New Notation for Serial Kinematic Chains

Berthold K. P. Horn

May 1987

Abstract: The description of a serial kinematic chain should be unique, unambiguous, simple to determine, easy to use and well-behaved when small changes are made in the arrangement of the elements of the chain. The notation currently in use, introduced by Denavit and Hartenberg, does not satisfy all of these criteria. It involves arbitrary choices, so that more than one description may apply to a given kinematic chain. More importantly, the parameters relating the links in the chain can be very sensitive to small changes in the physical arrangement of the chain. This is particularly true of so-called *ideal* chains, ones that permit closed-form solution of the inverse kinematic problem, since these often involve geometries where adjacent axes are parallel, perpendicular or intersect. A new notation is proposed here that does not suffer the above-mentioned short-comings. To demonstrate some of the advantages of the new notation, it is applied to the problem of finger-printing a robot arm and to the solution of the inverse kinematic problem of near-ideal arms.

Introduction

Let us start right away by defining the new notation:

Let the *home position* of a serial kinematic chain be an arbitrary position specified in terms of the joint variables. The home position may, for example, be chosen to be the one where all joint variables are zero. The *base coordinate system* is an arbitrary external coordinate system fixed with respect to the base of the kinematic chain. Erect a coordinate system in each link of the chain in such a way that the coordinate axes are parallel to those of the base coordinate system when the chain is in the home position. The kinematic chain is then fully specified when the following are given for the chain in the home position:

- The set of unit vectors $\{\hat{\mathbf{o}}_i^*\}$, parallel to the directions of motion of the prismatic joints.
- The set of unit vectors $\{\hat{\boldsymbol{\omega}}_i^*\}$, parallel to the axes of the revolute joints.
- The set of offset vectors $\{\mathbf{d}_i^*\}$ determined recursively with respect to the chosen base origin as follows:

The first reference point is the origin of the given base coordinate system. Drop a perpendicular from this reference point to the first revolute joint axis. This defines the first offset vector as well as a reference point on the first revolute joint axis. Now drop a perpendicular from this new reference point onto the second revolute joint axis. This defines the second offset vector and a new reference point. Continue in this fashion to the last revolute joint. Connect the last reference point so found to a chosen *tip reference point* in the final link. This defines the last offset vector.

The reference points so defined are the origins of the link coordinate systems. A prismatic joint does not yield a new reference point. The origin of the corresponding coordinate system is taken to be that of the last reference point encountered working outward from the base coordinate system.

This completes the description of the new notation for describing a kinematic chain. Note that there are no arbitrary choices. Thus a given kinematic chain has only one description with respect to a given base coordinate system and a tool reference point in the last link. Also, note that small changes in the kinematic chain can introduce only small changes in the description. If one of the axes is turned through a small angle, for example, the corresponding unit vector in the description changes, as does the reference point on the axis that is turned, as well as those on axes

further from the base. All of these changes are small, however. Similarly, if an axis is displaced by a small amount without turning, the corresponding offset, and offsets of axes further from the base, will change, but only a little.

The parameters are not unconstrained, since the direction vectors, $\hat{\mathbf{o}}_i^*$, and the axes vectors, $\hat{\boldsymbol{\omega}}_i^*$, have to be unit vectors and since each of the offsets, $\mathbf{d}_i^*$, has to be orthogonal to the following axis vector $\hat{\boldsymbol{\omega}}_i^*$. This means that the parameters are redundant, that is, there are more parameters than degrees of freedom. These parameters are, however, more convenient than alternate non-redundant sets of parameters, as we shall see.

Throughout this paper, as is customary, it is assumed that joint variables are controlled accurately or can be measured accurately. Modern methods for measuring joint variables appear to be adequate to assure that errors introduced by non-linearity will tend to be swamped by other contributions to inaccuracy in positioning of the last link. The methods presented here do not, however, depend on the assumption that the measured joint variable of a revolute joint is zero when there is a particular alignment of the links or that the measured joint variable of a prismatic joint is zero when there is a particular alignment of joints.

Notations for Kinematic Chains

There are several things that a kinematic notation has to provide:

- A standard coordinate system within each link of the chain.
- The means for determining the transformations between these coordinate system.
- Methods for using these transformations in the analysis of kinematics, statics and dynamics.

Denavit & Hartenberg developed such a notation, which was used by Uicker & Kahn in dealing with the dynamics of robot manipulators and popularized further by Pieper and more recently by Richard Paul in his book *Robot Manipulators*.

Review of Denavit & Hartenberg Notation

We find the following description of the notation of Denavit and Harten-

berg in Chapter 2 of Richard Paul's classic book *Robot Manipulators—Mathematics, Programming & Control*:

> We will now consider the specification of the A matrices. A serial link manipulator consists of a sequence of links connected together by actuated joints. For an n degree of freedom manipulator, there will be $n+1$ links and n joints. The base of the manipulator is link 0. Link 1 is connected to the base link by joint 1. There is no joint at the end of the final link. The only significance of links is that they maintain a fixed relationship between the manipulator joints at each end of the link.
>
> Any link can be characterized by two dimensions: the common normal distance a_i, and the angle α_i between the axes in a plane perpendicular to the common normal It is customary to call a_i the length and α_i the twist of the link. Generally, two links are connected at each joint axis.
>
> The axis will have two normals to it, one for each link. The relative position of two such connected links is given by d_i, the distance between the normals along the joint i axis, and θ_i, the angle between the normals measured in a plane normal to the axis. The quantities d_i and θ_i are called the distance and the angle between the links, respectively.
>
> In order to describe the relationship between links, we will assign coordinate frames to each link. We will first consider revolute joints in which θ_i is the joint variable. The origin of the coordinate frame of link i is set to be at the intersection of the common normal axes of joints i and $i+1$ and the axis of joint $i+1$. In the case of intersecting joint axes, the origin is at the point of the intersection of the joint axes. If the axes are parallel the origin is chosen to make the joint distance zero for the next link whose coordinate origin is defined. The z-axis for link i will be aligned with the axis of joint $i+1$. The x-axis will be aligned with any common normal which exists and is directed along the normal from joint i to joint $i+1$. In the case of intersecting joints, the direction of the x-axis is parallel or antiparallel to the vector cross-product $\mathbf{z}_{i-1} \times \mathbf{z}_i$. Notice that this condition is also satisfied for the x-axis directed along the normal between joints i and $i+1$. The angle θ_i is zero for the i-th revolute joint when $\mathbf{x}_{i-1}$ and $\mathbf{x}_i$ are parallel and have the same direction.
>
> In the case of a prismatic joint, the distance d_i is the joint variable. The direction of the joint axis is the direction in which the joint moves. The direction of the joint axis is defined but, unlike a revolute joint,

the position in space is not defined. In the case of a prismatic joint the length a_i has no meaning and is set to zero. The origin of the coordinate frame for a prismatic joint is coincident with the next defined link origin. The z-axis of the prismatic link is aligned with the axis of joint $i+1$. The x-axis is parallel or antiparallel to the vector cross product of the direction of the prismatic joint and $\mathbf{z}_i$. For a prismatic joint, we will define the zero position when $d_i = 0$.

With the manipulator in its zero position, the positive sense of rotation for revolute joints or displacement for prismatic joints can be decided and the sense of direction of the z-axes determined. The origin of the base link (zero) will be coincident with the origin of link 1. If it is desired to define a different reference coordinate system, then the relationship between the reference and base coordinate systems can be described by a fixed homogeneous transformation. At the end of the manipulator, the final displacement d_6 or rotation θ_6 occurs with respect to $\mathbf{z}_5$. The origin of the coordinate system for link 6 is chosen to be coincident with that of the link 5 coordinate system. If a tool (or end effector) is used whose origin and axes do not coincide with the coordinate system of link 6, the tool can be related by a fixed homogeneous transformation to link 6.

Having assigned coordinate frames to all links according to the preceding scheme, we can establish the relationship between successive frames $n-1$, n by the following rotations and translations:

rotate about $\mathbf{z}_{i-1}$, an angle, θ_i;

translate along $\mathbf{z}_{i-1}$, a distance d_i;

translate along rotated $\mathbf{x}_{i-1} = \mathbf{x}_i$ a length a_i;

rotate about $\mathbf{x}_i$, by the twist angle α_i.

This may be expressed as the product of four homogeneous transformations relating the coordinate frame of link n to the coordinate frame of link $n-1$. This relationship is called an A matrix.

Once the link coordinate frames have been assigned to the manipulator, the various constant link parameters can be tabulated: d_i, a_i, and α_i for a link following a revolute joint and θ_i and α_i for a link following a prismatic joint. Based on these parameters the constant sine and cosine values of the α_i's may be evaluated. The A matrices then become a function of the joint variable θ_i or, in the case of a prismatic joint, d_i. Once these values are known, the values for the six A_i transformation matrices can be determined.

Problems with Denavit & Hartenberg Notation

We see that there are several arbitrary choices. A few quotes from chapter 3 of John Craig's book *Introduction to Robotics—Mechanics & Control* illustrate this further:

> The distance a_i is measured along a line that is mutually perpendicular to both axes. This mutual perpendicular always exists and is unique, except when the axes are parallel, in which case there is an infinite number of mutual perpendiculars of equal length.
>
> In the case of intersecting axes, the twist α_i is measured in the plane containing both axes, but the sense is lost. In this special case one is free to assign the sign arbitrarily.
>
> These conventions have been chosen so that in a case where a quantity could be assigned arbitrarily, a zero value is assigned so that later caclulations will be as simple as possible.
>
> A final note on uniqueness: The conventions outlined above do not result in a unique attachment of frames to links. First of all, when we align the $\mathbf{z}_i$ axis with the axis of joint i, there are two choices for the direction of $\mathbf{z}_i$. Furthermore, in the case of intersecting joint axes (i.e. $a_i = 0$), there are two choices for the direction of $\mathbf{x}_i$, corresponding to the choice of direction for the normal of the plane containing $\mathbf{z}_i$ and $\mathbf{z}_{i+1}$. Also, when prismatic joints are present there is quite a bit of freedom in frame assignment.

Note that there are a number of arbitrary choices that have to be made when special alignments occur. As we note later such special alignments are not uncommon in practice, since they are needed to assure that the inverse kinematic problem has a closed-form solution[1].

Other suggestions have been made for standard coordinate systems in the links of kinematic chains. In work on dynamics, for example, the center of mass is a natural choice for the origin and the principal axes are natural choices for the directions of the coordinate axes of a link. So far, these alternate coordinate systems have been accomodated by introducing transformations to and from the coordinate systems established in the links using the Denavit & Hartenberg notation.

Kinematic Solution of Ideal Kinematic Chains

The *forward* kinematic problem is that of determining the position and

[1] It is interesting to note, by the way, that the link coordinate systems shown for the Stanford arm on the cover of Paul's book do not conform strictly to the notation of Denavit & Hartenberg.

orientation of the final link of the kinematic chain given the joint variables. This problem has a unique solution that can be computed directly. The *inverse* kinematic problem is that of finding a set of joint variables that will place the final link in a given position and orientation. There are typically several solutions for positions and orientations within the workspace, but these cannot be easily found for an arbitrary kinematic chain. An *ideal* kinematic chain is one for which a closed-form inverse kinematic solution exists. Almost all industrial robot arms are designed so that the inverse kinematic problem can be solved directly. Only a robot designed to operate solely in teach-by-showing or play-back mode can be useful without an efficient method for solving the inverse kinematics problem.

The solution methods for ideal kinematic chains depend on the intersection of certain joint axes, as well as parallel or perpendicular alignments of axes. Many arms with six joints, for example, are designed so that the inverse kinematic problem can be broken down into two subproblems, each with only three unknowns, by arranging for the axes of the last three joints to intersect in a point.

In practice, a kinematic chain departs slightly in the geometry from that specified in its design. This means that a closed-form kinematic solution does not exists for the actual arm. Thus methods must be developed for efficiently finding solutions of the inverse kinematic problem for these *near-ideal* kinematic chains.

Forward Kinematics

Let us use the notation

$$R = Rot(\hat{\boldsymbol{\omega}}, \theta)$$

for the clockwise rotation by an angle θ about the axis through the origin with direction specified by the unit vector $\hat{\boldsymbol{\omega}}$. It is not important whether this is represented using an orthonormal matrix, axis-and-angle or a unit quaternion (see Appendix). Let

$$\mathbf{x}' = R(\mathbf{x})$$

be the vector obtained by rotating $\mathbf{x}$, and let

$$R_1 \circ R_2$$

be the composition of the two rotations R_1 and R_2.

Working Backwards

Suppose that the kinematic chain has n_p prismatic joints and n_r revolute joints. Let $n = n_p + n_r$ be the total number of degrees of freedom. The

joints are numbered from 1 to n, while the links are numbered from 0 to n, with the fixed base being link number 0 and link n being the final link.

We wish to compute the position of the tip reference point in the base coordinate system for a given set of joint variables. One way to arrive at a procedure for doing this is to imagine the kinematic chain in the home position and then moving one joint at a time, starting with the one furthest from the base.

In the case of a prismatic joint one uses the recursive relationship

$$^{i}\mathbf{x} = {}^{i+1}\mathbf{x} + p_i\hat{\mathbf{o}}_i^*,$$

where p_i is the extend of the linear motion of the joint. In the case of a revolute joint one uses instead the recursive relationship

$$^{i}\mathbf{x} = {}^{i}R_{i+1}({}^{i}\mathbf{x}) + \mathbf{d}_i^*,$$

where ${}^{i}R_{i+1} = Rot(\hat{\boldsymbol{\omega}}_i^*, \theta_i)$, and θ_i is the angular motion of the joint. One starts with the initial offset ${}^{n}\mathbf{x} = \mathbf{d}_n^*$ and the desired result is, ${}^{0}\mathbf{x}$, the position of the tip reference point in terms of the base coordinates. A subscript is used to denote a quantity in a particular link, while a superprescript denotes a coordinate system in which a quantity is measured.

The forward kinematic computation is very simple, requiring $2n_r$ trigonometric function evaluations and only about $3n_p + 18n_r$ multiplications and $3n_p + 14n_r$ additions, if Rodrigues's formula is used to deal with the rotation of vectors (see Appendix).

To compute the orientation of the final link as well, we have to compose the rotations using

$$^{i}S_n = {}^{i}R_{i+1} \circ {}^{i+1}S_n$$

with ${}^{n}S_n = I$, a rotation of zero angle about an arbitrary axis. The composite rotation from the final link to the base coordinate system is then ${}^{0}S_n$. Composing the rotation about doubles the amount of work, but allows us to easily calculate the position in base coordinates of an arbitrary point in the final link using:

$$({}^{0}\mathbf{x}' - {}^{0}\mathbf{x}) = {}^{0}S_n({}^{n}\mathbf{d}' - {}^{n}\mathbf{d})$$

Working Forwards

If the rotations are composed, it is also possible to work in the direction from the base to the final link. In the case of a prismatic joint one uses the recursive relationship

$$^{i}\mathbf{y} = {}^{i-1}\mathbf{y} + {}^{0}S_i(p_i\hat{\mathbf{o}}_i*).$$

In the case of a revolute joint one uses instead

$$^{0}S_i = {}^{0}S_{i-1} \circ {}^{i-1}R_i$$

and

$$^{i}\mathbf{y} = {}^{i-1}\mathbf{y} + {}^{0}S_i(d_i^*).$$

Here ${}^{0}S_0 = I$ and $\mathbf{y}_0 = 0$.

Finger-printing

In order to be able to position the last link of a kinematic chain in any desired position and orientation within its work-space, it is neccessary to solve the inverse kinematics problem. This can only be done if the parameters of the kinematic chain are known. High accuracy cannot be achieved if these parameters are taken from the design plans without allowance for manufacturing tolerances. To accurately determine the joint variables that will place the last link in the desired position and orientation, it is neccessary to know the parameters of the actual kinematic chain. It has not proven practical to obtain these paramaters with sufficient precision by direct measurement on the disassembled chain.

Arm Calibration

Calibration procedures have been proposed where the position of a specified *calibration point* in the last link is accurately measured for a large number of joint variable combinations. The parameters of the arm are taken to be the ones that would place the calibration point in the measured positions given the corresponding joint variables.

One has to make the following decisions when one designs such a calibration procedure:

- What to measure.
- How many test positions to use.
- How to determine the parameters from the measured positions.

One might expect that one would have to measure the position and orientation of the last link for each set of joint variables. It is difficult to make accurate measurements of the attitude of a small rigid body in space, so it is fortunate that calibration procedures can be devised that rely only on positional information. In essence, each component of the measurement provides a constraint on the parameters, and one just has to make sure that there are at least as many constraints as there are parameters to be found. If the orientation of the final link is not measured, twice

as many test positions have to be used as would be needed were these measurements available.

It should also be apparent that it may be possible to gather the requisite information by measuring only one or two of the three components of the position. This simplifies the mechanics of the calibration procedure, but also makes the solution more sensitive to errors. In a similar vein, the measurements need not actually be the components of the position in the base coordinate system. They can be linear combinations of these components or even distances from fixed points whose coordinates are accurately known. All that matters is that a sufficiently large number of independent constraints is gathered. This flexibility is important, since apparatus for accurately measuring the position of a point in three-dimensional space is expensive and difficult to calibrate and use.

By position of the final link is usually meant the position of some reference point in the final link. It is important that this reference point be chosen carefully, since it must move when any one of the joint variables changes. It should not, for example, lie on the last joint axis of an arm whose last axis is revolute. Some thought should be given to the choice of this point, since the sensitivity of its position to a variation in one of the joint variables will determine the accuray with which the parameters of that joint can be found. There are, however, other factors to consider also. The reference point should be near the points in the final link that are likely to be of significance in the application, such as the center point of a tool to be attached to the final link.

The minimum number of test positions that have to be used in the calibration can be determined simply by considering the number of unknown parameters and the number of measurements taken in each test. We show later that the description of a kinematic chain with six revolute joints involves 27 parameters. So, if the only the position of a point in the final link is measured, then at least 9 test positions must be used to provide enough constraint to determine the parameters of the kinematic chain (since each measurement supplies three constraints). Measurement errors will lead to errors in the parameters. If the test positions are poorely chosen and do not sample the work-space adequately, the errors in the parameters can be quite large in relation to the measurement errors. This means that kinematic solutions for position and orientations that are not near those sampled will tend to be inaccurate.

These errors can be reduced by using more than the minimum number of test positions. In this case there will be more constraints than parameters and the resulting equations will almost certainly be inconsis-

tent. A least-squares procedure is then called for.

Problems with Denavit & Hartenberg Notation

Such a least-squares procedure may be developed using the Denavit & Hartenberg notation for kinematic chains. It turns out that such a procedure does not work well, in part because of the fact that the parameters are not *well behaved*. That is, small changes in the arrangement of the links in the kinematic chain can lead to large changes in some of the parameters. Consider, for example, two neighboring joint axes that are almost parallel. In determining the kinematic parameters, one has to find the shortest line connecting the two axes. The place where the two axes approach the closest, however, moves rapidly when the angle between the axes is changed. It can easily run off to infinity in one direction, only to make its appearance again at infinity in the opposite direction.

This kind of problem tends not to arise when a random arrangement of links and joints is assembled. It is certain to occur, however, with near-ideal chains, since these represent small departures from arms where axes are exactly parallel, exactly at right angles or intersect. Unfortunately, it is these near-ideal chains that we are most interested in. This means that we should base the least-squares method on other sets of parameters than those occuring in the Denavit & Hartenberg formulation.

Another problem is that the Denavit & Hartenberg notation uses four parameters per joint. This is just right for revolute joints as we shall see, but it is too much for a prismatic joint. The notation in this case is redundant. Additional constraints must be introduced to force uniqueness. This can be done, but complicates the least-squares procedure.

Number of Degrees of Freedom

A kinematic chain with n_p prismatic joints and n_r revolute joints is described by n_p unit vectors parallel to the motions of the prismatic joints and n_r unit vectors parallel to the joint axes of the revolute joints, as well as $(n_r + 1)$ offset vectors. It thus may seem that it takes $(3n_p + 6n_r + 3)$ parameters to fully specify such a kinematic chain. Note, however, that the vectors constituting the description have to satisfy the constraints

$$\hat{\mathbf{o}}_i^* \cdot \hat{\mathbf{o}}_i^* = 1, \quad \hat{\boldsymbol{\omega}}_i^* \cdot \hat{\boldsymbol{\omega}}_i^* = 1 \quad \text{and} \quad \mathbf{d}_i^* \cdot \hat{\boldsymbol{\omega}}_i^* = 0,$$

for $i = 1, 2, \ldots n$ where n is the number of links in the chain. There are $n_p + 2n_r$ constraints, so the number of degrees of freedom is actually

only

$$(3n_p + 6n_r + 3) - (n_p + 2n_r) = 2n_p + 4n_r + 3.$$

Alternatively, note that a unit vector defines a direction, and this can be specified by a point on a unit sphere. Thus a unit vector has but two degrees of freedom. Furthermore, the offsets have to be perpendicular to the following joint axis and thus also have only two degrees of freedom. The last offset is not so constrained and thus has a full three degrees of freedom. The total number of degrees of freedom is thus again seen to be

$$2n_p + 2n_r + 2n_r + 3 = 2n_p + 4n_r + 3.$$

A general-purpose kinematic chain in three dimensions must have six joints (at least three of which have to be revolute). A chain with three prismatic joints and three revolute joint is thus described by 21 paramaters, while an all-revolute arm has 27 parameters. The number of paramaters is reduced if additional assumptions are made, such as that the first axis passes through the origin or that the tool reference point lies on the last axis. Such additional constraints are of little interest in practice, however, since manufacturing variations will ensure that the first and last offsets cannot be assumed to be equal to zero without introducing some error in the kinematic solution.

Least-Squares Approach

Let the forward kinematics be represented by the vector-valued function

$$\mathbf{x} = \mathbf{f}(\mathbf{t}; \mathbf{p}),$$

where $\mathbf{t}$ is the vector of joint variables, $\mathbf{p}$ is the vector of kinematic parameters and $\mathbf{x}$ is the vector of measured components of the position and orientation of the last link in the chain[2]. The task is to determine the parameter vector $\mathbf{p}$, given m corresponding pairs of joint variable vectors $\mathbf{t}_i$ and measurement vectors $\mathbf{x}_i$. In the absence of measurement error and other disturbing effects, such as bending of the links, we would expect to be able to find a parameter vector $\mathbf{p}$ such that

$$\mathbf{x}_i = \mathbf{f}(\mathbf{t}_i; \mathbf{p}),$$

for $i = 1, 2, \ldots m$. In practice, we instead minimize the sum of squares of the residuals

$$\mathbf{e}_i = \mathbf{x}_i - \mathbf{f}(\mathbf{t}_i; \mathbf{p}).$$

[2] In what follows the vector $\mathbf{x}$ need only contain the components of the position and orientation of the final link actually measured in the calibration process.

That is, we minimize

$$E = \sum_{i=1}^{m} \mathbf{e}_i^T \mathbf{e}_i$$

by suitable choice of $\mathbf{p}$. This is an unconstrained minimization problem, subject to the usual numerical methods, as long as we choose a set of parameters that is not redundant. As mentioned earlier, however, there are definite advantages to the use of redundant parameters. Using the parameters introduced in this paper, for example, the minimization is constrained by the conditions

$$\hat{\mathbf{o}}_i^* \cdot \hat{\mathbf{o}}_i^* = 1, \quad \hat{\boldsymbol{\omega}}_i^* \cdot \hat{\boldsymbol{\omega}}_i^* = 1 \quad \text{and} \quad \mathbf{d}_i^* \cdot \hat{\boldsymbol{\omega}}_i^* = 0.$$

Gradient Methods

Many methods for non-linear optimization use the gradient of the function to be extremized. In the absence of constraints on the parameters we could simple set the derivative of the sum of squares of the errors equal to zero, that is,

$$\frac{\partial E}{\partial \mathbf{p}} = 0,$$

where the derivative of E with respect to the vector $\mathbf{p}$ is simply the vector whose components are the derivatives of E with respect to the components of $\mathbf{p}$. From this we obtain

$$\sum_{i=1}^{m} J_p^T \mathbf{e}_i = 0,$$

where

$$J_p^T = \frac{\partial \mathbf{f}^T}{\partial \mathbf{p}},$$

is a matrix whose rows are the derivatives of the row-vector $\mathbf{f}^T$ with respect to the components of $\mathbf{p}$. This leads to

$$\sum_{i=1}^{m} J_p^T \mathbf{x}_i = \sum_{i=1}^{m} J_p^T \mathbf{f}(\mathbf{t}_i; \mathbf{p}).$$

It is hard to proceed further without additional information about the structure of the forward kinematic function, $\mathbf{f}$, since this equation is likely to be highly non-linear. If we suppose, however, that we have a good approximation of the parameter vector, then we can use Taylor series expansion to linearize the equation locally[3]. Ignoring higher order terms, we have

$$\mathbf{f}(\mathbf{t}; \mathbf{p} + \delta\mathbf{p}) = \mathbf{f}(\mathbf{t}; \mathbf{p}) + J_p \, \delta\mathbf{p}.$$

[3]The equations are already linear in the offset vectors $\mathbf{d}_i*$.

Suppose that $\mathbf{p}$ is our current estimate, while $\mathbf{p}+\delta\mathbf{p}$ is the correct solution. Then

$$\sum_{i=1}^{m} J_p^T \mathbf{x}_i = \sum_{i=1}^{m} J_p^T (\mathbf{f}(\mathbf{t};\mathbf{p}) + J_p\, \delta\mathbf{p}),$$

which yields

$$\left(\sum_{i=1}^{m} J_p^T J_p \right) \delta\mathbf{p} = -\sum_{i=1}^{m} J_p^T \mathbf{e}_i.$$

These are the familiar *normal equations* for a linearized least-squares problem. The solution of the original problem may be found iteratively by means of this equation, with the Jacobian recomputed using the latest estimates of the optimal values of the unkown parameters.

Parameter Space and Constraints

Unfortunately, we are not dealing with an unconstrained minimization problem, so the iterative method suggested in the previous section is not directly applicable. We could introduce the constraints on $\hat{\mathbf{o}}_i^*$, $\hat{\boldsymbol{\omega}}_i^*$ and $\mathbf{d}_i^*$ using Lagrange multipliers, or at least add penalty terms that increase rapidly as one departs from the conditions

$$\hat{\mathbf{o}}_i^* \cdot \hat{\mathbf{o}}_i^* = 1, \quad \hat{\boldsymbol{\omega}}_i^* \cdot \hat{\boldsymbol{\omega}}_i^* = 1 \quad \text{and} \quad \mathbf{d}_i^* \cdot \hat{\boldsymbol{\omega}}_i^* = 0.$$

A more viable alternative is to always remain in the allowable subspace of parameters by only exploring changes in the parameters that are guaranteed not to violate the constraints. This is easy to do in the case of prismatic joints, one merely has to pick two directions orthogonal[4] to the unit vector $\hat{\mathbf{o}}_i^*$. Things are a little harder in the case of revolute joints. Focusing attention on one joint at a time, we see that we need four different ways of modifying the six numbers in the vectors $\hat{\boldsymbol{\omega}}_i^*$ and $\mathbf{d}_i^*$ in such a fashion that the constraints remain satisfied.

Feasible Variations of Parameters

The parameters can be changed as follows without violating the constraints:

---

[4] Things become even simpler if one wishes to allow for uncertainty in scaling, as well as the zero offset of the prismatic joint variable, since then the vector $\mathbf{o}_i^*$ in the direction of joint extension need not be unit vector and so $\mathbf{o}_i^*$ can be varied arbitrarily.

- The offset $\mathbf{d}_i$ can be rotated about the axes $\hat{\boldsymbol{\omega}}_i$. It is clear that it will remain orthogonal to the next axis. Here
$$\mathbf{d}_i' = \cos\alpha_1\,\mathbf{d}_i + \sin\alpha_1\,(\hat{\boldsymbol{\omega}}_i \times \mathbf{d}_i).$$
- The axis $\hat{\boldsymbol{\omega}}_i$ can be rotated about the offset $\mathbf{d}_i$. The orthogonality is preserved, as is the length of the axis vector. Here
$$\hat{\boldsymbol{\omega}}_i' = \cos\alpha_2\,\hat{\boldsymbol{\omega}}_i + \sin\alpha_2\,(\hat{\mathbf{d}}_i \times \hat{\boldsymbol{\omega}}_i).$$
- The offset $\mathbf{d}_i$ and the axis $\hat{\boldsymbol{\omega}}_i$ can be rotated together about a vector orthogonal to both. The lengths of the two vectors, as well as the angle between them is preserved by this operation. Here
$$\mathbf{d}_i' = \|\mathbf{d}_i\|\,(\cos\alpha_3\,\hat{\mathbf{d}}_i + \sin\alpha_3\,\hat{\boldsymbol{\omega}}_i)$$
while
$$\hat{\boldsymbol{\omega}}_i' = (\cos\alpha_3\,\hat{\boldsymbol{\omega}}_i - \sin\alpha_3\,\hat{\mathbf{d}}_i).$$
- The offset $\mathbf{d}_i$ can be extended along its length. Since the new offset points in the same direction as the old one, it will still be orthogonal to the axis of the next joint. Here
$$\mathbf{d}_i' = e^{\alpha_4}\,\mathbf{d}_i.$$

Note that we did not have to assume that the variations in $\hat{\boldsymbol{\omega}}_i$ and $\mathbf{d}_i$ are infinitesimal; the constraints remain satisfied for finite α_1, α_2, α_3, and α_4.

There is a problem with the scheme as presented so far in the unusual situation when $\mathbf{d}_i$ has zero length. This can be handled by generating an arbitary direction, $\mathbf{s}$ say, orthogonal to $\hat{\boldsymbol{\omega}}_i$. This direction can then be used instead of $\mathbf{d}_i$ to generate the three variations of $\hat{\boldsymbol{\omega}}_i$ above. Finnally, $\mathbf{d}_i$ can be extended in a direction orthogonal to both $\hat{\boldsymbol{\omega}}_i$ and $\mathbf{s}$ without violating the constraints, in order to generate the fourth needed variation of the parameters.

Infinitesimal Analysis of Parameter Changes

While the scheme above ensures that new parameter values generated satisfy the required constraints for any α_1, α_2, α_3, and α_4, in practice the changes chosen will be small in order to satisfy the assumption underlying the linearization of the least-squares problem. It is instructive to note how $\mathbf{d}_i$ and $\hat{\boldsymbol{\omega}}_i$ change when α_1, α_2, α_3, and α_4, are infinitesimal. We have for the four changes in the parameters introduced above:

$$\delta\mathbf{d}_i = \delta\alpha_1\,(\hat{\boldsymbol{\omega}}_i \times \mathbf{d}_i) \qquad \text{and} \qquad \delta\hat{\boldsymbol{\omega}}_i = \mathbf{0}.$$

$$\delta\mathbf{d}_i = \mathbf{0} \qquad \text{and} \qquad \delta\hat{\boldsymbol{\omega}}_i = \delta\alpha_2\,(\hat{\mathbf{d}}_i \times \hat{\boldsymbol{\omega}}_i).$$

$$\delta \mathbf{d}_i = \delta\alpha_3 \, \|\mathbf{d}_i\| \, \hat{\boldsymbol{\omega}}_i \qquad \text{and} \qquad \delta \hat{\boldsymbol{\omega}}_i = -\delta\alpha_3 \, \hat{\mathbf{d}}_i.$$

$$\delta \mathbf{d}_i = \delta\alpha_4 \, \mathbf{d}_i \qquad \text{and} \qquad \delta \hat{\boldsymbol{\omega}}_i = \mathbf{0}.$$

Note that the three incremental changes in $\mathbf{d}_i^*$ are orthogonal to one another. Also, the two incremental changes in $\hat{\boldsymbol{\omega}}_i^*$ are orthogonal to one another, as well as to $\hat{\boldsymbol{\omega}}_i^*$. The four variations indroduced above thus correspond to orthogonal directions in the space of feasible solutions. They yield a useful non-redundant local coordinate system.

Constrained Optimization

We can use the unconstrained least-squares method discussed above directly if we consider variations in the paramater vector not to be independent variations of $\hat{\mathbf{o}}_i^*$, $\mathbf{d}_i^*$ and $\hat{\boldsymbol{\omega}}_i^*$, but instead variations in the orthogonal directions in parameter space just described. While there are $(3n_p + 6n_r + 3)$ parameters, there are only $(2n_p + 4n_r + 3)$ directions to consider. The derivatives occuring in J_p^T are now not to be taken with respect to the original parameters of the kinematic chain, but with respect to the new parameters α_1, α_2, α_3 and α_4. In practice these derivatives can be estimated numerically by recomputing the forward kinematic solution for small changes in the corresponding directions in parameter space[5].

Obtaining a Good Initial Guess

The iterative method presented above requires an initial guess to get started. Typically a kinematic chain will be constructed according to some design. The parameters extracted from the design plan provide good initial guesses for the parameters of the actual arm.

Another alternative is to use a method that finds one axis at a time, starting in the home position. If the joint extension of a prismatic joint is varied, a point on the chain further from the base than this joint will trace out a straight line in space. This line will be parallel to the direction of extension of the joint. A line can be fitted to the positions of the point on the kinematic chain and its direction used as an estimate of the prismatic joint direction $\hat{\mathbf{o}}_i^*$.

[5] As usual, when estimating derivatives numerically, the increment has to be chosen carefully to reach a satisfactory compromise between round-off errors in the computer arithmetic when the step-size is too small and truncation errors when the step-size is too large.

Similarly, if the joint angle of a single revolute joint is varied, an point on the chain further from the base than this joint will trace a circle in space. This circle will lie in a plane perpendicular to the joint axis and the joint axis will pass through the center of the circle. Fitting a circle to the positions of the point on the kinematic chain as one joint angle is varied thus allows one to identify a point on the joint axis, as well as the direction of the axis, $\hat{\boldsymbol{\omega}}_i^*$.

Collecting information in this way about each joint in turn allows one to estimate the parameters of the kinematic chain. While the estimates are likely not to be good enough for use in the inverse kinematic solution program, they will be adequate as starting values for the iterative method described above for finding the least-squares solution for the parameters.

Kinematics of Non-Ideal Serial Chains

There are several ways of iteratively computing the solution of a non-ideal kinematic chain. These differ in complexity and computational efficiency.

Method of False Position

Suppose that $\mathbf{p}$ is the parameter vector of the ideal kinematic chain, while $\mathbf{p}'$ is that of the actual (non-ideal) chain. A closed form solution is available for the inverse kinematic of the ideal chain. It is also straight-forward to compute the forward kinematics for the non-ideal chain. An iterative scheme can be based on these two procedures. Suppose that we wish to place the terminal device in a position and orientation specified by the vector $\mathbf{x}$. The iteration is based on a modified or "false" goal position $\mathbf{x}_i$ for the ideal chain. Let the solution of the inverse kinematics of the ideal chain for this goal position $\mathbf{x}_i$ be $\mathbf{t}_i$, that is,

$$\mathbf{x}_i = \mathbf{f}(\mathbf{t}_i; \mathbf{p}).$$

The position and orientation of the actual chain given this joint angle vector is

$$\mathbf{x}_i' = \mathbf{f}(\mathbf{t}_i; \mathbf{p}').$$

This differs from the desired position by $\delta\mathbf{x}_i = (\mathbf{x}_i' - \mathbf{x})$. The notion behind this method is that we can get the actual arm closer to the correct position by pretending that we want the ideal chain to go to the modified goal position

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \delta\mathbf{x}_i$$

instead of $\mathbf{x}$. The iteration proceeds by solving the inverse kinematics of the ideal arm for this new goal position.

If the problem was linear, the exact solution would be found in one step. Since it is not linear, one only comes closer to the exact solution. The iteration can be started by letting $\mathbf{x}_1 = \mathbf{x}$. The iteration converges rapidly, provided that the parameters of the actual arm are close to those of the ideal arm. Even a single iteration provides accuracy in positioning significantly better than that to be expected if the inverse kinematic solution of the ideal arm is used directly.

Some trigonometric function evaluation can be avoided if one notes that the forward kinematics depends only on the cosine and sines of the joint angles. That is, one need not calculate the joint angles themselves (until the final step of the iteration). In the solution of the inverse kinematics of the ideal chain it is sufficient to determine the cosines and sines of the joint angles.

Definition of the Workspace

The *work-space* of a kinematic chain is the part of the space of positions and orientations that can be attained by the final link[6]. Note that this space has a higher dimension than the space in which the kinematic chain is embedded and that it may have a non-trivial topology[7].

For a kinematic chain in a plane, for example, the work-space is three-dimensional. Two of the dimensions correspond to the position of the reference point in the final link, while the third one corresponds to the direction in which the final link points. This third dimension "wraps around," since adding 2π to the orientation of the final link yields the same position and orientation. If there are three revolute joints, the work-space is cylindrical, with opposite faces of the cylinder identified.

For a kinematic chain in three-dimensional space, the work-space, as defined here, has six dimensions, three for position and three for orientation. The three dimensions for orientation also "wrap around" and can best be thought of in terms of the surface of a unit sphere in four dimensions.

Various projections and slices of the work-space that are easier to visualize are commonly used. For example, manufacturers often specify a

[6] In the discussion here we ignore mechanical limits on joint angles and joint extensions.

[7] The joint-variable space too has an interesting topology, since it "wraps around" in the direction of each of the joint variables corresponding to a revolute joint.

weak projected work-space, that is, the set of points reachable using some orientation of the final link. A more useful projection is a *strong projected work-space*, defined to be that set of points that can be reached using any orientation of the final link. The latter is obviously much smaller than the former (and in some cases is actually empty!).

Near the Boundary of the Workspace

The mapping from the space of joint variables to the work-space is many-to-one. That is, several different sets of joint variables will yield the same position and orientation of the final link. Different sets of joint variables yielding the same position and orientation are said to correspond to different arm *configurations*. One can think of this in terms of Riemann sheets, with the folds in the sheets on the boundary of the work-space. The superimposed sheets at any given point in the work-space correspond to different arm-configurations. The boundary of the work-space is where the number of solutions is reduced. The configuation of an arm can only be changed on the boundary of the work-space. *Singularities* occur on the boundaries of the work-space. These are places where finite velocities in the work-space require infinite velocities in some of the joint variables. This suggests that the iterative methods described above are likely to require a large number of iterations in parts of the work-space that are close to the boundary.

In addition, the inverse kinematic solution method presented above will not work at all in two cases:

- When the desired position and orientation is outside the work-space of the actual arm (but inside that of the ideal arm).
- When the desired position and orientation is outside the work-space of the ideal arm (but inside that of the actual arm).

In the first case, an inverse kinematic solution exist for the ideal chain, but none for the actual one, so the iteration cannot converge. In the second case, no inverse kinematic solution is found for the ideal arm, and so no starting values are available for the iteration. Because of these problems, and the need to stay away from singularities, it is suggested that positions and orientations near the boundary of the work-space not be used.

Summary and Conclusions

The new notation for describing serial kinematic chains with revolute joints has been shown to be unique, unambiguous, simple to determine, easy to use and well-behaved when small changes are made in the arrangement of the elements of the chain. It has also been shown that the notation has advantages when used in least-squares calibration procedures designed to recover the parameters describing the kinematic chain. Methods for computing the forward kinematics have been presented and an iterative method given for solving the inverse kinematics of near-ideal kinematic chains.

References

Brady, Michael, John M. Hollerbach, Timothy L. Johnson, Tomás Lozano-Pérez, & Matthew T. Mason (1982) *Robot Motion—Planning and Control*, MIT Press, Cambridge, Massachusetts.

Craig, John J. (1986) *Introduction to Robots—Mechanics & Control* Addison-Wesley Publishing, Reading, Massachusetts.

Denavit, J. & R.S. Hartenberg (1955) "A Kinematic Notation for Lower-Pair Mechanisms based on Matrices," *ASME Journal of Applied Mechanics*, pp. 215–221, June.

Hartenberg, R. S. & J. Denavit (1964) "Kinematic Synthesis of Linkages," McGraw-Hill, New York.

Deist, F. H. & L. Sefor (1967) "Solution of Systems of Non-Linear Equations by Parameter Variation," *The Computer Journal*, Vol. 10, No. 1, PP. 78-82, May.

Gupta, K. C. (1986) "Kinematic Analysis of Manipulators using the Zero Reference Position Description," *The International Journal of Robotics Research*, Vol. 5, No. 2, Summer.

Hollerbach, J. M. & G. Sahar (1983) "Wrist-partitioned inverse kinematic acceleration and manipulator control," *International Journal of Robotics Research*, Vol. 2, No. 4, pp. 61–76.

Horn, B.K.P. & Hirochika Inoue (1974) "Kinematics of the MIT-AI-VICARM Manipulator," MIT AI Working Paper 69, May.

Kahn, M.E. & B. Roth (1971) "The Near-Minimum-Time Control of Open-Loop Kinematic Chains," *Transactions of the ASME*, Series G, Vol. 93, pp. 164–172.

Paul, Richard P. (1972) "Modeling, Trajectory Calculation and Servoing of a Computer-Controlled Arm," Stanford AI Memo 177.

Paul, Richard P. (1981) *Robot Manipulators—Mathematics, Programming & Control*, MIT Press, Cambridge, Massachusetts.

Pieper, Donald Lee (1968) "The Kinematics of Manipulators under Computer Control," Stanford AI Memo 72.

Uicker, J.J. Jr. (1965) "On the Dynamic Analysis of Spatial Linkages using 4×4 Matrices," Ph.D. Dissertation, Northwestern University, Evanston, Illinois, August.

Uicker, J.J. Jr. (1967) "Dynamic Force Analysis of Spatial Linkages," *Transactions of the ASME*.

Appendix

Notation for Rotation

The transformations between the coordinate systems of the links in the chain are composed of translations and rotations. Few would argue that vectors are the appropriate way of representing the translational offsets. Things are not so clear when it comes to methods for representing rotation. The most commonly used notation for rotation in robotics and computer graphics is the orthonormal matrix with positive determinant. A vector is rotated simply by multiplying the matrix by the vector, and rotations are composed by multiplying the corresponding matrices. In dealing with the kinematics of serial chains, other notations also have their attractions.

Since the axes of the revolute joints and their angles of rotation are known, it is natural to consider the axis-and-angle notation, for example. Suppose the rotation is by an angle θ about an axis through the origin with direction specified by the unit vector $\hat{\boldsymbol{\omega}}$. Then Rodrigues's formula tells us that the vector $\mathbf{x}$ is rotated into

$$\mathbf{x}' = \mathbf{x}\cos\theta + (\hat{\boldsymbol{\omega}} \cdot \mathbf{x})\,\hat{\boldsymbol{\omega}}\,(1 - \cos\theta) + (\hat{\boldsymbol{\omega}} \times \mathbf{x})\sin\theta.$$

A slight savings in the computation effort is achieved if we use the equivalent form

$$\mathbf{x}' = \mathbf{x} + (\hat{\boldsymbol{\omega}} \times \mathbf{x}) \sin\theta + \hat{\boldsymbol{\omega}} \times (\hat{\boldsymbol{\omega}} \times \mathbf{x}) (1 - \cos\theta),$$

since the cross-product of $\hat{\boldsymbol{\omega}}$ and $\mathbf{x}$ is re-used. Unfortunately, there is no simple formula for composition of rotations using the axis-and-angle notation, so it is not directly useful in dealing with kinematic chains.

A related notation does, however, does yield a simple formula for composition. A quaternion $\mathring{\mathrm{q}}$ can be thought of as composed of a scalar part q_0 and a vector part $\mathbf{q}$. Multiplication of the quaternions $\mathring{\mathrm{p}}$ and $\mathring{\mathrm{q}}$ to produce the quaternion $\mathring{\mathrm{r}}$ can be defined in terms of the scalar and vector parts of the quaternions. If

$$(r, \mathbf{r}) = (p, \mathbf{p})\,(q, \mathbf{q})$$

then

$$r = pq - \mathbf{p} \cdot \mathbf{q} \qquad \text{and} \qquad \mathbf{r} = p\,\mathbf{q} + q\,\mathbf{p} + \mathbf{p} \times \mathbf{q}.$$

A quaternion can also be considered as composed of a real part and three different kinds of imaginary parts. These four numbers together are known as Euler's parameters.

A rotation through an angle θ about an axis given by the unit vector $\hat{\boldsymbol{\omega}}$ can be represented by a quaternion $\mathring{\mathrm{q}}$ of unit magnitude, whose scalar part q is $\cos(\theta/2)$ and whose vector part $\mathbf{q}$ equals $\sin(\theta/2)\hat{\boldsymbol{\omega}}$. That is,

$$\mathring{\mathrm{q}} = \left(\cos\frac{\theta}{2}, \sin\frac{\theta}{2}\hat{\boldsymbol{\omega}}\right).$$

A vector is rotated according to the rule

$$\mathring{\mathrm{x}}' = \mathring{\mathrm{q}}\,\mathring{\mathrm{x}}\,\mathring{\mathrm{q}}^*,$$

where $\mathring{\mathrm{q}}^*$ is the conjugate of $\mathring{\mathrm{q}}$, obtained by negating the vector part. Here $\mathring{\mathrm{x}}$ and $\mathring{\mathrm{x}}'$ are purely imaginary quaternions whose vector parts are equal to $\mathbf{x}$ and $\mathbf{x}'$ respectively. It follows from the above that composition of rotation corresponds to multiplication of unit quaternions. The formula for rotation of a vector can also be written explicitly in terms of the scalar and vector parts of the unit quaternion $\mathring{\mathrm{q}}$ as follows:

$$\mathbf{x}' = (q^2 - \mathbf{q} \cdot \mathbf{q})\,\mathbf{x} + 2\,(\mathbf{x} \cdot \mathbf{q})\,\mathbf{q} + 2\,q(\mathbf{q} \times \mathbf{x}).$$

It is important to realize that the choice of a notation for rotation is independent of the choice of a notation for the kinematic chain. The unit quaternion notation for rotation does, however, have a number of advantages that one may wish to exploit.

Computational Effort

The orthonormal matrix provides the most efficient representation when it comes to rotating a vector. It takes just 9 multiplications and 6 addi-

tions (3 dot-products). It is relatively expensive to rotate a vector using unit quaternions. The obvious algorithm, based on the formula given above, requires 22 multiplications and 14 additions[8]. In the computation of the direct kinematics, however, one not only has to rotate vectors, but also compose rotations. The orthonormal matrix is at a disadvantage here, since it takes 27 multiplications and 18 additions (9 dot-products), while multiplication of quaternions requires only 16 multiplications and 12 additions.

The forward kinematic computation involves one rotation of an offset vector and one composition of rotations per revolute axis. It thus takes 38 multiplications and 26 additions per revolute joint using unit quaternions, while orthonormal matrices require 36 multiplications and 24 additions. It thus appears that there is not much difference between the two notations if we consider only computational effort.

In the above analysis we have assumed the obvious implementations of the formulae for rotation of a vector and for composition of rotations. The computational effort can be reduced a little by more careful attention to the details. In the case of the multiplication of orthonormal matrices, for example, we can make use of the fact that the third column of the result must be orthogonal to the first two and that it must have unit magnitude. It can be computed from the first two columns by means of a cross-product. This reduces the effort to 24 multiplications and 15 additions/subtractions.

Similarly, the rotation of a vector using unit quaternions can be reduced to just 15 multiplications and 12 additions[9], using the formula

$$\mathbf{x}' = \mathbf{x} + 2\,q(\mathbf{q} \times \mathbf{x}) + 2\,\mathbf{q} \times (\mathbf{q} \times \mathbf{x}),$$

where the cross-product of $\mathbf{q}$ and $\mathbf{x}$ is reused. So it takes 31 multiplications and 24 additions per revolute joint using unit quaternions, while orthonormal matrices require 33 multiplications and 21 additions.

Once again we see that there is not a significant advantage to either of the two notations for rotations in terms of computational effort. Other considerations must motivate the choice of one over the other alternatives. It is also important to point out, once again, that the choice of a notation for rotation is independent of the choice of a notation for the kinematic chain.

---

[8] Here we count a multiplication by two as an addition.

[9] Again counting a multiplication by two as an addition.