Inverse kinematic solutions of 6-D.O.F. biopolymer segments

Jin Seob Kim and Gregory S. Chirikjian

Robotica / Volume 34 / Issue 08 / August 2016, pp 1734 - 1753
DOI: 10.1017/S0263574716000138, Published online: 13 April 2016

Robotica (2016) volume 34, pp. 1734–1753. © Cambridge University Press 2016
doi:10.1017/S0263574716000138

Inverse kinematic solutions of 6-D.O.F. biopolymer segments

Jin Seob Kim and Gregory S. Chirikjian*

Department of Mechanical Engineering, Johns Hopkins University
Baltimore, MD 21218, USA. E-mail: jkim115@jhu.edu

(Accepted February 24, 2016. First published online: April 13, 2016)

SUMMARY

We present two methods to find all the possible conformations of short six degree-of-freedom segments of biopolymers which satisfy end constraints in position and orientation. One of our methods is motivated by inverse kinematic solution techniques which have been developed for "general" 6R serial robotic manipulators. However, conventional robot kinematics methods are not directly applicable to the geometry of polymers, which can be treated as a degenerate case where all the "link lengths" are zero. Here, we propose a method which extends the elimination method of Kohli and Osvatic. This method can be applied directly to the geometry of biopolymers. We also propose a heuristic method based on a Lie-group-theoretic description. In this method, we utilize inverse iterations of the Jacobian matrix to obtain all conformations which satisfy end constraints. This can be easily implemented for both the general 6R manipulator and polymers. Although the extended elimination method is computationally faster than the Jacobian method, in cases where some of the joint angles are 180° (i.e., where the elimination method fails), we combine these two methods effectively to obtain the full set of inverse kinematic solutions. We demonstrate our approach with several numerical examples.

KEYWORDS: Biopolymer structure; 6R manipulator; Inverse kinematics; Eigenvalue problem; Lie-group-theoretic method; Jacobian matrix.

1. Introduction

During the past decades, the conformational analysis of polymers has been of great interest in the area of polymer science and biophysics. Several methods for this purpose have been utilized for simulating the motions of a polymer and sampling the space of preferred conformations. One of those methods includes Monte Carlo simulation. Regarding this famous statistical method, researchers have been interested in "local moves" for Monte Carlo sampling, which generates multiple conformations of a short polymer segment without causing any great change in global geometry.[1–3] This is especially useful when one is interested in the Monte Carlo sampling of polymer structures with fixed ends.[4] For example, suppose that we have a long chain-like biopolymer molecule. When we change one torsional angle in the middle of a polymer chain by a small amount, then the position and orientation of the last molecule of a chain can vary a lot. In this case, we need so-called "concerted rotation" or "local moves". In order to use this method, one needs to solve the inverse kinematics of a polymer with given end constraints. This can be solved efficiently when methods of robotic kinematics are applied. The simplified backbone geometry of a polymer is shown in Fig. 1. Looking at this figure, we can see that it is very similar to the geometry of the general robotic manipulator except when all of the adjacent axes of rotation intersect each other. In this paper, we review methods for solving the inverse kinematics problem for general 6R manipulators and propose modifications to deal with inverse kinematic problems for polymer geometries where the robotics techniques break down. We also present a method based on a Lie-group-theoretic description that can be applied and implemented easily compared with other methods to both the geometries of polymers and robot manipulators. Using

* Corresponding author. E-mail: gregc@jhu.edu

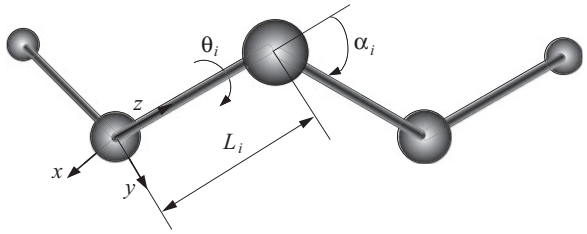

Fig. 1 The schematic conformation of a polymer used in this paper.

a combination of the proposed two methods enables us to obtain all true inverse kinematic solutions of 6-D.O.F. biopolymer structures even when there are kinematically degenerate cases such as when some of the joint angles are equal to 180°.

1.1. Literature review

Past theoretical works on "local movements" in polymer chains include the work of Gō and Scheraga.[5–7] Based on their pioneering work, Dodd at al. analyzed this motion with the concept of the "concerted rotation" method.[8] They were dealing with seven adjacent dihedral angles in polymer structures, one of which was the driving angle to generate the "concerted rotation". In this approach, one numerically solves non-linear algebraic equations in order to obtain the inverse kinematic solutions. Several researchers have applied this concerted rotation method to protein structures.[2,3,9–13] Another approach was introduced by Wedemeyer and Scheraga,[14] where the polynomial equations with respect to dihedral angles were derived through spherical geometric relations. On the other hand, some researchers in the chemical physics community have adopted Lee and Liang's method[15,16] to formulate the eigenvalue/eigenvector techniques of Manocha and Canny[17] for polypeptide kinematics including the special case which contains proline.[18]

Many researchers have studied the inverse kinematics of "general" 6*R* serial manipulators that work in all cases except for special combinations of link parameters and joint angles.[15,16,19–24] From these works, the maximum number of inverse kinematic solutions is shown to be 16. Among these, the formulation of Raghavan and Roth has been the basis of several other methods developed thereafter. In Raghavan and Roth's work, they constructed 14 equations from the basic homogeneous transformation equations and used the characteristic equation of a 12×12 matrix to get the inverse kinematic solutions.[23,24] To avoid lengthy calculations of the determinant of a 12×12 matrix and solving a 16th-order polynomial equation, Manocha and Canny formulated an eigenvalue problem to find the inverse kinematic solutions.[17] Based on Raghavan and Roth's work, they used matrix polynomials to get an augmented matrix whose size is 24×24. Then, the inverse kinematics problem can be reduced to an eigenvalue problem, which means it can be solved accurately and efficiently. Based on their formulation, Manocha *et al.* developed an extended version of Manocha–Canny formulation which can be applied to polymer structure.[25] In their work, the minimum size of a matrix for the eigenvalue problem is 32×32. Meanwhile, Kohli and Osvatic developed another method which computes inverse kinematics solutions as an eigenvalue problem by constructing a 16×16 matrix, which is linear in one suppressed variable.[26] Ghazvini also devised a method similar to that of Kohli and Osvatic.[27] His formulation also contained only one suppressed variable in the final form of matrix equation. However, in his formulation, matrices bigger than 16×16 were generated. Nielsen and Roth summarized the state-of-the-art techniques for solving the inverse kinematics of 6 degree-of-freedom serial manipulators and direct kinematics of parallel manipulators.[28] Their paper discusses power products and dialytic elimination which are based on the work of Raghavan and Roth.[23,24] Husty *et al.* proposed a method based on classical multi-dimensional differential geometry to solve a univariate polynomial equation for a general 6*R* robot manipulator.[29] A recent work develops a method to find intersection curves of four bivariate polynomial equations that can be derived based on Raghavan and Roth's formulation.[30]

1.2. Motivation and organization

Among the works discussed above, Kohli and Osvatic's work seems to be the least well known, but this work is interesting because it requires solving the smallest eigenvalue problem. It is similar to

that of Manocha and Canny in that they both used eigenvalue problems to get solutions. However, Kohli and Osvatic construct a 16×16 matrix in their work, which is smaller than that in Manocha and Canny's work and generates exactly the same number of eigenpairs as the maximum possible number of inverse kinematic solutions. Furthermore, the matrix equations appearing in their paper has only one suppressed variable. For these reasons, we adopt the method in Kohli and Osvatic's work. In the subsequent sections, we review that method and introduce our extension. We also explain why the direct application of techniques from manipulator inverse kinematics using eigenvalue/eigenvector formulation, such as work of Kohli and Osvatic[26] and that of Manocha and Canny,[17] does not work for the geometry of polymers. After that, we give a brief introduction to the Lie-group-theoretic notation, and explain so-called Jacobian method. Numerical examples follow thereafter.

2. Inverse Kinematics as an Eigenvalue/Eigenvector Problem

2.1. The method of Kohli and Osvatic

As mentioned in Section 1, we adopt Kohli and Osvatic's method to find the general inverse kinematic solutions. We give a brief explanation of their method below. For a more detailed explanation, see the work of Kohli and Osvatic.[26] Since Raghavan and Roth[23,24] built the fundamental formulation for both the work of Manocha and Canny[17] and that of Kohli and Osvatic,[26] we start the review with their formulation.

The basic matrix form of the equation which Raghavan and Roth used is

$$\mathbf{H}_1\mathbf{H}_2\mathbf{H}_3\mathbf{H}_4\mathbf{H}_5\mathbf{H}_6 = \mathbf{H}_{ee}, \tag{1}$$

where $\mathbf{H}_i, (i = 1, \ldots, 6)$ is a homogeneous transformation matrix according to the Denavit–Hartenberg (DH) representation, and can be expressed as[31]

$$\mathbf{H}_i = \begin{pmatrix} \cos(\theta_i) & -\sin(\theta_i)\cos(\alpha_i) & \sin(\theta_i)\sin(\alpha_i) & a_i\cos(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i)\cos(\alpha_i) & -\cos(\theta_i)\sin(\alpha_i) & a_i\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & L_i \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

where θ_i is a joint angle at joint i, α_i is a twist angle, a_i is a link length, and L_i is a offset at joint i. All these four symbols are called DH parameters. Looking at Fig. 1, since we choose the local z axis coincident with the backbone of a polymer, one can see that polymer structures have the characteristics as $a_i = 0$. $\mathbf{H}_{ee}$ is the position and orientation of the end effector. To avoid the analytical complexity of Eq. (1), they follow Tsai and Morgan[20] and rewrite the basic matrix equation in the new form:

$$\mathbf{H}_3\mathbf{H}_4\mathbf{H}_5 = \mathbf{H}_2^{-1}\mathbf{H}_1^{-1}\mathbf{H}_{ee}\mathbf{H}_6^{-1}. \tag{2}$$

By extracting the third and fourth columns in both sides (from first to third elements of each column excluding the last element which is 1), denoted as $\boldsymbol{l}$ and $\boldsymbol{p}$ respectively, six basic equations can be constructed. To be more specific, let the third and the fourth columns in the left-hand side be $\boldsymbol{l}_L$ and $\boldsymbol{p}_L$, respectively. Those in the right-hand side are denoted as $\boldsymbol{l}_R$ and $\boldsymbol{p}_R$, respectively. Then, the equations are obtained by setting $\boldsymbol{l}_L = \boldsymbol{l}_R$ and $\boldsymbol{p}_L = \boldsymbol{p}_R$. Since each equation has three components, the total number of equations becomes six. This description of the equations will be used throughout the paper. Furthermore, eight more independent scalar equations are obtained in the same way by considering the following terms:

$$\boldsymbol{p}\cdot\boldsymbol{p}, \quad \boldsymbol{p}\cdot\boldsymbol{l}, \quad \boldsymbol{p}\times\boldsymbol{l}, \quad (\boldsymbol{p}\cdot\boldsymbol{p})\boldsymbol{l} - (2\boldsymbol{p}\cdot\boldsymbol{l})\,\boldsymbol{p}$$

and combining these eight equations with basic six equations lead to 14 equations as in Raghavan and Roth's work.[23,24] Note that due to trigonometric rules these relations all have the same "power products" as $\boldsymbol{p}$ and $\boldsymbol{l}$, i.e., all of these terms are linear combinations of elements in the set $\{s_1s_2, s_1c_2, c_1s_2, c_1c_2, s_1, c_1, s_2, c_2, s_4s_5, s_4c_5, c_4s_5, c_4c_5, s_4, c_4, s_5, c_5, 1\}$, where $s_i = \sin(\theta_i)$ and $c_i = \cos(\theta_i)$, $(i = 1, 2, 4, 5)$.

At this point, Kohli and Osvatic used the two trigonometric relations $t_3 \sin(\theta_3) + \cos(\theta_3) = 1$ and $\sin(\theta_3) - t_3 \cos(\theta_3) = t_3$, where $t_3 = \tan(\theta_3/2)$, to get eight new equations. Finally, Kohli and Osvatic arranged the 14 equations as follows. First, one has six equations which are independent of θ_3 as

$$p_z,\ l_z,\ (\boldsymbol{p} \times \boldsymbol{l})_z,\ (\boldsymbol{p} \cdot \boldsymbol{p})l_z - (2\boldsymbol{p} \cdot \boldsymbol{l})p_z,\ \boldsymbol{p} \cdot \boldsymbol{l},\ \boldsymbol{p} \cdot \boldsymbol{p}, \tag{3}$$

and eight equations which are linearly dependent of t_3 as

$$\begin{aligned}
&l_x + t_3 l_y,\ l_y - t_3 l_x, p_x + t_3 p_y,\ p_y - t_3 p_x,\\
&(\boldsymbol{p} \times \boldsymbol{l})_x + t_3(\boldsymbol{p} \times \boldsymbol{l})_y,\ (\boldsymbol{p} \times \boldsymbol{l})_y - t_3(\boldsymbol{p} \times \boldsymbol{l})_x,\\
&(\boldsymbol{p} \cdot \boldsymbol{p})l_x - (2\boldsymbol{p} \cdot \boldsymbol{l})p_x + t_3[(\boldsymbol{p} \cdot \boldsymbol{p})l_y - (2\boldsymbol{p} \cdot \boldsymbol{l})p_y],\\
&(\boldsymbol{p} \cdot \boldsymbol{p})l_y - (2\boldsymbol{p} \cdot \boldsymbol{l})p_y - t_3[(\boldsymbol{p} \cdot \boldsymbol{p})l_x - (2\boldsymbol{p} \cdot \boldsymbol{l})p_x],
\end{aligned} \tag{4}$$

totalling 14 equations. Here, l_x, l_y and l_z, respectively denote the first, the second, and the third components of $\boldsymbol{l}$. The same notations are applied to $\boldsymbol{p}$.

With the first six equations in Eq. (3), which are independent of θ_3, one constructs the following matrix equation:

$$\mathbf{L}_1 \boldsymbol{y}_1 = \mathbf{R}_1 \boldsymbol{y}_2, \tag{5}$$

where $\mathbf{L}_1$ and $\mathbf{R}_1$ are 6×11 and 6×6 matrices, respectively. Here, vectors in the left-hand and right-hand side are defined as

$$\boldsymbol{y}_1 = \begin{bmatrix} c_4c_5 & c_4s_5 & s_4c_5 & s_4s_5 & c_4 & s_4 & c_5 & s_5 & c_2 & s_2 & 1 \end{bmatrix}^T$$

and

$$\boldsymbol{y}_2 = \begin{bmatrix} c_1c_2 & c_1s_2 & s_1c_2 & s_1s_2 & c_1 & s_1 \end{bmatrix}^T.$$

The remaining eight equations in Eq. (4), which are linearly dependent of t_3, can be arranged into the following matrix form:

$$\mathbf{L}_2 \boldsymbol{y}_1 = \mathbf{R}_2 \boldsymbol{y}_2, \tag{6}$$

where $\mathbf{L}_2$ and $\mathbf{R}_2$ are 8×11 and 8×6 matrices as functions of t_3, respectively. $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ are the same as in Eq. (5). Kohli and Osvatic combine Eqs. (5) and (6) to generate one matrix equation as

$$\mathbf{L}_3 \boldsymbol{y}_3 = \mathbf{0}, \tag{7}$$

where

$$\mathbf{L}_3 = \mathbf{L}_2 - \mathbf{R}_2 \mathbf{R}_1^{-1} \mathbf{L}_1$$

is an 8×11 matrix, and $\boldsymbol{y}_3$ is defined as

$$\boldsymbol{y}_3 = [\, c_4c_5 \quad c_4s_5 \quad s_4c_5 \quad s_4s_5 \quad c_4 \quad s_4 \quad c_5 \quad s_5 \quad c_2 \quad s_2 \quad 1 \,]^T.$$

As the next step, they make the substitution using the trigonometric relations:

$$s_4 = \frac{2t_4}{1+t_4^2} \quad c_4 = \frac{1-t_4^2}{1+t_4^2} \quad s_5 = \frac{2t_5}{1+t_5^2} \quad c_5 = \frac{1-t_5^2}{1+t_5^2},$$

where $t_4 = \tan(\theta_4/2)$ and $t_5 = \tan(\theta_5/2)$. Hence, the eight equations can be rearranged, after multiplied by the denominator $(1+t_4^2)(1+t_5^2)$, as

$$\mathbf{L}_4 \boldsymbol{y}_4 = \mathbf{0}, \tag{8}$$

where $\mathbf{L}_4$ is an 8×11 matrix, and $\boldsymbol{y}_4$ is defined as

$$\boldsymbol{y}_4 = \begin{bmatrix} t_4^2 t_5^2 & t_4^2 t_5 & t_4 t_5^2 & t_4^2 t_5 & t_4 t_5 & t_4^2 & t_4 & t_5^2 & J & K & 1 \end{bmatrix}^T,$$

where $J = \cos(\theta_2)(1 + t_4^2)(1 + t_5^2)$, and $K = \sin(\theta_2)(1 + t_4^2)(1 + t_5^2)$. They also multiply the eight equations by t_4 to get another eight equations, which, together with the previous eight equations in Eq. (8), gives a set of 16 equations, and the set of 16 equations can be arranged into the following matrix form:

$$\mathbf{L}_5 \boldsymbol{v} = \mathbf{0}, \tag{9}$$

where the column vector $\boldsymbol{v}$ is defined as

$$\boldsymbol{v} = \begin{bmatrix} t_4^3 t_5^2 & t_4^3 t_5 & t_4^3 & t_4^2 t_5^2 & t_4^2 t_5 & t_4^2 & t_4 t_5^2 & t_4 t_5 & t_4 J & t_4 K & t_4 & t_5^2 & t_5 & J & K & 1 \end{bmatrix}^T.$$

Here, J and K are the same as in the definition of $\boldsymbol{y}_4$. $\mathbf{L}_5$ is a 16×16 matrix, which is expressed as $\mathbf{A}t_3 + \mathbf{B}$, where $t_3 = \tan(\theta_3/2)$ and $\mathbf{A}$ and $\mathbf{B}$ are 16×16 matrices which can be treated as constant matrices. Consequently, the inverse kinematic solutions of a general $6R$ robot manipulator can be obtained by solving the generalized eigenvalue problem

$$\mathbf{B}\boldsymbol{v} = -t_3 \mathbf{A}\boldsymbol{v}, \tag{10}$$

where $\boldsymbol{v}$ is the 16×1 vector which appears in the left-hand side of Eq. (9), and t_3 is the corresponding eigenvalue.

After finding eigenvalues and corresponding eigenvectors of (10), one can find all the joint variables sequentially. For one value of t_3, one can determine θ_3, and one can also determine θ_2, θ_4 and θ_5 by finding J, K, t_4 and t_5, which can be calculated by normalizing the corresponding eigenvector, where "normalize" means that the last element of the eigenvector should be 1. After that, substitution of θ_2, θ_3, θ_4 and θ_5 into Eqs. (5) or (6) gives θ_1, and finally by Eqs. (1) or (2), we can determine θ_6.

3. Inverse Kinematics for Biopolymer Structures

The key to this solution method is to solve the eigenvalue problem in Eq. (10). If $\mathbf{A}$ is non-singular, then the problem reduces to the conventional eigenvalue problem as

$$-\mathbf{A}^{-1}\mathbf{B}\boldsymbol{v} = t_3 \boldsymbol{v}. \tag{11}$$

In the case when $\mathbf{A}$ is singular, we still have the possibility to get the correct eigenvalues and eigenvectors by using a generalized eigenvalue algorithm which deals with systems of the form:

$$\mathbf{B}\boldsymbol{v} = t_3(-\mathbf{A})\boldsymbol{v}. \tag{12}$$

However, if the system $(\mathbf{B} + t_3\mathbf{A})\boldsymbol{v} = \mathbf{0}$ is a singular pencil (meaning that $\det(\mathbf{B} + \mathrm{t}_3\mathbf{A}) = 0$ for all values of t_3), then solution techniques for the generalized eigenvalue problem such as the generalized Schur decomposition or QZ algorithm[32] do not give the correct answer.[33] The failure of inverse kinematics methods for the case of singular pencils has also been mentioned in the paper of Manocha and Canny.[17]

Unfortunately, a polymer structure such as a polypeptide chain or polypropylene possesses the above property due to the fact that all the link lengths of the structure are zero. In this case, another method is necessary because the algorithms of Raghavan and Roth and Manocha and Canny fail to yield solutions. Note that there are other possible methods to solve the problem in this situation.[21,29] However, we explore the eigenvalue/eigenvector formulation which we think has potential to be the fastest method. One can utilize the extended Manocha–Canny method,[25] in which the minimum size of the matrix for the eigenvalue problem is 32×32. In the subsequent section, we present the extended elimination method, which forms a smaller, and therefore more efficient, eigenvalue problem.

3.1. Implementation of the extended elimination method

If we apply Kohli and Osvatic's method to biopolymer structures, the rank of each resulting matrix, **A** and **B** in Eq. (10), becomes 14. If we calculate $\det(\mathbf{B} + t_3\mathbf{A})$ numerically, it becomes very close to zero regardless of the value of t_3, which means that this system is really a singular pencil. We circumvent this problem by the following procedures:

(1) Choose the same 14 rows from **A** and **B** to construct new 14×16 matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$. This also means that we choose 14 equations from among the original 16 equations. The reason why we can choose any 14 rows is that, if we check the rank of the augmented matrix [**A**, **B**] (formed by juxtaposition of **A** and **B**), we can see that its rank is 16, which means that **A** and **B** are degenerate in different ways.
(2) Multiply the 14 new equations by t_5. This procedure gives eight more new power products. It also generates 14 new equations, which, together with the previous 14 equations, gives a set of 28 equations, four of which are redundant.
(3) Choose 24 independent equations to construct 24×24 matrices $\mathbf{A}'$ and $\mathbf{B}'$ for which the rank of each matrix is 24. Unlike in step 1, in this step care must be taken so that the rank of $\mathbf{A}'$ is 24.

Then, the system can be rewritten as

$$\mathbf{L}_5'\boldsymbol{v}' = \mathbf{0}, \tag{13}$$

where $\boldsymbol{v}'$ is defined as

$$\begin{aligned}\boldsymbol{v}' = \big[&t_4^3t_5^3,\ t_4^3t_5^2,\ t_4^3t_5,\ t_4^3,\ t_4^2t_5^3,\ t_4^2t_5^2,\ t_4^2t_5,\ t_4^2,\ t_4t_5^3,\ t_4t_5^2,\ t_4t_5,\ t_4t_5J,\\ &\times t_4t_5K,\ t_4J,\ t_4K,\ t_4,\ t_5^3,\ t_5^2,\ t_5J,\ t_5K,\ t_5,\ J,\ K,\ 1\big]^T\end{aligned}$$

and $\mathbf{L}_5'$ is a 24×24 matrix which can be expressed as a linear combination of t_3 and constant 24×24 matrices, $\mathbf{A}'$ and $\mathbf{B}'$ as $\mathbf{L}_5' = t_3\mathbf{A}' + \mathbf{B}'$. Then, we can apply the eigenvalue problem algorithm, either as in Eqs. (11) or (12) to this system. In particular, if $\mathbf{A}'$ happens to be singular, then we can still apply the generalized eigenvalue solution techniques, as in Eq. (12), to it as long as the system is not a singular pencil, and in practice we have found that the new system is not a singular pencil. As for the rest of the joint variables, we can use a similar method to that of Kohli and Osvatic.[26] That is, by the eigenvalues and the corresponding eigenvectors which are normalized in the same way as in the previous section, we can determine θ_3, θ_4, θ_5 and θ_2. Then, substitution of these values into Eq. (5) gives θ_1. Finally, we can get the value of θ_6 through Eq. (2).

4. The Jacobian Method

In this section, we use a method based on a Lie-group-theoretic description to find all of the inverse kinematic solutions of the general $6R$ manipulator including the case of biopolymer geometry.

4.1. Notation and terminology

We review basic terminology in this subsection. Since we are dealing with rigid-body motion, we focus on the Euclidean motion group. See the references[34–36] for detailed explanations. Our notations follow Murray, *et al.*.[35] We also mention the notational difference (e.g., between Murray, *et al.*[35] and Chirikjian and Kyatkin[36]) to avoid possible confusion.

The Euclidean motion group (or "special Euclidean" group), $SE(3)$, is the set of all possible rotations and translations in three-dimensional space together with a composition rule. Let $g = (\mathbf{R}, \boldsymbol{b})$ be an element of $SE(3)$, then one can represent this element with a 4×4 matrix as

$$\mathbf{H} = \begin{pmatrix} \mathbf{R} & \boldsymbol{b} \\ \mathbf{0}^T & 1 \end{pmatrix}, \tag{14}$$

where $\mathbf{R} \in SO(3)$, and $\boldsymbol{b} \in \mathbb{R}^3$. By $\mathbf{R} \in SO(3)$, we mean that $\mathbf{R}\mathbf{R}^T = \mathbf{1}$ and $\det(\mathbf{R}) = +1$.

For a 3×3 skew-symmetric matrix $\boldsymbol{\Omega}$ of the form

$$\boldsymbol{\Omega} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix},$$

we define the dual vector $\boldsymbol{\omega}$ as

$$\boldsymbol{\omega} = (\boldsymbol{\Omega})^{\vee} = \begin{pmatrix} \omega_1 & \omega_2 & \omega_3 \end{pmatrix}^T.$$

Similarly, we can define the dual vector for a small (or infinitesimal) rigid-body motion, which is related to infinitesimal screw motion, as

$$\begin{pmatrix} \boldsymbol{\Omega} & \boldsymbol{v} \\ \mathbf{0}^T & 0 \end{pmatrix}^{\vee} = \begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{pmatrix}, \tag{15}$$

where $(\boldsymbol{\Omega})^{\vee} = \boldsymbol{\omega}$ and $\widehat{\boldsymbol{\omega}} = \boldsymbol{\Omega}$. The current notation directly follows Murray, *et al.*.[35] On the other hand, in Chirikjian and Kyatkin,[36] the definition of the infinitesimal rigid-body motion is defined as $\begin{pmatrix} \boldsymbol{\omega}^T & \boldsymbol{v}^T \end{pmatrix}^T$, which will affect the definition of the adjoint as shown later.

Given a rigid-body motion

$$\mathbf{H}(\boldsymbol{q}) = \begin{pmatrix} \mathbf{R}(\boldsymbol{q}) & \boldsymbol{b}(\boldsymbol{q}) \\ \mathbf{0}^T & 1 \end{pmatrix},$$

where $\boldsymbol{q} = (\ldots, q_i, \ldots)^T, (i = 1, \ldots, 6)$ denotes the parameters that describe the rigid body motion, we can define the "body" Jacobian matrix as

$$\mathbf{J}_b = \left[\left(\mathbf{H}^{-1} \frac{\partial \mathbf{H}}{\partial q_1} \right)^{\vee}, \ldots, \left(\mathbf{H}^{-1} \frac{\partial \mathbf{H}}{\partial q_i} \right)^{\vee}, \ldots, \left(\mathbf{H}^{-1} \frac{\partial \mathbf{H}}{\partial q_6} \right)^{\vee} \right]. \tag{16}$$

The adjoint of $g = (\mathbf{R}, \boldsymbol{b}) \in SE(3)$ is defined as

$$\mathrm{Ad}_{\mathbf{H}} = \begin{pmatrix} \mathbf{R} & \widehat{\boldsymbol{b}}\mathbf{R} \\ \mathbf{0} & \mathbf{R} \end{pmatrix}.$$

Note that if we define Eq. (15) as $\begin{pmatrix} \boldsymbol{\omega}^T & \boldsymbol{v}^T \end{pmatrix}^T$ as in Chirikjian and Kyatkin,[36] then $\widehat{\boldsymbol{b}}\mathbf{R}$ and $\mathbf{0}$ terms are switched, unlike as in the above expression.

Finally, if $\mathbf{X}$ is a screw matrix representing an infinitesimal rigid-body motion such as that in Eq. (15), the following relation holds

$$\mathrm{Ad}_{\mathbf{H}}(\mathbf{X})^{\vee} = \left(\mathbf{H}\mathbf{X}\mathbf{H}^{-1} \right)^{\vee}. \tag{17}$$

4.2. An inverse kinematic solution using the Jacobian

In this section, we discuss a method using the Jacobian matrix to find a single inverse kinematic solution of the general $6R$ manipulator. There have been several other methods for the inverse kinematics of robot manipulators other than those stated in Section 1.1. One example is a method using polynomial continuation.[22,37,38] In that formulation, the system is represented with a set of polynomial equations by introducing continuation parameters. One solves the system of equations with an initial value of the continuation parameters. Then, one generates solution paths in terms of these parameters to obtain all the solutions. This method has been successfully applied to the inverse kinematics of robot manipulators, even in the case when some of the joint angles are 180°.[22] Another example is the probabilistic approach.[36] This method has especially been applied to the

inverse kinematics and workspace generation of hyper-redundant manipulators.[39–44] It utilizes non-commutative harmonic analysis and generalized convolution on Lie groups to generate the workspace density (a probability density function on $SE(3)$). This has been applied to polymer chains.[45,46]

One of the most popular methods for numerical inverse kinematics relies on Jacobian iterations to update joint angles. Many variations on Jacobian-based iterative methods for solving serial chain inverse kinematics problems have been proposed in the literature. In the past, this method has been applied to the control of manipulators. One of the famous works is "resolved motion rate control".[47] In this work, Whitney used the Jacobian inverse and pseudo-inverse to obtain the increments of a vector of joint angles. Others include the work of Uicker *et al.*[48] and that of Isobe *et al.*,[49] etc. Those works used numerical methods such as Newton–Raphson or Newton's method to solve non-linear equations. Another type of method, for example, the work of Wang and Chen[50] includes the optimization technique called cyclic coordinate descent (CCD). This CCD method has been applied to robotic manipulators.[51] Moreover, some researchers in the area of computational biology have used this method to predict protein loop conformations.[52–54] Chirikjian also used a Jacobian-based approach to solve the inverse kinematics of a hyper-redundant manipulator treated as a continuous curve in space.[40] The novel parts of our method compared with others are: (1) the way we define the artificial path $\mathbf{H}_p(t)$; (2) the correction term which is used for finding one inverse kinematic solution and (3) the way we generate an initial set of candidate conformations to obtain all (rather than a single) inverse kinematic solution. This type of inverse kinematics solution technique using an artificial path has been successfully applied to determine the minimum energy conformation of double-helical DNA.[55] One advantage of this method is that it can be easily implemented in both the cases of biopolymers and robot manipulators in that one does not need to perform symbolic computations.

Let $\mathbf{H}_f$ be the desired pose of the end effector of a general $6R$ manipulator. We can get an initial guess of a pose of the end effector by using arbitrary values for the six joint variables θ_i's, where $i = 1, \ldots, 6$ and we denote this as $\mathbf{H}_0$.

We define an artificial function called $\mathbf{H}_p(t)$. This ideal path function is defined as

$$\mathbf{H}_p(t) = \mathbf{H}(t') \exp(t \cdot \log(\mathbf{H}^{-1}(t')\mathbf{H}_f)),$$

where $\mathbf{H}(t')$ is the pose of end effector at t', and $t' = t$, but when it comes to differentiation with respect to t, t' is treated as a constant. The notations $\log(\cdot)$ and $\exp(\cdot)$ describe the logarithm and exponential of matrices, which are well-defined quantities in the current context.

Note that $\mathbf{H}_p(0) = \mathbf{H}_0$ and $\mathbf{H}_p(1) = \mathbf{H}_f$. This function generates an artificial trajectory which is formed from the current frame of the end effector to the desired one, and pushes the end effector toward the desired pose. This suggested path function has an advantage in that it contains information on the current frame of an end effector as a feedback, which guarantees the convergence to the desired position and orientation.

In general, we can get one inverse kinematic solution by using this Jacobian-based method. In this context, the pose of an end effector is the product of six homogeneous transformation matrices as

$$\mathbf{H} = \mathbf{H}_1(\theta_1)\mathbf{H}_2(\theta_2)\mathbf{H}_3(\theta_3)\mathbf{H}_4(\theta_4)\mathbf{H}_5(\theta_5)\mathbf{H}_6(\theta_6).$$

Given this, the Jacobian matrix can be computed as

$$\mathbf{J}_b = \left[\ldots, \left(\mathbf{H}^{-1}\frac{\partial \mathbf{H}}{\partial \theta_i}\right)^{\vee}, \ldots\right],$$

where in this case one can find that each term can be computed by using Eq. (17) as

$$\left(\mathbf{H}^{-1}\frac{\partial \mathbf{H}}{\partial \theta_i}\right)^{\vee} = \mathrm{Ad}_{(\mathbf{H}_{i+1}\cdots\mathbf{H}_6)^{-1}}\left(\mathbf{H}_i^{-1}\frac{\partial \mathbf{H}_i}{\partial \theta_i}\right)^{\vee}, \text{ for } \; i = 1, \ldots, 5$$

and

$$\left(\mathbf{H}^{-1}\frac{\partial \mathbf{H}}{\partial \theta_6}\right)^{\vee} = \left(\mathbf{H}_6^{-1}\frac{\partial \mathbf{H}}{\partial \theta_6}\right)^{\vee}.$$

Since we have the expression of the Jacobian, we can integrate the velocity relation for $\mathbf{H}_p \in SE(3)$

$$\left(\mathbf{H}_p^{-1}(t)\dot{\mathbf{H}}_p(t)\right)^{\vee} = \mathbf{J}_b(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}$$

numerically to get an inverse kinematic solution. To implement this numerically, we use the following algorithm.

First, we calculate the increment at the kth step as

$$\dot{\boldsymbol{\theta}}_k = \mathbf{J}_b^{-1}\left(\mathbf{H}_p^{-1}(t_k)\dot{\mathbf{H}}_p(t_k)\right)^{\vee}. \tag{18}$$

However, this is not enough to get an accurate solution. Hence, we need to define the correction term which forces the actual pose of an end effector to the desired ideal path defined by $g_p(t)$ as

$$\boldsymbol{\theta}_k^c = \mathbf{J}_b^{-1}\left[\log\left(\mathbf{H}^{-1}(t_k)\mathbf{H}_p(t_k)\right)\right]^{\vee}. \tag{19}$$

With Eqs. (18) and (19), we can get a joint variable vector of the next step as

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \Delta t\ \dot{\boldsymbol{\theta}}_k + \boldsymbol{\theta}_k^c. \tag{20}$$

By using this algorithm, we can find one of the possible inverse kinematic solutions of the general $6R$ manipulator (and, in particular, a six-degree-of-freedom polymer segment). In the event that the path $\mathbf{H}_p(t)$ makes the manipulator pass through a singularity, a pseudo-inverse by singular value decomposition is used in place of the inverse in Eqs. (18) and (19).

4.3. How to find all the inverse kinematic solutions

In this section, we explain a method to obtain all of the inverse kinematic solutions for a particular end pose by applying the Jacobian method.

Assume that, after applying the method explained in the previous section, we can get one inverse kinematic solution. We use this one solution as a starting point to find all of the other solutions. We then add 0 (rad) and π (rad) to each of the joint angles, θ_i. This is based on the intuition that in a planar two-link manipulator, the solutions of one joint angle differ by π (rad) with each other.[35] Then, we have $2^6 = 64$ different initial guess vectors for a $6R$ manipulator. When we apply the Jacobian method to this set of 64 different guess vectors, the results, of course, do not give 64 different solutions since at most only 16 are possible. In theory, all of the 64 converge to a subset of all possible inverse kinematic solutions. In practice, this can be achieved when we use the smaller Δt. However, due to the computational cost, this Δt cannot be infinitesimally small. Another issue is that the Jacobian is not a smooth function of t. Because of these two issues, if Δt is not small enough in some paths, then those paths may not converge to the correct solutions.

For two elements of $SE(3)$, $g_1 = (\mathbf{R}_1, \boldsymbol{b}_1)$ and $g_2 = (\mathbf{R}_2, \boldsymbol{b}_2)$, we can define a metric between these two elements as[56,57]

$$d(g_1, g_2) = \sqrt{\|\boldsymbol{b}_1 - \boldsymbol{b}_2\|^2 + L^2\|\mathbf{R}_1 - \mathbf{R}_2\|^2}, \tag{21}$$

where $\|\mathbf{R}\| = \sqrt{\text{trace}(\mathbf{R}\mathbf{R}^T)}$. Here, L is a length scale to make distances of rotational part and translational part compatible, and can be related to the rotational inertia and the mass of a rigid body consisting of all atoms attached to each central molecule, which is depicted as a sphere in Fig. 1, in biopolymer structure.[57,58] Note that this metric has the property that it is invariant to the change of each element by shift. If two frames have a metric distance which is smaller than a certain criterion value, say 1×10^{-3}, then we can treat those two as being identical. In this way, we can discard the resulting vectors which do not converge, and representative joint vectors that have converged are

Table I. List of parameters used in example 1.

Joint	Link length a_i [Å]	Offset L_i [Å]	Twist angle α_i [°]
1	0	1.53	66
2	0	1.53	−68
3	0	1.53	66
4	0	1.53	−68
5	0	1.53	66
6	0	1.53	−68

Table II. List of real eigenvalues in example 1.

Number	Real eigenvalue
1	−5.9630
2	−5.7488
3	4.9660
4	4.3394
5	−0.8833
6	0.3470
7	1.0233
8	0.9004
9	0.9124
10	−0.3955
11	−0.5719
12	−0.5427

gathered to form an initial set of solutions. However, this initial set of solutions may not contain all possible solutions. In some cases, one or two solutions are missing. In order to make the solution set complete, we apply the above method with each joint vector in the solution set obtained previously as an initial solution used to generate 64 new guess vectors. Finally, the missing solutions, if any, can be found and all the resulting joint vectors converge to one in the set of solutions. In practice, this set corresponds to all the inverse kinematic solutions.

5. Numerical Examples

In this section, we demonstrate the methods explained in the previous sections with numerical examples. In Fig. 1 is shown the schematic conformation of a polymer including the kinematic parameters.

5.1. Example 1

Table I shows the structural constants used in the first example. This is very similar to the geometry of polypropylene which appears in the work of Dodd, Boone and Theodorou.[8] All the parameters follow the DH formalism.[31] As we discussed in the earlier section, all the link lengths are equal to zero. First, we try the following pose of the end effector:

$$\mathbf{H}_f = \begin{pmatrix} -0.8642 & -0.0270 & -0.5024 & -0.2884 \\ 0.4644 & -0.4272 & -0.7758 & -1.8013 \\ -0.1937 & -0.9037 & 0.3817 & 6.3649 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The real-valued eigenvalues calculated by the extended method are presented in Table II. In this case, there are 12 possible inverse kinematic solutions, which implies that this case corresponds to one that has the maximum possible number of solutions according to the work of Dodd, L. *et al.*[8] The 12 solutions are given in Table III and corresponding conformations are shown in Fig. 2.

Now let us apply the Jacobian method to this example. As described in Section 4.3, first we try with an arbitrary initial guess for the vector of joint angles, say $[0.1 \quad 0.2 \quad 0.3 \quad 0.4 \quad 0.5 \quad 0.6]^T$. With

Table III. The joint angles corresponding to the solutions [°].

i	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	−96.2800	−26.2700	−160.9600	38.4800	82.5500	−35.4000
2	81.9899	4.9596	−160.2643	−60.3541	−68.3477	−34.0301
3	−82.7885	−3.1069	157.2294	63.1350	88.3393	−29.6753
4	94.1134	29.5025	154.0461	−37.6020	−57.6850	−26.1063
5	6.4869	−62.1639	−82.9073	−137.4351	−1.1064	28.1057
6	−4.0928	66.6818	38.2765	−138.3936	−44.1007	−131.7876
7	−86.6833	−7.3910	91.3207	−56.8352	−87.9667	−103.5152
8	80.9276	62.1464	84.0005	−62.0489	−1.7489	27.4160
9	11.3907	78.4807	84.7535	135.6536	2.1915	−107.7752
10	34.5097	−35.7504	−43.1578	118.9050	52.5449	49.0738
11	−65.4235	−77.5112	−59.5267	87.4006	−13.2471	−122.3746
12	59.9160	−14.5902	−56.9800	91.1014	65.9502	42.8249

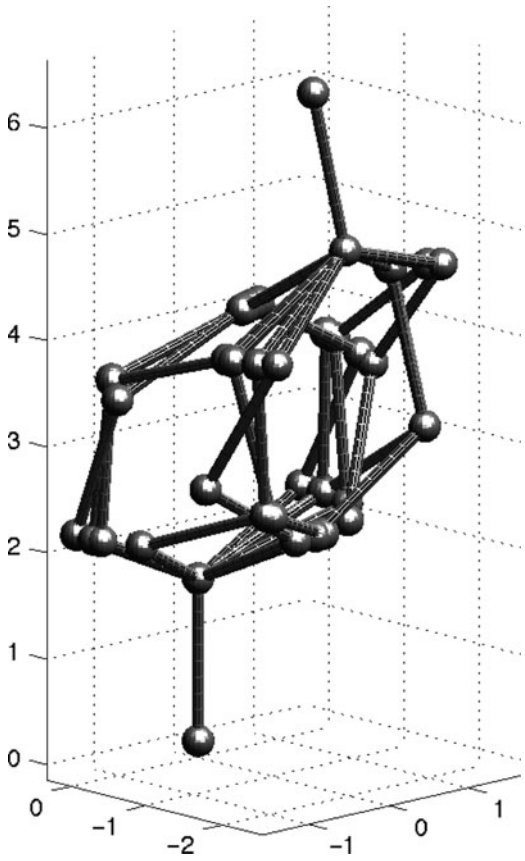

Fig. 2. The conformations of all the solutions in example 1.

the same geometric parameters and pose of the end effector as in the above example, we can get one inverse kinematic solution as

$$\boldsymbol{\theta} = \begin{pmatrix} -82.7885 \\ -3.1069 \\ 157.2294 \\ 63.1350 \\ 88.3393 \\ -29.6753 \end{pmatrix}.$$

Table IV. Comparison of running time.

Average (± std) [sec]	Extended elimination method	Jacobian method
Example 1	$3.4771 \times 10^{-3}(\pm 4.4242 \times 10^{-4})$	$0.1181(\pm 6.6641 \times 10^{-3})$
Example 2	$3.2032 \times 10^{-3}(\pm 2.4787 \times 10^{-4})$	$0.1219(\pm 6.9362 \times 10^{-3})$
	$(5.4271 \times 10^{-3}(\pm 5.6102 \times 10^{-4}))^*$	
Example 3	$3.4367 \times 10^{-3}(\pm 5.5903 \times 10^{-4})$	$0.1218(\pm 6.7818 \times 10^{-3})$

*Combined method

With this vector as an initial guess vector, we try the Jacobian method for the 64 different cases. In this example, fortunately we can generate 12 inverse kinematic solutions with only one initial guess vector. The results are exactly the same as in Table III and Fig. 2. However, in general, we should apply this method for each of the resulting set of vectors as an initial guess vector. It is worth noting that, when we add 0, $\pi/2$, π, and $3\pi/2$ instead of 0 and π, we obtain the same solution sets for all the examples in this section. This illustrates that the Jacobian method is practically good enough to obtain all inverse kinematic solutions.

As one can imagine, the Jacobian method should be slower than the extended elimination method, in part due to the iterative computation. In order to compare the computational efficiency between two methods, we measured time spent for each method, as respectively explained in Sections 3.1 and 4.3. Each method was implemented via Matlab script files to run in Matlab (version R2015) on a laptop computer (CPU 2.3 GHz Intel Core i7, OS X 10.10.4). To optimize the performance, the computationally heaviest parts (the part that finds all solutions in the extended elimination method, and the iteration part with 64 different initial sets in the Jacobian method) are replaced by C using "mex" function in Matlab. Twenty trials were performed for each method. Computation times were then measured by using "tic/toc" Matlab commands. Table IV shows the average computation times for the extended method and the Jacobian method for example 1, 2 and 3, respectively, in the format of average (± standard deviation). For all cases, the average computation times for the extended elimination method and the Jacobian method are about 3×10^{-3} and 0.1 s, respectively. Note that in the table, the case marked with an asterisk in Example 2 corresponds to the combined method that applies the extended elimination and the Jacobian methods together, which is useful in special situations that will be explained in more detail in the next example. The results clearly show that that the Jacobian method is computationally much more expensive than the extended elimination method. This is one drawback of the Jacobian method. However, as we will see in the following example, we can make use of the Jacobian method for a special purpose.

Note that, although we optimized our Matlab codes using "mex" function, it is difficult to directly compare the computational speed with other methods[11,52,53] partly due to the availability of the existing codes and some of them being written entirely in Fortran, C and python. Also, the technical skill of writing different codes in different languages becomes another factor that adds up to this difficulty. Finally, unlike the methods mentioned above, our method is to obtain all the possible conformations of 6-degree-of-freedom biopolymer structure. That being said, we believe that our approach, especially the eigenvalue approach, is at least as fast as other well-known methods especially in terms of obtaining all the possible conformations, because there is no iterative procedure involved and the dimension of the matrix for eigenvalue problem is the smallest among known ones.

5.2. Example 2

As the next numerical example, we consider the case which appears in Table V. This is similar to the simplified model of a protein molecule,[9,10] except that the $C - N$ bond is not fixed as 180°. Although each bond length of a backbone in a molecule can vary within some range, we treat here the bond length between each atom in a backbone as a covalent bond which has same bond length with $1.5\,\mathring{A}$.[59] Let the pose of an end effector be

$$\mathbf{H}_f = \begin{pmatrix} 0.5681 & -0.7792 & -0.2647 & 4.3548 \\ -0.0992 & 0.2544 & -0.9620 & -0.5697 \\ 0.8169 & 0.5728 & 0.0672 & 1.0536 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix}.$$

Table V. List of parameters used in example 2.

Joint	Link length a_i [Å]	Offset L_i [Å]	Twist angle α_i [°]
1	0	1.5	60
2	0	1.5	−60
3	0	1.5	70.53
4	0	1.5	−60
5	0	1.5	60
6	0	1.5	−70.53

Table VI. List of real eigenvalues in example 2.

Number	Real eigenvalues
1	5.9970
2	−2.9724
3	0.9590
4	0.8375

Table VII. The joint angles by the extended elimination method in example 2 [°].

i	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	166.3421	130.8356	161.0662	99.6300	0.6208	67.2585
2	14.2311	−93.6975	−142.8117	79.5403	131.5450	−161.2173
3	−148.7040	65.5293	87.6000	2.2625	29.2118	79.2319
4	43.2346	−37.7975	79.8894	−156.3153	64.0651	−133.0993

First, we apply the extended method to this example. Solving the eigenvalue problem of Eq. (13) gives four distinct real eigenvalues, which are shown in Table VI. In Table VII are shown the joint angles as well. In addition, Fig. 3 shows the conformations of each solution. Looking at Fig. 3, we can see that one of four solutions is not a true solution, which corresponds to the third real eigenvalue, 0.9590. If we further look at the last element of the corresponding eigenvector, which is to be used for normalization, it is -2×10^{-14}. In addition to the last element, we see that 17 elements are nearly zeros. Hence, normalizing this eigenvector leads to the incorrect answer. Actually, since the last element can be treated as zero, we should not normalize this eigenvector.

Note that there has been some work on using polynomial root finding for inverse kinematic solutions in the case when some joint angles are 180° (e.g., Manseur and Doty[21]). However, the work considered general robot manipulator geometries which did not include the special case of biopolymer structures. Other methods presented in the literature also have degeneracies. The method of Manocha, *et al.*[25] suffers from the same numerical issues as ours when some of the joint angles are 180°. The method of Coutsias, *et al.*[11] describes a polypeptide chain as a series of virtual $C_\alpha - C_\alpha$ bonds, and instead of using the original torsion angles it concentrates the degrees of freedom as spherical rotations at each vertex where virtual bonds are connected. The actual bonds and bond angles can then be reconstructed after the inverse kinematics solution for the virtual structure is found. Essentially this method substitutes the original polypeptide chain with a nonphysical proxy. Then, it solves the inverse kinematics problem for this proxy, and fits the pieces of the original structure to it in a second procedure. The non-linear change of coordinates employed in this approach distorts the original kinematics problem, and as such, does not have all of the same degenerate cases. However, this does not mean that it is free of degeneracies. As an analogy, the Jacobian matrix for any 6-D.O.F. manipulator will have singularities somewhere in the configuration space, and switching from one kinematic structure to another will not eliminate singularities, but rather will only move them to new locations. Likewise, using a non-physical proxy structure in place of the original will not

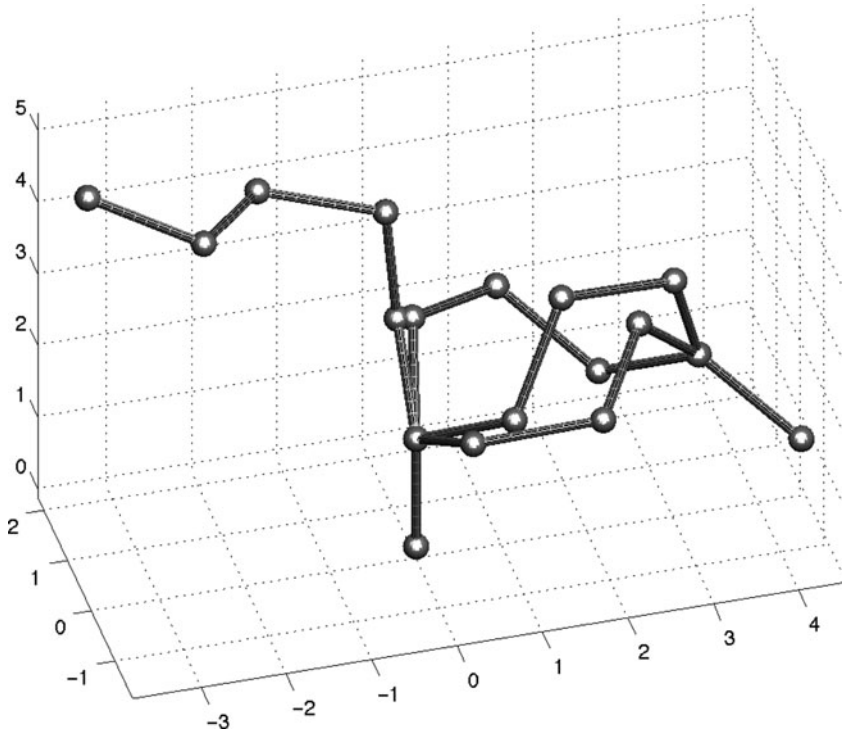

Fig. 3. The conformations of all the solutions by the extended method in example 2.

eliminate degenerate cases, but rather will only move them. The central issue then should not be how to remove all degenerate cases (which is impossible when using any change of variables involving a half-angle tangent substitution), but rather how to handle them gracefully. We have devised a simple approach to handle degenerate cases by slightly perturbing bond angles and/or end-effector poses. These small perturbations avoid the original degeneracies and produce structures that "almost" solve the original problem. That is, they produce joint-angle solutions that differ by small fractions of a degree from the solution that is sought. Following this, we use a single step of our Jacobian method to correct this approximation, thereby providing the exact solution to the original problem. Though the implementation is completely different than that in Coutsias, *et al.*, both of these methods invoke two-step procedures.

The details of this simple approach for handling the degenerate case, denoted as the "combined method", are the following. Keeping the structural parameters (offsets and twist angles) constant, we modify the target reference frame $\mathbf{H}_f$ which is a homogeneous transformation representation of $g_f = (\mathbf{R}_f, \boldsymbol{b}_f)$ by a small amount. To be more specific, let $\delta\theta$ be a small angle. We first define a new target frame $\mathbf{H}_f^n$, of which the corresponding $SE(3)$ element is $g_f^n = (\mathbf{R}_f^n, \boldsymbol{b}_f^n)$, by multiplying by a small rotation with $\delta\theta$ as $\mathbf{R}_f^n = \mathbf{R}_f \exp(\delta\theta\, \widehat{\mathbf{n}})$, where $\mathbf{n}$ is a unit vector defining the axis of rotation, and $\boldsymbol{b}_f^n = \boldsymbol{b}_f$. The vector $\mathbf{n}$ can be chosen as any unit vector, and specifically one of the standard basis vectors (e.g., $\mathbf{n} = \mathbf{e}_3$, i.e., a unit vector along z-axis) can be used. Then, we apply the extended elimination method to obtain the inverse kinematic solutions for this perturbed problem. With this as an initial guess, we apply the method in Section 4.2 to obtain a true solution for $\mathbf{H}_f$. Since we use a very small angle (e.g., $\delta\theta = 0.1^\circ$ or even smaller), the number of steps required for the method in Section 4.2 can be small by choosing a relatively large Δt, equivalent to the inverse of the total number of steps (recall $t \in [0, 1]$). The cases marked with an asterisk in Tables IV and XII were obtained with $\delta\theta = 0.05^\circ$ and $\Delta t = 0.2$, along with $\mathbf{n} = \mathbf{e}_3$.

Now, we apply the Jacobian method with any vector of joint angles in Table VII. For instance, let us choose the third set of joint angles which is, in fact, an incorrect one. We can generate a set of 64 different joint vectors by the joint vector chosen above, and then apply the Jacobian method for 64 cases. The results are shown in Table VIII and Fig. 4. By using the Jacobian method, we can get a

Table VIII. The joint angles by the Jacobian method in example 2 [°].

i	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	166.3421	130.8356	161.0662	99.6300	0.6208	67.2585
2	14.2311	266.3025	217.1883	79.5403	131.5450	198.7827
3	143.3000	180.0000	87.6000	180.0000	323.5000	38.3000
4	43.2346	322.2025	79.8894	203.6847	64.0651	226.9007

Table IX. List of DH parameters for protein structures. This is adopted from Manocha, *et al.*[25].

Joint	Link length a_i [Å]	Offset L_i [Å]	Twist angle α_i [°]
1	0	−5.81	8.67
2	0	9.44	70.05
3	0	−5.86	8.61
4	0	9.49	70.11
5	0	−5.78	8.68
6	0	9.42	70.12

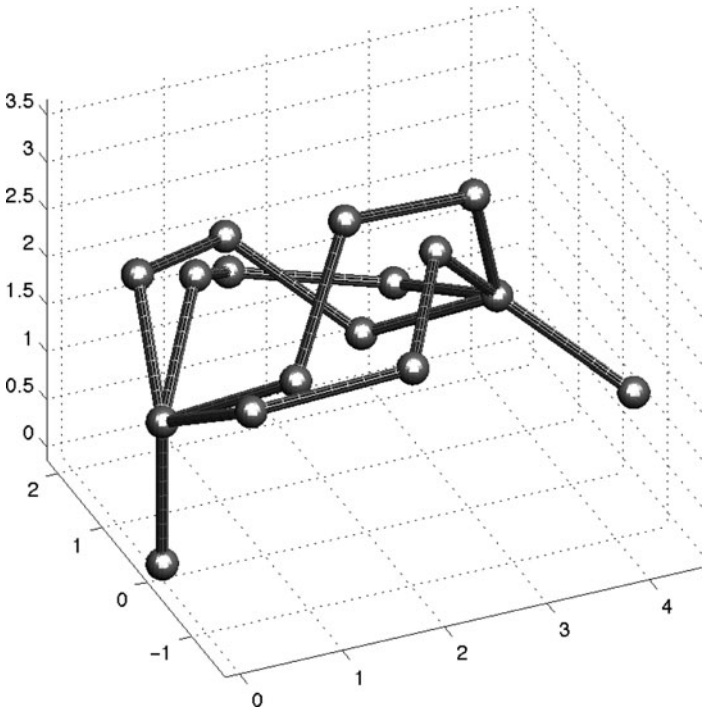

Fig. 4. The true inverse kinematic solutions in example 2.

set of solutions with true joint angle vectors, which the extended elimination method does not give. Looking at Fig. 4, we can see that all four of the solutions are correct.

5.3. Example 3

As the third example, we consider the polypeptide structure. In Fig. 5 is depicted the schematic picture of a polypeptide unit, which is adopted from Manocha, *et al.*.[25] All four atoms in the peptide unit, such as C_α, C and N, lie in the same plane. Noting that the each link $P - C_\alpha$ has the same rotation with $C_\alpha - C$ and $N - C_\alpha$, we can treat those two as the links in DH formalism. All DH parameters are shown in Table IX. Originally, this data was obtained from a segment of α helix.[25] Then, we apply

Table X. The joint angles by the extended elimination method in example 3 [°].

i	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	−137.8070	−84.4386	−55.4536	84.1689	−66.0047	−4.1227
2	−168.0328	−74.2095	−50.4896	76.5695	26.6843	−112.6164
3	43.1351	110.6350	−35.1735	21.9682	82.0679	−132.8294
4	49.0000	130.0000	−30.0000	20.0000	−60.0000	30.0000
5	36.7905	95.7387	20.9278	−24.6038	80.8806	−144.9628
6	44.8240	115.5686	20.4050	−21.3483	−78.8559	37.3791

Table XI. The joint angles by the Jacobian method in example 3 [°].

i	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6
1	48.9771	130.0056	330.0467	19.9652	299.9583	30.0309
2	36.7801	95.7592	20.8928	335.4277	80.8966	215.0271
3	44.8101	115.5937	20.3667	338.6866	281.1379	37.3918
4	191.9706	285.7823	309.5211	76.5626	26.6840	247.3806
5	222.2017	275.5502	304.5576	84.1604	293.9868	355.8842
6	43.1148	110.6383	324.8665	21.9368	82.0866	227.1408

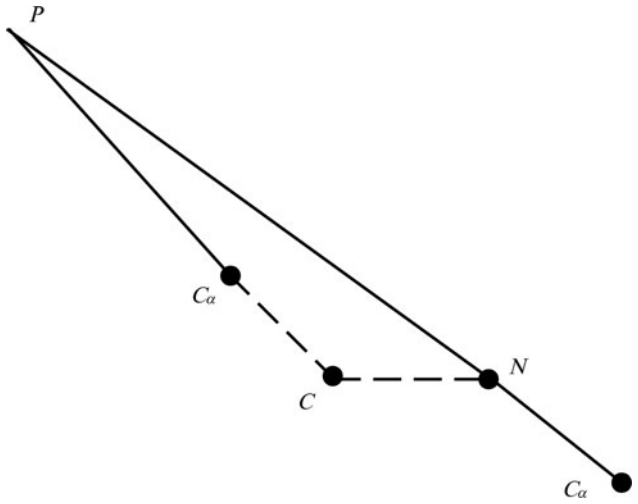

Fig. 5. Schematic representation of peptide unit by DH formalism. Real lines represent the links in actual calculations. This is adopted from Manocha, *et al.*[25]

our methods with the following pose of end effector

$$\mathbf{H}_f = \begin{pmatrix} -0.8923 & 0.0738 & 0.4455 & 3.3759 \\ -0.2368 & 0.7635 & -0.6009 & 4.1505 \\ -0.3845 & -0.6416 & -0.6637 & 0.6349 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix}.$$

In Tables X and XI, we show the resulting joint angles by the extended elimination method and the Jacobian method, respectively. Figure 6 shows all inverse kinematic solutions. Looking at Tables X and XI, one can find that two results are not exactly the same. This is due to the fact that the desired frame is expressed only with four digits. If we use longer digit numbers, for example as "format long" style in Matlab, then we have the exactly same results. As shown in the figure and tables, both our methods can be applied to the polypeptide structures.

Table XII. Results of example 4 (tripzip2 case): original structures.

	Extended elimination method		Jacobian method	
Trials	# of solutions	Computing time [s]	# of solutions	Computing time [s]
1	1 (2)	2.5757×10^{-3} ($\pm 8.4090 \times 10^{-5}$)	2	0.1122 ($\pm 5.6673 \times 10^{-3}$)
	2	$(4.9820 \times 10^{-3}$ ($\pm 2.9500 \pm 10^{-4}$))*		
2	6	2.6067×10^{-3} ($\pm 9.9450 \times 10^{-5}$)	6	0.0938 ($\pm 3.4573 \times 10^{-3}$)
3	4	2.6304×10^{-3} ($\pm 2.6050 \times 10^{-4}$)	4	0.1086 ($\pm 2.9198 \times 10^{-3}$)
4	2	2.6527×10^{-3} ($\pm 3.1320 \times 10^{-4}$)	2	0.1025 ($\pm 2.7672 \times 10^{-3}$)
5	6	2.6285×10^{-3} ($\pm 1.5590 \times 10^{-4}$)	6	0.0960 ($\pm 3.5481 \times 10^{-3}$)
6	4	2.5773×10^{-3} ($\pm 7.5420 \times 10^{-5}$)	4	0.1023 ($\pm 2.4818 \times 10^{-3}$)
7	2	2.5684×10^{-3} ($\pm 7.5904 \times 10^{-5}$)	2	0.1256 ($\pm 3.0442 \times 10^{-3}$)
8	2	2.5541×10^{-3} ($\pm 6.5750 \times 10^{-5}$)	2	0.1053 ($\pm 2.2375 \times 10^{-3}$)

*Combined method

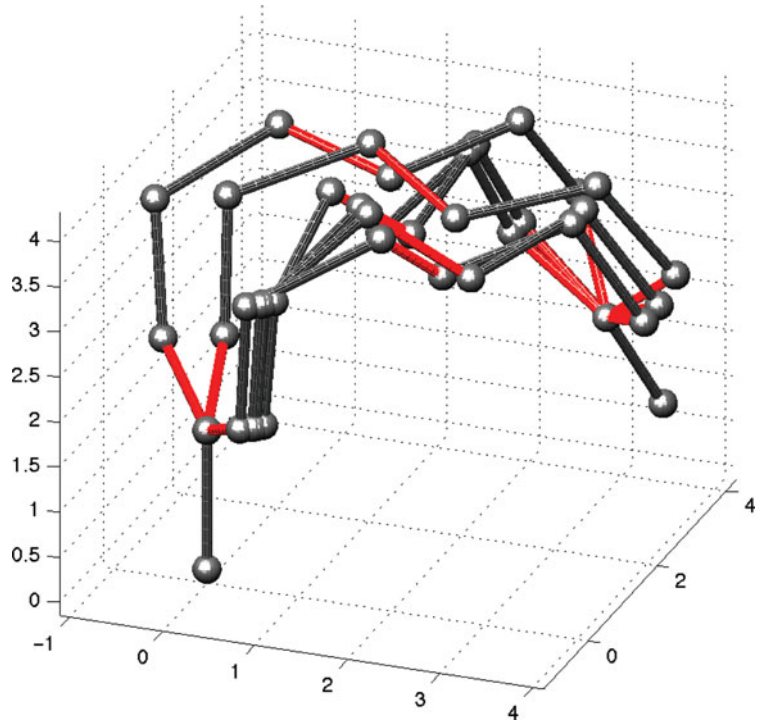

Fig. 6. The inverse kinematic solutions in the example 3. Red-colored bonds (color online) denote $C - N$ bond in polypeptide unit structure as shown in Fig. 5.

5.4. Example 4

As the final example, we consider the structure of tryptophan zipper (trpzip2, PDB code: 1LE1) which was used in protein folding simulations by Monte Carlo algorithm.[60,61] Given the whole structure, we randomly select eight three-peptide units (three consecutive residues), together with $N - C_\alpha$ and $C_\alpha - C$ bonds attached at the proximal and distal ends of the selected three-residue structures, which are used to calculate the initial and final poses according to DH parameterization (see Manocha, *et al.*[25] for more details). Also, we calculate offsets L_i and twist angles α_i ($i = 1, \ldots, 6$) from PDB coordinates of each structure based on the assumption that all four atoms C_α, C, N and C_α in a peptide unit are in the same plane.[25] Then, we solve for the inverse kinematics problem. Table XII shows the number of solutions and average computing time for each structure. As done earlier, computing time was calculated as the average of 20 trials given each three-residue structure. Note that in the first trial case, the extended elimination method gives two solutions (as shown in the parentheses),

Table XIII. Results of example 4 (tripzip2 case): perturbed structures.

	The extended method		Jacobian method	
Trials	# of solutions	Computing time	# of solutions	Computing time
1	2	2.5613×10^{-3} ($\pm 7.1410 \times 10^{-5}$)	2	0.1115 ($\pm 3.5357 \times 10^{-3}$)
2	6	2.7044×10^{-3} ($\pm 3.7720 \times 10^{-4}$)	6	0.0940 ($\pm 2.6699 \times 10^{-3}$)
3	4	2.6775×10^{-3} ($\pm 3.4480 \times 10^{-4}$)	4	0.1051 ($\pm 3.2846 \times 10^{-3}$)
4	2	2.6183×10^{-3} ($\pm 7.6130 \times 10^{-5}$)	2	0.1073 ($\pm 1.7974 \times 10^{-3}$)
5	6	2.6157×10^{-3} ($\pm 9.8490 \times 10^{-5}$)	6	0.1005 ($\pm 2.2602 \times 10^{-3}$)
6	4	2.6847×10^{-3} ($\pm 3.7270 \times 10^{-4}$)	4	0.1031 ($\pm 2.6349 \times 10^{-3}$)
7	4	2.6086×10^{-3} ($\pm 8.4520 \times 10^{-5}$)	4	0.0946 ($\pm 2.6288 \times 10^{-3}$)
8	2	2.6531×10^{-3} ($\pm 2.3180 \times 10^{-4}$)	2	0.1012 ($\pm 2.9837 \times 10^{-3}$)

but one of them is not correct because one of joint angles become very close to 180° (180.01° as obtained by the Jacobian method which could give all correct solutions). When we apply the combined method, then we could obtain two correct solutions again with a computation time almost as good as the extended elimination method alone (marked with an asterisk in Table 5.3). After that, we apply a slight angular perturbation (5°) about $N - C_\alpha$ attached at the proximal end, and apply our methods to obtain the inverse kinematic solutions. This process is to mimic "concerted rotations" moves used in Monte Carlo simulations for chainlike molecules. This small perturbation affects the initial pose of the chosen polypeptide structure, but we keep the final pose fixed, to seek the inverse kinematic solutions. Table XIII shows the results of the perturbation (the number of solutions and average computing time). In total, this example emphasizes the versatility of our methods even in more realistic situations.

6. Conclusions

In this paper, we have presented two methods for finding all the possible conformations of short end-constrained segments of polymers such as polypeptides and polypropylene. These segments have six free joint angles (dihedral angles) with end constraints in position and orientation. As for the first method in this paper, we have adopted and modified concepts from the inverse kinematics of the general $6R$ manipulator, which break down in the case of polymer geometries. We have extended an elimination method based on the work of Kohli and Osvatic which is suitable for the degenerate geometry that polymers usually have. We have also developed a heuristic Jacobian-based method, which can be implemented easily not only for a polymer but also for the general $6R$ manipulator in that it does not require any symbolic computations. This method utilizes an artificial path in Euclidean motion group. We compared the computational performance of these methods and found that the extended elimination method is computationally more favorable than the Jacobian method. However, when it comes to the kinematically degenerate cases such as when some of the joint angles are equal to 180°, the Jacobian method gives accurate solutions whereas the extended elimination method cannot. We have also presented the combined method which utilizes the extended method and the Jacobian method efficiently in degenerate cases. This method, which could generate all correct solutions in degenerate cases, is shown to be as efficient as the extended method in terms of computational speed. We have demonstrated this usefulness with appropriate numerical examples. We expect that our method can be applied to the protein folding algorithms and protein engineering such as drug design.

Acknowledgments

This work was supported by the National Institute of General Medical Sciences of the National Institutes of Health under award number R01GM113240. G. Chirikjian's contribution to this work was supported by the NSF IR/D program while serving as a program director under an IPA contract. The ideas are expressed by the authors and do not necessarily express those of the NSF.

References

1. G. Favrin, A. Irbäck and F. Sjunnesson, "Monte Carlo update for chain molecules: Biased Gaussian steps in torsional space," *J. Chem. Phys.* **114**, 8154–8158 (2001).
2. J. P. Ulmschneider and W. L. Jorgensen, "Monte Carlo backbone sampling for polypeptides with variable bond angles and dihedral angles using concerted rotations and a Gaussian bias," *J. Chem. Phys.* **118**, 4261–4271 (2003).
3. J. P. Ulmschneider and W. L. Jorgensen, "Monte Carlo backbone sampling for nucleic acids using concerted rotations including variable bond angles," *J. Phys. Chem.* **108**(43), 16883–16892 (2004).
4. D. Frenkel and B. Smit, *Understanding Molecular Simulation* (Academic Press, San Diego, 2002).
5. N. Gō and H. A. Scheraga, "Ring closure and local conformational deformations of chain molecules," *Macromolecules* **3**, 178–187 (1970).
6. N. Gō and H. A. Scheraga, "Ring closure in chain molecules with C_n, I, or S_{2n} symmetry," *Macromolecules* **6**, 273–281 (1973).
7. N. Gō and H. A. Scheraga, "Calculation of the conformation of *cyclo*-hexaglycyl," *Macromolecules* **6**, 525–535 (1973).
8. L. Dodd, T. Boone and D. Theodorou, "A concerted rotation algorithm for atomistic Monte Carlo simulations of polymer melts and glasses," *Mol. Phys.* **78**, 961–996 (1993).
9. E. Knapp, "Long time dynamics of a polymer with rigid body monomer units relating to a protein model: Comparison with the Rouse model," *J. Comput. Chem.* **13**, 793–798 (1992).
10. E. Knapp and A. Irgens-Defregger, "Off-lattice Monte Carlo method with constraints: Long-time dynamics of a protein model without nonbonded interactions," *J. Comput. Chem.* **14**, 19–29 (1993).
11. E. A. Coutsias, C. Seok, M. P. Jacobson and K. A. Dill, "A kinematic view of loop closure," *J. Comput. Chem.* **25**, 510–528 (2004).
12. C. H. Mak, "RNA conformational sampling: 1. Single-nucleotide loop closure," *J. Comput. Chem.* **29**, 926–933 (2008).
13. C. H. Mak, W-Y. Chung and N. D. Markovskiy, "RNA conformational sampling II: Arbitrary length multinucleotide loop closure," *J. Chem. Theory Comput.* **7**, 1198–1207 (2011).
14. W. Wedemeyer and H. A. Scheraga, "Exact analytical loop closure in prsotein using polynomial equations," *J. Comput. Chem.* **20**, 819–844 (1999).
15. H. Lee and C. Liang, "A new vector theory for the analysis of spatial mechanism," *Mech. Mach. Theory* **23**, 209–217 (1988).
16. H. Lee and C. Liang, "Displacement analysis of the general spatial 7-link 7R mechanism," *Mech. Mach. Theory* **23**, 219–226 (1988).
17. D. Manocha and J. Canny, "Efficient inverse kinematics for general 6R manipulators," *IEEE Trans. Robot. Autom.* **10**, 648–657 (1994).
18. M. G. Wu and M. W. Deem, "Analytical rebridging Monte Carlo: Application to *cis/trans* isomerization in proline-containing, cyclic peptides," *J. Chem. Phys.* **111**, 6625–6632 (1999).
19. J. Duffy and C. Crane, "A displacement analysis of the general spatial 7R mechanisms," *Mech. Mach. Theory* **15**, 153–169 (1980).
20. L-W. Tsai and A. Morgan, "Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods," *ASME J. Mech. Transm. Autom. Des.* **107**, 189–200 (1985).
21. R. Manseur and K. L. Doty, "A robot manipulator with 16 real inverse kinematic solution sets," *Int. J. Robot. Res.* **8**(5), 75–79 (1989).
22. C. Wampler and A. Morgan, "Solving the 6R inverse position problem using a generic-case solution methodology," *Mech. Mach. Theory* **26**(1), 91–106 (1991).
23. M. Raghavan and B. Roth, "Kinematic analysis of the 6R manipulator of general geometry," **In**: *Proceedings of the 5th International Symposium on Robotics Research* (H. Miura and S. Airmoto, eds.) (MIT Press, Cambridge, MA, 1990) pp. 263–270.
24. M. Raghavan and B. Roth, "Kinematic analysis of the 6R manipulator and related linkages," *ASME J. Mech. Des.* **115**, 502–508 (1993).
25. D. Manocha, Y. Zhu and W. Wright, "Conformational analysis of molecular chains using nano-kinematics," *Comput. Appl. Biosci.* **11**, 71–86 (1995).
26. D. Kohli and M. Osvatic, "Inverse kinematics of the general 6R and 5R, P serial manipulators," *ASME J. Mech. Des.* **115**, 922–931 (1993).
27. M. Ghazvini, "Reducing the Inverse Kinematics of Manipulators to the Solution of a Generalized Eigenproblem," **In**: *Computational Kimematics* (J. Angeles, et al., ed.) (Kluwer Academic Publishers, Springer, Netherlands, 1993) pp. 15–26.
28. J. Nielson and B. Roth, "On the kinematic analysis of robotic mechanisms," *Int. J. Robot. Res.* **12**(12), 1147–1160 (1999).
29. M. L. Husty, M. Pfurner and H-P. Schröcker, "A new and efficient algorithm for the inverse kinematics of a general serial 6R manipulator," *Mech. Mach. Theory* **42**(1), 66–81 (2007).
30. T. Rudny, "Solving inverse kinematics by fully automated planar curves intersecting," *Mech. Mach. Theory* **74**, 310–318 (2014).
31. M. Spong and M. Vidyasagar, *Robot Dynamics and Control* (John Wiley and Sons, New York, 1989).
32. G. Golub and C. Van Loan, *Matrix Computations* (The Johns Hopkins University Press, Baltimore, 1996).
33. E. Anderson *et al.*, *LAPACK User's Guide* (SIAM, Philadelphia, PA, 1999).

34. M. McCarthy, *An Introduction to Theoretical Kinematics* (MIT Press, Cambridge, MA, 1990).
35. M. Murray, Z. Li and S. Sastry, *A Mathematical Introduction to Robotic Manipulation* (CRC Press, Boca Raton, 1994).
36. G. S. Chirikjian and A. B. Kyatkin, *Engineering Applications of Noncommutative Harmonic Analysis* (CRC Press, Boca Raton, FL, 2001).
37. A. J. Sommese and C. W. Wampler, *The Numerical Solution to Systems of Polynomials Arising in Engineering and Science* (World Scientific, Singapore, 1985).
38. A. Morgan, *Solving Polynomial Systems Using Contiuation For Engineering and Scientific Problems* (Prentice-Hall, New Jersey, 1987).
39. I. Ebert-Uphoff and G. S. Chirikjian, "Inverse kinematics of discretely actuated hyper-redundant manipulators using workspace densities," *Proceedings of the IEEE International Conference on Robotics and Automation* (Minneapolis, MN, 1996) pp. 139–145.
40. G. S. Chirikjian, "Inverse kinematics of binary manipulators using a continuum model," *J. Intell. Robot. Syst.* **19**, 5–22 (1997).
41. J. Suthakorn and G. S. Chirikjian, "A new inverse kinematics algorithm for binary manipulators with many actuators," *Adv. Robot.* **15**(2), 225–244 (2001).
42. Y. Wang and G. S. Chirikjian, "Workspace generation of hyper-redundant manipulators as a diffusion process on SE(N)," *IEEE Trans. Robot. Autom.* **20**(3), 399–408 (2004).
43. A. B. Kyatkin and G. S. Chirikjian, "Computation of robot configuration and workspaces via the fourier transform on the discrete motion group," *Int. J. Robot. Res.* **18**(6), 601–615 (1999).
44. Y. Wang, "A fast workspace-density-driven inverse kinematics method for hyper-redundant manipulators," *Robotica* **24**, 649–655 (2006).
45. G. S. Chirikjian, "Conformational statistics of macromolecules using generalized convolution," *Comput. Theor. Polym. Sci.* **11**, 143–153 (2001).
46. J. S. Kim and G. S. Chirikjian, "A unified approach to conformational statistics of classical polymer and polypeptide models," *Polymer* **46**, 11904–11917 (2005).
47. D. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Mach. Syst.* **MMS-10**(2), 47–53 (1969).
48. J. Uicker, J. Denavit and R. Hartenberg, "An iterative method for the displacement analysis of spatial mechanisms," *ASME J. Appl. Mech.* **107**, 189–200 (1954).
49. T. Isobe, K. Nagasaka and S. Yamamoto, "A new approach to kinematic control of simple manipulators," *IEEE Trans. Syst. Man Cybern.* **22**(5), 1116–1124 (1992).
50. L. Wang and C. Chen, "A combined optimization method for solving inverse kinematics problem of mechanical manipulators," *IEEE Trans. Robot. Autom.* **7**(4), 489–499 (1991).
51. A. L. Olsen and H. G. Petersen, "Inverse kinematics by numerical and analytical cyclic coordinate descent," *Robotica* **29**(4), 619–626 (2011).
52. A. Canutescu and R. Dunbrack, "Cyclic coordinate descent: A robotic algorithm for protein loop closure," *Protein Sci.* **12**, 963–972 (2003).
53. W. Boomsma and T. Hamelryck, "Full cyclic coordinate descent: Solving the protein loop closure problem in C_α space," *BMS Bioinformatics* **6**, 159 (2005).
54. K. Al-Nasr and J. He, "An effective convergence independent loop closure method using forward-backward cyclic coordinate descent," *Int. J. Data Min. Bioinformatics* **3**(3), 346–361 (2009).
55. J. S. Kim and G. S. Chirikjian, "Conformational analysis of stiff chiral polymers with end-constraints," *Mol. Simul.* **32**(14), 1139–1154 (2006).
56. F. C. Park, "Distance metrics on the rigid-body motions with applications to mechanism design," *J. Mech. Des.* **117**, 48–54 (1995).
57. G. S. Chirikjian and S. Zhou, "Metrics on motion and deformation of solid models," *J. Mech. Des.* **120**(2), 252–261 (1998).
58. G. S. Chirikjian and Y. Yan, "Mathematical aspects of molecular replacement. II. Geometry of motion spaces," *Acta Cryst.* **A68**, 208–221 (2012).
59. B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter, *Molecular Biology of the Cell* (Garland Science, New York, 2000).
60. J. P. Ulmschneider and W. L. Jorgensen, "Polypeptide folding using Monte Carlo sampling, concerted rotation, and continuum solvation," *J. Am. Chem. Soc.* **126**, 1849–1857 (2004).
61. J. P. Ulmschneider and W. L. Jorgensen, "Monte Carlo vs molecular dynamics for all-atom polypeptide folding simulations," *J. Phys. Chem. B* **110**, 16733–16742 (2006).