RSD-16-87

Manipulator Kinematics and the Epsilon Algebra

Michael W. Walker

Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, Michigan 48109

August 1987

Center for Research on Integrated Manufacturing

Robot Systems Division
College of Engineering
The University of Michigan
Ann Arbor, Michigan 48109-2110

TABLE OF CONTENTS

Manipulator Kinematics and The Epsilon Algebra

Michael W. Walker

Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, Michigan

Abstract: A new algebra is defined for use in problems of manipulator kinematics. With this algebra the solution to the inverse kinematics problem can be solved for any time derivative of the joint position using the same program by simply changing the order of the algebra. An example computer program written in Ada is used for illustration.

1. Introduction

The position of a manipulator can be expressed either in joint coordinates or in Cartesian coordinates. Manipulator task are more easily specified in Cartesian coordinates. However, controllers usually control the positions of the joints. Therefore, the transformation from one coordinate system to another is an important task. We denote these transformations with the following equations.

$$y = f(\theta) \tag{1}$$

and the inverse function:

$$\theta = g(y) = f^{-1}(y) \tag{2}$$

In the study of manipulator kinematics y would be a vector that represents the position and orientation of the end-effector and θ would be a vector of joint positions.

Equation 1 is a statement of the forward kinematics problem of manipulators. That is, given the joint positions $\boldsymbol{\theta}$ calculate the position and orientation of the end-effector, $\mathbf{y}$. Equation 2 is a statement of the inverse kinematics problem of manipulators. That is, given the position and orientation of the end-effector, $\mathbf{y}$, calculate the joint positions $\boldsymbol{\theta}$.

The forward kinematics problem has a simple solution. The procedure is to start at the base of the manipulator and successively compute the position and orientation of each link out to the end-effector. Homogeneous transforms [1] are often used to specify the position and orientation of each link.

The inverse kinematics problem is not as simple. If fact a closed form solution to equation 2 for the general case is not known at this time. One of the fundamental results was given by Pieper [2] who showed that any manipulator in which the wrist axes intersect has a closed form solution. Most manipulators today are designed to satisfy this requirment.

Resolved rate control is another problem in kinematics. In this problem one must determine the velocity of the joints given the velocity of the end-effector. From equations 1 and 2 we see that:

$$\dot{\mathbf{y}} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}} \dot{\boldsymbol{\theta}} \tag{3}$$

and

$$\dot{\boldsymbol{\theta}} = \frac{\partial \mathbf{g}}{\partial \mathbf{y}} \dot{\mathbf{y}} = \left(\frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}}\right)^{-1} \dot{\mathbf{y}} \tag{4}$$

Again, the forward problem, equation 3, is very easy to solve, whereas the inverse problem, equation 4 is more difficult.

Whitney [3,4] presented the formulation of this problem. The joint velocities were obtained by solving the linear equation 3 rather than using equation 4 directly. The advantage is the closed form solution of equation 2 is not needed.

If the wrist axes intersect, then a closed form solution of equation 2 can be found. Paul [5] has shown how to obtain an expression for $\partial g/\partial y$ by directly evaluating the partial derivatives from the solution of the inverse kinematics problem obtained from equation 2. Equation 4 can then be used to obtain the joint velocities.

Featherstone [6] presented an efficient solution for the joint velocities when the wrist axes intersect. In this case, it is possible to determine the linear velocity of the point of intersection directly from the linear and angular velocity of the end-effector. The first three joint velocities can be obtained from the linear velocity of the point where the wrist axes intersect. Knowing the velocities of the first three joints the velocity of the three wrist axes can be obtained to get the correct angular velocity of the end-effector.

The problem of resolved acceleration [7] can also be greatly simplified if the wrist axes intersect. In this problem one determines the acceleration of the joint positions given the position and velocity of the joints and the position, velocity and acceleration of the end-effector. From equations 3 and 4 we see that:

$$\ddot{\mathbf{y}} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}} \ddot{\boldsymbol{\theta}} + \frac{d}{dt}\left[\frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}}\right]\dot{\boldsymbol{\theta}} \tag{5}$$

and

$$\ddot{\boldsymbol{\theta}} = \frac{\partial \mathbf{g}}{\partial \mathbf{y}} \ddot{\mathbf{y}} + \frac{d}{dt}\left[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}\right]\dot{\mathbf{y}} = \left(\frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}}\right)^{-1}\left[\ddot{\mathbf{y}} - \frac{d}{dt}\left[\frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}}\right]\dot{\boldsymbol{\theta}}\right] \tag{6}$$

Hollerbach and Sahar [8] presented the wrist partitioned resolved acceleration solution to this problem which is an extension Featherstones method.

In this paper an algebra is presented which in effect combines the results of Paul, Featherstone, Hollerbach and Sahar. With this algebra one can program the solution to the kinematics in position problems, Eq 1 and 2, and use this same program to solve for the kinematics problems in velocity, Eq 3 and 4, and acceleration, Eq 5 and 6. In fact, the inverse kinematics part of the program can be used to obtain any order time derivative of the joint positions.

We begin by introducting the algebra in the next section. We show some of the properties of the algebra and then define functions of the elements of the algebra. Next, the solution to the forward and inverse kinematics problem is presented. Then, the use of the Ada programming language is discussed. Finally, we conclude the paper by making some observations on the results.

2. The Algebra

In this section we define the algebra used. We begin by discussing some of the properties of the algebra and finish with a discussion of functions.

2.1. Properties of the ε-Algebra

The elements in the ε-algebra are ordered sets of n+1 real numbers. We call these ε-numbers and the order of the algebra is the integer n. Thus, we will denote the algebra by stating that it is an $\varepsilon^{(n)}$-algebra if the elements are ordered sets of n+1 real numbers or $\varepsilon^{(n)}$-numbers. We could denote these $\varepsilon^{(n)}$-numbers by listing their elements. For example, an $\varepsilon^{(2)}$-number is $\hat{x}$=(4,2 ,-2). However, the following notation

is more convenient.

$$\hat{x}=\varepsilon_0^{(2)}4 + \varepsilon_1^{(2)}2 - \varepsilon_2^{(2)}2$$

Thus, the order of the real numbers are denoted by the subscript on the epsilon and order of the algebra is denoted by the superscript. Symbols having a hat over them denote ε-numbers.

The operations of addition and multiplication are now easily defined. Addition of two ε-numbers are accomplished by adding the corresponding real number components. For example, let $\hat{x}$ and $\hat{y}$ be $\varepsilon^{(2)}$-numbers.

$$\hat{x}=\varepsilon_0^{(2)}4 + \varepsilon_1^{(2)}2 - \varepsilon_2^{(2)}2$$

and

$$\hat{y}=\varepsilon_0^{(2)}3 + \varepsilon_1^{(2)}1 + \varepsilon_2^{(2)}2$$

Then,

$$\hat{x}+\hat{y}=\varepsilon_0^{(2)}(4+3) + \varepsilon_1^{(2)}(2+1) + \varepsilon_2^{(2)}(-2+2)=\varepsilon_0^{(2)}7 + \varepsilon_1^{(2)}3 + \varepsilon_2^{(2)}0$$

Clearly, the ε-algebra satisfies the commutative and associative laws of addition, there exist a zero, and each ε-number has an additive inverse.

To multiply two ε-numbers together, we simply multiply them together as if the ε's were ordinary real numbers and then reduce products of ε's until a single ε or zero results. Reduction of the ε's is performed using the following formula.

$$\varepsilon_i^{(n)}\varepsilon_j^{(n)} = \begin{cases} \dfrac{(i+j)!}{i!j!}\,\varepsilon_{i+j}^{(n)} & \text{if } i+j \le n \\ 0 & \textit{otherwise} \end{cases}$$

For example, multiplying the $\hat{x}$ and $\hat{y}$ defined above gives:

$$\hat{x}*\hat{y} = (\varepsilon_0^{(2)}4 + \varepsilon_1^{(2)}2 - \varepsilon_2^{(2)}2)(\varepsilon_0^{(2)}3 + \varepsilon_1^{(2)}1 + \varepsilon_2^{(2)}2)$$

$$= (\varepsilon_0^{(2)}\varepsilon_0^{(2)}12 + \varepsilon_0^{(2)}\varepsilon_1^{(2)}4 + \varepsilon_0^{(2)}\varepsilon_2^{(2)}8)$$
$$+ (\varepsilon_1^{(2)}\varepsilon_0^{(2)}6 + \varepsilon_1^{(2)}\varepsilon_1^{(2)}2 + \varepsilon_1^{(2)}\varepsilon_2^{(2)}4)$$
$$- (\varepsilon_2^{(2)}\varepsilon_0^{(2)}6 + \varepsilon_2^{(2)}\varepsilon_1^{(2)}2 + \varepsilon_2^{(2)}\varepsilon_2^{(2)}4)$$

Using the reduction formula gives:

$$\hat{x}*\hat{y} = (\varepsilon_0^{(2)}12 + \varepsilon_1^{(2)}4 + \varepsilon_2^{(2)}8)$$
$$+ (\varepsilon_1^{(2)}6 + \varepsilon_2^{(2)}4 + 0)$$
$$- (\varepsilon_2^{(2)}6 + 0 + 0)$$
$$= (\varepsilon_0^{(2)}12 + \varepsilon_1^{(2)}10 + \varepsilon_2^{(2)}6)$$

To show that the associative law for multiplication holds, $(\hat{x}\hat{y})\hat{z}=\hat{x}(\hat{y}\hat{z})$, we note that the product of the three ε-numbers will result in a sum of products of three ε's. Thus, if one of the terms of the product on the left is of the form, $(\varepsilon_i^{(n)}\varepsilon_j^{(n)})\varepsilon_k^{(n)}$ the corresponding term on the right side of the equation will be of the form, $\varepsilon_i^{(n)}(\varepsilon_j^{(n)}\varepsilon_k^{(n)})$. Therefore, to show that the associative law holds for multiplication we only need to show that:

$$(\varepsilon_i^{(n)}\varepsilon_j^{(n)})\varepsilon_k^{(n)} = \varepsilon_i^{(n)}(\varepsilon_j^{(n)}\varepsilon_k^{(n)})$$

If $i+j+k>n$ then this equation is true since both sides are equal to zero. If $i+j+k\leq n$ then:

$$(\varepsilon_i^{(n)}\varepsilon_j^{(n)})\varepsilon_k^{(n)} = (\frac{(i+j)!}{i!j!}\varepsilon_{i+j}^{(n)})\varepsilon_k^{(n)}$$
$$= \frac{(i+j+k)!}{i!j!k!}\varepsilon_{i+j+k}^{(n)}$$

Similarly,

$$\varepsilon_i^{(n)}(\varepsilon_j^{(n)}\varepsilon_k^{(n)}) = \varepsilon_i^{(n)}(\frac{(j+k)!}{j!k!}\varepsilon_{j+k}^{(n)})$$

$$= \frac{(i+j+k)!}{i!j!k!}\,\varepsilon_{i+j+k}^{(n)}$$

Thus,

$$(\varepsilon_i^{(n)}\varepsilon_j^{(n)})\varepsilon_k^{(n)} = \varepsilon_i^{(n)}(\varepsilon_j^{(n)}\varepsilon_k^{(n)})$$

Therefore, the associative law for multiplication holds.

To show that the distributive law holds, $(\hat{a}(\hat{b}+\hat{c}) = \hat{a}\hat{b}+\hat{a}\hat{c}$, and $(\hat{b}+\hat{c})\hat{a} = \hat{b}\hat{a}+\hat{c}\hat{a})$, for ε-numbers we express the product of two $\varepsilon^{(n)}$-numbers, $\hat{x}$ and $\hat{y}$, in a different form:

$$\hat{x}*\hat{y} = \sum_{j=0}^{n}\sum_{k=0}^{n}\varepsilon_j^{(n)}\varepsilon_k^{(n)}(x_j y_k)$$

$$= \sum_{j=0}^{n}\sum_{k=0}^{n-j}\frac{(j+k)!}{j!k!}\,\varepsilon_{j+k}^{(n)}(x_j y_k)$$

Making a change of variables, $i=j+k$ gives:

$$\hat{x}*\hat{y} = \sum_{j=0}^{n}\sum_{i=j}^{n}\begin{bmatrix} i \\ j \end{bmatrix}\varepsilon_i^{(n)}(x_j y_{i-j})$$

Changing the order of summation gives:

$$\hat{x}*\hat{y} = \sum_{i=0}^{n}\varepsilon_i^{(n)}\Big(\sum_{j=0}^{i}\begin{bmatrix} i \\ j \end{bmatrix}(x_j y_{i-j})\Big) \qquad (7)$$

where $\begin{bmatrix} i \\ j \end{bmatrix}$ is the binomial coeficient. If we let $\hat{x}=\hat{a}$ and $\hat{y}=\hat{b}+\hat{c}$ then it is easy to show that the distributive law holds using this formula for products. Note, that 1 is the unity [1] ε-number.

Equation 7 is also useful to show that the commutitive law for multiplication, $\hat{x}*\hat{y} = \hat{y}*\hat{x}$, is satisfied. If when deriving equation 7 we had substituted $j=i-k$ instead

[1] For notational convienance we denote ε-numbers of the form $\varepsilon_0^{(n)}a+\varepsilon_1^{(n)}0+..+\varepsilon_n^{(n)}0$ where a is a real number simply as a.

of $k=i-j$ we would have obtained the formula:

$$\hat{x}^*\hat{y} = \sum_{i=0}^{n} \varepsilon_i^{(n)} \left(\sum_{k=0}^{i} \left[\frac{i}{j} \right] (y_k x_{i-k}) \right)$$

which is the same product obtained by using equation 7 and evaluating $\hat{y}^*\hat{x}$. Therefore, $\hat{x}^*\hat{y} = \hat{y}^*\hat{x}$.

We have shown that the ε-algebra satisfies the commutative and associative laws of addition and multiplication, there exist a zero and a unity, and each element has an additive inverse. We conclude that the ε-algebra is a commutitive ring with unity.

The reader may notice that the first order ε-algebra is identical to the dual number algebra. Dual number algebra is the bases of the screw theory which is used in the kinematic and dynamic analysis of mechanisms. In this theory [9] real 3x1 vectors are replaced by dual 3x1 vectors. That is, a dual 3x1 vector is composed of two real 3x1 vectors. The advantantage of using dual vectors is in the simple notation and geometrical insight they provide. For example, a force vector and moment vector acting on a beam can be represented by a single vector. The first real vector component is the real force vector and the second real vector component is the moment vector. The advantage of the ε-algebra over the dual number algebra is in determining the time derivatives of functions. The next section shows how any order time derivative of a function can be evaluated by simply using the corresponding order ε-algebra during the function's evaluation.

2.2. Functions of ε-numbers

In this section we define a class of functions of ε-numbers. Specifically, we are interested in functions which can be expressed in the form of a Taylor series. We begin by proving the following theorem.

Theorem 1: Let x be a real function of time and let a be any real number. Define the $\varepsilon^{(n)}$-number $\hat{x}$ to be:

$$\hat{x} = \varepsilon_0^{(n)} x + \varepsilon_1^{(n)} \frac{dx}{dt} + .. + \varepsilon_n^{(n)} \frac{d^n x}{dt^n}$$

Then,

$$(\hat{x}-a)^m = \varepsilon_0^{(n)} (x-a)^m + \varepsilon_1^{(n)} \frac{d(x-a)^m}{dt} + .. + \varepsilon_n^{(n)} \frac{d^n (x-a)^m}{dt^n}$$

where m is a positive integer.

Proof: We use mathematical induction as the method of proof. Since a is a constant, the statement is true for $m=1$. If it is true for some integer m, we will show it is also true for m+1 and therefore, for all m. Define

$$\hat{y}_m = (\hat{x}-a)^m$$

Denote the components of $\hat{y}_m$ as:

$$\hat{y}_m = \varepsilon_0^{(n)} y_{m,0} + \varepsilon_1^{(n)} y_{m,1} + \cdots + \varepsilon_n^{(n)} y_{m,n}$$

Using the formula for products of ε-numbers, Eq. 7, gives:

$$\hat{y}_{m+1} = \hat{y}_1 * \hat{y}_m = \sum_{i=0}^{n} \varepsilon_i^{(n)} \sum_{j=0}^{i} \begin{bmatrix} i \\ j \end{bmatrix} (y_{1,j} y_{m,i-j})$$

Using the formula in appendix A for derivative of products, letting $y_{1,0}=u_0$ and $y_{m,0}=v_0$, gives:

$$\hat{y}_{m+1} = \sum_{i=0}^{n} \varepsilon_i^{(n)} \frac{d^i}{dt^i} (y_{1,0} y_{m,0}) = \sum_{i=0}^{n} \varepsilon_i^{(n)} \frac{d^i}{dt^i} (y_{m+1,0})$$

Therefore,

$$(\hat{x} - a)^{m+1} = \varepsilon_0^{(n)} (x-a)^{m+1} + \varepsilon_1^{(n)} \frac{d(x-a)^{m+1}}{dt} + .. + \varepsilon_n^{(n)} \frac{d^n (x-a)^{m+1}}{dt^n}$$

Because of the intended application of ε-numbers we can restrict our discussion only to special functions. We consider all of those functions which can be represented in a Taylor series. That is, if y is a function of x then we can write y in the following form:

$$y = f(x) = \sum_{m=0}^{\infty} \frac{f(a)^{(m)}}{m!} (x - a)^m \tag{8}$$

where a is a particular value of x and $f(a)^{(m)}$ denotes the m-th derivative with repect to x. If x is a function of time then we can use the above formula for determining the i-th time derivative of $f(x)$. It is:

$$\frac{d^i f(x)}{dt^i} = \sum_{m=0}^{\infty} \frac{f(a)^{(m)}}{m!} \frac{d^i (x - a)^m}{dt^i} \tag{9}$$

We define $f(\hat{x})$ as the value of the Taylor series when $\hat{x}$ is substituted in for x and $\varepsilon^{(n)}$-algebra is used to carry out the indicated computations. That is,

$$f(\hat{x}) = \sum_{m=0}^{\infty} \frac{f(a)^{(m)}}{m!} (\hat{x} - a)^m$$

Many of the problems concerning convergence are avoided by restricting the domain of $\hat{x}$. Let x be a continuous function of time, t. Let it be continuously differentiable and let:

$$\hat{x} = \varepsilon_0^{(n)} x + \varepsilon_1^{(n)} \frac{dx}{dt} + .. + \varepsilon_n^{(n)} \frac{d^n x}{dt^n}$$

With this value of $\hat{x}$ we can use Theorem 1 to evaluate $f(\hat{x})$.

$$f(\hat{x}) = \sum_{m=0}^{\infty} \frac{f^{(m)}(a)}{m!} \left[\sum_{i=0}^{n} \varepsilon_i^{(n)} \frac{d^i (x-a)^m}{dt^i} \right]$$

Changing the order of summation gives:

$$f(\hat{x}) = \sum_{i=0}^{n} \varepsilon_i^{(n)} \left[\sum_{m=0}^{\infty} \frac{f(a)^{(m)}}{m!} \frac{d^i (x-a)^m}{dt^i} \right]$$

Substituting in equations 8 and 9 gives:

$$f(\hat{x}) = \varepsilon_0^{(n)} f(x) + \varepsilon_1^{(n)} \frac{df(x)}{dt} + .. + \varepsilon_n^{(n)} \frac{d^n f(x)}{dt^n} \tag{10}$$

Thus, given a continuous function, $f(x)$, and an $\varepsilon^{(n)}$-number, $\hat{x}$, whose components are the time derivatives of x, then the function $f(\hat{x})$ evaluates to an $\varepsilon^{(n)}$-number whose components are the time derivatives of $f(x)$. It is this property we use in solving the forward and inverse kinematics problem.

Equation 10 provides a simple method of evaluating functions of $\varepsilon^{(n)}$-numbers. Table 1 list the functions we will need in solving kinematics problems using $\varepsilon^{(2)}$-algebra. Note that only those ε-numbers whose first real component, x_0, is not zero have a multiplicative inverse.

Functions
$sin(\hat{x}) = \varepsilon_0^{(2)}\sin(x_0) + \varepsilon_1^{(2)}x_1 cos(x_0) + \varepsilon_2^{(2)}(-sin(x_0)x_1^2 + x_2\cos(x_0))$
$cos(\hat{x}) = \varepsilon_0^{(2)}\cos(x_0) - \varepsilon_1^{(2)}x_1 sin(x_0) + \varepsilon_2^{(2)}(-cos(x_0)x_1^2 - x_2\sin(x_0))$
$atan2(\hat{y},\hat{x}) = \varepsilon_0^{(2)}atan2(y_0,x_0) + \varepsilon_1^{(2)}(x_0y_1 - x_1y_0) + \varepsilon_2^{(2)}(x_0y_2 - x_2y_0)$
$\sqrt{\hat{x}} = \varepsilon_0^{(2)}\sqrt{x_0} + \varepsilon_1^{(2)}\frac{1}{2\sqrt{x_0}}x_1 + \varepsilon_2^{(2)}(2x_2 - \frac{x_1^2}{x_0})/4\sqrt{x_0})$
$\hat{x}^{-1} = \varepsilon_0^{(2)}x_0^{-1} - \varepsilon_1^{(2)}x_0^{-2}x_1 + \varepsilon_2^{(2)}(2x_0^{-3}x_1^2 - x_0^{-2}x_2)$
Identities
$\sin^2(\hat{x}) + \cos^2(\hat{x}) = 1$
$sin(\hat{x}+\hat{y}) = sin(\hat{x})cos(\hat{y}) + cos(\hat{x})sin(\hat{y})$
$cos(\hat{x}+\hat{y}) = cos(\hat{x})cos(\hat{y}) - sin(\hat{x})sin(\hat{y})$

Table 1: Functions of $\varepsilon^{(2)}$-numbers which are used in this paper. $\hat{x} = \varepsilon_0^{(2)}x_0 + \varepsilon_1^{(2)}x_1 + \varepsilon_2^{(2)}x_2$ and $\hat{y} = \varepsilon_0^{(2)}y_0 + \varepsilon_1^{(2)}y_1 + \varepsilon_2^{(2)}y_2$

3. Manipulator Kinematics

One of the basic problems in manipulator control is the forward and inverse kinematics problem. The forward kinematics problem is to compute the position and orientation of the end-effector, given the position of the joint variables. This problem is typically solved using the Hartenburg-Denavit notation for the manipulator kinematics. The procedure is to affix a coordinate system to each link. The position and orienation of the coordinate systems are defined using homogeneous transformation matrices. These homogeneous transforms are parameterized using four parameters defined by Hartenburg and Denivit. The transformation from link i-1 to link i is

usually denoted by the matrix A_{i-1}^{i}. To determine the transformation from one link to the next we simply multiply the transformations together. Thus, the transformation from link six coordinates to link 0 coordinates is given by:

$$A_0^6(q_1,q_2,..,q_6) = A_0^1(q_1)A_1^2(q_2)..A_5^6(q_6)$$

Usually, there is a tool attached to the end-effector whose position and orientation with respect to link six coordinates is given by the constant homogeneous transformation matrix T. A path planning routine determines the desired homogeneous transform of the tool with respect to the link 0 coordinates based upon the desired task. We call this matrix T_b. The objective of the controller is to drive the positions of the joints so that the following equation is always satisfied.

$$A_0^6(q_1,q_2,...,q_6)T = T_b$$

For the controller to do this it must determined the desired position of the joint variables based on this equation. This is the inverse kinematics problem. That is, determine the q_i given the matrix T_b.

The following two sections present the solution to the forward and inverse kinematics problems using the ε-algebra.

3.1. Forward Kinematics Problem

From the above it is seen that the forward kinematics problem is a relatively simple problem. It only involves the multiplication of a few well defined matrices together. What we are interested in here is to use the same equations as above, but to obtain not only the homogeneous transform of the tool with respect to the base coordinates but also its time derivatives. To do this, we use the ε-algebra defined above.

For example, to determine T_b and its first and second time derivatives we use the $\varepsilon^{(2)}$-algebra. Define the $\varepsilon^{(2)}$ joint position to be:

$$\hat{q}=\varepsilon_0^{(2)}q + \varepsilon_1^{(2)}\dot{q} + \varepsilon_2^{(2)}\ddot{q}$$

The $\varepsilon^{(2)}$ homogeneous transform of link i coordinates with respect to link i-1 coordinates is defined the same as A_{i-1}^{i} except that q_i is changed to $\hat{q}_i$ and all computations are done with $\varepsilon^{(2)}$-algebra. From the above discussion, we know the result will be equal to:

$$\hat{A}_{i-1}^{i} = \varepsilon_0^{(2)}A_{i-1}^{i} + \varepsilon_1^{(2)}\dot{A}_{i-1}^{i} + \varepsilon_2^{(2)}\ddot{A}_{i-1}^{i}$$

If we use $\varepsilon^{(2)}$ homogeneous transforms $\hat{A}_{i-1}^{i}$ in the equation for calculating T_b, the result is an $\varepsilon^{(2)}$ homogeneous transform $\hat{T}_b$ whose value is:

$$\hat{T}_b = \hat{A}_0^1(\hat{q}_1)\hat{A}_1^2(\hat{q}_2)..\hat{A}_5^6(\hat{q}_6)T = \varepsilon_0^{(2)}T_b + \varepsilon_1^{(2)}\dot{T}_b + \varepsilon_2^{(2)}\ddot{T}_b \qquad (11)$$

3.2. Inverse Kinematics Problem

The inverse kinematics problem is well known and there are many excellent sources [5, 10, 11, 12, 13] for reference. Although the approach in each reference may be slightly different, the ε-algebra is equally applicable to all. Since the purpose of this section is to illustrate the use of ε-algebra in the inverse kinematics problem, most of the details of the inverse kinematics solution will be left out and only those results which pertain to the use of ε-algebra will be presented.

The inverse kinematics solution is composed of two parts. The first part is to compute the position of the point of intersection of the wrist axes. This position is only a function of the first three joint variables. Hence, we have three equations and three unknowns from which we can solve for the first three joint positions.

The next part of the problem is to determine the positions of the last three joint variables which gives the correct orientation of the end-effector.

We start by modeling the kinematics of the manipulator. The Hartenburg-Denivit parameters for our example manipulator, the PUMA, are listed in Table A.1 in the appendix.

Using the parameters in Table A.1, we write the algebraic equations for $\hat{A}_0^3$ and $\hat{A}_3^6$. They are:

$$\hat{A}_0^3 = \begin{bmatrix} \hat{c}_1\hat{c}_{23} & -\hat{s}_1 & \hat{c}_1\hat{s}_{23} & a_2\hat{c}_1\hat{c}_2+a_3\hat{c}_1\hat{c}_{23}-d_2\hat{s}_1 \\ \hat{s}_1\hat{c}_{23} & \hat{c}_1 & \hat{s}_1\hat{s}_{23} & a_2\hat{s}_1\hat{c}_2+a_3\hat{s}_1\hat{c}_{23}+d_2\hat{c}_1 \\ -\hat{s}_{23} & 0 & \hat{c}_{23} & -a_2\hat{s}_2-a_3\hat{s}_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\hat{A}_3^6 = \begin{bmatrix} \hat{c}_4\hat{c}_5\hat{c}_6-\hat{s}_4\hat{s}_6 & -\hat{c}_4\hat{c}_5\hat{s}_6-\hat{s}_4\hat{c}_6 & \hat{c}_4\hat{s}_5 & d_6\hat{c}_4\hat{s}_5 \\ \hat{s}_4\hat{c}_5\hat{c}_6+\hat{c}_4\hat{s}_6 & -\hat{s}_4\hat{c}_5\hat{s}_6+\hat{c}_4\hat{c}_6 & \hat{s}_4\hat{s}_5 & d_6\hat{s}_4\hat{s}_5 \\ -\hat{s}_5\hat{c}_6 & \hat{s}_5\hat{s}_6 & \hat{c}_5 & d_6\hat{c}_5+d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The desired value of the matrix $\hat{T}_b$ is input to the inverse kinematics routine. Since the matrix T is a constant we can use equation 11 to numerically determine the desired value $\hat{A}_0^6$.

$$\hat{A}_0^6 = \hat{T}_b T^{-1}$$

For notational convienance we give names to the columns of $\hat{A}_0^6$.

$$\hat{A}_0^6 = \begin{bmatrix} \hat{n} & \hat{s} & \hat{a} & \hat{p} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We can now compute the value of the position vector, $\hat{r}$, where the wrist axes intersect. It is:

$$\hat{\mathbf{r}} = \hat{\mathbf{p}} - d_6\hat{\mathbf{a}} = \begin{bmatrix} \hat{r}_x \\ \hat{r}_y \\ \hat{r}_z \end{bmatrix}$$

Next, we compute the following functions.

$$r_1 = \sqrt{d_4^2 + a_3^2}\,; \qquad \hat{r}_2 = \sqrt{\hat{r}_x^2 + \hat{r}_y^2}$$

$$\hat{r}_3 = \sqrt{\hat{r}_x^2 + \hat{r}_y^2 - d_2^2}\,; \qquad \hat{r}_4 = \sqrt{\hat{r}_x^2 + \hat{r}_y^2 + \hat{r}_z^2 - d_2^2}$$

$$cos(\hat{\beta}_1) = \frac{\hat{r}_3}{\hat{r}_4}\,; \qquad sin(\hat{\beta}_1) = -\frac{\hat{r}_z}{\hat{r}_4}$$

$$cos(\hat{\beta}_2) = \frac{a_2^2 + \hat{r}_4^2 - d_4^2 - a_3^2}{2a_2\hat{r}_4}\,; \qquad sin(\hat{\beta}_2) = \sqrt{1 - \cos^2(\hat{\beta}_2)}$$

$$cos(\hat{\beta}_3) = \frac{a_2^2 + r_1^2 - \hat{r}_4^2}{2a_2 r_1}\,; \qquad sin(\hat{\beta}_3) = \sqrt{1 - \cos^2(\hat{\beta}_3)}$$

$$cos(\beta_4) = -\frac{a_3}{r_1}\,; \qquad sin(\beta_4) = \frac{d_4}{r_1}$$

From these we can compute the sine and cosine of $\hat{\theta}_1$, $\hat{\theta}_2$, and $\hat{\theta}_3$.

$$sin(\hat{\theta}_1) = \frac{\hat{r}_y\hat{r}_3 - \hat{r}_x d_2}{\hat{r}_2}\,; \qquad cos(\hat{\theta}_1) = \frac{\hat{r}_x\hat{r}_3 + \hat{r}_y d_2}{\hat{r}_2}$$

$$sin(\hat{\theta}_2) = \sin(\hat{\beta}_1)\cos(\hat{\beta}_2) + cos(\hat{\beta}_1)sin(\hat{\beta}_2); \qquad cos(\hat{\theta}_2) = \cos(\hat{\beta}_1)\cos(\hat{\beta}_2) - sin(\hat{\beta}_1)sin(\hat{\beta}_2)$$

$$sin(\hat{\theta}_3) = \sin(\hat{\beta}_3)\cos(\beta_4) - cos(\hat{\beta}_3)sin(\beta_4); \qquad cos(\hat{\theta}_3) = \cos(\hat{\beta}_3)\cos(\beta_4) + sin(\hat{\beta}_3)sin(\beta_4)$$

The joint variables $\hat{\theta}_1$, $\hat{\theta}_2$, and $\hat{\theta}_3$ can now be computed using the *atan* 2 function, which returns the angle given both the sine and the cosine of the angle:

$$\hat{\theta}_1 = atan\,2(\hat{s}_1,\hat{c}_1) \qquad \hat{\theta}_2 = atan\,2(\hat{s}_2,\hat{c}_2) \qquad \hat{\theta}_3 = atan\,2(\hat{s}_3,\hat{c}_3)$$

Since $\hat{\theta}_1$, $\hat{\theta}_2$, and $\hat{\theta}_3$ are now known, we can compute $\hat{\mathbf{A}}_0^3$ and use this to solve for the numerical value of $\hat{\mathbf{A}}_3^6$.

$$\hat{A}_3^6 = (\hat{A}_0^3)^{-1}\hat{T}_b = \begin{bmatrix} \hat{b} & \hat{c} & \hat{d} & \hat{e} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The sine and cosine of $\hat{\theta}_4$, $\hat{\theta}_5$, and $\hat{\theta}_6$ are now computed in the following order:

$$cos(\hat{\theta}_5) = \hat{d}_z; \qquad sin(\hat{\theta}_5) = \sqrt{1-\cos^2(\hat{\theta}_5)}$$

$$cos(\hat{\theta}_4) = \hat{d}_x/sin(\hat{\theta}_5); \qquad sin(\hat{\theta}_4) = \hat{d}_y/sin(\hat{\theta}_5)$$

$$cos(\hat{\theta}_6) = -\hat{b}_z/sin(\hat{\theta}_5); \qquad sin(\hat{\theta}_6) = \hat{c}_z/sin(\hat{\theta}_5)$$

where $\hat{b}_z$ is the z component of $\hat{\mathbf{b}}$, etc. The joint variables $\hat{\theta}_4$, $\hat{\theta}_5$, and $\hat{\theta}_6$ can now be computed using the *atan* 2 function:

$$\hat{\theta}_4 = atan\,2(\hat{s}_4,\hat{c}_4) \qquad \hat{\theta}_5 = atan\,2(\hat{s}_5,\hat{c}_5) \qquad \hat{\theta}_6 = atan\,2(\hat{s}_6,\hat{c}_6)$$

4. An Ada Programming Example

A program was written in Ada [14, 15] to illustrate the method. First the forward kinematics problem was programmed. The output of this program was the ε-homogeneous transform of the tool coordinates. This output was then used as input to the inverse kinematics program. The algebra used was of order 2 so that joint positions, velocities, and accelerations would be obtained. Ada was choosen as the programming language because of the overloading feature of the language.

The overloading feature of Ada allows two procedures or two functions to have identical names in the same scope provide that their profiles are different. The profile is the ordered list of the base types of the parameters and, for functions, the base type of the result as well. This feature applies not only to subprogram names but also to standard operators, such as '+', '-', '/', and '*'. For example, '*' is used to multiply two ε-homogeneous transforms and also to multiply two ε-numbers.

The algebra was defined in a package called ALGEBRA. There are two parts of this package, the specification part and the body part. A portion of the specification part is shown below:

```
package ALGEBRA is

    subtype index is INTEGER range 0..INTEGER'LAST;
    type epsilon is array(index range <>) of REAL;
    function "+" (x,y : in epsilon) return epsilon;
    function "*" (x,y : in epsilon) return epsilon;

end ALGEBRA;
```

The implementation of each function or procedure defined in the specification part of the package is defined in the body part of the package. For example, the two functions defined above were implemented with the following body.

```
package body ALGEBRA is

    function "+" (x,y : in epsilon) return epsilon is
        order : constant INTEGER := x'LENGTH - 1;
        w     : epsilon(0..order);
    begin
        for i in 0..order loop w(i) := x(i) + y(i); end loop;
        return w;
    end "+";

    function "*" (x,y : in epsilon) return epsilon is
        order : constant INTEGER := x'LENGTH -1;
        w     : epsilon(0..order);
        prod  : REAL;
    begin
        w(0) := x(0)*y(0);
        for i in 1..order loop
            prod := x(0)*y(i);
            for j in 1..i-1 loop
                prod := prod + binomial(i,j)*x(j)*y(i-j);
            end loop;
            w(i) := prod + x(i)*y(0);
        end loop;
        return w;
```

```
    end "*";

end ALGEBRA;
```

Note, binomial() in the function "*" is a precomputed array of the binomial coeficients.

After the algebra was implemented the functions listed in Table 1 were programmed. Finally, the forward and inverse kinematics solutions were programmed using the same equations as contained in the above sections.

The package ALGEBRA was written so that more than one order of algebra can use the same routines. The order of the algebra is determined by the LENGTH attribute of the input array variables. In the multiplication routine the order of the algebra is limited by the dimension of the predefined array binomial().

The homogeneous transforms used by the program were defined as 4x4 arrays of ε-numbers. A general function for multiplying two homogeneous transforms that would work for any order algebra was not possible, since the dimension of the ε-numbers must be declared when the homogeneous transform is defined. That is, "*" was defined for homogeneous transforms in the following way

```
package kinematics is

    type homogeneous is array (1..4,1..4) of epsilon (0..2);
    function "*" (x,y : in homogeneous) return homogeneous;

end kinematics;
```

The inverse kinematics section of code is written exactly as if one were solving only for position. For example, the code for solving for $\hat{\theta}_1$ was:

```
s(1) := (ry*r3 - d2*rx)/r2;     -- compute sine of θ1^
c(1) := (rx*r3 - d2*ry)/r2;     -- compute cosine of θ1^
theta(1) := atan2(s(1),c(1));   -- compute θ1^
```

where rx, ry, r2, r3, s(1), c(1), and theta(1) are ε-numbers and d2 is a real variable.

Note that "*" in d2*rx is another overload of the operator "*" which returns an ε-number.

Singular points are always an issue in the inverse kinematics problem. The problems which arise using the ε-algebra are similar to those encountered with real algebra. For example, evaluation of the square root of a negative number and division by zero.

5. Conclusion

A new algebra was defined for the use in solving the forward and inverse kinematics problem of manipulator control. The properties of the algebra were investigated and functions of an ε-numbers were defined. The fundamental result was that if $f(x)$ is a continuous function of the real variable x, and if x is a continuous function of time, and if we let the components of $\hat{x}$ be the time derivatives of x, then $f(\hat{x})$ evaluates to an ε-number whose components are the time derivatives of $f(x)$. It was this property that was used in the formulation and solution to the forward and inverse kinematics problem of manipulators.

Although any language could be used, the Ada language was used for illustration because of the ease of programming. The program was written as if only the forward and inverse problems in position were being solved. However, the same program can be used to solve for any time derivative of end-effector position for the forward kinematics problem or any time derivative of joint positions for the inverse kinematics problem, by simply changing the order of the algebra used by the program.

The use of the ε-algebra provides a simple method of solving the forward and inverse kinematics problem for manipulators. It is felt that the algebra is a useful tool

for the analysis of manipulator kinematics.

6. Acknowledgement

This work was supported under a TRW Fellowship Grant to the University of Michigan.

7. Appendix A

The following is an identity used in this paper.

Let u_0 and v_0 be continuous functions of time and define:

$$u_i = \frac{d^i u_0}{dt^i} \; ; \qquad v_i = \frac{d^i v_0}{dt^i}$$

Then:

$$\frac{d^i}{dt^i}(u_0 v_0) = \sum_{j=0}^{i} \left[\begin{matrix} i \\ j \end{matrix} \right] u_j v_{i-j}$$

where $\left[\begin{matrix} i \\ j \end{matrix} \right] = \frac{i!}{j!(i-j)!}$, the binomial coefficient, i non-negative integer and $\left[\begin{matrix} i \\ 0 \end{matrix} \right] = 1$.

The Hartenburg-Denivit parameters for the PUMA manipulator are listed in the following table.

Joint i	θ_i	α_i	a_i	d_i	Range
1	var	-90.0	0.0	0.0	-160 to +160
2	var	0.0	431.8mm	149.09mm	-225 to +45
3	var	90.0	-20.33mm	0	-45 to +225
4	var	-90.0	0.0	433.07mm	-110 to +170
5	var	90.0	0.0	0.0	-100 to +100
6	var	0.0	0.0	56.25mm	-266 to +266

Table A.1: PUMA Manipulator Link Coordinate Parameters (var denotes variable)

8. References

[1] J. Denavit, R.. Hartenberg, "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," *AME Journal of Applied Mechanics*, pp. 215-221, June 1955.

[2] D.L. Pieper, "The Kinematics of Manipulators under Computer Control," *Ph.D. Thesis,*, 1968.

[3] D.E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Machine Systems*, vol. MM-10, pp. 47-43, 1969.

[4] D.E. Whitney, "The mathematics of coordinated control of prostheses and manipulators," *J. Dynamicystems, Measurement. and Control*, pp. 303-309, Dec. 1972.

[5] R.P. Paul, *Robot Manipulators: Mathematics, Programming and Control.* Cambridge, Mass: MIT Press, 1981.

[6] R. Featherstone, "Position and velocity transformations between robot end effector coordinates and joint angles," *Int. J. Robotics Research*, vol. 2, pp. 35-45. June, 1983.

[7] J.Y.. Luh, M.W. Walker, R.P. Paul, "Resolved Acceleration Control of a Mechanical Manipulator," *IEEE Trans. Automatic Control*, vol. 25, pp. 468-474. 1980.

[8] J.M. Hollerbach, G. Sahar, "Wrist-Partitioned Inverse Kinematic Accelerations and Manipulator Dynamics," *International Conference on Robotics*, pp. 152-161, March 13-15, 1984.

[9] G.R. Pennock, A.T. Yang, "Application of Dual-Number Matrices to the Inverse Kinematics Problem of Robot Manipulators," *A.S.M.E J. Mech., Trans., & Auto. in Design*, vol. 107,2, 1985.

[10] Wesley E. Snyder, *Industrial Robots: Computer Interfacing and Control.* Englewood Cliffs, NJ: Prentice-Hall, 1985.

[11] Shimon Y. Nof, *Handbook of Industrial Robotics*. New York, NY: John Wiley & Sons, 1985.

[12] A.J. Critchlow, *Introduction to Robotics*. New York, NY: Macmillan, 1985.

[13] H. Asada, J.E. Slotine, *Robot Analysis and Control*. New York, NY: Wiley, 1986.

[14] A.N. Habermann, D.E. Perry, *Ada for Experienced Programmers*. Reading, Mass.: Addison-Wesley, 1983.

[15] *Ada Language Reference Manual*. Pittsburgh, Penn: Gensoft Corporation, Feb. 17, 1983.

UNIVERSITY OF MICHIGAN
3 9015 03527 5299