Mechanical Systems and Signal Processing 156 (2021) 107461

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp

Genetic-algorithm-based global design optimization of tree-type robotic systems involving exponential coordinates

Julien Amar [a,*], Kenji Nagase [a]

[a] Faculty of Systems Engineering, Wakayama University, 640-8510 Wakayama, Japan

ARTICLE INFO

Article history:
Received 30 July 2020
Received in revised form 9 November 2020
Accepted 14 November 2020
Available online 16 February 2021

Keywords:
Design optimization
Topology and geometry
Exponential coordinates
Tree-type system
Genetic algorithms

ABSTRACT

This paper proposes the simultaneous design optimization of the geometry (joint position and directions) and the topology (joint distribution and connection) of tree-type robotic systems based on an exponential coordinate system expression. Tree-type systems represent a versatile system expression of mechanical systems comprising multiple serial link chains branching from the root. Previously, we derived general closed-form formulas of the kinematics and dynamics of tree-type systems in exponential coordinates by introducing a connectivity matrix called the chain matrix. In this study, these results are first extended to floating base and closed-chain systems to enlarge the system framework. Next, the efficient coding of the system parameters by using a genetic algorithm (GA) is demonstrated. The closed-form formulas of the kinematics and dynamics allow for cost evaluations through numerical simulations with feedback control. Design examples of a robotic platform and a grasping/manipulation system illustrate the proposed optimization process.

1. Introduction

1.1. Background and motivation

The analysis, design, and control of robotic systems have attracted increasing attention in several domains [1–5]. Thus far, several robots have been designed for industrial and domestic applications [6–10]; such robots can perform various tasks efficiently and are therefore finding increasing use in daily life. In general robot designs, the system architecture, such as the joint distribution and connection (topology) as well as their position and movement directions (geometry), are first determined by considering intuitive or biologically inspired approaches. Then, for the choice of the architecture, physical parameters such as the mass/inertia and link length are designed or optimized, as reported in [11–13]. These designs are simple, and several satisfactory solutions can be obtained; however, bio-inspired designs might be inefficient or intractable if more specific or complex design specifications [12,14,15] are required. Therefore, a scheme involving systematic simultaneous optimization of the geometrical and topological properties is required [6,16].

To perform such an optimization, the system kinematics and dynamics must be determined to evaluate the costs, and the coordinates must be selected carefully to obtain an appropriate system parametrization. For robotic systems, the most widely used coordinates include the Denavit–Hartenberg (DH) parameters [1,17–19]. The DH parameters are the standard

* Corresponding author.
E-mail address: s159001@wakayama-u.ac.jp (J. Amar).

https://doi.org/10.1016/j.ymssp.2020.107461
0888-3270/

for industrial robots; they mostly have at most six degrees of freedom (DOF). However, these parameters are defined sequentially from the root joint to the tip and are not intuitive from a geometrical viewpoint, and it is generally difficult to use them in the optimization process according to the relevant dependencies. Some researchers optimized the kinematic path and other cost functions based on the DH parameters [20–23]. Alternative coordinates include quaternions, which are widely used in computer-assisted 3D representations and spacecraft attitude control [24,25]. The use of quaternions can suppress singularities in the attitude via the introduction of a redundant quadruple parameter expression. However, the resultant mathematics becomes somewhat complex, and therefore, the application of quaternions to robot manipulators remains rather limited. A quaternion-based optimization scheme for simple serial link robots was proposed in [26]. Exponential coordinates [2,27,28] are an alternative to these parameters, and they are especially useful for describing systems comprising serial link chains.

Exponential parameters include four parameters that define the joint properties: initial position, translational movement axis, rotational movement axis, and displacement along the axis. All these parameters are defined with respect to a single base frame, and thus, it is easy to change their properties without affecting other joint parameters. The position/orientation and velocity of a frame can be described by the exponential of the exponential parameters. The exponential function has elegant rules regarding the product and derivative, which is beneficial for deriving closed-form formulas of the kinematics and dynamics for serial link chains. One such optimization based on the product of exponentials was described in [29], in which only the kinematics of some specific structures was considered. In [30], tree-type systems [31] comprising multiple serial link chains branching from the root were considered. Inspired by the tensegrity principle of structural networks [32–35], we introduced a (0,1) element matrix called the chain matrix and derived closed-form formulas of the kinematics and dynamics of tree-type systems on exponential coordinates. Based on this formulation, the system geometry (joint position and directions) was determined from the exponential parameters, and the system topology (joint connection and distribution) was determined from the chain matrix. This parametrization is quite useful for the optimization, as a simple coding rule is formulated owing to the independence of the exponential parameters and the (0,1)-valued elements of the chain matrix. Physical parameters such as the mass and link length can be determined from the joint position and thus from the exponential parameters.

1.2. Relevance and contribution

Based on the results reported in [30], we propose a simultaneous (or global) geometry and topology optimization scheme for tree-type robotic systems based on exponential coordinates with the chain matrix expression. To enlarge the system framework, the results for tree-type systems are first extended to floating base systems (for humanoids, mobile robots, etc. [36,37]) and closed-chain systems (for parallel link manipulators, robotic platforms, etc. [38,39]). Global optimization problems are large scale and highly nonlinear and involve mixed continuous and discrete variables. In this study, we employ a genetic algorithm (GA) [40–42] because gradient-based approaches [43–46] may be inefficient for such problems. When coding the GA, the exponential parameters can be efficiently coded using binary strings with rate decoding in a unified manner. For chain matrix coding, the possible (0,1) patterns for tree-type systems are investigated in detail to exclude any unnecessary search spaces, and a minimal set of binary strings is defined. The closed-form kinematics and dynamics for tree-type systems can be obtained from the strings, and a controller of any type can be constructed from them. This allows performing numerical simulations to evaluate various costs (e.g., workspace, work, etc. [45–47]) in the optimization process, for which analytical formulas are not available. A cost evaluation process that uses the kinematics and dynamics with feedback control is also illustrated. Two numerical design examples of a robotic platform and a robotic hand are also presented to illustrate the proposed optimization process.

1.3. Organization

The rest of this paper is organized as follows. In Section 2, we briefly review the results of the kinematics and dynamics of tree-type systems. In Section 3, we describe their extension to floating base and closed-chain systems. Sections 4 and 5 respectively describe the GA coding of the exponential parameters and the chain matrix. Section 6 summarizes the complete GA procedure, and Section 7 describes the cost evaluation through numerical simulations with feedback control. Sections 8 and 9 present numerical examples. Finally, Section 10 presents the conclusions of this study.

In terms of notations, $\mathbb{R}$ represents the set of real numbers, and $\mathbb{N}$ and $\mathbb{B}$ respectively denote the sets of natural numbers and $\{0,1\}$. The superscript $(\cdot)^+$ represents the pseudo-inverse of the specified matrix.

2. Kinematics and dynamics of tree-type systems on exponential coordinates

Tree-type systems are a good general representation of robotic systems, and they represent robotic architectures comprising kinematic chains of rigid joints/links branching from the root. In our previous study [30], we derived closed-form formulas of the kinematics and dynamics of tree-type systems by using exponential coordinates by introducing the chain matrix. In this section, we briefly review the necessary terminologies and results obtained in [30].

2.1. Exponential parameters

A classic approach to represent a serial link chain is to use parameters that fully describe the joint properties, as the links only act as bridges between these joints. Exponential parameters can be used to realize such an approach. One of the main benefits of using exponential parameters is that they are all defined with respect to a fixed base frame, which allows the modification of the properties of a joint without affecting other joints. Another benefit is the elegant rules of the exponential function pertaining to the products and derivatives, which enables the formulation of the closed-form expression of the kinematics and dynamics of serial link manipulators. Four exponential parameters exist for every joint i ($i = 1, \cdots, n$) in a system:

- The position parameter $q_i \in \mathbb{R}^3$ is the 3D vector representing the coordinates of the initial position of joint i with respect to the base frame Σ_s.
- The translation parameter $v_i \in \mathbb{R}^3$ is the 3D vector representing the translation axis of joint i with respect to Σ_s. For a translational or prismatic joint, we set $\|v_i\| = 1$, whereas for a rotational joint, we set $v_i = 0$.
- The rotation parameter $\omega_i \in \mathbb{R}^3$ is the 3D vector representing the rotation axis of joint i with respect to Σ_s. For a rotational joint, we set $\|\omega_i\| = 1$, whereas for a prismatic joint, we set $\omega_i = 0$.
- The movement parameter $\theta_i \in \mathbb{R}$ represents the incremental movement of joint i along the direction v_i or ω_i. This parameter corresponds to the angle and translation for rotational and translational joints, respectively. θ_i is called the joint angle for clarity; however, θ_i can represent a displacement when the joint is translational.

We can locate several joints at the same location to describe a multi-DOF joint by setting $q_i = q_{i+1} = q_{i+2} = \cdots$.

From the exponential parameters, the twist $\widehat{\xi}_i$ (in matrix form) associated with the i_{th} joint is defined as follows:

$$\widehat{\xi}_i = \begin{cases} \begin{bmatrix} \widehat{\omega}_i & -\omega_i \times q_i \\ 0 & 0 \end{bmatrix} & \text{if the joint is pure rotation} \\ \begin{bmatrix} 0 & v_i \\ 0 & 0 \end{bmatrix} & \text{if the joint is pure translation,} \end{cases} \tag{1}$$

where $\widehat{\omega}_i$ is the skew-symmetric matrix equivalent to the vector product. The rotation matrix of the i_{th} link relative to the base frame Σ_s is described by the exponential of $\widehat{\omega}_i$ as $R_{si} = e^{\widehat{\omega}_i \theta_i}$. By using $e^{\widehat{\omega}_i \theta_i}$, the exponential of the twist can be described as

$$e^{\widehat{\xi}_i \theta_i} = \begin{cases} \begin{bmatrix} e^{\widehat{\omega}_i \theta_i} & \left(I - e^{\widehat{\omega}_i \theta_i}\right)(\omega_i \times v_i) + \omega_i \omega_i^{\mathrm{T}} v_i \theta_i \\ 0 & 1 \end{bmatrix} & (\omega \neq 0) \\ \begin{bmatrix} I & v_i \theta_i \\ 0 & 1 \end{bmatrix} & (\omega = 0). \end{cases} \tag{2}$$

2.2. Kinematics of rigid body

The motion of a rigid body is characterized by a frame attached to it. The configuration $g_{st} \in \mathbb{R}^{4\times4}$ of a frame Σ_t relative to a reference frame Σ_s is represented by the relative translation $p_{st} \in \mathbb{R}^3$ and the relative rotation $R_{st} \in \mathbb{R}^{3\times3}$ (orthogonal matrix) as

$$g_{st} = \begin{bmatrix} R_{st} & p_{st} \\ 0 & 1 \end{bmatrix}. \tag{3}$$

From (2), the exponential of a twist represents a configuration. For serial rink robots, the configuration of the tool frame Σ_t relative to the base frame Σ_s becomes a function of $\theta = [\theta_1, \theta_2, \cdots, \theta_n]^{\mathrm{T}} \in \mathbb{R}^n$. The rigid body motion $g_{st}(\theta)$, or the forward kinematics, is related to the reference configuration $g_{st}(0)$ in exponential parameters as

$$g_{st}(\theta) = e^{\widehat{\xi}_1 \theta_1} \ldots e^{\widehat{\xi}_n \theta_n} g_{st}(0). \tag{4}$$

To calculate the rigid body velocity of a frame from its configuration g, the adjoint transformation $\mathrm{Ad}_g \in \mathbb{R}^{6\times6}$ and its inverse are defined as follows:

$$\mathrm{Ad}_g = \begin{bmatrix} R & \hat{p}R \\ 0 & R \end{bmatrix}, \quad \mathrm{Ad}_g^{-1} = \begin{bmatrix} R^{\mathrm{T}} & -R^{\mathrm{T}}\hat{p} \\ 0 & R^{\mathrm{T}} \end{bmatrix} \tag{5}$$

($\widehat{p}$ is the skew-symmetric matrix equivalent to the vector product). We define two types of velocities of the frame Σ_t from the configuration g_{st} as follows. The spatial velocity $V_{st}^s = \left[\left(v_{st}^s\right)^{\mathrm{T}} \left(\omega_{st}^s\right)^{\mathrm{T}}\right]^{\mathrm{T}}$, where v_{st}^s and ω_{st}^s respectively denote the trans-

lational and rotational components, is the velocity of Σ_t as observed from Σ_s, and it is defined by the (1,1) and (1,2) blocks of $\widehat{V}^s_{st} = \dot{g}_{st} g_{st}^{-1}$. When g_{st} is obtained using (4), V^s_{st} can be expressed as

$$V^s_{st} = J^s_{st}\dot{\theta}, \tag{6}$$

with

$$J^s_{st} = \begin{bmatrix} \xi_1 & \xi'_2 & \dots & \xi'_n \end{bmatrix}, \text{ where } \xi'_\ell = \mathrm{Ad}_{\left(e^{\hat{\xi}_1\theta_1}\dots e^{\hat{\xi}_{\ell-1}\theta_{\ell-1}}\right)}\xi_\ell. \tag{7}$$

The body velocity $V^b_{st} = \left[\left(v^b_{st}\right)^{\mathrm{T}} \ \left(\omega^b_{st}\right)^{\mathrm{T}}\right]^{\mathrm{T}}$ represents the velocity of Σ_t as observed from Σ_t, and it is defined by the (1,1) and (1,2) blocks of $\widehat{V}^b_{st} = g_{st}^{-1}\dot{g}_{st}$. When g_{st} is obtained using (4), V^b_{st} can be expressed as

$$V^b_{st} = J^b_{st}\dot{\theta}, \tag{8}$$

with

$$J^b_{st} = \begin{bmatrix} \xi^\dagger_1 & \xi^\dagger_2 & \dots & \xi^\dagger_n \end{bmatrix}, \text{ where } \xi^\dagger_\ell = \mathrm{Ad}^{-1}_{\left(e^{\hat{\xi}_\ell\theta_\ell}\dots e^{\hat{\xi}_n\theta_n} g_{st}(0)\right)}\xi_\ell. \tag{9}$$

The spatial and body velocities are related to each other by the adjoint transformation as

$$V^s_{st} = \mathrm{Ad}_{g_{st}} V^b_{st}, \quad \text{or} \quad V^b_{st} = \mathrm{Ad}^{-1}_{g_{st}} V^s_{st}. \tag{10}$$

When the frames interchange their role, the following relation holds:

$$V^b_{ab} = -V^s_{ba}, \quad V^b_{ab} = -\mathrm{Ad}_{g_{ba}} V^b_{ba}. \tag{11}$$

It is also feasible to describe the spatial or body velocity from Σ_a to Σ_c by using an intermediate frame Σ_b:

$$V^s_{ac} = V^s_{ab} + \mathrm{Ad}_{g_{ab}} V^s_{bc} \tag{12}$$

$$V^b_{ac} = \mathrm{Ad}^{-1}_{g_{bc}} V^b_{ab} + V^b_{bc}. \tag{13}$$

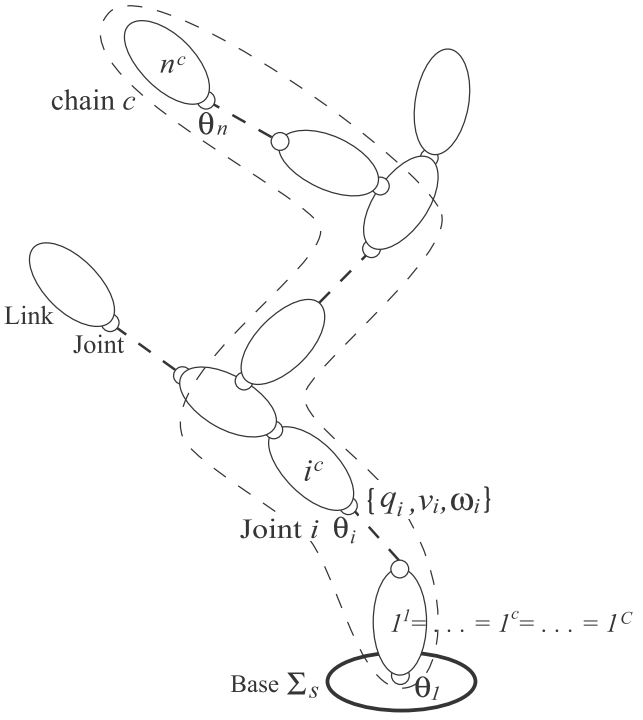


Fig. 1. Tree-type robotic system.

2.3. Tree-type systems and chain matrix

A tree-type system comprises a set of serial link chains branching from the root. We assume that each link is driven by a joint (one-to-one relation), and no closed path exists in the tree. The tree-type system is schematically illustrated in Fig. 1 and can be defined as follows:

- The base frame Σ_s is fixed, and the tree-type system starts from the root joint fixed in Σ_s.
- The chain is a set of serial links and joints starting from the root to one of the extremities of the system. The chains within the system are denoted by $c = 1, \cdots, C$.
- The joint angle is denoted by θ_i, where $i = 1, \cdots, n$ is the joint number in the full system. The joint number increases from the root to the tip and from the first chain to the last.
- The rigid links are denoted by i^c, where i is the joint number and c is the chain to which the joint belongs. A link can be a member of different chains. For the first link, $1^1 = 1^2 = \cdots = 1^C$.

To describe the system connection, we introduce the following $(0, 1)$-element matrix, that is, the chain matrix:

$$C_h = \begin{bmatrix} C_h(1,1) & \cdots & C_h(1,n) \\ \vdots & C_h(c,i) & \vdots \\ C_h(C,1) & \cdots & C_h(C,n) \end{bmatrix} \in \mathbb{R}^{C\times n}, \tag{14}$$

where the element $C_h(c, i)$ is set as 1 if a joint i exists in a chain c; otherwise, $C_h(c, i) = 0$. The ones in the c-th row represent the joints in chain c. Because $e^{0\cdot\widehat{\xi}_i\theta_i} = I_4$, we can describe the configuration of a link i^c in a unified manner by using C_h as, for example,

$$g_{si^c}(\theta) = \prod_{j=1}^{i}\left(e^{C_h(c,j)\widehat{\xi}_j\theta_j}\right) g_{si^c}(0). \tag{15}$$

The spatial and body velocities of the link, $V^s_{si^c}$ and $V^b_{si^c}$, can be calculated by (6) and (8), respectively, by using the configurations obtained from a calculation similar to (15).

The ones in the same column in C_h represent the multiplicity of the links in the total chains. To remove this multiplicity, we additionally define the simplified chain matrix $\widehat{C}_h \in \mathbb{R}^{C\times n}$ by replacing the ones in a column, except for the topmost to zero. The simplified chain matrix can be used to calculate the total kinetic and potential energies; for example, the kinetic energy can be defined by

$$T = \frac{1}{2}\sum_{c=1}^{C}\sum_{i=1}^{n}\widehat{C}_h(c,i)\left(J^b_{si^c}\dot{\theta}\right)^{\mathrm{T}}\mathcal{M}_{i^c}J^b_{si^c}\dot{\theta}, \tag{16}$$

where $\mathcal{M}_{i^c}$ is the generalized inertia matrix of the link i^c and $V^b_{si^c} = J^b_{si^c}\dot{\theta}$ is its body velocity. Note that it is unnecessary to calculate the terms for $\widehat{C}(c, i) = 0$.

2.4. Dynamics of tree-type systems

By using the exponential parameters and the chain matrix, we can systematically derive the closed-form kinematics and dynamics of tree-type systems. In [30], as a typical example of the dynamics of tree-type systems, arm-hand systems holding an object with the fingertips without slipping were considered. Their dynamics can be described by the following set of equations:

$$M_f\ddot{\theta} + C_f\dot{\theta} + N_f = \tau - J_h^{\mathrm{T}}F_f \quad (\text{arm} - \text{hand dynamics}) \tag{17}$$

$$M_o\dot{V}^b_{so} + C_oV^b_{so} + N_o = GF_f \quad (\text{object dynamics}) \tag{18}$$

$$J_h\dot{\theta} = G^{\mathrm{T}}V^b_{so} \quad (\text{constraint}) \tag{19}$$

where $\theta \in \mathbb{R}^n$ and $V^b_{so} \in \mathbb{R}^6$ are respectively the joint angles and the body velocity of the object, and $F_f \in \mathbb{R}^k$ is the fingertip force (constraint force), where k is the number of the constraint. $J_h \in \mathbb{R}^{k\times n}$ and $G \in \mathbb{R}^{6\times k}$ are respectively the manipulator Jacobian and the grasp map. To avoid slipping, the fingertip force F_f should be controlled to lie in the friction cone $\mathcal{FC}$ [2,48], i.e.,

$$F_f \in \mathcal{FC}. \tag{20}$$

The system dynamics are combined according to the control purposes. For grasping/manipulation control, the object motion and internal force are usually chosen as the control variables as follows. For the purpose of manipulation, we define $x_o = [p_{so}^{\mathrm{T}}, r_{so}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^6$, where p_{so} and r_{so} respectively denote the mass center and the rotation vector of the object. The object velocity $\dot{x}_o$ is related to the object body velocity by using a transformation matrix as [2,49,30]

$$V_{so}^b = T_o \dot{x}_o. \tag{21}$$

The system sometimes exhibits redundancy, and we can choose the redundant motion $x_r \in \mathbb{R}^{n-k}$ in addition to x_o in such cases. The redundant motion can be chosen from a combination of the joint angle and the object motion as

$$\dot{x}_r = T_h \dot{\theta} - T_g \dot{x}_o. \tag{22}$$

One typical choice of the redundant motion is the internal velocity $\dot{x}_n$ in which we choose $T_h = K_h^{\mathrm{T}}$ and $T_g = 0$, where K_h is the null-space matrix of J_h (i.e., $J_h K_h = 0$) [2]. By combining (22) and (19) with (21), V_{so}^b and $\dot{\theta}$ are respectively related to the augmented object motion $\dot{\bar{x}}_o = [\dot{x}_o^{\mathrm{T}} \;\; \dot{x}_r^{\mathrm{T}}]^{\mathrm{T}}$ as

$$V_{so}^b = \underbrace{[T_o 0]}_{\widehat{T}_o} \dot{\bar{x}}_o, \tag{23}$$

and

$$\underbrace{\begin{bmatrix} J_h \\ T_h \end{bmatrix}}_{\bar{J}_h} \dot{\theta} = \underbrace{\begin{bmatrix} G^{\mathrm{T}} T_o & 0 \\ T_g & I \end{bmatrix}}_{\bar{T}_o} \dot{\bar{x}}_o, \tag{24}$$

where $\bar{J}_h \in \mathbb{R}^{n \times n}$ is the augmented manipulator Jacobian and is assumed to be nonsingular for control purposes (see Section 7).

For the purpose of grasping, we note that the right-hand term in (18) corresponds to the manipulation force F_o, that is, $F_o = GF_f$. A decomposition is obtained by solving this equation for F_f as

$$F_f = G^+(GF_f) + K_G f_N \tag{25}$$

$$= G^+\left(M_o \dot{V}_{so}^b + C_o V_{so}^b + N_o\right) + K_G f_N, \tag{26}$$

where $G^+ \in \mathbb{R}^{k \times 6}$ and $K_G \in \mathbb{R}^{k \times (n-6)}$ respectively denote the pseudo-inverse and null space matrix of G, and f_N is the magnitude of the internal force [48,50].

Upon substituting (24) and (23) and their derivatives as well as (26) into (17), the motion equation in terms of the control variables $\bar{x}_o$ and f_N can be formulated as

$$\bar{M}_o \ddot{\bar{x}}_o + \bar{C}_o \dot{\bar{x}}_o + \bar{N}_o + J_h^{\mathrm{T}} K_G f_N = \tau, \tag{27}$$

where

$$\bar{M}_o = M_f \bar{J}_h^{-1} \bar{T}_o + J_h^{\mathrm{T}} G^+ M_o \widehat{T}_o \tag{28}$$

$$\bar{C}_o = M_f \bar{J}_h^{-1}\left(\dot{\bar{T}}_o - \dot{\bar{J}}_h \bar{J}_h^{-1} \bar{T}_o\right) + C_f \bar{J}_h^{-1} \bar{T}_o + J_h^{\mathrm{T}} G^+\left(M_o \dot{\widehat{T}}_o + C_o \widehat{T}_o\right) \tag{29}$$

$$\bar{N}_o = N_f + J_h^{\mathrm{T}} G^+ N_o. \tag{30}$$

This dynamic equation is used for the cost evaluation described in Section 7.2.

3. Extension to floating base and closed-chain systems

When considering tree-type systems, it was assumed that the root joint is fixed in the environment and that no closed path exists in the tree. However, this is not a challenging limitation as we can treat the systems resulting from these assumptions in our framework with a slight modification. These systems are known as floating base systems and closed-chain systems. This section describes these systems and the necessary modification procedures.

3.1. Floating base systems

Humanoids and mobile robots/vehicles are examples of floating base systems. Such structures can be described by adding artificial joints to ensure that the root joint has six DOFs (three translation and three rotation).

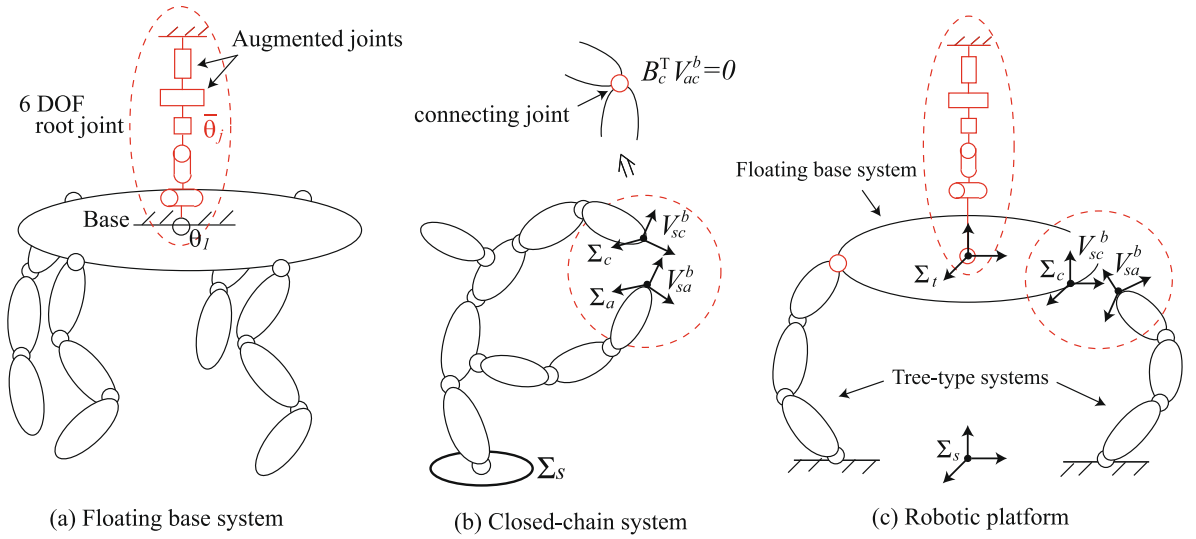


Fig. 2. Extension to general tree-type structures.

To illustrate this process, we consider the legged robot system shown in Fig. 2(a). The original tree-type system is shown in black; it comprises four chains with a revolute root joint fixed at the roof. To ensure that the base moves freely, we add 5-DOF joints (red joints) to the root joint. This modification can be performed by adding five columns of ones to the chain matrix from the left

$$\overline{C}_h = \left[\begin{array}{ccccc|ccccccc} 1 & 1 & 1 & 1 & 1 & 1 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & & & & & & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & 1 & & & C_h & & & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & & & & & & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & \cdots & \cdots & \cdots & \cdots & \cdots & 1 \end{array}\right], \tag{31}$$

and by defining the exponential parameters of the augmented joints $\overline{\theta}_j$ $(j = 1, \cdots, 5)$ according to the root joint property. If the root joint is rotational about the z-axis ($\omega_1 = [0, 0, 1]^{\mathrm{T}}$) located at q_1, the exponential parameters are

$$\bar{q}_1 = \cdots = \bar{q}_5 = q_1, \quad \text{and,} \quad \bar{v}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \bar{v}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \bar{v}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad \bar{\omega}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \bar{\omega}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}. \tag{32}$$

The mass $\overline{m}_{j^c}$ and inertia $\overline{I}_{j^c}$ of the links are set to zero.

After these modifications, the kinematics and dynamics of the floating base system can be calculated from the standard derivation process of tree-type systems.

3.2. Closed-chain systems

Examples of closed-chain systems include parallel link manipulators and robotic platforms, such as the Stewart platform. Such structures can be described by connecting multiple chains or tree-type systems by imposing geometrical constraints.

To illustrate this process, consider the parallel link manipulator shown in Fig. 2(b). The original tree-type system comprises three chains, and the objective is to connect two chain tips to construct a parallel link mechanism. To describe the constraint, we define frames Σ_a and Σ_c on each tip and choose their initial configurations to coincide with each other. Let V^b_{sa} and V^b_{sc} be the body velocity of these frames, and let V^b_{ac} be their relative body velocity. From (13), (11), and (8), by introducing an intermediate frame Σ_s (base frame), V^b_{ac} can be described by V^b_{sa} and V^b_{sc} as

$$V^b_{ac} = \mathrm{Ad}^{-1}_{g_{sc}} V^b_{as} + V^b_{sc} = -\mathrm{Ad}^{-1}_{g_{sc}} \mathrm{Ad}_{g_{sa}} V^b_{sa} + V^b_{sc} \tag{33}$$

$$= \underbrace{\left(-\mathrm{Ad}^{-1}_{g_{sc}} \mathrm{Ad}_{g_{sa}} J^b_{sa} + J^b_{sc}\right)}_{J^b_{ac}} \dot{\theta}. \tag{34}$$

The constraint can be represented by constraining the elements of $V^b_{ac} = \left[\left(v^b_{ac}\right)^{\mathrm{T}} \left(\omega^b_{ac}\right)^{\mathrm{T}}\right]^{\mathrm{T}}$ as

$$B_c^{\mathrm{T}} V_{ac}^b = B_c^{\mathrm{T}} J_{ac}^b \dot{\theta} = 0. \tag{35}$$

In (35), $B_c \in \mathbb{R}^{6\times n_c}$ is a constant matrix comprising unit vectors $e_i^{\mathrm{T}} \in \mathbb{R}^6$, where $e_1 = [1,0,0\,|\,0,0,0]^{\mathrm{T}}, e_2 = [0,1,0\,|\,0,0,0]^{\mathrm{T}}, \cdots, e_6 = [0,0,0\,|\,0,0,1]^{\mathrm{T}}$. In addition, the complement matrix $\overline{B}_c$ is defined such that the column vectors of B_c and $\overline{B}_c$ span $\mathbb{R}^6$. For example, if Σ_c can rotate freely with respect to Σ_a, $B_c = [e_1, e_2, e_3]$ and $\overline{B}_c = [e_4, e_5, e_6]$. If Σ_c can rotate only about the z-axis, $B_c = [e_1, e_2, e_3, e_4, e_5]$ and $\overline{B}_c = e_6$. The freedom of these constraints can be regarded as those of additional joints (a ball joint in the first example, and a 1-DOF revolute joint in the second example). These joints are considered the connecting joints. The connecting joints can be active, and $\overline{B}_c$ represents the direction of their joint force/torque. If several closed paths exist, their constraints are stacked in the form of (35).

According to the constraint, the constraint force λ_c arises in the direction of B_c of V_{ac}^b. If the connecting joint is active, the joint force/torque also arises in the direction of $\overline{B}_c$. Upon adding these constraints and forces to the original three-type system dynamics, the closed-chain system dynamics can be expressed as

$$\begin{cases} M\ddot{\theta} + C\dot{\theta} + N = \tau + \left(J_{ac}^b\right)^{\mathrm{T}} \left\{B_c\lambda_c + \overline{B}_c\overline{\lambda}_c\right\} & (\text{closed} - \text{chain system dynamcs}) \\ B_c^{\mathrm{T}} J_{ac}^b \dot{\theta} = 0 & (\text{constraint}) \end{cases} \tag{36}$$

3.3. Platform systems

Upon combining the procedures for the floating base and the closed-chain system, robotic platforms can be described by considering the kinematics and dynamics of the tree-type system. This situation is illustrated in Fig. 2(c). The original system comprises two tree-type systems (legs) and a floating base system (table). The kinematics and dynamics in terms of the joint velocity $\dot{\theta}$ can be readily derived as in (36) by following the procedure described in the previous sections.

For the design and control of robotic platforms, it is sometimes convenient to describe the kinematics and dynamics in terms of the table position/orientation. Let Σ_t be the tool frame fixed on the table, and V_{st}^b be the body velocity of Σ_t with respect to Σ_s. Assuming that the constraint holds, V_{st}^b can be described by the joint angle as $V_{st}^b = J_{st}^b\dot{\theta}$ from (8) as well as by the position and rotation vectors $x_t = \left[p_{st}^{\mathrm{T}} \;\; r_{st}^{\mathrm{T}}\right]^{\mathrm{T}}$ of Σ_t as $V_{st}^b = T_t\dot{x}_t$, similar to (21). Upon equating both equations, the relationship between $\dot{x}_t$ and $\dot{\theta}$ can be given by

$$V_{st}^b = T_t\dot{x}_t = J_{st}^b\dot{\theta}. \tag{37}$$

It is also possible to describe the constraint (35) in terms of $\dot{x}_t$. From (8), (13), and (37),

$$V_{sa}^b = J_{sa}^b\dot{\theta} \tag{38}$$

$$V_{sc}^b = \mathrm{Ad}_{g_{tc}}^{-1} V_{st}^b + V_{tc}^b = \mathrm{Ad}_{g_{tc}}^{-1} T_t\dot{x}_t, \tag{39}$$

and we use $V_{tc}^b = 0$ because both frames are fixed on the same link. Upon substituting (38) and (39) into (33) and applying the resultant equation to $B_c^{\mathrm{T}} V_{ac}^b = 0$, the constraint can be represented by

$$B_c^{\mathrm{T}}\mathrm{Ad}_{g_{tc}}^{-1} T_t\dot{x}_t = B_c^{\mathrm{T}}\mathrm{Ad}_{g_{sc}}^{-1}\mathrm{Ad}_{g_{sa}} J_{sa}^b\dot{\theta}. \tag{40}$$

Eqs. (37) and (40) form a kinematic relation between $\dot{\theta}$ and $\dot{x}_t$, and it is given by

$$\underbrace{\begin{bmatrix} T_t \\ B_c^{\mathrm{T}}\mathrm{Ad}_{g_{tc}}^{-1} T_t \end{bmatrix}}_{\bar{T}_t} \dot{x}_t = \underbrace{\begin{bmatrix} J_{st}^b \\ B_c^{\mathrm{T}}\mathrm{Ad}_{g_{sc}}^{-1}\mathrm{Ad}_{g_{sa}} J_{sa}^b \end{bmatrix}}_{\bar{J}_{st}} \dot{\theta}. \tag{41}$$

This kinematic equation is used for the cost evaluation described in Section 7.1.

4. Optimization using GA

As noted in the previous section, the kinematics and dynamics of tree-type systems can be described by the parameters defined from a single base frame, and we can conveniently use these aspects for system design. Because optimization problems are large scale and highly nonlinear with mixed continuous and discrete variables, a GA [40–42] is employed for the optimization. This section describes the necessary procedures to formulate the design optimization problem by using the GA.

4.1. GAs

The basic concept of a GA involves observing a population of N samples called a *string*. Each string is allocated a fitness that represents its performance and the probability of its survival through time. The population of strings undergoes three adaptation phases to produce the next generation by mimicking the process of nature evolution, as described below:

The first phase is called *reproduction*, in which a probability process is used to select the strings to give birth to their offspring. The probability of survival in the next generation is weighted by the fitness of the strings. Some of the best population, called the *elite*, are selected automatically and preserved in the next generation without performing the adaptation process described below.

The second phase is called *crossover*, in which the selected individuals are merged. In this process, at a specified probability, each string is cut into two parts, and the parts from different strings are then combined to produce new individuals. To avoid being stuck in the local minima, the crossover can be conducted for different levels of strings [51,52], such as for genes (elements of string), chromosomes (groups of elements), and individuals (whole set of groups). For example, if s_{j_k} and s_{j_l} are the elements of a string s_j and are the members of a group, we also conduct the crossover process for each element s_{j_k} and s_{j_l}, the group $s_{j_{kl}} = [s_{j_k}\ s_{j_l}]$, as well as the whole string s_j.

The final phase is called the *mutation* phase, in which one value in a string is changed to another random value with a specified probability. The mutation can also be conducted for different levels of strings. One cycle of these three phases produces a new set of strings called the *next generation*. The cycle is repeated until the best fitness becomes constant or the target value is attained.

To conduct the adaptation processes, the GA strings are usually represented by binaries. By adopting the binary strings, we can easily implement the adaptation process by exchanging or manipulating their bits. Therefore, the main challenge in design optimization is to enable the binary coding of the system parameters for the GA process. In the exponential formulation, the parameters that we wish to code into the strings are the exponential parameters $\{q_i, v_i, \omega_i\}$ and the chain matrix $\{C_h\}$. Note that the length of the links l_{i^c} is determined from the joint position q_i, and it is not a design parameter. The link mass m_{i^c} and inertia I_{i^c} can also be functions of l_{i^c}.

4.2. Binary string and rate decoding

Binary strings can be naturally associated with natural numbers. Consequently, we identify the string and the natural number and denote both by s; that is, $s \in \mathbb{B}^{n_s}$ or $s \in \mathbb{N}$, where $\mathbb{B}$ is the set $\{0, 1\}$ and $\mathbb{N}$ is the set of natural numbers. A binary string $s \in \mathbb{B}^{n_s}$ can be associated with a real number (rate or percentage) between $0 \leqslant p_s \leqslant 1$ by

$$p_s = \frac{s}{2^{n_s} - 1}. \tag{42}$$

Rate decoding p_s can be used in several ways depending on the problem. A real number x in a range $x_0 \leqslant x \leqslant x_1$ can be represented as

$$x = x_0 + (x_1 - x_0)p_s. \quad \text{(interval)} \tag{43}$$

A total number x can be distributed to x_i's by using multiple rate decoding strings p_{s_i} sequentially; for example,

$$x_1 = xp_{s_1}, x_2 = (x - x_1)p_{s_2}, \cdots. \quad \text{(distribution)} \tag{44}$$

We can align the position x_i accordingly between a range $x_0 \leqslant x_i \leqslant x_0 + x$ by

$$x_1 = x_0 + xp_{s_1}, x_2 = x_1 + (x - x_1)p_{s_2}, \cdots. \quad \text{(sequential)} \tag{45}$$

These values can be rounded if natural numbers are required.

The exponential parameters are immediately described by the rate strings with this decoding. Note that the following coding will be much more complicated if the joint parameters are defined relative to its parent joint (not from the base frame) like the DH parameters.

4.3. GA coding for exponential parameters $\{q_i, v_i, \omega_i\}$

The joint position q_i can be determined from the triplet position binary strings $s_{q_{i_x}} \in \mathbb{B}^{n_{q_{i_x}}}, s_{q_{i_y}} \in \mathbb{B}^{n_{q_{i_y}}}, s_{q_{i_z}} \in \mathbb{B}^{n_{q_{i_z}}}$. The position can be chosen within a region by using (43) or can be ordered in a region by using (45).

The joint movement direction $\{v_i, \omega_i\}$ can be determined from the joint type (rotation or translation) and the unit vectors v_i or ω_i. These vectors can be described by two real numbers ϕ_i and γ_i in the polar coordinates as

$$v_i = \begin{bmatrix} \cos(\phi_i)\cos(\gamma_i) \\ \cos(\phi_i)\sin(\gamma_i) \\ \sin(\gamma_i) \end{bmatrix}, \quad \omega_i = \begin{bmatrix} \cos(\phi_i)\cos(\gamma_i) \\ \cos(\phi_i)\sin(\gamma_i) \\ \sin(\gamma_i) \end{bmatrix}, \tag{46}$$

where $0 \leqslant \phi_i \leqslant 2\pi, 0 \leqslant \gamma_i \leqslant \frac{\pi}{2}$. The joint type can be specified by the one-bit binary string $s_{t_i} \in \mathbb{B}$. ϕ_i and γ_i can be described by the binary strings $s_{\phi_i} \in \mathbb{B}^{n_{\phi i}}$ and $s_{\gamma_i} \in \mathbb{B}^{n_{\gamma i}}$ with the interval decoding described in (43).

The next section describes the coding procedure of the chain matrix C_h.

5. GA coding for chain matrix

If the number of joints n and number of chains C are given, the chain matrix C_h is a $(0,1)$-valued $C \times n$ matrix. We can determine a $(0,1)$ matrix by the binary string for each element; however, most of them are infeasible from a structural viewpoint. In particular, the chain matrix can be coded simply by using $2C-1$ binary strings with range decoding, as described below.

5.1. Tree-type architectures

To examine the chain matrix characteristics from the tree-type system architecture, the example shown in the left-hand side in Fig. 3 is considered. The system comprises seventeen joints and five chains, and thus, the matrix is $C_h \in \mathbb{R}^{5\times 17}$. Recall that a row and a column of the chain matrix correspond to a chain and a joint, respectively; then, we can derive the chain matrix of the system as

$$C_h = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}. \tag{47}$$

It can be validated that the matrix elements are not random and have several series $(0,1)$ patterns arising from the tree-type system architecture. We utilize this aspect for the GA coding and determine the chain matrix from a smaller number of strings containing the key information. Toward this end, we note that the system comprises chains branching at the links driven by a joint, known as the branching joint. Subsequently, the system can be regarded as an assembly of the trunk and branches, as shown in the right-hand side of Fig. 3. Note that the branching joint is not included in the branch, and the branch starts from a link. When the start links are connected to a branching joint in the assembly, they are merged into the parent links. Therefore, the one-to-one relation of the joint and the link is retained, as shown in the left-hand side of Fig. 3.

In the following analysis, to avoid the multiplicity of the structure from different chain matrices, we assume that the joint numbers are assigned from the root to the tip from the first chain to the last and that each branch starts (sprouts) from the trunk tip or the elder (smaller number) branches, as indicated by the red and blue dotted lines on the right-hand side of Fig. 3. Note that the first branch always starts from the trunk tip.

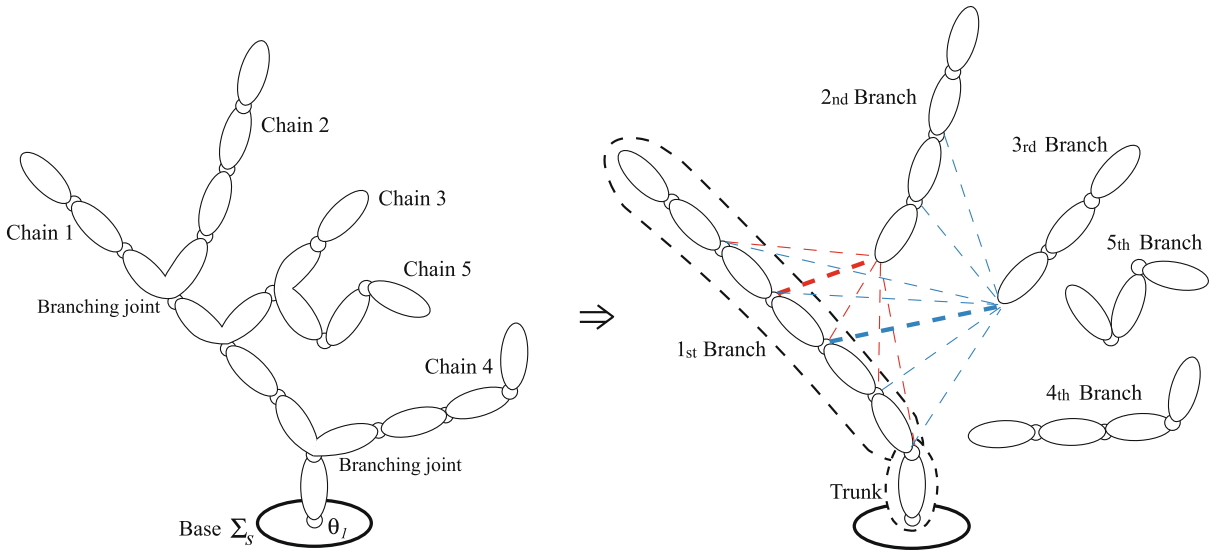


Fig. 3. Tree-type system and system components.

5.2. Construction of chain matrix from components

As mentioned previously, the system is characterized by the components (trunk and branches) and their connections to the branching joint. This section describes how the chain matrix is determined from this information.

The trunk and branches are characterized by the number of joints distributed to them. From these numbers, some of the elements of the chain matrix can be determined as

$$C_h = \left[\begin{array}{cc|cccccc|ccc|ccc|cc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & & & & & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & & & & & 0 & & & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & & & & & 0 & & & 0 & & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & & & & & 0 & & & 0 & & 0 & & & 0 & 1 & 1 \end{array}\right]. \tag{48}$$

trunk 1st branch ⋯

In (48), the partition of each block is first determined from the given numbers of joints. The columns of the first block correspond to the trunk joints, and thus, we fill one in the first block, as they are contained in all the chains. The columns of the remaining blocks correspond to the branch joints, and thus, we fill ones in the row corresponding to the branch number. The upper elements of the branch joints ones are zero according to the joint numbering assumption. The elements below one in the last column of each branch block are also zero because the tip joint cannot belong to other chains.

The system connection is determined by selecting the branching joint for each branch from the second branch to the last one. As shown in Fig. 3, the branching joint for the second branch is the fifth joint; therefore, we fill one in the fifth column in the second row. This process is continued until the last row (chain), which leads to the underlined ones in blue, as follows:

$$C_h = \left[\begin{array}{cc|cccccc|ccc|ccc|cc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & \underline{1} & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & \underline{1} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & \underline{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \underline{1} & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right]. \tag{49}$$

trunk 1st branch ⋯

From the position of the underlined ones (branching joints), the other elements in the same row can be determined. For the elements in the branch block involving the underlined ones (first branch block for the second row), we fill ones on the left and zeros on the right, as the chain branches off at the branching joint. The elements in the elder (or right-hand-side) branch blocks (third to fifth blocks for the second row) are set as zero from the numbering assumption. If younger branch blocks exist on the left-hand side of the underlined ones block (first and second branch blocks for the fifth row), we copy the (0,1) pattern from the row to which the branching joint belongs. For the fifth row, the branching joint belongs to the third row, and thus, we copy the elements of the first and second branch blocks in the third row. This process completes the chain matrix.

5.3. Strings for chain matrix

From the above observation, for a given n and C, the chain matrix can be constructed from.

- n_t: number of joints in trunk
- n_{b_i} $(i = 1, 2, \cdots, C-1)$: number of joints in branch i (for the last branch, $n_{b_C} = n - (n_t + n_{b_1} + \cdots + n_{b_{C-1}})$)
- n_{c_i} $(i = 2, \cdots, C)$: branching joint number for branch i (for the first branch, $n_{c_1} = n_t$)

These numbers ((2C-1) in total) can be coded using binary strings with the following rate decoding:

- For the numbers of joints n_t and n_{b_i}, we use strings $s_{n_t} \in \mathbb{B}^{n_{n_t}}, s_{n_{b_i}} \in \mathbb{B}^{n_{n_{b_i}}}$ and apply the distribution decoding in (44) with rounding.
- For the branching joint number n_{c_i}, we use a string $s_{n_{c_i}} \in \mathbb{B}^{n_{n_{c_i}}}$ and apply the interval decoding in (43) with rounding. The range of n_{c_i} is given by $n_{c_i} \in [0, (n_{b_1} - 1) + \cdots + (n_{b_{i-1}} - 1)]$ as the tip joint cannot be the branching joint. If $n_{c_i} = 0$, the trunk tip joint is the branching joint.

In the above example (49), the joint distributions are $n_t = 2, n_{b_1} = 5, n_{b_2} = 3, n_{b_3} = 2$, and $n_{b_4} = 3$. The number and possible ranges of the branching joint location are $n_{c_2} = 3 \in [0, 4], n_{c_3} = 2 \in [0, 6], n_{c_4} = 0 \in [0, 7], n_{c_5} = 7 \in [0, 9]$.

6. Overall GA procedure

It has been demonstrated that if the numbers of the joints n and chains C are given, the optimization parameters are the joint position q_i, joint direction $\{v_i, \omega_i\}$, and chain matrix C_h. Therefore, to minimize (or maximize) a cost J, the optimization problem can be stated as.

Optimization problem

- given: n and C
- find: q_i, v_i, ω_i, and C_h
- minimize (or maximize): J

The optimization parameters are coded into the GA strings as

GA strings

- $q_i : s_{q_{i_x}} \in \mathbb{B}^{n_{q_{i_x}}}, s_{q_{i_y}} \in \mathbb{B}^{n_{q_{i_y}}}, s_{q_{i_z}} \in \mathbb{B}^{n_{q_{i_z}}}$ (interval/sequential decoding) $(i = 1, \cdots, n)$
- $\{v_i, \omega_i\}$: $s_{t_i} \in \mathbb{B}$ (binary decoding), $s_{\phi_i} \in \mathbb{B}^{n_{\phi_i}}, s_{\gamma_i} \in \mathbb{B}^{n_{\gamma_i}}$ (interval decoding) $(i = 1, \cdots, n)$
- $n_t, n_{b_i} : s_{n_t} \in \mathbb{B}^{n_{n_t}}, s_{n_{b_i}} \in \mathbb{B}^{n_{n_{b_i}}}$ (distribution decoding) $(i = 1, \cdots, C-1)$
- $n_{c_i} : s_{n_{c_i}} \in \mathbb{B}^{n_{n_{c_i}}}$ (interval decoding) $(i = 2, \cdots, C)$

For the GA optimization, we assign the above strings for each individual. For the crossover and mutation operation for different levels, from the lower-level strings above, we define the higher-level string groups as follows:

String groups

- intermediate level: $s_{q_i} = \left[s_{q_{i_x}}, s_{q_{i_y}}, s_{q_{i_z}}\right]$, $s_{v_i} = \left[s_{t_i}, s_{\phi_i}, s_{\gamma_i}\right]$
- higher level: $s_q = \left[s_{q_1}, \cdots, s_{q_n}\right], s_v = \left[s_{v_1}, \cdots, s_{v_n}\right]$, $s_{C_h} = \left[s_{n_t}; s_{n_{b_1}}, \cdots, s_{n_{b_{n-1}}}; s_{n_{c_2}}, \cdots, s_{n_{c_n}}\right]$
- total system: $s = \left[s_q, s_v, s_{C_h}\right]$

For the individual j, we denote the total system string as s_j.

Fig. 4 shows a typical optimization process. From the initial population of N individuals (or strings) s_j, we obtain the optimization parameters $q_i, \{v_i, \omega_i\}$, and C_h from the decoding rule of the strings. From these parameters, the physical parameters of the link mass, length, and inertia (m_{i^c}, l_{i^c}, and I_{i^c}, respectively); twist ξ_i; and simplified chain matrix $\widehat{C}_h$ can be obtained. The kinematics/dynamics and a controller can be determined from these values (see Section 7 for details). By using the system and controller, a simulation can be performed to evaluate the fitness (or cost) J_j for each individual. If the best fitness attains the target value or becomes constant through the iteration, the optimal structure can be obtained. If not, the adaptation process is performed to yield the new generation, and the entire process is repeated.

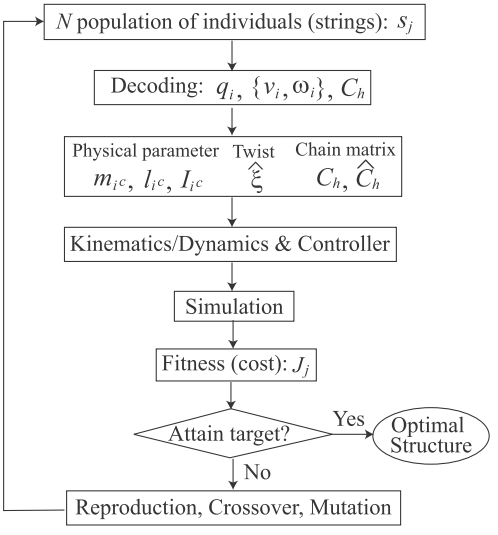


Fig. 4. Process flow for GA.

7. Cost evaluation using feedback control

For the cost evaluation, we conduct a numerical simulation for each individual obtained from the strings. Because the general closed-form formulas for the kinematics and dynamics of tree-type systems are available, a controller of any type can be constructed from them. This allows us to perform numerical simulations to evaluate a variety of costs during the optimization process, for which analytical calculation formulas are unavailable. This section illustrates this evaluation process.

7.1. Cost evaluation by kinematics control

One of the most important measures that can be evaluated by the kinematics is the region that the manipulators can access. This region is referred to as the workspace, and it is especially important in designing closed-link manipulators. Solving the inverse kinematics is the most straightforward approach, and it is efficient for some simple manipulators [53,54]. However, this approach is intractable in general as it requires solving several multivariable nonlinear equations [2]. This difficulty arises from tackling the position-level kinematics directly and can be alleviated by conducting a numerical simulation for the velocity-level kinematics with feedback control. This section illustrates this process by taking an example of the robotic platform shown in Fig. 5(a).

7.1.1. Basic idea

As is described in Section 3.3, the kinematics of the robotic platform between the joint angle and the tool flame is given by (41). This equation can be viewed as a linear equation of $\dot{x}_t \in \mathbb{R}^6$ characterized by the joint velocity $\dot{\theta} \in \mathbb{R}^n$ and the coefficient matrices $\overline{T}_t \in \mathbb{R}^{(6+n_c)\times 6}$ and $\bar{J}_{st} \in \mathbb{R}^{(6+n_c)\times n}$ determined from the current joint angle θ. From linear algebra, a necessary and sufficient condition for the solution existence can be expressed as

$$\mathrm{rank}\overline{T}_t = \mathrm{rank}\begin{bmatrix}\overline{T}_t & \bar{J}_{st}\dot{\theta}\end{bmatrix}. \tag{50}$$

If (50) holds, one of the solutions (minimum norm solution) for (41) can be given as

$$\dot{x}_t = \overline{T}_t^{+}\bar{J}_{st}\dot{\theta}, \tag{51}$$

where $\overline{T}_t^{+}$ is the pseudo-inverse of $\overline{T}_t$. If $\mathrm{rank}\overline{T}_t = 6$, (51) is the unique solution.

Eq. (50) can be used to evaluate the workspace. Suppose that the tool frame Σ_t is within the workspace at the current configuration θ and that we wish to move Σ_t toward the direction specified by $\dot{\theta}$. If (50) holds and has a solution $\dot{x}_t$, Σ_t can move along the direction $\dot{x}_t$. Consequently, for a small Δt, we can expect that $x_t + \dot{x}_t\Delta t$ is still within the workspace. If (50) does not hold, the specified direction is directed out of the workspace, and thus, Σ_t lies on the boundary of the workspace. Because the workspace is usually connected and has a smooth boundary, the workspace can be estimated by controlling the tool frame x_t along a path sweeping over a region of interest from the center until the condition (50) fails. The numerical simulation for this purpose can be conducted by using (41) as the system whose control input is $\dot{\theta}$.

In the following sections, we define some typical path alternatives and a controller for the workspace evaluation.

7.1.2. Trajectory for workspace evaluation

As is well known, a path $p(s) \in \mathbb{R}^3$ can be represented using a parameter between an interval $s = [0, 1]$. We can describe a segment of any interval using s:

$$l_k = ks \quad (\text{line}), \qquad \phi_k = 2\pi ks \quad (\text{angle}), \tag{52}$$

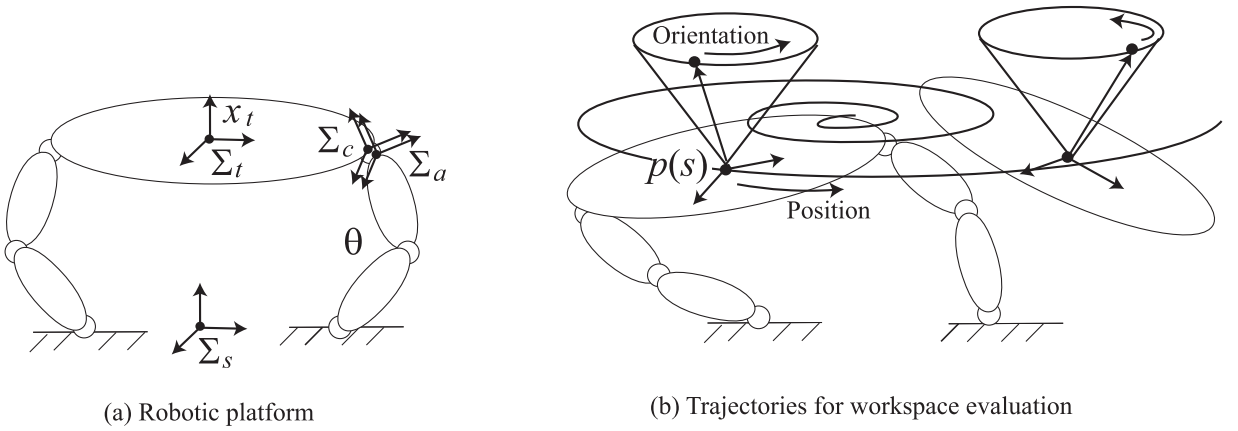
where $k \in \mathbb{R}$. By using these segments, we can describe a variety of paths; for example,

$$p(k_1) = \begin{bmatrix} r\cos(\phi_{k_1}) \\ r\sin(\phi_{k_1}) \\ h \end{bmatrix} (\text{circle}), \quad p(k_1,k_2) = \begin{bmatrix} l_{k_2}\cos(\phi_{k_1}) \\ l_{k_2}\sin(\phi_{k_1}) \\ h \end{bmatrix} (\text{spiral}), \tag{53}$$

$$p(k_1,k_2) = \begin{bmatrix} r\cos(\phi_{k_1}) \\ r\sin(\phi_{k_1}) \\ l_{k_2} \end{bmatrix} (\text{helix}), \quad p(k_1,k_2) = \begin{bmatrix} r(1-l_{k_2})\cos(\phi_{k_1}) \\ r(1-l_{k_2})\sin(\phi_{k_1}) \\ l_{k_2} \end{bmatrix} (\text{cone}), \tag{54}$$

$$p(k_1,k_2) = \begin{bmatrix} r\cos(\phi_{k_2})\cos(\phi_{k_1}) \\ r\cos(\phi_{k_2})\sin(\phi_{k_1}) \\ r\sin(\phi_{k_2}) \end{bmatrix} (\text{sphere}). \tag{55}$$

Fig. 5(b) shows the typical trajectories to evaluate the workspace for a robotic platform. The possible translational region of the table center x_t can be evaluated by a spiral trajectory expanding from the center position. At each translational position, the table is required to tilt by a certain angle, which can be evaluated by a circle trajectory for the table normal to rotate.



(a) Robotic platform

(b) Trajectories for workspace evaluation

Fig. 5. Trajectory for workspace evaluation.

The spiral for the translation can be produced by the second equation of (53) with $\phi_{k_1} = 2\pi k_1 s$ and $l_{k_2} = k_2 s$, where $k_1 \gg k_2$ (the translational rotation should be faster than the radius expansion). The circle trajectory for the tilting is given by the first equation of (53) with $\phi_{k_3} = k_3 s$, where $k_3 \gg k_1$ (the tilting rotation should be faster than the translational rotation).

7.1.3. Controller and controlled system kinematics

For the workspace evaluation, we design a tracking controller for the system (41). Because (51) is applicable if (50) holds, a simple choice for the controller is

$$\dot{\theta}_d = \left(\overline{T}_t^{+}\bar{J}_{st}\right)^{+}\left\{\dot{x}_{t_d} - K_p\left(x_t - x_{t_d}\right)\right\}, \tag{56}$$

where $K_p > 0$ is some constant matrix, and $\left(\overline{T}_t^{+}\bar{J}_{st}\right)^{+}$ is the pseudo-inverse matrix. If $\left(\overline{T}_t^{+}\bar{J}_{st}\right) \in \mathbb{R}^{6\times n}$ is row full rank, $\left(\overline{T}_t^{+}\bar{J}_{st}\right)\left(\overline{T}_t^{+}\bar{J}_{st}\right)^{+} = I$ holds. Then, by substituting (56) into (51), the closed-loop system becomes

$$\left(\dot{x}_t - \dot{x}_{t_d}\right) + K_p\left(x_t - x_{t_d}\right) = 0, \tag{57}$$

which yields $x_t \to x_{t_d}$.

The system response can be calculated by combining the controlled system $(\overline{T}_t\dot{x}_t = \bar{J}_{st}\dot{\theta}_d)$ and the system kinematics (41). The system kinematics with feedback control can be described as

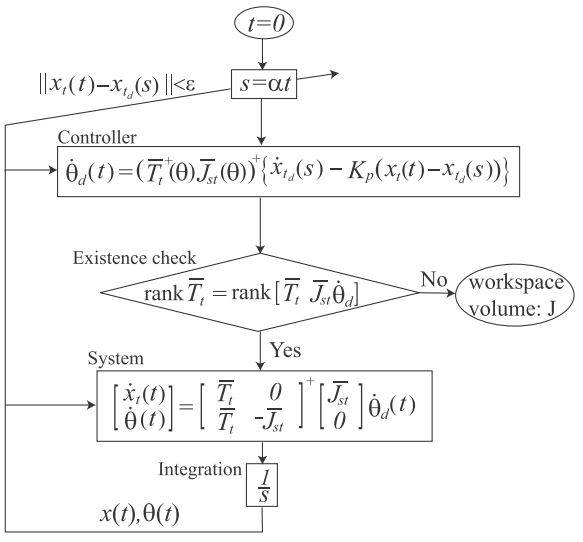


Fig. 6. Cost evaluation by kinematics control (robotic platform).

$$\begin{bmatrix} \overline{T}_t & 0 \\ \overline{T}_t & -\bar{J}_{st} \end{bmatrix} \begin{bmatrix} \dot{x}_t \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \bar{J}_{st} \\ 0 \end{bmatrix} \dot{\theta}_d. \tag{58}$$

7.1.4. Cost evaluation process

Fig. 6 shows the workspace evaluation procedure. We first choose a desired trajectory $x_{t_d}(s)$ to cover a region of interest, as described in Section 7.1.2. To adjust the speed of the desired trajectory, we introduce the adjusting parameter α and set $s = \alpha t$. The parameter α should be sufficiently small for the controlled system to follow $x_{t_d}(s)$. The control input $\dot{\theta}_d$ is calculated using (56), and the solution existence condition (50) is validated. If this condition holds, a numerical integration for (58) is performed and the time evolution is continued. If the condition does not hold, the tool frame reaches the workspace boundary, the simulation is terminated, and the workspace volume J is calculated from the area covered by $x_{t_d}(s)$ until this point. The trajectory speed parameter α can be adjusted by observing the tracking error $\|x_t - x_{t_d}\| < \varepsilon$ if necessary. Note that the position-level kinematics for θ do not need to be solved, as it is calculated using the numerical integration of (58).

We can choose any costs (manipulability, joint movement range, etc.) other than the workspace volume while guaranteeing the existence condition (50) by evaluating the cost during the simulation.

7.2. Cost evaluation by dynamics control

In some cases, we want to evaluate the costs involving the dynamic response including the input torque and the forces interacting with the environment. These costs can be evaluated by conducting the dynamical simulation with feedback control instead of kinematics control. This section illustrates this process by taking an example of the grasping/manipulation system.

7.2.1. Basic idea

As described in Section 2.4, the dynamics of the grasping/manipulation system is given by (17)–(19). For grasping and manipulation control, the following are assumed:

(A1) $\bar{J}_h \in \mathbb{R}^{n\times n}$ in (24) is nonsingular.
(A2) For any F_o, there exists f_N to make F_f satisfy (20).

The constraint Eq. (24) is a linear equation of $\dot{\theta}$. Therefore, the condition (A1) ensures that there exists a joint motion $\dot{\theta}$ satisfying the constraint for any object motion $\dot{\bar{x}}_t$. By contrast, the condition (A2) ensures that there exists an internal force f_N to regulate the fingertip force F_f inside the friction cone for any manipulating force F_o. These assumptions correspond to the manipulable and force closure conditions in the field [2].

We can evaluate the dynamic response of the grasping and manipulation system by controlling the augmented object motion $\bar{x}_o$ and the internal force f_N. The condition (A1) can be used to evaluate the workspace for the grasping/manipulation system.

7.2.2. Controller and controlled system dynamics

The system expression for grasping and manipulation is given by (27). A variety of tracking controllers can be designed, and a simple choice is a linearizing compensator with PID control [30], that is,

$$\tau = \tau_d = \overline{M}_o u_o + \overline{C}_o \dot{\bar{x}}_o + \overline{N}_o + J_h^{\mathrm{T}} K_G u_{f_N}, \tag{59}$$

where

$$u_o = \ddot{\bar{x}}_{o_d} - K_{d_o}\left(\dot{\bar{x}}_o - \dot{\bar{x}}_{o_d}\right) - K_{p_o}\left(\bar{x}_o - \bar{x}_{o_d}\right) \tag{60}$$

$$u_{f_N} = f_{N_d} - K_{I_f} \int \left(f_N - f_{N_d}\right) dt. \tag{61}$$

$K_{d_o}, K_{p_o}, K_{I_f} > 0$ are the control gains, and $\bar{x}_{o_d}$ and f_{N_d} denote the desired trajectories of $\bar{x}_o$ and f_N. Note that f_N in (61) can be calculated from F_f by using (25).

Upon substituting (59) with (60) and (61) into (27), the closed loop system becomes

$$\left(\ddot{\bar{x}}_o - \ddot{\bar{x}}_{o_d}\right) + K_{d_o}\left(\dot{\bar{x}}_o - \dot{\bar{x}}_{o_d}\right) + K_{p_o}\left(\bar{x}_o - \bar{x}_{o_d}\right) = 0 \tag{62}$$

$$\left(f_{f_N} - f_{N_d}\right) + K_{i_f} \int \left(f_N - f_{N_d}\right) dt = 0, \tag{63}$$

which yields $\bar{x}_o \to \bar{x}_{o_d}$ and $f_N \to f_{N_d}$.

The system response can be calculated by combining (17), (18) with (23), the time derivative of (24), and (59). The system dynamics with feedback control can be described as

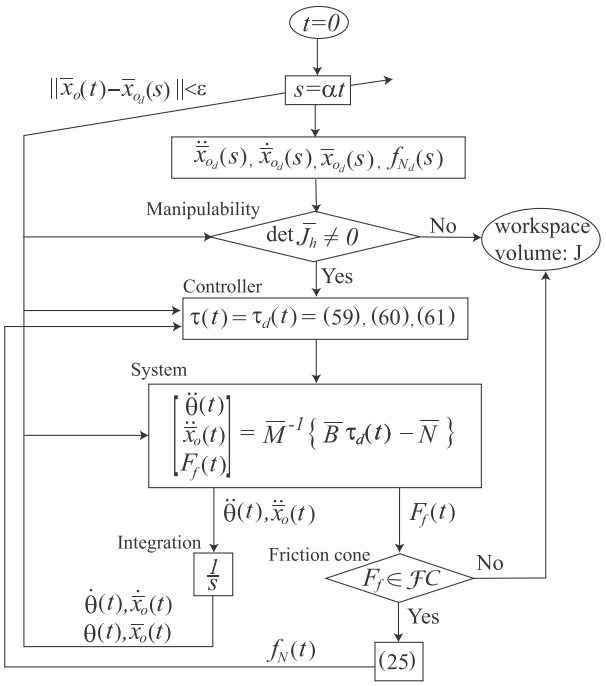


Fig. 7. Cost evaluation by dynamics control (grasping/manipulation).

$$\underbrace{\begin{bmatrix} M_f & 0 & J_h^{\mathrm{T}} \\ 0 & M_o\widehat{T}_o & -G \\ \bar{J}_h & -\bar{T}_o & 0 \end{bmatrix}}_{\bar{M}} \begin{bmatrix} \ddot{\theta} \\ \ddot{\bar{x}}_o \\ F_f \end{bmatrix} = \underbrace{\begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix}}_{\bar{B}} \tau_d - \underbrace{\begin{bmatrix} C_f\dot{\theta} + N_f \\ C_o\widehat{T}_o\dot{x}_o + N_o + M_o\dot{\widehat{T}}_o\dot{x}_o \\ \dot{\bar{J}}_h\dot{\theta} - \dot{\bar{T}}_o\dot{x}_o \end{bmatrix}}_{\bar{N}} \tag{64}$$

7.2.3. Cost evaluation process

Like Fig. 6 for the kinematics control, Fig. 7 shows the evaluation process by using dynamics control. From the region of interest, we first choose the desired trajectories $\bar{x}_{o_d}(s)$ and $f_{N_d}(s)$ according to Section 7.1.2. The desired trajectory for the internal force magnitude $f_{N_d}(s)$ can be a constant estimated a priori from the object weight. If the manipulable condition (A1) holds, the control input τ_d is calculated by (59) with (60) and (61). If not, the system reaches the boundary of the workspace. By applying the control input, the system response is calculated according to (64), and a numerical integration for the solutions $\ddot{\theta}$ and $\ddot{\bar{x}}_o$ is performed. For the fingertip force solution F_f, the friction cone condition (20) should be checked. If it holds, the time evolution is continued.

We can choose any cost (work, system mass, etc.) other than the workspace volume while guaranteeing the manipulability and friction cone conditions over the region.

In the next two sections, we illustrate the optimization process through two numerical examples. In the first example, we optimize the joint position of a robotic platform to maximize the workspace by using the kinematics control. In the second example, we optimize the grasping/manipulation system structure (geometry and topology) to minimize the link mass by using the dynamics control. In the following simulation, the Matlab/Simulink commercial package (R2018b, MathWorks Inc., Natick, MA) is used. The parameters for the numerical integration are set to auto with variable step size and the default options (relative error tolerance = 10^{-3}, maximum step = auto, etc.).

8. Optimization of robotic platform

In the first example, we consider a simple optimization problem of a robotic platform by using the kinematics control described in Section 7.1. In the design, the joint position of the base is optimized to maximize the workspace.

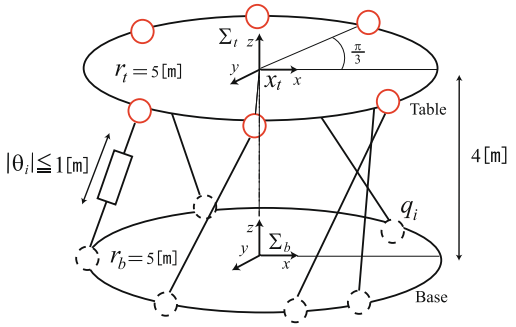


Fig. 8. Robotic platform for optimization.

8.1. Problem settings

For the first design, we consider the robotic platform shown in Fig. 8. The system comprises the base, table, and six connecting legs. Each leg has two ball joints on each end and one translational joint at the middle. All the ball joints are passive, and the translational joints are active to be controlled. The table joint position is fixed and is distributed evenly on the rim (interval of $\pi/3$). The base joint position is variable along the rim and is the design parameter. The translational joints have a maximum elongation of $|\theta_i| \leqslant 1$ [m]. The radius of the base and table is $r_b = r_t = 5$ [m]. The initial position of the table is $x_t = \left[p_{t_x}, p_{t_y}, p_{t_z} \,|\, \theta_{t_x}, \theta_{t_y}, \theta_{t_z}\right]^{\mathrm{T}} = [0, 0, 4 \,|\, 0, 0, 0]^{\mathrm{T}}$.

In the GA optimization, we design the base joint position

$$q_i = [r_b \cos\psi_i,\, r_b \sin\psi_i,\, 0]^{\mathrm{T}}, \tag{65}$$

that is parametrized by $0 \leqslant \psi_i \leqslant 2\pi$ $(i = 1, \cdots, 6)$. Therefore, we introduce the string $s_{q_{i_\psi}} \in \mathbb{B}^{n_{q_{i_\psi}}}$ and set its length (bits) as $n_{q_{i_\psi}} = 6$. The number of the population is set as $N = 50$, and the initial population is produced from random numbers. The adaptation process is conducted for the local-level string $s_{q_{i_\psi}}$ as well as the higher-level sting $s_q = \left[s_{q_1}, \cdots, s_{q_6}\right]$. Table 1 summarizes the probabilities of the adaptation process. For the reproduction, we use the *roulette wheel selection* [40] strategy, in which we distribute all the fitness-weighted strings onto a wheel and then generate a random number within that wheel to select the next offspring.

For each individual, we conduct a numerical simulation for the kinematics (58). The controller is given by (56), and the control gain is set as $K_p = 10$. The desired trajectory for the table position is a spiral and that for the table normal is a circle to tilt, as described in the example in Section 7.1.2. The trajectory parameters are set as $k_1 = 3, k_2 = 0.24$, and $h = 4$ for spiral translation and $k_3 = 10$ for tilting rotation. The simulation is stopped if any of the following three conditions fails: solution existence condition (50); maximum translational joint elongation $|\theta_i| \leqslant 1$; or no collision between the legs. The fitness (cost) is the simulation duration time T because it is directly correlated with the spiral expansion and thus the workspace.

8.2. Optimization results

Fig. 9 shows the evolution of the best individual fitness. The best time duration increases as the generation proceeds, and the optimal design achieves $T = 10$ [s]. Fig. 10(b) shows the spiral trajectory during the time duration (showing the workspace).

Fig. 10(a) and (b) respectively show the initial and final configuration of the best individual. The optimal parameters (angle interval) are

$$\psi_1 = \frac{\pi}{6}, \psi_2 = \frac{3\pi}{6}, \psi_3 = \frac{5\pi}{6}, \psi_4 = \frac{7\pi}{6}, \psi_5 = \frac{9\pi}{6}, \psi_6 = \frac{11\pi}{6}. \tag{66}$$

The results show that the design distributes the base joint evenly along the rim. The joints are located just below the table joints; therefore, the legs go straight up to the table from the base center. This result seems reasonable because we evaluate the workspace by sweeping the spiral region expanding from the center. The parallel vertical-leg configuration is beneficial to utilize the translational joint elongation effectively for the symmetrical circular region.

Table 1
GA adaptation probability (platform optimization).

Adaptation	String level	Values
Reproduction	–	Roulette Wheel Selection
	Number of elites	$N_l = 1$
Crossover	Higher level	$p_{c_H} = 0.300$
	Lower level	$p_{c_L} = 0.550$
Mutation	Higher level	$p_{m_H} = 0.0125$
	Lower level	$p_{m_L} = 0.0300$

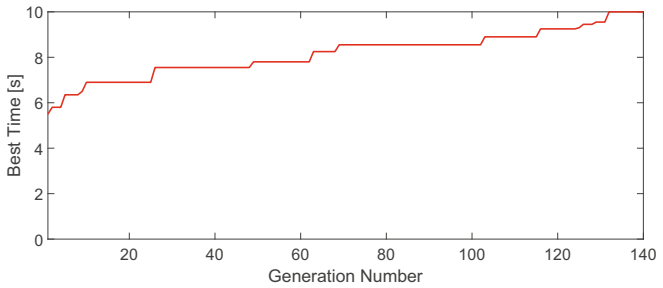
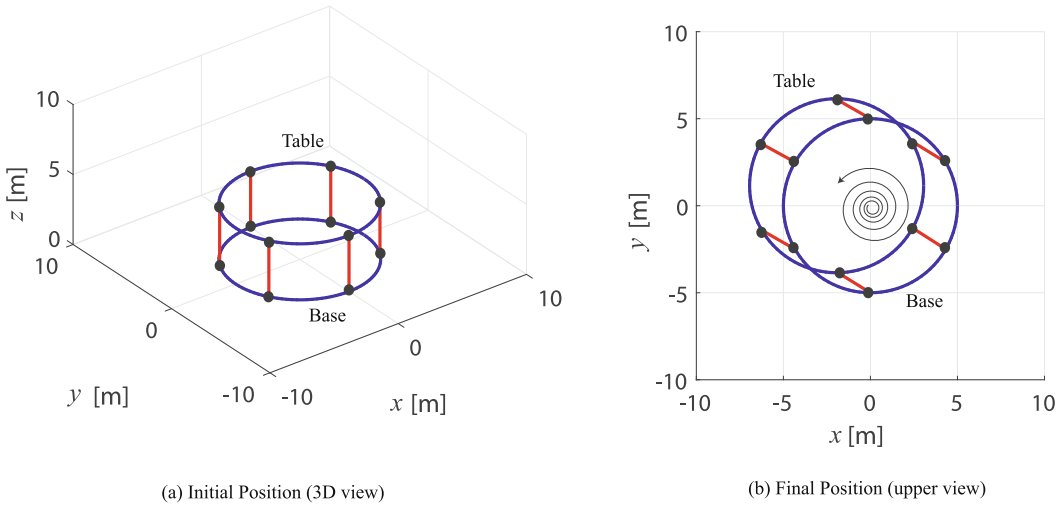


Fig. 9. Evolution of the best individual (platform optimization).



(a) Initial Position (3D view)

(b) Final Position (upper view)

Fig. 10. Initial and final configuration (platform optimization).

9. Optimization of grasping/manipulation system

In the second example, we consider an optimization problem of the grasping/manipulation system by using the dynamics control described in Section 7.2. In the design, both the joint position/direction (geometry) and the joint distribution/connection (topology) are optimized simultaneously.

9.1. Classical design

Before we proceed to the design optimization, we first conduct a grasping and manipulation simulation using the classical robot hand shown in Fig. 11. The physical parameters are shown in the figure.

In this design, the palm is connected on the roof thorough a rotational joint, and three 3-DOF fingers are installed on the edges of the palm ($n = 10$ and $C = 3$). The grasped object is a cube, and the contact points are chosen as in the right-hand-side figure. The initial object position is $x_o = \left[p_{o_x}, p_{o_y}, p_{o_z} \,|\, \theta_{o_x}, \theta_{o_y}, \theta_{o_z}\right]^{\mathrm{T}} = [0, 0, 0.8 \,|\, 0, 0, 0]^{\mathrm{T}}$. The system has $n - k = 10 - 9 = 1$

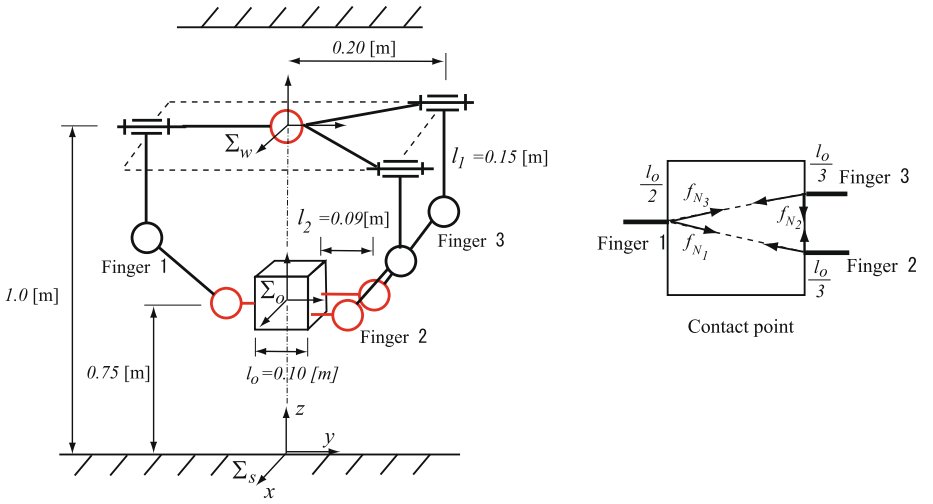
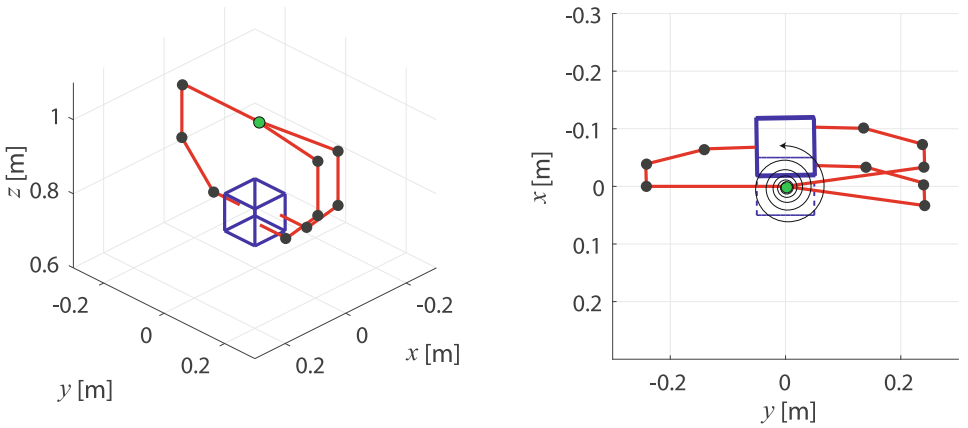


Fig. 11. Classical robot hand.



(a) Initial position (3D view)

(b) Final position (upper view)

Fig. 12. Initial and final configurations (classical hand).

redundancy, and we choose the internal velocity $\dot{x}_n$ for simplicity. We assume that the links have a cylindrical shape with radius of $r = 0.01$ [m] and are made of steel with a mass density of $\rho = 4000$ [kg/m^3]. The total mass $M = \sum \rho \pi r^2 l_i$ of the links is $M_{ref} = 2.3516$ [kg].

As in the first design example in Section 8, the desired trajectory for the object position is a spiral and that for the object normal is a circle to tilt. The trajectory parameters are set as $k_1 = 4.7, k_2 = 0.006$, and $h = 0.75$ for spiral translation and $k_3 = 10$ for tilting rotation. Fig. 12(a) and (b) respectively show the initial and final configurations. Fig. 12(b) also shows the translational spiral trajectory. The desired trajectory of the internal motion is set as $x_n \equiv 0$ and that of the internal force magnitude is set as $f_{N_d} = \left[f_{N_{d_1}}, f_{N_{d_2}}, f_{N_{d_3}}\right]^{\mathrm{T}} \equiv [10, 3, 10]^{\mathrm{T}}$ (see Fig. 11). The controller is given by (59), and the control gains are set as $K_{d_o} = 10 I_9, K_{p_o} = 300 I_9$, and $K_{I_f} = 0.01 I_3$. The dynamics of the system is calculated using (64), and during the task, the manipulability and friction cone conditions as well as whether the links collide with the object are checked.

According to the simulation, the classical hand can successfully conduct the task without violating the manipulability and friction cone conditions while avoiding collisions with the object. Fig. 13 shows the time history of the object motion x_o. The object motion (blue) is controlled properly to follow the desired trajectory (red).

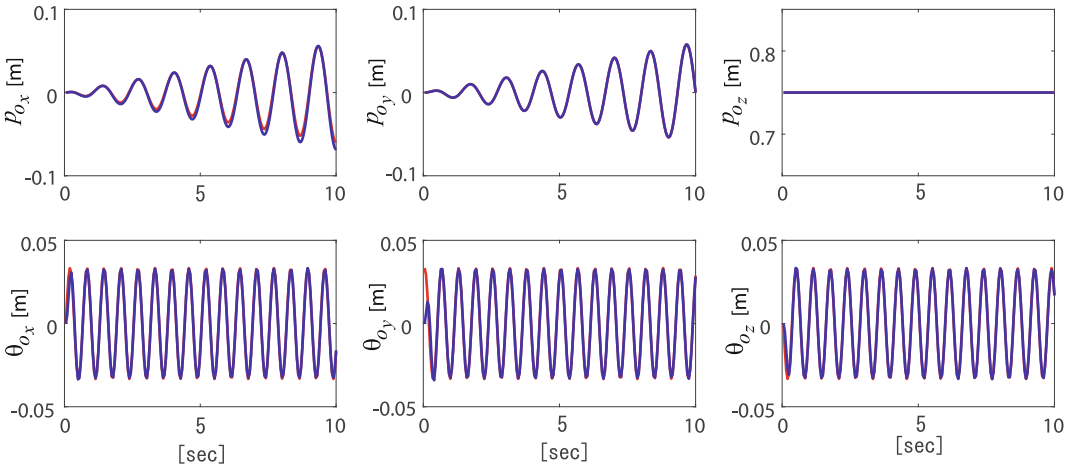


Fig. 13. Time history of object motion x_o (blue: actual, red: desired).

9.2. Optimization settings

Although the classical hand can conduct the task, a more efficient hand can possibly be designed for the given task. Below, we investigate the design of a manipulator that minimizes the total link mass while using the same number of joints $n = 10$ and fingers $C = 3$.

We design the joint position q_i, type (revolutional/translational), and movement direction (v_i or ω_i) as well as their tree-type architecture C_h. The position and orientation of the first joint (wrist) are fixed at their original values. The contact points and positions of the tip joints are also fixed for object grasping. This leaves six free joint positions and nine free joint orientations. Fig. 14 shows the possible joint region for the optimization.

The mass is calculated from the link length l_{i^c} as before, that is, $M = \sum \rho \pi r^2 l_{i^c}$. For a revolutional joint, the link length is the distance of the joint positions q_i. As for a translational joint, the length between the joints varies during the task; therefore, we assign a fixed length $l_t = 0.3$ [m] to represent its maximum elongation.

For the GA optimization, we use the strings described in Section 6. The length (bits) of the strings is set as $n_{q_{i_x}} = n_{q_{i_y}} = n_{q_{i_z}} = 4; n_{\phi_i} = n_{\gamma_i} = 3$; and $n_{n_t} = n_{b_1} = n_{b_2} = n_{C_2} = n_{C_3} = 3$. The total size of the string s_j for each individual is 150. The adaptation process is conducted by the different string group levels as described in Section 6. Table 2 summarizes the probability of the adaptation process.

The number of the population is set as $N = 50$. Candidates of the initial population are produced from random numbers, and we select the first $N = 50$ individuals that conduct the task successfully without violating the conditions or colliding. In

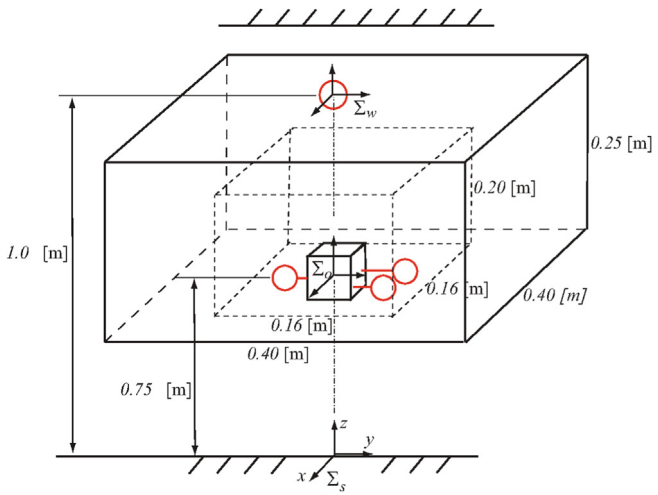


Fig. 14. Possible joint region for optimization.

Table 2
GA adaptation probability (grasping/manipulation optimization).

Adaptation	String level	Values
Reproduction	–	Roulette Wheel Selection
	Number of elites	$N_l = 1$
Crossover	Higher level	$p_{c_H} = 0$
	Intermediate level	$p_{c_I} = 0.18$
	Lower level	$p_{c_L} = 0.72$
Mutation	Higher level	$p_{m_H} = 0.0033$
	Intermediate level	$p_{m_I} = 0.0033$
	Lower level	$p_{m_L} = 0.0165$

this optimization, to save the computation time and avoid being stuck in local minima, we use a multistep optimization strategy. For a rough search, we first conduct four sets of optimizations up to $G = 50$ generations. Then, for a more precise search, we conduct the final optimization up to $G = 100$ generations by using the ten best individuals from the first optimization along with ten random individuals as the initial population ($N = 50$ in total).

9.3. Optimization results

Fig. 15 shows the evolution of the best individual in the first optimization. The solid blue line indicates the mass of the classical design, and we have three best individuals below the classical design. Fig. 16 shows the best configuration for each optimization, and the corresponding chain matrix is

$$C_{h_1} = \left[\begin{array}{cc|c|ccccccc|c} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \end{array}\right], \quad C_{h_2} = \left[\begin{array}{cc|cc|cccc|cc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right], \tag{67}$$

$$C_{h_3} = \left[\begin{array}{cccc|cc|ccc|c} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right], \quad C_{h_4} = \left[\begin{array}{ccc|c|ccc|ccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{array}\right]. \tag{68}$$

By using the best individuals described above, we conduct the final optimization. Fig. 17 shows the evolution of the best individual and Fig. 18, the best configuration after $G = 100$ generations. The optimal total mass is $M_{opt} = 1.6082$ [kg], which represents a 32% reduction compared with the classical design. Table 3 summarizes the exponential parameters (joints 6, 9, and 10 are the fingertip joints and their positions are not the design parameters), and the chain matrix is

$$C_{h_{opt}} = \left[\begin{array}{cc|cccc|ccc|c} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]. \tag{69}$$

We note the following about the optimization result:

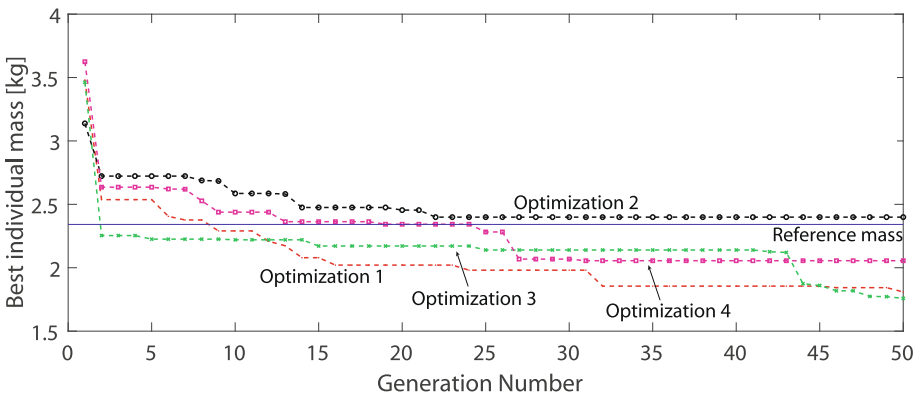


Fig. 15. Evolution of best individual (first optimization).

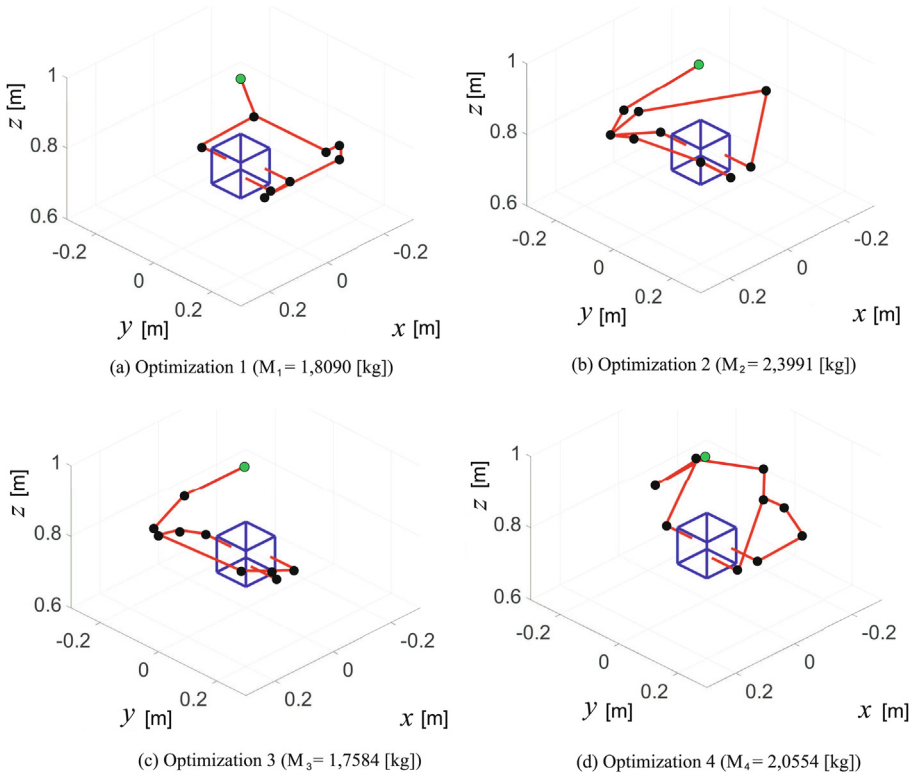


Fig. 16. Configuration of best individual (first optimization).

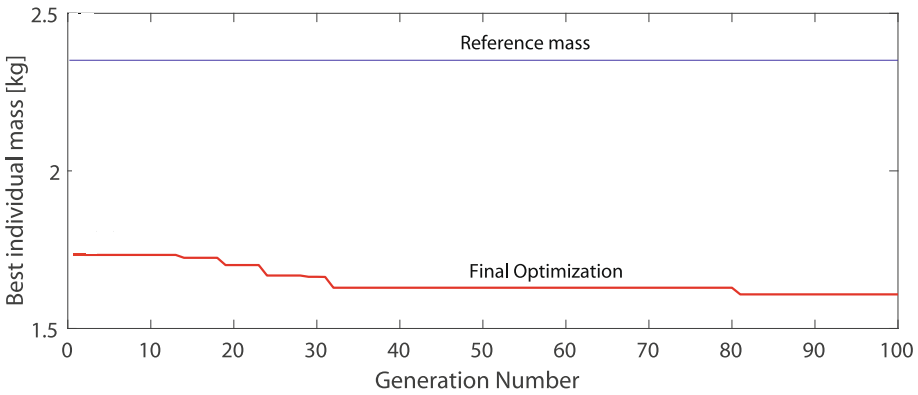
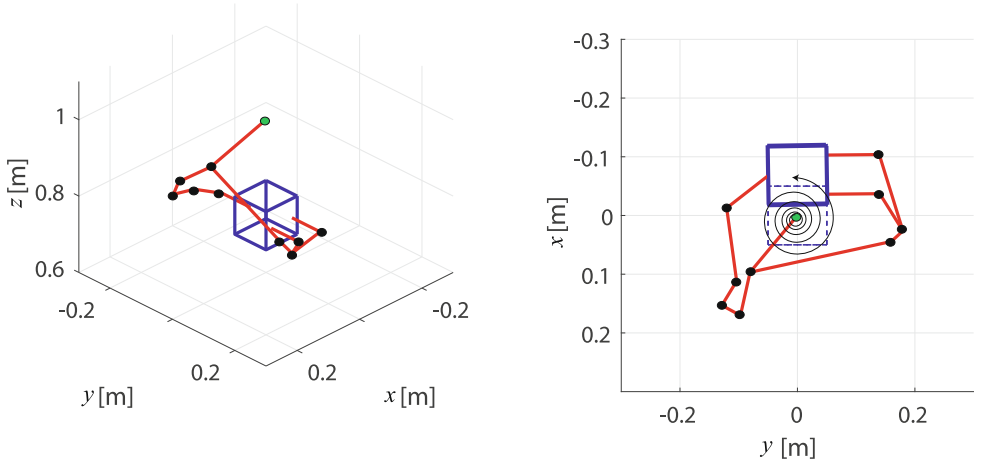


Fig. 17. Evolution of best individual (final optimization).

(1) For joint type selection, all joints are revolutional and no translational joint exists. This is reasonable because the introduction of translational joints automatically increases the mass by the fixed length of $l_t = 0.3$ irrespective of the joint position selection.
(2) For mass reduction, the total length of the link should be shortened while avoiding collisions. For this purpose, the design introduces an elbow (first joint) and locates the wrist or a branching joint (second joint) closer to the object in the lower left part of its original position. The two right-hand-side fingers branch at the end of the tree for the same reason.
(3) For manipulation purposes, the design distributes the joints equally to the left (thumb) and right fingers (index and middle) to accommodate the spiral motion. As a result, the thumb has more DOFs than the other fingers.



(a) Initial position (3D view)

(b) Final position (upper view)

Fig. 18. Configuration of best individual (final optimization).

Table 3
Exponential parameters of optimal design.

	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7	Joint 8	Joint 9	Joint 10
Joint position q_i	0.10 −0.08 0.88	0.16 −0.12 0.86	0.15 −0.15 0.80	0.11 −0.12 0.82	(0.00) (−0.14) (0.75)	0.11 0.16 0.79	0.10 0.18 0.76	(0.03) (0.14) (0.75)	(−0.03) (0.14) (0.75)
Joint type t_i (0 : rev., 1 : trans.)	0	0	0	0	0	0	0	0	0
Joint direction $\{v_i, \omega_i\}$	0.09 0.06 0.99	0.64 0.77 0.00	1.00 0.00 0.00	−0.02 −0.17 0.98	−0.02 0.11 0.99	−0.45 0.00 0.89	0.92 0.27 0.29	−0.29 0.57 0.77	0.40 0.92 0.00

10. Conclusions

This study proposes an approach to realize the simultaneous design optimization of the geometry and the topology of tree-type robotic systems based on exponential coordinates with a chain matrix expression. The previously obtained kinematics and dynamics of tree-type systems were extended to floating base systems and closed-chain systems. The exponential parameters were used to define the system geometry, and the chain matrix was used to define the system topology. It was demonstrated that these parameters can be efficiently coded into strings for GA optimization. The general closed-form formulas allowed us to automatically obtain the kinematics and dynamics and a controller of any type from a GA string. The cost evaluation process by using kinematics and dynamics control was also illustrated. The optimization was successfully demonstrated through numerical examples: a robotic platform was optimized to maximize the workspace and a grasping/-manipulation system was optimized to minimize the total mass while avoiding the violation of the manipulability/friction cone conditions and collisions with the object.

In the above designs, we fixed the system size (number of joints and chains) for simplicity, which can also be optimized. In a future study, we will extend our optimization to include these parameters.

CRediT authorship contribution statement

Julien Amar: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing - original draft, Writing - review & editing. **Kenji Nagase:** Conceptualization, Methodology, Resources, Writing - original draft, Writing - review & editing, Supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] J.J. Craig, Introduction to Robotics: Mechanics and Control, third ed., Pearson Prentice Hall, 2005.
[2] R.M. Murray, Z. Li, S.S. Sastry, A Mathematical Introduction to Robotic Manipulation, CRC Press, 1994.
[3] M.W. Spong, S. Hutchinson, M. Vidyasagar, Robot Modeling and Control, John Wiley & Sons, 2006.
[4] B.K. Rout, R.K. Mittal, Tolerance design of robot parameters using Taguchi method, Mech. Syst. Signal Process. 20 (8) (2006) 1832–1852.
[5] W. Li, Z. Li, Y. Liu, L. Ding, J. Wang, H. Gao, Z. Deng, Semi-autonomous bilateral teleoperation of six-wheeled robot on soft terrains, Mech. Syst. Signal Process. 133 (2019) 106234.
[6] R.J. Alattas, S. Patel, T.M. Sobh, Evolutionary modular robotics: Survery and analysis, J. Intell. Rob. Syst. 95 (2019) 815–828.
[7] V. Patidar, R. Tiwari, Survey of robotic arm and parameters, in: Coimbatore (Ed.), International conference on computer communication and informatics (ICCCI), 2016, pp. 1–6..
[8] S. Saeedvand, M. Jafari, H.S. Aghdasi, J. Baltes, A comprehensive survey on humanoid robot development, Knowl. Eng. Rev. 34 (2019) E20.
[9] T. Fujikawa, K. Hirakawa, S. Okuma, T. Udagawa, S. Nakano, K. Kikuchi, Development of a small flapping robot: Motion analysis during takeoff by numerical simulation and experiment, Mech. Syst. Signal Process. 22 (6) (2008) 1304–1315.
[10] K.J. Kaliński, M. Mazur, Optimal control of 2-wheeled mobile robot at energy performance index, Mech. Syst. Signal Process. 70–71 (2016) 373–386.
[11] L. Zhou, Y. Li, S. Bai, A human-centered design optimization approach for robotic exoskeletons through biomechanical simulation, Rob. Autonom. Syst. 91 (2017) 337–347.
[12] E.J. Van Henten, D.A. Van't Slot, C.W.J. Hol, L.G. Van Willigenburg, Optimal manipulator design for a cucumber harvesting robot, Comput. Electron. Agricul. 65 (2) (2009) 247–257.
[13] J. Fu, F. Gao, Optimal design of a 3-leg 6-DOF parallel manipulator for a specific workspace, Chin. J. Mech. Eng. 29 (2016) 659–668.
[14] F.C. Park, Optimal robot design and differential geometry, J. Mech. Des. 117 (B) (1995) 87–92.
[15] R. Datta, S. Pradhan, B. Bhattacharya, Analysis and design optimization of a robotic gripper using multiobjective genetic algorithm, IEEE Trans. Syst., Man Cybern.: Syst. 46 (1) (2016) 16–26.
[16] C. Zheng, Y. Zhang, J. Li, J. Bai, X. Qin, B. Eynard, Survey on design approaches for robotic manufacturing systems in SMEs, Procedia CIRP 84 (2019) 16–21.
[17] P.I. Corke, A simple and systematic approach to assigning Denavit-Hartenberg parameters, IEEE Trans. Rob. 23 (3) (2007) 590–594.
[18] G. Gao, G. Sun, J. Na, Y. Guo, X. Wu, Structural parameter identification for 6 DOF industrial robots, Mech. Syst. Signal Process. 113 (2018) 145–155.
[19] S. Singh, E. Singla, Realization of task-based designs involving DH parameters: a modular approach, Intel. Serv. Robot. 9 (2016) 289–296.
[20] G. Zeinoun, R. Sedaghati, F. Aghili, Optimal design parameters of reconfigurable robots with lockable joints, Trans. Can. Soc. Mech. Eng. 41 (1) (2017) 23–38.
[21] A. Csiszar, A combinatorial optimization approach to the Denavit-Hartenberg parameter assignment, IEEE International Conference on Advanced Intelligent Mechatronics (AIM) 2015 (2015) 1451–1456.
[22] S.F.P. Saramago, V. Steffen Jr, Optimization of the trajectory planning of robot manipulators taking into account the dynamics of the system, Mech. Mach. Theory 33 (7) (1998) 883–894.
[23] S. Ha, S. Coros, A. Alspach, J. Kim, K. Yamane, Computational co-optimization of design parameters and motion trajectories for robotic systems, Int. J. Rob. Res. 37 (13–14) (2018) 1521–1536.
[24] A. Valverde, P. Tsiotras, Spacecraft robot kinematics using dual quaternions, Robotics 7 (4) (2018) 64.
[25] D.R. Isenberg, Y.P. Kakad, Quaternion based computed-torque and feed-forward tracking controllers for a space robot, in: 2010 42nd Southeastern Symposium on System Theory (SSST), 2010, pp. 232–236.
[26] Y. Li, X. Yang, H. Wu, B. Chen, Optimal design of a six-axis vibration isolator via Stewart platform by using homogeneous Jacobian matrix formulation based on dual quaternions, J. Mech. Sci. Technol. 32 (2018) 11–19.
[27] R. Chhabra, M.R. Emami, A generalized exponential formula for forward and differential kinematics of open-chain multi-body systems, Mech. Mach. Theory 73 (2014) 61–75.
[28] C. He, S. Wang, Y. Xing, X. Wang, Kinematics analysis of the coupled tendon-driven robot based on the product-of-exponentials formula, Mech. Mach. Theory 60 (2013) 90–111.
[29] B. Yin, Z. Liang, X. Dai, J. Mo, S. Wang, Task-oriented configuration optimization of a lattice distortable reconfigurable robot, Proc. Inst. Mech. Eng., Part C: J. Mech. Eng. Sci. 230 (9) (2016) 1532–1543.
[30] J. Amar, K. Nagase, A unified framework for dynamics and control of tree-type systems using exponential coordinates, Mech. Syst. Signal Process. 131 (2019) 446–468.
[31] S.V. Shah, S.K. Saha, J.K. Dutt, Dynamics of Tree-Type Robotics Systems, vol. 62 of Intelligent Systems, Control and Automation: Science and Engineering, Springer, 2013..
[32] R. Motro, Tensegrity: Structural Systems for the Future, Kogan Page Science, 2003.
[33] R.E. Skelton, M.C. de Oliveira, Tensegrity Systems, Springer, 2009.
[34] J.Y. Zhang, M. Ohsaki, Tensegrity Structures, Form, Stability, and Symmetry, vol. 6 of Mathematics for Industry, Springer, 2015..
[35] K. Nagase, R.E. Skelton, Double-Helix tensegrity structures, AIAA J. 53 (4) (2015) 847–862.
[36] K. Kikuchi, K. Sakaguchi, T. Sudo, N. Bushida, Y. Chiba, Y. Asai, A study on a wheel-based stair-climbing robot with a hopping mechanism, Mech. Syst. Signal Process. 22 (6) (2008) 1316–1326.
[37] Á. Odry, R. Fullér, I.J. Rudas, P. Odry, Kalman filter for mobile-robot attitude estimation: Novel optimized and adaptive solutions, Mech. Syst. Signal Process. 110 (2018) 569–589.
[38] K. Cai, Y. Tian, X. Liu, S. Fatikow, F. Wang, L. Cui, D. Zhang, B. Shirinzadeh, Modeling and controller design of a 6-DOF precision positioning system, Mech. Syst. Signal Process. 104 (2018) 536–555.
[39] J. Deng, W. Chen, Y. Wang, S. Zhang, Y. Liu, Modeling and experimental evaluations of a four-legged stepper rotary precision piezoelectric stage, Mech. Syst. Signal Process. 132 (2019) 153–167.
[40] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, 1989.
[41] C.-H. Liu, C.-H. Chiu, Optimal design of a soft robotic gripper with high mechanical advantage for grasping irregular objects, IEEE International Conference on Robotics and Automation (ICRA) 2017 (2017) 2846–2851.
[42] E. Bjørlykhaug, O. Egeland, Mechanical design optimization of a 6DOF serial manipulator using genetic algorithm, IEEE Access 6 (2018) 59087–59095.
[43] B. Paden, S. Sastry, Optimal kinematic design of 6R manipulators, Int. J. Rob. Res. 7 (2) (1988) 43–61.
[44] F. Pierrot, V. Nabat, O. Company, S. Krut, P. Poignet, Optimal design of a 4-DOF parallel manipulator: From academia to industry, IEEE Trans. Rob. 25 (2) (2009) 213–224.
[45] D. Chablat, S. Venkateswaran, F. Boyer, Mechanical design optimization of a piping inspection robot, Procedia CIRP 70 (2018) 307–312.
[46] S. Rajappa, M. Ryll, H.H. Bülthoff, A. Franchi, Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers, IEEE International Conference on Robotics and Automation (ICRA) 2015 (2015) 4006–4013.
[47] S. Hwang, H. Kim, Y. Choi, K. Shin, C. Han, Design optimization method for 7 DOF robot manipulator using performance indices, Int. J. Precision Eng. Manuf. 18 (2017) 293–299.
[48] K. Nagai, T. Yoshikawa, Grasping and manipulation by arm/multifingered-hand mechanisms, in: Proceedings of the 1995 IEEE International Conference on Robotics and Automation, 1995, pp. 1040–1047.
[49] J. Kim, S.-H. Lee, F.C. Park, Kinematic and dynamic modeling of spherical joints using exponential coordinates, Proc. Inst. Mech. Eng., Part C: J. Mech. Eng. Sci. 228 (10) (2014) 1777–1785.

[50] T. Yoshikawa, Multifingered robot hands: Control for grasping and manipulation, Annu. Rev. Control 34 (2) (2010) 199–208.
[51] D. Corus, P.S. Oliveto, Standard steady state genetic algorithms can hillclimb faster than mutation-only evolutionary algorithms, IEEE Trans. Evol. Comput. 22 (5) (2018) 720–732.
[52] S.H. Chung, H.K. Chan, A two-level genetic algorithm to determine production frequencies for economic lot scheduling problem, IEEE Trans. Industr. Electron. 59 (1) (2012) 611–619.
[53] B. Dasgupta, T.S. Mruthyunjaya, A Newton-Euler formulation for the inverse dynamics of the Stewart platform manipulator, Mech. Machine Theory 33 (8) (1998) 1135–1152.
[54] O. Masory, J. Wang, Workspace evaluation of Stewart platforms, Adv. Rob. 9 (4) (1994) 443–461.