Inverse Kinematics of Binary Manipulators by Using the Continuous-Variable-Based Optimization Method

Yoon Young Kim, Gang-Won Jang, and Sang Jun Nam

***Abstract*—Hyper redundancy, high reliability, and high task repeatability are the main advantages of binary manipulators over conventional manipulators with continuous joints, especially when manipulators are operated under tough and complex work conditions. The precise and complex movement of a binary manipulator necessitates many modules. In this case, numerically efficient inverse kinematics algorithms for binary manipulators usually require impractically large memory size for the real-time calculation of the binary states of all joints. To overcome this limitation by developing a new inverse kinematics algorithm is the objective of this research. The key idea of the proposed method is to formulate the inverse kinematics problem of a binary manipulator as an optimization problem with real design variables, in which the real variables are forced to approach the permissible binary values corresponding to two discrete joint displacements. Using the proposed optimization method, the inverse kinematics of 3-D binary manipulators with many modules can be solved almost in real time (say, less than a second for up to 16 modules) without requiring a large memory size. Furthermore, some manipulation considerations, such as operation power minimization, can be easily incorporated into the proposed formulation. The effectiveness of the proposed method is verified through several numerical problems, including 3-D inverse kinematics problems.**

***Index Terms*—Binary manipulator, continuous variable optimization, inverse kinematics, real-time simulation.**

I. Introduction

A BINARY manipulator consists of serially connected modules that are composed of binary joints having two stable states. Because each joint has only two states, the workspace of a binary manipulator consists of finite discrete points only. The advantages of a binary manipulator are, however: 1) simplicity without the complex electronic circuit for feedback control; 2) high accuracy for repeated work; 3) high reliability for tough work due to hyper redundancy; 4) high stability for static loads; and 5) light mass. The binary manipulators have been considered for space exploration [1]–[5], medical, and other applications.

The inverse kinematics of a binary manipulator may not be treated as a direct extension of the inverse kinematics of a manipulator with continuous joints. If joint states were assumed to be continuous, solving the inverse kinematics for continuous joint state variables would be simpler. However, replacing the state values obtained for the manipulator with continuous joints by the nearest binary state values would locate the end-effector position of the binary manipulator quite off from the target position. The nonlinear relation between the joint states and the end-effector position may worsen the situation. When the binary joint states were directly used, on the other hand, the number of reachable points of a binary manipulator would increase exponentially as the number of modules increases. Therefore, it will be quite inefficient to make a list of all the joint statuses and their reachable targets for the inverse kinematics analysis. For instance, there exist $2^{3\times 7} = 2097152$ configurations for a binary manipulator with seven three-joint modules.

The most widely used algorithm for the inverse kinematics of a binary manipulator was developed by Chirikjian and his colleagues [6]–[10]. In their approach, the workspace of a binary manipulator is discretized into a finite number of pixels, and a workspace density is calculated for each pixel. Although their algorithm is very fast, it requires a large amount of computer memory for the discretized workspace. For example, if a 3-D workspace is discretized into $100 \times 100 \times 80$ pixels, and the orientation for each pixel is discretized every 10 degrees about the three orthogonal axes, a rough estimate of the total memory size is calculated to be

$$100 \times 100 \times 80 \times 36 \times 36 \times 36 \times 4 \text{ bytes} \approx 0.42 \text{ terabytes (integer type)}$$

which is almost impractical to deal with by current computer resources. To circumvent the memory limitation of the Ebert–Uphoff algorithm, Suthakorn and Chirikjian proposed a modified algorithm considering only the center of a workspace [11]. However, the accuracy of their algorithm is heavily dependent on the shape of modules and the workspace configuration. Another approach to the inverse kinematics of a binary manipulator is to use backbone curves to prescribe the main shape of the manipulator [12], [13].

In this paper, a totally different approach for the inverse kinematics of a binary manipulator is attempted. The inverse kinematics problem is treated as an optimization problem in a continuous variable space. The following is the essential idea of the proposed approach.

- The joint displacements are assumed to be continuous, and they are used as design variables of the optimization problem.
- Target tracing is imposed as a constraint equation of the optimization problem. Namely, the distance between the

Manuscript received January 18, 2005; revised June 12, 2005. This paper was recommended for publication by Associate Editor S. Ma and Editor I. Walker upon evaluation of the reviewers' comments. This work was supported by the Creative Research Initiatives Program of the Korea Ministry of Science and Technology. This paper was presented in part at the International Conference on Intelligent Robots and Systems, Sendai, Japan, September–October, 2004.

Y. Y. Kim and S. J. Nam are with the School of Mechanical and Aerospace Engineering and National Creative Research, Initiatives Center for Multiscale Design, Seoul National University, Seoul 151-742, Korea (e-mail: yykim@snu.ac.kr).

G.-W. Jang is with the School of Mechanical Engineering, Kunsan National University, Kunsan, Chonbuk 573-701, Korea.

Digital Object Identifier 10.1109/TRO.2005.858864

target and the end-effector should be smaller than or equal to a prescribed error.

- Pushing the continuous design variables to permissible binary values is stated as the optimization objective function. By employing a special mapping function, the design variables are effectively pushed toward the binary values during the optimization process.
- The optimization problem is solved by an efficient gradient-based optimizer for real-time calculation.

In the proposed algorithm, the numerical cost for the optimization process is dependent only on the number of design variables, i.e., the number of binary joints, not on the dimensions of the workspace. Therefore, the inverse kinematics problem of a 3-D binary manipulator with many modules, which is almost impossible to solve by the Ebert–Uphoff algorithm, can be handled by the proposed method.

If the continuous design variables are not sufficiently close to binary values, choosing the binary values nearest to the obtained continuous values usually causes a large error in the distance between the target and the end-effector. Therefore, it is very important to obtain the optimized continuous joint values close to the binary statuses. To achieve this goal, the objective of the optimization is set as minimizing the differences between joint displacements and binary states. A nonlinear mapping function used in other types of optimization problems [14], [15] is adopted to accelerate the convergence of the joint displacements to their binary states.

An ideal solution to avoid intermediate design variables is to directly use discrete design variables by employing nonlinear integer programming methods. Various nonlinear integer programming algorithms, such as branch-and-bound methods, dynamic programming methods, and their combinations [16]–[18] are available. However, these methods cannot solve large-size problems in real time (less than a few seconds). Alternatively, one may employ genetic algorithms to solve the inverse kinematics problem, as in [2] and [4]. In this paper, however, the inverse kinematics problem will be solved by a gradient-based optimization method.

To check the effectiveness of the proposed real-variable optimization method, several inverse kinematics problems for 2-D and 3-D binary manipulators are investigated. The orientation, as well as the position, of a target is considered for the inverse kinematics. The performance of the proposed method is checked in terms of the computation time.

II. Fundamentals

A. Configuration of a Binary Manipulator

Fig. 1 shows a 2-D binary manipulator having three-joint (or three-bit) modules [6], [12]. Frame F_i is attached to the center of the $(i+1)$th module bottom. The inertial reference frame and the end-effector frame are denoted as F_0 and F_B, respectively. All kinematic configurations of the ith module shown in Fig. 2(a) are illustrated in Fig. 2(b). It may be convenient to introduce the configuration set C_i of an N-dimensional binary manipulator such that

$$C_i = \left\{ g_i^1, g_i^2, \ldots, g_i^{K_i} \right\} \tag{1}$$

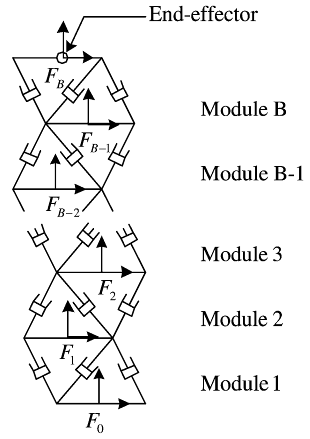
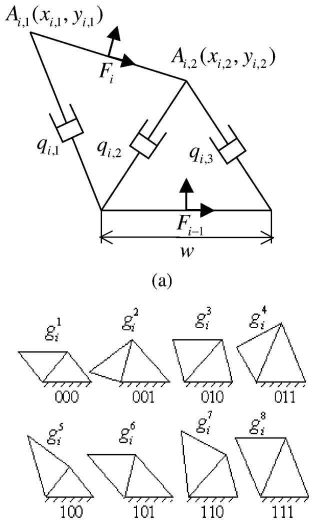


Fig. 1. 2-D binary manipulator with B three-bit modules.



(a)

(b)

Fig. 2. (a) 2-D module. (b) Its kinematic configuration C_i.

where K_i is the number of the configurations. In (1), g_i^j, an element of SE(N),[1] can be expressed in terms of the position vector $\mathbf{b}_i^j \in \mathbb{R}^N$, and the rotation matrix $\mathbf{R}_i^j \in$ SO(N)[2] [$K_i = 8$ and $N = 2$ for the module shown in Fig. 2(a)]

$$g_i^j = \begin{pmatrix} \mathbf{R}_i^j & \mathbf{b}_i^j \\ \mathbf{0} & 1 \end{pmatrix} \in \mathrm{SE}(N) \tag{2}$$

which will be also written as $g_i^j = (\mathbf{b}_i^j, \mathbf{R}_i^j)$.

Since the number of points that an end-effector can reach is finite, a prescribed target $g_p = (\mathbf{b}_p, \mathbf{R}_p)$ may not be reached by the end-effector of a binary manipulator. However, one can find the position and the orientation $g_B = (\mathbf{b}_B, \mathbf{R}_B)$ of the end-effector that satisfy the following condition:

$$D(g_B, g_p) = \sqrt{\|\mathbf{b}_B - \mathbf{b}_p\|^2 + L^2 \left\|\log \mathbf{R}_B^T \mathbf{R}_p\right\|^2} \leq \varepsilon \tag{3}$$

where $\|\cdot\|$ is the Euclidean norm[3] (see the Appendix), and a positive real number $\varepsilon \in \mathbb{R}^+$ is a given tolerance. In (3), L is a parameter mainly introduced to match units of the squared

[1] Special Euclidean groups of dimension N or a Euclidean group of rigid body motions.

[2] Special orthogonal groups of dimension N

$$\mathrm{SO}(N) = \{\mathbf{R} \in \mathbb{R}^{N \times N} : \mathbf{R}\mathbf{R}^{\mathrm{T}} = \mathbf{I}, \det \mathbf{R} = 1\}.$$

[3] The distance metric in (3) is introduced by Park [19].

terms. For actual implementation, the value of $L = 0.1$ was used. Other values of L may be used, but the value of $L = 0.1$ gave a good balance between the position error and the orientation error. Obviously, it is possible to obtain more than one g_B satisfying (3) due to the redundancy of the binary manipulator.

B. Forward Kinematics of a 2-D Binary Manipulator

In Fig. 2(a), the lengths of the joints are denoted as $q_{i,1}$, $q_{i,2}$, and $q_{i,3}$. The coordinates of the vertices $A_{i,1}$ and $A_{i,2}$ in the frame F_{i-1} are expressed as (see [20])

$$\begin{aligned} x_{i,1} &= \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \\ y_{i,1} &= \left(q_{i,1}^2 - \left(x_{i,1} - \frac{w}{2}\right)^2\right)^{\frac{1}{2}} \end{aligned} \tag{4a}$$

$$\begin{aligned} x_{i,2} &= \frac{q_{i,2}^2 - q_{i,3}^2}{2w} \\ y_{i,2} &= \left(q_{i,2}^2 - \left(\frac{q_{i,2}^2 - q_{i,3}^2 + w^2}{2w}\right)^2\right)^{\frac{1}{2}} \end{aligned} \tag{4b}$$

where

$$\begin{aligned} a &= (2x_{i,2} + w)^2 + 4y_{i,2}^2 \\ b &= -(4x_{i,2} + 2w)\left(x_{i,2}^2 - q_{i,1}^2 - \frac{5}{4}w^2 + y_{i,2}^2\right) + 4wy_{i,2}^2 \\ c &= \left(x_{i,2}^2 + q_{i,1}^2 - \frac{5}{4}w^2 + y_{i,2}^2\right)^2 - 4y_{i,2}^2\left(q_{i,1}^2 - \frac{1}{4}w^2\right). \end{aligned}$$

In the above, w is the width of the upper and lower base plates of the module. The position vector $\mathbf{b}_i$ and the orientation angle θ_i of F_i with respect to F_{i-1} are

$$\begin{aligned} \mathbf{b}_i &= \left(\frac{(x_{i,1} + x_{i,2})}{2}, \frac{(y_{i,1} + y_{i,2})}{2}\right) \\ \theta_i &= \tan^{-1}\frac{y_{i,2} - y_{i,1}}{x_{i,2} - x_{i,1}}. \end{aligned} \tag{5}$$

We use the following notation $g_{i-1,i}$ to express the configuration of F_i with respect to F_{i-1}:

$$g_{i-1,i} = \begin{pmatrix} \mathbf{R}_i & \mathbf{b}_i^{\mathrm{T}} \\ \mathbf{0} & 1 \end{pmatrix} \tag{6a}$$

where

$$\mathbf{R}_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{pmatrix}. \tag{6b}$$

Finally, the position and the orientation of the frame F_B relative to the base frame F_0 can be written in terms of $g_{i-1,i}$ as

$$\begin{aligned} g_B &= \prod_{i=1}^{B} g_{i-1,i} \\ &= \begin{pmatrix} \mathbf{R}_1 & \mathbf{b}_1^{\mathrm{T}} \\ \mathbf{0} & 1 \end{pmatrix}\begin{pmatrix} \mathbf{R}_2 & \mathbf{b}_2^{\mathrm{T}} \\ \mathbf{0} & 1 \end{pmatrix}\cdots\begin{pmatrix} \mathbf{R}_B & \mathbf{b}_B^{\mathrm{T}} \\ \mathbf{0} & 1 \end{pmatrix}. \end{aligned} \tag{7}$$

C. Forward Kinematics of a 3-D Binary Manipulator

Fig. 3 illustrates a three-bit module for a 3-D binary manipulator [2]–[5]. In the figure, revolute joints are used in the lower plate, and the rotation axes of the joints are in the direction of $\overrightarrow{B_{i,j}C_{i,j}} \times \overrightarrow{B_{i,j}O_{i-1}}$. Spherical joints are used in the upper plate, so the binary joints can rotate freely in all directions.

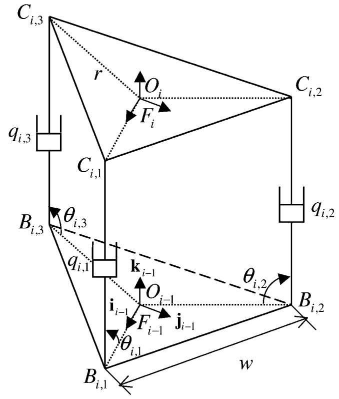


Fig. 3. 3-D module.

The direction vectors $\mathbf{q}_i$ of the binary joints are introduced to simplify the subsequent expressions

$$\begin{aligned} \mathbf{q}_{i,1} &= \frac{\overrightarrow{B_{\mathrm{i,1}}C_{\mathrm{i,1}}}}{|\overrightarrow{B_{\mathrm{i,1}}C_{\mathrm{i,1}}}|} = -\cos(\theta_{i,1})\mathbf{i}_{i-1} + \sin(\theta_{i,1})\mathbf{k}_{i-1} \\ \mathbf{q}_{i,2} &= \frac{\overrightarrow{B_{\mathrm{i,2}}C_{\mathrm{i,2}}}}{|\overrightarrow{B_{\mathrm{i,2}}C_{\mathrm{i,2}}}|} \\ &= \frac{1}{2}\cos(\theta_{i,2})\mathbf{i}_{i-1} - \frac{\sqrt{3}}{2}\cos(\theta_{i,2})\mathbf{j}_{i-1} + \sin(\theta_{i,2})\mathbf{k}_{i-1} \\ \mathbf{q}_{i,3} &= \frac{\overrightarrow{B_{\mathrm{i,3}}C_{\mathrm{i,3}}}}{|\overrightarrow{B_{\mathrm{i,3}}C_{\mathrm{i,3}}}|} \\ &= \frac{1}{2}\cos(\theta_{i,3})\mathbf{i}_{i-1} + \frac{\sqrt{3}}{2}\cos(\theta_{i,3})\mathbf{j}_{i-1} + \sin(\theta_{i,3})\mathbf{k}_{i-1} \end{aligned} \tag{8}$$

where $\mathbf{i}_{i-1}$, $\mathbf{j}_{i-1}$, and $\mathbf{k}_{i-1}$ are the unit vectors of F_{i-1}.

Using (8), the coordinates of the vertices $C_{i,1}$, $C_{i,2}$, and $C_{i,3}$ in F_{i-1} can be expressed as

$$\begin{aligned} \overrightarrow{O_{i-1}C_{i,1}} &= \{r - q_{i,1}\cos(\theta_{i,1})\}\mathbf{i}_{i-1} + q_{i,1}\sin(\theta_{i,1})\mathbf{k}_{i-1} \\ \overrightarrow{O_{i-1}C_{i,2}} &= \frac{1}{2}\{-r + q_{i,2}\cos(\theta_{i,2})\}\mathbf{i}_{i-1} \\ &\quad + \frac{\sqrt{3}}{2}\{-r - q_{i,2}\cos(\theta_{i,2})\}\mathbf{j}_{i-1} \\ &\quad + q_{i,2}\sin(\theta_{i,2})\mathbf{k}_{i-1} \\ \overrightarrow{O_{i-1}C_{i,3}} &= \frac{1}{2}\{-r + q_{i,3}\cos(\theta_{i,3})\}\mathbf{i}_{i-1} \\ &\quad - \frac{\sqrt{3}}{2}\{-r - q_{i,3}\cos(\theta_{i,2})\}\mathbf{j}_{i-1} \\ &\quad + q_{i,3}\sin(\theta_{i,3})\mathbf{k}_{i-1} \end{aligned} \tag{9}$$

where r is the distance from the center of the upper or the lower plate to its vertices. The solution $\theta_{i,j}$ satisfying the following constraint condition can be found by using the Newton–Raphson method:

$$|\overrightarrow{C_{i,1}C_{i,2}}| = |\overrightarrow{C_{i,2}C_{i,3}}| = |\overrightarrow{C_{i,3}C_{i,1}}| = w. \tag{10}$$

The maximum residual error of the Newton–Raphson method satisfying three equations of (10) was set to be 10^{-3} for all numerical problems. The Newton–Raphson method was used first in [2] to solve the forward kinematics of a 3-D structure.

By substituting the solution $\theta_{i,j}$ of (10) into (9), one can find the position vectors of the spherical joints. Then, the position $\mathbf{b}_i$ and the orientation $\mathbf{R}_i$ of F_i can be expressed in terms of $\overrightarrow{O_{i-1}C_{i,1}}$, etc., as

$$\mathbf{b}_i = \frac{\overrightarrow{O_{i-1}C_{i,1}} + \overrightarrow{O_{i-1}C_{i,2}} + \overrightarrow{O_{i-1}C_{i,3}}}{3} \tag{11a}$$

$$\mathbf{R}_i = \left[\mathbf{x}_i^{\mathbf{T}}, \mathbf{y}_i^{\mathbf{T}}, \mathbf{z}_i^{\mathbf{T}}\right] \tag{11b}$$

where $(\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i)$ are the unit vectors of F_i expressed in F_{i-1}

$$\mathbf{x}_i = \frac{\overrightarrow{O_iC_{i,1}}}{|\overrightarrow{O_iC_{i,1}}|} = \frac{\overrightarrow{O_{i-1}C_{i,1}} - \overrightarrow{O_{i-1}O_i}}{|\overrightarrow{O_{i-1}C_{i,1}} - \overrightarrow{O_{i-1}O_i}|} \tag{12a}$$

$$\begin{aligned}\mathbf{z}_i &= \frac{\overrightarrow{O_iC_{i,1}} \times \overrightarrow{O_iC_{i,2}}}{|\overrightarrow{O_iC_{i,1}} \times \overrightarrow{O_iC_{i,2}}|}\\ &= \frac{\{\overrightarrow{O_{i-1}C_{i,1}} - \overrightarrow{O_{i-1}O_i}\} \times \{\overrightarrow{O_{i-1}C_{i,2}} - \overrightarrow{O_{i-1}O_i}\}}{|\{\overrightarrow{O_{i-1}C_{i,1}} - \overrightarrow{O_{i-1}O_i}\} \times \{\overrightarrow{O_{i-1}C_{i,2}} - \overrightarrow{O_{i-1}O_i}\}|}\end{aligned} \tag{12b}$$

$$\mathbf{y}_{\mathrm{i}} = \mathbf{z}_i \times \mathbf{x}_i. \tag{12c}$$

The position and the orientation of the end-effector frame F_B relative to the base frame F_0 can be obtained by (7).

III. Proposed Inverse Kinematics by the Optimization Method

In this section, a new method to solve the inverse kinematics of binary manipulators will be presented. The problem to find the length $q_{i,j}$ of the binary joints to satisfy (3) will be formulated as an optimization problem. The length $q_{i,j}$ should be either $q_{i,j}^{\max}$ or $q_{i,j}^{\min}$. If the inverse kinematics problem is directly converted to an optimization problem with binary design variables, the resulting optimization problem will be almost impossible to solve within a reasonable computation time. The main contribution of this paper is to develop an efficient optimization-based inverse kinematics algorithm.

A. Problem Formulation

Our strategy is schematically described in Fig. 4. In the proposed approach, the binary variable design space is replaced by a continuous variable design space, in which an efficient gradient-based search method to find the values of $q_{i,j}$ can be used. The main hurdle in this relaxation is to ensure that the values of the design variables at the final optimal state become binary or "binary enough." So the success of the approach depends on how closely the optimized continuous design variables reach $q_{i,j}^{\max}$ or $q_{i,j}^{\min}$. Although we have considered several strategies, the following two strategies will be mainly considered here.

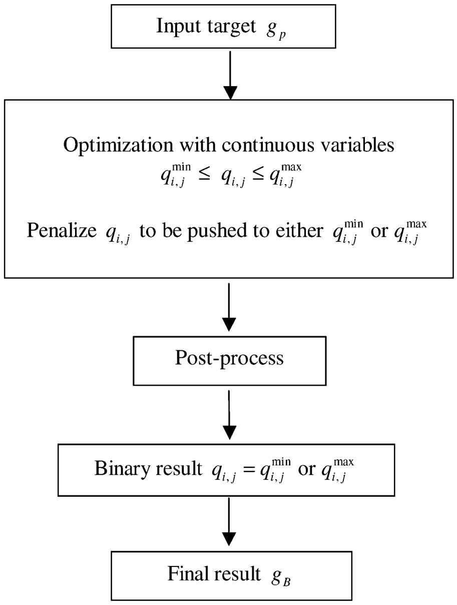

Input target g_p
Optimization with continuous variables
$q_{i,j}^{\min} \le q_{i,j} \le q_{i,j}^{\max}$
Penalize $q_{i,j}$ to be pushed to either $q_{i,j}^{\min}$ or $q_{i,j}^{\max}$
Post-process
Binary result $q_{i,j} = q_{i,j}^{\min}$ or $q_{i,j}^{\max}$
Final result g_B

Fig. 4. Schematic description of the proposed optimization-based method for the inverse kinematics of binary manipulators.

1) Optimization Strategy A (OS-A):

$$\underset{\mathbf{Q}}{\text{Minimize}}\ f = \sum_{i=1}^{B}\sum_{j=1}^{3} \alpha_i \left(q_{i,j}^{\max} - q_{i,j}\right)\left(q_{i,j} - q_{i,j}^{\min}\right) \tag{13a}$$

$$\text{subject to } h = D(g_B, g_p) \le \varepsilon \tag{13b}$$

where

$$\begin{aligned}\mathbf{Q} = \{&(q_{1,1}, q_{1,2}, q_{1,3}), (q_{2,1}, q_{2,2}, q_{2,3}), \ldots,\\ &(q_{B,1}, q_{B,2}, q_{B,3})\}\end{aligned} \tag{14a}$$

$$q_{i,j}^{\min} \le q_{i,j} \le q_{i,j}^{\max}, \quad (i = 1, 2, \ldots, B \text{ and } j = 1, 2, 3). \tag{14b}$$

In Optimization Strategy A (OS-A), the variables $q_{i,j}$ are treated as continuous variables as expressed in (14b). Note that the objective function f is to push continuous variables $q_{i,j}$ toward the permissible two discrete values $q_{i,j}^{\max}$ or $q_{i,j}^{\min}$. Furthermore, the distance between the prescribed target $g_p = (\mathbf{b}_p, \mathbf{R}_p)$ and the end-effector is treated as a constraint.

Instead of using (13), one may consider the following alternative formulation:

$$\underset{\mathbf{Q}}{\text{Minimize}}\ h = D(g_B, g_p)$$

$$\text{subject to } f = \sum_{i=1}^{B}\sum_{j=1}^{3} \alpha_i \left(q_{i,j}^{\max} - q_{i,j}\right)\left(q_{i,j} - q_{i,j}^{\min}\right) \le \varepsilon^*.$$

The disadvantage of using the above formulation is that local minima far from being sufficiently close to the target can be obtained only if the solution satisfies the constraint $f \le \varepsilon^*$. When (13) are employed, on the other hand, the closeness of the end-effector to the target can be controlled by the prescribed tolerance ε. Since the workspace of a binary manipulator is discrete, arbitrary targets may not be exactly reached by a binary

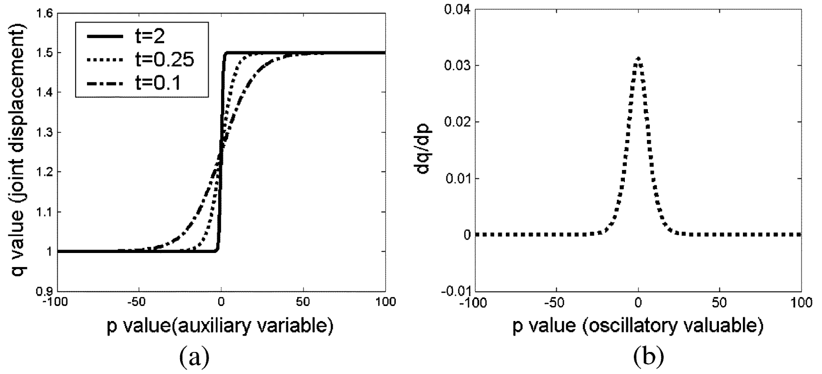


Fig. 5. (a) S-shaped nonlinear function mapping $p_{i,j}$ to $q_{i,j}$. (b) $dq_{i,j}/dp_{i,j}$ for $t = 0.25$ in (17c).

manipulator. Therefore, a point nearest to a given target satisfying (13b) will be obtained. Obviously, smaller values of ε can be used if more modules are used.

After the optimization is completed, some design variables may not be close enough to $q_{i,j}^{\max}$ or $q_{i,j}^{\min}$. In this case, they are postprocessed to take the binary values. If nonbinary-valued variables are those of the base module, the position of the end-effector after the postprocess would be significantly different from that before the postprocess; the difference between the computed $q_{i,j}$ and binary values in lower modules would cause a larger error in $D(g_B, g_p)$ than the difference in higher modules. Motivated by the above observation, the following form of the weighting parameter α_i is proposed:

$$\alpha_i = (B - i + 1)^n, \quad n \in \mathbb{R}^+. \tag{15}$$

When α_i in (15) is introduced, the binary condition on the design variables for lower modules is more strictly imposed. Other forms of the weighting parameter, such as $\alpha_i = \exp(-s \cdot i)$ $(s \in \mathbb{R}^+)$, may be used. However, the polynomial function in (15) was observed to outperform some functions considered during our numerical studies. The values of n between one and five were found to yield fast solution convergence.

2) Optimization Strategy B (OS-B):

$$\underset{\mathbf{P}}{\text{Minimize}}\, f = \sum_{i=1}^{B}\sum_{j=1}^{3} \alpha_i \left(q_{i,j}^{\max} - q_{i,j}\right)\left(q_{i,j} - q_{i,j}^{\min}\right) \tag{16a}$$

$$\text{subject to } h = D(g_B, g_p) \le \varepsilon \tag{16b}$$

where

$$\mathbf{P} = \{(p_{1,1}, p_{1,2}, p_{1,3}), (p_{2,1}, p_{2,2}, p_{2,3}), \ldots, (p_{B,1}, p_{B,2}, p_{B,3})\} \tag{17a}$$

$$p_{i,j}^{\min} \le p_{i,j} \le p_{i,j}^{\max}, \quad (i = 1, 2, \ldots, B \text{ and } j = 1, 2, 3) \tag{17b}$$

$$q_{i,j} = \frac{q_{i,j}^{\max} - q_{i,j}^{\min}}{1 + \exp(-t \cdot p_{i,j})} + q_{i,j}^{\min}, \quad t \in \mathbb{R}^+. \tag{17c}$$

While the joint displacements $q_{i,j}$ are directly used as design variables in OS-A, auxiliary variables $p_{i,j}$ are used as design variables in OS-B. Equation (17c) expresses a one-to-one S-shaped nonlinear correspondence between $q_{i,j}$ and $p_{i,j}$, which is plotted in Fig. 5(a). The S-shape function was introduced by Kim and Yoon [14] and Yoon and Kim [15] for structural topology optimization. In [15], it was shown that the high sensitivities near intermediate values [see Fig. 5(b)] help push intermediate variables to lower or upper bounds, and thus speed up solution convergence. The motivation to use (17c) here is exactly the same as in [15], although the optimization problems considered are not the same.

The optimization problems set up as OS-A and OS-B have local minima. Unfortunately, it was difficult to address this issue in an explicit mathematical fashion. However, these local minima are usually good enough to meet practical objectives, in practice.

The sensitivity of the objective function with respect to the auxiliary variable is simply given as

$$\frac{df}{dp_{i,j}} = \frac{df}{dq_{i,j}} \cdot \frac{dq_{i,j}}{dp_{i,j}}. \tag{18}$$

Fig. 5(b) illustrates the behavior of $dq_{i,j}/dp_{i,j}$ appearing in (18). Since $dq_{i,j}/dp_{i,j}$ has high values near $p_{i,j} = 0$, i.e., $q_{i,j} = (q_{i,j}^{\min} + q_{i,j}^{\max})/2$ where the joint has intermediate state values, the sensitivity $df/dp_{i,j}$ has also high values for intermediate joint displacements. Therefore, the intermediate joint values are pushed to move toward either $q_{i,j}^{\max}$ or $q_{i,j}^{\min}$.

To see the effect of the weighting parameter α_i and to compare the behavior of the solution strategies OS-A and OS-B, let us consider first the inverse kinematics problem for a 2-D binary manipulator with 10 modules. For this example, the values of $q_{i,j}^{\max} = 0.15$ and $q_{i,j}^{\min} = 0.1$ will be used. Unless stated otherwise, the initial values of $q_{i,j}$ and $p_{i,j}$ for the optimization problem will be chosen as $(q_{i,j}^{\max} + q_{i,j}^{\max})/2$ for OS-A, and $(p_{i,j}^{\max} + p_{i,j}^{\max})/2$ for OS-B, respectively.

The numerical results are depicted in Fig. 6, and the joint displacements obtained by OS-A and OS-B are listed in Table I. The target was $\mathbf{b}_p = (0.4, 0.9)$ ($\mathbf{R}_p$ not prescribed for simplicity). In Fig. 6, the binary manipulators with solid lines are the results by the optimization before the postprocessing of some intermediate values, and those with dotted lines are the results after the postprocessing. The postprocessing implies the processing to replace the optimized value of $q_{i,j}$ by the nearest upper $(q_{i,j}^{\max})$ or lower $(q_{i,j}^{\max})$ bound. Although the proposed optimization-based strategy does not explicitly control the closeness of $q_{i,j}$ to either $q_{i,j}^{\max}$ or $q_{i,j}^{\max}$, the postprocessing did not cause significant distance errors (for all numerical problems considered in this paper). Indeed, the typical error $D(g_B, g_p)$ after postprocessing was within $k\varepsilon$ (k = 2–3 for 2-D problems,

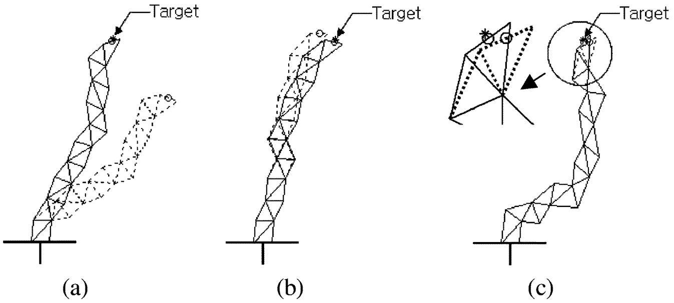


Fig. 6. Comparison of the results by various optimization strategies (solid line: result before postprocessing; dashed line: result after postprocessing). (a) OS-A with $\alpha_i = 1$. (b) OS-A with (15) $(n = 4)$. (c) OS-B with (15) $(n = 4, p_{i,j}^{\max} = 100, p_{i,j}^{\min} = -100$, and $t = 0.25)$.

TABLE I
OPTIMIZED VALUES OF $q_{i,j}$ BEFORE POSTPROCESSING

Module	OS-A (α_i =1)			OS-A (α_i from (15))			OS-B (α_i from (15))		
1	0.1	0.1	0.11	0.1	0.1	0.1	0.15	0.15	0.15
2	0.1	0.15	0.1	0.1	0.15	0.1	0.1	0.1	0.15
3	0.1	0,15	0.11	0.1	0.1	0.11	0.1	0.15	0.15
4	0.1	0.15	0.11	0.1	0.15	0.11	0.1	0.1	0.1
5	0.1	0.15	0.11	0.1	0.14	0.11	0.15	0.15	0.1
6	0.1	0.15	0.1	0.1	0.15	0.1	0.15	0.1	0.1
7	0.15	0.15	0.11	0.14	0.15	0.15	0.15	0.15	0.15
8	0.1	0.15	0.15	0.13	0.15	0.15	0.1	0.15	0.15
9	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15	0.15
10	0.15	0.15	0.15	0.15	0.15	0.15	0.14	0.1	0.15
Iterations	289			392			125		
$D(g_B, g_p)$ after post-processing	0.4153			0.2773			0.0285		

and $k = 8$–10 for 2-D problems). Thus, the tolerance ε used in (16b) should be smaller than the true error by a factor of k, as suggested above. If the postprocessed errors are larger than $k\varepsilon$, the problem is not considered to be converged and should be solved again with a smaller value of ε. However, no such situation was observed in the numerical problems worked out in this paper.

The strategy OS-A using the same values of α_i (i.e., $\alpha_i = 1$) yields many intermediate joint values and produces a large error after the postprocessing [see Fig. 6(a)]. The use of (15) for the weighting parameter shows a much better result for OS-A. The weighting exponent $n = 4$ is used in this example. However, the error $D(g_B, g_p)$ is not still negligible, as can be seen in Fig. 6(b). When OS-B using the auxiliary design variable is used, the error $D(g_B, g_p)$ meets design objectives. Note that the optimized results before and after the postprocessing are almost identical when OS-B with (15) is employed. This is because the nonlinear S-shaped mapping function in (17c) sufficiently pushes $q_{i,j}$ toward either $q_{i,j}^{\max}$ or $q_{i,j}^{\min}$. Table I clearly shows that the joint displacements obtained by OS-B have almost converged to $q_{i,j}^{\max}$ or $q_{i,j}^{\min}$. This example addresses the critical role of the nonlinear mapping function in the continuous-variable-based optimization formulation of the inverse kinematics of the binary manipulator.

When a binary manipulator is required to follow a certain locus, it will be desirable to minimize the number of moving joints. This requirement can easily be reflected in the proposed optimization-based strategy simply by adding an additional penalty term to the objective function. This strategy will be indicated as OS-B'.

3) Optimization Strategy B' (OS-B'):

$$\begin{aligned}\underset{\mathbf{P}}{\text{Minimize}}\ f = \sum_{i=1}^{B}\sum_{j=1}^{3}\{&\alpha_i(q_{i,j}^{\max} - q_{i,j})(q_{i,j} - q_{i,j}^{\min})\\ &+ \beta_i(q_{i,j} - q_{i,j}^{\text{prev}})^2\} \qquad (19\text{a})\\ \text{subject to } h = D(g_B, g_p) \le \varepsilon \qquad &(19\text{b})\end{aligned}$$

where $q_{i,j}^{\text{prev}}$ denotes the joint lengths at the previous state. Since the performance of OS-B works better than OS-A, as tested by the previous example and many other problems not presented here, only the strategy B is considered for locus-following problems. In (19a), an additional weighting parameter β_i is introduced to penalize the joint actuations. Assuming that more power is required to actuate joints of lower modules than those of upper modules, large values of β_i are used for low modules. Several penalization alternatives are possible, but we use the following penalization, as was used for α_i:

$$\beta_i = \beta_0(B - i + 1)^n, \quad \beta_0 \in \mathbb{R}^+ \quad \text{and} \quad 1 \le n \le 5. \qquad (20)$$

B. Sensitivity Analysis

The sensitivity of the objective function (19a) of OS-B' with respect to the auxiliary variable $p_{i,j}$ becomes

$$\begin{aligned}\frac{df}{dp_{i,j}} = \frac{df}{dq_{i,j}}\frac{dq_{i,j}}{dp_{i,j}} &= \left\{\alpha_i\left(q_{i,j}^{\max} + q_{i,j}^{\min} - 2q_{i,j}\right)\right.\\ &\left. + 2\beta_i\left(q_{i,j} - q_{i,j}^{\text{prev}}\right)\right\}\frac{t\left(q_{i,j}^{\max} - q_{i,j}^{\min}\right)\exp(-t\cdot p_{i,j})}{\{1 + \exp(-t\cdot p_{i,j})\}^2}. \qquad (21)\end{aligned}$$

The sensitivity of the constraint (19b) can be obtained by calculating the sensitivity of the position and the orientation of the end-effector frame, $dg_B/dp_{i,j}$. Using (7), one can find $dg_B/dp_{i,j}$ as

$$\begin{aligned}\frac{dg_B}{dp_{i,j}} &= \frac{d}{dp_{i,j}}\prod_{k=1}^{B} g_{k-1,k}\\ &= \prod_{k=1}^{i-1} g_{k-1,k}\frac{dg_{i-1,i}}{dp_{i,j}}\prod_{k=i+1}^{B} g_{k-1,k}\\ &= \prod_{k=1}^{i-1} g_{k-1,k}\frac{dg_{i-1,i}}{dq_{i,j}}\\ &\quad\times\prod_{k=i+1}^{B} g_{k-1,k}\frac{t\left(q_{i,j}^{\max} - q_{i,j}^{\min}\right)\exp(-t\cdot p_{i,j})}{\{1 + \exp(-t\cdot p_{i,j})\}^2}\end{aligned} \qquad (22)$$

$$\frac{dg_{i-1,i}}{dq_{i,j}} = \begin{pmatrix} \frac{d\mathbf{R}_i}{dq_{i,j}} & \frac{d\mathbf{b}_i^{\mathrm{T}}}{dq_{i,j}} \\ \mathbf{0} & 0 \end{pmatrix} \qquad (23)$$

where $d\mathbf{R}_i/dq_{i,j}$ and $d\mathbf{b}_i^{\mathrm{T}}/dq_{i,j}$ can be calculated explicitly by using the formulas for $\mathbf{R}_i$ and $\mathbf{b}_i$ given in Section II.

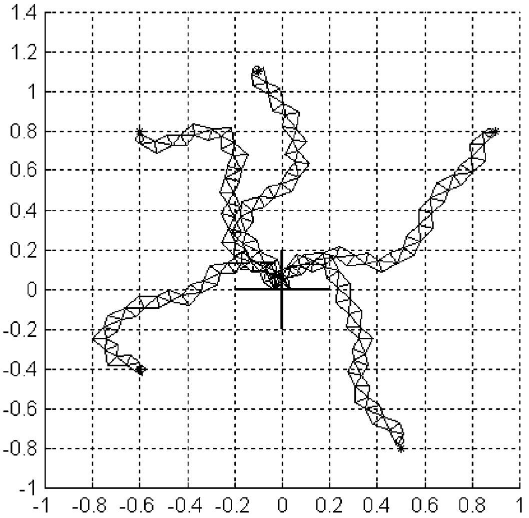


Fig. 7. Inverse kinematics results for the 2-D binary manipulator (o: end-effector; $*$: target).

IV. Applications and Verification

A. Case Study 1: 2-D Binary Manipulator

The inverse kinematics problem of a 2-D binary manipulator consisting of 20 modules was solved by the OS-B strategy proposed in the previous section. The 2-D binary manipulator was first introduced by Chirikjian and colleagues [7], [8], [11]. In this problem, we used $q_{i,j}^{\min} = 0.06, q_{i,j}^{\max} = 0.08, w = 0.06$, and $n = 5$ (for analysis convenience, the values such as $q_{i,j}^{\min}$, $q_{i,j}^{\max}$, and w will be treated as dimensionless quantities.) The tolerance ε in (16b) was set as $\varepsilon = 0.01$. Other parameters were set as $t = 0.25$ for (17c), $p_{i,j}^{\max} = 100$, and $p_{i,j}^{\min} = -100$. With these values determined after some numerical tests, fast solution convergence with design variables sufficiently close to binary values was achieved. In this paper, these values are used for all numerical problems considered. The following five targets of $g_p = (\mathbf{b}_p, \mathbf{R}_p)$ were considered:

$$\begin{pmatrix} b_{p,x} \\ b_{p,y} \\ \theta_p \end{pmatrix} = \begin{pmatrix} 0.9 \\ 0.8 \\ 0^\circ \end{pmatrix}, \begin{pmatrix} -0.1 \\ 1.1 \\ 30^\circ \end{pmatrix}, \begin{pmatrix} -0.6 \\ 0.8 \\ 90^\circ \end{pmatrix}, \begin{pmatrix} -0.6 \\ -0.4 \\ -80^\circ \end{pmatrix}, \begin{pmatrix} 0.5 \\ -0.8 \\ -90^\circ \end{pmatrix} \tag{24}$$

where the last component θ_p denotes the orientation angle used to define $\mathbf{R}_p$ in (6b).

In this investigation, a gradient-based nonlinear optimization algorithm, called the method of moving asymptotes (MMA) [21], is used for all examples. In MMA, the original nonconvex optimization problem is converted at every iteration to a convex subproblem, where the convexity is controlled by asymptote parameters. MMA is known to be very efficient for problems having many design variables. The detailed algorithm is given in [21].

Fig. 7 shows the five results obtained by OS-B. The initial value of $q_{i,j} = (q_{i,j}^{\max} + q_{i,j}^{\min})/2$ for the optimization was used. To check the accuracy and the CPU time by the proposed method, we solved the inverse kinematics for 2-D binary manipulators having from 4 up to 20 modules. The numerical results are summarized in Fig. 8. The CPU times and errors in Fig. 8 are the averages of 20 results obtained for different values of g_p, including those listed in (24). All the computations were done with an Intel Xeon, 2.6 GHz. Fig. 8 shows that the CPU time increases as the number of modules increases, but that the time does not grow exponentially.

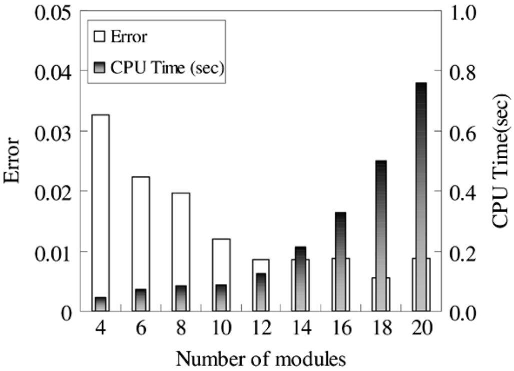


Fig. 8. CPU times and errors for Case Study 1 (2-D binary manipulators).

The solution convergence and accuracy can be affected by the module shapes of binary manipulators, equivalently, the workspace shape. The modified Ebert–Uphoff algorithm in [11] can yield inaccurate solutions for binary manipulators having crescent workspaces.[4] The effect of the workspace shape on solution accuracy is examined by considering two configurations: 20-module binary manipulator with $q_{i,j}^{\min} = 1.0, q_{i,j}^{\max} = 1.5$, and $w = 0.8$ for the case of a crescent workspace, and 20-module binary manipulator with $q_{i,j}^{\min} = 1.0, q_{i,j}^{\max} = 1.7$, and $w = 0.8$ for the case of a noncrescent workspace. While the algorithm in [11] produced 1.832 times larger errors in the crescent case than in the noncrescent case, the proposed method has almost the same accuracy for both crescent and noncrescent cases. (The actuation computation shows that the error for the crescent case was 0.902 times that for the noncrescent case. However, the computation time was only slightly increased.) From this test, the solution accuracy by the proposed optimization-based method is little influenced by the workspace shape.

As the next problem, the locus-tracing of the 2-D binary manipulator is considered. The manipulator consists of 10 modules having the same dimensions as the manipulator of the previous example. The end-effector is to be moved from ⓐ to ⓓ via the designated path ⓐ → ⓑ → ⓒ → ⓓ, as illustrated in Fig. 9. The locus-tracing patterns achieved by the strategies OS-B and OS-B' were plotted in Fig. 9(a) and (b), respectively. When OS-B' was employed, the previous statuses of the binary joints were used as the initial guess for the optimization strategy. In the result by OS-B', only a small number of joints in low modules were actuated due to the penalization parameter in (20). Table II shows the history of the joint configuration as the end-effector moves from ⓐ to ⓓ. Note that $q_{i,j}^{\min}$ and $q_{i,j}^{\max}$ correspond to the binary statuses 0 and 1, respectively. Table II also shows that the total number of actuated joints reduces when (20) is used for the inverse kinematics analysis.

[4] In the crescent workspace, the mean frame of all work points lies in a sparsely populated region of the workspace density function.

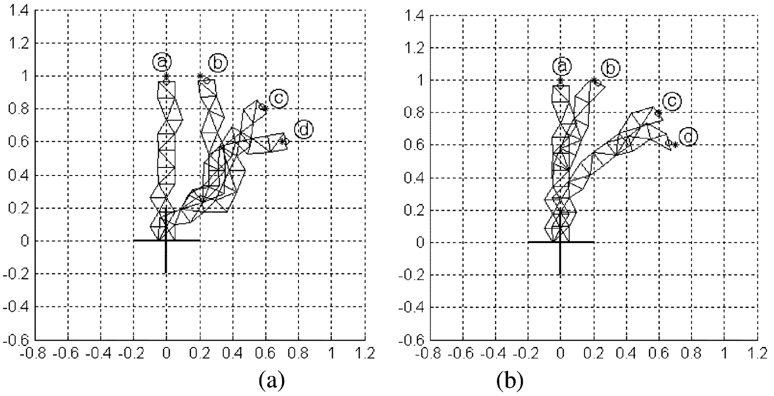


Fig. 9. Locus traced by employing (a) OS-B and (b) OS-B' ($\beta_0 = 0.05$).

TABLE II
JOINT CONFIGURATIONS FOR THE 2-D LOCUS-TRACING PROBLEM

Strategy	Target	Binary Values of Joint Displacements										Number of changed joints
		Base module ←									→ Top module	
OS-B	ⓐ	000	000	000	000	010	010	010	000	111	010	N/A
	ⓑ	110	100	111	001	011	011	110	110	001	010	14
	ⓒ	110	100	110	001	011	011	100	110	101	010	3
	ⓓ	110	100	011	001	110	110	100	100	011	010	9
OS- B'	ⓐ	000	000	000	000	010	010	010	000	111	010	N/A
	ⓑ	000	000	000	000	110	010	010	010	011	111	5
	ⓒ	010	000	000	100	110	010	010	010	111	111	3
	ⓓ	010	000	000	100	110	010	010	011	011	101	3

B. Case Study 2: 3-D Binary Manipulator

As Case Study 2, a 3-D binary manipulator with 20 modules is considered. 3-D binary manipulators were discussed at length by Dubowsky and colleagues [2]. Note that the inverse kinematics for 3-D binary manipulators is very difficult to deal with by the existing algorithms [7], [8], because of the excessive memory size. When the proposed method is employed, however, the required memory size depends only on the number of design variables, i.e., the number of binary joints, not on the dimensions of the given problem. Since both 2-D and 3-D binary modules consist of three binary joints, there is no difference between them as far as the memory size requirement is concerned. Obviously, the 3-D analysis is more complex than the 2-D analysis. The dimensions used for the 3-D manipulator were $q_{i,j}^{\min} = 0.06, q_{i,j}^{\max} = 0.08$, and $w = 0.06$. The distance error in (16b) was set as $\varepsilon = 0.01$. As before, we used $t = 0.25$ in (17c), $n = 5$ in (15), $p_{i,j}^{\max} = 100$, and $p_{i,j}^{\min} = -100$.

First, we considered the following target positions $g_p = (\mathbf{b}_p, \mathbf{R}_p)$ and solved for the joint statuses:

$$\begin{pmatrix} b_{p,x} \\ b_{p,y} \\ b_{p,z} \\ \alpha_p \\ \beta_p \\ \gamma_p \end{pmatrix} = \begin{pmatrix} 0 \\ 0.3 \\ 1.2 \\ 0^\circ \\ 55^\circ \\ 0^\circ \end{pmatrix}, \begin{pmatrix} -0.63 \\ 0.08 \\ 1.22 \\ 10^\circ \\ 20^\circ \\ 40^\circ \end{pmatrix}, \begin{pmatrix} 0.5 \\ -0.4 \\ 1.0 \\ 10^\circ \\ 60^\circ \\ 40^\circ \end{pmatrix}, \begin{pmatrix} -0.61 \\ -0.29 \\ 1.15 \\ 5^\circ \\ 10^\circ \\ 60^\circ \end{pmatrix}. \tag{25}$$

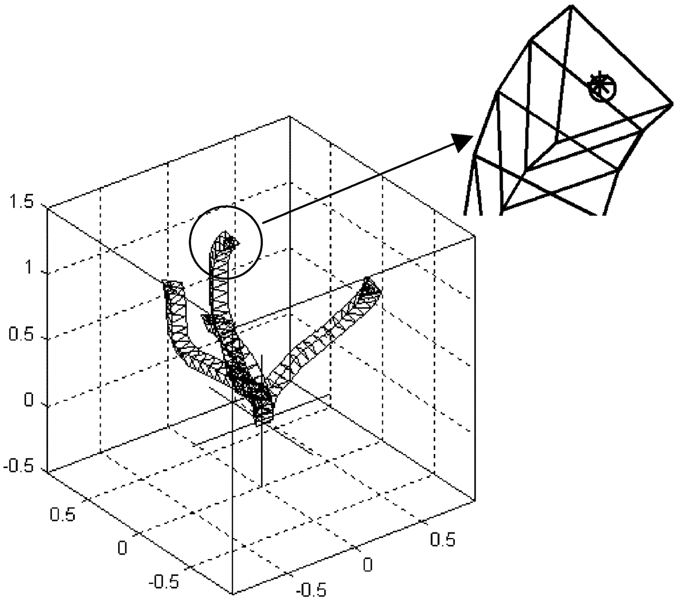


Fig. 10. Inverse kinematics results for the 3-D binary manipulator ($*$: target, o: end-effector).

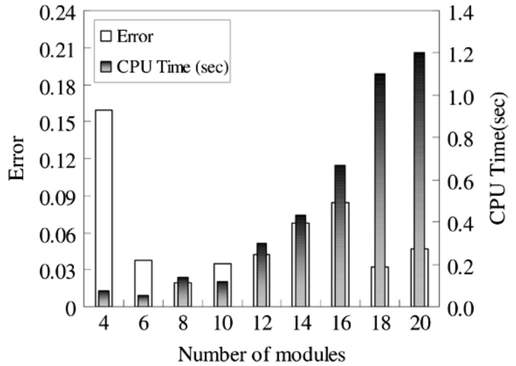


Fig. 11. CPU times and errors for Case Study 2 (3-D binary manipulators).

In (25), the symbols $\alpha_p, \beta_p, \gamma_p$ denote the rotation angles from which $\mathbf{R}_p$ is defined

$$\mathbf{R}_p = \mathbf{R}_z(\gamma_p)\mathbf{R}_y(\beta_p)\mathbf{R}_x(\alpha_p)$$

where $\mathbf{R}_x(\alpha_p)$, $\mathbf{R}_y(\beta_p)$, and $\mathbf{R}_z(\gamma_p)$ imply the rotation matrices about the x, y, and z axes, respectively.

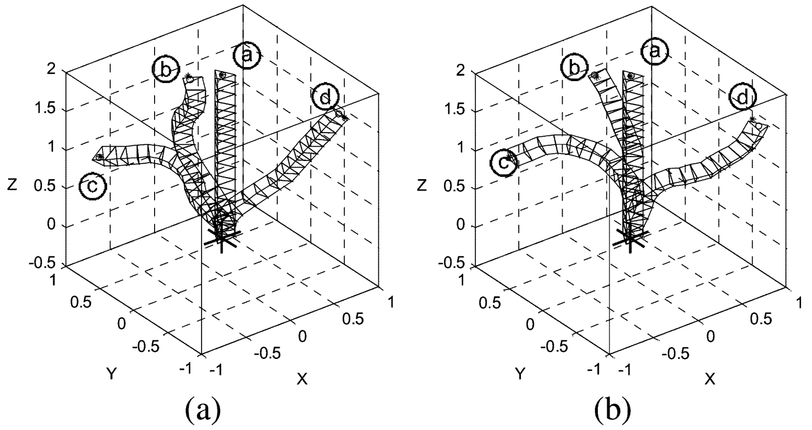


Fig. 12. Locus traced by employing (a) OS-B and (b) OS-B' ($\beta_0 = 0.05$).

TABLE III
JOINT CONFIGURATIONS FOR THE 3-D LOCUS-TRACING PROBLEM

Strategy	Target	State vector of joints																Number of changed joints
		Base module ←										→ Top module						
OS-B	ⓐ	111	111	111	111	111	111	111	000	000	000	000	000	111	111	111	000	N/A
	ⓑ	110	110	110	111	111	111	001	001	001	001	001	000	110	110	110	111	15
	ⓒ	010	010	000	000	001	101	110	110	010	010	010	010	000	001	001	111	31
	ⓓ	101	101	101	111	111	111	010	000	000	000	000	000	000	000	100	000	27
OS- B'	ⓐ	111	111	111	111	111	111	111	000	000	000	000	000	111	111	111	000	N/A
	ⓑ	111	111	111	111	111	111	110	110	000	000	000	000	000	110	000	000	10
	ⓒ	111	111	011	111	111	010	110	110	010	010	001	010	010	011	111	111	16
	ⓓ	101	111	011	101	101	101	101	100	000	110	000	000	010	010	010	111	15

Fig. 10 shows the results obtained by this approach. The targets were traced accurately, as may be clearly seen from Fig. 10. As in the previous 2-D example, the computation times to solve the inverse kinematics were examined and tabulated in Fig. 11 for 3-D manipulators having a different number of modules. As Fig. 11 indicates, the CPU time, even for the 3-D case, does not increase exponentially as the number of modules increases.

The locus-tracing for the 10-module 3-D binary manipulator was also considered. The objective was to trace sequentially the targets ⓐ, ⓑ, ⓒ, ⓓ that are marked in Fig. 12. The results by OS-B and OS-B' are plotted in Fig. 12(a) and (b), respectively, and Table III shows the binary statuses of all joints during the locus tracing. Again, the consideration of (20) reduces the number of the actuated joints (see Table III).

V. CONCLUSION

The new continuous-variable-based optimization method for the inverse kinematics of binary manipulators was presented. The key idea of the proposed method is to relax a binary-variable-based optimization problem to a continuous-variable-based problem. In the optimization, the distance between the end-effector and the target is constrained, and the objective is to push the continuous design variables toward the permissible binary values. The findings from this investigation are as follows.

1) 3-D inverse kinematics problems having a reasonable number of modules were solved almost in real time.
2) The use of the auxiliary variables mapped from the joint displacement variables by the nonlinear S-shaped function was very effective to push the design variables toward the binary values.
3) The locus tracing was successfully accomplished with the proposed optimization-based formulation. The number of the status-changed joints can be reduced if the corresponding condition is added as a penalty term in the objective function.
4) The system resource requirement and computation time to solve the inverse kinematics by the proposed algorithm do not directly depend on the workspace dimension.

APPENDIX

In [19], it is shown that members of a Lie group are exponentially mapped with members of a Lie algebra (see [22] for a background on Lie groups.) For example, for every $\mathbf{R} \in \mathrm{SO}(3)$ (a Lie group), there exists $[\boldsymbol{\omega}] \in \mathrm{so}(3)$ (a Lie algebra) such that

$$\mathbf{R} = \exp[\boldsymbol{\omega}] \tag{A1}$$

where $[\boldsymbol{\omega}] \in \mathrm{so}(3)$ is defined as

$$[\boldsymbol{\omega}] \triangleq \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}, \quad \boldsymbol{\omega} \in \mathbb{R}^3 \tag{A2}$$

and the exponential formula in (A1) is given as (see [19])

$$\exp[\boldsymbol{\omega}] = \mathbf{I} + \frac{\sin\|\boldsymbol{\omega}\|}{\|\boldsymbol{\omega}\|}[\boldsymbol{\omega}] + \frac{1-\cos\|\boldsymbol{\omega}\|}{\|\boldsymbol{\omega}\|^2}[\boldsymbol{\omega}]^2 \tag{A3}$$

where $\|\cdot\|$ denotes the Euclidean norm.

Therefore, the orientation part of the distance metric in (3) can be written as

$$\begin{aligned} \left\|\log \mathbf{R}_B^{\mathrm{T}}\mathbf{R}_p\right\| &= \left\|\log \mathbf{R}_B^{-1}\mathbf{R}_p\right\| = \|[\boldsymbol{\omega}_B] - [\boldsymbol{\omega}_p]\| \\ &= \|\boldsymbol{\omega}_B - \boldsymbol{\omega}_p\|. \end{aligned} \tag{A4}$$

Equation (A4) shows how the distance in $\mathrm{SO}(3)$ can be evaluated.

Acknowledgment

The authors appreciate discussions with Prof. F. C. Park at Seoul National University during this investigation.

References

[1] E. Paljug, T. Ohm, and S. Hayati, “The JPL serpentine robot: A 12 DOF system for inspection,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 1995, pp. 3143–3248.

[2] M. D. Lichter, V. A. Sujan, and S. Dubowsky, “Computational issues in the planning and kinematics of binary robots,” in *Proc. IEEE Int. Conf. Robot. Autom.*, Washington, DC, May 2002.

[3] ——, “Experimental demonstrations for a new paradigm in space robots,” in *Proc. 7th Int. Symp. Exp. Robot.*, Dec. 2000, pp. 225–234.

[4] V. A. Sujan, M. D. Lichter, and S. Dubowsky, “Lightweight hyper-redundant binary elements for planetary exploration robots,” in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron.*, 2001, pp. 1273–1278.

[5] M. Hafez, M. D. Lichter, and S. Dubowsky, “Optimized binary modular reconfigurable robotic devices,” *IEEE/ASME Trans. Mechatron.*, vol. 8, no. 1, pp. 18–25, Jan. 2003.

[6] I. Ebert–Uphoff and G. S. Chirikjian, “Efficient workspace generation for binary manipulators with many actuators,” *J. Robot. Syst.*, vol. 12, no. 6, pp. 383–400, 1995.

[7] ——, “Inverse kinematics of discretely actuated hyper-redundant manipulators using workspace densities,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 1996, pp. 139–145.

[8] Y. F. Wang and G. S. Chirikjian, “The comparison of two workspace-density-driven inverse kiematics methods for hyper-redundant manipulators,” in *Proc. ASME Des. Eng. Tech. Conf.*, vol. 5A, Montreal, QC, Canada, Sep. 2002, pp. 563–568.

[9] G. S. Chirikjian and I. Ebert–Uphoff, “Numerical convolution on the Euclidean group with applications to workspace generation,” *IEEE Trans. Robot. Autom.*, vol. 14, no. 1, pp. 123–136, Feb. 1998.

[10] Y. F. Wang and G. S. Chirikjian, “Workspace generation of hyper-redundant manipulators as a diffusion process on SE(N),” *IEEE Trans. Robot. Autom.*, vol. 20, no. 3, pp. 399–408, Jun. 2004.

[11] J. Suthakorn and G. S. Chirikjian, “A new inverse kinematics algorithm for binary manipulators with many actuators,” *Adv. Robot.*, vol. 15, no. 2, pp. 225–244, 2001.

[12] G. S. Chirikjian and J. W. Burdick, “Kinematically optimal hyper-redundant manipulator configurations,” *IEEE Trans. Robot. Autom.*, vol. 11, no. 6, pp. 794–806, Dec. 1995.

[13] K. R. Zanganeh and J. Angeles, “The inverse kinematics of hyper-redundant manipulators using splines,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 1995, pp. 2797–2802.

[14] Y. Y. Kim and G. H. Yoon, “Multi-resolution multi-scale topology optimization—A new paradigm,” *Int. J. Solids Struct.*, vol. 37, pp. 5529–5559, 2000.

[15] G. H. Yoon and Y. Y. Kim, “The role of S-shaped mapping functions in the SIMP approach for topology optimization,” *KSME Int. J.*, vol. 15, pp. 1496–1506, 2003.

[16] G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization*. New York: Wiley, 1988.

[17] O. K. Gupta and A. Ravindran, “Branch and bound experiments in convex nonlinear integer programming,” *Manage. Sci.*, vol. 31, pp. 1533–1546, 1985.

[18] K. M. Bretthauer and B. Shetty, “The nonlinear resource allocation problem,” *Oper. Res.*, vol. 43, pp. 670–683, 1995.

[19] F. C. Park, “Distance metrics on the rigid-body motions with applications to mechanism design,” *Trans. ASME*, vol. 117, pp. 48–54, 1995.

[20] D. S. Lees and G. S. Chirikjian, “An efficient method for computing the forward kinematics of binary manipulators,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 1996, pp. 1012–1017.

[21] K. Svanberg, “The method of moving asymptotes—A new method for structural optimization,” *Int. J. Numer. Meth. Eng.*, vol. 24, pp. 359–373, 1987.

[22] W. Boothby, *An Introduction to Differentiable Manifolds and Riemannian Geometry*. New York: Academic, 1975.

Yoon Young Kim received the B.S. and M.S. degrees from Seoul National University, Seoul, Korea, and the Ph.D. degree from Stanford University, Stanford, CA, in 1989.

He has been on the faculty of the School of Mechanical and Aerospace Engineering, Seoul National University, since 1991. He is also the Director of the National Creative Research Initiatives Center for Multiscale Design. His main research field is the optimal design of multiphysics systems, mechanisms, and transducers. He has served as Editor of several Korean and international journals, and as an organizing committee member of several international conferences.

Gang-Won Jang received the M.S. degree in 2000, and the Ph.D. degree in 2004, both from the School of Mechanical and Aerospace Engineering, Seoul National University, Seoul, Korea.

He is currently an Assistant Professor with the School of Mechanical Engineering, Kunsan National University, Chonbuk, Korea. His current interest concerns optimization for binary manipulators and mechanism designs.

Sang-Jun Nam received the B.S. degree in 2003 and the M.S. degree in 2005, both from Seoul National University, Seoul, Korea, where he is currently working toward the Ph.D. degree in the School of Mechanical and Aerospace Engineering.

His current research interest concerns optimization for binary manipulators and mechanism designs.