sensing is required for really competent general-purpose robots. Even in a highly structured environment where object location is presumably known, accidental collision may occur, and foreign object could intrude. Avoidance of damaging collision is imperative. However, even if the environment is structured as planned, it is often necessary to slow a working manipulator from a high slew rate to a slow approach just prior to touch. Since workpiece position accuracy always has some tolerance, proximity sensing is still useful.

Many robotic processes require sensors to transduce contact force information for use in loop closure and data gathering functions. Contact sensors, wrist force/torque sensors, and force probes are used in many applications such as grasping, assembly, and part inspection. Unlike tactile sensing which measures pressure over a relatively large area, force sensing measures action applied to a spot. Tactile sensing concerns extracting features of the object being touched, whereas quantitative measurement is of particular interest in force sensing. However, many transduction methods for tactile sensing are appropriate for force sensing.

In the last three decades, computer vision has been extensively studied in many application areas which include character recognition, medical diagnosis, target detection, and remote sensing. The capabilities of commercial vision systems for robotic applications, however, are still limited. One reason for this slow progress is that robotic tasks often require sophisticated vision interpretation, yet demand low cost and high speed, accuracy, reliability, and flexibility. Factors limiting the commercially available computer vision techniques and methods to facilitate vision applications in robotics are highlights of the subsection on vision.

Tactile and Proximity Sensors

A review of past investigations (see Nichols and Lee [1989] for details) has shown that a tactile sensor should have the following characteristics: most important, the sensor surface should be both compliant and durable, and the response of individual forcels should be stable, repeatable, free from hysteresis. The response must be monotonic, though not necessarily linear. The forcels should be capable of detecting

TABLE 14.5.1 Advantages and Disadvantages of Different Tactile Transduction Methods

Type	Advantages	Disadvantages
Resistive and conductive	Wide dynamic range Durability Good overload tolerance Compatibility with integrated circuitry	Hysteresis in some designs Limited spatial resolution Monotonic response, but often not linear
Capacitive	Wide dynamic range Linear response Robust	Susceptible to noise Temperature-sensitive Limiting spatial resolution
Magnetoelastic	Wide dynamic range Low hysteresis Linear response Robust	Susceptibility to stray fields and noise as circuitry requires
Optical	Very high resolution Compatible with vision technology No electrical interference problems	Some hysteresis, depends on elastomer in some designs
Piezoelectric and pyroelectric	Wide dynamic range Durability Good mechanical properties Capable of temperature as well as force sensing	Difficult to separate piezoelectric from pyroelectric effects Inherently dynamic
Thermal	Combined force and temperature	Slow in response

loads ranging from 0 to 1000 g, having a 1-g sensitivity, a dynamic range of 1000:1, and a bandwidth of approximately 100 Hz. Furthermore, forcers should be spaced no more than 2 mm apart and on at least a 10 × 10 grid. A wide range of transduction techniques have been used in the designs of the present generation of tactile sensors. These techniques are compared in Table 14.5.1 and the principles of transduction methods are described as follows.

Resistive and Conductive Transduction

This technique involves measuring the resistance either through or across the thickness of a conductive elastomer. As illustrated in Figure 14.5.1, the measured resistance changes with the amount of force applied to the materials, resulting from the deformation of the elastomer altering the particle density within it. Most commonly used elastomers are made from carbon or silicon-doped rubber, and the construction is such that the sensor is made up of a grid of discrete sites at which the resistance is measured.

A number of the conductive and resistive designs have been quite successful. A design using carbon-loaded rubber originated by Purbrick at MIT formed the basis for several later designs. It was constructed

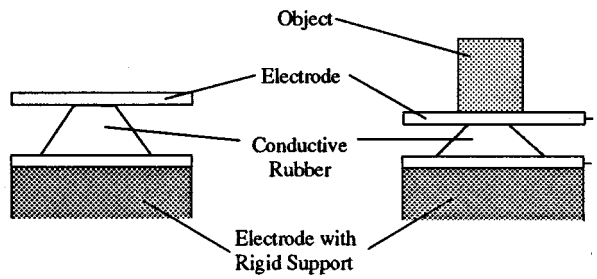


FIGURE 14.5.1 Resistive tactile element.

from a simple grid of silicon rubber conductors. Resistance at the electrodes was measured, which corresponds to loads. A novel variation of this design developed by Raibeit is to place the conductive sheet rubber over a printed circuit board (PCB) which incorporates VLSI circuitry, each forcel not only transduces its data but processes it as well. Each site performs transduction and processing operations at the same time as all the others. The computer is thus a parallel processor.

Capacitive Transduction

Capacitive tactile sensors are concerned with measuring capacitance, which is made to vary under applied load. A common sensor design is to use an elastomeric separator between the plates to provide compliance such that the capacitance will vary according to applied load. The capacitance of a parallel plate capacitor is proportional to its congruous area and the permitivity of dielectric, and inversely proportional to the separation of the plates. Alteration of any of the three parameters causes a change of capacitance. Since the capacitance decreases with decreasing congruous area, the sensor becomes rather cumbersome for design of small forcels.

To allow for a more compact design, an alternative tactile sensor array can be designed based on a moving dielectric element as illustrated in Figure 14.5.2. Each sensing element has two coaxial capacitor cylinders, acting as plates, fixed to a PCB. A dielectric element is spring-mounted in the space between the cylinders. The dielectric is displaced by contact with an external stimulus; hence it moves up and down between the capacitor plates as contact loads vary. A force-displacement relationship is thereby established.

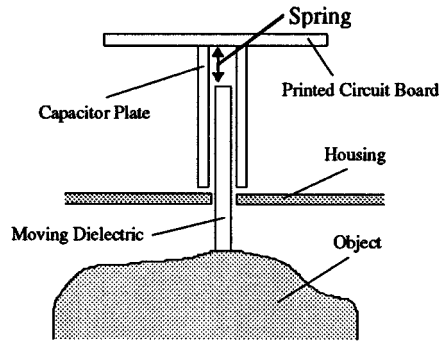


FIGURE 14.5.2 Mechanical/capacitive tactile element.

A novel slip sensor using the change in capacitance caused by relative contact movement between sensor and object is described by Luo (Nichols and Lee, 1989). The contacting sensor surface comprises a set of parallel rollers. Each roller is a half cylinder of conductive material, and a half cylinder of nonconductive material. The rollers are mounted in a nonconductive material.

The casing and rollers act as a variable capacitor. A slipping object will rotate the rollers, causing the capacitance to change, which is then measured, thereby facilitating a slip sensor. The sensor measures the change of phase angle, with the amount of phase shift providing a measure of the scale of slip. A highly linear relationship between detected phase shift angle and sensor output was established.

Magnetoelastic Transduction

Magnetoelastic sensors are a kind of inductive sensor that differs from those described above; they are not based on a change of geometry or on the position of conductive or capacitive materials. Instead, they are based on the Villari effect, consisting of reversible changes in the magnetization curve of a ferromagnetic material when it is subjected to a mechanical stress. It consists of changes of shape and volume during the magnetization process. Magnetoelastic materials undergo changes in their magnetic field when subjected to stress and therefore suggest themselves as possible transducers in tactile sensors.

Figure 14.5.3 illustrates this transduction principle in tactile sensor design. This method of tactile transduction has seen little development in robotics, although there are several papers on the subject (Fraden, 1993).

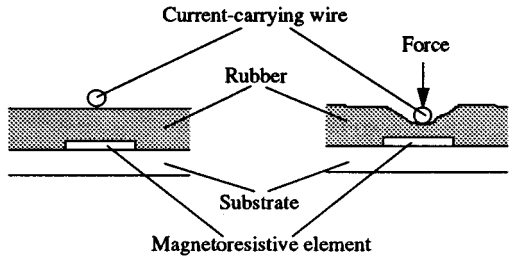


FIGURE 14.5.3 Magnetoresistive tactile element.

Fiber Optics Proximity and Tactile Sensors

The development of optical fiber technology and solid-state cameras has led to some interesting new tactile sensor designs. The capability for high-spatial-resolution images, freedom from electrical interference, and ease of separation of sensor from processing electronics are some of the attractions of incorporating optical transduction methods into tactile sensors. The following illustrates two different fiber optic sensor designs, a proximity sensor and a tactile sensor.

Figure 14.5.4 illustrates the basic principle of fiber optic proximity sensor. Light from a light-emitting diode (LED) is passed down a fiber optic cable to illuminate any proximal objects. A second cable picks up any reflected light from illuminated objects within a detection zone and directs it onto a photodiode. This simple technique can be built into a finger. The finger can sense contacts perpendicular to the finger axis, radially, and also axial contact at the fingertip. Several fiber optic cable pairs can be evenly spaced around the fingers and incorporated into a gripping system. Figure 14.5.4 illustrates this transduction method for proximity sensing.

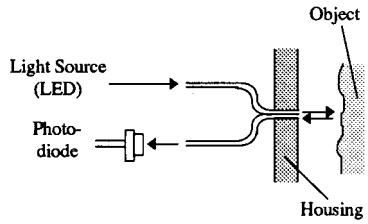


FIGURE 14.5.4 Optical proximity sensing.

Optical fibers are a type of dielectric waveguide. These waveguides channel light energy by "trapping" it between cylindrical layers of dielectric materials. In the most simple case, the fiber core is surrounded by a cladding which has a small refractive index. Light is lost from the core of a fiber when a mechanical bend or perturbation results in coupling between guided and radiation modes. The concept of monitoring light losses due to microbending can be found in several tactile sensor designs (Nichols and Lee, 1989; Tzou and Fukuda, 1992).

Piezoelectric/Pyroelectric Effect

The piezoelectric effect is the generation of a voltage across the sensing element when pressure is applied to it. Correspondingly, the pyroelectric effect is the generation of a voltage when the sensing element is heated or cooled. No external voltage is required, and a continuous analog output is available from such a sensor. Such sensors are most suited for sensing pressure changes or thermal variations. Figure 14.5.5 shows a design based on the piezoelectric effect for robotic applications (Fraden, 1993). The

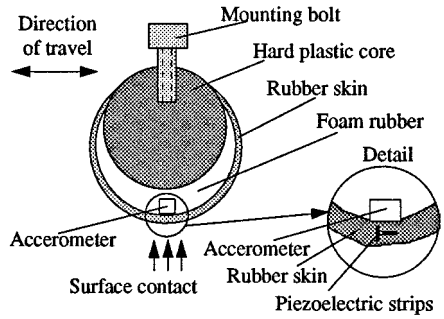


FIGURE 14.5.5 Schematic of piezoelectric sensor for a soft fingertip.

sensor includes piezoelectric strips directly interfaced with a rubber skin; thus the electric signal produced by the strips reflects movements of the elastic rubber which results from the friction forces.

Thermal Tactile Sensors

The thermal sensor (Nichols and Lee, 1989) is based on the detection of the change of thermal properties through contact of an object. The main function of the thermal sensor is to provide information about the material makeup of objects. The essential parts of each element of the thermal sensor are a heat source (such as a power transistor), a layer of material of known thermal conductivity (for example, copper) to couple the heat source to the touched object, and a temperature transducer (thermistor) to measure the contact-point temperature. The response time of the thermal sensor is relatively slow, typically in the order of several seconds. However, images representing the material constitution of the touching objects provide useful tactile data.

Force Sensors

Force sensors measure the force and represent its value in terms of an electrical signal. Examples of these sensors are strain gauges and load cells.

Strain Gauge-Based Force Sensor

A strain gauge is a resistive elastic sensor whose resistance is a function of applied strain or unit deformation. The relationship between the normalized incremental resistance and the strain is generally known as the piezoresistive effect. For metallic wire, the piezoresistance ranges from 2 to 6. For semiconductor gauges, it is between 40 and 200. Many metals can be used to fabricate strain gauges. Typical resistances vary from 100 to several thousand ohms. Strain gauges may be arranged in many ways to measure strains and are used typically with Wheatstone bridge circuits. As strain gauges are often sensitive to temperature variations, interfacing circuits or gauges must contain temperature-compensating networks.

Strain gauges are commonly used for six-degrees-of-freedom force/torque wrist sensors, force probes, flexural assemblies for force control, and micromotion detection. The Scheinman force-sensing wrist is a Maltese cross design, with one strain guage mounted on each of the 16 faces of the cross-webbings. The gauges are operated in eight voltage-divider pairs to measure distortions, and therefore forces, in six degrees of freedom in the hand coordinate system.

Other Force Sensors

Other methods include the vacuum diode force sensor, quartz force sensor, and piezoelectric force sensor. A piezoelectric sensor converts mechanical stress into an electric signal (Fraden, 1993). It is sensitive to changing stimuli only and insensitive to a constant force. As shown in Figure 14.5.6, the sensor consists of three layers where the PVDF film is laminated between a backing material (for example, silicon rubber) and a plastic film. When the PVDF is stressed, it results in a generation of electric charge

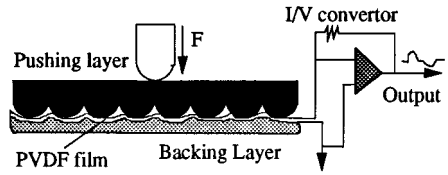


FIGURE 14.5.6 Piezoelectric force rate sensor.

flowing out of the film through a current-to-voltage (I/V) converter. The resulting output voltage is proportional to the applied force.

Figure 14.5.7 shows a typical structure fabricated by micromachining technology in a silicon wafer. As shown in the figure, the diode sensor has a cold field emission cathode, which is a sharp silicon tip, and a movable diaphragm anode. When a positive potential difference is applied between the tip and the anode, an electric field is generated which allows electrons to tunnel from inside the cathode to the vacuum. The field strength at the tip and quantity of electrons emitted (emission current) are controlled by the anode potential. When an external force is applied, the anode deflects and changes the field and the emission current.

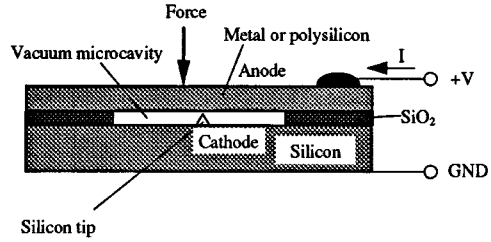


FIGURE 14.5.7 Schematic of a vacuum diode force sensor.

Figure 14.5.8 shows a quartz crystal force sensor. A quartz crystal is often used as a resonator in electrical oscillators. The basic idea behind the quartz force sensor's operation is that certain cuts of quartz crystal shift the resonant frequency when mechanically loaded.

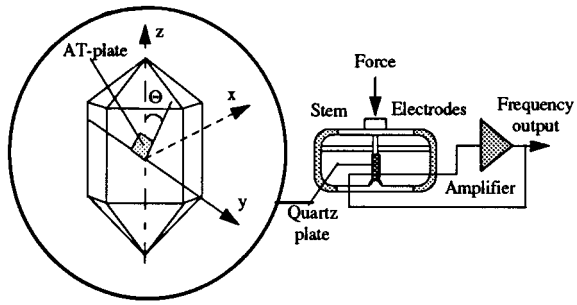


FIGURE 14.5.8 Quartz force sensor.

Vision

Many industrial tasks require sophisticated vision interpretation, yet demand low cost, high speed, accuracy, and flexibility. To be fully effective, machine vision systems must be able to handle complex industrial parts. This includes verifying or recognizing incoming parts and determining the location and orientation of the part within a short cycle time. Typical video-based vision systems conform to the RS-170 standard established in the 1950s, which defines the composite video and synchronizing signal that

the television industry uses. It specifies a standard frame rate for visual interpretation. The components required for building a video-based vision system generally include a video camera which outputs standard RS170 video signal, a frame grabber board which uses a flash analog-to-digital (A/D) converter to change the RS170 video signal into a series of n bit brightness values (gray levels) and fast memory components to store them, and a microcomputer which processes the images and computes the location and orientation of the part. See Ballard and Brown (1982) for information on vision processing techniques.

In addition to the error resulting from the timing mismatching between image acquisition hardware and the computer hardware, the RS170 video signal limits the readout of a complete frame at a rate of 30 fps (frames per second). An image of m rows by n columns has m × n *pixels* and so requires a substantial amount of memory and loading time. Among these m × n pixels, only a few carry the information on which a vision system will base a decision. This generally makes "frame grabbing" inherently wasteful.

Apart from the lack of appropriate hardware and the high equipment cost for robotic applications, a major problem often associated with the use of the RS170 video vision system is the excessive image processing time which depends on the illumination technique, the complexity of the geometry, and the surface reflectance of both the background and the objects to be handled.

Flexible Integrated Vision System

To overcome these problems, several vision systems were designed for robotic applications. Among these is a Flexible Integrated Vision System (FIVS) developed at Georgia Tech (Lee and Blenis, 1994), which offers performance and cost advantages by integrating the imaging sensor, control, illumination, direct digitization, computation, and data communication in a single unit. By eliminating the host computer and frame grabber, the camera is no longer restricted by the RS-170 standard and thus frame rate higher than 30 fps can be achieved.

Flexible Integrated Vision System Hardware. As shown in Figure 14.5.9, the central control unit of the flexible integrated vision system is a microprocessor-based control board. The design is to have all of the real-time processing performed using the microprocessor control board without relying on any other system or computer. Thus, it is desired to have the following features: (1) the microprocessor has an on-chip program memory and independent on-chip data memories. These memories must be externally expandable and accessible with zero wait states; (2) it has independent execution units which are connected by independent buses to the on-chip memory blocks. This feature provides the parallelism needed for high performance digital signal processing and high-powered computation of mathematically intensive algorithms. For these reasons, a digital signal processor (DSP) chip has been chosen.

The DSP-based control board is designed to communicate with several option boards in parallel to tailor the system for a number of applications. Each of these option boards is controlled independently by a programmable logic device (PLD) which receives a peripheral select signal, a read/write signal, and an address signal from the microprocessor control board. Typical examples of the option boards for the FIVS are the digital video head, a real-time video record/display/playback board, and an expandable memory board.

The video head consists of a m × n CCD array, the output of which is conditioned by high bandwidth amplification circuitry. The output is then sampled by a "flash" analog-to-digital converter (ADC). The DSP-based control board provides a direct software control of CCD array scanning and integration time, the intensity of the collocated illumination, and the real-time execution of a user-selectable vision algorithm imbedded in the EEPROM. In operation, the PLD decodes the control signals to initiate row shifts and column shifts in response to commands from the DSP-based control board. Particular row shifts and column shifts enable retrieving only a specific relevant area from an image. The PLD also provides control signals to ADC for performing the analog-to-digital conversion synchronized with row shifts, and enables the video buffer when the DSP reads or writes data to the VRAM.

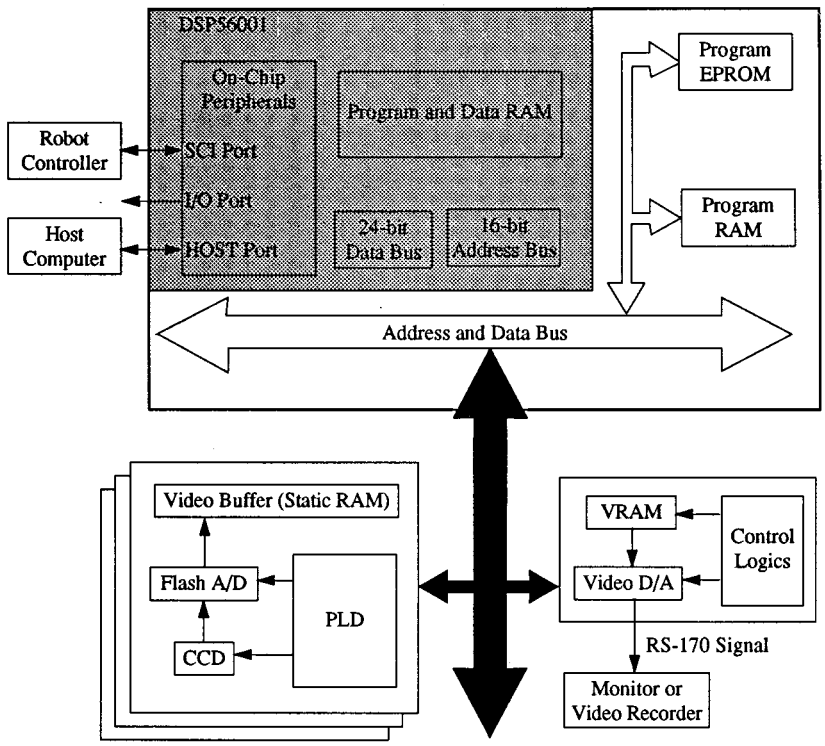


FIGURE 14.5.9 Schematic of a flexible integrated vision system.

Unlike conventional RS170-based systems which require pixel data to be stored in a video buffer before processing of pixel data can commence, the FIVS design provides an option to completely by-pass the video buffer and thus offers a means to process and/or to store the digitized pixel data by directly transferring the ADC output to the DSP. For real-time vision-based object tracking and motion control system applications, the scheme represents a significant saving in time and video buffer size required for processing an image. As an illustration, consider an image array of m × n pixels. The time needed to store the entire image (with no computation) in a memory at K MHz is (m × n)/K s and requires (m × n) bytes of memory. Typical array size of a CCD ranges from 200 × 160 to 4096 × 4096 of pixels. The corresponding video buffer and time required simply to store the entire image at a clock rate of 10 MHz would range from 32K btyes to 16 Mbytes and 3.2 to 1600 msec, respectively! Clearly, the option to completely bypass the video buffer offers a potentially usefu) solution to eliminate the frame storage prerequisite which is often required in conventional vision systems. Furthermore, this scheme completely eliminates the special hardware needed in acquiring the digitized pixel data for storage.

Flexible Integrated Vision System Imbedded Software. The vision system imbedded software includes the following functions. The first function is to give users the flexibility to control the CCD array scanning, integration time, and the intensity of the illumination. With the CCD under software control, partial frames can be "captured" instead of the customary full frame, reducing the cycle time required to capture and process an image. The ability to shift out partial frames is ideal for high-speed tracking applications where the approximate location is known from a prior image. By reducing the time to capture an image, the effective frame rate is increased. For example, shifting out 1/4 of an image can increase the frame rate up to 480 fps, not including the time required for illumination and image processing. This frame rate is 16 times the rate achievable from the RS-170 standard.

The second function is to offer an option to process the pixel data from the ADC directly without having to store the pixel data prior to processing. Although windowing process methods have been

suggested to perform object tracking under software control, these methods required that a partial window is stored before scanning can begin. The differences between the direct computation and the windowing process for object tracking are as follows: (1) in windowing process, the entire image must be stored and analyzed at least once before any subsequent windowing process can be performed in order to provide a reasonable estimate of the object location. Furthermore, if the initial field of view does not contain the object, this estimate must be repeated until an approximate area containing the object can be reasonably found. This prerequisite of storing the image is not necessary if the pixel data are directly processed; (2) after the initial estimate, a fixed window which must be sufficiently large in order to include the object in the field of view must be specified in the windowing process. In most conventional systems which output their ADC to the video buffer directly, a partial frame of the image as specified by the window must be stored. By providing a direct transfer the ADC output to the DSP and thus eliminating the windowing storing process, a significant fraction of time can be saved. This function provides an attractive feature to vision-based motion control applications.

The third function allows image processing to be performed in real time without a host computer. The algorithm that allows the user to customize the system for a specified task is preprogrammed in the EEPROM (electrically erasable programmable read only memory). Because it is impractical to preprogram every possible vision processing algorithm into the FIVS camera, it is desirable that the system can be reprogrammed easily. The main kernel provides a user interface whereby the user can customize the real-time processing for a particular task, from a library of algorithms. This function also provides an effective means to resolve software implementation issues prior to an on-line application. By previewing images of a sample part, the user may select an appropriate vision algorithm for an accurate computation of the location and orientation in real time. Once the algorithms and data are downloaded into the on-board EEPROM, the FIVS can function as an intelligent sensor and communicate directly with the robot controller without a host computer.

The fourth function, which incorporates a real-time display, allows the process controller to set up, to calibrate the vision system, or to analyze a failure mode (if any).

Illumination Considerations

Imaging sensors are characterized by their specific bandwidths or wavelengths of light which maximize the response of the sensor and will provide it an optimum operating environment. It is desired that the photodetector responds only to the light from the illumination source structured for the object but not that of ambient lighting. Otherwise, software compensation must be considered. To accomplish the objective, a typical sensor/illumination system design must consider the spectral matching of the camera imaging sensor/filter and a spectral illuminator while minimizing the effect of the ambient lighting.

Spectral Responsivity of Sensor. The two most commonly used camera imaging sensors are the charge-coupled device (CCD) and the charge injection device (CID). The CCD is responsive to wavelengths of light from below 350 nm (ultraviolet) to 1100 nm (near infrared) and has a peak response approximately at 800 nm. The CID offers a similar spectral response and has a peak spectral response about 650 nm. The relative response of a vidicon camera, however, depends significantly on the materials.

Spectral Characteristic of Typical Ambient Lighting. Depending on the spectral emissions of illumination sources used as general lighting in the factory environment, the influences of the ambient lighting can be effectively minimized or eliminated by means of spectral filtering. Gas discharge lamps generally have relatively high emission in the visible range and have little or no emission for wavelengths larger than 800 nm. Sun, tungsten lamps, and quartz-halogen-type lamps have a wide spectral emission.

Illumination Source. The spectral characteristics of three different spectral sources, namely, laser diodes, light-emitting diode (LED), and xenon strobes, are of particular interest since the spectral wavelengths of these sources well match the optimal response of the CCD and/or CID detectors. Pulsed GaAlAs laser diodes emit single frequency power in the 790- to 850-nm wavelength range. Irradiance at spectral wavelength in the range of 810 to 830 nm can also be produced from a xenon lamp. An

TABLE 14.5.2 Comparison Between Three Spectral Light Sources

Source	Wavelength (nm)	Unit cost (US $)	Life	Power
LED	570–630	1.00	5,000,000 hours (MTBF)	100 mW
Laser diode	790–840	200.00	250,000 hours (MTTF)	1 W (peak pulse power)
Xenon flashtubes	830–1000	10.00	1,000,000 flashes (0.3–4 flashes/sec)	25 W (500 V nominal)

AlGaAs LED is designed to concentrate the luminous flux into a narrow radiation pattern to achieve a narrow high peak intensity. A comparison of these sources is provided in Table 14.5.2.

Object and Background Reflectance. If the orientation of the parts can be characterized by the two-dimensional object silhouette and the environment can be structured, back lighting can be used to create silhouettes of the object. Alternatively, retroreflective materials (Lee and Blenis, 1994) can be used to create a unique background. Since most of the incident illuminance from the object is reflected or diffused away from the aperture, whereas that on the background surface is retroreflected, the object appears as a dark silhouette against a reliable bright-field background.

Retroreflective materials can be used as background in part presentation or as a landmark on parts. The choice clearly depends on the part design and manufacturing process. The most common retroreflective surface is in the form of sheeting due to its reliability and ease of application. Flexible retroreflective sheeting is made of countless microcube-corners or spheres enclosed in a weather-resistant transparent plastic film. Pigment or dye can be inserted into the film or the reflecting surface to reflect color. Four typical retroreflective sheetings are described as follows: (1) cube-corner retroreflective sheeting, (2) exposed glass beads, (3) enclosed glass beads, and 4) encapsulated glass beads. A detailed study of retroreflective sensing for robotic applications is given by Lee and Li (Lee and Blenis, 1994).

Vision Algorithms for Robotic Applications

Figure 14.5.10 illustrates a vision system for robotic part pickup applications (see also Section 14.9). Here the camera is mounted along with the gripper on the end effector mount of the robot. This allows complete freedom in positioning and orienting the camera for viewing. Placing the camera on the last link of a six-DOF robot enables the machine vision to view objects (parts) individually. The camera is oriented so that its line of sight is perpendicular to the plane on which the part is placed. However, at each position, the 3D position and orientation of the feature measured by the vision system are only relative to the vision sensor. The robot is driven by sensory information from the vision system as well as inputs from the off-line calibration.

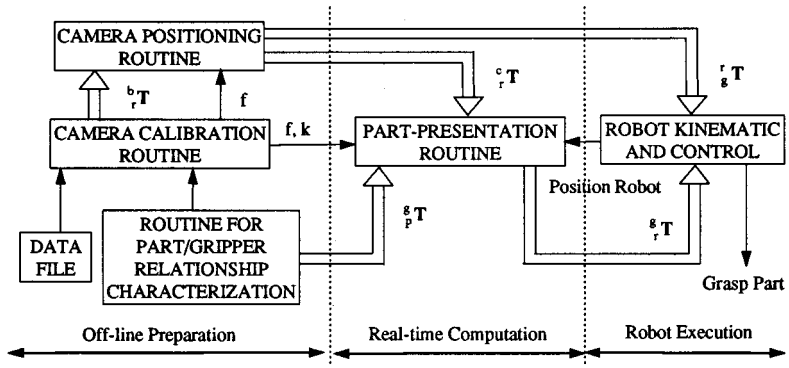


FIGURE 14.5.10 Vision system for robotic applications.

Thus, the basic components of the vision system for robotic part pickup include (1) system calibration, (2) image segmentation and feature extraction, and (3) communication protocol.

System Calibration. In order to determine the 3D position and orientation of the feature with respect to robot world coordinate, it is necessary to calibrate the relative homogeneous transformation between the two coordinate frames, one centered at the camera and the other at the gripper. Thus, the system calibration is to establish the relationship between the 3D world coordinates as seen by the robot and their corresponding 2D image coordinates as seen by the computer. The calibration algorithm consists of off-line calibrating the intrinsic parameters of the camera and the camera-gripper relationship and on-line calibrating the pallet location. The camera calibration technique originally established by Tsai and Lenz (1989) has been the basis for several later calibration routines.

Image Segmentation and Feature Extraction. Most existing industrial-vision systems and algorithms extract features from industrial objects against a high contrast background with controlled lighting. The processing of feature extraction usually begins by generating a binary image from the original gray-scale image by choosing an appropriate threshold. To eliminate noise caused by electromagnetic interference, and ignoring the other objects in the field of view, image segmentation is performed before the computation of the part location and orientation. An image segmentation algorithm is written to locate regions of pixels that are connected and to label each region (object) so that it can easily be picked out from the other regions in the image. After segmentation is complete, only the largest object in the image is examined for its features.

There are many practical methods for the identification of a given object in a scene. A part-recognition system consists of three major components, namely, feature extraction, object modeling, and matching. Most industrial parts-recognition systems are model-based systems in which recognition involves matching the input image with a set of predefined models of part. Models based on geometric properties of an object's visible surfaces or silhouette are commonly used because they describe objects in terms of their constituent shape features. Image features such as edge, corner, line, curve, hole, and boundary curvature define individual feature components of an image. Given a set of models that describes all aspects of all parts to be recognized, the process of model-based recognition consists of matching features extracted from a given input image with those of the models. There are many practical methods for identification of a given object in a scene. A tutorial on binary image processing for robot-vision applications is given by Kitchin and Pugh (1983). A more general comparative study of model-based object-recognition algorithms for robot vision is described by Chin and Dyer (1986).

Communication Protocol. To ensure data integrity during communications, DEC's Digital Data Communications Message Protocol (DDCMP) is used for communications with the vision system. DDCMP is an industrial standard communication protocol that is used to communicate with industrial robots. DDCMP ensures a reliable communications link with a minimum amount of overhead. DDCMP precedes all data transmissions with a header block describing the type of message and the length of any message data that follows.

Actuators

Actuators used for robotic manipulators can be broadly classified as follows: (1) electromechanical actuators, (2) fluid power actuators, and (3) new alternative actuators. Table 14.5.3 summarizes a further subdivision based on their operating principles. The choice of actuators for robotic applications depends on specific tasks. Relative comparisons to guide selection of common actuators are given in Table 14.5.4 and Figure 14.5.11, which shows the force vs. speed comparison for common actuators.

Direct-Drive Joint Motor

The direct-drive joint motor has been developed to eliminate the transmission mechanism between the motor and the links, thus eliminating friction and backlash introduced by gear motors. This results in an arm suitable for high-speed, fine torque control. A direct, drive design also allows for elegant

TABLE 14.5.3 Lower Power Actuator Principles

Electro-mechanical	Fluid power	Alternative concepts
Direct Current (DC) motor	Hydraulic actuators	Piezoelectric
Alternating Current (AC) motor	Pneumatic actuators	Magnetostrictive
Stepper motor		Electrochemical
Electromagnetic		Thermo-bimetal
Linear motor		Shape Memory Alloy
		Electrostatic

TABLE 14.5.4 Comparison between Common Actuators

		Non-linearity			Accuracy
Actuator type	**Static linearity**	**Friction**	**Backlash**	**Hysteresis**	**mm**
AC/DC motor with feed	A	B–C	B–C	B–C	0.005–100
Stepper motor with feed	A	B–C	B–C	B–C	0.01–50
Hydraulic cylinder		C			0.01–100
Pneumatic cylinder		C			0.1–100

Symbols: A good, negligible; B: average, common; C: bad, significant.

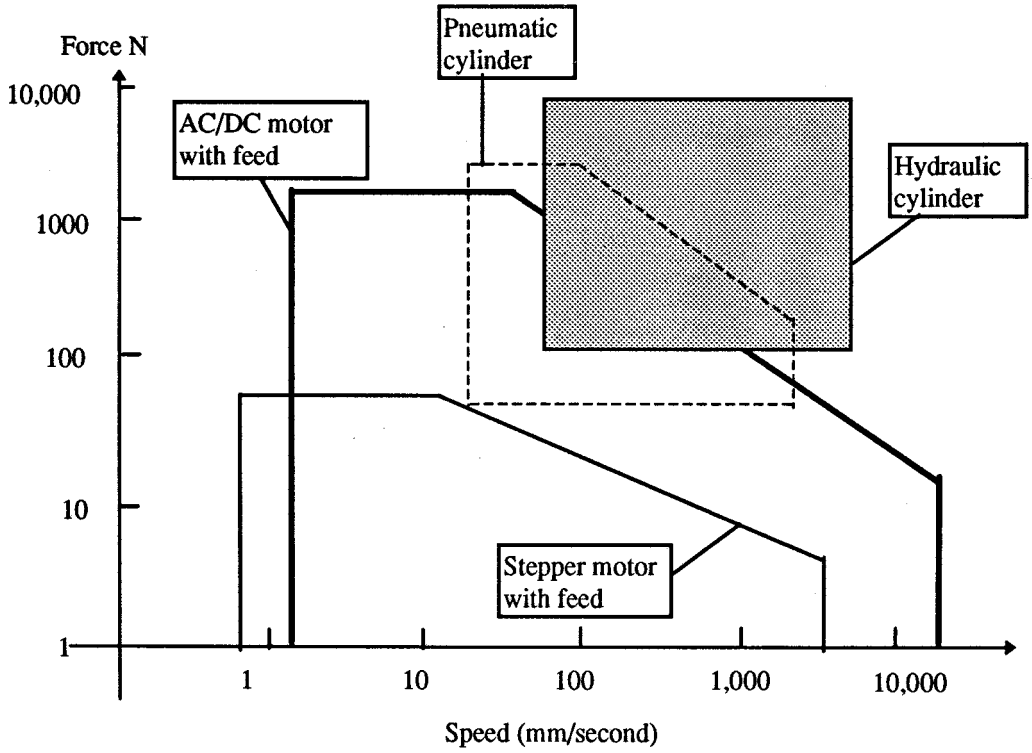


FIGURE 14.5.11 Force vs. speed for common actuators.

mechanical construction. All of the joints in a direct-drive arm are essentially identical in structure, consisting of a motor, a shaft encoder, bearings, and a housing. These components are mounted on a single shaft; a single bearing is used in the entire joint assembly. As a result, a direct drive joint generally has few but compact and more easily manufactured components than a gear-driven joint, an attractive feature for commercial production of arms.

Shape Memory Alloy (SMA) Wire

The properties of the shape memory alloy are associated with appearance and disappearance of martensite in the alloy structure. There are two kinds of martensites, namely, thermal martensite which is generated by cooling the shape memory alloy below martensite transition temperature, and stress-induced martensite which is generated by loading the stress on a shape memory alloy having an austenite structure. Shape memory effect (SME) is associated with the former, and superconductivity (SE) is associated with the latter. By making use of SME and SE, it is possible to use the shape memory alloy as an element of the actuator of a joint mechanism as shown in Figures 14.5.12 and 14.5.13. In Figure 14.5.12, mass 1 is driven toward the right side by heating the SMA wire A and cooling SMA wire B. Similarly, by reversing the direction of the heating and cooling, mass 1 can be made to move toward the left. An alternative SMA wire-actuated revolute joint using an ordinary spring is shown in Figure 14.5.13. The shape memory alloy joint mechanism has the advantage of being light in weight and simple. However, it generally has a relatively low efficiency and is slow in response.

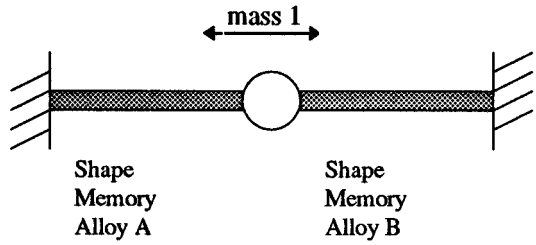


FIGURE 14.5.12 Model illustrating SME wire for joint mechanism.

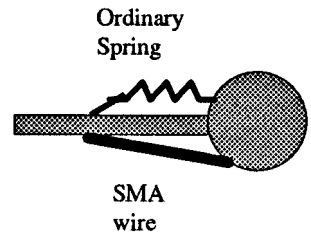


FIGURE 14.5.13 Schematic illustrating SME wire for revolute joint.

An increasing need for high performance robotic applications has motivated several researchers to direct their investigation efforts to new actuator concepts. In some applications such as high-speed plasma, water-jet and laser cutting, active joystick, and coordinate-measuring machines, the demands on workspace and the wrist force/torque are low, but the end effector must be oriented quickly, continuously, and isotropically in all directions. The performance of the popular three-consecutive-rotational-joints wrist, which possesses singularities within its workspace, is less than optimum. Several alternative designs have been developed, which present some attractive possibilities by combining pitch, roll, and yaw motion in single balljoint-like actuators. Among these design concepts are the spherical induction motor, the DC spherical servo motor, and the variable-reluctance (VR) spherical motor. Major developments for robotics are given as follows.

Spherical Induction Motor

In a spherical induction motor, three sets of windings are necessary to realize rotation about an arbitrary axis. The three windings are positioned to give rotations about the x, y, and z axes. By independently controlling the strength and phase of any two windings, one can realize a rotation vector at any point in the rotation plane of the two windings. Analyses of fields and torques in the spherical induction motor have been performed; however, realization of a prototype spherical induction motor remains to be

demonstrated. The mechanical design of a spherical motor is complex. Laminations are required to prevent unwanted eddy currents. Complicated three-phase windings must be mounted in recessed grooves in addition to the rolling supports for the rotor in a static configuration.

Spherical DC Servo Motor

The rotor of a spherical DC servo motor is a disk comprising a yoke with four permanent magnets attached to its periphery. The pivot bearing is constructed of three small radial ball bearings and has three DOF. Therefore, the rotor can incline and rotate around the three axes of Cartesian coordinates relative to the stator. The inclined and rotated angles are detected by rotary encoders attached to each axis. Three sets of windings are set at 30° apart around the z axis such that four electromagnetic force vectors can be obtained at the locations where the currents intersect the magnetic flux. They are controlled like three separated brushless motors. Although the DC spherical motor is characterized by its constructional simplicity, the range of inclination and the torque constant are rather limited.

Variable-Reluctance (VR) Spherical Motor

The structure of a VR spherical motor is shown in Figure 14.5.14, which consists of three subassemblies: a rotor, a stator, and a measuring system. The rotor is a smooth sphere in which m magnetic poles are embedded. The stator is a hollow sphere with n stator coils radially mounted on its inner surface. It also serves as a structure that holds together all the other functional elements which include the stator coils, the bearing, and the measurement system.

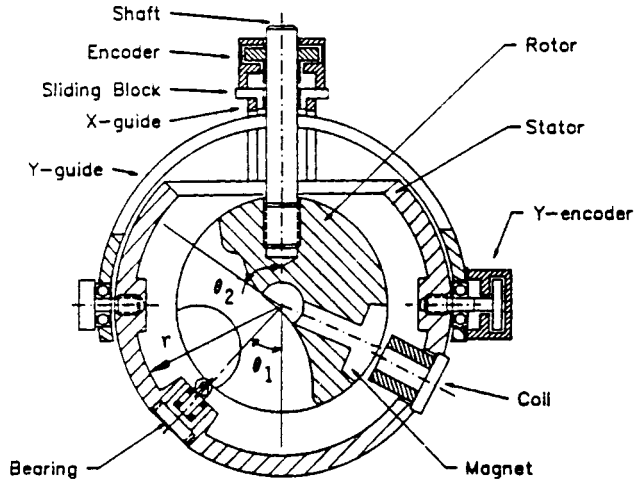


FIGURE 14.5.14 Schematic illustrating VR spherical motor.

In the operation of the VR spherical motor, the stator coils are energized individually using the control circuitry. A magnetic field is established which stores magnetic energy in the airgaps. The stored energy is a function of the relative position of the rotor and the stator. The motion of the spherical motor is thus generated as the rotor tends to move to a position such that the energy in the airgap is minimized. The VR spherical motor is characterized by the following features: (1) it offers a potential advantage of distributing the input power among several coils, each of which contributes a small fraction of the total mmf's required to generate a specified torque, and, thus, it allows a relatively low current per coil but a large surface area for heat dissipation; (2) there are multiple solutions to the selection of coil excitations, which allow an optimal electrical input vector to be chosen to minimize a preselected cost function.

As compared to its counterpart, a VR spherical motor has a relatively large range of inclination, possesses isotropic properties in motion, and is relatively simple and compact in design. The tradeoff, however, is that a sophisticated control scheme is required.

Multi-DOF Microactuators

Silicon exhibits very useful mechanical properties and its applications to microsensors and actuators have been extensively researched around the world. The fabrication of silicon-based components typically employs micromachining. Thin film and photolithographic fabrication procedures make it possible to realize a great variety of extremely small, high precision mechanical structures using the same processes that have been developed for electronic circuits. This technology has enabled the realization of many innovative microactuators operating on the basis of electrostatic to mechanical transduction.

14.6 Robot Programming Languages

Ron Bailey

The earliest industrial robots were simple sequential machines controlled by a combination of servo motors, adjustable mechanical stops, limit switches, and programmable logic controllers. These machines were generally programmed by a record and play-back method with the operator using a teach pendant to move the robot through the desired path. MHI, the first robot programming language, was developed at MIT during the early 1960s. It was an interpreted language which ran on a TX-O computer. WAVE, developed at Stanford during the early 1970s, was a general-purpose language which ran as an assembler on a DEC PDP-10 minicomputer. MINI, developed at MIT during the mid-1970s, was an expandable language based on LISP. It allowed programming in Cartesian coordinates with independent control of multiple joints. AL (Mujtaba, 1982), developed at Stanford, included some programming features of ALGOL and Pascal. VAL and VAL II (Shimano et al., 1984), developed by Unimation, Inc., was an interpreted language designed to support the PUMA series of industrial robots.

AML (A Manufacturing Language) (Taylor et al., 1982) was a completely new programming language developed by IBM to support the R/S 1 assembly robot. It was a subroutine-oriented, interpreted language which ran on the Series/1 minicomputer. Later versions were compiled to run on IBM-compatible personal computers to support the 7535 series of SCARA robots. MCL, developed by McDonnell Douglas, was an extension of the *APT* language (Automatic Programming of Tools) originally written for numerically controlled machine tools. RAIL, developed by AUTOMATIX, Inc., was an extension of Pascal designed to control robot welding and vision systems. Several additional languages (Gruver et al., 1984; Lozano-Perez, 1983) were introduced during the late 1980s to support a wide range of new robot applications which were developed during this period.

V+, developed by Adept Technologies, Inc., is a representative modern robot programming language. It has several hundred program instructions and reserved keywords. V+ will be used in the following sections to demonstrate important features of robot programming.

Robot Control

Program instructions required to control robot motion specify location, trajectory, speed, acceleration, and obstacle avoidance. Examples of V+ robot control commands are as follows:

MOVE	Move the robot to a new location
APPRO	Move to a location set back from a named position
DEPART_	Back away from the current location
DELAY	Stop the motion for a specified period of time
SPEED	Set the speed for subsequent motions
ACCEL	Set the acceleration and deceleration for subsequent motions
SINGLE	Limit motion of a joint
MULTIPLE	Allow full motion of the wrist joints
OPEN_	Open the hand
CLOSE_	Close the hand
RELAX_	Turn off air pressure to the hand

System Control

In addition to controlling robot motion, the system must support program editing and debugging, program and data manipulation, program and data storage, program control, system definitions and control, system status, and control/monitoring of external sensors. Examples of V+ control instructions are as follows:

EDIT	Initiate line-oriented editing
SEE	Initiate screen-oriented editing

STORE_	Store information from memory onto a disk file
LOAD	Read the contents of a disk file into memory
COPY	Copy an existing disk file into a new program
SPEED	Set the overall speed of robot motions
EXECUTE	Initiate execution of a program
ABORT	Stop program execution
DO	Execute a single program instruction
WATCH	Set and clear breakpoints for diagnostic execution
WHERE	Display current robot location
TEACH	Define a series of robot location variables
CALIBRATE	Initiate the robot positioning system
STATUS	Display the status of the system
TIME	Set and display current date and time
ENABLE	Turn on one or more system switches
DISABLE	Turn off one or more system switches

Structures and Logic

Program instructions are needed to organize and control execution of the robot program and interaction with the user. Examples include the following:

FOR	Execute a group of instructions a number of times
WHILE	Continue execution of a group of instructions until a condition is satisfied
DO	Execute a group of instructions until a condition is satisfied
IF	Control whether a group of instructions is executed
TYPE	Output a message to the terminal
ATTACH	Make the control pendant active
WRITE	Output a message to the manual control pendant
PENDANT	Receive input from the manual control pendant
PARAMETER	Set the value of a system parameter

Special Functions

Various special functions are required to facilitate robot programming. These include mathematical expressions and instructions for data conversion, manipulation, and transformation. Examples are as follows:

ABS	Absolute value
COS	Cosine function
SQRT	Square root
BCD	Convert from real to Binary Coded Decimal
DCB	Convert from binary to real
FRAME	Compute the reference frame based on given locations
TRANS	Compose a transformation from individual components
INVERSE	Return the inverse of the specified transformation

Program Execution

Organization of a program into a sequence of executable instructions requires scheduling of tasks, control of subroutines, and error trapping/recovery. Examples include the following:

PCEXECUTE	Initiate the execution of a process control program
PCABORT	Stop execution of a process control program

PCPROCEED	Resume execution of a process control program
PCRETRY	After an error, resume execution at the last step tried
PCEND	Stop execution of the program at the end of the current execution cycle

Example Program

This program demonstrates a simple pick and place operation.

```
 1  .PROGRAM move.parts()
 2  ; Pick up parts at location "pick" and put them down at "place"
 3  parts = 100                          ; Number of parts to be processed
 4  height1 = 25                         ; Approach/depart height at "pick"
 5  height2=50;
    Approach/depart height at "place"
 6  PARAMETER.HAND.TIME = 16             ; Setup for slow hand
 7  OPEN                                 ; Make sure hand is open
 8  MOVE start                           ; Move to safe starting location
 9  For i = 1 TO parts                   ; Process the parts
10  APPRO pick, height1                  ; Go toward the pick-up
11  MOVES pick                           ; Move to the part
12  CLOSEI                               ; Close the hand
13  DEPARTS height1                      ; Back away
14  APPRO place, height2                 ; Go toward the put-down
15  MOVES place                          ; Move to the destination
16  OPENI                                ; Release the part
17  DEPARTS height2                      ; Back away
18  END                                  ; Loop for the next part
19  TYPE "ALL done.", /I3, parts, "parts processed"
20  STOP                                 ; End of the program
21  .END
```

Off-Line Programming and Simulation

Computer-integrated manufacturing operations require off-line programming and simulation in order to layout production facilities, model and evaluate design concepts, optimize motion of devices, avoid interference and collisions, minimize process cycle times, maximize productivity, and ensure maximum return on investment. Commercially available software (e.g., ROBCAD and SILMA [SILMA Inc., 1992]) provides support for 3D workable layouts including robots, end effectors, fixtures, conveyors, part positioners, and automatic guided vehicles. Dynamic simulation allows off-line creation, animation, and verification of robot motion programs. However, these techniques are limited to verification of overall system layout and preliminary robot program development. With support for data exchange standards (e.g., *IGES* [International Graphics Exchange Specification], *VDAFS* [Virtual Data Acquisition and File Specification], *SET* [Specification for Exchange of Text]), these software tools can pass location and trajectory data to a robot control program which, in turn, must provide the additional functions (operator guidance, logic, error recovery, sensor monitoring/control, system management, etc.) required for full operation.

14.7 Robot Dynamics and Control

Frank L. Lewis

This section deals with the real-time motion control of robot manipulators. In Section 14.8 are covered higher-level planning and control functions, including the generation of the prescribed trajectory that is assumed given in this section. Robot manipulators have complex nonlinear dynamics that might make accurate and robust control difficult. Fortunately, robots are in the class of Lagrangian dynamical systems, so that they have several extremely nice physical properties that make their control straightforward. In this section will be discussed several control techniques including computed torque (e.g., *feedback linearization*), classical joint control, digital control, *adaptive control*, *robust control*, *learning control*, *force control*, and teleoperation. More information may be found in Lewis et al. (1993). The advances that made possible this modern approach to robot control were made by Craig (1985), Slotine and Li (Slotine, 1988), Spong and Ortega (Spong and Vidyasagar, 1989), and others.

Robot Dynamics and Properties

A robot manipulator can have either revolute joints or prismatic joints. The latter are actuators that function like an automobile antenna, extending and contracting on a linear axis. The values of the angles, for revolute joints, and link lengths, for prismatic joints, are called the link variables, and are denoted $q_1(t)$, $q_2(t)$, …, $q_n(t)$ for joints one, two, and so on. The number of links is denoted n; for complete freedom of motion in space, six degrees of freedom are needed, three for positioning and three for orientation. Thus, most commercial robots have six links. We discuss here robots which are rigid, that is which have no flexibility in the links or in the gearing of the joints; flexible robots are discussed in the section on control of flexible-link and flexible-joint robots.

The dynamics of robot manipulators with rigid links can be written as

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) + \tau_d = \tau \tag{14.7.1}$$

where $M(q)$ is the inertia matrix, $V_m(q,\dot{q})$ is the coriolis/centripetal matrix, $F(\dot{q})$ are the friction terms, $G(q)$ is the gravity vector, $\tau_d(t)$ represents disturbances, and $\tau(t)$ is the control input torque. The joint variable $q(t)$ is an n-vector containing the joint angles for revolute joints and lengths for prismatic joints. It is often convenient to write the robot dynamics as

$$M(q)\ddot{q} + N(q,\dot{q})\dot{q} + \tau_d = \tau \tag{14.7.2}$$

$$N(q,\dot{q}) \equiv V_m(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) \tag{14.7.3}$$

where $N(q,\dot{q})$ represents a vector of the nonlinear terms.

The objective of robot control is generally to select the control torques $\tau(t)$ so that the robot follows a desired prescribed motion trajectory or exerts a desired force. Examples include spray painting, grinding, or manufacturing assembly operations. The position control objective can be achieved by first defining a desired trajectory $q_d(t)$, which is a vector containing the desired values vs. time $q_{d_i}(t)$ of each of joint of the manipulator. This desired trajectory vector $q_d(t)$ is determined in a higher-level *path planner*, based on a even higher-level task decomposition, and then fed to the real-time motion control system. This section discusses the real-time motion control problem assuming that $q_d(t)$, or a similar desired force vector, is given.

The robot dynamics in Equation (14.7.1) satisfies some important physical properties as a consequence of the fact that they are a Lagrangian system. These properties significantly simplify the robot control problem. The main properties of which one should be aware are the following.

Properties of Robot Arm Dynamics

P1 The inertia matrix $M(q)$ is symmetric, positive definite, and bounded so that $\mu_1 I \le M(q) \le \mu_2 I$ for all $q(t)$. For revolute joints, the only occurrences of the joint variables q_i are as $\sin(q_i)$, $\cos(q_i)$. For arms with no prismatic joints, the bounds μ_1, μ_2 are constants.

P2 The coriolis/centripetal vector $V_m(q,\dot{q})\dot{q}$ is quadratic in $\dot{q}$ and bounded so that $\|V_m\dot{q}\| \le v_B\|\dot{q}\|^2$.

P3 The coriolis/centripetal matrix can always be selected so that the matrix $\dot{M}(q) - 2V_m(q,\dot{q})$ is *skew symmetric*. This is a statement of the fact that the fictitious forces in the robot system do no work.

P4 The friction terms have the approximate form $F(\dot{q}) = F_v\dot{q} + F_d(\dot{q})$, with F_v a diagonal matrix of constant coefficients representing the viscous friction, and $F_d(\cdot)$ a vector with entries like $K_{d_i}\,\mathrm{sgn}(\dot{q}_i)$, with $\mathrm{sgn}(\cdot)$ the signum function, and K_{d_i} the coefficients of dynamic friction. These friction terms are bounded so that $\|F(\dot{q})\| \le v_B\|\dot{q}\| + k_B$ for constants v_B, k_B.

P5 The gravity vector is bounded so that $\|G(q)\| \le g_B$. For revolute joints, the only occurrences of the joint variables q_i are as $\sin(q_i)$, $\cos(q_i)$. For revolute joint arms the bound g_B is a constant

P6 The disturbances are bounded so that $\|\tau_d(t)\| \le d$.

P7 The nonlinear robot terms are *linear in the parameters* of mass and friction so that one can write

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) = W(q,\dot{q},\ddot{q})\phi \tag{14.7.4}$$

where $W(q,\dot{q},\ddot{q})$ *is* a matrix of known robot functions and ϕ is a vector of mass and friction coefficient parameters, often unknown. The *regression matrix* $W(\cdot)$ can be computed for any specified robot arm.

The last property, P7, is especially useful in adaptive control approaches. The bounding properties are especially useful in robust control approaches. The *skew-symmetry* property P3 is vital for Lyapunov control proofs, which provide guaranteed tracking motion and often give the structure of the control loops. It essentially allows some very nice *linear systems techniques* to be used with the time-varying robot dynamics.

State Variable Representations and Computer Simulation

The nonlinear state-variable representation $\dot{x} = f(x, u)$, with $x(t)$ the internal state and $u(t)$ the control input, is very convenient for many applications, including the derivation of suitable control laws and computer simulation. Once the system has been put into state-space form, it can easily be integrated to obtain simulation time plots using, for instance, a Runge-Kutta integrator; many standard software packages have such integration routines, including MATLAB, MATRIX$_X$, and SIMNON.

It is supposed for convenience in this subsection that the disturbance $\tau_d(t)$ is equal to zero. There are three convenient state-space formulations for the robot dynamics in Equation (14.7.1). In the *position/velocity state-space form*, one defines the state as the $2n$-vector $x \equiv [q^T\ \dot{q}^T]^T$ and writes

$$\dot{x} = \begin{bmatrix} \dot{q} \\ -M^{-1}(q)N(q,\dot{q}) \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}(q) \end{bmatrix} \tag{14.7.5}$$

which is in state-space form with $u(t) \equiv \tau(t)$.

For computer simulation purposes, the matrix inversion $M^{-1}(q)$ is required at every integration time step. For arms with simple dynamics, it is often possible to invert the inertia matrix analytically off-line, reducing the on-line computational burden. Otherwise, it is more suitable to solve Equation (14.7.2) for $\ddot{q}$, required by the integration routine, using least-squares techniques to avoid the inversion of $M(q)$.

An alternative *linear* state-space equation in the form $\dot{x} = Ax + Bu$ can be defined as

$$\dot{x} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ I \end{bmatrix} u \tag{14.7.6}$$

with $u(t) \equiv -M^{-1}(q)\, N(q,\dot{q}) + M^{-1}(q)\tau$. This is known as the *Brunovsky Canonical Form.*

The third state-space formulation is the *Hamiltonian form,* which derives from Hamilton's equations of motion. Here the state is defined as the $2n$-vector $x = (q^T\ p^T)^T$, with $p(t) \equiv M(q)\dot{q}$ the *generalized momentum*. Then the state-space equation is

$$\dot{x} = \begin{bmatrix} M^{-1}(q)\, p \\ -\frac{1}{2}\left(I_n \otimes p^T\right)\dfrac{\partial M^{-1}(q)}{\partial q}\, p \end{bmatrix} + \begin{bmatrix} 0 \\ I_n \end{bmatrix} u \tag{14.7.7}$$

with the control input defined by $u = \tau - G(q)$ and $\otimes$ the Kronecker product (Lewis et al., 1993).

Cartesian Dynamics and Actuator Dynamics

Cartesian Dynamics

The dynamics in Equation (14.7.1) are known as the *joint-space dynamics*; they are expressed in the joint-space coordinates q. Cartesian coordinates referred to some frame, often the base of the robot manipulator, may be used to describe the position of the end effector of the robot arm. Denote the Cartesian coordinates of the end of the arm as $Y(t) = h(q)$, whose first three coordinates represent position and last coordinates represent orientation. The nonlinear function $h(q)$ gives the end effector Cartesian coordinates in termexcepts of the current joint positions q and is called the arm *kinematics transformation.* The *arm Jacobian* relates joint and Cartesian velocites and is defined as $J(q) \equiv \partial h(q)/\partial q$ so that

$$\begin{bmatrix} v \\ \omega \end{bmatrix} \equiv \dot{Y} = J(q)\dot{q} \tag{14.7.8}$$

where $v(t)$ is the linear velocity and $\omega(t)$ the angular velocity of the end effector. Both these velocities are 3-vectors. Differentiating this equation gives the *acceleration transformation* $\ddot{Y} = J\ddot{q} + \dot{J}\dot{q}$.

By differentiating Equation (14.7.1) one discovers that the dynamics may be written in Cartesian form as

$$\overline{M}\ddot{Y} + \overline{N} + f_d = F \tag{14.7.9}$$

where $\overline{M} \equiv J^{-T}MJ^{-1}$, $\overline{N} \equiv J^{-T}(N - MJ^{-1}\,\dot{J}\,J^{-1}\,\dot{y}\,)$, and the disturbance is $f_d \equiv J^{-T}\tau_d$. In the Cartesian dynamics the control input is F, which has three components of force and three of torque.

The important conclusion of this discussion is that the Cartesian dynamics are of the same form as Equation (14.7.2). Furthermore, it can be shown that the properties of the robot dynamics hold also in Cartesian form. Therefore, all the control techniques to be described in this section can be used for either the joint-space or the Cartesian dynamics.

Actuator Dynamics

The robot manipulator is driven by actuators which may be electric, hydraulic, pneumatic, and so on. Considering the case of electric motors it is direct to show that if the armature inductance is negligible, the dynamics of the arm plus actuators can be written as

$$\left(J_M + R^2M\right)\ddot{q} + \left(B_M + R^2V_m\right)\dot{q} + \left(RF_M + R^2F\right) + R^2G = RK_M v \tag{14.7.10}$$

where the robot arm dynamics are described by $M(q)$, $V_m(q,\dot{q})$, $F(\dot{q})$, $G(q)$, and J_M is the motor inertia, B_M is given by the rotor damping constant and back emf, and R has diagonal elements containing the

gear ratios of the motor/joint couplings. The control input is the motor voltage $v(t)$, with K_M the diagonal matrix of motor torque constants.

The important conclusion is that the dynamics of the arm-plus-actuators has the same form as the dynamics (Equation (14.7.1) and can be shown to enjoy the same properties of boundedness and linearityin-the-parameters. Therefore, the control methods to be described herein apply to this composite system as well. Similar comments hold for other sorts of actuators such as hydraulic. If the armature inductances of the electric motors are not negligible, then the arm-plus-actuators have a coupled form of dynamics such as those discussed in the section on control of flexible-link and flexible-joint robots. Then special control techniques must be used.

Computed-Torque (CT) Control and Feedback Linearization

For many years during the 1960s and 1970s the major techniques for robot dynamics control were based on the *computed-torque method*, which has many variants, including classical independent joint control. Recently, advanced mathematical techniques based on *feedback linearization* have been derived. For the rigid-link arms, these are equivalent.

It is assumed that the desired motion trajectory for the manipulator $q_d(t)$, as determined, for instance, by a path planner, is prescribed. Define the *tracking error* as

$$e(t) = q_d(t) - q(t) \tag{14.7.11}$$

and differentiate twice to see that the Brunovsky canonical form in Equation (14.7.6) can be written in terms of the state $x = [e^T \ \dot{e}^T]^T$ as

$$\frac{d}{dt}\begin{bmatrix} e \\ \dot{e} \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}\begin{bmatrix} e \\ \dot{e} \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} u \tag{14.7.12}$$

with

$$u \equiv \ddot{q}_d + M^{-1}(q)\left(N(q,\dot{q}) - \tau\right) \tag{14.7.13}$$

A two-step design procedure now suggests itself. First, use linear system design techniques to select a feedback control $u(t)$ that stabilizes the tracking error system in Equation (14.7.12), then compute the required arm torques using the inverse of Equation (14.7.13), namely,

$$\tau = M(q)\left(\ddot{q}_d - u\right) + N(q,\dot{q}) \tag{14.7.14}$$

This is a *nonlinear feedback control law* that guarantees tracking of the desired trajectory. It relies on computing the torque τ that makes the nonlinear dynamics of Equation (14.7.1), equivalent to the linear dynamics of Equation (14.7.12), which is termed *feedback linearization.*

Selecting proportional-plus-derivative (PD) feedback for $u(t)$ results in the *PD computed-torque controller*

$$\tau = M(q)\left(\ddot{q}_d + K_v\dot{e} + K_p e\right) + N(q,\dot{q}) \tag{14.7.15}$$

and yields the tracking error dynamics $\ddot{e} = -K_v\dot{e} - K_p e$, which is stable as long as the derivative gain matrix K_v and the proportional gain matrix K_p are selected positive definite. It is common to select the gain matrices diagonal, so that stability is ensured as long as all gains are selected positive.

The PD computed-torque controller is shown in Figure 14.7.1, which has a *multiloop structure*, with a nonlinear inner feedback linearization loop and an outer unity-gain tracking loop. Note that there are actually n outer loops, one for each joint. In this figure, $\underline{q} \equiv [q^T \ \dot{q}^T]^T$, $\underline{e} \equiv [e^T \ \dot{e}^T]^T$, $\underline{q}_d \equiv [q_d^T \ \dot{q}_d^T]^T$.

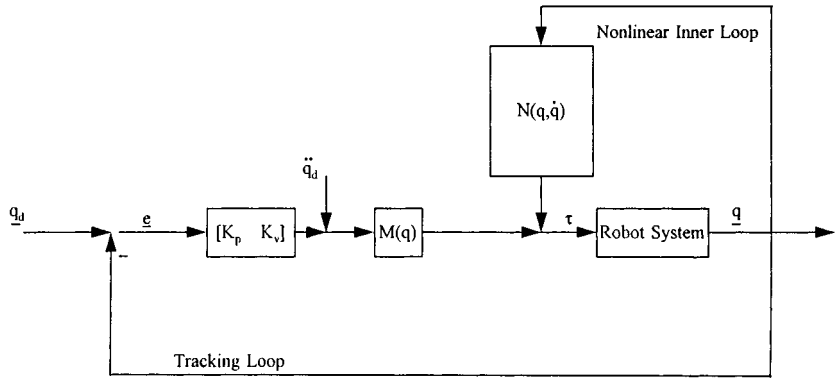


FIGURE 14.7.1 PD computed-torque controller.

To improve steady-state tracking errors, n integrators can be added, one to each joint controller, to place an integrator in the outer tracking loop in the figure. In fact, selecting $u(t)$ *as* a proportional-plus-integral-plus-derivative controller yields the *PID computed-torque controller*

$$\begin{aligned} \dot{\varepsilon} &= e \\ \tau &= M(q)\left(\ddot{q}_d + K_v\dot{e} + K_p e + K_i\varepsilon\right) + N(q,\dot{q}) \end{aligned} \tag{14.7.16}$$

which has its own dynamics and gives stable tracking as long as the integral gain K_i is not chosen too large.

Example 14.7.1 (Performance of PD and PID Computed-Torque Controllers)

The sort of performance to be expected from PD and PID CT controllers is illustrated here. It is desired for a 2-link robot arm to follow, in each of its joints, sinusoidal trajectories $q_d(t)$ with period of 2 sec.

Ideal PD CT Control. Since CT is theoretically an exact cancellation of nonlinearities, under ideal circumstances the PD CT controller yields performance like that shown in Figure 14.7.2, where the initial tracking errors go to zero quickly, so that each joint perfectly tracks its prescribed trajectory. In this figure are shown the plots for joint 1 tracking error $e_1(t)$ and joint 2 tracking error $e_2(t)$.

PD CT Control with Constant Unknown Disturbance. Now a constant unknown disturbance is added to the robot arm. As shown in Figure 14.7.3, the PD controller now exhibits steady-state tracking errors of $e_1 = -0.01$ rad, $e_2 = 0.035$ rad.

PID CT Control. If an integral term is now added to the outer loop to achieve PID CT control, even with a constant unknown disturbance, the simulation results look very much like the original plots in Figure 14.7.2; that is, the integral term has reduced the steady-state tracking errors to zero. □

A class of computed torque-like controllers is given by selecting

$$\tau = \hat{M}\left(\ddot{q}_d - u\right) + \hat{N} \tag{14.7.17}$$

where $\hat{M}, \hat{N}$ are approximations, estimates, or simplified expressions for $M(q)$, $N(q,\dot{q})$. An example is the *PD-gravity controller*

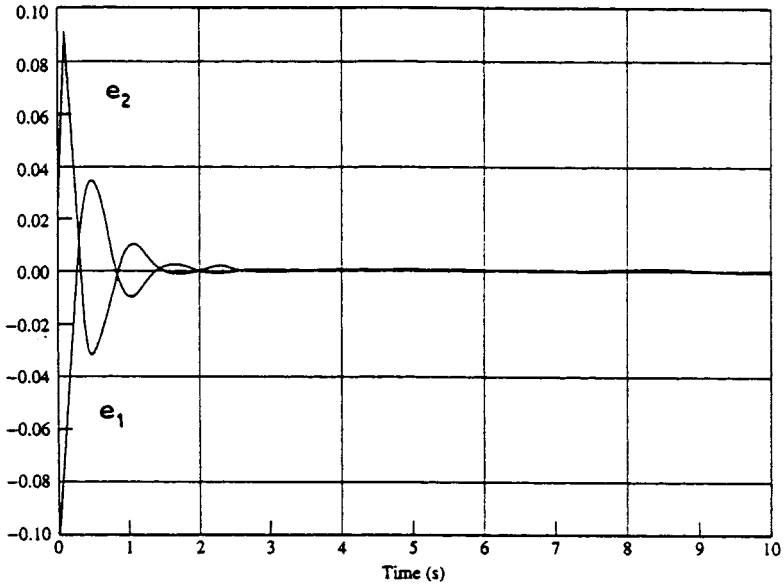


FIGURE 14.7.2 Joint tracking errors using PD computed-torque controller under ideal conditions.

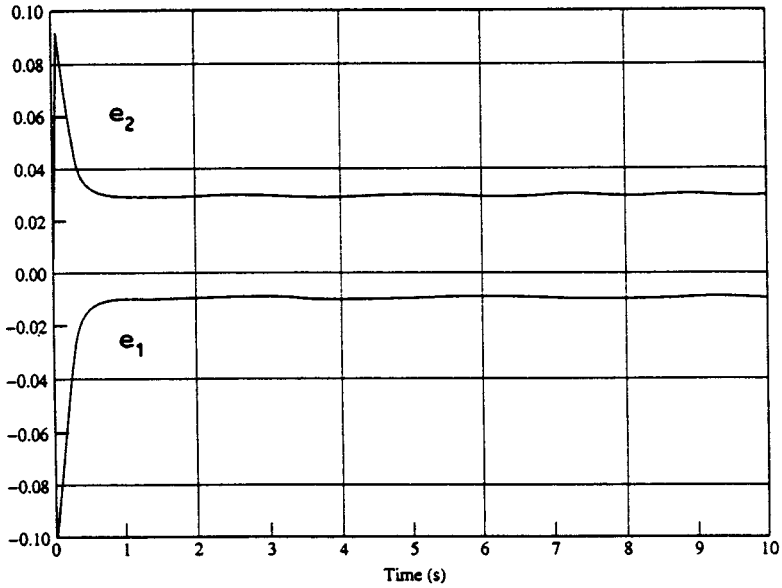


FIGURE 14.7.3 Joint tracking errors using PD computed-torque controller with constant unknown disturbance.

$$\tau = K_v\dot{e} + K_p e + G(q) \tag{14.7.18}$$

which selects $\hat{M} = I$ and only includes the gravity nonlinear terms, so that it is very easy to implement compared to full CT control. This has been used with good results in many applications.

If $\hat{M}, \hat{N}$ are selected, not as the actual inertia matrix and nonlinear terms, but only as approximations or simplified values, it is not always possible to guarantee stable tracking. In fact, the error dynamics of Equation (14.7.12) are then driven by *modeling mismatch errors*, which can degrade or even destabilize the closed-loop system.

Another computed torque-like controller is *PID classical joint control*, where all nonlinearities of the robot arm are neglected and one selects simply

$$\dot{\varepsilon} = e$$
$$\tau = K_v \dot{e} + K_p e + K_i \varepsilon \tag{14.7.19}$$

with the gain matrices diagonal, so that all the joints are decoupled. A PD classical joint controller is shown in Figure 14.7.4, which may seem familiar to many readers. The same figure may be drawn for each joint. In this figure, $d(t)$ represents the neglected nonlinear coupling effects from the other joints, and r *is* the gear ratio. The motor angle is $\theta(t)$ and $q(t)$ *is* the joint angle. The effective joint inertia and damping are J and B, respectively.

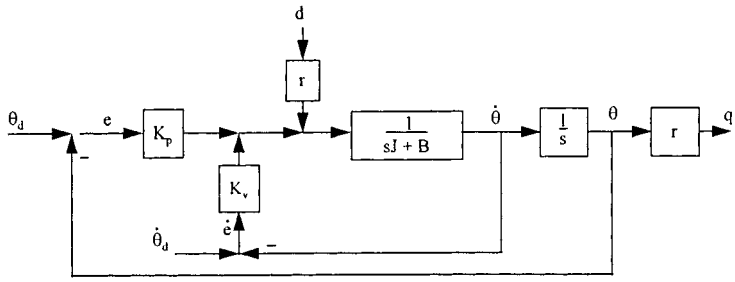


FIGURE 14.7.4 PD classical joint controller.

The simplified classical joint controller is very easy to implement, as no digital computations are needed to determine nonlinear terms. It has been found suitable in many applications if the PD gains are selected high enough, particularly when the gear ratio r is small. Unfortunately, if the gains are selected too high, the control may excite vibratory modes of the links and degrade performance. Moreover, practical applications often benefit by including additional terms such as gravity $G(q)$, desired acceleration feedforward $\ddot{q}_d(t)$, and various additional nonlinear terms.

Example 14.7.2 (Performance of PD-Gravity and Classical Joint Controllers)

The sort of performance to be expected from PD-gravity and classical joint controllers is shown in this example It is desired for a 2-link robot arm to follow, in each of its joints, sinusoidal trajectories $q_d(t)$ with period of 2 sec.

PD-Gravity Controller. The joint 1 and 2 tracking errors are shown in Figure 14.7.5. Note that the errors are small but not exactly zero, a reflection of the fact that the nonlinear coriolis/centripetal terms are missing in the controller. However, the DC error is equal to zero, since gravity compensation is used. (The gravity terms are effectively the "DC terms" of the robot dynamics.)

Classical PD Controller. The sort of behavior to be expected from classical (independent joint) control is illustrated in Figure 14.7.6. In this figure, the tracking errors are nonzero, but using large-enough PD gains can often make them small enough. Note that the DC error is no longer equal to zero; the offset is due to ignoring the gravity terms. □

Another important CT-like controller is the PD *digital controller* given by

$$\tau_k = M(q_k)\left(\ddot{q}_{d_k} + K_v \dot{e}_k + K_p e_k\right) + N(q_k, \dot{q}_k) \tag{14.7.20}$$

where the control input can only be computed at the *sample times*, $t_k = KT$, with T the sample period and k taking on integer values. Digital control is usually required in modern applications, as robot control

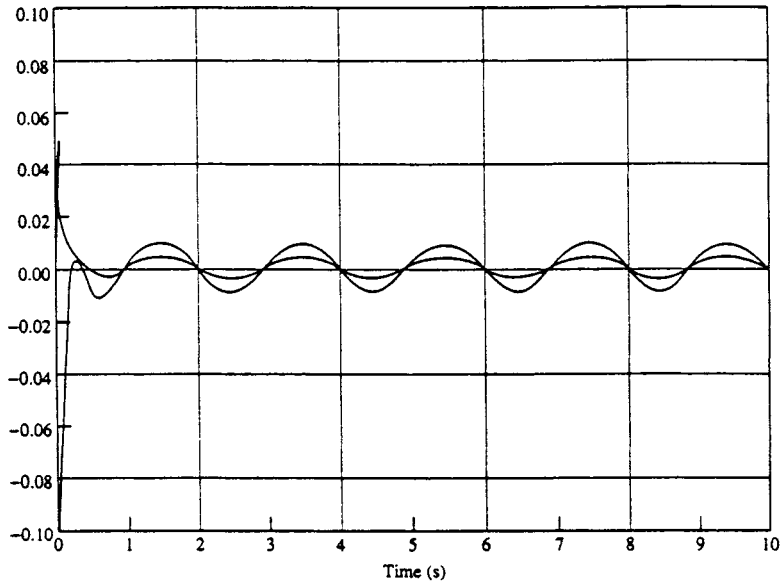


FIGURE 14.7.5 Joint tracking errors using PD-gravity controller.

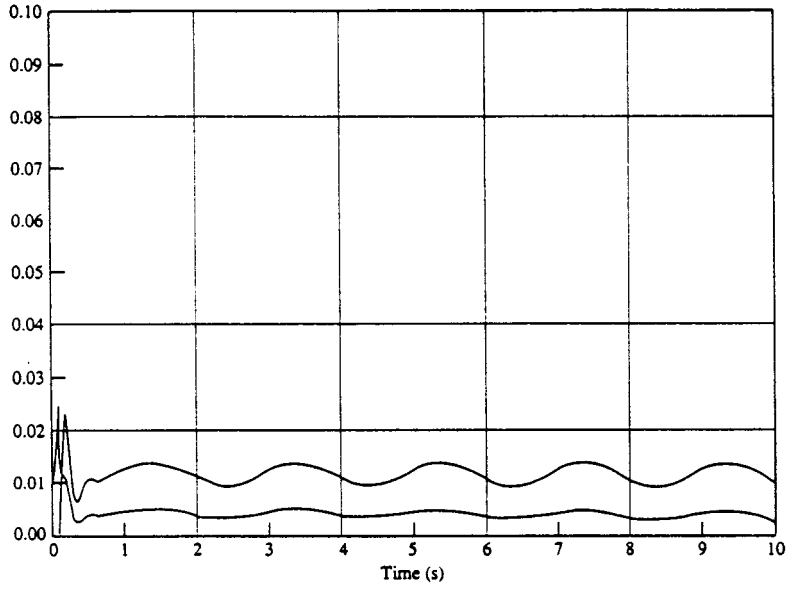


FIGURE 14.7.6 Joint tracking errors using classical independent joint control.

laws are generally implemented using microprocessors or digital signal processors. Unfortunately, the stability of robot digital controllers has not been generally addressed, so that the traditional approach relies on designing continuous-time controllers, meticulously proving stability, then sampling "fast enough" and holding one's breath.

In practice there are many other problems to be faced in robot controller implementation, including actuator saturation, antiwindup compensation, and so on. See Lewis et al., (1993).

Example 14.7.3 (Performance of Digital CT Controllers)

The performance of digital robot controllers has several idiosyncrasies of which one should be aware. In this example, it is desired for a 2-link robot arm to follow, in each of its joints, sinusoidal trajectories $q_d(t)$ with period of 2 sec.

Digital CT Controller. Using a sample period of T = 20 msec yields the tracking error plots shown in Figure 14.7.7. There the performance is quite good for the specific choice of PD gains. The associated control input for joint 2 is shown in Figure 14.7.8.

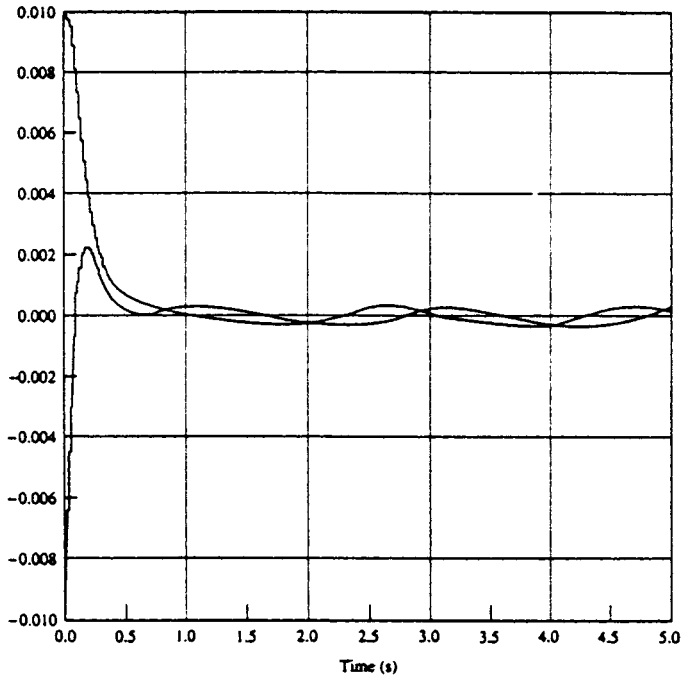


FIGURE 14.7.7 Joint tracking errors using digital computed-torque controller, T = 20 msec.

Limit Cycle of Digital Robot Controller. Unacceptable behavior in digital robot controllers can be due to integrator windup problems, selecting too large a sample period, selecting too small a sample period (so that there is not enough time to perform all control calculations in each period), or the ocurrence of *limit cycles*. If the sample period is selected as T = 100 msec, everything seems acceptable according to Figure 14.7.9, where the tracking errors are somewhat increased but still small. However, Figure 14.7.10 shoves the control torque for link 2, which has now entered a limit cycle-type behavior due to too large a sample period. □

Adaptive and Robust Control

Computed-torque control works very well when all the dynamical terms $M(q)$, $V_m(q,\dot{q})$, $F(\dot{q})$, $G(q)$ are known. In practice, robot manipulator parameters such as friction coefficients are unknown or change with time, and the masses picked up by the arm are often unknown. Moreover, computing nonlinear terms is difficult to do without exotic microprocessor-based hardware. Therefore, in applications simplified CT controllers that do not compute all nonlinear terms are used (e.g., classical joint control). These methods rely on increasing the PD gains to obtain good performance. However, large control signals may result and stability proofs of such controllers are few and far between. Adaptive and robust control techniques are useful in such situations to improve upon the performance of basic PD control

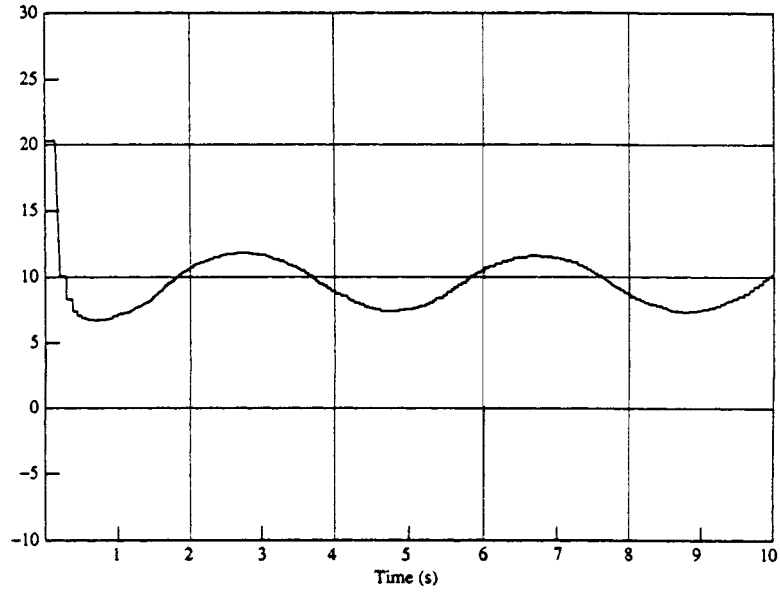


FIGURE 14.7.8 Joint 2 control torque using digital computed-torque controller, T = 20 msec.

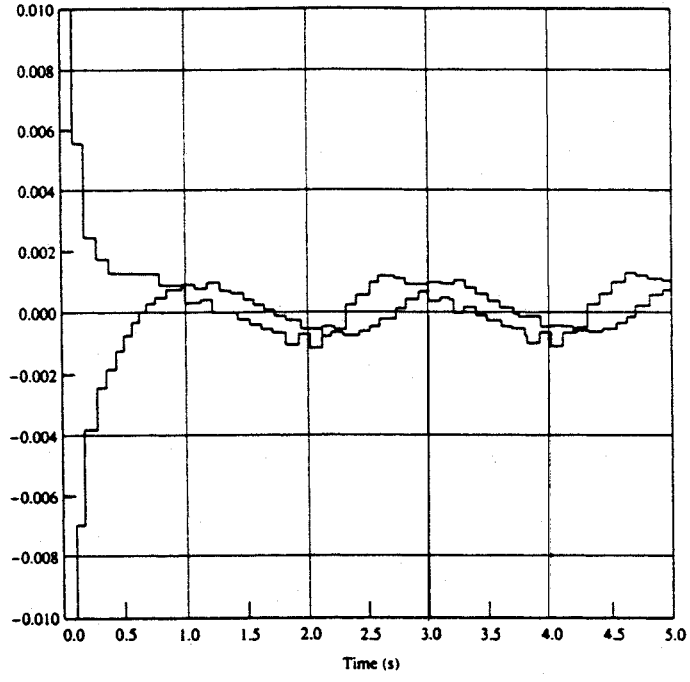


FIGURE 14.7.9 Joint tracking errors using digital computed-torque controller, T = 100 msec.

techniques, providing good performance that can be mathematically proven and so relied upon in applications. Such advanced techniques also extend directly to more complicated control objectives such as force control for grinding, polishing, and so on where straight PD methods are inadequate.

There are many sorts of adaptive and robust controllers (Lewis et al., 1993). A unifying design technique is presented here that extends as well to intelligent control using neural network and fuzzy

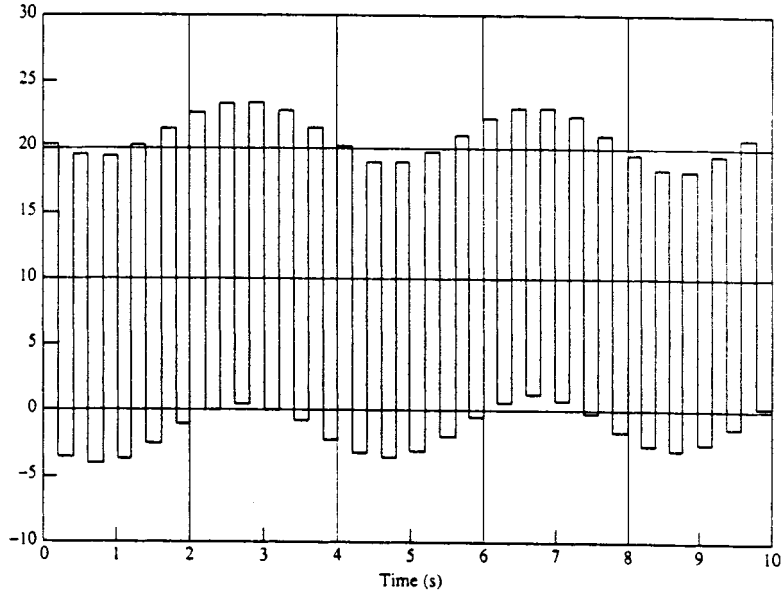


FIGURE 14.7.10 Joint 2 control torque using digital computed-torque controller, T = 100 msec.

logic techniques. Thus, given the desired trajectory $q_d(t)$, define the tracking error and *filtered tracking error* $r(t)$ by

$$e = q_d - q \tag{14.7.21a}$$

$$r = \dot{e} + \Lambda e \tag{14.7.21b}$$

with Λ a positive definite design parameter matrix. Common usage is to select Λ diagonal with large positive entries. Then Equation (14.7.21b) is a stable system so that $e(t)$ *is* bounded as long as the controller guarantees that the filtered error $r(t)$ *is* bounded.

Differentiating Equation (14.7.21b) and invoking Equation (14.7.1), it is seen that the robot dynamics are expressed in terms of the filtered error as

$$M\dot{r} = -V_m r + f(x) + \tau_d - \tau \tag{14.7.22}$$

where the *nonlinear robot function* is defined as

$$f(x) + M(q)\left(\ddot{q}_d + \Lambda\dot{e}\right) + V_m(q,\dot{q})\left(\dot{q}_d + \Lambda e\right) + F(\dot{q}) + G(q) \tag{14.7.23}$$

Vector x contains all the time signals needed to compute $f(\cdot)$ and may be defined, for instance, as $x \equiv [e^T\ \dot{e}^T\ q_d^T\ \dot{q}_d^T\ \ddot{q}_d^T]^T$. It is important to note that $f(x)$ contains all the potentially unknown robot arm parameters, except for the $V_m r$ term in Equation (14.7.22), which cancels out in the proofs.

A general sort of controller is now derived by setting

$$\tau = \hat{f} + K_v r - v(t) \tag{14.7.24}$$

with $\hat{f}$ an *estimate* of $f(x)$, $K_v r = K_v \dot{e} + K_v \Lambda e$ an *outer PD tracking loop*, and $v(t)$ an auxiliary signal to provide robustness in the face of disturbances and modeling errors. The estimate $\hat{f}$ and robustifying

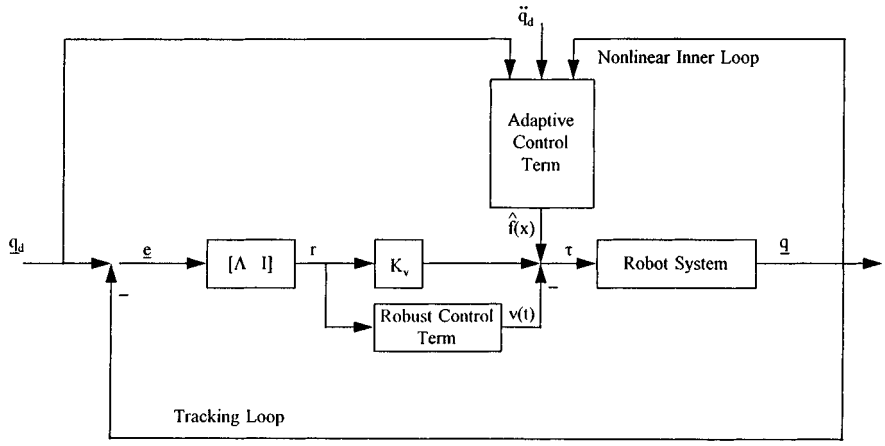


FIGURE 14.7.11 Adaptive filtered error controller.

signal $v(t)$ are defined differently for adaptive control, robust control, neural net control, *fuzzy logic control*, etc. The multiloop control structure implied by this scheme is shown in Figure 14.7.11.

Adaptive Controller

Using nonlinear stability proofs based on Lyapunov or passivity techniques, it can be shown that tracking error stability can be guaranteed by selecting one of a variety of specific controllers. One such is the adaptive controller shown in Figure 14.7.11 and described by the equations

$$\begin{aligned} \tau &= W(w)\hat{\phi} + K_v r \\ \dot{\hat{\phi}} &= \Gamma W^T(w) r \end{aligned} \tag{14.7.25}$$

where Γ is a tuning parameter matrix, generally selected diagonal with positive elements. Matrix $W(x)$ is the (known) regression matrix chosen such that $f(x) = W(x)\phi$, with all the unknown parameters placed into the vector ϕ; $W(x)$ must be computed off-line in a design phase for the specific robot arm to be controlled (Lewis et al., 1993). In the adaptive controller, the second equation represents the internal dynamics of the controller, where the estimate $\hat{\phi}$ of the unknown parameter vector is produced by dynamic on-line tuning. The robot control input $\tau(t)$ *is* then given in terms of $\hat{\phi}$ by the first equation. Though it need not be computed to produce the control inputs, the estimate of the nonlinear function is given by $\hat{f}(x) = W^T(x)\hat{\phi}$.

The adaptive controller shown in Figure 14.7.11 has a multiloop structure with an outer PD tracking loop and an inner nonlinear adaptive loop whose function is to estimate the nonlinear function required for feedback linearization of the robot arm.

Robust Saturation Controller

Another filtered error controller is the robust saturation controller

$$\begin{aligned} \tau &= \hat{f} + K_v r - v \\ v &= \begin{cases} -r\dfrac{F(x)}{\|r\|}, & \|r\| \geq \varepsilon \\ -r\dfrac{F(x)}{\varepsilon}, & \|r\| < \varepsilon \end{cases} \end{aligned} \tag{14.7.26}$$

where $\hat{f}$ is an estimate for $f(x)$ that is not changed on-line — for instance, a PD-gravity-based robust controller would use $\hat{f} = G(q)$, ignoring the other nonlinear terms. In computing the robust control term $v(t)$, ε is a small design parameter, $\|\cdot\|$ denotes the norm, and $F(x)$ is a known function that bounds the uncertainties $\|f - \hat{f}\|$. The intent is that $F(x)$ is a simplified function that can be computed even if the exact value of the complicated nonlinear function $f(x)$ is unknown.

Variable Structure Robust Controller

Another robust controller is the *variable structure robust controller*

$$\tau = \hat{f} + K_v r - v$$
$$v = -\big(F(x) + \eta\big)\mathrm{sgn}(r) \tag{14.7.27}$$

where sgn(·) *is* the signum function and $F(x)$ *is* a known function computed to bound the uncertainties $\|f - \hat{f}\|$. The design parameter η is selected as a small value. This controller takes advantage of the properties of sliding mode or variable structure controllers to provide its robustness.

In adaptive controllers the primary design effort goes into selecting a dynamic estimate $\hat{f}$ that is tuned on-line. By contrast, in robust controllers, the primary design effort goes into selecting the robust term $v(t)$. An advantage of robust controllers is that they have no dynamics, so they are generally simpler to implement. On the other hand, adaptive controllers are somewhat more refined in that the dynamics are learned on-line and less control effort is usually needed. Furthermore, in adaptive control it is necessary to compute the regression matrix $W(x)$, while in robust control it is necessary to compute the bounding function $F(x)$.

Example 14.7.4 (Performance of Adaptive and Robust Robot Controllers)

This example illustrates the sort of performance to be expected from adaptive and robust controllers. In this example, it is desired for a 2-link robot arm to follow, in each of its joints, sinusoidal trajectories $q_d(t)$ with period of 2 sec.

Adaptive Control. In adaptive control, the controller dynamics allow for learning of the unknown parameters, so that the performance improves over time. Typical plots are like those in Figure 14.7.12, where the errors start out large but then converge to zero, and the parameter (mass) estimates converge to constant values.

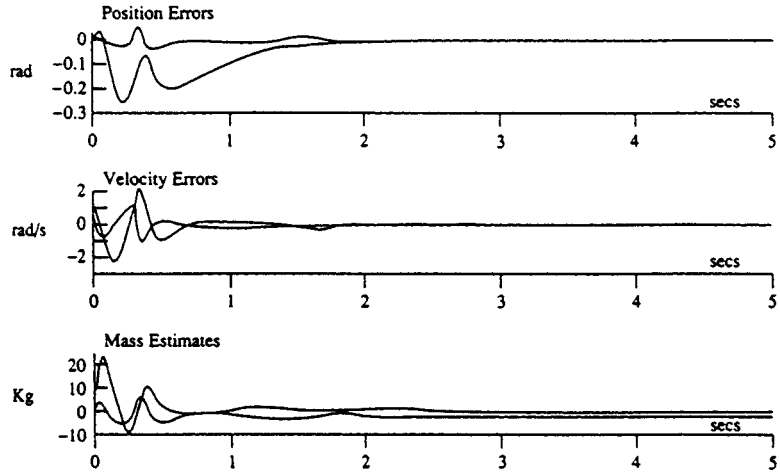


FIGURE 14.7.12 Typical behavior of adaptive controller.

Robust Control. In typical robust controllers, there are no controller dynamics so that the performance does not improve with time. However, with good designs (and large-enough control gains) the errors are bounded so that they are small enough. Typical plots are like those in Figure 14.7.13, where the errors are always small, though nonzero, but do not become smaller with time. □

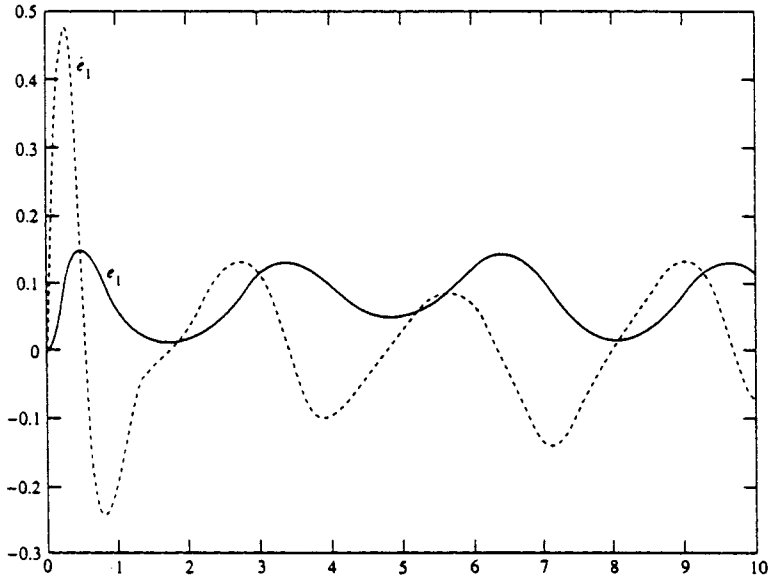


FIGURE 14.7.13 Typical behavior of robust controller.

Learning Control

In many industrial applications robot manipulators are used to perform the same task repeatedly, such as in spray painting, short assembly operations, component insertion, and so on. In such repetitive motion situations, information from one iteration can be recorded and used to improve the performance on the next iteration. This is termed repetitive motion learning control. Using the filtered error approach of the section on adaptive and robust control, it is direct to derive the learning controller of Sadegh et al. for the robot arm in Equation (14.7.1).

Let $\ell = 1, 2, \ldots$ denote the iteration number of the trajectory repetition. Then, using information from the $(\ell - 1)st$ iteration, the controller for the ℓth iteration is given by

$$\begin{aligned} \tau_\ell &= \hat{f}_\ell + K_v r - v \\ v &= -\left(K_p e + K_s \|e\|^2 r\right) \\ \hat{f}_\ell &= \hat{f}_{\ell-1} + K_L r \end{aligned} \tag{14.7.28}$$

where the filtered error is $r = \dot{e} + \Lambda e$, $e = q_d - q$, with $q_d(t)$ the specified trajectory to be followed repeatedly. The gains K_v, K_p, K_s are positive diagonal design matrices, and K_L is a positive diagonal learning gain matrix. The function $\hat{f}_\ell$ is a learning term that uses its value on the previous iteration to improve on an estimate for a nonlinear function appearing in the error analysis.

Control of Flexible-Link and Flexible-Joint Robots

If the robot arm has flexible links, flexible joints, or fast motor dynamics, the control schemes just discussed must be modified. There are two basic cases to consider: *flexible-link robots* and flexible-joint robots; fast motor dynamics can be considered a special case of the latter.

Flexible-Link Robots

In the case of flexible-link robots, the links have significant vibratory modes that cannot be neglected. In this case one may perform an analysis using, for instance, the Bernoulli-Euler model, obtaining an infinite dimensional (partial differential equation) model, which can then be truncated to a finite dimensional (ordinary differential equation) model using, for instance, assumed mode shape techniques. The result is a model such as

$$\begin{aligned} M_{rr}\ddot{q}_r + M_{rf}\ddot{q}_f + V_{rr}\dot{q}_r + V_{rf}\dot{q}_f + F_r(\dot{q}_r) + G_r(q_r) &= B_r\tau \\ M_{fr}\ddot{q}_r + M_{ff}\ddot{q}_f + V_{fr}\dot{q}_r + V_{ff}\dot{q}_f + K_{ff}q_f &= B_f\tau \end{aligned} \tag{14.7.29}$$

which describes the coupling between the rigid modes $q_r(t)$ and the flexible modes $q_f(t)$. In these equations, the quantities M, V, F, G are defined basically as in Equation (14.7.1) and K_{ff} is a matrix of flexible mode stiffness constants.

The complete dynamics are now described by the vector $q = [q_r^T \; q_f^T]^T$. The control objective is to control the link-tip positions to follow a desired trajectory $q_d(t)$ while making small the flexible modes q_f. In the pinned-pinned modes shape method, for instance, the link-tip positions are given by $q_r(t)$. The major problem is that there are now *more degrees of freedom in* $q(t)$ *than control inputs available in* τ. This complicates the control problem greatly; however, a key property is that the matrix B_r is *nonsingular* in flexible-link manipulators.

It can be shown by using a singular perturbation approach, followed by the filtered error approach of the section on adaptive and robust control, that all the basic controllers just described can be used for flexible-link arms if an *additional inner control loop* is added for vibration management. That is, to the control toque $\tau(t)$ generated by Equation (14.7.24), is added the boundary-layer correction (fast) control term, manufactured by an inner loop, given by

$$u_F = -\frac{K_p}{\varepsilon^2}q_f - \frac{K_d}{\varepsilon}\dot{q}_f + \frac{K_p}{\varepsilon^2}\bar{q}_f \tag{14.7.30}$$

where ε is a small parameter (determined according to the time-scale separation imposed by the elements of the stiffness matrix K_{ff}). The slow manifold term $\bar{q}_f$ is a function of the slow control $\bar{u}$ (which is found as before), the variable $q(t)$, and some system parameters. It is possible to avoid measurements of the flexible mode rates $\dot{q}_f$.

Flexible-Joint Robots

The case of flexible-joint robots is in some ways the "dual" problem to that of flexible links. The dynamics of a robot arm driven by motors through rigid joints are given by Equation (14.7.10) for which the controllers described in previous subsections can be used. The dynamics of a robot arm driven by motors through joints with flexibility that is not negligible are given by

$$\begin{aligned} M\ddot{q}_r + V_m\dot{q}_r + F_r(\dot{q}_r) + G_r(q_r) &= K_J(q_f - q_r) \\ J_M\ddot{q}_f + B_M\dot{q}_f + F_M(\dot{q}_f) + K_J(q_f - q_r) &= v \end{aligned} \tag{14.7.31}$$

where $q_r(t)$ is the robot joint variable vector, $q_f(t)$ are the motor angles, and quantities are defined as per the discussions on Equations (14.7.1) and (14.7.10). It is assumed for simplicity that the gear ratio is $R = 1$. The stiffnesses of the joint motor couplings are on the diagonals of the joint stiffness matrix K_J.

The flexible-joint controller problem can be confronted using either a singular perturbation approach (work by M. Spong) or a *backstepping* approach. Using backstepping, it is found that the same basic structure of controller can be used as in Figure 14.7.30, but now the controller has multiple loops, with two adaptive systems required. The extra loop arises since the control input $v(t)$ controls directly the motor angles, which provide indirectly an input into the arm dynamics to control $q_r(t)$, the quantity of actual interest.

Force Control

In many industrial applications it is desired for the robot to exert a prescribed force normal to a given surface while following a prescribed motion trajectory tangential to the surface. This is the case in surface finishing etc. A hybrid position/force controller can be designed by extension of the principles just presented.

The robot dynamics with environmental contact can be described by

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) + \tau_d = \tau + J^T(q)\lambda \tag{14.7.32}$$

where $J(q)$ *is* a Jacobian matrix associated with the contact surface geometry and λ (the so-called "Lagrange multiplier") is a vector of contact forces exerted normal to the surface, described in coordinates relative to the surface.

The prescribed surface can be described by the geometric equation $\phi(y) = 0$, with $y = h(q)$ the Cartesian position of the end of the arm and $h(q)$ the kinematics transformation. The constraint Jacobian matrix $J(q) \equiv \partial\{\phi[h(q)]\}/\partial q$ describes the joint velocities when the arm moves on the surface; in fact, the normal velocity is $J(q)\,\dot{q} = 0$. According to the implicit function theorem, on the surface $\phi(q) = 0$ one may find a function $\gamma(\cdot)$ that $q_2 = \gamma(q_1)$, where the reduced variable $q_1(t)$ corresponds to motion in the plane of the surface and $q_2(t)$ represents dependent variables. The robot, constrained for motion along the surface, satisfies a *reduced-order* dynamics in terms of $q_1(t)$. Defining the extended Jacobian $L(q_1) \equiv [I^T\ \partial\gamma^T/\partial q_1]^T$, the relation of q_1 to the full joint variable q is given via $\dot{q} = L(q_1)\,\dot{q}_1$. For further details see McClamroch and Wang (1988) and Lewis et al. (1993).

The hybrid position/force control problem is to follow a prescribed motion trajectory $q_{1d}(t)$ tangential to the surface while exerting a prescribed contact force $\lambda_d(t)$ normal to the surface.

Define the filtered motion error $r_m = \dot{e}_m + \Lambda e_m$, where $e_m = q_{1d} - q_1$ represents the motion error in the plane of the surface and Λ is a positive diagonal design matrix. Define the force error as $\tilde{\lambda} = \lambda_d - \lambda$, where $\lambda(t)$ is the normal force measured in a coordinate frame attached to the surface. Then a hybrid position/force controller has the structure

$$\tau = \hat{f} + K_v L(q_1) r_m + J^T\left[\lambda_d + K_f\tilde{\lambda}\right] - v \tag{14.7.33}$$

This controller has the basic structure of Figure 14.7.11, but with an inner force control loop. In this controller, the nonlinear function estimate inner loop $\hat{f}$ and the robustifying term $v(t)$ can be selected using any of the techniques mentioned heretofore, including adaptive control, robust control, intelligent control, and so on. A simplified controller that may work in some applications is obtained by setting $\hat{f} = 0$, $v = 0$, and increasing the PD motion gain K_v and force gain K_f.

Teleoperation

In teleoperation, a human operator conducts a task, moving a master robot manipulator and thus defining motion and force commands. The master is connected through a communication channel to a slave robot manipulator in a remote location whose purpose is to mimic the master, thus performing the commanded motions and exerting the commanded forces on its environment. A typical teleoperation system is depicted in Figure 14.7.14. Task performance is enhanced if the human operator has information on the contact force being exerted by the slave manipulator. A convenient way of providing this information is to "reflect" the contact force to the motors on the master so the operator can feel a resistive force indicative of the contact force.

FIGURE 14.7.14 Typical robotic teleoperation system.

To focus on the issues peculiar to teleoperation, one can consider simplified dynamics for the master and slave arms given, respectively, by

$$M_m \dot{v}_m = F_h + \tau_m \tag{14.7.34}$$

$$M_s \dot{v}_s = -F_e + \tau_s \tag{14.7.35}$$

where the human operator input torque is F_h and the contact force exerted by the slave is F_e. In actual systems, one should include the nonlinear coriolis, centripetal, friction, and gravity terms (see Equation (14.7.1)), so that a preliminary feedback linearization (computed torque control) is needed to get the dynamics in this simplified form. Moreover, the Jacobians associated with the force inputs should also be considered (see Equation (14.7.32)).

The control problem is to provide motor control torques τ_m, τ_s so that the slave velocity $v_s = \dot{q}_s$ equals the commanded (master) velocity $v_m = \dot{q}_m$ and the environmental contact force F_e is proportional to the commanded force F_h (there could be a desired force amplification). In Figure 14.7.14, F_s *is* the sensed force resulting from the contact force F_e, the reflected force provided to the master robot is F_{md}, and v_{sd} is the desired velocity command for the slave. A straightforward control scheme for teleoperation is given by

$$\begin{aligned} \tau_m &= -K_m v_m - F_{md} \\ \tau_s &= -K_s v_s + F_s - \alpha_f F_e \end{aligned} \tag{14.7.36}$$

where K_m, K_s are positive master and slave control gains and α_f *is* a positive force gain. The selection of τ_s closes a local force control loop around the slave manipulator.

The key to successful control now lies in the appropriate definition of F_s, F_{md}, and v_{sd}. A naive definition of the sensed force is $F_s = F_e$, the contact force. It has been observed in experiments that this definition is unsuitable and results in instability. Therefore, a *coordinating torque* F_s is defined based on the slave velocity error $e_s(t) \equiv v_{sd}(t) - v_s(t)$ so that

$$\begin{aligned} \dot{\varepsilon} &= e_s = v_{sd} - v_s \\ F_s &= K_p e_s + K_i \varepsilon \end{aligned} \tag{14.7.37}$$

Though it may seem odd to define F_s in terms of the velocity error, it can be shown that, taking into account the impedance relationship of the environment, $F_e = Z_e v_s$, this definition makes the coordinating torque dependent on the contact force F_e. In fact, this definition results in the *passivity* of the slave dynamics referred to the variables (v_s, F_s).

Now, it can be shown that stable teleoperation results if one selects

$$\begin{aligned} F_{md} &= F_s \\ v_{sd} &= v_m \end{aligned} \tag{14.7.38}$$

Unfortunately, if there is any delay in the communications channel, this simple scheme is doomed to failure and results in unstable control. One technique for repairing this problem is to remove the force reflection and let the operator rely on transmitted visual information to infer the contact forces. In practical applications, this can result in the exertion of excessive forces that break tools and fixtures.

It has been shown in Anderson and Spong (1989) that if there is a time delay T in the communications channel, one may modify the controller as shown in Figure 14.7.15 to obtain stable teleoperation regardless of the magnitude of T. In this figure N is a positive scaling factor introduced since the force and velocity signals may differ by orders of magnitude. This modification makes all blocks in the diagram *strictly passive*, so that stability can be shown using circuit analysis techniques. The teleoperation controller with time delay compensation is given by the torques Equation (14.7.36), the coordinating torque in Equation (14.7.37), and the modified reflected force and slave velocity commands given by

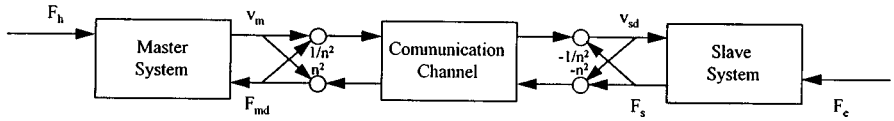


FIGURE 14.7.15 Passive robotic teleoperation system using active control.

$$\begin{aligned} F_{md}(t) &= F_s(t-T) + n^2\left[v_m(t) - v_{sd}(t-T)\right] \\ v_{sd}(t) &= v_m(t-T) + \frac{1}{n^2}\left[F_{md}(t-T) - F_s(t)\right] \end{aligned} \tag{14.7.39}$$

It is noted that in this modified controller, part of the reflected force is derived from the slave velocity error and part of the slave velocity command is derived from a force error term.

14.8 Planning and Intelligent Control

Chen Zhou

The previous section dealt with servo-level joint control of robot manipulators. This section deals with higher-level functions of planning and control, including generation of the prescribed joint trajectories that are required for servo-level control. Robots are designed to accomplish various tasks such as spot welding in assembly, part loading in material handling, or deburring in processing. A task consists of a series of complex motions of various joints and the end effector. The execution of robot tasks is controlled by robot programs. Robot programming can be classified into three levels: (1) joint level, (2) manipulator level, and (3) task level (see Leu, 1985). At the joint and manipulator levels, a task is decomposed into a set of explicit paths. The robot program is essentially a series of move statements to instruct the robot to pass through a sequence of paths. The programming at these two levels involves tedious specification of points, paths, and motions.

The difficulties involved at the first two levels of programming led to research and development for task level programming. At the task level, a robot program is a sequence of goals or objective states of the tasks, such as inserting a peg or deburring an edge. Due to the omission of explicit path and kinematic instructions by the programmer, the robot must know its configurations, its environment and the goal locations, such as the location of materials, as well as the obstacles within the envelope. The robot controller has to construct a set of collision-free paths that are optimized in terms of time, motion, or other control characteristics.

Task planning is a process in which the detailed motion control involved in a task (or a subtask) is determined by software or algorithms. The objectives can include proper grasp configurations, collision-free paths, minimum travel distance, minimum travel time, and avoidance of *singularities*. In a singular configuration, the robot may lose a degree of freedom or lose the ability to provide designed power. For complex tasks, task decomposition can be performed to provide more specific control of the robot. *Path planning* is a process of finding a continuous path from an initial robot configuration to a goal configuration without collision. It is a very important component in task planning.

Task planning includes several special cases. First, errors can occur during execution of a task. Various sensors have been developed to detect error conditions. Since the occurrence of an error is random, there is uncertainty associated with task planning when error and error recovery are concerned. Second, multiple robots are often used in robotic applications. The simplest case involves two arms. An important issue in two-arm task planning and control is the coordination of the two arms. Third, in robotic manufacturing cells, robots must coordinate with other equipment in the cell and the control of the robot can often affect the performance of the entire cell. Therefore, cell control is also discussed in this section. At the end of this section, we also mention artificial intelligence as applied to planning and man–machine interface.

Path Planning

Path planning involves finding a continuous path from an initial robot configuration C_{init} to a goal configuration C_{goal} without collision. Figure 14.8.1 illustrates an example of path planning in two-dimensional space. The small rectangular object represents a *mobile robot* or an end effector of a manipulator, and the other objects represent the obstacles within the working envelope. The dashed line shows the path of the center point of the robot. The four small rectangles show the orientation of the robot at the initial, goal, and two intermediate configurations.

Often, the collision-free path is not unique. In addition to avoiding collision, one can also add requirements for smoother motion, shorter traveling distance, shorter traveling time, or more clearance from the obstacles. Therefore, path planning can also involve optimization with respect to certain performance measures.

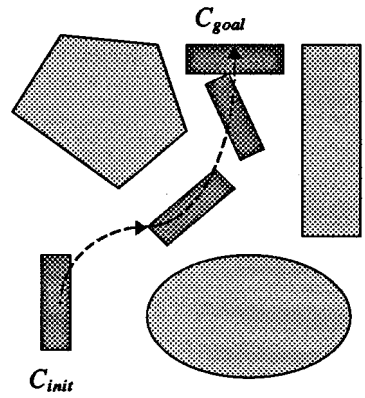


FIGURE 14.8.1 Illustration of path-planning problem.

The abstract model of the path-planning problem can take different forms depending on the application characteristics. It can be in two dimensions or in three dimensions. The concern can be the end effector alone or the entire robot arm. The end effector can be considered as a solid body or an infinitesimal point. Different considerations can have significant implications on the solution methodology and complexity. In this handbook, we will only discuss the simplest cases in which a point end effector in two-dimensional space is concerned. The readers are referred to Latombe (1991) for more complex procedures.

Road Map Approach Based on Visibility Graph

The road map approach is one of the earliest path-planning methods. The obstacles are modeled as polygons. A *visibility graph* is a nondirected graph. The nodes of the graph are the vertices of the polygons, the initial point and the goal point. The links of the graphs are straight-line segments that connect a pair of nodes without intersecting with any obstacles. A reduced visibility graph for the example is shown in Figure 14.8.2. A reduced visibility graph does not contain links that are dominated by other links in terms of distance. The elliptical obstacle is approximated by a hexagon. In the visibility graph, all the paths consisting of successive links that connect C_{init} to C_{goal} represent semicollision-free paths. The coarse line represents one of these paths. The use of the term "semicollision free" is due to the fact the path may actually contact an obstacle. It is clear that the path is not unique. In the example the possible paths can be $C_{init}\,AC_{goal}$, $C_{init}\,BC_{goal}$, or $C_{init}\,CD_{goal}$. Some offer shorter travel distances while others offer smoother paths. This method can be extended to include circular end effector and obstacles which have lines and arcs as boundaries.

Road Map Approach Based on Voronoi Diagram

For the same problem described above, one can create a *Voronoi diagram* based on the vertices and line segments of the obstacles and the working envelope and use this graph to generate a collision-free path. A Voronoi diagram is a diagram that consists of lines having equal distance from the adjacent objects. Obviously, the Voronoi diagram does not touch the obstacles and can provide collision-free paths. A Voronoi diagram in a polygonal space with polygonal obstacles is composed of straight line segments and parabolas. When both adjacent object segments are straight lines or vertices, the segment of the Voronoi diagram is a straight line. When one object segment is a point while the other is a line segment, the segment of Voronoi diagram is a parabola. Two additional links need to be created to connect the C_{init} and C_{goal} to the Voronoi diagram. Any set of links that connects C_{init} and C_{goal} through the diagram represents a collision-free path. Unlike the road map approach based on visibility graph, this approach tends to maximize the clearance between the robot and the obstacles. For the characteristics and creation of the Voronoi diagrams, the reader is referred to Okabe et al. (1992).

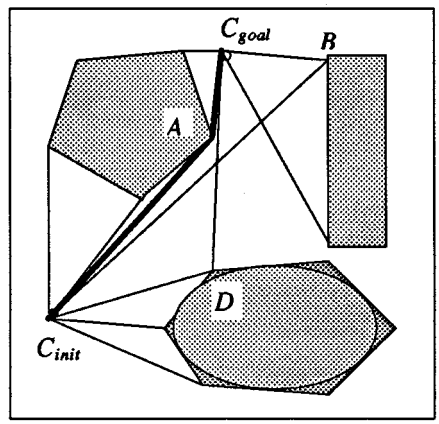


FIGURE 14.8.2 Road map method based on visibility graph.

Cell Decomposition Approach

In the *cell decomposition* approach, the robot free space is decomposed into simple connected geometric shapes, or cells such that a path can be generated between any two points within a cell. When Euclidean distance is used (as when using a Cartesian robot), convex polygonal cells satisfy this requirement. The simplest way to generate cells is the line sweeping method. An example in which the work envelope and the obstacles are polygons is shown in Figure 14.8.3. The two shaded areas are obstacles. In this example, the decomposition is done by sweeping a vertical line across the work envelope. A cell is formed whenever a vertex is encountered by the sweeping line. After decomposition, a *connectivity graph* is constructed. A connectivity graph is a nondirected graph. The nodes of the graph are the cells. If two cells share a common edge, they are connected and a link is drawn between the two nodes. The existence of a collision-free path can be found by searching the connectivity graph to see if there exists a path that connects two nodes containing C_{init} and C_{goal}. If such a path exists, one can construct collision-free paths by determining paths in the cells and connect the paths in adjacent cells.

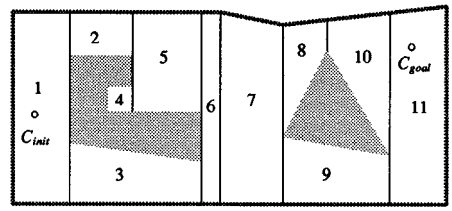
Apparently, the path and decomposition are not unique. One can select different paths and decompositions to optimize other measures. Figure 14.8.3(d) shows another possible decomposition of the same space.

Potential Field Approach

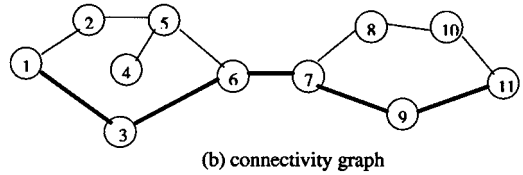
The idea of the *potential field* method is to represent the robot work space as a potential field which has peaks and a valley. The valley is at the goal configuration, while the peaks are at the location of the obstacles. The robot, represented by an article, will roll naturally away from the peaks and toward the valley in such a terrain. Mathematically, this is done by creating an artificial potential field with peaks as obstacles and a valley as the goal, and by using a search procedure to plan a descending path to the valley. The artificial potential field is composed of an attractive potential with its lowest point as the goal configuration, and a repulsive potential for each of the obstacles. An example is shown in Figure 14.8.4. The dimension of the robot envelope is 10 wide and 10 high. Here *ZG* is the attractive potential function, *ZR* is the repulsive potential function for all obstacles, and *Z* is the combined potential function. In this example, the attractive potential function *ZG* is a parabolic well:

$$ZG(x,y) = \frac{A}{2}\left[\left(x - G_x\right)^2 + \left(y - G_y\right)^2\right] \tag{14.8.1}$$

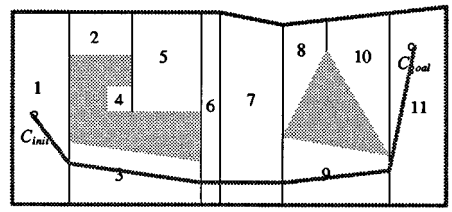
where *A* is a constant used to adjust the magnitude of the attractive field, G_x = 2 and G_y = 1 are the coordinates of C_{goal}. *ZR* represents three cylindrical obstacles with diameters 2, 1, and 1, respectively.



(a) cell decomposition



(b) connectivity graph



(c) a possible collision free path.

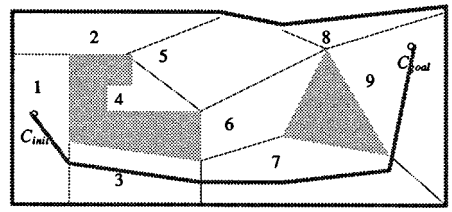


FIGURE 14.8.3 Cell decomposition method.

The center location of the three obstacles are at (2, 5), (7, 7), and (6, 3). Let ZR_i be the obstacle repulsive fields of i^{th} cylindrical obstacle. The repulsive function for i^{th} obstacle is

$$ZR_i(x,y) = \begin{cases} \dfrac{B_i}{2}\left[\dfrac{1}{\sqrt{\left(x-R_{x,i}\right)^2+\left(y-R_{y,i}\right)^2}} - C_i\right] & \text{if } \dfrac{D_i}{2} \ge \sqrt{\left(x-R_{x,i}\right)^2+\left(y-R_{y,i}\right)^2} < D_i \\ Z_i & \text{if } \sqrt{\left(x-R_{x,i}\right)^2+\left(y-R_{y,i}\right)^2} \le \dfrac{D_i}{2} \\ 0 & \text{otherwise} \end{cases} \tag{14.8.2}$$

where B_i is a constant for the adjustment of the height of i^{th} peak, D_i is the diameter of the i^{th} cylindrical obstacle, and $R_{x,i}$ and $R_{y,i}$ are the center coordinates of the cylinder. The characteristic of this function

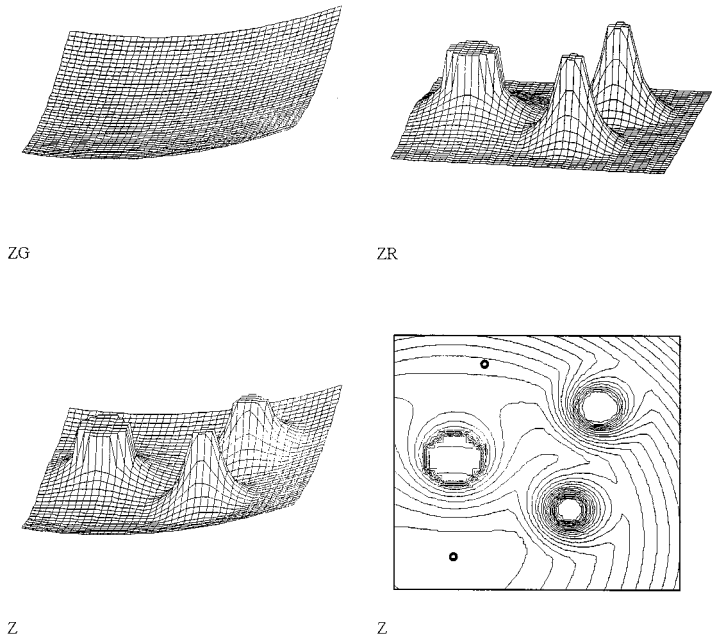


FIGURE 14.8.4 Potential field method example.

is that the potential at any point with distance more than twice the diameter of the closest obstacle is zero. Also, the potential inside the obstacle is a constant and equal to the potential at the boundary of the obstacle, where C_i is a constant to accomplish the former and the Z_i is a constant for the latter. The total potential is the sum of all the terms:

$$Z = ZG + \sum_{i=1}^{3} ZR_i \tag{14.8.3}$$

There are several techniques for potential guided path planning. The simplest is the depth first planning. In depth first planning, a prespecified step δ is predefined. The path will be found iteratively using

$$\begin{aligned} x_{n+1} &= x_n + \delta \frac{\partial Z(x_n, y_n)}{\partial x} \\ y_{n+1} &= y_n + \delta \frac{\partial Z(x_n, y_n)}{\partial y} \end{aligned} \tag{14.8.4}$$

where x_0 and y_0 are at C_{init}. Depth first planning is very fast for certain situations but may cause trapping some local point in others.

Error Detection and Recovery

In the execution of a task, errors can occur. The errors can be classified into several categories: hardware error, software error, and operational error. The hardware errors include errors in mechanical and electrical mechanisms of the robot, such as failure in the drive system or sensing system. Software errors can be bugs in the application program or control software. Timing with cooperative devices can also

be called software error. The operational errors are the errors in the robot environment that are external to the robot system such as jamming of parts or collision with obstacles.

The sensors used in error detection can be classified into several categories: tactile sensors for sensing contact and existence, proximity sensors for sensing location or possible collision, force/torque sensors for sensing collision and jamming, and vision for sensing location, orientation, and existence.

The occurrence of an error normally causes interruption of the normal task execution. Error recovery can be done at three levels. At the lowest level, the task is not resumable. Upon detection of an error, the task is interrupted automatically. The error must be corrected manually and the task must start again manually. At the second level, the task can be resumed. Upon detection of an error, the error can be corrected manually and the task can be continued from the point where the error occurred. At the third level, upon detection of an error, the error will be corrected automatically and the task execution is continued. Figure 14.8.5 shows an example of error and recovery in an insertion task.

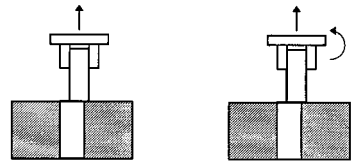


FIGURE 14.8.5 Jamming error detection and recovery.

In this insertion example, a peg is misaligned with the hole. The misalignment can be caused by wrong location of the fixture, wrong positioning of the peg before pickup, etc. The jamming error can be detected by a vertical force sensor in the gripper. At the lowest level, this can be used to trigger an emergency stop and an alarm. An operator can withdraw the gripper, remove the peg, and start the cycle again. At the second level, the operator can correct the problem and continue the insertion after the error condition is corrected. At the third level, additional sensory information is required. For example, additional force sensors can be used to identify the jamming torque. The torque information can be used to redirect the end effector to put the peg in the hole correctly. Since the occurrence of error is random in nature, the incorporation of error recovery introduces uncertainty into task planning. Artificial intelligence methods are often employed in error recovery.

Two-Arm Coordination

Many robotic applications require multiple arms, such as lifting heavy weights in handling or assembling two components that require simultaneous motion. In such applications, a special planning and control issue is the coordination of two arms. Figure 14.8.6 shows an example of two-arm application. In two-arm applications, two arms form a closed chain. Each arm acts as a constraint on the other and can have different contributions to the motion of the part. In terms of constraints, one or both arms can have rigid grasps. Two arms may also be controlled to remain in contact at a point, along a line, or on a surface. In terms of control, two-arm coordination can rely on a master/slave relationship, a push/pull relationship, or other relationships. In the master/slave relationship, one arm is controlled in force mode (master) while the other is controlled in position mode (slave). These constraints and control relationships require different controls from both controllers. Please see Hayati et al. (1989) for more discussion.

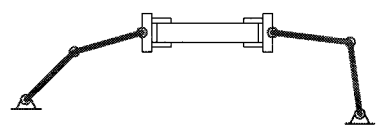


FIGURE 14.8.6 Two-arm coordination.

Workcell Control

Robots are used in many workcells for machine loading/unloading, assembly, part *kitting*, packaging, etc. The task planning associated with workcell control has its unique characteristics. A robot in a workcell often interacts with other resources in the cell such as a machine, a conveyor, a fixture, or a container. These interactions often require exact coordination. The coordination is done based on the clocks or interlocks implemented through *discrete inputs* (DI) and *discrete outputs* (DO). A DO is a binary signal that can be set in one of the two states and a DI is a binary sensing that can detect one of the two possible states from a DO.

In a flexible robotic manufacturing cell, alternative actions often exist for robot control, such as which assembly task to perform first, or which machine to serve first. The ordering of these different alternatives can directly affect the utilization, throughput, and other measures of the cell. Due to its discrete nature, the cell control optimization problem is combinatorial in nature and expands rapidly with problem size. As a result, dispatching rules in scheduling are often employed as rules in rule-based cell controllers.

Additional concerns in cell control relate to two commonly used terms in production system control: *blocking* and *deadlock* (or *locking*). Blocking is a condition in which material cannot be transported to its next location because of a temporary resource unavailability. This can cause the waste of capacity of the current machine. Deadlock is a condition in which two resources mutually require the service of the other but neither can provide the required service at the current state. Therefore, the system will be deadlocked. Examples of blocking and locking are given in Figure 14.8.7.

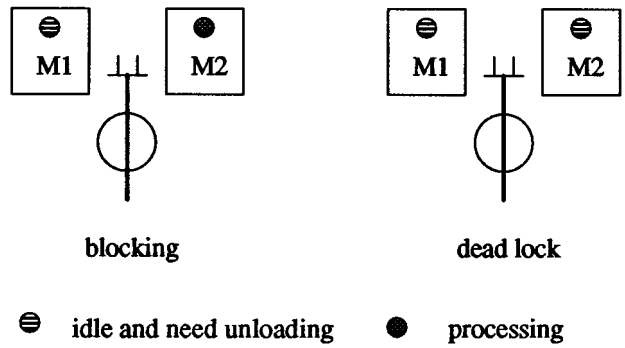


FIGURE 14.8.7 Example of blocking and locking.

In the figure on the left, the part on M1 needs to be transported to M2. However, M2 is in processing state and therefore the part in M1 has to wait for its completion. In the figure on the right, both parts on M1 and M2 are finished and need to be transported to the other machine. Suppose the robot has a single end effector; it is not able to transport either part to the other machine. One possible solution to blocking and deadlock problems is to add buffers. In the example, if a buffer location is added to the cell, the blocking and deadlock can be temporarily resolved. However, the addition of buffers to a cell can increase the cost of the cell and the complexity of the control. The real-time solution to blocking and deadlock problems lies in the identification of possible blocking or deadlocks and prevents them from occurring. Blocking or deadlock avoidance and system optimization are combinatorial in nature. The number of states expands exponentially with the number of states for each resource and number of part types. Therefore, rule-based systems and AI techniques find wide acceptance in cell controls.

Planning and Artificial Intelligence

Artificial intelligence (AI) is a branch of computer science studying the characteristics associated with human intelligence such as reasoning, acquiring knowledge, applying knowledge, and perceiving the environment. Path or task planning is the application of reasoning, knowledge acquisition, and perception

of the environment. Therefore, robot planning is closely related to the study of AI and certain AI techniques can be applied to robot planning and control. Some of the areas of research in AI that apply to robotic planning are problem solving, expert systems, learning, and machine vision.

The methodologies in path planning can be considered as problem solving. In assembly, when multiple components are assembled with precedence requirements, AI techniques can also be applied. Expert systems or rule-based systems solve problems in a discrete space domain based on rules derived from experts or other sources. Finally, machine vision has enjoyed a rapid increase in robotic applications. Machine vision can acquire complex environment information rapidly. Various algorithms can be used to extract useful path planning information such as locations of obstacles, end effectors, tools, etc. and can be used in real-time robot motion control. The reader is referred to Winston (1984) for more AI discussion.

Man-Machine Interface

Robots can commonly be programmed or controlled through teach pendants or a computer. A teach pendant is a small key pad that allows the user to move the robot, record positions, and enter simple programs. Modern robots are also accompanied by programming and control software that runs in microcomputers or workstations. The software environment often includes an editor, menu-driven robot control, and diagnostic utilities. More intelligent robot control programming is commonly supported in this environment than is available through the teach pendant.

Control programs can also be generated off-line. In *off-line programming*, the spatial configuration of the robot and work environment is modeled in the computer. A programmer is presented with a 2D or 3D world model of the robot and its environment graphically. The programmer will specify the locations and paths in this model rather than working with a real robot. Off-line programming has the potential to improve robot productivity and simplify the procedures of creating complex robot programs.

14.9 Design of Robotic Systems

Kok-Meng Lee

For manufacturing in which the manufacturing facility is concerned with similar volumes of production and a wider range of parts, the assembly line/mass production method is often not cost effective. It is often desirable to group equipment units together into workcells that can, in composite, perform an entire family of related operations on the product. The work-in-progress enters the workcell, remains while several functions are performed, and then leaves the workcell.

The individual equipment units that are used in the workcell (for both processing and materials handling) can consist of combinations of manual, semiautomatic, and fully automated equipment. However, in this section, the term "workcell" refers to a grouping of the robot and its peripheral equipment to assemble any of a large variety of products with little or no human intervention, driven by electronically designed data. An assembly robot is a comparatively simple mechanism whose function is to position parts and tools in the space of its work volume accurately. It is a comparatively low-cost machine of high precision of positioning and great reliability. Its simplicity, however, excludes the possibility of human-type actions like form recognition and its prehensile tools are very far from having the number of degrees of liberty a human hand has. If we concede that an assembly robot can by no means compete with a human being in a complex task, we also have to acknowledge that an assembly robot is capable of executing monotonous tasks with consistently high precision, thereby increasing the quality of the product. It can also keep up a fast production line indefinitely. Recognizing this difference between a human and a robot is essential in the design of a robotic system.

The remainder of the section is organized as follows. A set of design considerations for designing an assembly robot workcell is first presented. Layouts for a typical robotic workcell are then discussed. Experience so far has shown that in most instances, it is a feeder that fails in the workcell, not the robot. Feeding methods must be carefully considered when designing a workcell and are discussed at the end of the section.

Workcell Design and Layout

Design Considerations

Assembly systems can be broadly classified as manual, fixed, and flexible systems in relation to the complexity of the product to be assembled and to the production volume as shown in Figure 14.9.1. Flexible robotic workcells are typically used for less complex products at low or medium production volume, while for increasing product complexity, the cells designed for a single-purpose task can be linked into assembly lines. Apart from product volume and complexity, the design of the robotic workcell depends on several factors: namely, number of part types, end-of-arm tooling exchange, as well as product design.

Number of Part Types. A typical workcell consists of a robot and its peripherals made up of part-presentation mechanisms, feeders, conveyor, and end-of-arm tooling. For a small number of part types, parts are presented to the robot by feeders or magazines. As these take up space, only a limited number of different parts can be fed to one robot. In mechanical assembly normally a maximum of five to six different parts can be presented in this way.

To extend the robot's accessibility to a large number of parts, mechanized component feeding systems can be mounted on data-driven carousel conveyors spaced around the robot, each with a fixed dispensing point within reach of the robot gripper. The carousel can accommodate up to several hundred positions onto which magazines, tapes, or other modular dispensing systems can be attached. With multiple programmable carousels, the robot can access several thousand different parts. The application of the mechanized carousel is useful when only a few of each part type from thousands of styles may be used. Other alternatives are (1) kitting, in which all components to be assembled are kitted in a loosely

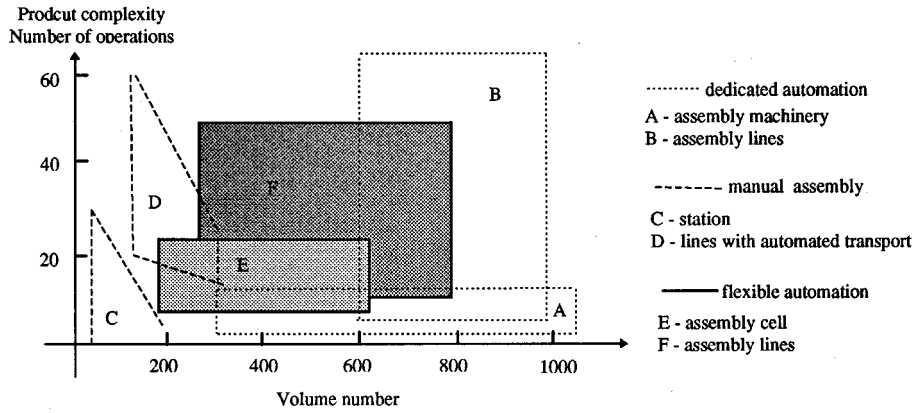


FIGURE 14.9.1 Classification of assembly systems.

palletized waffle pack, and followed by more accurate location using standard machine vision; and (2) the use of accurate totes for robot handling.

End-of-Arm Tooling Exchange. Many systems use different gripper exchange systems in order to cope with different parts. Tool exchanges are often considered as "nonproductive" since they do not contribute to assembly operations. The exchange is serially coupled to the assembly operations. This means that the cycle time increases due to the extra time needed for pickup and drop-off for tool changes as well as travel time between the assembly point and the end-of-arm tooling station. In order to reduce time loss due to the gripper, exchange should be minimized and/or in parallel with other activities, and the distance between pick-up point and assembly point should be very short. This problem could be avoided if a fast-revolving gripper head is used provided that space, weight, and cost of the revolving head do not pose a problem. Alternatively, the pallet carries batch-specific equipment such as grippers, fixtures, and end-of-arm tooling and can be presented to the robot on a conveyer in a similar fashion as the parts.

Product Design. Product design for flexible automation cells includes the following criteria: task operations based on flexible assembly cells for specific product families which must be able to assemble the variants of these product families using programming, fast changeover from one product to another within a flexible assembly cell, and reuse of standard elements for new assembly tasks

In addition to product complexity and volume, two other criteria should be considered in the construction of flexible assembly cells. First, since only a few products are generally suitable for fully automatic assembly, manual working processes are often essential with a large number of products. Flexible assembly cells must be constructed so that manual work stations can be included following ergonomic principles. Second, since the type-specific peripheral costs will increase in relation to the number of individual parts in the product to be assembled, part-specific feeders must be minimized for the economic use of flexible assembly cells.

Workcell Layout

Workcell design and layout in a flexible automation system depend on the nature of the manufacturing processes, the product design, and the material handling system as a whole. The manufacturing systems are classified as electronic product assembly, subassembly of electrical and mechanical components, and kitting cell for large-scale manufacturing.

Electronic Product Assembly. Flexible workcells are commonly used for the assembly of integrated circuit boards (PCB), where a combination of interchangeable part-feeding mechanisms are used to present parts to robots. Since a majority of the processes involved are carried out in the linear, vertical plane, robots of SCARA or gantry construction are best suited for these assembly tasks. The workcell

consists of a robot and its peripherals made up of part-presentation mechanisms, feeders, conveyor, and end-of-arm tooling.

Figure 14.9.2 shows the organization of a typical workcell for assembly of a family of circuit boards (Decelle, 1988), which is a part of the in-line component insertion, inspection, and repair assembly line. Circuit boards to be assembled are secured on panels and flow through the workcell on a conveyor. Each of the circuit boards is characterized by a bar-coded serial number that permits product tracking, data collection, and testing through the assembly. Boards requiring assembly are positioned over an elevator mechanism in the workspace of the robot. The mechanism lifts the board slightly and uses the tooling holes on the panel to locate the circuit board. Two digital signals interface the conveyor to the workcell — one signals the robot that the board is ready for assembly and the other signals the conveyor to index the board to the next workcell. Components are fed to the robot by using feeders. The feeder singles out components to a walking-beam mechanism that transfers parts through lead-cutting, lead-straightening, and lead-verification operations and on to the lead-locating nest for robot pickup. The activities of the robot and its peripherals in a workcell are coordinated by a host computer. The workcell is set up and monitored through the host computer. Through the host computer, the operator provides the workcell the code to be assembled, the components in the feeders, and the configuration of the feeders.

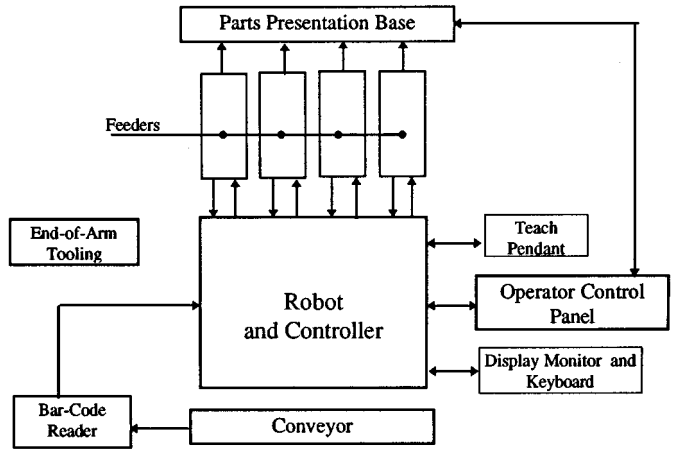


FIGURE 14.9.2 Workcell for elctronic assembly.

Subassembly of Electrical and Mechanical Components. Unlike PCB assembly, packaging and designs of small electrical and mechanical components are generally nonstandardized. Thus, the problem of automated flexible assembly workcells lies in the presentation of parts and the degree of flexibility of assembly of small components often involves both product design and layout considerations extensively.

Figure 14.9.3 shows a self-contained flexible workcell for assembly of small mechanical parts with a circular indexing table. Modular part-feeding equipment such as vibrator feeder bowls and special-purpose trays are placed around the indexing table to feed and to orient small components to the robot. Typical mechanical operations such as riveting, screwing, welding, inserting, pressing, and so on are achieved through quick changeover end-of-arm tooling. The circular indexing table arrangement is advantageous where end-of-arm tooling changes are necessary for handling different parts. It allows changes of end-of-arm tooling to take place while other operations are continuing.

With complex products, assembly in a single flexible assembly cell is not always feasible. In this case, a flexible assembly line can be designed to link self-contained independent workcells (Figure 14.9.3) so that they can be engaged or disengaged as required to allow adaptability in connection with product model change. Alternatively, standard carriers or pallets can be used to present a large number of different part-types to a robot. Each pallet carries a large number of identical parts, unoriented but

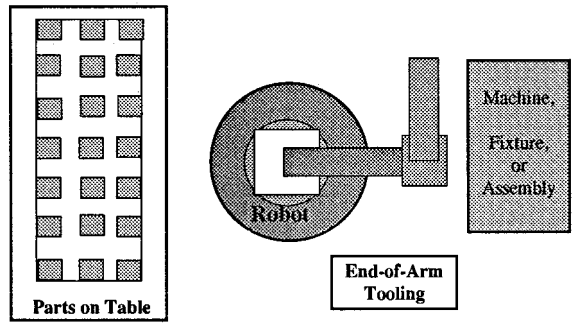


FIGURE 14.9.3 Typical single-purpose workcell.

with the right side up, and placed on a flat board. Standard machine vision was used to detect the orientation of the parts.

Kitting Cell for Large-Scale Manufacturing. In the field of large-scale manufacturing such as automobile manufacturing, engine assembly, and machining processes, where the setup time of specialized tools for each task is excessive, the work is generally distributed into several cycle zones. As an example, actual cutting time (production time) represents a value between 5 and 20% of average machine utilization time that includes nonproductive time accountable by workpiece load/unload, too) change/setting, and workpiece inspect.

To avoid a high level of wear and tear on tools due to constant conversion, the cycle zone is commonly divided into individual operating cells which may be interconnected in series, parallel, or a combination of series and parallel. A typical workcell (Figure 14.9.3) consists of a robot, a part-feeder, an end-of-arm tooling section, and the manufacturing process. The parts are contained in a regularly spaced pallet, which are transported by means of an automated guided vehicle (AGV) or a conveyor to the loading tables and are fed to process by the robot. The most common approach in automated part presentation for machine loading is the use of specially designed pallets for each part family to maintain sufficient position accuracy for a completely preprogrammed robot picking.

In the case of assembly, purchased parts or parts to be processed are kitted onto one kit tray in a single location. Kitting is the process of taking parts from bulk and placing them on a kit tray, which is an organized group of parts. Concentrating the material delivery system and its control to one area is the main benefit of the kitting cell. In addition to efficient use of floor space by eliminating duplicate equipment at each assembly cell, the feeders and tooling are universal — the same equipment is being used all the time for all parts, thus maximizing utilization while minimizing capital expense. The material delivery equipment is eliminated at the assembly cycle times. Also, having all the parts for an assembly on a carrier permits changes in the process route during machine downtime or blockages.

Figure 14.9.4 shows a layout of the kitting cell. An overhead gantry takes bins of parts and dumps them in the appropriate feeders (indicated in Figure 14.9.4 as F1, … F7). The feeders fill the lanes with an initial quantity and replenish them as parts are kitted. The parts come to rest in nests at the end of the feeder lanes. Here the vision system verifies the correct part family, performs some quality checks, and determines the position and orientation for the robot to pick the parts. Should the vision reject the part, the nest will dump the part and a new part will be fed in for an inspection. Using a quick changeover gripper, multiple parts are kitted onto a tray. Once all the parts are on the kit tray, the tray is indexed to the inspection station for verification that all parts are placed. The robot takes the completed kit tray and places it on the assembly conveyor to an idle station, ready to be picked up by an AGV.

Part-Feeding and Transfers

The term “part-feeding” refers here to feeding workpieces from pallets using a preprogrammed robot for subsequent processes such as machining or assembly. The cost to feed parts to a robot for either

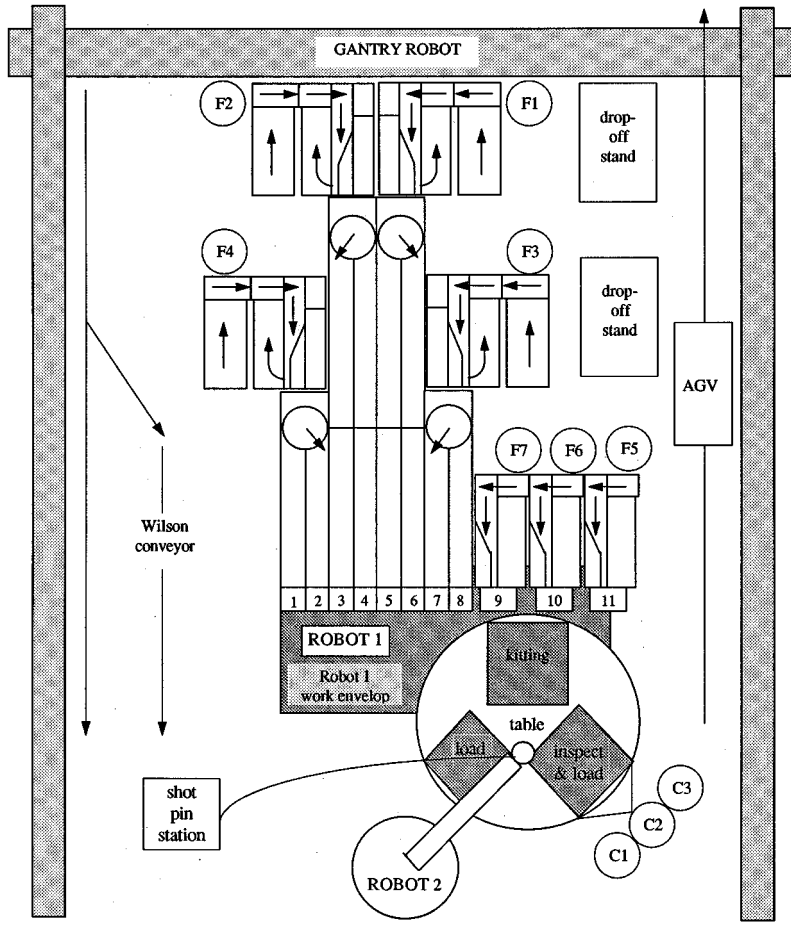


FIGURE 14.9.4 Schematics of the kitting cell.

machine loading or assembly in a flexible manufacturing system (FMS) has often been underestimated, which may comprise as high as two thirds of the overall investment and is usually the source of a large percentage of work stoppages and defects. A general review of existing *mechanical part feeders* can be found in Lee (1991).

The basic kinds of part-feeding may be classified as follows: (1) mechanical feeders which are designed to feed and to orient the parts-dedicated part-feeding apparatus, (2) dimensionally dedicated pallets which are specially designed for each part family to maintain the position/orientation, and (3) machine vision.

Mechanical Feeders

The commonly used mechanical feeders for robotic assembly are bowl feeders, vibratory feeders, and programmable belt feeders. For large volume manufacturing, the employment of the dedicated mechanical part-feeding apparatus may be justified. However, mechanical feeders consume a lot of room around the workcell, often fail due to jamming, and, most significantly, generally require retooling when a component is changed or tool wear is caused by jamming.

Vibratory Bowl Feeders. Vibratory bowl feeders (Boothroyd and Dewhurst, 1985) are most commonly used as mechanical feeders for robotic assembly. The basic component of a bowl feeder consists of a

vibratory bowl, a rotating disk, and an orienting track. Parts to be fed to the robot are separated into a single line and oriented to move to feeding end. These feeders, in general, are not designed to be easily converted to feed new part types. The cost of the bowl feeders can be broadly divided into two parts: special purpose equipment cost and general purpose equipment cost. Typically, changeover would involve replacing the bowl, orientation track, feed track, and escapement, which contribute to special-purpose equipment cost. Only the vibratory drive unit could be reused. This general-purpose portion of the feeder is approximately 30% of the feeder cost.

One way to lower the cost of the bowl feeder per part is to deliver different parts to a robot assembly station using multiple layer vibratory bowl feeders. A multilayer bowl feeder has several bowls mounted in stacked fashion, and in each bowl a different kind of part is stored. The design of multipart vibratory feeders aims at reducing the cost of the vibratory feeders by sharing the general-purpose hardware cost over several parts and by reducing the special-purpose tooling cost. Two basic forms are available: bowl type and in-line type.

To change over this multipart vibratory feeder to other part types, the orienting tracks must be replaced. An effective way to reduce wear is to separate the function of orienting from feeding. The function of the multilayer vibratory feeders is to restrict feeding parts to a separation unit. Parts of several different types are fed but not oriented from a vibratory feeder. In most cases, the workpieces must be held by a mechanical pusher against a pair of orthogonal datum planes on a relatively flat surface with the "right side up." A machine vision system is then used to locate and/or to sort the orientation of the parts using two-dimensional binary images, which is a great deal easier to store and to process.

Vibratory Belt Feeders. In vibratory belt feeders, parts are fed by a vibratory conveyer belt (Boothroyd and Dewhurst, 1985). The principle of the vibratory belt feeder is to produce a vibratory motion on the surface of the brushplate. The motion is obtained by pulling the brushplate sharply down and back and then allowing it to spring up and forward. This action, repeated at high speeds (approximately 3600 times per minute at 60 Hz power supply), produces a definite vibrating movement on the brushplate surface, permitting parts to be conveyed in a smooth and easily controlled manner.

The orienting systems used on these belt feeders may be a mechanical device, an optical sensor, or a vision system. A machine vision system is often used to locate and/or to sort the orientation of the parts using two-dimensional binary images. A line-scan camera is commonly used to create the silhouettes of the workpieces, and in some cases the product designs can be reviewed to simplify the vision algorithm and to reduce the system cost. Since a robot gripper can grasp parts from a queue on the feeder itself, belt feeders do not require any special-purpose tooling for feed track or escapement and thus offer several advantages over the vibratory bowl feeder for robotic assembly

Dimensionally Specific Pallets

One of the most common approaches is the use of specially designed trays or totes for each part family to maintain sufficient accuracy for a completely preprogrammed robot picking. A particular form of these dimensionally precise feeders is known as tape-and-reel for feeding parts of relatively small sizes, which can be placed on tapes of standard width. For some devices that are large, heavy, ceramic, or have fragile leads, tapes are very expensive and impractical.

In general, the dimensionally specific pallets are well suited for large volume production where changes of part types are not frequent. The operational cost of the design-specific pallets includes packaging costs for transport, construction cost for pallet alignment, and engineering cost for new pallet designs.

Vision-Based Flexible Part-Feeding

For flexible manufacturing, where a large variety of product sizes and component types are encountered, the part-feeding system must have the ability to adapt to a changing product design without costly hardware redesign or time-consuming software reengineering. This need has been addressed as a general industrial vision-based bin-picking problem by several authors.

In manufacturing automation applications, the processing speed of acquiring and analyzing an image must be comparable to the speed of execution of the specific task. The attempt to duplicate human

perception by obtaining a three-dimensional detailed image of the part often calls for time-consuming computation and does not necessarily determine the location and orientation of a given part with the accuracy required for successful part acquisition by the robot. Moderate location inaccuracies pose no difficulty for human operators since they use vision, hand-eye coordination, and sense of touch to locate and correctly load the part.

However, if the orientation of the parts can be characterized by the two-dimensional object silhouette, retroreflective materials can be used as a background in generic part presentation (Lee and Li, 1991). Most surfaces on objects exhibit a combination of diffuse and specular reflection. A point on an ideal diffuse-reflecting surface appears equally bright from all viewing directions. Surfaces covered with papers and matte paints may be considered as reasonable approximations. An ideal specular reflector is one that reflects any incident light ray as a single ray in the same plane as the incident ray and the surface normal. The basic principle of the retroreflective vision sensing is to structure the surface reflectance of the pallet or the landmarks so that it is much brighter than objects generally characterized by diffuse or specular surfaces. In practice, a number of nonpredictable factors such as measurement noise, the uniformity of the surface reflectivity, and the uniformity of illumination, which occur on both the object and the background, can be eliminated by a relatively simple technique. If part design can be modified, brightly illuminated retroreflective landmarks can be intentionally created on objects for location tracking. Low cost landmarks could be incorporated in design by using retroreflective liquid paints on existing features. Alternatively, generic landmarks can be constructed by applying solid glass beads on the reflected surface of standard fastening devices such as screw heads.

14.10 Robot Manufacturing Applications

John W. Priest and G. T. Stevens, Jr.

Product Design for Robot Automation

Identifying automation opportunities early in product design is important because product design requirements to facilitate robotic manufacturing are often unique and must be integrated early in the product design process. Some overall manufacturing problems for using robots and some design solutions to resolve these problems are listed in Table 14.10.1.

TABLE 14.10.1 Design Solutions for Robots

Problems in Utilizing Robotics	Design Solutions to Assist Production
Location accuracy and repeatability	Design for vertical assembly; use chamfered edges for mating surfaces; tolerance leeway for mating parts
Part feeding and orientation	Design parts which can be easily fed, provide notches, guide pins, or slots for part orientation; select parts from vendors that will deliver in easy-to-feed packaging
Programming robot and associated equipment	Design simplification; use common parts for different products, part reductions; part families
Application problems with fasteners (screws, washers, and nuts)	Minimize the use of all fasteners; utilize snap fits where possible
Downtime caused by jams and misfeeds due to poor part quality	Select vendors that produce high-quality parts

Rossi highlighted the product designer's role in robotics stating this problem (Rossi, 1985):

> Often designs are made in such a fashion that one cannot access a certain area with a robot. Humans can get around obstacles and operate within those designs easily, but robots cannot because they are not quite as flexible as human beings. I think that this is the single most important item that has kept us from being further along than we are. What happens is that users try to apply a robot to something that's been designed without robotic assembly in mind. They usually run into a problem. Either the robot cannot handle it at all or the users find that they have got to put a lot of additional engineering design into a particular workcell, or perhaps into an end effector, in order to get around the problem. All this does is add to the price tag, and cost is very much in consideration when one is trying to sell these systems. A situation arises where robots are no longer attractive because of all the additional things that need to be done.

In summary, the product must be designed for the manufacturing process and the robot. For more information, the reader should review Boothroyd (1994), Bralla (1986), Priest (1988), and Tanner (1994). Table 14.10.2 shows some design rules for robotic assembly.

TABLE 14.10.2 Design Rules for Robotic Assembly

Product should have a base part on which to build assemblies in a top-down, straight-line motion direction
Base should be stable and facilitate orientation
Parts should be able to be added in layers
Use guide pins, chamfers, and tapers to simplify and self-align the layering of parts
All parts should accommodate handling by a single gripper and be comparable with popular feeding methods
Sufficient access is available for the gripper
Avoid the use of bolt-and-nut assembly
Parts should be able to be pushed or snapped together; when screws are necessary for repair, they should all be the same size
High quality parts are used
Vendors deliver parts that are compatible with the selected part feeder mechanism

Economic Analysis

Economic analyses for robotic applications are similar to those for any manufacturing equipment purchase and usually use minimum annual revenue requirements, present value methods, or break-even analyses. Since robots are a flexible method of automation, a unique aspect of robotics is manufacturing's ability to reuse the robot after its initial production run for other applications in later years. For many companies, this subsequent use of the robot can be shown in the economic evaluation. Some other unique benefits in robotic economic analysis that may be included are improved quality, higher precision, ability to run longer shifts, and reduced floor space. Unfortunately, some unique disadvantages of robot analysis are software integration complexity, inability to respond quickly to product design changes, and process reliability.

In general, there are several situations where robots are more likely to make economic sense. These are

A. Sufficient volume to spread investment costs over many units
   1. High volume
   2. Stable product design
   3. Multishift operations
B. Robot is used on more than one product
   1. Limited number of different products on same production line
C. Part handling problems occur when performed manually
   1. Parts that are very large, heavy, or bulky
   2. Parts that are very fragile or easily damaged
   3. Parts that are extremely small
D. Extremely difficult manufacturing process without using robot or automation
   1. Many processes, especially in electronics, cannot be performed without robots or some type of automation
E. Safety and health concerns of process
   1. Safety and health costs can be significant

The type of data concerning the robot system that is required for an economic analysis is shown in Table 14.10.3.

TABLE 14.10.3 Economic Cost and Savings for Robot Applications

Investment costs
Robot purchase price — for many applications this is a much smaller part of the costs than expected (25 to 45%)
Other equipment (part feeders, conveyors) — this includes the cost of hardware interfaces
Design of end effector, special fixtures, and other equipment — most applications require the design of a unique end effector and special fixtures
Software design and integration — can be a much higher cost than expected due to the complexity of interfacing different equipment controllers
Installation including facility modifications — usually a small cost for robot system
Technical risk — this is the risk of whether the system will perform up to the specifications in areas such as performance, quality, precision, etc.
Operating costs
Training — costs of training operators, engineering and maintenance personnel
Product design changes — cost required to modify the robotic software and hardware when design changes or modifications are made to the product
Operating, utilities, and maintenance — typical costs found for most manufacturing equipment
Savings
Direct labor — labor savings caused by the robotic system
Ergonomic and health — benefits of lower number of job injuries, workers compensation costs, and compliance with OSHA regulations
Quality — improved quality may result due to lower scrap and waste
Precision — robots can often perform tasks at a much higher precision (i.e., lower variability) than manual operations resulting in fewer defects and better product performance

Cost Justification for Robots

In this section an example of a robot justification study is presented. This example uses the discounted cash flow method resulting in the calculation of a rate of return (often referred to as the internal rate of return).

The rate of return, R, is defined by Equation (14.10.1) as

$$0 = \sum_{j=0}^{n} \frac{X_j}{(1+R)^j} = \sum_{j=0}^{n} X_j (P/FR, j) \tag{14.10.1}$$

where

X_j = the net total cash flow for year j
n = number of years of cash flow

Basically, the rate of return, R, is the interest rate that makes the sum of the discounted cash flows equal zero.

The net cash flows, X_j, in Equation (14.10.1) can be defined by Equation (14.10.2):

$$X_j = (G - C)_j - (G - C - D)_j (T) - K + L_j \tag{14.10.2}$$

where

G_j = gross income (savings, revenues) for year j
C_j = the total costs for year j exclusive of book (company depreciation and debt interest)
D_j = tax depreciation for year j
T = tax rate (assumed constant)
K = total installed cost of the project (capital expenditure)
L = salvage value in year j

A numerical example is now presented using the data given in Table 14.10.4. The values in Table 14.10.4 should not be considered representative. They are used only for example purposes.

TABLE 14.10.4 Data for Numerical Example

Project costs	
Purchase price of robot	= $40,000
Cost of end effector	= $10,000
Software integration	= $20,000
Cost of part feeders	= $10,000
Installation cost	= $ 4,000
	$84,000
Yearly operating costs	= $10,000
Yearly savings	
Labor	= $60,000
Quality	= $10,000
Tax rate	= 40%

The first question that must be answered is, what is the capital recovery period (life of the economic study)? In this example 3 years is used. The next question is, what is the yearly tax depreciation? This example uses MACRS (3 years). Therefore, the percentages used to determine the yearly tax depreciation amounts are those given in Table 14.10.5. It should be noted that Table 14.10.5 implies there are 4 years of tax depreciation. This is because the MACRS system uses a half-year depreciation convention. In this example, it is assumed there is sufficient taxable income from other operations that allow the use of the depreciation amount in year 4. In this example, the salvage value is assumed to be zero.

TABLE 14.10.5 MACRS Percentages

Year	Percentage
1	33.33
2	44.45
3	14.81
4	7.42

Using Equation (14.10.2), it is now possible to generate the cash flows shown in Table 14.10.6.

TABLE 14.10.6 Net Cash Flows

EOY	K&L	G	C	D	X
0	$84,000	—	—	—	84,000
1		$70,000	$10,000	$27,997	47,199
2		70,000	10,000	37,338	50,976
3		70,000	10,000	12,440	40,976
4	L = 0	—	—	6,224	2,490

Some sample calculations follow:

$$x_0 = -\$84,000$$

$$x_1 = (70,000 - 10,000) - (70,000 - 10,000 - 27,997)(.40)$$

$$= \$47,199$$

$$x_4 = -(-6224)(.40)$$

$$= \$2,490$$

With the cash flows in Table 14.10.6 the rate of return can be determined using Equation (14.10.1):

$$0 = -84,000 + 47,199(P/FR,1) + 50,935(P/FR,2) + 40,976(P/FR,3) + 2,490(P/FR,4) \qquad (14.10.3)$$

To determine R in Equation (14.10.3), a trial-and-error solution is required. Assuming 20%, the right-hand side of Equation (14.10.3) gives $15,619 and with 25%, it is $–17,262. Therefore, the rate of return, using linear interpolation, is

$$\begin{array}{cc} 20\% & \$15,619 \\ R & 0 \\ 25\% & -17,262 \end{array}$$

$$R = 20 + \frac{15,619}{32,881}(5)$$

$$= 22.38\%$$

This rate of return (22.38%) is now compared to a minimum acceptable (attractive) rate of return (MARR). If R ≥ MARR, the project is acceptable. Otherwise, it is unacceptable. It is pointed out that the definitions of cash flow and MARR are not independent. Also, the omission of debt interest in Equation (14.10.2) does not, necessarily, imply that the initial project cost (capital expenditure) is not

being financed by some combination of debt and equity capital. When total cash flows are used, the debt interest is included in the definition of MARR as shown in Equation (14.10.4).

$$\text{MARR} = k_e(1-c) + k_d(1-T)c \tag{14.10.4}$$

where

k_e = required return for equity capital
k_d = required return for debt capital
T = tax rate
c = debt ratio of "pool of capital" used for current capital expenditures

In practice, it is not uncommon to adjust (increase) k_e and k_d for project risk and uncertainties in the economic climate. There are other definitions of cash flow definitions (equity and operating) with corresponding MARR definitions. A complete discussion of the relationship between cash flow and MARR definitions is given in Stevens (1994).

Assembly

Assembly is projected to be the largest area of growth for robots. Key design goals for robotic assembly are to ensure high-quality parts, minimize the use of fasteners and cables, and provide accessibility so that parts can be easily fed and oriented by automated equipment. Designing to facilitate the use of robotics requires a review of their capabilities. Although assembly robots are often shown as stand-alone equipment, they require considerable amounts of support tooling and auxiliary equipment. These include part feeders, end effectors, special fixturing, and a material handling system. Except in the case of robots with vision or special sensors, parts with which the robot will interact must be precisely located and oriented. This may require additional tooling or special vendor packaging.

Assembly is defined as the combining of two parts into one entity. This combining process may include (1) the use of mechanical fasteners (i.e., screws, snap fits, rivets, etc.); (2) joining processes such as welding, brazing, soldering, etc.; (3) application of adhesives; (4) the simple process of placing two parts together to be joined together later.

Robotic assembly is the use of robots to perform one of these assembly processes. A typical set of tasks for robotic assembly using mechanical fasteners might be

1. Go to location (x_1, y_1, z_1) and grasp part A (assumed to be properly positioned and oriented).
2. Place part A in a fixtured assembly position (x_2, y_2, z_2, including proper orientation).
3. Go to location (x_3, y_3, z_3) and grasp part B (assumed to be properly positioned and oriented).
4. Place part B on part A (x_4, y_4, z_4) including proper orientation.
5. When an additional process is needed, fasten or join part A to part B using process tooling.

As can be seen in this simple list of tasks, developing robotic assembly system focuses on getting the parts to be assembled in the proper position and orientation and the combining process itself. Because of this, the rest of this section will describe the parameters of these two aspects: part feeding and presentation and the combining process.

Part Feeding and Presentation

Robot assembly requires the robot to go to a predefined location and grasp a part. The part may be positioned and oriented or it may not. Since a positioned and oriented part is preferred, part-feeding methods that can perform this task are desired. The most popular types are

1. Vibratory bowl feeders
2. Pallets and trays
3. Specialized feeders
4. Special vendor packages

5. Conveyors

Vibrating bowl feeders are one of the most popular methods due to the large number of parts that it can feed and its cost effectiveness. Pallets, in turn, are popular for many electronic parts and fragile parts where the part cannot withstand the forces found in a vibrating bowl feeder. Specialized feeders are those feeders that are usually designed for a particular type of part. These can include tube feeders, magazine feeders, and slides. Vendors can often provide parts in specialized shipping packages which keep the parts in the proper position and orientation. Finally, when the parts are delivered by conveyor, special fixturing and stops can often be placed on the conveyors to position/orient the part.

When the part is not positioned or oriented, additional sensors must be added to the system. Commonly used sensors are

Machine/robotic vision
Simple sensors such as photodiodes
Tactile/touch sensors

Robotic vision is becoming more popular in assembly as their purchase, software integration, and installation costs continue to decrease. Although most robot manufacturers offer vision systems as an option, they are still an expensive addition to the system. Simple on/off sensors can be used in certain cases when only limited data are required. Tactile sensors can sometimes be used to touch/feel the part to identify specific features of the part or to recognize its location.

Combining Process

After the parts are placed together, the combining process will often require the robot to perform some process operation. This can include an additional equipment such as a fastening gun for a screw, adhesive applicator and pump for a bonding operation, or a solder gun for soldering. Most robot manufacturers can offer equipment for the various types of assembly processes.

For more information on robotic assembly, the reader should review Asfahl (1992), Groover (1986), Klafter et al. (1989), and Sandler (1991).

14.11 Industrial Material Handling and Process Applications of Robots

John M. Fitzgerald

Replacing humans with robots to perform processes has often led to failure. The reason is that the robots are often mechanically capable of the manipulation while being incapable of process planning and control. Thousands of robot installations have failed because replacing the manual method with the automatic method lacked adaptability to process related variation. The human operators had been using their cognitive abilities to do the job. A vast majority of successful robot implementations past and present have a very important common aspect: repeated execution of fixed programs with little or no on-line modification of path or position.

Process robot planning and programming still usually require the efforts of highly skilled technicians. Often, complex programs cost too much and take too long. Continuously controlling and varying path manipulation parameters for real-time process control is difficult. Many processes are not known well enough to describe their control algorithmically. In a few applications sensors are becoming more common for adapting robot plans to changes in the environment. Setup, seam tracking, positioning, conveyor tracking, and now automatic programming for painting and finishing are becoming practical as sensor costs and computation costs continue to decline.

In this section robotic material handling and process applications are presented from an automation system perspective focusing on the robot's manipulation functions. Manipulation is considered a manufacturing material transformation and a transportation process factor. Programming and control are viewed as the means of integrating robot manipulation as part of the manufacturing process. The reader who is interested in a specific application is encouraged to first review the relevant process technology sections of this book.

Implementation of Manufacturing Process Robots

Manipulation as a Process Requirement

The starting point of automation system design is a thorough understanding of the process to be automated. Implementation of a process robot requires a focus on manipulation as a process factor. The pose and path requirements of the process are independent of the manipulator used.

It is useful to conduct a static spatial analysis of manipulation requirements and then examine the mechanical and dynamic requirements when designing or selecting a process robot manipulator.

A spatial description of the relative positions and orientations of the workpiece and tool during processing provides the basis for describing the required manipulation. *Tool poses* are graphed in an appropriate reference frame, usually the frame of the workpiece, or in the case of machine loading, the work holding fixture may be used. Path requirements are secondary for these applications. The path taken does not affect the process. For *continuous path processes* entire paths must be graphed or mapped. If continuous analytical descriptions of the path are not available, a sampling of discrete points along the required path can be used to represent the space occupied by the path. The result in both cases is a Cartesian mapping of spatial requirements of pose and path. A description of the pose and path precision requirements should be included. Next the mechanical and dynamic requirements are defined. Payload and force reactions at each position and along the path must be understood. Other important dynamic requirements such as acceleration and power should be quantified. The manipulation requirements are the basis for design and selection of both the robot arm and the controller.

Manipulation Capability of Process Robots

The basic mechanical capability of the robot mechanism to perform the manipulation work is determined by its mechanical structure, kinematic configuration, and drive mechanism. There are several applications including painting, palletizing, spot welding, and arc welding for which specific types of robot arm

designs have evolved driven by process needs. Although predisposed by design to perform a particular process, these robots have no innate process capability and are not guaranteed to perform in a specific application. Specifications of gross robot performance characteristics such as reach, *repeatability,* accuracy, and payload are usually readily available from their manufacturers. A well-defined set of process manipulation requirements when compared with published robot specifications usually isolates the field of mechanically qualified candidates. It is more difficult to characterize and evaluate a robot's capability for complex motion. The exact working of the robot's trajectory generation software is usually not known by end users and can only be evaluated by indirect testing. Acceleration and load capacity are usually specified, and there are some standard methods for specifying path performance, but the robot's dynamic behavior and performance are difficult to measure. Specific performance testing is usually required to prove manipulability for process robot applications.

Integration of Manipulation Control and Process Control

Achieving manipulator and process control integration depends upon robot programming and external data access. For any given application the required motion execution may be possible, but programming may be too difficult to be practical. Establishing that the robot is capable of coordinated motion can be done by reviewing the specifications or by conducting motion tests. As an illustration of the importance of programmability consider, for example, a situation in which a complex series of twisted curves define a robot tool path. If two robots with identical kinematic structure and joint trajectory generation capability differ in their programming in that one is capable of executing paths following user-defined mathematical functions and the other is only capable of executing paths defined by closely spaced taught poses, the difference in programming effort could easily amount to hundreds of hours. For each application encountered the programming methods must be assessed to determine if the required motion is programmable in a practical sense.

Access by process robot programs to external data is becoming more important. Although most process robots now work without any external process feedback, this is beginning to change rapidly with the development of improved low cost sensor systems and methods. Virtually all robots are capable of discrete digital and analog signal input and output. Most may also be equipped with standard serial and parallel communication capacity. If sensor information is to be used for set-up positioning or real-time path adjustment, the robot controller must have the communication and control to convert data into information that can be used to modify path and position commands. In cases of extreme path complexity, path planning systems external to the robot controller may be needed to create the paths. Testing will verify the ability of the robot controller to accept and execute externally generated motion sequence data.

Industrial Applications of Process Robots

Palletizing and Depalletizing

Many products are packaged in boxes of regular shape and stacked on standard pallets for shipping. Robots are commonly used to palletize and depalletize boxes because they can be programmed to move through the array of box positions layer after layer. Although palletizing is more common than depalletizing, there is no major functional difference in the manipulation requirements. Transport distances of several feet are common. Stack heights usually do not exceed 5 ft. Payload weight can be in excess of 100 lb. When standard servo-driven joint actuators are used accuracy and repeatability will usually be far better than the required box positioning precision.

Palletizing typically requires four axes of controlled motion — three for translation and a fourth for yaw to orient the box. Cylindrical coordinate robots are favored in palletizing because they have large vertical lift and a compact footprint allowing more of the floor area in the workspace for conveyors and pallets. When larger workspace is needed gantry robots must be used. Continuous duty cycles are not uncommon and robot power is important for maximizing throughput. The most technically demanding aspect of system design is the gripper. Vacuum grippers are popular for lifting boxes by their tops, but other more complex gripping methods are sometimes needed. Payloads must be carefully positioned

with respect to the robot's wrist and other links to balance gravitational and dynamic loading. Load shifting during high acceleration moves can result in dropping or mislocating the box.

Palletizing position arrays are usually taught or programmed relative to a corner or keystone box position as a reference so that the entire array can be shifted by redefining that one position. Programs are simple and easily modified to adapt to changes in box dimensions. Monitoring is done by checking the state of discrete proximity and vacuum sensors. A proximity sensor mounted on a gripper will indicate if an object is at an expected location; or the same simple proximity sensor may be used to stop the robot in the correct location to pick up a box from a stack of unknown height when the top of the box is encountered. Vacuum pressure switches are often used to verify acquisition by suction cup. A simple proximity switch can be used to signal the presence of an expected package at the pick-up point. With careful timing and additional sensor inputs, items can be transported to and from moving conveyors.

Packaging

Packaging is often a combination of palletizing and assembly-type actions. A collection of objects which may not be identical are inserted into a box or other container. The robot may also be required to assemble, place dunnage, seal, or mark the package. Insertion may simply require positioning the pack item over the opening of the package and dropping it. Boxes most often are supplied partially assembled, printed, and folded flat. Usually human operators or a special machine will open and prepare the box for packing; rarely will the robot be used for this purpose. Often the robot can be used to place cardboard layer separators, foam, or cardboard holding forms and other protective dunnage in the box. Finally, sealing and marking operations may be performed by the robot. Pack items may require complicated assembly-type motions such as rotations and curved moves to clear other packed items.

Three to six axes of motion may be needed. Packing items with a variety of sizes, shapes, and other varying physical properties into one package have the potential to complicate motion and tooling requirements. Grippers can be designed with multiple functions or they can be designed to be exchanged by the robot at tool storage racks. When material throughput is high, a single robot may be dedicated to each pack item. Simple programming methods are employed such as teach programming. Discrete sensors are useful for monitoring grip status of pack items.

Machine Tending: Loading and Unloading

Forges, stamping process, some machine tools, and molding machines are now commonly tended by robots. Historically these types of machines have been loaded by human operators. Now these jobs are considered to be too arduous and hazardous. An important benefit of robotic machine loading is improved product quality resulting from consistent machine cycles. Robots eliminate the inconsistencies of human-paced loading and as a result the cycle can be precisely repeated. For heated molding, stamping, and forging processes, part formation and release are sensitive to the thermal state of the machine. If a machine is left open for loading for differing amounts of time each cycle, significant cooling variations result in potential sticking and geometric flaws. When robots are used, the process can be tuned to the consistent robot loading cycle.

Machine loading is usually more demanding than other material handling applications because part orientation and placement are critical and may require locating mechanisms such as tooling pins and pads and/or sensor logic to guarantee interface between the robot and the serviced machine. Accuracy is usually not an important factor because the loading stations are permanently located in the robot workspace, but repeatability requirements may be as small as several thousandths of an inch. Payloads can range from a few ounces to several hundred pounds. Grippers for machine loading may also require tooling pins and pads to locate and orient parts and to mate precisely with the machine's part holding fixture. The gripper may dock with the holding fixture and then transfer the part when loading clearances are very tight.

The entire range of robot types, sizes, and configurations is used for machine tending. Articulated arm robots are needed when dexterous manipulation is required to transport parts through the maze of clamps and spindles and other protrusions and obstacles found on some machines or when part orientation

must change for loading. Applications where the robot is dedicated to loading a single part into a single machine in high volume production are not uncommon. Position programming is usually done by teaching. It is common to monitor discrete sensors in the gripper and the loaded machine to insure proper loading before cycling the process machine.

Sorting

Discrete parts are often sorted during production, usually as a condition of transfer to the next production station. The sort characteristics are usually distributed in some unpredictable manner so that individual inspection and handling are required. The diflference between sorting and other transfer or loading robot applications is that the disposition of the part is based on information gained during the sort. The robot must have the programming functions to support multiple preprogrammed path execution triggered by the logical sort outcome conditions.

Part Dipping

Many processes require controlled manipulation of parts temporarily submerged in some working fluid or coating material. Some common part dipping processes are the following.

Investment Casting. Intricately shaped and often delicate wax forms are coated with a slurry of stucco material which cures to form a mold. Later the wax is melted and drained from mold which can then be filled with molten metal. The dipping motion must be carefully controlled to prevent trapping bubbles and distorting the wax shape.

Solder Pretinning. Electrical contact pads and component leads are coated by dipping in molten solder as a preliminary step to assembly and soldering of the connections. A temperature-dependent flux reaction is required to achieve wetting by the solder so the robot must hold the component submerged in molten solder for a precise delay period. Speed of withdrawal is a major process variable for controlling the coating thickness of solder.

Conformal Protective Coating. Some electrical and mechanical components are dipped in liquid polymers to seal out moisture, air, and contamination. The viscosity of the polymer and the speed of insertion into the fluid must be controlled so that flow into small features occurs without trapping bubbles of air. Once submerged the component may be reoriented to several poses and to allow air bubbles to escape.

Quenching. Heat treating is a commonly used method of improving alloy properties. Various fluids are used as cooling baths. Controlling insertion and manipulation is important for control of cooling rate.

Dipping processes require precision of part insertion and withdrawal so velocity and acceleration must be programmable and repeatable. The stirring requirements may require the use of a two- or three-axis wrist in addition to the translation motion axes. Grippers may require special cleaning or cooling capability either on board or at service stations located conveniently in the robot's reach.

Resistance Spot Welding

Robotic spot welding (see Figures 14.11.1 and 14.11.2) is the most pervasive robot application in the automotive industry. Resistance spot welds are formed by tightly clamping steel pieces together with opposing contact electrodes and then passing a large amount of current through the joint, welding the metal while producing a spray of molten sparks along with loud noise. Then the joint is held momentarily until the weld solidifies. Welds are made at discrete positions by moving the robot-mounted gun to pretaught poses. The spot welding process parameters, pressure and temperature, are controlled with the separate gun controller. Weld location and therefore positioning of the gun are critical.

Dexterity, payload, and quickness are critical operational requirements for spot welding robots. Gun pose repeatability is critical for consistently locating weld joints. Access to joint locations is limited because both electrodes must reach the weld site while maintaining clearance between gun frame and workpiece edges. Large articulated arm robots are typically used for most spot welding applications because of the dexterity needed and because the weight of the welding gun and associated robot-mounted

FIGURE 14.11.1 Six-axis articulated arm robots spot weld automobile bodies on a transfer line. (Courtesy of Nachi Robotics, Ltd.)

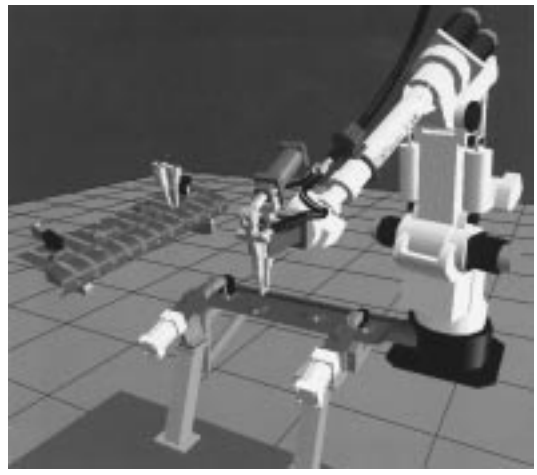


FIGURE 14.11.2 Spot welding operation simulated in off-line programming environment. (Courtesy of Deneb Robotics, Inc.)

apparatus often exceeds 200 lb. Fixed cycle programs are typical which may require several man-months to develop and less than a minute to execute. The robot spot welding path position names, path order, and control logic can be developed off-line, but lack of robot positioning accuracy characteristic of large articulated robots requires the weld positions for each individual robot to be taught by posing and recording them manually. This takes advantage of the robot's repeatability which is often orders of magnitude better than its accuracy. Unfortunately, when a robot that has been teach programmed is replaced by another robot, even an identical model, hours or days of teaching will be required to bring the replacement robot on line. Practical new PC-based calibration methods which eliminate this problem by effectively improving accuracy are now becoming commercially available.

Drilling

Hole drilling is a precision machining process. Most robots cannot hold a drill spindle rigidly enough to overcome the drilling reactions and most robots cannot generally move in a precise enough straight line to feed the drill. Drilling robots use special drilling end effectors which locate and dock onto the work piece or a fixture. The robot wrist and arm must be compliant and forceful enough to hold the drilling end effector firmly into location against the fixture or workpiece. Drilling end effectors have a spindle motor and a feed mechanism which execute a separately controlled drilling cycle while the robot holds the end effector in position.

The robot's only contribution to the process is to move the drilling end effector into its docking or holding place. Drilling robots have been used most successfully in the aerospace industry because airframe structures require thousands of holes to be placed precisely and in complex orientations. Manipulability requirements for drilling are similar to those for spot welding. The drilling end effector weight will tend to be less than a welding gun but tool holding force and reach usually impose the requirement for large robots.

Fastening

Robots are commonly used for applying threaded fasteners in the automobile industry for fastening wheels, and in the electronics industry for screwing components to circuit boards and circuit boards into chassis. Robots are also used for riveting in airframe fabrication.

Fastening is an end effector position-and-hold application. The robot does not follow the threaded fastener as it turns and travels into place; the end effector uses a slide or cylinder for that purpose. Automatic nut runners and screwdrivers and the associated hardware feeding apparatus are broadly available. Since a human is no longer operating the fastening tool other means of process control are needed. Usually fastener angular displacement, longitudinal displacement, and torque can be monitored and correlated with signatures or patterns characterized for specific fastener joints. Manipulator arm and control system requirements are similar to other position-and-hold applications. Very large torque may be encountered. Torsion bars or other static mechanisms are often needed to prevent the arm from being torque loaded.

Inspection

Robot inspection involves relative part/sensor manipulation to compare, measure, or detect a physical characteristic of the objective workpiece. Sensors used in robotic inspection include chemical detectors, computer vision systems, infrared detectors, sonar, laser radar, radiation detectors, capacitive proximity sensors, touch probes, X-ray cameras, particle/photon detectors, thread probes, and go-no go gauges. Robot inspection applications cover the range of manipulation from end effector position-and-hold to continues in-contact path motion. In some cases the kinematic structure of the robot is used as a spatial measuring device by incorporating surface sensors or probes in the last link as robot end effectors (Figure 14.11.3). The forward kinematic solution of the joint angle measurements sampled at a contact pose give the position in Cartesian space of the contact point. If the manipulator is stationary during the measurement then the robot's Cartesian positioning error must be added. If the robot is calibrated the error may be almost as small as the repeatability (0.001 to 0.020 in. for most industrial servo-driven

arms). If the arm is moving while measurements are made significant error may be added because of delay in sampling the manipulator joint positions.

FIGURE 14.11.3 Robots inspect pick-up truck body prior to final assembly. (Courtesy of Fanuc Robotics, N.A.)

Programming considerations are critical because robot inspection often requires data collection at a huge number of discrete positions. When CAD data are available, off-line programming of inspection may be possible, particularly for position and sense-type inspections. Sensor tool pose requirements can be quickly and accurately defined in the CAD environment and the pose data transformed into robot workspace coordinates. When hundreds or thousands of inspection poses are required manual teach programming may be too time consuming and cost prohibitive, especially when a mix of different parts must be inspected.

Paint and Compound Spraying

Paint spraying is a major application in the automotive industry. Painting booths are hazardous because the paint material is often toxic, carcinogenic, and flammable. Human painters often wear required protective clothing and breathing equipment. The paint fan projecting from the arm-mounted spray gun must be manipulated smoothly along paths that are often curved and complex.

Most robots used for painting are especially designed for that purpose. They usually have large reach, small payloads, and repeatability is usually larger than that of other types of robots and may exceed ±0.010 in. Painting robots are typically six DOF articulated arms, often with supplemental axes to pitch the paint gun and to traverse alongside a moving line. The potential for ignition of solvents and suspended particles may require taking precautions to eliminate ignition sources associated with the sparking of motors and other electrical components. Until brushless DC motors became commonly available for robot actuation virtually all painting robots were hydraulic because of the motor sparking problem. Lead-through teaching (also called teach-playback) is typical for painting. Off-line programming of painting is becoming more popular and some special software packages are available. Figure 14.11.4 shows an image from a simulation used in validating painting robot motion programs. Some compound spraying is done with smaller general-purpose robots, for example, spraying of protective coatings in the electronics industry. Circuit boards may require unexpected dexterity in order to point the spraying nozzle correctly to coat board features.

Compound Dispensing

Compound dispensing refers to laying a bead of fluid material on a surface. Application examples include caulking car bodies, sealing windshields, placing solder masking on circuit boards, gluing subassemblies,

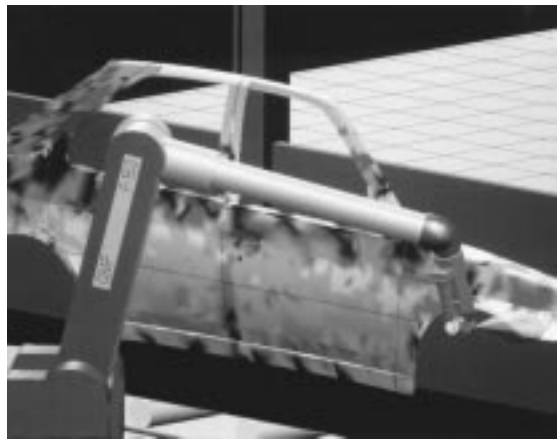


FIGURE 14.11.4 Painting operation simulated in off-line programming environment. Estimated paint thickness is illustrated by shading. (Courtesy of SILMA, Inc.)

solder paste dispensing, and decorating candies and cakes. Precision of placement and amount is critical. Smooth controlled paths are often essential. Position accuracy and tool path velocity accuracy are both important requirements.

All types and configurations of robots are used for dispensing. Many applications require only three DOF. When obstructions must be maneuvered around to gain access to the dispense locations, five or six DOF are needed. Payloads are usually small. Dispense speed may be limited by either the robot's ability to track a path at high speeds or by the dynamics of the dispensing process. In automotive applications the fixed cycle mode of operation is common. A robot program to lay a bead of sealer along the edge of a windshield is a taught path requiring good dynamic path repeatability of the robot. In electronic circuit board fabrication and decorating cakes, each workpiece may have a different dispense pattern; teaching paths are not practical in this situation. Some method of off-line programming must be used.

Cutting

Many engineering materials are produced and supplied as stacked or rolled flat plates or sheets. Further fabrication can require forming and/or cutting these materials into precise shapes. Robots are frequently used to manipulate a variety of cutting tools along paths that are often complex and curved. Many cutting processes are also used to produce fine features such as holes and slots. Common robotically manipulated cutting processes are listed below.

Laser. Molten metal heated by collimated intense light is blown away by a gas jet. Most common use is for thinner metals (0.50 in. or less) and on a variety of other thin materials.

Waterjet. A high velocity water jet is formed by forcing very high pressure water through a small orifice in the range of 0.008 to 0.040 in., which can cut a variety of nonmetals.

Abrasivejet. After a high velocity water jet is formed it passes through an abrasive mixing chamber where abrasive particle are entrained in the jet. A variety of metals and other tough and hard materials can be precisely cut; many materials can be cut with good control up to 1.0 in. thick. Cut thickness in excess of 6.0 in. has been reported.

Plasma Arc. Molten metal heated by an electric arc is blown away by a gas jet. Plasma arc cutting is commonly used to cut patterns in plate steel.

Router. A rotary cutter is most often piloted either on the workpiece or a guide fixture for precise trimming or chamfering edges of plate and sheet material.

Knife. A variety of knife types, some ultrasonically assisted, are employed to cut mostly nonmetals.

The cutting tool path and pose must be precisely controlled to achieve accurately patterned piece parts. The demands on manipulator performance are primarily determined by the interaction of desired part geometry, material thickness, and material properties. Feed rate, tool stand-off, beam, jet, and arc angle are all cutting process control variables which must be adjusted to material characteristics for good results. An extreme case of cutting manipulator performance demand is the combination of thin easily cut material with complex shape, and small geometric tolerance. This requires high speed coordinated motion of five or more axes which must be kept on track. This translates to a requirement for high performance servo-control elements in order to achieve high rates of mechanical response and joint angle position and velocity precision. Some type of advanced programming method such as a CAD/CAM may be required for high part mix applications. When extreme precision and complexity are required, as in many aerospace applications, precision fixtures incorporating tool guides may be used to force the tool path to repeat with near-zero deviation. In the case of contact tools such as routers and wheel knives the end effector must be capable of bearing preload forces applied in excess of the tool reaction forces to eliminate tool bounce-induced path errors.

Equipment and tooling for many of the robotic cutting processes may be complex and expensive. End effectors can easily cost tens of thousands of dollars and require difficult and cumbersome wiring and plumbing. Routing of laser wave guides and high pressure tubing for water jets from power source to robot end effector requires skill and experience in both design and installation.

The majority of cutting robots are three-axis natural Cartesian machines specially designed to cut flat sheet materials. Cutting speed and accuracy performance are aided by their easily calculated kinematics and easily predicted dynamics. Path planning and path generation are also simplified with flat parts. CNC is commonly used, and many automatic nesting and path programming software systems are readily available. There is some use of tool position sensors for part location during setup. In-process sensor-based tracking for edge cutting has been implemented with success, but is rare. Several five-axis gantry-style machines have been implemented for cutting complex aerospace materials including impregnated broadcloth patterns and composite wing skins. Articulated arm robots may be used when less precision is needed, as in trimming automobile carpet or making cut-outs in large plastic moldings.

Arc Welding

Arc welding is a metal joining process that uses intense heat produced by an electric arc between an electrode and the metal parts being welded. The weld pool and arc are always shielded by inert gas or a chemical vapor. In gas metal arc welding (sometimes called metal inert gas welding), which is the most common robotic arc welding, an electrode of filler metal wire is fed through a gun into the weld pool site as the robot manipulates the gun along the weld path. The hazards of arc welding include: intense ultraviolet, visual band and radio frequency radiation, toxic fumes, and noise. The pose (position and orientation) of the welding gun with respect to the joint or seam is a major arc control parameter. The feed rate of the gun is important in control of penetration and other weld characteristics. Unlike spot welding, manipulation is an arc welding process control variable.

Most arc welding robots operate in fixed cycle mode, which means they execute or play back a programmed sequence. If assemblies are presented to the robot with consistent seam geometry, then the path can be taught once, stored, and then executed repeatedly for each assembly. Given that all other relevant process variation is within accepted limits, the system will produce satisfactory output. However, weld seam position and seam shape variations may influence the process, especially in larger assemblies. When there is variation in the upstream sizing, cutting, fit-up, and jigging of weld assemblies, the location and orientation of the weld seam will vary. Also, as the weld progresses, localized thermal expansion and residual stresses can force seam distortion. A range of methods for adapting the robot system to these variations exists, from correcting pretaught path plans at setup time, through actively altering robot motion in “real time.” A sustained high level of academic and commercial research and development effort has resulted in practical methods of automatic weld seam tracking and process control.

Rapid deployment in recent years is a direct result of sensor integration for seam tracking Seam tracking methods correct the path to compensate for errors in location and orientation of the welding tip based on sensor data. Commonly used sensors include mechanical probes, computer vision, laser edge detection and ranging, and arc current and voltage. Because of the extreme environment of the region surrounding an active welding tip, sensors are often housed in protective chambers. Typically the errors are measured and calculated in a convenient reference frame in the three-dimensional workspace of the robot system, the same space in which the tool path is described. In some cases the error is measured by tool-mounted sensors relative to the moving reference frame of the tool. The preprogrammed path is then shifted by mathematical transform in the reference frame of the tool. An important aspect of seam tracking is the use of sensors to detect the sides of the weld channels as boundaries for automatic side-to-side weaving. This is usually done with "through-the-arc" sensing in which arc current is monitored as an indicator of clearance between the welding tip and the channel edge. While tracking in the direction of the seam, transverse motion commands are given so that the tip approaches one edge until the edge is sensed and then the motion is commanded in the direction of the other edge. Weld penetration, filler deposit amount, and weld bead shape can be controlled in-process by variable control of the welding speed or feed rate. Arc welding robot systems which use sensors to adjust the robot path in real time (computation is fast enough to respond to sensor data with useful path adjustments) are among the most advanced or intelligent robotic applications found in practical industrial use.

Robots used for arc welding must be capable of precisely executing taught paths. Motion must be smooth and precisely controlled. Velocity control is important but the speeds required are not high, 2 in./sec, while welding is faster than most applications require. Higher velocities are important to reduce cycle time for applications with lengthy arc-off motion. Welding robots must have good reach and dexterity. Five DOF is required as a minimum and six DOF adds to gun maneuverability. There is normally no forced contact with the weld seam and the welding gun's weight is usually less than 20 lb, so robot payload requirements are light. If real-time path altering is required, the robot's motion generation functions must have programmable interfaces with the sensor systems. The robot's controller must have a means of accepting data and manipulating it for use with high level functions in the robot's native programming language.

Finish Machining

Few material-forming processes produce finished parts. Most machining operations leave burrs and sharp edges. Large aircraft wing skins are milled by three-axis terrace cutting leaving small steps which must be blended to prevent fatiguing stress concentrations. Complex curved surfaces like ship propellers and aircraft landing gear are machined with rounded milling tools which leave a pattern of tool marks which must be ground off. Cast parts require gate and sprue removal and deflashing. Many parts must have their surfaces conditioned for appearance or subsequent plating and coating operations. Stamping and forging of automobile door panels and engine components leave "imperfections" which are finished out by hand. Die cast surfaces of hardware for door handles, faucets, furniture, and appliances are ground and polished. Finishing removes material to reduce waviness, reduce roughness, remove burrs and sharp edges, and to remove flaws. Manipulation is a finish machining process control variable. Tool pose, applied pressure, feed rate, and tool path must be controlled. Smaller parts are finished using fixed-floor or bench-mounted tool stands. For larger work pieces the finishing tool is mounted on the robot. Finish machining is very demanding of the manipulator because continuous path control is required while maintaining contact between part and tool.

Medium- to large-sized robots are usually required for surface finishing because end effector weight and tool reaction force are additive when calculating payloads. A further margin of payload is usually required to offset the fatiguing effects of vibration and cyclic loading from tool reactions. Tool point positioning accuracy is less important than tool orientation and feed rate. In general, higher rates of surface curvature will require greater robot path precision. Six DOF robots are often required. Edge machining tool paths are constrained by burr geometry, finishing tool characteristics, end effector geometry, and part geometry. A single part may have edge features in several directions and orientations.

The robot may require an assortment of different tools and a tool changer to reach all edges. If these measures do not allow access then multiple setups may be needed to present all features for finishing.

Most robotic surface finish machining applications use compliant abrasive processes. Force control is required in some applications to keep tool pressure constant. Force controllers are most often incorporated in the end effector or in the tool stand. Figure 14.11.5 shows a robot equipped with a force-controlled finishing end effector using a servo-controlled pneumatic actuator that is capable of applying consistent tool pressure. Through-the-arm robot force-control is available from some robot manufacturers, but its usefulness is limited to applications requiring slow feed rates because of slow mechanical response.

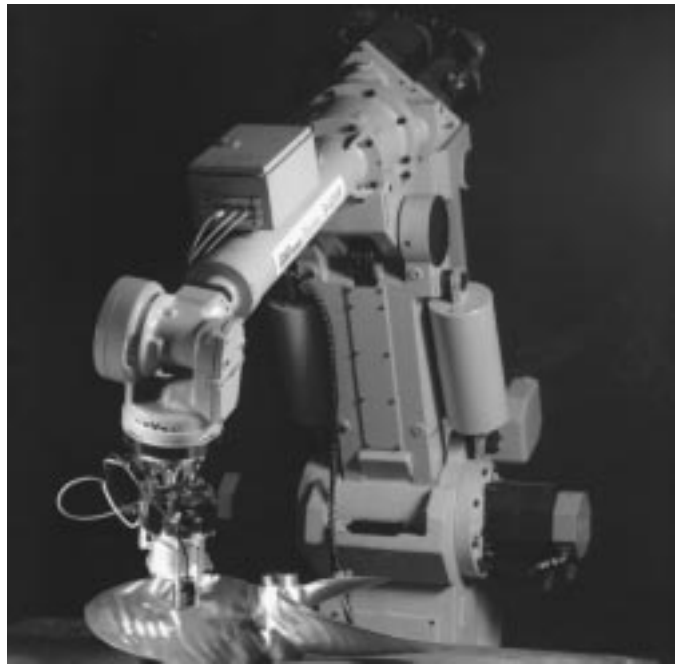


FIGURE 14.11.5 Robot equipped with force-controlled end effector grinds tool marks from ship propellor. (Courtesy of The Automation and Robotics Research Institute, The University of Texas at Arlington.)

Path planning and programming of edge and surface finishing for complex-shaped parts can be very difficult and time consuming. Both tool position and tool pose are critical in obtaining the correct tool contact area. Tedious paths programmed using the teach method require hundreds of hours to develop because of the large number of taught points. Generating the path control sequence is a major problem in manufacturing operations which produce a variety of complex-shaped parts. An example is in polishing large asymmetric-shaped aircraft skin panels. A more difficult automation problem is robotic spot finishing of flaws and other anomalous regions of the part surface when their location and extent are not known before set-up time. The reason is that the paths must be planned, generated, and executed on-line. This type of motion generation system requires part modeling and computational functions not available on most robot controllers. Figure 14.11.6 shows an operator digitizing a part surface to be modeled in preparation for automatic path generation by the PC.

FIGURE 14.11.6 Workpiece surface is digitized and modeled in preparation for automatic tool path generation. (Courtesy of The Automation and Robotics Research Institute, The University of Texas at Arlington.)

14.12 Mobile, Flexible-Link, and Parallel-Link Robots

Kai Liu

This section will discuss nonstandard robots, including mobile robots, lightweight flexible-link robots, and parallel-link robots. These robots are often more suitable than standard serial-link commercial robots for certain applications.

Mobile Robots

Traditionally, standard robots are fixed in position. They are mounted on a rigid base and bolted to the floor so that they can withstand the forces and torques applied when the arm manipulates objects. However, fixed-base robots cannot cope with a large variety of applications in which a robot will operate in large and unstructured domains. A special type of manipulator, that is, a mobile robot, is often required in these applications.

In tomorrow's flexible manufacturing system (FMS) environment, mobile robots will play an important role. They will transport parts from one workstation to others, load and unload parts, remove undesired objects from floors, and so on. In addition to indoor mobile robots, there are some other outdoor occasions where mobile robots may take on heavy responsibilities. Examples include construction automation, military missions, handling of harmful materials, hazardous environments, interplanetary exploration, and so on.

Classifications of Mobile Robots

Mobile robots can be classified by driving mechanism as wheeled mobile robots, legged mobile robots, and treaded mobile robots. Some other types of mobile robots, for instance, the *underwater mobile robots*, the *autonomous aerial mobile vehicle*, and so on, are also available but are not included in this discussion.

Wheeled Mobile Robots. Mobile robots using wheels for locomotion are called *wheeled robots*. Two driving configurations are used in today's wheeled mobile robot — steer-drive and differential-drive. The former uses two driving wheels to make the vehicle move forward and backward. The heading angle is controlled by an independent steering mechanism. Since the driving action is independent of the steering action, the motion control of the vehicle is somewhat simplified. However, due to physical constraints, this configuration cannot turn in a very small radius. This shortcoming makes it less attractive in some industrial applications. Differential-drive configuration mobile robots, on the other hand, have two independent driving wheels positioned at opposite sides of a cart base, arranged parallel to one another. Their speeds can be controlled separately. Thus, by appropriately controlling the speed of each driving wheel, this mechanism is able to drive the vehicle forward and backward, as well as steer its heading angle by differential speed commands. Even though this configuration requires a somewhat more complex control strategy than the steer-drive configuration, its capability of making small-radius turns, even making turns on-the-spot, makes it the first choice in many industrial applications.

Some commercial wheeled mobile robots include the *CyberGuard Autonomous Survelliance Robot* manufactured by Cyberworks Inc., Canada; *B12 Mobile Robot Base* manufactured by Real World Interface, Inc., Dublin, NH; *LabMate Mobile Robot Platform* manufactured by Transitions Research Corporation, Danbury, CT; and *R-20 Mobile Robot* manufactured by Arrick Robotics, Euless, TX.

Legged Mobile Robots. While most mobile robots use wheels for locomotion because of the simplicity of the moving mechanism design and control, some other mobile robots use legs for locomotion. These types of mobile robots are called *legged robots.* The primary advantages of legged robots include their ability to traverse rough terrain with good body stability and minimal ecological damage. In order to maintain good stability, it is sufficient that at any time there are three points in contact with the ground. Therefore, most legged robots use at least four legs, or even six or eight legs. As long as the legged mobile robots' center of gravity is within the triangle formed by the three contact points, stability is

guaranteed. Compared with the wheeled robots, the control of legged robots is much more difficult. Much has been learned about multilegged locomotion from studies of balancing and hopping on a single leg. In particular, biped running can be viewed as successive hopping on alternating legs, since both legs never contact the ground simultaneously. Some examples of legged mobile robots include *ODEX I* manufactured by Odetics.

Treaded Mobile Robots. Another type of mobile robot, the treaded robot, moves much like a tank. An example of the treaded robot is the *ANDROS MARK V* manufactured by REMOTEC, Inc. at Oak Ridge, TN. It is something of a hybrid between a walking and a rolling vehicle. *ANDROS* can ascend/descend 45° stair/slopes by lowering its front and rear auxiliary tracks. It has all-terrain capabilities that are ideal for performing missions in rough outside terrain or in rubble-strewn, damaged buildings.

Sensors and Measurements

To navigate in unknown and unstructured areas, the mobile robot must have the capability of sensing the real world, extracting any useful information from the data acquired, and interpreting the information to understand the environment surrounding it, especially the situation in front of it. Several sensor systems for mobile robot navigation have been reported in the literature (Elfes, 1987). Of these, stereo vision systems and active rangefinding devices are the most used sensor systems. The former extracts range information from pairs of images to build a 3D world map. However, due to the high computational expense — a 3D map may require 1 min to generate — stereo vision systems have not to date been generally used for real-time navigation control. Active rangefinding devices do not suffer from this problem because they can deliver range information directly

Two kinds of rangefinding devices are available: laser rangefinders and ultrasonic range transducers. Even though laser rangefinders can provide fast response with high resolution, a relatively long measurement range, and high measurement precision, the required systems structure and configurations are very complicated, which makes the system itself very expensive. On the other hand, sonar systems are simple and low cost (probably orders of magnitude less expensive than laser-based systems), though the measurements have lower resolution and lower precision.

Determining range by means of sonar systems is a simple process. A short burst of ultrasonic sound is first transmitted by an ultrasonic range transducer, then an echo is expected to be received by the same transducer. If in a reasonable time period no reflected signal is detected, it is assumed that there are no objects in the area of interest. Otherwise, the time for round-trip propagation is determined and the distances to any objects are calculated. The transducer yields a 3-dB full angle beamwidth of 50 KHz at approximately 12 to 15°, depending on the signal frequency and transducer diameter. Thus, to scan the whole area surrounding the mobile robot, at least 24 to 30 transducers, of which the transmit/receive axis lies in the same horizontal plane, are needed.

Vision systems, also sometimes useful in robot sensing, usually consist of one or more video cameras and an image processor. The vision system can provide the richest source of information, which is, in fact, needed in certain applications such as road following, object identification, and so on.

Feedback from rotary and linear actuators used in wheeled and/or legged mobile robots is provided by position sensors and/or velocity sensors. This information is then processed for estimating position and orientation of the mobile robot in world coordinates.

By far the most commonly used position sensor is the *optical encoder,* which uses marks to indicate position. The typical encoder has a track for each binary digit of information. The encoder is mounted on the servo motor. When the motor rotates certain degrees, the absolute rotation position of the axis can be read from the digital output of the encoder. The resolution of the encoder is equal to $(360/2^n)$ degree, where n is the number of tracks. If an 8-track encoder is used, then a 1.4°/step resolution can be attained.

Other types of position sensors used in mobile robot systems include synchros, resolvers, potentiometers, linear variable differential transformers (LVDT), rotary variable differential transformers (RVDT), amplitude-modulated laser radars, and laser interferometers.

Conventional servo design requires that the servo controller include a "velocity term" in its transfer function. Without the velocity term, a servo system will usually exhibit an undamped, resonant behavior and can be highly unstable. In principle, the signal from a joint position sensor can be electronically differentiated to obtain joint velocity. However, if the joint position sensor has a noisy output, differentiating the position sensor signal can effectively magnify the noise sufficiently to make the servo system unstable or unreliable. To overcome this difficulty, several velocity sensors are available for directly measuring the joint velocity. A *DC tachometer* system consists of a voltage meter (or a current meter) and a small DC generator (sometimes called a "speed-measurement generator"). The latter is usually constructed with a permanent-magnet stator and a multipole wound armature. The armature is connected directly to the rotating shaft of the servo motor which is used to drive the manipulator joint. When the small permanent magnet DC generator rotates with the servo motor, its output voltage (when driving a high-impedance load) varies in proportion to the rotation speed of the armature. Voltage output variations can then be translated into speed changes or used as a feedback signal to control the robot arm velocity.

Supplementary position and orientation information can also be supplied by inertial guidance sensors, terrestrial magnetic field sensors, or inertial reference systems (IRS).

Navigation

Autonomous navigation of mobile vehicles has been studied by many researchers. In Elfes (1987), a sonar-based navigation system for an autonomous mobile robot working in unknown and unstructured environments was developed. The workspace is classified into "probably empty regions," "somewhere occupied regions," and "unknown regions" based on the interpretation of the data obtained from the sonar system. In this scheme, as more and more data are received, the first two regions may increase, and the uncertainty of these regions also decreases. It is reported that after a few hundred readings, a sonar map covering a thousand square feet with up to 0.1-ft position accuracy can be made. Another navigation scheme uses a stereo vision system to control a mobile base autonomously operating in a complex, dynamical, and previously unknown environment. A pair of stereo cameras is mounted on the mobile base to generate a symbolic world model. Based on this model, the desired trajectories are specified for the driving motors.

Although the schemes described above work well in specific environments, path planning and navigation control are always separated into two isolated issues. The path planning mechanism designs a smooth path from an initial position to a goal position by providing profiles of position and velocity, or profiles of position and heading angles, in Cartesian space. It assumes that perfect knowledge of the system dynamics and the environments is always available and that the position and orientation of the vehicle are measurable absolutely. After the desired trajectories have been designed, the navigation mechanism will take charge of driving the mobile robot to follow the prescribed trajectory as closely as possible. Even though each mechanism may work well through closed-loop control, the whole navigation system is an open-loop system. Static path planning strategies do not provide the essential adaptability necessary for coping with unexpected events. The success of navigation control depends mostly on the accuracy of absolute measurements of position, velocity, orientation, and their rates of change. All of these must be measured in (or transformed to) Cartesian space. This is a very expensive and difficult job.

Other possible closed-loop navigation control schemes use intelligent control techniques, for instance, fuzzy-logic control. In such a control scheme, the path-planning mechanism and trajectory-following mechanism are often integrated, not separated. The path is planned dynamically and is always up-to-date. All the information that the system needs to know such as "where is the goal (the dock)," "what is the required final orientation (the docking angle)," "what is the present orientation (the present heading angle)," "what is the present distance between the car and the goal," "what is the present distance between the car and any obstacles," "what is the safe turning radius (the minimum curvature radius)," and so on is easily captured through sensing the environment surrounding the car using onboard sensors (e.g., sonar) that yield relative information.

The advantages of such intelligent control strategies are evident. They unite navigation and maneuvering into a single set of algorithms. Full and accurate knowledge of the system dynamics is not required. The only knowledge needed are the correlations between the control actions (acceleration, steering, etc.) and the performance ("behaviors") of the system. The absolute measurement of the position and velocity in Cartesian space is not required. Only information about relative locations is necessary, and this is always available. Tight coupling between sensor data and control actions provides the adaptability necessary for coping with unexpected events. Actually, there is no path planning to be performed; the driving mechanism reacts immediately to perceived sensor data as the mobile robot navigates through the world.

Flexible-Link Robot Manipulators

Most robots used in today's manufacturing systems are rigid-link manipulators. Making the robot links and drives extremely stiff to minimize vibrations allows rigid-link robots to track a desired trajectory with very high degree of accuracy, often using standard classical (PID) control schemes. However, the price paid for this includes heavy manipulators, a low payload-weight-to-arm-weight ratio, high power consumption, and slow response rates to motion control commands.

With the growing demand from industry automation for lower manufacturing costs, higher motion speeds, better performance, and easier transportation and setup, the rigid-link manipulators may, sooner or later, be replaced by some sort of lightweight mechanical structures, such as flexible-link robots. While lightweight flexible manipulators have certain inherent advantages over rigid-link robots, they impose more stringent requirements on system modeling and controller design because of the vibrations of the flexible modes.

Modeling of Flexible-Link Robots

One essential step toward successful control synthesis is to obtain an accurate dynamic model for flexible-link manipulators. The flexible manipulator dynamics can be derived on the basis of a recursive Lagrangian assumed-modes method (Book, 1984). The motion of robots with link flexibility is governed by partial differential equations that must be satisfied inside a given domain defining the flexible structure, and by boundary conditions to be satisfied at points bounding this domain. Therefore, the dynamic model of flexible-link manipulators consists of highly coupled nonlinear integro-partial differential equations. Control of a structure using a formulation based on partial differential equations is extremely difficult. To reduce the complexity of the model, the assumed-modes method is used to produce a set of nonlinear ordinary differential equations based on an orthonormal series expansion of the flexure variables.

A solution to the flexible motion of links is obtained through a truncated modal approximation, under the assumption of small deflections of the links. The dynamic equations of motion for an *n*-degree-of-freedom manipulator with up to *m* flexible links can be written as

$$\mathbf{M}(\mathbf{q},\delta)\begin{bmatrix}\ddot{\mathbf{q}}\\ \ddot{\delta}\end{bmatrix}+\mathbf{D}\left(\mathbf{q},\dot{\mathbf{q}},\delta,\dot{\delta}\right)\begin{bmatrix}\dot{\mathbf{q}}\\ \dot{\delta}\end{bmatrix}+\begin{bmatrix}0 & 0\\ 0 & \mathbf{K}_f\end{bmatrix}\begin{bmatrix}\mathbf{q}\\ \delta\end{bmatrix}+\begin{bmatrix}\mathbf{F}_r(\mathbf{q},\dot{\mathbf{q}})\\ 0\end{bmatrix}+\begin{bmatrix}\mathbf{G}_r(\mathbf{q})\\ 0\end{bmatrix}=\begin{bmatrix}I\\ \mathbf{B}_f\end{bmatrix}\tau$$

where $\mathbf{q} = [q_1\ q_2\ \ldots\ q_n]^T$ is the vector of rigid joint variables, $\delta = [\delta_1\ \delta_2\ \ldots\ \delta_m]^T$ is the vector of deflection variables, $\mathbf{M}(\mathbf{q}, \delta) \in R^{(n+m)\times(n+m)}$ is the inertia matrix, $\mathbf{D}(\mathbf{q},\dot{\mathbf{q}},\delta,\dot{\delta}) \in R^{(n+m)\times(n+m)}$ contains both rigid and flexible coriolis/centripetal terms and terms representing the interactions of the joint rigid variables with the deflections, $\mathbf{K}_f \in R^{m\times m}$ is the stiffness matrix, $\mathbf{F}_r(\mathbf{q},\dot{\mathbf{q}}) \in R^n$ is the friction, and $\mathbf{G}_r(\mathbf{q}) \in R^n$ is the gravity term. The control input is $\tau \in R^n$ and the input matrix $[\mathbf{I}\ \mathbf{B}_f]^T \in R^{(n+m)\times n}$ is generally a function of $(\mathbf{q}, \delta)$ depending on the boundary conditions (e.g., pinned-pinned, pinned-free, clamped-free) chosen by the designer. Note that the coriolis/centripetal matrix $\mathbf{D}(\mathbf{q},\dot{\mathbf{q}},\delta,\dot{\delta})$ can take several different definitions. However, among these definitions, there exists one such that the derivative of the inertia matrix $\mathbf{M}(\mathbf{q}, \delta)$ and the coriolis/centripetal matrix are related in a very particular way, that is, $\dot{\mathbf{M}}(\mathbf{q},\delta) - 2\,\mathbf{D}(\mathbf{q},\dot{\mathbf{q}},\delta,\delta)$ is *skew symmetric*, a property that is often very useful for controls design.

It is important to realize that the rank of the control effectiveness matrix rank ($[\mathbf{I}\ \mathbf{B}_f]^T$) *is* less than ($n + m$), that is, *rank* ($[\mathbf{I}\ \mathbf{B}_f]^T = n < (n + m)$. This means that the number of degrees of freedom is greater than the number of control inputs and is the major source of the problems in controlling flexible-link manipulators.

Control of Flexible-Link Robots

Control of mechanical manipulators to maintain accurate position and velocity is an important problem. Rigid-link manipulators are designed to be mechanically stiff precisely because of the difficulty of controlling flexible members. The major objectives in control of a robotic system with link flexibility are to command the tip of the flexible-link manipulator to move from one position to another as quickly as possible (point-to-point minimum-time control), or to follow preplanned desired trajectories (trajectory-following control) while keeping the oscillations of the flexible modes as small as possible. The inherent large nonlinearities of flexible-link manipulators make their control very difficult. The link flexibility makes the robot arm itself sensitive to external excitation; a small impulse signal may cause the flexible modes to oscillate wildly.

Several conventional control techniques have been studied by robotics researchers for the control of flexible-link manipulators. They fall into different categories. The first category includes some approximate techniques such as linear systems approaches, linear minimum-time control, decentralized approaches, and input-shaping techniques. Conventional control techniques cannot usually obtain very satisfactory results for fast desired motions.

The second category includes some new control approaches that take into account many of the nonlinearities. Among them are variable structure control, adaptive control, and the inverse dynamics approach (where the whole control signal is composed of a causal part and an anticausal part). To minimize residual vibrations, several constraints must be applied for the desired tip trajectories. As pointed out in Kwon and Book (1990), for a specified rigid mode trajectory, there is associated a unique flexible mode trajectory. The interactions between the desired rigid motion and the associated required flexible motion (e.g., the inverse dynamics) are very complicated and parametrically sensitive.

Since the number of independent control inputs is less than the number of the output variables in the case of flexible-link arms, the control problem is characterized as having *reduced control effectiveness.* The so-called "model matching conditions" do not hold, and the conventional control techniques usually used in the control of rigid-link robots (e.g., computed-torque control) cannot be directly applied to the control of flexible-link arm. This problem can be solved by a model-order reduction based on a singular perturbation strategy, in which the rigid modes are treated as slow-state variables, while the flexible modes and their time derivatives are treated as fast-state variables. Another approach is the "reduced-order computed torque scheme" that first removes the nonlinearities that are in the range of the control input matrix, then uses PD state-feedback loop to convert the flexible system to a set of uncoupled point-mass-like systems.

A third category of controllers includes various intelligent control schemes such as neural networks (NN) (Lewis et al., 1995) and fuzzy logic control (FLC). Many of the drawbacks mentioned above can be overcome by either FLC or NN control if these are used in conjunction with sound control engineering design practices (e.g., singular perturbation and/or feedback linearization techniques). The reason is obvious: FLC and NN are *model-free* control schemes applicable to a wide range of dynamical systems that are ill understood and ill defined. The primary reason for this model-free characteristic is the 'universal approximation property, shared by NN and FLC. Thus, by careful design, effective control actions can be generated without extended analyses based on a precise, explicit mathematical function.

Parallel-Link Robots

By far the most widely used commercial robots are the serial-link manipulators, whose links and joints alternate with one another in an open kinematic chain. This serially connected configuration is similar to that of the human arm, with each link connecting only to two neighboring links through either prismatic

or revolute joints, except for the last link which attaches to the end effector and the robot base which attaches to the floor. The advantage of the serial chain structural arrangement is that it provides a large work volume and dexterous manipulability; however, it suffers from a lack of rigidity and from accumulated actuator errors. Especially at high speed and high dynamic loading operating conditions, the serial-link manipulators show poor dynamic performance. To improve the dynamic performance and achieve high precision operations, the robot links must be made with high rigidity, which results in heavy robots with low force-output-to-manipulator-weight ratio. On the other hand, if the links can be arranged parallel to one another in a closed kinematic chain structure such that the major force components add together, then high precision operations and high force-output-to-manipulator-weight ratios can be achieved.

The Stewart Platform

The most popular and successful parallel mechanical structure is the so-called Stewart platform, which was first proposed by Stewart (1965) in 1965. As a manufacturing manipulator, the Stewart platform has two fundamental characteristics which set it apart from machine tools and industrial robots — it is a *closed kinematic system* with *parallel links.* The Stewart platform link ends are simply supported, making the manipulator system far more rigid in proportion to size and weight than any serial link robot. Furthermore, the links of the Stewart platform are arranged so that the major force components of the six actuators add together, yielding a force-output-to-manipulator-weight ratio more than one order of magnitude greater than most industrial robots.

The original Stewart platform was designed for an aircraft simulator and consisted of six linear hydraulic actuators acting in parallel between the base and the upper platform, as shown in Figure 14.12.1. All the links are connected both at the base and at the upper platform. Thus, by changing the length of each link, the position and orientation of the upper platform are able to be controlled.

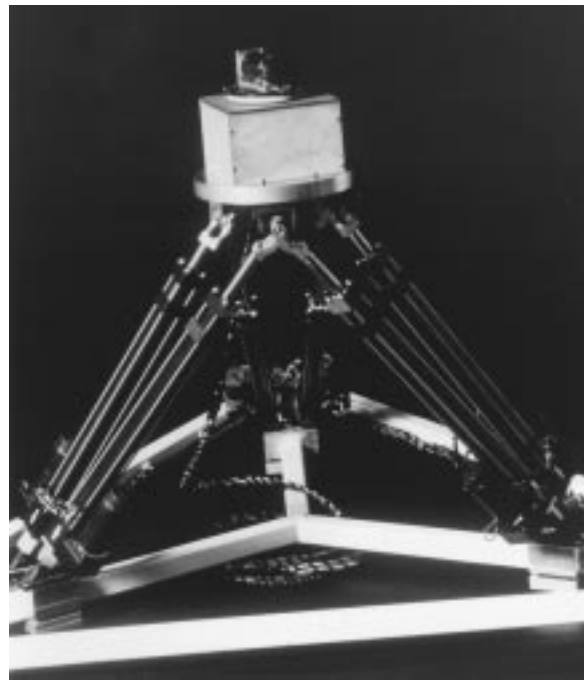


FIGURE 14.12.1 A six-degree-of-freedom *Stewart platform manipulator* developed at the Automation & Robotics Research Institute, The University of Texas at Arlington.

Some nice features of the original Stewart platform include:

- The manipulator design has six degrees of freedom, three for position and three for orientation.
- The six linear actuators are driven by six motors with each motor reacting on the base to avoid interaction between motors. Actually the manipulator can move when five of the jacks are locked merely by adjustment of the remaining jack.
- To achieve the maximum performance for a given power source, each motor operates directly on the same load (the upper platform). This makes a high payload-to-structure-weight ratio that at certain points of the workspace amounts to nearly six times the lifting capability of each individual actuator.
- Having low friction losses, with a powerful hydraulic system the manipulator can respond to commands very quickly.

It is interesting to note that the Stewart platform was not the first parallel link mechanical structure used in industry. As early as the 1950s, McGough devised a similar device for studying tire-to-ground forces and movements. The system had been in operation since 1955, but was never made known to the public until 1965 when Stewart published his journal article.

Advantages and Problems of the Stewart Platform

The Stewart platform appears simple and refined to the point of elegance. The performance mentioned above can be achieved using relatively inexpensive commercially available servo-actuator technology. The Stewart platform uses a closed kinematic chain which is structurally extremely strong and rigid, and is capable of distributing loads throughout the system. The actuator errors are not cumulative, allowing for high precision operations. However, the same closed kinematic structure that provides mechanical stiffness also complicates the forward kinematics analysis. This problem is an impediment to the derivation of dynamic equations and hence control schemes for real-time trajectory generation, which is necessary for industrial application of the manipulator (e.g., surface finishing applications).

It is known that in the case of fully parallel structures, the inverse kinematics (that is, solving for the corresponding lengths of the links given the position and orientation of the upper platform in Cartesian space) is relatively straightforward. However, the forward kinematics analysis for the fully parallel mechanism (e.g., given the length of each link, solve for the position and orientation of the upper platform in Cartesian space) is very challenging. The reason is that the kinematic equations are highly coupled and highly nonlinear.

Much effort has been devoted to finding an efficient algorithm for computing an accurate kinematic solution. To solve for the Cartesian position of the upper platform in terms of the given link lengths, thirty (30) nonlinear algebraic equations must be solved simultaneously, or polynomials of order 16 in a single variable must be solved. Due to the time-consuming nature of these procedures, it is difficult to compute the kinematic solutions on-line in real time. In Liu et al. (1993), a simplified algorithm was proposed which required to solve for only three (3) simultaneous nonlinear algebraic equations. Since the Stewart platform requires complex kinematics computations for trajectory following control, it is difficult to achieve real-time control capable of supporting high bandwidth motion.

The common feature of the forward kinematics analyses mentioned above is that there is no explicit analytical solution. Even for Liu et al.'s algorithm, it is still required to solve three nonlinear algebraic equations by numerical methods. Since there is no explicit expression available for the forward kinematics, deriving the Jacobian matrix and dynamic equations directly in link space and studying the singularity become impossible.

It is known that the Jacobian provides a transformation path which allows a two-way transformation between the link space and Cartesian space. If the Jacobian is not singular, then velocity in link space can be uniquely transformed to the corresponding velocity in Cartesian space. Particularly, if there is no movement in link space, then there is no movement in Cartesian space, so that the Stewart platform will remain rigidly fixed. However, at singular configurations, the transformation path from link space

to Cartesian space is blocked. In this case, even though there is no movement in link space, the upper platform can lose rigidity, still possibly moving along some directions. In other words, at singularities, the Stewart platform may gain extra degrees of freedom. The problem becomes even worse in that, in this situation, forces or torques in Cartesian space cannot be transformed to link space, that is, at singular positions the Stewart platform cannot be controlled to move in all directions and cannot exert force in all directions. From the applications viewpoint, investigating the conditions under which there will be singularities is important.

Thus, while the parallel link manipulators afford structural advantages, they also present severe difflculties for controller design. The control problems associated with such structures are not easy, as the systems do not satisfy most of the assumptions made in the controls literature (e.g., linearity in the parameters and feedback linearizability). Therefore, most existing control algorithms do not work well.

Manufacturing Applications of the Stewart Platform

Since proposed by Stewart (1965) in 1965, various applications of the Stewart platform have been investigated for use as aircraft simulators, as robot wrists, in mechanized assembly, and in active vibration control. As a manufacturing manipulator, the Stewart platform has great potential in automating many light machining applications such as surface finishing, edge finishing, routing, and profile milling. New manipulator applications to manufacturing processes requiring high force and power output such as combined assembly pressing are also possible. There are several light machining applications that a Stewart platform manipulator would perform with less set-up complexity and tooling cost than a serial link robot or a standard machine tool. In Figure 14.12.2 is shown a Stewart platform developed as a surface finishing milling machine.

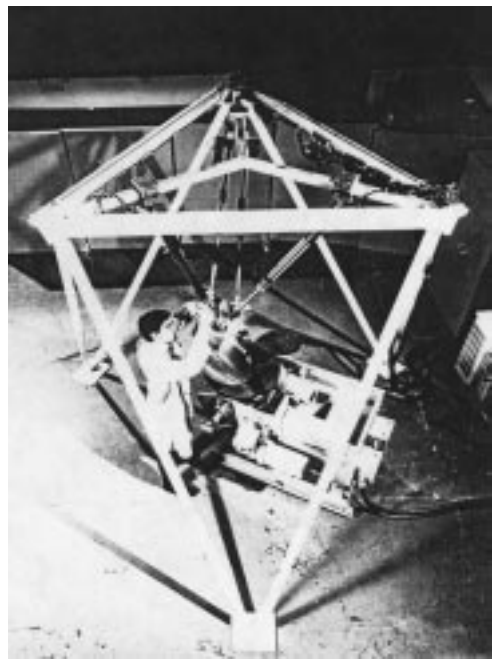


FIGURE 14.12.2 Stewart platform automated surface finishing cell.

Many applications of robotic manipulators to high precision routing can be found in the aerospace industry. A common application is trimming wing skin edges. The precision is usually achieved through the employment of expensive templates that provide a precise guide bearing for the tool to follow as it is held in the naturally compliant grip of a robotic manipulator. The major advantage for using a Stewart platform as a routing machine is that it can follow the contours of many aerospace parts without the

need for a tool guide; it is stiff enough to track the part precisely while withstanding the router cutting reactions. The cost of this contouring capability as compared to a standard five- or six-axis routing machine would be much lower.

The Stewart platform would be superior to any serial link robot as a drilling head manipulator. Virtually all applications of drilling robots in aerospace manufacturing require the use of expensive and complex end effectors of part jigs to compensate for the inaccuracy and lack of stiffness of the robots. Drilling jigs for some parts can cost as much as ten times more than the robot itself. Special end effectors are often used to apply preloads to prevent the drill from "walking" and chattering. The Stewart platform could perform many drilling tasks unaided by special tooling because of its stiffness and precision.

The industrial robot has generally not been considered to be a good milling manipulator. The Stewart platform could potentially perform contour milling of some materials with near-machine-tool accuracy. A Stewart platform milling machine with stiffness and dexterity characteristics intermediate between a large serial link robot and a five-axis mill could be built at or below the cost of a commercial serial link robot. A Stewart platform milling machine successfully used for industrial applications is shown in Figure 14.12.2.

When a direct contact tool like a grinder is used it is important to control both the tool position and the forces involved so that the substrate is not damaged. For example, when grinding mold scale a very aggressive tool may be needed, and the normal force applied can be as large as 40 to 50 lbs so long as the penetration into the surface is precisely controlled. The reactions in the surface tangent plane can be very large and could cause oscillations if not held rigidly. A Stewart platform with a constant force suspension for its tool could apply very large force with very high stiffness in one direction while being compliant and forceful in the normal direction.

Defining Terms

Accuracy. The degree to which the actual and commanded positions (of, e.g., a robot manipulator) correspond for computed as opposed to taught positions.

Adaptive Control. A large class of control algorithms where the controller has its own internal dynamics and so is capable of learning the unknown dynamics of the robot arm, thus improving performance over time.

AML. A Manufacturing Language — a robot programming language.

APT. Automatic Programming of Tools — a robot programming language.

Cell Decomposition. An approach to path planning where the obstacles are modeled as polygons and the free space is decomposed into cells such that a straight line path can be generated between any two points in a cell.

Compliance. The inverse of "stiffness" — useful in end effectors tooling whenever a robot must interact with rigid constraints in the environment.

Computed-Torque Control. An important and large class of robot arm controller algorithms that relies on subtracting out some or most of the dynamical nonlinearities using feedforward compensation terms including, e.g., gravity, friction, coriolis, and desired acceleration feedforward.

End Effector: Portion of robot (typically at end of chain of links) designed to contact world:

- **Compound.** A cluster of multiple end effectors and tooling mounted on the robot wrist.
- **Active.** An end effector with sensing and servo control of the grasp forces and/or finger motions.
- **Prehensile.** An end effector that holds parts between fingertips or encircled by fingers.
- **Vacuum.** A nonprehensile end effector that uses suction cups to hold parts.
- **Dextrous.** A hand with the ability to manipulate parts in the fingers and actively control grasp forces.

Feedback Linearization. A modern approach to robot arm control that formalizes computed-torque control mathematically, allowing formal proofs of stability and design of advanced algorithms using Lyapunov and other techniques.

Flexible-Link Robot. Lightweight mechanical structures where vibration and flexibility of the links must be taken into account in controller design. They possess favorable features including lower manufacturing costs, higher motion speeds, better performance, and easier transportation and setup.

Force Control. A class of algorithms allowing control over the force applied by a robot arm, often in a direction normal to a prescribed surface while the position trajectory is controlled in the plane of the surface.

Forward Kinematics. Identification of task coordinates given configuration.

Fuzzy Logic Control. A multilevel logic controller, which is different from the conventional dual (two-level) logic in which only two values (true and false) may be assigned to each state variable. Fuzzy logic controllers have advantages in being robust to disturbances and not requiring an explicit mathematical model for the design process. They consist of three parts: the fuzzifier, the rulebase, and the defuzzifier.

Grasp Isotropy. A measure of how uniformly forces and motions can be controlled in different directions.

IGES. International Graphics Exchange Specification — a data exchange standard.

Inverse Kinematics. Identification of possible configurations given task coordinates.

I/O Device. Input/output device — a port through which external information is connected to a computer. I/O devices may be A/D, which converts analog signals to digital, D/A, which converts digital signals to analog, or binary, which passes digital signals.

Joint Variables. Scalars specifying position of each joint — one for each degree of freedom.

Kitting. The process of taking parts from bulk and placing them on a *kit tray,* which is an organized group of all parts required to assemble a single product or subassembly.

Learning Control. A class of control algorithms for repetitive motion applications (e.g., spray painting) where information on the errors during one run is used to improve performance during the next run.

Linearity in the Parameters. A property of the robot arm dynamics, important in controller design, where the nonlinearities are linear in the unknown parameters such as unknown masses and friction coefficients.

Manipulator Jacobian. Matrix relating joint velocities to task coordinate velocities - configuration dependent.

Mechanical Part Feeders. Mechanical devices for feeding parts to a robot with a specified frequency and orientation. They are classified as vibratory bowl feeders, vibratory belt feeders, and programmable belt feeders.

Mobile Robot. A special type of manipulator which is not bolted to the floor but can move. Based on different driving mechanisms, mobile robots can be further classified as wheeled mobile robots, legged mobile robots, treaded mobile robots, underwater mobile robots, and aerial vehicles.

Path Planning. The process of finding a continuous path from an initial robot configuration to a goal configuration without collision.

PD-Gravity Control. A special case of computed-torque control where there is a PD outer control loop plus a gravity compensation inner control loop that makes the DC values of the tracking errors equal to zero.

Pinch Grasp. A grasp in which a part is clamped between fingertips.

Pixel. Picture element — one point of an image matrix in image processing terminology.

Prismatic Joint. Sliding robot joint which produces relative translation of the connected links.

Redundant Manipulator. Manipulator for which the number of joint variables is greater than the number of task coordinates.

Remote-Center Compliance (RCC). A compliant wrist or end effector designed so that task-related forces and moments produce deflections with a one-to-one correspondence (i.e., without side effects). This property simplifies programming of assembly and related tasks.

Revolute Joint. Rotary robot joint producing relative rotation of the connected links.

Robot Axis. A direction of travel or rotation usually associated with a degree of freedom of motion.

Robot Joint. A mechanism which connects the structural links of a robot manipulator together while allowing relative motion.

Robot Link. The rigid structural elements of a robot manipulator that are joined to form and arm.

Robust Control. A large class of control algorithms where the controller is generally nondynamic, but contains information on the maximum possible modeling uncertainties so that the tracking errors are kept small, often at the expense of large control effort. The tracking performance does not improve over time so the errors never go to zero.

SCARA. Selectively compliant assembly robot arm.

SET. (Specification for Exchange of Text) — a data exchange standard.

Singularity. Configuration for which the manipulator jacobian has less than full rank.

Skew Symmetry. A property of the dynamics of rigid-link robot arms, important in controller design, stating that $\dot{M} - {}^1/_2 V_m$ is skew symmetric, with M the inertia matrix and V_m the coriolis/centripetal matrix. This is equivalent to stating that the internal forces do no work.

Stewart Platform Manipulator. A special type of parallel-link robot consisting of six identical linear actuators in parallel, an upper platform, and a base. One end of each actuator connects to the base, and the other to the upper platform with two- or three-degrees-of-freedom joints. This manipulator has a greater force-to-weight ratio and finer positioning accuracy than any commercial serial-link robot.

Task Coordinates. Variables in a frame most suited to describing the task to be performed by manipulator.

VDAFS. (Virtual Data Acquisition and File Specification) — a data exchange standard.

Visibility Graph. A road map approach to path planning where the obstacles are modeled as polygons. The visibility graph has nodes given by the vertices of the polygons, the initial point, and the goal point. The links are straight line segments connecting the nodes without intersecting any obstacles.

Voronoi Diagram. A road map approach to path planning where the obstacles are modeled as polygons. The Voronoi diagram consists of line having an equal distance from adjacent obstacles; it is composed of straight lines and parabolas.

Wrap Grasp. A grasp in which fingers envelope a part, to sustain greater loads.

References

Anderson, R.J. and Spong, M.W. 1989. Bilateral control of teleoperators with time delay. *IEEE Trans. Robotics Automation.* 34(5):494–501.

Asfahl, C.R. 1992. *Robotics and Manufacturing Automation.* 2nd ed. John Wiley & Sons, New York.

Ballard, D.H. and Brown, C.M. 1982. *Computer Vision.* Prentice-Hall, Englewood Cliffs, NJ.

Book, W.J. 1984. Recursive Lagrangian dynamics of flexible manipulator arms. *Int. J. Robotics Res.* 3(3):87–101.

Boothroyd, G. and Dewhurst, P. 1985. Part presentation costs in robot assembly. *Assembly Automation,* 138–146.

Boothroyd, G., Dewhurst, P., and Knight, W. 1994. *Product Design for Manufacture and Assembly.* Marcel Dekker, New York.

Bottema, O. and Roth, B. 1979. *Theoretical Kinematics,* North Holland, Amsterdam.

Bralla, J.G. (ed.). 1986. *Handbook of Product Design for Manufacturing,* McGraw-Hill, New York, 7-75, 7-100.

Craig, J. 1985. *Adaptive Control of Mechanical Manipulators.* Addison-Wesley, Reading, MA.

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

Critchlow, A.J. 1985. *Introduction to Robotics.* Macmillan, New York.

Decelle, L.S. 1988. Design of a robotic workstation for component insertions. *ATEJT Tech. J.* 67(2):15–22.

Denavit, J. and Hartenberg, R.S. 1955. A kinematic notation for lower-pair mechanisms based on matrices, *J. Appl. Mech.* 22:215–221,

Duffy, J. 1980. *Analysis of Mechanisms and Robot Manipulators,* John Wiley & Sons, New York.

Elfes, A. 1987. Sonar-based real-world mapping and navigation. *IEEE J. Robotics Automation.* RA-3(3):249–265.

Fraden, J. 1993. *AIP Handbook Of Modern Sensors, Physics, Design, and Applications.* American Institute of Physics, New York.

Fu, K.S., Gonzalez, R.C., and Lee, C.S.G. 1987. *Robotics.* McGraw-Hill, New York.

Fuller, J.L. 1991. *Robotics: Introduction, Programming, and Projects.* Macmillan, New York.

GMF Robotics Training and Documentation Department. 1985. *Paint Processing: Concepts and Practices.* GMF Robotics Corporation, Troy, MI.

Groover, M.K., Weiss, M., Nagel, R.N., and Odrey, N.G. 1986. *Industrial Robotics: Technology, Programming, and Applications.* McGraw-Hill, New York.

Gruver, W.A., Soroka, B.I., and Craig, J.J. 1984. Industrial robot programming languages: a comparative evaluation. *IEEE Trans. Syst. Man Cybernetics.* SMC-14(4).

Hayati, S., Tso, K., and Lee, T. 1989. Dual arm coordination and control. *Robotics.* 5(4):333–344.

Hollerbach, J.M., Hunter, I.W., and Ballantyne, J. 1992. A comparative analysis of actuator technologies for robotics. In *Robotics Review 2,* O. Khatib, J. Craig, and T. Lozano-Perez, (eds.). MIT Press, Cambridge, MA, 299–342.

Hollis, R.L., Allan, A.P., and Salcudean, S. 1988. A six degree-of-freedom magnetically levitated variable compliance fine motion wrist. In *Robotics Research, the 4th Int. Symp.,* R. Bolles and B. Roth., (eds.). MIT Press, Cambridge, MA, 65–73.

Jacobsen, S., Wood, J., Knutti, D.F., and Biggers, K.B. 1984. The Utah/M.I.T. dextrous hand: work in progress. *Int. J. Robotics Res.* 3(4):Winter.

Jamshidi, M., Lumia, R., Mullins, J., and Shahinpoor, M. *Robotics and Manufacturing: Recent Trends in Research, Education, and Applications*, Vol. 4. ASME Press, New York.

Klafter, R.D., Chmielewski, T.A., and Negin, M. 1989. *Robotic Engineering: An Integrated Approach.* Prentice-Hall, Englewood Cliffs, NJ.

Latombe, J.C. 1991. *Robot Motion Planning.* Kluwer Academic Publishers, Amsterdam.

Lee, K.-M. 1991. Flexible part-feeding system for machine loading and assembly. I. A state-of-the-art survey. II. A cost-effective solution. *Int. J. Prod. Economics.* 25:141–168.

Lee, K.-M. and Blenis, R. 1994. Design concept and prototype development of a flexible integrated vision system, *J. Robotic Syst.*, 11(5):387–398.

Lee, K.-M. and Li, D. 1991. Retroreflective vision sensing for generic part presentation. *J. Robotic Syst.* 8(1):55–73.

Leu, M.C. 1985. Robotics software systems. *Rob. Comput. Integr. Manuf.* 2(1):1–12.

Lewis, F.L., Abdallah, C.T., and Dawson, D.M. 1993. *Control of Robot Manipulators.* Macmillan, New York.

Lewis, F.L., Liu, K., and Yesildirek, A. 1995. Neural net robot controller with guaranteed tracking performance. *IEEE Trans. Neural Networks.* 6(3):703–715.

Liu, K., Fitzgerald, J.M., and Fewis, F.L. 1993. Kinematic analysis of a Stewart platform manipulator. *IEEE Trans. Ind. Electronics.* 40(2):282–293.

Lozano-Perez, T. 1983. Robot programming. *Proc. IEEE.* 71(7):821–841.

McClamroch, N.H. and Wang, D. 1988. Feedback stabilization and tracking of constrained robots. *IEEE Trans. Automat. Control.* 33:419–426.

Mujtaba, M.S. 1982. The AL robot programming language. *Comput. Eng.* (2):77–86.

Nichols, H.R. and Lee, M.H. 1989. A survey of robot tactile sensing technology. *Int. J. Robotics Res.* 8(3):3–30.

Nomura, H. and Middle, J.E. 1994. *Sensors and Control Systems in Arc Welding.* Chapman and Hall, 2-6 Boundary Row, London SE1 8HN, UK.

Okabe, A., Boots, B., and Sugihara, K. 1992. *Spatial Tessellations, Concepts and Application of Voronoi Diagrams.* John Wiley & Sons, New York.

Pertin-Trocaz, J. 1989. Grasping: a state of the art. In *The Robotics Review 1*, O. Khatib, J. Craig, and T. Lozano-Perez, Eds. MIT Press, Cambridge, MA, 71–98.

Priest, J.W. 1988. *Engineering Design for Producibility and Reliability.* Marcel Dekker, New York.

Rossi, M. 1985. Dialogues. *Manuf. Eng.* October: 41:24.

Sandler, B.Z. 1991. *Robotics, Designing the Mechanisms for Automated Machinery.* Prentice-Hall, Englewood Cliffs, NJ.

Shimano, B.E., Geschke, C.C., and Spalding, C.H., III. 1984. Val-II: a new robot control system for automatic manufacturing. *Proc. Int. Conf. Robotics.* March 13–15:278–292.

Siciliano, B. 1990. Kinematic control of redundant robot manipulators: a tutorial. *J. Intelligent Robotic Syst.* 3(3):201–210.

SILMA Incorporated. 1992. *SILMA CimStation Robotics Technical Overview.* SILMA Incorporated, 1601 Saratoga-Sunnyvale Road, Cupertino, CA.

Slotine, J.-J. 1988. Putting physics in control: the example of robotics. *Control Syst. Mag.* 8 (December):12–15.

Snyder, W.E. 1985. *Industrial Robots: Computer Interfacing and Control.* Prentice-Hall, Englewood Cliffs, NJ.

Spong, M.W. and Vidyasagar, M. 1989. *Robot Dynamics and Control.* John Wiley & Sons, New York.

Stauffer, R.N. 1984. Robotic assembly. *Robotics Today.* October.

Stevens, G.T. 1994. *The Economic Analysis of Capital Expenditures for Managers and Engineers.* Ginn Press, Needham Heights. MA.

Stewart, D. 1965. A platform with six degrees of freedom. *Proc. Inst. Mech. Engr.* (*London)* 180(15):371–386.

Tanner, W.R. 1994. Product design and production planning. In *CRC Handbook for Robotics.* CRC Press, Boca Raton, FL, 537.

Taylor, R.H., Summers, P.D., and Meyers, J.M. 1982. AML: a manufacturing language. *Int. J. Robotics Res.* (1):19–41.

Tsai, R.Y. and Lenz, R.K. 1989. A new technique for fully autonomous and efficient 3D robotics hand/eye calibration. *IEEE Trans. Robotics Automation.* 5(3):345–358.

Tzou, H.S. and Fukuda, T. 1992. *Precision Sensors, Actuators, and Systems.* Kluwer Academic, Publishers, Amsterdam.

Winston, P.H. 1984. *Artificial Intelligence.* Addison-Wesley, Reading, MA.

Wright, P.K. and Cutkosky, M.R. 1985. Design of grippers. In *The Handbook of Industrial Robotics,* S. Nof, (ed.). John Wiley & Sons, New York, chap. *2*.4.

Additional Reading

For a less mathematical treatment of robotics, including topics in manufacturing and programming, see the book by Fuller (1991). For further reading on information flow and computer science aspects of robotics, see the chapter on "Robotics" in the *CRC Handbook of Computer Science Engineering.* More details on manufacturing and industrial robot applications are found in Asfahl (1985) and Groover et al. (1986). For more on dynamics and control of robot manipulators one may examine books by Lewis et al. (1993) or Spong and Vidyasagar (1989). Robotics including topics in control, vision processing, and programming aspects is discussed in Fu et al. (1987). A constant source of articles on all aspects of robotics is the *IEEE Transactions on Robotics and Automation.*