SCIENCE CHINA
Technological Sciences

**Confined spaces path following for cable-driven snake robots with prediction lookup and interpolation algorithms**

Lei TANG, LiMin ZHU, XiangYang ZHU and GuoYing GU

Citation: SCIENCE CHINA Technological Sciences **63**, 255 (2020); doi: 10.1007/s11431-019-1440-2

View online: http://engine.scichina.com/doi/10.1007/s11431-019-1440-2

View Table of Contents: http://engine.scichina.com/publisher/scp/journal/SCTS/63/2

Published by the Science China Press

SCIENCE CHINA
Technological Sciences

•Article•

February 2020 Vol.63 No.2: 255–264
https://doi.org/10.1007/s11431-019-1440-2

# Confined spaces path following for cable-driven snake robots with prediction lookup and interpolation algorithms

TANG Lei[1,2], ZHU LiMin[1,2], ZHU XiangYang[1,2] & GU GuoYing[1,2*]

[1] *Robotics Institute, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China;*
[2] *State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai 200240, China*

Received May 22, 2019; accepted September 2, 2019; published online December 11, 2019

While cable-driven snake robots are promising in exploring confined spaces, their hyper-redundancy makes the collision-free motion planning difficult. In this paper, by combining the prediction lookup and interpolation algorithms, we present a new path following method for cable-driven snake robots to high-efficiently slither into complex terrains along a desired path. In our method, we first discretize the desired path into points, and develop the prediction lookup algorithm to efficiently find the points matched with joints of the robot. According to geometric relations between the prediction lookup results and link length of the robot, we develop the interpolation algorithm to reduce the tracking errors caused by the discretization. Finally, simulations and experiments of inspections in two confined spaces including the obstacle array and pipe tank system are performed on our custom-built 25 degree of freedoms (DOFs) cable-driven snake robot. The results demonstrate that the presented method can successfully navigate our snake robot into confined spaces with high computational efficiency and good accuracy, which well verifies effectiveness of our development.

**cable-driven snake robot, confined spaces, path following, prediction lookup, interpolation**

**Citation:** Tang L, Zhu L M, Zhu X Y, et al. Confined spaces path following for cable-driven snake robots with prediction lookup and interpolation algorithms. Sci China Tech Sci, 2020, 63: 255–264, https://doi.org/10.1007/s11431-019-1440-2

## 1 Introduction

Developments of cable-driven snake robots have surged in recent years both in academia and industry for their potential applications in exploring confined spaces such as intricate industry devices [1–5] and minimally invasive surgery [6,7]. With the increase of DOFs, collision-free motion planning of cable-driven snake robots becomes very difficult because of the hyper-redundancy [8,9], nonlinear and multi-lever relationships between the motors, cables, joints and end effector [10–14], especially in confined spaces.

Many efforts have been made to address this problem, which can be roughly classified into four types: Jacobian based methods [15–18], geometric heuristic methods [19–21], curve based methods [22–25] and path following methods [26–29]. Jacobian based methods get their solutions based on the kinematics model of robots. For instance, gradient projection and extended techniques are commonly used Jacobian based methods for obstacle avoidance. However, such methods suffer from cumbersome computation and local minima. A more intuitive way is to use the geometric heuristic methods. These methods view the robot links and joints as geometric lines and points, meanwhile some heuristic algorithms such as cyclic coordinate descent [19] and forward and backward reaching algorithms [20] are used to find suitable configurations of robots. However, with the increase of DOFs and obstacles, the dimensions of motion planning become very high, which exasperates the computational efficiency. Curve based methods are developed to decrease the dimensions of motion planning of snake robots.

*Corresponding author (email: guguoying@sjtu.edu.cn)

The curves employed by these methods have much fewer parameters than the DOFs of robots they represented [26]. However, in confined spaces, if the body of robot deviates the path of its end effector, there are likely that the robot gets stuck on obstacles [30]. Path following methods are popular for navigating the robot into confined spaces. They navigate the robot with its head and body keeping tracking of a desired path. For planar line and B-spline paths, Conkur [26] investigated how to follow them by hyper-redundant robots. Alternatively, Tanaka et al. [30] developed control methods for path following motion for wheeled snake robots. However, they were mainly applicable to the planar motion, and could not be used for cable-driven snake robots. For three dimensional snake robots, Palmer et al. [29] adopted an optimization approach to navigate the snake robot to follow desired trajectory, which made the motion of the robot not smooth. Recently, Andersson [31] approximated snake robots to continuous curve using the bisection search. There are only simulation verifications, and the efficiency of bisection search is affected by initial values. To our knowledge, there is rare study on efficiently navigating cable-driven snake robots into three dimensional confined spaces with experimental validation.

In this paper, we develop a computational efficient path following method combining the prediction lookup and interpolation algorithms for cable-driven snake robots to explore the confined spaces with experimental validation. The developed method contains three steps: path generation, joint position search and inverse resolution. Firstly, polynomial splines are used to generate the desired path, which negotiates obstacles in confined spaces. Secondly, the desired path is discretized into points, and the robot moves along the path by matching joints with the discretized points. The prediction lookup algorithm searches joint positions with joint information of previous moments, which leads to high computational efficiency. The discretization loses some points of the path, which may give rise to tracking errors. Thus, according to geometric relationships between the prediction lookup results and link length, the interpolation algorithm acquires accurate joint positions. Finally, with the known joint positions, joint angles and motor control inputs are calculated by kinematics of the robot. Simulations and experiments are performed on a 25 DOFs cable-driven snake robot platform. The simulation and experimental results show that the computation time is greatly reduced while maintaining high accuracy, which validates effectiveness of the presented method. Major contribution of this paper is that a high-efficiency and high-accuracy confined spaces path following method for cable-driven snake robots with prediction lookup and interpolation algorithms is developed with simulation and experimental validations.

The rest of the paper is organized as follows. Sect. 2 introduces the path following problem and the 25 DOFs cable-driven snake robot. Details of the developed method including the prediction lookup and interpolation algorithms are presented in Sect. 3. Inspection simulations of two confined spaces, namely a board obstacle array and a pipe tank system, are presented in Sect. 4. In Sect. 5, experiments of the two simulations are conducted on the 25 DOFs cable-driven snake robot platform to validate the effectiveness and efficiency of the presented method. Sect. 6 concludes this paper.

## 2 Problem statement of path following

### 2.1 Path following

As shown in Figure 1, path following means that snake robots slither into confined spaces along a desired path and conduct inspection or maintenance in the place of interest ($\mathbf{P}_m$). The desired path is collision-free, and generated by some way-points ($\mathbf{P}_1$–$\mathbf{P}_m$) that the robot needs to pass through. $\mathbf{P}_1$–$\mathbf{P}_m$ are determined by the tasks, and $m$ is the number of way-points. In the path following process, the head of the snake robot keeps tracking of the path, meanwhile its body matches with the path.

In this sense, the path following method has three major concerns: desired path generation, determining the configuration matched with the path in each step, calculating the joint angles and motor control inputs. The desired path can be generated by curve based methods, and the joint angles and motor control inputs can be calculated according to the kinematics. How to match the snake arm with the desired path at each step efficiently and accurately is the key issue in path following, which is the main concern of this paper. The essence of this problem is to efficiently find points accurately matched with the joints under the rigid link length constraint. Therefore, the prediction lookup and interpolation algorithms are put forward to solve this problem efficiently.

### 2.2 System description

A cable-driven snake robot designed to fulfill confined spaces exploration is introduced briefly here. In the development, the robot consists of end effector, snake arm, actuation system and linear slider, which are shown in Figure 1. The snake arm with 12 sections has a length of 1235 mm and a diameter of 45 mm. Each section contains a universal joint, a 70 mm rigid bar, two 4 mm disks and two 6.5 mm joint bars. To reduce weight and inertia, all the components are hollow. The snake arm is driven by 36 cables which are connected to motors by actuation system. The motors and control system are put at rear of the snake arm, which further improves efficiency of the arm. A linear slider is utilized to provide feed motion for the snake robot. For detailed information, the readers may refer to refs. [8,15].

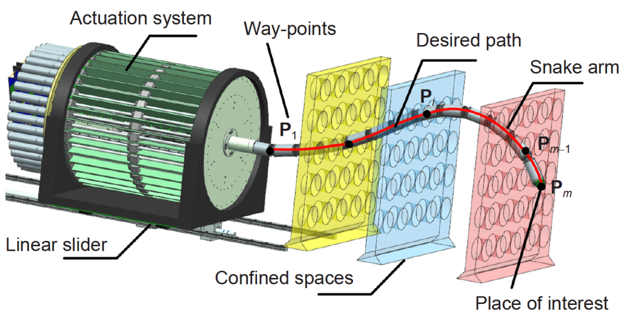


**Figure 1** (Color online) Schematic of cable-driven snake robot and path following motion.

## 3 Path following method

The presented path following method consists of path generation, prediction lookup and interpolation algorithms and inverse resolution, which is not restricted to the specific configuration of our 25 DOFs cable-driven snake robot. Without losing generality, our method can also be applied for other hyper-redundant cable-driven snake robots or continuum robots. The difference may lie in the inverse resolution according to the kinematics of different robots, which relies on the transformation matrix between two adjacent joints or sections.

### 3.1 Path generation

Polynomial splines are employed to generate the desired path, which are composed of piecewise $k$ order polynomials with $k$–1 order derivatives at the junction points between two spline segments. Given $m$ points, $m$–1 spline segments can be determined. As shown in Figure 2, way-points $\mathbf{P}_h$ and $\mathbf{P}_{h+1}$ are employed to generate the $h^{\text{th}}$ spline segment, and the position vector $\mathbf{r}_h$ of each point on it is represented by

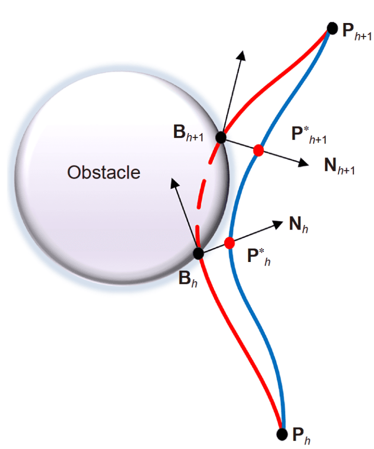


**Figure 2** (Color online) Schematic of the path generation and obstacle avoidance.

$$\mathbf{r}_h(u)=\sum_{j=1}^{k+1}\mathbf{a}_{hj}u^{j-1},\quad 0\le u\le u_{h+1},\tag{1}$$

where $h$=1,…, $m$–1, is the index of spline segment. The curve parameter is $u$, and $u_{h+1}$ is the parameter value at the end of $h^{\text{th}}$ segment. The coefficients of $h^{\text{th}}$ segment are $\boldsymbol{a}_{hj}$, which is determined by the boundary conditions such as the position and first derivatives vector of the initial and end points. The index $j$ is determined by the order of splines. For example, for the cubic splines, $k$=3, $j$=1,…, 4, and $\boldsymbol{a}_{hj}$ can be denoted by eq. (2). $\mathbf{P}_h'$ is the tangential vector at $\mathbf{P}_h$.

$$\begin{aligned}
&\mathbf{a}_{h1}=\mathbf{P}_h,\\
&\mathbf{a}_{h2}=\mathbf{P}'_h,\\
&\mathbf{a}_{h3}=3(\mathbf{P}_{h+1}-\mathbf{P}_h)/u_{h+1}^2-(2\mathbf{P}'_h+\mathbf{P}'_{h+1})/u_{h+1},\\
&\mathbf{a}_{h4}=2(\mathbf{P}_h-\mathbf{P}_{h+1})/u_{h+1}^3+(\mathbf{P}'_h+\mathbf{P}'_{h+1})/u_{h+1}^2.
\end{aligned}\tag{2}$$

The cubic splines have the advantages of passing through way-points, smoothness with continuous second order derivative. According to the condition of second order continuous at the junction points $(\mathbf{r}_h^{k-1}(u_{h+1})=\mathbf{r}_{h+1}^{k-1}(0))$, curve parameter $u$ is normalized with $u_{h+1}$, and $\lambda$ ($\lambda=u/u_{h+1}$) is set as the new curve parameter. The position vectors $r_h$ can be rearranged as eq. (3).

Path segments between way-points can be generated according to eqs. (1)–(3). However, in confined spaces, the path may collide with obstacles. An adjustment approach is presented for the collision-free feasible path.

$$\begin{aligned}
&\mathbf{r}_h(\lambda)=\mathbf{P}_h(t)\mathbf{f}_h(\lambda)^{\mathrm{T}},\quad 0\le\lambda\le 1,\\
&\mathbf{P}_h(t)=[\mathbf{P}_h\ \ \mathbf{P}_{h+1}\ \ \mathbf{P}'_h\ \ \mathbf{P}'_{h+1}],\\
&\mathbf{f}_h(\lambda)=[f_{1h}\ \ f_{2h}\ \ f_{3h}\ \ f_{4h}],\\
&f_{1h}=2\lambda^3-3\lambda^2+1,\\
&f_{2h}=\lambda^2(-2\lambda+3),\\
&f_{3h}=\lambda(\lambda-1)^2,\\
&f_{4h}=\lambda^2(\lambda-1).
\end{aligned}\tag{3}$$

As shown in Figure 2, obstacles can be modeled as a number of spheres or cylinders. The distances among the obstacles and the planned spline segments are calculated. When the $h^{\text{th}}$ spline segment collides with obstacles, the start and end points of the collision are denoted as $\mathbf{B}_h$ and $\mathbf{B}_{h+1}$, with their normal directions $\boldsymbol{N}_h$ and $\boldsymbol{N}_{h+1}$, respectively. To avoid collision, $\mathbf{B}_h$ and $\mathbf{B}_{h+1}$ are translated along their normal direction by a small distance. The small distance is determined as follows. The distances from the points between $\mathbf{B}_h$ and $\mathbf{B}_{h+1}$ to the surface of the collided obstacle are calculated, and the maximum distance is obtained and recorded as $s_{\max}$. The small distance that $\mathbf{B}_h$ and $\mathbf{B}_{h+1}$ need to be translated is equal to $s_{\max}$ plus the feeding displacement $s$. The points after translation ($\mathbf{P}^*_h$ and $\mathbf{P}^*_{h+1}$) are utilized as new way-points for spline path generation. The above process is repeated until there is no collision happens.

Remark: It is worthy of mentioning that commonly used curves such as polynomial, Bezier, and B-spline curves, can be used for the path generation. Without losing generality, we employ the commonly used polynomial splines in this work to verify the effectiveness of the developed path following method.

### 3.2 Prediction lookup algorithm

The joints and links of cable-driven snake robot should be matched with the desired path. This problem can be formulated as eq. (4). As shown in Figure 3, with the position of the $i^{\text{th}}$ joint $\mathbf{C}_i$ and length of $i^{\text{th}}$ link $L_i$, the position of the next joint $\mathbf{C}_{i+1}$ should be found on the desired path $\mathbf{r}(\lambda)$.

$$\begin{aligned} &\|\mathbf{C}_{i+1}(t)-\mathbf{C}_i(t)\| = L_i, \\ &\text{s.t.} \quad \mathbf{C}_i(t) \in \mathbf{r}(\lambda). \end{aligned} \tag{4}$$

An efficient way for finding the joint positions is to discretize the planed curve $\mathbf{r}(\lambda)$ to a set of points $\mathbf{M}(In)$ with same distance $sd$ between two adjacent points, and search them numerically. $\mathbf{M}$ notifies the set of points and $In$ is the index of the points. A prediction lookup algorithm is developed to improve the efficiency of the search process. This algorithm predicts and searches joint positions of the next moment $\mathbf{C}_i(t+1)$ with the joint information at previous moment $\mathbf{C}_i(t)$ and feed motion of the base, which achieves high efficiency. The algorithm is presented in detail as below.

The inputs of the algorithm are the points on the desired path $\mathbf{M}(In)$, base feed displacement $s$, link lengths of each section $L_i$, the initial positon of the first joint $\mathbf{C}_1(0)$, which coincides with the start point of $\mathbf{r}(\lambda)$. The Outputs of the algorithm are indexes of the points coincided with the joints.

The index of the $In^{\text{th}}$ discrete point coincided with the $i^{\text{th}}$ joint at $t$ moment is defined as $In(i, t)$, and its incremental is $\Delta In(i, t)= In(i+1, t)–In(i, t)$. At the moment of $t$=0, the first joint $\mathbf{C}_1(0)$ is the same as $\mathbf{M}(1)$, ($In(1,0)=1$), and the index of the point coincided with other joints $In(i, 0)$ are determined by the link length. Generally, assume that $In(i, t)$, the index of the point of $i^{\text{th}}$ joint at the moment of $t$, is known, $In(i, t+1)$, the index of the points coincided with the joints at the next moment, can be predicted as follows.

At the moment of $t$+1, the linear slider feeds by a small displacement $s$. The snake arm moves along the path accordingly, which makes the joint move by a small displacement. As shown in Figure 3, with small displacement and invariant link length, the difference between the point indexes of adjacent joints at the moments of $t$ and $t$+1 are almost equal. Thus, the relationship between the point indexes of joints with such movement can be denoted by

$$In(i+1,t+1)-In(i,t+1) \approx In(i+1,t)-In(i,t). \tag{5}$$

At the moment of $t$=0, the estimation of $\Delta In(i, t)$ can be denoted by eq. (6). As all the initial joint angles are zero, the point of the first joint with feeding displacement $s$ can be determined by eq. (7).

$$\Delta In(i,0) = L_i / sd, \tag{6}$$

$$In(0,t+1) = In(0,t) + s / sd. \tag{7}$$

All the indexes of the point of joints at the moment of $t(In(i, t))$ are known, that is, $\Delta In(i, t)$ is known. Thus, $In(i+1, t+1)$ can be predicted by eq. (8) according to eq. (5).

$$In(i+1,t+1) = In(i,t+1) + \Delta In(i,t). \tag{8}$$

With the prediction, the candidate points of actual joint position ($\mathbf{C}_{i+1}(t+1)$) can be determined as $\mathbf{M}(In(i+1, t+1))$, $\mathbf{M}(In(i+1, t+1)–1)$ and $\mathbf{M}(In(i+1, t+1)+1)$. The distances among the candidate points and $\mathbf{C}_i(t+1)$ are notified as $L_m$, $L_l$, and $L_r$, which is shown in eq. (9). The difference among $L_m$, $L_l$, $L_r$ and the link length $L_i$ are defined as $D_1$, $D_2$, and $D_3$, which are denoted by eq. (10).

$$\begin{aligned} &L_m = \| \mathbf{C}_i(t+1)\mathbf{M}(In(i+1,t+1)) \|, \\ &L_l = \| \mathbf{C}_i(t+1)\mathbf{M}(In(i+1,t+1)-1) \|, \\ &L_r = \| \mathbf{C}_i(t+1)\mathbf{M}(In(i+1,t+1)+1) \|, \end{aligned} \tag{9}$$

$$D_1 = \|L_m - L_i\|, D_2 = \|L_l - L_i\|, D_3 = \|L_r - L_i\|. \tag{10}$$

The values of $D_1$, $D_2$, and $D_3$ are compared, and the minimal of them is found. If $D_1$ is the minimal, the first candidate point $\mathbf{M}(In(i+1, t+1))$ is the position of the joint $\mathbf{C}_{i+1}(t+1)$. If all the joint positions are found, the process is terminated, else the $\Delta In(i, t+1)$ is recorded and the process of eqs. (6)–(9) is repeated to find the position of the next joint $\mathbf{C}_{i+2}(t+1)$; If $D_2$ is the minimal, the points $\mathbf{M}(In(i+1, t+1)–1)$, $\mathbf{M}(In(i+1, t+1)–2)$ and $\mathbf{M}(In(i+1, t+1))$ are set as the new candidates, and repeat eqs. (8) and (9) to find $\mathbf{C}_{i+1}(t+1)$; If $D_3$ is the minimal, the points $\mathbf{M}(In(i+1, t+1)+1)$, $\mathbf{M}(In(i+1, t+1))$ and $\mathbf{M}(In(i+1, t+1)+2)$ are set as the new candidates, and repeat eqs. (8) and (9) to find $\mathbf{C}_{i+1}(t+1)$. The pseudo-code of the prediction lookup algorithm is presented in Figure 4.

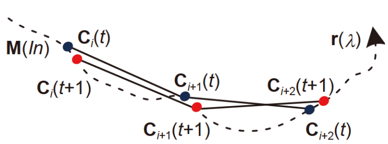


**Figure 3** (Color online) Illustration of the prediction lookup algorithm.

### 3.3 Interpolation algorithm

As shown in Figure 5, $\mathbf{C}_i$ is the position of $i^{\text{th}}$ joint. $\mathbf{M}_{In}$ and $\mathbf{M}_{In+1}$ are the candidate points for $\mathbf{C}_{i+1}$, and $\mathbf{M}_{In}$ is the best solution for the link length constraint found by the above lookup algorithm. They are all known points, which makes up the plane $\mathbf{C}_i\mathbf{M}_{In}\mathbf{M}_{In+1}$. The circle with the center of $\mathbf{C}_i$ and a radius of $L_i$ intersects with $\mathbf{C}_i\mathbf{M}_{In}$, $\mathbf{C}_i\mathbf{M}_{In+1}$ and the path at

```
For i=2: n+1 % n is the number of sections.
  In(i+1, t+1)=In(i, t+1)+ΔIn(i, t)
  Flag=0; % if flag=1 means the position of the joint in this section is found.
  While ~flag
      C_{i+1}(t+1)_1= M (In (i+1, t+1)); C_{i+1}(t+1)_2= M (In(i+1, t+1)-1);
      C_{i+1}(t+1)_3= M (In (i+1, t+1) +1)
      L_1=|| C_i (t+1) C_{i+1}(t+1)_1||; L_2=|| C_i (t+1) C_{i+1}(t+1)_2 ||; L_3=|| C_i (t+1) C_{i+1}(t+1)_3 ||
      D_1=|| L_1- L_i ||; D_2=|| L_2- L_i ||; and D_3=|| L_3- L_i ||;
      MinD=min (D_1, D_2, D_3);
     % find the point most satisfying the constraint of the length of the section.
      Switch MinD
      Case 1
       C_{i+1}(t+1) = M (In(i+1, t+1));
       ΔIn (i, t+1)= In(i+1, t+1) - In(i+1, t); flag=1;
     Case 2
       C_{i+1}(t+1)_1= M (In (i+1, t+1)-1); C_{i+1}(t+1)_2= M (In (i+1, t+1)-2);
       C_{i+1}(t+1)_3= M (In (i+1, t+1));
     Case 3
       C_{i+1}(t+1)_1= M (In (i+1, t+1) +1); C_{i+1}(t+1)_2= M (In (i+1, t+1));
       C_{i+1}(t+1)_3= M (In (i+1, t+1) +2);
    End
  End
End
```

**Figure 4** Pseudo-code of the prediction lookup algorithm.

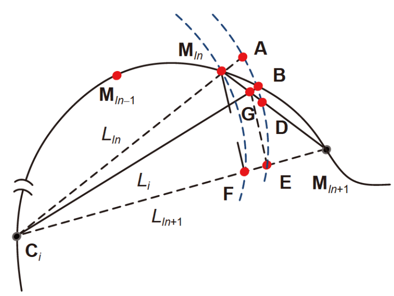


**Figure 5** (Color online) Demonstration of the interpolation algorithm.

**A**, **E** and **B**, respectively. The point **A** is on the line $\mathbf{C}_i\mathbf{M}_{In}$ which satisfies the constraint of $\|\mathbf{C}_i\mathbf{A}\| = L_i$. The length of $\mathbf{C}_i\mathbf{M}_{In}$ and $\mathbf{C}_i\mathbf{M}_{In+1}$ are $L_{In}$ and $L_{In+1}$, respectively. Thus, the real solution of eq. (4) is a little different from $\mathbf{M}_{In}$, which cannot be found because of the discretization. In this sense, the discretization may result in tracking errors. Therefore, an interpolation is developed to compensation such errors.

According to the geometric relationships as shown in Figure 5, we can see the point **F** is on the line $\mathbf{C}_i\mathbf{M}_{In+1}$ with $\|\mathbf{C}_i\mathbf{F}\| = \|\mathbf{C}_i\mathbf{M}_{In}\| = L_i$, and point **G** is on the line $\mathbf{C}_i\mathbf{B}$ with **EG** parallel to $\mathbf{FM}_{In}$. With such geometric relationships, eq. (11) can be obtained, and the position of **G** can be solved by eq. (12). Thus, **G** is the position for the next joint $\mathbf{C}_{i+1}$, which is accurate enough and has lightweight computation.

$$\frac{\mathbf{FE}}{\mathbf{FM}_{In+1}} = \frac{\mathbf{M}_{In}\mathbf{G}}{\mathbf{M}_{In}\mathbf{M}_{In+1}} = \frac{L_i - L_{In}}{L_{In+1} - L_{In}}, \tag{11}$$

$$\mathbf{G} = \mathbf{M}_{In} + \frac{L_i - L_{In}}{L_{In+1} - L_{In}}\mathbf{M}_{In}\mathbf{M}_{In+1}. \tag{12}$$

### 3.4 Inverse resolution

With all the joint positions known, the joint angles are calculated by the inverse resolution. The kinematics of the snake robot is briefly introduced as the foundation of the inverse resolution. Kinematics model between joint and task spaces is established by the product of exponentials (POE) formula. The parameters of POE are listed in Table 1. With the parameters, the transformation matrix from joint to end effector is denoted as eq. (13). The two rotational angles of $i^{\text{th}}$ section are $\alpha_i$ and $\beta_i$, respectively. More detail information can be found in refs. [8,15].

$$\mathbf{T} = e^{\hat{\xi}^1_{y1}\alpha_1} e^{\hat{\xi}^1_{x1}\beta_1} \cdots e^{\hat{\xi}^1_{yi}\alpha_i} e^{\hat{\xi}^1_{xi}\beta_i} \cdots e^{\hat{\xi}^1_{yn}\alpha_n} e^{\hat{\xi}^1_{xn}\beta_n} e^{\hat{\xi}^1_{st}}. \tag{13}$$

According to joint positions ($C_i$) and kinematics, the angle of each joint can be calculated as follows. $Z$ axis of each section is presented by eq. (14). With the kinematics transformation matrix (13), the relationship between $Z$ axis of section $i$+1 and $i$ can be denoted as eq. (15). According to eqs. (14) and (15), joint angles of each section can be denoted by eq. (16). As a conclusion, the flowchart of the presented path following method is shown in Figure 6.

$$\mathbf{Z}_i = \mathbf{C}_{i+1} - \mathbf{C}_i / \|\mathbf{C}_{i+1} - \mathbf{C}_i\| = [x_i, y_i, z_i]^{\mathrm{T}}, \tag{14}$$

$$\mathbf{Z}_i = \begin{bmatrix} \cos\alpha_i & \sin\alpha_i\sin\beta_i & \sin\alpha_i\cos\beta_i \\ 0 & \cos\beta_i & -\sin\beta_i \\ -\sin\alpha_i & \cos\alpha_i\sin\beta_i & \cos\alpha_i\cos\beta_i \end{bmatrix} \mathbf{Z}_{i-1}, \tag{15}$$

$$\begin{aligned} \beta_i &= \arcsin(-y_i), \\ \alpha_i &= \arcsin(x_i / \cos(\beta_i)). \end{aligned} \tag{16}$$

## 4 Simulation and case study

In this section, simulations of two confined spaces inspections are performed on a standard commercial desktop running an i7-4790 (3.6 GHz) processor to demonstrate the effectiveness of the presented method. The first case as shown in Figure 7(a) is an array of obstacle boards with circle holes of 100 mm diameter. The second case as shown in Figure 7(b) is the pipe tank system.

### 4.1 Simulation results

For the first case, the cable-driven snake robot should carry its end effector to the target point (Way-point 1) which is

**Table 1** Parameters of POE

| Description | Symbol | Value |
|---|---|---|
| TY in world frame | $\xi^1_{y,i}$ | $(0,1,0, -(i-1)(2h+L),0,0)^{\mathrm{T}}$ |
| TX in world frame | $\xi^1_{x,i}$ | $(1,0,0,0, (i-1)(2h+L),0)^{\mathrm{T}}$ |
| TI in world frame | $\xi^1_{st,i}$ | $(0,0,0,0,0, (n-1)(2h+L))^{\mathrm{T}}$ |

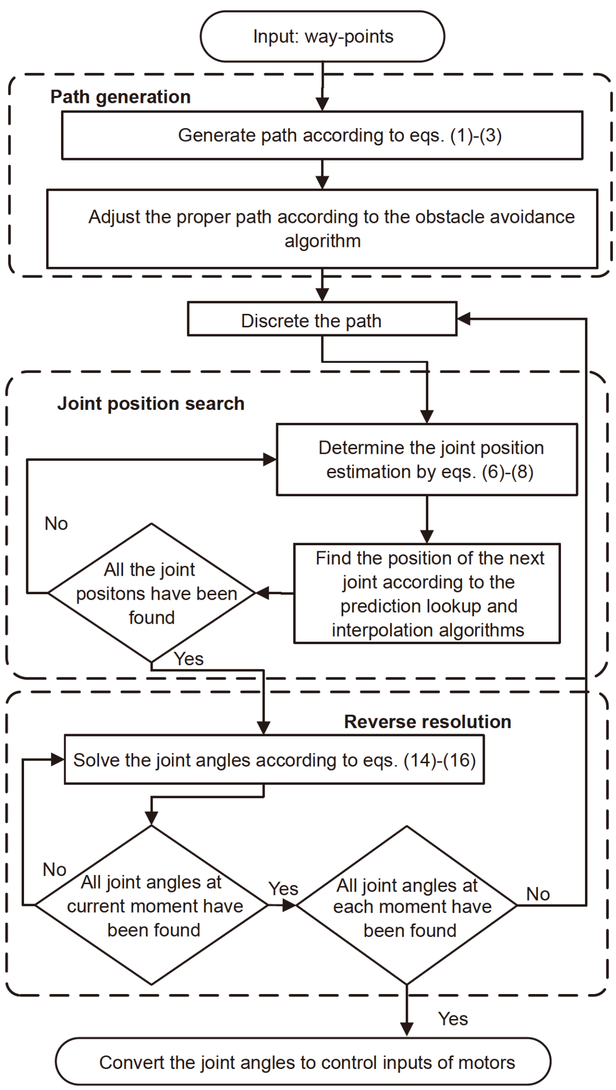


**Figure 6** Flowchart of the path following method.

behind the third board. The entrance (Way-point 3) is at the hole on the first board, and the middle point (Way-point 2) on the second board is selected for the robot to pass through. The three way-points are utilized to generate the desired path. The snake robot slithers into the obstacle array with a step of 3.5 mm. There are 312 steps in total with each step 150 ms, which is the control period of the snake robot. Figure 8 shows the images of the robot slithering into obstacle array at $t$= 0, 22, 34, 46 s, respectively.

For the second case, the entrance (Way-point 1) is at the elliptic hole of the pipe tank system, which is the unique entrance for inspection task. The snake robot is planned to inspect the corner area between the back board and pipe 3. Thus, a middle point target point (Way-point 2) between pipe 1 and 2 and a target area (Way-point 3) between pipe 3 and back board is employed to generate the desired path. The step length is also set 3.5 mm with each step 150 ms. There are 303 steps in inspection of the pipe tank system. Figure 9 shows the images of inspections at $t$= 0, 20, 32, 45 s, respectively. A video of these simulations is available in Movie S1 in Supporting Information.

### 4.2 Performance evaluation

There are mainly two concerns on the path following method developed in this paper: computation efficiency and accuracy. The pipe tank system inspection case is employed to study the computation efficiency and accuracy of the developed method. The computation time and deviations between the end effector and the desired path at each step are measured.

#### 4.2.1 Computation time

An iterative search method presented in ref. [32] searched the position of the joint positions sequentially with the discretized path. However, the iterative method directly searches the joint position according to the link length constraint as presented in eq. (4), resulting medium efficiency. The pre-

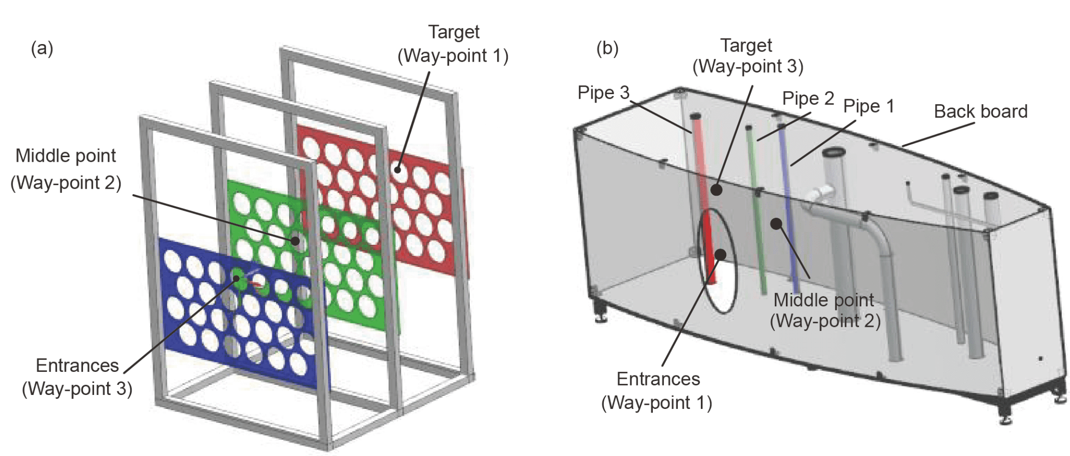


**Figure 7** (Color online) Three dimensional view of the simulated confined space. (a) Obstacle array; (b) pipe tank system.

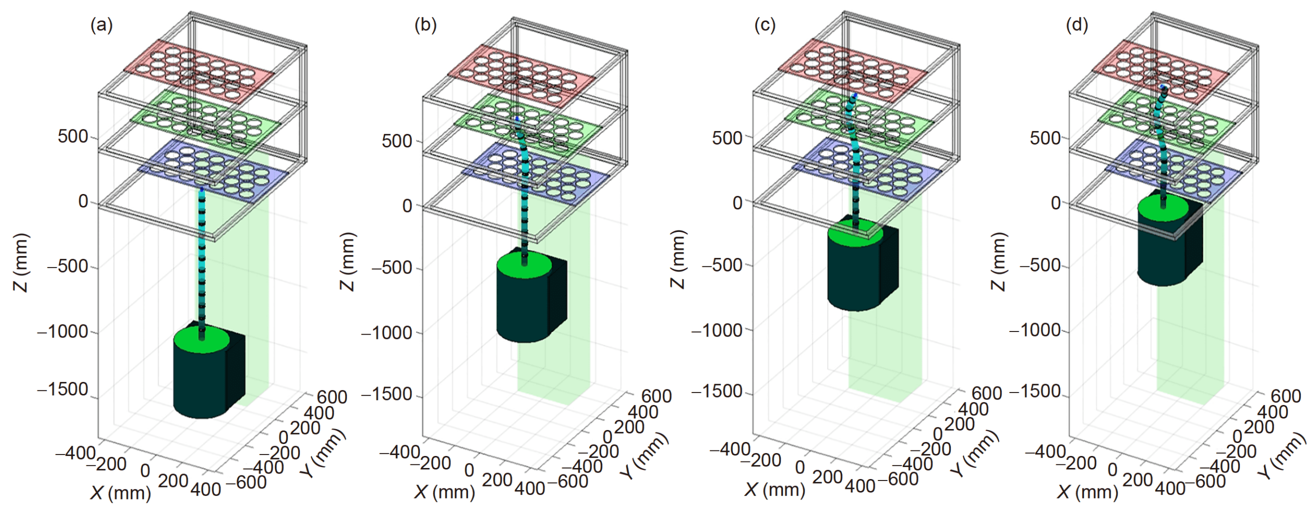


**Figure 8** (Color online) Simulation results of the inspection of the obstacle array.

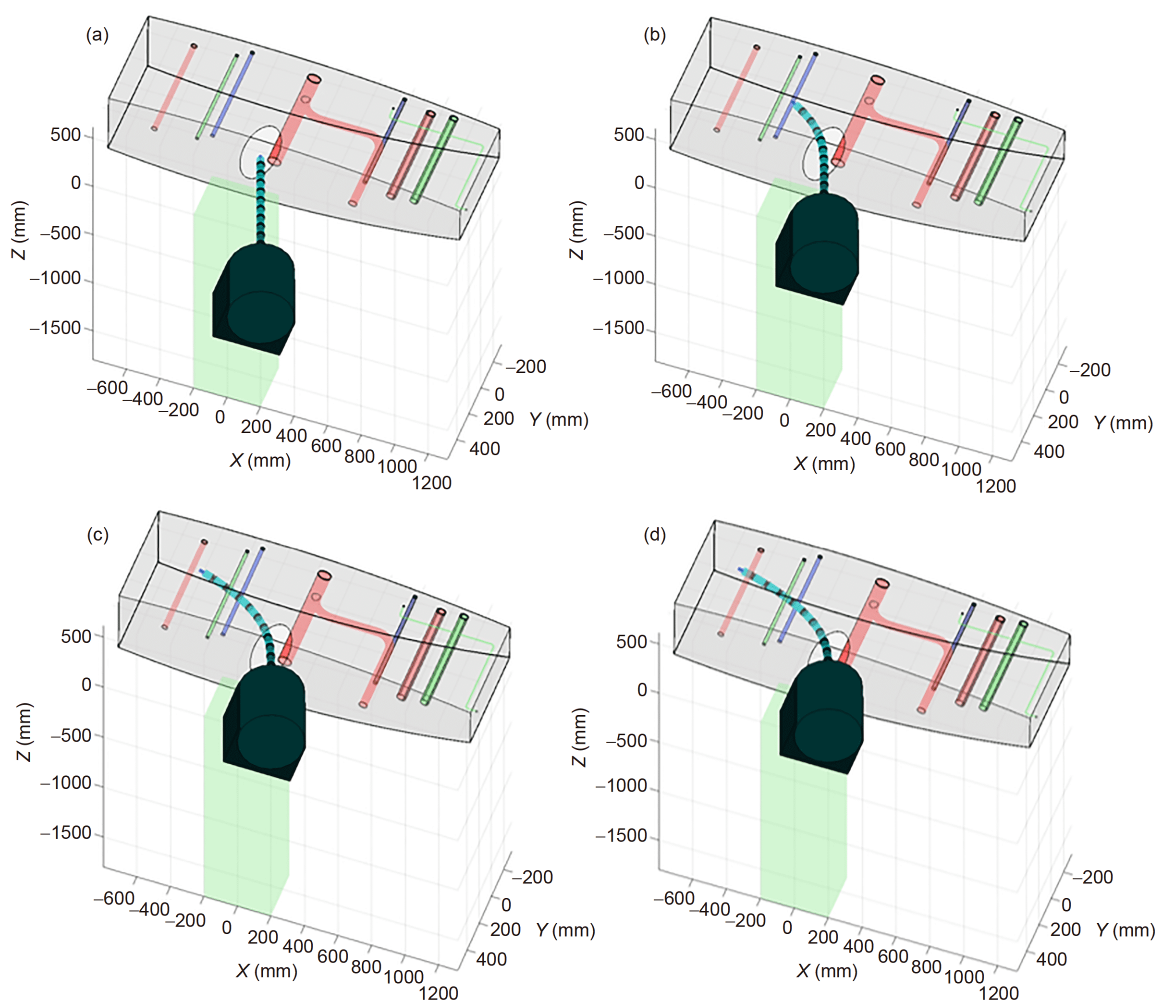


**Figure 9** (Color online) Simulation results of the inspection of the pipe tank system.

diction lookup algorithm in this paper utilizes the information of the joint positions at previous moments to predict and search the joint positions of the next moment, which is more efficient than direct search with link length constraint. The computation time of each step by the iterative search and prediction lookup algorithms are shown in Figure 10(a). The

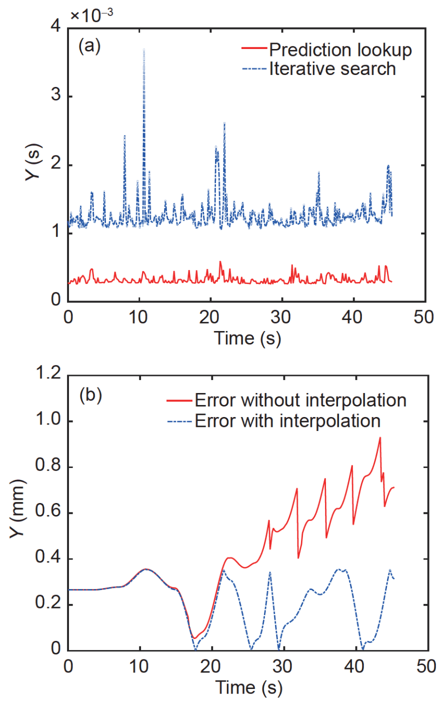


**Figure 10** (Color online) Computation efficiency and end effector accuracy. (a) Computation time of the iterative search and prediction lookup algorithms; (b) tracking errors with and without the interpolation algorithm.

average time of one step by these two algorithms are 1.29 and 0.32 ms, respectively. For 25 DOFs snake robot, the computation time spent finding the desired positions of joints by the optimization search [29], bisection search [31], iterative search [32] and prediction lookup algorithms are listed in Table 2. It can be seen that the prediction lookup algorithm improves the computation efficiency greatly.

### 4.2.2 Deviation errors

The errors between the end effector and desired path are an important accuracy index for the robot to follow the path. Thus, the actual position of the end effector is calculated by forward kinematics eq. (13) with the joint angles. Errors between the desired and actual trajectories of the end effector with and without the presented interpolation algorithm are calculated. Figure 10(b) shows the detailed deviations. The root mean square error (RMSE) and variance value error (VVE) of the end effector with and without interpolation algorithm are listed in Table 3. The results indicate that RMSE and VVE with interpolation are reduced by about 48%, and 72%, respectively, compared with the method without interpolation algorithm.

**Table 2** Computation time of one step by four algorithms for 25 DOFs snake robot

| Method | Average computation time (ms) |
|---|---|
| Optimization search | 390 |
| Bisection search | 6 |
| Iterative search | 1.29 |
| Prediction lookup | 0.32 |

**Table 3** Tracking errors with and without interpolation algorithm

| | Without interpolation | With interpolation |
|---|---|---|
| RMSE (mm) | 0.4674 | 0.2429 |
| VVE ($mm^2$) | 0.0394 | 0.0107 |

## 5 Experiment validation

In this section, experiments are conducted on the prototype platform for further verification of the presented path following method. The prototype is controlled to inspect target points in the obstacle array and pipe tank system as the same in the simulations.

### 5.1 Experimental setup

The experimental platform consists of mechanical mechanism and control system, which are shown in Figure 11(a). Way-points are given to the computer through the interface, and the desired path and control inputs of motors are generated by the presented method. The motion commands of motors and slider are delivered to control boards by Controller Area Network (CAN) bus and STM32 (F103ZET6), respectively. The control data distributed to the EPOS2 cards drives the snake arm to follow the desired path with the feed motion of the slider.

The hardware architecture and its power supply system are shown in Figure 11(b). The motors of the snake arm and slider are MAXON (EC-MAX30) and YASKAWA (SGM7G), respectively. Their control cards (EPOS2 and SGD7S) receive data from upper computer by CAN and STM32 respectively. The camera on tip of the snake arm sends the images it captured to the computer in real-time. The power supply cabinet delivers AC 220V to the computer and slider motor, and the STM32 and MAXON motor share DC 24V.

### 5.2 Experiment result

Inspection experiments of the two simulation cases are performed on the 25 DOFs cable-driven snake robot platform. Figure 12 is the static motion images of snake robot slither into obstacle array at $t$=0, 22, 34, and 46 s, respectively. For better presentation, side and front views of the motion are in the main and upper left graphs of Figure 12(a)–(d), respectively.

The motion of the inspection of pipe tank system is shown

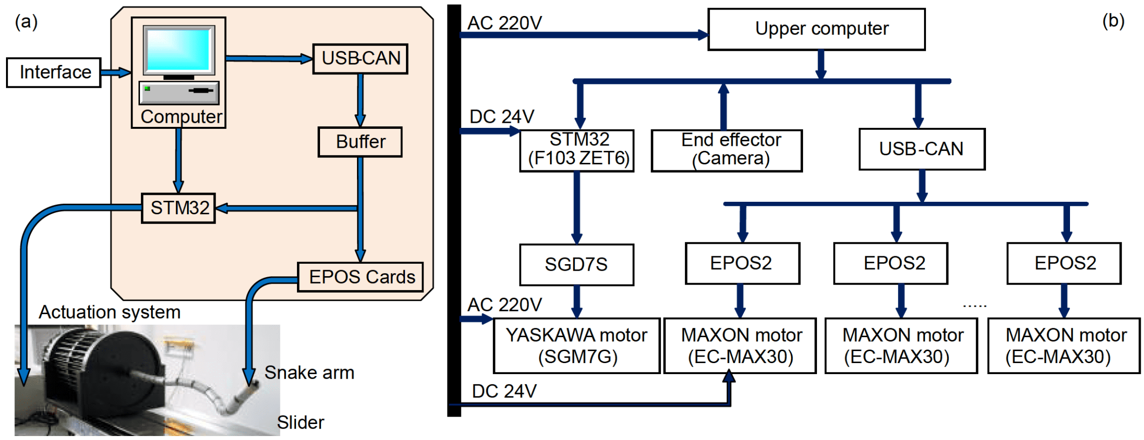


**Figure 11** (Color online) Diagram of the control system and hardware architecture of the snake robot experimental platform. (a) Control system; (b) hardware architecture.

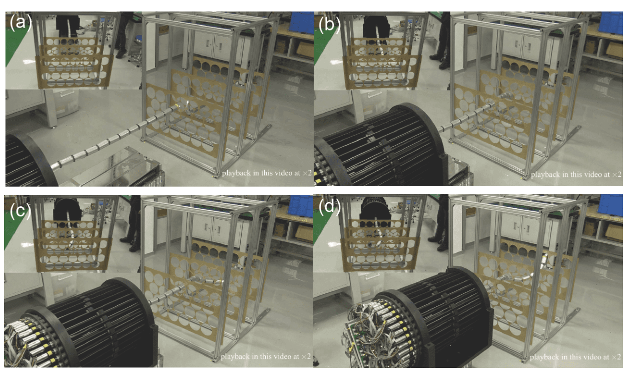


**Figure 12** (Color online) Experimental results of the inspection of the obstacle array. (a)–(d) are the images at $t$=0, 22, 34, and 46 s, respectively.

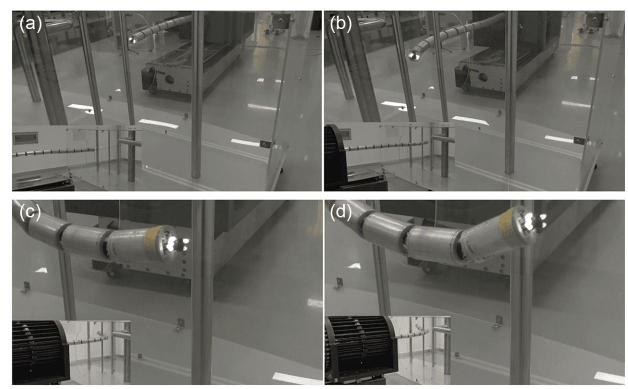


**Figure 13** (Color online) Experimental results of the inspection of the pipe tank system. (a)–(d) are the images of at $t$=0, 20, 32, and 45 s, respectively.

in Figure 13. The static motion images at $t$=0, 20, 32, and 45 s are shown in (a)–(d), respectively. The lower left and main graphs of Figure 13(a)–(d) are the side and front views of the motion, respectively.

The experimental results verify that the cable-driven snake robot can slither into and inspect the confined spaces by the presented path following method effectively and efficiently. A video of these experiments is available in Movie S2 in Supporting Information.

## 6 Conclusion and discussion

A path following method with prediction lookup and interpolation algorithms is presented in this paper for the cable-driven snake robot to explore confined spaces. The prediction lookup algorithm is developed to efficiently search the suitable point for each joint with joint information of previous moments. Meanwhile, an interpolation algorithm is presented to get accurate joint positions. This algorithm uses geometric relationships to compensate the errors brought by the discretization of continuous path. The advantage of this method is that it can achieve relative high computation efficiency and accuracy.

Simulations and experiments are performed with a 25 DOFs cable-driven snake robot to verify the presented method. Simulation results show that the presented method achieves average computation time of 0.32 ms of one step. The RMSE and VVE with interpolation algorithm are reduced by about 48%, and 72%, respectively, compared with the method without interpolation. Finally, inspection experimental results of the obstacle array and pipe tank system validate the effectiveness of the presented method on exploring confined spaces by cable-driven snake robot.

In the experiment, we can see that our robot can move smoothly through the confined spaces. This is because: (i) before the robot is powered on, all the antagonistically controlled cables are pre-tensioned and calibrated; (ii) all the motors are controlled simultaneously by a buffer structure in each control period, and their control inputs are updated at the same time; (iii) the interpolation trajectory algorithm is implemented for smooth movements.

*This work was supported by the National Natural Science Foundation of China (Grant Nos. 51435010, and 91848204).*

**Supporting Information**

The supporting information is available online at tech.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

1 Buckingham R O, Graham A C. Dexterous manipulators for nuclear inspection and maintenance case study. In: Proceedings of the International Conference on Applied Robotics for the Power Industry. Montreal: IEEE, 2010
2 Axinte D, Dong X, Palmer D, et al. MiRoR—Miniaturised robotic systems for holistic *in-situ* repair and maintenance works in restrained and hazardous environments. IEEE/ASME Trans Mechatron, 2018, 23: 978–981
3 Li T, Ma S G, Li B, et al. Axiomatic design method to design a screw drive in-pipe robot passing through varied curved pipes. Sci China Tech Sci, 2016, 59: 191–202
4 Palermo E. Tesla unveils snakelike robot charger for electric cars. New York: Live Science, 2015. http://www.livescience.com/51791-tesla-electric-car-robot-charger.html
5 Wu J, Gao Y, Zhang B, et al. Workspace and dynamic performance evaluation of the parallel manipulators in a spray-painting equipment. Robotics Comput-Integrated Manufacturing, 2017, 44: 199–207
6 Cianchetti M, Ranzani T, Gerboni G, et al. Soft robotics technologies to address shortcomings in today's minimally invasive surgery: The STIFF-FLOP approach. Soft Robotics, 2014, 1: 122–131
7 Li Z, Wu L, Ren H, et al. Kinematic comparison of surgical tendon-driven manipulators and concentric tube manipulators. Mechanism Machine Theor, 2017, 107: 148–165
8 Tang L, Wang J, Zheng Y, et al. Design of a cable-driven hyper-redundant robot with experimental validation. Int J Adv Robot Syst, 2017, 14: 1729881417734458
9 Wu J, Wang J, Wang L, et al. Dynamics and control of a planar 3-DOF parallel manipulator with actuation redundancy. Mechanism Machine Theor, 2009, 44: 835–849
10 Lau D, Oetomo D, Halgamuge S K. Generalized modeling of multi-link cable-driven manipulators with arbitrary routing using the cable-routing matrix. IEEE Trans Robot, 2013, 29: 1102–1113
11 Lau D, Oetomo D, Halgamuge S K. Inverse dynamics of multilink cable-driven manipulators with the consideration of joint interaction forces and moments. IEEE Trans Robot, 2015, 31: 479–488
12 Mustafa S K, Lim W B, Yang G, et al. Cable-driven robots. In: Handbook of Manufacturing Engineering and Technology. Nee A Y C, ed. London: Springer, 2015. 2169–2228
13 Xu J H, Xiao M B, Ding Y. Modeling and compensation of hysteresis for pneumatic artificial muscles based on Gaussian mixture models. Sci China Tech Sci, 2019, 62: 1094–1102
14 Wu J, Yu G, Gao Y, et al. Mechatronics modeling and vibration analysis of a 2-DOF parallel manipulator in a 5-DOF hybrid machine tool. Mechanism Machine Theor, 2018, 121: 430–445
15 Tang L, Huang J, Zhu L M, et al. Path tracking of a cable-driven snake robot with a two-level motion planning method. IEEE/ASME Trans Mechatron, 2019, 24: 935–946
16 Zhang Z, Zheng L, Yu J, et al. Three recurrent neural networks and three numerical methods for solving a repetitive motion planning scheme of redundant robot manipulators. IEEE/ASME Trans Mechatron, 2017, 22: 1423–1434
17 Li M, Kang R, Branson D T, et al. Model-free control for continuum robots based on an adaptive kalman filter. IEEE/ASME Trans Mechatron, 2018, 23: 286–297
18 Liu T, Jackson R, Franson D, et al. Iterative Jacobian-based inverse kinematics and open-loop control of an MRI-guided magnetically actuated steerable catheter system. IEEE/ASME Trans Mechatron, 2017, 22: 1765–1776
19 Martín A, Barrientos A, del Cerro J. The natural-CCD algorithm, a novel method to solve the inverse kinematics of hyper-redundant and soft robots. Soft Robotics, 2018, 5: 242–257
20 Aristidou A, Lasenby J. FABRIK: A fast, iterative solver for the Inverse Kinematics problem. Graphical Model, 2011, 73: 243–260
21 Ananthanarayanan H, Ordóñez R. Real-time Inverse Kinematics of ($2n$+1) DOF hyper-redundant manipulator arm via a combined numerical and analytical approach. Mechanism Machine Theor, 2015, 91: 209–226
22 Chen Y, Cai Y, Zheng J, et al. Accurate and efficient approximation of clothoids using bézier curves for path planning. IEEE Trans Robot, 2017, 33: 1242–1247
23 Saha M, Isto P. Manipulation planning for deformable linear objects. IEEE Trans Robot, 2007, 23: 1141–1150
24 Miao Y J, Gao F, Zhang Y. Gait fitting for snake robots with binary actuators. Sci China Tech Sci, 2014, 57: 181–191
25 Tang L, Zhu L, Zhu X, et al. A serpentine curve based motion planning method for cable-driven snake robots. In: Proceedings of the 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP). Stuttgart: IEEE, 2018
26 Conkur E S. Path following algorithm for highly redundant manipulators. Robotics Autonomous Syst, 2003, 45: 1–22
27 Gill R J, Kulic D, Nielsen C. Spline path following for redundant mechanical systems. IEEE Trans Robot, 2015, 31: 1378–1392
28 Garriga-Casanovas A, Rodriguez y Baena F. Complete follow-the-leader kinematics using concentric tube robots. Int J Robotics Res, 2018, 37: 197–222
29 Palmer D, Cobos-Guzman S, Axinte D. Real-time method for tip following navigation of continuum snake arm robots. Robotics Autonomous Syst, 2014, 62: 1478–1485
30 Tanaka M, Tanaka K, Matsuno F. Approximate path-tracking control of snake robot joints with switching constraints. IEEE/ASME Trans Mechatron, 2015, 20: 1633–1641
31 Andersson S B. Discretization of a continuous curve. IEEE Trans Robot, 2008, 24: 456–461
32 Wang J, Tang L, Gu G, et al. Tip-following path planning and its performance analysis for hyper-redundant manipulators. J Mech Eng, 2018, 54: 18–25