Motion Control Algorithms for Sensor-Equipped Robots

M. Shoham
Assistant Professor,
Department of Mechanical Engineering,
Columbia University, New York, N.Y.

Y. Koren
Professor,
Department of Mechanical Engineering,
The University of Michigan,
Ann Arbor, Mich. 48109

This paper deals with the development of kinematic algorithms for the control of sensor-equipped robots. The kinematics is solved in the sensor coordinate system, which reduces the computation efforts, and allows the elimination of the first joint encoder. Simplification of the algorithms can be obtained when approximations are used to solve the inverse kinematics. Three control algorithms based on approximations are presented. However, with these algorithms, convergence to the target is not always guaranteed. A Theorem which specifies the sufficient conditions required for a trajectory to converge to a target point is proved. Based on this Theorem robot parameters can be selected in the design stage of the manipulator. This is illustrated for several types of manipulators.

1 Introduction

This paper deals with kinematical features of intelligent robots having a sensor located at the arm. A typical sensor might be a force-torque transducer or a camera located at the wrist section. With such robots, the motion commands are generated by the sensor and the robot must execute them in real time. This requires that the robot algorithms include control in sensor-oriented coordinates, rather than in world coordinates as in conventional robots. It is claimed that control in sensor (or object) oriented coordinates may be comparable in complexity to that required in conventional robots [1]. In this paper, it is shown that sensor-oriented control may be much simpler, and also much faster, compared with the control in world coordinate system.

Based upon a kinematic analysis, the paper proposes motion algorithms for sensor-equipped robots. Simplification of the algorithms can be obtained when approximations are used to solve the inverse kinematics. However, when using the approximations the robot might diverge from the target point. Therefore, sufficient conditions which guarantee the convergence in this case must be found.

The kinematic approach in this paper is based upon the resolved motion rate control method [2, 3] in which the required velocity of the end effector, s, is related to the joint variables θ_i by the equation

$$\mathbf{s} = \mathbf{J}\dot{\theta} \tag{1}$$

where $\mathbf{J}$ is a Jacobian 6×6 matrix, in a 6 degrees of freedom (DOF) robot, $\dot{\theta}$ is a vector of six joint[1] speeds, and s is given by

$$\mathbf{s} = \begin{bmatrix} \mathbf{v} \\ -- \\ \Omega \end{bmatrix} \tag{2}$$

where $\mathbf{v}$ is a three-element vector describing the velocity of the tool center point (TCP), and Ω is a three-element vector describing the angular velocity of the tool coordinate system.

The vector s and the matrix $\mathbf{J}$ must be given in the same coordinate system, for example in the equation

$$\mathbf{s}_w = \mathbf{J}_w \dot{\theta} \tag{3}$$

the subscript w implies that s and $\mathbf{J}$ are both given in the world coordinate system.

2 The Jacobian in the Sensor Coordinate System

The resolved motion rate control can be used for a sensor coordinate systems (SCS) as well. In the case of a sensor attached to the arm, the required velocity of the tool is provided by the sensor. The relationship between the tool velocity s, given in SCS, and the joint speeds is given by equation (1), but in this case $\mathbf{J}$ is expressed in the SCS.

The reference velocity of the tool is obtained by detecting the deviation between the target and the actual positions of the arm with the aid of the sensor. Examples might be vision or force sensors. With a vision sensor attached to the arm, the actual position and orientation of the target are detected. The velocity commands s are generated in the sensor in order to move the arm toward the object [4, 5].

Another example is a force sensor attached to the end of the arm and used to assemble a shaft into a long hole. The required force is constant during this assembly operation. The sensor measures the actual force between the robot tip and the body, and compares it to the required one. The error is translated into velocity command s, which causes a motion that eliminates, or reduces, this error [6].

The required joint speeds are derived from equation (1) by inverting the Jacobian:

$$\dot{\theta} = \mathbf{J}^{-1}\mathbf{s} \tag{4}$$

The computation of the Jacobian is based upon current joint variables (i.e., angles and link lengths) which are

[1] In this paper the word "joint" is used to denote a degree of freedom (or an axis of motion), whether it be a revolute joint or a prismatic joint.

Contributed by the Dynamic Systems and Control Division for publication in the Journal of Dynamic Systems, Measurement, and Control. Manuscript received by the Dynamic Systems and Control Division July 1985; revised manuscript received September 1986.

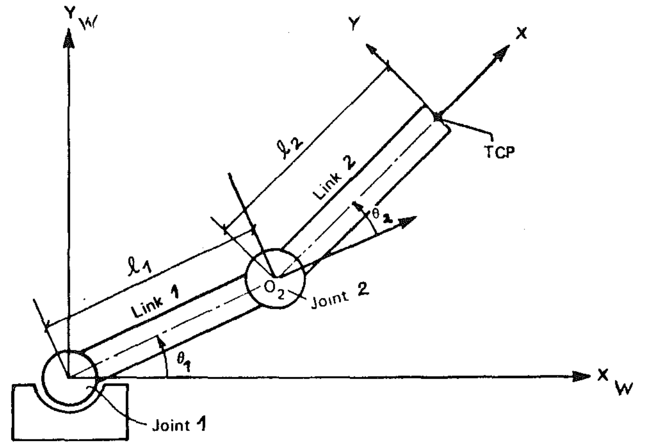

Fig. 1 Two-link planar manipulator

measured by the joint position detectors (e.g., encoders).

Several manipulators are analyzed below to demonstrate the advantage of expressing the Jacobian $\mathbf{J}$ in the sensor coordinate system.

Two-Joint Manipulators. The first example deals with the two-link planar manipulator used in SCARA[2]-type robots and shown in Fig. 1. Both rotary joints have the Z-axis as the axis of rotation, with θ_1 and θ_2 the joint angles corresponding to joints 1 and 2. The lengths for links 1 and 2 are l_1 and l_2, respectively. The Jacobian matrix which relates the joint velocities to the Cartesian velocities in the world coordinate system is [7, 8]

$$\mathbf{J}_w = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin\psi & -l_2 \sin\psi \\ l_1 \cos\theta_1 + l_2 \cos\psi & l_2 \cos\psi \end{bmatrix} \tag{5}$$

where $\psi = \theta_1 + \theta_2$.

Let us assume that a sensor located near the arm end provides two velocities v_x and v_y which comprise the velocity vector $\mathbf{v}$ (in SCS) at which the TCP should move

$$\mathbf{v} = \begin{bmatrix} v_x \\ v_y \end{bmatrix} \tag{6}$$

Note that in this particular case $\mathbf{s} = \mathbf{v}$.

The relation between the velocity vector in the sensor coordinate system and the world coordinate system is

$$\mathbf{s}_w = \mathbf{C}\mathbf{s} \tag{7}$$

and $\mathbf{C}$ is a transformation matrix. Eliminating s from equations (1) and (7) and comparing the results with equation (3) yields

$$J = \mathbf{C}^{-1} J_w \tag{8}$$

In this particular case the transformation matrix is

$$\mathbf{C} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \tag{9}$$

Substituting equations (5) and (9) into (8) yields

$$\mathbf{J} = \begin{bmatrix} l_1 \sin\theta_2 & 0 \\ l_2 + l_1 \cos\theta_2 & l_2 \end{bmatrix} \tag{10}$$

The derivation of the joint angle velocities from the sensor velocities is obtained by finding the inverse Jacobian from equation (10).

$$\begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \end{bmatrix} = \frac{1}{l_1 l_2 \sin\theta_2} \begin{bmatrix} l_2 & 0 \\ -(l_2 + l_1 \cos\theta_2) & l_1 \sin\theta_2 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \tag{11}$$

This result should be compared with the derivation of the joint velocities from the Cartesian velocities v_x and v_y given in the world coordinate system [8]

$$\begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \end{bmatrix} = \frac{1}{l_1 l_2 \sin\theta_2} \begin{bmatrix} l_2 \cos\psi & l_2 \sin\psi \\ -(l_1 \cos\theta_1 + l_2 \cos\psi) & -(l_1 \sin\theta_1 + l_2 \sin\psi) \end{bmatrix}_w \begin{bmatrix} v_x \\ v_y \end{bmatrix}_w \tag{12}$$

Besides the simpler expression of $\mathbf{J}^{-1}$ in equation (11) as compared with $\mathbf{J}^{-1}$ in equation (12), there is another significant result: $\mathbf{J}^{-1}$ is independent of θ_1. In conventional SCARA-type robots to which a sensor was added, that means eliminating real-time calculation of $\sin\theta_1$ and $\cos\theta_1$, and consequently increasing the allowable velocity of the manipulator. However, if a SCARA robot is designed at the outset as an intelligent robot operating with an appropriate sensor, then the designer might omit the encoder of joint 1. The relative position between the end effector and the object is measured continuously by the sensor, and the joint position is not required for the velocity calculations. Since a typical SCARA robot contains only three servo-controller axes (θ_1, θ_2, and Z), saving one encoder has an impact on the robot cost.

Three-Joint Manipulators. Similar results are obtained when analyzing cylindrical and spherical robots. If the joint variables of a cylindrical robot are called θ_1 (base angle), d_2 (vertical translation), and d_3 (horizontal translation), the Jacobian in sensor coordinates located at the arm end can be written as

$$\mathbf{J} = \begin{bmatrix} d_3 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{13}$$

where d_3 is the variable length of the second link. Notice that neither θ_1 nor d_2 appear in equation (13).

The world coordinate Jacobian for a spherical robot is [8]

$$\mathbf{J}_w = \begin{bmatrix} -d_3 \sin\theta_1 \cos\theta_2 & -d_3 \cos\theta_1 \sin\theta_2 & \cos\theta_1 \cos\theta_2 \\ d_3 \cos\theta_1 \cos\theta_2 & -d_3 \sin\theta_1 \sin\theta_2 & \sin\theta_1 \cos\theta_2 \\ 0 & d_3 \cos\theta_2 & \sin\theta_2 \end{bmatrix} \tag{14}$$

and the transform matrix is

$$\mathbf{C} = \begin{bmatrix} -\sin\theta_1 & -\cos\theta_1 \sin\theta_2 & \cos\theta_1 \cos\theta_2 \\ \cos\theta_1 & -\sin\theta_1 \sin\theta_2 & \sin\theta_1 \cos\theta_2 \\ 0 & \cos\theta_2 & \sin\theta_2 \end{bmatrix} \tag{15}$$

The sensor coordinate system is defined in coordinate frame s as shown in Fig. 2. (Y is pointing up in the direction of θ_2.) Substitution of equations (14) and (15) into equation (8) yields the Jacobian matrix as expressed in the sensor coordinate system

$$\mathbf{J} = \begin{bmatrix} d_3 \cos\theta_2 & 0 & 0 \\ 0 & d_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{16}$$

Notice that again θ_1 does not appear in equation (16), which

[2] SCARA stands for selective compliance assembly robot arm.

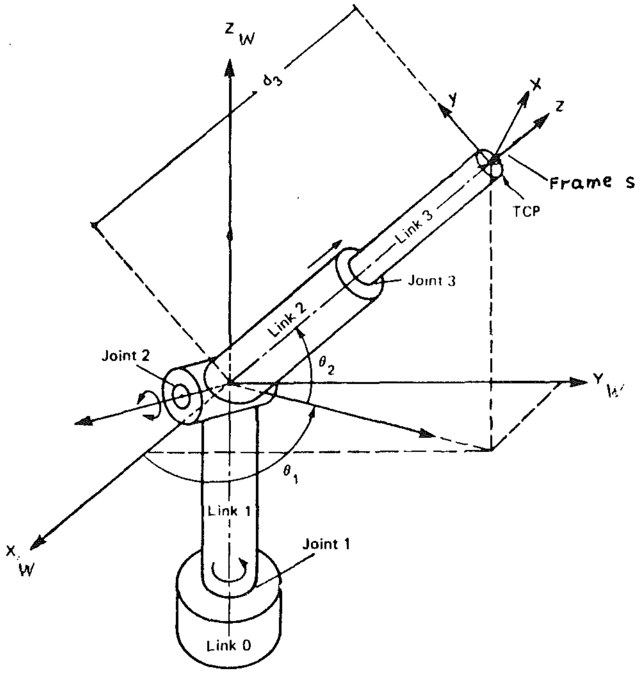

Fig. 2 Three-DOF spherical robot

consequently eliminates real-time calculations of $\cos\theta_1$ and $\sin\theta_1$.

Although it has been shown above that omitting the first joint encoder is possible if **J** is computed in sensor coordinates and the motion commands are given in sensor coordinates as well, there might be practical reasons for avoiding the full implementation of this result. Such cases occur, for example, when the endpoint sensor cannot sense the entire workvolume, or when a joint-velocity signal is needed for the joint controller. In the first case, however, it is sufficient to add a low-resolution encoder or a potentiometer to bring the endpoint to a region where the sensor is active. In the latter case, an inexpensive tachometer can be mounted at the joint. Notice, however, that trigonometric calculations involving θ_1 are eliminated, which speeds-up the real-time computation.

Another common outcome of the last two examples is that **J** is a diagonal matrix (Even if the sensor axes were to be chosen differently, each row and each column of **J** would have only one entry.) Having a diagonal matrix **J** means that each sensor output controls a single corresponding joint, and consequently complex real-time calculations are avoided. The matrix **J** is diagonal if and only if the joints move the arm end instantaneously in orthogonal directions.

In practice only three types of industrial robots satisfy this condition: Cartesian, cylindrical, and spherical arms. Therefore, the use of these types of robots is more appropriate for end-of-arm sensor-based tasks. The SCARA-type and articulated robots contain two parallel axes for rotation, and therefore their Jacobian is not a diagonal matrix. Nevertheless, from the viewpoint of robot accessibility, manipulators could be ranked as follows: articulated, spherical, cylindrical, and Cartesian. As a consequence of these two features, a reasonable conclusion is that from control viewpoint, a spherical coordinate arm is the most appropriate for 3-D sensor-equipped intelligent robot. The particular application, however, must be considered when a robot is selected.

3 The Jacobian Approximation in Six-Joint Manipulators

In the previous section three-joint robot arms were discussed. However, in order to position and orient the end effector in space, six joints are required. Accordingly we assume that the robot is equipped with a six-component sensor. The sensor is able to produce three components of the required linear velocity of the TCP in three orthogonal axes, and three components of the required rotational rates (i.e., angular velocities) about these axes. The three velocity components create the velocity vector **v** and the three rotational rates comprise the vector **Ω**. These six velocities are the required variables provided by the sensor. The velocities of the three-arm-joints create the vector $\dot{\theta}_a$, and the three velocities of the wrist comprise the vector $\dot{\theta}_w$. These four vectors are related by

$$\begin{bmatrix}\mathbf{v}\\ \mathbf{\Omega}\end{bmatrix}=\mathbf{J}\begin{bmatrix}\dot{\theta}_a\\ \dot{\theta}_w\end{bmatrix} \tag{17}$$

where **J** (the Jacobian in SCS) is a 6×6 matrix. Having obtained **J**, we may find the required commands $\dot{\theta}_i$ by inverting it to obtain

$$\begin{bmatrix}\dot{\theta}_a\\ \dot{\theta}_w\end{bmatrix}=\mathbf{J}^{-1}\begin{bmatrix}\mathbf{v}\\ \mathbf{\Omega}\end{bmatrix} \tag{18}$$

The obvious problem with this method is that an inversion of a 6×6 matrix in real time is required. However, the structure of **J** enables the approximation of the 6×6 matrix inversion by two 3×3 inversions. (Approximations of non 6×6 matrices are also discussed in the examples of this text.)

The matrix **J** can be partitioned into four 3×3 matrices

$$\mathbf{J}=\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{U} & \mathbf{W}\end{bmatrix} \tag{19}$$

where **A** represents the relationship between the arm joint velocities and the end effector translational velocities; **W** represents the relationship between the wrist joint velocities and the angular velocities of the end effector; **B** represents the effect of the wrist joint velocities on the end effector translational velocity, which is obviously small compared with the effect of the arm joints (namely, $\|\mathbf{A}\dot{\theta}_a\| > \|\mathbf{B}\dot{\theta}_w\|$); finally, **U** represents the effect of the arm joint velocities on the orientation of the end effector, which is small in practical motions (namley $\|\mathbf{W}\dot{\theta}_w\| > \|\mathbf{U}\dot{\theta}_a\|$).

The inverse of **J** is a 6×6 matrix consisting of four 3×3 sub-matrices

$$\mathbf{J}^{-1}=\begin{bmatrix}\mathbf{K} & \mathbf{L}\\ \mathbf{M} & \mathbf{N}\end{bmatrix}. \tag{20}$$

where

$$\mathbf{K}=[\mathbf{A}-\mathbf{B}\mathbf{W}^{-1}\mathbf{U}]^{-1}$$

$$\mathbf{M}=-\mathbf{W}^{-1}\mathbf{U}\mathbf{K}$$

$$\mathbf{N}=[\mathbf{W}-\mathbf{U}\mathbf{A}^{-1}\mathbf{B}]^{-1}$$

$$\mathbf{L}=-\mathbf{A}^{-1}\mathbf{B}\mathbf{N}.$$

Since $\|\mathbf{A}\dot{\theta}_a\| > \|\mathbf{B}\dot{\theta}_w\|$ and $\|\mathbf{W}\dot{\theta}_w\| > \|\mathbf{U}\dot{\theta}_a\|$ the matrices **K** and **N** can be approximated by $\mathbf{K}\simeq\mathbf{A}^{-1}$ and $\mathbf{N}\simeq\mathbf{W}^{-1}$, which yields

$$\mathbf{J}^{-1}=\begin{bmatrix}\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{W}^{-1}\\ -\mathbf{W}^{-1}\mathbf{U}\mathbf{A}^{-1} & \mathbf{W}^{-1}\end{bmatrix} \tag{21}$$

which means that the inversion of **J** was reduced to two 3×3 inversions of **A** and **W**. In general, equation (21) can be used to control robots in which the matrices **A** and **W** are non-singular, and it is not limited to 6-DOF robots.

If this or other appoximations (which will be introduced below) are used in equation (18), the joints would not move in the desired velocity, and, in turn, the robot endpoint would not move exactly toward the target. However, since the robot is equipped with a sensor, the error is detected, and a new motion-command toward the target is generated in the next

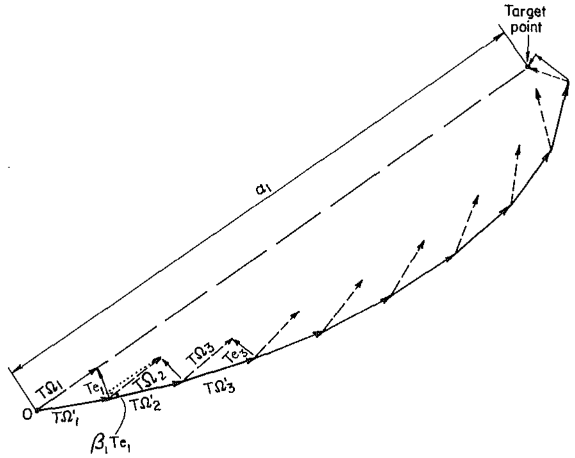

Fig. 3 Trajectory of an approximated motion control algorithm

iteration. This is shown in Fig. 3, where the dashed lines represent the motion-commands and the solid line represents the actual trajectory (subscripts indicate iterations). If certain conditions (discussed below) are satisfied, the robot endpoint will eventually reach the target.

Notice that even when the correct value of J is used in equation (18), an error is generated in practical robot systems which are of sampled-data type. The reason is that J changes continuously, but the sampled-data controller uses a constant J during the sampling period T. Since errors are anyhow generated, approximation methods which enable the reduction of T are justified.

There are robots in which $\mathbf{B} = \mathbf{0}$[9–11]. In this case the *exact* solution is

$$\mathbf{J}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} & \mathbf{0} \\ -\mathbf{W}^{-1}\mathbf{U}\mathbf{A}^{-1} & \mathbf{W}^{-1} \end{bmatrix} \tag{22}$$

The matrix $\mathbf{B}$ is zero if and only if the three rotary axes of the wrist meet at one point and the Jacobian relates the velocities of this point to the joint speeds.

Similarly, an exact solution of $\mathbf{J}^{-1}$ is obtained if $\mathbf{U} = \mathbf{0}$ is substituted in equation (21). However, $\mathbf{U} = \mathbf{0}$ is valid only for a Cartesian robot. (If the arm has rotary joints, their corresponding columns in $\mathbf{U}$ contain at least one non-zero element). As a conclusion of the last two observations we see that $\mathbf{J}$ may be diagonal in a Cartesian robot in which the three rotary axes of the wrist intersect at one point at which the sensor and the TCP are located. This is the ideal structure of a sensor-equipped robot from a motion control viewpoint, but such a robot only rarely meets the application requirements.

4 Motion Control Algorithms

This section proposes approximation algorithms and analyses their convergence conditions.

4.1 Approximation With Orientation Error. Although we observed that $\mathbf{U} = \mathbf{0}$ only for a Cartesian robot, let us assume that $\mathbf{J}^{-1}$ is approximated by substituting $\mathbf{U} = \mathbf{0}$ in equation (21) for any type of robot. Assuming that $|\mathbf{A}| \neq \mathbf{0}$ and $|\mathbf{W}| \neq \mathbf{0}$, this yields

$$\mathbf{J}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}\mathbf{W}^{-1} \\ \mathbf{0} & \mathbf{W}^{-1} \end{bmatrix} \tag{23}$$

If the control program applies this approximation, the corresponding velocity commands to the joints are

$$\dot{\boldsymbol{\theta}}_a = \mathbf{A}^{-1}\mathbf{v} - \mathbf{A}^{-1}\mathbf{B}\mathbf{W}^{-1}\boldsymbol{\Omega} \tag{24}$$

$$\dot{\boldsymbol{\theta}}_w = \mathbf{W}^{-1}\boldsymbol{\Omega} \tag{25}$$

Equations (24) and (25) are denoted henceforth as the 1st motion control algorithm (MCA) or MCA1.

Using the 1st MCA, the joint velocities yields the following tool velocities

$$\mathbf{v}' = \mathbf{A}\dot{\boldsymbol{\theta}}_a + \mathbf{B}\dot{\boldsymbol{\theta}}_w \tag{26}$$

$$\boldsymbol{\Omega}' = \mathbf{U}\dot{\boldsymbol{\theta}}_a + \mathbf{W}\dot{\boldsymbol{\theta}}_w \tag{27}$$

where $\mathbf{v}'$ and $\boldsymbol{\Omega}'$ are the actual velocities of the sensor attached to the arm.

The effect of the approximation is determined by substituting equations (24) and (25) into (26) and (27), which yields

$$\mathbf{v}' = \mathbf{v} \tag{28}$$

$$\boldsymbol{\Omega}' = \mathbf{U}\dot{\boldsymbol{\theta}}_a + \boldsymbol{\Omega} \tag{29}$$

That means that despite the usage of the approximation the exact desired velocity $\mathbf{v}$ is obtained, but it causes an angular velocity error of

$$\mathbf{e} = \boldsymbol{\Omega} - \boldsymbol{\Omega}' = -\mathbf{U}\dot{\boldsymbol{\theta}}_a \tag{30}$$

This error causes an error in the tool orientation. Substituting equation (24) into (30) gives the error as function of the sensor commands

$$\mathbf{e} = -\mathbf{U}\mathbf{A}^{-1}\mathbf{v} + \mathbf{G}\boldsymbol{\Omega} \tag{31}$$

where

$$\mathbf{G} = \mathbf{U}\mathbf{A}^{-1}\mathbf{B}\mathbf{W}^{-1} \tag{32}$$

As a result of this error the manipulator endpoint moves along the required trajectory with the wrong wrist orientation. However, since the manipulator is equipped with a sensor, the error can be detected and corrected. The determination of the conditions under which the manipulator is capable to correct the orientation error and converge to the target with the proper orientation is of extreme importance. Satisfying these conditions allow the motion control algorithm to apply two 3 × 3 matrix inversions (i.e., equation (23)) rather than one 6 × 6 inversion, thereby decreasing the control-program execution time and consequently speeding up the robot response.

4.2 Convergence Conditions. It was shown that despite the use of the approximation, the required endpoint velocity is obtained, and therefore the TCP will reach the desired point. As a consequence, we can assume in the forthcoming analysis without loss of generality, that the TCP is in the desired position but with the wrong orientation. Since position was achieved, there is no need to manipulate the velocity $\mathbf{v}$, namely $\mathbf{v} = \mathbf{0}$. The control system of the robot is a discrete-type system. At each iteration i the desired angular velocity $\boldsymbol{\Omega}(i)$ is given by the sensor, and the corresponding joint velocities are calculated according to equations (24) and (25). The manipulator moves, but as a result of the approximation an error $\mathbf{e}(i)$ is generated at the end of the iteration. Notice that the matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{U}$, $\mathbf{W}$, and $\mathbf{G}$ depend on the joint variables and have different values at each iteration. According to equation (31) with $\mathbf{v} = \mathbf{0}$, the error at the end of the first iteration is

$$\mathbf{e}(1) = \mathbf{G}(0)\boldsymbol{\Omega}(1) \tag{33}$$

where $\mathbf{G}(0)$ represents the value of $\mathbf{G}$ before the first iteration (this value is constant during the iteration) and $\boldsymbol{\Omega}(1)$ is the orientation command during the first iteration. Subsequently, a new command $\boldsymbol{\Omega}(2)$ is generated by the sensor. As seen from Fig. 3 (the dotted line is a shifted vector $T\Omega_1$) the relationship between two successive commands is

$$\boldsymbol{\Omega}(2) = \boldsymbol{\Omega}(1) + \beta_1\mathbf{e}(1) \tag{34}$$

where β_1 is a scaler. The angular motions are depicted in Fig. 3 in a planar configuration where subscripts indicate iterations. The scalar β_1 can be approximated for a large $\alpha(i)$ by

$$\beta_i = \frac{\|\Omega(i)\|T}{\alpha(i) - \|\Omega(i)\|T} \tag{35}$$

where T is the iteration time, and $\alpha(i)$ is the angle about the $\Omega(i)$ axis between the actual and the desired orientations of the tool. Notice that $1 > \beta > 0$ always exists (excluding the last step). At the end of the second iteration the error is

$$e(2) = G(1)\Omega(2) \tag{36}$$

Substitution of equation (34) into (36) yields

$$e(2) = G(1)\Omega(1) + F(1)e(1) \tag{37}$$

where $F(1) = \beta_1 G(1)$.

Similarly, the sensor command at the $(i+1)$ iteration is

$$\mathbf{\Omega}(i+1) = \mathbf{\Omega}(i) + \beta_i \mathbf{e}(i) \tag{38}$$

and the consequent error is

$$\mathbf{e}(i+1) = \mathbf{G}(i)\mathbf{\Omega}(i+1) \tag{39}$$

Substituting equation (38) into (39) yields

$$\mathbf{e}(i+1) = \mathbf{F}(i)\mathbf{e}(i) + \mathbf{G}(i)\mathbf{\Omega}(i) \tag{40}$$

where

$$\mathbf{F}(i) = \beta_i \mathbf{G}(i) \quad \text{for} \quad i = 1, 2, \ldots \tag{41}$$

Equation (40) represents a discrete linear process with initial conditions given by equation (33). In order to test convergence, it is considered as a control system. Stability of this system means that with bounded input $\mathbf{\Omega}$ the error $\mathbf{e}$ is bounded, and with zero input the error $\mathbf{e}$ converges to zero, namely, the manipulator will reach the target point with the correct orientation.

It is known (e.g., [12]) that the linear time-invariant discrete system

$$\mathbf{x}(i+1) = \mathbf{Qx}(i) + \mathbf{RU}(i) \tag{42}$$

is stable if the eigenvalues λ_j of the $p \times p$ matrix $\mathbf{Q}$ satisfy the condition

$$|\lambda_j| < 1 \text{ for } j = 1, 2, \ldots, p \tag{43}$$

Equation (40), however, describes a time-varying process. Since $\beta_i < 1$ the matrix $\mathbf{F}(i)$ in equation (40) can be replaced by $\mathbf{G}(i)$ for stability tests, reducing thereby the time dependence. The elements of $\mathbf{G}(i)$ vary smoothly and slowly within the work volume. Therefore, at each instant the time-varying parameters in equation (40) can be considered as being fixed at the current value and the process can be treated as time invariant. (By using this "freezing time" assumption designers have been very successful in designing autopilots [13].) This implies the following result which states a sufficient condition for the stability of a trajectory.

Result 1. *A sensor-equipped robot using the 1st MCA can reach a target point located within its work volume if the eigenvalues of the matrix* **G** *satisfy*

$$|\lambda_j(i)| < 1 \textit{ for } j = 1, \ldots, k$$

along the trajectory.

In robot manipulators, k (the dimension of **G**) can be 1, 2, or 3. This result might be used to determine an effective work volume in which all the eigenvalues satisfy $|\lambda| < 1$ or to dictate parameter design of the manipulator (e.g., ratio between length of links) to guarantee convergence in the entire work volume. This is demonstrated in the continuation of this paper.

An angular convergence trajectory to the target point is depicted in Fig. 3. The manipulator endpoint trajectory in this case is a straight line to the target, and the orientation corrections are taking place during the motion and after reaching the point. This situation is reversed in the next algorithm, in which the correct orientation is achieved, but the endpoint deviates from the straight-line trajectory.

4.3 Approximation with Velocity Error. An alternative approximation algorithm might be the one given by equation (22), namely

$$\dot{\theta}_a = \mathbf{A}^{-1}\mathbf{v} \tag{44}$$

$$\dot{\theta}_w = -\mathbf{W}^{-1}\mathbf{UA}^{-1}\mathbf{v} + \mathbf{W}^{-1}\mathbf{\Omega} \tag{45}$$

Equations (44) and (45) are denoted as the 2nd motion control algorithm or MCA2.

The effect of this approximation is determined by substituting equations (44) and (45) into (26) and (27), which yields

$$\mathbf{v}' = \mathbf{v} + \mathbf{B}\dot{\theta}_w \tag{46}$$

$$\mathbf{\Omega}' = \mathbf{\Omega} \tag{47}$$

This means that when using this algorithm, the exact desired orientation is obtained, and the effect of the approximation is a velocity error

$$\mathbf{e}_v = \mathbf{v} - \mathbf{v}' = -\mathbf{B}\dot{\theta}_w \tag{48}$$

Notice that Fig. 3 can apply to this approximation if instead of $\mathbf{\Omega}$ and α, the linear velocity $\mathbf{v}$ and a distance d are written in the drawing.

Substituting equation (45) into (48) gives the velocity error as a function of the sensor commands

$$\mathbf{e}_v = \mathbf{Hv} - \mathbf{BW}^{-1}\Omega \tag{49}$$

where **H** is an $m \times m$ matrix defined by

$$\mathbf{H} = \mathbf{BW}^{-1}\mathbf{UA}^{-1} \tag{50}$$

Sufficient convergence conditions of this algorithm can be summarized in the following result.

Result 2. *A sensor-equipped robot using the 2nd MCA can reach a target point located within its work volume if the eigenvalues of the matrix* **H** *satisfy*

$$|\lambda_j(i)| < 1 \textit{ for } j = 1, \ldots, m$$

along the trajectory.

The proof of this result follows along the lines of the proof of Result 1, with the assumption that $\mathbf{\Omega} = \mathbf{0}$ instead of $\mathbf{v} = \mathbf{0}$.

4.4 General Approximated Motion Algorithm. The matrix in equation (21) can also be used as a motion control algorithm:

$$\dot{\theta}_a = \mathbf{A}^{-1}\mathbf{v} - \mathbf{A}^{-1}\mathbf{BW}^{-1}\mathbf{\Omega} \tag{51}$$

$$\dot{\theta}_w = -\mathbf{W}^{-1}\mathbf{UA}^{-1}\mathbf{v} + \mathbf{W}^{-1}\mathbf{\Omega} \tag{52}$$

Equations (51) and (52) are denoted as the 3rd motion control algorithm or MCA3. This approximation algorithm causes the following errors:

$$\mathbf{e}_v = \mathbf{v} - \mathbf{v}' = \mathbf{Hv} \tag{53}$$

and

$$e_w = \mathbf{\Omega} - \mathbf{\Omega}^1 = \mathbf{G\Omega} \tag{54}$$

Comparing equations (53) and (54) with (49) and (31), respectively, one can see that the errors in this case are smaller, but the control algorithm requires more algebraic operations, which, in turn, result in longer sampling periods.

Following the proof of Result 1, one might think that the convergence conditions in this case require that the eigenvalues of both matrices **G** and **H** satisfy $|\lambda| < 1$. However,

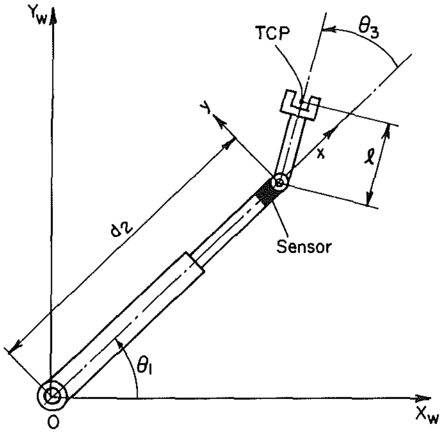

Fig. 4 Planar polar manipulator

testing the eigenvalues of only one of these matrices is sufficient, as is explained below.

Let us define: $\mathbf{P} = \mathbf{UA}^{-1}$ and $\mathbf{Q} = \mathbf{BW}^{-1}$. From equations (32) and (50): $\mathbf{G} = \mathbf{PQ}$ and $\mathbf{H} = \mathbf{QP}$. The matrices $\mathbf{P}$ and $\mathbf{Q}$ might be either square matrices with equal dimensions (3×3 in a 6-DOF manipulator), or matrices in which the number of rows in $\mathbf{P}$ equals the number of columns in $\mathbf{Q}$, and vice versa. For example, if the dimensions of $\mathbf{J}$ in equation (19) are $n \times n$ those of A are $m \times m$, and those of $\mathbf{W}$ are $k \times k$, then the dimensions of $\mathbf{P}$ are $k \times m$ and those of $\mathbf{Q}$ are $m \times k$ (where $n = m + k$). (E.G., in a SCARA-type manipulator $n = 4$, $m = 3$, and $k = 1$.) The resulting $\mathbf{G}$ is a $k \times k$ matrix and $\mathbf{H}$ is an $m \times m$ matrix. If $m \geq k$, then testing the eigenvalues of $\mathbf{G}$ is sufficient, and if $m \leq k$, then the test of $\mathbf{H}$ is sufficient; the excess eigenvalues (i.e., $|m-k|$) are zero [14], therefore satisfying the condition $|\lambda| < 1$.

The convergence conditions in this case are the following.

Result 3. *A sensor-equipped robot using the 3rd MCA can reach a target point within its work volume if all the eigenvalues of either the matrix* $\mathbf{G}$ *or* $\mathbf{H}$ *satisfy* $|\lambda(i)| < 1$ *along the trajectory.*

Based on the discussion above and on Results 1, 2, and 3, the following theorem states the sufficient condition for convergence.

Theorem. *A sensor-equipped robot using the 1st, 2nd, or 3rd MCA can reach a target point within its work volume if the eigenvalues of either the matrix* $\mathbf{G}$ *or* $\mathbf{H}$ *satisfy* $|\lambda(I)| < 1$ *along the trajectory.*

According to this Theorem, if $\mathbf{G}$ and $\mathbf{H}$ have different dimensions then the test of the matrix with the smaller dimensions is sufficient regardless of the algorithm which is used. The selection of the appropriate MCA depends on the particular application of the robot and on the structure of the manipulator as well as on the sensor sensitivity to velocity and orientation deflections.

5 Examples

The Theorem stated above might be used in determining the effective work volume or in the design of manipulators as is demonstrated next.

5.1 Planar Polar Robot. As an illustrative example consider the 3-DOF planar polar robot shown in Fig. 4. A sensor is attached to the second link, and the equations are written in the sensor coordinate system. It should be noticed that the location of the coordinate system in which the equations are written does not change the convergence conditions. The Jacobian which relates the TCP velocities (given in the sensor coordinates) to the joint speeds is

$$\begin{bmatrix} v_x \\ v_y \\ \Omega_z \end{bmatrix} = \left[\begin{array}{cc:c} -lS_3 & 1 & -lS_3 \\ d_2 + lC_3 & 0 & lC_3 \\ \hdashline 1 & 0 & 1 \end{array}\right] \begin{bmatrix} \dot{\theta}_1 \\ \dot{d}_2 \\ \dot{\theta}_3 \end{bmatrix} \tag{55}$$

where $S_3 = \sin\theta_3$ and $C_3 = \cos\theta_3$. Notice that the Jacobian is independent of θ_1.

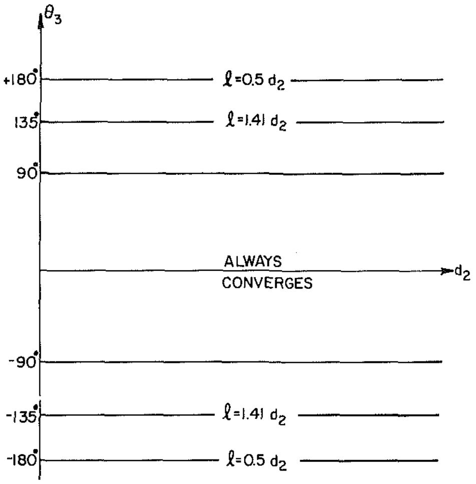

Fig. 5 Convergence regions of planar polar robot

The Jacobian in equation (55) can be partitioned into four submatrices (**A**, **B**, **U**, and **W**) shown with the dashed line (see equation (19)). In order to find the convergence condition of MCA1 (equations (24) and (25)), the eigenvalues of the matrix **G** have to be found. The matrix **G** is defined in equation (32), and in this case is a scalar

$$G = \frac{lC_3}{d_2 + lC_3}$$

Since G is a scalar we have $G = \lambda$.

In order to reach a target point while using the 1st MCA approximation, the following condition must be satisfied

$$|\lambda| = \left|\frac{lC_3}{d_2 + lC_3}\right| < 1 \tag{56}$$

For $d_2 > 0$ two conditions are derived from equation (56)

$$C_3 > -\frac{d_2}{l} \tag{57}$$

and

$$C_3 > -\frac{d_2}{2l} \tag{58}$$

The more restrictive convergence condition in this case is given by equation (58) and is illustrated in Fig. 5. Exactly the same convergence condition must be satisfied if the 2nd MCA or the 3rd MCA are used.

Simulation results of a polar robot using the 2nd MCA are shown in Fig. 6. The length of the second link is $l = 0.2$. The initial location in Fig. 6(*a*) is $x = 0.3$, $y = 0.173$, and orientation = 60 deg. The target location is $x = 0.4$, $y = 0.25$, and orientation = 60 deg. At the initial location $\lambda = 0.50$, and at the target location $\lambda = 0.45$. The trajectory converges to the target point as shown in Fig. 6(*a*). In Fig. 6(*b*) the initial location is $x = -0.15$, $y = 0.234$, and orientation = 150 deg. The target location is $x = 0.25$, $y = 0.35$, and orientation = 150 deg. At the initial location $\lambda = -1.5$, and at the final location $\lambda = -0.9$. The convergence condition is not fulfilled and the trajectory diverges as seen in Fig. 6(*b*). Simulation results

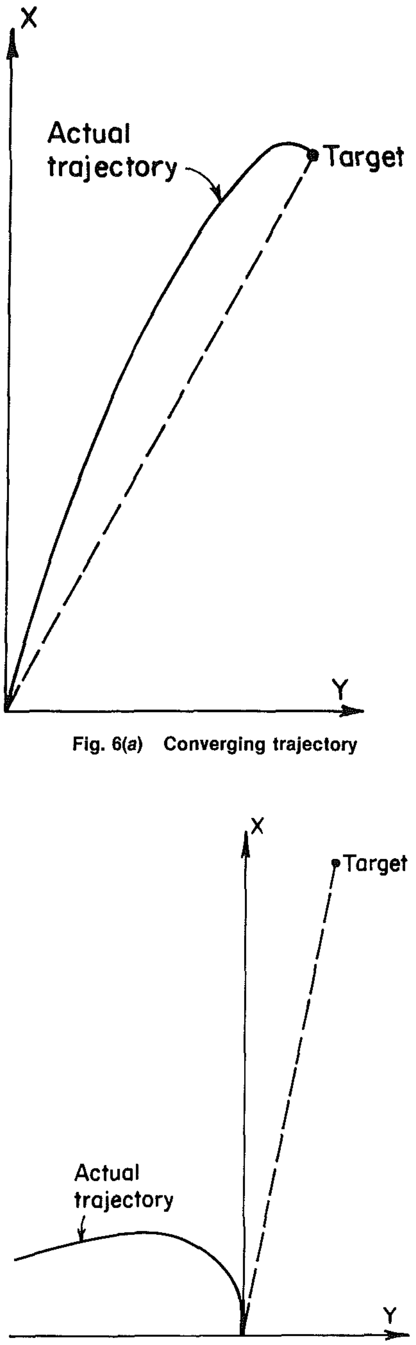

Fig. 6(*a*) Converging trajectory

Fig. 6(*b*) Diverging trajectory

Fig. 6 Simulation results of a planar polar robot controlled by the 2nd MCA

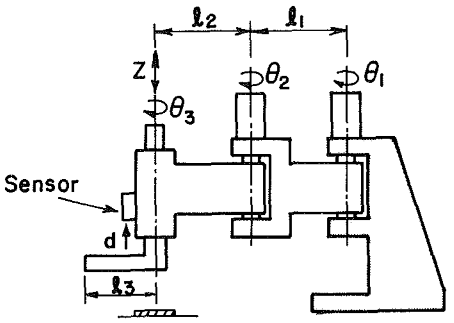

Fig. 7 SCARA-type robot

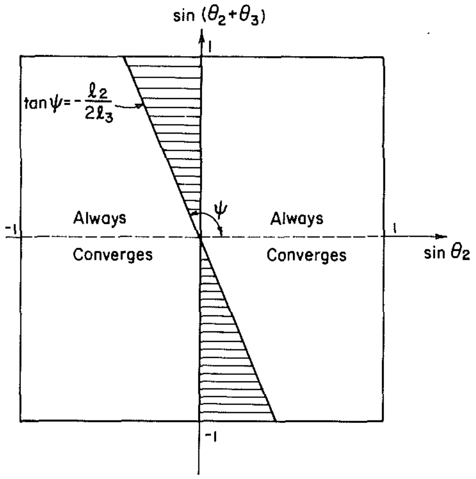

Fig. 8 SCARA robot convergence regions

show that if $|\lambda| < 1$ at the initial and the target points the trajectory converges, and it is not necessary to check in advance the values of λ along the trajectory.

Notice from equation (58) that in the polar robot, if the designer guarantees that $d_2/l > 2$, then the robot converges to the target point in the entire work volume of the manipulator despite of using the approximation motion algorithms.

5.2 SCARA Robot. The implementation of the approximation algorithms can be demonstrated on a SCARA-type manipulator (see Fig. 7). The sensor is attached to the second link and the Jacobian is written in the sensor coordinate system as follows

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \Omega_z \end{bmatrix} = \left[\begin{array}{ccc|c} l_1S_2 - l_3S_3 & -l_3S_3 & 0 & -l_3S_3 \\ l_1C_2 + l_2 + l_3C_3 & l_2 + l_3C_3 & 0 & l_3C_3 \\ 0 & 0 & 1 & 0 \\ \hline 1 & 1 & 0 & 1 \end{array}\right] \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{d} \\ \dot{\theta}_3 \end{bmatrix} \tag{59}$$

where $S_2 = \sin\theta_2$, $S_3 = \sin\theta_3$, $C_2 = \cos\theta_2$, and $C_3 = \cos\theta_3$. Also in this case **G** is a scalar given by

$$G = \frac{l_3S_{23}}{l_2S_2 + l_3S_{23}} \tag{60}$$

Where $S_{23} = \sin(\theta_2 + \theta_3)$
The convergence condition in this case is

$$\left| \frac{l_3S_{23}}{l_2S_2 + l_3S_{23}} \right| < 1 \tag{61}$$

which yields the following two conditions:

$$\frac{S_{23}}{S_2} > -\frac{l_2}{l_3} \tag{62}$$

and

$$\frac{S_{23}}{S_2} > -\frac{l_2}{2l_3} \tag{63}$$

The more restrictive condition is given by equation (63) and is illustrated in Fig. 8. Notice that if S_2 and S_{23} are both positive or both negative, the convergence condition is always satisfied.

Two simulation results of a SCARA robot using the 2nd MCA are shown in Fig. 9. The link lengths in this case are $l_1 = l_2 = l_3 = 0.4$. The initial location in Fig. 9(*a*) is $x = 1.10$, $y = 0.155$, and the orientation is -21.3 deg. The target point is x

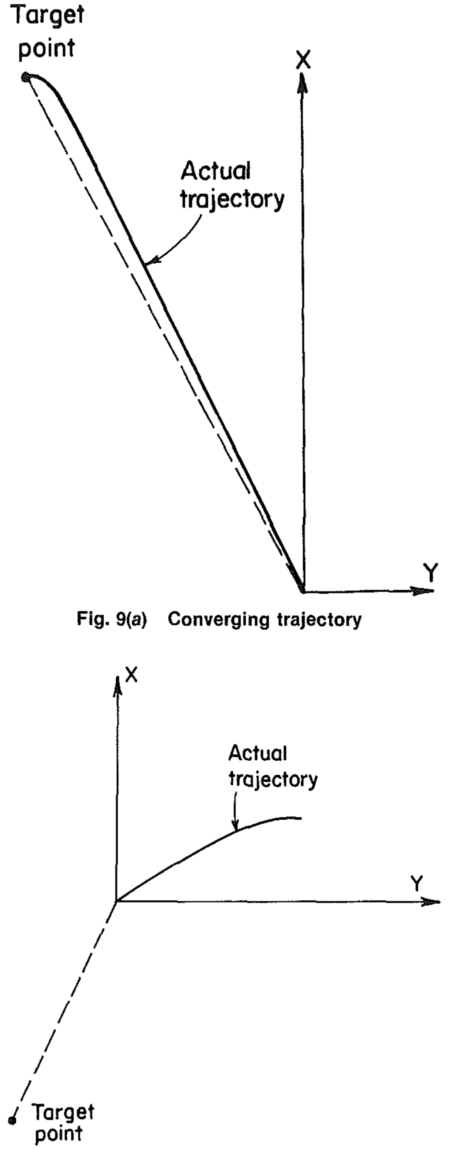

Fig. 9(*a*) Converging trajectory

Fig. 9(*b*) Diverging trajectory

Fig. 9 Simulation results of a SCARA robot controlled by the 2nd MCA

= 1.116, y = 0.124, and the orientation is -21.3 deg. At the initial location $\lambda = 0.695$ and at the target location $\lambda = 0.718$. The trajectory converges to the target point as shown in Fig. 9(*a*). In Fig. 9(*b*) the initial location is $x = 1.0, y = -0.037$, and the orientation is -50 deg. The target location is $x = 0.95, y = -0.06$, and the orientation is -50 deg. At the initial location $\lambda = 1.62$, and at the final location $\lambda = 1.70$. The convergence condition is not fulfilled and the trajectory diverges as shown in Fig. 9(*b*).

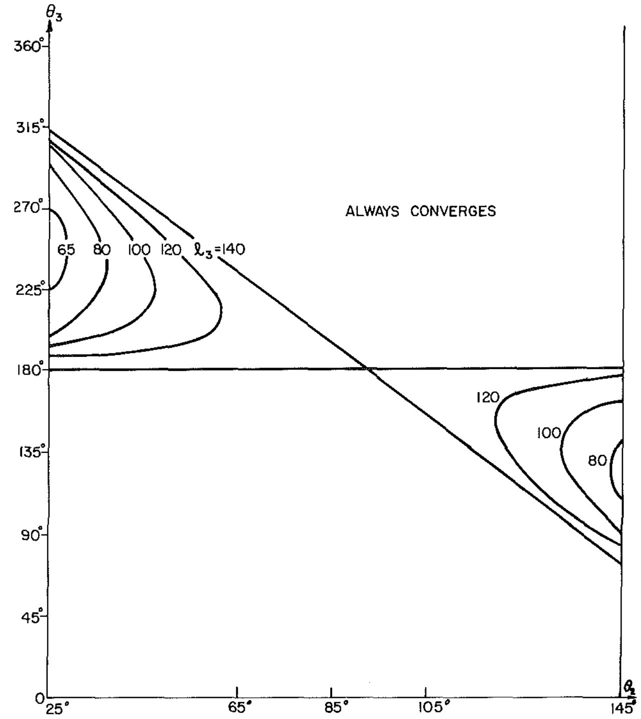

Fig. 10 HIRATA ARH-300 robot convergence regions

It is difficult to obtain a design rule which will guarantee convergence in the entire work volume of a SCARA type manipulator. In practical manipulators, however, the range of θ_2 is limited. The minimal θ_2 dictates a maximum value of l_3/l_2 for which convergence always exists despite of using the approximation algorithms. In the Hirata ARH-300, for example, θ_2 can vary in the limits 25 deg $< \theta_2 <$ 145 deg. The second link length is $l_2 = 275$ mm. As a consequence if $l_3 <$ 60 mm then the robot always converges to any target point in the entire work volume of the manipulator despite of using the approximate algorithms. With the increasing of l_3, the region in which trajectories might diverge increases as seen in Fig. 10. Therefore it is recommended to work with a tool length $l_3 <$ 60 mm when controlling the robot with one of the proposed MCA algorithms. In a case that l_3 must be greater and convergence is required in the entire work volume, the range of θ_2 should be reduced with the aid of available limit-switches.

5.3 Six-DOF Robot. As an example of a 6-DOF robot consider a spherical robot with a wrist having three revolute joint axes intersecting at one point (P) as shown in Fig. 11. Assume that the robot is equipped with a sensor located at point P and is capable of measuring the required velocity and angular velocity of the TCP.

The Jacobian which relates the TCP velocity to the joint variable speeds, given in the sensor coordinate system, is

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix}_S = \left[\begin{array}{ccc:ccc} d_3C_2 + lC_2C_5 - lS_2C_4S_5 & 0 & 0 & lC_4S_5 & lS_4C_5 & 0 \\ -lS_2S_4S_5 & d_3 - lC_5 & 0 & lS_4S_5 & -lC_4C_5 & 0 \\ -lC_2S_4S_5 & -lC_4 - S_5 & 1 & 0 & -lS_5 & 0 \\ \hdashline 0 & 1 & 0 & 0 & C_4 & S_4S_5 \\ C_2 & 0 & 0 & 0 & S_4 & -C_4S_5 \\ -S_2 & 0 & 0 & 1 & 0 & C_5 \end{array}\right] \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{d}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \\ \dot{\theta}_6 \end{bmatrix} \quad (70)$$

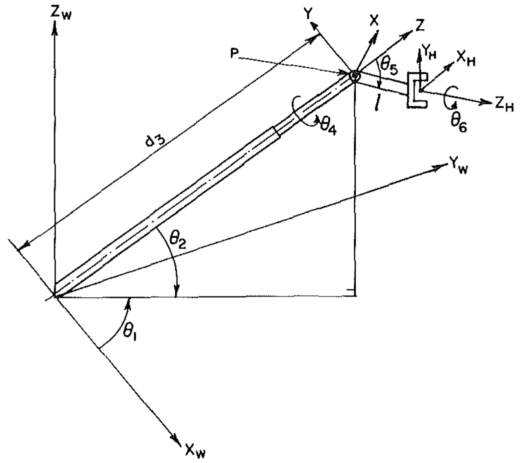

Fig. 11 Six-DOF spherical robot

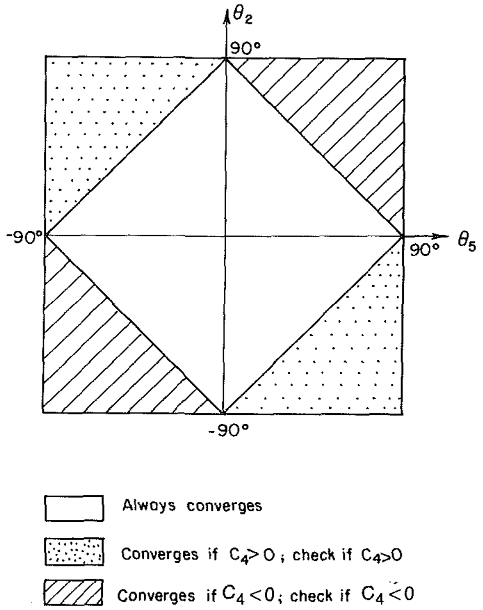

Fig. 12 Convergence regions of a six-DOF spherical robot

The Jacobian may be partitioned into four 3×3 matrices (**A**, **B**, **U**, and **W**) defined in equation (19) and shown with the dashed line in equation (70).

By using these definitions, the matrix **G** is derived from equation (32)

$$\mathbf{G}=\frac{l}{\det(A)}\begin{bmatrix} -d_3C_2C_5-lC_2C_5^2+lS_2C_4S_5C_5 & lS_2S_4S_5C_5 & d_3C_2S_4S_5+lC_2S_4S_5C_5 \\ 0 & d_3C_2C_5-lC_2C_5^2 & d_3C_2C_4S_5-lC_2C_4S_5C_5 \\ 0 & -d_3S_2C_5+lS_2C_5^2 & d_3S_2C_4S_5+lS_2C_4S_5C_5 \end{bmatrix} \qquad (71)$$

where $\det(A) = d_3^2C_2 - d_3lS_2C_4S_5 - l^2C_2C_5^2 + l^2S_2C_4S_5C_5$.

The convergence conditions state that the eigenvalues of **G** satisfy $|\lambda_j| < 1$ ($j = 1, 2, 3$). After some algebraic manipulations and assuming that $d_3 > l$ and $\cos\theta_2 > 0$, the convergence conditions become

$$C_2C_5-S_2C_4S_5>0 \qquad (72)$$

and

$$\frac{d_3}{2l}>C_5 \qquad (73)$$

One can design a manipulator with $d_3 > 2l$, and then the restrictive condition becomes equation (72). Assuming that θ_5 is limited by -90 deg $<\theta_5<90$ deg, convergence regions in the robot work volume can be determined from equation (72) and are shown in Fig. 12. At least within 50 percent of the work volume convergence always exists. However, since in most practical jobs C_4 will not contain values of $+1$ and -1 along the same trajectory, the effective work volume is much larger.

6 Conclusions

Kinematic algorithms have been developed for the control of a robot equipped with a sensor located on its arm. It has been shown that when computations are performed in the sensor coordinate system, the algorithms become simpler and one joint variable is eliminated from the Jacobian matrix. This permits the removal of the respective encoder from the robot and thereby reducing its cost.

From control viewpoint, the spherical manipulator has been found as the most appropriate structure for 3-DOF sensor-equipped robots. In the spherical robot a complete decoupling of the joint command is achieved, which facilitate the control of this robot.

Simplification of the motion control algorithms (and a consequent improved speed) of sensor-equipped robots can be obtained when approximations are used to solve the inverse kinematics. The subsequent inaccuracies in the trajectories can be corrected by the sensor during the robot motion toward the target point. Accordingly, three motion control algorithms are proposed. However, error analysis reveals that convergence to the target point is not always guaranteed. A Theorem which states the stability conditions of a trajectory has been found. Robot parameters (e.g., tool length or range of joint angles) can be selected to ensure stable convergence to any target point when controlling the robot with the simple approximation algorithms.

Acknowledgment

The work described in this paper was conducted as part of a project sponsored by Elron Inc., Haifa, Israel, with Dr. Zohar Ophir as project monitor. The authors gratefully acknowledge this support and the helpful advice of Dr. Ophir.

References

1 Johnson, T. L., "Feedback Control," Chapter 3 of *Robot Motion*, MIT Press, Cambridge, 1982, p. 145.

2 Whitney, D. E., "The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators," ASME Journal of Dynamic Systems, Measurement, and Control, Dec. 1972, pp. 303-309.

3 Pieper, D. L., "The Kinematics of Manipulators under Computer Control," Ph.D. thesis, Stanford Univ., 1969.

4 Loughlin D., et al., "Line Edge and Contour Following with Eye-in-Hand Vision System," *14th Symp. on Industrial Robots*, Sweden, Oct. 1984, pp. 553-559.

5 Shoham, M., Fainman, Y., and Lenz, E., "An Optical Sensor for Real-Time Positioning, Tracking and Teaching of Industrial Robots," *IEEE trans. on Industrial Electronics*, Vol. IE-31, No. 2, May 1984, pp. 159-163.

6 Whitney, D. E., "Force Feedback Control of Manipulator Fine Motions," ASME Journal of Dynamics Systems, Measurement, and Control, Vol. 99, No. 2, June 1977, pp. 91-97.

7 Brady, M., et al., *Robot Motion*, MIT Press, Cambridge, 1982.

8 Koren, Y., *Robotics for Engineers*, McGraw-Hill, New York, 1985.

9 Walderon, K. J., Wang, S. L., and Bolin, S. J., "A Study of the Jacobian Matrix of Serial Manipulators," ASME *Journal of Mechanical Design*, Vol. 109, 1984.

10 Renaud, M., "Contribution a la Modelisation et a la Commande Dynamique des Robots Manipulators," Ph.D. thesis, Paul Sabatier de Toulouse Univ., 1980.

11 Featherstone, R., "Position and Velocity Transformation Between Robot and Effector Coordinates and Joint Angles," *International J. of Robotic Research*, Vol. 2, No. 2, 1983.

12 Kuo, B. C., *Digital Control Systems*, SRL Publishing Co., Champaign, Ill., 1977.

13 Hsu, J. C., and Meyer, A. U., *Modern Control Principles and Applications*, McGraw-Hill, New York, 1968, p. 160.

14 Noble, B., and Daniel, J. W., *Applied Linear Algebra*, Prentice-Hall, 1977, p. 298.