2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Effective Trajectory Tracking with Convex-Optimization Based Obstacle-Avoidance Method for Continuum Robot

Ping Deng†, Rui Peng†, Duo Tang, Xiao Cao and Peng Lu

***Abstract*—A cable-driven continuum robot with high redundancy is capable of performing the tip trajectory tracking task while simultaneously satisfying additional safety constraints, such as joint limits or external obstacles in the environment. To address these challenges, efficient motion planning methods are required. This paper proposes a quadratic programming based method in conjunction with convex polytopes based distance computation. Our methodology integrates safety constraints based on the robots' posture states, thus enabling barriers evasion in dynamic situations. Simulation outcomes demonstrate effective trajectory tracking in the presence of various objects and provide a comprehensive performance evaluation based on the generated robot state. Finally, real-world experiment was conducted on a prototype of a three-segment cable-driven continuum manipulator, which confirmed the efficacy of the proposed obstacle avoidance approach. The approach is versatile and can be adapted to similar multiple segments cable-driven continuum robotic systems by designing the robot parameters, enabling the success of tip trajectory tracking tasks under complex obstacle conditions.**

***Index Terms*—Continuum robots, Obstacle-avoidance and control, Trajectory tracking.**

I. INTRODUCTION

In recent years, continuum robots (CRs) have received increasing attention as effective solutions to travel intricate and curving pathways and perform tasks in constrained or hard-to-reach situations, particularly in the field of medical surgery [1], [2]. Compliant continuum robots differ from traditional robots designed with stiff links and joints in that they have thin, flexible structures made of elastic materials. This unique framework enables sophisticated deformations [3]. Recent research has shown a growing focus on the development and application of continuum arms [4]. [5], Object avoidance is a critical area of focus in continuum robotics research. It refers to the difficulty of computing a route for the robot to traverse from one location to another while avoiding obstacles and simultaneously respecting the physical constraints of the system [6].

Industrial automation relies on rigid robotic manipulators for precise positioning [7], but their inflexibility limits adaptability in dynamic, obstacle-filled environments. This rigidity can cause joints to reach mechanical limits, hindering obstacle avoidance and raising safety and durability concerns. Despite advances in neural network and deep learning methods for trajectory tracking [8], [9], the physical constraints of rigid manipulators remain a major challenge for reliable operation in complex settings.

Unlike rigid manipulators, continuum robots exhibit inherent compliance enabling their widespread use in surgical applications and conferring enhanced obstacle avoidance potential [10]. However, operation in complex environments remains challenging. Some trajectory tracking methods, such as data-driven approaches [11] and model-based strategies [12], are confined to 2D scenarios and lack scalability to complex settings. Model Predictive Control (MPC) for dynamic obstacle avoidance [13] suffers from high computational demands and requires extensive tuning of control gains and Prescribed Performance Function (PPF) parameters via simulation and experimentation. Additionally, the border detection technique in [14] is restricted to static obstacles, limiting efficacy in dynamic environments; the neural network-based method for redundant manipulators [15] fails to accommodate moving obstacles or irregular geometries, hampering practicality. While a number of studies demonstrate good computational efficiency for trajectory tracking, they cannot handle obstacles [16]–[18]. These limitations highlight the need for more adaptable, efficient control solutions for continuum robots in complex dynamic settings.

In this paper, we extend quadratic programming and graph theory techniques originally developed for rigid manipulators [19], [20] to the context of continuum manipulators for obstacle avoidance. By modeling the continuum robot's workspace as convex polytopes, we propose a novel framework that enables fast and effective trajectory tracking for high-speed end-effector tasks while ensuring collision avoidance. We provide a tracking method with obstacle avoidance. Compared to existing methods, our approach achieves improved computational efficiency and real-time performance in complex three-dimensional dynamic environments with multiple obstacles. This work advances geometric motion planning by combining accuracy and speed in a framework tailored to continuum robots.

II. DESIGN AND KINEMATICS OF CONTINUUM ROBOT MODELING

A. Continuum Robot System Description

This subsection provides a detailed description of the kinematics of piecewise constant curvature (PCC) - based continuum manipulators, which builds upon the mechanisms of continuum robots as presented in previous studies [21].

This work is supported by General Research Fund under Grant 17204222.*(Corresponding author: Peng Lu)*

†Ping Deng and Rui Peng equally contribute to this work.

The authors are with the Department of Mechanical Engineering, The University of Hong Kong. (Email: lupeng@hku.hk).

This work is supported in part by the Research Grants Council of HK (grant 14203917) and in part by the Chinese National Engineering Research Centre for Steel Construction Hong Kong Branch (grant BBV8).

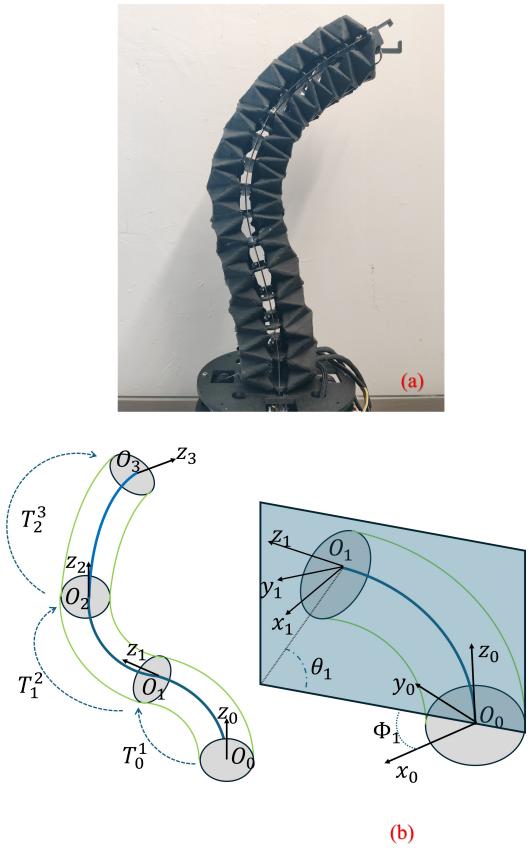

Fig. 1. Illustration of a continuum robot. (a). Physical prototype of a three-section continuum robot. (b). Continuum robot model diagram with a constant curvature segment.

The torso of the continuum robot measures 0.45 meters in length and is composed of a 0.6mm NiTi alloy flexible backbone aligned along the neutral axis, eighteen disks each 64mm in diameter uniformly distributed along the backbone, and twelve actuation tendons threaded through regularly spaced apertures in the disks. The bending motion of the robot in multiple directions is precisely controlled by modulating the extension and contraction of the actuation tendons. The k^{th} segment (where $k \leq n$, n denotes the number of robot segments) is defined by the parameters $(\theta_k, \phi_k, \kappa_k)$ of the PCC model [22], illustrated in Fig. 1, where $\theta_k \in [0, \pi]$, $\phi_k \in [-\pi, \pi]$, and $\kappa_k = \theta_k / L_k$. θ_k symbolizes the bending angle, ϕ_k indicates the bending direction angle, and κ_k signifies the curvature of the k^{th} segment. The location of the k^{th} segment can be expressed by the parameters as:

$$\boldsymbol{p}_{k-1}^{k} = \begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = \frac{1}{\kappa_k} \begin{bmatrix} \sin\phi_k \left(1 - \cos\theta_k\right) \\ \cos\phi_k \left(1 - \cos\theta_k\right) \\ \sin\theta_k \end{bmatrix} \tag{1}$$

where $\boldsymbol{p}_{k-1}^{k}$ indicates the position vector of the end of the k^{th} segment in the $k-1$ frame, x_k, y_k, and z_k are the end position coordinates, κ_k is the curvature of the k^{th} segment, ϕ_k is the bending direction angle, and θ_k is the bending angle. The rotation matrix $\boldsymbol{R}_{k-1}^{k}$ can be expressed as:

$$\boldsymbol{R}_{k-1}^{k} = \mathrm{Rot}\left(\hat{\boldsymbol{z}}_{k-1}, \phi_k\right) \cdot \mathrm{Rot}\left(\hat{\boldsymbol{y}}_{k-1}, \theta_k\right) \cdot \mathrm{Rot}\left(\hat{\boldsymbol{z}}_{k}, -\phi_k\right) \tag{2}$$

where $\boldsymbol{R}_{k-1}^{k}$ denotes the rotation matrix from the $k-1$ frame to the k frame, $\mathrm{Rot}\left(\hat{\boldsymbol{z}}_{k-1}, \phi_k\right)$ represents the rotation matrix around the z-axis of the $k-1$ frame by angle ϕ_k, $\mathrm{Rot}\left(\hat{\boldsymbol{y}}_{k-1}, \theta_k\right)$ represents the rotation matrix around the y-axis of the $k-1$ frame by angle θ_k, and $\mathrm{Rot}\left(\hat{\boldsymbol{z}}_{k}, -\phi_k\right)$ represents the rotation matrix around the z-axis of the k frame by angle $-\phi_k$.

The rotation matrix $\boldsymbol{R}_{i-1}^{i}$ can be expanded as:

$$\boldsymbol{R}_{k-1}^{k} = \begin{bmatrix} c_{\phi_k}^2 \left(1 - c_{\theta_k}\right) + c_{\theta_k} & -c_{\phi_k} s_{\phi_k} \left(1 - c_{\theta_k}\right) & s_{\phi_k} s_{\theta_k} \\ -c_{\phi_k} s_{\phi_k} \left(1 - c_{\theta_k}\right) & s_{\phi_k}^2 \left(1 - c_{\theta_k}\right) + c_{\theta_k} & -c_{\phi_k} s_{\theta_k} \\ -s_{\phi_k} s_{\theta_k} & -c_{\phi_k} s_{\theta_k} & c_{\theta_k} \end{bmatrix} \tag{3}$$

where c_{ϕ_k} and s_{ϕ_k} denote the cosine and sine of ϕ_k, respectively, c_{θ_k} and s_{θ_k} denote the cosine and sine of θ_k, respectively.

The transformation matrix $\boldsymbol{T}_{k-1}^{k}$ can be represented as:

$$\boldsymbol{T}_{k-1}^{k} = \begin{bmatrix} \boldsymbol{R}_{k-1}^{k} & \boldsymbol{p}_{k-1}^{k} \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{4}$$

where the position vector is represented by $\boldsymbol{p}_{k-1}^{k}$, the rotation matrix is represented by $\boldsymbol{R}_{k-1}^{k}$, and the transformation matrix from the $k-1$ frame to the k frame is indicated by $\boldsymbol{T}_{k-1}^{k}$. Thus, given the base position $\boldsymbol{O}_B$, the position of the endpoint of the k^{th} segment $\boldsymbol{p}_k$ can be computed in Eq.(5) as k denotes the number of segments of the continuum arm and when $k = 1$ and $k = 2$ denote the first and second segments respectively. $k = n$ means the last segment of the manipulator and the end-effector position is also denoted by $\boldsymbol{p}_{tip}$.

$$\begin{bmatrix} \boldsymbol{p}_k \\ 1 \end{bmatrix} = \prod_{j=1}^{k} \begin{bmatrix} \boldsymbol{R}_{j-1}^{j} & \boldsymbol{p}_{j-1}^{j} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{O}_B \\ 1 \end{bmatrix} \tag{5}$$

The Jacobian matrix $\boldsymbol{J}_0^k$ is defined as:

$$\begin{aligned} \boldsymbol{J}_0^k &= \frac{\partial \boldsymbol{p}_k}{\partial \boldsymbol{q}_k} \\ &= \begin{bmatrix} \frac{\partial \boldsymbol{p}_k(x)}{\partial \boldsymbol{q}_1} & \frac{\partial \boldsymbol{p}_k(x)}{\partial \boldsymbol{q}_2} & \cdots & \frac{\partial \boldsymbol{p}_k(x)}{\partial \boldsymbol{q}_{2k}} \\ \frac{\partial \boldsymbol{p}_k(y)}{\partial \boldsymbol{q}_1} & \frac{\partial \boldsymbol{p}_k(y)}{\partial \boldsymbol{q}_2} & \cdots & \frac{\partial \boldsymbol{p}_k(y)}{\partial \boldsymbol{q}_{2k}} \\ \frac{\partial \boldsymbol{p}_k(z)}{\partial \boldsymbol{q}_1} & \frac{\partial \boldsymbol{p}_k(z)}{\partial \boldsymbol{q}_2} & \cdots & \frac{\partial \boldsymbol{p}_k(z)}{\partial \boldsymbol{q}_{2k}} \end{bmatrix} \end{aligned} \tag{6}$$

where $\boldsymbol{J}_0^k$ denotes the Jacobian matrix of the k^{th} segment, $\boldsymbol{p}_k$ is the position vector of the end of the k^{th} segment, and $\boldsymbol{q}_k = [\theta_1, \phi_1, \ldots, \theta_k, \phi_k]$ is the joint vector.

The velocity of k^{th} segment $\dot{\boldsymbol{p}}_k$ can be calculated as:

$$\dot{\boldsymbol{p}}_k = \boldsymbol{J}_0^k \cdot \dot{\boldsymbol{q}}_k \tag{7}$$

where $\dot{\boldsymbol{p}}_k$ represents endpoint's velocity in the k^{th} segment, $\boldsymbol{J}_0^k$ represents the k^{th} segment's Jacobian matrix, and $\boldsymbol{q}_k$ represents the joint velocity vector.

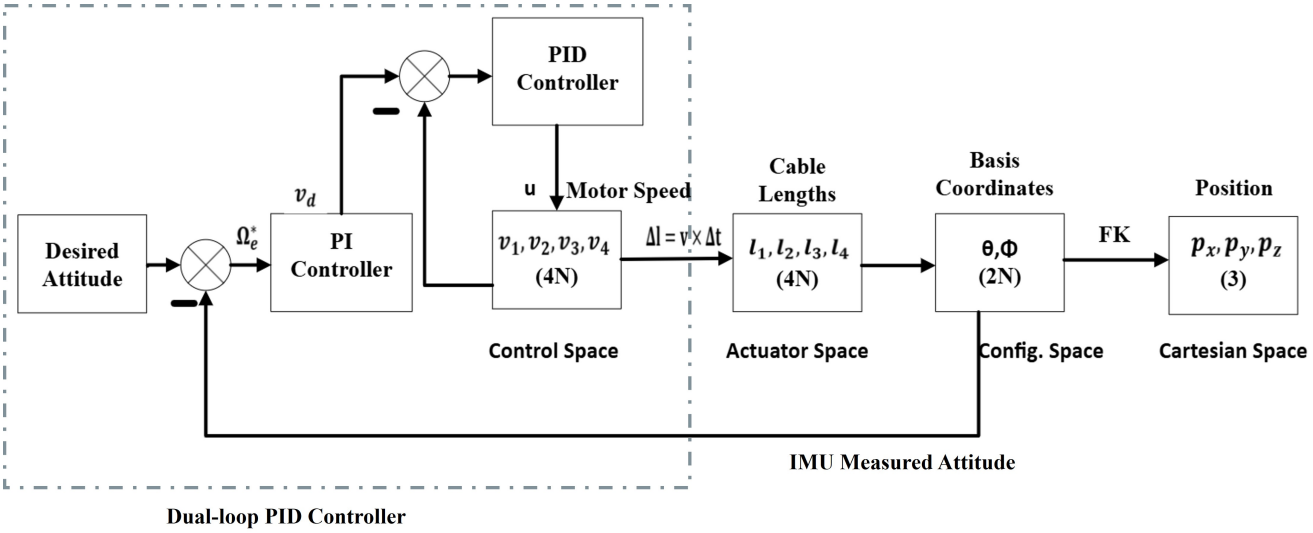

Fig. 2. The diagram block illustrates the mechanics of the N segment continuum manipulator system with a kinematic dual-loop PID controller based on IMU attitude data.

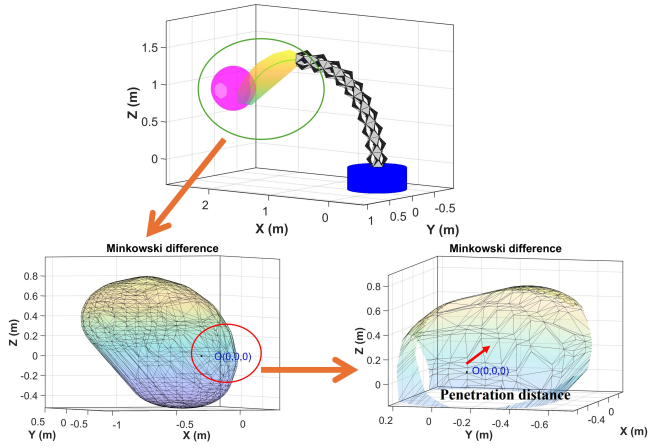

Fig. 3. Illustration of the GJK distance algorithm. The third segment of the continuum arm contacts the sphere object, the GJK distance show the penetration depth.

The definition of the error $\boldsymbol{e}$ is:

$$\boldsymbol{e} = \|p_{tip} - p_d\|_2 \tag{8}$$

where the Euclidean norm is represented by $\|\cdot\|_2$ and the error vector between the desired position p_d and the actual location p_{tip} is shown by $\boldsymbol{e}$.

B. IMU based controller

This section presents a dual-loop PID controller meant to reduce the attitude errors of three end disks in a mechanical system. The illustration is shown in Fig. 2. The internal angular loop in the controller and the outside attitude loop taken together give exact control over the orientation and angular speed of the system. The difference between the IMU-measured attitude $\boldsymbol{\epsilon}^*_{IMU}$ and the desired quaternion attitude $\boldsymbol{\epsilon}^*_d$ is the attitude error $\boldsymbol{\epsilon}^*_e$:

$$\begin{aligned} \boldsymbol{\epsilon}^*_e &= \boldsymbol{\epsilon}^*_d - \boldsymbol{\epsilon}^*_{IMU} \\ \boldsymbol{\epsilon}^*_d &= \boldsymbol{\epsilon}_\phi \otimes \boldsymbol{\epsilon}_\theta \end{aligned} \tag{9}$$

where $\boldsymbol{\epsilon}_\phi = [\cos(\phi/2), 0, \sin(\phi/2), 0]$ denotes rotation around the z axis by angle ϕ and $\boldsymbol{\epsilon}_\theta = [\cos(\theta/2), 0, \sin(\theta/2), 0]$ denotes bending about the y axis by angle θ. The angular controller is intended to reduce the angular errors by utilizing the attitude error. $\boldsymbol{v}_d$ is the desired motor speed. :

$$v_d = K_\Omega \boldsymbol{\Omega}^*_e + I_\Omega \int \boldsymbol{\Omega}^*_e \tag{10}$$

where K_Ω and I_Ω are the proportional and integral gains of the angular controller, respectively. The angular error matrix $\boldsymbol{\Omega}^*_e$ is defined as:

$$\boldsymbol{\Omega}^*_e = K_{\boldsymbol{\epsilon}} \boldsymbol{\epsilon}^*_e - \boldsymbol{\Omega}^* \tag{11}$$

Here, $K_{\boldsymbol{\epsilon}}$ is the proportional gain of the attitude controller, and $\boldsymbol{\Omega}^*$ is the reference angular velocity matrix. The control input matrix u is calculated based on the motor speed error $\boldsymbol{v}_e$, which is the difference between the desired motor speed $\boldsymbol{v}_d$ and the actual motor speed $\boldsymbol{v}$:

$$\boldsymbol{v}_e = \boldsymbol{v}_d - \boldsymbol{v} \tag{12}$$

The control input matrix u is then given by:

$$u = K_v \boldsymbol{v}_e + I_v \int \boldsymbol{v}_e + D_v \dot{\boldsymbol{v}}_e \tag{13}$$

where K_v, I_v, and D_v are the proportional, integral, and derivative gains of the velocity controller, respectively.

III. End Effector Task with Obstacle Avoidance

A. GJK distance

The Gilbert-Johnson-Keerthi (GJK) distance algorithm is an exact iterative method based on convex geometry, used to determine the shortest distance and the coordinates of the closest points between two convex polytopes [20]. Its main advantage lies in the fact that it only requires decomposing the manipulator and obstacles into several convex polytopes, rather than processing their entire shapes, to accurately compute the minimal distance. The GJK algorithm performs collision detection with a time complexity of O(M + N), where M and N represent the numbers of vertices of the two convex polytopes involved. During each iteration, the algorithm focuses on selecting search directions close to the origin, which contributes to rapid convergence. Fig. 3 demonstrates how the GJK distance algorithm is applied in the context of a continuum robotic arm, highlighting its effectiveness in handling complex geometries for collision avoidance.

Let A and B be two convex polytopes, where A is derived from a bending cylinder section and B consists of external objects. Their respective sets of surface points are represented as $S_A \in \mathcal{R}^3$ and $S_B \in \mathcal{R}^3$. The shortest distance between the two convex polytopes is defined by

$$d(S_A, S_B) = min\{\|p_A - p_B\| : p_A \in S_A, p_B \in S_B\} \quad (14)$$

where $\|\cdot\|$ denotes the Euclidean norm. The GJK proximity algorithm operates by progressively determining the Minkowski difference set $C = A \ominus B$ through successive applications of the support mapping. This process iteratively constructs a simplex approximation until convergence. The final distance measurement corresponds to the minimum norm from the terminal simplex $D \subset C$ to the coordinate origin. Let the vertex set of simplex D be denoted as $S_D \subset \mathcal{R}^3$, with $GJK(D) \in S_D$ representing the minimal-norm vertex in S_D. As a result, we possess

$$d(S_A, S_B) = \|GJK(D)\| = min\{\|h\| : h \in S_D\} \quad (15)$$

The geometric penetration between convex polyhedra A and B is confirmed when the origin resides within the convex hull of simplex D, where the penetration depth is quantified by $\|h\|$.

B. Optimization-based Trajectory Tracking method

SLSQP (Sequential Least Squares Quadratic Programming) algorithm is a gradient-based optimization method for solving nonlinearly constrained optimization problems

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) = \frac{1}{2}e^2, \\ \text{s.t.} \quad & h_i(\boldsymbol{x}) \geq 0, \quad i = 1, \ldots, n_h; \end{aligned} \quad (16)$$

where $\boldsymbol{x} = [q_1, q_2, \ldots, q_{2n}]^T \in \mathcal{R}^{2n}$, $f : \mathcal{R}^{2n} \to \mathcal{R}$, $h_i : \mathcal{R}^{2n} \to \mathcal{R}$, n_h depends on the inequality constraints and in this paper, the system doesn't have equality constraint. The SLSQP algorithm shown in Algorithm 1 functions by solving a sequence of quadratic programming (QP) subproblems. Each subproblem is a quadratic approximation of the original problem, and is solved to determine the search direction for the next iteration.The QP subproblem at iteration k is:

Algorithm 1: SLSQP Algorithm

Input: Initial guess $\boldsymbol{x}_0 \in \mathcal{R}^6$; initial Lagrange multiplier $\lambda_0 \in \boldsymbol{C}$; initial Hessian approximation $H_0 \in \mathcal{R}^{6\times6}$; convergence threshold tolerance; Maximum iteration number max_iter
Result: Optimal solution $\boldsymbol{x}^*$ and its corresponding objective function value $f(\boldsymbol{x}^*)$
Initialization: Compute the initial gradient $\nabla f(\boldsymbol{x}_0)$ and the constraint functions at $\boldsymbol{x}_0$.
while *not converged and current iteration* $k <$ *max_iter* **do**
  Minimize over $\boldsymbol{s}$: $0.5\boldsymbol{s}^T H_k \boldsymbol{s} + \nabla f(\boldsymbol{x}_k)^T \boldsymbol{s}$
  Subject to:
   $\mathrm{h}_i(\boldsymbol{x}_k) + \nabla h_i(\boldsymbol{x}_k)^T \boldsymbol{s} \geq 0, \quad i = 1, \ldots, n$
  Solve the problem using Eq.(19) and satisfy Eq.(20)
  $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}$
  Calculate $\nabla f(\boldsymbol{x}_{k+1})$ and $h(\boldsymbol{x}_{k+1})$
  If $\|\nabla f(\boldsymbol{x}_{k+1})\| <$ tolerance, set converged = true
end
Return Optimal solution $\boldsymbol{x}^*$ and its corresponding objective function value $f(\boldsymbol{x}^*)$

$$\begin{aligned} \min \quad & g(\boldsymbol{s}) = \nabla f(\boldsymbol{x}_k)^T \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T H_k \boldsymbol{s} \\ \text{s.t.} \quad & z(\boldsymbol{s}) = h_i(\boldsymbol{x}_k) + \nabla h_i(\boldsymbol{x}_k)^T \boldsymbol{s} \geq 0, \quad i = 1, \ldots, n_h; \end{aligned} \quad (17)$$

$$\begin{aligned} \nabla f(\boldsymbol{x}_k)^T &= (\boldsymbol{p}_{tip} - \boldsymbol{p}_d)^\top J_0^k \\ H_k &= H_{k-1} - \frac{\boldsymbol{s}_{k-1}\boldsymbol{s}_{k-1}^T}{\boldsymbol{s}_{k-1}^T y_{k-1} \boldsymbol{s}_{k-1}} + \frac{H_{k-1} y_{k-1} y_{k-1}^T H_{k-1}}{y_{k-1}^T H_{k-1} y_{k-1}} \\ y_{k-1} &= \nabla f(\boldsymbol{x}_k)^T - \nabla f(\boldsymbol{x}_{k-1})^T \end{aligned} \quad (18)$$

where $\nabla f(x_k)$ is the gradient of the objective function x_k, y_{k-1} is the difference in gradient, H_k using a quasi-Newton method to approximate and s is the search direction.The Lagrangian function is constructed as follows to solve the subproblem:

$$L(\boldsymbol{s}, \lambda) = g(\boldsymbol{s}) + \sum_{i=1}^{m} \lambda_i z_i(\boldsymbol{s}) \quad (19)$$

To solve the QP subproblem, it is essential to determine the parameters s and λ that satisfy the Karush-Kuhn-Tucker

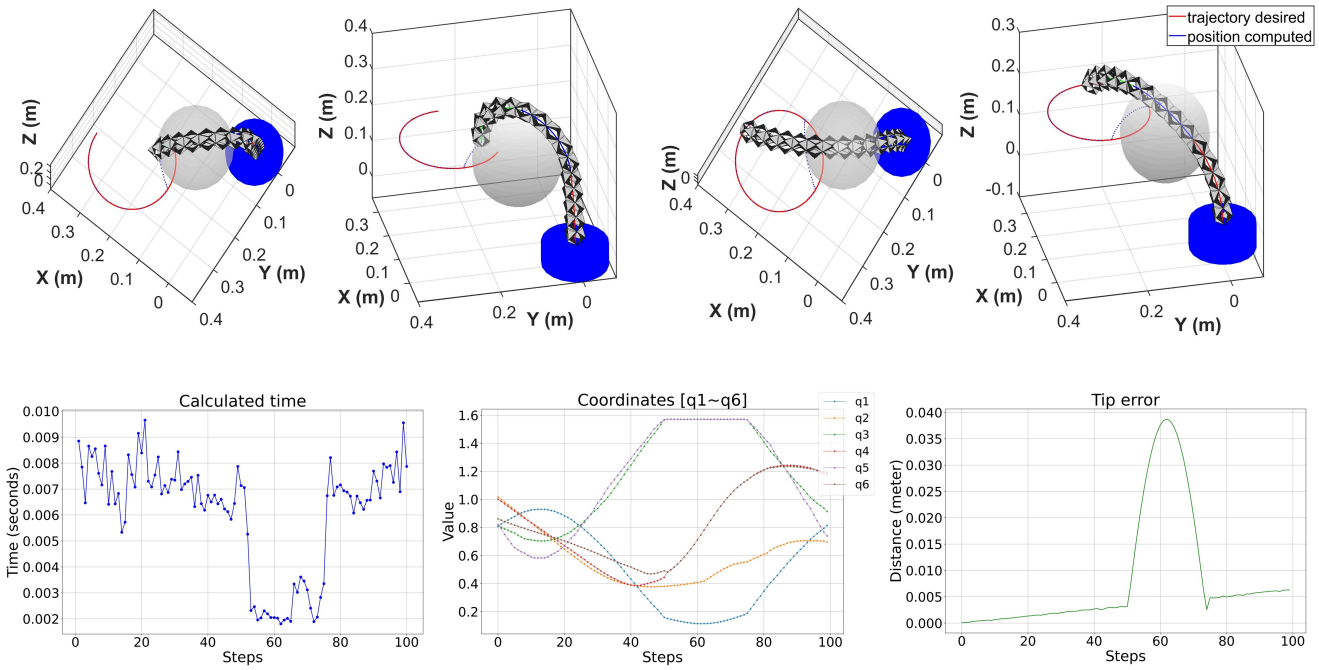

Fig. 4. Simulation 1: Static environment trajectory tracking with a sphere obstacle. The red horizontal circle represents the desired path, the desired circle origin is placed at $[0.24, 0.24, 0.16]^\top$ with radii set as 0.12 m, while the blue points represent the computed position. The corresponding calculated time, manipulator coordinates and tip position error are given.

(KKT) conditions

$$\begin{aligned} \nabla_x L(\boldsymbol{s}, \lambda^*) &= 0, \\ z_i(\boldsymbol{s}) &\geq 0, \quad i = 1, \ldots, n_h; \\ \lambda_i^* &\geq 0, \quad i = 1, \ldots, n_h; \\ \lambda_i z_i(\boldsymbol{s}) &= 0. \end{aligned} \tag{20}$$

Once the QP subproblem is solved, the solution s is used to update the current iterate:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s} \tag{21}$$

Define constraints $h_i(x)$ in Eq.(16) considering the safe distance between manipulator and objects, basis limits and manipulator joints constraints

$$h_i(x) = \begin{cases} d(S_A, S_B) > 0, \\ \quad q < q_{max} \\ \quad \Delta q < b \\ \quad \dot{p}_k < \dot{p}_{k,max} \end{cases} \tag{22}$$

where S_A denotes the three segments of the continuum manipulator, S_B denotes the several objects, q denotes the manipulator coordinates

According to the algorithm, a set of solution $\boldsymbol{x}^*$ can be put into the IMU-based controller to propose a set of motions which can be used in complex scenarios.

IV. Simulation

This section evaluates our numerical method, incorporating safety constraints, to demonstrate its effectiveness in trajectory tracking and obstacle avoidance within both static and dynamic environments. The proposed approach utilizes a three-segment Cable-Driven Continuum Robot (CDCR) model explicitly designed to account for interactions with

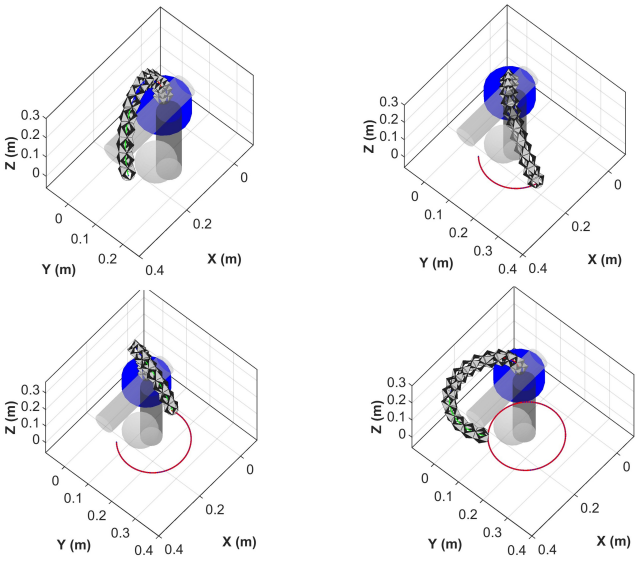

Fig. 5. Simulation 2A: Static environment trajectory tracking with multiple obstacles. Left: The desired circle origin is placed at $[0.24, 0.24, 0.16]^\top$ and orientation is defined by normal vector $[1, 0, 1]^\top$.

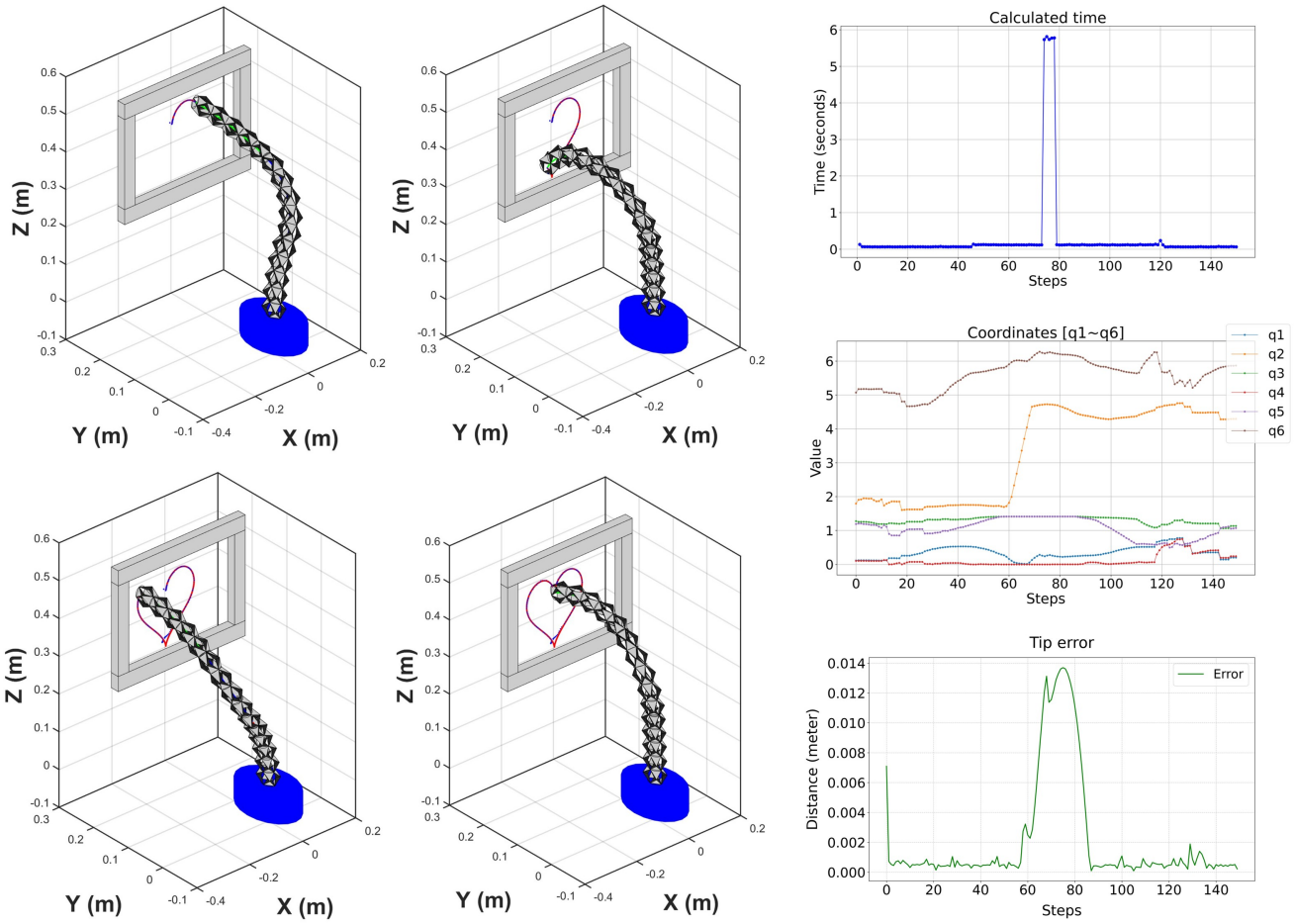

Fig. 6. Simulation 2B: A heart-shaped trajectory tracking for the continuum arm subject to the constraint of a window-shaped obstacle.

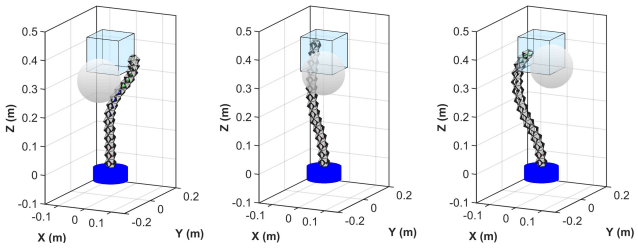

Fig. 7. The object avoidance of a moving spherical obstacle with the CDCR tip confined to a cubic region

external objects. Simulations were conducted in C++ within the ROS-rviz platform, while data rendering visualization was performed using MATLAB. The process involved calculating multiple three-dimensional locations along the robot's structure via the inverse solution of a refined, computationally efficient compressible curvature forward kinematics. All experiments were executed on a laptop equipped with an Intel Core i7-13700H CPU@2.40GHz 16GB RAM.

A. *Static single object avoidance*

The first simulation results are demonstrated in Fig. 4. In this simulation, three cameras are employed to capture and display the posture of the robot from different angles. A gray sphere, positioned at $[0.12, 0.11, 0.16]^\top$ is used as a reference object with a radius of $0.13\,\mathrm{m}$. The desired trajectory for the end-effector is designed as a circular trajectory with a radius of $0.13\,\mathrm{m}$ and 20 points are created inside the sphere object deliberately. The method corresponds with CDCR's kinematics by computing optimal inverse kinematics (IK) solutions at each sampled point along the trajectory. The computation of the maximal single IK takes approximately $0.00965\,\mathrm{s}$. The objective of the CDCR is to follow the path that has been generated while avoiding collisions with obstacles. When it comes to finding a solution that is feasible, the average amount of time is around $0.00611\,\mathrm{s}$.

B. *Static multiple objects avoidance*

The second simulation provides two different scenarios. Fig. 5 depicts the trajectory tracking task constrained by three obstacles. The CDCR is preceded by a gray sphere with a radius of $0.07\,\mathrm{m}$ is located at $[0.26,\ 0.20,\ 0.20]^\top$ and

TABLE I
CONTROL PERFORMANCE AND ACCURACY COMPARISON BETWEEN OUR STUDY AND THE EXISTING WORKS

Reference	Robot type (initial length L)	Inverse kinematics control method	Position error	Computation Time (language/software)	Open or closed loop
[5]	Multi-segment pneumatic-driven (L=110 mm)	Reduced FEM-based	7.9 mm ($<$ 8%)	75 ms per inverse step (SOFA)	Closed loop
[6]	Multi-segment pneumatic-driven (L=104 mm)	Analytical Jacobian-based	6.0 mm ($<$ 6%)	133 ms per inverse step (MATLAB)	Open loop
[16]	Two-segment magnetic-driven (L=70 mm)	Inverse kinematics model based	6.0 mm ($<$ 9%)	6.5 ms per inverse step (MATLAB)	Closed loop
[17]	Multi-segment pneumatic-driven (L=89 mm)	Inverse statics (Cosserat rod model) based	3.4 mm ($<$ 4%)	77 ms per inverse step (MATLAB)	Open loop
[18]	Tensegrity continuum robot (L=700 mm)	Dynamics (DAEs) model based	12.0 mm ($<$ 1.7%)	less than 10 ms per inverse step (MATLAB)	Closed loop
Our work	Multi-segment cable-driven continuum robot (L=450 mm)	Inverse kinematics model with single or multiple obstacles	5 mm ($<$ 2%)	6 ms for single and 9 ms for multiple obstacles per step (C++)	Closed loop

* The percentage indicates the relative position error in relation to the original length of the robot.

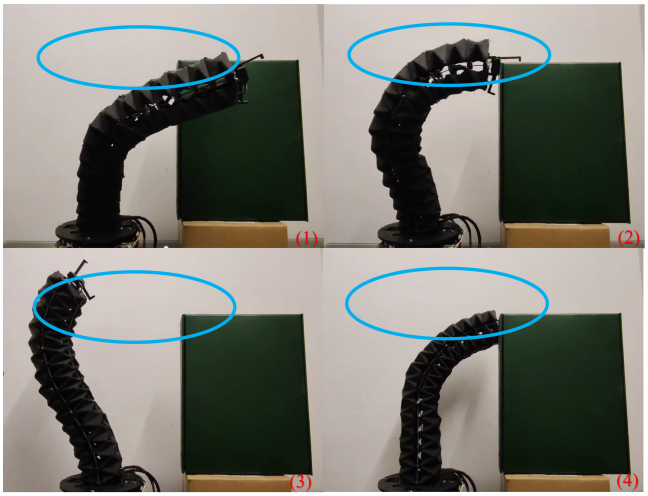

Fig. 8. Lab experiment: Fixed-base object avoidance with a static rectangular box as the obstacle, where the blue circle denotes the desired trajectory which is parallel to the ground.

two cylinders, each having a radius of 0.1 m and a height of 0.3 m, to simulate intensive obstacles. The multiple different shape objects serve to facilitate precise tracking in complex environments along a specified circular trajectory with a radius of 0.13 m. The maximal single IK calculation requires approximately 0.00419 s, while the CDCR efficiently tracks paths with obstacle avoidance, yielding an average solution time of about 0.00239 s. The simulation results demonstrate that the proposed method achieves precise path tracking while maintaining computational efficiency, as evidenced by a tip error of 0.003813 m relative to the total continuum body length ($L = 0.45$ m), representing an accuracy of approximately 0.095%. Fig. 6 depicts the trajectory tracking task constrained by a window-shaped obstacle.The window has an overall width of 0.48 meters and height of 0.32 meters, with its four edges each consisting of a rectangular prism. The CDCR will navigate through this window to perform operations, while avoiding colliding with the window's four edges during the motion planning. The maximum single IK computation step requires approximately 5.4 s when the CDCR tip is close to and positioned below the window bottom, whereas the average computation time is roughly 0.009 s in other cases. A comparison of tracking control of continuum robot in existing works with ours is shown in TABLE I.

C. Dynamic object avoidance

The third simulation results, containing three snapshots under a fixed camera, are demonstrated in Fig. 7. The workspace within which the end-effector can operate is defined as a $0.1\,\text{m} \times 0.1\,\text{m} \times 0.1\,\text{m}$ cubic region, with its center located at $[0, 0, 0.4]^\top$ and the origin tip position is set at $[0, 0, 0.4]^\top$. The dynamic ball is generated to contact the CDCR body. The algorithm works by seeking the least tip position change in the workspace with a small configuration space change. The average time required to find a feasible solution is approximately 0.09471 s.

V. VALIDATION

This segment examines the efficacy of the obstacle avoidance tracking method that employs the three-segment CDCR in a laboratory setting. A high-precision RGB-d motion capture camera is utilized to monitor the tip position of the CDCR.

The total duration of the experiment is 25 seconds. The developed method showcased exceptional performance on the

continuum robot in experimental tests. The CDCR base is located at $[0, 0, 0]^\top$, and the obstacle is located at $[0, 0.15, 0]^\top$ with a height of 0.4 m. Using our method, the robot can track the desired path while performing obstacle avoidance calculations and recalculate the path when the desired point is within the obstacle. Fig. 8 illustrates the snapshots of the experiment. The cumulative tip tracking error between the desired and actual positions is approximately 0.035m while the horizontal desired circle trajectory has a center point at $[0, 0, 0.4]^\top$ and a radius of 0.2m. Our method ensures safe and rapid trajectory tracking. We emphasize its resilience in complex situations, prioritizing safety. This combined focus considerably elevates its value for real-world applications.

VI. Conclusion

The paper presents an effective obstacle avoidance trajectory tracking method for continuum robots in complex environments. With continuum robots operating in dynamic situations, the primary goal is to achieve collision-free tracking. The strategy, which innovatively combines convex geometry with continuum robot in developing the collision detection algorithm, has proven effective in handling complex and dynamic tracking control scenarios through simulations and tests. The method can identify the nearest point at the minimum distance to satisfy the safety constraints when the desired trajectory intersects with an obstacle. In multiple obstacle circumstances, the average tip error is approximately 0.095% relative to the robot's body length. Additionally, the method performs well when facing an approaching obstacle, enabling flexible maneuvering. Future research will concentrate on enhancing the robustness, flexibility, and dynamic avoidance capabilities of the algorithm to handle unexpected obstacles that impede robot movement and manipulation. Furthermore, the current method does not consider force-related interactions. In future development, the continuum robot's body may encounter external disturbances.

References

[1] J. Lai, K. Huang, B. Lu, Q. Zhao, and H. Chu, "Verticalized-tip trajectory tracking of a 3d-printable soft continuum robot: Enabling surgical blood suction automation," *IEEE/ASME transactions on mechatronics*, vol. 27, no. 3, pp. 1–1, 2022.

[2] J. Li, D. Li, C. Wang, W. Guo, Z. Wang, Z. Zhang, and H. Liu, "Active collision avoidance for teleoperated multi-segment continuum robots toward minimally invasive surgery," *The International Journal of Robotics Research*, vol. 43, no. 7, pp. 918–941, 2024.

[3] C. Armanini, F. Boyer, A. T. Mathew, C. Duriez, and F. Renda, "Soft robots modeling: A structured overview," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 1728–1748, 2023.

[4] R. Peng, Y. Wang, M. Lu, and P. Lu, "A dexterous and compliant aerial continuum manipulator for cluttered and constrained environments," *Nature Communications*, vol. 16, no. 1, p. 889, 2025.

[5] P. Chaillou, J. Shi, A. Kruszewski, I. Fournier, H. A. Wurdemann, and C. Duriez, "Reduced finite element modelling and closed-loop control of pneumatic-driven soft continuum robots," in *2023 IEEE International Conference on Soft Robotics (RoboSoft)*, 2023, pp. 1–8.

[6] T. Greigarn, N. L. Poirot, X. Xu, and M. C. ÃavuÅoÄlu, "Jacobian-based task-space motion planning for mri-actuated continuum robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 145–152, 2019.

[7] Y. Tong, J. Liu, H. Zhou, Z. Ju, and X. Zhang, "Adaptive tracking control of robotic manipulators with unknown kinematics and uncertain dynamics," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 4, pp. 5252–5269, 2024.

[8] P. Yu, N. Tan, Z. Zhong, C. Hu, B. Qiu, and C. Li, "Unifying obstacle avoidance and tracking control of redundant manipulators subject to joint constraints: A new data-driven scheme," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 16, no. 5, pp. 1861–1871, 2024.

[9] M. Yang, Y. Zhang, N. Tan, and H. Hu, "Concise discrete znn controllers for end-effector tracking and obstacle avoidance of redundant manipulators," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3193–3202, 2022.

[10] J. Li, D. Li, C. Wang, W. Guo, Z. Wang, Z. Zhang, and H. Liu, "Active collision avoidance for teleoperated multi-segment continuum robots toward minimally invasive surgery," *The International Journal of Robotics Research*, vol. 43, no. 7, pp. 918–941, 2024.

[11] Y. Gao, Z. Chen, F. Zhong, X. Li, and Y.-H. Liu, "Data-driven modeling and high-precision tracking control of a soft continuum manipulator: Enabling robotic sorting of multiwire cables," *Advanced Intelligent Systems*, vol. 6, no. 10, p. 2300827, 2024. [Online]. Available: https://advanced.onlinelibrary.wiley.com/doi/abs/10.1002/aisy.202300827

[12] H. Zhang, H. Jin, Z. Liu, Y. Liu, Y. Zhu, and J. Zhao, "Real-time kinematic control for redundant manipulators in a time-varying environment: Multiple-dynamic obstacle avoidance and fast tracking of a moving object," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 28–41, 2020.

[13] J. Nie and Y. Wang, "Safe obstacle avoidance planning-control scheme for multiconstrained mobile manipulators," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 12, pp. 14 154–14 163, 2024.

[14] J. Lai, B. Lu, Q. Zhao, and H. K. Chu, "Constrained motion planning of a cable-driven soft robot with compressible curvature modeling," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4813–4820, 2022.

[15] Z. Zhang, S. Chen, X. Zhu, and Z. Yan, "Two hybrid end-effector posture-maintaining and obstacle-limits avoidance schemes for redundant robot manipulators," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 754–763, 2020.

[16] D. Lin, W. Chen, K. He, N. Jiao, Z. Wang, and L. Liu, "Position and orientation control of multisection magnetic soft microcatheters," *IEEE/ASME Transactions on Mechatronics*, vol. 28, no. 2, pp. 907–918, 2023.

[17] J. Shi, S.-A. Abad, J. S. Dai, and H. A. Wurdemann, "Position and orientation control for hyperelastic multisegment continuum robots," *IEEE/ASME Transactions on Mechatronics*, vol. 29, no. 2, pp. 995–1006, 2024.

[18] F. Li, H. Yang, G. Gu, Y. Wang, and H. Peng, "Position and orientation tracking control of a cable-driven tensegrity continuum robot," *IEEE Transactions on Robotics*, vol. 41, pp. 1791–1811, 2025.

[19] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake, "Motion planning around obstacles with convex optimization," *Science Robotics*, vol. 8, no. 84, p. eadf7843, 2023. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.adf7843

[20] B. Ma, Z. Xie, B. Zhan, Z. Jiang, Y. Liu, and H. Liu, "Actual shape-based obstacle avoidance synthesized by velocityâacceleration minimization for redundant manipulators: An optimization perspective," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 10, pp. 6460–6474, 2023.

[21] P. Luo, S. Yao, Y. Yue, J. Wang, H. Yan, and M. Q.-H. Meng, "Efficient rrt*-based safety-constrained motion planning for continuum robots in dynamic environments," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 9328–9334.

[22] C. Della Santina, C. Duriez, and D. Rus, "Model-based control of soft robots: A survey of the state of the art and open challenges," *IEEE Control Systems Magazine*, vol. 43, no. 3, pp. 30–65, 2023.