Using Interval Methods for Solving Inverse Kinematic Problems

Albert Castellet and Federico Thomas

Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Gran Capità 2-4, 08034 Barcelona, Spain. E-mail: acastellet@iri.upc.es and fthomas@iri.upc.es.

Summary. In this paper we present an algebraic analysis of the closure equation of arbitrary single loop spatial kinematic chains, which allows us to use an interval method based on interval cuts for solving their inverse kinematics.

The solution of a kinematic equation can be factored into a solution of both its rotational and its translational components. We have obtained general and simple expressions for these equations that are used to perform cuts. A branch and prune strategy can be used to get a set of boxes as small as desired containing the inverse kinematic solutions. If the kinematic chain is redundant, this approach can also provide a discretized version of the solution set.

The mathematics of the proposed approach are quite simple and much more intuitive than continuation or elimination methods. Yet it seems to open a promising field for further developments.

1. Introduction

Solving loops of kinematic constraints [11] is a basic requirement when dealing with inverse kinematics, task-level robot programming, assembly planning, or constraint-based modeling problems. This problem is difficult due to its inherent computational complexity (i.e., it is NP-complete) and due to the numerical issues involved to guarantee correctness and to ensure termination.

Two basic approaches have been used for solving this problem: continuation and elimination methods [10, 7]. The former is based upon homotopy techniques to solve a system of polynomial equations [13]. They compute the solutions of the algebraic system by following paths in the complex space. They are robust but slow. The latter approach is based on an algebraic formulation, eliminating variables from a system of equations, and is used along with algorithms for finding roots of univariate polynomials [9]. They can be slow because of symbolic expansion and usually do not work for kinematic chains with special geometries.

Recently, interval methods for solving systems of non-linear equations have attracted much attention and have been explored by a variety of authors [4]. They have already been used to solve some kinematic problems proving to be robust but sometimes slow compared to continuation methods [5].

We have developed an algebraic analysis that allows us to obtain the closure equations of an arbitrary kinematic chain in a well-conditioned way so that they can be readily used by an interval method [2]. The method adopted in our experiments consist in improving the known bounds on the

possible solutions using a set of inference rules called *interval cuts* as defined in [5].

In our case, the interval method receives a box, i.e. an interval tuple of the variables of rotation and translation, specifying the initial bounds. Then, it returns a set of boxes containing the different solutions. When the kinematic chain is redundant, it is also shown how this method is able to provide a discretized version of the underlying self-motion manifold. These ideas are beening implemented using BIAS-PROFIL [6], a portable C++ class library.

This paper is structured as follows. Section 2 describes the used formulation for the closure equations of an arbitrary kinematic chain. Section 3 gives an introduction to interval methods and in particular to those involving *interval cuts*. In Section 4, our specific cuts are extensively explained. We conclude in Section 5.

2. Background

Any closed kinematic chain can be described as a circular list of screws $X_1, X_2, \ldots, X_n$, each one being orthogonal to the next one, so that its configuration is determined by the angles ϕ_i around X_i and the offsets d_i along X_i. Then, its associated loop equation can be expressed as

$$\prod_{i=1}^{n} \mathbf{T}(d_i)\mathbf{R}(\phi_i)\mathbf{Z} = \mathbf{I}, \tag{2.1}$$

where $\mathbf{T}(d_i)$ stands for a translation along the x-axis, $\mathbf{R}(\phi_i)$, a rotation around the x-axis and $\mathbf{Z}$, a rotation of $\pi/2$ radians around the z-axis. This equation corresponds to the loop equation of what in [12] is called the *n-bar mechanism* (Figure 2.1).

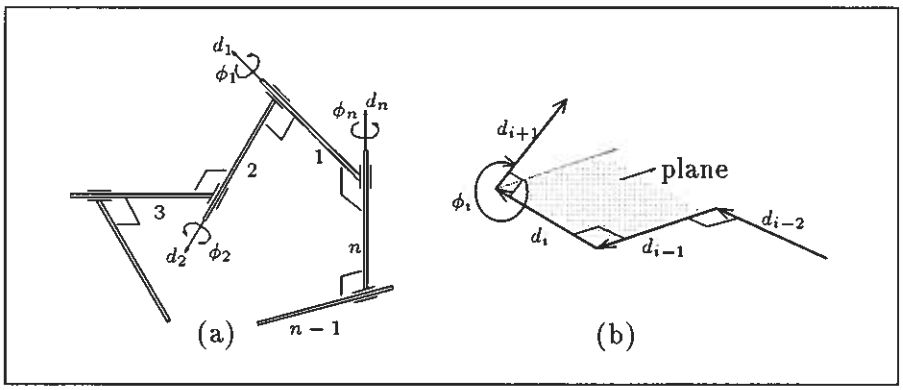

Fig. 2.1. The n-bar mechanism (a) and definitions of the involved degrees of freedom (b).

If we rename $\alpha_{(i-1)/2} = \phi_i + \pi$ and $a_{(i-1)/2} = d_i$ when i is odd and $\theta_{i/2} = \phi_i + \pi$ and $t_{i/2} = d_i$ when i is even, α_i, a_i, θ_i and t_i are the Denavit-Hartenberg parameters of the mechanism, where the odd bars correspond to links and the even bars to joints [3](Figure 2.2). Therefore, any single loop mechanism with $n/2$ links can be represented by an n-bar mechanism by restricting some of its degrees of freedom.

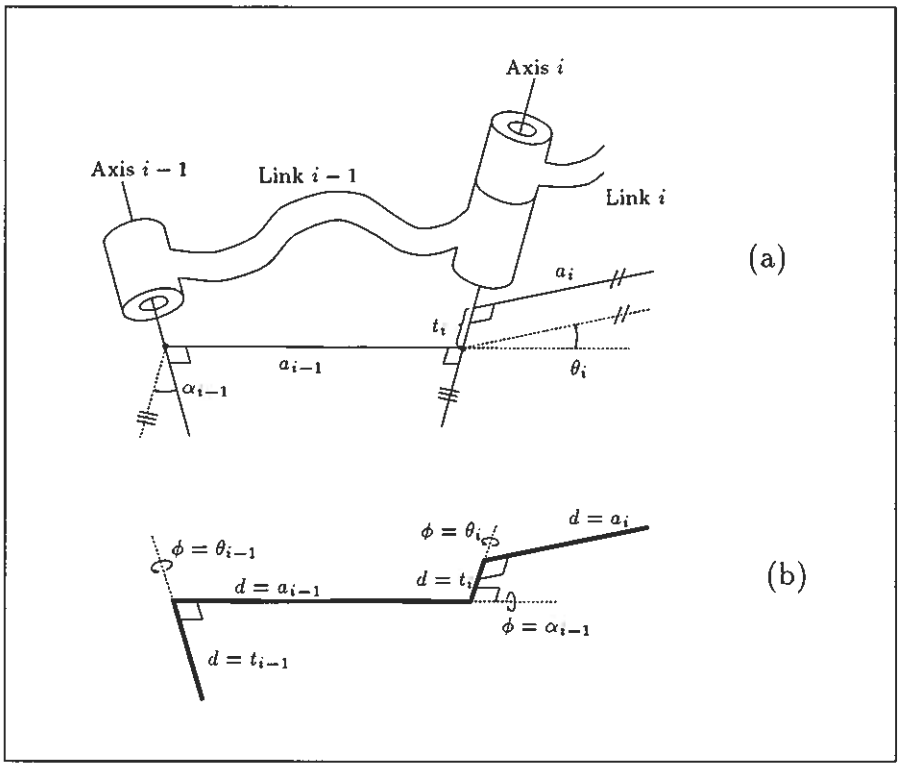

Fig. 2.2. The Denavit-Hartenberg parameters of a mechanism (a) and the corresponding ones in an n-bar mechanism (b).

Three bars are enough to reach any point in 3D space with arbitrary orientation. Then, if the mechanism described by the Denavit-Hartenberg parameters is not closed (e.g. a manipulator), we can always close the associated n-bar mechanism with three bars, representing the position of the last bar (the end-effector in a manipulator) with respect to the base.

As an example, the Denavit-Hartenberg parameters of the PUMA 560 are those of Table 2.. The associated n-bar mechanism will have the vectors of rotations and translations of Table 2..

ϕ_{13}, ϕ_{14}, ϕ_{15}, d_{13}, d_{14} and d_{15} are used to represent the end-effector's position with respect to the robot base.

Table 2.1. Denavit-Hartenberg parameters of the PUMA 560.

i	α_{i-1}	a_{i-1}	t_i	θ_i
1	0	0	0	θ_1
2	$-\pi/2$	0	0	θ_2
3	0	a_2	t_3	θ_3
4	$-\pi/2$	a_3	t_4	θ_4
5	$\pi/2$	0	0	θ_5
6	$-\pi/2$	0	0	θ_6

Table 2.2. The vectors of rotations and translations of an n-bar mechanism respresenting the PUMA 560.

i	1	2	3	4	5	6	7	8	9	10
ϕ	π	$\theta_1+\pi$	$\pi/2$	$\theta_2+\pi$	π	$\theta_3+\pi$	$\pi/2$	$\theta_4+\pi$	$-\pi/2$	$\theta_5+\pi$
d	0	0	0	0	a_2	t_3	a_3	t_4	0	0

i	11	12	13	14	15
ϕ	$\pi/2$	$\theta_6+\pi$	ϕ_{13}	ϕ_{14}	ϕ_{15}
d	0	0	d_{13}	d_{14}	d_{15}

Equation (2.1) can be factored into the following two equations [12]:

$$\mathbf{F}(\boldsymbol{\phi}) = \prod_{i=1}^{n} \mathbf{R}(\phi_i)\mathbf{Z} = \mathbf{I} \tag{2.2}$$

and

$$\mathbf{T}(\boldsymbol{\phi}, \mathbf{d}) = \sum_{i=1}^{n} \mathbf{A}_1^{i-1}(\boldsymbol{\phi}) \begin{pmatrix} d_i \\ 0 \\ 0 \end{pmatrix} = \mathbf{0} \ , \tag{2.3}$$

where

$$\mathbf{A}_k^l(\boldsymbol{\phi}) \triangleq \begin{cases} \mathbf{I} & k = l+1 \\ \prod_{j=k}^{l} \mathbf{R}(\phi_i)\mathbf{Z} & k \leq l \ . \end{cases}$$

Equations (2.2) and (2.3) are called the *rotation and translation equations*, respectively.

The rotation equation can be extracted directly from the original loop equation by simply removing all translations. It only assures that the final orientation of the chain is the same as the first one, without constraining the translation values. The translation equation, however, involves both translations and rotations. It states that the chain really closes, i.e. that the last bar ends at the beginning of the first one without constraining its orientation.

Thus, the solutions to both the rotation equation (2.2) and the translation equation (2.3) are the solutions to the loop equation (2.1).

3. Interval methods

Interval methods manipulate upper and lower bounds on variables and are based on interval arithmetic [4]. They have been used to solve systems of non-linear equations, global optimization problems and to avoid rounding errors due to floating-point representation of real numbers in computers. There are many variants of these methods; some of them consist in improving the bounds of the variables using a set of inference rules called *interval cuts* [8]. An interval cut is a procedure which operates on a set of constraints and a current box, reducing this box by deriving a new bound on one of the variables. A box can be successively reduced by applying interval cuts to all variables with all the constraining equations.

In our case, an interval method would receive a box, i.e. two interval tuples $<I_{d_1}, \ldots, I_{d_n}>$ and $<I_{\phi_1}, \ldots, I_{\phi_n}>$ specifying the initial range of the elements in $\mathbf{d}$ and $\boldsymbol{\phi}$, respectively. In general, this box is highly degenerate, since usually most of the variables are fixed by the mechanism's geometry. Only the variables corresponding to the degrees of freedom of the mechanism will vary within a range delimited by design constraints. When performing an interval cut, the box will be reduced in one direction.

After reducing the box, three possibilities arise. First, the pruning operation may have resulted in an empty box, in which case we return failure. Second, it may be the case that the interval associated with each variable has reached a width below a specified accuracy. In this case we terminate and return the box. If the pruning operation results in a box which is not of sufficient accuracy, then we can split the box and two branches are generated. Then, solutions on each branch are recursively searched. This is what in [5] is called a branch and prune strategy. This method would also be able to provide a discretized version, up to a given resolution, of the underlying self-motion manifold of a redundant mechanism (i.e., the set of all solutions [1]).

Besides the wide variety of heuristics for finding useful cuts and for determining when to branch, the key point is to efficiently generate cuts to prune the box.

In [8] and [5], three different cuts are described: the Newton cut, which is based on the natural evaluation of interval functions, the Snake cut and the Taylor cut, which are faster at the end of the process. These three cuts follow the same schema: they evaluate the constraint equation for a fixed value of the variable we want to cut. Then, by evaluating derivatives of the constraint equation, we can infer the possible range of values of these variables. The aforementioned three cuts differ in the kind of evaluation used for the equation and for its derivatives.

4. Our cuts

The constraint equations used to perform the cuts in our problem are the rotation equation (2.2) and the translation equation (2.3). It is possible to isolate the variable we want to cut and evaluate directly its range of possible values for both equations. In doing so, we are minimizing the errors of the three previous cuts, since we are evaluating directly the variable we want to cut and, thus, the errors due to the evaluation of its derivative are not introduced. This is why we often call them *exact cuts*.

In our case, three different exact cuts are possible: two cuts for the variables of rotation –derived from both the rotation and translation equation– and one cut for the variables of translation –derived from the translation equation.

4.1 Cutting ϕ_i with the rotation equation

The rotation equation (2.2) can be written as

$$\mathbf{A}_1^{i-1}(\boldsymbol{\phi})\mathbf{R}(\phi_i)\mathbf{Z}\mathbf{A}_{i+1}^{n}(\boldsymbol{\phi}) = \mathbf{I} \ .$$

In this equation, the only matrix that involves ϕ_i is $\mathbf{R}(\phi_i)$, which can be isolated as follows:

$$\mathbf{R}(\phi_i) = (\mathbf{Z}\mathbf{A}_{i+1}^{n}(\boldsymbol{\phi})\mathbf{A}_1^{i-1}(\boldsymbol{\phi}))^t \ .$$

Let us define $\mathbf{V}^i$ as:

$$\mathbf{V}^i \triangleq (\mathbf{Z}\mathbf{A}_{i+1}^{n}(\boldsymbol{\phi})\mathbf{A}_1^{i-1}(\boldsymbol{\phi}))^t \ .$$

Since $\mathbf{R}(\phi_i) = \mathbf{V}^i$, evaluating $\mathbf{V}^i$ in a box of $\boldsymbol{\phi}$ will give us all possible values of $\mathbf{R}(\phi_i)$ for values of $\boldsymbol{\phi}$ in that box. Thus, the chain can be effectively closed if

$$1 \in v_{11}^i \quad \text{and} \quad 0 \in v_{12}^i, v_{13}^i, v_{21}^i, v_{23}^i \ ,$$

where v_{jk}^i denotes the (j, k) element of matrix $\mathbf{V}^i$.

If the previous conditions hold, the possible values for ϕ_i will be:

$$\phi_i = \arccos(v_{22}^i \cap v_{33}^i) \cap \arcsin(v_{32}^i \cap -v_{23}^i) \tag{4.1}$$

The interval of possible values for ϕ_i can be cut by intersecting the initial range of ϕ_i with the interval obtained from (4.1).

Note that the arcsin function gives only values between $-\pi/2$ and $+\pi/2$. However, we need to evaluate this function in an interval and get all possible values between 0 and 2π. Note that the result encompasses up to three intervals. Something similar happens with the arccos function. Then, the intersection of intervals in (4.1) can lead to as much as four disjoint intervals. For the sake of simplicity, we take as the final interval for ϕ_i the convex hull of the resulting intervals.

4.2 Cutting ϕ_i with the translation equation

We can also isolate ϕ_i from the translation equation (2.3):

$$\sum_{k=1}^{i} \mathbf{A}_1^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} + \mathbf{A}_1^i(\phi) \left[\sum_{k=i+1}^{n} \mathbf{A}_{i+1}^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} \right] = \mathbf{0} \ .$$

Multiplying by $(\mathbf{A}_1^{i-1}(\phi))^t$ we get

$$(\mathbf{A}_1^{i-1}(\phi))^t \sum_{k=1}^{i} \mathbf{A}_1^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} + \mathbf{R}(\phi_i)\mathbf{Z} \sum_{k=i+1}^{n} \mathbf{A}_{i+1}^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} = \mathbf{0} \ .$$

Defining

$$\mathbf{w}_0^i \triangleq -(\mathbf{A}_1^{i-1}(\phi))^t \sum_{k=1}^{i-1} \mathbf{A}_1^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix}$$

and

$$\mathbf{w}_1^i \triangleq \sum_{k=i+1}^{n} \mathbf{A}_{i+1}^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} \ ,$$

we can write

$$\begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} - \mathbf{w}_0^i = \mathbf{R}(\phi_i)\mathbf{Z}\mathbf{w}_1^i \ .$$

In other words,

$$\begin{pmatrix} 0 & -1 & 0 & d_i \\ \cos\phi_i & 0 & -\sin\phi_i & 0 \\ \sin\phi_i & 0 & \cos\phi_i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} w_{11}^i \\ w_{12}^i \\ w_{13}^i \end{pmatrix} = \begin{pmatrix} w_{01}^i - d_i \\ w_{02}^i \\ w_{03}^i \end{pmatrix} \ ,$$

where w_{0j}^i and w_{1j}^i stand for the (j) element of vectors $\mathbf{w}_0^i$ and $\mathbf{w}_0^i$ respectively.

The chain can be effectively closed if

$$0 \in (w_{01}^i + w_{12}^i - d_i)$$

and the possible values for ϕ_i will be

$$\phi_i = \arcsin\left(\frac{w_{03}^i w_{11}^i - w_{13}^i w_{02}^i}{{w_{11}^i}^2 + {w_{13}^i}^2}\right) \bigcap \arccos\left(\frac{w_{02}^i w_{11}^i - w_{13}^i w_{03}^i}{{w_{11}^i}^2 + {w_{13}^i}^2}\right) \ . \quad (4.2)$$

Then, we can cut the initial interval of ϕ_i by intersecting it with the range obtained in (4.2).

In this case, two observations have to be made. First, the division of intervals can cause problems if the denominator includes the origin. In equation (4.2), the denominators are greater or equal than 0. Then the division

leads to a single interval, possibly extended to infinity. However, this does not imply any difficulty, since we have to intersect this interval with $[-1, 1]$, since the arcsin and arccos functions are only defined within this interval. Second, note that the translation equation depends on where we place the first bar of the n-bar mechanism in the chain. This can be easily seen by observing that the translation equation (??) does not involve the last angle ϕ_n. In general, the result using (4.2) will be different from the result using a translational equation where the first angle is another one. Therefore, we have n different translation equations and each angle ϕ_i can be cut using them. This was not the case with the rotation equation, since it is the same wherever we take the reference.

4.3 Cutting d_i with the translation equation

In order to isolate d_i in (2.3), we express it as follows:

$$\sum_{k=1}^{i-1} \mathbf{A}_1^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} + \mathbf{A}_1^{i-1}(\phi) \begin{pmatrix} d_i \\ 0 \\ 0 \end{pmatrix} + \sum_{k=i+1}^{n} \mathbf{A}_1^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} = 0 \ .$$

Multiplying this equation by $\left(\mathbf{A}_1^{i-1}\right)^t$, we get:

$$\begin{pmatrix} d_i \\ 0 \\ 0 \end{pmatrix} = -\left(\mathbf{A}_1^{i-1}(\phi)\right)^t \sum_{k=1}^{i-1} \mathbf{A}_1^{k-1}(\phi) \begin{pmatrix} d_i \\ 0 \\ 0 \end{pmatrix} - \sum_{k=i+1}^{n} \mathbf{A}_i^{k-1}(\phi) \begin{pmatrix} d_k \\ 0 \\ 0 \end{pmatrix} =$$

$$= \mathbf{w}_0^i - \mathbf{A}_i^i(\phi)\mathbf{w}_1^i \ .$$

Defining

$$\mathbf{w}_3^i \triangleq \mathbf{w}_0^i - \mathbf{A}_i^i(\phi)\mathbf{w}_1^i \ ,$$

the chain can be closed if

$$0 \in w_{32}^i, w_{33}^i$$

and the allowed values for d_i are:

$$d_i = w_{31}^i \ . \tag{4.3}$$

We can cut the initial interval of d_i by intersecting it with the range obtained in (4.3).

As explained in the previous subsection, we can cut d_i with n different translation equations.

For the three described exact cuts we have used the natural evaluation of interval functions described in [5], extended with the arcsin and arccos functions. Since our evaluations involve many times the same variables, the final result is a wider interval than the actual range of possible values for that function. This is the main limitation of the described cuts.

We have done some preliminary experiments, which show that, first, the performance of exact cuts depends on the variables that have fixed values and, second, they are not able to cut the initial box to the maximum by themselves. As a consequence, the proposed cuts should be seen as complementary to the general procedure described in [5]. Basically, they can be used to accelerate the pruning process, which actually is the interest of domain-dependent cuts.

5. Conclusions

Interval methods based on interval cuts can be used to solve inverse kinematic problems. We have presented a formulation of the closure equation of single loop kinematic chains which can be used to derive domain-dependent cuts, herein called exact cuts.

By adding our exact cuts to the Newton, Snake and Taylor cuts described in [8], the pruning process will be more informed and hence faster. We are now experimenting when our cuts work best and in which order the cuts should be applyed to cut the box in the most efficient way.

Acknowledgement. This work has been partially supported by the Spanish CICYT under contract TIC96-0721-C02-01. The first author is currently supported by a grant from the Catalan Government (CIRIT. No. FI/94-3.003).

References

[1] J. W. Burdick, "On the Inverse Kinematics of Redundant Manipulators: Characterization of the Self-Motion Manifolds," *IEEE Proc. Int. Conf. Robotics Automat.*, 1:264-270, Aug. 1989.

[2] A. Castellet and F. Thomas, "Towards an Efficient Interval Method for Solving Inverse Kinematic Problems," *IEEE Proc. Int. Conf. Robotics Automat*, 1997.

[3] J. J. Craig, "Introduction to Robotics: Mechanics and Control," Addison-Wesley Pub. Comp., 1989.

[4] E. Hansen, "Global Optimization Using Interval Analysis," Pure and Applied Mathematics, New York, Marcel Dekker, Inc., 1992.

[5] P. van Hentenryck, D. McAllester and D. Kapur, "Solving Polynomial Systems using a Branch and Prune Approach," to appear, *SIAM Journal of Numerical Analysis*, also available through http://www.ai.mit.edu/people/dam/dam.html.

[6] O. Knüppel, "PROFIL-Programmer's Runtime Optimized Fast Interval Libraries," Technical Report of the Technische Univertät 93.4, Hamburg-Harburg, 1993, also available through http://www.ti3.tu-harburg.de/indexEnglisch.html.

[7] D. Manocha and Y. Zhu, "A Fast Algorithm and System for the Inverse Kinematics of General Serial Manipulators," *IEEE Proc. Int. Conf. Robotics Automat.*, 4:3348-3353, 1994.

[8] D. McAllester P. van Hentenryck, and D. Kapur, "Three Cuts for Accelerating Interval Propagation," A.I. Memo no.1542, Massachusetts Institute of Technology, 1995.

[9] M. Raghavan and B. Roth, "A General Solution for the Inverse Kinematics of All Series Chains," *Proc. of the 8th CISM-IFTOMM Symposium on Robots and Manipulators*, 1990.

[10] B. Roth, "Computational Advances in Robot Kinematics," *Advances in Robot Kinematics and Computational Geometry*, eds. A. J. Lenarčič and B. B. Ravani, Netherlands: Kluwer Academic Publishers, pp. 7-16, 1994.

[11] F. Thomas, "Graphs of kinematics constraints," in *Computer-Aided Robotic Assembly Planning*, eds. S. Lee and H. de Mello, Kluwer Academic Publishers, 1991.

[12] F. Thomas, "On the N-bar Mechanism, or How to Find Global Solutions to Redundant Single Loop Kinematic Chains," *IEEE Proc. Int. Conf. Robotics Automat*, 1:403-408, 1992.

[13] C. Wampler and A. P. Morgan, "Solving the 6R Inverse Position Problem Using a Generic-Case Solution Methodology," *Mechanism and Machine Theory*, 26(1):91-106, 1991.