INTELLIGENT INDUSTRIAL ROBOTS

Wanek Golnazarian
46 Mansfield Ave.
Essex Jct., VT 05452
and
Ernest L. Hall
Center for Robotics Research, ML 72
University of Cincinnati
Cincinnati, OH 45221

ABSTRACT

An intelligent industrial robot is a remarkably useful combination of a manipulator, sensors and controls. The use of these machines in factory automation can improve productivity, increase product quality and improve competitiveness. Robots have been created to perform a wide variety of tasks spanning from educational robots in classrooms, to arc welding robots in the automobile industry, to teleoperated robot arms and mobile robots in space. This chapter focuses on the application of industrial robots within a manufacturing setting, and their contribution to further enhance their capabilities for flexibility in automation. In proper selection of a robot, one has to consider the various *robot characteristics* such as the way the robot links are connected and controlled at each joint. Next, a comprehensive review of the *robot kinematics, dynamics and control strategies* along with a treatment of the *artificial neural network* will describe the recent attempts to advance the development of intelligent control systems for industrial robots.

1. Introduction

The Robot Industries Association (RIA) has defined an *industrial robot* as "a reprogrammable multi-functional manipulator designed to move material, parts, tools or specialized devices, through variable programmed motions for the performance of a variety of tasks." The most common types of manipulators may be modeled as an open kinematic chain of rigid bodies called *links*, interconnected by *joints*. Some have closed

kinematic chains such as four bar mechanisms for some links. The typical industrial robot is mounted on a fixed pedestal base which is connected to other links. The end-effector attaches to the free end and enables the robot to manipulate objects and perform required tasks. Hard automation is differentiated because of the single function. Computer numerical control (CNC) machines have a smaller variety of tasks.

A more general definition of a robot is: a general-purpose, reprogrammable machine capable of processing certain human-like characteristics such as judgment, reasoning, learning, and vision. Although industrial robots have been successfully used in a variety of manufacturing applications, most robots used are deaf, dumb, blind, and stationary [Hall and Hall, 1985]. In fact, they have been used more like automated machines where the jobs are repetitive, dirty, dangerous, or very difficult. An industrial robot is limited in sensory capabilities (vision and tactile), flexibility, adaptability, learning and creativity.

Current researchers are attempting to develop *intelligent robots*. Hall and Hall [1985] define an intelligent robot as one that responds to changes to its environment through sensors connected to its controller. Much of the research in robotics has been concerned with vision (eyes) and tactile (fingers). *Artificial intelligence* (AI) programs using heuristic methods have somewhat solved the problem of adapting, reasoning, and responding to changes in the robot's environment. For example, one of the most important considerations in using a robot in a workplace is human safety. A robot equipped with sensory devices that detect the presence of an obstacle or a human worker within its work space, could automatically shut itself down in order to prevent any harm to itself and/or the human worker.

Kohonen [1988] suggests a higher degree of learning is possible with the use of *neural computers*. The intelligent robot is supposed to plan its action in the natural environment, while at the same time performing non-programmed tasks. For example, the learning of locomotion in an unknown environment is extremely difficult to achieve by formal logic programming. Typical robot applications in manufacturing assembly tasks may require locating components and placing them in random positions.

2. Automation and Robotics

In the manufacturing industry, the term *automation* is very common. It was introduced in the 1940s at the Ford Motor Company, where specialized machines helped manufacture high volume production of mechanical and electrical parts. However, the high cost of tooling for new models limit production flexibility. This type of automation is referred to as *hard or fixed automation*. The advantage is high production rate. Hard automation is still being used for the production of light bulbs at General Electric at a rate of two billion light bulbs per year. [Asfahl, 1992].

In the early 1950s numerically controlled (NC) machine tools were introduced. Later, NC machines evolved into computer numerical control (CNC) machines. Since CNCs can be easily reprogrammed through software to accommodate high product variety relative to hard automation, they are referred to as *soft or flexible automation.* The reprogrammability of flexible automation equipment gives it a key advantage over hard automation. Robots and their development are a natural extension of the concepts of NC and CNC. The robot's reprogrammable feature has enhanced the flexibility of the automation systems and is referred to as an example of soft or flexible automation [Asfahl, 1992].

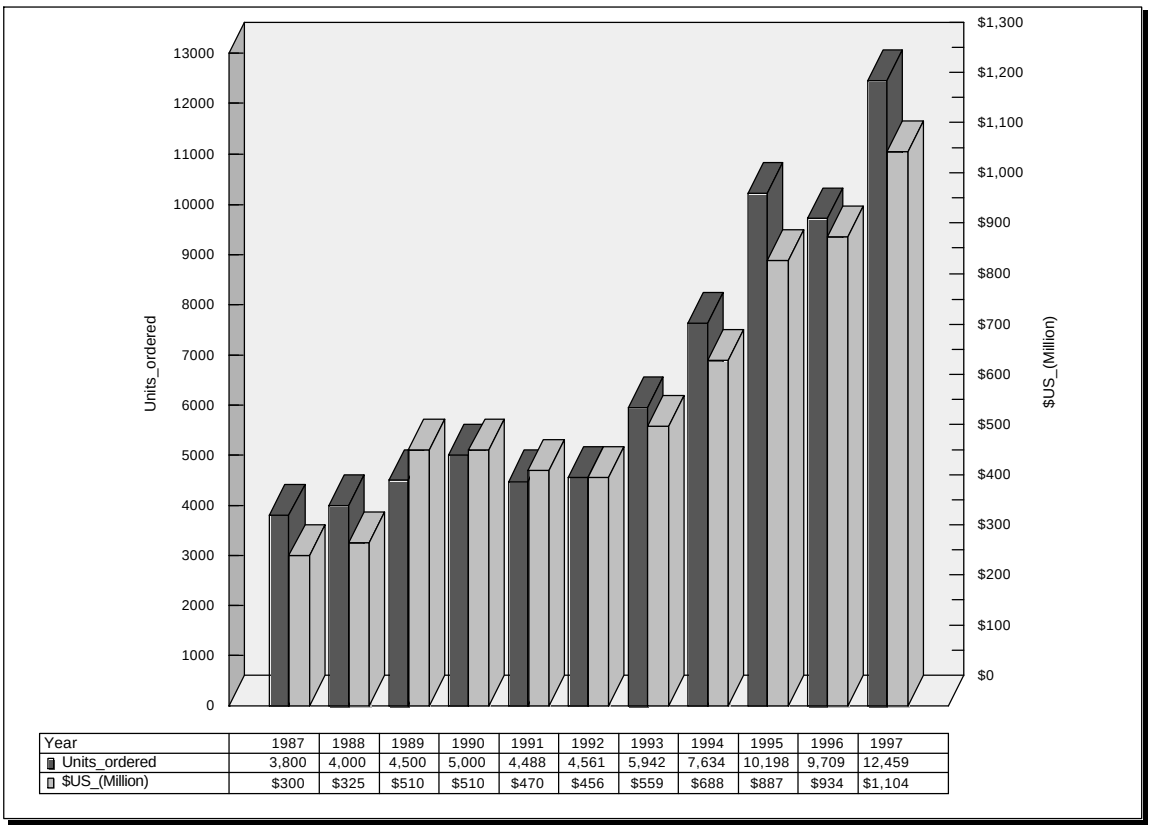
Industrial robots were first commercially marketed in 1956 by a firm named Unimation. In 1961, Ford Motor Company was the first to use a Unimate robot to unload a die-casting machine [Odrey, 1993]. Since then, the automobile industry has been largely responsible for development of the flexible manufacturing system (FMS) with industrial robots. Introducing robot technology into the factories has improved productivity, quality, and flexibility which could not be realized on the basis of hard or fixed automation structure. However, robots in the early 1980's were limited in capabilities and performance due to their drive mechanisms, controller systems and programming environment. They weren't well suited for most manufacturing tasks and were often too expensive. As a result, the increase of robot installation through the mid 80's turned into a significant slump which lasted into the early nineties [Holusha, 1994]. Due to the steep decline in robot orders,

many US manufacturers chose to pull out and ceded the market to foreign competitors. Adept Technology Inc. (San Jose, CA) is the only major US robot manufacturer to survive in the $700 million market with about $60 million in annual sales [Sinton, 1995].

The introduction of robots is often justified on the basis that they perform consistently and productively. Often a few people suffer the loss of employment. Others believe robotic technology creates skilled jobs with greater creativity. However, the question of the social impact of robotics has yet to be adequately addressed [McKerrow, 1991].

Lower cost, greater reliability, and targeting tasks that are too difficult or dangerous for humans have led to a renewed interest in robotics during the early nineties. Finally, US manufacturers are realizing the significant impact robots can have in improving productivity, quality, flexibility, and time-to-market. Process repeatability and final product uniformity are more important than labor cost. And unlike dedicated machinery (fixed automation) which is designed to perform a specific task, today's robot can be used for multiple products with ease. It has become the critical element in many applications such as welding, sealing, and painting. Other applications (material handling, assembly, and inspection) in non-automotive industries such as electronics, consumer products, pharmaceutical, and service, are maturing rapidly [Weil, 1994]. According to the Robotic Industries Association (RIA), the robot industry is in a recovery mode, particularly in the US, as manufacturers invest in robotics to stay competitive. Figure 1 illustrates the renewed strength in robotics since 1987 [Holusha, 1994].

Record breaking shipments from US manufacturers has totaled 12,459 robots valued at $ 1.1 billion in 1997. This represents a 172% increase in robotic systems and a 136% increase in revenues since 1992. According to new statistics released by the Robotic Industries Association, the worlds population of installed robots at the end of 1997 exceeded 500,000. The country that has the largest population of industrial robots is Japan (400,000); it is followed by the USA (80,000), and then the Western European nations combined (120,000).



Year	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997
Units_ordered	3,800	4,000	4,500	5,000	4,488	4,561	5,942	7,634	10,198	9,709	12,459
$US_(Million)	$300	$325	$510	$510	$470	$456	$559	$688	$887	$934	$1,104

Figure 1 Industrial robot market

Industrial robot applications are not limited to automotive industries. A summary of robot applications, along with the share of robot market in the USA for the year 1995, is displayed in Table 1. Traditional applications of spot and arc welding, and spray painting continue to dominate. The market share for assembly robots has grown over the past decade. The discussion that follows, however not all-inclusive, offers an overview of such applications by type. In addition, excellent reviews of existing robot applications are given by Odery [1993] and examples and case studies in the textbook by Asfahal [1992].

TABLE 1. USA robot market [RIA, 1995]

Application	Percent
welding	53.0
material handling	24.0
assembly	10.0
spray coating	8.5
inspection	1.0
other	3.5

Welding. Welding is the process of joining metals by fusing them together, commonly used in the fabrication industry. Spot and arc welding are the top industrial robot applications as of the early 1990's. Continuous arc welding is a more difficult process for a robot than spot welding. An arc weld has to be placed along a joint between two pieces of metal. Part misalignment or dimensional variations in parts are the major causes of problems often encountered in robot arc welding. Researchers are investigating the variety of sensors that can provide feedback information for the purpose of guiding the weld path [Klafter, Chmielewski, and Negin, 1989].

Robotic arc welding cells provide the following advantages: higher productivity as measured by greater "arc-on" time, reduced worker fatigue, improved safety, and decreased idle time. Broshco, a division of Jay Plastic (Mansfield, OH), is a producer of robotic welded assemblies for automobile seating units. It purchased a Motoman ArcWorld 2000, with six axes of motion in order to improve productivity per man hour, weld quality, and work environment. The installation is part of the automation strategy for the new product line [Rottenbach, 1992].

Material handling. Applications of this type refer to grasping and movement of work parts from one point to another. Examples include machine loading and unloading, automated palletizing, and automated warehousing. Material handling applications are

typically simple with regard to degrees of freedom required. Specialized gripper design is an active area of research, where quick retooling enables the robot during the production cycle. Automated palletizing operations may require additional degrees of freedom with more sophisticated drive mechanisms, controllers, and expert programming features [Hall, Slutzky, and Shell, 1989]. In GE's Electrical Distribution & Control plant (Morristown, TN) five CG-120 gantry robots (C&D Robotics) palletize a range of product sizes (cartons range from 8x8 in^2. to 12x40 in^2.). Since 1989, GE's robots have provided versatility over manual and conventional palletizing and eliminated injuries caused by manual lifting [Labrenz, 1992].

Spray coating. Spray coating is the process of applying paint or a coating agent in thin layers to an object, resulting in a smooth finish. Industrial robots are suitable for such applications, where a human worker is in constant exposure to hazardous fumes and mist which can cause illness and fire. In addition, industrial robots provide a higher level of consistency than the human operator. Continuous-path control is required to emulate the motions of a human worker, with flexible multiple programming features for quick changeovers. Hydraulic drives are recommended to minimize electrical spark hazards. Chrysler Corp. has found an alternative process to fill the seams on its new LH vehicles to eliminate warping and inconsistent filling. In 1992, a four-robot station (Nachi Robotic Systems Inc.) at Chrysler's completely retooled plant (Ontario, Canada) successfully replaced the manual filling of the seams with silicon-bronze wire. [Sauer, 1992].

Assembly. Many products designed for human assembly cannot be assembled automatically by industrial robots. The integration of product design and assembly design, belongs to the concept of *design for manufacturability* [Boothroyd, Poli, and Murch, 1982]. More recent research in design for assembly has been completed at the University of Cincinnati [Hoekstra, 1992]. Design for manufacturability results in the design of factories for robots. Fewer parts, complex molding, and subassemblies which allow a hierarchical approach to assembly has lead to robotic applications. For example, the IBM Proprinter, which was designed for automatic assembly, uses 30 parts with mating capabilities (screwless) to assemble and test in less than five minutes. For part-mating

applications such as inserting a semiconductor chip into a circuit board (peg-in-hole), remote center compliance (RCC) devices have proved to be an excellent solution. More recently, Reichle (Wetzikon, Switzerland), a midsize manufacturer of telecommunications switching equipment, needed a system to automate the labor intensive assembly of electronic connectors. Using hardware and software from Adept Technology Inc. (San Jose, CA), three AdeptOne robots reduce manpower requirements from 10 to 2. In addition, the system provides speed and a high degree of robotic accuracy [Farnum, 1992].

Inspection & Measurement. With a growing interest in product quality, the focus has been on "zero-defects". However, the human inspection system has somehow failed to achieve its objectives. Robot application of vision systems have provided services in part location, completeness and correctness of assembly products, and collision detection during navigation. Current vision systems, typically two-dimensional systems, compare extracted information from objects to previously trained patterns for achieving their goals. Coordinate measuring machines (CMM) are probing machining centers used for measuring various part features such as concentricity, perpendicularity, flatness, and size in a three dimensional rectilinear or polar coordinate systems. As an integrated part of a flexible manufacturing system, the CMMs have reduced inspection time and cost considerably, when applied to complex part measurement.

Machine vision applications require the ability to control both position and appearance in order to become a productive component of an automated system. This may require a three-dimensional vision capability which is an active research area [Nurre and Hall, 1989]. At Loranger Manufacturing (Warren, PA), 100% inspection of the rim of an ignition part is required for completeness. Using back lighting and with the camera mounted in line, each rim is viewed using pixel connectivity. When a break in pixel is detected an automatic reject arm takes the part off the line [Davies, 1992].

Other processing applications for robot use include machining (grinding, deburring, drilling, and wire brushing) and water jet cutting operations. These operations employ

powerful spindles attached to the robot end-effector, rotating against a stationary piece. For example, Hydro-Abrasive Machining (Los Angeles, CA) uses two gantry robots with abrasive water jet machining heads. They cut and machine anything from thin sheet metal to composites several inches thick with tolerances of .005 in. for small parts to .01 in. for larger parts [Davies, 1992]. Flexible manufacturing systems combined with robotic assembly and inspection, on the one hand, and intelligent robots with improved functionality and adaptability on the other hand, will initiate a structural change in the manufacturing industry for improved productivity for years to come.

3. Robot Characteristics

In this section an industrial robot is considered to be an open kinematic chain of rigid bodies called links, interconnected by joints with actuators to drive them. A robot can also be viewed as a set of integrated subsystems [Klafter, Chmielewski, and Negin, 1989]:

1. **Manipulator.** The mechanical structure that performs the actual work of the robot, consisting of links and joints with actuators.
2. **Feedback devices**. Transducers that sense the position of various linkages and/or joints that transmit this information to the controller.
3. **Controller.** Computer used to generate signals for the drive system so as to reduce response error in positioning and applying force during robot assignments.
4. **Power source.** Electric, pneumatic, and hydraulic power systems used to provide and regulate the energy needed for the manipulator's actuators.

The manipulator configuration is an important consideration in the selection of a robot. It is based on the kinematic structure of the various joints and links and their relationships with each other. There are six basic motions or degrees of freedom to arbitrarily position and orient an object in a three-dimensional space (three arm and body motions and three wrist movements). The first three links, called the major links, carry the gross manipulation tasks (positioning). Examples are arc welding, spray painting, and water jet cutting applications. The last three links, the minor links, carry the fine manipulation tasks (force/tactile). Robots with more than six axes of motion are called redundant degree of freedom robots. The redundant axes are used for greater flexibility such as obstacle avoidance in the workplace. Examples are parts assembly, and machining applications.

Typical joints are revolute (R) joints, which provide rotational motion about an axis, and prismatic (P) joints, which provide sliding (linear) motion along an axis. Using this notation, a robot with three revolute joints would be abbreviated as RRR, while one with two revolute joints followed by one prismatic joint would be denoted RRP.

There are five major mechanical configurations commonly used for robots: Cartesian, cylindrical, spherical, articulated, and SCARA (Selective Compliance Articulated Robot for Assembly). Workplace coverage, particular reach and collision avoidance, are important considerations in the selection of a robot for an application. Table 2 provides a comparative analysis of the most commonly used robot configurations along with their percent of use. Details for each configuration are documented by Ty and Tien [1993]. Figure 2 shows the arm geometries for the most commonly used robot configuration: a) Cartesian (PPP), b) cylindrical (RPP), c) articulated (RRR), d) spherical (RRP), and e) SCARA (RRP). However, there are other configurations used in either research or specialized applications.

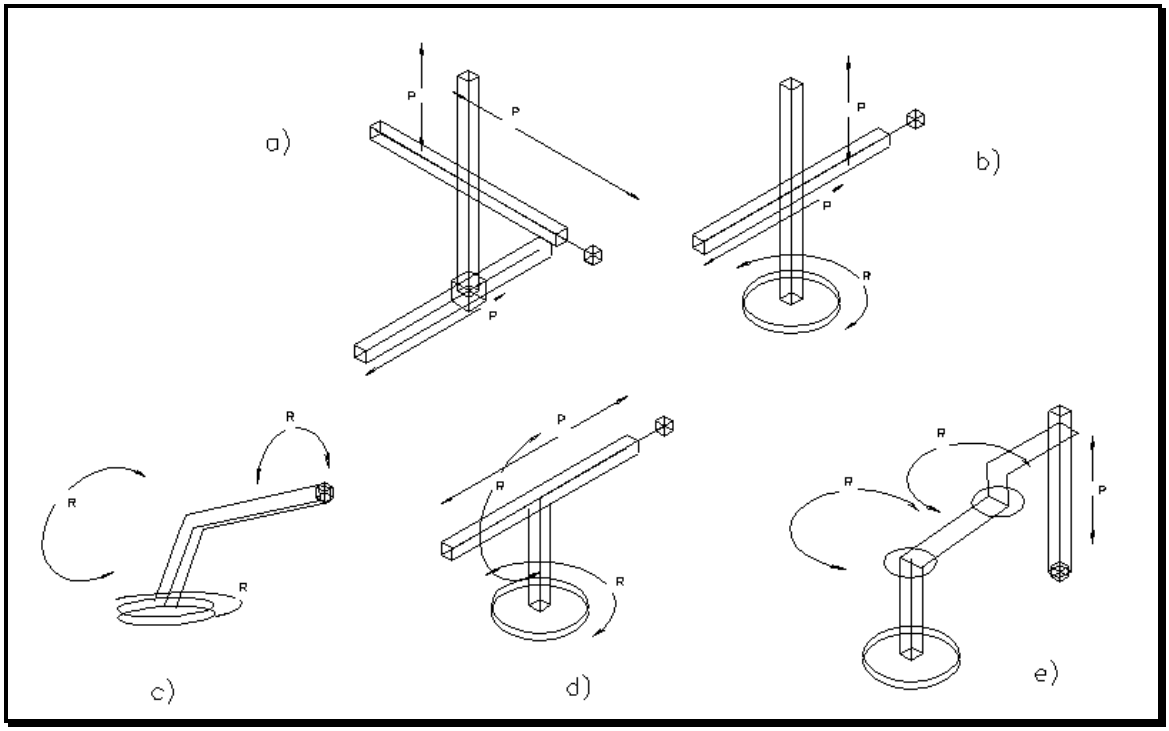


Figure 2 Common robot arm geometries

The gantry configuration is essentially a Cartesian configuration with the robot mounted on an overhead track. A redundant robot configuration may be used when more than six degrees of freedom is needed to reach a particular orientation. All the above configurations are rigid serial links. A parallel robot configuration known as Steward platform, also exists. Lightweight flexible robot arms also exist for faster speed, and low energy consumption. Other classifications are possible based on transmission type**.** Industrial robots can be direct-driven arms (DDArm) and indirect driven arms. Most industrial robots used today are indirect-drive geared mechanisms. This drive mechanism may suffer from poor dynamic response under heavy mechanical load and gear friction, and backlash.

Table 2. Comparisons of robot configurations

Robot Application	Confg.	% use	Advantage	Disadvantage
Cartesian assembly & mach. loading	PPP	18	linear motion in 3D; simple kinematics; rigid structure	poor space utilization; limited reach; low speed
Cylindrical assembly & mach. loading	RPP	15	good reach; simple kinematics	restricted work space; variable resolution
Spherical automotive manufacturing	RRP	10	excellent reach; very powerful w/ hydraulic Drive	complex kinematics; variable resolution
Articulated spray coating	RRR	42	maximum flexibility; large work envelope; high speed	complex kinematics; rigid structure; difficult to control
SCARA* assembly & insertion	RRP	15	horiz. compliance; high speed; no gravity effect	complex kinematics; variable resolution; limited vert. motion

* **S**elective **C**ompliance **A**rticulated **R**obot for **A**ssembly
(Source for the percent of use: V.D. Hunt, Robotics Sourcebook, New York: Elsevier, 1988.)

In DDArms no gears or other mechanical power conversion devices are used. High torque motors are directly coupled to each joint, eliminating gear friction and increasing stiffness and speed. However, they are more difficult to control because the inertia changes and gravity effects are no longer suppressed by high gear ratios [Kaiser, 1993].

Experimental DDArms have been built at the MIT AI Laboratory and CMU Robotics Institute. The AdeptOne robot (four-axis SCARA) is the first commercially available direct-drive robot.

4. Robot Control Strategies

Robot manipulators typically perform a given task repeatedly. Yoshikawa [1990] defines the fundamental elements of tasks performed by robots as:

1. **Gross manipulation**: to move the end-effector, with or without a load, along a desired path (*position control*).
2. **Fine manipulation**: to exert a desired force on an object when in contact with it (*force control*).

Industrial manipulators require a high level of accuracy, speed, and stability in order to provide the mobility and flexibility needed to perform the range of tasks required in a manufacturing setting. An industrial robot is useful when it is capable of controlling its movement and the forces it applies to its environment. Accuracy, repeatability, and stability are important considerations when designing a suitable robot controller.

One of the major objectives of a robot is to position its tool from one point to another while following a planned trajectory. This is called controlled path motion or the *motion trajectory problem*. Motion trajectory control of the end-effector applies to the tasks in the first category, gross manipulation, as defined by Yoshikawa. Robots, in general, use the first three axis for gross manipulation (position control) while the remaining axis orient the tool during the fine manipulation (force or tactile control). The dynamic equations of an industrial robot are a set of highly nonlinear differential equations. For an end-effector to move in a particular trajectory at a particular velocity a complex set of torque (force) functions are to be applied by the joint actuators. Instantaneous feedback information on position, velocity, acceleration, and other physical variables can greatly enhance the performance of the robot.

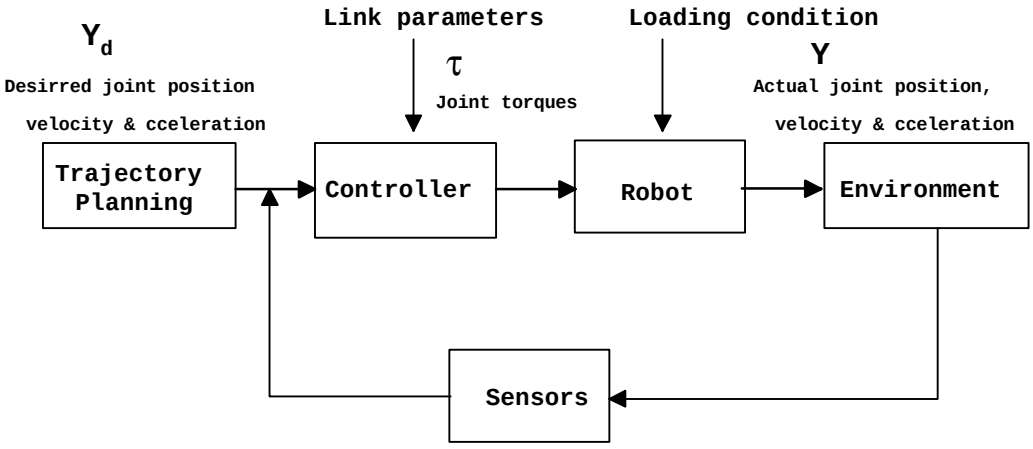


Figure 3 - Basic control block for robot trajectory system

In most systems, conventional single loop controllers track the tasks which are defined in terms of a joint space reference trajectory. In practice, the tracking error is compensated through an iterative process which adjusts the reference input so that the actual response (Y) of the manipulator is close to the desired trajectory (Y_d). When following a planned trajectory, control at time **t** will be more accurate if the controller can account for the end-effector's position at an earlier time. Figure 3 represents the basic block diagram of a robot trajectory system interacting with its environment.

With increasing demands for faster, more accurate and reliable robots, the field of robotics has faced the challenges of reducing the required on-line computational power, calibration time, and engineering cost when developing new robot controllers. If the robot is to be controlled in real time the algorithms used must be efficient and robust. Otherwise, we will have to compromise the robot control strategies, such as reducing the frequency content of the velocity profile at which the manipulator moves.

The robot arm position control is a complex kinematic and dynamic problem and has received researchers' attention for quite some time. During the last several years, most research on robot control has resulted in effective but computationally expensive algorithms. A number of approaches have been proposed to develop controllers that are robust and adaptive to the nonlinearities and structural uncertainties. However, they are also computationally very difficult and expensive algorithms to solve. As of this day, most robot controllers use joint controllers that are based on traditional linear controllers and are ineffective in dealing with the nonlinear terms such as friction and backlash.

One popular robot control scheme is called "computed-torque control" or "inverse-dynamics control". Most robot control schemes found in robust, adaptive or learning control strategies can be considered as special cases of the computed-torque control. The computed-torque-like control techniques involves the decomposition of the control design problem into two parts[Koivo, 1989]:

(1) a primary controller, a feedforward (inner-loop) design to track the desired trajectory under ideal conditions.
(2) a secondary controller, a feedback (outer-loop) design to compensate for undesirable deviations (disturbances) of the motion from the desired trajectory based on a linearized model.

The primary controller compensates for the nonlinear dynamic effects, and attempts to cancel the nonlinear terms in the dynamic model. Since the parameters in the dynamic model of the robot are not usually exact, undesired motion errors are expected. These errors can be corrected by the secondary controller. Figure 4 represents the decomposition of the robot controller showing the primary and secondary controllers.

It is well known that humans perform control functions much better than the machine-like robots. In order to control voluntary movements, the central nervous system must determine the desired trajectory in the visual coordinates, transform its coordinate to the body coordinate, and finally generate the motor commands [Miyamoto *et al.*, 1988]. The

human information processing device (brain) has been the motivation for many researchers in the design of intelligent computers often referred to as neural computers.

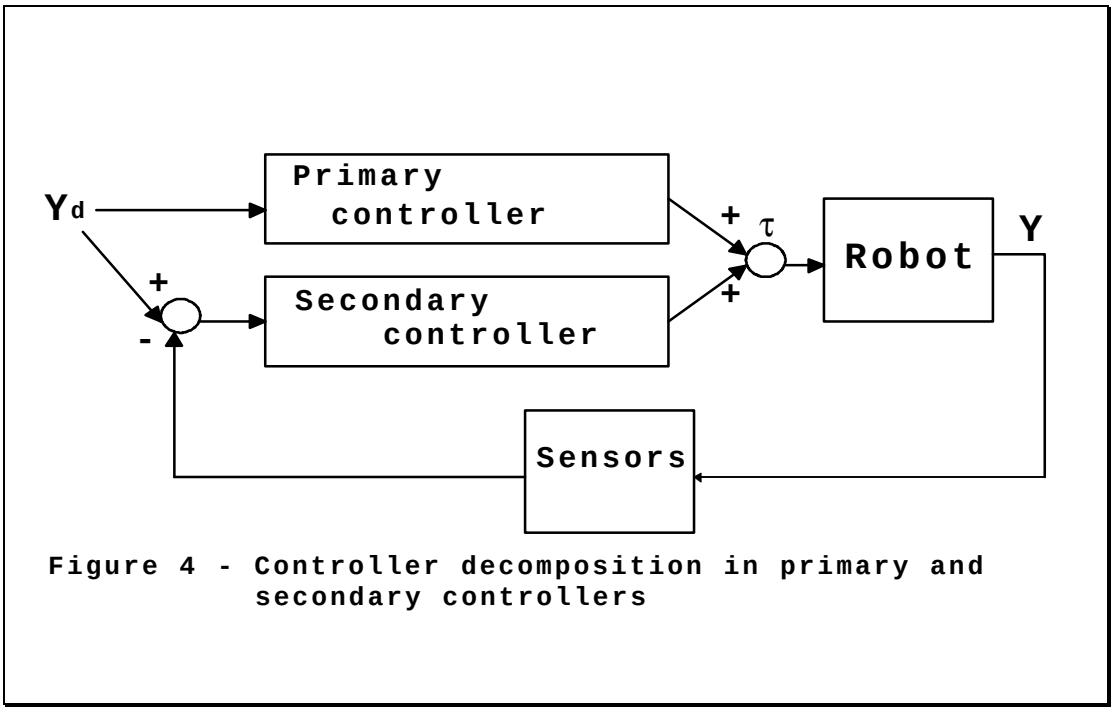


Figure 4 - Controller decomposition in primary and secondary controllers

Psaltis, *et al.* [1988] describe the neural computer as a large interconnected mass of simple processing elements (artificial neurons). The functionality of this mass, called the *artificial neural network*, is determined by modifying the strengths of the connections during the learning phase. This basic generalization of the morphological and computational feature of the human brain has been the abstract model used in the design of the neural computers.

Researchers interested in neural computers have been successful in computationally intensive areas such as pattern recognition and image interpretation problems. These problems are generally static mapping of input vectors into corresponding output classes using a feedforward neural network. The feedforward neural network is specialized for

the static mapping problems. Whereas in the robot control problem, nonlinear dynamic properties need to be dealt with and a different type of neural network structure must be used. Recurrent neural networks have the dynamic properties, such as feedback architecture, needed for the appropriate design of such robot controllers.

5. Artificial Neural Networks

ANNs are highly parallel, adaptive and fault tolerant dynamical systems modeled like their biological counterparts. The phrases "neural networks" or "neural nets" are also used interchangeably in the literature, which refer to neurophysiology, the study of how the brain and its nervous system work. ANNs are specified by the following definitions [Lippman, 1987]:

(1) Topology. It describes the networked architecture of a set of neurons. The set of neurons are organized into layers which are then classified as either *feedforward networks* or *recurrent networks.* In feedforward layers, each output in a layer is connected to each input in the next layer. In a recurrent ANN, each neuron can receive as its input a weighted output from other layers in the network, possibly including itself. Figure 5 illustrates three simple representations of the ANN topologies.

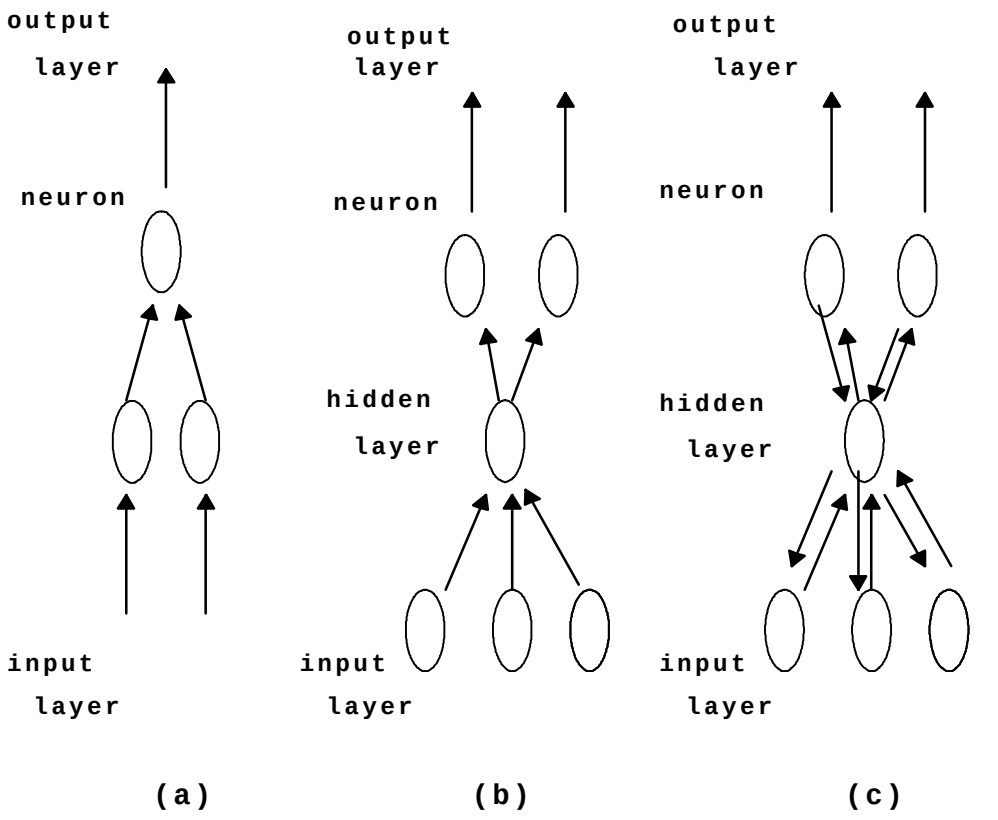


Figure 5 - ANN topologies: (a) single-layer feedforward (b) multilayer feedforward; (c) multilayer recurrent

(2) Neuron. A computational element that defines the characteristics of input/output relationships. A simple neuron is shown in Figure 6, which sums N weighted inputs (called activation) and passes the result through a nonlinear transfer function to determine the neuron output. Some nonlinear functions that are often used to mimic biological neurons are: unit step function and linear transfer-function. A very common formula for determining a neuron's output is through the use of sigmoidal (squashing) functions:

$$g(x) = (1 + e^{-kx}) \qquad \text{range of (0,1)} \qquad (1a)$$

$$g(x) = \tanh(kx) \qquad \text{range of (-1,1)} \qquad (1b)$$

For various values of the slope parameter, k, these functions are continuous and have derivatives at all points.

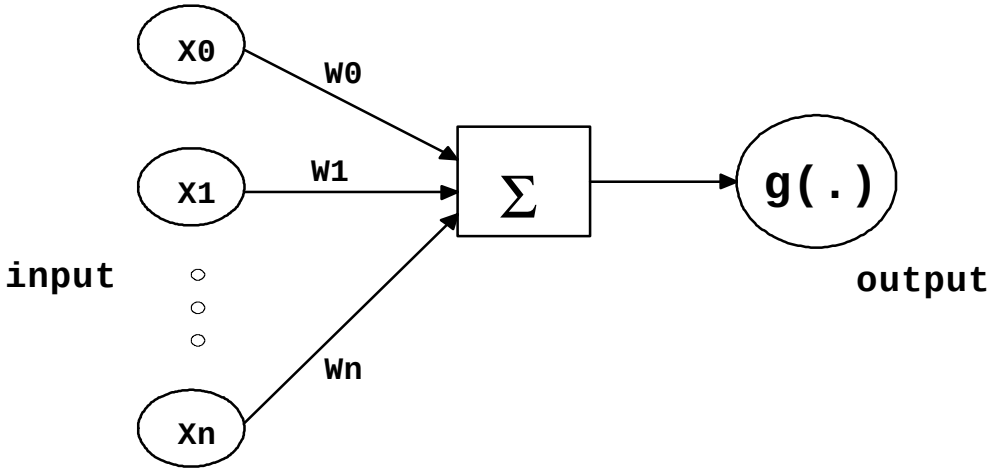


Figure 6 - McCulloch and Pitts neuron

(3) The Learning Rules. Given a set of input/output patterns, ANNs can learn to classify these patterns by optimizing the weights connecting the nodes (neuron) of the networks. The learning algorithms for weight adaptation can be described as either supervised or unsupervised learning. In supervised learning, the desired output of the neuron is known, perhaps by providing training samples. During supervised training, the network compares its actual response, which is the result of the transfer function described above, with the training example and then adjusts its weight in order to minimize the error between the desired and its actual output. In unsupervised training, which there are no teaching examples, built-in rules are used for self-modification, in order to adapt the synaptic weights in response to the inputs to extract features from the neuron. Kohonen's self-organizing map is an example of unsupervised learning [Chester, 1993].

One of the first models of an artificial neuron, was introduced in 1943 by McCulloch and Pitts and is shown in Figure 6. The model, known as the McCulloch-Pitts neuron,

computes a weighted sum of inputs (x_i) and outputs (unit step function) a binary value (Y) according to whether this sum is above or below a certain threshold (θ)[Hertz *et al.*, 1991].

$$Y = g(\sum w_i x_i - \eta) \qquad (2)$$

where g(p)= +1 if p>=0; 0 otherwise.

McCulloch and Pitts proved that a synchronous network of neurons (M-P network), described above, is capable of performing simple logical tasks (computations) that is expected of a digital computer. In 1958, Rosenblatt introduced the "perceptron", in which he showed how an M-P network with adjustable weights can be trained to classify sets of patterns. His work was based on Hebb's model of adaptive learning rule in the human brain [Hebb, 1949], in which he stated the neuron's interconnecting weights change continuously as it learns [Vemuri 1988].

In 1960, Bernard Widrow introduced its ADALINE (**ADA**ptive **LIN**ear element), a single-layer perceptron, and later extended it to what is known as MADALINE, multilayer ADALINE [Widrow and Lehr, 1990]. In MADALINE, Widrow introduced the steepest descend method to stimulate learning in the network. His variation of learning is referred to as the Widrow-Hoff rule or delta rule.

In 1969, Minsky and Papert reported on the theoretical limitations of the single layer M-P network, by showing the inability of the network to classify the exclusive-or (XOR) logical problem. They left the impression that neural network research is a farce and went on to establish the "artificial intelligence" laboratory at MIT. Hence, the research activity related to ANNs went to sleep until the early 1980s when the work by Hopfield, an established physicist, on neural networks rekindled the enthusiasm for this field. Hopfield's autoassociative neural network (a form of recurrent neural network) solved the classic hard optimization problem (traveling salesman) [Hopfield and Tank, 1985].

Other contributors to the field, Steven Grossberg and Teuvo Kohonon, continued their research during the seventies and early eighties (referred to as the "quiet years" in the literature). During the "quiet years", Steven Grossberg [1982 and 1988] worked on the mathematical development necessary to overcome one of the limitations reported by Minsky and Papert [1969]. Teuvo Kohonon [1982] developed the unsupervised training method, the self-organizing map. Later, Bart Kosko [1988] developed bi-directional associative memory (BAM) based on the works of Hopfield and Grossberg. Robert Hecht-Nielson [1990] pioneered the work on neurocomputing.

It wasn't until 1986 that the two-volume book, by McClleland and Rumelhart, titled *Parallel Distributed Processing* (PDP), exploded the field of artificial neural networks [Hecht-Nielson, 1990]. In this book (PDP), a new training algorithm called *The Backpropagation* method (BP), using the gradient search technique was used to train a multilayer perceptron to learn the XOR mapping problem described by Minsky and Papert [Rumelhart, Hinton, and Williams, 1986]. Since then, ANNs have been studied for both design procedures and training rules (supervised and unsupervised), and are current research topics. An excellent collection of theoretical and conceptual papers on neural networks can be found in books edited by Vemuri [1988], and Lau [1992]. Interested readers can also refer to a survey of neural networks book by Chapnick [1992] categorized by: theory, hardware and software, and how-to books.

The multilayer feedforward networks, using the BP method, represent a versatile nonlinear map of a set of input vectors to a set of desired output vectors on the spatial context (space). During the learning process, an input vector is presented to the network and propagates forward from input layers to output layers to determine the output signal. The output signal vector is then compared with the desired output vector resulting in an error signal. This error signal is backpropagated through the network in order to adjust the network's connecting strengths (weights). Learning stops when the error vector has reached an acceptable minimum [Lippman, 1987]. An example of feedforward network consisting of three layers is shown in Figure 7.

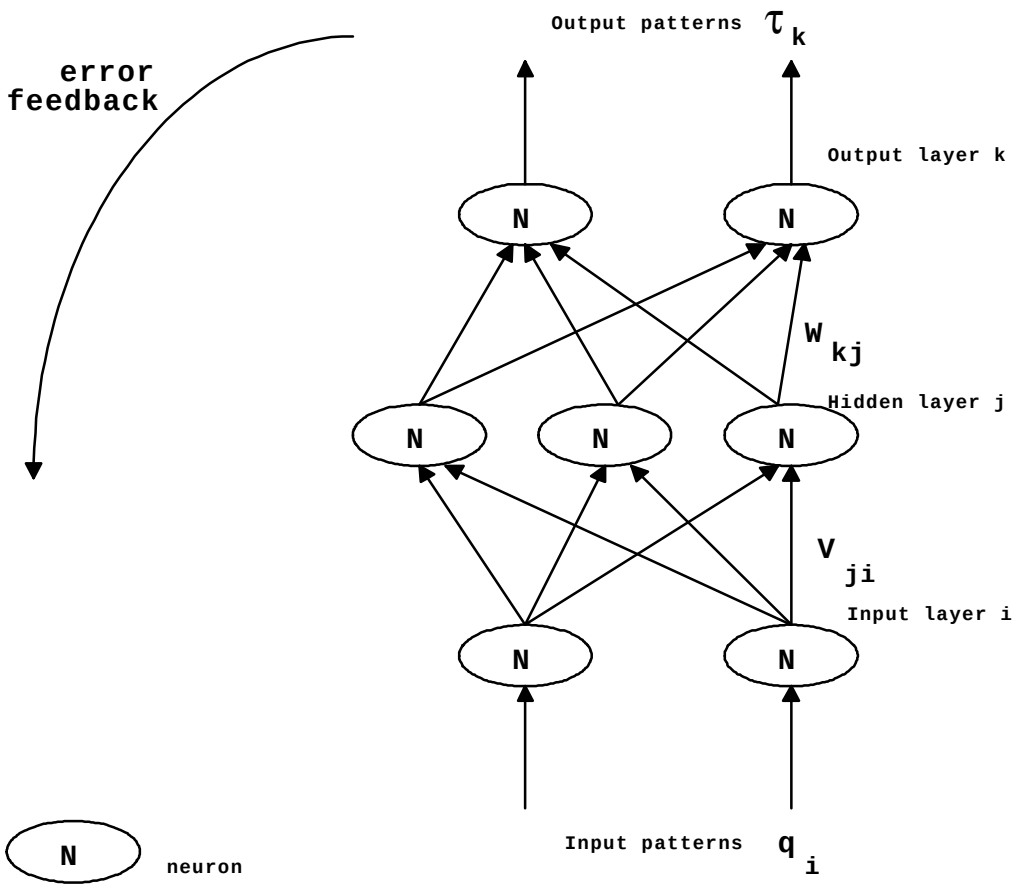


Figure 7 - Artificial Neural Network - Feedforward

Many studies have been undertaken in order to apply both the flexibility and the learning ability of backpropagation to robot control on an experimental scale [Josin, Charney and White, 1988; Kuperstein and Wang, 1990; Guez, Ahmad and Zelinsky, 1992]. In a recent study, an ANN utilizing an adaptive step size algorithm based on random search technique, improved the convergence speed of the BP method for solving the inverse kinematic problem for a two-link robot [Golnazarian, Hall, and Shell, 1992]. The robot control problem is a dynamic problem, where the BP method only provides a static mapping of the input vectors into output classes. Therefore, its benefits are limited. In addition, like any other numerical method, this novel learning method has limitations (slow convergence rate, local minimum). Attempts to improve the learning rate of BP have

resulted in many novel approaches [Jacobs, 1988; Baba, 1989]. It is necessary to note that the most important behavior of the feedforward networks using the BP method is its classification ability or the generalization to fresh data rather than temporal utilization of past experiences.

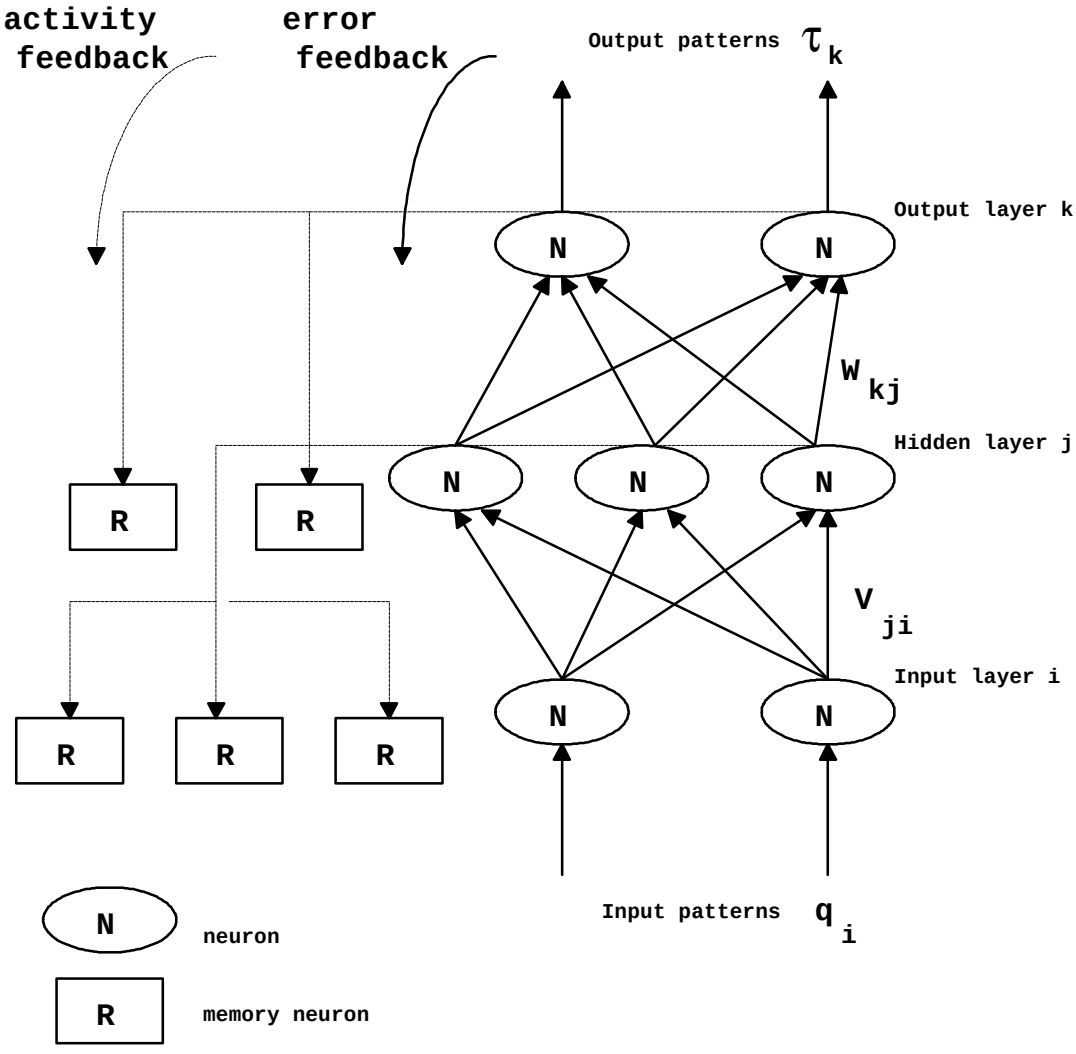


Figure 8 - Artificial Neural Network - Recurrent

A *recurrent network* is a multilayer network in which the activity of the neurons flows both from input layer to output layer (feedforward) and also from the output layer back to the input layer (feedback) in the course of learning [Hecht-Nielsen, 1990]. In a recurrent network each activity of the training set (input pattern) passes through the network more

than once before it generates an output pattern, where in standard BP only error flows backward, not the activity. This network architecture can base its response to problems on both spatial (space) and temporal (time) contexts [Pearlmutter, 1989; Pineda, 1987]. Therefore, it has a potential in modeling time-dependent processes such as in robotic applications. Figure 8 represents the recurrent ANN architecture.

It is clearly evident that a recurrent network will require a substantial memory in simulation (more connections) than a standard BP. Recurrent networks computing is a complex method, with a great deal of record keeping of errors and activities at each time phase. However, preliminary results indicate that they have the ability to learn extremely complex temporal patterns where data is unquantified with very little preprocessing i.e. stock market prediction, and Fourier transforms relationships [Caudill and Butler, 1992]. In feedforward networks where the training process is memoryless, each input is independent of the previous input. It is advantageous, especially in repetitive dynamical systems, to focus on the properties of the recurrent networks to design better robot controllers.

6. Robot Arm Kinematics

This section provides the mathematical formulations for the kinematic and dynamic analysis of robot manipulators. These formulations are then considered in design of the control algorithms for a four-axis SCARA industrial robot (AdeptOne). The treatment of robot manipulator kinematics and dynamics presented here is patterned primarily after discussions found in Lee [1982], Spong and Vidyasagar [1989], and Schilling [1990]. Additional comprehensive sources of robot manipulator kinematics and dynamics can be found in investigations by Paul [1981], Wolovich [1986], Craig [1986], Asada and Slotine [1986], and Fu, Gonzalez and Lee [1987].

The purpose of a robot manipulator is to position and interface its end-effector with the working object. For example, a robot has to pick up a part from a certain location, put it down in another location, and so on. Robot arm kinematics deals with the geometry of robot arm motion as a function of time (position, velocity, and acceleration) without

regards to the forces and moments that cause it. Specifically, one studies the functional relationship between the joint displacements and the position and orientation of the end-effector of a robot as shown in Figure 9. The problem of finding the end-effector position and orientation for a given set of joint displacements is referred to as the *direct kinematic problem*. Thus, for a given joint coordinate vector **q** and the global coordinate space **x**, it is to solve:

$$x = f(q) \tag{3}$$

where **f** is a nonlinear, continuous and differentiable function. This equation has a unique solution. On the other hand, given the end-effector position and orientation, the *inverse kinematic problem* calculates the corresponding joint-variables to drive the joint servo controllers by solving:

$$q = f^{-1}(x) \tag{4}$$

The solution to this equation, also called the arm solution, is not unique. Since trajectory tasks are usually stated in terms of the reference coordinate frame, the inverse kinematics problem is used more frequently.

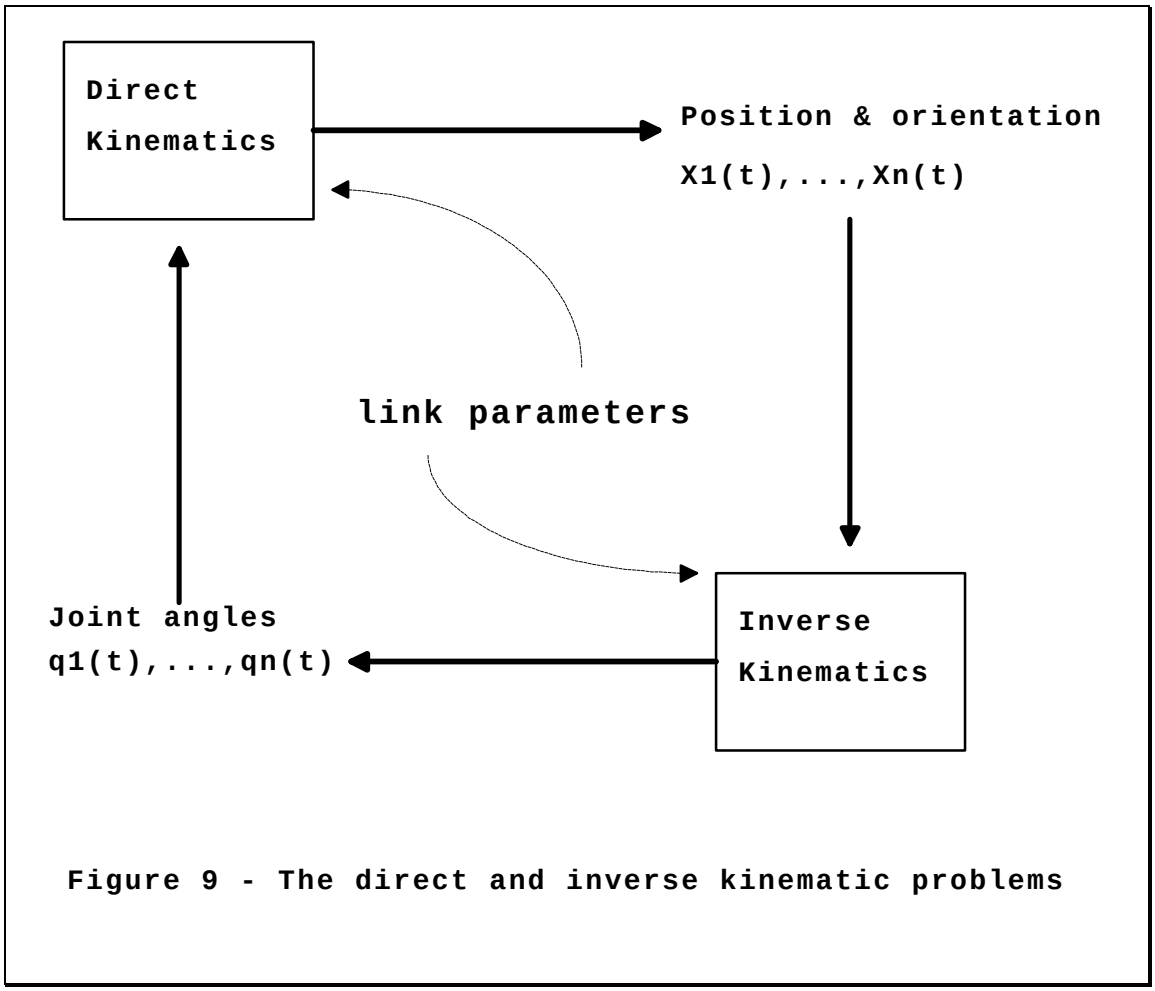


Figure 9 - The direct and inverse kinematic problems

Homogeneous Transformation Matrix. Before further analyzing the robot kinematic problem, a brief review of matrix transformations is needed. Figure 10 illustrates a single vector defined in the {i} coordinate frame $\mathbf{P}^{i}=(x',y',z')$. The task is to transform the vector defined with the {i} coordinate frame to a vector with the {i-1} coordinate frame, $\mathbf{P}^{i-1}=(x,y,z)$. Simply, this transformation is broken up into a rotational part and a translational part.

$$\mathbf{p}^{i-1} = \mathbf{R}_i \mathbf{p}^{i} + \mathbf{d}_i \qquad (5)$$

Since rotation is a linear transformation, the rotation between the two coordinate frames is given by

$$\mathbf{p}^{i-1} = \mathbf{R}_i \mathbf{p}^i \qquad (6)$$

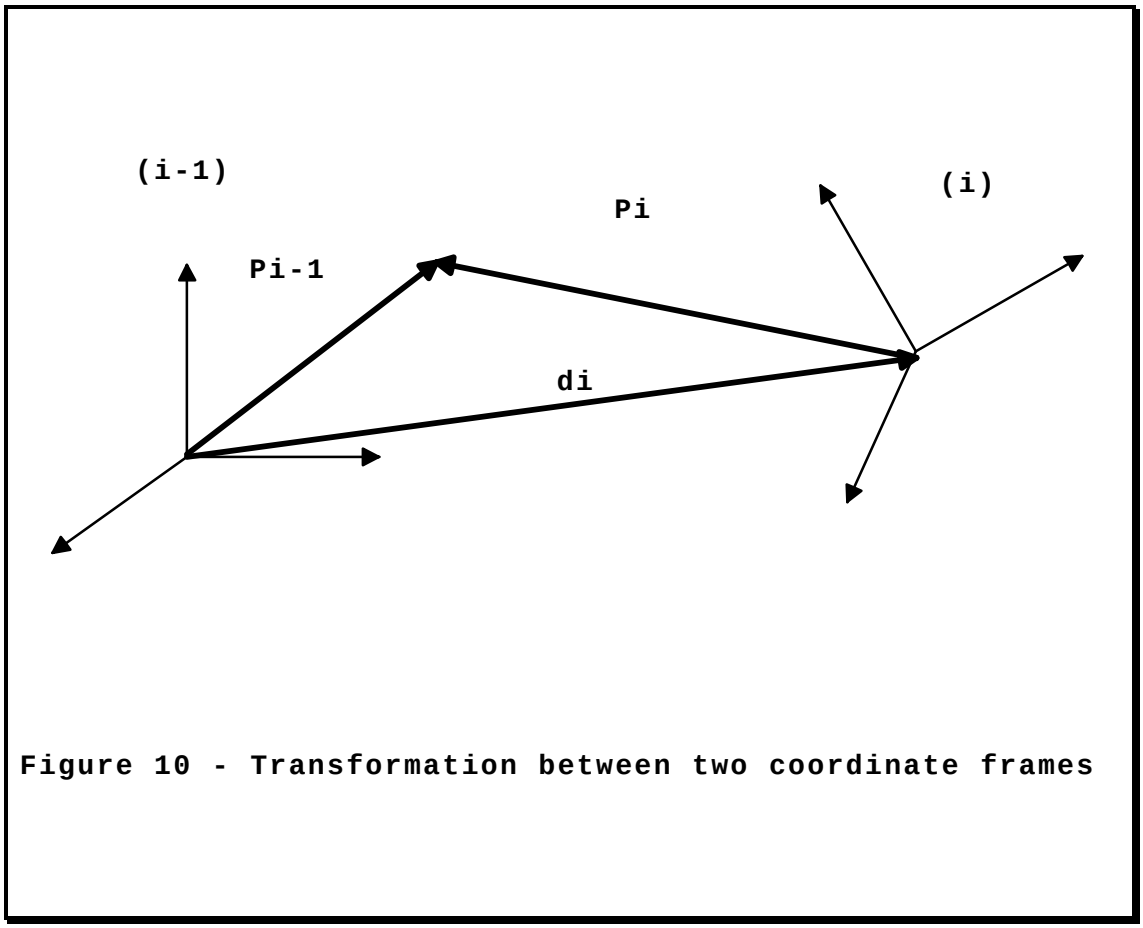


Figure 10 - Transformation between two coordinate frames

Here, $\mathbf{R}_i$ is 3x3 matrix operation about x, y, and z axis. These matrices are

$$R_x() = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\eta & -\sin\eta \\ 0 & \sin\eta & \cos\eta \end{bmatrix} \qquad (7a)$$

$$R_y(\ ) = \begin{bmatrix} \cos\eta & 0 & \sin\eta \\ 0 & 1 & 0 \\ -\sin\eta & 0 & \cos\eta \end{bmatrix} \qquad (7b)$$

$$R_z(\ ) = \begin{bmatrix} \cos\approx & -\sin\approx & 0 \\ \sin\approx & \cos\approx & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (7c)$$

The following general statements can be made:

1) A coordinate transformation **R** represents a rotation of a coordinate frame to a new position.
2) The columns of **R** give the direction cosines of the new frame axes expressed in the old frame.
3) It can be extended to a product of more than two transformation matrices.

To complete the transformation in Figure 10, translation between frames {i} and {i-1} still needs to take place.

$$\mathbf{d_i} = \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} \tag{8}$$

However, translation is a non-linear transformation, hence the matrix representation of equation (5) can only be in 3x4 form:

$$p^{i-1} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = R_i \begin{bmatrix} x^{\circlearrowleft} \\ y^{\circlearrowleft} \\ z^{\circlearrowleft} \end{bmatrix} + \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} \tag{9}$$

$$p^{i-1} = \begin{bmatrix} R_i & d_i \end{bmatrix} \begin{bmatrix} p^i \\ 1 \end{bmatrix}$$

Equation (9), where $[\mathbf{R_i}\ \mathbf{d_i}]$ is a (3x4) matrix and $[\mathbf{P_i}\ 1]^T$ is a (4x1) vector, can only be used to transform the components of **p** from frame {i} to {i-1}. Due to singularity of the matrix above, the inverse of the transformation cannot be achieved. To incorporate the inverse transformation in equation (9), the concept of *homogeneous coordinate representation*[1] replaces the (3x4) transformation matrix with a (4x4) transformation matrix by simply appending a final (1x4) row, defined as [0 0 0 1], to $[\mathbf{R_i}\ \mathbf{d_i}]$. Correspondingly, the $\mathbf{P_{i-1}}$ vector will be replaced by (4x1) vector of $\mathbf{P_i} = [\mathbf{P_{i-1}}\ 1]^T$.

[1] The representation of an n-component position vector by an (n+1)-component vector is called homogeneous coordinate representation.

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} & & & dx \\ & R_i & & dy \\ & & & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^{①} \\ y^{①} \\ z^{①} \\ 1 \end{bmatrix} \tag{10}$$

It can be seen that the transformation equation (5) is equivalent to the matrix equation (10). The (4x4) transformation matrix, noted $\mathbf{H_i}$, contains all the information about the final frame, expressed in terms of the original frame:

$$H = \begin{bmatrix} R & d_i \\ 0 & 1 \end{bmatrix} \tag{11}$$

Using the fact that $\mathbf{R_i}$ is orthogonal it is easy to show that the inverse transformation $\mathbf{H^{-1}}$ is given by

$$H^{-1} = \begin{bmatrix} R_i^T & -R_i^T d_i \\ 0 & 1 \end{bmatrix} \tag{12}$$

The matrix $\mathbf{H_i}$ has homogenized the representation of translation and rotations of a coordinate frame. Therefore, matrix $\mathbf{H_i}$ is called the *homogeneous transformation matrix*. The upper left (3x3) submatrix represents the rotation matrix; the upper right (3x1) submatrix represents the position vector of the origin of the rotated coordinate system with respect to the reference system; the lower left (1x3) submatrix represents perspective transformation for visual sensing with a camera; and the fourth diagonal element is the global scaling factor. In the robot manipulator, the perspective transformation is always a zero vector and the scale factor is one. The frame transformation is now given by:

$$\mathbf{P^{i-1} = H_i P^i} \tag{13}$$

This transformation, represented by the matrix $\boldsymbol{H_i}$, is obtained from simpler transformations representing the three basic translations along (three entries of $\mathbf{d_i}$), and

three rotations (three independent entries of $\mathbf{R}_i$) about the frames axes of x, y, and z. They form the six degrees of freedom associated with the configuration of **P**. These fundamental transforms, expressed in a 4x4 matrix notation, are shown as:

$$\text{Trans}(dx,dy,dz) = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14a}$$

$$\text{Rot}(x,\theta) = \begin{Vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\eta & -\sin\eta & 0 \\ 0 & \sin\eta & \cos\eta & 0 \\ 0 & 0 & 0 & 1 \end{Vmatrix} \tag{14b}$$

$$\text{Rot}(y,\theta) = \begin{Vmatrix} \cos\eta & 0 & \sin\eta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\eta & 0 & \cos\eta & 0 \\ 0 & 0 & 0 & 1 \end{Vmatrix} \tag{14c}$$

$$\text{Rot}(z,\theta) = \begin{Vmatrix} \cos\eta & -\sin\eta & 0 & 0 \\ \sin\eta & \cos\eta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{Vmatrix} \tag{14d}$$

The homogeneous transformation matrix is used frequently in manipulator arm kinematics.

The Denavit-Hartenberg Representation. The most commonly accepted method for specifying frame position and finding the desired transformation matrices is attributed to the Denavit and Hartenberg (D-H) representation [Denavit and Hartenberg, 1955]. In this method, an orthonormal Cartesian coordinate system is established on the basis of three rules [Wolovich, 1986]:

a) The z_{i-1} axes lies along the axis of motion of the ith joint.

b) The x_i axis is normal to the z_{i-1} axis, pointing away from it.

c) The y_i axis complete the right-hand rule coordinate system.

This is illustrated in Figure 11. Note that joint i joins link i-1 with link i. Frame i, which is the body frame of link i, has its z axis located at joint i+1. If the joint is revolute, then the rotation is about the z axis. If the joint is prismatic, the joint translation is along the z axis.

The D-H representation depends on four geometric parameters associated with each link to completely describe the position of successive link coordinates:

a_i = the shortest distance between z_i and z_{i-1} along the x_i

α_i = the twist angle between z_i and z_{i-1} about the x_i

d_i = the shortest distance between x_i and x_{i-1} along the z_{i-1}

θ_i = the angle between x_{i-1} and x_i about the z_{i-1}

For a revolute joint, $\boldsymbol{\theta_i}$ is the variable representing the joint displacement where the adjacent links rotate with respect to each other along the joint axis. In prismatic joints in which the adjacent links translate linearly to each other along the joint axis, $\mathbf{d_i}$ is the joint displacement variable, while $\boldsymbol{\theta_i}$ is constant. In both cases, the parameters a_i and α_i are constant, determined by the geometry of the link.

In general we denote the joint displacement by $\mathbf{q_i}$, which is defined as:

$\mathbf{q_i} = \boldsymbol{\theta_I}$ for a revolute joint

$\mathbf{q_i} = \mathbf{d_i}$ for a prismatic joint

Then, a (4x4) homogeneous transformation matrix can easily relate the ith coordinate frame to the (i-1)th coordinate frame by performing the following successive transformations:

1) Rotate about z_{i-1} axis an angle of θ_i, Rot(z_{i-1},θ_i)
2) Translate along the z_{i-1} axis a distance of d_i, Trans(0,0,d_i)
3) Translate along the x_i axis a distance of a_i, Trans(a_i,0,0)

4) Rotate about the x_i axis an angle of α_i, Rot(x_i,α_i)

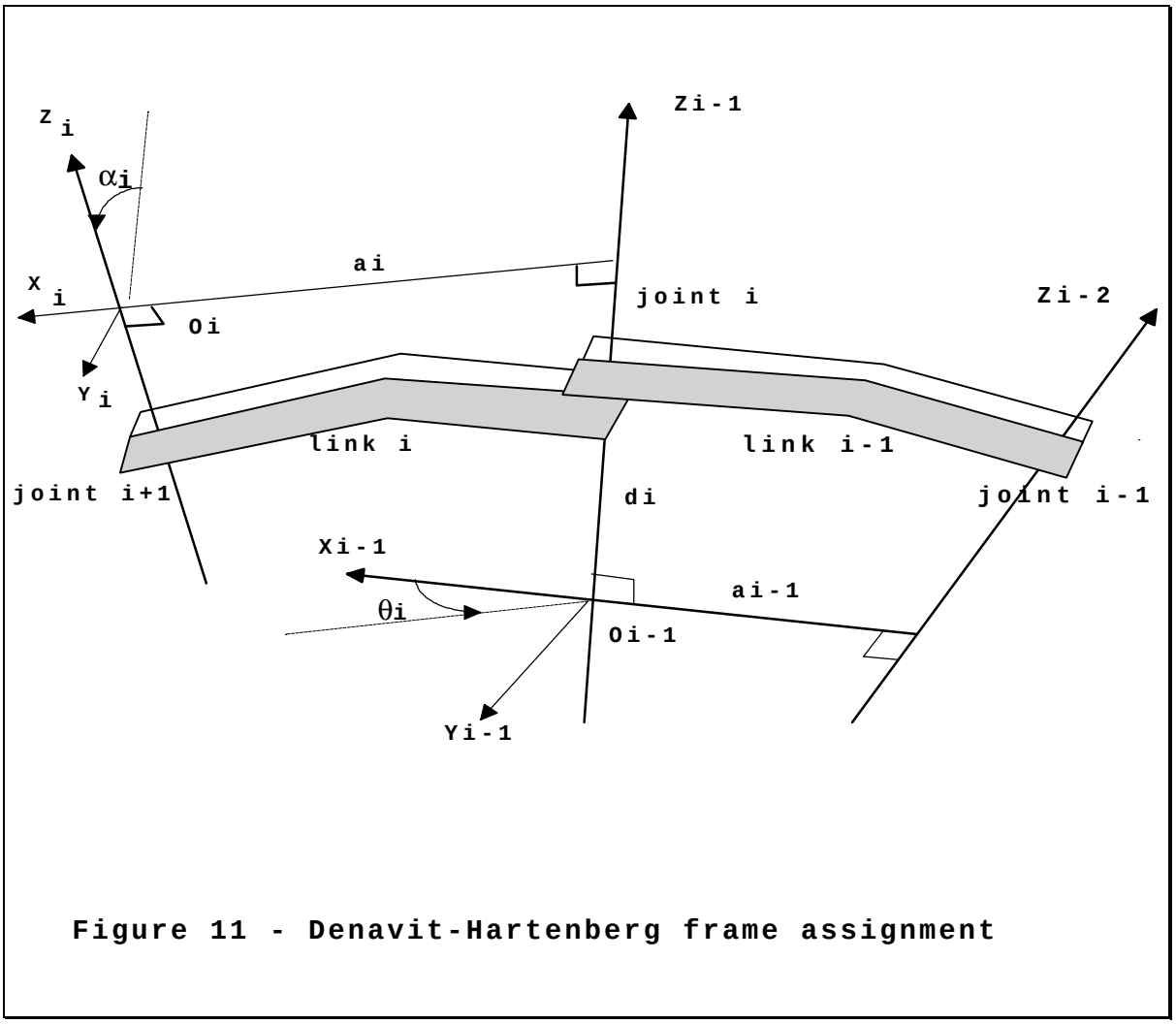


Figure 11 - Denavit-Hartenberg frame assignment

The operations above result in four basic homogeneous matrices. The product of these matrices yields a composite homogeneous transformation matrix $^{i}\mathbf{A}_{i-1}$. The $^{i}\mathbf{A}_{i-1}$ matrix is known as the D-H transformation matrix for adjacent coordinate frames, {i} and {i-1}. Thus,

$$^{i}A_{i-1} = \text{Trans}(0,0,d_i)\ \text{Rot}(z_{i-1},\theta_i)$$

$$\text{Trans}(a_i,0,0)\ \text{Rot}(x_i,\alpha_i)$$

$$^{i}A_{i-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \% \begin{bmatrix} \cos\eta_i & -\sin\eta_i & 0 & 0 \\ \sin\eta_i & \cos\eta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \%$$

$$\begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -\cos\alpha & \sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$ {}^{i}A_{i-1} = \begin{bmatrix} \cos\theta & -\cos\alpha\sin\theta & \sin\alpha\sin\theta & a_i\cos\theta \\ \sin\theta & \cos\alpha\cos\theta & -\sin\alpha\cos\theta & a_i\sin\theta \\ 0 & \sin\alpha & \cos\alpha & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (15)$$

Using the $^{i}\mathbf{A}_{i-1}$ matrix in figure 10, vector $\mathbf{P}_i$ expressed in homogeneous coordinates with respect to coordinate system {i}, relates to vector $\mathbf{P}_{i-1}$ in coordinate {i-1} by

$$\mathbf{P}_{i-1} = {}^{i}\mathbf{A}_{i-1}\,\mathbf{P}_i \qquad (16)$$

where $\mathbf{P}_{i-1} = (x,y,z,1)^T$ and $\mathbf{P}_i = (x',y',z',1)^T$.

Manipulator Arm Kinematic Equations. The transformation matrix of equation (16) relates points defined in frame {i} to frame {i+1}. For a robot manipulator with 6 links, the position of the end-effector (last link) with respect to the base is determined by successively multiplying together the single (D-H) transformation matrix that relates frame {6} to frame {0}:

$$\begin{aligned} {}^{6}T_5 &= {}^{6}A_5 \\ {}^{6}T_4 &= {}^{5}A_4\,{}^{6}T_5 = {}^{5}A_4\,{}^{6}A_5 \\ {}^{6}T_3 &= {}^{4}A_3\,{}^{6}T_4 = {}^{4}A_3\,{}^{5}A_4\,{}^{6}A_5 \\ {}^{6}T_2 &= {}^{3}A_2\,{}^{6}T_3 = {}^{3}A_2\,{}^{4}A_3\,{}^{5}A_4\,{}^{6}A_5 \\ {}^{6}T_1 &= {}^{2}A_1\,{}^{6}T_2 = {}^{2}A_1\,{}^{3}A_2\,{}^{4}A_3\,{}^{5}A_4\,{}^{6}A_5 \\ {}^{6}T_0 &= {}^{1}A_0\,{}^{6}T_1 = {}^{1}A_0\,{}^{2}A_1\,{}^{3}A_2\,{}^{4}A_3\,{}^{5}A_4\,{}^{6}A_5 \end{aligned} \qquad (17)$$

Generalized for n degree-of-freedom, the base frame {0} is assumed to be fixed. This is taken as the inertial frame with respect to which a task is specified. The body frame {n} is the free moving end-effector. The columns of the overall homogeneous transformation matrix, $^{n}\mathbf{T}_0$, corresponds to the position and orientation of the end-effector $\mathbf{(x_n)}$, expressed

in the base frame. This transformation matrix, $^{n}\mathbf{T_0}$, will be a function of all n joint variables $\mathbf{(q_n)}$, with the remaining parameters constant.

$$^{n}T_0(x_n) = {}^{1}A_0(q_1)^{2}A_1(q_2)^{3}A_2(q_3)...{}^{n}A_{n-1}(q_n) \qquad (18)$$

The final transformation matrix $^{n}\mathbf{T_0}$, also called the arm matrix, defines the final configuration of any end-effector with respect to the inertia frame {0}, depicted in Figure 12. The tool origin represents any appropriate point associated with the tool frame (or the transporting object). The origin of (O_t) frame can be taken either at the wrist or at the tool tip, or placed symmetrically between the fingers of the end-effector (gripper). The $^{n}\mathbf{T_0}$ matrix may be written as

$$^{n}T_0 = \begin{bmatrix} R_i & d_i \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} x_n & y_n & z_n & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (19)$$

$$= \begin{bmatrix} n & s & a & d \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_y & a_y & d_y \\ n_z & s_z & a_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where three mutually perpendicular unit vectors, as shown in Figure 12, represent the tool frame in Cartesian coordinate system. In the above equation:

n = normal unit vector, normal to the fingers of the robot arm, following the right-hand rule.

s = unit sliding vector, pointing to the direction of the sideward motion of the fingers (open and close).

a = unit approach vector, normal to the tool mounting plate of the arm.

Equation (20) represents the direct (forward) kinematic problem for a robot manipulator: given the joint displacements **(Θ)** and the link parameters, find the position and the orientation of the end-effector **(X)** in the base frame:

$$\mathbf{X} = \mathbf{f}(\Theta) \tag{20}$$

where **f** is a nonlinear, continuous and differentiable function. This equation has a unique solution. This is best illustrated by an example.

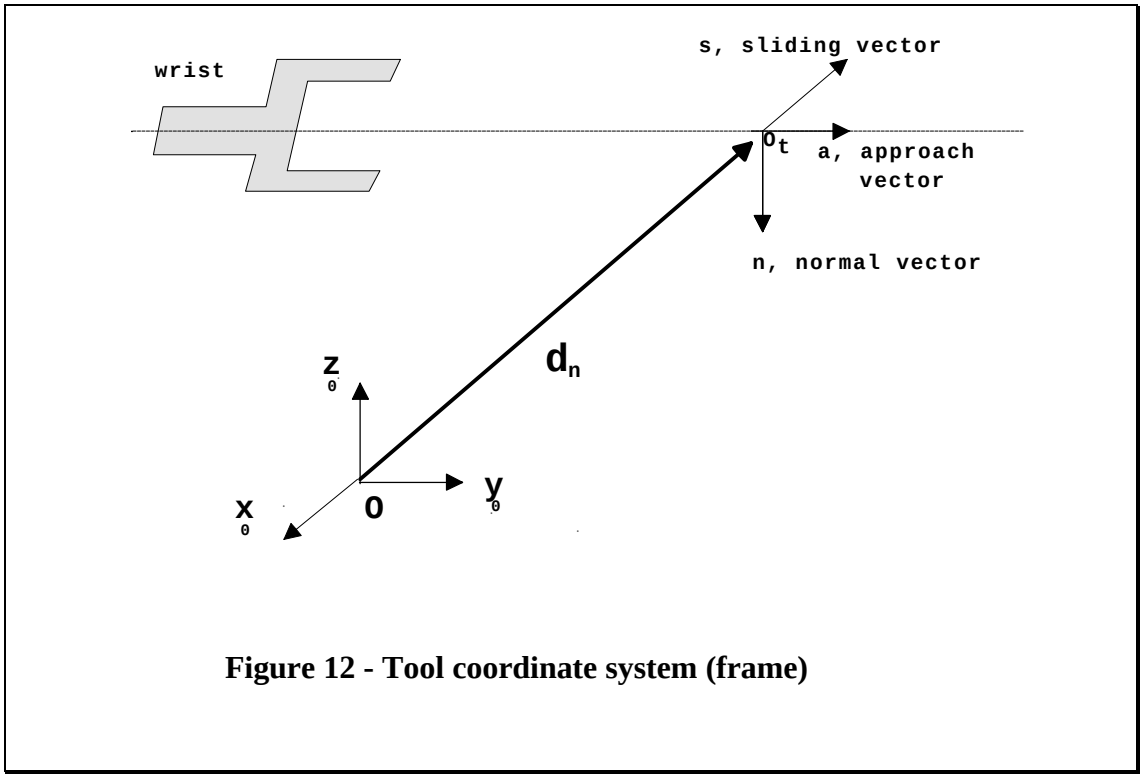


Figure 12 - Tool coordinate system (frame)

Consider the Four-axis horizontal-jointed SCARA robot, AdeptOne in Figure 13. This manipulator is unique because it is the first commercial robot to implement a *direct-drive*[2] system for actuation. The robot consists of an RRP arm and a one degree-of-freedom wrist, whose motion is a roll about the fourth vertical axis. The (D-H) link kinematic parameters are given in Table 3 [Schilling, 1990].

[2] Direct-drive is an electrical drive in which no gear reducer is used. Therefore the rotor of the electric motor is directly coupled to the load, hence the mechanical gears are not needed. This eliminates gear friction and backlash and allows for clean, precise, high-speed operation [Asada and Youcef-Toumi, 1987].

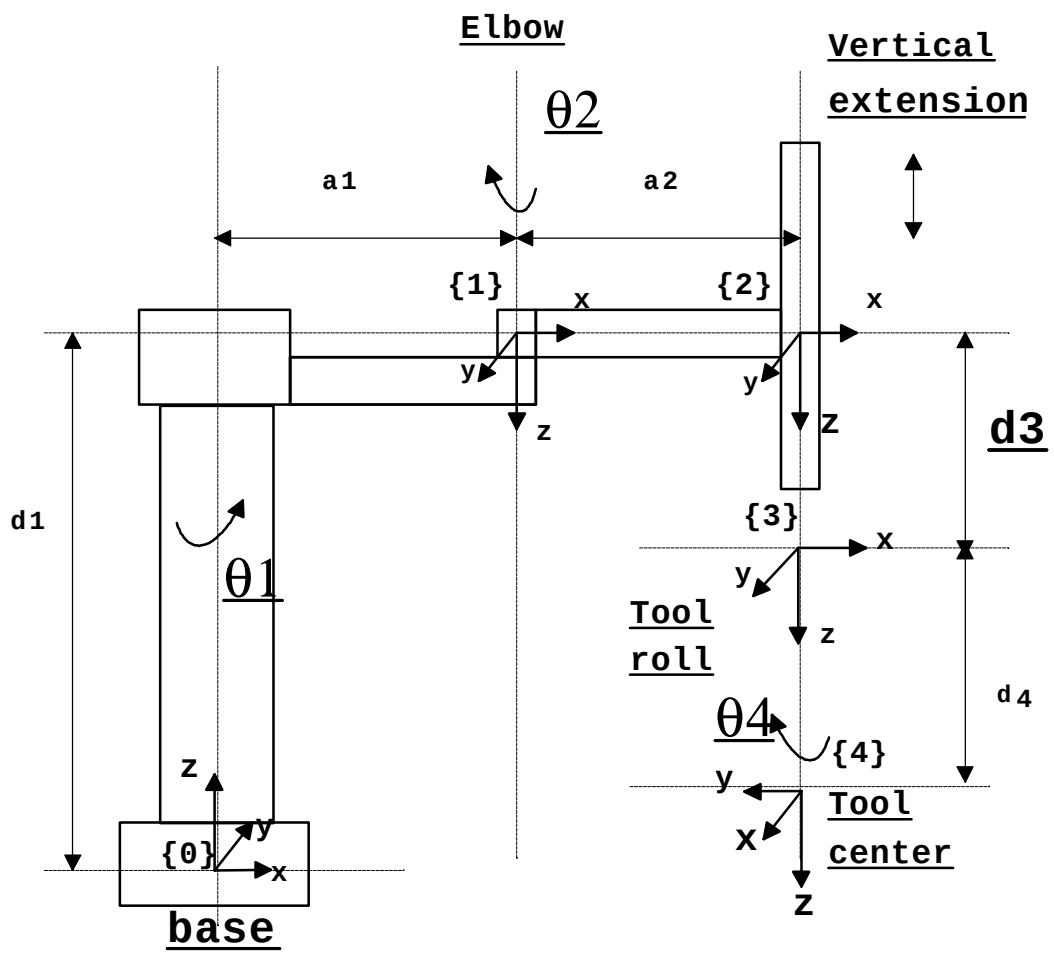


Figure 13 - A four-axis SCARA robot (AdeptOne)

Table 3. Kinematic Parameters for a four-axis SCARA robot

Axis	θ	d	a	α
1	q_1	d_1 = 877 mm	a_1 = 425 mm	π
2	q_2	0	a_2 = 375 mm	0
3	0	q_3	0	0
4	q_4	d_4 = 200 mm	0	0

All the joint axes are parallel. The joint variable (vector-form) is $\mathbf{q} = [\theta_{1,}\ \theta 2,\ d_{3,}\ \theta 4]^{T}$**.** The first two joint variables, $\boldsymbol{\theta_1}$ and $\boldsymbol{\theta_2}$, are revolute variables which establish the horizontal component of the tool position. The third joint variable $\mathbf{d_3}$, a prismatic joint, determines the vertical component of tool origin. Finally, the last joint variable $\boldsymbol{\theta_4}$, which is of revolute kind, controls the tool orientation.

Using the values from Table 3 in equation (15), the ${}^{\mathbf{i}}\mathbf{A}_{\mathbf{i-1}}$ matrices are as follows

$$ {}^{1}A_{0} = \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & 0 & a_1\cos\theta_1 \\ \sin\theta_1 & -\cos\theta_1 & 0 & a_1\sin\theta_1 \\ 0 & 0 & -1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (21a) $$

$$ {}^{2}A_{1} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (21b) $$

$$ {}^{3}A_{2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \theta_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (21c) $$

$$ {}^{4}A_{3} = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & 0 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0_1 \\ 0 & 0 & 1 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (21d) $$

The forward kinematic solution, using equation (18) is therefore given by

$$ {}^{tool}T_{base(x4)} = {}^{4}T0(x4) $$

$$ = {}^{1}A0(\theta 1)\,{}^{2}A1(\theta 2)\,{}^{3}A2(d3)\,{}^{4}A3(\theta 4) \qquad (22) $$

$$= \begin{bmatrix} \cos(\theta_1 - \theta_2 - \theta_4) & \sin(\theta_1 - \theta_2 - \theta_4) & 0 & a_1 \cos\theta_1 + a_2 \cos(\theta_1 - \theta_2) \\ \sin(\theta_1 - \theta_2 - \theta_4) & -\cos(\theta_1 - \theta_2 - \theta_4) & 0 & a_1 \sin\theta_1 + a_2 \sin(\theta_1 - \theta_2) \\ 0 & 0 & -1 & d_1 - d_3 - d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In equation (22) the rotation matrix $^{\mathbf{tool}}\mathbf{R}_{\mathbf{base}}$, the (3x3) upper left sub-matrix, expresses the orientation of tool frame {4} relative to the base frame {0} as

$$^{tool}R_{base} = \begin{bmatrix} n & s & a \end{bmatrix} = \begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix}$$

$$^{tool}R_{base} = \begin{bmatrix} \cos\theta_1 - \theta_2 - \theta_4) & \sin(\theta_1 - \theta_2 - \theta_4) & 0 \\ \sin(\theta_1 - \theta_2 - \theta_4) & -\cos(\theta_1 - \theta_2 - \theta_4) & 0 \\ 0 & 0 & -1 \end{bmatrix} \qquad (23)$$

Note that the approach vector a = [0 0 -1] is fixed, and independent of the joint variables. This is one of the characteristics of the AdeptOne robot, or even all SCARA robots, which are designed to manipulate objects directly from above. Industrial applications such as circuit board assembly, is the common area of use for this robot.

The vector $\mathbf{d}_i$, the right column vector in equation (22), represents position of the tool frame {4} relative to the base frame {0} as

$$d_i = \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} = \begin{bmatrix} a_1 \cos\theta_1 + a_2 \cos(\theta_1 - \theta_2) \\ a_1 \sin\theta_1 + a_2 \sin(\theta_1 - \theta_2) \\ d_1 - d_3 - d_4 \end{bmatrix} \qquad (24)$$

The inverse kinematic solution. For the purpose of driving and controlling a robot, it is necessary to solve equation (20) for the joint variables since the actuator variables are the joint variables. This is the inverse (backward) kinematic problem associated with a robot manipulator: given the position and the orientation of the end-effector **(X)** in the base frame, find the joint displacement **(Θ):**

$$(\Theta) = \mathbf{f}^{-1}(\mathbf{X}) \qquad (25)$$

The backward solution algorithm is generally more difficult than the forward solution. In the three-dimensional space, six coordinates are needed to specify a rigid body (three position coordinates and three angles of rotation). Since the six equations generated are nonlinear trigonometric functions and are coupled, a simple and unique solution for **q** may not even exist. For a high number of degrees of freedom the inverse kinematic problem can result in very complicated solution algorithms [Fu *et al.*, 1987).

Some simplification is possible by properly designing the robot geometry. For example, when the axes for the three revolute joints of a six degree-of-freedom robot coincide at the wrist of the end-effector, it is possible to decouple the six equations in (15) into two sets of three simpler equations [Pieper, 1968]. The first set of equations decide the position of the first three joint variables. Once the first three joint variables are determined, the last three joint variables are obtained such that the end-effector has the correct orientation.

Recall the four-axis SCARA (AdeptOne) example whose forward kinematic solution is defined by equation (22). Suppose that the position and orientation of the final frame (tool frame) is given as equation 19

$$^{tool}T_{base}(x_4) = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_y & a_y & d_y \\ n_z & s_z & a_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

To find the corresponding joint variables $[\theta_{1,}\ \theta 2,\ d_{3,}\ \theta 4]$, we must solve the following simultaneous set of nonlinear trigonometric equations:

$$n_x = \cos(\eta - \eta_2 - \eta_4)$$
$$n_y = \sin(\text{illegible})$$
$$n_z = 0$$
$$s_x = \sin(\text{illegible})$$

$$
\begin{aligned}
s_y &= -\cos(\theta_1 - \theta_2 - \theta_4) \\
s_z &= 0 \\
a_x &= 0 \\
a_y &= 0 \\
a_z &= -1 \\
d_x &= a_1 \cos(\theta_1) + a_2 \cos(\theta_1 - \theta_2) \\
d_y &= a_1 \sin(\theta_1) + a_2 \sin(\theta_1 - \theta_2) \\
d_z &= d_1 - d_3 - d_4
\end{aligned}
$$

Note that, the SCARA robot has only four degrees of freedom. Therefore, not every element of the orientation matrix allows a solution for equation (22). The complete solution to the inverse kinematic problem of finding the joint variables [d] in terms of the end-effector position and orientation for the SCARA manipulator is shown in table 4.

Table 4. Inverse kinematics of a four-axis SCARA robot

Base joint:

$$q_2 = \arctan 2\left[\pm\sqrt{1-r^2} \quad r \right] \text{ , where } r^2 = \frac{d_x^2+d_y^2-a_1^2-a_2^2}{2a_1a_2}$$

Elbow joint:

$$q_1 = \arctan 2\left[d_x \quad d_y \right] - \arctan 2\left[a_1 + a_2 \cos\theta_2 \quad a_2 \sin\theta_2 \right]$$

Vertical extension joint:

$$q_3 = d_1 - d_4 - d_z$$

Tool roll joint:

$$q_4 = q_1 - q_2 - \arctan 2\left[n_y \quad n_x \right]$$

This inverse kinematics solution is not unique due to multiplicity of q_2. Complete derivation for the solution for the four-axis SCARA (AdeptOne) robot manipulator can be found in textbooks by Spong and Vidyasagar [1989] and Schilling [1990]. In general, the inverse kinematic problem can be solved by various methods. The methods used usually are algebraic, geometric or iterative. The common approach is to use a closed form

solution to the inverse kinematic problem. However, these solutions are manipulator-dependent and still often too difficult to solve in closed form. In many cases (non closed form) the iterative method is required with kinematically redundant robots [Fu *et al.*, 1987).

One popular method is to use the N-dimensional Newton-Raphson algorithm or its modified version to solve the non-linear equations [Oh *et al.*, 1984]. Fast iterative techniques for the inverse kinematic problems have been reported based on nonlinear least squares minimization with accurate and rapid convergence [Goldenberg and Lawrence, 1985]. Other techniques based on the geometric relationship between the various links and joints have been reported, depending on the points of reference chosen. In general, most researchers resort to numerical methods to obtain the inverse solution.

The use of iterative techniques raises the problem of accurate real-time implementation of manipulator kinematic control. The need to compute the inverse Jacobian at several points, importance of closeness of the initial solution to the exact solution (otherwise the algorithm diverges) and accumulation of the linearization error, reflects the computational complexity of the methods for on-line use.

Artificial neural network (ANN) theory has provided an alternative solution for solving the inverse kinematic problem. ANNs are highly parallel, adaptive and fault tolerant dynamical systems modeled like their biological counterparts [Rumelhart *et al.*, 1986]. Application of ANNs to this problem is to train the network with the input data in the form of pairs of end-effector positions and orientations and the corresponding joint values. After the training is completed, the ANN can generalize and give good results (joint angles) at new data points (position and orientation).

Recently, artificial neural networks (ANNs) have been augmented with an iterative procedure using the Newton-Raphson technique resulting in an increase in computational efficiency by two-fold for the PUMA 560 robot [Guez *et al.*, 1992]. An ANN based scheme has also been proposed for computing manipulator inverse kinematics where no

prior knowledge of the manipulator kinematics is required [Nguyen and Patel, 1990]. In the event that the physical structure of a robot is changed (or damaged) during an operation, ANN architecture can supplement the existing robot controller to learn the new transformations quickly without repairing the robot physically [Josin *et al.*, 1988]. In a recent study, an ANN utilizing an adaptive step size algorithm based on random search technique, improved the convergence speed for solving the inverse kinematic problem for a two-link robot [Golnazarian *et al.*, 1992].

Manipulator motion kinematic equations. Previously, we have described the forward and inverse arm solutions relating joint positions and end-effector position and orientation. However, the manipulator is stationary at a specific configuration. Typically, a robot is in motion to perform a task such as spray painting, arc welding and sealing. Continuous-path motion control is required in these applications where the tool must travel a specific path in a prescribed time (trajectory). One must then determine and control the velocity and acceleration of the end-effector between points on the path.

The forward kinematic (20) relates the end-effector position to the joint displacements. When the end-effector is in motion, an infinitesimal directional change in its position is determined by differentiating the kinematic equations (20) with respect to time, this yields:

$$dx_m = \frac{\partial x_m}{\partial q_1} dq_1 + \frac{\partial x_m}{\partial q_2} dq_2 + \frac{\partial x_m}{\partial q_3} dq_3 + \ldots + \frac{\partial x_m}{\partial q_n} dq_n$$

$$dx_m = J(q)\, dq_n$$

$$\dot{X}_m = J(q)\, \dot{q}_n \qquad (27)$$

The J(q) matrix, called *manipulator Jacobian or Jacobian*, defines the linear transformation from joint coordinates to Cartesian coordinates, n is the number of joints of the manipulator and m is the dimensionality of the Cartesian coordinate of the tool under consideration. *The Jacobian*, is one of the most important quantities in the analysis and control of robot motion. It is used for smooth trajectory planning and execution, in the derivation of the dynamic equations, and in transformation of forces applied by the end-

effector into the resultant torque generated at each joint. The generalized Cartesian velocity vector, $\dot{X}$ is defined as

$$\dot{X} = \begin{bmatrix} \text{❍} \\ \text{⌘} \end{bmatrix}$$

where x, y z] represents the linear velocity and the x y z the angular velocity of the tool.

Consider the four-axis SCARA robot whose kinematic description has been developed. The Jacobian of the SCARA robot is

$$J(q) = \left[\begin{array}{cccc} -a_1 \sin\eta_1 - a_2 \sin(\eta_1 - \eta_2) & a_2 \sin(\eta_1 - \eta_2) & 0 & 0 \\ a_1 \cos\eta_1 + a_2 \cos(\eta_1 - \eta_2) & -a_2 \cos(\eta_1 - \eta_2) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & -1 \end{array}\right] \quad (28)$$

The first three rows of J(q) correspond to linear tool displacement, while the last three correspond to angular tool displacement. Then a joint space trajectory q(t) corresponding to x(t) can be obtained by inverting the Jacobian along with the inverse kinematic solution.

$$\dot{\approx} = J(\approx)^{-1}\dot{X} \quad (29)$$

By differentiating the equation (29) the desired joint accelerations are found as well,

$$\ddot{\approx} = \dot{J}(\approx)^{-1}\dot{X} + J(\approx)^{-1}\ddot{X} \quad (30)$$

The problem with solving for the joint space differentials (velocity and acceleration) using the Jacobian is that at certain point in joint space, the tool Jacobian may lose its rank. That is, there is a reduction in the number of linearly independent rows and columns. The numerical solutions of equations (29) and (30) produce very large differential values. The

points at which J(q) loses rank are called *joint-space singularities*. There are two types of joint-space singularities:

(1) boundary singularity, which occurs when the tool tip is on the surface of the work envelope. These are not particularly serious, because they can be avoided by mechanical constraints.

(2) interior singularity occurs inside the work envelope when two or more axes become collinear. These are more serious, because cancellation of counteracting rotations about an axis causes the tool tip position to remain constant.

From inspection of the Jacobian for the SCARA robot, equation (28), if and only if the upper left 2x2 submatrix becomes singular (|J(q)| = 0), the J(q) loses rank.

$$
\begin{aligned}
\det = |J(q)| &= [-a_1 \sin\theta_1 - a_2 \sin(\theta_1 - \theta_2)][-a_2 \cos\theta_2] \\
&\quad - [a_2 \sin(\theta_1 - \theta_2)][a_1 \cos\theta_1 + a_2 \cos(\theta_1 - \theta_2)] \\
&= a_1 a_2 \sin\theta_1 \cos(\theta_1 - \theta_2) + a_2^2 \sin(\theta_1 - \theta_2)\cos(\theta_1 - \theta_2) \\
&\quad - a_1 a_2 \sin(\theta_1 - \theta_2)\cos\theta_1 - a_2^2 \sin(\theta_1 - \theta_2)\cos(\theta_1 - \theta_2) \\
&= a_1 a_2 [\sin\theta_1 \cos(\theta_1 - \theta_2) - \cos\theta_1 \sin(\theta_1 - \theta_2)] \\
&= a_1 a_2 \sin\theta_2
\end{aligned}
\qquad (31)
$$

If sin( )=0, the Jacobian matrix is singular and has no inverse. This will occur when the elbow angle q_2 is an integer multiple of . The tool tip is on the outer surface of the work envelope (arm is reaching straight out). Whereas when $|q_2|=$ the arm is folded inside the surface of the work envelope.

7 Robot Arm Dynamics

Similar to the robot kinematic problem, there are two types of robot dynamic problems, a direct (forward) dynamic problem and an inverse dynamic problem, as shown in Figure 14. The problem of direct or forward dynamics is to calculate the joint trajectories (position, velocity, and acceleration), q(t), given the force (torque) profile, (t), which causes the desired motion trajectory:

$$\mathbf{q} = \mathbf{f}(\ ) \qquad (32)$$

This problem is important in computer simulation studies of robot manipulators. However, the vast majority of robot manipulators are driven by actuators which supply a force for the prismatic joint and a torque for the revolute joint to generate motion in a prescribed trajectory. The controller must also call for torques to compensate for inertia when each link is accelerated. Therefore, the problem of inverse dynamic is to design efficient control algorithms to compute accurate force (torque) (t), which causes the desired motion trajectories q(t):

$$= \mathbf{f}^{-1}(\mathbf{q}) \qquad (33)$$

This section presents the inverse dynamic equation of motion for robot manipulators. The robot arm kinematic solution along with the velocity and the acceleration of the link coordinates will be used to obtain the inverse dynamic model for the four-axis SCARA robot (AdeptOne).

In order to control the robot manipulator in real time, at adequate sampling frequency, it is necessary to balance the generalized torques accurately and frequently from four sources [McKerrow, 1991]:

1) dynamic source, arising from the motion of the robot:
   (i) inertia, mass property resisting to change in motion, proportional to joint acceleration.
   (ii) Coriolis, vertical forces derived from the link interactions, proportional to the product of the joint velocities.
   iii) centripetal forces, constraining rotation about a point, proportional to the square of the joint velocity.
2) static source, arising from friction in the joint mechanism.
3) gravity source, arising from force of gravity on each link.
4) external source, arising from external loads (tasks) on the end-effector.

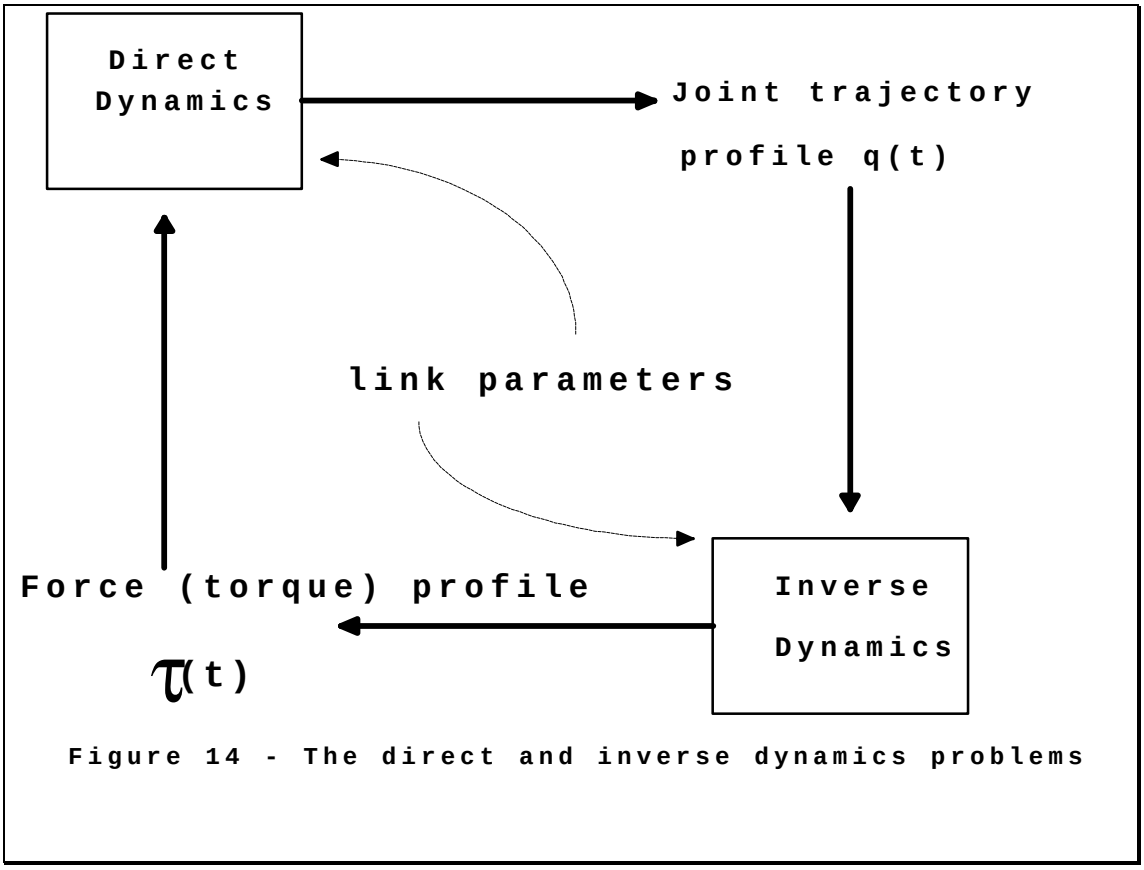


Figure 14 - The direct and inverse dynamics problems

We can formulate the following expression for the inverse dynamics problem:

$$\blacklozenge = M(\vec{x})\ddot{\vec{x}} + C(\vec{x},\dot{\vec{x}}) + F(\dot{\vec{x}}) + G(\vec{x}) + \blacklozenge_d \qquad (34)$$

where, $M(\vec{x})$, (nxn) inertia matrix of the robot.

$C(\vec{x},\dot{\vec{x}})$, (nx1) vector of centrifugal and Coriolis terms.

$F(\dot{\vec{x}})$, (nx1) friction vector.

$G(\vec{x})$, (nx1) gravity vector.

$\blacklozenge_d$, disturbance due to unknown loading

Detail discussions on structure and dynamic properties of the robot equations of motion can be found in Lewis *et al.* [1993] and Schilling [1990]. An excellent collection of early kinematics and dynamics research articles, discussed in this section, is also available by Lee *et al.* [1986].

The dynamic characteristics of robot manipulators are highly nonlinear and therefore require a great number of mathematical
operations. There are two forms of inverse dynamic solutions for a robot: (1) closed form, and (2) recursive form. If manipulators are kinematically and dynamically simple in design, an analytical expression for the closed form dynamic equations can be derived [Craig, 1986]. Thus, the final analytical expression will have simple physical interpretation in terms of all the dynamic properties described above.

An analytical approach based on the Lagrange's energy function, known as Lagrange-Euler method (L-E), results in a dynamic solution that is simple and systematic. In this method, the kinetic energy (K) and the potential energy (P) are expressed in terms of joint motion trajectories. The resulting differential equations then provide the forces (torques) which drive the robot. Closed form equations result in a structure that is very useful for robot control design and also guarantee a solution. However, its drawback is that it requires redundant calculations. For instance, when a force (torque) is applied to the end-effector, of a serial link manipulator, the joint interaction results in considerable duplication of calculation in the subsequent link equations.

This duplication can be avoided in the recursive form and calculation can be made more efficient. In addition, the L-E method is inefficient, mainly because it uses the (4x4) homogeneous transformation matrices that are somewhat sparse due to combination of rotation and translation. Various attempts have been reported to simplify and improve the computational efficiency of the L-E formulation [Hollerbach, 1980] and [Renaud, 1984]. In general, these approximations, when used for control purposes, result in suboptimal dynamic performance (lower speed and position inaccuracy) [Lee, 1982].

One recursive approach called the Newton-Euler (N-E) formulation, has the advantage of speed and accuracy for on-line implementation [Luh *et al*., 1980]. The N-E method is based on the Newton's mass center theorem and the Euler's theory of kinetic momentum applied at each robot link. Each link is considered to be a free body and the equations of motion are obtained for each link in a recursive manner. It uses a set of forward and

backward recursive equations. The forward iteration propagates kinematic information from base frame to the end-effector. Once the forward iteration is complete, the backward iteration calculates and propagates the joint forces (torques) exerted on each link from end-effector back to the base frame. The N-E formulation is simple and very fast. However, the derivation is messy (vector cross product terms) and recursive equations destroy the structure of the dynamic model, which is very important for the design of robot controllers [Lee, 1982].

The advantage of using recursive form (numerical) over the closed form (analytical) is only the speed of computation, particularly as the number of axes increases. The L-E method has a computational complexity of order $O(n^4)$, n being the number of axis. This is in contrast to N-E formulation which has a computational complexity of order O(n).

Lee *et al.* [1983] also obtained an efficient set of closed form solutions based on the generalized d'Alembert (G-D) principle that retain the structure of the problem with a moderate computational complexity of order $O(n^3)$. A symbolic program called Algebraic Robot Modeler (AMR) has also been reported to generate efficient customized dynamic equations for a variety of robots [Neuman and Murray, 1987].

In the following section, the complete dynamic model of the four-axis SCARA robot (AdeptOne) is derived based on the L-E formulation (see Figure 13). The (AdeptOne) SCARA robot is kinematically simple, and its unique direct drive actuating system eliminates gear friction and backlash. It has a closed form dynamic solutions which will provide the simple physical interpretations (inertia, centrifugal and Coriolis forces, friction and gravity) necessary to design the robot controller.

The Lagrange-Euler formulation. The Lagrange-Euler method describes the dynamic behavior of a robot in terms of work and energy stored in the system. The constraining forces are eliminated during the formulation and the closed form solution is derived independent of any coordinate system. This method is based on the utilization of [Spong and Vidyasagar, 1989]:

1) The (4x4) Denavit-Hartenberg matrix representation, $^{i}\mathbf{A}_{i-1}$, which describes the spatial relationship between the ith and (i-1)th link coordinate frames.
2) Fundamental properties of kinetic energy (K) and potential energy (P).
3) Lagrange's equation of motion [Koivo, 1989]:

$$\blacklozenge_i = \frac{d}{dt}\left[\frac{\partial L}{\partial \dot{q}_i}\right] - \frac{\partial L}{\partial q_i} \qquad i=1,2,\ldots,n \quad links \tag{35}$$

where L = lagrangian function = (K - P)

q_i = generalized coordinates of the robot arm

$\dot{q}_i$ = first derivative of q_i

$\blacklozenge_i$ = generalized torques corresponding to q_i

The generalized torques act on link i in the direction of the q_i coordinate frame. In the case of revolute joint, it is composed of torque vector, or when prismatic it is a force vector.

Since potential energy is only position dependent, equation (35) can be further defined as

$$\blacklozenge_i = \frac{d}{dt}\left[\frac{\partial K}{\partial \dot{q}_i}\right] - \frac{\partial K}{\partial q_i} + \frac{\partial P}{\partial q_i} \tag{36}$$

Let us begin by deriving the kinetic energy stored in a moving robot manipulator link (i) with both translation and rotation, in three dimensional space:

$$^{i}K = \tfrac{1}{2}\, ^{i}m\, ^{i}v_0^{T}\, ^{i}v_0 + \tfrac{1}{2}\, ^{i}\omega_0^{T}\, ^{i}I_0\, ^{i}\omega_0 \tag{37}$$

where ^{i}m is the mass of link i

^{i} is the (3x1) linear velocity vector of the center mass with respect to reference frame

^{i} is the (3x1) angular velocity vector of the link i with respect to reference frame

$^{i}\boldsymbol{I_0}$ is the (3x3) inertia tensor matrix of link i

Since energy is additive, the total kinetic energy stored in the whole arm linkage is then given by

$$K = \sum_{i=1}^{n} {}^{i}K \tag{38}$$

Recall the homogeneous transformation matrix, $^{i}\mathbf{T_0}$, that gives the position and orientation of any link, $^{i}\mathbf{r}$, with respect to base frame as

$$r = {}^{i}T_o \, {}^{i}r \tag{39}$$

Differentiation with respect to time gives the velocity of the link position, ^{i} , with respect to base frame as

$$ {}^{i}\dot{\theta}_0 = \frac{dr}{dt} = \frac{d}{dt}({}^{i}T_0){}^{i}r = \sum_{j=1}^{i} \frac{\partial {}^{i}T_o}{\partial \theta_j} \dot{\theta}_j \, {}^{i}r \tag{40}$$

Substituting expression (40) in (37) and subsequently in (38) yields

$$K = \sum_{i=1}^{n} \frac{1}{2} {}^{i}m \left(\sum_{j=1}^{i} \frac{\partial {}^{i}T_o}{\partial \theta_j} \dot{\theta}_j \, {}^{i}r \right)^{T} \sum_{j=1}^{i} \frac{\partial {}^{i}T_o}{\partial \theta_j} \dot{\theta}_j \, {}^{i}r$$

$$+ \frac{1}{2} \left(\sum_{j=1}^{i} \frac{\partial {}^{i}T_o}{\partial \theta_j} \dot{\theta}_j \, {}^{i}r \right)^{T} {}^{i}I_0 \sum_{j=1}^{i} \frac{\partial {}^{i}T_o}{\partial \theta_j} \dot{\theta}_j \, {}^{i}r$$

$$K = \sum_{i=1}^{n} \frac{1}{2} {}^{i}m \sum_{j=1}^{i} \sum_{k=1}^{i} \frac{\partial {}^{i}T_o}{\partial \theta_j}^{T} {}^{i}r^{T} \, {}^{i}r \frac{\partial {}^{i}T_o}{\partial \theta_k} \dot{\theta}_j \dot{\theta}_k$$

$$+ \frac{1}{2} \sum_{j=1}^{i} \sum_{k=1}^{i} \frac{\partial {}^{i}T_o}{\partial \eta_j}^{T} {}^{i}r^{T} \, {}^{i}I_0 \, {}^{i}r \, \frac{\partial {}^{i}T_o}{\partial \eta_k} \dot{\eta}_j \dot{\eta}_k \qquad (41)$$

Asada and Slotine [1986] suggest to rewrite the expressions in the equation (41) by using the (nxn) *manipulator inertia tensor matrix*, **M**,

$$M = \sum_{i=1}^{n} \left[{}^{i}m \sum_{j=1}^{i} \sum_{k=1}^{i} \frac{\partial {}^{i}T_o}{\partial \eta_j}^{T} {}^{i}r^{T} \, {}^{i}r \, \frac{\partial {}^{i}T_o}{\partial \eta_k} + \sum_{j=1}^{i} \sum_{k=1}^{i} \frac{\partial {}^{i}T_o}{\partial \eta_j}^{T} {}^{i}r^{T} \, {}^{i}I_0 \, {}^{i}r \, \frac{\partial {}^{i}T_o}{\partial \eta_k} \right] \qquad (42)$$

The matrix **M** incorporates all the mass properties of the entire arm linkage. The manipulator inertia matrix, also called mass matrix, is symmetric and in quadratic form. Since the kinetic energy is positive, unless the robot is at rest, the inertia matrix is *positive definite*.

$$K = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} M_{ij} \dot{\eta}_i \dot{\eta}_j \qquad (43)$$

Note that M_{ij}, component of inertia matrix **M**, is a function of joint variables **q** and represents coupling between the i and j links. The diagonal term M_{ii} represents the self-inertial coefficients. Since the mass matrix **M** is positive definite, the coefficient of coupling falls between zero (no inertial interaction) and one (tightly coupled).

Since the mass matrix **M** involves Jacobian matrices, which are configuration dependent and can vary, two links can be highly coupled in one configuration and completely decoupled in another. Desirably, the manipulator inertia matrix would be diagonal with constant self-inertial terms. This will allow the dynamic properties to stay the same for all configurations and, the control algorithms simple [Tourassis and Neuman, 1985].

The kinetic energy depends on q and dq/dt. Therefore,

$$K(q,\dot{q}) = \tfrac{1}{2}\dot{q}^T M(q)\dot{q} \qquad (44)$$

The potential energy of position vector defined in (36) in a gravity field $\mathbf{g} = [g_x\ g_y\ g_z\ 0]$ is

$${}^iP = -{}^i m\, g\, {}^iT_o\, {}^i r \qquad (45)$$

Then the total arm potential energy is

$$P = -\sum_{i=1}^{n} m_i\, g\, {}^iT_o\, {}^i r \qquad (46)$$

Note that the potential energy depends only on the joint variable q.

The terms required in the Lagrangian equation (36) are now given by

1st term: $$\frac{d}{dt}\left[\frac{\partial K}{\partial \dot{q}_i}\right] = M(q)\ddot{q} + \dot{M}(q)\dot{q}$$

2nd term: $$\frac{\partial K}{\partial q_i} = \frac{1}{2}\frac{\partial}{\partial q}(\dot{q}^T M(q)\dot{q})$$

3rd term: $$\frac{\partial P}{\partial q_i} = -\sum_{i=1}^{n} m_i\, g\, {}^iT_o\, {}^i r$$

Therefore, the arm dynamic equation is

$$\blacklozenge = M(q)\ddot{q} + \dot{M}(q)\dot{q} - \frac{1}{2}\frac{\partial}{\partial q}(\dot{q}^T M(q)\dot{q}) + \sum_{i=1}^{n} m_i g\, {}^iT_o\, {}^i r \qquad (47)$$

Defining the Coriolis/centripetal vector as

$$C(q,\dot{q}) = \dot{M}(q)\dot{q} - \frac{1}{2}\frac{\partial}{\partial q}(\dot{q}^T M(q)\dot{q}) \qquad (48)$$

The final dynamic form defined in (34) is derived. The friction and disturbance terms can be added to complete the dynamic equation. The robot dynamic equation represents nonlinearities due to coordinate transformations which are trigonometric. Additional nonlinearities appear in both kinetic and potential energy terms due to Coriolis and centrifugal terms. The Coriolis terms represent the velocity coupling of links j and k felt at joint i, in torque or force form. Whereas, the centrifugal terms reflect the velocity coupling of only link j at joint i. The Coriolis and centrifugal coefficients are small and become important when the robot is moving at high speed.

The gravity coefficients arise from the potential energy stored within individual links. These coefficients also vary with configuration. Equation (47) provides the dynamic properties based on the assumption that no loads are attached to the arm. Effect of a load on the dynamic coefficients is additive and therefore can be considered as a point mass and an extension of the last link (tool). Robot manipulators use actuators (electric or hydraulic) to move. Since the actuator and motor inertias are decoupled values acting only at the joints, they can be treated as additive terms to the mass matrix [Lewis *et al.*, 1993]. Also, note that each torque (force) computation involves three summation over a possible range of n joints. This creates a computation complexity in the order of $O(n^4)$. This high order of calculations is very slow for on-line implementation. In practice, the manipulator dynamic must be modeled accurately for precise and high speed motion control.

Dynamic model of the SCARA robot. Tasks performed by robots are defined previously as gross manipulation and fine manipulation tasks. Motion trajectory control of the end-effector applies to the tasks in the first category. Robots, in general, use the first three axis for gross manipulation (position control) while the remaining axis orient the tool during the fine manipulation (force or tactile control).

Robots, in parts assembly applications, are to pick a component up with a vertical movement, move it horizontally and then downwards vertically for insertion. These can be achieved by the four-axis SCARA robots. Examples of robots which belong to this class include the AdeptOne robot, the intelledex 440 robot, and the IBM 7545 robot. The

first three axes of a SCARA robot position the end-effector, while the fourth axis orients the tool through a roll motion [Schilling, 1990].

This section provides the dynamic equations of AdeptOne SCARA robot, based on the L-E formulation discussed above. AdeptOne robot is a direct-drive robot. As there is no gearbox, the friction at each joint is very small. The vector of joint variables is

$$\begin{bmatrix} \theta_1 & \theta_2 & d_3 & \theta_4 \end{bmatrix}$$

where, the third joint is prismatic while the remaining three joints are revolute.

From reviewing the results of Table 4, it is clear that the tool roll angle θ_4 has no effect on the end-effector position, therefore θ_4 can be ignored in our investigation of the motion trajectory control. The mass of the fourth link and the attached end-effector is also assumed to be very small in comparison with the masses of the other links.

The link-coordinate diagram for this robot is shown in Figure 13 for the first three axes. The three links are assumed to be thin cylinders of mass m_1, m_2, and m_3. The vertical column height d_1 is stationary, and when joint 1 is activated, only rod of length a_1 (mass of m_1) rotates. The dynamic relationships are governed by the equation 47.

$$\blacklozenge = M(q)\ddot{q} + \dot{M}(q)\dot{q} - \frac{1}{2}\frac{\partial}{\partial q}(\dot{q}^T M(q)\dot{q}) + \sum_{i=1}^{n} m_i g\, {}^{i}T_o\, {}^{i}r$$

where the matrix M(q) is (nxn) symmetric and positive definite with elements $m_{ij}(q)$, and n is the number of joints. We will adopt the following notation:

l_i, *length of link* i

m_i, mass of link i

mass moment of inertia of link i about its center of

mass of thin cylinder $I = (m_i a_i^2)/12$.

g, gravity constant.

Then the dynamic equation for the 1st joint of the manipulator would be

$$\blacklozenge_1 = \left[\left(\frac{m_1}{3} + m_2 + m_3\right) a_1^2 + (m_2 + 2m_3) a_1 a_2 \cos\theta_2 + \left(\frac{m_2}{3} + m_3\right) a_2^2 \right] \ddot{\theta}_1 \qquad (49)$$

$$-\left[\left(\frac{m_2}{2}+m_3\right)a_1a_2\cos\theta_2+\left(\frac{m_2}{3}+m_3\right)a_2^2\right]\ddot{\theta}_2$$
$$-(m_2+2m_3)\,a_1a_2\sin\theta_2\dot{\theta}_1\dot{\theta}_2+\left(\frac{m_2}{2}+m_3\right)a_1a_2\sin\theta_2\dot{\theta}_2^2$$

Since axis 1 is aligned with the gravitational field, the gravity term on joint one is zero.

2nd joint:

$$\tau_2 = -\left[\left(\frac{m_2}{2}+m_3\right)a_1a_2\cos\theta_2+\left(\frac{m_2}{3}+m_3\right)a_2^2\right]\ddot{\theta}_1 + \left(\frac{m_2}{3}+m_3\right)a_2^2\ddot{\theta}_2+\left(\frac{m_2}{2}+m_3\right)a_1a_2\sin\theta_2\dot{\theta}_1^2 \quad (50)$$

Again, there is no gravitational loading on joint two.

3rd joint:

$$\tau_3 = m_3\ddot{\theta}_3 - gm_3 \quad (51)$$

Joint three is a prismatic joint, used to establish the vertical tool position, and is completely independent of the first two joints. Thus, there is no Coriolis and centrifugal forces on this joint. However, The mass of the third joint effects the motion of the first two joints by acting as a load.

Equations (49-51) are referred to as the inverse dynamic equations of the three-axis SCARA robot. One needs to know the resulting motion trajectories: θ_i $\dot{\theta}_i$ $\ddot{\theta}_i$ before one can calculate the joint torques, . These equations are time varying, nonlinear, and coupled differential equations. The trajectory problem of this robot manipulator revolves around finding the joint torques $_i$(t) such that the joint angles $_i$(t) track the desired trajectories $_{id}$(t). Torque-based control techniques are built directly on the dynamic models of the robot manipulators.

8 Robot Neural Controller

In order to design intelligent robot controllers, one must also provide the robot with a means of responding to problems on the temporal context (time) as well as spatial (space). It is the goal of the intelligent robot researchers to design a neural learning controller to

utilize the available data from the repetition in the robot operation. The neural learning controller based on the recurrent network architecture has the time-varying feature that once a trajectory is learned, it should learn a second one in a shorter time.

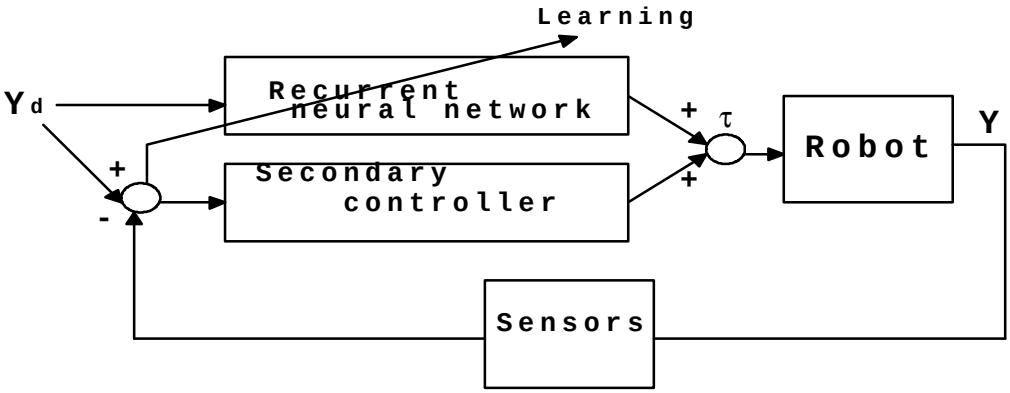


Figure 15 - Recurrent neural learning controller

In Figure 15, the time-varying recurrent network will provide the learning block (primary controller) for the inverse dynamic equations discussed above. The network compares the desired trajectories: θ_d, $\dot{\theta}_d$, $\ddot{\theta}_d$, with continuous paired values for the three-axis robot θ $\dot{\theta}$ $\ddot{\theta}$: ♦, at every instant in a sampling time period. The new trajectory parameters are then combined with the error signal from the secondary controller (feedback controller) for actuating the robot manipulator arm.

Neural networks can be applied in two ways in the design of the robot controller described in Figure 4: (1) system identification model and (2) control. ANNs can be used to obtain the system model identification which can be used to design the appropriate controller. They can also be used directly in design of the controller [Narendra and Parthasarathy, 1990] once the real system model is available. Neural network approaches to robot control is discussed in general by Psaltis *et al.* [1988], and Yabuta and Yamada [1992]. These approaches can be classified into:

1. **Supervised control**, a trainable controller unlike the old teaching pendant, allows responsiveness to sensory inputs. A trainable neuromorphic controller reported by Guez and Selinsky [1988] provides an example of a fast, real time, and robust controller.
2. **Direct inverse control**, is trained for the inverse dynamic of the robot. Kung and Hwang [1989] used two networks on-line in their design of the controller.
3. **Neural adaptive control**, neural nets combined with adaptive controllers, results in greater robustness and ability to handle nonlinearity. Chen [1990] reported the use of the BP method for a nonlinear self-tuning adaptive controller.
4. **Backpropagation of utility**, involves information flowing backward through time. Werbos's backpropagation through time is an example of such technique [Werbos, 1990].
5. **Adaptive critic method**, uses a critic evaluating robot performance during training. Very complex method; requires more testing [Werbos, 1991].

In the direct inverse control approach, the recurrent neural network will learn the inverse dynamic of the robot in order to improve the controller performance. The neural network model replaces the primary controller (see Figure 4). In this approach a feedback controller (secondary controller) will be used to teach the network initially. As learning takes place, the neural network takes full control of the system.

Kawato and his research group were successful to use this approach in trajectory control of a three degree-of-freedom robot [Kawato *et al.*, 1987; Miyamoto e*t al*., 1988]. Their approach is known as feedback-error-learning control. However, their neural network structure was simply the linear collection of all nonlinear dynamic terms (they called them subsystems) in the dynamic motion equation. Learning was purely for the estimates of the subsystems. As the degree of freedom increases the network size needs to increase (order of n^4). For example, for six degree-of-freedom 942 subsystems are needed, compared with 43 for the three degree-of-freedom. However, due to the parallel processing capability of the neural network, the implementation of Kawato's method is still an attractive method.

Goldberg and Pearlmutter [1989] have demonstrated the utility of the feedback-error-learning approach for the motion control of the first two joints of the CMU DDArm II, using temporal windows of measured positions as input to the network. The output of the network is the torque vector. Newton and Xu [1993] used this approach to control a flexible space robot manipulator (SM^2) in real-time. The trajectory tracking error was reduced by 85% when compared to conventional PID control scheme. More recently, Lewis *et al.* [1995] developed an on-line neural controller, based on the robot passivity properties (system cannot go unstable if the robot cannot create energy), using a similar approach with good tracking results. The feasibility and performance of the feedback-error-learning control with global asymptotic stability has also been reported [Kawato, 1990 and Patino *et al.*, 1994]. The design of a compact and generic *recurrent network* has also shown promising results in replacing the need for custom subsytems-type design such as the one by Kawato's group [Golnazarian, 1995]. The proposed controller performs based on the systematic design approach and the recurrent network's time-varying feature.

References

1. Asada, H. and Slotine, J.E., Robot Analysis and Control, pp. 133-183, John Wiley & Sons, Inc., New York, 1986.

2. Asfahl, C.R., Robots and Manufacturing Automation, pp. 1-10, John Wiley & Sons, Inc., New York NY, 1992.

3. Baba, N., "A new approach for finding the global minimum of error function of neural networks", Neural Networks, vol. 2, pp. 367-373, 1989.

4. Boothroyd, G., Poli, C., and Murch, L.E., Automatic Assembly, Marcel Dekker, New York NY, 1982.

5. Caudell, M. and Butler, C., Understanding Neural Networks: Computer Exploration, Advanced Networks, vol. 2, pp. 79-112, The MIT Press, Cambridge MA, 1992.

6. Chapnick, P., "Lots of neural nets books", AI Expert, pp. 21-23, June 1992.

7. Chen, F., "Back-propagation neural networks for nonlinear self-tuning adaptive control", IEEE Control Systems Magazine, pp. 44-48, April 1990.

8. Chester, M., Neural Networks: A Tutorial, Prentice Hall, Englewood Cliffs New Jersey, 1993.

9. Craig, J.J., Introduction to Robotics: Mechanics and Control, Addison-Wesley Publishing Company, Reading MA, 1986.

10. Davies, O., "Vision Fires Up Parts Inspection", Managing Automation, vol. 7(6), part 2 of 2, pp. 14, 16 & 36, June 1992.

11. Denavit, J. and Hartenberg, R.S., "A kinematic notation for lower-pair mechanisms based on matrices", Journal of Applied Mechanics, pp. 215-221, June 1955.

12. Farnum, G., "Robots Figure in Flexible Assembly", Managing Automation, vol. 7(6), part 2 of 2, pp. 9, June 1992.

13. Fu, K.S., Gonzalez, R.C., and Lee, C.S.G., Robotics: Control, Sensing, Vision and Intelligence, pp. 201-223, McGraw-Hill, New York NY, 1987.

14. Goldberg, K.Y., and Pearlmutter, B.A., "Using backpropagation with temporal windows to learn the dynamics of the CMU direct-drive arm II", Advances in Neural Information Processing Systems I", D.S. Touretzky (ed.), pp. 356-363, Morgan Haufmann Publishers Inc., Palo Alto CA, 1989.

15. Goldenberg, A.A. and Lawrence, D.L., "A generalized solution to the inverse kinematics of robotic manipulator", Journal of Dynamics Systems, Measurement, and Control, Vol. 107, pp. 103-106, March 1985.

16. Golnazarian, W., Hall, E.L., and Shell, R.L., "Robot control using neural networks with adaptive learning steps", SPIE Conference Proceedings, Intelligent Robots and Computer Vision XI: Biological, Neural Net, and 3-D Methods, vol. 1826, pp. 122-129, 1992.

17. Golnazarian, W., “Time-varying Neural Networks for Robot Trajectory Control”, Ph.D. Dissertation, University of Cincinnati OH, 1995.

18. Grossberg, S., Studies of Mind and Brain, Vol. 70 of Boston Studies in the Philosophy of Science, D. Reidel Publishing Company, Boston MA, 1982.

19. Grossberg, S.,(ed.), Neural Networks and Natural Intelligence, The MIT Press, Cambridge MA, 1988.

20. Guez, A. and Selinsky, J., "A trainable Neuromorphic Controller", Journal of Robotic Systems, Vol. 5(4), pp. 363-388, 1988.

21. Guez, A., Ahmad, Z., and Selinsky, J., "The application of neural networks to robotics", Neural Networks: Current Applications, P.G.J. Lisboa (ed.), pp. 111-122, Chapman & Hall, London, 1992.

22. Hall E.L., Slutzky G.D., and Shell R.L., "Intelligent Robots for Automated Packaging and Processing", Quality Use of the Computer: Computational Mechanics, Artificial Intelligence, Robotics and Acoustic Sensing, vol. 177, pp. 141-146, 1989.

23. Hall, E.L. and Hall, B.C., Robotics: A User-Friendly Introduction, pp. 1-8, Saunders College Publishing, Holt, Rienhart and Wilson, Orlando FL, 1985.

24. Hebb, D.O., The Organization of Behavior: A Neuropsychological Theory, Wiley, New York NY, 1949.

25. Hecht-Nielsen, R., Neurocomputing, pp. 182-190, Addison-Wesley, Reading MA, 1990.

26. Hertz, J., Krogh A., and Palmer, R. G., Introduction to the Theory of Neural Computation, Addison-Wesley, Reading MA, 1991.

27. Hokestra, R. L., "Design for assembly", Ph.D. Dissertation, University of Cincinnati, Cincinnati OH, 1992.

28. Hollerbach, J., "A recursive lagrangian formulation of manipulator dynamics and a

comparative study of dynamics formulation complexity", IEEE Transactions on Systems, Man, and Cybernetics", vol. SMC-10, no. 11, pp. 730-736, November 1980.

29. Holusha, J., "Industrial robots make the grade", The New York Times, pp. c1 & d5, Wednesday September 7th, 1994.

30. Hopfield, J.J. and Tank, T.W., "'Neural' computation of decisions in optimization problems", Biological Cybernetics, vol. 52, pp. 141-152, 1985.

31. Jacobs, R.A., "Increased Rates of Convergence Through learning rate adaptation", Neural Networks, vol. 1, pp. 295-307, 1988.

32. Josin, G., Charney, D., and White, D., "Robot control using neural networks", IEEE international conference on neural networks, vol. 2, pp. 625-631, 1988.

33. Kaiser, D., "Valves, Servos, Motors, and Robots", Process/Industrial Instruments and Controls Handbook, Fourth edition, D.M. Considine (editor-in-chief), pp. 9.86-9.87, McGraw-Hill, New York NY, 1993.

34. Kawato, M., "Feedback-error-learning neural network for supervised motor learning", Advanced Neural Computers, R. Eckmiller (ed.), pp. 365-372, Elsevier Science Publishers B.V. (North-Holland), 1990.

35. Kawato, M., Furukawa, K., and Suzuki, R., "A hierarchical neural-network model for control and learning of voluntary movement", Biological Cybernetics, vol. 57, pp. 169-185, 1987.

36. Klafter, R.D., Chmielewski, T.A., and Negin, M., Robotic Engineering: An Integrated Approach, pp. 244-247, Prentice Hall, Englewood Cliffs New Jersey, 1989.

37. Kohonen, T., "Introduction to neural computing", Neural Networks, vol. 1, pp. 3-16, 1988.

38. Kohonen, T., "Self-organized formation of topologically correct feature maps", Biological Cybernetics, vol. 43, pp. 59-69, 1982.

39. Koivo, A.J., Fundamentals for Control of Robotic Manipulators, pp. 306-338, John Wiley & Sons, Inc., New York NY, 1989.

40. Koivo, A.J. and Guo T., "Adaptive linear controller for robotic manipulator", IEEE Transactions on Automatic Control, vol. AC-28, number 2, pp. 162-171, February 1983.

41. Kosko, B., "Bi-directional Associative Memories", IEEE Transactions on System, Man, and Cybernetics", vol. 18(1), pp. 49-60, Jan-Feb. 1988.

42. Kung S. and Hwang J., "Neural network architectures for robotic applications", IEEE Transactions on Robotics and Automation, vol. 5(5), pp. 641-657, February 1989.

43. Kuperstein, M. and Wang, J., "Neural controller for adaptive movements with unforeseen payload", IEEE Transactions on Neural Networks, vol. 1(1), pp. 137-142, 1990.

44. Labrenz, D., "Robots Lend Muscle to Palletizing", Managing Automation, vol. 7(6), part 2 of 2, pp. 16-20, June 1992.

45. Lau, C. (Ed.), Neural Networks: Theoretical Foundations and Analysis, IEEE Press, New York NY, 1992.

46. Lee, C.S.G., "Robots Arm Kinematics, Dynamics, and Control", Computer, Vol. 15(12), pp. 62-80, December 1982.

47. Lee, C.S.G., Lee, B.H., and Nigham, R., "Development of the generalized d'Alembert equations of motion for mechanical manipulator", Proceedings of the 22nd Conference on Decision and Control, December 14-16 1983.

48. Lewis, F.L., Abdallah, C.T., and Dawson, D.M., Control of Robot Manipulators, pp. 140-158, Macmillan Publishing Company, New York NY, 1993.

49. Lewis, F.L., Liu, K., and Yesildirek, "Neural net robot controller with guaranteed tracking performance", IEEE Transactions on Neural Networks, vol. 6(3), pp. 703-

715, May 1995.

50. Lippman, R.P., "An introduction to computing with neural nets", IEEE ASSP Magazine, 4(2), pp. 4-22, April 1987.

51. Luh, L.Y.S., Walker, M., and Paul, R., "Resolved-acceleration Control of Mechanical Manipulators", IEEE Transactions on Automatic Control, AC-25, pp. 468-474, 1980.

52. McKerrow, P.J., Introduction to Robotics, pp. 14-23, Addison-Wesley, Reading MA, 1991.

53. Minsky, M.L. and Papert, S.A., Perceptrons, Cambridge MA 1969.

54. Miyamoto, H., Kawato, M., Setoyama, T., and Suzuki, R., "Feedback error learning neural network model for trajectory control of a robotic manipulator", Neural Networks, vol. 1, pp. 251-265, 1988.

55. Narendra, K.S. and Parthasarathy, K., "Identification and control of dynamical systems using Neural networks", IEEE Transactions on Neural Networks, vol. 1(1), pp. 4-27, 1990.

56. Neuman, C.P. and Murray, J.J., "Customized computational robot dynamics", Journal of Robotics Systems, vol. 4(4), pp. 503-526, August 1987.

57. Newton, R.T. and Xu, Y., "Real-time implementation of neural network learning control of a flexible space manipulator", IEEE International Conference on Robotics and Automation, pp. 135-141, Atlanta Georgia, May 2-6, 1993.

58. Nguyen, L. and Patel, R.V., "A neural network based strategy for the inverse kinematics problem in robotics", Proceedings of International Symposia on Robotics and Manufacturing, vol. 3, pp. 995-1000, 1990.

59. Nurre, J.H. and Hall, E.L., "Three Dimensional Vision for Automated Inspection", Proceedings of Robots 13 Conference, pp. 16-1 to 16-11, Gaithersburg MD, May 7-11, 1989.

60. Odrey N.G., "Robotics: applications", The Electrical Engineering Handbook, R.C. Dorf (editor-in-chief), pp. 2175-2182, CRC Press, Inc., Boca Raton, Florida, 1993.

61. Oh, S., Orin, D., and Bach, M., "AN inverse kinematic solution for kinematically redundant robot Manipulators". Journal of Robotic Systems, vol. 1(3), pp. 235-249, 1984.

62. Patino, D., Carelli, R., and Kuchen, B., "Stability analysis of neural networks based adaptive controllers for robot manipulators", Proceedings of the American Control Conference, pp. 609-613, Baltimore MD, June 1994.

63. Paul, R.P., Robot Manipulators, MIT Press, Cambridge MA, 1981.

64. Pearlmutter, B.A., "learning state-space trajectories in recurrent neural networks", Neural Computation, vol. 1, pp. 263-269, 1989.

65. Pieper, D., "The kinematics of manipulators under computer control", Ph.D. Dissertation, Stanford University CA, 1968.

66. Pineda, F.J., "Recurrent backpropagation and the dynamical approach to adaptive neural computation", Neural Computation, vol. 1, pp. 161-172, 1989.

67. Psaltis D., Sideris A., and Yamamura A. A., "A multilayered neural network controller", IEEE Control Systems Magazine, vol. 8(2), pp. 17-21, 1988.

68. RIA, “Robotics Industry Has Best Year Ever in 1997”, Robotics Industries Association, Ann Arbor MI, http://www.robotics.org, 1998.

69. Renaud, M., "An efficient iterative analytical procedure for obtaining a robot manipulator dynamic model", Robotics Research, M. Brady (ed.), pp. 749-768, 1984.

70. Rottenbach, J., "Quality Takes a Seat via Welding", Managing Automation, vol. 7(6) part 2 of 2, pp. 16, June 1992.

71. Rumelhart, D.E., Hinton, G.E., and Williams, R.J. "Learning internal representation by error propagation", Parallel distributed processing: exploration in the microstructure of

cognition, D.E. Rumelhart and J.L. McClelland (eds.), vol. 1, pp. 318-362, MIT Press, Cambridge MA, 1986.

72. Sauer, K., "Robotic Arc Spray Eliminates Warping", Managing Automation, vol. 7(6) part 2 of 2, pp. 10, June 1992.

73. Schilling, R.J., Fundamentals of Robotics: Analysis and Control, Prentice Hall, Englewood Cliffs New Jersey, 1990.

74. Sinton, P., "Faster, flexible robots tackle the job", San Francisco Chronicle, Section B, pp. 1, column 6, May 15, 1995.

75. Spong, M.W. and Vidyasagar, M., Robot Dynamics and Control, Wiley, New York NY, 1989.

76. Tourassis, V.D. and Neuman, C.P., "Properties and structure of dynamic robot models for control engineering applications", Mechanism and Machine Theory, vol. 20(1), pp. 27-40, 1985.

77. Ty, A.J. and Tien, C.H., "Robotics: robot configuration", The Electrical Engineering Handbook, R.C. Dorf (editor-in-chief), pp. 2154-2162, CRC Press, Inc., Boca Raton Florida, 1993.

78. Vemuri, V., "Artificial neural networks: an introduction", Artificial Neural Networks: Theoretical Concepts, V. Vemuri (ed.), pp. 1-12, IEEE Computer Society Press, 1988.

79. Weil, M., "New competitiveness spurs record robot sales", Managing Automation, vol. 9(6) part 2 of 2, pp. 5-8, June 1994.

80. Werbos, P., "Backpropagation through time: what it does and how it does it", Proceedings of the IEEE, vol. 78, pp. 1550-1560, 1990.

81. Werbos, P., "An overview of neural networks for control", IEEE Control Systems Magazine, vol. 11(1), pp. 40-42, January 1991.

82. Widrow, B., and Lehr, M.A., "30 years of adaptive neural networks: Perceptron, madaline, and backpropagation", Proceedings of IEEE, vol. 78, pp. 1415-1442, 1990.

83. Wolovich, W.A., Robotics: Basic Analysis and Design, Holt, Rinehart and Winston, New York NY, 1986.

84. Yabuta, T. and Yamada, T., "Neural network controller characteristics with regard to adaptive control", IEEE Transactions on System, Man, and Cybernetics", vol. 22(1), pp. 170-176, January 1992.

85. Yoshikawa, T., Foundations of Robotics: Analysis and Control, pp. 1-12, The MIT Press, Cambridge MA, 1990.